PROCEEDINGS

COMMON MEETING

CINCINNATI, OHIO

SEPTEMBER 6-7-8, 1967

TABLE OF CONTENTS

SESSION NUMBER 0

SPEAKERS
    JIM STANSBURY
    LAURA AUSTIN
    CHARLES MAUDLIN

DISCUSSION

WELCOME TO NEW MEMBERS. EXPLANATION OF KINDS OF SESSIONS AVAILABLE. SUGGESTIONS OF SPECIFIC SESSIONS OF INTEREST TO NEW MEMBERS. DISCUSSION OF INFORMATION TO GIVE TO SECRETARY-TREASURER TO OFFICIALLY BECOME MEMBER.

SESSION NUMBER W.1.1

GENERAL SESSION

SPEAKERS

JAMES STANSBURY, PRESIDENT

TRANSCRIPTION

GENERAL ASSEMBLY

COMMON, CINCINNATI, 6 SEPTEMBER, 1967

JAMES STANSBURY, CHAIRMAN -

At Boston, I took off on the members enthusiastically, wildly, and with not too much result, but I can't preach to the people that attend today. We've tried to set up a meeting that meets the objectives and requirements that you people gave to us at Boston, and in conversation afterwards. We're still making mistakes, but I think we've done far better than we did then. Hope you will enjoy it and benefit from it.

I'd like to say one thing about the interests of COMMON. I've had numerous inquiries from people who aren't certain whether COMMON is what they want to join, or not. They would join GUIDE possibly; if they had a large machine, they might join SHARE. They aren't qualified for these on the basis of machines, so they say, fine, we'll join COMMON. The special groups in this organization are the ones you people want. If you people have a common interest then get together with a birds of a feather session, organize a project, and, if you convince the Executive Board that you are serious, you'll get to be a project. The people here are the only ones to make those projects run - make them beneficial. That's about all I intend to say.

I'll start off here by introducing members of the Executive Board, then I intend to have each one of our Divisional Managers present a short description of what his division is doing. Jim Tunney has some corrections on the agenda and IBM has requested some time to make a short presentation, which they will think will be of interest to all members - a new hardware announcement.

On my left over here is Dick Pratt, Executive Board Member, next to him is Norman Goldman, President of the Eastern Region. Paul Bickford, who was appointed to fill the vacancy created by Don Jardine's resignation. Bill Lane, Western Region President. On my right, Frank Maskiell, Executive Board Member, and on the far right, Chuck Maudlin, Secretary-Treasurer. Since Chuck is a rather ethical character, I'm going to apologize for him. There have been a great many problems with communications with the Secretary-Treasurer - lack of installation support. He's changed installations; he assures me that he has his backlog down now to a reasonable value, and that in the future we can expect reasonably prompt responses from him. Is Laura Austin here? Jim Taylor?

We'll start out with the Applications Division since we have the Division Manager present.

FRANK MASKIELL -

I wasn't prepared to dissertate on the Applications Divisions. We have six projects in the Applications Division. First, I might state the objectives of the Applications Division. You all will have a chance to read them more accurately in the COMMON Reference Manual, which will be forthcoming shortly, I understand. In the Applications Division,, we are particularly concerned with problems of installations which are not particular machine problems. We're concerned with what you're doing on the machine,on the computer, not what computer you are using to accomplish the results of these problems. To this end we have these six projects, specifically, they are the Techniques Project, which is concerned with mathematical packages, some statistical work. We have the Electric Utilities Project, what has operated in the past, particularly in the Eastern Region, as the Electric Utilities Team. We have the Petro-Chemical Engineering Project, hopefully serving the needs of installations in these industries. The Civil-Mechanical Project, who at this meeting, I believe, will be meeting withsome of the members of Cepa. We have the Education Project, which gathers together the large number of university people who, in some sense of the word, have problems all of their own. Meeting at Cincinnati, for the first time, are a fair number of individuals interested in numerical control. This is a project which is just now getting started. We hope to see it grow as others find information available in this area, and can be served by this as a project.

CHAIR -

An now that Laura Austin is here, the lady sitting on the far right, I'm going to ask her to discuss the Administration Division next. She also has some announcements to make.

LAURA AUSTIN -

Thank you Jim.

I would like to point out some of the Sessions to be held for the Administration Division first. The Administration Division is a working division of COMMON. It is one from which you, as a individual, will not realize a great deal of benefit for your installation, in terms programs that you have obtained through participation in a project. It is a division in which we feel you can contribute a great deal to COMMON, and thereby benefit your installation through the overall operation of COMMON, and through what we hope will be an efficient operation of COMMON. The first session I would like to call to your attention is at 10:30 on Thursday, which is the session on nominations. We will be, within the next year, drawing up a slate of nominations for officers of COMMON. Since COMMON has grown to be such a large organization, it is difficult for us to get around to know each member. We would like to have people who are interested in serving on the Executive Board of COMMON, or who are interested in serving as Chairmen of Committees, or as Managers of Divisions, to make themselves known at this nominating sessions, so that we can become better acquainted with you, and so that we can have a slate of people to draw from for our nominations for the next year. We feel that you will benefit a great deal from taking part in COMMON, from getting right into the working side of COMMON, whether it is as a Committee Chairman, or as a Project Chairman, or as an Executive Board Member.

The next Session that I would like to call to your attention is at 8:30 on Friday. This is the Future Meetings Session. In this one, we will be discussing the locations of future meetings, the cities and hotels, and we also would like to solicit help from members of COMMON for Program Chairman and Local Arrangements Chairman. Again, you contribute a great deal to COMMON by offering your services in this area. If you will be at all available for any of these positions, we would like to have you come to this meeting on Friday. We can give you more detail as to what is involved in carrying out these duties at that time. Then, at 10:30 on Friday, is the Program Library Project. We are hoping this will be one of the largest sessions of the conference. In this particular session, we will be discussing the use of the PREP forms, what the PREP form is, what its purpose is, and how it can be of help to your. We will be discussing the Program Shipment Analysis form, in which we are trying to work with IBM in finding out where the problems are in the shipment of programs, and in the distribution of them. We will also be discussing the new 360, 1130, 1800 Library, and I'm sure this is going to be of interest to most members. We will be discussing the standards for submittal, the procedures for ordering, and what catalogs will be available, etc.

Another part of the Administrative Division has been the Reference Manual Committee. This has been largely made up of one person - Mr. B. R. Russell, who has contributed a great deal of time and effort in preparing for us a reference manual. I'm happy to say that today, I have a preliminary copy of the Reference Manual for COMMON.

This will be mailed to every installation in about two weeks. You should be receiving the complete binder, dividers and the material. Now, this first issue of the Reference Manual is not complete. There are some sections that have not been submitted yet. We felt that we wanted to get it out to the members as soon as possible. The things that it does contain are something about the history of COMMON, what are the advantages of membership, the obligations of membership, what is the coming calendar for COMMON meetings, the organizational structure. We hope here to have completed before too long a list of all the projects that are currently active in COMMON - their scope and objectives. The Reference Manual will not include current reports on project progress. These will be covered in CAST, but the Reference Manual will give the scope and objective of each project; it will give you a list of all of the Project Chairmen, so you will know who to contact when you want to have some correspondence regarding a project. It also contains information for you about the information services of COMMON. In other words, how do you submit things to CAST, what can you expect to see in CAST and the Newsletter. How do you submit information to the Newsletter, and what is the purpose served by the Newsletter. And some information regarding the Program Information Dept., better known to you as the Program Library. This will talk about the ordering procedures for material from the different machine type libraries, will talk about submittal procedures for programs, the PREP form, and the shipment analysis form. We hope the Reference Manual will be of great benefit to you. There is one other thing it will contain, a membership list which is indexed by company name and by user group number. The company names are in alphabetical order, and will give you the installation number. The full address of the installation representative is given under the installation code listing. Later, it will also have a geographic list, listed according to region of the country. I think that this describes what we have for you in the Reference Manual, and I'm sure you will be looking forward to receiving this within the next two weeks.

CHAIR -

The next division which we will consider is the Installation Management -

PAUL BICKFORD -

Thank you, Jim. I want to welcome all of you this morning to our meeting. I would like to begin by outlining the objectives of our Division. We primarily exist for helping to form guide lines to our management in the areas of Personnel and Operation Management. We presently consist of two projects. One in Personnel and Operations and, in a way, one in Education. As a result of the Boston meeting, we outlined some activities for this meeting, and are currently responsible for such presentations as the Job Description and Personnel Selection Presentation. Also a presentation on Programmer Evaluation. We also requested IBM to make a presentation on the Systems Reference Library and one on 360 Operator Training. We would like to solicit at this time any of you who are interested in participating in the activities of our group. We will have a planning session this coming Friday. We have about 30 people in our Projects. We would like to have more people participate actively in our group. So, if you new members or old members are interested in becoming involved in these activities, please be at the Planning Meeting on Friday. Thank you.

CHAIR -

I would like to particularly suggest to all members, new and old, - The Thursday afternoon session on the IBM Customer Engineer relationship should cover a discussion of the APARS, and of something called a Programming Systems Memorandum, which has just been made available by subscription. It lists all the APARS for a given programming system, and comes out every two weeks. It's the best source I know of to find out what bugs are where and when. It's extremely useful. Jim Taylor is not here; Dick, do you want to discuss the Systems Division?

DICK PRATT -

Frank said he was unprepared, I'm extremely unprepared. About all I can tell you about the Systems Division is that it consists of machine-oriented sessions, and you will find these in your program. Some of them have probably been changed. These will be announced. There are sessions for 1620, 1130, 1800 & 360, and these will take most of the day today, and part of the day tomorrow. I imagine you can find the session you are interested in by just looking through your machine type. There are some sessions scheduled at the same time you might want to split yourself between - somebody has already complained to me about that - unfortunately, there is only so much time. You just sort of have to take a choice or go back and forthe. I don't really have any information as to what is scheduled in any of these machine-oriented sessions, except for the 360. The 360 will have some presentations by IBM on PL-I, which has just come out in DOS, and on

DOS, Version 3, which has just been announced, and will be available in April. And, while I am standing up, I would like to say that I'd like to meet very briefly after this session with the Chairmen of the 360 committees, so we can sort of get things straightened out. I assume the rest of the mcahine-oriented sessions are already straightened out. If they are not, I don't know what to tell you to do about it.

CHAIR -

I should say that Dick is not the Systems Division Manager. He's the Chairman of the 360 Project, as well as a member of the Executive Board.

There is one comment that I would like to make about that - particularly in the 1620 area - there will be applications papers presented within the Systems Division. Where we felt that they were sufficiently machine-oriented that they would not be of benefit to anyone other than a user of a particular machine type, we put the applications papers in the machine-oriented sessions.

Bill Lane has an announcement, and Jim Tunney has some modifications on the agenda.

BILL LANE -

I notice that this is about the first and last General Session according to the agenda. I notice also that, according to the agenda, Friday afternoon until 3 P. M. there is a general time set out for planning of the next meeting. The next meeting will be in San Francisco at the Sheraton Palace and, getting word in for the people of the Golden State, we'd sure like to have you come out. We're quite proud of San Francisco and if you want something to see that is different go out to Ashbury and see the Hippies. I went out to check about a week ago, or two weeks ago, I gues it was, and they're fantastic. For those of you who are worried, they are not around the Sheraton Palace. But, anyway, I would ask that you harken to the call for papers, and also the call for help, because the Executive Board can't put on the meeting by themselves, and I sure as heck can't put it on by myself. We'd like you all to help. We'd like to have you consider today, to-morrow, and Friday morning, rather than just Friday afternoon, as to what you would like in the next meeting, and what type of papers you think would be appropriate, Please, when you do find these out, either get information to the Division Chairman or, better yet, get information both to the Division Chairman and to me.

In CAST 7 there is a little questionnaire-type form that all you do is check things off and send it back to me - it requires a five cent stamp now, I guess. We'd sure appreciate it. I'm looking forward to a good meeting today, tomorrow and Friday, as well as in San Francisco. Come on out. It's great in December - the weather's not bad. In fact it might even be warmer than in the East.

LAURA AUSTIN -

In conjunction with Bill Lane's announcement, I might mention for this Future Meetings Session that we have for Administration Division - so that you can be considering this, maybe I should tell you where the meetings are going to be so you would know whether you'd want to volunteer to be Program Chairman or Local Arrangements Chairman.
December of '67 will be San Francisco, as Bill mentioned -
April of '68 will be in Chicago - so anybody from the Chicago area - we'd be interested in having volunteers to help on that program.
September of '68 will be Philadelphia.
December of '68 will be Houston, Texas.
April of '69 will be Los Angeles.

That is as far ahead as I will go right now, but, if you are from any of these local cities, we'd certainly like for you to consider volunteering for help on those programs.

CHAIR -

Thank you, Laura.

Jim Tunney has some announcements and program changes -

JIM TUNNEY, PROGRAM CHAIRMAN -

Since there is going to be a series of program changes through out this meeting, I am going to try each time to go down the list in the same sequence these things appear in your program so that you can make the changes as I go -

The first change is on Page 14, I believe, the Session W2.2, which is 1130 - OK, page 12, in W2.2, in which there is the 1130 project, there will be a presentation on 1130 Commercial Applications by Mr. J. Elan.

Then on Page 16, W4.4, the PL-I presentation has been moved from there. That session will be chaired by Mr. Mc Ilvain, rather than Richard Pratt, and will include comments on DOS. The PL-I presentation has been moved to T2.1, which is on page 20 - that's the next item. T2.1, which is again Don Mc Ilvain's session, will have the presentation by IBM on PL-I. T2.1 is PL-1 under DOS.

The next change is on Page 22, which is T3.1. T3-1, of course, has the DOS Version 3, which is correct as stated. Down under T3.3, the paper on Expanded AUTOSPOT for 1130, which is B in T3.3, will be given by Charles Newman, instead of D. Carlson. Page 23 - The Education Project there - those presentations from IBM will be made by Mr. H. Codow& G. Wolf. Their names somehow were omitted here. That is in T3.6.

Down at the bottom of that page, in T3.8, R. Brennan is going to give that paper.

I'd like to talk to Jim Fisher after session here. His paper right now is scheduled in F3.1, and there is some question as to whether that's when it will be presented.

Also, on page 27, in F1.7, the paper by Mr. Groft will be moved to another session. I don't know where yet, but the one that is scheduled for 9:15 on AC Circuit Analysis will really take place at 8:30 inthat session. I gues that's all I have.

CHAIR -

As usual there are always afterthoughts -

PAUL BICKFORD -

One little note - we are interested in forming a panel here at the meeting of people, two or three people, interested in discussing 360 operator training. We would like for people interested to participate in this panel. It will be Session F2.2, on page 28. So if you are interested, and would like to participate in a spontaneous discussion here, please meet me up here after this session. Also, we are interested in getting together people who are interested in a CAI project - Computer Assisted Instruction - Bill Lane is interested in meeting with these people, so if you will see him here after the session, he will speak with you then.

CHAIR -

We are trying to cut this session short today, so that there will be time for registration for the people that didn't get there last night. IBM has requested permission to make a presentation on some new hardware -

Paul Manikowski of IBM will make the presentation.

CHAIR -

I heard the laugh when Paul read the statement of intent. How many of you people have received CAST 7? Possibly half. There is a letter in that, which is basically the same as the speech by Watson to GUIDE, discussing IBM's policy regarding Program Announcements. I don't know - we have not requested any specific coverage here, so I'll give you the gist of it.

IBM has adopted a policy that program announcements will be deferred until such time as they are reasonably certain that the program will do what the intend for it to do. In some cases, it may even mean that the program announcement will not be made until a program is actually in Alpha or Beta Tests. Because, in many cases, this would be too late to do the user any good, they have indicated that they will make a statement of intent, which indicates what they are trying to do, but makes no committment on their part to do it. Their men are saying - we are going to try to do this, it may not be exactly what we say here. We will do something in the area, but we may even abandon the project. It's not very good, but possibly better than slipping programs, and having unsatisfactory programs issued. In addition to that, there was a statement made by Watts Humphrey, of IBM, at SHARE XXIX, to the effect that there would be no extension to the FORTRAN AND COBOL capabilities of OS. Any extension of capabilities would be done in PL-I, unless there was very serious market pressure to implement an extension to FORTRAN OR COBOL.

With that - that's about what we had for the General Meeting. I know that there were a great many of you who didn't get registered last night. This hopefully will give you time to do so. Also give you time to catch up on the breakfast you missed.

Dave, or Jim Tunney - You know when the morning coffee break will occur and where, Jim? There will be coffee at 10 o'clock, outside in the registration foyer area. I'll see you then.

Pages 13 and 14 were not made available when Proceedings were published.

SESSION NUMBER W.2.2

SPEAKERS

NO FORMAL PRESENTATION. MEETING WAS CHAIRED BY LARRY ARMBRUSTER.

DISCUSSION

REQUESTS FOR INFORMATION AND ASSISTANCE COVERING THE FOLLOWING TOPICS WERE MADE

MIXTURES OF FORTRAN & ASSEMBLER LANGUAGE
COMMERCIAL SUBROUTINES W/ OVERLAPPED I/O
ALLOWANCE FOR MECHANICAL FAILURE IN IDEAL FORTRAN
PLOTTER MALFUNCTIONS
BETTER ASSEMBLER LANGUAGE INSTRUCTION MATERIAL
SOURCE CODING OF THE OPERATING SYSTEM
DISK COPYING PROBLEMS ON THE 1800
EARLY MORNING START PROBLEM
USE OF OTHER PLOTTERS RATHER THAN 1627

GENE LESTER OF IBM WILL PRESENT A TALK ON PRIORTY, INTERRUPT PHILOSOPHY AT SESSION W4.2.

A PROPOSAL TO SPLIT INTO SCIENTIFIC AND COMMERCIAL SUB-PROJECTS WAS VOTED DOWN.

A SHOW OF HANDS INDICATES THAT ALMOST ALL PRESENT USE ASSEMBLER LANGUAGE TO SOME EXTENT.

SESSION NUMBER W.2.4. & W.3.4.

SPEAKERS
    360 USERS THEMSELVES
    MODERATED BY R.L. PRATT & D.R. MC ILVAIN

DISCUSSION

OPEN DISCUSSION ON PROBLEMS ARISING IN 360 INSTALLATIONS - MOST ATTENDEES CONCERNED WITH DOS. LACK OF PROPER IBM ATTENDENCE HINDERED RESPONSE TO MANY ITEMS. THE FOLLOWING ITEMS WERE REVIEWED

DISCUSSION OF ERROR DIAGNOSTICS IN FORTRAN & LACK OF INTERPRETATION.

BETTER REFERENCING AND INDEXING OF MANUALS IS NECESSARY FOR THEIR EFFICIENT USE.

PSM NOW TO BE DISTRIBUTED BY IBM WILL GIVE USERS A BETTER REFERENCE TO EXISTING APAR'S, ANTICIPATED CORRECTION TIME, AND POSSIBLY IMMEDIATE TEMPORARY CORRECTION. THE RETAIN SYSTEM FOR THE FE'S AND SECOM FOR SE'S IS IN USE BY IBM TO AID THE DISTRICT OFFICES TO BETTER SUPPORT THE USERS.

FORTRAN DOES NOT AUTOMATICALLY OVERFLOW UPON SENSING CHANNEL 12 PUNCH. A PATCH EXISTS FOR LEVEL 9, BUT WOULD NOT FIX THE MOST RECENT ISSUE OF DOS.

029'S ARE SENSITIVE TO REPRODUCING HEX-PUNCHED CARDS AND ARE LIKELY TO BREAK THE CODE PLATE, PRINTING OR NOT. IBM MENTIONED THAT THE 029 HAS A NEW FEATURE AVAILABLE (CODE INHIBIT FOR $3.00 A MONTH) TO LOCK OUT THE EXTENDED SET, REDUCING THE KEYBOARD ENTRY POSSIBILITY FROM 64 TO 48 CHARACTERS. IT WAS REPORTED THAT AN 024 IS SATISFACTORY FOR REPRODUCING THESE CARDS BUT SOME HAVE HAD POOR EXPERIENCE HERE TOO.

REVIEW OF PL/I EXPERIENCE UNDER DOS. EXPERIENCE WAS LIMITED BUT INDICATED GOOD ACCEPTANCE WITH THE COMMENT OF POOR OBJECT TIME DIAGNOTICS.

IT WAS SUGGESTED THAT A FORM BE MADE AVAILABLE TO USERS FOR USE IN SUBMITTING PROGRAMMING TIPS TO THE NEWSLETTER. THIS WILL BE PURSUED BY THE DOS COMMITTEE.

JOB ACCOUNTING (AUTOMATICALLY) IS STILL A DESIRED FEATURE IN THE SYSTEM SUPPORT. HOPEFULLY SHARE'S PRESSURE IN THIS AREA WILL AID OUR REQUESTS.

THE DSR TYPE 3 PROGRAM IS AVAILABLE FROM THE LOCAL OFFICE FOR INCLUSION AT SYSGEN TIME FOR LOGGING OF DIAGNOSTICS.

SESSION NUMBER W.2.6 OS PROJECT

SPEAKERS
NO SCHEDULED SPEAKERS.

DISCUSSION
WE PLANNED AN AGENDA FOR THE FOLLOWING SESSIONS. THE THIRTEEN ATTENDEES DISCUSSED PROBLEMS OF CONCERN TO THEM BRIEFLY. AFTER ESTABLISHING AN AGENDA, IBM'S REPLY TO BOSTON RECOMMENDATIONS WAS READ AND COMMENTED UPON. EVERYONE IN THE OS COMMITTEE AGREES THAT COMMON SHOULD SUPPORT THE EFFORTS OF USASI X3.6 TO OBTAIN A NATIONAL STANDARD FOR HAND CODED GRAPHICS.

SESSION NUMBER W.3.2

SPEAKERS

JIM ELAM OF IBM SPOKE ON 1130 COMMERCIAL APPLICATION PROGRAMMING.
DON GARDNER SPOKE ON SOCALS AND REMEDIES HE HAS FOUND.

DISCUSSION

DAVE DUNSMORE HAS BEEN ELECTED CO-CHAIRMAN OF THE 1130 PROJECT.
THE FOLLOWING SUBJECTS WERE OPENED TO THE FLOOR FOR DISCUSSION-

- IMPROPER DIMENSIONING
- DISK READ ERRORS

SESSION NUMBER W.3.6

SPEAKERS

MR. G.W. GOESCH, IBM CORP. ON THE SYSTEMS REFERENCE LIBRARY

COMMON
Cincinnati, Ohio

PROJECT:	Management Installation Division, Operation Project
SUBJECT:	The Systems Reference Library
SPEAKER:	Mr. G. W. Goesch, Manager, Product Publications IBM Corporation, San Jose, Calif. Telephone (408) 227-7100
FOR PRESENTATION:	Wednesday, September 6, 3:30 PM, Session IV 8 Pages Text

The System Reference Library

I'm Gordon Goesch, Product Publications Manager, IBM, San Jose, California. While I am not involved in the development of all types of IBM publications, other areas of our organization have the same mission for similar publications in various parts of the country.

When one looks at a manual, it is not impressive, there doesn't seem to be much to it, but when you get into all tho ramifications in publications, it's a little like an iceberg in the ocean - most of it is below the surface of the water.

We haven't solved all our problems and I'm not sure that we will completely - but you may be assured that we (like yourselves) are certainly constantly trying to improve our operation.

That is the reason I am always happy to talk to groups such as yours about our publications. Because it gives us an opportunity to discuss with you our Publication Library, its organization, its purpose, and revision service - as well as to update you on the library's operation procedures - because even with a good system - you must understand how to use it - if it is to be effective for you.

I think that this type of a meeting can be a two-way street for information: We develop the literature; you provide feedback. We do get feed back from you via Reader's Comments forms - we want more of your comments and we certainly appreciate them.

I don't know how knowledgeable you are about our publications; therefore, for the benefit of the new members and also for the updating of the veteran members, I'll run through a few slides which I think will tell you the publication story.

Slide #1
BOL

To begin, we have what we call the IBM Branch Office Library (BOL). BOL contains much information. Not all, but the major portion of BOL is made up of publications for users of our equipment.

You have probably seen the publications display in the main convention lobby. The display, I am sure, contains publications of interest to you. Many of the publications on display have been published since your last COMMON meeting. Feel free to examine them in depth but please do not take them away, as there is only one copy of each and we want many people to benefit from the display.

Slide #2
FYI

In order to call to your attention what publications we have For Your Information

Slide #3 Swamped	and to help you avoid being swamped by ordering blindly as the man shown in the slide
Slide #4 SRL	we have developed the Systems Reference Library (SRL). As you probably know the SRL is a rather extensive library system.
Slide #5 Library Shelves	However, before you start pulling arm fulls of manuals from its library shelves
Slide #6 Mr. SRL Helps Select	let Mr. SRL help you make the correct selection.
Slide #7 What, Where, How	He will acquaint you with the SRL and tell you: a. What is available b. Where you can get the information c. and, how you can go about getting it.
	First, you must make some determinations:
Slide #8 System	a. What System or Systems interest you
Slide #9 System X	b. What size library do you want - everything for that system or just those parts that pertain to your specialty?
	Considerable thought and effort was spent in the design of the Systems Reference Library - for both the needs of the reader and the type of publications necessary to support our products.
Slide #10	Each SRL is an encyclopedia for a <u>particular</u> system - with separate publications for <u>Major Subject</u> areas. It consolidates all the basic reference literature necessary for you in: Planning Programming Installing, and Operating that system.
Slide #11 SRL Key	The key you need for opening any of the System Reference Libraries
Slide #12	is the SRL Bibliography for the <u>given</u> system.
	Currently there are 13 major System Reference Libraries, ranging from the 1130 to the System 360 - and of course each system has its own separate Bibliography. You will find,

however, that Bibliographies make cross references to pertinent publications of other systems.

Each Bibliography is actually an Index of all the current publications about a specific system. In it, publications are listed both by subject code and by machine number - and it contains abstracts describing each available publication. (We will discuss subject code a little later.)

You may have noted that now Programming Logic Manuals (PLMs) are listed in the Bibliographies. While they are subject to restricted distribution, they are available if a real need is evidenced. The PLM details the internal logic of the program (like a large map of the listings).

Slide #13 Bibliography & SRL Newsletter

Now, how do we update the Bibliography?

Each Bibliography has its own Newsletter (its color is green) and it is issued monthly (when there are changes). The Newsletter updates the Bibliography, provides abstracts of new publications, and lists Type I programs with their latest modifications.

Actually, the Bibliography newsletter is a current "accumulative Index of Publications and Programs" available for a given system.

From the publishing mechanics point of view, Bibliographies are periodically scheduled for revision - when that occurs, the current information from the newsletter is merged into the Bibliography.

Slide #14 SRL Masthead

Each SRL publication, listed in a system bibliography, is identified by a file number and a form number, located on the upper right hand corner of the publication cover - as shown in the slide.

The file number performs two functions: the first part, specifies the system (s) number; the last two digits the subject code

The Subject Code is made up of a group of two-digit numbers (00-99) assigned to the various system components, e.g.

00	Includes, Bibliographies, System Summaries, Configurators
01	Machine System (CPU)
03	Input/Output Units
05	Magnetic Tape Units
20-50	Programming Systems

A recent change now places Application Program manuals under code 60.

The subject code 13 shown in this slide, indicates that this publication is about Special and Custom features.

The form number is self explanatory; however, the form number suffix indicates the editorial level of the publication.

Slide #15
SRL, TNL

Because of the dynamic nature of computer technical information, frequent changes occur.

When changes occur, technical newsletters (TNLs) are issued to update the publications involved.

Consequently, not only is the Bibliography updated by its own newsletter but each and every SRL manual can have its own TNL.

In most cases TNL packages are made up of an identifying cover page and replacement change pages for the parent publication -- when you receive such a TNL you merge it into its parent manual and throw away the old pages.

Incidentally when a publication is ordered, you will automatically receive the latest technical (suffix number) level copy as well as all the outstanding TNLs available against that publication.

Slide #16
TNL
Mast Head

This slide shows the upper right hand corner of a TNL. It indicates how the TNL identifies itself with its parent publication.

It carries the file number and form number of the parent publication it updates (and it is form number suffix sensitive)

Below that is the TNL's own number, publication date of the TNL, and the form numbers of any previous TNLs outstanding against the parent publication.

Incidentally, all page replacement TNL pages carry similar identifying information.

Outstanding TNLs are incorporated into the parent publication when it is being revised. TNLs may also be merged into the parent publication when it is being reprinted.

Information regarding the technical level of a publication and the TNLs that may have been merged into it may be found inside the front cover of any SRL manual.

Slide #17
SRL, NL
Parent Pub.
Manual

This slide shows a manual, a corresponding TNL, and a green SRL newsletter. With this combination on hand you have all the publishing reference you need for the parent manual.

Keep in mind that the best SRL publishing information source is the green SRL newsletter, because it not only updates its own parent publication (the bibliography) but also lists all the existing publications that are current for that system as well as their outstanding TNLs

Slide #18
2 Biblios.
2 NLs

and of course that each major system has its own bibliography.

Slide #19
In-Out

Thus, the SRL system keeps you well posted on what is in (current) and what is out (obsolete) for an effective library.

Slide #20
Wrapped Up

Basically, you have at your disposal a "living-doll" of a library system.

Via the SRL, you can develop and maintain a library for one system or more - and tailor your library to your own needs.

Slide #21
DPT

An additional source of information is the "Data Processing Techniques" (DPT) Bibliography (Form F20-8172). It indexes a series of publications of techniques for Study, Analysis, Design, Implementation, Programming, Documentation, Installation, Operation, Scientific, etc.

This Bibliography is also updated by its own Green Newsletter.

Slide #22
Series of
DPTs

This slide shows some of the DPT manuals.

Slide #23
KWIC

Another excellent Index for your use is the KWIC Index of Marketing Publications (Form 320-1621).

Slide #24
KWIC &
TNL

It is published quarterly and updated by a monthly TNL.

The KWIC Index is based on an abbreviated 30-position publication title - listing and cross-referencing publications by the important words in the title.

The KWIC Index is listed in five ways:

1. Alphabetical
2. Machine or System Type
3. Form Number
4. Type I & II Programs in System Sequence
5. Type III & IV Programs in System Sequence

Slide #25
3 Way List
1-2-3

For example, this slide shows 3 separate word listings for a single publication - "Programs for Petroleum Engineering"

Slide #26
2 More List
4-5

and here, the same publication listed by machine number and by form number.

The latest KWIC Index was prepared from 11, 546 publication titles which generated 29, 122 listings.

Actually, the KWIC Index lists many publications other than SRL publications, such as Executive Guides and Brochures, tools and techniques manuals, applications manuals and briefs, educational material,

Another way of putting it is that <u>all</u> SRL publications are <u>Marketing Publications</u> but all marketing publications <u>are not</u> SRL.

Hence the KWIC Index provides wide IBM marketing publication coverage.

Therefore, to start a System Reference Library, it is necessary that you contact your IBM representative, and work through him, using the three key publications we have talked about, the:

1. Bibliography
2. its SRL newsletter
3. KWIC Index of marketing publications·

You can select and build your own reference library to support your system.

However, please do not order your publications by writing to Product Publications (the address shown on the manual) or our Distribution Center in Mechanicsburg, as it will only delay the order. You must order through your local IBM representative.

Now that you have established the base for your library, let's discuss its <u>maintenance.</u>

a. If you desire, you may continue ordering specific publications through the IBM Branch Office.
b. However, you may prefer to subscribe to the "Publication Revision Service" that the System Reference Library offers.

Slide #27
SRL, TNL
Revision
Service

The Revision Service provides automatic shipping of revised manuals and newsletters directly to you without having to go through the IBM Branch office each time.

However, Subject codes 00-60 only are supplied by the Revision Service. This excludes installation supplies, education literature, and other supplementary information.

Here's how it works:

Slide #28
Subscrip.
Card

The IBM representative fills out a subscription card with you, using the green SRL Newsletter to indicate those publications on which you want the updating afforded by the subscription service. After the card is approved, IBM Distribution Center takes over and a single copy of each TNL or revision involving the indicated items will be mailed to you.

Slide #29
Mailman
w/Pkg.

Remember that only one copy per subscription can be mailed. Additional copies must still be ordered through your IBM representative, just as were your initial manuals. Any changes in your Revision Service are made via a new subscription card. Again, if multiple copies are ordered through the Revision Service, it only complicates and delays matters.

It is important to set up a library and assign responsibility for updating and maintenance to make the service effective.

Slide #30
SRL & Rev
Pkg

Now, you are all set, you know <u>what</u> is available, you know <u>how</u> to select the publications, and you know <u>where</u> to get them.

Slide #31
The Key
Is Yours

The key is yours.

Slide #32
Key to
Knowledge

Use this key to open up a tremendous amount of timely knowledge about your system through the <u>IBM Systems Reference Library.</u>

Slide #33
SRL

I hope that the result of all this, will make you as happy as the man in the next slide - careful planning of your library may help!

Most of you have probably noticed the Reader's Comment Form that is appearing on the back of many manuals these days. Some of you may have filled one or more out. The response to these has often been very gratifying and we appreciate it.

I want to encourage you to use them as it is one of the means, along with meetings like this, by which we get the feedback from our readers that is essential if we are to improve our publications and make them more useful to you.

This form is self-addressed and post-paid and will be directed to the correct publications group. As a reminder, please don't use the form to order manuals as we have to redirect such orders back to the Branch Office with a consequent delay.

On behalf of the publications groups, I want to thank you for your attention and for your comments and pledge to you our whole-hearted effort in providing first rate publications support for your systems.

Thank you again for giving me this opportunity to talk to you.

SESSION NUMBER W.3.7.

SPEAKERS
1. WADE NORTON
2. NORMAN GOLDMAN
3. NORMAN GOLDMAN

DISCUSSION
1. REREAD
2. GENERATING WITH A 2 DRIVE CUSTOMIZED SYSTEM.
3. ACCOUNTING ROUTINES.
4. VARIETY OF TOPICS.

SESSION NUMBER W.4.1

SPEAKERS
    MR. HERBERT RUDERFER

DISCUSSION
    MR. HERBERT RUDERFER PRESENTED BOTH THE FACILE & FACET PAPERS.
    MR. WILLIAM SILER PRESENTED HIS PAPER IN ANOTHER SESSION.

SESSION NUMBER W.4.2

SPEAKERS

GENE LESTER OF IBM SPOKE ON PHILOSOPHY.

PAUL MANIKOWSKI OF IBM SPOKE ON EDUCATION INCLUDING COURSES, PROGRAMMED INSTRUCTION, AND LITERATURE AVAILABLE.

DISCUSSION

TWO NEW PIECES OF LITERATURE TO AID IN ASSEMBLER LANGUAGE ARE AVAILABLE.

1. PROGRAMMING THE 1130 AND 1800 BY R.K. LOUDEN - PRENTICE HALL.
2. P.I. COURSE - CONTACT DON JOHNSON
   HINSDALE CENTRAL HIGH SCHOOL
   HINSDALE, ILLINOIS

SESSION NUMBER W.4.4

SPEAKERS
MEETING MODERATED BY D.R. MCILVAIN.

DISCUSSION
THE PRESENT STATUS OF THE DOS COMMITTEES IN GUIDE & SHARE WAS GIVEN. ESSENTIALLY NO COMMENTS HAD BEEN RECEIVED FROM THE GROUP AT LARGE PRIOR TO THIS MEETING - PRE-SUBMITTAL IS NECESSARY FOR EFFICIENT OPERATION OF THIS COMMITTEE. COMMENTS ON FORTRAN & COBOL WERE PRESENTED BY MESSRS. GWILLIAM & CUNNINGHAM. THE RESPONSE FROM IBM TO OUR QUESTIONS ON FORTRAN WAS RECEIVED BUT WAS VERY POOR AND THE QUESTIONS WILL HAVE TO BE RESUBMITTED. GUIDE IS MEETING IN ENDICOTT WITH THE IBM DOS IMPLEMENTATION GROUP LATE IN SEPTEMBER, 1967 AND COMMONS DOS COMMITTEE WILL HAVE REPRESENTION AT THIS MEETING AND WILL EXPLORE SOME OF THE ITEMS OF INTEREST TO COMMON.

SESSION NUMBER W.4.6

SPEAKERS

PANEL DISCUSSION I
DR. R. GABRIEL
MR. M. GOLDBERG
MR. N. GOLDMAN
MR. P. KOEPSELL

PANEL DISCUSSION II
DR. R. GABRIEL
MR. H.B. KERR
MR. P. KOEPSELL
MR. D. LA PORTE

DISCUSSION

1. ECONOMIC JUSTIFICATION OF THE UNIVERSITY COMPUTING INSTALLATION.
2. EQUIPMENT SELECTION FOR COLLEGES AND UNIVERSITIES.

ATTENDANCE 36

SESSION NUMBER W.4.7

SPEAKERS

FRED W. MATEJCEK, COMPUTER CENTER ON NIU SYSTEM/240 A DATA RETRIEVAL SYSTEM AS APPLIED TO A LIBRARY INVENTORY

NIU System/240

A Data Retrieval System As Applied To a Library Inventory

Fred W. Matejcek
Computer Center
Northern Illinois University

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

Much of the success of this system, for it has been operating successfully more than a year now, must be attributed to: Dr. Robert Hunyard, Head of Northern Illinois University's Audio-Visual Department for recognizing the problem and the possibility of a computerized solution; Mr. Clyde Givens, Director of the Computer Center, on whose analysis the system was initiated; and Michael Roldan, Director of the Film Library and his staff, who learned to use the various facets of the system as they were designed to be used, without which the best design never becomes operational. Finally, the existence of the system is in no small way due to the patience of my spouse, Elizabeth.

LIST OF ILLUSTRATIONS

INTRODUCTION

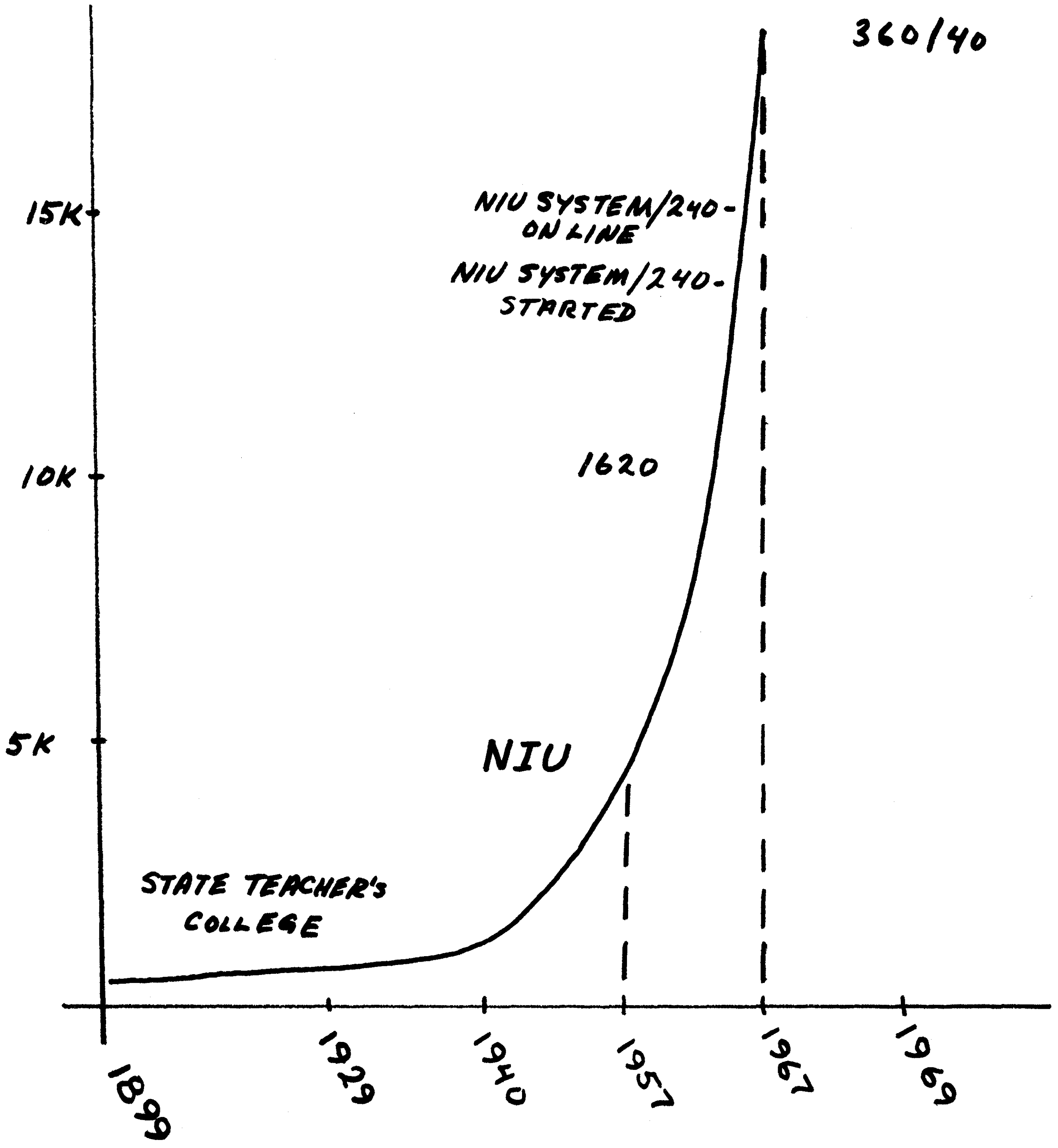
NIU System/240; Northern Illinois University and the system we developed. It took on the project number "240" identifying the application area with which we were dealing within the University, Audio-Visual Aids. This is a data retrieval system developed on the 1620 to solve the inventory problem faced by the film library at NIU in the AVA Department. The manual solution to the problem was unable to handle the growth both in terms of the student body (see next page) and something refered to as the "information explosion". This is the first point we see under "justification" for the system, the economic aspect. Extra clerical staff to handle the volume could not be hired for the same expenditure involved in the operation of System/240.

The second point under justification is better and faster service using System/240. User requests that are received in the morning are confirmed by the afternoon mail. This is in comparison to the normal two week response time by other libraries in the area.

The third and last point is the additional predictive information to be gotten from the system based on the records kept by the system. So much for justification. How new or unique is the system?

NORTHERN's Growth

360/50



A review of the literature shows that NIU System/240 was, at the time it went on-line in April of 1966, the only application of its kind in the country, and is to date the most comprehensive of exisiting systems, irrespective of hardware. How did we arrive at this solution? To answer this question we will look at the problems facing analysts when embarking upon the creation of any data retrieval system to solve an inventory problem and then look at how we faced these problems.

THE PROBLEM

To give structure to the following discussion we will separate the problems and their respective solutions into four classes. This taxonomy covers the basic problems faced in the design and implementation of a data retrieval system.

Our first classification covers the creation of files; something common to the early phases of most systems, because in creating a new system or converting an old system from manual or other means, the files must be put into machine usable form. The second class of problems concerns the actual servicing of requests by users for the item contained in our inventory. The third is the maintaining of the inventory. We must have a means of deciding how many of a given item to have on hand. The last basic class of problems is that of maintaining records. Our problems demand that we maintain records on both our users or customers and our inventory. Before elaborating on these four problem areas and providing the corresponding solutions to the problems, let's look at the special conditions which are placed on our example application, the NIU film library.

A RECIRCULATING INVENTORY

A film library is a special sort of inventory. It embodies a recirculating scheme by which films are supplied to the user and then returned to the shelves of our inventory. Therefore the normal input which maintains the "on hand" number of items that we have decided upon, is from the user, not a supplier. In other, or what we may call "one-way" types of inventories, the input which maintains the "on hand" number in the inventory is from an outside supplier. The only time which the recirculating inventory gets its input from an outside supplier is when we wish to change the "on-hand" level because of increased demands. In designing the system, this recirculating scheme forces special considerations in three areas. All three fall within the scope of the inventory file maintenance.

First, because we "reuse" items, we must keep track of each and every item within a given type, rather than simply being concerned with the total number of that item available. We must know whether or not print three of a given film will be back in the library in time to fill another request.

Second, we need a method to tell us when to go to the outside supplier and beef up our inventory. This is more complicated in the recirculating inventory than in the one-way inventory.

The third manifestation of our special conditions is possibly a blessing in disguise. Because quite often orders are placed more than a year in advance, our files must be large and complex, but the potential blessing comes in the preview of demands to come that we can glean from these advance orders. But this is only potential. We must take advantage of it.

Having now looked at the justification for the system, the structure of the discussion to follow and the special conditions a film library places on an inventory system, (and hopefully having put you in mind of the special conditions of your particular inventory problem) let's see what we did to cope with the four general classes of problems embodied in the design and implementation of a successful inventory system.

THE NIU SOLUTION

First is the area of creating new files. We will consider our basic tools in terms of hardware and then software. The hardware consists of an IBM 1620 Model II with 60K, which has been updated to include a 1311 disk drive and 1443 printer. The 1311 disk drive is the medium that we use for our mass storage of records. The software will also be considered in two parts; that which is supplied by IBM and that which is supplied by the Northern Illinois University Computer Center.

The Monitor I is the normal support system supplied with the 1311 disk drive but as used, has been somewhat modified from the original IBM version. Though we do compile FORTRAN we do this utilizing another disk thus enabling us to delete FORTRAN II-D, its subroutines, and some unused utilities from our production disk and in turn make this room available to System/240. Since three of the four basic programs are in SPS and 90 percent of our running is execution rather than compiling we do not really lose any flexibility.

The four programs developed as the full compliment in NIU System/240 are named AV-LOD, AV-DLY, AV-RPT and QIKLOD. AV-RPT is the only one written in Fortran and is used for our quarterly reports. AV-LOD and AV-DLY are the only ones that are disk resident and used on a daily basis. QIKLOD is a key part of our back-up.

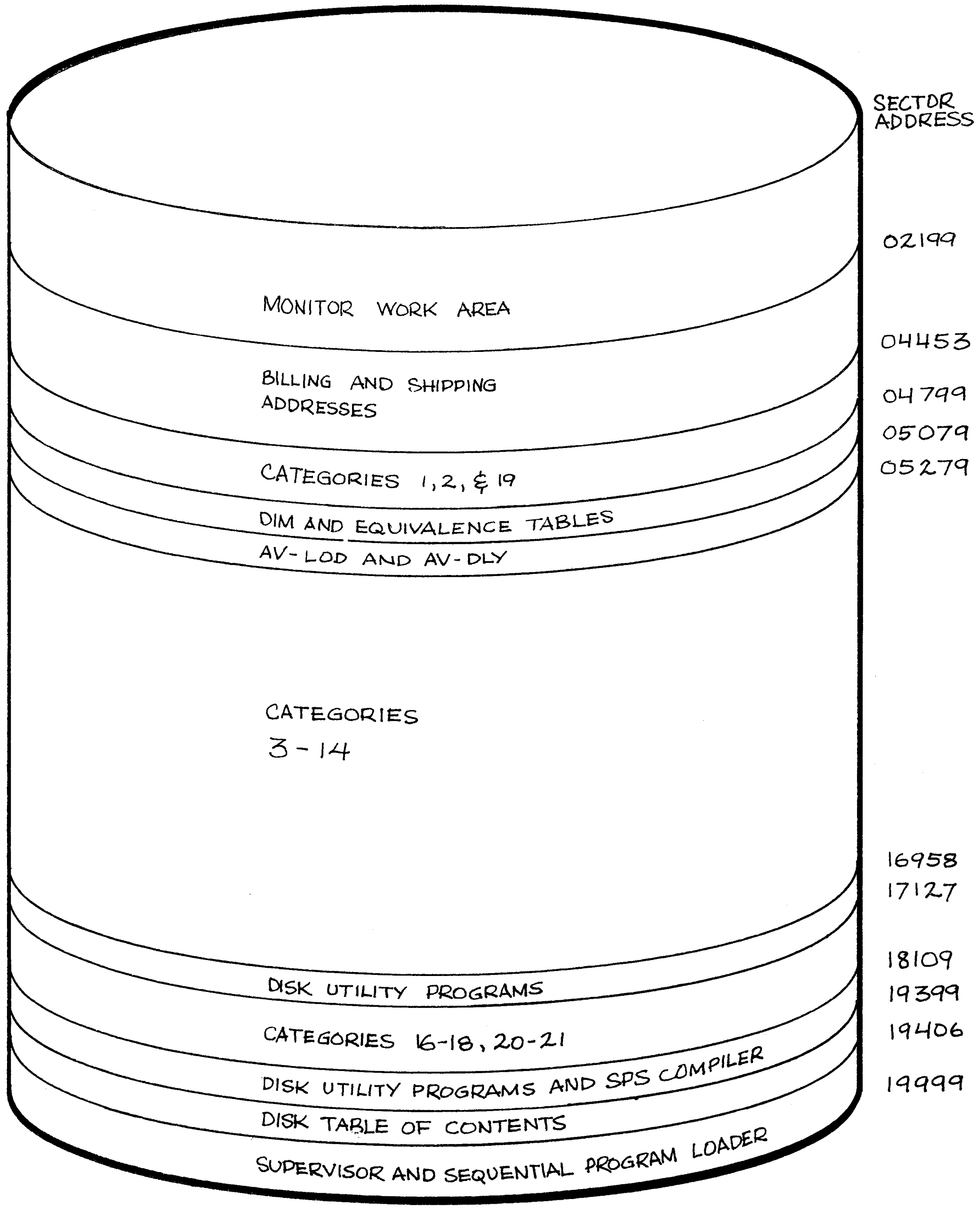
Since we have only a one drive system, a strict disk to disk back-up can not be provided, but with QIKLOD and the card output that is produeed during our daily procedures we are able to provide card back-up. Figure I shows the status of our disk and if you are familiar with the normal organization you can see where things have been rearranged or deleted. This includes limiting the Monitor work area to the minimum 11 cylinders.

There are basically two files in the NIU/CC software. The one is a file that contains all the information pertaining to our users. This file is broken again into two parts that may be termed our "billing and shipping files". We have run into the necessity of creating these two files by the nature of our users.

In dealing with schools or other large institutions we are often asked to bill a central office or school district, but in turn they want the items, in our case films, sent to individual users within their system. So we are faced with the problem of billing to one address and shipping to several addresses within that particular users juristiction. We will look closer at these files as we get into the structure of the actual records. The other file that we keep on the disk is a complete file of all the items, again in our case these items are films.

Our film library is divided up into 21 logical categories by subject matter. We use these logical categories as actual physical categories in our disk organization. For each category

Figure I



SECTOR ADDRESS
02199
MONITOR WORK AREA
04453
BILLING AND SHIPPING ADDRESSES
04799
05079
05279
CATEGORIES 1, 2, & 19
DIM AND EQUIVALENCE TABLES
AV-LOD AND AV-DLY
CATEGORIES 3-14
16958
17127
18109
DISK UTILITY PROGRAMS
19399
19406
CATEGORIES 16-18, 20-21
DISK UTILITY PROGRAMS AND SPS COMPILER
19999
DISK TABLE OF CONTENTS
SUPERVISOR AND SEQUENTIAL PROGRAM LOADER

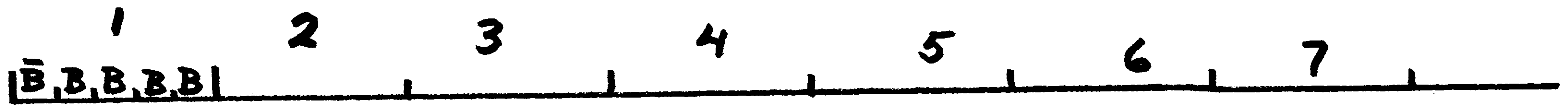
that you would find in the film catalogue you find a corresponding physical category on the disk. Down near the bottom of Figure I you will find a "Disk Table of Contents" at sector location 19406. This disk table of contents is our systems table of contents that tells the software what the status of our files is. In other words, what condition the last program that operated on these files left them in. Referring then to the next figure, Figure 2, we see the "Dynamic Table of Contents", which is in reality the table we see located at 19406 of Figure I. The static table exists only in the programs themselves because it is actually unchanging from the time of the original compiling of the program. On the other hand, any of the programs in System/240 may modify the information stored in the dynamic table of contents, but every time it is modified it is restored to the disk destroying the old information. The static table contains one entry per category.

Each entry is the beginning sector address of a given category. Looking for category 1, we would pick out the first field and would find that a given address in that field is the beginning sector address of category 1.

The dynamic table of contents contains three entries per category on the disk. The first field represented by "A's" is the active level to which the category is filled. In its initial form, when a category is empty before we have loaded any films to it, we would find that the field of "A's" or the active level

DISK TABLEs of CONTENTS

Static -



Dynamic -

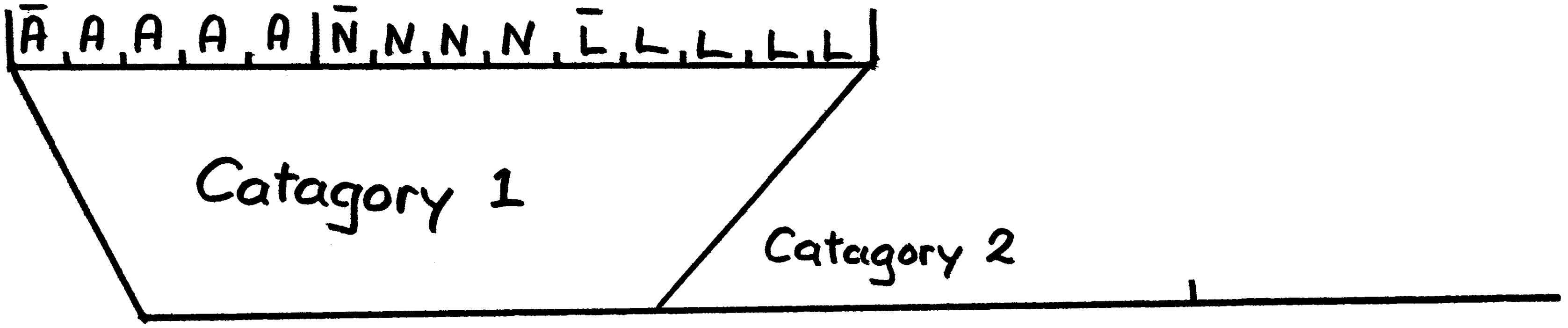


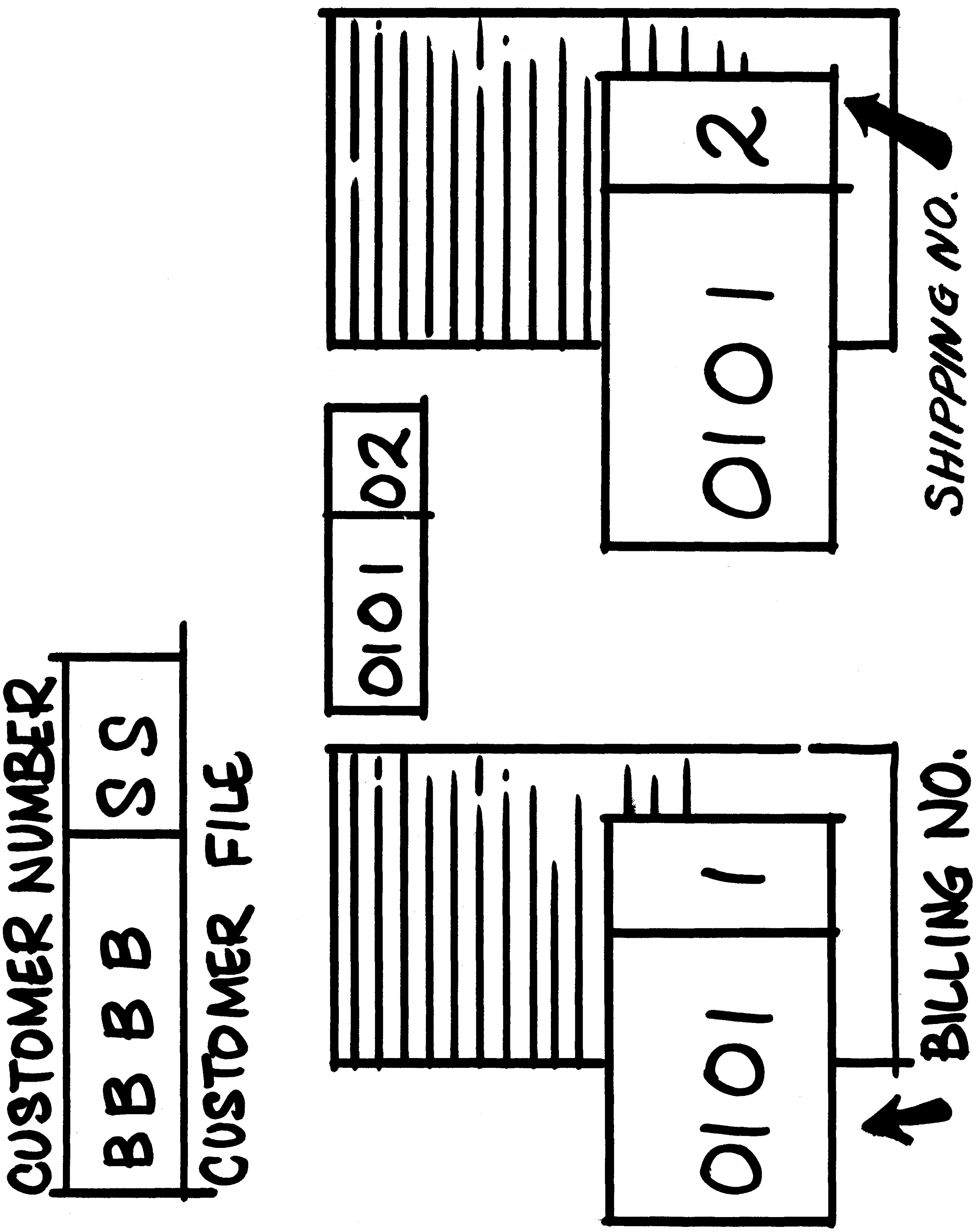
Figure 2

would be exactly equal to the field of "B's" or the beginning address of a given category, for the active level would be the beginning address of the category. The second field, or the field of "N's" indicates the last film number within that given category. Each category starts with a film number one and goes on to the last number within that category. The last number is the number that appears in this field so that in our search we are able to check immediately the last number and see whether or not the film specified is a legal film for that category, i.e. if it falls within this range. The last field or the field of "L's" as it appears, is the limit to which we may fill this category. When a category is completely filled, the converse of the situation we mentioned earlier occurs. The field of "A's" and the field of "L's" are equal because the category has been filled to its limit. Within that static table there are two additional entries, one is the beginning address of the customer, or the billing file; the other is the beginning address of the shipping file.

Now going to the organization of our two customer files, if we have an individual requesting a film and the request indicates that he is to have the film sent to the address he is to be billed at, this customer is entered in only one of our two files - he is entered in our billing file. If though, we have a customer that wishes to be billed at an address separate from the address to which he is having the film shipped, he is entered in both files. We give each customer a four digit number. If he is a customer with the same billing and shipping address this is all the number that he gets. If the customer does desire a separate shipping address, one separate shipping address or several, then on the initial cards that we use to create his records on our disk file the two cards that contain his billing address also contain a 1 in card column 7.[1] This indicates that there will be shipping addresses to follow these billing addresses and these will have the same four digit customer number but will also have a two digit shipping number tacked onto the end of this. These records are then filed on our shipping address file. Then we have a billing address number which is tied to one or several shipping addresses and a sequence number which makes each shipping address unique. If we look at Figure 3, we see the example of customer 101 in our billing file with a 1 coded following the 101 which indicates that in the shipping file we will find at least one shipping address if not more for him. When we make a request for the customer number 10102 this request is serviced by going to the billing file, finding customer 101, finding out whether or not the 02 is valid for this customer - in other words whether or not the customer does have separate shipping addresses by the fact that there is a one following his number on the file and then going to the shipping file and finding customer two or shipping address two of customer 101.

[1]Card Formats appear in Appendix A.

Figure 3

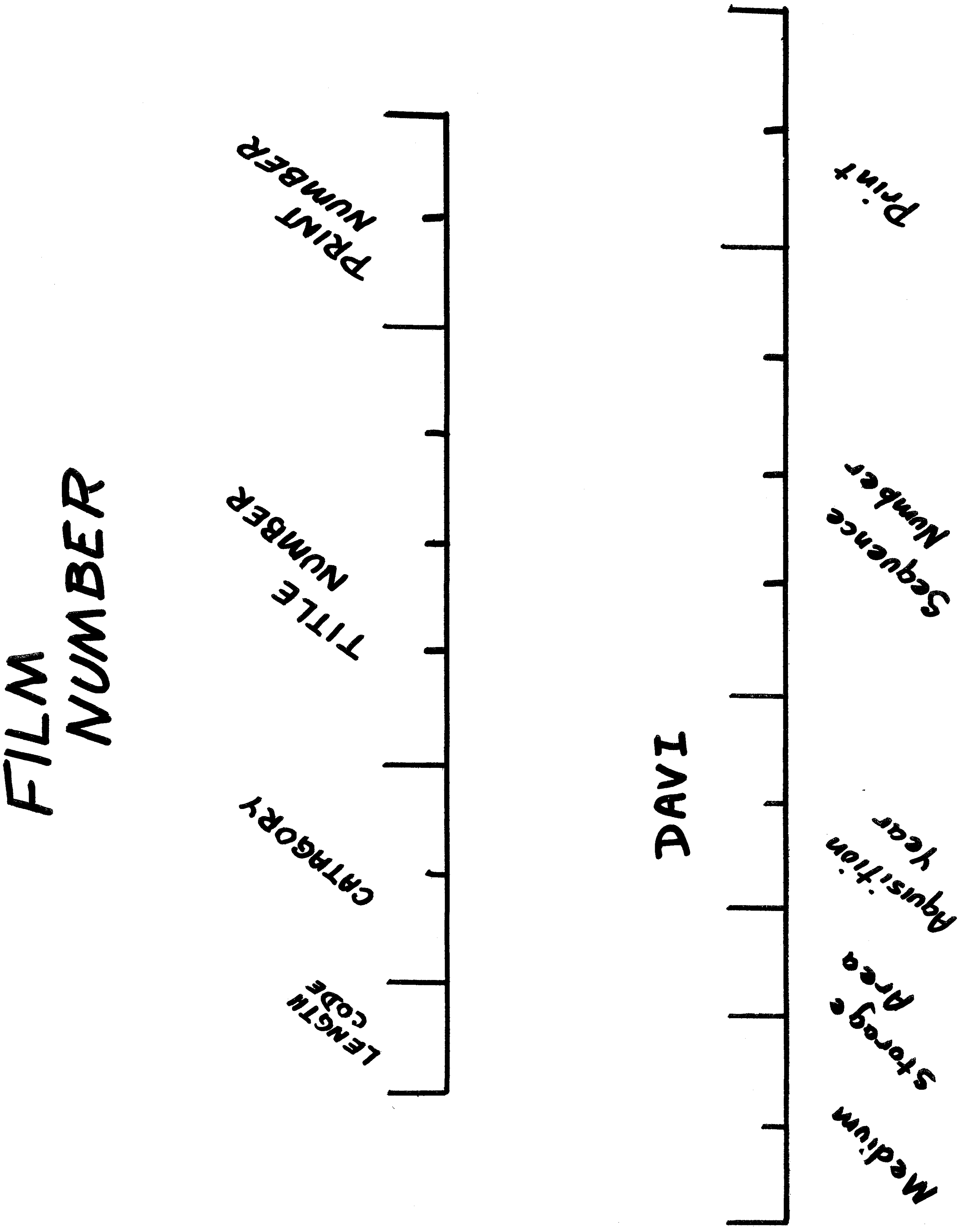


CUSTOMER NUMBER
BBBBB
SS
CUSTOMER FILE
0101
02
0101
2
SHIPPING NO.
0101
1
BILLING NO.

Moving on to the organization of our film file, we'll look at the key to this file or our film number. In Figure 4, we find an example of our film number. The first digit of this film number is a length and color code. The two digit category which is the logical category mentioned above and the physical category that we find on the disk. We have then within the category a four digit number which uniquely identifies a particular title of a film. Tacked on to the back of that we have a two digit number which uniquely identifies a given print for that given film. Here again we see our special conditions showing up. In this recirculating inventory we have to keep track of not only the total number of a given item but of each and every item as a unique entity. This is our print number. We have multiples of a given item, i.e. a given title of a film. Very popular films have several prints.

A footnote to this; the DAVY group which is to the educational film industry as COMMON is to IBM, is trying to come up with a coding system to be presented to the Federal Government as a suggested national standard for the entire educational film producing industry. Though we did develope Northern's numbering scheme independently, we find a great resemblence to the DAVY number in the one created for System/240 at Northern. We have the two digit medium which corresponds to our category, one digit storage area which corresponds directly to our length code, a sequence number of four digits (this is our "title number") and the two digit print number which we just refered to. In addition they carry a two digit year of acquisition number. This

Figure 4


FILM NUMBER
PRINT NUMBER
TITLE NUMBER
CATAGORY
LENGTH CODE
DAVI
Print
Sequence Number
Aquisition year
Storage Area
Medium

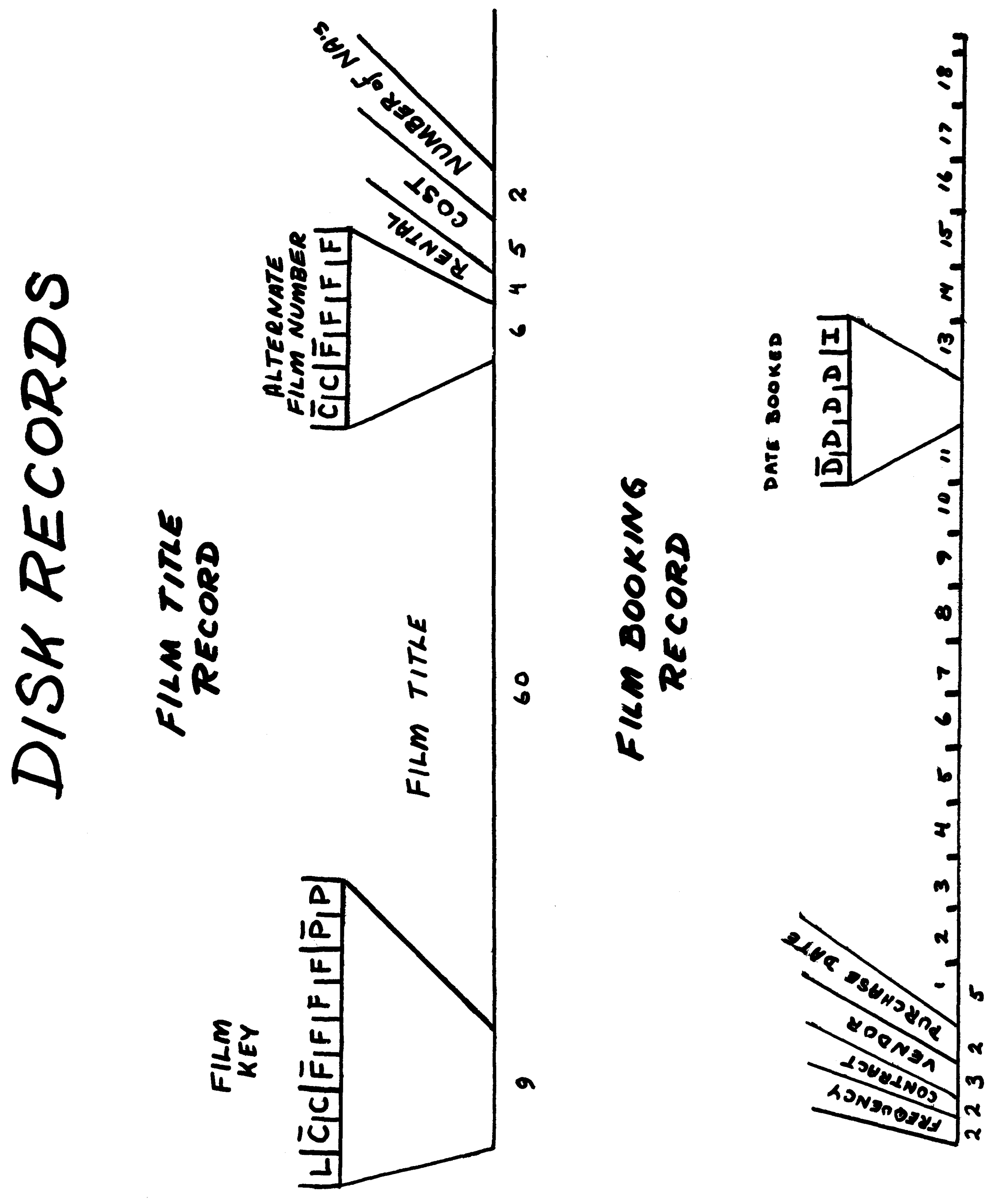
number we carry internal to our record but do not put on our key.

If we look then at Figure 5, we have the two formats of our film records. The first is the general information on a given film, information similar to that which we would expect to find on any inventory system. We have the film key which we just mentioned above, the film title which corresponds to the item description, a rental amount which would correspond to a selling price, the cost of the film to us which would correspond to a users purchase price and the number of times this particular item was requested by a user plus the number of times we were unable to fill a customer's request because the film was not available at that time. In addition we have the alternate film number.

The alternate film number is a number that is used if the customer indicates that the time period that he has requested is very important to him but that the particular film is not important and that a film covering comparable material would be satisfactory to him if we can make a suggestion. This alternate film is automatically booked for him on the given dates if he indicates that we can make this substitution for him.

The second record is also an outgrowth of our special conditions - added problems to the normal inventory system that we had to cope with. This is the record that we keep on each individual print, each individual item within an item type. This gives us summary information, frequency of use, the contract

Figure 5



DISK RECORDS
FILM TITLE RECORD
FILM KEY
L C̄ C F̄ F F F P̄ P
FILM TITLE
ALTERNATE FILM NUMBER
C̄ C F̄ F F F F
RENTAL
COST
NUMBER of NR.'s
9
60
6 4 5 2
FILM BOOKING RECORD
DATE BOOKED
D̄ D D D I
FREQUENCY
CONTRACT
VENDOR
PURCHASE DATE
2 2 3 2 5
1 2 3 4 5 6 7 8 9 10 11 13 14 15 16 17 18

under which we obtain this particular print, may it be through a government grant, under some sort of leasing condition, or some sort of share-the-rent basis. We indicate the vendor from which it was purchased and follow this by the purchase date (which DAVY has put into their record number as was mentioned earlier). The rest of the record is devoted to the particular dates on which this item will be out of the inventory. We say "will be" very correctly because as this particular item is checked for requests, we also check for bookings of that film which have been sent out and returned. These bookings are cleared from our records at the time they become obsolete. Date coding is as follows:

The date is calculated relative to April 1, 1966 and translated to a four digit number, i.e. April 1, 1966 = $\bar{0}$001. This shipping date is stored in the four digit disk code and the return date plus one (extra day in shop for checking) is expressed in a one digit increment to the shipping date. This code is capable of a twenty-one day block out using five digits. One day indicated by the four digit date plus a maximum of a 20 day increment. Increments are expressed as follows:

Code	# Days Increment From Shipping Date
0	1
1	2
2	3
.	.
.	.
9	10
$\bar{0}$	11
$\bar{1}$	12
.	.
.	.
$\bar{9}$	20

Any date earlier than April 1, 1966 is translated and printed as blank.

Examples:

April 1, 1966 to April 6, 1966 = $\bar{0}0015$

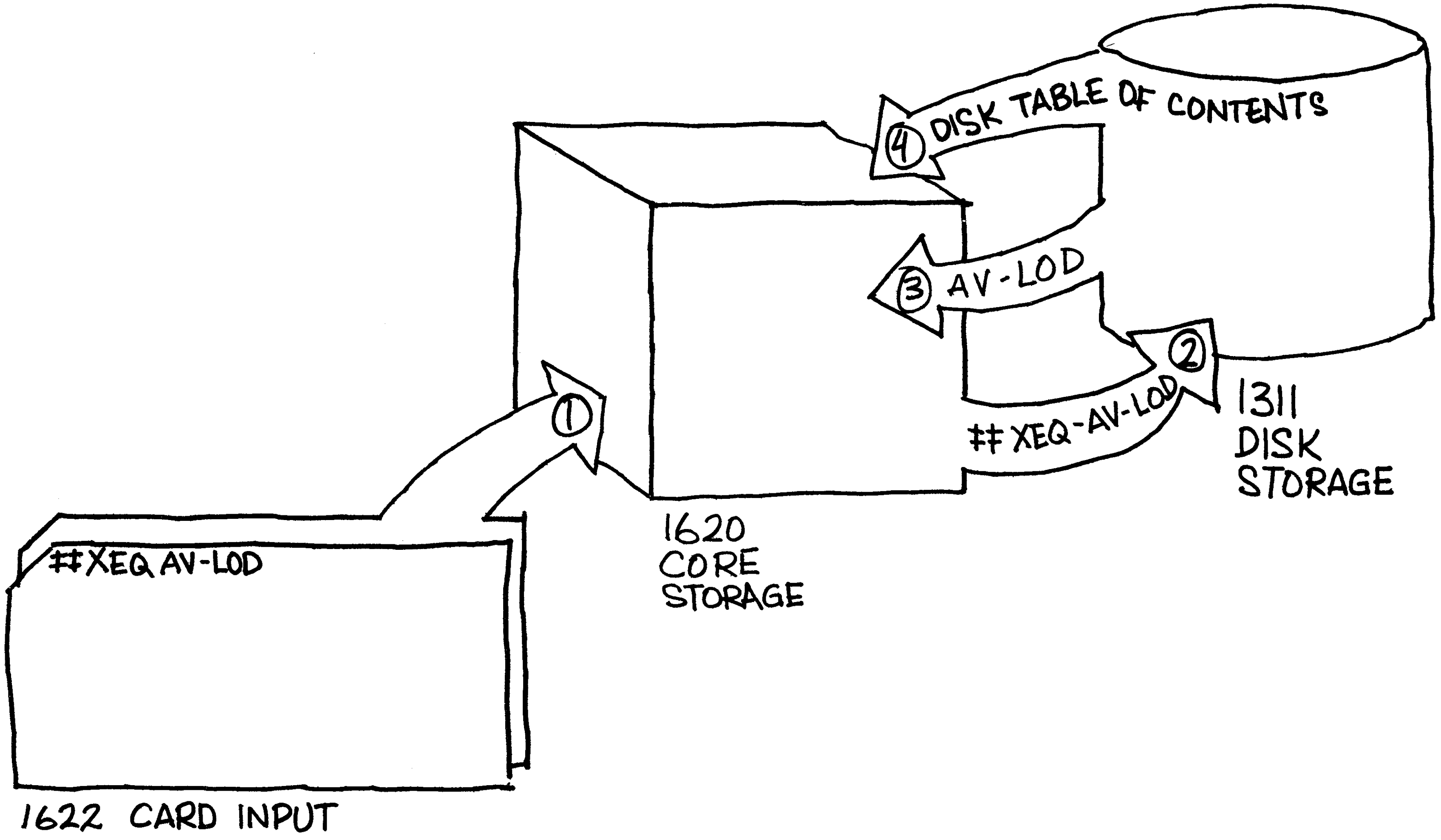
April 9, 1966 to April 28, 1966 = $\bar{0}009\bar{9}$

AV-LOD is the program that creates, maintains, updates, and restructures (when necessary) the above mentioned files. Figure 6 shows the normal invoking of a disk resident program in addition to the extra step included in all System/240 programs; retrieving the DTC.

Figure 6

NIU SYSTEM / 240 DAILY PROGRAM

INVOKING THE SYSTEM



SERVICING REQUESTS

Looking back now to the four parts of our problem, we have run through the first one, creating our new files. Now we want to move on to servicing our requests. Let us look first at the types of input or types of requests that we can get. In the final form the input to the computer is in card form. How do we get it in card form? There are three ways, with a fourth under consideration. First we may have a phone call from our on-campus users or off-campus users for an immediate request delivery. The pertinent information on that particular request is coped down by a clerk and then edited and submitted to keypunching. The second form is that of a request received through the mail which is edited and sent again to keypunching. The last of the existing types is received in card form. Let's take a look at exactly how we accomplish this, how we relieve ourselves of not only keypunching but also the clerical editing and put the responsibility for the validity of the information on the user. In order to do this we supply the user with two bits of information, one is the KWIC index, a sample portion of which can be found in Figure 7. The other is a master card file, with a card per film, or partially punched request cards.

Here we have the titles of our films alphabetically listed by each significant word in the title. In other inventories

Figure 7

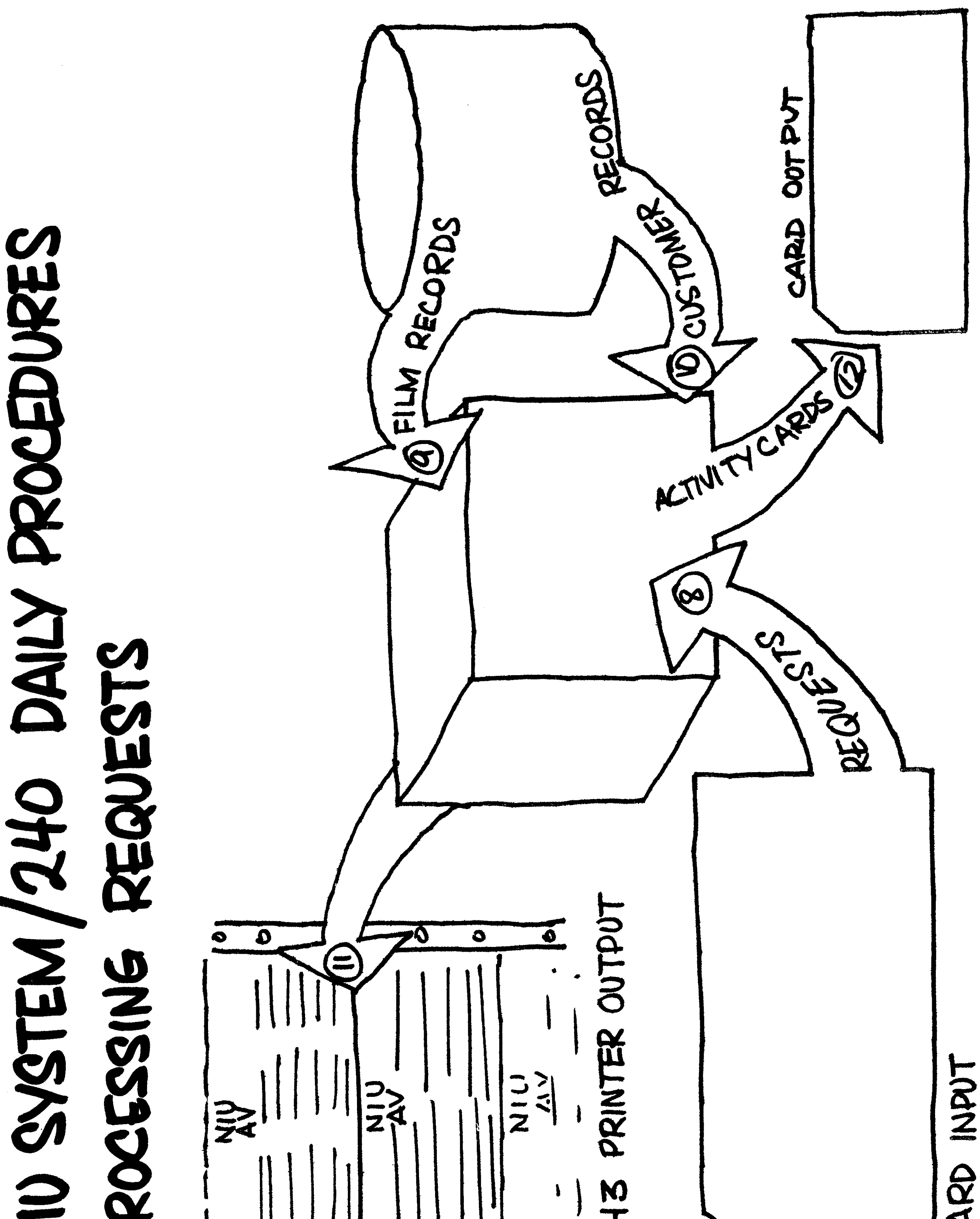
KWIC INDEX

```
   LEGEND OF JOHNNY | APPLESEED 00650*                           512007701
   HOME ELECTRICAL  | APPLIANCES 00190*                          217001301
   GAS LAWS THEIR   | APPLICATIONS 0200230*                      406025101
   GRANT LEE AT     | APPOMATTOX 0400395*                        603025401
BAL YOUR DIET HLTH  | APPRNCE 0100190*                           208006801
       MECHANICAL   | APTITUDES 0100190*                         205023401


                    | GARDEN PLANTS HOW. THEY GROW 0100190*      214016901
    DEMONSTRATING   | GAS LAWS 00285*                            406009401
      HEAT ENERGY   | GAS LAWS 00575*                            606037701
                    | GAS LAWS THEIR APPLICATIONS 0200230*       406025101
          LAWS OF   | GASES 0100190*                             206014701
                    | GASOLINE AGE 0200485*                      503042101


  HEAT ENERGY GAS   | LAWS 00575*                                606037701
              GAS   | LAWS THEIR APPLICATIONS 0200230*           406025101
     PRESIDENTIAL   | LEADERSHIP 0400425*                        603030101
         STYLE OF   | LEADERSHIP 00450*                          605014901
       DEVELOPING   | LEADERSHIP 0100190*                        205021301
```

this will be a description of the item. A user can then, assuming half-way reasonable descriptions of your items, look up a given idea and find several films on the idea or subject. In the example in Figure 7, if he were looking for something on gas laws or applications of gas laws, he could look under APPLICATIONS and find a film referenced as 406025101. The user could look under GAS and find "Gas Laws and Their Applications" also referenced as 406025101. Under LAWS he would find the same film with the same number. Using this reference number and going to the card reference file which we supply the user, the customer need only duplicate this master card and punch the dates required into the reproduced card. If he wants his name to appear on the address he can also include at the end of the card in the "attention to" columns the particular name that he wishes on the address. (The card is identified by a 5 in card column 1 and the format can be found in Appendix A with the others in the system.) This card is then sent directly to us and becomes input to our system.

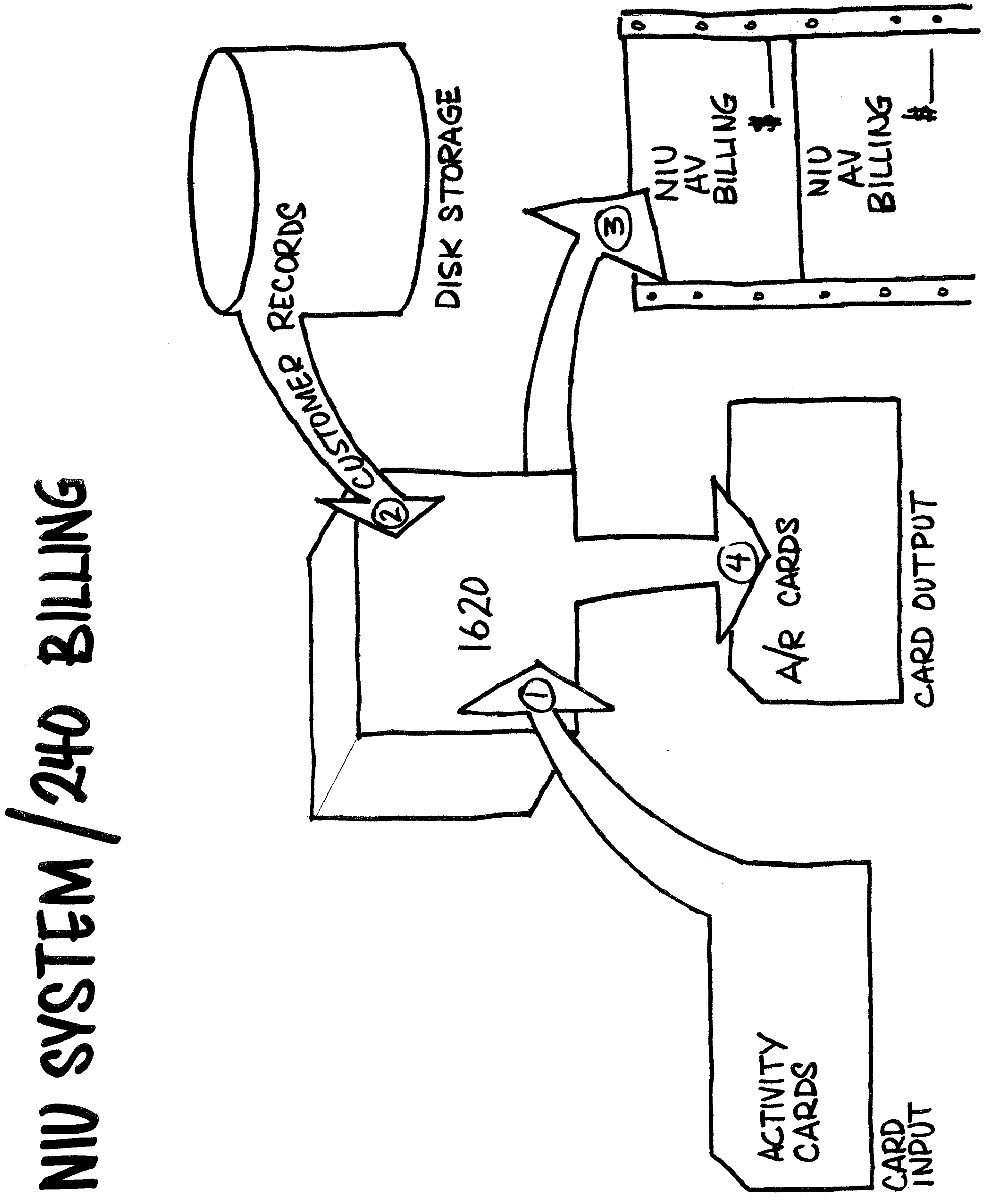
Yet under consideration is a mark sense form to be distributed to faculty and customers. The dates and film number would be coded on this and processed by our 1230. The cards output from this would be in request card form and processable by the machine. These then are our inputs. We have covered the method of invoking the system in which we pull the program off the disk and begin reading our request cards. Figure 8 shows the program steps involved in processing a request. The

Figure 8


NIU SYSTEM/240 DAILY PROCEDURES
PROCESSING REQUESTS
9 FILM RECORDS
10 CUSTOMER RECORDS
CARD OUTPUT
ACTIVITY CARDS 12
8 REQUESTS
11
NIU AV
NIU AV
NIU AV
1443 PRINTER OUTPUT
CARD INPUT

program upon reading a request pulls the required film from the disk file, the required customer from the disk file, and checks to see if the film is available for the dates requested, if so, it begins printing out a confirmation slip, if not, it prints out the confirmation slip indicating that the film is not available for that date but if it is we also punch out an activity card which is used in our internal system. (The activity card form also appears in Appendix A). The form of the confirmation is in the Appendix B and this is a multi-part form. The upper part seen in the Appendix is the top sheet, the lower part on the Appendix representation of this form is the bottom sheet of the four part form. The top sheet is sent out directly to the customer immediately after processing his request to indicate the status of his request and the lower right hand corner of the fourth carbon becomes a shipping label for the item. This portion is torned off, the back of it is already gummed and this is stuck directly on the film mailing case. In the case of customers who supply us with card input to the system, we also supply them card form output that goes into their accounting systems and provides them with some statistics on our performance for them. Our billing operation is then performed on the basis of these activity cards that we have produced during the successful booking of a film for a customer. When the film comes back, the packing slip which was a part of the confirmation is matched against one of these activity cards. When a successful match has been made, i.e. the film has been received back from the customer, these cards provide input to our billing system, the activity chart of which appears in Figure 9. As these activity cards are read, the customer information on the disk providing us an address to bill to is pulled off the disk. Pertinent film descriptions that apply to the customer's bill are also pulled and this detailed information is printed on his bill. We then print the bill and also punch a card which becomes input to our University accounts receivable system. (The billing form is included in Appendix C.) This then covers our second point of servicing the requests to our customers. We now move to the third point of the problem and the third point of our solution - maintaining our inventory.

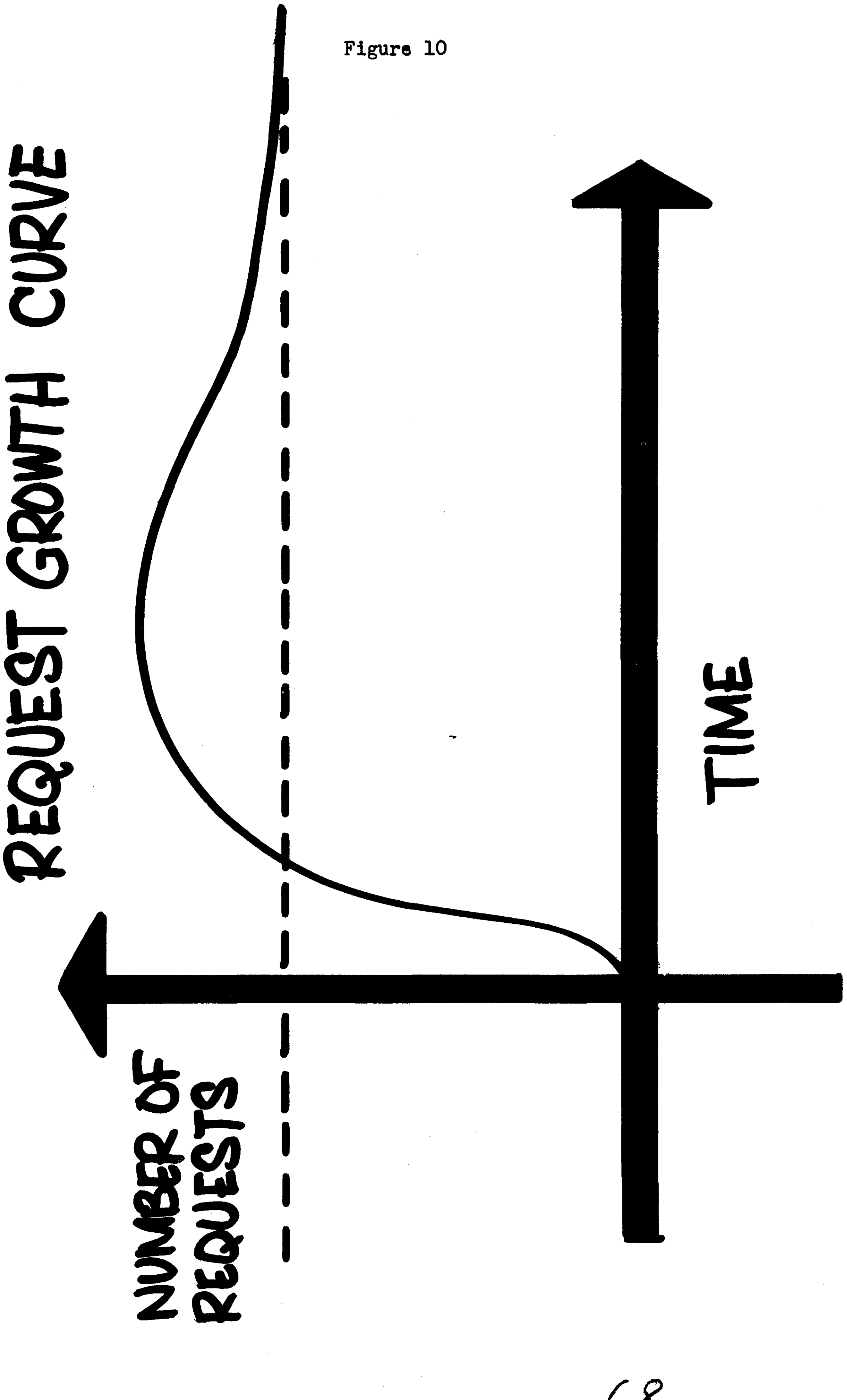
Figure 9


NIU SYSTEM/240 BILLING
CUSTOMER RECORDS
DISK STORAGE
1620
NIU AV BILLING $
NIU AV BILLING $
A/R CARDS
CARD OUTPUT
ACTIVITY CARDS
CARD INPUT
1
2
3
4

MAINTAINING THE INVENTORY

If we look at Figure 10, which is labeled REQUEST GROWTH CURVE, we are trying to develop here a buyer's indicator by keeping records on particular films. This curve describes the demand on a new item from the time of the user's awareness of its availability to some time out beyond that. As the demand for this film grows; as we go up on the axis labeled "number of requests", we can also mark it off in intervals indicating number of prints of that film, or number of that item which we have to keep in our inventory in order to satisfy these requests. When we reach the doted line, we have a choice to make, whether to purchase, lease, or by some other means obtain the items in order to catch the peak of this growth and get these requests, or at this time to stop our purchasing and to realize that we are going to have to take some "not availables" or find that we are unable to satisfy the customer's request, but in so doing not have obtained films that will not be in demand beyond time "y". It is our job to come up with an equation for this curve. At the current time based on our recrods, we can find a breaking point - point B - where the curve starts to come back down or it starts to change direction somewhat. When this happens we can cut off or we can lease items

Figure 10



for a short amount of time in order to fill in this area and pick up these extra requests. But our job is simply to supply this buying information to management so that intelligent decisions can be made on growth and trends within the inventory. This is the heart of the successful inventory and the conclusion of the third portion of our solution.

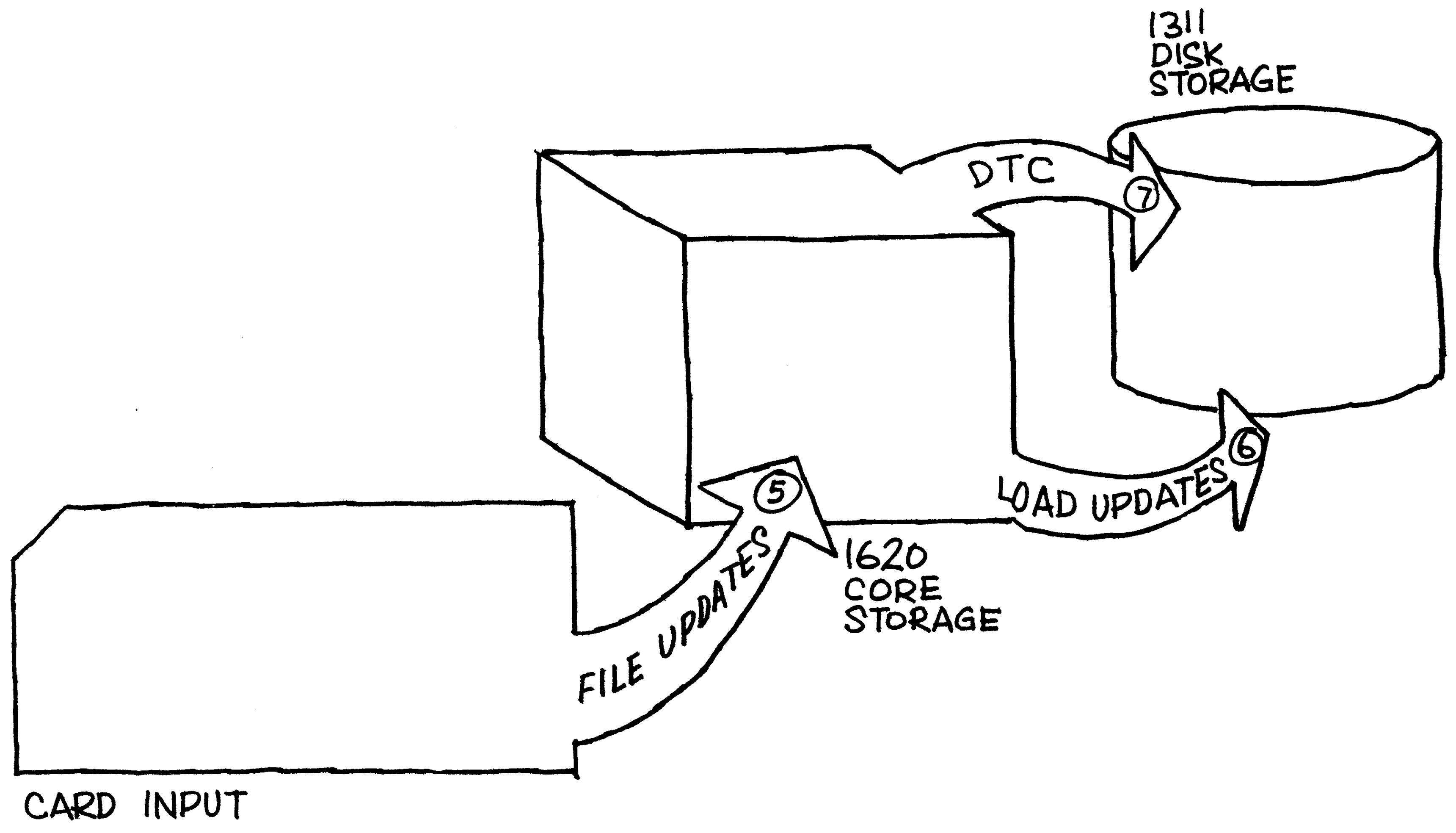
MAINTAINING RECORDS

Having covered the creation of new files, the servicing of requests and the maintaining of our inventory, let's go now to the maintaining of our records. We find that in the maintaining of our records we are going back to the same system that provided us with the ability to create these records. We have if you refer back to Figure I, the disk layout and Figure 2 the Disk Table of Contents provided for our user department a system that very closely parallels IBM's index sequential on the 360. We have come up with an index sequential type sytem for a 1620 disk system. We have quite the same capabilities here - the system can create files, add to the files, delete from the files, process the files randomly or sequentially. All of these abilities are itemized in Appendix D.

The updating is done on a daily basis, in other words, the AV-LOD program which provides both the creation and updating ability is run daily just before the AV-DLY program which services the requests. Figure 11 shows the operation of AV-LOD in the update mode, but it should be noted that in our system, creation is a special case of updating. At this point we must look closer at the back-up system that we mentioned earlier.

Figure 11

NIU SYSTEM/240 DAILY PROCEDURES
UPDATING THE SYSTEM



If we have to duplicate our files we have to go through three steps. Number one is to load the monitor system on the disk, number two we have to load our basic files which amount to our customers two part file, billing and shipping, and our film records for each category. The third step is then to reload all the dates booked on the films we have just loaded. The loading of the customers and the films is done again by AV-LOD, but the loading of the dates booked for a given film is done by a unique program named QIKLOD. This utilizes our activity cards which are kept up to date on the basis of our billing, i.e. for any films that have been returned and are no longer out of the library the transaction or activity card is pulled from our card file and stored elsewhere. So when loading up the periods booked for a given film we have only current bookings. This particular program allows us to load somewhere in the neighborhood of 5,000 bookings in an hour. The daily booking program that provides us with the confirmations allows us to book something like a thousand in an hour. We usually spend anywhere from 3 or 4 minutes to 15 minutes on the 1620. Obviously in time periods this small the most significant time factor is not computer time, but set-up time and general operator intervention.

Having looked at the four parts of the problem and in turn the four parts of our solution, which involves creation of new files with program AV-LOD, servicing of requests using AV-DLY, maintaining of an inventory by producing a buyers indictor for management and maintaining our records through the use of QIKLOD and AV-LOD and having talked a little about the pleasantly surprising performance we have gotten from the system, lets look at future projections for the system and areas of growth.

FUTURE PROJECTIONS

Referring to Northern's enrollment growth curve in the Introduction to our discussion, we can see the installation of our Model 40 360 in August of 1967. With the installation of the machine under our belt, we are well on our way to converting System/240 to 360 COBOL. As we mentioned earlier, a large part of the development of System/240 was devoted to producing our own Index Sequential system for the 1620. This having been done for us on the 360, the strict conversion is going rather quickly. Of course in a 360 we have additional hardware capabilities not found in the 1620. Two of these are the teleprocessing and multiprocessing capacities.

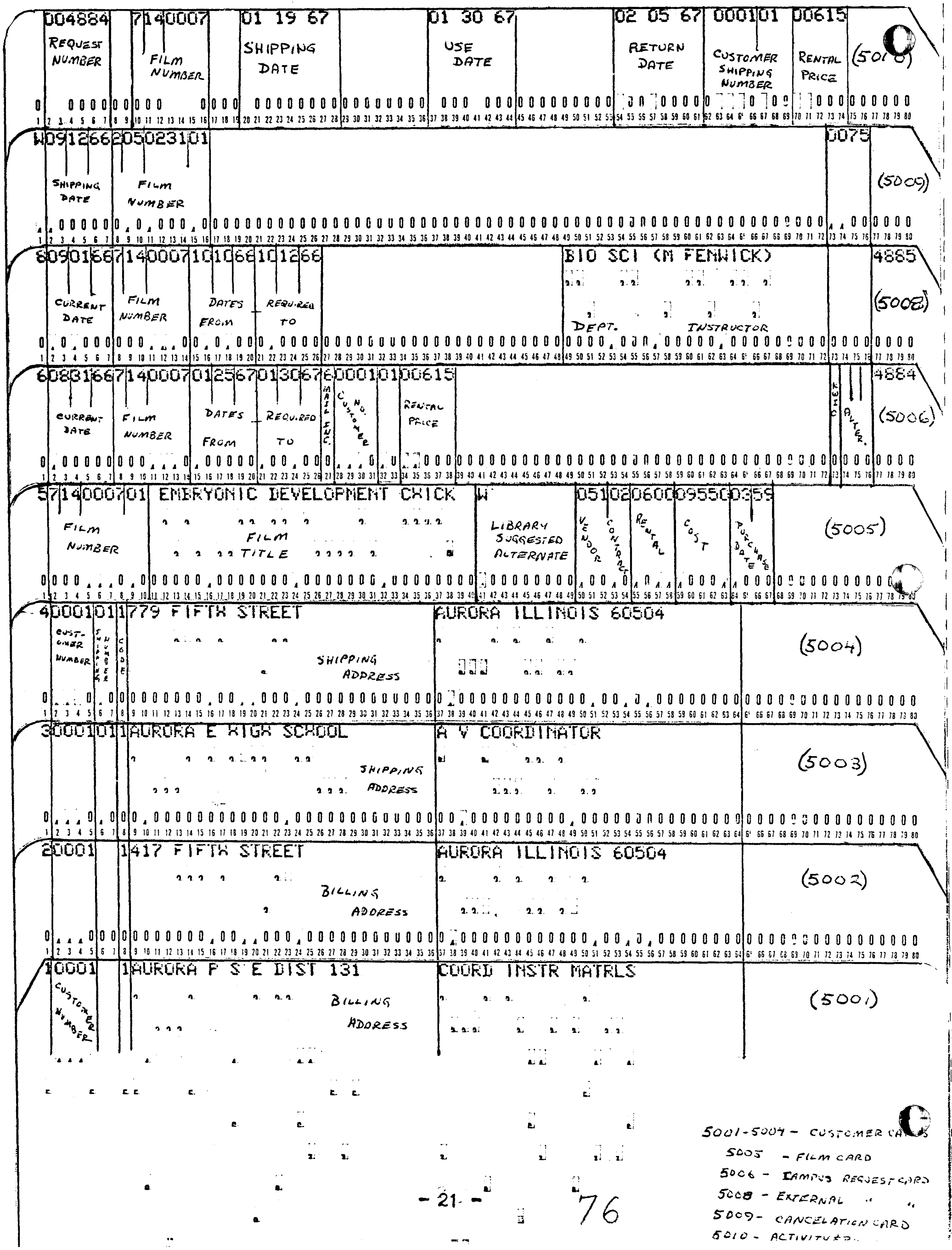
Using teleprocessing we can do several things, two of which are; (1) give "immediate" response on film requests and (2) develop a cirriculum-building-tool in the University. In the first case, not only can an immediate "yes" or "no" response be given on a request, but if the response is "no" a calander showing remaining available dates can be displayed on the CRT. In the second case we can expand upon our existing quick index and on request, display available material pertinent to the subject matter to be covered plus indicating proper sequences.

The multiprocessing capability gives us the technology to be able to tie all the state libraries together so that a request given to any of the libraries would exhaust the states potential, i.e. over the phone, in a matter of seconds any individual could have the entire states film resources at his command.

We have now a data retrieval system that is updated daily. With the new hardware to fill the gap we will have a real-time system.

We feel at this point that the ability of the system to grow with increasing demands and greater the capabilities of the hardware is limited only by the extent to which we wish to persue it.

APPENDIX A



004884 7140007 01 19 67 01 30 67 02 05 67 000101 00615
REQUEST NUMBER | FILM NUMBER | SHIPPING DATE | USE DATE | RETURN DATE | CUSTOMER SHIPPING NUMBER | RENTAL PRICE
(5010)

W0912662050231O1 0075
SHIPPING DATE | FILM NUMBER
(5009)

80901667140007101066101266 BIO SCI (M FENWICK) 4885
CURRENT DATE | FILM NUMBER | DATES FROM | REQUIRED TO | DEPT. | INSTRUCTOR
(5008)

6083166714000701256701306760001010O0615 4884
CURRENT DATE | FILM NUMBER | DATES FROM | REQUIRED TO | MAIL INC. | CUST. NO. | RENTAL PRICE | CHEK | ALTER.
(5006)

5714000701 EMBRYONIC DEVELOPMENT CHICK W 05102060009550O359
FILM NUMBER | FILM TITLE | LIBRARY SUGGESTED ALTERNATE | VENDOR | CONTRACT | RENTAL | COST | PURCHASE DATE
(5005)

400010111779 FIFTH STREET AURORA ILLINOIS 60504
CUSTOMER NUMBER | SHIPPING | CODE | SHIPPING ADDRESS
(5004)

3000101 1AURORA E HIGH SCHOOL A V COORDINATOR
SHIPPING ADDRESS
(5003)

20001 1417 FIFTH STREET AURORA ILLINOIS 60504
BILLING ADDRESS
(5002)

10001 1AURORA P S E DIST 131 COORD INSTR MATRLS
CUSTOMER NUMBER | BILLING ADDRESS
(5001)

5001-5004 - CUSTOMER CARDS
5005 - FILM CARD
5006 - CAMPUS REQUEST CARD
5008 - EXTERNAL " "
5009 - CANCELATION CARD
5010 - ACTIVITY ...

APPENDIX B

CONFIRMATION COPY

NORTHERN ILLINOIS UNIVERSITY
EDUCATIONAL FILM LIBRARY
DE KALB, ILLINOIS 60115

IMPORTANT

REQUEST NUMBER	SHIPPING DATE	SHOW DATE	DUE BACK DATE
FILM NUMBER	FILM TITLE		** RENTAL AND SERVICE CHARGE $
CUSTOMER NUMBER	CUSTOMER ORDER NO.		

BILL TO

SHIP TO

* PLEASE NOTE ABOVE FILM IS A SUBSTITUTE BECAUSE FILM ORDERED IS NOT AVAILABLE. ** 15¢ INSURANCE CHARGE ADDED FOR EACH FILM.

NORTHERN ILLINOIS UNIVERSITY
EDUCATIONAL FILM LIBRARY
DE KALB, ILLINOIS 60115

IMPORTANT

REQUEST NUMBER	SHIPPING DATE	SHOW DATE	DUE BACK DATE
FILM NUMBER	FILM TITLE		
CUSTOMER NUMBER	CUSTOMER ORDER NO.		

FROM: AUDIO VISUAL CENTER
NORTHERN ILLINOIS UNIVERSITY
DE KALB, ILLINOIS 60115

BILL TO

TO:

PACKING SLIP - NOT AN INVOICE

RETURN REQUESTED LIBRARY MATERIALS
16 MM MOTION PICTURE - NON FLAMMABLE

APPENDIX C

C-4458

NORTHERN ILLINOIS UNIVERSITY
EDUCATIONAL FILM LIBRARY
DEPARTMENT OF INSTRUCTIONAL MATERIALS
DE KALB, ILLINOIS 60115

CUSTOMER NO.	INVOICE NO.	INVOICE DATE	PAGE NO.

** 15¢ INSURANCE CHARGE HAS BEEN ADDED FOR EACH FILM. ▼

REQUEST NUMBER	SHOW DATE	FILM NUMBER	FILM TITLE	CUSTOMER ORDER NUMBER	** RENTAL AND SERVICE CHARGE

PAY LAST AMOUNT IN THIS COLUMN

INSTRUCTIONAL MATERIAL
TRUST NO. 52428

SIGNATURE OF INDIVIDUAL PREPARING INVOICE/STATEMENT

MAKE CHECKS PAYABLE TO: NORTHERN ILLINOIS UNIVERSITY

MAIL TO: NORTHERN ILLINOIS UNIVERSITY. BURSAR'S OFFICE, DE KALB, ILLINOIS 60115

APPENDIX D

AV-LOD
MANIPULATIONS

	FILMS		ADDRESSING	
	TITLES	PRINTS	BILLING	SHIPPING
LOADING	1	1	1	1
AUGMENTING	2	3	4	5
REPLACEMENT	-6-		-7-	

BASIC REQUIREMENTS AND END RESULTS	TYPEWRITER RESPONSE
1. INFORMATION LOADED AS IT APPEARS IN THE FILM AND CUSTOMER FILE.	'GAP' CC NNNN FFFF LEAVING A GAP IN FILM NUMBERS C - CATAGORY N - NEXT OPEN NUMBER F - NEXT FILM NUMBER PLUS APPROPRIATE ERROR MESSAGES 'ER' XX XX - IDENTIFICATION NUMBER OF ERROR
2. OPERATION FILLS GAPS IN TITLE LIST OUTPUTED ON TYPEWRITER FROM STEP ONE AS 'GAP' --INPUT FILM CARD WITH NUMBER OF PRINTS TO BE LOADED IN CC 9-10.	'AUGT' NNNNNNNPP LL FILLING TITLE NUMBER GAPS N - FILM NUMBER TO BE INSERTED P - PRINT NUMBER FOR SAID FILM L - NUMBER OF PRINTS STORED TO DATE FOR SAID FILM
3. OPERATION ADDS PRINTS. INPUT FILM CARD WITH 9-10 BLANK AND THE NUMBER OF PRINTS TO BE ADDED APPEARING IN CC 19-80.	'AUGP' NNNNNNNPP LL INSERTING PRINT
4. REPLACING POSITIONS HELD BY DUMMY NUMBERS. AMOUNTS TO RE-PLACEMENT (7)	
5. OPERATION ADDS SHIPPING ADDRESSES FOR BILLING ADDRESS-ALREADY HAVING ONE OR MORE SHIP-PING ADDRESS.	'INSS' NNNNSS (SHIPPING ADDRESS) N - CUSTOMER NUMBER TO BE INSERTED S - CORRESPONDING SHIPPING NUMBER
6. OPERATION REPLACES FILM W/(0 IN CC 80), OR W/O(BLANK IN CC 80) CLEARING OF BOOKINGS ON ALL PRINTS. REPLACES WITH SAME NUMBER OF PRINTS. INPUT FILM CARD W/ CC 9-10 BLANK.	'FILM REPLACEMENT' NNNNNNN- N - FILM NUMBER WITH STORED BOOKINGS TO BE REPLACED
7. OPERATION REPLACES AND INITIALIZES BILLING AND SHIP-PING ADDRESS INDICATED.	'CUSTOMER REPLACEMENT' (BILLING ADDRESS)

SESSION NUMBER T.1.1

SPEAKERS

MRS. JOYCE FODER, ENGINEERING COMPUTING LABORATORY ON DISK
    DATA STORAGE ROUTINE DDSR

DISK DATA STORAGE ROUTINE DDSR

Mrs. Joyce Fodor
Engineering Computing Laboratory
B554 Engineering Building
1415 W. Johnson Street
Madison, Wisconsin 53706

Description/Purpose

DDSR is a program written in SPS II-D to facilitate permanent disk storage of users' data from FORTRAN or SPS programs. The blocks of data are given a name and table entries that are completely compatible with the MONITOR I system. The user need not know what sectors on the disk are available.

Machine Configuration Required

1. 1620 Model I or II
2. 20 K memory
3. Card I/O
4. Indirect Addressing
5. 1 Disk drive

General Program Description

Since rapid processing of the MONITOR system tables required approximately 10000 core positions, and it was undesirable to take this much core from the user's program, the system uses the area from 02402 to 12000. In order to do this, the routine must first store the contents of these locations on disk so that it can restore them before returning to the main program. For this reason the short form of the subroutines cannot be used when using these subroutines. Both the disk write and the data recovery subroutines have routines which store these core locations in cylinder fifteen of the work cylinders before calling link to the main read and write routines. The read and write routines then search the tables to find the particular entry called or the required storage, and process the data.

All data blocks are given both DIM entries and Equivalence table entries. When the routine is used to write data on disk permanently, the routine also makes an entry in the sequential table.

Data blocks may be read from or written into disk in the normal course of any FORTRAN program, and may be used as often as the user desires, but the maximum amount of core that may be stored at one time is 20000 digits. This corresponds roughly to a singly dimensioned matrix of dimension 2000. If larger blocks of storage are required the arrays may be broken into parts and stored that way.

Warnings

This program might not work with the short form of the FORTRAN subroutine because core positions 2402 to 12000 are stored on disk and these core locations are used as work area. If the call to these subroutines, and INDATA or OUTDATA are located above this address the system should work.

All entries in an array should be defined prior to storing them on disk. If the array is doubly subscripted this is very important. If it is a singly subscripted array and all the elements from one to the desired element is defined this is adequate.

FORTRAN CALLING PROCEDURES

To Store Data

CALL INDATA (IND, LF, LK, NR, ARRAY)

IND =	1	if data to be stored is a single fixed point variable or a a fixed point array.
	3	if data to be stored is a single floating point variable or a floating point array.
LF		Is the floating point mantissa length for the program being used.
LK		Is the fixed point word length for the program being used.
NR		Is the number of elements to be stored, the dimension of the array if it is single subscripted, or M * N for a doubly subscripted array.
A		Is the name of the array or number, fixed or floating point to be stored.

To Call a Data Block

CALL OUTDAT (IND, LF, LK, NR, ARRAY)

where parameters are the same as above.

Naming the Data Block

Whenever either of the routines is called, one data card will be read from the card reader. This card should have name of the block data to be read or written left justified in card columns 7 to 12. Care must be taken to be sure that this card appears in the proper location in the data deck or error will result and the job will be terminated.

DISK DATA STORAGE ROUTINE DDSR (1620 - 01.1.036) can be ordered from the 1620 program library through your local representative or:

International Business Machines Corporation
DP Program Information Department
40 Saw Mill River Road
Hawthorne, N. Y. 10532

SESSION NUMBER T.1.2

SPEAKERS

G. ROEMER OF IBM SPOKE ON THE CSMP FOR THE 1130. THE TOPICS INCLUDED WERE THE DIFFERENCE BETWEEN DIGITAL AND ANALOG COMPUTERS, BLOCK PROGRAMMING AN ANALOG COMPUTER, USAGE OF THE CSMP PACKAGE, AND EXAMPLES.

P. WOODROW SPOKE ON BUFFERED AND OVERLAPPED I/O. THIS PACKAGE WILL BE RELEASED TO COMMON SHORTLY.

MEETING WAS CHAIRED BY LARRY WHELEN.

OVERLAPPED I/O FOR 1130 FORTRAN PROGRAMS

Peter J. Woodrow

Aeronautical Research Associates of Princeton, Inc.
Princeton, New Jersey

For most types of commercial computer applications, computation is at a minimum and input/output operations are numerous. Thus it is desirous, even at the cost of increasing core requirements somewhat, to use I/O routines that operate as efficiently timewise as is possible. Unfortunately, the 1130 FORTRAN I/O routines were designed to require minimum core. As a result, all FORTRAN I/O routines (e.g. PRNTZ, CARDZ) use a common buffer that is filled by SFIO. While presumably, on output at least, the I/O routines could return immediately with SFIO waiting for completion before starting a new line, this is not the case at present. All I/O routines wait internally for the completion of the requested operation. This means generally that I/O speeds from FORTRAN programs are almost halved.

In order to speed up our commercial data processing, we decided to write some FORTRAN-callable routines, primarily for use with COMET (since they all use A2 format in effect). These routines were each written for a specific purpose but are presumably general enough for different applications. We have never tried using these routines with other FORTRAN I/O, primarily because we did not wish to have SFIO loaded (approximately 800 words), but they should work above modification level No. 5- if the interrupt levels used by FORTRAN I/O are completely different from those used by these routines. In particular, the card routine cannot be used by a FORTRAN program that does FORTRAN I/O on paper tape or the keyboard/printer.

One point to note is that none of the routines is very complex since all usë standard assembler language I/O subroutines. There would almost certainly be no great difficulty encountered in writing a special version of PRNTZ to replace the supplied version and to run at a speed twice as fast as the current FORTRAN version. However, some core would have to be sacrificed. Replacing CARDZ with a faster version is somewhat more difficult for reasons explained below in the brief write-ups that follow.

SPCDR

This routine has three entry points; SPCDR, SRERD, and SPCDP. The primary reason for writing this routine was not so much to increase card reading speed as it was to perform some special error checking and code conversion. If it is necessary for computation to be done by the FORTRAN program in order to determine whether to stacker select, this high card reading speed is impossible. In addition, in most commercial applications, each card is printed and hence(for the 1130) print speed is the overriding consideration. For our application it was highly undesirable for the program to accept characters that would not print on the 1132 printer (as does the FORTRAN card read routine). In addition, since we have an 029 keypunch with the left-zero-insertion feature, we felt it would be mandatory to accept -Ø (11-Ø punch) which CARDZ translates into a blank. Following are the various calls that one might use.

CALL SPCDR(NC, IFV, IFTV, IAREA, NERRC)

where

NC	-	must contain the number of columns that are to be read
IFV	-	a vector containing field end points. The routine performs minor checking for specified fields on the card. If, for example, the fields were 1 - 9, 1Ø - 19, ... etc., then IFV(1) = 9, IFV(2) = 19, etc. N.B. The last IFV element must be greater than or equal to NC.
IFTV	-	a vector containing on return to the calling program the type of field corresponding to IFV. The following codes are possible 1 = field contains only blanks 2 = field contains only numeric information (no blanks)

3 = field contains blanks and numeric information
4 = field contains only alphanumeric characters
5 = field contains blanks and alphanumeric characters
6 = field contains alphanumeric and numeric, but no blanks
7 = field contains a mixture of all three; alphanumeric, numeric, and blanks

N.B. A (11-numeric) punch in the last column of a field is considered numeric (i.e. indicates the field is to be considered negative).

IAREA - a one-word integer vector dimensioned to contain the input characters. IAREA must be dimensioned at least NC/2 words (NC even) or (NC + 1)/2 words (NC odd). The characters are packed two per word as in the conventional A2 format.

NERRC - returned as a zero if no errors were sensed; returned as a 1 if any character read is not printable on the 1132 printer (the only exception being -Ø(11 - Ø punch)). Characters in error are replaced by blanks.

The above routine converts characters and checks field types as the card is being read. After the card has been read, the routine checks the error indicator provided by CARD1 (indicating a 1442 sensed error) and retries the read and conversion if there was an error.

In order to more fully understand the operation of the following two calls, one must be aware of the fact that SPCDR contains an internal buffer into which the card image is read. This is not destroyed by conversion and is thus available for the following two calls.

CALL SRERD(NC, IFV, IFTV, IAREA, NERRC)

This call is provided so that one may reread the same card with different specified fields and thus first sense the type of card to be processed and then actually check the appropriate fields. No card reading takes place on this CALL. The explanation of the parameters is the same as that for SPCDR above.

CALL SPCDP(NC, OAREA, NERRC)

where

NC	-	contains the number of columns to be punched
OAREA	-	is a vector containing the characters to be punched (see IAREA above)
NERRC	-	returns the error code
		0 = no error
		1 = at least one character to be punched not printable on 1132 printer (except -Ø)
		2 = an already punched column (read on call to SPCDR) is to be punched with a different, non-blank character; i.e.,overpunching is not allowed

If an error is sensed, the card will not be punched and an immediate return to the calling program is instead executed.

Note that if SRERD is called after a call to SPCDP, it will in effect be reading the "union" of the original card that was read and the card to be punched, i.e. the final version of the card as it would appear if it were read again after punching.

Note also that for proper operation all cards must be read by SPCDR prior to being punched by SPCDP; otherwise, spurious errors are likely to be generated.

None of the above routines is particularly speedy. Their main advantage is the thorough error checking that is done by them rather than by the FORTRAN program. If maximum speed is desired with no automatic error checking, then SFCDR which follows should be used instead.

SFCDR

This routine has four entry points; SFCDR, SFCDS, SFCDF, and SFCDP. It was designed for a quite different environment. Here we wished to read a considerable number of cards, operate rather extensively on each card and maintain the results in core. As a result, speed was the primary objective with no desire to stacker select the cards. As a result, the routine was designed with two buffers that alternate. Thus SFCDR immediately starts reading the next card (unless otherwise instructed) and then proceeds to convert the present, desired card. As a result of this process conversion "on the fly" is not necessary and CARDØ is used. However, the SPEED conversion routine is used and thus all EBCDIC characters are converted properly and at maximum speed. The FORTRAN CALL's are as follows:

CALL SFCDR(NW, IAREA, NC)

where

NW	-	is the number of words in IAREA to be filled (i.e. 1/2 the number of characters). The number of characters must thus be even.
IAREA	-	a one-word integer vector that is to contain the converted card characters packed two per word as in conventional A2 format. IAREA must be dimensioned at least NW words long.
NC	-	a code used to control the reading of the next card while this one is being processed. If NC is odd, no new card read is started (see SFCDS below). If NC is even, then the next card read is started unless the first column of the present card happens to contain the card image code equivalent of NC. Thus, if the last card of one's deck always had a 7 - 9 punch in column one, then presumably one would not want to start reading any card following a card with a 7 - 9 punch in column 1. The Subroutine Manual

gives the Hex IBM Card Code equivalent of a 7 - 9 punch as ØØ5Ø which translates to a decimal integer of 8Ø. Hence, NC should equal 8Ø for this example. If you wish to start reading the next card in all circumstances, then NC should be set to 2.

CALL SFCDS

This call is to allow the user's FORTRAN program to make a more extensive test before starting a read operation on the next card. Presumably, NC would be set to 1 in the call to SFCDR, a test quickly made on the card, and then a call to SFCDS if the next card is to be read. Note that SFCDS just starts a read operation; it performs no actual reading and, in fact, has no effect if the next card is already in the process of being read.

CALL SFCDF

This call causes a wait until current card operations are complete and then initiates a card feed (with no read taking place at all) on the next card. Note that if a card read operation were under way at the time, then the next call to SFCDR will convert <u>that card</u> and not the one after the card being fed through.

CALL SFCDP(NW, OAREA)

where

NW	-	contains the number of words to be punched (see SFCDR)
OAREA	-	contains the characters to be punched (see IAREA above)

If a card read was under way at the time of this call, then the read is completed, the punch data moved into the second buffer, and the punch operation initiated. A call to SFCDR at this time will cause proper conversion of the card that was read (and which has already been punched by the call to SFCDP). If, however, another call to SFCDP occurs prior to the call to SFCDR, then the call to SFCDR will cause conversion of the data that was punched into the second card via the second call to SFCDP.

Note that this routine always keeps track of whether the next card read has been started or not; hence, it is unnecessary to call SFCDS at any time unless the obvious resultant speed increase is desired.

SPRNT

This routine contains three entry points; SPRNT, SPRPT, and SPRPC and is used to obtain high speed line-printer operation from a FORTRAN program. Double buffering is not used because it would result in only a very slight increase in speed at a cost of an additional 60 words of storage. The calls are as follows:

CALL SPRNT(NW, OAREA)

where

NW	-	contains the number of words to be printed (the number of characters must be even and is 2 * NW)
OAREA	-	is a one-word integer vector containing the characters to be printed, packed two characters per word as in the conventional A2 format

The print operation is started on this call and a wait for completion occurs either on the next call or on a call to either of the following routines.

CALL SPRPT(NC)

where

NC	-	is returned as a 1 if a page eject occurred after the last line was printed and is returned as a Ø otherwise.

Use of this routine will cause a definite degradation in the printer speed. Generally, maximum speed will be obtained if an internal line counter is used to control page overflow. At any rate, the call to SPRPT should be made immediately prior to the printing of the next line (via a call to SPRNT). Note that the page eject to the top of the next page (channel 1 punch in carriage control tape) has already occurred automatically if NC is returned as one.

CALL SPRPC(NC)

where

NC - contains a code for printer forms control. NC is multiplied by 16 and transferred as is to PRNT1 (see Subroutine Library Manual) for printer control. Thus NC = 16 will cause an immediate skip to channel 1 and NC = 2Ø8 will cause an immediate space of 1

Note that the forms control command is merely initiated by this call; it is completed on another call to any of the above entry points.

FINAL NOTES

1) All of the above sets of routines operate independently of one another (except that SFCDR or SPCDR may be used, but not both by the same program). In general, their operations overlap one another with no problems and they have been used in a number of applications with no difficulty.

2) All sets of routines contain internal buffers into which the characters are moved prior to a call to the proper I/O routine. Hence the various character vectors may be operated on or changed immediately after a call even though the operation may only have been initiated.

3) Since two of the sets of routines may have I/O operations under way concurrent with computation, PAUSE may not work (Modification Level 5 should work, I hope). It it does not, then use of IOND (in CSP Version 2) is suggested.

4) I will be happy to send out binary decks, but may not get around to submitting programs to COMMON for some time. I personally do not feel that there is anything unusual or complicated about these routines and make no comparisons with IDEAL or CSP, Version 2. They were written for our purposes and as a result have features peculiar to our needs. It may easily be that these features are wasted in your application.

```
// ASM
*LIST
*PRINT SYMBOL TABLE
000B      225C3119             ENT      SPCDR      SPECIAL CARD INPUT SUBR
0000      22645644             ENT      SRERD
008B      225C3117             ENT      SPCDP
0000 1    0001         SRERD   DC       *          REREAD CARD WITH NEW
0001 0    6973                 STX    1 EXIT+1          FIELDS
0002 0    6A74                 STX    2 EXIT+3
0003 01   65800000             LDX   I1 SRERD
0005 0    6905                 STX    1 SPCDR
0006 00   C5800000             LD    I1 0
0008 01   D40000DF             STO   L  IAREA
000A 0    700D                 MDX      RETRY
000B 1    000C         SPCDR   DC       *
000C 0    6968                 STX    1 EXIT+1     CALL SPCDR(NC,IFV,IFTV,
000D 0    6A69                 STX    2 EXIT+3         IAREA,NERRC)
000E 01   6580000B             LDX   I1 SPCDR
0010 00   C5800000             LD    I1 0
0012 01   D40000DF             STO   L  IAREA
0014 20   03059131             LIBF     CARD1
0015 0    1000                 DC       /1000
0016 1    00DF                 DC       IAREA
0017 1    00DA                 DC       ERROR
0018 0    C061         RETRY   LD       ZERO
0019 0    D061                 STO      ERRID
001A 0    C101                 LD     1 1
001B 0    D03E                 STO      IFAD+1
001C 0    D027                 STO      IFADX+1
001D 0    C102                 LD     1 2
001E 0    D042                 STO      OFAD+1
001F 0    C103                 LD     1 3
0020 0    D05B                 STO      IARAD
0021 0    C104                 LD     1 4
0022 0    D05A                 STO      EROAD
0023 0    C056                 LD       ZERO
0024 01   D480007D             STO   I  EROAD
0026 0    C057                 LD       ONE
0027 0    D057                 STO      CCNT
0028 0    6100                 LDX    1 0
0029 0    C050                 LD       ZERO
002A 0    D055                 STO      CHTYP
002B 01   6680007F     LOOPA   LDX   I2 CCNT
002D 01   C60000DF             LD    L2 IAREA
002F 01   4C04002D             BSC   L  LOOPA+2,E
0031 0    D04F                 STO      CHAR
0032 0    62CE                 LDX    2 -50
0033 0    C04D         LOOPB   LD       CHAR
0034 01   F6000162             EOR   L2 CHARL+50
0036 01   4C18003E             BSC   L  MATCH,+-
0038 0    7201                 MDX    2 1
0039 0    70F9                 MDX      LOOPB
003A 0    C043                 LD       ONE
003B 01   D480007D             STO   I  EROAD
003D 0    62CE                 LDX    2 -50
003E 01   C6000194     MATCH   LD    L2 EBCCL+50
0040 01   4C100048             BSC   L  NOTS,-
0042 0    C03C                 LD       CCNT
0043 01   95000045     IFADX   S     L1 *
0045 0    E03C                 AND      MSKAB
0046 01   EE000194             OR    L2 EBCCL+50
```

```
0048 0  E837        NOTS  OR        CHTYP
0049 0  D036              STO       CHTYP
004A 0  C034              LD        CCNT
004B 01 4C040085          BSC   L   SLCH,E
004D 01 C6000194          LD    L2  EBCCL+50
004F 0  E033              AND       MSKCH
0050 01 EC80007C          OR    I   IARAD
0052 01 D480007C          STO   I   IARAD
0054 01 74FF007C          MDX   L   IARAD,-1
0056 01 7401007F    CKEF  MDX   L   CCNT,1
0058 0  C026              LD        CCNT
0059 01 9500005B    IFAD  S     L1  *
005B 01 4C080066          BSC   L   CKEC,+
005D 0  C022              LD        CHTYP
005E 0  180C              SRA       12
005F 0  E024              AND       MSKCT
0060 01 D5000062    OFAD  STO   L1  *
0062 0  C017              LD        ZERO
0063 0  D01C              STO       CHTYP
0064 0  71FF              MDX    1  -1
0065 0  1000              SLA       0
0066 0  C018        CKEC  LD        CCNT
0067 0  9077              S         IAREA
0068 01 4C08002B          BSC   L   LOOPA,+
006A 20 03059131    TSTLC LIBF      CARD1
006B 0  0000              DC        /0000
006C 0  70FD              MDX       TSTLC
006D 0  C00D              LD        ERRID
006E 01 6580000B          LDX   I1  SPCDR
0070 01 4C200018          BSC   L   RETRY,Z
0072 0  7105              MDX    1  5
0073 0  6905              STX    1  EXIT+5
0074 01 65000076    EXIT  LDX   L1  *
0076 01 66000078          LDX   L2  *
0078 01 4C00007A          BSC   L   *
007A 0  0000        ZERO  DC        0
007B    0001        ERRID BSS       1
007C    0001        IARAD BSS       1
007D    0001        EROAD BSS       1
007E 0  0001        ONE   DC        1
007F    0001        CCNT  BSS       1
0080    0001        CHTYP BSS       1
0081    0001        CHAR  BSS       1
0082 0  4000        MSKAB DC        /4000
0083 0  00FF        MSKCH DC        /00FF
0084 0  0007        MSKCT DC        /0007
0085 01 C6000194    SLCH  LD    L2  EBCCL+50
0087 0  1008              SLA       8
0088 01 D480007C          STO   I   IARAD
008A 0  70CB              MDX       CKEF
008B 0  0000        SPCDP DC                       CALL SPCDP(NC,OAREA,NERRC)
008C 0  69E8              STX    1  EXIT+1
008D 0  6AE9              STX    2  EXIT+3
008E 01 6580008B          LDX   I1  SPCDP
0090 00 C5800000          LD    I1  0
0092 0  D04C              STO       IAREA
0093 0  C101              LD     1  1
0094 0  D0E7              STO       IARAD
0095 0  C0E8              LD        ONE
0096 0  D0E8              STO       CCNT
```

```
0097 0  C0E7          PLP   LD        CCNT
0098 01 4C0400D6            BSC   L   SRCH,E
009A 01 C480007C            LD    I   IARAD
009C 01 74FF007C            MDX   L   IARAD,-1
009E 0  E0E4          PLPR  AND       MSKCH
009F 0  D0E1                STO       CHAR
00A0 0  62CE                LDX    2  -50
00A1 01 C6000194      PLPA  LD    L2  EBCCL+50
00A3 0  E0DF                AND       MSKCH
00A4 0  F0DC                EOR       CHAR
00A5 01 4C1800AB            BSC   L   PMAT,+-
00A7 0  7201                MDX    2  1
00A8 0  70F8                MDX       PLPA
00A9 0  C0D4                LD        ONE
00AA 0  700F                MDX       EXITP
00AB 01 C6000162      PMAT  LD    L2  CHARL+50
00AD 0  D0D3                STO       CHAR
00AE 01 6680007F            LDX   I2  CCNT
00B0 01 4C1800BF            BSC   L   PSTO,+-
00B2 01 C60000DF            LD    L2  IAREA
00B4 01 4C1800BF            BSC   L   PSTO,+-
00B6 0  F0CA                EOR       CHAR
00B7 01 4C1800BF            BSC   L   PSTO,+-
00B9 0  C01B                LD        TWO
00BA 00 D5800002      EXITP STO   I1  2
00BC 0  7103                MDX    1  3
00BD 0  69BB                STX    1  EXIT+5
00BE 0  70B5                MDX       EXIT
00BF 0  C0C1          PSTO  LD        CHAR
00C0 01 D60000DF            STO   L2  IAREA
00C2 01 7401007F            MDX   L   CCNT,1
00C4 0  C0BA                LD        CCNT
00C5 0  9019                S         IAREA
00C6 01 4C080097            BSC   L   PLP,+
00C8 20 03059131            LIBF      CARD1
00C9 0  2000                DC        /2000
00CA 1  00DF                DC        IAREA
00CB 1  00DA                DC        ERROR
00CC 0  C0AD          PREID LD        ZERO
00CD 0  D0AD                STO       ERRID
00CE 20 03059131      TSTPF LIBF      CARD1
00CF 0  0000                DC        /0000
00D0 0  70FD                MDX       TSTPF
00D1 0  C0A9                LD        ERRID
00D2 01 4C2000CC            BSC   L   PREID,Z
00D4 0  70E5                MDX       EXITP
00D5 0  0002          TWO   DC        2
00D6 01 C480007C      SRCH  LD    I   IARAD
00D8 0  1808                SRA       8
00D9 0  70C4                MDX       PLPR
00DA 1  00DB          ERROR DC        *
00DB 0  C0A2                LD        ONE
00DC 0  D09E                STO       ERRID
00DD 01 4C8000DA            BSC   I   ERROR
00DF    0051          IAREA BSS       81
0130 0  0000          CHARL DC        /0000
0131 0  8420                DC        /8420
0132 0  8120                DC        /8120
0133 0  80A0                DC        /80A0
0134 0  8000                DC        /8000
```

98

```
0135 0   4420                 DC      /4420     $                  0080
0136 0   4220                 DC      /4220     *                  0090
0137 0   4120                 DC      /4120     )                  0100
0138 0   4000                 DC      /4000     -                  0110
0139 0   3000                 DC      /3000     /                  0120
013A 0   2420                 DC      /2420     ,                  0130
013B 0   0120                 DC      /0120     '                  0140
013C 0   00A0                 DC      /00A0     =                  0150
013D 0   9000                 DC      /9000     A                  0160
013E 0   8800                 DC      /8800     B                  0170
013F 0   8400                 DC      /8400     C                  0180
0140 0   8200                 DC      /8200     D                  0190
0141 0   8100                 DC      /8100     E                  0200
0142 0   8080                 DC      /8080     F                  0210
0143 0   8040                 DC      /8040     G                  0220
0144 0   8020                 DC      /8020     H                  0230
0145 0   8010                 DC      /8010     I                  0240
0146 0   6000                 DC      /6000     (-ZERO)            0250
0147 0   5000                 DC      /5000     J                  0260
0148 0   4800                 DC      /4800     K                  0270
0149 0   4400                 DC      /4400     L                  0280
014A 0   4200                 DC      /4200     M                  0290
014B 0   4100                 DC      /4100     N                  0300
014C 0   4080                 DC      /4080     O                  0310
014D 0   4040                 DC      /4040     P                  0320
014E 0   4020                 DC      /4020     Q                  0330
014F 0   4010                 DC      /4010     R                  0340
0150 0   2800                 DC      /2800     S                  0350
0151 0   2400                 DC      /2400     T                  0360
0152 0   2200                 DC      /2200     U                  0370
0153 0   2100                 DC      /2100     V                  0380
0154 0   2080                 DC      /2080     W                  0390
0155 0   2040                 DC      /2040     X                  0400
0156 0   2020                 DC      /2020     Y                  0410
0157 0   2010                 DC      /2010     Z                  0420
0158 0   2000                 DC      /2000     0                  0430
0159 0   1000                 DC      /1000     1                  0440
015A 0   0800                 DC      /0800     2                  0450
015B 0   0400                 DC      /0400     3                  0460
015C 0   0200                 DC      /0200     4                  0470
015D 0   0100                 DC      /0100     5                  0480
015E 0   0080                 DC      /0080     6                  0490
015F 0   0040                 DC      /0040     7                  0500
0160 0   0020                 DC      /0020     8                  0510
0161 0   0010                 DC      /0010     9                  0520
0162 0   1040          EBCCL  DC      /1040                        0530
0163 0   404B                 DC      /404B     .
0164 0   404D                 DC      /404D     (                  0550
0165 0   404E                 DC      /404E     +                  0560
0166 0   4050                 DC      /4050     &                  0570
0167 0   405B                 DC      /405B     $                  0580
0168 0   405C                 DC      /405C     *                  0590
0169 0   405D                 DC      /405D     )                  0600
016A 0   A060                 DC      /A060     -                  0610
016B 0   4061                 DC      /4061     /                  0620
016C 0   406B                 DC      /406B     ,                  0630
016D 0   407D                 DC      /407D     '                  0640
016E 0   407E                 DC      /407E     =                  0650
016F 0   40C1                 DC      /40C1     A        99        0660
0170 0   40C2                 DC      /40C2     B                  0670
```

```
0171 0   40C3                    DC      /40C3      C                         0680
0172 0   40C4                    DC      /40C4      D                         0690
0173 0   40C5                    DC      /40C5      E                         0700
0174 0   40C6                    DC      /40C6      F                         0710
0175 0   40C7                    DC      /40C7      G                         0720
0176 0   40C8                    DC      /40C8      H                         0730
0177 0   40C9                    DC      /40C9      I                         0740
0178 0   A0D0                    DC      /A0D0      (-ZERO)                   0750
0179 0   A0D1                    DC      /A0D1      J                         0760
017A 0   A0D2                    DC      /A0D2      K                         0770
017B 0   A0D3                    DC      /A0D3      L                         0780
017C 0   A0D4                    DC      /A0D4      M                         0790
017D 0   A0D5                    DC      /A0D5      N                         0800
017E 0   A0D6                    DC      /A0D6      O                         0810
017F 0   A0D7                    DC      /A0D7      P                         0820
0180 0   A0D8                    DC      /A0D8      Q                         0830
0181 0   A0D9                    DC      /A0D9      R                         0840
0182 0   40E2                    DC      /40E2      S                         0850
0183 0   40E3                    DC      /40E3      T                         0860
0184 0   40E4                    DC      /40E4      U                         0870
0185 0   40E5                    DC      /40E5      V                         0880
0186 0   40E6                    DC      /40E6      W                         0890
0187 0   40E7                    DC      /40E7      X                         0900
0188 0   40E8                    DC      /40E8      Y                         0910
0189 0   40E9                    DC      /40E9      Z                         0920
018A 0   20F0                    DC      /20F0      0                         0930
018B 0   20F1                    DC      /20F1      1                         0940
018C 0   20F2                    DC      /20F2      2                         0950
018D 0   20F3                    DC      /20F3      3                         0960
018E 0   20F4                    DC      /20F4      4                         0970
018F 0   20F5                    DC      /20F5      5                         0980
0190 0   20F6                    DC      /20F6      6                         0990
0191 0   20F7                    DC      /20F7      7                         1000
0192 0   20F8                    DC      /20F8      8                         1010
0193 0   20F9                    DC      /20F9      9                         1020
0194                             END
```

```
                                SYMBOL TABLE

CCNT   007F     CHAR   0081     CHARL  0130     CHTYP  0080     CKEC   0066
CKEF   0056     EBCCL  0162     EROAD  007D     ERRID  007B     ERROR  00DA
EXIT   0074     EXITP  00BA     IARAD  007C     IAREA  00DF     IFAD   0059
IFADX  0043     LOOPA  002B     LOOPB  0033     MATCH  003E     MSKAB  0082
MSKCH  0083     MSKCT  0084     NOTS   0048     OFAD   0060     ONE    007E
PLP    0097     PLPA   00A1     PLPR   009E     PMAT   00AB     PREID  00CC
PSTO   00BF     RETRY  0018     SLCH   0085     SPCDP  008B     SPCDR  000B
SRCH   00D6     SRERD  0000     TSTLC  006A     TSTPF  00CE     TWO    00D5
ZERO   007A

NO ERRORS IN ABOVE ASSEMBLY.
```

```
// ASM
*LIST
*PRINT SYMBOL TABLE
0000      22183119             ENT      SFCDR      CALL SFCDR(NW,IAREA,NC)
0057      22183122             ENT      SFCDS      CALL SFCDS
0093      22183106             ENT      SFCDF      CALL SFCDF
005D      22183117             ENT      SFCDP      CALL SFCDP(NW,IAREA)
0000 1    0001        SFCDR    DC       *
0001 0    6936                 STX    1 RESIR+1
0002 01   65800000             LDX   I1 SFCDR
0004 00   C5800000             LD    I1 0
0006 0    D034                 STO      WCNT
0007 0    904E                 S        K40
0008 0    4830                 BSC      -Z
0009 0    1010                 SLA      16
000A 0    804B                 A        K40
000B 0    7103                 MDX    1 3
000C 0    692D                 STX    1 EXIT+1
000D 01   4C080037             BSC   L  RESIR,+
000F 0    1001                 SLA      1
0010 0    D01B                 STO      CHCNT
0011 0    C1FE                 LD     1 -2
0012 0    D01D                 STO      TAD+1
0013 00   C580FFFF             LD    I1 -1
0015 0    D026                 STO      CODE
0016 01   7401003E             MDX   L  CCNT,+1
0018 0    4026                 BSI      RCD
0019 0    C023                 LD       BUFA
001A 0    D00F                 STO      CFAD
001B 0    D00F                 STO      CFAD+1
001C 0    D011                 STO      FAD+1
001D 0    6100                 LDX    1 0
001E 20   03059130             LIBF     CARD0
001F 0    0000                 DC       /0000
0020 0    70FD                 MDX      *-3
0021 0    C01A                 LD       CODE
0022 01   4C040028             BSC   L  CONV,E
0024 01   F480002A             EOR   I  CFAD
0026 01   4420003F             BSI   L  RCD,Z
0028 20   225C5144    CONV     LIBF     SPEED
0029 0    0000                 DC       /0000
002A 1    002B        CFAD     DC       *
002B 1    002C                 DC       *
002C 1    002D        CHCNT    DC       *
002D 01   C500002F    FAD      LD    L1 *
002F 01   D4000031    TAD      STO   L  *
0031 0    7101                 MDX    1 1
0032 01   74FF0030             MDX   L  TAD+1,-1
0034 01   74FF003B             MDX   L  WCNT,-1
0036 0    70F6                 MDX      FAD
0037 01   65000039    RESIR    LDX   L1 *
0039 01   4C00003B    EXIT     BSC   L  *
003B      0001        WCNT     BSS      1
003C      0001        CODE     BSS      1
003D      0001        BUFA     BSS      1
003E 0    0000        CCNT     DC       0
003F 1    0040        RCD      DC       *
0040 0    C811                 LDD      BUFAD
0041 0    18D0                 RTE      16
0042 0    D80F                 STD      BUFAD
0043 0    D007                 STO      RAD
0044 0    800F                 A        ONE
```

```
0045 0  D0F7                 STO      BUFA
0046 0  C00E                 LD       K80
0047 01 D480004B             STO   I  RAD
0049 20 03059130             LIBF     CARD0
004A 0  1000                 DC       /1000
004B 1  004C         RAD     DC       *
004C 01 74FF003E             MDX   L  CCNT,-1
004E 0  1000                 NOP
004F 01 4C80003F             BSC   I  RCD
0052    0000         BUFAD   BSS   E  0
0052 1  0098                 DC       CBUFA
0053 1  00EA                 DC       CBUFB
0054 0  0001         ONE     DC       1
0055 0  0050         K80     DC       80
0056 0  0028         K40     DC       40
0057 1  0058         SFCDS   DC       *
0058 0  C0E5                 LD       CCNT
0059 01 4410003F             BSI   L  RCD,-
005B 01 4C800057             BSC   I  SFCDS
005D 1  005E         SFCDP   DC       *
005E 0  6931                 STX    1 PRESI+1
005F 01 6580005D             LDX   I1 SFCDP
0061 0  C101                 LD     1 1
0062 0  D01B                 STO      PFAD+1
0063 00 C5800000             LD    I1 0
0065 0  90F0                 S        K40
0066 0  4830                 BSC      -Z
0067 0  1010                 SLA      16
0068 0  80ED                 A        K40
0069 0  D0D1                 STO      WCNT
006A 0  7102                 MDX    1 2
006B 0  6926                 STX    1 PEXIT+1
006C 01 4C08008F             BSC   L  PRESI,+
006E 0  1001                 SLA      1
006F 0  D01B                 STO      PCHCN
0070 0  6100                 LDX    1 0
0071 0  C8E0                 LDD      BUFAD
0072 0  18D0                 RTE      16
0073 0  D8DE                 STD      BUFAD
0074 0  D019                 STO      PAD
0075 0  80DE                 A        ONE
0076 0  D013                 STO      PCFAD+1
0077 0  80C3                 A        WCNT
0078 0  D010                 STO      PCFAD
0079 0  D006                 STO      PTAD+1
007A 0  C010                 LD       PCHCN
007B 01 D480008E             STO   I  PAD
007D 01 C400007F     PFAD    LD    L  *
007F 01 D5000081     PTAD    STO   L1 *
0081 0  7101                 MDX    1 1
0082 01 74FF007E             MDX   L  PFAD+1,-1
0084 01 74FF003B             MDX   L  WCNT,-1
0086 0  70F6                 MDX      PFAD
0087 20 225C5144             LIBF     SPEED
0088 0  0001                 DC       /0001
0089 1  008A         PCFAD   DC       *
008A 1  008B                 DC       *
008B 1  008C         PCHCN   DC       *
008C 20 03059130             LIBF     CARD0
008D 0  2000                 DC       /2000
```

103

```
008E 1  008F        PAD   DC      *
008F 01 65000091    PRESI LDX  L1 *
0091 01 4C000093    PEXIT BSC  L  *
0093 1  0094        SFCDF DC      *
0094 20 03059130          LIBF    CARD0
0095 0  3000              DC      /3000
0096 01 4C800093          BSC  I  SFCDF
0098    0052        CBUFA BSS     82
00EA    0052        CBUFB BSS     82
013C                      END
```

```
                              SYMBOL TABLE

BUFA   003D    BUFAD  0052    CBUFA  0098    CBUFB  00EA    CCNT   003E
CFAD   002A    CHCNT  002C    CODE   003C    CONV   0028    EXIT   0039
FAD    002D    K40    0056    K80    0055    ONE    0054    PAD    008E
PCFAD  0089    PCHCN  008B    PEXIT  0091    PFAD   007D    PRESI  008F
PTAD   007F    RAD    004B    RCD    003F    RESIR  0037    SFCDF  0093
SFCDP  005D    SFCDR  0000    SFCDS  0057    TAD    002F    WCNT   003B

NO ERRORS IN ABOVE ASSEMBLY.
```

```
// ASM
*LIST
*PRINT SYMBOL TABLE
0000      225D9563              ENT        SPRNT
0072      225D95E3              ENT        SPRPT
0080      225D95C3              ENT        SPRPC
0000 1    0001          SPRNT   DC         *                  SPECIAL FORTRAN PRINT SUBR
0001 0    6929                  STX     1  EXIT+1                 FOR ACCOUNTING
0002 0    6A2A                  STX     2  EXIT+3
0003 01   65800000              LDX    I1  SPRNT
0005 0    C101                  LD      1  1
0006 0    D018                  STO        MFOA+1
0007 20   176558F1      TSTPL   LIBF       PRNT1
0008 0    0000                  DC         /0000
0009 0    70FD                  MDX        TSTPL
000A 00   C5800000              LD     I1  0
000C 0    D028                  STO        OAREA
000D 0    9023                  S          CONST
000E 01   4C080012              BSC    L   *+2,+
0010 0    C020                  LD         CONST
0011 0    D023                  STO        OAREA
0012 0    C01F                  LD         OARAD
0013 0    8021                  A          OAREA
0014 0    801E                  A          ONE
0015 0    D00B                  STO        MTOA+1
0016 0    6200                  LDX     2  0
0017 0    7102                  MDX     1  2
0018 0    6916                  STX     1  EXIT+5
0019 0    C016                  LD         ZERO
001A 0    D019                  STO        EPGFL
001B 0    9019                  S          OAREA
001C 00   D4000001              STO    L   /0001
001E 01   C6000020      MFOA    LD     L2  *
0020 01   D5000022      MTOA    STO    L1  *
0022 0    72FF                  MDX     2  -1
0023 0    1000                  SLA        0
0024 0    7101                  MDX     1  1
0025 0    70F8                  MDX        MFOA
0026 20   176558F1              LIBF       PRNT1
0027 0    2000                  DC         /2000
0028 1    0035                  DC         OAREA
0029 1    008F                  DC         ERROR
002A 01   6500002C      EXIT    LDX    L1  *
002C 01   6600002E              LDX    L2  *
002E 01   4C000030              BSC    L   *
0030 0    0000          ZERO    DC         0
0031 0    003C          CONST   DC         60
0032 1    0035          OARAD   DC         OAREA
0033 0    0001          ONE     DC         1
0034      0001          EPGFL   BSS        1
0035      003D          OAREA   BSS        61
0072 1    0073          SPRPT   DC         *
0073 0    69B7                  STX     1  EXIT+1
0074 01   65800072              LDX    I1  SPRPT
0076 20   176558F1      TSTPF   LIBF       PRNT1
0077 0    0000                  DC         /0000
0078 0    70FD                  MDX        TSTPF
0079 0    C0BA                  LD         EPGFL
007A 00   D5800000              STO    I1  0
007C 0    7101          SPREX   MDX     1  1
007D 0    69B1                  STX     1  EXIT+5                     106
007E 0    6AAE                  STX     2  EXIT+3
```

```
007F 0   70AA                  MDX       EXIT
0080 1   0081         SPRPC    DC        *
0081 0   69A9                  STX     1 EXIT+1
0082 01  65800080              LDX    I1 SPRPC
0084 00  C5800000              LD     I1 0
0086 0   1004                  SLA       4
0087 0   E005                  AND       MSK
0088 0   8005                  A         CCWD
0089 0   D001                  STO       CCDWD
008A 20  176558F1              LIBF      PRNT1
008B 1   008C         CCDWD    DC        *
008C 0   70EF                  MDX       SPREX
008D 0   0FF0         MSK      DC        /0FF0
008E 0   3000         CCWD     DC        /3000
008F 1   0090         ERROR    DC        *
0090 0   C0A2                  LD        ONE
0091 0   D0A2                  STO       EPGFL
0092 01  4C80008F              BSC     I ERROR
0094                           END
```

SYMBOL TABLE

```
CCDWD 008B      CCWD  008E      CONST 0031      EPGFL 0034      ERROR 008F
EXIT  002A      MFOA  001E      MSK   008D      MTOA  0020      OARAD 0032
OAREA 0035      ONE   0033      SPREX 007C      SPRNT 0000      SPRPC 0080
SPRPT 0072      TSTPF 0076      TSTPL 0007      ZERO  0030

NO ERRORS IN ABOVE ASSEMBLY.
```

SESSION NUMBER T.1.4.

SPEAKERS
J.L. TUNNEY, JR., CHAIRMAN S/360 HARDWARE COMMITTEE

DISCUSSION
THE MEETING CONSISTED OF A SOUND-OFF ON HARDWARE PROBLEMS, AN INFORMAL SURVEY OF MINIMUM CONFIGURATIONS THOUGHT TO BE REQUIRED FOR VARIOUS SOFTWARE PACKAGES, AND A SURVEY OF USERS OF NON-STANDARD EQUIPMENT.

REPORT ON MEETING OF SYSTEMS DIVISION, 360 PROJECT, HARDWARE COMMITTEE - 9/7/67

Attendance: Approx. 100

Chairman: James L. Tunney
J. R. Ahart, Inc.
627 Salem Avenue
Dayton, Ohio 45406
Telephone 513 - 278-4754

The first meeting of this committee since its organization was divided into three sections:

1. A discussion of users' hardware problems and how they have been remedied.

2. A discussion of software systems and what minimum hardware requirements have proven to be in practice.

3. A survey among those attending of new or unusual equipment that they might have on their "360".

SECTION I - HARDWARE PROBLEMS

"1442" Card Read-Punch

The majority of the complaints about this unit involved an apparent lack of checking circuits. Reports included machines that would punch and read without cards, others instances where the program would proceed normally without the punches actually going down, and still other problems of punching without feeding the cards. One user having both a "1442" and a "2501" found that in order to "CATALR" from the "1442", both SYSIN and SYSRDR had to be assigned to the "1442".

"1443" Printer

Most problems with the "1443" were evidently blamed on static electricity by the FE's. One suggestion was to install a humidifier, another was to hang tinsel on metal frame over which the paper moves, a third attendee mentioned that the box of paper must set on the steel frame. Richard Pratt said that their printer could throw the type-bar without leaving the ready state!

"2501" Card Reader

Four attendees reported that dust accumulated in one of the read stations and caused frequent read checks. It was felt that on heavily used equipment once-a-week PM was not sufficient to relieve the problem and a design change is needed.

"2540" Card Read-Punch

Read-side: One person reported a machine that read 1200 cards per minute instead of 1000/minute when delivered. This caused excessive reader checks and proved difficult to find.

Punch-side: Three installations reported excessive numbers of partly punched holes without punch checks. New dies proved to be the only effective remedy.

"1403" Printer

One user reported that on the QN print chain which is the PL/I set with 45 preferred graphics, the single quote mark is a non-preferred character. The net result is that any line having an apostrophe prints at 330 lines/minute instead of 1100! Another reported an oil leak that took 3 days to fix. Moral-put a pan under it! Another installation reported that transfer of static electricity from a belt to a wire resulted in character substitution.

"2311" Disk Drive

Six users reported oil leaks or hydraulic problems. Several more said they frequently couldn't I.P.L. from certain drives. One said the fix was to power the "2311" down, slam the lid, and restart it! Of more serious concern was the report that I.B.M. is currently installing desensitizers on the heads of all "2311's". At least two people reported that they had problems reading old files after this engineering change. A theory was advanced that possibly old "2311's" were already below sensitivity specs!

"029" Keypunch

Duplicating ECBDIC cards with or without the printing mechanism engaged is reported to damage the code plates. I.B.M. offers a special "024" for this purpose but the users felt that the "026" should be modified to do the job since it is the "360" Keypunch!

SECTION II - MINIMUM HARDWARE REQUIREMENTS FOR AVAILABLE SOFTWARE

T.O.S. No comment other than no amount of core will speed it up!

D.O.S. 32K and 2 Disks should be minimum.

D.O.S.-FORTRAN IV 32K minimum.

D.O.S.-R.P.G. 32K minimum.

D.O.S.-COBOL 32K marginal, 64K adequate.

D.O.S.-PL/I same as FORTRAN - 32K minimum.

D.O.S.-Bill of Matl. Proc. with COBOL 64K and several disks.

M.I.T. Civil Eng. Pkg. 128K and 3 disks (comparable to what used to fit on 40 or 60K "1620"!

Proj. Mgmt. System Rumor suggests 4 disks or 4 tapes!

D.O.S. Utilities and Sort-Merge Good even in 16K!

SECTION III - NEW OR UNUSUAL EQUIPMENT

Record Overflow on "2841" An excellent feature for $10/month that is rendered useless by lack of software support under D.O.S.

Storage Protect Out of 12 reported installations, 3 said it took 2 months or longer to get it to work!

"1012" Paper Tape Read-Punch R.P.Q. Leonard Sites, Sunstrand Aviation, Denver, Colorado.

"1231" Optical Reader Miami-Dade Junior College, Miami, Florida reported that this is a good unit but that it is only supported under B.P.S. Lack of D.O.S. support results in an unrecoverable unit check when D.O.S. is in core and someone readies the "1231" or other unsupported device!

"565" CALCOMP Plotter on "2701" Bethlehem Steel Company reports that it runs at ½ "1620" speed under this hook-up.!

"470" CALCOMP Plotter off-line, 9 channel mag tape. Don McIlvain reports that 9 channel tape characters are very unusual and some extra BAL programming is required for this arrangement.

"2314" Disk Unit Lear-Siegler Company, Grand Rapids, Michigan.

I.B.M. Visual Display Terminal Also Lear-Siegler Company.

Data Cell Data Corporation, Dayton, Ohio

R.C.A. Video-data Terminal Also Data Corporation

SESSION NUMBER T.1.6.

SPEAKERS

PANEL ON JOB DESCRIPTIONS AND PERSONNEL SELECTION IN MEDIUM TO SMALL INSTALLATIONS

PAUL BICKFORD, DEPAUW UNIVERSITY
ROBERT CORNELL, FED. RES. BANK, MINNEAPOLIS
DR. L.H. BAKER, PIONEER HI-BRED CORN CO.
DR. PAUL HERWITZ, I.B.M.

Presentation Summary of a Member of a Panel

on

"Personnel Selection and Job Descriptions in Medium-to-Small Computer Installations".

By

Paul A. Bickford

INTRODUCTION

The comments below are centered around a small university (2,400 students) Computer Center which has an IBM 1620 20k, 1311 configuration for education, a compliment of unit record equipment and an IBM 1401 8k, 3-1311 system for University Administration Data Processing. Some of the main tasks of the installation are: (1) Maintenance of a large Alumni file (2) Student Admission files (3) Registration (4) Grade processing and reporting (5) Maintenance of Student Information files and (6) Developments of Campaign Accounting files. The Administrative Data Processing group has always operated under a closed shop operation philosophy whereas the Education group has always operated under an open shop philosophy. All facilities are located in one building. The Staff consists of eight people, one Secretary, two Key Punch operators, one File Clerk, one Programmer, one Machine Room Supervisor, Part-time Delivery Boy and a Director.

PERSONNEL SELECTION

The procedures for hiring people are straight forward and derived primarily out of necessity because of a very limited reservoir of prospective experienced people. Attributes that the prospective employee's must have is a pleasant personality and who is reasonably easy to get along with. This is mandatory because of the size of the installation. One is constantly rubbing elbows and communicating with another member of the staff and emotional friction here would be more detrimental in disrupting output than an ailing CPU.

EDUCATION

A high school diploma is required. College experience is desired for the positions of: Programmer and Machine Room Supervisor.

SEX

Women are employed for the positions of File Clerk and Key Punch operators. Men are employed for the positions of Machine Room Supervisor, Programmer and Machine operators.

AGE

Women:	18 to 60
Men:	18 to ? depends upon persons experience and our needs. Presently, the oldest man is 31 years.

JOB DESCRIPTIONS

Because the installation is small, there is an overlapping of job responsibilities. Each person must perform numerous and varied tasks in order that continunity may be given to the flow of work thru the Computer Center. Sickness and vacations sometime present serious difficulties because the overlapping of responsibilities never seems to be adequately or properly defined. The job titles are general ones and the "other" tasks each person performs will be described.

KEY PUNCH OPERATOR(S)

Each must know how to operate the 548 Interpreter, 083 Sorter, 514 Reproducer, 085 Collator and the 407 Accounting Machine in addition to the 026 Key Punch and 056 Verifier. They must eventually become familiar with (know by heart) the field formats of 20 plus different cards. On some jobs, the Key Punch operator will receive information directly in the mail, key punch and verify it, select the proper Interpreter board (and sometimes wire one) process the cards thru the 548 and then file them in the proper file. Many times the operator must refer to a master code book in order that she may supply neede source information codes to punch into cards. Currently, one operator is learning to program the 1401 in RPG language.

FILE CLERK

This individual primarily performs the task of hand filing updates to the 75,000 card Alumni file which constantly changes. Presently, this procedure is followed even though the Alumni files reside on 1311 disks, and shortly this hand task will be eliminated. There are another 15 files that constantly change and the file clerk spends most of her time performing this function. However, she must also be able to Key Punch and operate the other Unit Record equipment when either of the other two operators are ill or on vacation.

SECRETARY

Besides performing the routine tasks of a secretary ie, typing letters, distributing mail and filing etc. she must also be able to Key Punch and operate most of the Unit Record equipment. She also updates Alumni address changes in an Alumni Directory which is 3 feet thick. Code sheets that are sent to the Computer Center are supplied with additional codes (such as the student and alumni alphabetic identification numbers) by the Secretary.

MACHINE ROOM SUPERVISOR

Much of the daily routine is closely watched and controlled by the Machine Room Supervisor. He operates, wires and programs all equipment. He accepts job requests (verbally or in writting) and schedules them in a manner that will keep all equipment as busy as possible. With the many tasks that must "mesh" perfectly before a job is completed he must be constantly aware of the state of completion that each job is at so that work can progress along. This "art" of scheduling seems to work reasonably well in our small shop environment. The peak work periods seem to always present serious scheduling problems and sometimes some jobs get completely neglected for awhile. The Machine Room Supervisor carries the heaviest responsibility load in seeing that jobs are finished in a reasonable length of time and are well done. He often communicates directly with the Faculty as well as all Administrative offices. One-fourth to one-half of his time, approximately, is spent in programming the less complex "One-shot" jobs that occur frequently.

PROGRAMMER

Is responsible for programming the more complex jobs such as Development Fund Accounting, general Alumni and Student file maintenance. He frequently takes requests directly, does what ever systems work is necessary, writes and key punches his own program as well as assemble and test the program. On half of the jobs he is the only person who knows precisely how his programs function. Having only one person responsible for major jobs leaves a small installation in jeopardy. The

Programmer also writes the operating instructions for all of his programs and is supposed to flow chart them also.

He is also responsible for knowing the more intimate details of the IBM Sort Package, the Disk Utility Programs and Disk File Organization routines.

DIRECTOR

Last, but I hope not least, the Director has the ultimate responsibility of the complete operation and he spends most of his time communicating with the various departments of the University and monitors the progress of new needed applications. Sometimes he plays the role of Programmer for a week or two, Machine Room Supervisor for awhile besides carrying on the usual routine of answering complaints (we have a few) and carrying on with correspondence. He also teaches a course in Basic Computer Programming.

PROGRAMMER SELECTION AT THE FEDERAL RESERVE BANK OF MINNEAPOLIS

Robert Cornell

OUTLINE

sic Structure

A. Programming staff of our company can be broken down into 5 basic units.
   1. Programmer
   2. Procedures Analyst
   3. Systems Analyst B
   4. Systems Analyst
   5. Senior Systems Analyst

   Advancement along the scale follows naturally in that order, with the "programmer" being the basic unit, or starting point.

C. Even from the beginning, a programmer must be trained as a high-powered individual in the many levels of work into which he may advance. His duties include:
   1. Present job analysis.
      a. Discuss job to be programmed with supervisor or department head in charge of the operation to be programmed.
      b. Study and learn the "ins and outs" of the job by working with the clerk; acquire a thorough understanding of all facets of the job, including special operating methods, periodic, regular and irregular error procedures.
   2. Program design.
      a. Determine program objectives through consultation with bank and systems personnel.
      b. Construct a flow chart of the work to be done by the computer.
   3. Program writing/testing.
      a. Writes the program - ideally in any one of several programming languages; selection of the one most suitable to the job.
      b. Designs test procedure and sees to the preparation by keypunchers of a "test-data deck".
      c. Run, test, debug program.
   4. Installation/Implementation.
      a. Assist in preparation of operator manuals.
      b. Performs any necessary training of departmental and data processing personnel concerned with the handling of the job.
   5. Special Projects.
      a. Normal procedure is to assign new program~~s~~mers to a "system" when hired.
      b. Most programmers will also be required, from time to time, to help in the planning and programming of any special one-time shots.

Hiring Techniques

A. We hire on three things: 1) Pat score, 2) willingness to work; and, 3) ability to learn.

B. Requirements

1. Education - High school diploma with good record, indication of math interest. Any additional education, including college level statistics, economics, accounting, or even IBM courses in functional wiring or programming are considered; if absent, a willingness to take course work to fill in the gaps must be present.
2. Previous experience. We promote from within and train ourselves. Experience required varies with the job, and may be as low as none for a programmer, to 5 years or more of bank experience for a Senior Systems Analyst. We subscribe to the basic belief that "Persons with an aptitude for this type of work (programming) can usually gain job knowledge rapidly, but without the aptitude, no amount of experience will produce a competent programmer.
3. Communications - In all levels of work, a fair amount of communication with other bank personnel will be required. The importance of finding people who are willing and able to communicate <u>concretely and calmly</u> with others cannot be over emphasized. In his limited supervisory level of work, the ability to communicate will be of great value.
4. Personal characteristics.
   a. Programming at any level requires initiative and ingenuity. People must be found with the ability to absorb through observation, to suggest and implement improvement.
   b. Programming is time consuming work, and requires a good amount of initiative to complete a project <u>on schedule</u>.
   c. Honesty important. As an employee moves into the level of systems analysis, his principal goal will be to devise improved methods, reduce costs and improve bank services. Among these as well will be the responsibility to provide safeguards against falsification and embezzlers, and against destruction - whether inadvertant or careless - of valuable records.

Management Installation Division

ORGANIZATION AND SELECTION

by

Dr. Paul S. Herwitz
IBM Corporation
Old Orchard Road
Armonk, New York 10504
(914) 765 - 4543

Thursday, 8:30 A.M.

Text, 7 pages
Graphics, 5 pages

ORGANIZATION AND SELECTION

Although in the final analysis organization is people oriented, in IBM we have a structure of jobs which seems to work very well in all of our programming areas whether the group is large or small. I will describe this structure to you in rather general terms without getting into a completely detailed description of the job responsibilities and requirements. As a matter of fact, this is an opportune time to talk about the subject because I have been involved during the past nine months in a study of the job descriptions that have existed for programmers for the last five years. We are just about at the end of this now, and are making a number of revisions. In order to describe the key positions correctly we conducted extensive interviews of programmers throughout the company, and as a result the new descriptions give a pretty accurate description of the basic programming jobs being performed today.

First of all (Figure 1) there are two ways to enter programming in IBM - with or without a college degree. Generally speaking the college graduate can have any degree although we prefer a technical degree. (A little later on I'll give you a breakdown of the educational background of our people.) When the college graduate is hired he enters in a classification called pre-professional which is not exempt from the Federal Wage and Hours Laws. Depending upon the division in IBM in which he is hired, our new pre-professional may have anywhere from six to twenty-six weeks of programmer training. The twenty-six week training course is populated by candidates for our programming systems activities. The first twelve weeks of this course are formal class lecture with some hands-on experience. During the last fourteen weeks the trainee is a member of a kind of programming job shop, if you will, where he is managed by experienced programming managers and where his assignment is to program actual applications on a kind of sub-contract basis. This is a type of very careful on-the-job training that is closely supervised by people who have extensive management and training experience. At the end of the twenty-six week period, those trainees who have successfully completed their assignment (and not all of them do) are given a permanent assignment in one of the regular programming groups.

Those new hires who do not go into the twenty-six week course, but who have a shorter term - six or eight weeks, receive formal lectures and, of course, some hands-on training. At the end of this formal training period they go directly into a permanent assignment with one of the programming groups.

Once the trainee has received his first permanent assignment, and from then on, he is under the direct technical supervision of an experienced programmer. This experienced programmer will usually be someone with at least two or more years of experience, and may be the actual first line manager of the project.

From this point on, then, the pre-professional's progress depends upon what he can demonstrate. On the average, after twelve to eighteen months, the pre-professional is eligible for promotion to exempt status. If he does not show that he is ready for promotion after two years, then we feel he is not eligible to continue as a programmer. This means either reassignment, or termination from the company.

For the non college graduate, we have several programming technician positions which are designed to take care of two problems. The first problem we are trying to solve is that of taking care of the non-professional activities that go with every programming job. When I say non-professional I am really speaking in the legal sense as implied by the Fair Labor Standards Act; and I am not trying to raise the question of whether programming really is a profession or not. Remember that to be eligible for a professional exemption one must have as his primary duty "work requiring knowledge of an advanced type in a field of science or learning customarily acquired by a prolonged course of specialized intellectual instruction or study, and this work must require the consistent exercise of discretion and judgment." Alternatively, the work must be "original or creative in character in a recognized field of artistic endeavor" and the result must depend "primarily on the invention, imagination, or talent of the employee." Clearly there is much routine activity that goes along with all programming jobs but does not meet the definition just stated for professional exemption. Some of these activities are for example, setting up job decks for machine processing, key punching correction cards, preparing in-put data for automated flowcharting, to some extent generating data for program testing, assembling program documentation and writing procedures for the computer operator to use, and up-dating operating systems by use of specifically designed utility programs, and so on. Using these activities as a basis, we have defined our first technician level job to consist primarily of such activities. We think that one Programming Operations Aide can perform these duties for a group of about a half dozen full time programmers.

The educational requirement for this position is a high school diploma. This is a new position that has been in existence in IBM for less than a year, and it is frankly experimental. We won't know for another year or two just how well it will work out. If the candidate has college credits but no college degree, after his initial training period he becomes a Programming Technician. The job of the Programming Technician includes some of the activities of the Programming Operations Aide, but also includes coding of well defined sub-routines from detailed flowcharts.

At the top of our technician path we have a Senior Programming Technician. His duties are primarily to flowchart, code, and debug complete computer programs from well defined specifications. These programs are usually at the level of routine utility programs, and are essentially always self-contained. The Senior Programming Technician is of course also responsible for the documentation of his program, and more or less for the installation of the program in an operational enviroment.

The second problem we hope to solve by use of the technician career path is, of course, the problem of scarcity of qualified programmers. We hope to attack this problem in two ways. First of all, if indeed we do succeed in defining the non-professional activities that accompany the program task and in delegating these to our technicians, then we have relieved the professional programmer of some of his less productive activity; and he will devote his time to more creative tasks, such as systems specification and design. The net result then is to let the more creative programmer spend more of his time in creative activity, thus increasing the total productivity of the entire programming group.

Secondly its clear that many of the non-professional activities I have described don't require the attention of a college graduate. Thus, use of the technician positions permits us to consider a group of prospects that is much broader than we could consider without these positions. Our first level technician requires only a high school or prep school diploma. The top level technician requires two years of college or the equivalent.

We think the technician career path is quite good-the first level technician, the Programming Operations Aide, can either learn programming, or can go into machine operations. The top level technician, the programming specialist, is certainly sophisticated enough to go into operations management of a reasonably large computing center. On the other hand, if indeed he truly demonstrates a keen understanding of programming, and if his manager makes the business judgment that he can be successful in programming, then he to can be considered for promotion to the same exempt position to which we promote the pre-professional. This is not a loophole; it is up to management to present evidence that the man truly has learned enough about the job that he can be expected to perform in all ways as well as the pre-professional who was promoted. Our statistics show that some twenty or twenty-one percent of our professional programmers do not have a college degree.

Our first key position, then, is that held by our first level exempt programmer. Traditionally he is known as an Associate Programmer. The position is key because the Associate Programmer must truly demonstrate that his career lies in programming.

The next key position, and probably the most important one we have, is a position at what we call the staff level (Figure 2). There are really three jobs at this level - all equally important to the company, and all comparably rewarded. It is at this level that our first true manager appears. This Project Programmer is in all respects a manager; he is responsible for hiring, training, supervising, salary administration, promotion, and firing. We expect him to be technically qualified to direct a project, and to be fully involved technically in the programming activities of the project. Ideally he manages a group of six to ten programmers. If the group is small, he can spend roughly seventy percent of his time in technical matters, the remaining time being given over to administrative and personnel activities. If the project is larger, then he will typically have in his group a programmer at his own level who will act as a project leader. This project leader is called a Staff Programmer and is the second of the three jobs I mentioned. The project leader has all of the technical responsibilities that the project manager has, but does not have the administrative and personnel responsibilities.

Roughly half of our staff level programmers are managers, another quarter perform the project leader duties, and the remaining staff level programmers-the third job-act in the classical staff capacity - that is, they perform as an extension of management. Generally speaking they act as trouble shooters, as gatherers and analysisers of data, as technical consultants, and to some extent can make decisions when the authority has been delegated to them. They also double in brass as project leaders when circumstances warrant-usually for short periods of time.

We are organized this way because we want our projects to be small, and we think the person in the best position to perform the managerial duties is really the leader of the project. After all, he is the man on the spot who is in the best position to know the capabilities of the people who work for him and, therefore, given adequate training, he is in the best position to make the managerial decisions that relate to his people.

The use of the Staff Programmer as a project leader is really a compromise. Since IBM's programming activities are expanding almost as rapidly as the programming field itself, it is always difficult to find people who are well qualified to assume the combined technical and administrative responsibilities of first line management as we conceive it. Moreover, many of our technically qualified people are just not interested in becoming managers. Thus there is always a scarcity of first level managers and we are faced with the necessity of using technical project leaders. The problem we have to guard against is that if a project grows too large then its first level manager may tend to shift the administrative and personnel responsibilities onto the shoulders of the project leader. The project leader then becomes a manager without portfolio and without accountability.

From the technical standpoint, the project manager or project leader works from objectives. He is responsible for a complex project, and he is responsible for specification, negotiation of interfaces, implementation, documentation, the whole works. In general, he is not responsible for the total programming system or large application, but for a complex but recognizable sub-section. Typical examples would be a FORTRAN compiler, a set of mathematical subroutines, etc. The total systems or applications responsibility rests with the next level or even the one beyond that. I won't go into detail about these next two levels except to say that we have both managerial and non managerial positions at these higher levels as you can see from Figure 2. We have dual promotional opportunities in the managerial and technical activities.

A little arithmetic (Figure 3) will show that if we restrict the project manager to a group of six and if we restrict the second and third level managers each to having six managers report to him, then if we include the managers in our counts we can take care of a group of fourty-three people with only two levels of management and this grows to 259 people if we have one third level manager. Analogously, if we allow each first level manager to manage ten people but still say that a second or third level manager may only have six managers reporting to him then the three levels of management include 403 people. In the first case if the ratio of people to first level managers is six to one then the overall man-manager ratio comes out five to one. If the first level ratio is ten to one, then the overall ratio becomes slightly more than eight to one.

I have just one more comment on the responsibilities of the project manager. Clearly if the programs to be written are not so complex as to require six programmers or more on the particular project, there is no reason why a project manager may not have the responsibility for implementing a number of small programs. In this case he still manages a small group in which one or more of his programmers are responsible for a complete program.

We think the entire structure is quite flexible, and have found that it works very well for us over a variety of technical activities.

As I promised earlier, I want to show you a breakdown of the educational background of our programmers as of the end of 1966. Figure 4 shows the educational level and Figure 5 the field of study of our exempt programmers. Four percent of the records I consulted did not give the educational level, and fifteen percent did not carry the field of study. Otherwise, the figures should be self-explanatory.

Finally, I'd like to turn my attention now briefly to the question of selection. I say briefly because I don't have too much to tell you. I have talked about our educational requirements. Beyond this, all inexperienced candidates are required to take the current version of the PAT test. We are presently involved in a study which we hope will help us better validate the PAT test and which we hope will also give us a lead on other tests which might possibly be used in the selection process. Presently our study has given us ample indication that we don't understand all the ramifications of the PAT test. This test was originally validated against performance in training programs. It has never been satisfactorally validated to my knowledge in IBM against actual programmer performance. We are attempting to do this now, and will make the results available after the study is completed.

We also have indications that there will be one or two other tests that might be very useful in predicting whether or not a programmer will be successful. Unfortunately, at this point in time it is premature to discuss our findings. Again, though, we will be happy to make our information available when the study is complete.

Beyond this, I can only tell you that the judgments you must make when you hire programmers are bound to be subjective. In my own experience I have found that the interviewers who are right more often than wrong in the subjective judgments are the interviewers who are most people oriented. In lieu of any other more objective indicators, I think you must select as interviewers those people on your technical staff who appear to be most sensitive to the people around them.

EARLY PROGRAMMING POSITIONS

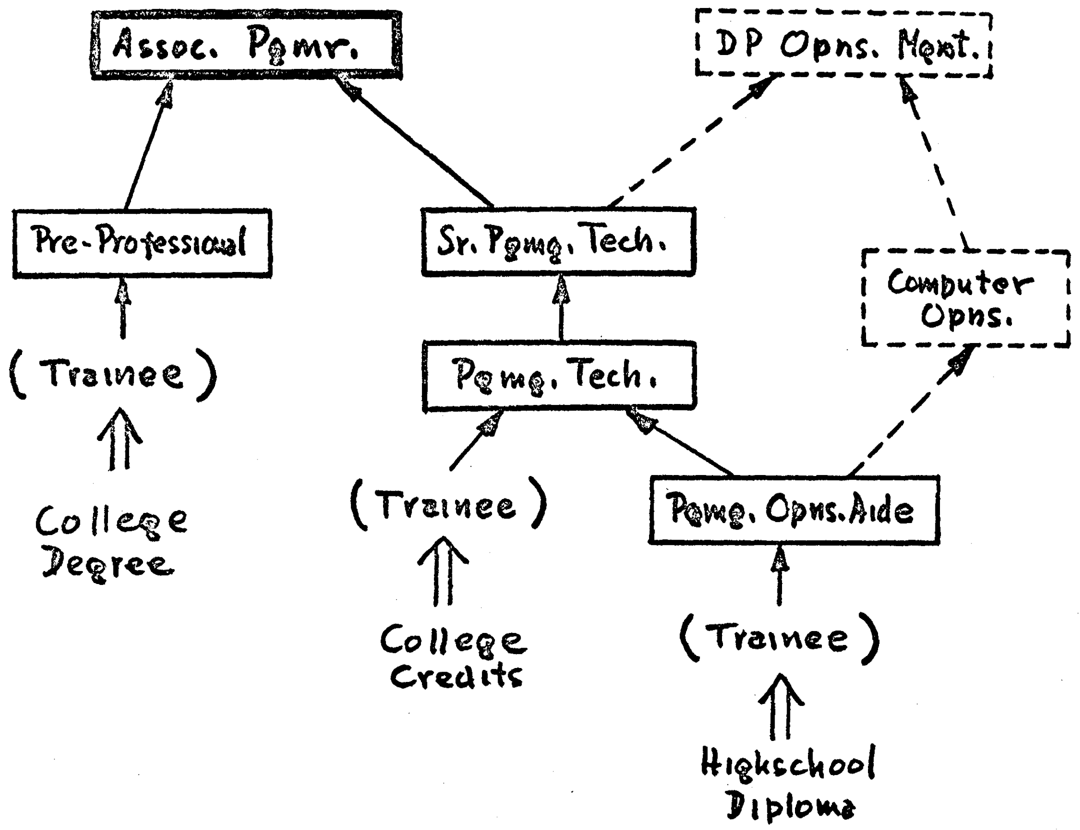


Figure 1

EXEMPT PROGRAMMING POSITIONS

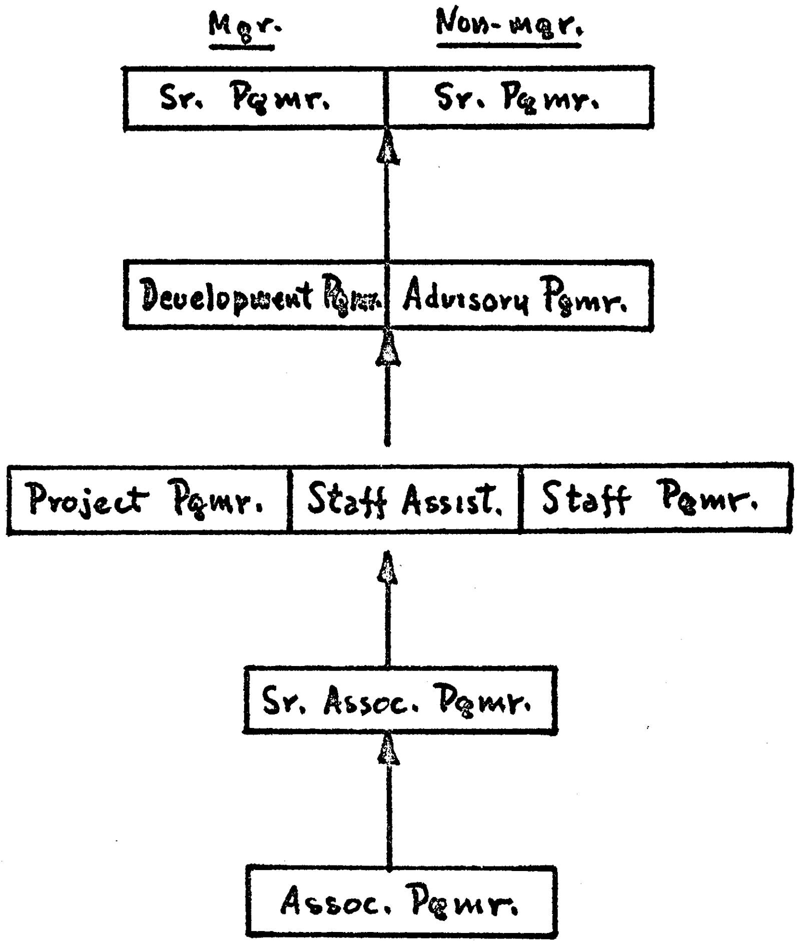
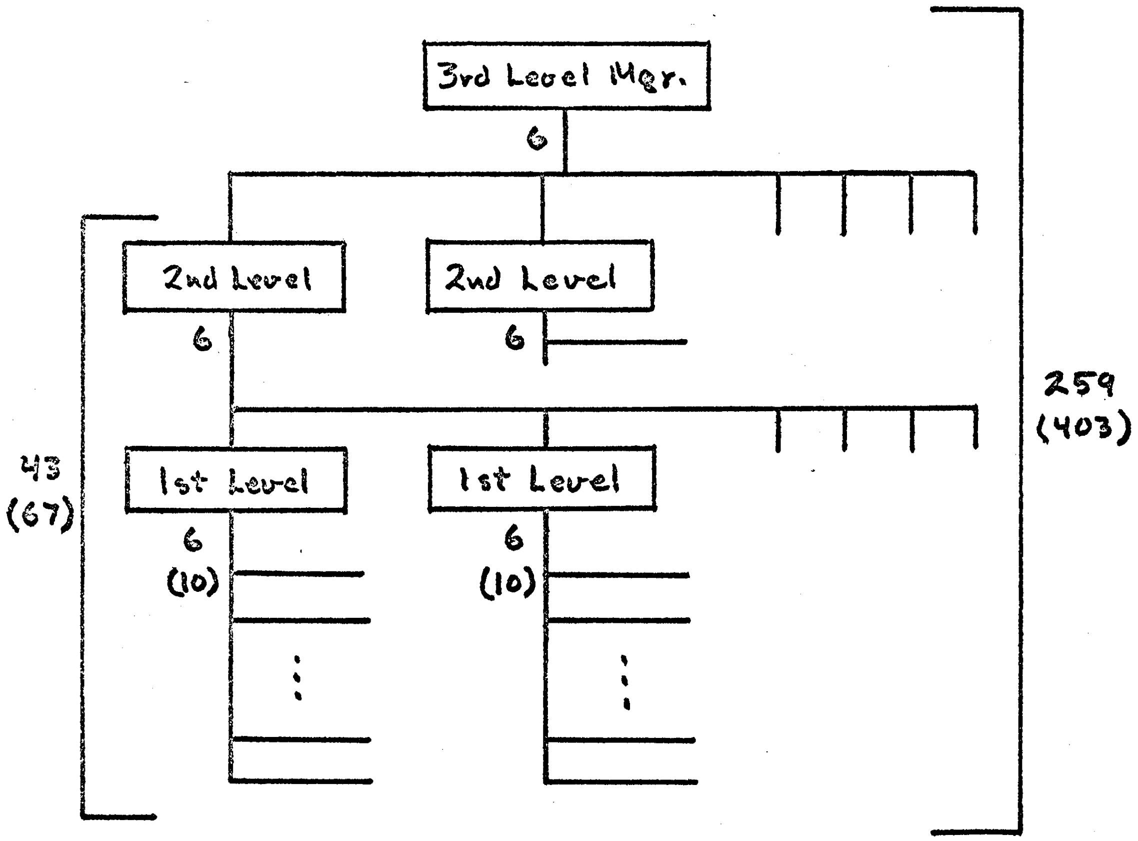


Figure 2.

MANAGEMENT SPAN



If | **Then**

$$\frac{\text{Non-mgrs}}{\text{1st Level mgrs}} = \frac{6}{1} \qquad \frac{\text{Non-mgrs}}{\text{All Mgrs}} = \frac{5}{1}$$

$$\left(\frac{\text{Non-mgrs}}{\text{1st Level mgrs}} = \frac{10}{1}\right) \qquad \left(\frac{\text{Non-mgrs}}{\text{All Mgrs}} = \frac{8}{1}\right)$$

Figure 3.

PROGRAMMERS by EDUCATION LEVEL

	%
Associate of Arts	2
College credit, no degree	12
No college credit	9
	23
BA or BS	58
MA or MS	14
PhD	1
	96

Figure 4.

PROGRAMMERS by FIELD of STUDY

	%
Business	10.5
Education	2
Engineering	15
Fine and Applied Arts	.5
Liberal Arts	8
Math	41
Science	8
	85

Figure 5.

SESSION NUMBER T.2.1.

SPEAKERS

DAN FULLAN (IBM) PRESENTED WM. GARRISON (IBM), MODERATED BY D.R. MC ILVAIN

DISCUSSION

PRESENTATION ON PL/I BY IBM, INDICATING THE INTENT OF PL/I, USAGE & EXPERIENCE TO DATE, AND FUTURE PLANS (INCLUDING ADDITIONAL PUBLICATIONS FOR USER EDUCATION).

FUTURE PLANS FOR PL/I UNDER DOS INCLUDE

1. ADDITIONAL DOCUMENTATION
2. INCLUSION OF INITIAL ATTRIBUTE
3. SCIENTIFIC SUBROUTINES WILL BE AVAILABLE FOR OS PL/I AS TYPE 3 SUPPORT.
4. EXPANSION OF OBJECT TIME DIAGNOSTICS
5. CONSIDERATION OF CREATION OF IS FILE UNDER DOS PL/I.
6. SIFTS FROM COBOL OR FORTRAN ARE BEING CONSIDERED.
7. PL/I OBJECT PROGRAMS WILL BE AVAILABLE FOR OBJECT TIME EXECUTING IN FOREGROUND UNDER RELEASE OF DOS SCHEDULED FOR 4/68.

THE PHILOSOPHY BEHIND PL/I HAS BEEN THAT SUPPORT OF EXTENSIONS IN 3RD GENERATION EQUIPMENT & TECHNOLOGY WOULD BE MADE PREFERENTIALLY INTO PL/I OVER FORTRAN OR COBOL.

SESSION NUMBER T.2.3.

SPEAKERS
    LAURA AUSTIN

DISCUSSION
        VOLUNTEERS WERE SOLICITED FOR SERVICE ON THE THE NOMINATING COMMITTEE, EXECUTIVE BOARD, AND PROJECTS. THERE WERE 10 PEOPLE IN ATTENDANCE. NAMES WILL BE REFERRED TO THE APPROPRIATE PERSONS.

SESSION NUMBER T.2.4.

SPEAKERS

BRIAN SWAIN OF SHAWINIGAN ENGINEERING SPOKE ON THE 1130 SINGLE DISK SORT PROGRAM. W.C. BLACKNEY OF DOW SPOKE ON RUNNING A MEMORYSCOPE THRU THE 1130, 1627 PLOTTER ATTACHMENT.

DISCUSSION

LARRY WHALEN CHAIRED THE MEETING. WE HAVE DECIDED TO ESTABLISH A COMMITTEE TO INVESTIGATE FORTRAN AND MONITOR V2. THIS COMMITTEE WILL BE FORMALLY ORGANIZED AT A LATER DATE.

PMERG - A FAST SORT-MERGE SUBROUTINE FOR IBM 1130

by

B. J. SWAIN

THE SHAWINIGAN ENGINEERING COMPANY LIMITED

SEPTEMBER 1967

139

CORRIGENDUM

The following errors have been observed in the version of this program which was distributed at the Cincinnati COMMON.

1. APPENDIX 3 (Source Listings)

   Subroutine IPTSK, statement 204 + 2

   Remove the following statement:

   ```
   IF(IR)290,207,299
   ```

   and replace by

   ```
   IF(IR)299,207,290
   ```

   The previous version of this statement caused alphabetic sorting to be performed in descending order if KEY(1,I) is positive.

2. APPENDIX 1 - Users Guide

   Page 10.

   The equation for the calculation of the length of IWØRK is incorrectly stated as

   ```
   2*(N*IBLØK+IREP)+10
   ```

   replace this with the equation

   ```
   2*(N*IBLØK+5 ,
   ```

   No program error is associated with this correction.

These corrections have been incorporated in the following text.

TABLE OF CONTENTS

Table of Contents (cont'd)

ABSTRACT

This subprogram sorts a disk file into ascending or descending order as determined by any number of control fields. It is self-adjusting to make use of available storage, and if the sequence length requires, sorts the file by sections, which are subsequently merged. The maximum length is determined by available disk space. Maximum speed is achieved by reduction of manipulation of data on the disk. The subprogram is coded in 1130 Fortran, and is compatible with COMET and IDEAL.

DISCLAIMER

PMERG - A FAST SORT-MERGE SUBROUTINE FOR IBM 1130.

Purpose

An IBM 1130 computer was installed at The Shawinigan Engineering Company Limited offices in Montreal, Canada in January 1967. Although this machine was primarily intended for the solution of engineering problems, it was planned to use it for a considerable range of commercial applications. The principal ones of these were Payroll, Labour Distribution, and Cost Analysis. It was recognized that a fast sort-merge program was required, principally in order to produce sorted reports for these commercial applications. A subprogram was therefore written in Fortran. In order to incorporate the subprogram into several program packages it was made self-adjusting so that the amount of working storage could be allocated to suit the available core. Flexibility as to the number and type of sort keys (control fields) was also required.

Features Incorporated.

The following features, which will be discussed in more detail below were incorporated in order to achieve the aims required of the program

a) Modular Structure

b) A fast sorting Algorithm

c) Extraction of sort keys from the records.

Features Incorporated (cont'd)

d) Record blocking

e) Rejection of unwanted records before sorting.

f) Self adjusting working storage.

Modular Structure.

The modular structure of the subprogram allows it to be incorporated into program packages being written by different programmers. The subprogram itself has an argument list which describes completely the file to be sorted, the number, position and type of the sort keys, the available working core storage and disk work files, and the disposition of the output. Two subprograms are called by the principal subprogram to perform first the sort and then merge operation. Since these are called only once they may be localled against each other, thus freeing additional core space for working storage.

Sorting Algorithm.

The sorting algorithm used is Shells method, which is a successive merging process of sequences which start at 2 or 3 entries in length and are subsequently merged until a single file results. Advantage is taken of any previous ordering of the file, by recognizing in the merging process that when one sequence is exhausted then examination of the other sequence is not required. This algorithm has been found to be superior in speed to algorithms incorporating sorting by exchange, and does not require excessive coding.

Extraction of Keys from Records.

In view of the large access time required for the 2315 disk, it is vital that the amount of information removed and written on the disk is reduced as far as possible. The program was therefore written as a tag-sort. The output from the subroutine is a file of one word records, each entry in the file being a pointer to the record to be processed in the main file in accordance with the required order. Also,the program was built around the concept that the sort keys would be removed from the record on the disk and only these moved in the subsequent processing. The procedure of removing these sort keys from the record also increases the number of records which can be sorted in one phase prior to the merging process. During the sort phase the working storage contains records organized as follows:

Pointer to record in main file

Sort keys

Pointer to indicate position in working storage of high order key.

In the sorting process only the pointer to indicate the position in working storage of the high order key is moved thus increasing the speed of the sorting procedure. At the end of the sorting phase disk records are written consisting of the following

Record number

Sort keys

Extraction of Keys from Records (cont'd)

These subsidiary records are stored on working files and subsequently merged. In the last phase of the merging process the record numbers only are written onto a disk file which then forms the pointer file to the main file.

Record Blocking.

In order to speed the transfer of information from and to the disk a system of record blocking is employed. Blocking subroutines have been written and incorporated into the sort-merge program, to assemble records into blocks before writing them on the disk or after retrieving them from the disk. The number of records contained in one block can be adjusted by the user.

Rejection of Unwanted Records.

It is frequently found that for a particular report only a portion of a file is required. Time can therefore be saved if only those records which are required are sorted and the remainder not processed. The user is permitted to employ this technique by writing a subroutine to determine for each record whether or not it is required in the sort. The name of this subroutine is included in the argument list in the sort-merge subprogram and in an external statement in the main program calling the sort-merge subroutine

<u>Self-Adjusted Working Storage.</u>

The limited core available on the IBM 1130 demands that space allocated to variable storage is somewhat restricted. At the same time, for a sorting program, it is obviously desirable that the maximum use be made of core storage in order to reduce the processing time. In order to achieve a balance between these two mutually exclusive requirements, a feature was included in the program whereby the amount of available working storage was used as a parameter supplied by the user thus permitting the program to calculate the maximum number of records which would be sorted in one phase, the number of phases which would be merged at one cycle and hence the number of merge cycles which would be required in order to complete the merging process. During the sorting process the number of records which can be sorted in one phase is calculated by dividing the available working storage length by the length of the subsidiary record containing of the pointer to the records in the main file, the sort keys and the pointer to indicate the position in working storage of the high order key. During the merging process working storage contains buffers to hold the subsidiary records from the working files of the disk. The number of these which can be contained simultaneously is calculated by dividing the working storage by the length of one buffer. In the demonstration program given in the appendix to this paper, the available

Self-Adjusted Working Storage (cont'd)

working storage is 1,850 words. Each record sorted generates 10 words to be contained in working storage during the sorting process. Thus 185 records are sorted simultaneously, and the total file, from which 811 records have been extracted, is sorted in 5 phases. The read buffers have a length of 325 words, which permits these 5 phases to be merged simultaneously.

Performance.

The following sorting times have been observed:

1) Sorting 10,000 10 word records having 3 integer sort keys - 37 minutes.
2) Sorting 811 32 word records having 5 sort keys, integer, double word integer, real and alphabetic, occupying a total of 8 words - 2½ minutes (This is the demonstration program included in appendix 3 of this paper).

The program has been in regular use sorting reports for commercial applications, the longest of these requires sorting 7,000 records having 6 sort keys occupying 7 words. Sort time for this job is 20 minutes.

Compatibility.

The sort-merge subprogram is compatible with COMET (library program No. 3.0.002) for handling alphabetic arrays, and IDEAL (library program No. 3.0.004) for handling double word integers.

Compatibility (cont'd)

It can be made compatible with the IBM Commercial Subroutine Package by making appropriate changes to the calls to subprograms for comparison of alphabetic arrays, by substituting variable length integers for double-word integers, and by appropriately increasing the storage allocation for sort keys. Due to the increased space required for data storage, when using the Commercial Subroutine Package, in comparison with that required for COMET and IDEAL, the performance of the sort-merge program would be adversely effected.

Conclusions.

A sort-merge subroutine coded in Fortran has been presented. It has been shown that with careful coding it is entirely feasible to perform a disk sort on the IBM 1130 computer. It is possible to sort files containing as many as 10,000 records in a reasonable length of time. This covers the normal requirements of a small computer installation.

Appendix 1 USERS GUIDE

Calling the Sort-merge Subprogram.

PMERG is a subprogram to sort a disk file into ascending or descending order, using sort merge partitioning to reduce the core working storage requirement. The maximum sequence length is determined by the available disk storage only.

The argument list is as follows:

```
CALL PMERG (IFILE, N, J, L, KEYS, IKEY, IWØRK,
            NWØRK, IBUF, IBLØK, IWRK1, IWRK2,
            IFPT, IUSE, KØUNT)
```

IFILE	=	Number of disk file containing information to be sorted. (File should be written completely on the disk before calling PMERG).
N	=	Logical record length in IFILE.
J	=	Position in IFILE of first record to be sorted.
L	=	Number of logical records to be sorted.
KEYS	=	Previously defined array to determine position and type of sort keys in file record. In the calling program, the

Calling the Sort-merge Subprogram (cont'd)

array must appear in DIMENSION statement, with dimension (4,IKEY). For the I'th sort key, the value to be assigned to the elements of KEYS are as follows:

Sort key type	One word integer	Double word integer	Alphabetic	Real
KEYS(1,I)	1 -1 if descend ing order	2 -2 if descend ing order	3 -3 if descend ing order	4 -4 if descend ing order
KEYS(2,I)	Position in record of high-order word of key.			
KEYS(3,I)	Not used		Character posi tion of left most character relative to KEYS(2,I)	Not used
KEYS(4,I)	Not used		No. of cha- racters	Not used

Calling the Sort-merge Subprogram (cont'd)

IKEY	=	Number of sort keys.
IWØRK	=	Working storage array, used to hold sort keys and as a transfer buffer for disk records. Required length is 2*(N*IBLØK+5). IBLØK is defined below. IWØRK should appear in a DIMENSION statement in the calling program, and should be as long as possible to reduce number of merge cycles.
NWØRK	=	Length of IWØRK. Corresponds to dimensioned size in calling program.
IBUF	=	Read buffer for disk records. Required size is N*IBLØK+5. N*IBLØK is physical record size to appear in define file statement for all disk files i.e. IFILE,IWRK1,IWRK2,IFPT. IBUF should appear in a DIMENSION statement in the calling subprogram.
IBLØK	=	Number of logical records blocked to form one physical record on disk.
IWRK1	=	Number of disk file for first working storage. Required number of physical records is (KØUNT-1)/(N*IBLØK/IREP)+1. KØUNT and IREP are defined below.

Calling the Sort-merge Subprogram (cont'd)

IWRK2	=	Number of disk file for second working storage. Required number of records same as IWRK1.
IFPT	=	Number of disk file to contain returned values. Entries in IFPT are record numbers corresponding to required order. Required number of physical record is (KØUNT-1)/(N*IBLØK)+1. Logical record length is 1.
IUSE	=	The name of a function subprogram supplied by the user, having the form FUNCTION IUSE (IREC) whose purpose is to determine if a record is to be included in the sort. IREC is a single subscripted array containing one record. The function returns 1 if this record is to be included, and 2 if this record is not to be included. IUSE is a dummy name. The true name assigned must be included in an EXTERNAL statement in the calling program. Use of the subprogram INTAK included in Appendix 3 will cause all records to be included.

<u>Calling of Sort-merge Subprogram</u> (cont'd)

KØUNT	=	Output variable. The number of records found to be in this sort, when examined by subprogram IUSE.

IREP is logical record length of record containing sort keys and pointer to record to which keys belong.

Allow
- 1 word for pointer
- 1 word for each integer key
- 2 words for each double word integer key
- 1 word for every 2 characters of alphabetic key
- 2 words for each real standard precision key
- 3 words for each real extended precision key.

Requirements for Calling Program.

The calling program must comply with the following requirements:

1) It must use the *ØNE WØRD INTEGERS option.

2) If double word integers are used,it must use standard precision real variables.

3) It must contain an EXTERNAL statement defining the name of the user-writer function subprogram,referred to as IUSE in the argument list of PMERG.

4) It must contain DIMENSION statements, allocating storage for IBUF and IWØRK.

5) It must contain DEFINE FILE statements for the disk files. The physical record length for all four files used by the sort-merge subprogram is the same, being N*IBLØK.

6) It must define the KEYS array.

7) It must define all other arguments in the CALL statement, except KØUNT, which is an output variable.

Use of Record Blocking.

The record blocking subroutines DØPEN, DCLØS, DPUT and DGET are used by this subprogram, and their use for all record handling is recommended.

The following definitions and standards are used for all record blocking subroutines

LOGICAL RECORD - the record stored in the buffer.

Use of Record Blocking (cont'd)

PHYSICAL RECORD - the record stored in the file

PHYSICAL RECORD LENGTH as defined in the DEFINE FILE statement is either:

LOGICAL RECORD LENGTH * Number of Logical Records in Block
or
320

whichever is less.

All the following subroutines use the file buffer IBUF, which contains control words to regulate the reading and writing of records, and also is used for assembling the block of records for transfer to and from the disk. The following is the appearance of the file buffer.

IBUF(1)	=	File number
IBUF(2)	=	Logical record length
IBUF(3)	=	Number of logical records in block
IBUF(4)	=	File type. = 1 when block has only been used for GET operations (i.e. is identical to equivalent records on disk) = 2 when block has been used for PUT operations.
IBUF(5)	=	Number of first logical record in the block now resident in the file buffer. Set to zero if no record has been transferred to the buffer.
IBUF(6) etc.,	=	Blocked records.

Use of Record Blocking (cont'd)

IBUF must appear in the calling program with dimension

LOGICAL RECORD LENGTH * Number of RECORDS IN BLOCK + 5

This should be set as large as core space permits.

The subroutine arguments lists are as follows:

```
CALL DØPEN (IBUF, N, J, L )
```

To open a file buffer. Sets initial values to the file control words. No transfer of information to or from disk takes place.

IBUF	=	Name of file buffer
N	=	File number
J	=	Logical record length
L	=	Number of logical records in block.

```
CALL DGET (IBUF, K, IA)
```

To transfer logical record K of the file to the array IA. If the required record is already resident in the buffer, it is immediately transferred. If it is not, the block of records in the buffer is stored if necessary, and the correct block of records obtained from disk. Transfer then takes place.

IBUF	=	Name of file buffer
K	=	Required logical record
IA	=	Array to contain record obtained.

Note that IA is integer. If real values are required, they can be obtained by use of a suitable EQUIVALENCE statement, in which

Use of Record Blocking (cont'd)

real variables are assigned to EVEN locations in the array IA. The real variables then occupy the designated location, and the next lower location in the array e.g.

```
        DIMENSION IA(40)
        EQUIVALENCE (B,IA(16))
     B occupies  IA(16) and IA(15)
```

Note: The use of an EQUIVALENCE statement in this way is not strictly speaking permitted. However, it works satisfactorily, providing nothing is done to force the addresses of real variables onto uneven word numbers. To prevent this, the following rules should be observed.

1) Equivalence only to even locations in the integer array, IA.

2) The integer array IA should be dimensioned to an even number of words.

3) If IA is in COMMON, then any previous variables in COMMON together occupy an even number of words.

e.g.

```
COMMON IX, IY(2),IZ,IA(20)  }
EQUIVALENCE (IA(2),B)       }     is valid

COMMON IX, IA(20)           }
EQUIVALENCE (IA(2),B)       }     is not valid
```

<u>Use of Record Blocking</u> (cont'd)

CALL DPUT (IBUF, K, IA)

To transfer the array IA to logical record K of the file. Operation is similar to DGET. The contents of IA are transferred only as far as the file buffer, and not written on disk.

CALL DCLØS (IBUF)

To close the file. If a block or records requires transfer to disk, the transfer is made.

CALL DUSE (IBUF, K, M, KEY, KL)

The subroutine is called by DGET and DPUT to perform the disk reading and writing operations, and to generate a pointer to the required record in the file buffer. It is not normally called directly by the user.

<u>Errors:</u>

One error message is included in the sort-merge subroutine PAUSE 3333 indicates that the working storage allocated is too small to permit at least two phases of the file to be merged.

Appendix 2 - PROGRAM STRUCTURE

The following programs are included in the package:

PMERG	- Principal subprogram, calls SRTPH and MRGPH to perform sorting and merging operations respectively.
SRTPH	- Extracts keys from records. Performs sorting operation. Makes up a file containing sort keys and pointers to records included in the sort.
MRGPH	- Performs merging operation. Outputs the file of pointers to records in the main file.
IPTSK	- Function subprogram to compare two sets of sort keys, in order to determine which record should be processed first.
DØPEN	- Initializes a disk buffer for blocking operations.
DGET	- Gets disk records - transfers logical disk records from disk buffer to primary storage.
DPUT	- Puts disk records - transfers logical disk records from primary storage to the disk buffer.
DCLØS	- Terminates a disk writing operation.
DUSE	- Called by DGET and DPUT to transfer blocks of records to and from the disk, as required.
MINØ	- Function subprogram to determine the minimum two variables.

Program Structure (cont'd)

QCØMP) -) QGRAB)	COMET subprograms
DISGN)) INT)) SD)	IDEAL subprograms

In addition, the demonstration program requires the use of the following subprograms.

LARGE -	Subroutine to determine if a record is to be included in the sort.
MDIA -	IDEAL subprogram
QPASS)) QSHUV)	COMET subprograms

Also included

INTAK - Subroutine to include all records in the sort.

Core Storage Requirements.

Storage requirements are as follows:

PMERG	82 words
SRTPH	644 "
MRGPH	662 "
IPTSK	334 "
DØPEN	56 "
DGET	68 "

Core Storage Requirements (cont'd)

DPUT	68	words
DCLØS	132	"
DUSE	232	"
MINØ	50	"
QCØMP	52	"
QGRAB	12	"
DISGN	30	"
INT	22	"
SD	34	"
TOTAL	2,478	"

This figure is reduced if SRTPH and MRGPH are localled.

```
// JOB                                     SWAIN SORT-MERGE PROGRAMS
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE PMERG(IFILE,N,J,L,KEYS,IKEY,IWORK,NWORK,IBUF,IBLOK,
     $IWRK1,IWRK2,IFPT,IUSE,KOUNT)
C     SUBROUTINE TO SORT A DISK FILE, USING SORT-MERGE PARTITIONING,
C     TO REDUCE CORE WORKING STORAGE REQUIREMENT. SEQUENCE LENGTH
C     DETERMINED BY DISK WORKING STORAGE FILE LENGTH
C  CARDS MARKED EXT IN COLS 70-72 REFER TO EXTENDED PRECISION          EXT
C  FLOATING POINT KEYS. THESE STATEMENTS SHOULD BE SUBSTITUTED         EXT
C  FOR THE PRECEDING STANDARD PRECISION STATEMENTS, IF EXTENDED        EXT
C  PRECISION IS REQUIRED.  NOTE THAT HIGH PRECISION INTEGERS           EXT
C  CANNOT BE USED WITH EXTENDED PRECISION.
C     IFILE=NUMBER OF DISK FILE CONTAINING INFORMATION TO BE SORTED
C     N    =LOGICAL RECORD LENGTH IN IFILE
C     J    =POSITION IN IFILE OF FIRST RECORD TO BE SORTED
C     L    =NUMBER OF LOGICAL RECORDS TO BE SORTED
C     KEYS =TABLE TO CONTAIN SORT KEYS, DIMENSIONED (4,IKEY) WHERE
C           IKEY IS NUMBER OF SORT KEYS. ENTRIES IN KEYS, FOR THE
C           I'TH SORT KEY ARE AS FOLLOWS
C           KEYS(1,I) = 1 FOR INTEGER
C                     = 2 FOR DOUBLE WORD INTEGER
C                     = 3 FOR ALPHABETIC
C                     = 4 FOR REAL
C           ENTER KEYS(1,I) NEGATIVE IF SORT IS REQUIRED IN DESCENDING
C           ORDER BY THIS KEY
C           KEYS(2,I) = POSITION IN LOGICAL RECORD OF HIGH-ORDER WORD
C                       OF SORT KEY
C           KEYS(3,I) = POSITION OF LEFTMOST CHARACTER WITH RESPECT
C                       TO KEYS(2,I) - ALPHABETIC KEYS ONLY
C           KEYS(4,I) = NUMBER OF CHARACTERS-ALPHABETIC KEYS ONLY
C     IKEY =NUMBER OF SORT KEYS
C     IWORK=WORKING STORAGE ARRAY. USED TO HOLD SORT KEYS AND AS A
C           READ BUFFER FOR MERGING.  REQUIRED LENGTH
C           IS 2*(N*IBLOK+5).  IBLOK IS DEFINED BELOW
C           IWORK SHOULD BE AS LONG AS POSSIBLE TO REDUCE NUMBER OF
C           MERGE CYCLES
C     NWORK=LENGTH OF IWORK. CORRESPONDS TO DIMENSIONED SIZE IN
C           CALLING PROGRAM
C     IBUF =READ BUFFER FOR DISK RECORDS. REQUIRED SIZE IS N*IBLOK+5
C           N*IBLOK IS PHYSICAL RECORD SIZE TO APPEAR IN DEFINE FILE
C           STATEMENT FOR ALL DISK FILES I.E. IFILE,IWRK1,IWRK2,IFPT
C     IBLOK=NO. OF LOGICAL RECORDS BLOCKED TO FORM ONE PHYSICAL RECORD
C           ON DISK
C     IWRK1=NUMBER OF DISK FILE FOR FIRST WORKING STORAGE. REQUIRED
C           NUMBER OF PHYSICAL RECORDS IS (KOUNT-1)/(N*IBLOK/IREP)+1
C           KOUNT + IREP ARE DEFINED BELOW
C     IWRK2=NUMBER OF DISK FILE FOR SECOND WORKING STORAGE. REQUIRED
C           NUMBER OF RECORDS SAME AS IWRK1
C     IFPT=NUMBER OF DISK FILE TO CONTAIN RETURNED VALUE. ENTRIES IN
C           IFPT ARE RECORD NUMBERS CORRESPONDING TO REQUIRED ORDER
C           REQUIRED NUMBER OF PHYSICAL RECORDS IS (KOUNT-1)/(N*IBLOK)+1
C           LOGICAL RECORD LENGTH IS 1
C     IUSE =FUNCTION SUBPROGRAM SUPPLIED BY USER HAVING THE FORM
C                 FUNCTION IUSE(IREC)
C           WHOSE PURPOSE IS TO DETERMINE IF A RECORD IS TO BE
C           INCLUDED IN THE SORT
C                 IREC IS SINGLE SUBSCRIPTED ARRAY CONTAINING
C                 ONE RECORD
```

```
C                 FUNCTION RETURNS 1 IF THIS RECORD IS TO BE INCLUDED
C                 FUNCTION RETURNS 2 IF THIS RECORD IS NOT INCLUDED
C     IUSE IS A DUMMY NAME. THE TRUE NAME ASSIGNED MUST BE INCLUDED IN
C     AN EXTERNAL STATEMENT IN THE CALLING PROGRAM
      EXTERNAL IUSE
C
C     KOUNT=NO. OF RECORDS FOUND TO BE IN THIS SORT, WHEN EXAMINED
C           IN SUBPROGRAM IUSE
C           IREP IS LOGICAL RECORD LENGTH OF RECORD CONTAINING SORT
C           KEYS AND POINTER TO RECORD TO WHICH KEYS BELONG
C           ALLOW 1 WORD FOR POINTER
C                 1 WORD FOR EACH INTEGER KEY
C                 2 WORDS FOR EACH HIGH PRECISION INTEGER KEY
C                 1 WORD FOR EVERY 2 CHARACTERS OF ALPHABETIC KEY
C                 2 WORDS FOR EACH REAL STANDARD PRECISION KEY
C                 3 WORDS FOR EACH REAL EXTENDED PRECISION KEY
C
      DIMENSION KEYS(4,10),IWORK(100),IBUF(320)
C  FOR INDEXING ONLY. TRUE SIZE IN CALLING PROGRAM
      CALL SRTPH(IFILE,N    ,J,L,KEYS,IKEY,IWORK,NWORK,IBUF,IBLOK,IWRK1,
     $IREP,IBLKK,IUSE,KOUNT)
      CALL MRGPH(N,KEYS,IKEY,IWORK,NWORK,IBUF,IBLOK,IWRK1,IREP,IBLKK,
     $KOUNT,IWRK2,IFPT)
      RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR PMERG
 COMMON      0  VARIABLES      2  PROGRAM     80

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE SRTPH(IFILE,N    ,J,L,KEYS,IKEY,IWORK,NWORK,IBUF,IBLOK,
     $IWRK1,IREP,IBLKK,IUSE,KOUNT)
C  PERFORM SORT PHASE OF SORT-MERGE. OUTPUT IS FILE OF SORT KEYS AND
C  POINTERS TO RECORDS. FOR ARGUMENT DEFINITION, SEE CALLING PROGRAM
C
      DIMENSION KEYS(4,10),IWORK(500),IBUF(325)
C  FOR INDEXING ONLY. TRUE SIZE IN CALLING PROGRAM
C
C  INITIALISE
C  JLAST IS POSITION IN IFILE OF LAST RECORD TO BE SORTED
      JLAST=J+L-1
C  JF IS POINTER TO RECORDS IN IFILE
      JF=J
      KOUNT=0
C  JUMP IS INTERVAL BETWEEN SUCCESSIVE ENTRIES IN IWORK
      JUMP=0
C  INITIALISE THIS PHASE
C  NNOW IS LENGTH OF SEQUENCE IN THIS PHASE
  150 NNOW=0
C  JG IS POINTER TO LAST POSITION IN IWORK ACTUALLY IN USE
      JG=0
C  LOC IS LOCATION OF HIGH-ORDER SORT KEY FOR NEXT RECORD
      LOC=2
      CALL DOPEN(IBUF,IFILE,N,IBLOK)
C  RETURN HERE FOR NEW RECORD
  151 IF(JF-JLAST) 152,152,153
  152 IF(JG+JUMP-NWORK) 154,154,153
C  ANOTHER RECORD IS REQUIRED
  154 CALL DUSE(IBUF,JF,0,1,KL)
      IF(IUSE (IBUF(KL+1))-1) 155,155,156
C  THIS RECORD TO BE INCLUDED IN SORT
C  ENTER RECORD NO IN WORKING STORAGE
  155 JG=JG+1
      IWORK(JG)=JF
C  ENTER KEYS IN WORKING STORAGE
      DO 103 IK=1,IKEY
      KEYTP=KEYS(1,IK)
      IF(KEYTP) 513,514,514
  513 KEYTP=-KEYTP
  514 JB=KL+KEYS(2,IK)
      GO TO(104,105,106,105),KEYTP
C  KEY IS INTEGER
  104 JG=JG+1
      IWORK(JG)=IBUF(JB)
      GO TO 103
C  KEY IS HIGH PRECISION INTEGER OR REAL
  105 DO 111 I=1,2
C 105 DO 111 I=1,3                                                      EXT
      JG=JG+1
      IWORK(JG)=IBUF(JB-1)
C     IWORK(JG)=IBUF(JB-2)                                              EXT
  111 JB=JB+1
      GO TO 103
C KEY IS ALPHABETIC
  106 KK=KEYS(4,IK)
      CALL QPASS(IBUF(JB),KEYS(3,IK),IWORK(JG+1),1,KK)
      JG=JG+(KK+1)/2
  103 CONTINUE
C  FIRST RECORD IN SORT, SET IREP AND JUMP
      IF(JUMP) 158,158,157
  158 IREP=JG
      JUMP=IREP+1
      IBLKK=N*IBLOK/IREP
C  WRITE LOCATION OF HIGH-ORDER SORT KEY AND
C  INCREMENT SORTING SEQUENCE COUNTERS
  157 JG=JG+1
      IWORK(JG)=LOC
      LOC=LOC+JUMP
      NNOW=NNOW+1
C  INCREMENT RECORD COUNTER
  156 JF=JF+1
      GO TO 151
C  SEQUENCE COMPLETE, SORT BY SHELL'S METHOD.  ONLY ENTRIES SHOWING
C  LOCATION OF KEYS(LOC) ARE MOVED
  153 CALL DCLOS(IBUF)
      M=NNOW
   96 IF(M-1) 208,208,98
   98 M=(M+2)/3
      MJ=M*JUMP
      M1J=MJ+JUMP
      DO 99 IJ=M1J,JG,JUMP
      IMJ=IJ-MJ
      I1J=IJ+JUMP
      DO 97 LLJ=JUMP,IMJ,MJ
      JP1J=I1J-LLJ
      JPJ=JP1J-MJ
      LOC=IWORK(JPJ)
      LOC1=IWORK(JP1J)
      IF(IPTSK(KEYS,IKEY,IWORK,LOC,LOC1)-1) 99,99,299
C  ELEMENTS ARE OUT OF ORDER
  299 IWORK(JPJ)=LOC1
   97 IWORK(JP1J)=LOC
   99 CONTINUE
      GO TO 96
C SORT COMPLETE WRITE KEYS ON WORK FILE
  208 CALL DOPEN(IBUF,IWRK1,IREP,IBLKK)
      JPJ=0
      IF(NNOW)301,301,302
  302 DO 510 IPHAS=1,NNOW
      JPJ=JPJ+JUMP
      KOUNT=KOUNT+1
      LOC=IWORK(JPJ)
  510 CALL DPUT(IBUF,KOUNT,IWORK(LOC-1))
  301 CALL DCLOS(IBUF)
C  IF FILE INCOMPLETE, GO BACK TO SORT ANOTHER PHASE
      IF(JF-JLAST) 150,150,159
  159 RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR SRTPH
 COMMON        0  VARIABLES      28   PROGRAM     616

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE MRGPH(N,KEYS,IKEY,IWORK,NWORK,IBUF,IBLOK,IWRK1,IREP,
     $IBLKK,KOUNT,IWRK2,IFPT)
C  PERFORMS MERGE PHASE. INPUT IS FILE OF POINTERS AND SORT KEYS IN
C  IWRK1.  OUTPUT IS FILE OF POINTERS IN IFPT
C
      DIMENSION KEYS(4,10),IWORK(100),IBUF(320)
C  FOR INDEXING ONLY. TRUE SIZE GIVEN IN CALLING PROGRAM
      DIMENSION NREAD(10),NTRAN(10),IREC(10),NRECL(10),ISW(10)
C         TABLE OF CONSTANTS FOR MERGE PHASES,ENTRIES CORRESPOND TO
C     PHASES BEING MERGED
C     NREAD =POINTERS TO START OF READ BUFFERS IN IWORK
C     NTRAN =POINTERS TO START OF TRANSFER BUFFERS IN IWORK
C     IREC =POINTERS TO LOGICAL RECORD NOW BEING PROCESSED
C     NRECL =NUMBERS OF LAST RECORD IN PHASES NOW BEING PROCESSED
C     ISW   =PROCESSING SWITCH FOR PHASES
C                =1 WHEN FURTHER RECORDS EXIST TO BE PROCESSED
C                =2 WHEN PHASE IS EXHAUSTED
C  CALCULATE CONSTANTS REQUIRED FOR MERGING
C  NPHAS IS NUMBER OF LOGICAL RECORDS TO BE SORTED IN ONE PHASE
      IF(KOUNT)622,622,301
  301 NPHAS=NWORK/(IREP+1)
C  NPH IS REQUIRED NUMBER OF SORT PHASES
      NPH=(KOUNT-1)/NPHAS+1
      NBUF=IREP*IBLKK+5
C  NBUF IS LENGTH OF READ BUFFERS IN WORKING STORAGE
      NWAYS=MINO(NWORK/NBUF,10)
C  NWAYS IS NO OF WAYS SORT PHASES ARE MERGED
      NRR=1
      DO 512 ITEST=1,NWAYS
      NREAD(ITEST)=NRR
  512 NRR=NRR+NBUF
C  TEST LENGTH OF WORKING STORAGE
      IF(NWAYS-1) 505,505,506
C  WORKING STORAGE TOO SMALL
  505 PAUSE 3333
      CALL EXIT
C  INITIALISE FOR FIRST MERGE CYCLE
C  IFLIP IS SWITCH TO DETERMINE WHICH WORK FILE IS TO BE READ
  506 IFLIP=1
C  RETURN HERE FOR NEW MERGE CYCLE. NOWPH IS PHASE BEING PROCESSED
  601 NOWPH=0
      JNOW=1
      GO TO(603,604),IFLIP
  603 IGET =IWRK1
      IPUT =IWRK2
      GO TO 625
  604 IGET= IWRK2
      IPUT= IWRK1
  625 DO 609 ITEST=1,NWAYS
      II=NREAD(ITEST)
  609 CALL DOPEN(IWORK(II),IGET,IREP,IBLKK)
C  TEST FOR LAST MERGE CYCLE. IF SO
C  OUTPUT FILE OF POINTERS TO IFILE
      IF(NPH-NWAYS) 620,620,621
  621 CALL DOPEN(IBUF,IPUT,IREP,IBLKK)
      GO TO 605
  620 CALL DOPEN(IBUF,IFPT,1,N*IBLOK)
```

169

THE

```
C  RETURN HERE FOR NEW MERGE PHASE. SET CONSTANTS
  605 IHIGH=0
C  IHIGH IS NO. OF WAYS THIS PHASE IS TO BE MERGED
  606 IHIGH=IHIGH+1
      IBASE=NOWPH*NPHAS
      I=IBASE+1
      IREC(IHIGH)=I
      ISW(IHIGH)=1
      II=NREAD(IHIGH)
      CALL DUSE(IWORK(II),I,1,1,KL)
      NTRAN(IHIGH)=KL+II
C  ADVANCE TO NEXT PHASE, AND TEST FOR END OF CYCLE OR FILE
      NOWPH=NOWPH+1
      IF(KOUNT-IBASE-NPHAS) 633,633,634
C  NOT END OF FILE
  634 NRECL(IHIGH)=IBASE+NPHAS
      IF(IHIGH-NWAYS) 606,635,635
C  END OF FILE
  633 NRECL(IHIGH)=KOUNT
  635 CONTINUE
C  RETURN HERE FOR NEW MERGE GROUP
  629 ITAKE=0
      DO 618 ITEST=1,IHIGH
C  COMPARE KEYS, IF EITHER PHASE IS EXHAUSTED, THEN OTHER RECORD IS
C  USED. ITAKE IS BUFFER NO. IN WHICH RECORD TO BE PROCESSED IS FOUND
      IF(ITAKE) 623,613,623
  623 IF(ISW(ITEST)-2) 615,618,615
  613 IF(ISW(ITEST)-2) 616,618,616
  616 ITAKE=ITEST
      GO TO 618
  615 IF(IPTSK(KEYS,IKEY,IWORK,NTRAN(ITAKE),NTRAN(ITEST))-2)618,616,616
  618 CONTINUE
      IF(ITAKE) 617,617,626
C  TRANSFER RECORD TO FORM NEW FILE
  626 IJ=NTRAN(ITAKE)-1
      CALL DPUT(IBUF,JNOW,IWORK(IJ))
      JNOW=JNOW+1
C  GET NEW RECORD
      I=IREC(ITAKE)+1
      IREC(ITAKE)=I
      IF(I-NRECL(ITAKE)) 630,630,631
  630 II=NREAD(ITAKE)
      CALL DUSE(IWORK(II),I,1,1,KL)
      NTRAN(ITAKE)=KL+NREAD(ITAKE)
      GO TO 629
C  PHASE IS EXHAUSTED
  631 ISW(ITAKE)=2
      GO TO 629
C  THIS PHASE COMPLETE. TEST IF LAST PHASE
  617 IF(NOWPH-NPH) 605,624,624
C  THIS MERGE CYCLE COMPLETE. SET CONSTANTS FOR NEXT CYCLE
  624 NPHAS=NPHAS*NWAYS
      NPH=(NPH+NWAYS-1)/NWAYS
      IFLIP=2/IFLIP
      CALL DCLOS(IBUF)
C  TEST FOR LAST CYCLE
      IF(NPH-1) 622,622,601
  622 RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR MRGPH
 COMMON       0  VARIABLES     72  PROGRAM    590

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      FUNCTION IPTSK(KEYS,IKEY,IGHST,KG1,KG2)
C     KEYS=ARRAY CONTAINING DEFINITION OF SORT KEYS
C     IKEY= NO OF KEYS
C     IGHST=ARRAY CONTAINING STRING OF SORT KEYS
C     KG1  =POINTER TO MAJOR SORT KEY,FIRST RECORD
C     KG2  =POINTER TO MAJOR SORT KEY,SECOND RECORD
C     OUTPUT IS 1 IF FIRST RECORD IS TO BE SELECTED
C               2 IF SECOND RECORD IS TO BE SELECTED
C     IF KEYS ARE IDENTICAL, THEN ORIGINAL POSITION IS TESTED TO
C     DETERMINE OUTPUT
      DIMENSION KEYS (4,10),IGHST(500)
      DIMENSION IP(2),IQ(2)
C     DIMENSION IP(4),IQ(4)                                         EXT
      EQUIVALENCE (P,IP(2)),(Q,IQ(2))
C     EQUIVALENCE(P,IP(3)),(Q,IQ(3))                                EXT
      JG=KG1
      JG1=KG2
      DO 201 IK=1,IKEY
      KEYIK=KEYS(1,IK)
      KEYTP=KEYIK
      IF(KEYIK) 215,216,216
  215 KEYTP=-KEYTP
  216 GO TO(202,203,204,203),KEYTP
C         COMPARE INTEGER KEYS
  202 IGJG=IGHST(JG)
      IGJG1=IGHST(JG1)
      IF(IGJG) 210,211,211
  210 IF(IGJG1) 212,290,290
  211 IF(IGJG1) 299,212,212
  212 IF(IGJG-IGJG1) 290,205,299
  205 JG=JG+1
      JG1=JG1+1
      GO TO 201
C         COMPARE HIGH PRECISION KEYS
  203 DO 209IX=1,2
C 203 DO 209 IX=1,3                                                 EXT
      IP(IX)=IGHST(JG)
      IQ(IX)=IGHST(JG1)
      JG=JG+1
  209 JG1=JG1+1
      IF(KEYTP- 2) 214,214,213
  214 CALL SD(R,P,Q)
      IF(INT(R)) 290,201,299
  213 IF(P-Q) 290,201,299
C         COMPARE ALPHABETIC KEYS
  204 KL=KEYS(4,IK)
      CALL QCOMP(IGHST(JG),1,IGHST(JG1),1,KL,IR)
      IF(IR) 299,207,290
  207 JGINC=(KL+1)/2
      JG=JG+JGINC
      JG1=JG1+JGINC
  201 CONTINUE
C     KEYS ARE IDENTICAL
      IF(KG1-KG2) 291,291,292
  290 IF(KEYIK) 292,291,291
  291 IPTSK=1
      RETURN
```

```
                                                                PAGE 02
  299 IF(KEYIK) 291,292,292
  292 IPTSK=2
      RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR IPTSK
 COMMON       0  VARIABLES      22  PROGRAM    312

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE DOPEN(IBUF,N,J,L)
C         THIS SUBROUTINE OPENS A FILE BUFFER,SETTING INITIAL VALUES  TO
C         THE FILE CONTROL WORDS.
C
      DIMENSION IBUF(325)
C         IBUF = NAME OF FILE BUFFER
C         N    = FILE NUMBER
C         J    = CORE RECORD LENGTH
C         L    = NUMBER OF RECORDS IN BLOCK
C
      IBUF(1) = N
      IBUF(2) = J
      IBUF(3) = L
C
C         IBUF(4) = FILE TYPE
C                 =1 WHEN BLOCK HAS ONLY BEEN USED FOR TRANSFERRING A
C                    RECORD  FROM IBUF TO AN ARRAY(FOR GET OPERATIONS)
C                 =2 WHEN BLOCK HAS BEEN USED TO TRANSFER A RECORD FROM
C                    AN ARRAY TO IBUF(FOR PUT OPERATIONS)
C                 SET = 1 INITIALLY
C
      IBUF(4) = 1
C         IBUF(5) = NUMBER OF FIRST RECORD IN BLOCK
C
C                 SET = 0 INITIALLY
C
      IBUF(5) = 0
      RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR DOPEN
 COMMON       0  VARIABLES       6  PROGRAM      50

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE DGET(IBUF,K,IA)
C THIS SUBROUTINE TRANSFERS RECORD K OF A FILE TO ARRAY IA.
C IF THE REQUIRED RECORD IS ALREADY RESIDENT IN THE BUFFER, IT IS
C IMMEDIATELY TRANSFERRED
C IF IT IS NOT IN THE BUFFER, THE BLOCK OF RECORDS IN THE BUFFER
C IS STORED IF NECESSARY, AND THE CORRECT BLOCK OF RECORDS
C OBTAINED FROM THE DISK, AFTER WHICH TRANSFER OF THE RECORD TAKES PLACE
C
      DIMENSION IBUF(325),IA(100)
C     IBUF = NAME OF FILE BUFFER
C     K    = RECORD TO CONTAIN ARRAY IA
C     IA   = REQUIRED ARRAY
C
      CALL DUSE(IBUF,K,1,1,KL)
C HERE TO TRANSFER RECORD FROM IBUF
      J= IBUF(2)
      DO 5 JJ = 1,J
      IA(JJ) = IBUF(KL)
    5 KL = KL + 1
      RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR DGET
 COMMON      0  VARIABLES      6  PROGRAM     62

END OF COMPILATION
```

THE

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE DPUT(IBUF,K,IA)
C THIS SUBROUTINE TRANSFERS AN ARRAY IA TO RECORD K OF A FILE.
C IF THE REQUIRED BLOCK IS ALREADY RESIDENT IN THE BUFFER, THE ARRAY
C IS  IMMEDIATELY TRANSFERRED
C IF IT IS NOT IN THE BUFFER, THE BLOCK IN THE BUFFER IS STORED
C IF NECESSARY, AND THE REQUIRED BLOCK OBTAINED FROM THE DISK,
C AFTER WHICH TRANSFER OF THE ARRAY TAKES PLACE.
C THE CONTENTS OF IA ARE NOT WRITTEN ON THE DISK.
C
      DIMENSION IBUF(325),IA(100)
C     IBUF = NAME OF FILE BUFFER
C     K    = RECORD TO CONTAIN ARRAY IA
C     IA   = REQUIRED ARRAY
C
      CALL DUSE(IBUF,K,1,2,KL)
C HERE TO TRANSFER RECORD FROM IA
      J= IBUF(2)
      DO 5 JJ = 1,J
      IBUF(KL) = IA(JJ)
    5 KL = KL + 1
      RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR DPUT
 COMMON      0  VARIABLES      6  PROGRAM     62

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE DCLOS(IBUF)
C
C        THIS SUBROUTINE CLOSES THE FILE
C        IF A BLOCK OF RECORDS REQUIRES TRANSFER TO THE DISK, THE TRANSFER
C        IS MADE.
      DIMENSION IBUF(325)
C
C        IBUF = NAME OF FILE BUFFER
C
      KK = IBUF(4)
      GO TO (3,4),KK
    3 RETURN
    4 N = IBUF(1)
      LL = (IBUF(2)*IBUF(3)) + 5
C
C        LL = NUMBER OF LAST WORD IN BLOCK
C
      K1 = IBUF(5)
      L = MIN0(320,(LL-5))
C
C         L     = FILE RECORD LENGTH
C
      NREC = (LL + L - 6)/L
C
C         NREC = NO OF FILE RECORDS IN A BLOCK
C
      K3 =((K1/IBUF(3))*NREC) + 1
C
C         K3    = RECORD NUMBER OF THE FIRST FILE RECORD IN THE BLOCK
C                 CONTAINING RECORD K
C
      WRITE(N'K3)(IBUF(I),I =6,LL)
      RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR DCLOS
 COMMON      0  VARIABLES     16  PROGRAM    116

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      SUBROUTINE DUSE(IBUF,K,M,KEY,KL)
      DIMENSION IBUF(325)
C         THIS SUBROUTINE GENERATES A POINTER TO ONE WORD OF A DISK
C         RECORD, SO THAT WORD CAN BE REFERENCED IN A SUBSEQUENT
C         OPERATION
C         IF THE BLOCK CONTAINING THE REQUIRED RECORD IS ALREADY IN
C         THE BUFFER, THE POINTER IS GENERATED IMMEDIATELY
C         IF IT IS NOT IN THE BUFFER, THE BLOCK IN THE BUFFER IS
C         STORED IF NECESSARY, AND THE REQUIRED BLOCK OBTAINED
C         FROM  THE DISK, AFTER WHICH GENERATION OF THE POINTER TAKES
C         PLACE
C         IBUF =NAME OF FILE BUFFER
C         K    =RECORD REQUIRED
C         M    =WORD REQUIRED
C         KEY  =SWITCH TO INDICATE IF SUBSEQUENT USE OF POINTER WILL
C               CAUSE RECORD TO BE CHANGED. IF SO, KEY=2. IF NOT, KEY=1
C         KL   =POINTER TO WORD REQUIRED
C     FIRST 5 WORDS OF IBUF ARE CONTROL WORDS
C         IBUF(1)= FILE NUMBER  (N)
C         IBUF(2)= LOGICAL RECORD LENGTH (J)
C         IBUF(3)= NO. OF LOGICAL RECORDS IN A BLOCK (L)
C         IBUF(4)= FILE TYPE DEFINITION =1 INITIALLY
C                                       =2 WHEN BUFFER HAS BEEN MODIFIED
C                   BY PUT OPERATIONS, AND HENCE IS DIFFERENT FROM DISK
C         IBUF(5)= NO. OF FIRST RECORD IN BLOCK =0 IF BUFFER HAS NOT
C                   BEEN FILLED
      L=IBUF(3)
      K1= ((K-1)/L)*L+1
C         K1  =RECORD NO. OF FIRST LOGICAL RECORD IN THE BLOCK CONTAIN-
C              ING RECORD K
      J=IBUF(2)
      K2=IBUF(5)
      IF(K1-K2) 1,2,1
    1 N=IBUF(1)
      L2=J*L
C         L2  =NO. OF WORDS IN BLOCK
      L1=MINO(320,L2)
C        L1   =PHYSICAL RECORD LENGTH
      NREC=(L2+L1-1)/L1
C         NREC=NO OF LOGICAL RECORDS IN PHYSICAL RECORD
C              -SAME AS BLOCKING FACTOR IF PHYSICAL RECORD LENGTH.LE.320
      LL=L2+5
      KK=IBUF(4)
C  INSERT THE FOLLOWING STATEMENTS IF A TRACE ON SW 15 IS REQUIRED
C  THESE CAUSE THE VALUES OF ALL PARAMETERS TO BE PRINTED WHENEVER
C  A PHYSICAL RECORD IS TRANSFERRED TO OR FROM DISK
C     CALL DATSW(15,JJJ)
C     IF(JJJ-1) 998,997,998
C 997 WRITE(3,999)(IBUF(III),III=1,5),K,M,KEY
C 999 FORMAT(' ',10I5)
C 998 CONTINUE
      GO TO(3,4),KK
C
C         HERE TO STORE BLOCK ON DISK
C
    4 K3= (K2/L)*NREC+1
C         K3  =RECORD NO. OF THE FIRST LOGICAL RECORD IN THE BLOCK
```

```
C                   CONTAINING RECORD K
      WRITE(N'K3)(IBUF(I),I=6,LL)
C
C              HERE TO READ NEW DISK BLOCK
    3 K3= (K1/L)*NREC+1
      READ(N'K3)(IBUF(I),I=6,LL)
      IBUF(5)=K1
C              REQUIRED DISK BLOCK IS IN BUFFER
    2 KL =(K-K1)*J+M+5
      IF(KEY-1) 6,6,5
C              HERE FOR PUT OPERATIONS
    5 IBUF(4)=2
    6 RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR DUSE
 COMMON       0  VARIABLES     16   PROGRAM    216

END OF COMPILATION
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      FUNCTION MINO(I,J)
C
C           FUNCTION SUBPROGRAM TO CHOOSE THE SMALLEST VALUE
C           OF TWO INTEGERS.
C
      IF(I)210,211,211
  210 IF(J)212,290,290
  211 IF(J)299,212,212
  212 IF(I-J)290,290,299
  290 MINO = I
      RETURN
  299 MINO = J
      RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR MINO
 COMMON       0  VARIABLES       2  PROGRAM      48

END OF COMPILATION
```

```
// JOB                                    SWAIN SORT-MERGE LISTING
// * COMET AND IDEAL SUBROUTINES USED BY SORT-MERGE PROGRAM
// ASM
*LIST SOURCE PROGRAM
                          *     THIS IS QCOMP                                    COMP0010
                          *                                                      COMP0020
0000     180D6517               ENT      QCOMP                                   COMP0030
0000     0001             QCOMP BSS      1                                       COMP0040
0001 0   6A2B                   STX    2 SAVE&1     SAVE XR2                     COMP0050
0002 0   282B                   STS      SAVES      SAVE STATUS                  COMP0060
0003 01  66800000               LDX   I2 QCOMP      LOAD XR2 WITH CALL&1         COMP0070
0005 0   C205                   LD    X2 5          LOAD ACC WITH RET PARA ADD   COMP0080
0006 0   D024                   STO      STOR1&1    STORE IN  STORE PARA  INST   COMP0090
0007 0   C200                   LD    X2 0          LOAD SC AREA ADD TO ACC      COMP0100
0008 0   1001                   SLA   X  1          SHIFT LEFT ONE BIT           COMP0110
0009 00  96800001               S     I2 1          SUB COLUMN NO. SC. CHAR      COMP0120
000B 0   800C                   A        TWO        ADD TWO                      COMP0130
000C 0   D024                   STO      SCRAD      STORE INTO SCRAD             COMP0140
000D 00  C6800004               LD    I2 4          LOAD CHARACTER COUNT         COMP0150
000F 0   D022                   STO      CHCNT      STORE INTO CHCNT             COMP0160
0010 0   C202                   LD    X2 2          LOAD SECOND AREA ADDRESS     COMP0170
0011 0   1001                   SLA   X  1          SHIFT LEFT ONE BIT           COMP0180
0012 00  96800003               S     I2 3          SUB COL NO FROM ACC          COMP0190
0014 0   8003                   A        TWO        ADD TWO                      COMP0200
0015 0   7206                   MDX   X2 6          ADD 5 TO XR2                 COMP0210
0016 0   6A19                   STX    2 RETRN&1    STORE XR2 IN RET INSTR.      COMP0220
0017 00  D4000002         INST  STO   L  2          STORE ACC INTO XR2           COMP0230
0019 0   C017             LOOP  LD       SCRAD      LOAD SCRAD TO ACC            COMP0240
001A 30  181D9042               CALL     QGRAB      GET FIELD ONE CHARACTER      COMP0250
001C 0   D016                   STO      ATEMP      STORE CH INTO ATEMP          COMP0260
001D 00  C4000002               LD    L  /0002      LOAD XR2 TO ACC              COMP0270
001F 30  181D9042               CALL     QGRAB      GET FIELD TWO CHAR           COMP0280
0021 0   9011                   S        ATEMP      SUBTRACT ATEMP FROM ACC      COMP0290
0022 01  4C20002A               BSC   L  STOR1,Z    BRANCH IF NOT ZERO           COMP0300
0024 01  74FF0031               MDX   L  SCRAD,-1   DECREMENT SCRAD              COMP0310
0026 0   72FF                   MDX   X2 -1         DECREMENT XR2                COMP0320
0027 01  74FF0032               MDX   L  CHCNT,-1   DECREMENT CHCNT              COMP0330
0029 0   70EF                   MDX      LOOP       NOT ZERO, GO TO LOOP         COMP0340
002A 01  D400002C         STOR1 STO   L  *          STORE IN RETURN VARIABLE     COMP0350
002C 01  6600002E         SAVE  LDX   L2 *          RESTORE XR2                  COMP0360
002E 0   2000             SAVES LDS   X  0          RESTORE STATUS               COMP0370
002F 01  4C000031         RETRN BSC   L  *          RETURN                       COMP0380
0031 1   0032             SCRAD DC       *                                       COMP0390
0032 1   0033             CHCNT DC       *                                       COMP0400
0033 1   0034             ATEMP DC       *                                       COMP0410
0018                      TWO   EQU      INST&1                                  COMP0420
0034                            END                                              COMP0430

     NO ERRORS IN ABOVE ASSEMBLY.
```

THE

```
// ASM
*LIST SOURCE PROGRAM
                              *      THIS IS QGRAB                                         GRAB0010
                              *                                                            GRAB0020
0000      181D9042                   ENT      QGRAB                                        GRAB0030
0000      0001                QGRAB BSS       1                                            GRAB0040
0001 0    1881                      SRT    X  1           DIVIDE BY 2                      GRAB0050
0002 0    D002                      STO       LOAD&1      STORE WORD ADD INTO LOAD         GRAB0060
0003 0    1091                      SLT    X  17          PUT REMAINDER INTO CARRY         GRAB0070
0004 01   C4000006            LOAD  LD     L  *           LOAD WORK TO ACC                 GRAB0080
0006 0    4802                      BSC       C           GO TO MASK INST IF CARRY O GRAB0090
0007 0    1808                      SRA    X  8           RIGHT JUSTIFY CHARACTER          GRAB0100
0008 0    E002                      AND       MASK        MASK EXTRA BITS                  GRAB0110
0009 01   4C800000                  BSC    I  QGRAB       RETURN                           GRAB0120
000B 0    00FF                MASK  DC        /00FF                                        GRAB0130
000C                                END                                                    GRAB0140

     NO ERRORS IN ABOVE ASSEMBLY.
```

THE

```
// ASM                                                                                        IDEAL961
*LIST                                                                                         IDEAL962
0000     042621D5           ENT      DISGN                                                    IDEAL963
                      *CALL DISGN(DBL INT,ISIGN)                                              IDEAL964
                      *WHERE ISIGN IS RETURNED -1,0,=1 DEPENDING UPON THE                     IDEAL965
                      *DBL INT BEING RESPECTIVLY NEGATIVE,ZERO,POSITIVE                       IDEAL966
0000     0001         DISGN BSS       1          RETURN  CALL & 1 HERE                        IDEAL967
0001 0   6915               STX     1 INDX1&1                                                 IDEAL968
0002 01  65800000           LDX    I1 DISGN                                                   IDEAL969
0004 00  CD800000           LDD    I1 0          GET DOUBLE INTEGER                           IDEAL970
0006 01  4C280011           BSC    L  NEG,Z&     BRANCH IF NEG ACCUM                          IDEAL971
0008 01  4C08000C           BSC    L  ZACC,&     BRANCH IF ZERO ACCUM                         IDEAL972
000A 0   C010         POS   LD        PONE       LOAD ISIGN WITH PLUS ONE                     IDEAL973
000B 0   7006               MDX       STORE                                                   IDEAL974
000C 0   1090         ZACC  SLT       16         SHIFT EXT TO ACC                             IDEAL975
000D 0   4820               BSC       Z          SKIP IF ZERO                                 IDEAL976
000E 0   70FB               MDX       POS                                                     IDEAL977
000F 0   C00C               LD        ZERO                                                    IDEAL978
0010 0   7001               MDX       STORE                                                   IDEAL979
0011 0   C008         NEG   LD        NONE                                                    IDEAL980
0012 00  D5800001     STORE STO    I1 1                                                       IDEAL981
0014 0   7102               MDX     1 2                                                       IDEAL982
0015 0   6903               STX     1 BACK&1     LOAD BRANCH BACK INSTRUCT                    IDEAL983
0016 00  65000000     INDX1 LDX    L1 *-*        RESTORE INDEX ONE                            IDEAL984
0018 00  4C000000     BACK  BSC    L  *-*        BRANCH BACK TO MAIN PROG                     IDEAL985
001A 0   FFFF         NONE  DC        -1                                                      IDEAL986
001B 0   0001         PONE  DC        +1                                                      IDEAL987
001C 0   0000         ZERO  DC        0                                                       IDEAL988
001E                        END                                                               IDEAL989

      NO ERRORS IN ABOVE ASSEMBLY.
```

```
// FOR                                                                   IDEAL993
*ONE-WORD INTEGERS                                                       IDEAL994
*LIST ALL                                                                IDEAL995
      FUNCTION INT(DBLIN)                                                IDEAL996
C INT IS A FUNCTION WHICH TESTS THE SIGN OF A DBL INT WITH AN IF STATE  IDEAL997
C -1 IF MINUS,0 IF 0,=1 IF POSITIVE IS RETURNED.                         IDEAL998
C FUNCTION INT CALLS DISGN                                               IDEAL999
C SAMPLE   IF(INT(DIONE))NEG STATEMENT,ZERO STATEMENT,POS STATEMENT NUMBIDEAL00&
      CALL DISGN(DBLIN,ISIGN)                                            IDEAL00A
      INT=ISIGN                                                          IDEAL00B
      RETURN                                                             IDEAL00C
      END                                                                IDEAL00D

VARIABLE ALLOCATIONS
 INT   =0000   ISIGN=0002

CALLED SUBPROGRAMS
 DISGN   SUBIN

CORE REQUIREMENTS FOR INT
 COMMON       0  VARIABLES      4   PROGRAM      18

END OF COMPILATION
```

184

```
// ASM                                                                          IDEAL769
*LIST                                                                           IDEAL770
0000    22100000             ENT      SD         SUBROUTINE NAME                IDEAL771
                        *CALL SD   (A,B,C) WHERE A=B-C                          IDEAL772
                        *DOUBLE INTEGERS HAVE STD PREC REAL VARIABLE NAMES      IDEAL773
                        *IDEAL 1130 FORTRAN ERROR CODE IS /DEAF IN ACCUM.       IDEAL774
                        *TO DISPLAY STATEMENT ALLOCATION ADDR IN ERROR, HIT     IDEAL775
                        *START,ACCUM HAS FORTRAN STATEMENT ALLOCATION ADDR.     IDEAL776
                        *HIT START TO CONTINUE.OUTPUT IS SET TO ZERO.           IDEAL777
0000    0001            SD   BSS      1          SUBROUTINE ENTRY POINT         IDEAL778
0001 0  690F                 STX    1 INDX1&1                                   IDEAL779
0002 0  280F                 STS      STATS                                     IDEAL780
0003 0  2000                 LDS      0          INITIALIZE OVERFLOW            IDEAL781
0004 01 65800000             LDX   I1 SD         CALL&1 ADDR IN INDEX ONE       IDEAL782
0006 00 CD800001             LDD   I1 1          LOAD B OF A B-C                IDEAL783
0008 00 9D800002             SD    I1 2          SUBTRACT C OF A B-C            IDEAL784
000A 0  4801                 BSC      0          SKIP IF OVERFLOW IS OFF        IDEAL785
000B 0  7009                 MDX      TOBIG      GO TO IDEAL ERROR DISPLAY      IDEAL786
000C 00 DD800000        OUT  STD   I1 0          MOVE TO A OF A=B-C             IDEAL787
000E 0  7103                 MDX    1 3                                         IDEAL788
000F 0  6904                 STX    1 BACK&1                                    IDEAL789
0010 00 65000000        INDX1 LDX  L1 *-*                                       IDEAL790
0012 0  2000            STATS LDS     0                                         IDEAL791
0013 00 4C000000        BACK BSC   L  *-*                                       IDEAL792
0015 00 74000032        TOBIG MDX  L  50,0                                      IDEAL793
0017 0  70FD                 MDX      TOBIG      INTERRUPT SERVICE LOOP         IDEAL794
0018 0  C006                 LD       HDEAF      IDEAL FORTRAN ERROR CODE       IDEAL795
0019 0  3000                 WAIT                DISPLAY ERROR CODE IN ACC      IDEAL796
001A 0  C0E5                 LD       SD         LOAD ENTRY ADDR,SUB ORG+2,     IDEAL797
001B 0  9004                 S        H01C4      AND DISPLAY STATEMENT          IDEAL798
001C 0  3000                 WAIT                ALLOCATION ADDR IN ACCUM.      IDEAL799
001D 0  10A0                 SLT      32         CLEAR TO OUTPUT ZERO VALUE     IDEAL800
001E 0  70ED                 MDX      OUT                                       IDEAL801
001F 0  DEAF            HDEAF DC      /DEAF      IDEAL FORTRAN ERROR CODE       IDEAL802
0020 0  01C4            H01C4 DC      /01C4      DISKZ ORGIN +2                 IDEAL803
0022                         END                                                IDEAL804

     NO ERRORS IN ABOVE ASSEMBLY.
```

```
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      FUNCTION LARGE(IREC)
C  THIS IS A SAMPLE SUBPROGRAM TO SATISFY THE CALL TO DUMMY FUNCTION
C  IUSE IN SUBROUTINE MRGPH
C  THE PURPOSE OF THIS FUNCTION SUBPROGRAM IS TO DETERMINE IF A
C  RECORD IS TO BE CONTAINED IN A SORT, ACCORDING TO RULES ESTABLISHED
C  BY THE USER. IN THIS EXAMPLE, A RECORD WILL BE OMITTED IF WORD 10
C  IS GREATER THAN 5
C
C  IREC IS ARRAY CONTAINING RECORD
C  FUNCTION RETURNS 1 IF RECORD IS TO BE INCLUDED, 2 IF OMITTED
      DIMENSION IREC(32)
      IF(IREC(10)-5    ) 1,1,2
    1 LARGE=1
      RETURN
    2 LARGE=2
      RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR LARGE
 COMMON       0  VARIABLES       2   PROGRAM      32

END OF COMPILATION
```

```
// * COMET AND IDEAL SUBROUTINES USED BY DEMONSTRATION PROGRAM
// ASM                                                                   IDEAL001
*LIST                                                                    IDEAL002
002D     14044240            ENT      MADI        SUBROUTINE NAME        IDEAL003
                       *CALL MADI (INPUT ARRAY,FLD START,FLD END,DBL INT)  IDEAL004
                       *CALL MADI MOVES AND CONVERTS A1 TO DBL WD INTEGER  IDEAL005
                       *AN 11 ZONE OVER RT MOST DIGIT NEGATIVE CONVERSION  IDEAL006
                       *ALL ZONES EXCEPT RT MOST ARE STRIPPED OFF.         IDEAL007
                       *DOUBLE INTEGERS HAVE STD PREC REAL VARIABLE NAMES  IDEAL008
                       *                                                   IDEAL009
0000     14062240            ENT      MASI        SUBROUTINE NAME        IDEAL010
                       *CALL MASI (INPUT ARRAY,FLD START,FLD END,INTEGER)  IDEAL011
                       *CALL MASI MOVES AND CONVERTS A1 TO SIG WD INTEGER  IDEAL012
                       *AN 11 ZONE OVER RT MOST DIGIT NEGATIVE CONVERSION  IDEAL013
                       *ALL ZONES EXCEPT RT MOST ARE STRIPPED OFF.         IDEAL014
                       *                                                   IDEAL015
009E     14109040            ENT      MDIA        SUBROUTINE NAME        IDEAL016
                       *CALL MDIA (OUT ARRAY,START,END,DBL INT,EDIT CNTRL) IDEAL017
                       *CALL MDIA MOVES AND CONVERTS DBL WD INT TO A1      IDEAL018
                       *EDIT CNTRL 0,INSERTS 11 ZONE AT FLD END,NO ZERO SU IDEAL019
                       *EDIT CNTRL -N,- AT END+1,ZERO SUPP TO N RT DIGITS  IDEAL020
                       *USE INT VARI FOR NEG CNTRL TO SAVE 5 WDS PER CALL  IDEAL021
                       *EDIT CNTRL +N,-AT END+2,N DECI PL,ZERO SUPP TO DEC IDEAL022
                       *EDIT CNTRL N GREATER THAN FIELD WIDTH,N IS +0 OR-0 IDEAL023
                       *DOUBLE INTEGERS HAVE STD PREC REAL VARIABLE NAMES  IDEAL024
                       *                                                   IDEAL025
005D     14889040            ENT      MSIA        SUBROUTINE NAME        IDEAL026
                       *CALL MSIA (OUT ARRAY,START,END,INTEGER,EDIT CNTRL) IDEAL027
                       *CALL MSIA MOVES AND CONVERTS SIG WD INT TO A1      IDEAL028
                       *EDIT CNTRL 0,INSERTS 11 ZONE AT FLD END,NO ZERO SU IDEAL029
                       *EDIT CNTRL -N,- AT END+1,ZERO SUPP TO N RT DIGITS  IDEAL030
                       *USE INT VARI FOR NEG CNTRL TO SAVE 5 WDS PER CALL  IDEAL031
                       *EDIT CNTRL +N,-AT END+2,N DECI PL,ZERO SUPP TO DEC IDEAL032
                       *EDIT CNTRL N GREATER THAN FIELD WIDTH,N IS +0 OR-0 IDEAL033
                       *                                                   IDEAL034
                       *IDEAL 1130 FORTRAN ERROR CODE IS /DEAF IN ACCUM.   IDEAL035
                       *TO DISPLAY STATEMENT ALLOCATION ADDR IN ERROR, HIT IDEAL036
                       *START,ACCUM HAS FORTRAN STATEMENT ALLOCATION ADDR. IDEAL037
                       *HIT START TO CONTINUE.OUTPUT IS SET TO ZERO.       IDEAL038
                       *                                                   IDEAL039
0000 0  0000           MASI  DC         *-*       SUBROUTINE ENTRY POINT   IDEAL040
0001 0  C0FE                 LD         MASI      CALL&1 ADDRESS           IDEAL041
0002 0  D071                 STO        ARGMT&1   SET UP FOR ARGMT ADDR LOAD IDEAL042
0003 0  406A                 BSI        SAVE      SAVE XRS & GET PARAMETERS  IDEAL043
0004 01 74FF010E             MDX   L    BACK&1,-1 ADJ RETURN ADDR FOR 4 PARA IDEAL044
0006 0  C054                 LD         HD480     STO I SINGLE INT INST.   IDEAL045
0007 0  D04E                 STO        OUT       AT OUT INSTRUCTION       IDEAL046
0008 0  1010                 SLA        16        CLEAR ACC                IDEAL047
0009 0  2000                 LDS        0         INITIALIZE OVERFLOW      IDEAL048
000A 01 D4000090       ASI   STO   L    WORK      LOAD REMAINING INTEGER   IDEAL049
000C 01 C48000E8             LD    I    FLDST     FIELD CHARA TO ACC       IDEAL050
000E 0  E04B                 AND        H0F00     A1 DIGIT MASK            IDEAL051
000F 0  1808                 SRA        8         SHIFT TO BINARY DIGIT SIGL IDEAL052
0010 01 D4000092             STO   L    DIGIT     HOLD INTEGER DIGIT       IDEAL053
0012 01 C4000090             LD    L    WORK      LOAD PARTIAL INTEGER     IDEAL054
0014 0  1002                 SLA        2         WORK TIMES FOUR          IDEAL055
0015 0  807A                 A          WORK      PLUS ONE IS FIVE         IDEAL056
0016 0  1001                 SLA        1         DOUBLE EQUALS TEN TIMES  IDEAL057
0017 01 4C010117             BSC   L    TOBIG,O   TOO BIG IF OVERFLOW      IDEAL058
0019 0  8078                 A          DIGIT     ADD DIGIT 187            IDEAL059
001A 01 4C290117             BSC   L    TOBIG,O&Z  TOBIG IF OVERFLOW OR NEG IDEAL060
```

```
001C 01 74FF00E8             MDX   L  FLDST,-1     THIS IS NOW NEXT COLUMN      IDEAL061
001E 01 74FF00E9             MDX   L  FLDWD,-1     THESE COLUMNS ARE YET TO G   IDEAL062
0020 0  70E9                 MDX      ASI          SKIP THIS IF FLDWD IS ZERO   IDEAL063
0021 0  D06E                 STO      WORK         SAVE CONVERTED INTEGER       IDEAL064
0022 01 C48000EA             LD    I  FLDND        GET END COLUMN CHARACTER     IDEAL065
0024 0  E077                 AND      HF000        ISOLATE ZONES                IDEAL066
0025 0  F074                 EOR      HD000        ACC IS ZERO IF 11 ZONE       IDEAL067
0026 01 4C20002B             BSC   L  LDASI,Z                                   IDEAL068
0028 0  1010                 SLA      16           CLEAR ACC                    IDEAL069
0029 0  9066                 S        WORK         MAKE COMPLIMENT INTEGER      IDEAL070
002A 0  702B                 MDX      OUT          GO TO STORE NEG INTEGER      IDEAL071
002B 0  C064           LDASI LD       WORK         LOAD POSITIVE INTEGER        IDEAL072
002C 0  7029                 MDX      OUT          RESTORE                      IDEAL073
                       *                                                        IDEAL074
002D 0  0000           MADI  DC       *-*          SUBROUTINE ENTRY POINT       IDEAL075
002E 0  C0FE                 LD       MADI         CALL&1 ADDRESS               IDEAL076
002F 0  D044                 STO      ARGMT&1      SET UP FOR ARGMT ADDR LOAD   IDEAL077
0030 0  403D                 BSI      SAVE         SAVE XRS & GET PARAMETERS    IDEAL078
0031 01 74FF010E             MDX   L  BACK&1,-1    ADJ RETURN ADDR FOR 4 PARA   IDEAL079
0033 0  C028                 LD       HDC80        STD I DBL INT INST.          IDEAL080
0034 0  D021                 STO      OUT          AT OUT INSTRUCTION           IDEAL081
0035 0  10A0                 SLT      32           CLEAR ACC & EXT              IDEAL082
0036 0  2000                 LDS      0            INITIALIZE OVERFLOW          IDEAL083
0037 0  D858           ADI   STD      WORK                                      IDEAL084
0038 01 C48000E8             LD    I  FLDST        FIELD CHARA TO ACC           IDEAL085
003A 0  E01F                 AND      H0F00        A1 DIGIT MASK                IDEAL086
003B 0  1898                 SRT      24           SHIFT TO BINARY DIGIT DBL    IDEAL087
003C 0  D855                 STD      DIGIT        HOLD DBL INTEGER DIGIT       IDEAL088
003D 0  C852                 LDD      WORK         LOAD PARTIAL DBL INTEGER     IDEAL089
003E 0  1082                 SLT      2            WORK TIMES FOUR              IDEAL090
003F 0  8850                 AD       WORK         PLUS ONE IS FIVE             IDEAL091
0040 0  1081                 SLT      1            DOUBLE EQUALS TEN TIMES      IDEAL092
0041 01 4C010117             BSC   L  TOBIG,O      TOO BIG IF OVERFLOW          IDEAL093
0043 0  884E                 AD       DIGIT        ADD DIGIT                    IDEAL094
0044 01 4C290117             BSC   L  TOBIG,O&Z    TOO BIG IF OVERFLOW OR NEG   IDEAL095
0046 01 74FF00E8             MDX   L  FLDST,-1     THIS IS NOW NEXT COLUMN      IDEAL096
0048 01 74FF00E9             MDX   L  FLDWD,-1     THESE COLUMNS ARE YET TO G   IDEAL097
004A 0  70EC                 MDX      ADI          BRANCH TO MADI ROUTINE       IDEAL098
004B 0  D844                 STD      WORK         SAVE CONVERTED DBL INTEGER   IDEAL099
004C 01 C48000EA             LD    I  FLDND        GET END COLUMN CHARACTER     IDEAL100
004E 0  E04D                 AND      HF000        ISOLATE ZONES                IDEAL101
004F 0  F04A                 EOR      HD000        ACC IS ZERO IF 11 ZONE       IDEAL102
0050 01 4C200055             BSC   L  LDADI,Z      BRANCH IF NOT AN 11 ZONE     IDEAL103
0052 0  10A0                 SLT      32           CLEAR ACC & EXT              IDEAL104
0053 0  983C                 SD       WORK         MAKE COMPLIMENT DBL INT      IDEAL105
0054 0  7001                 MDX      OUT          GO TO STORE NEG DBL INTEG    IDEAL106
0055 0  C83A           LDADI LDD      WORK         LOAD POSITIVE DBL INTEGER    IDEAL107
0056 01 D48000EC       OUT   STO   I  ARG4A        INT TO PROG VARIABLE         IDEAL108
0058 01 4C00010A             BSC   L  INDX1                                     IDEAL109
005A 0  0F00           H0F00 DC       /0F00        DIGIT ISOLATION MASK         IDEAL110
005B 0  D480           HD480 DC       /D480        STO INDIRECT INST            IDEAL111
005C 0  DC80           HDC80 DC       /DC80        STD A&Q INDIRECT INST        IDEAL112
                       *                                                        IDEAL113
005D 0  0000           MSIA  DC       *-*          SUBROUTINE ENTRY POINT       IDEAL114
005E 0  C0FE                 LD       MSIA         CALL&1 ADDRESS               IDEAL115
005F 0  D014                 STO      ARGMT&1      SET UP FOR ARGMT ADDR LOAD   IDEAL116
0060 0  400D                 BSI      SAVE         SAVE XRS & GET PARAMETERS    IDEAL117
0061 0  1090                 SLT      16           CLEAR EXTENSTION             IDEAL118
```

THE

```
0062 01 C48000EC          LD   I  ARG4A       LOAD INTEGER                   IDEAL119
0064 01 442800B3          BSI  L  NEG,&Z      BRANCH IF NEGATIVE             IDEAL120
0066 0  D029              STO     WORK        SAVE INTEGER                   IDEAL121
0067 0  C028        SIA   LD      WORK        LOAD WORK INTEGER              IDEAL122
0068 0  1890              SRT     16          SET UP DIVIDEND IN A&Q         IDEAL123
0069 0  A82F              D       TEN         GET LOW DIGIT TO EXT           IDEAL124
006A 0  D025              STO     WORK        SAVE NEW INTEGER QUOTIENT      IDEAL125
006B 0  1098              SLT     24          SHIFT DIGIT TO ACC             IDEAL126
006C 0  404E              BSI     TEST        TEST FOR NEG & FIELD WIDTH     IDEAL127
006D 0  70F9              MDX     SIA         GET NEXT DIGIT                 IDEAL128
                    *                                                        IDEAL129
006E 0  0000        SAVE  DC      *-*         RETURN ADDR HERE               IDEAL130
006F 01 6D00010B          STX  L1 INDX1&1     SAVE INDEX ONE                 IDEAL131
0071 01 2C00010C          STS  L  STATS       SAVE STATUS                    IDEAL132
0073 00 65000000    ARGMT LDX  L1 *-*         ARGUMENT ONE ADDR TO INDX1     IDEAL133
0075 0  C100              LD    1 0           ARRAY ADDR TO ACCUM            IDEAL134
0076 00 95800002          S    I1 2           SUBTRACT FIELD END COLUMN      IDEAL135
0078 01 D40000F3          STO  L  SIGN&1      SAVE SIGN ADDR                 IDEAL136
007A 0  801D              A       ONE         ADD ONE                        IDEAL137
007B 0  D06E              STO     FLDND       FIELD END ADDR                 IDEAL138
007C 0  C100              LD    1 0           ARRAY ADDR TO ACCUM            IDEAL139
007D 0  801A              A       ONE         ADD ONE                        IDEAL140
007E 00 95800001          S    I1 1           SUBTRACT FIELD START COLMN     IDEAL141
0080 0  D067              STO     FLDST       FIELD STARTING ADDR            IDEAL142
0081 0  8016              A       ONE         ADD ONE                        IDEAL143
0082 0  9067              S       FLDND       SUBTRACT FIELD END ADDRESS     IDEAL144
0083 0  D065              STO     FLDWD       STORE FIELD WIDTH              IDEAL145
0084 0  D066              STO     ZSCNT       FIELD WIDTH TO ZERO SUPP       IDEAL146
0085 0  C103              LD    1 3           ARG4 ADDR TO ACCUM             IDEAL147
0086 0  D065              STO     ARG4A       SAVE ARGUMENT 4 ADDRESS        IDEAL148
0087 0  C104              LD    1 4           ARGUMENT 5 ADDR TO ACCUM       IDEAL149
0088 0  D064              STO     ARG5A       SAVE ARGUMENT 5 ADDRESS        IDEAL150
0089 0  7105              MDX   1 5           ADJUST RETURN ADDRESS          IDEAL151
008A 01 6D00010E          STX  L1 BACK&1      LOAD BRANCH BACK INST          IDEAL152
008C 0  1010              SLA     16          CLEAR ACC                      IDEAL153
008D 0  2000              LDS     0           SET OVERFLOW & CARRY OFF       IDEAL154
008E 01 4C80006E          BSC  I  SAVE        RETURN TO SUBROUTINE           IDEAL155
                    *                                                        IDEAL156
0090    0002        WORK  BSS  E  2           CONVERSION WORK AREA           IDEAL157
0092 0  0000        DIGIT DC      0           DOUBLE WORD DIGIT AREA         IDEAL158
0093 0  0000              DC      0           SECOND WORD OF DIGIT AREA      IDEAL159
0094 0  0000        EVEN  DC   E  0           EVEN WORD ONE                  IDEAL160
0095 0  0000              DC      0           EVEN WORD TWO                  IDEAL161
0096 0  0000        ODD   DC      0           ODD WORD ONE                   IDEAL162
0097 0  0000              DC      0           ODD WORD TWO                   IDEAL163
0098 0  0001        ONE   DC      1           ONE                            IDEAL164
0099 0  000A        TEN   DC      10                                         IDEAL165
009A 0  D000        HD000 DC      /D000       11 ZONE MASK                   IDEAL166
009B 0  DFFF        HDFFF DC      /DFFF       NEGATIVE  ZONE MASK            IDEAL167
009C 0  F000        HF000 DC      /F000       ZONE ISOLATION MASK            IDEAL168
009D 0  F040        HF040 DC      /F040       DIGIT MASK FOR A1 FORMAT       IDEAL169
                    *                                                        IDEAL170
009E 0  0000        MDIA  DC      *-*         SUBROUTINE ENTRY POINT         IDEAL171
009F 0  C0FE              LD      MDIA        CALL&1 ADDRESS                 IDEAL172
00A0 0  D0D3              STO     ARGMT&1     SET UP FOR ARGMT ADDR LOAD     IDEAL173
00A1 0  40CC              BSI     SAVE        SAVE XRS & GET PARAMETERS      IDEAL174
00A2 01 CC8000EC          LDD  I  ARG4A       LOAD DOUBLE WORD INTEGER       IDEAL175
00A4 01 442800B3          BSI  L  NEG,&Z      BRANCH IF NEGATIVE             IDEAL176
00A6 0  D0EE              STO     EVEN&1      STORE EVEN WORD OF DBL INT     IDEAL177
```

```
00A7 0  1090                SLT      16          SHIFT ODD WORD TO ACC         IDEAL178
00A8 0  D0EE                STO      ODD&1       STORE ODD WORD OF INTEGER     IDEAL179
00A9 0  C8EA         DIA    LDD      EVEN        EXTRACT NEXT DIGIT            IDEAL180
00AA 0  A8EE                D        TEN         GET LOW DIGIT TO EXT          IDEAL181
00AB 0  D0E9                STO      EVEN&1      SAVE NEW EVEN WORD            IDEAL182
00AC 0  1090                SLT      16          SHIFT DIGIT TO ACC            IDEAL183
00AD 0  88E8                AD       ODD         ADD ODD WORD                  IDEAL184
00AE 0  A8EA                D        TEN         GET LOW DIGIT TO EXT          IDEAL185
00AF 0  D0E7                STO      ODD&1       SAVE NEW ODD WORD             IDEAL186
00B0 0  1098                SLT      24          SHIFT DIGIT TO ACC            IDEAL187
00B1 0  4009                BSI      TEST        TEST FOR NEG & FIELD WIDTH    IDEAL188
00B2 0  70F6                MDX      DIA         GET NEXT DIGIT                IDEAL189
                     *                                                         IDEAL190
00B3 0  0000         NEG    DC       *-*         ENTRY FOR NEG ROUTINE         IDEAL191
00B4 0  D8DB                STD      WORK        STORE DOUBLE INTEGER          IDEAL192
00B5 0  C0E2                LD       ONE         SET NEGATIVE SWITCH ON        IDEAL193
00B6 0  D037                STO      NEGSW       SET NEGATIVE SWITCH ON        IDEAL194
00B7 0  10A0                SLT      32          CLEAR ACC & EXT               IDEAL195
00B8 0  98D7                SD       WORK        COMPLIMENT VALUE IN WORK      IDEAL196
00B9 01 4C8000B3            BSC   I  NEG         GO TO CONVERSION ROUTINE      IDEAL197
                     *                                                         IDEAL198
00BB 0  0000         TEST   DC       *-*         TEST RETURN                   IDEAL199
00BC 0  E8E0                OR       HF040       MASK DIGIT FOR A1 FORMAT      IDEAL200
00BD 01 D48000EA            STO   I  FLDND       OUTPUT RECOVERED DIGIT        IDEAL201
00BF 0  C02E                LD       NEGSW       LOAD SWITCH CONDITION         IDEAL202
00C0 01 4C0800CD            BSC   L  POS,&       BRANCH TO POS IF SW OFF       IDEAL203
00C2 01 C48000ED            LD    I  ARG5A       EDIT CONTROL PARAMETER        IDEAL204
00C4 01 4C20010F            BSC   L  MINUS,Z     BRANCH IF NOT 11 ZONE CODE    IDEAL205
                     * INSERT 11 ZONE OVER FIELD END DIGIT                     IDEAL206
00C6 01 C48000EA            LD    I  FLDND       GET END DIGIT OF FIELD        IDEAL207
00C8 0  E0D2                AND      HDFFF       INSERT 11 ZONE WITH DIGIT     IDEAL208
00C9 01 D48000EA            STO   I  FLDND       PUT DIGIT WITH 11 ZONE        IDEAL209
00CB 0  1010         SW     SLA      16          CLEAR ACC                     IDEAL210
00CC 0  D021                STO      NEGSW       SET NEG SWITCH TO ZERO        IDEAL211
00CD 01 740100EA     POS    MDX   L  FLDND,&1    THIS IS NOW NEXT COLUMN       IDEAL212
00CF 01 74FF00E9            MDX   L  FLDWD,-1    THESE COLUMNS ARE YET TO G    IDEAL213
00D1 0  7001                MDX      AGAIN       SKIP THIS IF FLDWD IS ZERO    IDEAL214
00D2 0  7002                MDX      EDIT                                      IDEAL215
00D3 01 4C8000BB     AGAIN  BSC   I  TEST        GET NEXT DIGIT                IDEAL216
                     *                                                         IDEAL217
00D5 01 C48000ED     EDIT   LD    I  ARG5A       EDIT CONTROL PARAMETER        IDEAL218
00D7 01 4C18010A            BSC   L  INDX1,&-    OMIT SUPP ZEROS IF ZERO       IDEAL219
00D9 01 4C1000E0            BSC   L  DEC,-       NO DECI POINT                 IDEAL220
00DB 0  800F                A        ZSCNT       CALC LEFT ZERO MAX COUNT      IDEAL221
00DC 01 4C2800FC            BSC   L  SLZ,&Z      N GT FLDWD,CNTRL IS -0        IDEAL222
00DE 0  D00C                STO      ZSCNT       MAX LEFT ZERO SUPPRESS        IDEAL223
00DF 0  701C                MDX      SLZ         BRANCH TO SUPPRESS ZEROS      IDEAL224
00E0 0  D008         DEC    STO      FLDWD       STORE WIDTH OF DECI FIELD     IDEAL225
00E1 0  C009                LD       ZSCNT       ENTIRE FIELD WIDTH            IDEAL226
00E2 0  9006                S        FLDWD       SUBTRACT DECI WIDTH           IDEAL227
00E3 01 4C1000EF            BSC   L  NORM,-      N GT FLDWD,CNTRL IS &0        IDEAL228
00E5 0  C0B2                LD       ONE         LOAD ONE                      IDEAL229
00E6 0  D002                STO      FLDWD       SET FLDWD TO ONE FOR SIGN     IDEAL230
00E7 0  700A                MDX      SIGN        BRANCH TO SIGN                IDEAL231
00E8 0  0000         FLDST  DC       0           FIELD START ADDR              IDEAL232
00E9 0  0000         FLDWD  DC       0           FIELD WIDTH                   IDEAL233
00EA 0  0000         FLDND  DC       0           FIELD END ADDR                IDEAL234
00EB 0  0000         ZSCNT  DC       0           ZERO SUPPRESS MAX COUNT       IDEAL235
```

```
00EC 0  0000        ARG4A DC       0           ARGUMENT 4 ADDRESS          IDEAL236
00ED 0  0000        ARG5A DC       0           ARGUMENT 5 VALUE            IDEAL237
00EE 0  0000        NEGSW DC       0                                       IDEAL238
                    *                                                      IDEAL239
                    * SHIFT RIGHT TO FIELD END PLUS TWO FOR DECI POINT     IDEAL240
00EF 0  D0FB        NORM  STO      ZSCNT       MAX LEFT ZERO COUNT         IDEAL241
00F0 01 740100E9          MDX   L  FLDWD,1     ADJUST FOR SIGN POSITION    IDEAL242
00F2 00 65000000    SIGN  LDX   L1 *-*         LOAD  SIGN ADDRESS          IDEAL243
00F4 0  C100        NEXT  LD     1 0           LOAD CHARACTER              IDEAL244
00F5 0  D1FF              STO    1 -1          SHIFT RIGHT ONE COLUMN      IDEAL245
00F6 0  7101              MDX    1 &1          INCREMENT INDEX FOR NEXT    IDEAL246
00F7 01 74FF00E9          MDX   L  FLDWD,-1    DECREMENT DECI WIDTH CNTR   IDEAL247
00F9 0  70FA              MDX      NEXT        SKIP IF LAST DECI POSITION  IDEAL248
00FA 0  C01A              LD       H4B40       LOAD DECIMAL POINT          IDEAL249
00FB 0  D1FF              STO    1 -1          INSERT DECIMEL POINT        IDEAL250
                    * SUPPRESS LEADING ZEROS IN OUTPUT FIELD PER COUNT     IDEAL251
00FC 01 74FF00EA    SLZ   MDX   L  FLDND,-1    THIS IS NEXT COLUMN RIGHT   IDEAL252
00FE 01 C48000EA          LD    I  FLDND       LOAD LEFT CHARACTER         IDEAL253
0100 01 E400005A          AND   L  H0F00       ISOLATE DIGIT               IDEAL254
0102 01 4C20010A          BSC   L  INDX1,Z     BRANCH OUT IF NOT ZERO      IDEAL255
0104 0  C00F              LD       H4040       LOAD BLANK TO REPLACE ZERO  IDEAL256
0105 01 D48000EA          STO   I  FLDND       OUTPUT BLANK TO FIELD       IDEAL257
0107 01 74FF00EB          MDX   L  ZSCNT,-1    MAX ZERO SUPP SKIP EXIT     IDEAL258
0109 0  70F2              MDX      SLZ         SKIP THIS IF MAX CNT ZERO   IDEAL259
010A 00 65000000    INDX1 LDX   L1 *-*         RESTORE INDEX REG ONE       IDEAL260
010C 0  2000        STATS LDS      *-*         RESTORE STATUS              IDEAL261
010D 00 4C000000    BACK  BSC   L  *-*         BRANCH BACK TO FORTRAN PRO  IDEAL262
                    * INSERT MINUS SIGN AT FIELD END PLUS ONE              IDEAL263
010F 0  C006        MINUS LD       H6040       MINUS SIGN                  IDEAL264
0110 01 D48000F3          STO   I  SIGN&1      AT FIELD END PLUS ONE       IDEAL265
0112 01 4C0000CB          BSC   L  SW                                      IDEAL266
0114 0  4040        H4040 DC       /4040       BLANK                       IDEAL267
0115 0  4B40        H4B40 DC       /4B40       PERIOD                      IDEAL268
0116 0  6040        H6040 DC       /6040       DASH                        IDEAL269
                    *                                                      IDEAL270
0117 00 74000032    TOBIG MDX   L  50,0                                    IDEAL271
0119 0  70FD              MDX      TOBIG       INTERRUPT SERVICE LOOP      IDEAL272
011A 0  C009              LD       HDEAF       IDEAL FORTRAN ERROR CODE    IDEAL273
011B 0  3000              WAIT                 DISPLAY ERROR CODE IN ACCU  IDEAL274
011C 01 C4000074          LD    L  ARGMT&1     DISPLAY ENTRY ADDRESS       IDEAL275
011E 0  9004              S        H01C4       AND DISPLAY STATEMENT       IDEAL276
011F 0  3000              WAIT                 ALLOCATION ADDR IN ACCUM.   IDEAL277
0120 0  10A0              SLT      32          CLEAR TO OUTPUT ZERO INTEG  IDEAL278
0121 01 4C000056          BSC   L  OUT                                     IDEAL279
0123 0  01C4        H01C4 DC       /01C4       DISKZ ORGIN +2              IDEAL280
0124 0  DEAF        HDEAF DC       /DEAF       IDEAL FORTRAN ERROR CODE    IDEAL281
0126                      END                                              IDEAL282

     NO ERRORS IN ABOVE ASSEMBLY.
```

```
// ASM
*LIST SOURCE PROGRAM
                          *      THIS IS QPASS                                        PASS0010
                          *                                                           PASS0020
0000      185C18A2               ENT        QPASS                                     PASS0030
0000      0001            QPASS BSS         1                                         PASS0040
0001 0    6A20                   STX     2  SAVE&1       SAVE XR2                     PASS0050
0002 01   66800000               LDX    I2  QPASS        LOAD XR2 WITH CALL&1         PASS0060
0004 0    C200                   LD     X2  0            LOAD SOURCE AREA TO ACC      PASS0070
0005 0    1001                   SLA    X   1            SHIFT LEFT ONE BIT           PASS0080
0006 00   96800001               S      I2  1            SUB COLUMN NO OF SC FIELD    PASS0090
0008 0    801E                   A          TWO          ADD TWO                      PASS0100
0009 0    D01B                   STO        SCRAD        STORE ACC INTO SCRAD         PASS0110
000A 00   C6800004               LD     I2  4            LOAD CHARACTER COUNT TO AC   PASS0120
000C 0    D019                   STO        CHCNT        STORE IN CHCNT               PASS0130
000D 0    C202                   LD     X2  2            LOAD DEST FIELD ADD TO ACC   PASS0140
000E 0    1001                   SLA    X   1            SHIFT LEFT ONE BIT           PASS0150
000F 00   96800003               S      I2  3            SUB DEST CH NO FROM ACC      PASS0160
0011 0    8015                   A          TWO          ADD TWO                      PASS0170
0012 0    7205                   MDX    X2  5            ADD 5 TO XR2                 PASS0180
0013 0    6A10                   STX     2  RETRN&1      STORE IN RETURN INSTR        PASS0190
0014 00   D4000002               STO    L   /0002        STORE ACC IN XR2             PASS0200
0016 0    C00E            LOOP   LD         SCRAD        LOAD SCRAD TO ACC            PASS0210
0017 30   181D9042               CALL       QGRAB        GET SOURCE CHARACTER         PASS0220
0019 30   18888925               CALL       QSHUV        PUT IN DESTINATION FIELD     PASS0230
001B 0    72FF                   MDX    X2  -1           DECREMENT XR2                PASS0240
001C 01   74FF0025               MDX    L   SCRAD,-1     DECREMENT SCRAD              PASS0250
001E 01   74FF0026               MDX    L   CHCNT,-1     DECREMENT CHCNT              PASS0260
0020 0    70F5                   MDX        LOOP         NOT ZERO, GO TO LOOP         PASS0270
0021 01   66000023        SAVE   LDX    L2  *            RESTORE XR2                  PASS0280
0023 01   4C000025        RETRN  BSC    L   *            RETURN                       PASS0290
0025 1    0026            SCRAD  DC         *                                         PASS0300
0026 1    0027            CHCNT  DC         *                                         PASS0310
0027 0    0002            TWO    DC         2                                         PASS0320
0028                             END                                                  PASS0330

     NO ERRORS IN ABOVE ASSEMBLY.
```

```
// ASM
*LIST SOURCE PROGRAM
                             *     THIS IS QSHUV                                   SHUV0010
                             *                                                     SHUV0020
0000      18888925                 ENT      QSHUV                                  SHUV0030
0000      0001           QSHUV BSS      1                                          SHUV0040
0001 0    D013                     STO      TEMP     STORE CHARACTER INTO TEMP     SHUV0050
0002 00 C4000002                   LD    L  2        LOAD XR2 TO ACC               SHUV0060
0004 0    1881                     SRT   X  1        DIVIDE BY 2                   SHUV0070
0005 0    D005                     STO      AND&1    STORE WORD ADD IN MASK INS    SHUV0080
0006 0    1091                     SLT   X  17       MOVE REMAINDER INTO CARRY     SHUV0090
0007 0    C00F                     LD       MASK     LOAD MASK TO ACC              SHUV0100
0008 0    4802                     BSC      C        SKIP IF CARRY OFF             SHUV0110
0009 0    1808                     SRA   X  8        SHIFT RIGHT IF CARR ON        SHUV0120
000A 01 E400000C         AND       AND   L  *        MASK DESTINATION WORD         SHUV0130
000C 0    D009                     STO      TEMP1    STORE INTO TEMP1              SHUV0140
000D 0    C007                     LD       TEMP     LOAD CHARACTER TO ACC         SHUV0150
000E 0    4802                     BSC      C        GO TO COMBINE CHAR IF CARY    SHUV0160
000F 0    1008                     SLA   X  8        SHIFT LEFT IF CARRY OFF       SHUV0170
0010 0    E805                     OR       TEMP1    COMBINE CHARACTERS            SHUV0180
0011 01 D480000B                   STO   I  AND&1    STORE INTO DESTINATION        SHUV0190
0013 01 4C800000                   BSC   I  QSHUV    RETURN                        SHUV0200
0015 0    0000           TEMP      DC       0                                      SHUV0210
0016 0    0000           TEMP1     DC       0                                      SHUV0220
0017 0    FF00           MASK      DC       /FF00                                  SHUV0230
0018                               END                                             SHUV0240

     NO ERRORS IN ABOVE ASSEMBLY.
```

```
// JOB T  *SE10  42192          .1  SWAIN CALL ME
// FOR
*IOCS(TYPEWRITER,1132 PRINTER,DISK)
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
C  DEMONSTRATE PMERG BY SORTING 1000 RECORDS OF 32 WORDS
C  USE 5 SORT KEYS, OCCUPYING 8 WORDS
C  MAJOR KEY IS WORD 6, INTEGER
C  SECOND KEY IS WORDS 14 AND 13, TREAT AS DOUBLE WORD INTEGER
C         (SIGN AND HIGH-ORDER BITS IN WORD 14, LOW ORDER BITS WORD 13)
C  THIRD KEY IS WORD 21, TREAT AS 2 ALPHABETIC CHARACTERS
C  FOURTH KEY STARTS IN WORD 16, REAL
C         (SIGN AND HIGH-ORDER BITS IN WORD 16
C          LOW ORDER BITS AND EXPONENT IN WORD 15)
C  MINOR KEY IS WORDS 30 THRU 32 TREAT AS 3 ALPHABETIC CHARACTERS,
C         STARTING FROM RIGHT CHARACTER OF WORD 30
      EXTERNAL LARGE
C  LARGE IS NAME OF PROGRAM TO DETERMINE IF RECORD IS TO BE
C  INCLUDED IN SORT
      DIMENSION IREC(32)
      DIMENSION KEYS(4,5),IWORK(1850),IBUF(325)
      DIMENSION AREC(3)
      DIMENSION NAME(25),IOUT(11)
      EQUIVALENCE(AREC(1),IREC(16)),(DOUB,IREC(14))
      DEFINE FILE 1(101,320,U,I1)
      DEFINE FILE 2(30,320,U,I2)
      DEFINE FILE 3(30,320,U,I3)
      DEFINE FILE 4(4,320,U,I4)
C  FILE 1 IS FILE TO BE SORTED.
C  FILES 2 AND 3 ARE WORK FILES
C  FILE 4 IS OUTPUT. CONTAINS POINTERS TO INDICATE PROCESSING ORDER
C  AS DETERMINED BY SORT KEY INFORMATION
C
C  GENERATE 1010 RECORDS OF RANDOM NUMBERS
C  PMERG WILL EXAMINE LAST 1000 OF THESE TO SELECT THOSE TO BE
C  INCLUDED IN SORT, AND THEN SORT THOSE INCLUDED
      CALL DOPEN(IBUF,1,32,10)
      K=31525
      L=899
      IX=1000
      IK=5
      IB=10
      IL=IB+IX-1
C  CREATE ALPHABETIC CHARACTERS IN NAME. ALL VALID PRINTER CHARACTERS
C  ARE INCLUDED
      NAME(1)=16459
      NAME(2)=19790
      NAME(3)=20571
      NAME(4)=23645
      NAME(5)=24673
      NAME(6)=27517
      NAME(7)=32449
      NAME(8)=-15677
      NAME(9)=-15163
      NAME(10)=-14649
      NAME(11)=-14135
      NAME(12)=-11822
      NAME(13)=-11308
      NAME(14)=-10794
```

TH

```
      NAME(15)=-10280
      NAME(16)=-9758
      NAME(17)=-7196
      NAME(18)=-6682
      NAME(19)=-6168
      NAME(20)=-5648
      NAME(21)=-3598
      NAME(22)=-3084
      NAME(23)=-2570
      NAME(24)=-2056
      NAME(25)=-1792
C  STORE RANDOM INTEGER IN RANGE -9 TO +9 IN WORDS 1-14
      DO 1 I=1,IL
      DO 2 J=1,14
      K=K*L
    2 IREC(J)=K/3277
C  STORE RANDOM REAL NUMBER IN RANGE -10 TO +10 IN WORDS 15-20
      DO 5 J=1,3
      K=K*L
    5 AREC(J)=FLOAT(K)/3277.
C  STORE RANDOM ALPHABETIC CHARACTER IN WORDS 21-32
      DO 6 J=1,24
      K=K*L
      M=K/1311+25
    6 CALL QPASS(NAME,M,IREC(21),J,1)
    1 CALL DPUT(IBUF,I,IREC)
      CALL DCLOS(IBUF)
C  CREATE  KEYS TABLE, TO INDICATE LOCATION AND TYPE OF SORT KEYS
C  MAJOR KEY IS INTEGER, WORD 6
      KEYS(1,1)=1
      KEYS(2,1)=6
C  SECOND KEY IS HIGH PRECISION INTEGER, HIGH-ORDER PART IN WORD 14
C  SORT INTO DESCENDING ORDER BY THIS KEY
      KEYS(1,2)=-2
      KEYS(2,2)=14
C  THIRD KEY IS ALPHABETIC, FIRST CHARACTER  WORD 21 COL 1, 2 CHARACTERS
      KEYS(1,3)=3
      KEYS(2,3)=21
      KEYS(3,3)=1
      KEYS(4,3)=2
C  FOURTH KEY IS REAL, STARTS IN WORD 16
      KEYS(1,4)=4
      KEYS(2,4)=16
C  FIFTH KEY IS ALPHABETIC, FIRST CHARACTER WORD 30 COL 2, 3 CHARACTERS
      KEYS(1,5)=3
      KEYS(2,5)=30
      KEYS(3,5)=2
      KEYS(4,5)=3
C
C  PERFORM SORT-MERGE
      WRITE(1,102)
  102 FORMAT('START SORT-MERGE')
      CALL PMERG(1,32,IB,IX ,KEYS,IK,IWORK,1850,IBUF,10,2,3,4,
     $LARGE,KOUNT)
      WRITE(1,103)KOUNT
  103 FORMAT('END SORT-MERGE',I6,' RECORDS INCLUDED')
C
C  LIST FIRST 10 RECORDS IN SORTED FILE
```

```
      CALL DOPEN(IBUF,1,32,10)
      CALL DOPEN(IWORK,4,1,320)
      DO 4 I=1,10
      CALL DGET(IWORK,I,IPT)
      WRITE(3,101)IPT
  101 FORMAT(' RECORD NO.',I5)
      CALL DGET(IBUF,IPT,IREC)
      CALL QPASS(NAME,1,IOUT,21,1)
      CALL MDIA(IOUT,1,10,DOUB,-1)
    4 WRITE(3,104)(IREC(J),J=1,12),IOUT,AREC,(IREC(J),J=21,32)
  104 FORMAT(' ',12I3,11A1,1X,3F8.4,1X,12A2)
      STOP
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS
 IOCS

CORE REQUIREMENTS FOR
 COMMON        0  VARIABLES    2324   PROGRAM     674

END OF COMPILATION
```

```
// XEQ        L  1
*LOCAL,SRTPH,MRGPH

FILES ALLOCATION

    1 0432  0065
    2 0497  001E
    3 04B5  001E
    4 04D3  0004

STORAGE ALLOCATION

R 47  0010 (HEX) WORDS AVAILABLE

          CALL TRANSFER VECTOR

          DISGN   1C66
          QCOMP   1C32
          INT     1BBA
          SD      1B94
          QSHUV   1B7C
          QGRAB   1B70
          MINO    1A78
          IPTSK   1940
          DUSE    1855
          MDIA    178C
          DGET    16B1
          LARGE   168D
          PMERG   1638
          DCLOS   15C6
          DPUT    1576
          QPASS   1546
          DOPEN   1498
          MRGPH   1D25  LOCAL
          SRTPH   1CF8  LOCAL

          LIBF TRANSFER VECTOR

          FSUB    1BCC
          SDRED   0D9E
          SDCOM   0DC2
          SDIX    0D94
          SDWRT   0DEC
          FARC    1B4E
          NORM    1B24
          EBCTB   1B21
          GETAD   1AE0
          IFIX    1AB4
          PAUSE   1AA8
          SUBIN   1820
          STOP    1814
          SIOAF   0FEF
          SIOAI   0FFC
          SIOIX   1071
          SIOI    0FF7
          SCOMP   0FDF
          SWRT    0FD6
          FSTOX   145A
          FDIV    14F4
          FLOA1    14E6
          SUBSC    14C8
          FSTO     145E
          FLD      147A
          PRNTZ    1398
          WRTYZ    1360
          SFIO     10AD
          SDFIO    0DF1
          DISKZ    00F4

          SYSTEM ROUTINES

          ILS02    1F6F
          FLIPR    1C90
       0B3C (HEX) IS THE EXECUTION ADDR.


 RECORD NO.   764
  -8 -2  4  2 -3 -9  0  0 -5 -7 -4 -6     589830     4.8669 -4.3433 -4.8663 XHLNP2LYVQ.8BAW'MGCPQMI)
 RECORD NO.   975
   1 -1 -4 -8  4 -9  1 -1  9  0  9  0     589816     3.2807  9.5688  2.8877 CODHXQ*(C=7'(5FT Z5(B&-.
 RECORD NO.   861
  -7 -5  9 -6 -8 -9 -6  9  9 -8  0  3     458760     3.5761 -4.8565 -6.3280 .D14+)K=02,J+&*U-PB*H.X8
 RECORD NO.   349
  -4  3 -2 -1  4 -9 -7  4 -7  1  8  7     458753     1.7012  9.5175 -3.2032 LW(N((/H.=I$( 22R LZF6K*
 RECORD NO.   773
  -8 -7  6 -4 -8 -9 -4  4  6  4  3  2     458753     5.5974 -7.5419 -0.6350 7=,VNZ8K=+K6OZZC$R7ZIA.M
 RECORD NO.   860
   0  4  0 -5  7 -9 -5 -8  0 -4  0 -7     458750     8.7339 -7.7415 -0.0619 V+W ,KS7LVOFWD/E*6-62.//
 RECORD NO.   632
  -5  0 -3  0  7 -9 -2  6  4 -1  0  5     393219    -7.4418  9.3521  8.0967 K6&ILI1C*,4RM16I41TYHZYW
 RECORD NO.    74
  -7  0 -2 -5 -5 -9  0  0  6  0 -5 -5     393216     4.5422  3.7445  6.5855 N.+FF/MT.BAC-FL2LLE2UX(2
 RECORD NO.   533
   3  0 -9  0  0 -9 -6 -7  8  1 -8  0     393207    -1.5309  3.5706 -9.7946 ,4V$G)UK56SCC3SZ/UY*+++K
 RECORD NO.   965
  -4 -6  0  9  1 -9  5  6  3  3 -9 -6     327684     8.8004 -7.9325  8.1925 Y'RJDO1M7TYC,,L U2$Z6CS2
```

```
START SORT-MERGE

END SORT-MERGE   811 RECORDS INCLUDED
```

```
// JOB    *SE10  42192          •2  SWAIN
// FOR
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      FUNCTION INTAK(IREC)
C  THIS FUNCTION CAUSES ALL RECORDS TO BE INCLUDED IN A SORT
      INTAK=1
      RETURN
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS

CORE REQUIREMENTS FOR INTAK
 COMMON      0  VARIABLES      2  PROGRAM     14

END OF COMPILATION
```

SESSION NUMBER T.2.5
SPEAKERS

PANEL ON PROGRAMMER EVALUATION
A.S. GLOSTER, II, OAK RIDGE ASSOC. UNIV.
S.A. LYNCH, U.S. REDUCTION CO.
DR. L.H. BAKER, PIONEER HI-BRED CORN CO.
DR. PAUL HERWITZ, I.B.M.

PROFESSIONAL PROGRAMMERS AND ANALYSTS: PROBLEMS IN PERFORMANCE EVALUATION *

By: Arthur S. Gloster II

Oak Ridge Associated Universities is a nonprofit corporation engaged in research and educational activities located in Oak Ridge, Tennessee. The primary function of the corporation's data processing center is to apply electronic data processing techniques, where feasible, to research and administrative projects. The center contains 27 employees of which 13 are analyst/programmers. The center is divided into three groups: 1) scientific applications consisting of 6 personnel and a group leader, 2) commercial applications consisting of 5 personnel and a group leader, and 3) operations section consisting of a supervisor and 10 other employees. Oak Ridge Associated Universities has on site an IBM 1800 disk/tape system which is utilized by approximately 40% of the programming personnel. Approximately 60% of the personnel use IBM 360-50, 360-75 and CDC equipment located in the area. The analysis and programming function comprise a significant portion of the operating costs of the ORAU data processing center.

After discussion with personnel from other installations, we found that there are no reliable standards by which costs can be calculated in advance, schedules established, and the performance of personnel evaluated. Although the methods we used to establish schedules and costs are subjective and arbitrary, they in no way approach the accuracy of the methods developed in the hardware area. For example, IBM has established rates for EAM equipment and has even produced a slide rule to use for estimating job times.

A means of evaluating the professional analyst/programmer's job and a means of evaluating his effectiveness is highly desirable but rarely accomplished. Such means would be helpful predictors in determining the staff needed for a particular application or a data processing installation. We have found that records of intangibles, such as the time for the application analysis and problem definition, flow charting, coding, debugging, checkout, and finally documentation would have to be maintained continuously to have a base for predicting analyst/programmer costs and for personnel evaluation. In predicting costs of computer programs and evaluation, we would like to be able to have a magic number representing the proper number of analysts or programmers that could be applied to a given situation and for our center as a whole, but we found this to be impractical because we were measuring intangibles by subjective means. We found that attempting to save expenses by minimizing or restricting the availability of professional personnel caused equipment to be used ineffectively. The more the programmer is annoyed with accounting for his time and the more detailed the account for nonproductive time, the less apt he will be in cooperating in a program that keeps up with all of the various functions he performs.

Programming personnel at ORAU are engaged in numerous types of jobs; therefore, standards of work evaluation could not effectively reflect the variety of tasks they encounter. One programmer may be responsible for coding X number of instructions with relatively small amounts of logic development, while another programmer may be responsible for extensive logic development with relatively few instructions. Thus, it becomes difficult to measure the amount of work required on each program, and the total work effort performed by a programmer cannot be assessed in standards of comparison with another programmer.

At ORAU, we believe that the group leader of either the scientific section or the commercial section, depending on the particular area, should look at the problem in advance and then meet with the data processing manager to establish reasonable target dates for each of the previously-mentioned phases on the basis of the nature of the problem and on their past experience. The manager or group leader must have a detailed knowledge of the problem under study because it is his responsibility to prepare the cost estimate and schedule. He must keep up with the allocation of funds and judge progress of the application. After giving several methods of job measurement trial, we have found that there is no substitute for experience in the area of predicting costs, measuring work, and evaluating programming personnel. Programmers respect a supervisor who gives them a job and can tell them what performance is expected. The supervisor is also respected by his staff if he remembers that good supervision is the least supervision needed to get the job done.

*From Oak Ridge Associated Universities, Oak Ridge, Tennessee, under contract with the United States Atomic Energy Commission.

Programmer Evaluation in U. S. Reduction Co.

S. A. Lynch

U. S. Reduction Co. is a producer of secondary aluminum alloys. Annual sales are over $60,000,000. There are approximately 900 employees. Corporate headquarters and one plant are located in East Chicago, Indiana. There are four other plants in four other states.

Data-processing-services employees number six. In addition to a manager and assistant manager, there are two programmers, one operator and one key-puncher. In addition to these individuals, two other company employees are closely related; one functions as a senior systems analyst, the second as a special project researcher who does his own programming. Finally, use has been made of two outside programmers on a contract or hourly basis.

U. S. Reduction Co. uses two IBM 1130 Systems and some time on System 360/Models 20 and 30. An expanded 1130 configuration has been ordered.

Methods of evaluation of programmer-effectiveness are based on subjective criteria, augmented by certain measurable phenomena. These include:

a. Demonstrated dedication to task and cooperation during its fulfillment.

b. Speed of accomplishment of assigned tasks.

c. Feedback from users serviced by applications programmed by the programmer in question.

d. Feedback from IBM personnel dealing with the programmer on technical matters.

e. Extent and clarity of program documentation and general ease of implementation.

f. Infrequency of undefined error halts, accuracy of output, and speed of job execution.

g. Personal evaluation by the programmer during the semi-annual salary review interview.

h. In all of the above full recourse must be made to comparisons with the evaluator's personal experience as a programmer himself and with other programmers he has known.

Management Installation Division

PROGRAMMER EVALUATION

by

Dr. Paul S. Herwitz
IBM Corporation
Old Orchard Road
Armonk, New York 10504
(914) 765 - 4543

Thursday, 10:30 A.M.

Text, 12 pages

PROGRAMMER EVALUATION

Whenever the subject of Programmer Evaluation comes up in conversation, at meetings, and as the topic for panel discussion, I have the feeling that the participants are looking for easy answers. As an IBMer who has been in this business for a long time, I'll have to disappoint you if you expect me to give you an algorithm for judging the work that programmers do; because I don't believe one ever will exist. It's easy to understand that in a profession that's only 20 years old, in which the technology is in its infancy, in which 40% or 50% of the people working in it probably have been working in it for about two years, in which an operating system which was a new concept in 1962 is old-hat in 1967, in which an experienced professional is ancient if he's been in the business ten years, and in which a manager is highly experienced if he has five years of management -- it's easy to understand that most of us don't really know what to expect of the practitioners of this peculiar craft called programming. I'm disturbed by the idea some people seem to have that there may be a magic formula and some sort of mechanical procedure that produces an evaluation automatically without anyone's effort and judgment going into the process. You know, programmers are a clever lot; and if someone comes up with such a scheme, they'll find a way to beat it.

The evaluation of the work of professionals is just not an easy task. It's especially difficult in a profession that's as young as ours. To do the job properly requires a lot of experience, a lot of skill, -- and especially it requires a lot of energy on the part of the manager. It also requires an understanding of a company's philosophy -- a knowledge of what the company expects to accomplish by evaluating its people, and a knowledge of its standards. I think, also, that there is an opportunity now to improve the case for professionalism in programming. In an activity that's so new, -- which has been entered by so many people so recently -- in which the demand for practitioners far exceeds the supply -- it comes as no surprise that advancement has been a matter of expediency more often than we care to admit. This only works to the detriment of the profession. We expect the number of programmers and analysts to double in the next four or five years, and we expect it to triple in ten. It's time to take stock. The opportunity won't last forever.

As I have said, the evaluation of the professional's performance on the job is really very difficult. It's a continuing problem -- one with no pat solution, only guidelines. Now one of the guidelines I'm going to speak about later on is knowing what the job is. And this applies to the first line manager as well. This manager must know that evaluation is his job and he, himself, will be measured on how well it's done. This is something that's not well understood by new, young managers, and I think the reason lies partly in today's pressure environment and partly in the way we choose managers. Many of us say that the programmer lives in a state of constant crisis. That may be true, but I don't think it's a characteristic that applies only to the life a programmer leads. I think if you will look at any production-oriented business -- and I mean by that any activity in which you produce a product -- I think you'll find the same kind of perpetual crisis, or continuing pressure situation throughout the business world and industry today. Requirements are always changing, technology is constantly changing, and invention is made according to schedule. This means management is generally facing new technical problems every day. Combining this with the explosive growth that has taken place in the programming field, it's easy to understand the tendency to choose as a new manager the person in the group who has been technically most competent. Unfortunately, this doesn't mean that our new manager is the person most likely to succeed as a manager of people. Technical competence is, of course, a most necessary attribute of a manager, but I believe that in the long run the person who is really well oriented to the people who work for him will produce much more than the purely technically-oriented manager. The situation, of course, tends to be self-perpetuating. The new manager probably learns more from his manager than from anyone else. But the second line manager, himself, is so busy with the problems of meeting schedules and changing requirements that he is not likely to take the time to tutor his new subordinate in the personnel aspects of the job. Moreover, the second line manager probably lived in the same environment when he first became a manager and learned very little of this aspect of the job from his second line manager. Now this is a pretty vicious cycle, but it can be broken. Breaking it depends upon management in general realizing the value of doing the job properly. Just plain common sense tells us that there can be a real payoff.

In order to become a really top notch manager, a person must have considerable insight which usually comes only after very extensive experience -- job insight and insight into people -- what they're

capable of and what motivates them. In lieu of this extensive experience, the problem is -- how can we do a job which in some sense can be considered competent? Well, one thing we can do is attempt to give our managers better education in the form of instruction by his manager, and education in terms of formal training. I think most companies probably do a pretty good job of training new managers in various administrative aspects of their job, such as a knowledge of company policies, what procedures to follow, when it's necessary to requisition supplies, how to fill out the forms necessary to give a man a raise, and so on. But although most companies have forms to be used and to be filled out before and after an appraisal and counseling interview is held, since the evaluation process is so judgmental in nature, it's very difficult to tell anyone how to make the decisions necessary so that he <u>can</u> fill in the forms. My firm belief is that you learn this only by trying, by reviewing your decisions with your own manager and by trying again. And, of course, it's always extremely useful for a new first line manager to talk it over with other first line managers who have had more experience than he. At best, the only specific instructions that can be given can only be in the form of guidelines; and we will discuss a number of guidelines later.

Let's take a look, now, at the reasons for evaluating our people. In IBM if we ask why we evaluate programmers, the answer is easy to understand. And I think our reasons are pretty much the same as yours. Evaluation really has a triple payoff -- for the company, for the manager, and for the programmers.

If we look at the process from the company's viewpoint there are three important reasons. First, we want to improve our products and our services by finding an optimal deployment of our human resources. Second, we want to be certain that the salaries we pay are equable throughout all of our programming activities. And third, something that isn't often considered, IBM's attitude toward its programmers and toward the profession in general is bound to have a major impact on all programmers. So we feel that learning to do the job properly is a challenge and an opportunity to help the profession.

If we look at evaluation from the manager's viewpoint, the reasons are similar but more specific. First of all, the manager has to find the people most capable of doing the job he needs done. This means he must find the best way to utilize the people he has -- and then he must measure them against his requirements. If he is really a good manager (and he, himself, will be evaluated on this attribute), he will look beyond the immediate requirements of his own project and will decide whether or not some of his people can contribute more somewhere else. Secondly, since the responsibility for salary recommendations lies with the first line manager, and since increases are given for merit, the manager must decide who merits a raise, what the amount should be, and when the increase should be given. Finally, the manager must continually strive to improve the performance of the people who work for him. This means he must evaluate, he must instruct, and he must encourage his people. This is his opportunity to contribute to the profession.

The third part of the evaluation payoff comes when the programmer is told openly and honestly what the manager thinks of the programmer's work. All of us want answers to a lot of questions -- our morale depends on them. How do I stand? How good was the job I did? Where am I going? Will the jobs I do continue to be interesting -- to be challenging? Even if the answers to the questions are not what the programmer hopes for, his morale will be high if there is no subterfuge on the manager's part. It's amazing how hard people will work if criticism is constructive. And it's surprising how much they will learn if they are given timely instruction.

Now let's try to define some guidelines that will help us realize the benefits of a good evaluation program. The guidelines I shall propose are really a large number of questions that have to be asked and answered on a continuing basis. They will take quite a bit of the manager's time, skill, and energy to answer in any responsible fashion. And this is one reason why I constantly preach that a manager shouldn't have more than about eight programmers reporting to him -- six would be better.

We are trying to evaluate programmers in terms of their present performance and their potential for future assignments. We'll start with present performance.

First and foremost, I think, it's necessary to know what the job is that you expect the man to do. It's just as important that the man know the job you expect him to do. Once there is agreement on what the job is, I think half the battle is won. Moreover, one of the factors that one must use in the evaluation is the level of detail necessary in describing the job; and this must take into account the programmer's experience and position level.

Now it must be possible to measure how well the job is done. In order to do this it follows that some quantifying "job parameters" must be agreed upon. By job parameters I mean things such as the length of time necessary to complete the program, some indication of the size of the program, some indication of the performance expected of the program when complete, and some statement of the functional capability expected of the program. Remember, of course, that it may not be possible to state such parameters when the program is first conceived; and so the job assignment may very well be to study the proposed program and arrive at such parameters.

Now depending upon the length of time needed to complete the program, it may be necessary to state some parameters that will act as checkpoints. After all, it's easy enough to keep track of a programmer who's working on a 1-week assignment, but it may be extremely difficult to keep track of the work a programmer does when his assignment takes him 2 or 3 months or more. In the latter case, it may be necessary to block out the job into sub-jobs, each of which has some sort of a time checkpoint and appropriate functional specifications, and so on. Checkpoints generally call for a review. This, of course, can be a very informal thing or it can be quite a formal undertaking if several programmers are involved in projects which culminate in a common checkpoint. But these checkpoints are necessary not only to see that the project progresses properly but also in terms of constant evaluation of the people doing the work. Once the job is complete, of course, the other half of the battle begins; and that is the assessment of whether or not the job was completed as expected. It's not difficult to know whether a person completed the job according to the required schedule or not; but if he didn't, what are the circumstances surrounding this? Was he permitted to do the job without interference? Were the specifications changed during the middle of the job? Does this program interface with other programs that changed and, therefore, required this program to be changed?

Again, it's easy to measure the size of the finished program against the agreed-upon size. But if the program is too large, we have to ask whether the circumstances are extenuating or not. Was the projected size based upon knowledge of similar routines which had been done before for other machines or other circumstances? Should there have been a direct carry-over from the previous program to this? If this program is really a brand new program, was the amount of work underestimated? Were additional functional capabilities included in the program beyond the functional specifications? Again, is the size of the finished program considerably less than projected when the job was first described? If so, is anything missing? Or was it just easier than we expected? Or did we discover new programming techniques? Similarly, was the job completed ahead of time? If so, was the estimate over-conservative? And if so was that the manager's fault or the programmer's fault?

How we measure the performance of the program is probably one of the most difficult questions we could ask, particularly if the program has any magnitude. But at least we can compare the performance to other programs with similar functional specifications if such programs exist. And the likelihood is that they do. Does this program perform better or worse than the old program? In either case, why?

Finally, in regard to functional specifications, does the completed program really have all the functional capabilities specified? If not, why is some capability missing? Was the feasibility of such a capability not proved? Or was it something not understood by the programmer in the first place? Again, if there is more functional capability in the program than asked for, what did we pay for this additional capability? Did we get it free in terms of schedule, size oı program, and performance? Probably not. If not, was the price worth paying? One begins to see that these are not easy questions to answer. And, therefore, it should be obvious that when I said evaluation was a difficult and time-consuming job, I really meant it.

Once the initial assessment in terms of job parameters is made, we have to undertake an additional evaluation which is probably even more difficult. What is the quality of the work? We have already addressed most of the external aspects of quality; that is, whether or not the program met its size, performance, and functional specifications. One other external aspect is the correctness of the program. How well has this program been debugged? Of course, if the program came in on time, this means that it was debugged on time. Whether it came in on time or not, a good manager will ask what are the nature of the errors that did turn up? What was the debugging plan? Did the debugging plan take all reasonable contingencies into consideration? And I guess, finally, would a better debugging plan have produced the desired results in a shorter period of time?

Of course, to answer these questions, the manager must be pretty familiar with the written code. As a matter of fact, I don't know any way to judge the internal quality of a program without becoming thoroughly familiar with the program itself. This, of course, implies, too, that a manager can only make a quality judgment based on his own experience. Another factor regarding the internal quality of a program is the quality of the remarks that document the program. Are these remarks succinct and to the point? Do they appear where they are needed and not where they are redundant? Another factor one must look at is does the program use proven techniques, or is it innovative? I think that nine times out of ten the use of proven techniques will get the desired result with much less heartache than will innovation. This is not to say there is no place for innovation because there is. Again, though, it's a judgmental factor as to when one should be innovative. Finally, is the program elegant? Now how do you answer that one? Well, I suppose you look for simplicity; you look for coding techniques that save time, or produce more function, and that are well documented; and you ask whether the job is workmanlike. Are all the steps there, do they lead logically from one to another to arrive at the desired result in the shortest possible time? An elegant program to me is one which is characterized by simplicity, lucidity, novelty, and good workmanship.

Before I talk about potential for future assignments, I would like to pause a moment and look back over what I've said. At this point the manager has evaluated a programmer's work with respect to a particular job assignment; that is, he has determined whether the job came in on time, that it met functional, performance, and size specifications, and that it was of a certain quality. So now the manager should be able to answer two questions. First of all -- does the programmer meet the requirements for this job? If the answer is yes, then the second question must be -- can he do a better job and is there something more challenging for him to do? If the answer to the first question was no, if the job requirements were not met, then the manager must ask -- why not? And here he must make a real value judgment -- was the programmer over his head, or was he not working up to his capabilities? If he was over his head, can he be taught? Or should he go elsewhere? And probably the most difficult question -- if he was not working up to his capabilities, what are the chances that he will next time? If the chances are poor, the manager has a real problem. If there were extenuating personal circumstances or job-related circumstances, what can the manager do to help alleviate these? Finally, if there are personal problems, the manager had best stay out. If there are job-oriented problems, then the manager has a responsibility to try to correct the situation.

Now let's look at the potential for the future. I'm going to assume that the man being evaluated has performed satisfactorily in his present assignment and that the assignment has been challenging. Otherwise, he has either not met the job requirements or has not shown any potential for promotion or added responsibility. But if his last performance has been acceptable, then we have to look to the future. First, I guess, we would ask how broad is the man's experience? And here we get into the old argument of whether a person should be a generalist or a specialist. The argument, of course, is that the specialist can produce a given program without having to reinvent the wheel. My concern is that our people have specialized without having been exposed to the broad basic fundamentals of programming as a whole. This, of course, is due to the nature of the rapid growth in programming, but it doesn't really excuse the situation. There is no question that this is an age of specialization, but specialization usually comes after extensive schooling in the broad fundamentals of the discipline. Specialization without a broad base to start from is really a pretty unhealthy situation. Here again is one of the opportunities that we have to improve the profession as a whole. And I really don't think we will be in good shape until the colleges and universities have developed well-rounded curricula in the computing sciences. Whenever possible, I think it's best to attempt to give our programmer some breadth of experience.

Directly related with the question of how broad this man's experience is is the question -- how up to date is his technical knowledge? It's very easy for a man who has spent one or two years in a high pressure project to forget some of the experiences he had in previous projects. This, again, I think is due to the lack of a well-rounded initial education. If a man has a good base to begin with, he is less likely to completely forget what he has learned in an area he is not working in currently than if he has had only brief experiences in several different areas. Several people have suggested that at least part of the evaluation of a programmer be based upon a well-defined skills inventory. I suspect this is a rather difficult thing to construct, but it would be interesting to try. Remember, however, that this is not a panacea. It is only one of the tools that a manager must draw upon in evaluating one of his people.

Another characteristic we must try to evaluate is the man's technical foresight. How well does he foresee problems? Is he able to avoid the problems, or bring them to his management's attention early enough so that appropriate steps may be taken? Or does he evade the problems? Another attribute to consider is the man's problem-solving ability. This is always considered a key characteristic in the search for new programming talent. But how do we evaluate it? That's a very difficult thing to do without seeing him in operation. I expect, though, that watching the man in action gives a pretty good clue as to his ability to face problems by himself. And I think that's what we really mean. Does he solve the problem himself, or does he need to come to his management for help? A corollary question, does he come to his management for help when this is really necessary -- when the problem is outside of his range of experience? This leads directly to another characteristic, his judgment and decision-making ability. When the man is faced with alternatives, how often does he make the right choice? Can he in some fashion characterize the probabilities of success for the various alternatives? Is his judgment good -- does he make the right choice more often than not?

The next area that we have to evaluate is the man's self-knowledge and his self-improvement activities. Does he understand his own limitations and his own talents? What is he doing to remove his limitations? Does he have his own plan, or must he be led? Remember that one way to improve one's capabilities and one's experiences is to constantly seek out new and broadening assignments. Does he read the literature of the field? Does he go to seminars? Does he go to meetings and bring back new ideas? Just how wide are his horizons?

One of my pet peeves is the man who cannot communicate. Over and over again I've run into people who cannot write clearly, cannot speak clearly, and it's questionable as to how clearly they think. Although many programmers tend to be reserved and introverted, this is not true of all of them by any means. You might think if a programming shop or a programming specialty is a one-man operation in a particular installation, that it doesn't matter whether he can communicate or not. Well, nothing could be farther from the truth. If a programmer is to be successful in working in a team, he has got to be able to communicate both orally and in writing. Particularly, if a man is going to be an analyst of any sort, he should be able to communicate clearly with the people he has to deal with in attempting to offer his services. So many times I've heard people who are not computer-oriented say that they would love to use the facilities of a computing center but they can't communicate with the analysts or the programmers. I think it's up to us to see that we, as programmers, are able to provide the right kind of information and to show the uninitiated how our services might be used. And it does no good to do this with a chip on the shoulder. So one of the major questions that I would ask is how well does the programmer communicate with his peers, his management, and with novices and the uninitiated? I can't emphasize strongly enough that I think the man who can't communicate, although he may be a fine programmer, is a man whose usefulness is highly specialized.

Now let's consider what the programmer has done for the profession. Has he had an opportunity to make a contribution? What has he done within his company? Has he made any technical advances? Does he make suggestions within his own area -- timely suggestions for new programs? Does he see ways to simplify what's being done? Does he attend seminars? Does he write papers and make presentations? Does he go to computer meetings? What kind of contacts has he made? Has he met people whom he can stimulate intellectually and who can stimulate him intellectually? If he makes presentations, does he have something to say? How well is he respected within his own area, within his company, and by others in the profession outside of his company? I grant you that with so many people who are so new in the profession, not a lot have been able to make contributions. But I think again that it's our responsibility to stimulate our people into contributing. And not on a haphazard basis either. They need the guidance of those of us who have had experience.

Now I want to discuss one other item that needs consideration both from the standpoint of present performance and potential for future assignments. This is one which most of us don't like to talk about too much, but it's something that's extremely important in this day and age. That's the question of cost control. Primarily, we have to ask how well our programmer has utilized machine time. This, of course, ties in with his success in debugging. But it's of utmost importance today because it's a major contributing factor to the cost of production of programs. The main question is could the man have used less machine time to debug his program? Three other cost factors are the use of supplies, the use of travel, and the use of overtime. Although the question of overtime is more likely

something that a manager will be judged on, unless a project has been directed to go into overtime, a person's ability to do his work during regular hours is one that should be looked at. I guess if overtime is necessary in a particular instance, we have to go back to the question of whether or not the job was completed on time. Of course, there are always extenuating circumstances such as illness, absence for other personal reasons, etc.

I'm certain that all of you can think of many other questions that one might ask in the evaluation of a programmer. I don't claim to be all inclusive -- I'm just trying to suggest the general areas that I think are important to evaluate. Of course, remember also that most of the questions must be answered in light of the position a man holds. We ask much less of the newcomer than we do of the oldtimer. We ask considerably less of the junior programmer than we do of the senior programmer. As a matter of fact, this implies that position descriptions exist and that they're well written, that they're understood, and that they're generally applicable to the work being done. Remember also that good position descriptions will address many of the areas in which I have asked questions and hopefully will give some guidance as to what is expected of the person in the particular job. A good position description can be one of the manager's valuable aids for evaluation. Good position descriptions also should address promotability. I think there should be well defined criteria for promotion with indications as to what criteria may be sidestepped and under what circumstances. I'm certain most of you have seen cases where a programmer was not promoted in one department, was then transferred, and shortly thereafter promoted in another department. At the very least this suggests that the two areas were not looking at the same criteria for promotion. This is an extremely unfortunate situation and one that's hard to handle.

I guess I can't tell you what kind of answers you need to these questions to be able to say that a programmer is outstanding or merely meets requirements or always exceeds requirements or does a very poor job. Standards are going to vary from company to company. I think most of us will find that we don't even have standards within our own company. So we really have a job here in providing the standards applicable to our own situation. There is a lot of work involved; but, as I indicated earlier, we think it's a challenge and an opportunity.

Let's remember another thing, too. The manager who looks constantly at all the attributes and all the characteristics of the programmer he is evaluating and who instructs him and who criticizes him constructively all the time is a paragon that just doesn't exist. If most of us can look at many of these questions on a relatively constant basis and can do our best to instruct and criticize constructively, then I think that we're doing quite a remarkable job. And I guess in closing I finally want to remind you that when you try to answer many of these questions and when you try to set up your own standards, don't forget the man who is given an impossible task. We need a hero medal -- second class.

ADDRESS BY

MR. G. W. WOERNER, JR.
VICE-PRESIDENT & REGIONAL MANAGER
MIDWESTERN REGION
I.B.M. CORP.

COMMON Meeting
September 7, 1967
Cincinnati, Ohio

Address by G. W. Woerner, Jr. - Vice-President & Regional Manager - MWR

Thanks for inviting me to your meeting and to the very fine luncheon this noon.

I must say it has been a real pleasure and I have been tremendously impressed by the pace of your program and the am'·tious goals which you have set for yourselves here in Cincinnati.

I am sure you must know how greatly IBM values the working relationship which has evolved over the years between our people and your organization. There is absolutely no substitute for the direct insight into your individual and collective problems which can be gained through this relationship.

With the complexity of hardware, software and applications increasing at such astounding rates, we need your advice, counsel and constructive criticism badly. It provides us with a fundamental reference point from which to develop our plans and focus our resources on areas that best meet your needs. Incidentally, you may find it interesting that two particular subjects are coming through loud and clear.

The first concerns the general area of our method of providing you with manuals and technical information.

The second question concerns the various types and levels of supporting education required from IBM.

I can assure you that we are addressing these two particular points with exceptional urgency as a direct result of the emphasis which this organization has placed on them.

As I observe the many contributions your organization has made to its members, I am especially impressed with the quality and quantity of the Type IV Library. Today it contains over 700 programs written by members of the COMMON Organization. Your massive contribution to the 1620 Library has been of significant value to your own members, to IBM and to the state of the art as a whole. This experience should indicate the value of now turning your efforts to the equally productive Models 20, 1130, and 1800 to build a similar library for your own use of these systems.

There are an awful lot of people these days sitting back and watching us in the computing industry. In fact, it has become a popular pastime to comment upon the effects of computing and automation on our society.

We, as IBM'ers, are especially interested in these commentaries. And I, for one, have been particularly concerned about one dimension of the industry which has been getting a lot of attention lately and I would like to spend the remaining time I have with you this noon talking about it.

It has to do with the broad conclusion reached by a number of contemporary writers that automation and, more specifically, the computer constitute a major threat to the individual in our society. These writers point out specific instances where automation has unfavorably impacted the individual and then conclude categorically that these instances constitute a trend at the end of which is the complete extinction of the individual in tomorrow's society. And, that we are destined to live in a faceless world with standardized values, regimented routines of life and mechanical modes of thought. More often than not, the computer is used as a symbol of that threat.

I guess the whole thought stream got started as early as the mid '30's with Aldous Huxley's "Brave New World". He pictured a technological society living in doped-up bliss under the watchful eye of a powerful tyrant.

George Orwell depicted a similar grim scene of life under the ever-present electronic eye and ear of Big Brother.

Today, writers such as Galbraith and Vance Packard are further soothsayers of the impending doom. Packard, in his book "The Naked Society", says, "In the Western World today, swirling forces are causing whole populations willy-nilly to change their attitudes, ideals and behavior patterns". One of these forces he describes as "the electronic eyes, ears and memories".

Supplementing these more renown writers are self-appointed investigators who use the public media to broadcast sensational exposes of how man today is losing the battle with automation.

Most often they take some form of the theme of a man's personal identity being replaced by a number of holes in a punched card.

They usually conclude that these mysterious forces are somehow thrusting men pell mell toward a conformative life of grim, mushy blahdom overseen by the sterile surveillance of the master machine.

Before we rush to take a position for or against these observations, let's pause to define what it really is that we're afraid of.

Psychologists tell us that each of us has a compelling need to establish his own personal worth, and that a key part of that need is identity as an individual.

I would define individuality as the desire of each man in our society to direct his own destiny. To set himself apart from the rest of society as a person who has unique tastes, thoughts and goals. He needs to have the prerogative of living as he wants and developing himself toward ends which he has defined. He doesn't want society or governments or employers or any other group (much less machines) to plan and manage his life for him. He wants the freedom of choice which allows him to select the kind of food, clothing, shelter, education, employment and leisure which suits him and his family best. The uniqueness of this pattern which he chooses identifies him as an individual and establishes his identity.

What we're afraid of then is that our society is becoming so organized that the above choices will not be ours, but will be provided according to some kind of plan. We're afraid that those in seats of power are developing some plan based upon statistical models and norms describing what each of us should want and need.

We're afraid of the similarity of our housing, the food in the supermarkets, the clothes on the racks, the cars on the lots, the standardization of our educations -- and the impersonal treatment by service organizations.

In short, we're afraid because the patterns we see developing across our society appear to be a challenge to our compelling need for individuality and personal identity.

I must join in these observations and agree that many of these trends are, in fact, in motion. I can't agree, however, with the conclusion that automation, or computers, have been the cause of these trends nor do they constitute the current threat.

Computers and their use are a result rather than a cause. A result of the very same basic force which is responsible for the threat to our individuality. That force stems from the historic insistence on the part of the "have nots" to join the "haves". Whether they be the poor or the emerging nation reaching for food and shelter, or the middle class stretching for higher living standards, they represent a constantly growing demand for a higher standard of living for more people.

Man's success in meeting these demands, many times with the help of machines but sometimes without, has resulted in a society geared to volume rather than to individual needs.

Now, whether we look at consumer products, human rights, education or personal services, this demand and response goes through three stages.

In the first stage -- the benefit is only available to a select portion of the population. They are the "haves". They have an exclusive, if you will, on the benefit -- and it uniquely distinguishes them from their fellows.

Stage two is a natural follow-on -- the "have nots" demand the benefit and society's ingenuity to supply it is motivated. The result is that more and more "have nots" become "haves". But in doing so, the benefit loses a big ingredient -- its exclusiveness and thus, its individuality to the possessor.

The third stage then is, or should be at least, an attempt to stylize or expand the benefit into unique subsets so that even after mass production and distribution, the benefit regains what it lost in stage two --------- an appeal to the individual.

Let me illustrate with some examples --

In the early 1900's every automobile was designed specially for the buyer and then was hand made. This was the first stage. You can observe several things about it --

1. Only a few could be produced.

2. Only a very small percent of the population could afford them.

3. Ownership of custom automobiles was certainly a prestigious and individualistic thing.

Within a few years, however, pressure developed from those who did not have this luxury and we enter stage two -- as a response to the demand.....Henry Ford's ingenuity created the technique of mass production and he cranked out huge quantities or one model.....the Model "A" at a price that more people could afford, and in the color they wanted.....as long as it was black.

At this point, Ford satisfied the demand of the masses but he lost something in the process. He sacrificed the idea of a "custom" car designed and built specifically for an individual. The customer no longer had a car which represented his individual tastes and distinguished him from his neighbors. All had black Model "A's".

But the story doesn't end here -- rather it moves to a third stage and this is the key point. Today we can select an automobile from an impressive array of models, styles, color variations, and features which suit our individual tastes. In fact, that myriad of options and combinations is so great that every single car produced by Ford alone this year could have been different from all others without going outside the standard features.

What brought on stage three? What accomplished this return to the custom car? A demand for individuality responded to by more sophisticated techniques of automation, aided in no small way, by the computer.

Let's take another example to illustrate the point.....education.

Stage one.....in the good old days -- the right to education was enjoyed only by the few, primarily royalty -- in fact, as late as 1900, only 94,000 people in the United States graduated from high school.

Today we consider education to be the birth right of every man, woman, and child -- not just royalty.

The result is the phenomenon realized in stage two -- a high school education is a standard for the majority of our population and over a half million students are receiving college degrees each year.

However, companion with this has been the depersonalized multiversity with campuses swarming with 20 to 30,000 students.

- Overcrowded classrooms - where an instructor addresses a hall with upwards of 2 to 300 students.....

and

- Standardized curricula - where the pace of instruction and the content is geared to the average rather than to the individual.

We readily see that in our attempt to satiate the demand, we've already sacrificed the small classroom environment, the tailored pace of instruction, the personal guidance and touch of the teacher, and to a great degree, the ability of the individual student to pursue courses of study which are to his particular interest.

Interestingly enough, in this example one can't even associate the move to the second stage with automation. For we accomplished both the response to the demand of the many and the loss of individuality without it.

Stage three is still in its infancy. Significant developments are already on the scene in the form of computer aided instruction. A system is now available and in use which allows a student with his own visual display terminal to learn with the computer in a conversational mode.....the process is personalized to the extent that the material is presented and dealt with at the learning rate of the student. The human touch is preserved by the teacher who, upon being relieved of the mechanics of presenting volumes of material, has time to devote herself to that student who is having problems or needs counseling.

These techniques show promise of moving education solidly into stage three. A stage which satisfies the demand for higher education for more and more people, and yet preserves the integrity of the individuals.

I myself have explored many other examples and I know each of you could come up with many of your own. But these two suffice to illustrate my premise. Let me review.....

First, there are today, trends in our society which <u>are</u> a continuing threat to each of us as we pursue our personal identity and individuality.

Second, these trends do <u>not</u> stem from the use of automation or the computer, but rather are a by-product of man's organized response to the steadily increasing demand of the many for the benefits enjoyed by the few.

And finally, the computer has demonstrated its capability to contribute to the development of the last stage of these trends.....the pursuit of our individuality.

The advance of technology and automation cannot, and should not, be stopped -- for it is not the adversary of the individual -- but the ally.

So I would say to you as members of this organization.....as key participants in the field of automation.....we must be prepared to proceed with confidence -- for there is no shame in progress -- our efforts are <u>not</u> a betrayal of <u>ourselves</u>, but rather an expansion of our opportunity.

And this point must be made to our critics.

In fact, you might want to think about <u>adding</u> this one to the objectives of your group.

I'd like to thank you again for the opportunity of joining with you today and wish all of you a very successful conclusion to your meeting.

########

SESSION NUMBER T.3.1.

SPEAKERS

DAN FULLAN (IBM) - PRESENTATION ON DOS III .
J. ALVAREZ & W. SELSMEYER - QUESTION & ANSWER SESSION ON DOS.
MODERATED BY D.R. MC ILVAIN

DISCUSSION

DOS III (ACTUALLY A NEW RELEASE OF DOS II, NOT FORMALLY NAMED DOS III) - SEE ATTACHED WRITE-UP. THIS IS SCHEDULED FOR RELEASE IN SMALL PART 11/67 & COMPLETE 4/68. QUESTION & ANSWER SESSION FOLLOWED.

SYSTEM/360

DISK OPERATING SYSTEM

IMPROVEMENTS

Scheduled Improvements to DOS

I would like to spend the next few minutes or so discussing some significant improvements to the Disk Operating System. These improvements which will be available with subsequent releases of DOS include Additional Device Support, namely the 2314, Additional Features in the Supervisor to improve performance, for example, seek separation, Increased Capabilities, such as expanded multiprogramming capabilities, and a number of related features to simplify operations, for example, label handling. Let's go over these now in a little more detail.

2314 Support

The 2314 direct access storage facility has received wide acceptance in the intermediate systems marketplace. The programming support provided for this device reflects this acceptance. Full system support will be provided for the 2314. This includes system residence, that is, residence for all IBM components currently supplied in DOS/360, except Autotest, systems input and output on the 2314, full systems libraries support, full data management support including QTAM, Sequential Access, Direct Access, Index Sequential Access, and Device Independent Access. All current DASD utility functions will include support for the 2314. Additional support in the form of new special purpose utility programs will be provided for both the 2314 and the 2311. These utilities include a Volume Table Of Contents Display, an Initialize Disk Program, an Assign Alternate Track Program, a program to copy Disk to Disk, and programs to Copy and Restore Disk with Tape and Cards. Full language support is provided for both compilation and object time support. This includes the Assembler, COBOL, Basic FORTRAN and RPG. We intend to provide PL/I and Sort/Merge support. Information will be provided at a later date. All support will take advantage of the increased capacity of the 2314 as well as the speed of the unit itself.

Simplified Label Handling

We have modified the labeling procedures in DOS to provide for a simpler more efficient operating environment. These procedures will involve the use of fewer cards and label cards that are reusable. The formats have been improved and more room has been provided for standard labels. In addition, standard labels can now be used for Index Sequential and Direct Access files.

Two types of standard labels will be provided. Standard labels will be available to programs operating in all three partitions while partition standard labels will be available to programs in the specified partition only. A total of six tracks for user labels and partition standard labels are available. On the 2311 four additional tracks and on the 2314 fourteen additional tracks will be provided for standard labels. We have then the possibility of labels appearing in one of three parts of the label cylinder. It may appear as a user label associated with the specified partition, as a standard label associated with that partition, or as a standard label available to all partitions.

The increased capacity for standard labels and the ability to use standard labels for direct access and index sequential files should greatly reduce the number of label cards and handling required in an operating environment.

The number of cards required for label handling has been reduced. We have a DSKL card which replaces the VOL and DLAB cards required today. The parameters for this card may be expressed in a variable format, some of the parameters have been made optional. Date need no longer be expressed as absolute creation and expiration dates, a retention period may be substituted. This will enable the DSKL card to be reusable, that is, once label cards have been set up for a production run, the same cards may be used each time the run is executed.

A new EXTNT card has been provided. In this card the upper and lower limits of the extents do not have to be expressed in absolute form. You may specify relative track, that is, the sequential number of tracks relative to zero where the extent is to begin and the number of tracks the extent is to contain. On input files the information that is supplied by the extent cards can also be taken from the label itself so that in many cases an extent card will not be required.

Data sets on tape are defined by means of a new TLAB control card which replaces the current VOL and TPLAB cards. Here again a retention period may be specified instead of an expiration date. When a volume label is not found on a labeled output file, it will automatically be created based upon a serial number supplied by the operator. A multi-volume file may be opened at other than the first volume.

To summarize the label processing capabilities of DOS, the number of cards required for disk labels has been reduced from three to two and in some cases only a single card will be required. A single card will now suffice for tape labels instead of the two cards previously required. In addition, since retention periods can be specified these cards are reusable, that is, new label cards need not be punched on a daily basis. The expansion of the standard label facilities both for the number of standard labels that can be stored and the types of files that can use standard labels provides for an even further reduction in the label cards required in an operating environment. I am sure that those of you involved in day to day operations will appreciate the reduced burden on operations personnel made possible by these improvements.

Seek Separation

The seek separation feature was designed to improve the performance of systems running under DOS. This feature, provided by a supervisor option, enables the supervisor to separate a seek from its associated read or write so that the seek can be separately scheduled. This means that multiple seeks can be issued to devices on a channel and the reads and writes scheduled as the seeks have been completed. As this is a supervisor function, it will automatically apply to programs written at any language level and operating in all three partitions. The benefits derived from this feature will increase the larger the number of devices on a channel. The implementation of this feature is such that when a seek has been issued to a device the arm cannot again be shifted until the I/O operation that initiated the seek has been completed. In other words arm stealing has been prevented. Following the issuance of a seek the channel is available for scheduling other I/O operations. In a multiprogramming environment this feature is particularly important where the different partitions have a mix of input/output requests for a single channel with multiple direct access devices.

The seek separation feature is specified at Systems Generation time and will add approximately 200 bytes to the size of the supervisor plus 4 bytes per direct access device.

Full Track Add

We have improved the efficiency of the index sequential access method when adding records to the file by utilizing a full track add feature. This feature enables the user to provide room in core for up to as many physical records as can be contained on one full track. When a new record is added to an index sequential file that record is inserted in sequence and records that have a higher key are shifted to make room for it. The highest record on that track will then be placed in the overflow area. Without the core data feature when a logical record is added to a block each block on that track must be read, the records internally shifted, and then written back to disk. When the core data feature is specified, two or more physical records up to a full track will be read at one time. The records shifted internally and then written out together. For example, when room in core is provided for a full track, it will take 4 1/2 revolutions or less to add a record to that track. While without the feature, 3 revolutions will be required for each physical record on that track. Let us suppose a blocking factor of 4 physical records per track. When adding records to that track using the core data feature for a full track add, 4 1/2 revolutions are required compared to 12 revolutions without the feature or approximately a 300% improvement. To utilize this feature, an I/O size parameter must be specified in the DTFIS to reserve room in core for the additional data area. Core data equals yes must be specified in the index sequential module itself. The additional core requirements for this feature are 16 bytes per DTF and 175 bytes in the index sequential module itself.

Cylinder Index

We will also provide the capability to maintain all or part of the cylinder index in core. This will have a significant impact in efficiency during random retrieval, since the seek and read time of the index itself can be completely eliminated. This facility is achieved by specifying the name and size of the area reserved for the cylinder index in the DTF and specifying core index equals yes in the index sequential module. Where there is not sufficient room in core for the entire index it is possible to make only a portion of it core resident.

The core requirements to use the cylinder index in core feature are minimal. An additional 24 bytes are required for the DTF plus 150 bytes in the index sequential module. This of course is in addition to the core required for the index itself.

Both the full track add and the cylinder index feature are available only in the Assembler language.

Multiprogramming

In the area of multiprogramming we have made a number of improvements to programs operating in foreground partitions. In the current system foreground programs are initiated by the operator individually by a foreground initiator. We refer to this as Single Programming Initiation (SPI). Included in the improvements for multiprogramming is another method of initiating foreground partitions called Batch Job Foreground (BJF). This option will allow a batch job stream to be executed in the foreground partitions. Other multiprogramming improvements include individual communication regions, checkpoint/restart and the use of system logical units in the foreground partition. These features allow for execution of all user programs regardless of source language in the foreground partition operating under BJF.

Let's examine foreground batch job capabilities. In addition to the background partition, one or both foreground partitions can be operated in the batch job mode if sufficient Input/Output and CPU facilities are available. The requirements for operating foreground programs in batch job mode are a minimum 10K partition size, separate systems input and output files for the partition and the specification at system generation time for batch job program support. The batch job foreground option will add approximately 350 bytes to the basic size of multiprogramming supervisor. This includes the storage requirements for a communications region for each of the foreground partitions. If disk system input and output is desired for the foreground, an additional 250 bytes or a total of approximately 600 bytes will be required in the supervisor. All system class logical units except SYSLNK will be usable by foreground partitions regardless of whether they are operated in the batch job or single program initiator mode. However, if the single program initiator is used then SYSRDR, SYSIPT, SYSPCH, SYSLST must be assigned to unit record devices. The IBM supplied utility programs will be distributed to run in the background area but may be link edited by the user to operate in the foreground with batch job initiation. No other types of IBM supplied programs are intended to be available to the foreground partitions.

Let's take a look at some of the environments that are possible with the Disk Operating System. In a 16K system we can run a batch job stream, where 6K is available to the supervisor and 10K available to background programming. One restriction in this environment is the inability to compile COBOL programs since the COBOL compiler requires a minimum of 14K.

In a 24K system with a supervisor that runs between 8 and 10K, we can have a background program of 10K plus up to two foreground partitions operating with a single program initiator.

At the 32K level besides operating a 10K background program, we can have one foreground program operating in the batch job mode and a second foreground program operating with a single program initiator.

At the 64K level we can have three programs operating in a batch job environment, background, foreground 1 and foreground 2. The supervisor requirements would be at least 10K, with 10K minimum requirements for background and foreground programs. The additional 24K that I have shown available for the background could be distributed amongst all three partitions.

Let's summarize the improvements in the multiprogramming capabilities of DOS. I have mentioned the features available for background programs, foreground programs operating with the batch job initiation, and foreground programs operating under the single program initiator. Program initiation itself is automatic when working from a batch job stream for both background and foreground programs. Under the SPI it is still operator initiated. Individual communication region will be provided for each foreground program operating under the batch job mode. The Checkpoint/Restart facilities now available to background programs will also be available to foreground programs operating under the batch job mode. The systems service programs such as the librarian and linkage editor, the Language Processors and Sort/Merge can be executed only in the background partition. Utility programs will be distributed for execution in the background partition but can be link edited by the user for execution in the foreground where a batch job mode has been specified. The multiprogram system utility macros as today will be available in all partitions. Minimum partitions sizes are for background program 10K, 14K if you want to compile COBOL programs, 10K for batch job initiation in the foreground and 2K when the single program initiator is used for foreground programs.

Device Independence

There will be a new device independent access method for systems units. This access method DTFDI and the device independent module will support sequential processing of unblocked records for files on SYSLST, SYSIPT, SYSPCH, and SYSRDR. Using this new access method the user can change device assignments at object time without having to reassemble source program and without explicit knowledge of the functional characteristics of the assigned device. Besides providing device independence at object time the use of the device independent module will reduce core storage requirements since one module can be used instead of several as are required today.

Private Libraries

We are adding a new facility to DOS to provide for a librarian function that will allow users to create and use private source and private relocatable libraries on packs other than the system resident pack. This new facility will give the user greater flexibility in allocating library space with the result that more room on the system pack itself will be available for an expanded core image library. The number of private libraries is not limited, as many private libraries as desired may be created. However, only one of private source and/or relocatable library can be operative on the system at any one time. It is also possible to use the relocatable and source libraries on the systems resident pack in combination with private relocatable and source libraries. In addition, it is possible to eliminate the source and relocatable libraries on the systems residence pack. Library maintenance and service functions will apply to these private libraries.

CSERV

We have added a new librarian service function called CSERV. This function will enable the user to display, punch, or display and punch a specified phase or complete programs from the core image library. This facility will give the user the ability to transfer programs or phases from the core image library of one systems resident pack to the core image library of another systems resident pack. Since SYSPCH may be assigned to tape or disk an intermediate card step will not be necessary. The display, the punch, and display punch functions can be used for a phase, a program, or the entire library.

New Resident Utility Programs

We have provided in addition to the existing utility programs six additional utility programs resident under DOS.

The VTOC display program will enable the user to display the labels contained in the volume table of contents of a disk pack. This will enable him to more easily keep track of his files and their extents. The output of the VTOC program may be directed to a printer, a tape file or a disk pack. Labels are identified by their locations within the VTOC and their format types. There are no special utility modifier cards required for this program. All that is necessary to execute the VTOC program will be a job card, assign cards, and an execute card.

Another utility program is the initialize disk program. This program prepares the disk pack for use on the 2311 or 2314 disk drive. The initialize disk program first checks the volume table of contents to verify that there are no unexpired files on the pack. It then generates home addresses for each track, does a surface analysis assigning alternate tracks when required and pre-formats the volume table of contents.

The alternate track assignment program is used to assign alternate tracks to replace defective tracks on the 2311 or 2314 at any time other than when initializing the pack. When an alternate track is assigned the records contained on the defective track may transfer to the alternate track. Full diagnostics are also provided.

Three separate programs are provided to copy and restore disk packs. These are copy and restore disk with card, copy and restore disk with tape, and copy disk to disk. These copy programs may be used to copy the entire volume or just specified files. When the copy volume function is used the entire contents of the volume including the IPL records, volume labels, and the VTOC will be copied. The copy file function permits the transferring of a data file from disk to cards, tape, or another disk pack.

Availability

There is one more important subject to be covered. That is the availability of these improved facilities of the Disk Operating System. The improvements in the tape label area will be available from the Program Information Department November 17, 1967. The rest of the improvements that I have discussed for the Disk Operating System will be available April 5, 1968.

SESSION NUMBER T.3.2.

SPEAKERS

NO FORMAL SPEAKERS. WADE NORTON, ACT'G. CHAIRMAN, HAD BEEN SCHEDULED TO PRESENT A PAPER ON RUST ENGINEERING CO.'S. 16K OS WHICH RUNS ON A 65K MACHINE. THE OS PROJECT FELT THAT MOST OF THEIR MEMBERS ALREADY KNEW THE ADVANTAGES, AND THAT THE PRESENTATION COULD BETTER BE MADE AT A LATER MEETING WHEN MORE PUBLICITY COULD ATTRACT MORE DOS' ERS W/ OR PLANNING 65K CORE.

DISCUSSION

WE THEN DISCUSSED SAN FRACISCO (OR LATER MEETINGS). THE NEXT MEETING NEEDS TO COVER

A. LINK EDIT (I), SCHEDULER (I)
B. LINK EDIT (II), SCHEDULER (II)
C. OS FOR THE USER OF SMALL 360'S, PRESENTATION OF PAPER ON RUST'S SYSTEM & PANEL OF OTHER SMALL OS'ES.

ON RUST'S SYSTEM & PANEL OF OTHER SMALL OS'ES.

D. PLANS & ORGANIZATION SESSION.
E. SQUAWK SESSION.

SESSION NUMBER T.3.3
SPEAKERS

HARISH J. JAGTIANI, RICHMOND ENGINEERING CO., INC. ON FORTRAN PROCESSOR FOR DRILLING TUBESHEETS ON NC MACHINE

FORTRAN PROCESSOR FOR DRILLING TUBESHEETS ON NC MACHINE

by
Harish J. Jagtiani
Richmond Engineering Company, Inc.
Richmond, Virginia

presented at
COMMON, Cincinnati Meeting
September 6, 7, & 8, 1967

NUMERICAL CONTROL

Numerical Control (or NC) is a control system using punched tape or other automatic control devices to direct the operation of machines and machine systems. Numerical Control provides a highly accurate and efficient means of positioning or controlling the path of a tool. Machine tools are classified as being point-to-point or continuous in operation.

"Point-to-Point" implies that machining operations are only at certain locations on a part and that the cutter is retracted from the piece before it moves to another location. Point-to-Point, particulary valuable for precision drilling, boring, reaming, involves automatic coordinate setting. Careful layout, therefore, is less dependent upon the operator. "Continuous" (or contouring) implies that there is no retraction of the cutter as it produces a path in moving throughout a specified area. Point-to-Point machines operate in two dimensions; continuous machines in two or three dimensions.

NUMERICAL CONTROL AT RECO

The application that we are particularly concerned with, at Richmond Engineering is for NC Point-to-Point machine tools. In the manufacture of process heat transfer equipment, a good portion of ~~the~~ work is involved in the drilling of holes in the tube sheets and baffle plates of heat exchangers and the bolt holes in the flanges. With a view to secure some of the benefits from automation, Richmond Engineering decided to acquire a NC drilling machine. In 1961, we purchased Pratt and Whitney's Tape-O-Matic numerical drilling machine. It is a single machine with a table working surface of 30" x 20" and a transverse table travel of 15". This machine is capable of taking care of the drilling for RECO Standard Products up to 16" diameter. The experience gained with the Tape-O-Matic proved that NC machines could certainly provide tremendous advantages. With an increase in business activity Richmond Engineering decided to purchase a much larger drill press which would not only produce dollar savings but replace some of the existing manual drills and still have enough of excess capacity to handle increased business.

In late 1964, an order was placed for the American NC 819-32 Travelling Openside Boring and Drilling machine. This has an actual work area up to 92 inches wide and length of 15 feet. This machine has now been installed and is currently in operation.

The job planning procedure for this machine was similiar in many ways to the Tape-O-Matic. The machine has a "floating zero" or a "full zero shift" feature, as shown in figure 1. The reference point is set at the center of the tubesheet, which then allows the parts programmer to give all axis dimensions relative to the workpiece. The programming done manually was hand-written on a process data sheet.

DATA PROCESSING FOR NUMERICAL CONTROL MACHINES

In the preparation of "programs" or "instructions" to govern the movements and processes of the numerically controlled devices, there was the problem of cummulative tolerance errors being introduced through manual computational procedures.

An investigation was undertaken to determine what aid could be offered to alleviate the problem of computational tedium and inherent accuracy difficulties associated with manual preparation for NC processes. The thought of computations and accuracy naturally suggested electronic computers as a possible means for solution.

IBM has developed a simple language AUTOSPOT (AUTOmatic System for Positioning Tools). AUTOSPOT is a general purpose computer program designed to aid the parts programmer perpare instructions for NC Point-to-Point machine tools. The parts programmer can describe the required operations in a familiar language, without calculations, repetitions, and the tedium of output format preparation.

The AUTOSPOT general program processor receives the input information, performs various operations, and delivers outputs to the post processor. The post processor converts the information into the data required to perform the machining by a specific machine tool. Since there was not an existing post processor for the American drill and writing up on would involve considerable time, several months, investigation was directed to prepare an aid to our specific problem that would specify the locations of the centers of equally spaced holes on the tubesheet and also the locations of the centers of any number of equally spaced holes on the bolt circle.

In this effort to relieve time consuming, laborious, and technician type work from the shoulders of the part programmer, a generalized computer program has been developed to perform position calculations for tubesheet holes and bolt circles configurations. These configurations are common enough in our line of products and for a firm using numerical control equipment to justify such an approach.

EXPLANATION OF TYPICAL LAYOUT

From a typical layout of a tubesheet you will notice the tube holes are equally spaced and so are the rows of tube holes. Further the layout on one side of the center line is generally a mirror image of the other side.

There are two configurations of tube holes that we are concerned with. The tubes are either on a triangular pitch or a square pitch.

Lets consider the center of the tubesheet as the reference point with coordinates of (0.0, 0.0). Knowing the pitch of the holes, one can easily compute the coordinates of each hole in each row.

Hence for the first row the coordinates of holes marked A is (1.0, 0.0), B (2.0, 0.0) and so on. For the second row the coordinates maybe (1.0, 0.5), (2.0, 1.5) and so on.

We can compute these coordinates manually and put them down on a process sheet.

Now if a row of holes was defined by the following statement,

SY = 0.0, SX = 16.0, EX = -16.0, NH = 0.0

we can compute the coordinates of each hole in this row.

```
Pitch = (EX - SX) / (NH - 1)
      = (16.0 - ( -16.0)) / (33-1)
      = 1.0
```

Hence

```
N  1   Y  0.0   X - 16.0
N  2   Y  0.0   X - 15.0
N  3   Y  0.0   X - 14.0

          .

N  32  Y  0.0   X + 15.0
N  33  Y  0.0   X + 16.0
```

This procedure is the logic of the computer program. The flowchart further outlines this routine to compute the coordinates of each hole, figure 2.

FILLING THE INPUT FORM

A typical input data form filled up is shown in figure 3.

The master card contains some general information followed by ICODE, the reference point (corresponding to the Datum Reference to permit the specifications of point coordinates relative to the most convenient origin), total of detail specification cards, total number of holes to be drilled, type of pitch,

pitch and bolt circle specifications if any.

The detail card specifies coordinates for each row or part of each row in some cases. It contains SX, SY, EY, NH, JT and EXCEPT.

Where	SX	- Starting X coordinate
	SY	- Starting Y coordinate
	EY	- Ending Y coordinate
	NH	- Total number of holes
	JT	- Total number of exception holes
	EXCEPT	- Position of exception holes

EDITING THE INPUT

After the input form has been filled and the cards punched we run an edit of the input data. The edit program checks the input data for completeness and consistency.

The edit program checks the following:

1. Code
2. Bolt circle specifications
3. Pitch of the holes
4. Spacing of the rows of tubes
5. Total number of holes
6. Total number of detail cards

and so on.

The output listing from the edit may appear on the first run, somewhat as shown by figure 4.

After the edit is run the errors - if any - are corrected, and the input deck is run through the processor. The processor punches the output card and also lists each block of the data for the NC machine on the printer. A listing of the output is as shown in figure 5.

USING THE 3 - SPINDLE DRILL

On the American NC Drill we can also put a 3-spindle attachment. Preparing a tape to run with the 3-spindle drill is naturally somewhat different from that for the single drill.

The attachment we have and the usual pitch of the holes we maintain limits drilling three consecutive holes. Instead we have to drill 3 alternate holes at one setting.

For example if we had 6 holes equally spaced in a row, we first center the drill over hole number 3 and drill holes 1, 3, and 5. Then we position drill over hole number 4 and drill holes 2, 4, and 6. In other words we drill groups of 6 holes in 2 operations.

The processor automatically takes care of this. Now we do have rows of holes of less than 6. These holes we pick up on the single spindle drill. The processor hence is designed to segregate groups of holes to be drilled with the 3 spindle drill and those with the single spindle. These two data sets we store on the disk. Each record in the first set is accessed and cards punched out for the same. Then the same is done for the second set of records.

The general flowchart of the processor is shown in figure 6.

LIMITATIONS OF THE PROCESSOR

Obviously the processor we are talking about has its limitations. It is not as versatile or powerful as the AUTOSPOT processor. For instance, one half of the tubesheet is generally symmetrical about the axis. If the parts program were written in AUTOSPOT language, just one statement would have sufficed to invert the pattern on one side to the other. But with our present processor we have to define each row of the other half again. Of course this is not too difficult, because all the parts programmer has to do is to reverse the sign for the SX dimensions. However, this is not what you may call very sophisticated. We also have the problem of not being able to define patterns other than the bolt circle or a row.

Not long back we had a job which had a pattern of 3 concentric circles each with 36 holes and another pattern within the triangle formed by the circles. There were roughly 8000 holes to be drilled. Doing the programming manually would have taken ages. So we wrote a special program which consisted of routines from the original processor. This was not difficult but certainly took some effort and time, which could easily be saved by using AUTOSPOT.

PUNCHED TAPE PREPARATION

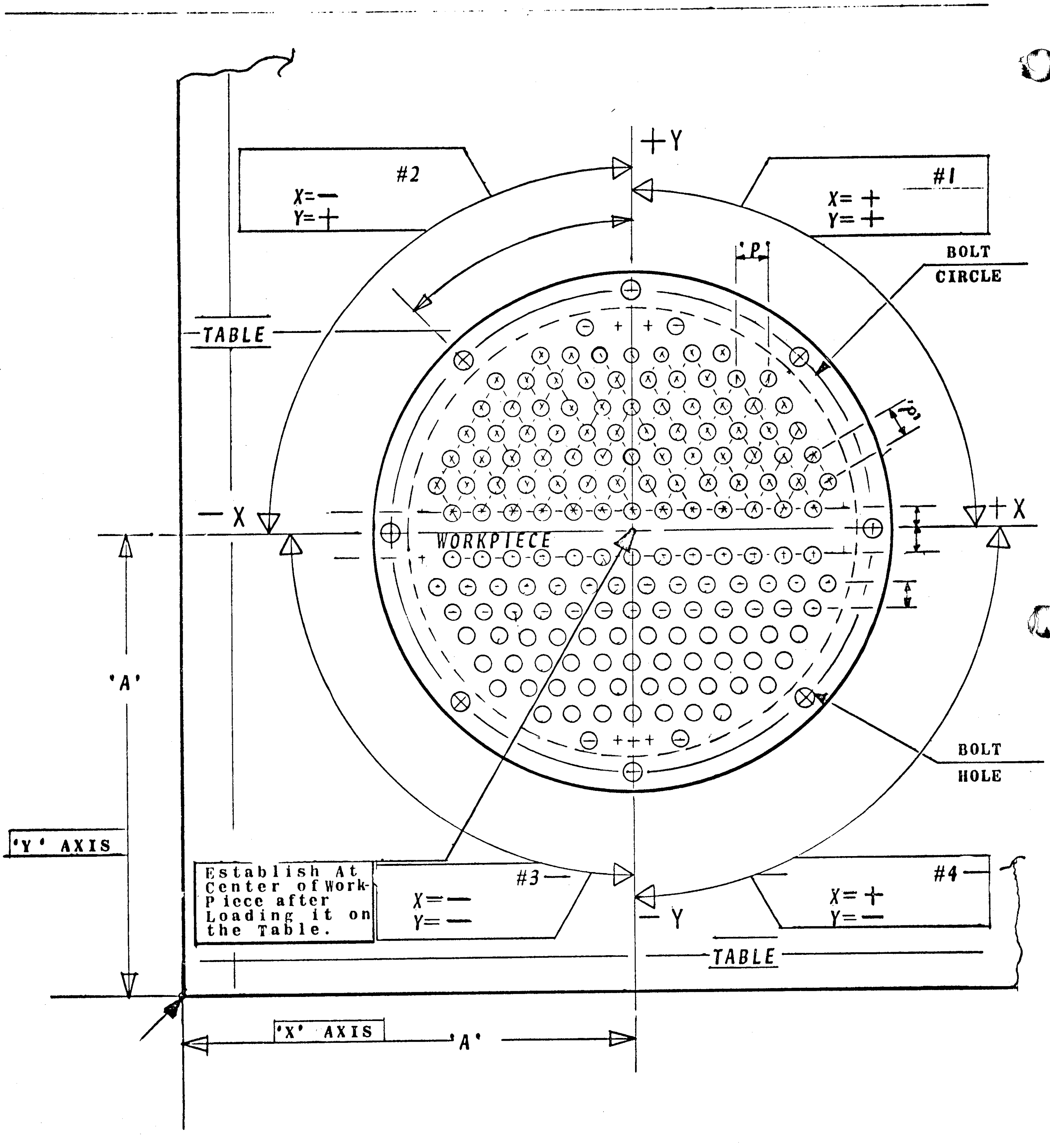
The data input to the machine tool is one inch perforated 8 channel tape in accordance with EIA standards.

Initially the tape was manually punched on a Friden Flexowriter. This was both time consuming and laborious task full with human errors and omissions, thus offsetting some of the advantages gained by using the computer.

When we changed to card output from the computer we added a Friden Automatic Card-to-Tape Convertor to the 2201 Flexowriter. Now the cards are fed in the card reader and the tape is automatically punched out.

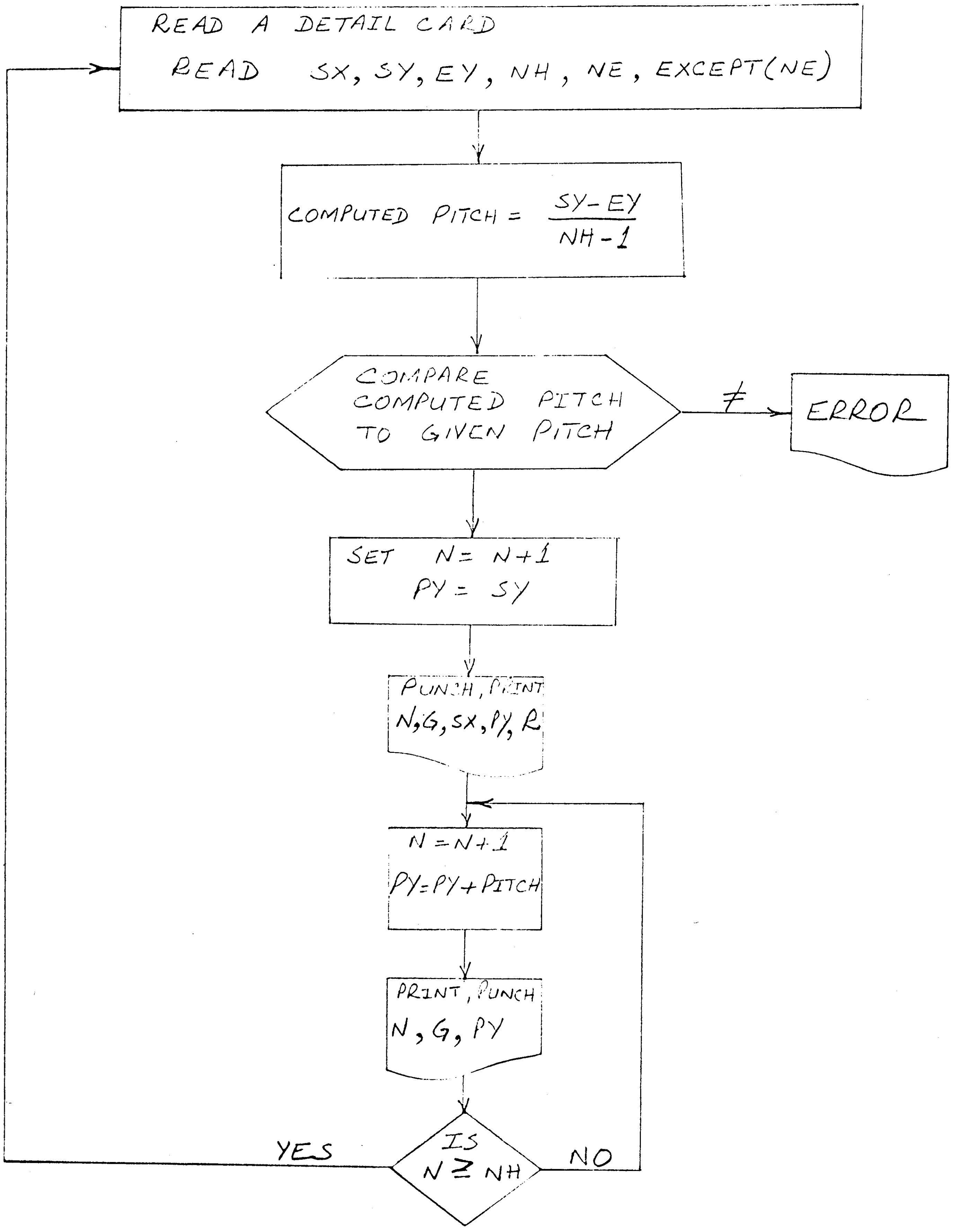
CONCLUSION

This processor as you can see is not very sophisticated or powerful as many others. However it has proved extremely easy to use for the parts programmer. The programmer does not need any knowledge of computers or computer languages. It has considerably reduced the lead time in preparing tapes for the NC machines -- from days to a couple of hours. And finally it has helped a small manufacturing company with somewhat limited skills and resources to increase its productivity and product quality.



+Y
#2
X=−
Y=+
#1
X=+
Y=+
'P'
BOLT
CIRCLE
TABLE
'P'
−X
+X
WORKPIECE
'A'
BOLT
HOLE
'Y' AXIS
Establish At Center of Work-Piece after Loading it on the Table.
#3
X=−
Y=−
#4
X=+
Y=−
−Y
TABLE
'X' AXIS
'A'

FIGURE 1



READ A DETAIL CARD
READ SX, SY, EY, NH, NE, EXCEPT(NE)
COMPUTED PITCH = $\frac{SY-EY}{NH-1}$
COMPARE COMPUTED PITCH TO GIVEN PITCH
≠
ERROR
SET N = N+1
PY = SY
PUNCH, PRINT
N, G, SX, PY, R
N = N+1
PY = PY + PITCH
PRINT, PUNCH
N, G, PY
IS N ≥ NH
YES
NO

FIGURE 2

Program "NCRECO" RICHMOND ENGINEERING COMPANY, INC. Sheet 1 Of 2

NUMERICALLY CONTROLLED DRILL INPUT

Input Prepared By D.E. Curl RECO Job No. 7057.30 RECO Drawing No. D-67202-1

Input Checked By ______ Customer Name ______

icard	job number	by	month	date	year	icode	reference point		total detail cards	total no. of holes	tubesheet o. d.	no. of passes	jp	pitch	no. of bolts	bolt diameter	bolt circle diameter	angle			lv
							x-axis	y-axis										deg	min	sec	
x	xxxxx.xx	aaa	xx	xx	xx	x	xxxx.xxx	xxxx.xxx	xxx	xxxx	xxx.xxxx	xx	x	x.xxxx	xxx	x.xxx	xxx.xxxx	xxx	xx	xx	x
1	2-8	9-11	12-17				19-25	26-32		36-39	40-46			50-54	55-57	58-61	62-68	69-76			
1	7057.30	DEC	08	21	67	3	40	40	38	1270	57.8750	2	1	1.2500	68	0875	51.2500	—	—	—	1

icard	starting x-coordinate	starting y-coordinate	ending y-coordinate	no. of holes	jt	exception holes									
x	xxxx.xxxx	xxxx.xxxx	xxxx.xxxx	xxx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx
1	2-9	10-17	18-25			31-50									
2	00	- 23.75	+ 23.75	39											
2	1.0825	+ 24.375	- 24.375	40											
2	2.1650	- 23.75	+ 23.75	39											
2	3.2475	+ 24.375	- 24.375	40											
2	4.3300	- 23.75	+ 23.75	39											
2	5.4125	+ 24.75	- 24.75	40											
2	6.4950	- 23.75	+ 23.75	39											
2	7.5775	+ 23.125	- 23.125	38	01	20									
2	8.6600	- 22.5	+ 22.5	37											
2	9.7425	+ 21.875	- 21.875	36											
2	10.825	- 21.25	+ 21.25	35											
2	11.9075	+ 20.625	- 20.625	34											
2	12.9900	- 20	+ 20	33											
2	14.0725	+ 19.375	- 19.375	32											
2	15.125	- 18.75	+ 18.75	31											

NOTES:

ALL COLUMNS SHOULD BE CODED FOR EACH LINE. FILL WITH ZEROES WHERE NECESSARY.

ICODE	= 1	for tubesheet only
	= 2	for bolt circle only
	= 3	for tubesheet & bolt circle
	= 4	for tubesheet (3-spindle drill)
	= 5	for tubesheet & bolt holes (3-spindle drill)
JP	= 1	for triangular pitch
	= 2	for square pitch
LV	= 0	if angle is given
	= 1	for bolt circle holes to be straddled about center line
JT	=	total number of exception holes

Form #60-12/66

FIGURE 3

```
LISTING AND CHECKOUT OF INPUT DATA FOR AMERICAN NC DRILL
=======================================================

1 7057.30  DEC   8 21 67 3 40.0000 40.0000  35 1275 57.8750  2 1 1.2500  68 0.875 51.2500   0  0  0 1

2    0.0      -23.7500   23.7500   39  0  0  0  0  0  0  0  0  0  0  0

2    1.0825    24.3750  -24.3750   40  0  0  0  0  0  0  0  0  0  0  0

2    2.1650   -23.7500   23.7500   39  0  0  0  0  0  0  0  0  0  0  0

2    6.4950   -23.7500   23.7500   38  0  0  0  0  0  0  0  0  0  0  0
DETAIL CARD NO...  4   SPACE BETWEEN THIS ROW AND PREVIOUS - COMPUTED # 4.3300   GIVEN # 1.0825
DETAIL CARD NO..  4   COMPUTED PITCH # 1.2838   GIVEN PITCH # 1.2500

2    4.3300   -23.7500   23.7500   39  0  0  0  0  0  0  0  0  0  0  0
DETAIL CARD NO...  5   SPACE BETWEEN THIS ROW AND PREVIOUS - COMPUTED #-2.1650   GIVEN # 1.0825

2    5.4125    24.3750  -24.3750   40  0  0  0  0  0  0  0  0  0  0  0

2    3.2475    24.3750  -24.3750   41  0  0  0  0  0  0  0  0  0  0  0
DETAIL CARD NO...  7   SPACE BETWEEN THIS ROW AND PREVIOUS - COMPUTED #-2.1650   GIVEN # 1.0825
DETAIL CARD NO..  7   COMPUTED PITCH # 1.2188   GIVEN PITCH # 1.2500

2    7.5775    23.1250  -23.1250   38  0  0  0  0  0  0  0  0  0  0  0
DETAIL CARD NO...  8   SPACE BETWEEN THIS ROW AND PREVIOUS - COMPUTED # 4.3300   GIVEN # 1.0825

2    8.6600   -22.5000   22.5000   37  0  0  0  0  0  0  0  0  0  0  0

2    9.7425    21.8750  -21.8750   36  0  0  0  0  0  0  0  0  0  0  0

2   10.8250   -21.2500   21.2500   35  0  0  0  0  0  0  0  0  0  0  0

2   11.9075    20.6250  -20.6250   34  0  0  0  0  0  0  0  0  0  0  0

2   12.9900   -20.0000   20.0000   33  0  0  0  0  0  0  0  0  0  0  0

2   14.0725    19.3750  -19.3750   32  0  0  0  0  0  0  0  0  0  0  0
NUMBER OF DETAILCARDS   ACTUAL#  38   GIVEN#  35


NUMBER OF HOLES  ACTUAL# 1270   GIVEN# 1275
```

```
    JOB NO...  7057.30      SPECIFIED BY...DEC      DATE...****** 1967
                                 ICODE... 3

            RECO PROCESSOR FOR AMERICAN NC DRILLING MACHINE

            TUBESHEET HOLES COORDINATE DATA
               TUBESHEET SIZE                         57.875
               TOTAL NUMBER OF TUBES                1270
               TOTAL NUMBER OF ROWS                   38
               NUMBER OF PASSES                        2
               PITCH - TRIANGULAR                      1.250

            EQUALLY SPACED BOLT CIRCLE HOLES COORDINATE DATA
               BOLT CIRCLE DIAMETER                   56.2500
               NUMBER OF EQUALLY SPACED HOLES        68
               NOMINAL HOLE SIZE                       1.0000
               ZERO ANGLE CLOCKWISE FROM VERTICAL
                   DEGREES                             2
                   MINUTES                            38
                   SECONDS                            49

               REFERENCE POINT    X-COORDINATE       40.000
                                  Y-COORDINATE       40.000


         N 1000     G 80     X 000.000     Y 000.000               M  0
         N 1001     G 81     X  40.000     Y  16.250      R  1
         N 1002     G 81                   Y  17.500      R  1
         N 1003     G 81                   Y  18.750      R  1
         N 1004     G 81                   Y  20.000      R  1
         N 1005     G 81                   Y  21.250      R  1
         N 1006     G 81                   Y  22.500      R  1
         N 1007     G 81                   Y  23.750      R  1
         N 1008     G 81                   Y  25.000      R  1
         N 1009     G 81                   Y  26.250      R  1
         N 1010     G 81                   Y  27.500      R  1
         N 1011     G 81                   Y  28.750      R  1
         N 1012     G 81                   Y  30.000      R  1
         N 1013     G 81                   Y  31.250      R  1
         N 1014     G 81                   Y  32.500      R  1
         N 1015     G 81                   Y  33.750      R  1
         N 1016     G 81                   Y  35.000      R  1
         N 1017     G 81                   Y  36.250      R  1
         N 1018     G 81                   Y  37.500      R  1
         N 1019     G 81                   Y  38.750      R  1
         N 1020     G 81                   Y  40.000      R  1
         N 1021     G 81                   Y  41.250      R  1
         N 1022     G 81                   Y  42.500      R  1
         N 1023     G 81                   Y  43.750      R  1
         N 1036     G 81                   Y  60.000      R  1
         N 1037     G 81                   Y  61.250      R  1
         N 1038     G 81                   Y  62.500      R  1
         N 1039     G 81                   Y  63.750      R  1
         N 1040     G 81     X  41.082     Y  64.375      R  1
         N 1041     G 81                   Y  63.125      R  1
         N 1042     G 81                   Y  61.875      R  1
         N 1043     G 81                   Y  60.625      R  1
```

FIGURE 5

DR. JOHN MANIOTES
COMPUTER TECHNOLOGY DEPT.
PURDUE UNIVERSITY
CALUMET CAMPUS
HAMMOND, IN 46323

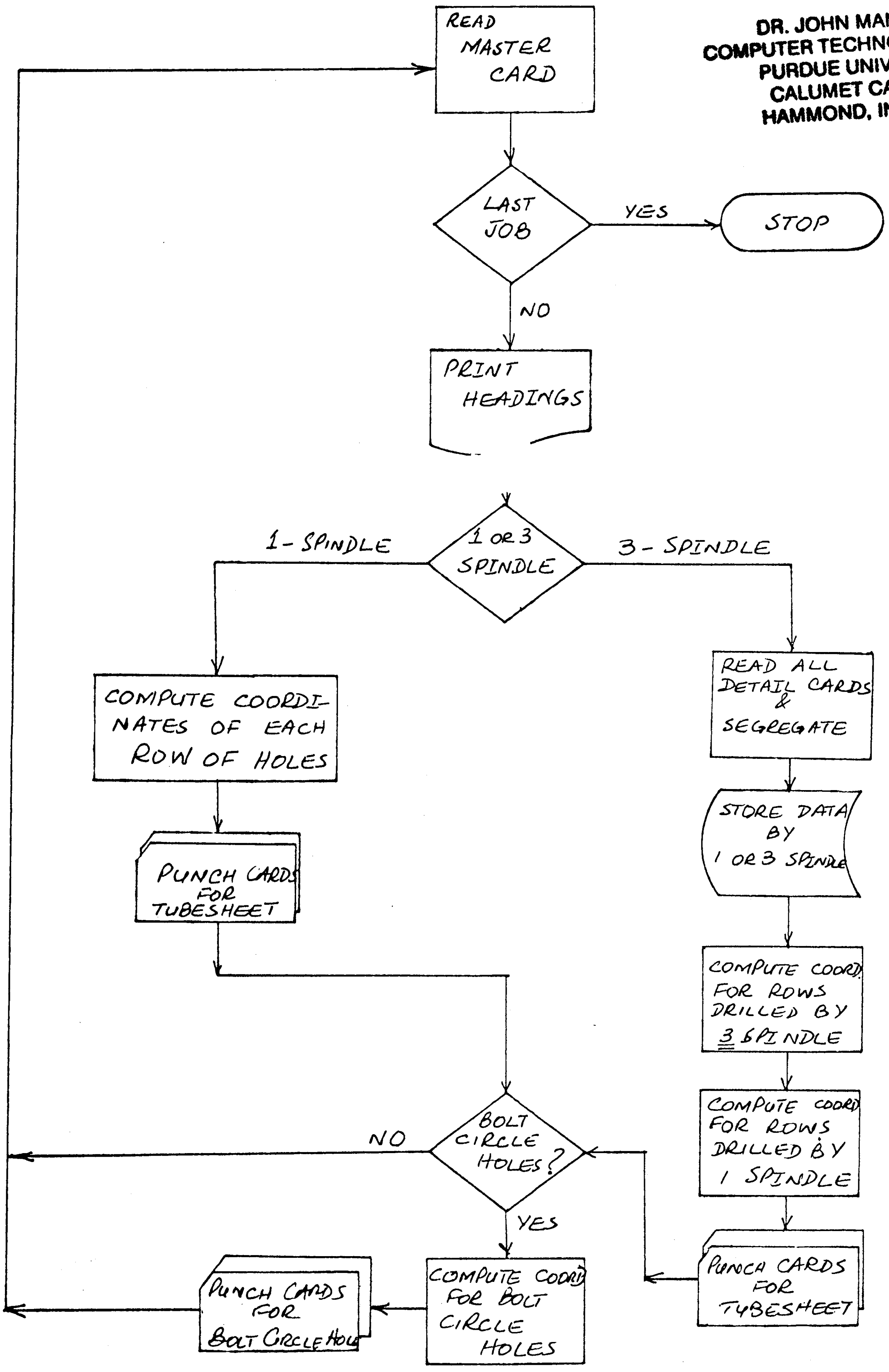

READ MASTER CARD
LAST JOB
YES
STOP
NO
PRINT HEADINGS
1 OR 3 SPINDLE
1 - SPINDLE
3 - SPINDLE
COMPUTE COORDINATES OF EACH ROW OF HOLES
PUNCH CARDS FOR TUBESHEET
READ ALL DETAIL CARDS & SEGREGATE
STORE DATA BY 1 OR 3 SPINDLE
COMPUTE COORD. FOR ROWS DRILLED BY 3 SPINDLE
COMPUTE COORD FOR ROWS DRILLED BY 1 SPINDLE
PUNCH CARDS FOR TUBESHEET
BOLT CIRCLE HOLES?
NO
YES
COMPUTE COORD FOR BOLT CIRCLE HOLES
PUNCH CARDS FOR BOLT CIRCLE HOLES

FIGURE 6

SESSION NUMBER T.3.4

SPEAKERS

R.D. BRENNAN, IBM SCIENTIFIC CENTER ON CHEMICAL ENGINEERING
APPLICATIONS OF CSMP

Petro-Chemical Engineering Project

Subject: CHEMICAL ENGINEERING APPLICATIONS OF CSMP

R. D. Brennan
IBM Scientific Center
2670 Hanover Street
Palo Alto, California 94304
(415) 327-2300

T.3.4 3 p.m. Sept. 7, 1967

CHEMICAL ENGINEERING APPLICATIONS OF CSMP

The 1130 Continuous System Modeling Program, an adaptation of PACTOLUS, is a new digital-analog simulator specially developed for online experimentation by the design engineer. 1130 CSMP, in combination with the computing power of the IBM 1130, has demonstrated itself to be an effective and economical tool for simulation studies across the breadth of engineering practice and the physical and biological sciences. It provides a library of 25 standard elements plus five "special" elements that the user himself may define. Interaction with the program is simple and convenient via the console keyboard and the console entry switches. From the keyboard, or via punched cards, the user enters 1130 CSMP language statements defining the configuration and associated parameters. During entry, automatically typed instructions and diagnostics guide the user through the procedures. These tell him how to initiate data entry, how to select the variables for printer and plotter output, and how to specify the integration interval, total run time, and output intervals. He may interrupt a simulation run at will to modify or extend the simulation, and need not follow a rigid schedule.

The Continuous System Simulator Approach

The S/360 Continuous System Modeling Program offers the best available illustration of what can now be done in continuous system simulation. This IBM program is an advanced version of DSL/90, the most powerful of previous efforts of this kind. S/360 CSMP incorporates a number of novel features: it can, for example, incorporate procedural coding within the definition of a macro element; it can conveniently calculate initial and terminal conditions (thus placing the entire simulation under programmed control). These new features provide the engineer a significantly more powerful and sophisticated tool for simulation. Working with a basic set of functional elements for modeling the components of continuous systems, the S/360 CSMP user specifies the interconnection of these functional elements by the (FORTRAN) equivalent of common mathematical notation for functional dependencies, namely, $Y = F(X)$. S/360 CSMP thus represents a rapprochement between analog block modeling and conventional digital programming. In addition to the block modeling capability shared with digital-analog simulators, this kind of program provides the power and convenience of algebraic and logical statements. Its FORTRAN-like language can be used as either a parallel (non-procedural) or as a procedural programming language.

The relative advantages of these two programs were compared within the context of a chemical engineering example; this example was previously described in connection with PACTOLUS in the March 1966 issue of <u>Instruments and Control Systems</u>. (An expanded version of this simulation study is expected to appear in a future issue of Genie Chemique.) Several other interesting applications in chemical engineering are described in the Proceedings of the IBM Scientific Computing Symposium on Digital Simulation of Continuous Systems, IBM Form No. 32-01943.

Both programs are available from IBM as Type II supported application programs. The Application Descr iption Manuals for 1130 CSMP and 360 CSMP are available under Form No. H20-0209-1 and H20-0240-1 respectively.

SESSION NUMBER T.3.4

SPEAKERS

CUSTOMER - I.B.M. RELATIONSHIP
ROBERT LUKEMAN, SALES SECTION
R.C. METEER, SYSTEMS ENGINEERING
G.P. MONJEAU, FIELD ENGINEERING

Presented by
J. R. Lukeman
IBM Corporation
112 East Post Road
White Plains, N.Y. 10601

COMMON MEETING
September 6 - 9, 1967

(Panel Discussion on Customer - IBM Relationship with respect to FE, SE, and Sales - Sales Section)

With the rapid changes we have been experiencing over the past decade in the data processing industry with regard to hardware, programming systems and applications, it is quite natural that there likewise be a corresponding change in the role of the IBM account representative. This change has not been an attempt to alter the primary responsibility of the account representative, which has always been and will continue to be "service to the customer", but one of implementation of that responsibility.

The complexities of the industry today have brought about a need for a type of salesman who is not only knowledgeable in all aspects of data processing, but one who acts as an interface between the customer and the IBM Company. No one individual is able to be an expert in all facets of data processing and still possess the ability to comprehend every conceivable application area the customer may have in his business. For this reason there is a trend to cover our larger accounts with a self sufficient team made up of experienced employees with both marketing and systems engineering backgrounds. Their job is twofold. One, (Slide 1) they must represent all of the service, experience and technical expertise of the IBM Company to you the customer, and two, (Slide 2), they must represent you, the customer, to IBM.

(Slide 3) - In carrying out this twofold responsibility, a thorough knowledge of both organizations is a necessity. As far as your business goes, the IBM account representative must have detailed knowledge of the applications you presently are running as well as those that are in the development stage. This includes not only the volume figures, input sources and output requirements, but the interrelationship of these applications as well. In addition, he must know your organization, where the data processing function fits into the corporate structure and what the evolution of this organization has been. Another important aspect the account representative must be keenly aware of, and one that our company is putting a great deal of emphasis on, is knowledge of the particular industry, whether it be finance, education, insurance, manufacturing, and so on. Knowlege of not only what is occuring in the industry today, but what are the trends of the industry. Finally, it is imperative that the account representative have a good working relationship with the top executive management of his customer so he knows what the long range goals of the business are and can make long range recommendations regarding data processing. As you are all aware, the implementation of a data processing function, if done correctly, takes many weeks and months of planning, testing, installing and converting. If this time spent is not consistent with the long range objectives of the business it can be disastrous to all concerned. To sum it up, the account representative must work hand in hand with every level of management (Slide 4) aspect of your business and become as knowledgeable as you yourselves are.

Once this knowledge is gained, and it is a continuing process to say the least, the account representative must couple it with the knowledge, experience and education he has received from the IBM Company. This knowledge covers the full spectrum of IBM (Slide 5) products, services and resources as well as our company policies and business practices. It is the responsibility of the team manager to be sure that he has developed the required talents within the team itself and to be cognizant of these resources so he may utilize them as the situation dictates. These include such services as industry specialists, product specialists, application specialists, operating system and programming specialists and many more who are available from district, region and division headquarters to assist the local people in fulfilling their primary role - that of services to you the customer. The account representative must have in-depth knowledge of the products capabilities, relationship to other hardware, characteristics, potential and most importantly the application value to solve a particular problem or set of problems. Other information that the account representative must be intimately familiar with in order to be of maximum service to the customer is, of course, our own internal policies and business practices so as to be in a position to advise and counsel in regard to contracts, deliveries, prices and stay withing the general business rules that we must follow.

The account representatives can work hand in hand with you in such matters as (Slide 6) providing professional guidance and advice on organization, both within the data processing group and the relationship of data processing with the other departments of the business. Advice on budgetary matters very often can be offered by the account representative in light of what other firms are doing and trends within the industry group. Personnel matters is another area where the experience of the account representative can be used to offer suggestions on position descriptions, salary ranges, apptitudes, and labor markets. This advice can materially assist you in your day to day operations as well as long range planning. Planning sophisticated applications such as Management Information Systems or Total Integrated Systems, Educational programs, Teleprocessing systems and the long range data processing plan, can all be facilitated within the assistance of the account representative.

The things I've been mentioning all have been areas where you, as customers, can benefit the most in developing a working relationship with your IBM account representative. In doing so, the IBM representative is fulfilling one of his primary responsibilities, and that is to make sure that every account in his territory is a satisfied customer.

A mutual respect for the other person's activities must exist between the customer's personnel and the account representative.

This mutual respect refers to the relationship the account representative must have for the supporting services available to him from the IBM Company, as well. In acting as the interface between the customer and the IBM Company, he must properly utilize these services in the best interest of both. The group that he usually works closest with, of course, are the Systems Engineers. We are fortunate to have with us today an SE who's qualifications make him one of the finest examples of service excellence in the IBM Company. I'd like to turn the conference over to Russ Meteer.

COMMON MEETING - Cincinnati, September 6-8

Operations Committee

Customer - IBM Relationship with Respect to FE, SE, and Sales - (SE Section)

R. C. (Russ) Meteer

IBM Corporation
112 East Post Road
White Plains, New York 10601

Area Code 914, WH9-1900
Extension 6153

Thursday, September 7, 1967 3:30 p.m. Session VIII

3 pages of text

COMMON Meeting
September 6-8, 1967
(Panel Discussion on Customer-IBM Relationship with Respect to FE, SE, and Sales-SE Section)

As the complexities of the Data Processing industry have increased over the years, it has become unrealistic to expect any one person to provide all of the knowledge and talents necessary to meet the varying needs of a computer installation. I am sure if you, as a group, look at your own operations, you will find Data Processing being used for engineering and scientific computation, accounting and record keeping, and management guidance and control. Each of these areas poses unique requirements. The Systems Engineer brings specific talents and sources of information to bear on these varied problems.

Generally, Systems Engineering will be most active during the installation planning phase. The installation of a Data Processing system consists of six major elements: (Slide 1) design, programming, testing and debugging, documentation, conversion, and implementation.

The most important factor which affects each of these six elements is your in-house capability to accomplish them. Education is, and has been for many years, one of the important initial phases of your experience with IBM, and is the beginning of the development of this in-house capability. The transition from formal classroom training to the practical application of your newly gained knowledge to systems design, programming, and productive use of the system is one of the key elements in the relationship between the Systems Engineer and you as a customer. The Systems Engineer brings to you the technical guidance necessary to make this transition.

The services provided to you by the Systems Engineer will vary to a great degree depending on the use you are making of your system. The common element in any case is the importance of making the transition from theory to practice.

You might ask why IBM is so concerned with customer self-sufficiency. Only you can make the most effective use of the system in your organization. Many studies over the last few years have shown that the key to a successful computer installation is user participation and management attention.

Let's discuss the ways in which Systems Engineers will help you develop your data processing skills during the installation planning. Design is essentially defining what is to be done. If your applications are engineering oriented, this probably involves formulation and the translation of the formulas to FORTRAN. If your jobs are more accounting-oriented, then design of the systems flow and a definition of terms is involved. In either case, you as the user know best what your problems are, and what results are desired. The Systems Engineer can often assist you by showing you tried and proven approaches to obtain a solution. In addition, he can discuss your own ideas with you and help you to evaluate their effectiveness in achieving the desired result.

Programming is one thing which you learned in some detail in your education program. There is a difference, however, between classroom case studies and actual programs. Here the Systems Engineer provides guidance and helps you in the use of various programming techniques until you have developed a proficiency of your own which allows you to work independently.

Testing and debugging usually provides the first opportunity that a customer has to operate the system. Those first unsure moments at the console change very rapidly to an "old friend" relationship with the proper guidance and counsel provided during the early testing phases.

Probably the most neglected and overlooked phase of installation planning is that of documentation. And yet, for a smoothly operating installation, documentation is essential. The plan in a programmer's mind is not a suitable substitute for readily available documentation. Providing that you, the customer, have made normal progress in developing a programming and testing proficiency, documentation can proceed without a great deal of SE involvement. I would say that the main contribution in this area is one of constant nagging to be sure that the job is actually accomplished. We humans seem to have an aversion to writing things down.

Conversion is largely a clerical function, but it requires close coordination and monitoring to insure that all the records of your company are correctly transcribed into the format intended for processing by the programs which have been developed.

And finally, implementation. As the saying goes, "the proof of the pudding is in the eating." I am sure that all of you have had a few sleepless nights during that period when you were implementing the programs on the system and making them work with real live data. At this time, the Systems Engineer will be close at hand, but, hopefully, he will have been able to help you develop to the point where you are able to do much of the implementation yourself and the crises of implementation are minimized.

There is one thing I would like to make quite clear and that is that the SE is not always all-knowing. He sometimes runs down blind alleys and finds problems which he cannot solve. However, he has available to him a number of sources of information which can help him to help you. (Slide 2) In each district, a Field Systems Center is available to him as a source of technical information and guidance. The Field Systems Center encompasses the Education Center, Test and Datacenter, and a Systems Design and Installation Center, better known as SD&I. It is this last group, SD&I, which provides the necessary technical expertise to assist the Systems Engineer in questions on programming languages, operating systems and application-oriented programs, such as simulation, netword design, etc. (Slide 3) The Field Systems Center in turn can go to one of our programming centers, either in Poughkeepsie, or Endicott in order to research problems and questions for which there are no answers at a district level. The programming centers have direct contact with the implementors and development groups to resolve any questions which the programming centers cannot answer.

Another source of Systems Engineering assistance, which is available in many of the metropolitan locations, is the Installation Center. The Installation Center concept is one in which the Systems Engineering installation planning assistance is centralized in order to concentrate the various resources of information together in one location so that they might bear directly on the customer's problem. In the Installation Center the customer may do his programming and testing in a machine environment having constant guidance available from one of the Installation Center personnel. Currently these centers support the 1130, S/360 Model 20, and punched card equipment. Experience shows that the Installation Centers accelerate the installation planning cycle and the customer's learning process.

In the past, the Systems Engineer also has been heavily involved in the maintenance of Type I programs after installation. This function has now been assumed by the Customer Engineer and will be discussed by Mr. Monjeau from the Field Engineering Division.

We see then the role of the Systems Engineer as one of guidance, information and counsel with the ultimate objective of developing customer capability. Self-sufficiency on your part will provide you with the means to realize the greatest return on your Data Processing dollar.

Abstract of Presentation to be Made by G. P. Monjeau,
Field Engineering Division

Subject: FE - Customer Relationship

Enclosed are copies of the VU - graph foile which will be shown with the presentation.

The presentation will open with a recap of the Customer Engineer and a brief rundown on what he does. Next is a description of the technical back-up for the CE for both hardware and software followed by aids to the CE.

Following this is the discussion of the customer - CE relationship, what is required and what can be gained.

The subject of APAR's will next be discussed, touching on the function of the CE and SDD's roll in APAR processing.

The last item to be covered is problem determination. Essentially, this covers who the customer sees when he has a problem.

FIELD ENGINEERING DIVISION

(CUSTOMER ENGINEERS)

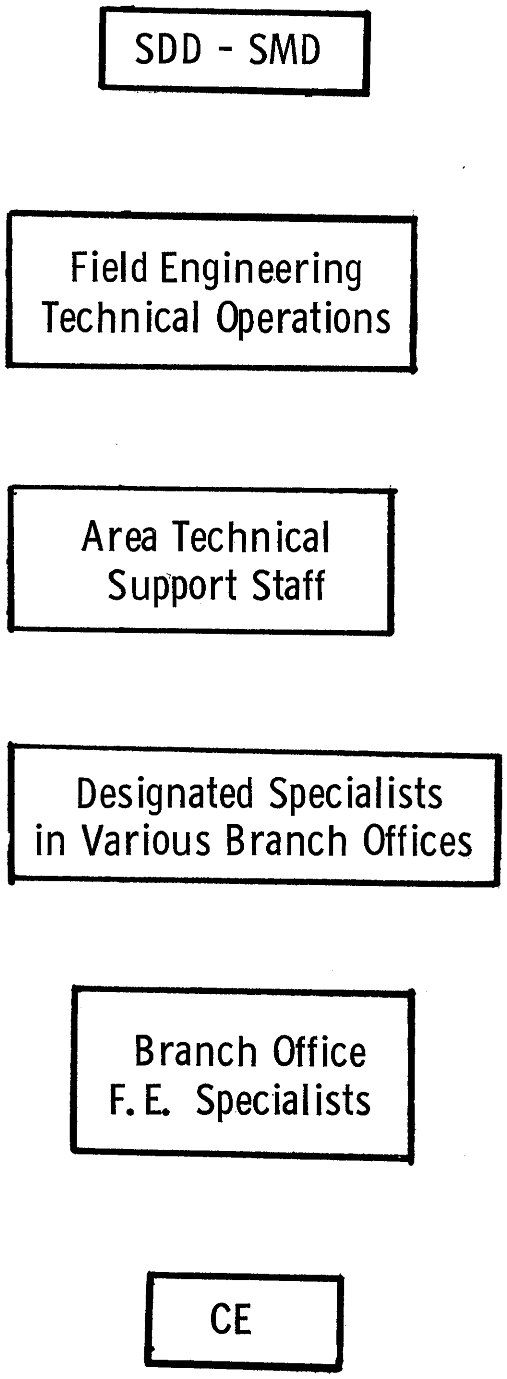
WHAT DOES THE CE DO?

1. Physical Planning
   - Environment
   - Space

2. Installs
   - Teams
   - Shakedown

3. Production
   - PM
   - EC's
   - Unscheduled Interruptions
   - Program Support

CE Technical Support


SDD - SMD
Field Engineering Technical Operations
Area Technical Support Staff
Designated Specialists in Various Branch Offices
Branch Office F. E. Specialists
CE

AIDS TO CE

Education	Hands on Seminars CAI Self-Study Classroom (formal)
Documentation	Microfiche Retain
Tools	Diagnostic Programs Test Equipment

CUSTOMER - CE

RELATIONSHIP

CE MUST

KNOW OPERATION

KNOW WORKLOAD

KNOW CUSTOMER

HOW CAN YOU HELP THE CE DO HIS JOB?

1. P. S. A. L.
2. Trouble Log
3. P. M.
4. E. C.

A P A R

A	-	Authorized
P	-	Program
A	-	Analysis
R	-	Report

C E

1. Identifies
2. Submits
3. Bypasses - if possible
4. Applies SDD supplied temporary fixes (PTF's)

S D D

1. Responds
2. Creates permanent fixes
3. Creates temporary fixes when required
4. Lends on-site assistance when required

TYPE I PROGRAMS

<u>INTERNAL</u>

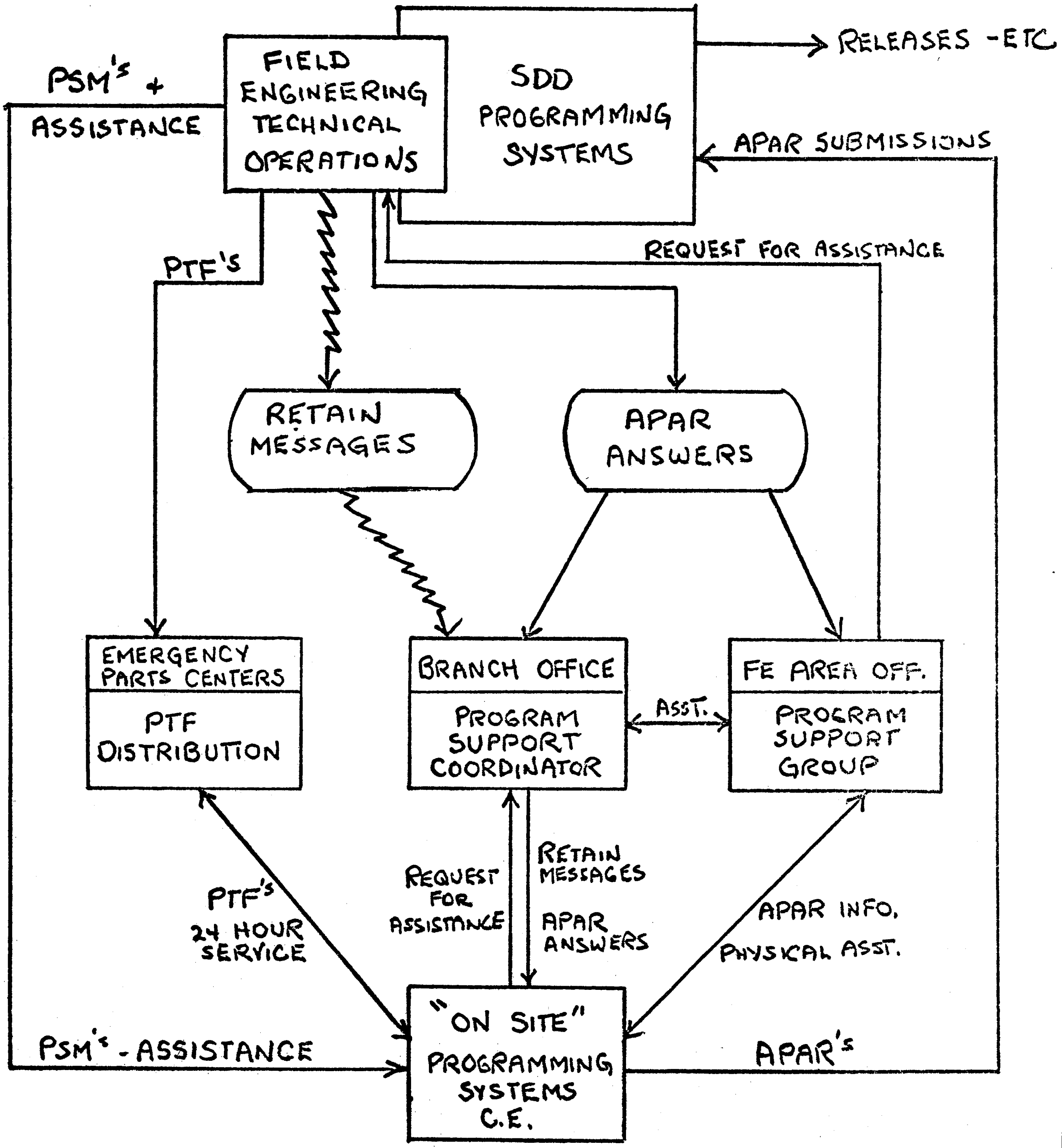
<u>EXTERNAL</u>

Problem Determination

What is it?

Problem Known	Hardware Type I Program Internal	Contact CE
	Type I External Other IBM Supplied Programs User Program Unit Record Control Panel	Contact SE
Problem Unknown	SE is there SE and CE are there	Contact SE
	CE is there Neither SE or CE is there	Contact CE

SUPPORT RESOURCES AVAILABLE TO YOUR CUSTOMER ENGINEER



SESSION NUMBER T.3.6

SPEAKERS

MR. G. WOLF, IBM EDUCATION DEVELOPMENT, SAN JOSE, CALIF.
MR. H. CADOW, IBM EDUCATION DEVELOPMENT, POUGHKEEPSIE, N.Y.

DISCUSSION

THE IBM EDUCATION EFFORT FOR THE 1130, 1800, AND 360 SYSTEMS INCLUDING PROGRAMMED INSTRUCTION MANUALS, EDUCATION CENTER COURSES AND THE CUSTOMER EXECUTIVE PROGRAM.

ATTENDANCE 39

Presentation on 360 Operator Training

by Harry Cadow

(1) The purpose of this presentation is to describe the education program for operations. In the context of this presentation, operations includes all those non-management people whose responsibility it is to keep the wait time of the 360 to a minimum. This includes the traditional button pusher through the lead operator or section chief.

(3) Before getting into the discussion proper, let me state what the operations education program does not solve, but what the user must certainly consider. These are the areas of Job Titles, salary ranges, career paths, and personnel selection. Job titles as a consideration has always been a frustrating topic in data processing. As you'll see in a few moments we've overcome the frustration by literally ignoring it. Salary ranges is also a topic that generates lively discussion - "what should you pay an operator?" is a tough question to answer because it is entwined with the problem of job titles - just what is an operator? Complicating the matter even further is the problem of career paths for operations personnel. Some installations have regarded the position of operator as a training ground for becoming a programmer if the operator is "any good". It is somewhat difficult in these installations to determine what happens to an operator if he doesn't make it to programming. One SHARE member said that operators who don't make it to programming in his shop leave to go to another company for a considerable raise in pay. A few installations have made considerable effort to develop career paths within operations. One company has a series of five positions, any one of which an employee in computer operations can aspire to. The promotion from one level to another is

based upon several measurable items. 1) Classes attended and marks received, 2) Length of time in the position, and 3) Management evaluation. The last consideration concerns personnel selection. "How does one go about choosing an applicant for a position in data processing operations?" At present there are a number of tests under development for assisting an operation's manager to help choose a potentially successful operator.

Now back to why job titles are more or less ignored. The main reason is that as a guide line for IBM in developing operations courses, very few users could agree on the description of "operator". Hence, we could not use "job title" description as an aid in developing courses and texts. (This problem existed not only with the job title of operator, but also with such other data processing areas as programming and systems analysis.) Therefore, in the area of operations, we listed the tasks that must be performed.

(2) Here we see five operation tasks. The first one concerns device handling. This includes such activities as obtaining and returning tape reels and disk packs to the data bank or library, mounting and dismounting tapes and disks, putting the correct cards in card readers and punches, and putting paper and proper carriage control tape into the printer. The second task has to do with responding to operational messages printed on the console typewriter and entering commands to the supervisor program via the console typewriter. When I say "entering commands" I do not envision the console operator deciding on the spur of the moment what command to issue. Rather, I see the console operator referring to a

well documented run sheet with a list of all the commands he may make and the conditions under which he may make them.

The third task has to do with system updates. This includes activities from the addition or replacement of modules to systems libraries, to a complete system generation.

The fourth task has to do with a very complex operation, that of identifying the cause of a problem. Identifying the cause of a problem can mean doing a complete diagnosis, recommending and effecting the fix, or, it can mean calling either the systems programmer, IBM SE or IBM CE. The task of system restart is also an operation of varying implication. It could simply mean re-IPL-ing and putting the job control cards of the errant job at the beginning of the input stream, or it could mean first running a series of utility programs to restore the system to a checkpoint condition and then re-IPL-ing. Running the utility programs is not too difficult - it is choosing the appropriate ones and preparing the proper control cards that requires systems know-how.

(2a) These five tasks lend themselves to categorization as to personnel requirement - production people or analytical people. It's not that the production people aren't analytically minded or that the analytical people don't produce - it's more that the production people work from a recipe and aren't required to make spur of the moment decisions, and that the analytical personnel must deal with unusual and unpredicted events.

(2b) The people requirements of production and analytical lend themselves very nicely to functions of what has been called System Operation and Operation Control. The tasks of device handling, responding to messages and issuing commands, and updating the system come under the function of System Operation. The tasks of system restart and identifying the causes of problems are the function of personnel in operation control. Also, operation control is partially responsible for system update, viz., laying out the systems residence pack and selecting the modules for a system generation.

Now let's see what IBM has in the way of educational materials and courses to assist you.

(17) First of all, IBM has released within the past 4 weeks, four System Operation Student Texts to be used for self study. The student text packages consist of a text, a book of illustrations, and a hands-on exercise. There is one student text for DOS using the System/360 Model 40, one for DOS with the Model 30, one for TOS with the Model 30, and one for any 360 model greater than a 30 for OS.

Each of the four student texts assumes some basic knowledge.

This basic knowledge is primarily concerned with some very simple data processing concepts. One of the ways the aspiring systems operation student could have acquired these concepts is by having worked in, or been close to, a 360 data processing installation. Some of basic concepts and knowledge are:

1. The relationship between input/processing/output
2. Hexadecimal numbering system
3. Decimal to Hex (and vica versa) conversion
4. Function and description of I/O devices and media
5. Purpose and function of Channels
6. Relationship between compiler, source program, and object program
7. Importance of documenting the jobs to be run

(18) What if the systems operation trainee doesn't have these concepts and knowledge in his repertoire? The answer is, "He enrolls in the programmed instruction course Computing Systems Fundamentals or CSF for short." CSF is a major revision of the PI course called BCS which was released for use in 1964. CSF is concerned only with system/360 topics whereas BCS addressed the entire IBM computer line manufactured in the early nineteen sixties. BCS you may recall, required somewhere between 30 and 40 hours for the student to finish. On the other hand the entire CSF course requires 15 - 20 hours.

(19) Whereas the entire CSF course consists of six units, the required pre-requisites for a systems operation trainee are Units I, IV, V, and VI.

Unit I provides the student with a basic knowledge of problem solution, numbering systems, and conversion, computing devices, and I/O media. Flowcharts, decision tables, programming principles, and programming languages are discussed in an introductory manner.

Unit IV discusses the process of creating a program from previously prepared decision tables and flowcharts. Short sample problems are used to show the development of programs in COBOL, FORTRAN, RPG, and PL/I. Also presented is a brief introduction to program compilation. Unit V discusses how the computing system (CPU, storage, I/O devices) solves a data processing problem. Information is traced through input devices, processing it, and finally as it emerges from output units. Numbering systems, I/O devices, storage devices, and processing units are illustrated and discussed.

Unit VI is devoted to the procedures for collecting and packaging the documentation created during the programming and testing of a data processing problem, and the documentation required for operations.

The amount of time required for a student to finish these four units varies from six to 14 hours.

(20) You've probably been wondering about Units II and III: Unit II concerns "Defining a Problem" and Unit III is about Analyzing a problem. Although these two units are primarily targeted for a programmer or systems analyst trainee and not absolutely essential for a systems operation student, we recommend that if time permits, the student take these two units. To elaborate a little, Unit II concerns problem definition in detail. The student is shown <u>methods</u> for determining input and output data requirements as well as the calculations required to solve his data processing problem. Sample I/O documents are discussed throughout the text. Unit III is about the use

of decision tables and flowcharts. Sample problems are analyzed by means of decision tables and are flowcharted using the standard flowcharting symbols.

We in IBM Education appreciate your concern with conservation of time. That is why Units II and III are not mandatory - - - but we do strongly recommend them. If the systems operation student does take these two units, he will have a better understanding of the steps involved in producing those computer programs for which he'll be responsible in his role as an operator.

(21) Now let's take a look at the systems operation student texts. The three texts for the Models 30 and 40 DOS and Model 30 TOS are identical in organization. Section 1 is an introduction to the System/360 computer. It can be classified as a review of CSF. Section 2 describes the role of the operator. It also very appropriately tells what the role of an operator is not. One other major topic of Section 2 concerns a description of the Programs to be run and the Run Book. Section 3, "Sample Programs" consists of detailed step-by-step procedures for getting the components of a system operational. A description of sample programs on sample run sheets are used to give cohesiveness to the step-by-step procedures. Section IV is devoted to DOS/TOS terminology and how the operator communicates with the supervisor. Included in communications is a discussion on I/O device assignments. Also discussed in Section 4 are numerous messages that the operator may receive and what actions might be taken on these messages.

After the student has finished the text, he should perform the appropriate systems exercise. This exercise will provide directed experience in operating I/O devices, interpreting operator messages, and issuing commands. The exercise requires from 1/2 hour to 1 hour.

(22) For Systems Operation Model 40 OS, the organization of the text is somewhat different. Section I is an introduction to the operating system - specifically, the function of the job scheduler. When the student has finished this section he is able to define such terms as Reader/Interpreter, Initiator/Terminator, data sets, Volume Table of Contents, DASDI, Job Control Cards, data set disposition, and, IPL. Section II which is entitled hardware familiarization is basically the same as "Sample Programs" for DOS. Section III "System Exercise" describes the OS message format, lists the publications "Messages and Completion Codes" and "OS Operator's Guide" and how they are to be used, and, a description of the exercise. One half to one hour of time is required for the student to run the exercise.

(23) Now let's turn our attention to the courses recommended for the personnel who perform the operation control tasks.

The initial course the student takes may be one or more of three:

1. CSF (which has already been described)
2. S/360 Introduction
3. Fundamentals of Programming Languages

S/360 Introduction is a 5 day class in the education center. The major objective is to have the student able to describe the functional characteristics and general principles of operation of the S/360.

Fundamentals of Programming Languages is also a 5 day course and is for people with <u>no</u> data processing experience. It is primarily a CSF review and a description of the various programming languages with analyses of programs written in the different languages.'

If the installation is to transmit and receive data via terminals and common carrier transmission facilities, it is recommended that the operation control student attend the 1 day Data Communications Concepts course. Here he will find out how data communications can effect the operation of the data processing installation.

The Problem Oriented languages are COBOL, PL/I, FORTRAN, and RPG. Some of these courses are currently available in both classroom form and PI form, viz. FORTRAN and COBOL. In general, RPG is available only in PI form. Although PL/I is currently being taught in education centers only, it will be available in PI form within the next 6 months or so.

The Assembler language coding course is also available in either form. For those students who study ALP in PI form, there is a 3 day hands-on wrap up class available at the education center.

The <u>facilities class</u> teaches the student how the operating system functions and what facilities, in the form of pre-programmed routines, are available. This includes the selection and use of utilities and the requirements for using various levels of Access Methods. The DOS/TOS Facilities class lasts for 3 days whereas the OS version is scheduled for 5 days.

The System Generation (or Sysgen) course covers the considerations for generating a system when provided with the following items:

1. Released tapes from PID
2. Sysgen manuals
3. The machine configuration and operating system options or features to be included.

The last item is sort of misplaced. It belongs of course just after S/360 introduction. This is where it will be placed in the future.

CONSIDERATIONS

JOB TITLES

SALARY RANGES

CAREER PATHS

PERSONNEL SELECTION

3

OS OPERATION

TASK	*REQUIREMENT*	*FUNCTION*
DEVICE HANDLING	PRODUCTION	
MESSAGES & COMMANDS	PRODUCTION	SYSTEM OPERATION
SYSTEM UPDATES	PRODUCTION ANALYTICAL	
IDENTIFY CAUSE OF PROBLEM	ANALYTICAL	OPERATION CONTROL
RESTART SYSTEM	ANALYTICAL	

SYSTEM OPERATION TRAINING MANUALS

TEXT
ILLUSTRATION
EXERCISE-HANDS ON

MODEL 40 DOS	(light) BLUE COVER
MODEL 30 DOS	RED COVER
MODEL 30 TOS	(dark) BLUE COVER
ANY 360 MODEL>30 OS	GOLD COVER

PREREQUISITE

COMPUTING SYSTEMS FUNDAMENTALS

UNITS I, IV, V, VI.

COMPUTING SYSTEMS FUNDAMENTALS

UNIT I	WHAT COMPUTERS DO
UNIT IV	PROGRAMMING A PROBLEM
UNIT V	SOLVING A PROBLEM
UNIT VI	DOCUMENTING A PROBLEM

COMPUTING SYSTEMS FUNDAMENTALS

RECOMMENDED IF TIME PERMITS

UNIT II	DEFINING A PROBLEM
UNIT III	ANALYZING A PROBLEM

295

SYSTEMS OPERATION
MODELS 30 & 40 DOS

296

21

SYSTEMS OPERATION

MODEL 40 OS

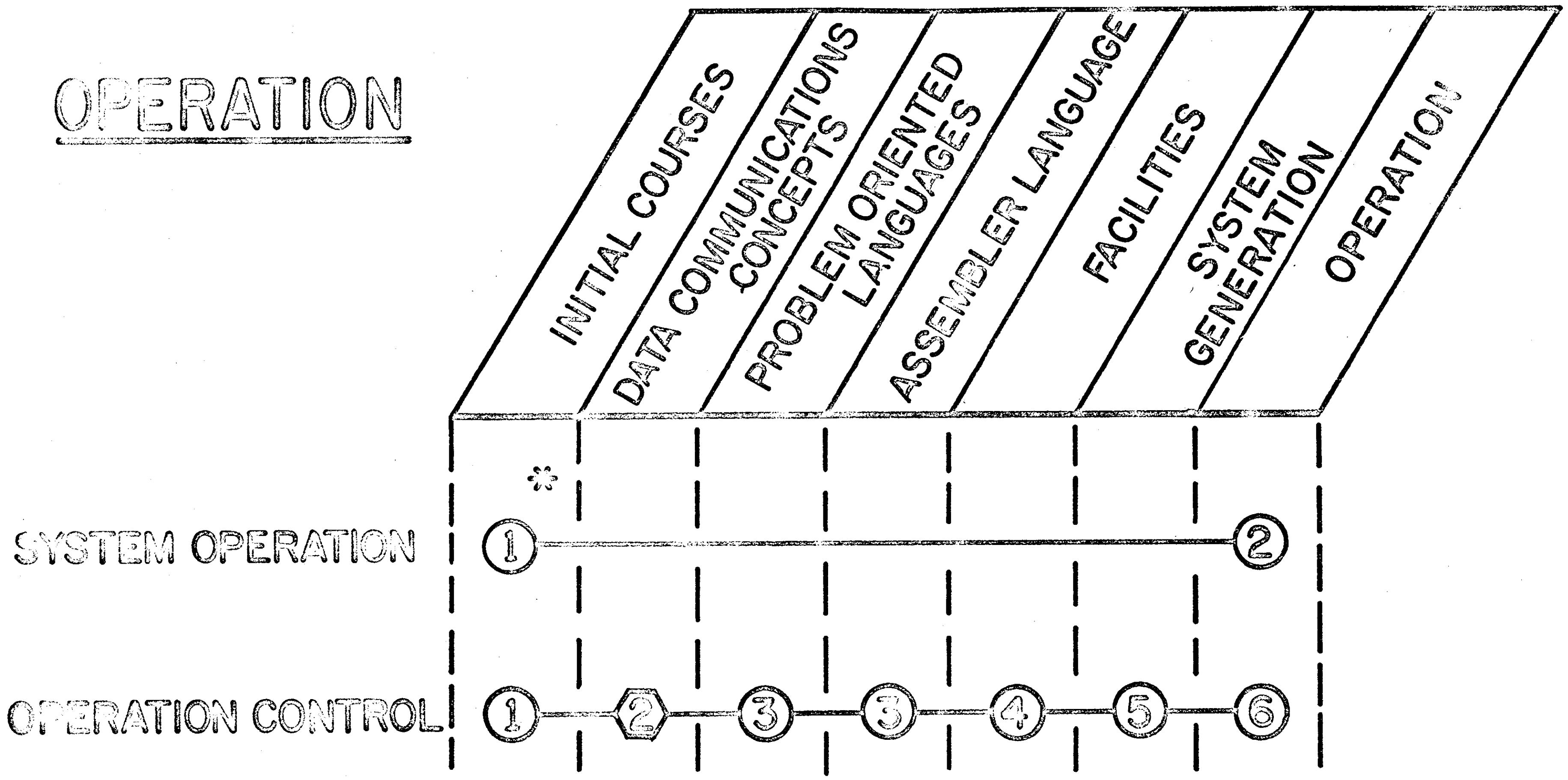
SECTION I INTRO TO OPERATING SYSTEM

297

SECTION II HARDWARE FAMILIARIZATION

SECTION III SYSTEM EXERCISE

OPERATION



* CSF PI ONLY

SESSION NUMBER T.4.1

OPEN BOARD MEETING

SPEAKERS

JAMES STANSBURY, PRESIDENT

TRANSCRIPTION

OPEN BOARD MEETING

COMMON - 7 SEPTEMBER, 1967

The transcription below indicates that we have much to learn about procedures for taped sessions. I think that the idea is good; it's the only way to get the verbal information at these meetings into the proceedings.

Many of the questions from the floor could not be understood (on the tape); comments were generally clear, since most of those used the floor microphones. When I could understand or remember the gist of the question, I have put it in; in some cases, I have reconstructed the question from the reply. In a few places, I have been forced to delete passages; the context was not clear.

In the future, we will have the floor mikes checked before we start, to eliminate the feed back that was present at Cincinnati.

The rest is up to you. Name, rank, and serial number, please, and - use the mikes.

The transcription follows:

CHAIR -

This is not exactly an Executive Board Meeting, in the sense that we don't vote on business, but we do have the Executive Board up here as targets. We also have some things that we'd like your opinions on. Some feed back from the members - and it's a chance for you to bring up any gripes or complaints, pats on the back or anything else that you think may be appropriate. I'd like to say before I start that there is some trouble with the mikes. They must be pushed to talk. There is a switch on each one of them. Turn on the switch whenever you have anything to say. There's a report that I will present tomorrow at the Contributed Program Library Session on the 360 Contributed Program Library. This is a joint COMMON, SHARE GUIDE proposition. The minimum standards that have been distributed by PID do not represent COMMON'S standards. They are minimal standards acceptable to all user groups. We're free to exceed them in anyway we want to. Just not contradict them.

Second item of business that I have - COMMON has been asked to participate in a Contributed Program Library Catalogue sponsored by the Joint User's Group and ACM. The Executive Board is in favor of this. What it means is that the programs in the library of any participating group would be part of a composite catalogue of all User Group Program Libraries, regardless of the manufacturer and machine affiliation. At present the ACM is talking about a subscription price of some $3 to $5 a year for a quarterly catalogue. You don't have to subscribe if COMMON participates - that's up to you. But I would like to know how you feel about our making the COMMON Library Catalogue a part of this composite Contributed Program Library catalogue. First, is there anyone who wants any further discussion, explanation of how it works, or anything of that sort? Have you seen anything on it, heard anything about it, anything else. Comments from the floor please.

QUESTION - What would be published? Titles, authors -

ANSWER - Titles, authors, abstracts - any program entered in that catalogue would be available to anyone regardless of user group affiliation.

QUESTION - How available?

ANSWER - On order - the orders would be handled through your User Group, so in practice they are restricting it to members of organized user groups, but it would be available. There was a stipulation that a reasonable reproduction cost could be charged.

The general idea, frankly, is that COMMON'S Library, along with those of several other user groups, is winding up in other catalogues, anyway. Such things as COSMIC and some of the others. We feel that we wouldn't be losing anything by making our library entries available. We do feel that we might be gaining something in having the information about the other User Group libraries. This would not affect the COMMON Library catalogues in any sense. The only additional thing that would be required is that someone who enters a program under these circumstances would have to fill out two submission blanks. One would go to PID, the other, without the program materials, would go to the Contributed Program Library liaison group committee. We are not promising to retro-fit existing libraries. If you don't want a formal vote on it, we don't need it, but I would like to have some opinion as to what you think of the idea. In the back, please and you may have to use the mike -

QUESTION - How do SHARE and GUIDE feel about this?

ANSWER - At the Executive Board level, SHARE and GUIDE are both in favor of this. I don't know if they have had a resolution, a vote on it, because, frankly, we do not have a response from IBM yet as to their willingness to participate. They promised such a response by, hopefully, the first of October. That's the reason I am asking for the feed back at this time.

FLOOR -

I think I could express the opinion of many of us here that this idea could be considered twofold - one, we have a centralized way of obtaining information and, second, with this type of catalogue, and the whole combination of papers being presented in one place, we could all have these papers available when we enlarge, let's say, to different systems. We're a 1620 user now, and have a 360 on order. I'm sure that many of you will express the same opinion.

I'm throughly in favor of it.

I would like to add one comment to that. On the Contributed Program Library, COMMON'S Contributed Program Library, there will be separate catalogues for the 360, the 1800 and 1130, and the 1620. A user will receive the Library catalogue corresponding to his machine type. The 1130 and 1800 catalogues are combined. T The official rule is that the user will receive only his own machine type catalogue, so you normally wouldn't know if you were a 1130 user what was in the 360 library without some such procedure as this. Incidentally, you can get the catalogue for another machine type by convincing your branch office that you need it.

QUESTION - Wouldn't the submitter have to sign a disclaimer?

ANSWER - They do have to sign a disclaimer when they send it in. I mean now.

QUESTION - Just from a legal point of view - Would the individual whose programs are now in the library have to be consulted on this, before we make any committment?

ANSWER - They had to sign a disclaimer when they sent it in. It has always been required.

FLOOR -

The disclaimer gives any third party the right to reproduce the program and give it to anyone they desire.

CHAIR -

AND I might add that IBM does exactly that. Apparently, they are legally obligated to not use the Program Library as a means of sales advantage. As a consequence, any requester can obtain a copy of any program in our library, regardless of whether or not he's even using an IBM machine. These are the present rules.

May I ask for a vote - a show of hands - as to those who would agree that we should participate. Very good. Thank you. There was to be some discussion of our proposed by-laws. How many people have read them enough to comment? They are published in CAST 7 which about half of the people received, and there were additional copies made available here.

FLOOR -

No time to read them.

I know it, but this is the last chance we'll have for discussion. Would you like a run-down from Laura as to the general changes we're making?

LAURA AUSTIN -

I think generally you'll find that there are relatively few changes except for wording. I'll just briefly go through them here, and see if I can pick them out. I wasn't prepared to do this, and I haven't a marked copy here.

We did change it to include the new machine types. We've changed the name throughout the by-laws from 1620 User's Group to COMMON. We've included the machine types that have since been made eligible for membership in COMMON. The purpose and framework of the organization remains the same as it was. In loss of membership, there has been a change, because we're doing away with the regional structure. Where we used to have two meetings in each region each year, we voted to cut it down to three meetings a year. So we've had to change our requirements for attendance at meetings. If an installation is unrepresented at four consecutive meetings you will be removed from membership. This should give you a period of about 18 months to make a meeting and still maintain your membership. We've added one other stipulation, so that we can keep our mailing list up-to-date. Each year, the Secretary-Treasurer will send out a registration form which registers your correct address, your machine type, and so on. If this is unreturned after two consecutive mailings, (we mail it, and if we don't get any answer, we try again) then you will be removed from the rolls. Now, on that mailing, I shouldn't say we'll mail again if we don't get any answer. If the mailing is unreturned, we will mail again to the installation, without an individual addressee, because the installation maintains the membership, not the individual. Also, any installation failing to respond to two consecutive mail ballots will be removed from the membership. We've provided for abstentions to be counted as voting. If you wish to abstain, don't just throw your ballot in the waste basket. Return it, because you will be counted as having returned a ballot, even though you are abstaining from the voting.

On the elective officers, there will no longer be regional Presidents, but they will be Executive Board Members at large. The Executive Vice-President will be elected from the Board by the Board members, but the President, Secretary-Treasurer, and the Board Members will be nominated from and by the membership at large. In the elections, half of the Elective Board will be nominated every odd numbered year - or every year half of the Board Members will be elected, so we will not have a complete turn-over at any one time. That means that we will be having elections every year, instead of just every two years as now. Half of the Board members will be elected every year. The President and Secretary-Treasurer will be elected for a two-year term of office, as will the Board Members. The election for the President and Secretary-Treasurer will only be every other year, but there will be Board Members elected every year.

There is some change in the wording on the Teller, so we won't have more than one Teller from any one installation. This was not restricted in the former by-laws.

On a quorum, we've had a difficult time in the past. You may remember in New Orleans, for instance, where we had to declare the people elected because there wasn't even a quorum voting for the people, so we didn't in reality have an election. We are reducing what is required for a quorum. So we've said that, for formal business at a meeting, at least one quarter of the member installations must have voting representation at the meeting to constitute a quorum. Then, for conducting business by mail, at least one-third installations must reply to amke up a quorum, and as I mentioned, abstention shall be counted. We hope it won't be so difficult to get a quorum anymore on our voting.

The decisions by committees have remained essentially as before, about proposed standards, on meetings - this will remain the same. We have changed the by-laws to state three meetings a year, that they will be distributed geographically and temporally. This means essentially we will have a meeting in the East, Mid-West, West, Mid-West, East, Mid-West, West, so we will hit the Mid-West twice as often as the others. We felt that this allows the East to get to both theirs and the Mid-West, and the West to get to theirs and the Mid-West. So this the plan for coming meetings.

We've also changed, under distribution of information, that any material, except the membership list, which is distributed through any COMMON channels, is assumed to be non-proprietary. We did feel that the membership list should be considered proprietary to COMMON. It will not be sent out just on request, because we've had difficulty with people putting COMMON members on mailing lists, so we have restricted its use. Of course, in submitting a program, as we've already mentioned, you submit a disclaimer at the time, which does make it open for any kind of reproduction.

You'll note that incorporated in the new by-laws is the proposal that was submitted last year regarding financing for Executive Board Members who must attend every meeting during the year. There will be some subsidization made by COMMON, if necessary, to have a Board Member in attendance.

We have also left it open that, if it is felt necessary in the future, we might put in a subscription feed for CAST, if the cost becomes prohibitive under our normal financing situation. As you know, the only way we receive finances for COMMON right now is through the registration at the meetings. We take the cost of the meeting, and then some additional to cover the expenses of COMMON. The issues of CAST that we have sent out have run us in the neighborhood of $1200 per issue of CAST. You can see that we are running into considerable expenses in operating COMMON, but we have felt, from the feed back we've gotten from the membership, that CAST is well worth this. If we are to continue to issue CAST on a regular basis, and to include all the information we have in the past, we are going to have to have some means of financing this. So we've left it open in the by-laws that, if deemed necessary, we might put in a subscription fee. However, this would be submitted to the membership for consideration before it was done. It's not going to be anything that is just railroaded through. I think that this basically covers the changes in the by-laws. Do any of the other Board Members think of any I have skipped over?

CHAIR -
Thank you, Laura.

I would merely like to say one thing. I hope that the members see fit to approve these by-laws, because I don't want to spend the rest of my life running from the sheriff. Obviously, I am approving expenses of Executive Board Members to get them to these meetings; I'm not authorized to spend your money this way, but I am doing so.

There was some question at Boston about a statement of finances. Chuck had to return to Dallas unexpectedly. Our present treasury is something like $4750. We will make in the neighborhood of $2500 from this meeting. As far as the Executive Board is concerned, we've already stipulated that the registration fee for San Francisco will be $25, and you will get only one luncheon out of it.

The feeling at Boston was that higher registration feed would be acceptable to the membership in preference to dues, and that a subscription fee for CAST would be acceptable in lieu of dues. They did not want dues inlieu of a meeting attendance requirement. As a comparison, SHARE has considered imposing a subscription fee for SSD; but the suggestion was voted down by the members. Every thing that they get from a registration fee is net - There are no group activities that are scheduled without charge. SCIDS registration is additional, and on a usage basis. SKIDS is, a SHARE organized drinking session. Six 'til two every night during the meeting. It's very well patronized.

In regard to the quorum, we felt that abstentions were legitimate votes for COMMON members. There are many of you new members who, for one reason or another, feel that you aren't qualified to make a decision, but are willing to go along with the majority. Now, we've asked you to deliberately abstain. The vote on the by-law amendment that we proposed earlier was very much in favor - I think something like 80% - but there were not enough votes returned to establish a quorum, so it was defeated. We're trying to get around that, but we didn't go as far as SHARE. In SHARE, if 20 ballots are returned from a mail vote, the majority of the votes determines whether the thing passes, because every installation has an opportunity to consider a mail ballot. I don't think that's quite legitimate. We didn't go that far. Are there any comments from the floor?

QUESTION - How many is 25% of the membership?

ANSWER - At the moment our membership is something like 1275. We are not a quorum here.

QUESTION - Is the omission of machine requirements deliberate?

ANSWER - It was deliberate. We'll accept anything from a 20 to a 91.

QUESTION - I would like to ask one question. Is there any provision in the by-laws now for bonding of either the Secretary-Treasurer or the Local Events Chairman? I know that Mr. Mauldin is undoubtedly a very honest person. Dave Dunsmore is undoubtedly a very honest person, but to protect them and COMMON I would hope there would be some way of providing for a bonding of them or of anyone who must handle organization funds. I don't think there would be an organization that could operate effectively without some sort of a bonding procedure.

ANSWER - I think you have a good point here on the Local Arrangements Chairman, and I know that Dave, himself, raised this question. It does not state in the by-laws, and perhaps should, about the Local Arrangements Chairman, but it does about the Secretary-Treasurer. It says he should be the primary financial officer of the group, and, as such, will be required to be bonded at the expense of COMMON. I might add that Chuck Mauldin has been requested by the Executive Board to have himself bonded at the expense of COMMON for his own protection. Further comments -

COMMENT - I think it has been rather the concensus of the people at this meeting that the by-laws as amended are going to be quite acceptable to the membership. I realize that it will take a mail ballot to insure this, but I think we could give you a show of hands here to indicate our approval

CHAIR -

Well, since the majority of you have not had a chance to read them, or at least so indicated, I don't think it would be too appropriate at this time.

IN THE BACK -

We don't need a show of hands. What we mean is just five minutes of people's time to take the time to mail the ballot.

CHAIR -

That is exactly right. I've forgotten exactly how many, but something like 250 ballots were returned on that by-law amendment. As I said, it was about 80% in favor of it, but there were not enough under our existing by-laws to establish a quorum.

QUESTION - Can you give some indication as to when the ballot will be mailed and how long they will have to return it.

ANSWER - Yes. The amended by-laws, the proposed by-laws, were published in CAST 7. About half of you, about 40% of you, I think, received them before you came to this meeting. The rest of you should find them when you get back. Under the by-laws there is a required discussion period - under the existing by-laws and under the proposed by-laws. We have to submit it to you, and we have to give you time to send in written discussion, if you wish, prior to requesting a vote. That written discussion and the ballot, together with Executive Board comments, will be published in CAST 8, which is scheduled for sometime around October 1 or October 15. You then have approximately a month to get your ballots back. Comments should come in very promptly, please, and we would appreciate favorable or unfavorable. We're most interested in some idea of what you want.

COMMENT - I HAVE read the by-laws. They look good to me, but, from what you said a moment ago, I would suggest consideration of the question of the size of the quorum necessary to conduct business at a meeting. If , in fact, a group of this size is not competent enough to make decisions binding upon the whole, I think we have a much too conservative position expressed, and it might be appropriate to lower the quorum when we do have a chance to discuss back and forth, and each one adequately hear the opinion of everyone else. Now, secondly, I take exception to your interpretation of what I believe to be the argument presented at Boston. I do not believe there was a strong opposition to the payment of dues from this membership if it was rewarded by adequate, efficient administration.

Thirdly, I say that, maybe because of what we said, planning and the conduct of this meeting has been vastly superior, and I say thanks to whoever it was.

CHAIR -

To some extent I can take credit for that. (LAUGHTER)

No, this is the reason for my guilty conscience for subsidizing the attendance of sufficient Executive Board members to establish a quorum of the Executive Board at the planning meetings we had for this. I don't mean that I can take credit for the work. Work has been Dave Dunsmore and Jim Tunney, Laura Austin, Division Chairman, Managers, Eric Zielinski of IBM. They are the people who must really take credit, but I think the Executive Board contributed a great deal to this, and I would like to have my guilty conscience sa~~l~~ved. (Clapping).

We do hope and expect a better run meeting at San Francisco than we had here. We are aware of the problem with the late agenda. They were mailed on time. The next ones are going to have to go out first class, which means more money. It is going to cost us about $1,000 for a first mailing about the 1st of October, a preliminary agenda about the 1st of November. That's the reason for the deadlines that I mentioned at lunch.

IN THE BACK -

Do you know how many installations we have represented here? How many people we have here?

ANSWER - The people are approximately 450. The number of installations, however, is by no means anything like that. There are around 50 attendees from IBM, and I know of several installations that have multiple attendees. I couldn't begin to guess, but somewhere in the neighborhood of 200 or 250 installations.

FROM THE BACK -

Couldn't hear-----------------

CHAIR -

I would like to have that in writing, please. To CAST.

DICK PRATT -

I'm not sure, maybe somebody can correct me about this if I'm wrong, but I believe the only way to get this provision in the proposed by-laws is to defeat them, and then propose a different set of by-laws with this change in it. If this is true, and I think it is, I would like to suggest to you that, if this is the only objection you have, and if your other objections are similarly minor, you approve the by-laws as they stand, and then propose these things as amendments, because we absolutely cannot operate legally under the present by-laws.

CHAIR -

Before we even comment on this - under our present by-laws we have to have two-thirds to approve these amended by-laws. If we get that done, then we can conduct business on a reasonable basis. That's one advantage.

FROM THE FLOOR - SAM LYNCH, U.S. PRODUCTION CO. -

I agree with Dick, that if this is the only thing that prevents us from getting these by-laws, for crying out loud, let's get these, and then work on them. But how can we find out right now whether Dick's assumption is correct? You know, if it is, then I won't even bother writing CAST and making the suggestion, and I'm sure the gentlemen back there won't either, but can we find out now what the legality is?

CHAIR -

Chuck Mauldin knows, and he's not here. (After I got home. It's two thirds of the membership in favor, to amend the by-laws).

In any case, from our existing knowledge, I think we'd better do it.

DICK PRATT -

The point is, I don't think there is anyway, either under the existing by-laws or under the proposed by-laws, to amend a proposal to be voted on by mail. You either accept it, or defeat it as it is stated. I think that's correct.

CHAIR -

I believe that one of the alternatives in mail discussion is to submit an alternative proposal, but then that must be discussed, so it would definitely delay it for two to three to four issues of CAST, which is sometime next Spring.

FROM THE FLOOR -

I don't want to delay this, but, Dick, you imply that the restriction is on the mail ballot. You know, this thing could be changed before it is mailed to us.

CHAIR -

It has already been mailed.

FROM THE FLOOR -

Not in the form of a ballot - in the form simply of read this.

CHAIR -

This form has been mailed - must be discussed, and then the ballot submitted.

FROM THE FLOOR -

O.K. In other words, the procedure has started, and we should not abort them, but carry through. I would certainly go along. I withdraw my comment for now.

FROM THE FLOOR - WADE NORTON, from Southern Service, Birmingham.

I would like to ask the question if there is any way in which the proposed by-laws are not as good as the existing by-laws?

Whether we have reached an ultimate is not necessarily the question. Whether we have ever reached an ultimate is subject to some question in my mind, so would it not be better to go ahead and make progress in increments as we have been doing since we got our first computer. (Clapping of Hands)

FROM THE FLOOR -

I have intermittenly received copies of CAST and various letters from COMMON. If we are going to mail out a mail ballot, let's please make sure that all the members receive a copy. Myself for one would like to receive a copy.

CHAIR -

All right, I can speak for Chuck on this. As you know, he has had some trouble with his installation. He's still having trouble with his installation, but not from lack of support. He informs me that he has reduced his back log of mail from yea high, about four feed, to around three inches and that, within roughly two weeks, there will be a membership list complete with addresses, which includes every application that he has. I will not guarantee that there is no incorrect information in there, of course. There should be no problem with people registering or changing address, and not having it processed within a reasonable length of time.

LAURA AUSTIN -

I'd like to say that in the new members' meeting yesterday, some came up and said they had not received their mail. At that time, Chuck checked the list which he has, which is current as of last Friday, and all but one of the people's names were on there. We hope that things are in much better shape right now than they were a week ago.

CHAIR -

My reason for giving the two weeks delay was that Chuck also said that he has 75 new member's names in Dallas, which are going in immediately to Pat Lonergan for the final up-dating on their membership list.

FROM THE FLOOR -

I'm just wondering - I had problems here with Chuck, and it is possible when you have something like a ballot that it could be flagged in some way so that you know it needs immediate attention, or urgent attention, and not be put in with everything else in CAST. (Sorry - everyone talked at onee here)

CHAIR -

Well, we assume that you would at least look over the table of contents. We do try to segregate such items, as a matter of fact, generally not the front page, but we'll try to make it the back page, normally.

FROM THE FLOOR -

Different colored paper.

CHAIR -

Different colored paper is a suggestion. I don't know whether we can do it or not, but we can certainly consider it.

FROM THE FLOOR -

If it were just typed in upper case in the table of contents - that would tend to catch someone's eye. Action is required.

CHAIR -

Well, I'll get hold of Chuck after I get back to New York, give him a call, and suggest that he can possibly arrange to have an asterisk, or some sort of indication, put beside the item in the table of contents that requires immediate attention and your action.

FROM THE FLOOR-

If it's official action that is required of a member and this is pointed out --

CHAIR -

Something of that sort, but I think I will have to insist that you do at least look over the table of contents. It is rarely more than two pages.

FROM THE FLOOR -

I am wondering, as a suggestion, would it be possible to get regional responsibility for contacting the members in the local area to find out if contact has been made or just to remind them to mail this in.

CHAIR -

I might say that one of the reasons that we put that failure to respond to two mail ballots in there, is that if you don't pay attention, you aren't going to be a member very long. That won't affect this vote, but if you pass these by-laws that provision will be in there.

FLOOR -

In reply to that suggestion, it's my opinion that what we need is responsible individuals who will take action without being led by the hand, and not somebody that you have to constantly follow up and follow up.

FLOOR -

He might have been referring, tho, to those people who might not have received it because they weren't on an address list.

CHAIR -

I might add in connection with this that quite a few of the local installations, Long Island Lighting for one, did not receive an agenda till very late, because the third class mail service on Long Island is lousy. Jene Lewis called me, and I find that other members do the same thing; they will tend to call the nearest Board Member, contact like that, if they know that something is coming up. You certainly should know because you would have to miss two copies, the one that carried the item being discussed and the one with the ballot. I think that to some extent it is back to a matter of installation responsibility. If you don't get something you should, then find out why.

FLOOR -

May I make a suggestion - If you don't get CAST 7 write to Chuck. This way you will be sure of getting your ballot. When you get back to your installation, if you haven't got CAST 7, then write a letter to Chuck right away so you can find out what they trouble is and be sure you get your ballot.

CHAIR -

Which takes care of only the people here.

FLOOR -

That was tried four weeks ago and I still haven't gotten an answer back. I sent a telegram.

CHAIR -

To where? Terre Haute? Or Denton?

FLOOR -

Denton.

CHAIR -

He's not here to defend himself. I don't know the circumstances.

FLOOR -

We have a problem apparently to get this particular ballot approved by a large majority. Would it not be worthwhile to make a special first class mailing of this ballot only in an effort to get this back? Otherwise, you are going to have multiple mailings, anyway, and you are going to wind up with the same cost. It would be roughly five cents apiece if you do it that way, rather than putting it in the next CAST or some other method. This is a critical point. Once we get past this our quantities are down. We don't have to have the same reply.

CHAIR -

CAST goes out first class in any case.

FLOOR -

I know, but it is lost.

CHAIR -

A separate mailing is certainly something ....

FLOOR -

I went back. I didn't know I had it. I went back and looked in CAST and found it. I had gone right past it the first time and you're not talking to people who are here. We'll all look for it. We're not the majority, by a long shot.

CHAIR -

Well worth considering. Thank you.

CHAIR -

The gentlemen in the back.

FLOOR -

You shouldn't have to go through a special first class mailing just to get you to read the appropriate place in CAST, or the appropriate ballot. That's what Cast is all about. This is what we are trying to do - save a little bit of extra money. For heaven sakes, please read it page by page. Please read the letters from Joe Doakes to see what this is all about.

CHAIR -

I have a gentleman down there.

FLOOR -

I just want to say in this case, as in so many cases, working with the practical situation is going to require an awfully lot more than working with the theoretical one. We can look out for this, but what about the ones who are not here?

CHAIR -

An there happens to be about 800 to 900 installations that are not represented at this meeting.

FLOOR -

How many must vote to establish a quorum for this?

CHAIR -

This has to be a mail ballot, which means two-thirds, something like

800 or 900. However, the present by-laws require the approval of two-thirds of the members. They would all have to vote yes.

FLOOR -

You have 300 people here and there are roughly 500..............

CHAIR -

We have 300 people, but there are not 300 installations represented here, I don't believe.

FLOOR -

Nevertheless, the first class mailing will call attention to it. This has developed into a very difficult situation right now to make our initial requirement for a quorum.

CHAIR -

Thank you.

FLOOR -

A very interesting point is that, if we can get this one ballot passed, we can kick out all the dead beats who don't answer, and then we'll have a quorum again. But in the meantime, we have some number of dead beats who never answer. O.K. so they don't read CAST so we'll save some expenses by not sending them CAST or anything else after this. We have to get this passed, because for now we can't kick them out for not answering the ballot.

CHAIR -

I agree that almost any measures are warranted to get some response immediately.

FLOOR -

I recommend you that you put this first class mailing to a vote.

CHAIR -

I hear your words, and, personally, I am inclined to agree with you, but I will have to review our financial status. About 200 for postage, plus some labor charges, of course. Folding and stuffing the envelopes and so forth.

FLOOR -

Suppose we don't get back enough votes, what happens then?

CHAIR -

Then we keep resubmitting it.

FLOOR -

I apologize immensely before I even start, but I'm going to say it. We are a society of professionals in Data Processing. We're supposed to be experts in fine procedures and good systems. We are in charge of installations spending lots of money for high powered equipment. Let's not be penny ante about running our own organization. I detect this in all together too many responses of the Executive Board and elected officers, and, to my chagrin, I hear it from the floor. Let's spend $60. Let's spend $1,000. Let's get an organization that works for our benefit.

CHAIR -

At the moment, if we go wildly over budget, the registration for San Francisco will not be $25 - it'll be $35. (Laughter and clapping)

FLOOR -

We'll mail them out first class.

CHAIR -

They will go out first class. CLAPPING

CHAIR -

There were two other hands raised. Same comment, or generally so? Further comments?

FLOOR -

I just wanted to pose a question, Jim. Is there any reason why or why not our registration fee could not float between certain bounds depending upon our particular situation and expenses...........

CHAIR -

It does now.

[illegible]

FLOOR -

I didn't know if we were illegal in this respect.

CHAIR -

No, we aren't. We decide how much we need. We have to try to get it out of meeting registration. That's the reason we left in our amended by-laws the ability to impose a subscription feed on CAST. It is a little unfair, because of this fluctuation in the registration feed. It might always happen, we'll say, that one particular region wound up having to pay the highest registration fee. We are considering other methods of financing. Definitely. But at the moment we can't even legally impose an subscription fee for CAST. Are there any comments?

FLOOR -

I just wanted to second again what the last gentleman said. We spend many, many thousands of dollars on computers. Man-time cost thousands of dollars. Half an hour of computer time is not really thought of as being expensive. Half an hour of computer time is $25 or $30. Please don't think small. We've got bigger things to get out of this instead of worrying about $10 more in registration fees.

CHAIR -

Part of our problem here is there are a lot of 1130 installations, and 1130 installations are one and two man installations. And they do have cost to worry about. They are in the same fix the 1620 group was years ago. I'm talking about the majority now.

FLOOR -

Neither the typist nor I could get this clearly enough for transcription. Comments about the Boston meeting. JCS

CHAIR -

I know this. It was a very poor meeting.

FLOOR -

I understand that people just sat and talked. I agree with this gentleman - if it's needed, we should go ahead and attempt to get our finances straightened out now with a higher registration or with some nominal charge for CAST. For instance fifty cents a year.

CHAIR -

At the moment there is a subscription price for IBM employees who want to get extra copies - you know - for branch offices - things like that. This is $15 a year and that is about a break even price.

FLOOR -

About a $1 an issue.

CHAIR -

Approximately that.

FLOOR -

Let's charge it, then.

CHAIR -

At the moment, as I say, under our existing by-laws we cannot charge for them. We can do it only if we are supply extra copies or something of that sort. Copies other than the one that a member installation is entitled to.

CHAIR -

(Section deleted; couldn't transcribe).

FLOOR -

It seems to me, Jim, that the message is clear that you ought to spend the money to put on a quality program, and most of the people are going to pay the money. The people who don't aren't going to recognize the importance of operating their own installation, whether it's a one-man installation, or what it is. Let's spend the money and do a high class job.

CHAIR -

We agree with that and we did it. And as I said I have a guilty conscience.

CHAIR -

O.K. let's wind this up. You people have indicated that you are largely in favor and I am quite pleased, I might add, with your response. Is this pertaining to this?

FLOOR -

American Electronic Labs - I heard you talking about the COMMON meeting in Boston about not being well run. That was the first COMMON meeting I ever attended, and I don't agree with you at all. In fact, I disagree violently. This meeting has been run very weel, but, to me in Boston, whether it was planned well or not, it ran very well.

CHAIR -

I agree. It wasn't planned very well, but it ran well.

FLOOR -

It got this meeting off the ground. A lot of things that happened here resulted from that, so I see no reason for anybody to say to themselves - you know - we have to look back at Boston, and say we didn't do a good job. You did. I was very happy with it. In fact, I probably wouldn't be here if not for that.

CHAIR -

And we didn't do a good job here in my sense of the word, and we expect to do a better one in San Francisco.

Bill Lane, incidentally, needs a Local Arrangement Chairman for San Francisco. He's handling both ends of the deal, but he's nearly 200 miles from San Francisco. He can't work as Program Chairman, and as Local Arrangements Chairman. He shouldn't be working as either, but we feel that we can't swap Program Chairmen at this late date.

FLOOR -

I feel like the mandate is clear to those of us who are here, altho we can't vote - we could vote, but you would tell us that we don't have a quorum - so why don't we do that? (Comment concerned registration fees for San Francisco).

CHAIR -

Never mind. I think the majority is quite clear. We hear that, and will consider that in discussing our San Francisco plans tomorrow. There will be an Executive Board Meeting at lunch to discuss the results that we are hearing now. Bill Lane has some comments. He's the guilty party for San Francisco.

BILL LANE -

I hear all of these things. We haven't discussed the raise in the fees, but I can almost guarantee they are going up. The first mailing for the San Francisco meeting will contain approximately the following material, and will come out approximately October 1. It will contain three hotel registration cards. Please get them in. It will contain a blurb about how great San Francisco is; I think it is the best city in the world. I'm biased. It will have some information regarding the wive's program. We'd like to see a lot of wives out there. You ladies bring your husbands along. Maybe we can send them out to North Beach. It will have some general information on the meeting, and will be about October 1. We expect, hope, and ask for returns on this information. We would also ask that if you have some questions, if you have some items that wish to be discussed, that you get to your project leaders, or to your Division Chairman, before you go home tomorrow, so that they will get on the agenda. I am not about to put anybody's "non"-return on the agenda. I'm not going to make an agenda for you. A slot will be there, but information will be printed only if it is submitted. And submitted means by deadline. There will be a second mailing approximately (both of these are going out first class; that's already been decided). November 1 which will contain, for those of you who did not get your hotel card in, another hotel card, and the agenda. This is the tentative agenda. The final agenda will be handed out at the meeting. Please get to your Division Chairman today. If you can't find him today, corral him tomorrow. If you can't find him tomorrow, the names of the various people are in the agenda. If you can't find anybody else, get the information to me, and I will send it to your Division Chairman, but let's get it out there. San Francisco can be a great meeting. In the past, the Western regions have had a slightly lighter attendance than we had in New Orleans. I was not at the Boston meeting. A lighter attendance does not necessarily mean a poor meeting, and, while we anticipate maybe a lighter attendance this time, start twisting arms right now - you know the boss - tell him how you have to go to San Francisco. We will have a tour for you of IBM's campus facilities in San Jose. IBM has a gentleman who has offered to arrange bus service, and I'm going to take him up on that. There will be available at the Data Center, and possibly other Places, computers that include the following: 1130, 360-20, 30, 40, 44, 50, and I think there is a 65 down there, isn't there? All you IBM'ers. You don't know? What's the matter with you guys? Well, anyway, there's up through a 50 down there. There are Data cells; you know, that's the little tweezer gadget. Anyway, the San Francisco Data Center is well equipped. There are a number of pieces of equipment there. You will be allowed to reserve time in the evenings to play games, if you want. You will take care of this, won't you, Pat? It has been suggested, I think casually , that we also, for those of you who are interested, hold a tour of the Napa Valley. I don't know whether any of you know what the Napa Valley is, but that's the greatest wine-growing country in the world, and there are lots of wineries out there. This is by preference at a later date. Anyway, San Francisco, the 11th through the 13th of December. We're going to see everyone of you there. It's the last month of this year. Shortly after Thanksgiving and before Christmas. You can get all your Christmas shopping done in San Francisco and things like that. Get your papers in. There is something to be said about an unstructured meeting, an unplanned meeting, but as an engineer it bugs the heck out of me. I want to see some programs, and you are the people who are making the programs. O.K.? Thank you.

CHAIR -

After that sales talk I give up, but I agree with Bill. There will be a report by me on the 360, 1130 and 1800 CPL minimal standards that have been distributed by PID for those machines. The 1620 would not have received a copy of them. Again, these are minimal standards. I'll tell you what we came up with, why we chose them, and what SHARE'S reaction to them was at the Program Library Meeting tomorrow. I'll try to make that the last part of the meeting, in case there are any conflicts with other meetings. If any of you are interested, or have any ideas at all, about what you want to do with the Program Library - fine. Otherwise, instructions are adequate. Any other business?

FLOOR -

You have listed on your agenda discussions about IBM - customer relations. For those of us who are first attending, I'd like to hear what you have to say about the financial relations.

CHAIR -

Fine. IBM publishes our proceedings for us at the moment. They have suggested that this would be a legitimate fiscal responsibility for us, but quite agree that we are not in position to take it over at the moment. We have to have a reserve in the treasury before we can do that. They maintain our Program Library; there's no question about their continuing to do that. CAST and the Newsletter, which preceeded CAST, and is now part of CAST, have both historically been published by COMMON. IBM provides certain help at these meetings in the sense of IBM personnel - They give us an indirect subsidy in the shape of IBM people attending the meeting and paying the registration fees. There is no direct subsidy from - Oh, Yes, the coffee breaks, audio visual equipment, special equipment of that nature, signs - the administration facilities, for instance, they have a 2400 Zerox in there for us, badges, material of this nature - they do pay for us. I should say that there is no cash subsidy. They help us negotiate with hotels.

JUDI GREENE-

I would like to say thank you to the people of IBM. Those of us who were at the 1130 meetings - we got a lot of very important information, and a good deal of it was impromptu; we appreciate it very much - those people who did speak to us.

CHAIR -

Incidentally, since there are new members present and Arnold Smith was not introduced at the new members' meeting, or at the General Assembly: stand up, Arnie, and take a bow - you and Pat, both.

On the transportation to the boat - there is a map illustrating the route, something like five blocks. I'm told they are rather long blocks, but we had planned that most of the people would walk down there. Cab, that's fine.

FLOOR -

Can we park?

DAVE DUNSMORE -

There is parking on the wharf down there. The boat is to be loaded between 5:30 and 6:30. Cocktail hour until 7:30, and at 10:30 we will be back. I will try to arrange for cabs to be present in that general area, because it is an up-hill walk back.

CHAIR -

If there are no other comments, I'll entertain a motion that we adjourn. Second? All in favor, move.

```
SESSION NUMBER  T.4.2.

DISCUSSION
       PLANNING SESSION FOR FUTURE MEETINGS OF THE EDUCATION PROJECT
    OF THE APPLICATIONS DIVISION.
    ATTENDANCE   20
```

SESSION NUMBER F.1.1

SPEAKERS

W.K. THOMSON

J.R. AHART, INC., DAYTON, OHIO

DISCUSSION

USING 360/30-40 R.P.G. FOR FUN & PROFIT.

MR. THOMSON DISCUSSED HIS COMPANIES EXPERIENCE WITH R.P.G. HE POINTED OUT IT'S POWERS, IT'S SHORTCOMINGS, AND WAYS TO OVERCOME SOME OF THE SHORTCOMINGS. THE PRESENTATION WAS ABOUT AN HOUR IN LENGTH WITH A QUESTION AND ANSWER PERIOD FOLLOWING. APPROX. 50 PEOPLE ATTENDED.

USING 360/30-40 R.P.G. FOR FUN AND PROFIT

PRESENTED AT THE

COMMON

MEETING SEPTEMBER, 1967

CINCINNATI, OHIO

LIAM K. THOMSON
GRAMMER

J. R. AHART, INC.
DAYTON, OHIO

TABLE OF CONTENTS

INTRODUCTION

Using R.P.G. for <u>FUN</u> and <u>PROFIT</u>? Prior to the System "360" Happy R.P.G. users were few and far between, but we feel it has come of age and if given a chance can find its place in any installation. The intent of this paper is to encourage the use of R.P.G. and to pass on the good points and weak points we have found through eight months of use.

There are six different coding sheets used in preparing a program: File Description, File Extension, Line Counter, Input, Calculation, and Output. (See Attachment 1 for examples) The R.P.G. manual has a section for each sheet which covers all the entries for it. We have found no drawbacks in using coding forms over free form which we had been used to.

On our first programs, we seemed to be bumping heads with the I/P, calculation, O/P cycle that is the fixed logic of an R.P.G. program. (See Attachment 2) We felt restricted because we couldn't control the occurrence of the I/P cycle. We have found that by making card, disk, and printer layouts and then planning our programs from these the logic fits many applications very well.

I. COMPILER EFFICIENCIES AND DEFICIENCIES

To date we have found only one logic error in the compiler. If the same name is used with different tag operations on the calculation specifications, it is not diagnosed and the last one encountered is used. This is similar to duplicate statement numbers in Fortran or duplicate labels in Assembly Language.

If the level break field of the first record of a file that has level breaks specified is zero or blank, the level break indicator is not on for the first detail cycle.

The compile time for programs will depend mostly on your peripheral gear. With a (1443 Printer, 1442 Punch, 2501 Reader) a (441) source statement program can be compiled and cataloged in five minutes and seventeen seconds, (60) source statements or less takes two minutes. If you take object decks, the speed of the punch will be the controlling element of compilations. You will get larger object decks from R.P.G. than you experience in other languages because all fields are initialized. To date, for a 32K machine, the largest object deck we have generated is 500 cards and the smallest is 100 cards. Because of the speed of the compiler, we have found it economical on some jobs to use a compile and execute rather than maintain a source and object deck.

II. LANGUAGE PROBLEMS THAT CAN BE SOLVED AT SOURCE LEVEL

Our early programs were frequently canceling due to reading an undefined record type. The HO indicator is used for canceling a job for this reason and eleven others, so the programmer has no way of deciding anything based on HO. We have found that leaving the record identification code blank on single record type files and using a blank specification as the last one of multiple record type file and then checking record type on the calculation specs and taking action from resulting indicators used on input gives the programmer good control. (See Examples)

We found it was very difficult to obtain a skip to a new page and put out headings after level breaks by trying to condition headings on overflow alone. By using the level break indicator at detail time in a (or) situation with overflow - Group Printing Feature - we have eliminated this problem. (See Examples)

We found we were getting some extra pages in reports on which we were doing an internal line count. This is because if a print file has been specified and an overflow indicator is not used as one of the first three indicators of any O/P line the compiler generates an automatic skip to channel one on overflow. We eliminated this by using a dummy line that is never executed with overflow as one of its indicators.

Also there were some unexplainable lines at the beginning of our reports. These were the result of the program looking for first page output and allowing any detail conditions met to print. We find if you have to use a not condition for printing a detail line, it must have N1P specified also to stop its printing on first page O/P.

We noticed that printer output was not at full print speed in programs that had little calculation to slow them down. We found that this is a result of the program having to wait on a line to finish printing in order to check overflow. The use of the line counter specification as an internal carriage tape gives you maximum printer efficiency. Unfortunately, it was not working properly prior to release 11 of D.O.S.

III. CODING TECHNIQUES WE HAVE FOUND TO PRODUCE MORE EFFICIENT PROGRAMS

The GO TO operation on the calculation specification was said to be one of the improvements over 1400 R.P.G. We have found it to a powerful instruction for two reasons. One is probably obvious that is the ability to branch backwards in the calculation coding. But an additional use we find of great value is that it gives you the ability to create your calculation in modular blocks of routines to handle specific parts of a problem rather than use a resulting indicator to allow or inhibit the execution of coding. We found that the latter approach not only produced larger and less efficient programs, but was extremely confusing to write and maintain. (See Examples)

The total time block of coding is entered only once regardless of the number of level breaks that have occurred. All total time calculations that can be done are executed at once, and then all total output is done.

A second improvement over 1400 R.P.G. is the use of multiple input files. We have found several uses of this feature. One is specifying the same file with two different names so you can process it twice in the same program. An example would be where totals are needed to calculate percentages of detail lines. If you specify the same file as a primary and a secondary with matching record, R.P.G. will give you the detail records for a group twice. You can add them as they are processed as a primary file and then calculate percentages and print them as they are processed as a secondary file.

IV. EXECUTION PROBLEMS CAUSED BY D.O.S. DEFICIENCIES

When a disk file is specified as blocked and a full block is not generated one time a record with a key the length of the data is generated and a blank data record written after it.

If a utility created a disk file and it has not been sorted by sort merge, an R.P.G. program will not recognize the end of file written by the utility.

When a file has been specified as an update file, the last record to be updated is not returned to the file.

All of these problems have existing apar's to correct various levels of D.O.S.

V. CHAINING (INDEX-SEQUENTIAL FILES)

If file is unblocked, make the input record size ten bytes larger than the record length to handle records from an independent overflow area.

Be sure to include a LBLTYP card when link editing the program.

If key fields have been packed you can get a no record found condition because the signs are different.

Chaining indicator is specified on chaining file only.

When creating the index sequential file, be sure the key field is reproduced in the record for all files, blocked and unblocked.

If at all possible do not pack key fields, because there is no ability to chain from a card file if keys are packed.

If no record is found HO is turned on.

VI. CHAINING (DIRECT ACCESS METHOD)

The Addrout option of the sort merge program can be used to produce a table of keys and associated track addresses. This table can be used to convert keys to a chaining address through a table LOKUP operation.

This method will allow the user to add records with an R.P.G. program. By putting blank key fields in the original file, when a no record found occurs in the table LOKUP for a key then a LOKUP for a blank key will give a track address for new record, and setting of a resulting indicator will let the R.P.G. program know this is a new record.

VII. TABLES

If a HI/EQUAL or LOW/EQUAL LOKUP is to be performed, the table must be specified as ascending on the file extension sheet or an invalid LOKUP will occur. Also the use of the resulting indicators is <u>low</u> gives next highest value, <u>high</u> gives lowest value.

VIII. ASSEMBLY LANGUAGE SUPPORT BY USE OF THE EXIT INSTRUCTION

We have found that the ability to interface R.P.G. and the assembler language through the RLABL, ULABL, and exit commands is very beneficial. The linkage is not complicated as the examples we have included will bear out. The routines can be in the relocatable library and will autolink.

The first routine we found to be of help was the ability to block and unblock a segment of an input record by a displacement and length. The need for this arose when we wanted to loop on the calculation specifications and move across an input record picking up different fields and returning an updated value. (See Block and Unblock Routines)

Another routine we have made extensive use of was a subroutine that allows the programmer to cause the program to loop between detail calculation and detail output until he resets it. This allows the formated output of tables at any time in the program. (See R.P.G. LOOP)

A great aid in debugging is a routine that PDUMPS the status of all the resultings indicators and the label fields at any time specified. (See PDUMP)

As mentioned before the HO indicator can be turned on for 12 different reasons. Logically, all of these cannot occur in one program but several could at any one time, and no message is issued to distinguish one from another. Therefore we developed a routine to analyze the reason it is on and print it on the console. (See HALT)

Also we found there was a need to give the operator some ability to align special forms prior to starting output. Since you cannot open or close an individual file in R.P.G. we developed a routine to allow some alignment. (See ADJUS)

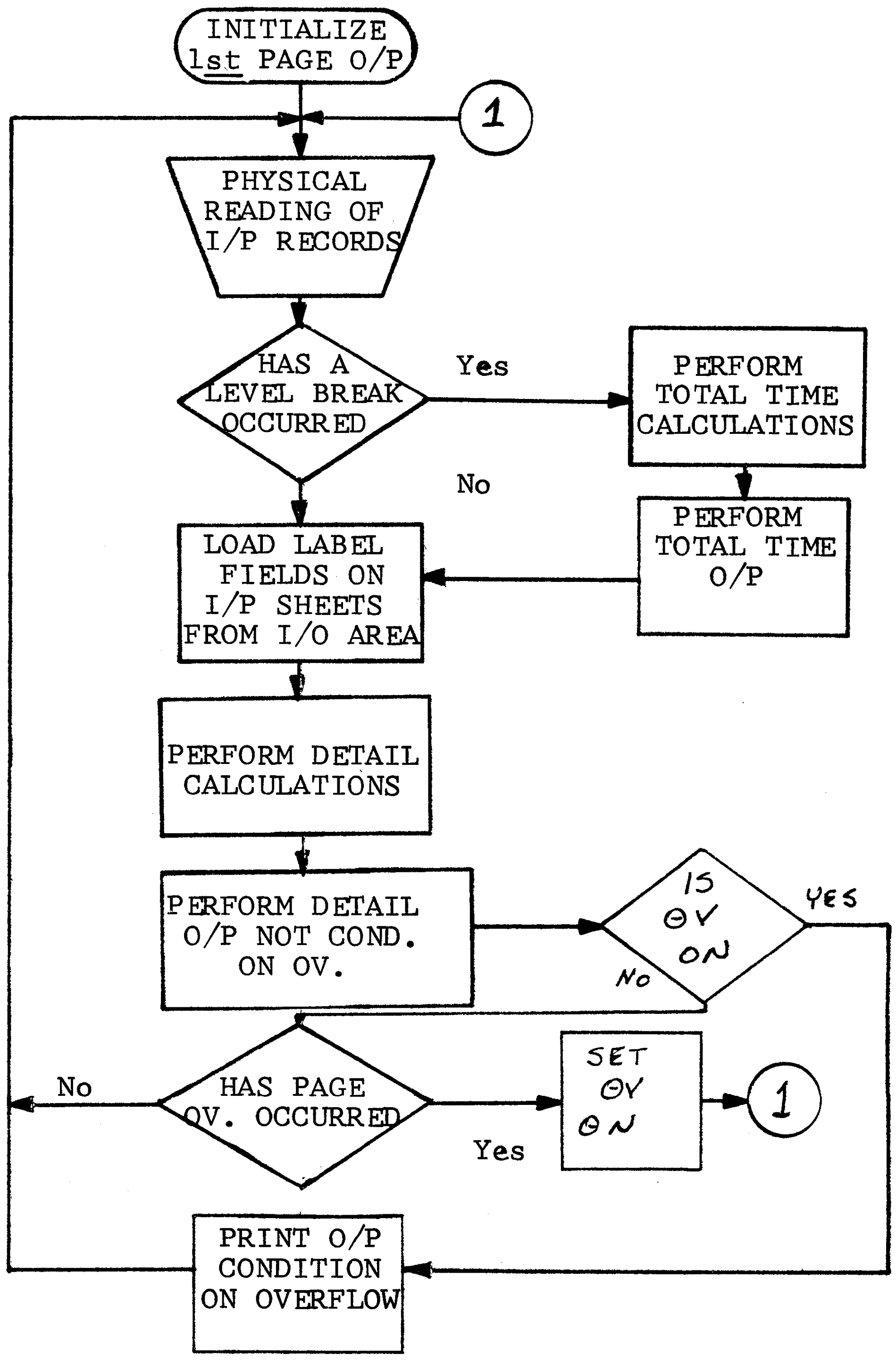
Finally we did not like the idea of having to read one card to get the current date so we wrote a routine to get the date from the communication region. (See DATER)

IX. IMPROVEMENTS WE WOULD LIKE TO SEE.

1. Ability to create and add records to an index sequential file.
2. Ability to control when chaining should or should not occur.
3. Make the date from the communication regions available in the same manner as page number is.
4. Allow the programmer to specify if a table should be filled by R.P.G. or just have space reserved and initialized. This could be done by leaving out the from file name on the extension sheet.
5. Allow a file to be closed and re-opened by the programmer.
6. Don't cancel a program because of a data exception either set field to blanks or zeros or give control to a special routine on calc specs for handling this.
7. Allow program to have a separate block of coding that is executed on first page only. (Calculation and O/P).
8. Use a binary table LOKUP if an ascending or descending table is used.

ATTACHMENT 2

Using D.O.S. 360/30-40 R.P.G.



INITIALIZE 1st PAGE O/P
1
PHYSICAL READING OF I/P RECORDS
HAS A LEVEL BREAK OCCURRED
Yes
No
PERFORM TOTAL TIME CALCULATIONS
PERFORM TOTAL TIME O/P
LOAD LABEL FIELDS ON I/P SHEETS FROM I/O AREA
PERFORM DETAIL CALCULATIONS
PERFORM DETAIL O/P NOT COND. ON OV.
IS OV ON
YES
NO
HAS PAGE OV. OCCURRED
No
Yes
SET OV ON
1
PRINT O/P CONDITION ON OVERFLOW

SUBROUTINE NAME: BLK and UNBLK

LANGUAGE: RPG (16 K.D.O.S.)

PURPOSE: To allow the programmer to retrieve fields from an I/O area by changing a length and displacement value in the calculation coding.

CALLING SEQUENCE:

```
        C               EXIT     BLK       (TO STORE IN I/O AREA)
        6                28      33

        C               EXIT     UNBLK      (TO RETRIEVE FROM I/O AREA)
        6                28      33
        C               RLABL               IPBLK        (see Note 2)
        C               RLABL               PIECE        (see Note 1)
        C               RLABL               LNGTH            30
        C               RLABL               DISP             30
        6                28                 43               31
```

NOTE 1: PIECE may be established as an Alpha or numeric field. If packed data is going to be retrieved PIECE must be numeric, otherwise RPG will pack the field again if it is moved from PIECE.

NOTE 2: IPBLK must be an Alpha field and can have length established on I/P specs.

OPERATION:

BLK Takes DISP and adds it to address of IPBLK and moves from left to right out of PIECE the length of LNGTH into that address.

UNBLK Works the same as BLK except the move is from IPBLK to PIECE.

```
* BLK
* WRITTEN BY R. C. DICE JR. IBM DAYTON
*
BLOK     TITLE '*** GENERALIZED  O/P  BLOCKING ROUTINE -- DOS RPG ***'
BLK      START
         EXTRN IPBLK,DISP,PIECE,LNGTH
         USING BLK,15
         STM   2,5,SAVE
         LM    2,5,REGS
         ZAP   DWRD,0(2,3)         GET 'DISP' (2 BYTES)
         CVB   3,DWRD
         ZAP   DWRD,0(2,5)         GET 'LNGTH' (2 BYTES)
         CVB   5,DWRD
         AR    2,3                 ADD 'DISP' TO 'IPBLK'
         BCTR  5,0                 SUBTRACT ONE FROM 'LNGTH' FOR 'EX'
         EX    5,MOVE              MOVE 'PIECE' TO 'IPBLK(DISP+1)'
         LM    2,5,SAVE
         BR    14
         SPACE
MOVE     MVC   0(0,2),0(4)
REGS     DC    A(IPBLK,DISP,PIECE,LNGTH)
SAVE     DS    4F
DWRD     DS    D
         END
 END OF DATA
```

```
* UNBLK
* WRITTEN BY R. C. DICE JR. IBM DAYTON
*
UNBK     TITLE '*** GENERALIZED  I/P  UNBLOCKING ROUTINE -- DOS RPG **'
UNBLK    START
         EXTRN IPBLK,DISP,PIECE,LNGTH
         USING UNBLK,15
         STM   2,5,SAVE
         LM    2,5,REGS
         ZAP   DWRD,0(2,3)           GET 'DISP' (2 BYTES)
         CVB   3,DWRD
         ZAP   DWRD,0(2,5)           GET 'LNGTH' (2 BYTES)
         CVB   5,DWRD
         AR    2,3                   ADD 'DISP' TO 'IPBLK'
         BCTR  5,0                   SUBTRACT ONE FROM 'LNGTH' FOR 'EX'
         EX    5,MOVE                MOVE 'IPBLK(DISP+1)' TO 'PIECE'
         LM    2,5,SAVE
         BR    14
         SPACE
MOVE     MVC   0(0,4),0(2)
REGS     DC    A(IPBLK,DISP,PIECE,LNGTH)
SAVE     DS    4F
DWRD     DS    D
         END
  END OF DATA
```

```
DOS/360*RPG*CL 2-4              3000                    10/04/67             PAGE 0001

          00 000 H
                 F* TEST RPG LOOPING FEATURE
                 F*
001              FCARDS    IPE F   80  80            READ01 SYSRDR
002              FLIST     O    V 120 120       OV  LPRINTERSYS005
                 F*
003              LLIST     0040106012
                 I*
004              ICARDS    AA  01
005              I                                          1  80 ACARD
                 C*
006              C   01N02                Z-ADD5.        LOOPCT  20
007              C             LOOPCT     SUB  1.        LOOPCT     02
008              C   02                   SETON                     L1
009              C   02                   GOTO LOOPST
010              C                        SETOF                     L1
011              CL1           LOOPST     TAG
                 O*
012              OLIST     D  201    OV
013              O        OR         1P
014              O                                    10 'LOOP TEST'
015              O         D  1      01
016              O                                    25 'DETAIL O/P CYCLE TAKEN'
017              O         T  1      L1 02
018              O                                    27 'LOOP CYCLE AT TOTAL TIME'
019              O                          ACARD    120
020              O         T  1      L1N02
021              O                                    21 'REGULAR TOTAL TIME'
```

347

THIS LOOP IS POSSIBLE WITH THE
D.O.S REL. 13 COMPILER

```
LOOP TEST

  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 1
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 1
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 1
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 1
  DETAIL O/P CYCLE TAKEN
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 2
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 2
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 2
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 2
  DETAIL O/P CYCLE TAKEN
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 3
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 3
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 3
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 3
  DETAIL O/P CYCLE TAKEN
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 4
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 4
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 4
  LOOP CYCLE AT TOTAL TIME           INPUT RECORD 4
  DETAIL O/P CYCLE TAKEN
  REGULAR TOTAL TIME
```

SUBROUTINE NAME: RPGLOOP

LANGUAGE: RPG (16 K D.O.S.)

PURPOSE: To obtain multiple calculation and O/P cycle for a single I/P cycle.

CALLING SEQUENCE:

A. To start a loop

```
C               EXIT SETL
6              28   33
```

COMMENTS: If the loop is set at total time the program will immediately branch to the beginning of the calculation specifications without any further total time calculations or output occurring. The loop will be between detail calculation and detail output.

If the loop is set at detail time the program will next execute the statement immediately following the exit. The loop will again be between the detail calculation and detail output.

B. To end a loop

```
C               EXIT RESETL
6              28   33
```

COMMENTS: If the loop was set at total time the program will immediately return to the total time statement following the exit to setl.

NOTE: The exit to resetl must be at detail time regardless of where the setl occurred.

If the loop was set at detail time the program will return to the statement following the exit to resetl.

OVERFLOW

Output will occur as it normally would.

```
* LOOP
* WRITTEN BY R. C. DICE IBM DAYTON
*
LOOP     TITLE '*** DOS RPG LOOP CONTROL MONITOR ***'
RPGLOOP  START
         ENTRY SETL
         ENTRY RESETL
         SPACE
         USING SETL,15
         SPACE
SETL     CLI   SWITCH,X'FF'
         BCR   8,14
         MVI   SWITCH,X'FF'
         STM   13,14,SAVEREGS
         L     14,X'E8'(3)
         LM    13,14,X'F4'(14)
         STM   13,14,SAVEAREA
         B     MONITOR
         SPACE
         USING RESETL,15
         SPACE
RESETL   CLI   SWITCH,X'00'
         BCR   8,14
         MVI   SWITCH,X'00'
         L     14,X'E8'(3)
         MVC   X'F4'(8,14),SAVEAREA
         LM    13,14,SAVEREGS
         BR    14
         SPACE
SWITCH   DS    X
SAVEREGS DS    2F
SAVEAREA DS    2F
         SPACE
MONITOR  L     7,X'E4'(3)      PERFORM
         L     15,X'B8'(3)        DETAIL
         BALR  14,15                 CALCULATIONS
         L     7,X'E4'(3)      PERFORM
         L     15,X'C0'(3)        DETAIL
         BALR  14,15                 OUTPUT
         TM    X'157'(13),X'02'   TEST FOR
         BZ    18(14)                OVERFLOW
         L     7,X'E4'(3)      PERFORM
         L     15,X'114'(3)       OVERFLOW
         BALR  14,15
         SPACE
         BALR  14,0            BRANCH BACK TO MONITOR
         SH    14,6(14)
         BR    14
         DC    H'40'
         END
 END OF DATA
```

SUBROUTINE NAME: PDUMP

LANGUAGE: RPG (16 K D.O.S.)

PURPOSE: To obtain program status at selected points during execution.

CALLING SEQUENCE:

```
C                    EXIT PDUMP
6                   28   33
```

RETURN: The general registers and the entire indicator and symbol tables are dumped on the printer. Processing resumes at end of dump.

```
* PDUMP INDICATORS AND LABELED FIELDS OF AN RPG PROGRAM
* WRITTEN BY R. C. DICE JR. IBM DAYTON
*
PDUMP    START
         USING PDUMP,15
         STM   0,1,SAVE
         ST    3,START
         MVC   FINISH,X'94'(3)
         LA    0,START
         LA    1,ROUTINE
         SVC   2
         LM    0,1,SAVE
         BR    14
SAVE     DS    2F
ROUTINE  DC    C'$$BPDUMP'
START    DS    F
FINISH   DS    F
         END
 END OF DATA
```

SUBROUTINE NAME: HALT

LANGUAGE: RPG (16 K D.O.S.)

PURPOSE: To allow programmer to determine the reason HO has been turned on.

CALLING SEQUENCE:

```
              C       I  I       EXIT HALT
              6      10 11      28   33
                                II is any indicator(s)
```

RETURN: Typed Message of Cause of Error

```
H0 ON if cause is not known.
H1 ON if undefined record read.
H2 ON if sequence error.
H3 ON if no record found. (Index Seq. or Direct
                           Access)
H4 ON if wrong length record check.
```

```
* HALT  ANALYSIS PROGRAM
* WRITTEN BY R. C. DICE IBM DAYTON
*
HALT      START
          USING *,15
          STM   1,9,SAVE
          COMRG
          LA    2,RESULT-1
          LM    4,8,REGS
          L     9,ADDR
          MVC   NAME,24(1)
          CLC   288(3,3),ZERO
          BC    8,WRITE2
          AR    2,6
LOOP1     IC    8,LENGTH(7)
LOOP2     CLC   288(1,3),0(2)
          BC    8,WRITE1
          AR    2,6
          AR    4,5
          BCT   8,LOOP2
          AR    3,6
          BCT   7,LOOP1
WRITE1    AR    4,9
          ST    4,ADDR
WRITE2    EXCP  CONSOL
          LA    7,X'85'
          COMRG
          AH    7,8(1)
          AH    7,44(1)
          IC    8,13(0,2)
          AR    8,7
          NI    0(7),X'00'
          OI    0(8),X'F0'
          WAIT  CONSOL
          ST    9,ADDR
          LM    1,9,SAVE
          BR    14
CONSOL    CCB   SYSLOG,CONSOLE
CONSOLE   CCW   9,MSG,X'60',1
          CCW   1,MESG,X'60',33
ADDR      CCW   9,MESSAGE,X'60',24
          CCW   9,MSG,X'00',1
FIX       DC    A(MESSAGE)
SAVE      DS    9F
REGS      DC    4F'24,24,1,3'
ZERO      DC    F'0'
LENGTH    DC    X'00060304'
RESULT    DC    X'02100408'
          DC    X'804020'
          DC    X'FF8040200410'
HALTINDC  DC    X'0001000203000400000400000300'
MESG      DC    C'***** '
NAME      DS    CL8
MSG       DC    C' TERMINATED DUE TO '
MESSAGE   DC    C'PROGRAMMER REQUEST      '
          DC    C'INVALID CHAINING REQUEST'
          DC    C'UNDEFINED RECORD TYPE   '
          DC    C'SEQUENCE ERROR  (MR)    '
          DC    C'RECORD SEQUENCE ERROR   '
          DC    C'DAM RECORD NOT FOUND    '
```

```
                DC    C'DAM DATA CHECK              '
                DC    C'DAM WRONG LENGTH RECORD     '
                DC    C'ISAM INVALID KEY LENGTH     '
                DC    C'ISAM DASD ERROR             '
                DC    C'ISAM WRONG LENGTH RECORD'
                DC    C'ISAM -EOF- WITHIN LIMITS'
                DC    C'ISAM DUPLICATE RECORD       '
                DC    C'ISAM NO RECORD FOUND        '
                DC    C'NO APPARENT REASON          '
                END
        END OF DATA
```

<u>SUBROUTINE NAME:</u> DATER

<u>LANGUAGE:</u> RPG (16 K D.O.S.)

<u>PURPOSE:</u> To retrieve the date from the communications region.

<u>CALLING SEQUENCE:</u>

```
          C          ULABL                  DATE        8
          6         28                     43          51

          C          EXIT DATER
          6         28   33
```

<u>RETURN:</u> Date as it appears in communication region (MM/DD/YY).

```
* DATER
*
* PICK UP DATE FROM COMRG FOR RPG
*
DATE1     START
DATER     BALR  15,0
          USING *,15
          STM   1,14,HDRG
          COMRG
          MVC   DATE,0(1)
          LM    1,14,HDRG
          BALR  14,14
          ENTRY DATE,DATER
HDRG      DS    14F
DATE      DC    CL8' '
          END   DATER
 END OF DATA
```

```
* ADJUS
*
* WRITTEN BY R. C. DICE JR. IBM DAYTON
ADJS     TITLE '''ADJUST'' SUBROUTINE TO ALIGN FORMS'
ADJUST   START
         USING ADJUST,15
LOOP     LA    1,PRINT                 PRINT A LINE ON THE PRINTER
         EXCP  (1)                       AND SKIP TO CHANNEL 1
ERR      LA    1,CONSOLE               ASK OPERATOR IF RETRY DESIRED
         EXCP  (1)                       AND READ HIS REPLY ('Y' OR 'N')
         WAIT  (1)                     WAIT FOR REPLY
         OI    REPLY,X'40'
         CLI   REPLY,C'Y'
         BE    LOOP                    'Y' - GO PRINT ANOTHER LINE
         CLI   REPLY,C'N'
         BNE   ERR                     NOT 'N' - ASK AGAIN
         BR    14                      'N' - ALL DONE
         SPACE
PRINT    CCB   SYSLST,PR
PR       CCW   X'89',LINE,X'00',L'LINE      PRINT, SKIP TO CH. 1
LINE     DC    C'01234567890123456789'
         SPACE
CONSOLE  CCB   SYSLOG,CON
CON      CCW   X'01',MSG,X'40',L'MSG+1      WRITE, NO AUTO-RETURN
         CCW   X'0A',REPLY,X'00',1          READ INQUIRY
MSG      DC    C'PRINTER RETRY  (''Y'' ''N'') ='
         DC    X'16'                   BACKSPACE
REPLY    DS    C
         END
  END OF DATA
```

SESSION NUMBER F.1.2

SPEAKERS

MAINTENANCE OF AN INSTALLATION PROGRAM LIBRARY

ARNE HOVERSTAD
JOHN JOHNSON
DICK THOMAS

Requirements for the Program Library

Common Meeting September 6, 7, 8, 1967

Arne Hoverstad
John Johnson
Dick Thomas
Federal Reserve Bank of Minneapolis

I would like to open my breif comments with two disclaimers. First, since I work for a Quasipublic institution, the FRB of Minneapolis, it is customary for me to point out that the views and beliefs here expressed are those of the author and not necessarily those of the FRB of Minneapolis. Second, in standing up and talking about the way that we handle our program library, we don't mean to imply in any way that we have this particular problem completely under control. However, we do have some techniques that we use, some records that we keep and we find them helpful to us.

We think of the program library as requiring two kinds of attention; planning and discipline. Under planning it is necessary or at least desirable to make some formal effort at describing the conditions, the environment, and the output that you want from your program library. It is certainly worthwhile to make an effort at writing down a formal objective, something more comprehensive than "we want a program library." This can be sort of a "chicken and egg procedure" with the other considerations that must go into the library. Let's list and comment briefly on each of those considerations.

1. What are the contents of the library?

The answer seems at first obvious. It is a program library; therefore, it contains programs. However, there is more than one kind of program. For instance, our own library is divided into IBM programming systems, common programs, user-written programs, (that is our own programs to perform our own data processing work,) and our own subroutines that are common to several of our programs. In addition, we maintain in the program library a disk register; (a list of the usage of the 20 or so 1316 Disk Packs that we carry,) and a program name register; this last to avoid confusion between programs. The user might also consider filing data in a program library. We at the moment file our data separately. This is historical data, cost data, in our case economic research data. We will probably continue to file data under a specific and separate system although there will be some direct responsibility for the program librarian for this sort of thing.

2. The second major consideration, it seems to us, is the system of filing that will be used. Should we generate some sort of classification numbering system similar to that used in the common library? Should we file programs by name or title alone? Should perhaps programs be filed under a major classification of the disk pack on which or against which they are run?

We file our program decks for the most part according to the disk pack, since we feel that the time that we are in a real hurry to get to a program or a set of programs is when we have just cobbered a disk pack and it must be completely reloaded. We do some cross-indexing by purpose in order to get at other "natural sets" of programs.

3. Security is a problem to any installation. How much security does the installation require? We have two kinds of security problems at our installation. One kind we do not have is the question of proprietary programs, such as exists in some of the chemical or other processing industries. We do, however, have data that cannot be regarded as being in the public domain, that is individual transaction or balance data from commercial banks which they would be very unhappy if we published or allowed to be published, and of course we have the problem of providing security against loss of data or programs. We handle the first problem by purely informal methods. All people who work with or who have access to this data are made aware of the consequences and required security level of the numbers. The second problem we handle rather conventionally with back-up files. We have available to us a large and secure vault and maintain updated copies of programs and data within the vault. We update this backup copy at least once a month, more often if we develop a large volume of changed programs or additional data.

4. The fourth consideration and in many ways the most important is what should a program librarian be, and this we really don't have an answer for. We have tried using one of our more senior programmer analysts as a program librarian on a part-time basis. This has not worked very well. He has many other things to do and almost all of them are more interesting than working on and maintaining a program library. For the past year, we have had a non-programmer, non-data processer, non-analyst as a program librarian. He is an older man and has had a great deal of experience in the bank in senior clerical and first-line supervision positions. Our experience has been good. A man of this kind can and will do the work, shows the responsibility that is required for work of this nature and we think that this may be the answer to the problem. He requires a great deal more guidance than a trained or experienced programmer analyst might. This guidance can sometimes get to be a bore but at least something is getting done.

Planning a program library is quite easy, actually, anyone of us, I am sure, can sit down and draw up a list of specifications and filing procedures, and considerations and personnel qualifications for an ideal program library. The real secret we feel is discipline. If the staff and management of an installation is convinced that the program library that they have specified is an important and vital part of their work and are willing to devote time and attention in limited quantities to its successful maintenance and continuation, then there exists no problems. Unfortunately, the reverse seems to be true. Once a program is done, the problem is solved, the exciting part is over, it is difficult to maintain interest in seeing that the results are properly recorded and kept track of. Only if a data processing management keeps considerable pressure and maintains a continuing interest in the program library will any scheme or design no matter how beautiful be successful. This I think is our message. A program library is a necessary evil and unless you worry about it every day you are not going to have a workable one.

Thank you very much.

SESSION NUMBER F.1.4

SPEAKERS

MR. D. DUQUETTE, FORDHAM UNIVERSITY
MR. P. FALCONELLO, FORDHAM UNIVERSITY ON A COORDINATED STUDENT DATA SYSTEM FOR COLLEGES AND UNIVERSITIES ITS PHILOSOPHY, METHODOLOGY AND APPLICATIONS.

FORDHAM UNIVERSITY COMPUTING CENTER

BRONX, NEW YORK

A COORDINATED STUDENT DATA SYSTEM

by

P. Falconello and D. Duquette

FORWARD

Our intention in the presentation of this paper is to give a brief discussion of a COORDINATED STUDENT DATA SYSTEM FOR UNIVERSITIES. Our source of material is the experience which was gained throughout the development of various University EDP systems. The major applications which contributed to this effort were the Admission's System, the Registrar's System and the Alumni System Therefore, a discussion of these systems forms the nucleus of our presentation. It should be mentioned that this presents only initial thinking on a proposed CSDS.

STUDENT DATA

Student data may be defined as that personal, academic and financial information of each student which is vital to the daily operations of the various University departments.

It is our purpose to discuss an approach to a unified coordinated system of centrally maintained student data. Our basic premise is that our current EDP systems of admission, registration and alumni represent a tangible parallel to the input, processing, and output of such an integrated system.

FILES

The Coordinated Student Data System contains three data files.

1. Admission Master File
2. Student Data File
3. Alumni Association File

The system is based on the sequential flow of data through these files and the use of such data by the various University departments.

IDENTIFICATION

The student records in each of the three above files will be identified by SSN. For those who do not have SSN, because they are members of a religious order or because they are foreign exchange students, the Computing Center can generate a set of SSN that are invalid by the Social Security Administration standards, but are valid in the Coordinated Student Data System. This can be accomplished by generating a group of numbers and setting the first three digits to zero. Lists of available invalid SSN are prepared by the Computing Center and sent to the Admission Office. As a valid SSN becomes available to any of the offices they can inform the Computing Center of the new SSN and the change is made, at the same time the Computing Center will prepare a printed sheet of the updated record and forward it to the originating office.

SOURCE DATA

The original source of primary information to the

Student Data File (SDF) is traced to the source documents which create the Office of Admissions Master File (AMF) of applicants. The documents which create this file, their source, and the information which they provide, are as follows:

1. Application for Admission to the University.

A formal application is submitted to the Office of Admissions by an applicant. This form should be designed to satisfy the need for a document of both detail and utility. It should provide for that detailed information which is needed for admission decisions and also facilitate the key punching of basic information which is carried in the AMF. This formal application is submitted to the Office of Admissions by the applicant. After conversion to machine sensible form, an additional record is generated in a subsequent update of the AMF. The information extracted from the admission application will comprise the primary personal and academic data of each record on the SDF.

2. High School Transcript.

From this transcript, the applicant's high school academic history is entered in the AMF and subsequently carried to the SDF. The transcript may be available in punch card form from automated secondary school systems.

3. CEEB Scores.

The applicant's SAT and Achievement scores are entered in the AMF directly from Educational Testing Service (ETS). These scores are currently available in punch card form and the majority of these cards contain a social security number. For those cards which have a blank SSN field, a separate file can be created which is periodically matched, by name, against the AMF.

4. Decision Card.

When the Office of Admissions arrives at a decision regarding an applicant, a Decision card is submitted for updating the status of his record on the AMF.

5. CSS Financial Need Analysis.

CSS Financial Need Analysis con't.

Initial information regarding a candidate's scholarship application is entered in the AMF via a CSS card from the Office of Financial Aid. The CSS (College Scholarship Service) card is currently available from ETS in punched card form, and usually contains a SSN.

6. Pre-registered Transaction.

At the receipt of an applicant's deposit on tuition, the Bursar will provide a transaction card which will update the applicant's record status, thereby changing him to the category of "student". This pre-registered card may be pre-punched at "accept" decision time and enclosed with the Letter of Acceptance forwarded to the student.

7. Scholarship/Loan Transaction

From the office of Financial Aid, a transaction card which provides the type and amount of aid which has been awarded to the candidate, is used to update his record on the AMF. This transaction card may be a pre-punched mark-sense card which is prepared prior to committee meeting.

8. List of State Approved Loans.

From the individual states, a list of students and the amount of loans authorized, is provided to the University. The student record is updated with the current amount and the cumulative amount is subsequently brought forward to the SDF. This procedure may also be facilitated through the use of punch cards made available by some states.

9. Transfer Status Transaction.

In the case of transfer students who are accepted with advance standing, a Transfer Status Transaction is forth coming from the Office of Admissions after evaluation of credits to be transferred. This transaction may be in pre-punched, mark-sense card form, created during the update in which a transfer applicant is entered on the AMF. This information should be supplemented with the actual courses transferred when transition is made to the SDF.

TRANSFER OF DATA BETWEEN FILES

At the end of the Admissions processing cycle, the Computing Center will update the SDF with the pre-registered students from the AMF. This is the first of two file transition points in the system. In this case, pre-registered students on the AMF are physically added to the SDF. The records of transfer students with advance standing will be supplemented with those courses and grades which have been accepted for transfer. This transition takes place immediately prior to registration time. Those records which are not added to the SDF may be dumped to a card file for later use in statistical analysis.

Registration affords the major input of primary source data to the SDF. In most cases, freshman registration will be the only time that new personal data is entered into the file. Normally, only changes to personal data will occur at subsequent registration. Courses registered for, and class schedules, represent the input of primary academic data at registration. Whether registration requires physical attendance in all cases, or is accomplished through a pre-registration with a latter mail registration, a simple well-designed form and class cards will facilitate updating the SDF.

Some other primary sources of input to the SDF are:

1. The course grades which are provided on the class cards at semester end.
2. Requests for financial aid, such as scholarships, loans, and grants. In such cases additional financial information is entered into the SDF by cards, scholarship loan cards, state loan cards, etc.
3. Participation in student activities, such as societies, sporsports, clubs, etc., causes random updating of such information.

The other transition point in this system occurs at the end of the registrar's processing cycle. The Computing Center will process the SDF by updating class standings for all students on file. At the same time the Alumni File will be updated with graduating seniors from the SDF. Basic information such as ID number, name and address, student's school attended, year of graduation, degrees awarded extra curricular activity and other information will comprise the basic input to the Alumni File. Additional information will be entered to the Alumni File by the Placement Office and the Alumni Association. Through the use of specially designed forms the Placement Office will make available information on students dealing with that office, such as business address, industry classification, job classification and level of position. The Alumni Association through the use of questionnaires will serve as another means of acquiring the updating business information of the Alumni.

MAINTENANCE

In each of the three files, responsibility for maintenance falls on the corresponding Office of Primary Responsibility (OPR); that is, the Admissions Office is responsible for maintaining the Admissions file, the Registrar's Office is responsible for maintaining the Student Data File, and the Alumni Office is responsible for the Alumni Association file.

In order to facilitate the maintenance of the file, the Computing Center, in cooperation with each of the above mentioned offices, can develop a number of data forms. These should be designed to alter or add any or all information contained in the records. As changes or additions become necessary, the Office of Primary Responsibility fills out the proper forms and sends them to the Computing Center, where they will be key punched and processed against the proper file. While updating the files, updated records are printed and sent back to the originating department. These records are then added to master books that are periodically prepared by the Computing Center for each of the three offices.

USES

The uses of the Coordinated Student Data System are many. We will attempt to list the most important of these uses and briefly describe each application.

A. Admissions.

The following is a list of products and uses of the AMF by the Office of Admissions:

1. Missing Information Lists.

On a scheduled basis or as required, a listing can be produced which itemizes any critical information which may be missing from an applicant's record.j This list can be used by the Office of Admissions to view the conditions of the file and send out request for information when needed. Work loan peaks resulting from rapid growth periods, can be alleviated by producing such reminders by computer and mailing them direct to the applicant.

A further use of missing information lists is found in the processing of special applicants. In the case of foreign students, religious, veterans and other, a decision may be reached promptly when the presence of all critical information is deemed impossible.

2. Summary Information Labels.

A summary of information on each applicant can be produced as his record becomes complete. This data can be printed on labels, index cards, lists, etc., and forwarded to the Office of Admissions to be entered on the applicant's folder. The type of information contained on a label may be Name, date, address, high school, SAT and Achievement scores, High School average, program applied for, predicted QPI and other significant data. Subsequent labels may be produced by a change in any information h shown.

Occassionally, these labels may be used by other departments, such as the Deans of the various schools. For example, the class sectioning of entering freshmen Biology majors can be facilitated with such handy information.

3. Decision Cards.

Concurrently with producing the first summary label, a decision card can be pre-punched with ID information. This card is then filed into the applicant's folder to be later mark-sensed with the Committee on Admissions decision. The Decision card will then be used to update the applicant's record on the AMF.

4. Pre-Registered Transaction Cards.

A pre-registered card may be mailed to each applicant with the Letter of Acceptance. This pre-punched card may be produced simultaneously with the decision card or it may be punched only after an "accept" decision has been made. The card will then accompany the student's deposit to the Bursar and, thereafter, be used to update the AMF.

5. **Advance Standing Card.**

A pre-punched mark-sensed card can be produced and forwarded to the Office of Admissions at the time a transfer student is accepted. After evaluating the credits offered for transfer, the advance standing card can be marked with those courses accepted for transfer and used to update the SDF after the AMF transition.

6. **Letters of Acceptance/Rejection.**

During the AMF update at which a decision card is entered, a letter of decision may be generated to be mailed directly to the applicant.

7. **Admissions Master File Analysis.**

On a scheduled basis, such as weekly or semi-monthly, a YTD comparative report may be produced as a by-product of the file update. Such a report can be a cincise analysis of the file. Totals and other ratios of current year to prior year can be presented for all combinations of programs and for each category of record status. This report provides a most effective decision-making tool which may be utilized in maintaining control of class sizes, preparing progress reports, and determining teacher scheduling and resource planning.

8. Mailing Labels and Addressed Envelopes.

A significant advantage of a computerized admission system is the efficient production of bulk mailings. Mailing labels or addressed envelopes, or cards, maybe generated for any of all portions of the file.

9. High School Applicants' Lists.

For each High School, a list of graduates who have applied to the University can be prepared. Such a list may give the applicant's name, the action taken, and whether a scholarship was given. Aside from answering inquiries from the High Schools, the list can prove helpful during recruiting time.

10. Master Book.

An edited listing of the entire file may prove helpful in answering random inquiries. It may also suffice as a record of changes made, if none is otherwise prepared.

11. Statistical Reports.

A wide variety of statistical reports can be generated as a by-product of maintaining detailed records on all applicants. The following is an example of a few such statistics.

a) Frequency distributions based on Verbal and Math SAT scores, and rank in class can be generated for each program of each school by appoied, accepted and pre-registered applicants.

b) Geographic and demographic analysis of quantity and quality are often helpful.

c) Correlations of actual versus predicted QPI for the various schools, programs, etc. give an indication of the validity of such a factor.

B. Registrar

The uses that the Registrar Office can make of the SDF are many. Some of the major uses are as follows:

1. Class Lists.

These lists are prepared at the beginning of each semester; they may contain the following information: course name, title and credits, instructor's name, time and place of meeting, and an alphabetical list of each student taking the course. These lists can be used by the instructor to record the student progress during the term.

2. Class Cards.

Class cards are produced in course order at the beginning of each semester. For each course, a card for every student taking that course is produced. The cards may contain the student's name and number, and the course received by students on the SDF.

3. Student Schedules.

Student schedules may contain the following information: student name and Id number, student address, and a list of courses with time and place of meeting. These schedules are mailed to the students and another set is retained by the Registrar for their records.

4. Grade Reports.

At semester's end, grade reports are produced. Each grade report may contain the following information: student name and ID number, student address, and a list of courses with the corresponding grade, number of credits, YTD, and cumulative credits, quality points and indices. One copy of these grade reports is mailed to the student, and the other is retained by the Registrar for their records.

5. Transcripts.

Transcripts reflecting a student's achievements during his scholastic career are produced periodically. These transcripts may contain the following information: student name and ID number, student address, and a list of courses with their corresponding description and the grades received for these courses, any honors awarded, etc.

6. Ranking Lists.

Lists in YTD and cumulative index order are produced, showing rank in class for every student on the SDF.

C. Deans and Department Chairmen.

1. Grade Analysis.

The grade analysis report is produced at the end of each semester in order to aid the Deans in evaluating individual teacher performances. For each course, the grade analysis report contains the following information: course name, instructor's name, number of credits, the number of students taking the course and a distribution of grades awarded by the teacher; that is, the number of "A's" given, the number of "B+'s" given, the number of "B's", etc.

2. Instructor's Schedule.

Instructor's schedules may contain the following information: instructor's name and a list of courses the instructor is responsible for teaching. Each course will contain the time and place that it meets. These schedules may be used in locating individual instructors.

3. Honor Lists.

Honor lists can be prepared showing the names of students eligible for awards.

4. Problem Students.

Lists of students whose index is below a certain level can be produced periodically and forwarded to the Dean's Office, where individual attention may be given these students.

5. Graduating Seniors.

Complete records on graduating seniors can be produced to determine if any deficiencies exist in their programs.

6. Test Scoring.

Pre-punched test scoring cards may be produced for any course by request of the instructor. These cards may contain the ID number, student name, and course number. They will be used as input to a test scoring program.

D. Library

1. Student Master File.

A Student master file can be made available to the library. This master file may show the student's ID number, address, etc. It may be used to verify student status.

2. Reminders.

Reminders on loaned books and overdue notices may be produced for the library. This should greatly facilitate collection of loaned books.

E. Physical Plant.

1. Room Schedules.

Room schedules reflecting its use can be made available to physical plant in order to facilitate service schedules and allocation of rooms on special requests.

F. Central Mailing

Central mailing may use the CSDS to produce mailing labels for all general and selective mailings.

G. Alumni Association.

A major report which may be produced from the Alumni file is missing information lists. Periodically missing information lists, designed to facilitate the collection of data to the Alumni file, can be produced. A typical missing information list might be one containing all Alumni who do not have a telephone number on file.

1. Directories.

Alumni directories can be alphabetical, geographic, or class and school produced from the Alumni file.

2. Fund Raising

Of all the areas in which the Alumni file is used by the Alumni Association, fund raising is undoubtedly the most important. For this reason, a more detailed account of this application will be given.

a) Worker Assignment Lists.
Lists of all Alumni with a better than average giving history can be produced. These lists may be used to aid in recruiting workers that will participate in the annual fund raising campaign.

b) Phonothon Pledge Cards.
The phonothon pledge cards may consist of two parts. One part may contain personal information, giving the history of the alumni; and the other may contain his name, address, and a place to write a pledge. These cards are used by the workers to contact the alumnus by telephone and try to solicit a pledge.
One part of the pledge card is mailed to the Alumnus reflecting his amount pledged; the other part is used to enter the result of the telephone conversation in the Alumni File.

c) Reminders.

During the campaign, monthly reminders can be produced from the Alumni file. These reminders are produced for all alumni who have not fulfilled their pledge. They show the amount pledged, the amount given to date (if any), and a balance due. Reminders are then mailed to the Alumnus in the hope that a payment for the amount pledged will be sent.

d) Receipts.

Daily receipts of payments can be produced and mailed to the donors at the same time that the file is updated.

e) Block Sheets.

Block sheets may be listings showing a detailed list of donors and amount given by each donor. They also show the total daily intake.

f) Worker Reports.

Worker reports can be produced periodically to report to the individual workers the progress of their group. These reports may contain the worker's name, followed by a list of prospects, the amount pledged, and the amount given by each prospect.

g) Progress Reports.

Reports can be produced to reflect the amount of progress to the campaign director. They may contain the names of all alumni who have pledged and the amount given by the alumni, if any. Totals such as total pledge, total given, total number of pledges and total number of donors can also be produced to reflect the overall progress.

h) Year-end Analysis.

A detailed year end analysis reflecting the fund raising campaign can be produced. This analysis may range from a general university-wide performance analysis to a selective geographic and school-class analysis.

H. Bursar

The office of the Bursar can extract information from both the Admission Master File and the Student Data File.

Totals and exception lists may comprise the major use of the Admissions Master File by the Bursar. Receipts of tuition payment may be produced by computer for the Bursar, but often, the students cancelled check is regarded as sufficient receipt. A list of all pre-registered Freshman, including amount of tuition prepaid, may be produced prior to registration.

The following are two uses of the SDF by the Bursar:

1. Tuition and Fees Payment Transaction.

Prior to registration, a set of pre-punched payment cards may be produced which contain the type and amount of any financial aid authorized. These cards may then be used to record any payment made on tuition and fees, and later used to update the SDF.

2. Post-registration Totals.

After registration, a summary of totals can be provided for the Bursar. Totals, such as amount of tuition and fees paid and balances due, can be helpful in verification of account balances; as well as enhancement of the internal control.

I. Financial Aid

The Office of Financial Aid may utilize both the AMF and the SDF. The following may be output from the AMF regarding incoming Freshman.

1. Summary Information Labels.

A label of summary information can be provided for all applicants who have been accepted for admission. This information can be used for cross reference on all who apply for financial aid.

2. Financial Aid Applicants' List.

Prior to the Financial Aids Award Committee meeting, a list can be produced of those applicants for which a CSS card has been received. This list may be ordered by predicted QPI, adjusted need, or some other criteria. It should contain the pertinent personal, academic, and financial information which is needed to make award decisions.

3. Financial Aid Autherization Transaction.

A set of marksense cards may be punched concurrently with the financial aid applicants list for use by the awards committee. As awards are made, the amount and type may be marked on the card for later use in updating the AMF.

4. Financial Aid Award Letters.

As the AMF is updated with the marksense authorization cards, a letter of award can be prepared for mailing to the student.

5. Financial Aid Candidate List.

After the pre-registration deadline, a list of those students who have accepted the awards can be produced as a follow-up procedure. This listing may be used as a final office copy for report preparations and inquiries.

6. Summary Totals.

In conjunction with producing the candidate list, a complete register of summary totals can be periodically printed. Such item and amount accumulations will be very useful in maintaining control of funds as well as preparation of required Federal and State reports.

The SDF is used for processing undergraduates by the Office of the Fimancial Aid in essentially same manner as the AMF is used for processing incoming freshman. The significant difference is the use of University academic data in lieu of high school data and board scores.

J. <u>Clubs, Societies, Associations, Etc.</u>

The SDF may be used by the various University clubs, societies, and other organizations whose end may be academic, athletic, social, etc. A variety of information is available for soliciting potential members, or contacting present members. The major use then is special mailing labels or pre-addressed envelopes.

K. **President's Office**

The office of the President, or the Board of Trustees, require the additional advantage of an information system over a record keeping system. That advantage is the developement of informative statistics from which decisions on policy and future planning can be made. Prior year to current year comparison totals and ratios, with the myriad of variables, could render the recordkeeping a small portion of the system. For example, to know the relative quantity of applicants requesting dormitory residence over commuters could facilitate the decision to build a new dormitory.

CONCLUSION

A Coordinated Student Data System must encompass all offices of the University that have a need for student data. Such an all-inclusive system will enjoy the following advantages. Non-duplication of effort is avoided through singular introduction of source data. Through the establishment of Offices of Primary Responsibility (OPR), ease of file maintenance is achieved. Economic availability of data for all University departments is gained through centrally maintained student data. Consistency of information is assured through the inherent compatibility of common source data. Last, but not least, the University will realize an abundance of research data to be used in the analysis of University performance.

With the addition of the necessary terminal equipment; the CSDS is potentially a real-time system, thus providing the special advantages of such a system.

STUDENT DATA FILE

Sample Information

	DATA	TYPE	PRIMARY SOURCE	FORM	OFFICE OF PRIMARY RESPON.
1.	Social Security Number	C	Admissions Application	KP	Registrar
2.	Name	C	Admission Application	KP	Registrar
3.	Home Address	C	Admission Application	KP	Registrar
4.	Local Address	C	Registration Form	KP	Registrar
5.	Residence Code	C	Registration Form	KP	Registrar
6.	Phone Number	C	Registration Form	KP	Registrar
7.	Sex	C	Admission Application	KP	Registrar
8.	Campus Residence Mail Box Number	C	Registration Form	KP	Registrar
9.	Date of Birth	C	Admission Application	KP	Registrar
10.	Marital Status	C	Admission Application	KP	Registrar
11.	Veteran Status	C	Admission Application	KP	Registrar
12.	Citizen	C	Admission Application	KP	Registrar
13.	Birth Place	C	Admission Application	KP	Registrar
14.	Parent/Guardian Name	C	Registration Form	KP	Registrar
15.	Parent/Guardian Address	C	Registration Form	KP	Registrar
16.	Father/Mother Alumnus	C	Admission Application	KP	Registrar

	DATA	TYPE	PRIMARY SOURCE	FORM	OFFICE OF PRIMARY RESPON.
17.	Spouse's Name	C	Registration Form	KP	Registrar
18.	Religion	C	Registration Form	KP	Registrar
19.	Student Hospital Insurance	C	Registration Form	KP	Registrar
20.	Student Activities	C	Student Affairs Form	KP	Registrar
21.	High School Name	A	Admission Application	KP	Admissions
22.	High School Address	A	Admission Application	KP	Admissions
23.	High School CEEB	A	Admission Application	KP	Admissions
24.	Type of High School	A	Admission Application	KP	Admissions
25.	Year of Graduation (High School)	A	Admission Application	KP	Admissions
26.	High School Average	A	High School Transcript	KP/PC	Admissions
27.	High School Rank	A	High School Transcript	KP/PC	Admissions
28.	SAT Scores	A	ETS CEEB Card	PC	Admissions
29.	Achievement Scores	A	ETS CEEB Card	PC	Admissions
30.	Transfer Status	A	Admission Application	KP	Admissions
31.	Year of Graduation Present Program	A	Admission Application	KP	Admissions
32.	School Enrolled	A	Admission Application	KP	Registrar

	DATA	TYPE	PRIMARY SOURCE	FORM	OFFICE OF PRIMARY RESPON.
33.	Program Enrolled	A	Admission Application	KP	Registrar
34.	Teacher Certification Program	A	Admission Application	KP	Registrar
35.	Matriculated	A	Registration Form	KP	Registrar
36.	Courses Currently Registered for	A	Registration Form	KP	Registrar
37.	Class Schedule of above	A	Grade Report	KP	Registrar
38.	Grades	A	Grade Report	KP	Registrar
39.	Loans Authorized	F	Financial Aid Form	M/S	Financial Aid
40.	Scholarships	F	Financial Aid Form	M/S	Financial Aid
41.	Work Grants	F	Financial Aid Form	M/S	Financial Aid
42.	Grants in Aid	F	Financial Aid Form	M/S	Financial Aid
43.	Family Financial Status	F	ETS CSS Card	PC	Financial Aid

SESSION NUMBER F.1.5

SPEAKERS

R.A. EDWARDS, IBM CORPORATION ON LABORATORY AUTOMATION BASED SYSTEMS

Presentation

at

COMMON Meeting

Cincinnati, Ohio

LABORATORY AUTOMATION BASED SYSTEMS

September 8, 1967

R. A. Edwards
IBM Corporation
112 East Post Road
White Plains, New York

384

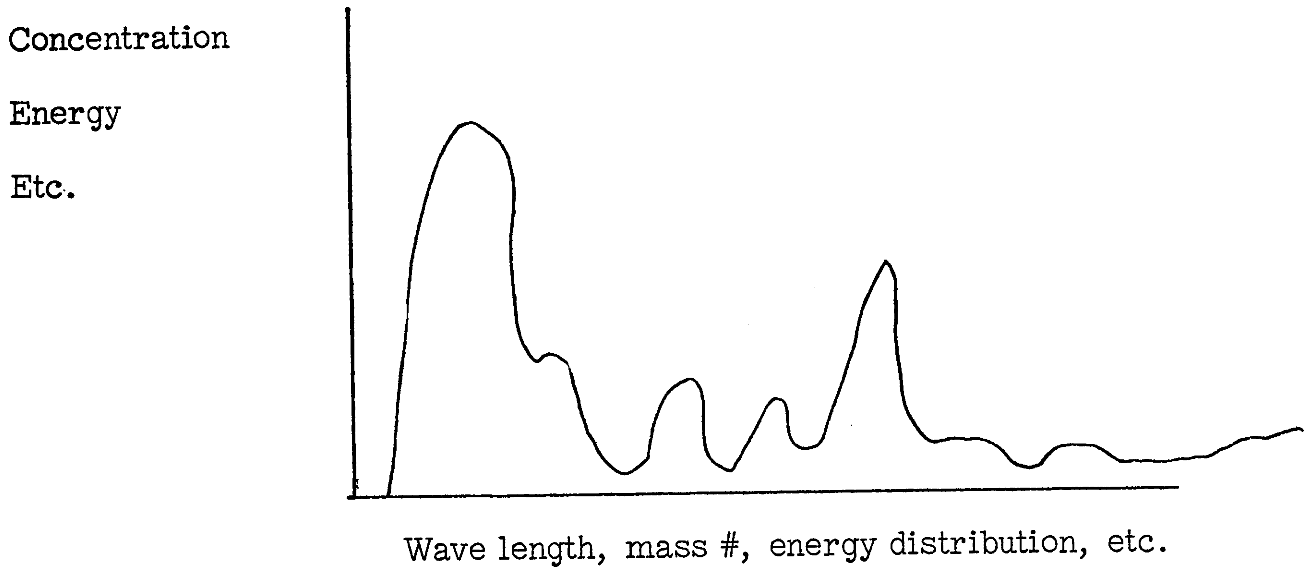
LABORATORY AUTOMATION BASED SYSTEMS (LABS)

Definition

LAB Systems are defined as the application of largely sensor based 1800s and 1130s in the laboratory to perform functions such as:

A.	Data Acquisition	Recording of data at a rate dictated by characteristics of the instrument and the experiment.
		Data Handling - Data Massaging prior to analysis.
		Filtering
		Averaging
		Scaling
		Validity Checking
B.	Data Analysis	Selection of appropriate analysis routine based on:
		Accuracy Requirements Speed of Response
		Analysis for purposes of interpretation of instrument output.
C.	Data Management	Identification-Comparison of interpreted data to known standards.
		Display of experimental data.
		Cataloguing of experimental data.
D.	Experimental Management	Reset of experimental facility to next condition
		Real time interaction between the scientist and his experiment

A laboratory is also defined as a facility in which various levels of instrument (sensors) usage can take place. In addition, at least 80% of the instruments found in a laboratory fall into the spectrometer category; i.e., they produce a graphic output as shown below which is characteristic of a material/compound being analyzed.



Common Instruments

These instruments in a research laboratory can be involved in activities such as:

Pilot Plant Studies
Quality or Production Control
Testing (Components or Systems)
Materials Analysis (Analytical Chemistry)
Basic Research

It is therefore possible to find anywhere from one or multiples of one to twelve different types of instruments in any given laboratory. For instance, a petrochemical laboratory may house multiples of a dozen of the following types of instruments. The mix is dependent upon their concentration on a particular classification of chemistry.

Types of Instruments

Analytical

Amino Acid Analyzers
Mass Spectrometer
Nuclear Magnetic Resonance Spectrometer
Emission Spectrometer
X-Ray Diffractometer
Multichannel Analyzer
Infrared/Ultraviolet/Visable Micronove Spectrometers
Electron Spin Resonance Spectrometer
Spectro Photometer
Ph Analyzer
CHN Analyzer
O_2 Analyzer
Gas Chromatograph

Mechanical

Tensile Strength Analyzer
Dynamometer
Film Thickness & Hardness
Vibration Table

Medical/Clinical

Electrocardiograph
Electroencephlograph
Electromyograph
Electrophoresismeter
Auto Analyzers
Coulter Cell Counters
Cell Scanner

Nuclear

Pulse Height Analyzers
Spark Chambers
Film Digitizers
Neutron Time of Flight Analyzers
X-Ray Flourescense
Electron Spectrometer
Beta, Gamma, Alpha Counting Systems

Other

Viscosimeters
Refractometers
Hardness
Flame Photometers

Incentives

The incentives to automate the laboratory are both tangible and intangible...both benefits are clear advantages over any alternate automation approach.

General

a. All the simple and obvious research has been completed. Today scientists require elaborate experimental facilities which may take years to plan and/or build. The ideas behind the use of these new experimental devices must be tested within a one or two year period after completion of the facility. The experimentalist feels compelled to publish his findings within that period or he may be pre-empted by someone else taking a possibly more advanced approach.

b. Because of the precise timing requirements of the experiments conduct/control or the quantity of data generated, the computer approach is the only reasonable way the experiment can be accomplished...some experiments generate thousands of events which have to be individually analyzed to substantiate statistically the existence of a new material.

c. The computer can tell the experimentalist when things may be going wrong...is the experiment data valid? Also, the cost of running experimental equipment may be significant...an on-line computer may prevent malfunctions in its operation, thus increasing its safety level as well as availability to the researcher. In addition, the speed of the computer will allow optimum usage of the experimental subject...as in the case of a fast changing radioactive isotope which is decaying into other materials.

d. The digital computer's availability and versatility now allows it to simulate experiments common to the physical sciences. Such work often uses higher order languages which are written in terms understood by analog computer users or people used to solving complex mathematical problems.

e. The computer system allows the experimentalist to obtain improved accuracy either directly by means of the A/D converter and the "absolute" nature of the digital computers problem solving capabilities or indirectly through the statistical treatment of meager or vast amounts of experimental data to obtain correlations. It has been stated that instruments have had their accuracies improved by one to two decades through the use of the digital computer.

Specific

a. Assurance of greater precision for data acquisition, data analysis and experiment control.

> Data handled by a LAB System in digital form (assuming a properly designed front end) will provide significantly greater accuracy than in conventional techniques.

Conventional Techniques

Method of Recording	Instrument Error	Readability
Strip Chart (inked)	1/4%	1 part in 150
Oscillograph (heat sensitive)	2%	1 part in 50
AM Magnetic Tape	3-5%	
FM Magnetic Tape	1.75-3%	
PDM/PLM	1-2%	

Note: When the total data acquisition routines associated with conventional techniques are considered, the accumulative errors have been as high as 14%. Significant additional errors are usually introduced in the system through manual analysis of the data and control of the experiment.

LAB Systems Techniques (with a suitably designed front end) operate with an instrument error in the order of .01-.1% and has a readability of 1 part in 16,000.

b. Consistent (24 hr./day) experimental observation and control. The sampling of sensor based data can take place under programmed computer control, on an interrupt basis at varying time intervals: 1 sec., .1 sec., .01 sec., .001 sec., etc., or on a demand response basis. The important point is that the computer can faithfully take data on a fixed or variable basis presented by the experimental conditions during the life of the experiment.

c. Decreased incidence of lost or worthless data due to instrumentation or system malfunction.

A typical input from an instrument may contain valueless data. In fact, a very large fraction of the data generally falls outside of the areas of interest to the experimentalist.

The comparitor circuits or editing routines of a LAB System can insure that only data within limits is digitized and stored. In comparison when blind recording techniques (FM or analog magnetic tape) are utilized, it is common to witness periods of lost data in large blocks. Digital systems can be programmed to self-checking.

d. Release of scientists from routine tasks such as data taking, scaling, calculating, experiment control.

A typical single crystal analysis may require 2,000 different adjustments to the goniometer/scintillation counter head; in addition in excess of 100,000 pieces or data must be extracted from an analog record on a strip chart recorder. The magnitude of this problem is obvious, considering that the maximum manual digitizing rate is 8 points/minute excluding the goniometer control problem. A LAB System can do these tasks automatically.

e. Increased control in the conduct of the experiment.

Research at times operates on the fringes of known operating limits. It must push the state of the art...enter the unknown. For examples, there are today complex analytical instruments associated with small reaction chambers that operate at high termperatures and pressures. The computer monitoring of strain gauge placed on the exterior of the chamber will allow the experiment to proceed under control at the computed limits of the rated design.

f. Conservation of expensive materials such as reactants, isotopes, etc.

Often times experiments require exact amounts of reactants or isotopes which may be decaying into other isotopes at a rate fast enough to produce erroneous results. Such experiments are enhanced through the use of an on-line computer which can insert or inject samples on a closed loop basis in reproductably exact quantities. In addition, if a known or changeable amount of material must be introduced which involves time dependency, the on-line computer can conduct the experiment "flawlessly" within the given time frame and thereby limit the effect of an isotope's degradation upon the experimental results. An ancillary benefit in such environments is that the computer can maintain an inventory of isotopic materials under use and at hand. This is required by AEC regulations.

g. The fringe benefit of obtaining a greater understanding of the complex research devices/instruments.

Instruments such as a mass spectrometer contain power supplies, ionization gauges, sweep voltage controls, etc. The monitoring of the critical points in a mass spectrometer's components will allow the researcher to quickly determine if the equipment is "in phase", i.e., for a given situation (setting) are all other subsystem outputs realistic. The data correlation power of the computer will allow quicker debug time of the instrument. In addition, a history of machine operation/malfunction can be retained in disk memory to pinpoint wear in the components of the instrument.

h. All experimental data is in a retrievable form.

Data in a digital form, on any recording medium (except printed on paper), i.e., magnetic tape, disk files, data cells, can be quickly retrieved for analysis purposes at a later date. Also, the data can be added to during the experiment. Such capabilities are required to identify materials from a large data base such as ASTM tables. These tables are used today in an off line manner to identify compounds.

i. A "Quick Look" at experimental results:

The results of an experiment's progress may apparent from raw data. The on-line computer will allow the research to analyze an envelope of data (a sampling of data over a given time frame) and print such data on a typewriter or display it on a CRT. This ability proves invaluable especially in complex experiments where computed theoretical results can be compared with the analyzed "quick look" results. Such information may cause a complete redirection or reappraisal of the experiment.

j. "Self-Checking" Capabilities

The on-line digital computer is a self-checking device...the only absolute device in the experimental system. It has the capability, if properly programmed, to impose dummy voltages, bit patterns, etc., in control circuits, and by putting the normal control element in a hold or bypassed condition, check the correct functioning of all components. The same program can even "exercise" the control elements during an experiment Similar capabilities allow the on-line system to standardize instruments automatically on a timed or as required basis to eliminate base line drift problems.

k. Operator Training

Routines can be written which will allow the new analyzer operator to simulate the conduct of an experiment. As systems become more complex, such capabilities will prove more useful. Training can utilize the computer's spare time.

l. Exact Time Correlation
The correlation of time with experimental procedures is an often overlooked problem. When recognized it becomes an expensive problem to solve. Even if strip chart recorders have time printed on them it is impossible to accurately relate it to manual experimental procedures. On the other hand, the digital clock in the on-line computer can allow all phases of experimental conduct to be time correlated to the microsecond level if necessary. Time can be in relative or absolute units.

m. Batch and Time Shared Usage of the Central Processing Unit
The normal laboratory use of a computer today is in a batch mode, i.e., the experimentalist takes his deck of cards to the computer center for solution. Now programming systems exist which allow a single computer to provide batch capability while at the same time conducting on-line tasks such as data acquisition and closed loop control of experiments.

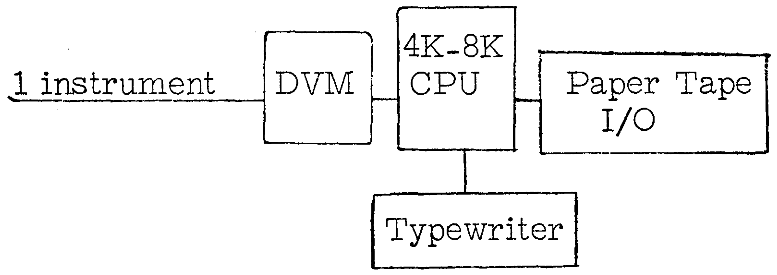
n. Efficient Facility Usage
The ability of the on-line computer to compute results of an experiment within minutes after its completion can greatly increase the utilization of complex analytical equipment.

o. New and Timely Experimentation
This incentive cannot be overlooked. The technical world changes very rapidly and what is research today will be routine in the future. Instrument design and data rates are changing. Experimental procedures are being enhanced by graphics, information retrieval and on-line interaction of the researcher with his experiment. Only the computer can provide this flexibility on a controlled and timely basis.

Systems Approaches

The laboratories which are automating today are taking four systems approaches.

1. 1 (one) instrument - very small CPU - 1 (one) researcher

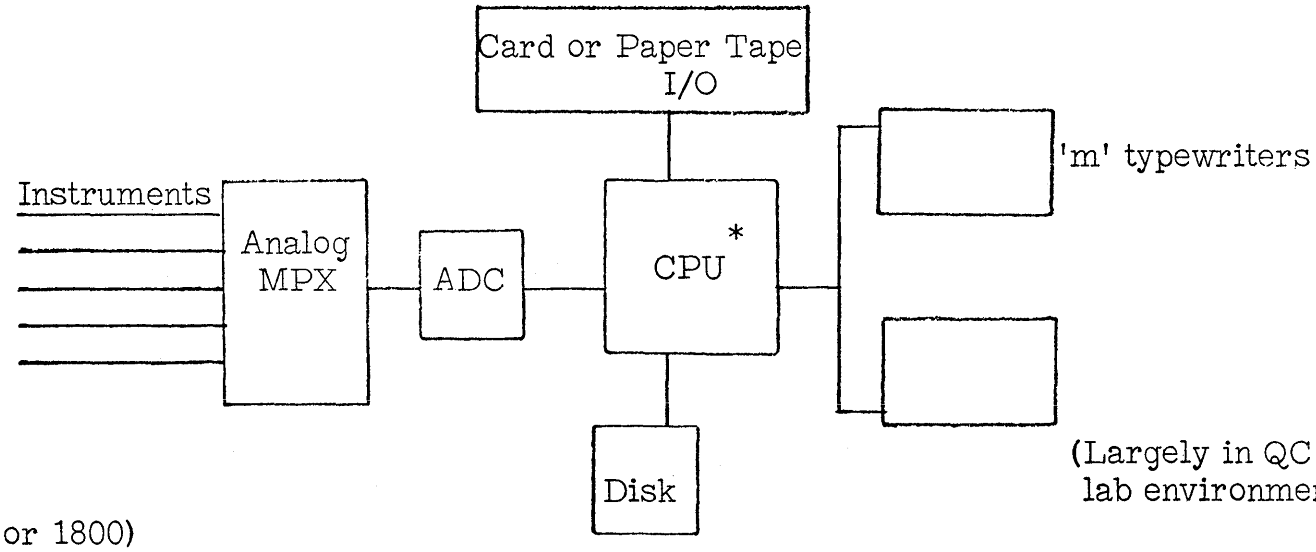


(Largely in basic research environment)

Considerations

Low initial investment	Programming in machine language
Dedicated system	No growth system
	Unsupported

2. Multiple instruments of same type - small CPU - 1 supervisor



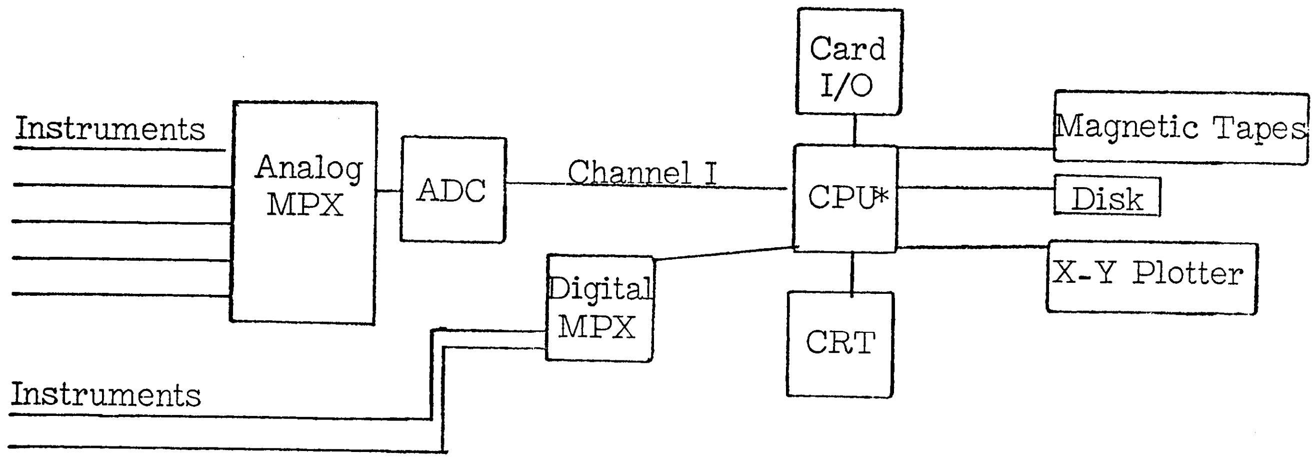
(Largely in QC lab environment)

*(1130 or 1800)

Considerations

Easily customized front end	low CPU usage factor
fixed programs, low demand on CPU	difficult to add different instrument
easily justified	fixed time slices in data acquisition mode
easy to support under a TSX	

3. Multiple instruments of different types - small CPU - many different users



*(1130 or 1800)

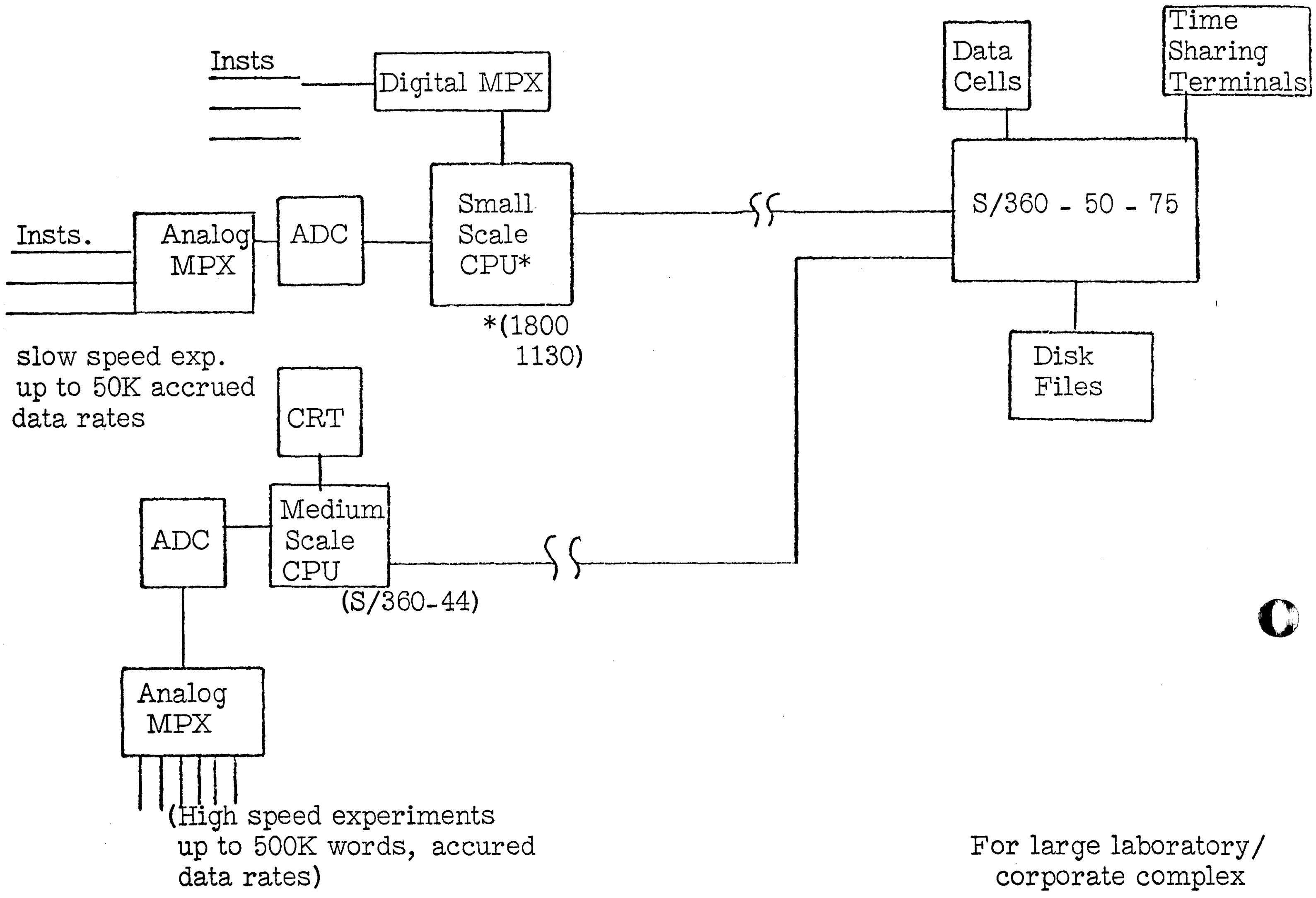
Configuration for multi-purpose laboratory facility

Considerations

Capable of satisfying many different users on a demand response basis	High initial cost
Possible to dynamically adapt to experiments	More sophisticated programming required
Remote link to 360	
Capable of more than data acquisition tasks	

394

4. Total integrated laboratory information system



CONSIDERATIONS

Total approach	Programming requirements are complex
Capable of widest range of laboratory applications	Price is high unless remote S/360 is time shared
Fast turn around time	
Offers large CPU for total problem solution	

Modes of System Operation

Each different type of laboratory could conceivably write its own unique monitor system to fit the various levels of instrument usage which is conducted there. The five levels of activity found in typical laboratories is shown in Table I.

Relevance of 1800 & 1130

The ability of the 1130 and 1800 to satisfy the demands of the LAB System applications can be summarized as follows:

1130

Hardware:
The OEM and SAC channels on the 1130 allow this machine to accept an analog or digital I/O (instrument based) front end. At least two manufacturers have announced A/D converters which can tie to the 1130.

Software:
No IBM TSX monitor exists for the 1130. However, it is known that at least three users are writing I/O subroutines to support an ADC front end. It is also known that two 1130's are tied to x-ray and neutron analyzers in a closed loop fashion under some monitor system.

1800

Hardware:

The I/O flexibility of the 1800 makes it an ideal growth system for laboratory automation programs. The newly announced digital multiplexor RPQ allows data to stream directly into assigned core memory locations on a demand reponse basis at data rates of 50K words/second. Thus each experimentalist will think he has a computer always available to him during his asynchronous method of operation.

Software:

Up to 50 chromatographs at one user's laboratory are now operating under the 1800's TSX. The GCs are connected to the 1800 via the 1894 System Engineering LAB special automatic gain change ADC front end.

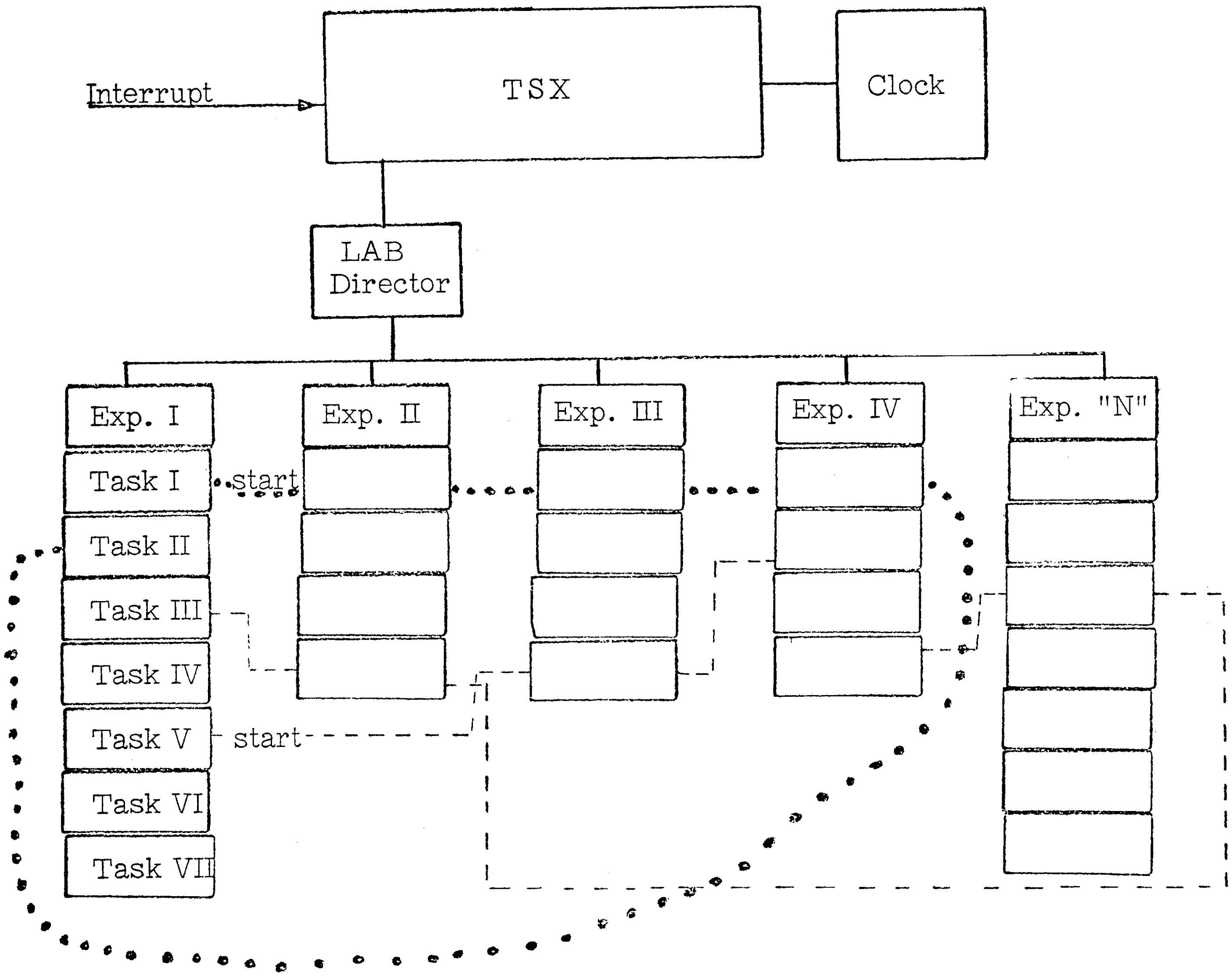
Laboratory Automation in IBM

IBM has now in progress at its own technical centers a number of laboratory automation programs in support of its own research requirements.

An 1800 at the Thomas J. Watson Research Center is tied to an x-ray diffractometer in a closed loop fashion. In addition, a plate film reader is also connected. At IBM's San Jose Research facility an Electron Paramagnetic Resonance Spectrometer is tied to an 1800 also in a closed loop mode. In both cases, the 1800's are controlling the experiment. Both facilities plan the attachment of additional instruments in the future.

These IBM research laboratory automation efforts have generated two differing operating systems approaches to facilitate the interaction between multiple different instruments and the 1800. The main features of these operating systems is a so called "Lab Director Program" which satisfies the demands of slow response analytical instruments in the basic research environment.

These LAB directors are written to allow the computer to respond to the various tasks associated with each type of instrument in an efficient manner; i.e., not under complete TSX control. So long as no interrupts arise, the LAB director allows the 1800 to handle multiple different instruments with relative ease. Modes of operation under the LAB directors are as shown in the following diagram.



● ● ● ● ● ● ● serial tasking

– – – – – random tasking

in equal or variable time slices

Multi-Tasking
Lab Environment

Summary

In summary, one finds the researcher or other lab personnel in the process of modifying the existing TSX for the 1800 to adapt it to his particular laboratory needs. Type III application programs for the gas chromatograph, mass spectrometer, and Instron tensile strength tester have been written by IBM and run under TSX. A plea is made to COMMON members to put in PID their instrument oriented application programs.

Table I	PILOT PLANT OPERATION	QUALITY CONTROL	COMPONENT TESTING	MATERIALS ANALYSIS	BASIC RESEARCH
Basic operational philosophy of on-line computer	frequent changes in a large math model program - low batch demands	infrequent changes to program, computer dedicated... no background	frequent changes in small programs	infrequent changes in analysis routines, frequent roll in & out of analytical subroutines	frequent changes in programs associated with wide variety of experiments
Instrument Characteristics	Slow response, wide variety of "process" instruments plus few analyzers	slow response, multiple instruments of one or two types	wide response range, mainly slow. Generally one instrument type to satisfy	slow response insts., are general wide variety of analytical insts.	wide range of inst. response + wide variety of instrument
General philosophy of system operation	to get the most out of the pilot plants performance	to get the most out of the insts. with as few technicians as possible	to recognize faults in products final assembles & components as soon as possible	to get the most out of instruments	to get the most out out of researcher
Turn Around Requirements	Seconds	Minutes	Seconds to Hours	Minutes	Milliseconds to Seconds
Computer Tasks	data acquisition optimization	data acquis. limit checking report generation	data acquisition performance summary report generation plotting	data acquisition data cleanup table look up report generation	data acquisition data analysis data storage & retrieval experiment control
Monitor Characteristics	none - process demands will use computer	scan instrument at fixed time interval - no interrupts	scan instruments at fixed time intervals - rare interrupts	scan instruments at fixed time interval when CPU is interrupted	demand response mode necessary... no time for monitor overhead
Instrument Usage Factor	100%	85-90%	100%	50-60%	10-15%

GENERAL OPERATIONAL CHARACTERISTICS OF LABORATORY OPERATION

SESSION NUMBER F.1.6

SPEAKERS
    LAURA AUSTIN

DISCUSSION
        A CALL WAS ISSUED FOR PROGRAM CHAIRMAN AND LOCAL ARRONGEMENTS CHAIRMAN. BOTH WERE LOCATED FOR THE PHILADELPHIA MEETING FOR SEPTEMBER, 1968. WE STILL NEED BOTH FOR CHICAGO IN APRIL, 1968. FUTURE SITES WERE LISTED AND DISCUSSION WAS HELD REGARDING CONTENT AND FORMAT OF MEETINGS.

SESSION NUMBER F.1.7

SPEAKERS

DANIEL J. LAMPONE
EDMUND E. COLAN ON A 1620 COMPUTER PROGRAM FOR A.C. CIRCUIT
ANALYSIS UTILIZING TRANSISTOR Y PARAMETER EQUIVALENT
CIRCUIT MODELING TECHNIQUES

August 28, 1967

A 1620 COMPUTER PROGRAM FOR A.C. CIRCUIT ANALYSIS UTILIZING TRANSISTOR "Y" PARAMETER EQUIVALENT CIRCUIT MODELING TECHNIQUES

by

Daniel J. Lampone
and
Edmund E. Colan

Central Division
Sylvania Electronic Systems,
Sylvania Electric Products Inc.
Williamsville, N.Y. 14221

403

TITLE: A 1620 Computer Program for A. C. Circuit Analysis Utilizing Transistor "Y" Parameter Equivalent Circuit Modeling Techniques

Authors: Daniel J. Lampone, Sr. Engineer
Edmund E. Colan, Research Engineer
Sylvania Electronic Systems-Central
Wehrle Drive and Cayuga Road
Williamsville, New York 14221

I. INTRODUCTION

This paper presents a simple but accurate method of obtaining circuit A. C. analysis by use of the IBM 1620 computer. The unique feature in this program utilizes the transistor in its "Y" parameter equivalent circuit and some circuit transformation techniques to allow insertion of program input information in a normal topological manner. The need for the use of "Y" parameter representation became apparent when problems were encountered using a predecessor program employing mesh current analysis and the transistor "h" parameter model for high frequency problems. High frequency representation of the "h" parameter model utilizing discrete components for the complex quantities created problems in circuits involving tuned amplifiers. Details of this will be presented later. Another reason for resorting to the "Y" nodal analysis was that most vendor's information specify "Y" parameters for high frequency transistors. In the mesh current method, inaccuracies resulted when converting "Y" parameters to "h" parameters since conversion formulae were approximate. Before getting into an actual circuit solution, let us quickly illustrate the basic "Y" model and sample nodal setup. Figure 1 shows the basic four-terminal "Y" parameter equivalent circuit. The use of the "Y" parameter equivalent circuit is essential in nodal circuit analysis since units of current must be maintained, with the solution of nodal voltages being the outcome by matrix algebra. This is illustrated by reference to the example passive circuit and nodal equations shown in Figure 2.

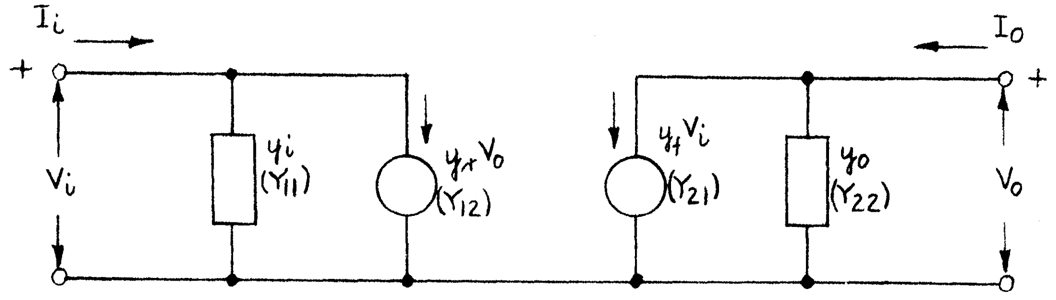
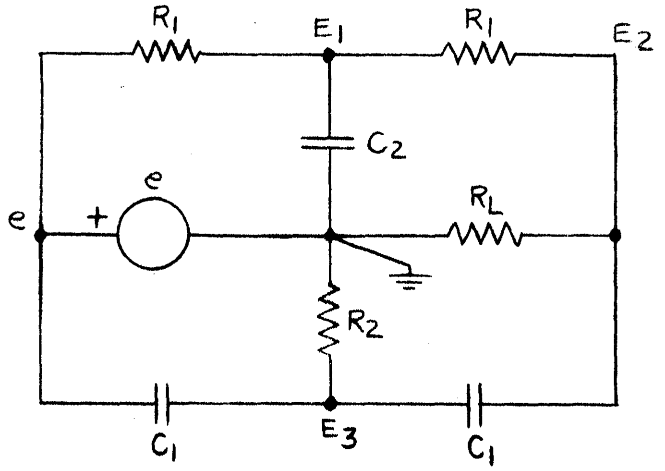


FIGURE 1



EQUATIONS:

$$E_1(2G_1 + j\omega C_2) + E_2(-G_1) + E_3(0) = eG_1$$

$$E_1(-G_1) + E_2(G_1 + G_L + j\omega C_1) + E_3(-j\omega C_1) = 0$$

$$E_1(0) + E_2(-j\omega C_1) + E_3(2j\omega C_1 + G_2) = e j\omega C_1$$

FIGURE 2

The matrix solution is $[E]\ [Y] = [I]$ where the coefficient matrix is made up of admittances rather than resistances in a mesh current solution. The right side of the equation is current sources associated with respective nodes.

II. A.C. EQUIVALENT CIRCUIT SETUP

This discussion will be centered around a sample RF amplifier circuit shown in Figure 3. The circuit has been simplified for purposes of the analysis. Decoupling networks, d-c biasing components, and AGC circuitry were removed since they have no effect in the a-c analysis. The circuit consists of two cascoded pairs of transistors and four tuned circuits (varactor tuned). Figure 4 shows the a-c equivalent circuit with all circuit components and parameters modified by "turns-ratio squared" factors to allow for elimination of circuit ideal transformers.

It would be best to interject at this time how to arrive at the a-c equivalent circuit since the success of the solution hinges greatly on the correct methods used at this point in the analysis. It was necessary to do some hand calculations because of the limited capacity of the 1620 computer or else we would be into the category of a highly complicated a-c analysis used in the more publicized and more automated programs allowed by the larger computers. Certain steps must be followed in setting up the a-c equivalent network in order to show how to arrive at Figure 4. First, starting with the original circuit, substitute the equivalent circuit of the transformers. Most methods for these were taken from "Communication Networks", Volume 1, by Guillemin. The obvious intentions were to arrive at an equivalent "T-network" and the following example and Figure 4A shows the steps taken.

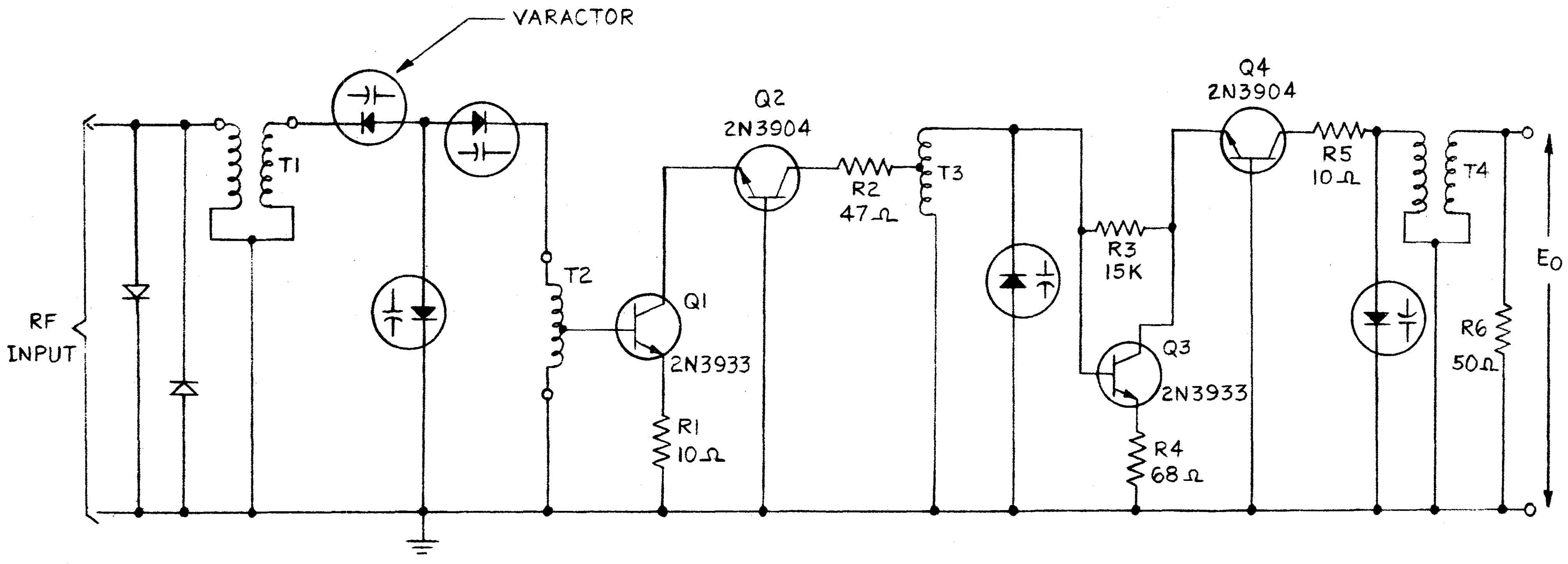
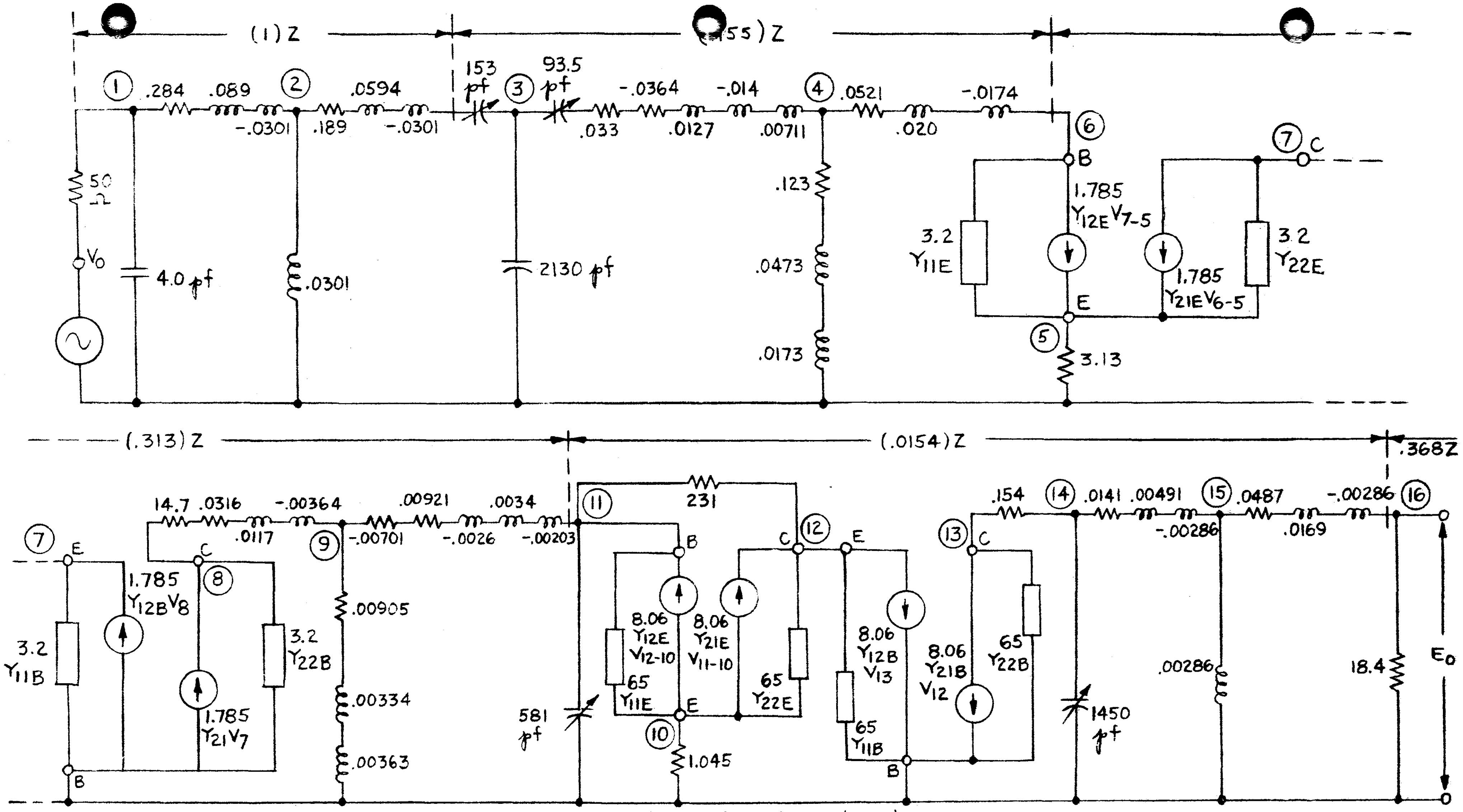


FIGURE 3

R.F. AMPLIFIER

407

408



$$\frac{1}{a_1}\cdot\frac{1}{a_2} = \frac{1}{.394}\cdot\frac{1}{1.42} = 1.785$$

$$\frac{1}{a_1^2}\cdot\frac{1}{a_2^2} = \frac{1}{.313} = 3.2$$

$$\frac{1}{a_1}\cdot\frac{1}{a_2}\cdot\frac{1}{a_3} = \frac{1}{.394}\cdot\frac{1}{1.42}\cdot\frac{1}{.222} = 8.06$$

$$\frac{1}{a_1^2 a_2^2 a_3^2} = \frac{1}{.0154} = 65$$

$$E_0 = \frac{V_{16}}{.606}$$

2N3933 AT 56 MHz

$Y_{11E} = (1.1 + j\,4.2) \times 10^{-3}$ MHOS

$Y_{12E} = (0 + j\,.16)$

$Y_{21E} = (51 - j\,13)$

$Y_{22E} = (.015 + j\,.4)$

2N3904 AT 56 MHz

$Y_{11B} = (64.5 - j\,43) \times 10^{-3}$ MHOS

$Y_{12B} = (-.27 - j\,.51)$

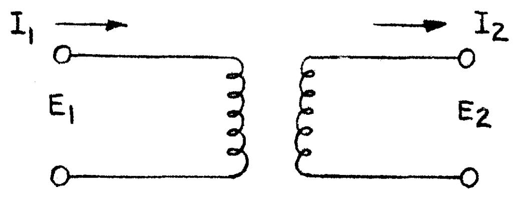
$Y_{21B} = (-60 + j\,49)$

$Y_{22B} = (.27 + j\,.84)$

-5-

FIGURE 4

A.C. EQUIVALENT CIRCUIT



$$b_{11} = R_1 + jwL_1$$

$$b_{22} = R_2 + jwL_2$$

$$b_{12} = jwM = b_{21}$$

Defining the Mutual Inductance as Negative Numerically

(Arbitrary; for convenience only)

we can say:

$$b_{11}I_1 - b_{12}I_2 = E_1$$

$$-b_{21}I_1 + b_{22}I_2 = E_2$$

One type of network which can be made equivalent to the transformer is shown below. It consists of the T-network and an ideal transformer. Various special forms result by giving the ratio "a" of the ideal transformer particular values.

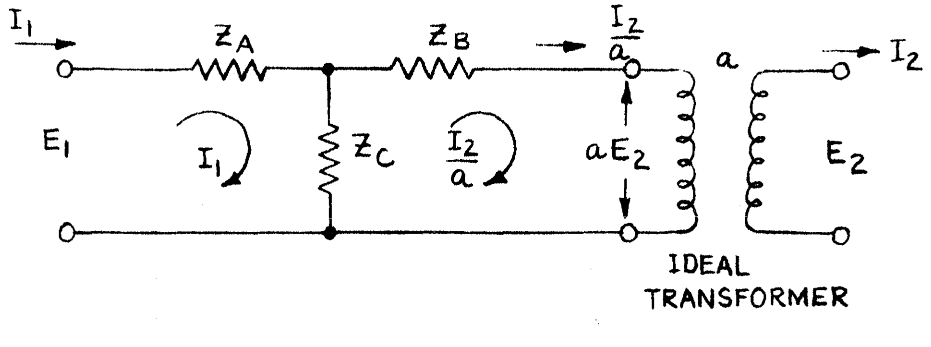


Figure 4A

For the T-structure alone:

$$\left.\begin{aligned} (Z_A + Z_C)I_1 - \frac{Z_C I_2}{a} &= E_1 \\ - Z_C I_1 + (Z_B + Z_C)\frac{I_2}{a} &= aE_2 \end{aligned}\right\}$$

or:

$$\left.\begin{aligned} (Z_A + Z_C)I_1 - \frac{Z_C}{a} I_2 &= E_1 \\ - \frac{Z_C}{a} I_1 + \frac{Z_B + Z_C}{a^2} I_2 &= E_2 \end{aligned}\right\}$$

If this network with the ideal transformer is to be identical in behavior to the original transformer in Figure 4A, then,

$$\frac{Z_C}{a} = b_{12}$$

$$Z_A + Z_C = b_{11}$$

$$\frac{Z_B + Z_C}{a^2} = b_{22}$$

which gives

$$Z_A = b_{11} - ab_{12} \qquad = R_1 + jwL_1 - jwaM$$

$$Z_B = a^2b_{22} - ab_{12} \qquad = a^2(R_2 + jwL_2) - jwaM$$

$$Z_C = ab_{12} \qquad = jwaM$$

Therefore we have the circuit in Figure 4B:

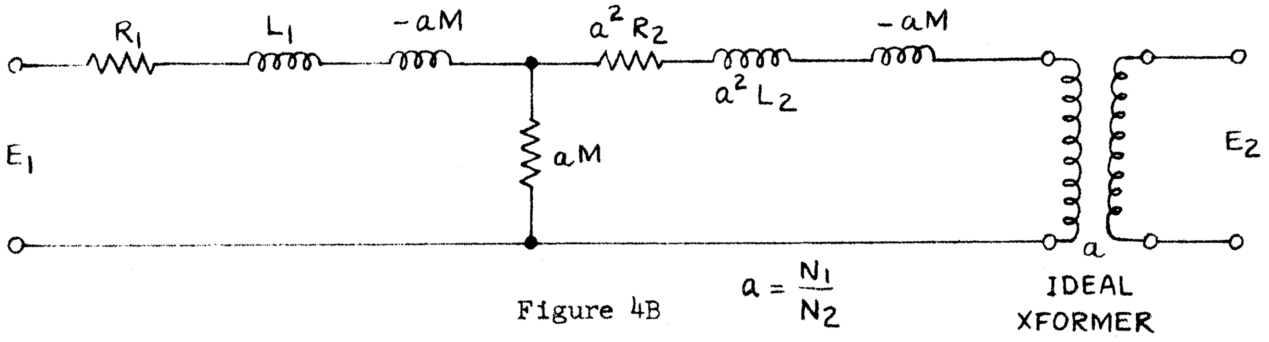


Figure 4B

One must keep in mind that the transformer conversions need only be derived once and retained for reference in anticipation of future problems involving transformers. Initial preparation of these equivalent networks of transformers is painstaking but nevertheless valuable.

By assigning any value to the ratio a, we immediately have the impedances in the T-structure to be used in conjunction with an ideal transformer having this ratio in order to replace the given transformer.

A special case: when we make a = 1 the ideal transformer vanishes (this is the case for a 1:1 transformer).

Then we have:

$$Z_A = b_{11} - b_{12}$$
$$Z_B = b_{22} - b_{12}$$
$$Z_C = b_{12}$$

and this is:

$$Z_A = R_1 + jw(L_1 - M)$$
$$Z_B = R_2 + jw(L_2 - M)$$
$$Z_C = jwM$$

This result is illustrated in Figure 4C as follows:

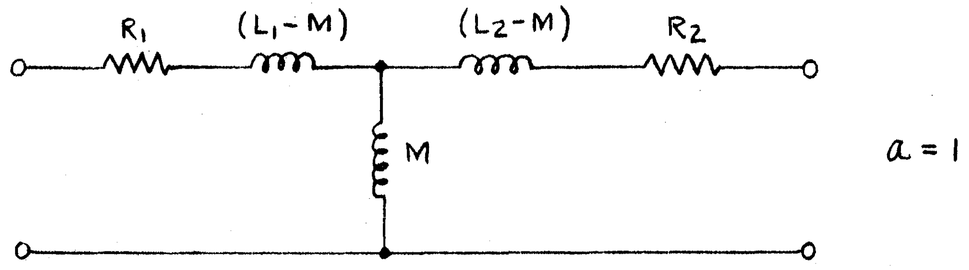
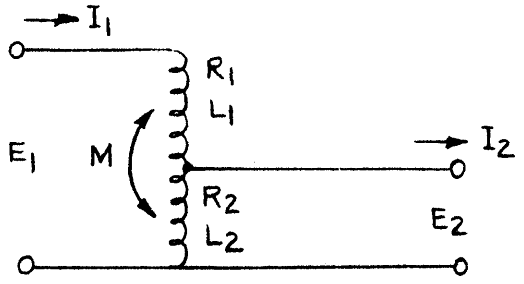


Figure 4C

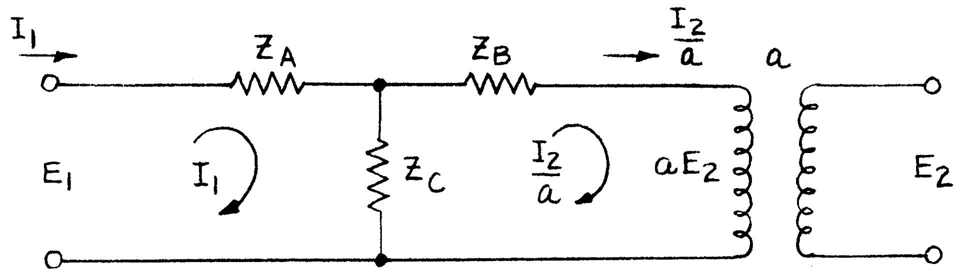
Our original circuit also showed step-down and step-up auto-transformers. For purposes of brevity, the solution for the equivalent circuit of the auto-transformer follows without detailed discussion in order to approach the next point.



$$b_{11} = R_1 + R_2 + jw(L_1 + L_2 + 2M)$$

$$b_{22} = R_2 + jwL_2$$

$$b_{12} = b_{21} = R_2 + jw(L_2 + M)$$



$$(Z_A + Z_C)I_1 - Z_C \frac{I_2}{a} = E_1$$

$$-\frac{Z_C}{a} I_1 + \frac{Z_B + Z_C}{a^2} I_2 = E_2$$

per above:

$$Z_A + Z_C = b_{11} = R_1 + R_2 + jw(L_1 + L_2 + 2M)$$

$$\frac{Z_B + Z_C}{a^2} = b_{22} = R_2 + jwL_2$$

$$\frac{Z_C}{a} = b_{12} = b_{21} = R_2 + jw(L_2 + M)$$

Substituting:

$$Z_C = a\left[R_2 + jw(L_2 + M)\right]$$

$$Z_A = R_1 + R_2 + jw(L_1 + L_2 + 2M) - a\left[R_2 + jw(L_2 + M)\right]$$

$$Z_B = a^2(R_2 + jwL_2) - a\left[R_2 + jw(L_2 + M)\right]$$

Impedances are:

$$Z_C = aR_2 + aL_2 + aM$$

$$Z_A = R_1 + (1-a)R_2 + L_1 + (1-a)L_2 + (2-a)M$$

$$Z_B = (a^2-a)R_2 + (a^2-a)L_2 - aM$$

which yields the following equivalent circuit shown in Figure 4D:

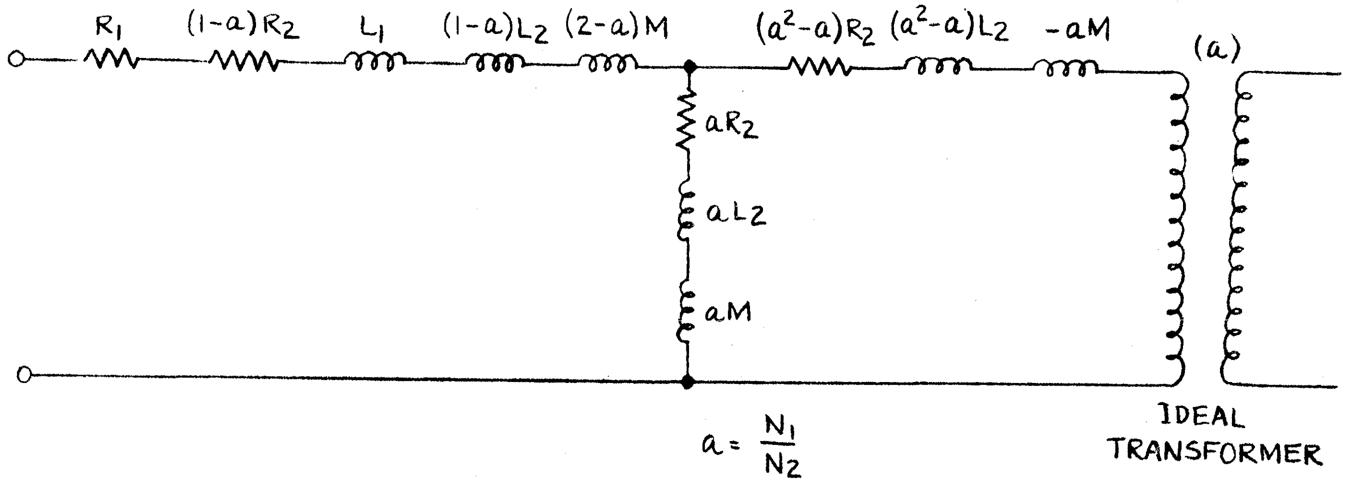


Figure 4D

Step-Down Autotransformer

As you can see, each equivalent circuit of the transformers includes an ideal transformer. The next step is to eliminate the ideal transformers in order to avoid discontinuities in the circuit solution. In order to eliminate ideal transformers all components succeeding an ideal transformer must have impedances modified by the "turns-ratio-squared" factor $\left[\frac{N_1}{N_2}\right]^2$.

Now, looking back at Figure 4, the circuit was split up at four different points where the ideal transformers should have existed. For this circuit the following turns-ratios existed for the four transformers:

$$a_1 = .394 \quad a_2 = 1.42 \quad a_3 = .222 \quad a_4 = 4.88$$

As shown on Figure 4, the first section impedences succeeding the first ideal transformer get multiplied by $.394^2 = .155$, the following section by $\left[(1.42)^2(.155)\right]$ and so on. Notice, the multiplication factors become cumulative for section after section. This is an absolute necessity since the source generator must see all components in the proper perspective.

Conversely, it becomes necessary to modify the "Y" parameters of the transistors (admittances) by multiplying Y_{11} and Y_{22} by $1/a^2$, $Y_{21}v_1$ and $Y_{12}V_2$ by $1/a$. This can be understood easier by observing the following calculations.

For the 2N3933 of the first cascoded pair:

$$\frac{1}{a_1} x \frac{1}{a_2} = \frac{1}{.394} x \frac{1}{1.42} = 1.785$$

$$\frac{1}{a_1^2 a_2^2} = \frac{1}{.313} = 3.2$$

Now:

$$Y_{11E} = (1.1x10^{-3} + j\ 4.2x10^{-3})3.2 = 3.52x10^{-3} + j\ 13.42x10^{-3}$$

$$Y_{12E} = (0x10^{-3} - j.16x10^{-3})1.785 = 0 - j.285\ x\ 10^{-3}$$

$$Y_{21E} = (51x10^{-3} - j\ 13x10^{-3})1.785 = 91x10^{-3} - j\ 23.2x10^{-3}$$

$$Y_{22E} = (.015x10^{-3} + j.4x10^{-3})\ 3.2 = .048x10^{-3} + j\ 1.28x10^{-3}.$$

The same multiplication factors apply to the common base 2N3904 of the first cascoded pair.

For the second cascoded pair, the same process is carried out, only now the multiplication factors obviously become:

$$\frac{1}{a_1} \times \frac{1}{a_2} \times \frac{1}{a_3} = \frac{1}{.394} \times \frac{1}{1.42} \times \frac{1}{.222} = 8.06$$

$$\frac{1}{a_1^2\ a_2^2\ a_3^2} = 65$$

We now arrive at what looks like Figure 4, the a-c equivalent circuit. The foregoing set-up seems a little messy (yet straightforward) but after experience with a couple of circuits, adeptness, efficiency and accuracy are very much enhanced.

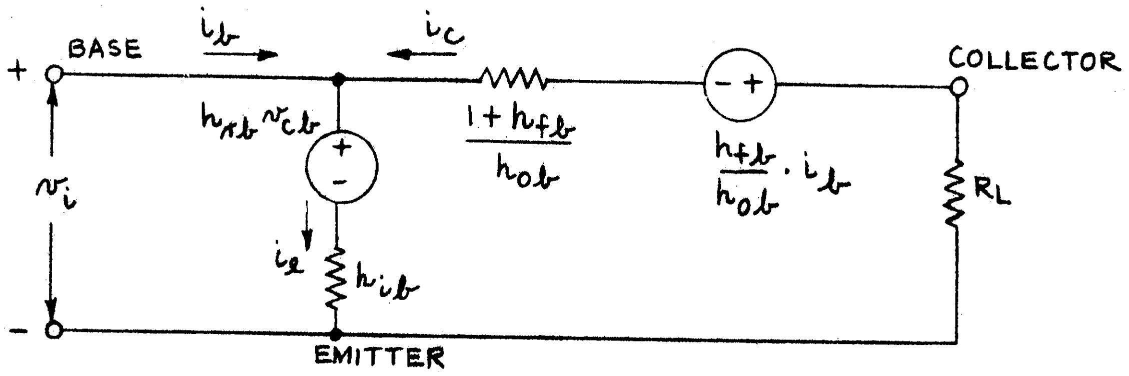
III. PROBLEM AREAS

Minor problems existed and we will now discuss them.

Originally, it was intended to solve this circuit by the loop (or mesh) current method using the "hybrid" equivalent network for the transistor. The "hybrid" equivalent network is shown in Figure 5 with a few rules governing its insertion into the circuit, This "h" equivalent was derived from information contained in the text "Transistor Circuit Analysis and Design" by F.C. Fitchen.

The information in Figure 5 is only included as extraneous information to serve as a guide in cases where choice is made to apply a mesh current solution, for example, low frequency analog circuits.

As I previously mentioned, most vendor's information gives "Y" parameters for high frequency transistors, and these are given in complex quantities. Converting to "h" parameters is necessary to perform mesh current analysis and these obviously become complex. Utilizing discrete components for the complex quantities



Circuit Insertion Rules

1. $h_{rb} \cdot v_{cb}$ is always same as v_i in sign.

2. Polarity of $\frac{h_{fb}}{h_{ob}} \cdot i_b$ must always be related to v_i as follows: either both must be positive at collector and base (terminals) respectively (as shown in sketch), or both negative at these terminals. Base current (i_b) must be consistent with v_i.

3. v_{cb} is interpreted as voltage at collector relative to base. Polarity of the voltage generator ($\frac{h_{fb}}{h_{ob}} \cdot i_b$) is indicated (as determined in 2). Voltage drop and polarity in resistor $\frac{1+h_{fb}}{h_{ob}}$ is determined by direction of current flow assumed for i_c.

4. When inserting parameter data into equations, don't forget that h_{fb} is negative.

5. The generators will take on polarities with respect to loop currents according to the following manner:

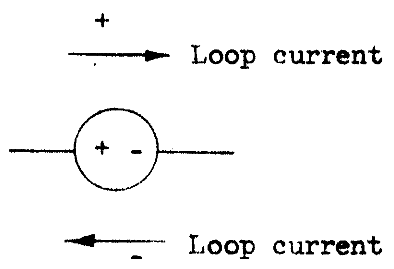


Figure 5

created problems where tuned circuits could interact with the transistor capacities. Reference to the grounded base configuration in Figure 6 will illustrate this.

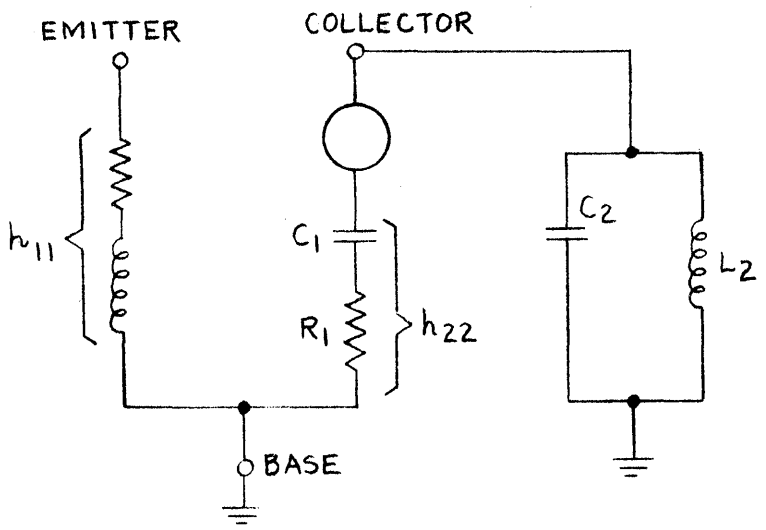


Figure 6

The capacitor C_1 and resistor R1 are the components resulting from the conversion. During computer solution, C_1, which is small, resonated with the combination of C_2 and L_2. C_2 and L_2 formed a high value of X_L rather than a high resistive value for the tuned circuit. Consequently, while attempting to tune a circuit for maximum output voltage, the solution proceeded to find the peak series resonant point rather than the antiresonant point which is desired. The series resonant point causes a much higher output voltage, E_o, than the parallel peak. (This can be demonstrated mathematically). This is a false resonant condition, even though E_o is higher, because of interaction between transistor parameters and actual external tank circuits. The series resonant peak is close to the antiresonant peak and the skirts of the response curve blanket any identification of the actual circuit parallel resonant condition. Conversely, a solution for antiresonance by searching for the minimum loop current does not succeed because C_1 eliminates the occurrence of the current "dip".

Attempts to utilize a current generator in the h-parameter transistor model with paralleled h_{22} components were not successful. Introduction of the current generator - or any equivalent alterations - could not be made compatible with the loop analysis.

Therefore, without the success of using the "h" equivalent loop (or mesh) current analysis, we proceeded to investigate the "Y" equivalent nodal analysis.

Some nodal analysis problems also occurred but were "ironed" out after some proper manipulation. It so happened that with the circuit previously shown containing two (2) pairs of cascoded transistors, it was found that the input admittance of the second transistor in the pair (the grounded base) loaded down the grounded emitter transistor output so that a loss resulted rather than a gain across the pair. This problem was circumvented by combining the transistor data so that two cascaded transistors are combined into one single model to represent the cascoded pair.

The combined admittance matrix takes on the following form:

$$Y' = \begin{vmatrix} Y_{11E} & Y_{12E}\dfrac{y_{12B}}{y_{21B}} \\ Y_{21E} & y_o \end{vmatrix}$$

$$\text{where } y_o = y_{22B} - \frac{y_{21B}y_{12B}}{Y_{22E}+y_{11B}}$$

The new combined "Y" model of a common emitter - common base pair has its elements varied by the above formulae (the Y_{11E} and Y_{21E} remained unchanged obviously). Another small problem that occurred in the process of determining the y-parameters for a cascoded pair is where the grounded emitter transistor has a degenerating emitter resistor. The emitter resistor is combined by matrix manipulation as shown in Figure 7.

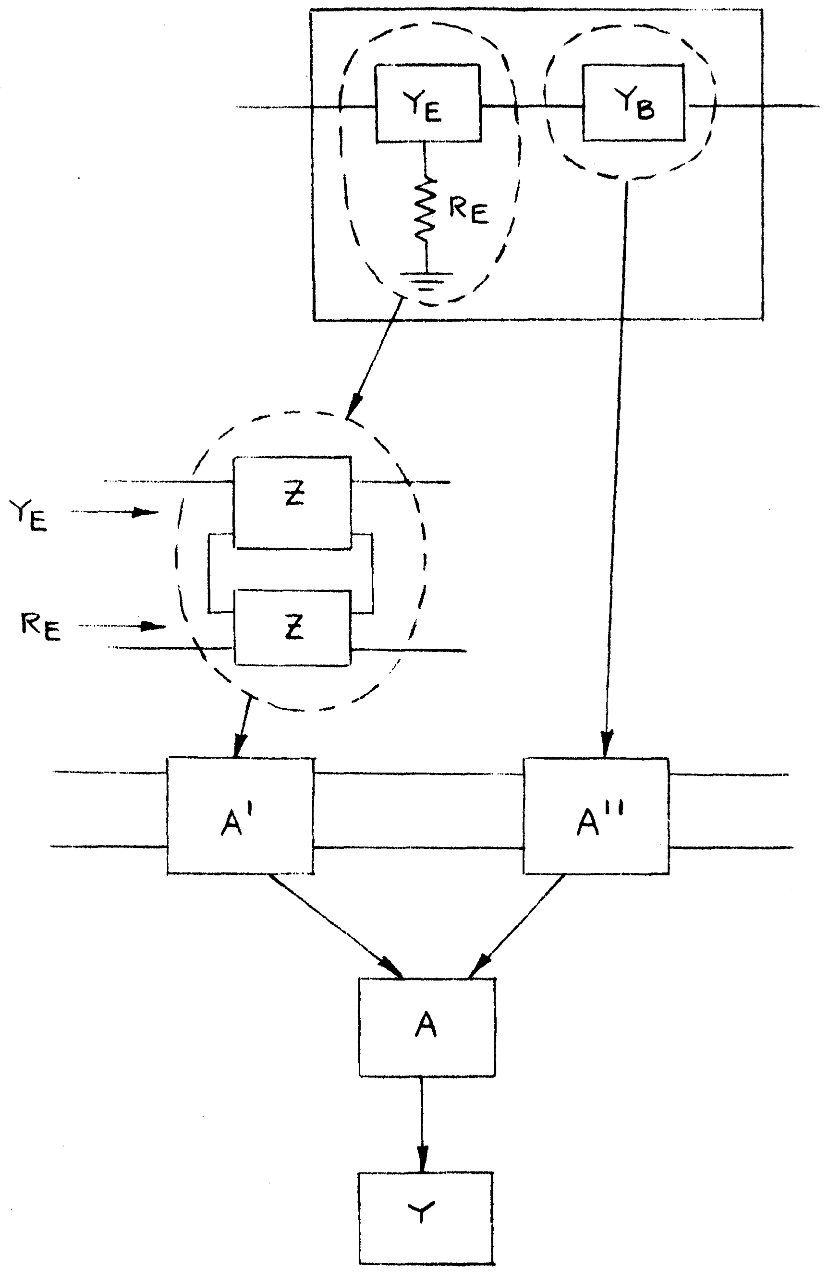


FIGURE 7

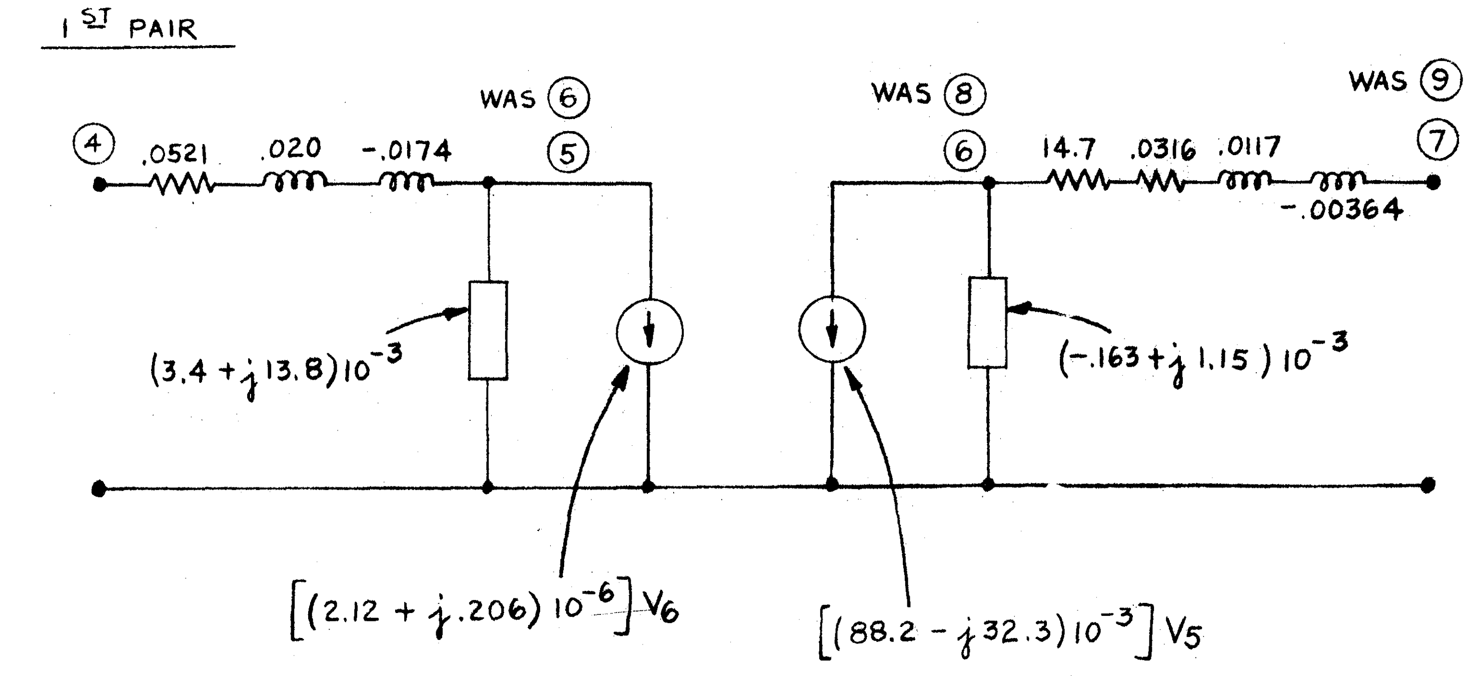
A short computer program was written to accomplish what is shown in Figure 7. The y- parameter data initially used was the transistor data modified by the "turns-ratio squared" factors to fit the A.C. equivalent circuit with ideal transformers removed. This modified data was then combined with the emitter resistor value to obtain the resultant matrix for each cascoded pair. This method, however, yielded improper parameter data. Oddly enough, this particular problem was rectified by reversing the order of solution. By first combining the basic transistor data with the emitter resistor and then modifying by the "turns-ratio squared" factors, the proper data finally resulted for the cascoded pair.

By combining the transistor data for each cascoded pair, the circuit reduces to a 12 node problem. Figure 8 shows the new revised areas containing the resultant combined "Y" parameters of each cascoded pair. The emitter resistors have now been combined into the new "Y" parameters representing each cascoded pair.

IV. COMPUTER PROGRAM

The final category that will be discussed is the actual computer input data set-up. With most people this should be elementary since matrix algebra determines the final solution and a little knowledge of circuit analysis can aid the programmer in inputing the information. The technical portion that I have previously discussed would probably need the assistance of an engineer to properly set-up the a-c equivalent circuit. But again, practice with a few circuits would lead to more familiarity and shorter turn-around time.

An explanation of presenting the input information follows. By referring to Figure 9 and Figure 4 with the combined cascode pair configuration of Figure 8, the input information is easily understandable.


1ST PAIR
WAS 6
WAS 8
WAS 9
4
.0521
.020
-.0174
5
6
14.7
.0316
.0117
7
-.00364
$(3.4 + j13.8)10^{-3}$
$(-.163 + j1.15)10^{-3}$
$[(2.12 + j.206)10^{-6}]V_6$
$[(88.2 - j32.3)10^{-3}]V_5$

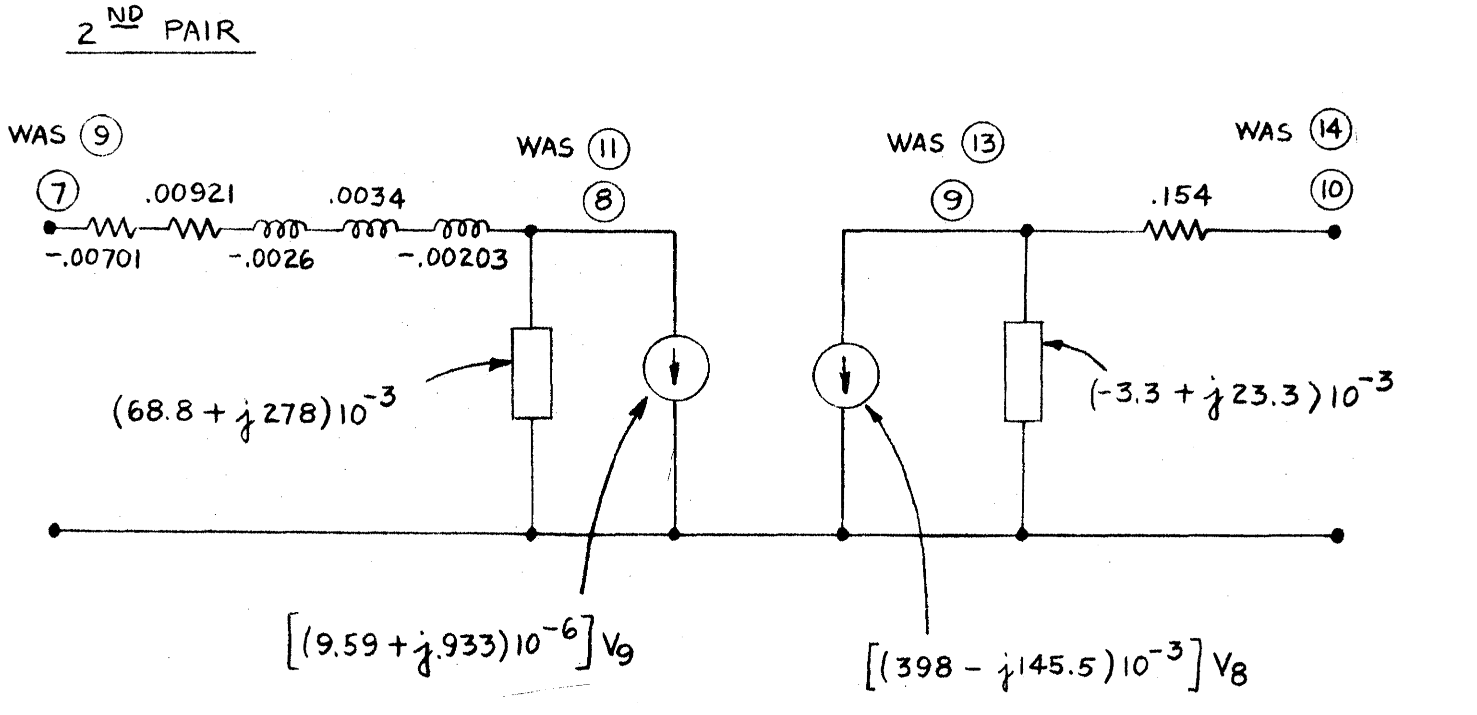

2ND PAIR
WAS 9
WAS 11
WAS 13
WAS 14
7
.00921
.0034
8
9
.154
10
-.00701
-.0026
-.00203
$(68.8 + j278)10^{-3}$
$(-3.3 + j23.3)10^{-3}$
$[(9.59 + j.933)10^{-6}]V_9$
$[(398 - j145.5)10^{-3}]V_8$

FIGURE 8

Read cards order:

1st card: N and Frequency

where N = number of nodes

2nd set of cards: II, JJ, R, XL, C, ITYP (Impedance Branches Only)

where: II = matrix row element (node number)

JJ = matrix column element (node number)

R, XL, C = resistance in ohms, inductance in henries, capacitance in farads associated with II, JJ impedance branch.

ITYP = 0 if symmetrical, 1 if asymmetrical

All impedance branches are symmetrical. The YV (current generators) branches are sometimes asymmetrical and will be assigned as such when we present the admittance branches later. By looking at Figure 9 and Figure 4, three branches are common to node 1 and must be listed separately in the 1-1 position. The program combines these in the final 1-1 position in the coefficient matrix. A branch need not be repeated when symmetrical and off-diagonal (for instance the 1-2 common branch) i.e., the program makes **A** (II, JJ) = A (JJ, II), however, the off-diagonal elements should be assigned a minus (-) sign. Also, the program converts all these impedance entries to admittances in complex notation to keep the matrix solution in proper units. Notice II cannot be larger than JJ.

3rd set of cards: Follow these cards with a blank card.

4th set of cards: II, JJ, YR, YI, ITYP of Y parameters (admittances)

where: II = matrix row element

JJ = matrix column element

YR = real part of Y parameter

YI = imaginary part of Y parameter

ITYP = 0 if symmetrical, 1 if asymmetrical

INPUT INFO - RF AMPLIFIER
12 NODES

```
ZZJOB 5
ZZXEQSIMPAD1                   3
   12          56.E+06
    1    1     50.                0.                   0.                0
    1    1     0.                 0.                   4.      E-12      0
    1    1     .284               .0589     E-06       0.                0
    1    2     -.284              -.0589    E-06       0.                0
    2    2     .284               .0589     E-06       0.                0
    2    2     0.                 .0301     E-06       0.                0
    2    2     .189               .0293     E-06       153.    E-12      0 )
    2    3     -.189              -.0293    E-06       -153.   E-12      0 } TUNE
    3    3     .189               .0293     E-06       153.    E-12      0 )
    3    3     0.                 0.                   2130.   E-12      0
    3    3     -.0034             .00581    F-06       85.0    E-12      0 )
    3    4     .0034              -.00581   E-06       -85.0   E-12      0 } TUNE
    4    4     -.0034             .00581    E-06       85.0    E-12      0 )
    4    4     .123               .0646     E-06       0.                0
    4    4     .0521              .0026     E-06       0.                0
    4    5     -.0521             -.0026    E-06       0.                0
    5    5     .0521              .0026     E-06       0.                0
    6    6     14.7316            .00806    E-06       0.                0
    6    7     -14.7316           -.00806   E-06       0.                0
    7    7     14.7316            .00806    E-06       0.                0
    7    7     .00905             .00697    E-06       0.                0
    7    7     .0022              -.00123   E-06       0.                0
    7    8     -.0022             .00123    E-06       0.                0
    8    8     .0022              -.00123   E-06       0.                0
    8    8     0.                 0.                   1040.   E-12      0 TUNE
    9    9     .154               0.                   0.                0
    9   10     -.154              0.                   0.                0
   10   10     .154               0.                   0.                0
   10   10     0.                 0.                   1500.   E-12      0 TUNE
   10   10     .0141              .00205    E-06       0.                0
   10   11     -.0141             -.00205   E-06       0.                0
   11   11     .0141              .00205    E-06       0.                0
   11   11     0.                 .00286    E-06       0.                0
   11   11     .0487              .01404    E-06       0.                0
   11   12     -.0487             -.01404   E-06       0.                0
   12   12     .0487              .01404    E-06       0.                0
   12   12     18.4               0.                   0.                0

    5    5     3.4       E-03     13.8      E-03       0
    5    6     2.12      E-06     .206      E-06       1
```

-20-

120

FIGURE 9
PAGE 1

```
   6   5    88.2    E-03    -32.3    E-03    1
   6   6    -.163   E-03     1.15    E-03    0
   8   8    68.8    E-03    278.     E-03    0
   8   9    9.59    E-06    .933     E-06    1
   9   8    398.    E-03    145.5    E-03    1
   9   9    -3.3    E-03    23.3     E-03    0

    1  .02             0.

ZZZZ
```

FIGURE 9
PAGE 2

In this set of cards, II can be larger than JJ. This happens in the case of current generators which must be assigned ITYP = 1. The reason for making this asymmetrical assignment is to prevent the program from switching the element from the A(II, JJ) position in the matrix to the A(JJ, II) position since the generator looks like an off-diagonal element. Switching should not happen in the case of off-diagonal current generators in the "Y" model because they are not symmetrical elements by nodal analysis theory.

NOTE: For current generators, direction (or sign) is positive leaving node.

5th - Follow these cards with a blank card.

6th set of cards: J, VR, VI

where J = node number associated with source I.

VR = real part of source I.

VI = imaginary part of source I.

NOTE: In Figure 9 this is obviously the source generator (node 1) but must be considered current, therefore assuming a one (1) volt source and source resistance = 50 ohms gives I = 1/50 = 0.02 as shown.

7th - Follow these cards with a blank card.

8th - End of job.

The elements marked "tune" represent the capacitors we tuned in this circuit to obtain desired results. Different values must be used for these cards obviously and program repeated until best results are obtained. This is not too cumbersome since computer running time is short (i.e., a 12 node run approximately 5 minutes 1620 time).

Now, for the purposes of practice and application let us create a more complicated (but fictitious in this case) problem using the circuit originally set forth in Figure 4. The input information is shown in Figure 10. The impedance branches are straightforward as before and need not be discussed but the admittance branches containing the "Y" parameters of the cascoded transistors and emitter resistors become much more "messy". For this reason, and in the original case which the authors had to do, the "Y" matrix elements associated with respective node equations should be set-up as shown in Figure 11. Page 2 of Figure 10 shows input elements sectioned off by the circled node numbers to serve as a guide when looking at the "Y" element equations in Figure 11. Notice, the "A"s and "S"s refer to asymmetrical and symmetrical elements respectively. Notice also, and as similar in the case of impedances, symmetrical admittance branches need not be repeated in input data.

There are a few sense switch controls in the program mainly for convenience. For instance, Sense Switch 1 On will punch output of coefficient matrix and together with Sense Switch 3 will punch out input matrix. Sense Switch 2 will reinvert matrix once after initial inversion. Sense Switch 4 On will <u>not</u> punch out R, L and C's and all final admittances. Initially, as we generally do, leave sense switch 4 off in order to cross check admittance values with possible slide rule calculations. Then after beginning to tune circuits, putting Sense Switch 4 On will produce only final node voltages. This saves time getting punch bound.

The foregoing discussion is complete as possible and should provide 1620 users a convenient engineering tool for solving A-C circuit analysis. It has been the intent of this paper to provide 1620 users with a method to solve difficult problems such as the R.F. amplifier presented which can normally be solved by larger computers and more complicated programs.

Node 5 $\underbrace{(Y_{11E} + Y_5 + Y_{22E})}_{S}V_5 - \underset{S}{(Y_{11E})}V_6 - \underset{S}{(Y_{22E})}V_7 - (1.785\, Y_{12E})(\underset{A}{V_7} - \underset{S}{V_5}) - (1.785\, Y_{21E})(\underset{A}{V_6} - \underset{S}{V_5})$ ---- = 0

Node 6 $\underbrace{(Y_{4-6} + Y_{11E})}_{S}V_6 - \underset{S}{(Y_{4-6})V_4} - \underset{S}{(Y_{11E})V_5} + \underbrace{(1.785\, Y_{12E})(V_7 - V_5)}_{A}$ ---- = 0

Node 7 $\underbrace{(Y_{22E} + Y_{11B})V_7}_{S} - \underset{S}{(Y_{22E})V_5} + \underbrace{(1.785\, Y_{21E})(V_6 - V_5)}_{A} - \underbrace{(1.785\, Y_{12B})V_8}_{A}$ ---

Node 8 $\underbrace{(Y_{8-9} + Y_{22B})V_8}_{S} - \underbrace{(1.785\, Y_{21B})V_7}_{A} - \underset{S}{(Y_{8-9})}V_9$ ----

Node 10 $\underbrace{(Y_{11E} + Y_{10} + Y_{22E})}_{S}V_{10} - \underset{S}{(Y_{11E})}V_{11} - \underset{S}{(Y_{22E})}V_{12} + (8.06\, Y_{12E})(\underset{A}{V_{12}} - \underset{S}{V_{10}}) + (8.06\, Y_{21E})(\underset{A}{V_{11}} - \underset{S}{V_{10}})$ ----

Node 11 $\underbrace{(Y_{11-12} + Y_{11E} + Y_{9-11} + Y_{11})}_{S}V_{11} - \underset{S}{(Y_{11-12})}V_{12} - \underset{S}{(Y_{9-11})}V_9 - \underbrace{(8.06\, Y_{12E})(V_{12} - V_{10})}_{A} - \underset{S}{(Y_{11E})}V_{10}$ ----

Node 12 $\underbrace{(Y_{11-12} + Y_{22E} + Y_{11B})}_{S}V_{12} - \underset{S}{(Y_{11-12})}V_{11} - \underset{S}{(Y_{22E})}V_{10} - \underbrace{(8.06\, Y_{21E})(V_{11} - V_{10})}_{A} + \underbrace{(8.06\, Y_{12B})V_{13}}_{A}$ ----

Node 13 $\underbrace{(Y_{13-14} + Y_{22B})V_{13}}_{S} + \underbrace{(8.06\, Y_{21B})V_{12}}_{A} - \underset{S}{(Y_{13-14})}V_{14}$ ---

Figure 11 16 Node RF Amplifier

INPUT INFO - RF AMPLIF[illegible] - EXAMPLE PROBLEM
16 NODES

```
ZZJOB 5
ZZXEQSIMPAD1                3
  16          56.E+06
   1   1    50.             0.                 0.             0
   1   1    0.              0.                 4.      E-12   0
   1   1    .284            .0589    E-06      0.             0
   1   2    -.284           -.0589   E-06      0.             0
   2   2    .284            .0589    E-06      0.             0
   2   2    0.              .0301    E-06      0.             0
   2   2    .189            .0293    E-06      153.    E-12   0 ⎫
   2   3    -.189           -.0293   E-06      -153.   E-12   0 ⎬ TUNE
   3   3    .189            .0293    E-06      153.    E-12   0 ⎭
   3   3    0.              0.                 2130.   E-12   0
   3   3    -.0034          .00581   E-06      96.5    E-12   0 ⎫
   3   4    .0034           -.00581  E-06      -96.5   E-12   0 ⎬ TUNE
   4   4    -.0034          .00581   E-06      96.5    E-12   0 ⎭
   4   4    .123            .0646    E-06      0.             0
   4   4    .0521           .0026    E-06      0.             0
   4   6    -.0521          -.0026   E-06      0.             0
   5   5    3.13            0.                 0.             0
   6   6    .0521           .0026    E-06      0.             0
   8   8    14.7316         .00806   E-06      0.             0
   8   9    -14.7316        -.00806  E-06      0.             0
   9   9    14.7316         .00806   E-06      0.             0
   9   9    .00905          .00697   E-05      0.             0
   9   9    .0022           -.00123  E-06      0.             0
   9  11    -.0022          .00123   E-06      0.             0
  10  10    1.045           0.                 0.             0
  11  11    .0022           -.00123  E-06      0.             0
  11  11    0.              0.                 1185.   E-12   0   TUNE
  11  11    231.            0.                 0.             0
  11  12    -231.           0.                 0.             0
  12  12    231.            0.                 0.             0
  13  13    .154            0.                 0.             0
  13  14    -.154           0.                 0.             0
  14  14    .154            0.                 0.             0
  14  14    0.              0.                 1500.   E-12   0   TUNE
  14  14    .0141           .00205   E-06      0.             0
  14  15    -.0141          -.00205  E-06      0.             0
  15  15    .0141           .00205   E-06      0.             0
  15  15    0.              .00286   E-06      0.             0
  15  15    .0487           .01404   E-06      0.             0
  15  16    -.0487          -.01404  E-06      0.             0
```

FIGURE 10
PAGE 1

```
 16  16    .0487              .01404  E-06    0.              0
 16  16    15.4               0.              0.              0

  5   5    3.52     E-03      13.4    E-03    0
  5   5    .048     E-03      1.28    E-03    0
  5   5    0.                 -.285   E-03    0
  5   5    91.      E-03      -23.2   E-03    0
  5   6    -3.52    E-03      -13.4   E-03    0      5
  5   7    -.048    E-03      -1.28   E-03    0
  5   7    0.                 .285    E-03    1
  5   6    -91.     E-03      23.2    E-03    1
  6   5    0.                 .285    E-03    1
  6   6    3.52     E-03      13.4    E-03    0      6
  6   7    0.                 -.285   E-03    1
  7   5    -91.     E-03      23.2    E-03    1
  7   6    91.      E-03      -23.2   E-03    1
  7   7    .048     E-03      1.28    E-03    0      7
  7   7    207.     E-03      -137.   E-03    0
  7   8    .482     E-03      .91     E-03    1
  8   7    107.3    E-03      -87.5   E-03    1
  8   8    .865     E-03      2.69    E-03    0      8
 10  10    71.7     E-03      274.    E-03    0
 10  10    .977     E-03      26.     E-03    0
 10  10    0.                 1.29    E-03    0
 10  10    -411.    E-03      105.    E-03    0
 10  11    -71.7    E-03      -274.   E-03    0      10
 10  12    -.977    E-03      -26.    E-03    0
 10  12    0.                 -1.29   E-03    1
 10  11    411.     E-03      -105.   E-03    1
 11  10    0.                 -1.29   E-03    1
 11  11    71.7     E-03      274.    E-03    0      11
 11  12    0.                 1.29    E-03    1
 12  10    411.     E-03      -105.   E-03    1
 12  11    -411.    E-03      105.    E-03    1
 12  12    .977     E-03      26.     E-03    0      12
 12  12    4200.    E-03      -2800.  E-03    0
 12  13    -2.18    E-03      -4.11   E-03    1
 13  12    -483.    E-03      395.    E-03    1
 13  13    17.6     E-03      54.7    E-03    0      13

    1  .02          0.

ZZZZ
```

FIGURE 10
PAGE 2

SESSION NUMBER F.2.2

SPEAKERS

HARRY CADOW ON 360 OPERATOR TRAINING.
GARTH GROFT ON OPEN SHOP OPERATION IN AN ENGINEERING DEPARTMENT.

OBTAINING MAXIMUM RESULTS USING A COMPUTER FOR ENGINEERING DESIGN

by

Garth E. Groft

York Division, Borg-Warner Corporation

ABSTRACT

For more than five years a major manufacturer of refrigeration and air-conditioning equipment has made comprehensive use of an electronic computer as an engineering tool. Its use is directed toward increasing the productivity of the design and test engineers, thereby reducing the cost of engineering.

It is well known that a substantial part of engineering is involved with tedious and repetitive calculations. Much of this work can be done more quickly and more accurately by a computer. Both the quantity and quality of product design and development are enhanced as faster, more accurate, solutions allow more time to explore a wider range of design possibilities and to conduct more extensive test programs.

The facility is operated on an "open shop" basis whereby the engineers have direct 24 hour/day access to the computer and are encouraged to use it as freely as they do a slide rule. Courses are frequently given to train the engineers so that they themselves can program and operate the computer. To date, 94 engineers have completed the courses. This arrangement minimizes the number of steps in the communication chain, and has resulted in a most effective use of the computer for design and development work. This paper discusses in detail the manner in which an "open shop" operation is achieving outstanding results.

OBTAINING MAXIMUM RESULTS USING A COMPUTER FOR ENGINEERING DESIGN

by

Garth E. Groft

York Division, Borg-Warner Corporation

For more than five years a major manufacturer of refrigeration and air conditioning equipment has made comprehensive use of an electronic computer as an engineering tool. Its use is directed toward increasing the productivity of the design and test engineers, thereby reducing the cost of engineering.

PURPOSE OF THE ENGINEERING COMPUTER

It is well known that a substantial part of engineering is involved with tedious and repetitive calculations. Much of this work can be done more quickly and more accurately by a computer. By using the computer, the engineer is no longer limited to approximations, oversimplified formulas, and short-cut methods that have been in wide use for many years. The computer can be programmed to include equations in their entirety and to consider factors that previously had to be ignored.

The computer provides more reliable, more consistent results. The engineer is assured that a computer program will repeatedly handle data in precisely the same manner - that is to say, all calculations are performed the same way each time the computer is used. Hence all data has a common basis of analysis. He is further assured that the computer is not subject to errors caused by distractions, nervous tension or fatigue.

Both the quality and quantity of product design and development are enhanced, as faster, more accurate, more reliable solutions allow more time to explore a wider range of design possibilities and to conduct more extensive test programs. The computer has contributed significantly toward reducing the total design and development time cycle. The effectiveness of the computer in saving time is emphasized by the fact that the work being done by this installation is equivalent to an additional staff of more than 40 engineers using slide rules and desk calculators.

GROWTH OF THE FACILITY

The first engineering computer facility was an IBM 1620 Model I Paper Tape System, installed in December 1961. Acceptance of the computer as an integral part of engineering was exceptionally favorable. This was due largely to the extensive personnel conditioning, training, programming and general preparations which took place before the equipment arrived. The demand for the computer quickly exceeded expectations as well as its capacity, necessitating upgrading just a year later, to a faster, more efficient card system. The core storage was also increased from 20,000 to 40,000 6-bit digits. This provided sufficient thruput until early 1966. Additional capabilities were then provided by converting to a 1620 Model II having twice the speed of the Model I. This system provided more effective operation and permitted a higher level of programming capability. A later appraisal of future engineering computing needs and proposed improvement in test data collection equipment and techniques dictated the further change to an IBM 1800 Data Acquisition and Control System. This was installed in April, 1967.

The 1800 is ideally suited to engineering use. It can execute both process control and non-process control programs concurrently. Batch processing and time sharing at the computer console or from remote terminals is provided by the system. The system software and most of our program library reside on two disk packs of 512,000 words each. A core storage of 16384 16-bit words was found to be equivalent to some 50-60,000 digits. We have also determined that for our work the 1800 is 2-1/2 to 3 times faster than the equipment it replaced, or about 15 times faster than the original tape system.

COMPUTER TRAINING COURSE

Frequent courses are given to train engineers so that they themselves can program and operate the computer. These 50 hour courses are divided into two-hour sessions, so as to minimize interruption of the engineer's regular work. The first part of the course presents 1800 FORTRAN.

Because FORTRAN is machine independent, the engineer can be problem-oriented, i.e., he can direct his attention toward the method of solving the problem itself. The FORTRAN language is quickly learned and a program can be written with relative ease. Examples of FORTRAN statements from actual programs are used in these sessions to illustrate the flexibility and practicality of the FORTRAN language. In the interest of saving training time and avoiding confusion, the design and internal workings of the computer are pointedly avoided.

The second phase of the course concerns the three major areas of difficulty which tend to discourage our engineers from writing computer programs:

1). Methods for representing tabular or graphical data in a computer program,

2) Techniques for solving trial and error problems, and

3) Means for referencing properties of refrigerants and other fluids.

Curve fitting and surface fitting techniques are presented as a likely method for representing graphs and tabular data which form smooth curves. The applications of table look-up procedures are reviewed. Normally this approach is not recommended as the equations resulting from curve or surface fitting are easier to program, require less storage and generally provide faster, more accurate results.

Techniques for solving trial and error problems are treated in detail. The advantages and disadvantages of various methods are explored in detail with actual programming examples.

Sub-programs for the properties of commonly used refrigerants have been written from equations developed by a major manufacturer of refrigerants. These same equations generated the tabular refrigerant data frequently used in manual calculations. These properties can be referenced freely in any program and are as easy to use as sine, cosine, and square root determinations.

<u>Examples</u>:

Pressure of saturated liquid P1 = PFT(T1)

Specific volume of saturated vapor V1 = VPT(P1,T1)

Enthalpy of vapor H1 - HPT(P1,T1)

Enthalpy of vaporization HLAT1 = HFGT(T1)

Where T1 = Temperature in ^{o}F

P1 = Pressure in psia

The refrigerant properties include acoustic velocity, heat capacity at constant pressure and at constant volume; enthalpy and entropy of vaporization; specific volume, pressure, temperature, enthalpy, and entropy of the saturated liquid, and of the superheated and saturated vapor; liquid and vapor viscosity; and X and Y compressibility functions. Other properties will be added to this set as they are needed. Subprograms for the properties of water, air, carbon dioxide, methane, ammonia, lithium bromide, chlorine, and other such fluids are also available.

During the third and final phase of the course the engineer programs several typical applications in class. This training is to prepare him for the ultimate objective of the course - to write a useful program himself and get actual results on the computer. Experience has shown that only by actually writing and running a computer program does the engineer acquire good recall of the course content. The course throughout avoids the commonly encountered textbook approach. Instead, material, examples and illustrations gathered from everyday experience are used.

The engineer continues to receive training on an individual basis each time he uses the computer facility. He is given an Engineering Computer Guidebook, an in-house publication which provides detailed programming and operating information specifically designed to promote a greater degree of self-sufficiency and improve the overall productivity of the engineer using the 1800 computer.

To date, 94 engineers have completed the FORTRAN course and most of these are now making effective use of the computer. Our own Program Library has grown to 175 programs covering a broad range of applications. The major areas where the computer is utilized are:

1. Design
2. Data reduction/analysis
3. Performance ratings
4. Physical science
5. Mathematics, statistics and reliability
6. Information processing and retrieval
7. Management science
8. Numerical paper tape control of machine tools

OPEN SHOP OPERATION

Before the initial computer installation, it was recognized that it is easier to train engineers to write their own programs than to familiarize a machine language oriented programmer with the broad aspects and fine variations of engineering design and development.

For this reason, the computer has always been used on a strictly "open shop" basis, whereby the engineers operate the computer and do most of their own programming. Direct 24 hour/day access to the computer is provided and they are encouraged to use it as freely as they do a slide rule. Computer oriented help is available, however, for consultation and to assist in the more involved problems.

A magnetic type schedule board is used to reserve computer time. Permanent reservations are provided for certain test activities which go on around the clock. All other reservations are made as needed. Conflicts in scheduling or other such difficulties are resolved by computer personnel.

The keypunching of new programs and input data cards is done by a keypunch operator. The engineers, however, are trained to punch their own cards so that they are able to use the computer after hours without assistance.

The "open shop" operation minimizes the number of steps in the communications chain. The engineer does not have to explain his problem to a programmer who, no matter how intelligent, resourceful, and eager to help, rarely has as much knowledge and experience as an engineer in his own bailiwick. The problem of retrieving and relating all pertinent information and ensuring that nothing is lost in the translation is eliminated when the engineer does his own programming. While programs written by engineers may not always be as efficient as those created by programming specialists, the qualitative benefits of "open shop" programming far outweigh the disadvantages of any slightly slower running times.

Like any commercial product, a program is subject to occasional or frequent revision as dictated by technological innovations and advances. The engineer who does not write his own program frequently experiences difficulty and delays in keeping the program up to date. Furthermore, as the engineer operates the computer, he acquires valuable feedback regarding the effectiveness of his own program and can detect unsatisfactory program performance, implement necessary improvements, and prevent needless delays caused by incorrect input data.

Moreover, hands-on experience contributes greatly toward building up the confidence of the engineer in utilizing the computer facilities. It removes the mystery, awe, and fear often associated with computers. The engineer who uses the computer himself no longer feels that it is a gigantic brain with which he is in fierce competition.

This attitude, although diminishing with each passing year, presented a real problem with early installations and is still of consequence. Under "open shop" conditions, the engineer acquires a genuine appreciation for the computer and gradually realizes that, even though inanimate, it is a valuable partner in a closely knit engineering team.

The "open shop" philosophy coupled with "in-house" training has proven a most effective approach to computer usage in design and development work. It is undoubtedly the basic reason that this installation is achieving such outstanding results.

G.E.G.
7/19/67

SESSION #		MC CRACKEN	1800 FORTRAN LANGUAGE REFERENCE MANUAL
1	INTRODUCTION TO FORTRAN	CHAP. 1, PAR. 1.1 & 1.2	
2	CONSTANTS, VARIABLES & EXPRESSIONS (INCLUDING SUB-SCRIPTED VARIABLES)	CHAP. 1, PAR. 1.3 - 1.5, PROB. 1 - 7 CHAP. 5, PAR. 5.1, 5.2, 5.5	PP. 1 - 5, 18 - 19 (DIMENSION)
3	ARITHMETIC STATEMENTS QUIZ #1 ON SESSIONS 1 - 3 (1/2 HOUR)	CHAP. 2, PROB. 1 - 3 CHAP. 5, PROB. 1, 2, 3, 8, 9, 10	P. 6 (COL. 1) PP. 22 - 23 (FORTRAN SUPPLIED SUBPROGRAMS)
4	CONTROL STATEMENTS	CHAP. 3, PAR. 3.4 & 3.5 CHAP. 4, PROB. 1 - 13	P. 6 (COL. 2) - P. 7 (DO STATEMENT) P. 9 (PAUSE, STOP, END)
5	DO STATEMENTS	CHAP. 5, PAR. 5.3, PROB. 5, 6, 7, 11 CHAP. 6, PROB. 1 - 9	P. 7 (DO STATEMENT) - P. 9 (CONTINUE)
6	DO STATEMENTS	CHAP. 6, PROB. 10, 13 - 17	P. 7 (DO STATEMENT) - P. 9 (CONTINUE)
7	REVIEW & QUIZ #2 ON SESSIONS 1 - 6 (1 HOUR)		
8	DISCUSSION OF QUIZ #2 INPUT/OUTPUT STATEMENTS		PP. 10 - 14
9	INPUT/OUTPUT STATEMENTS	CHAP. 3, PROB. 1 - 7	PP. 15 - 17 (OMIT MANIPULATIVE INPUT/ OUTPUT STATEMENTS)
10	INPUT/OUTPUT STATEMENTS	CHAP. 7, PROB. 1, 5, 7, 9	PP. 10 - 17
11 & 12	STATEMENT FUNCTIONS, SUBPROGRAMS & SPECIFICATION STATEMENTS	CHAP. 8	PP. 19 - 25 P. 26 (CALL DATSW STATEMENT)
13	REVIEW	HANDOUT - PROB. SETS 1 - 7	ALL STATEMENTS

SESSION #		MC CRACKEN	1800 FORTRAN LANGUAGE REFERENCE MANUAL
14	QUIZ #3 ON SESSIONS 8 - 12 INPUT/OUTPUT,FORMAT, STATEMENT FUNCTIONS, SUBPROGRAMS (1-1/2 HOURS)		
15	DISCUSSION OF QUIZ #3 CASE STUDY 1 CASE STUDY 8 FLOW DIAGRAM	CHAP. 9, PP. 62 - 63 CHAP. 9, PP. 73 - 77	
16 17 18	METHODS OF ITERATION		
19	GENERAL DISCUSSION OF PROJECT DEVELOPMENT TO DATE		
20	COMPUTER DEMO - GENERAL SESSION		
21	CASE STUDY 2 MOTOR EFFICIENCY CALCULATIONS	CHAP. 9, P. 64	
22	COMPUTER DEMO - WITH CASE STUDY 2		
23	CASE STUDY 4 HEAT TRANSFER PROB.	CHAP. 9, PP. 66 - 68	
24	COMPUTER DEMO - WITH CASE STUDY 4		
25	TEST & EXECUTE PROJECT PROGRAMS		

SESSION NUMBER F.2.3

SPEAKERS

D. GARDNER, ON SUBROUTINE DATAR

Subroutine DATAR

Summary:

I am going to speak to you today about the subroutine DATAR and some of the history behind its conception.

DATAR is a format-free input subroutine designed to read numerical input data with or without a decimal point. A blank space is used as the delimiter (separator) and as many pieces of data (or as few) may be punched on a data card as desired. The call statement is CALL DATAR(K,X) which initiates the reading of a data card and the beginning of the search for K pieces of data which will be returned as floating point (real) variables in the vector X.

Background:

We at General Foods have enjoyed the usage of "format-free" input capability as long ago as 1961 when the first Fortran compiler was made available for the IBM 1620. The only problem with the input capability of that compiler was that each time the READ statement was encountered, a new data card was not read. That is, the reading of new data carried on in the input area of the compiler where the reading of old data left off. This caused problems when one had more data on a data card than one wanted to read. This problem was "solved" when the next compiler from IBM and all subsequent ones for the 1620 (and the present compiler for the 1130) incorporated the format concept of input data that we are all familiar with. Other compilers for the 1620 (AFIT and PDQ-written by users) attempted to give format-free input capability, but each had its limitations.

The few programs having the input capability we wanted were written in SPS (the assembler language for the 1620) and utilized a general purpose format-free input routine also written in SPS.

There were many other capabilities lacking from the earlier compilers for small scientific computers such as the 1620 which drove others to write a group of input/output subroutines (also written in SPS) for use with commercial applications utilizing programs written in Fortran. These subroutines were called FORCOM, an acronym for Fortran Commercial.

At the Denver COMMON meeting in July, 1966 Jim Kokie of IBM was explaining the FORCOM subroutines written for the 1130 (subsequently changed from a Type III program to a Type II one entitled "1130 Commercial Subroutine Package"). He made what appeared to be an astounding statement when he said, "The interesting aspect of seven of these eight subroutines is that they are written in Fortran!" Astounding, indeed, for he had just said in effect that with 1130 Fortran one now has assembler language maneuverability at a higher programming level.

Method:

The method employed in DATAR and the other routines I have mentioned is a relative simple one. In 1130 Fortran (and also the 360 Fortrans) one has the capability of reading information from a data card in A1 format from 1 to 80 columns. When the variable used for reading is an integer variable, the resultant value after the read (under A1) is a numerical one which can be used for comparing and other purposes with no problems (Data read under an A2 format results in similar numerical-type values). Table 1 attached shows the numerical equivalent values for all valid 1130 characters which have been read under an A1 format. (This is p. 150 from the 1130 Commercial Subroutine Package, Version 2, Program Reference Manual, H20-0241-2).

This means, of course, that one can easily identify any valid 1130 character which is to be read in on cards by storing the appropriate decimal equivalents in your program and comparing these stored values with the ones read-in under the A1 format. If we wanted to recognize a slash (/) mark, for instance, one might put the statement K=24896 in his program, read the appropriate variable in A1 format, and compare the two. One can do many nice programming tricks this way.

It is worthy to note in passing the relationship among the decimal equivalents of the ten digits. Each is 256 larger than the preceding one which allows for one to translate the decimal equivalent (K) of a digit read in A1 format into its digit equivalent (I) with a single Fortran statement:

```
I=(K+4032)/256
```

The letters can be related to the numbers 1-26 in a similar way, but one must program around the two discontinuities noted between the letters I-J and R-S.

Remarks:

DATAR was possible only because of the capability of reading characters on a data card in an A1 format. The attached abstract, flow chart, and Fortran listing of the subroutine clearly shows what is taking place. However, a few remarks are in order concerning the philosophy of this format-free input subroutine.

1. Everytime the call statement CALL DATAR(K,X) is encountered in a program, a new data card will be read and scanning started in column 1. As many data cards as necessary will be read until the K pieces of data (separated by at least one blank column) have been encountered. All numbers with or without decimal points will be converted to floating point and returned in the vector X.

2. Only numeric data can be read by DATAR. Any character other than a decimal point (.), minus sign (-), plus sign (+), or the digits (0-9) will cause an error message to print on the printer. If an error is encountered, the routine will halt on a PAUSE and will return to the first statement in the subroutine after PROGRAM START is pushed. (Thus, one can fix the error and continue at the point of the calling statement).

   Note: Both an 026 plus sign (12 punch) and an 029 plus sign (12-6-8) can be recognized.

3. A decimal point alone, a minus or plus sign alone, or a minus or plus sign with a decimal point alone will result in the variable being set to zero.

4. The subroutine at present scans columns 1-72 for data. If any other limit besides 72 is desired, a one statement change is all that is necessary.

)ther Applications:

I have already mentioned the Commercial Subroutine Package ›ut it is worth mentioning again, particularly Version 2 which :ontains some twenty-three subroutines most of which are ritten in Fortran.

Clearly, one could use this method very effectively when ne wants to generate alphabetic information from within one's rogram. Of course, the "H" specification would also work nless the information is to be printed in variable places n the page (like the word "TOTAL" as a heading of a column f figures whose placement on the page is variable depending pon problem size).

I have used this idea many times in many programs, some of hich are shown below:

1. In a "scatterplot" program each X-Y value is plotted as an "A". If more than one data point falls within the same grid, it would show as a "B" etc.

2. In another program I convert the digits 1-6 to the letters A-F which is quite easily done (could be done, of course, for all the letters).

3. The following table shows five separate row labels, any one of which could be encountered in a program. The index i is used to generate the proper row label.

i	Condition
1	A
2	A B
3	A B C
4	A B C D
5	A B C D E

4. In another program (familiar to some of you) the row labels were generated from the i index according to the following correspondence.

i	Row Label
1	A
2	B
3	A B
4	C
5	A C
6	B C
7	A B C

439

5. In the same program as (4) other row labels were generated in this way:

i	Row Label		
1	1	1	1
2	1	1	2
3	1	2	1
4	1	2	2
5	2	1	1
6	2	1	2
7	2	2	1
8	2	2	2

As before, the index _i_ generated the numbers in A1 format so that blanks could be printed if the problem size were smaller than three.

6. In a one column distribution program one could separately identify all digits, letters, and special characters.

In closing, I shall only say that the use of the Fortran language in this way is only limited by one's imagination.

DSG/mh

TABLE 1

EBCDIC CHARACTERS AND DECIMAL EQUIVALENTS

A	-16064	S	-7616	blank	16448
B	-15808	T	-7360	. (period)	19264
C	-15552	U	-7104	< (less than)	19520
D	-15296	V	-6848	(	19776
E	-15040	W	-6592	+	20032
F	-14784	X	-6336	&	20544
G	-14528	Y	-6080	$	23360
H	-14272	Z	-5824	*	23616
I	-14016	0	-4032	)	23872
J	-11968	1	-3776	- (minus)	24640
K	-11712	2	-3520	/	24896
L	-11456	3	-3264	,	27456
M	-11200	4	-3008	%	27712
N	-10944	5	-2752	#	31552
O	-10688	6	-2496	@	31808
P	-10432	7	-2240	' (apostrophe)	32064
Q	-10176	8	-1984	=	32320
R	-9920	9	-1728		

Subroutine DATAR

Purpose:

To read data in "free" format from cards, each piece separated from each other by at least one blank column (between cols 1-72)

Usage:

Call DATAR(KTOT,A)

Description of Parameters:

input	KTOT -	number of pieces of data to be read
output	A -	the output vector containing the KTOT pieces of data

Remarks:

An error condition will result if any character other than the digits 0-9, a plus sign (+), a minus sign (-), or the decimal point (.) is encountered.

The "end of card" specification is set at column 72. If another specification is wanted (say, column 80 for a complete card scan); change the first card of the subroutine (DATAR 13) accordingly (i.e., KK = 80).

Subroutines Required:

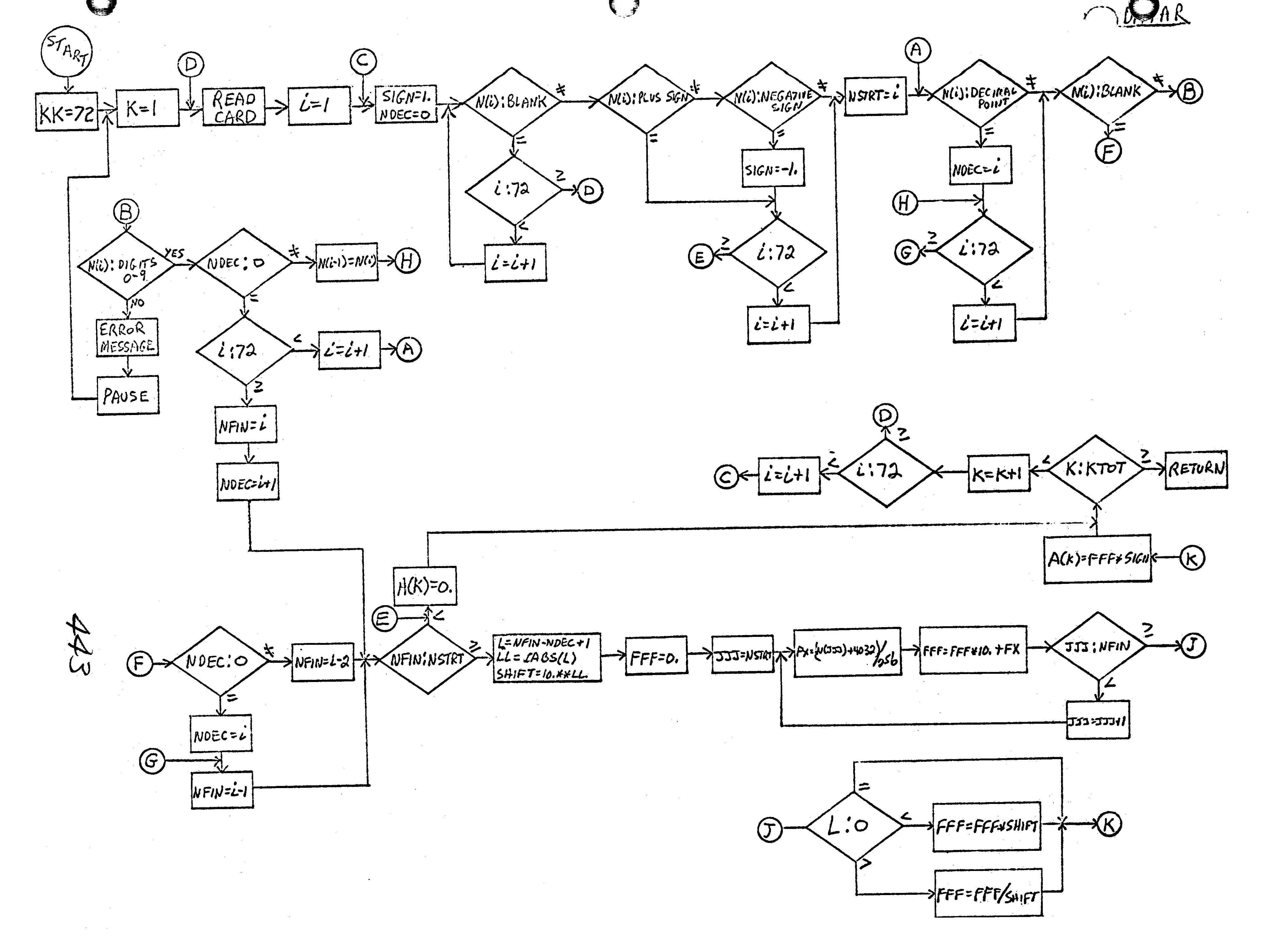
None

Input - Output Devices Used:

1442 Card Reader, 1132 Printer

Method:

Each data card is read with the format 80 A 1. Each character is scanned and the proper numbers formed arithmetically.



D AR
START
KK=72
K=1
D
READ CARD
i=1
C
SIGN=1.
NDEC=0
N(i):BLANK
≠
=
i:72
≥
D
<
i=i+1
N(i):PLUS SIGN
≠
=
N(i):NEGATIVE SIGN
≠
=
SIGN=-1.
i:72
≥
E
<
i=i+1
NSTRT=i
A
N(i):DECIMAL POINT
≠
=
NDEC=i
H
i:72
≥
G
<
i=i+1
N(i):BLANK
≠
B
=
F
B
N(i): DIGITS 0-9
YES
NO
ERROR MESSAGE
PAUSE
NDEC:0
≠
N(i-1)=N(i)
H
=
i:72
<
i=i+1
A
≥
NFIN=i
NDEC=i+1
D
≥
i:72
<
i=i+1
C
K=K+1
<
K:KTOT
≥
RETURN
A(K)=FFF*SIGN
K
A(K)=0.
E
<
NFIN:NSTRT
≥
L=NFIN-NDEC+1
LL=IABS(L)
SHIFT=10.**LL
FFF=0.
JJJ=NSTRT
FX=(N(JJJ)+4032)/256
FFF=FFF*10.+FX
JJJ:NFIN
≥
J
<
JJJ=JJJ+1
F
NDEC:0
≠
NFIN=i-2
=
NDEC=i
G
NFIN=i-1
J
L:0
=
<
FFF=FFF*SHIFT
>
FFF=FFF/SHIFT
K

```
// DUP
*DELETE             DATAR
// FOR
*ONE WORD INTEGERS
      SUBROUTINE DATAR(KTOT,A)
C
C        FREE FORMAT INPUT ROUTINE
C
C              KTOT=NUMBER OF ELEMENTS TO BE READ AND PLACED IN *A*
C              A=OUTPUT VECTOR OF VARIABLES
C
      DIMENSION A(1)
      DIMENSION N(80)
C
C        SET *END OF CARD* AT COLUMN 72
C
      KK=72
C
    1 K=1
C
C        READ DATA CARD
C
  103 READ(2,40)(N(I),I=1,KK)
   40 FORMAT(80A1)
C
C        INITIALIZE AND SET-UP
C
      I=1
  137 SIGN=1.
      NDEC=0
C
C        CHECK FOR BLANK
C
  105 IF(N(I)-16448) 132,101,132
C
C        CHECK FOR END OF CARD (COL *KK*)
C
  101 IF(I-KK) 104,103,103
  104 I=I+1
      GO TO 105
C
C         CHECK FOR PLUS SIGN (12-6-8 CARD PUNCH-029)
C
  132 IF(N(I)-20032) 138,133,138
C
C        CHECK FOR PLUS SIGN (12 CARD PUNCH-026)
C
  138 IF(N(I)-20544) 102,133,102
C
C        CHECK FOR MINUS SIGN(-)
C
  102 IF(N(I)-24640) 108,106,108
  106 SIGN=-1.
C
C        CHECK FOR END OF CARD (COL *KK*)
C
  133 IF(I-KK) 107,128,128
  107 I=I+1
  108 NSTRT=I
```

```
C          CHECK FOR DECIMAL POINT(.)
C
  100 IF(N(I)-19264) 112,109,112
  109 NDEC=I
C
C          CHECK FOR END OF CARD (COL *KK*)
C
  110 IF(I-KK) 111,115,115
  111 I=I+1
C
C          CHECK FOR BLANK
C
  112 IF(N(I)-16448) 130,113,130
C
C          ACCEPT DIGITS 0-9, REJECT ALL OTHER CHARACTERS
C
  130 IF(N(I)+4032) 131,122,135
  135 IF(N(I)+1728) 122,122,131
  131 WRITE(3,140) N(I),I
  140 FORMAT(1H1,21HNON-NUMERIC CHARACTER,1X,A1,1X,9HIN COLUMN,I3)
      PAUSE
      GO TO 1
C
C          HAS DECIMAL POINT BEEN ENCOUNTERED
C
  113 IF(NDEC) 116,114,116
C
C          NO
C
  114 NDEC=I
  115 NFIN=I-1
      GO TO 117
C
C          YES
C
  116 NFIN=I-2
C
C          CHECK FOR (.), (-), (+), (-.), OR (+.) BY THEMSELVES
C
  117 IF(NFIN-NSTRT) 128,118,118
C
C          IF (.), (-), (+), (-.), OR (+.)  SET A(K)=0.
C
  128 A(K)=0.
      GO TO 129
  118 L=NFIN-NDEC+1
C
C          L=0  NO SHIFT
C          L GREATER THAN 0   SHIFT LEFT
C          L LESS THAN 0   SHIFT RIGHT
C
      LL=IABS(L)
      SHIFT=10.**LL
C
C          TRIM PRECEDING ZERO(ES) FROM NUMBER
C
      KKK=NSTRT
      DO 162 JJJ=KKK,NFIN
      IF(N(JJJ)+4032) 161,162,161
  162 NSTRT=NSTRT+1
```

```
      GO TO 128
C
C         * GET * ROUTINE - A(K) RETURNED AS FLOATING POINT VARIABLE
C
  161 FFF=0.
      DO 160 JJJ=NSTRT,NFIN
  160 FFF=FFF*10.+FLOAT((N(JJJ)+4032)/256)
      IF(L) 150,152,151
  150 FFF=FFF*SHIFT
      GO TO 152
  151 FFF=FFF/SHIFT
  152 A(K)=FFF*SIGN
C
C         CHECK FOR LAST VARIABLE
C
  129 IF(K-KTOT) 120 119,119
  120 K=K+1
C
C         CHECK FOR END OF CARD (COL *KK*)
C
      IF(I-KK) 121,103,103
  121 I=I+1
      GO TO 137
C
C         HAS DECIMAL POINT BEEN ENCOUNTERED
C
  122 IF(NDEC) 124,123,124
C
C         YES - SHIFT DECIMAL DIGITS ONE SPACE TO LEFT
C
  124 N(I-1)=N(I)
      GO TO 110
C
C         NO - CHECK FOR END OF CARD (COL *KK*)
C
  123 IF(I-KK) 125,126,126
  125 I=I+1
      GO TO 100
  126 NFIN=I
      NDEC=I+1
      GO TO 117
  119 RETURN
      END
// DUP
*STORE      WS  UA  DATAR
```

SESSION NUMBER F.2.4

SPEAKERS

DAVID J. MARTIN, CHATTANOGA STATE TECH. INST., ON AN EXPANDED
    EDUCATIONAL COMPUTER SYSTEM

The 1041

AN EXPANDED EDUCATIONAL COMPUTER SYSTEM

A paper

Submitted by

David J. Martin
Associate Professor
Data Processing
Chattanooga State Technical Institute

COMMON Meeting
Cincinnati, Ohio
September 7, 1967

In the fall of 1965, the State of Tennessee entered upon a new adventure in the field of higher education, when Chattanooga State Technical Institute opened its doors for the first time. The purpose of this school was to provide, during a two-year period of study, a junior college education geared, not so much to the development of theories, as to the practical application of these theories.

One of the fields of study included in its program is a data processing curriculum which is divided into three distinct areas. The first of these is, of course, the various programming courses. In the 14 courses in this category, the student is introduced to variety of subjects. From an introduction beginning with unit-record equipment and its applications, he progresses into machine-level language programming languages. From higher level languages such as COBOL, FORTRAN, AND PL/1, he then goes into Systems Design and Analysis.

In keeping with our philosophy that the data processor needs to have a well-rounded educational backgroung, the student is also required to take a group of courses called related studies. Among these courses are English, Public Speaking, Mathematics, and Social Science. Additionally, the student chooses to complete his course of study with courses either in Accounting or in Mathematics and Science.

Our students' courses in computer programming initially included FORTRAN and 1620 SPS during their first year. In order to expand their programming background to include work with other systems, we planned to provide training and experience on the 1401 during their second year. Since we believe it is necessary for a student to compile and execute programs before he can begin really to understand a computer-programming system, we faced a very definite problem. How could we teach the 1401 when our school only has a 1620 computer? The ideal solution was to rent a 1401 computer; therefore, we called up our "friendly IBM representative" and asked the cost for renting an appropriately equipped system. When we heard the price, we realized that our ideal solution would have to be compromised somewhat. Then one of our faculty members suggested that some public-spiritec company might be happy to allow us to use their 1401 system free of charge or at a reasonable cost. But when we realized the amount of machine time and/or cost involved, this idea also faded into the background. And so, you guessed it, I was appointed the task of making our 1620 meet this new challenge.

Up until this point, I had always believed that the 1620 users-group library was a worthwhile undertaking. Now it had a chance to prove itself. At the first of September a year ago I ordered a group of programs known as the 141 Educational Computer System. It included two 1620 programs--a simulator and an SPS assembler.

The simulator portion of the 141 System provided for the simulated execution of 14 1401-type instructions. Among these were the basic input-output instructions--read, write, and punch--the arithmetic instructions of addition and subtraction, the data manipulation instructions of move and load, the various forms of the branch instruction and some few miscellaneous instructions. The indicators that can be tested are those that result from a compare operation; high, low, equal, and unequal. The only provisions made for carriage control on the printer was the testing for channel 12 and skipping to channel 1, this function being automatically performed by the simulator, rather than being controllable by the program. Also provided were subroutines for multiplication, division, zero suppression, and editing of fields in the form of a dollar and cents amount.

We used this system through the entire fall quarter and were very pleasantly surprised with the results obtained. However, it had several dificiencies that we wanted to overcome, so in December of last year I proposed to begin work on a project that led to the development of the 1041 simulator.

One of the first problems encountered in the 141 System concerned the amount of core storage available. Of the thousand positions available the first three hundred were reserved for use as input-output areas. In addition, the four subroutines mentioned above required approximately one hundred positions apiece; therefore the average student problem was limited to less than five hundred positions, which meant that the complexity and size of the various problems ahd to be limited. As a result, the first change was to increase the amount of core to two thousand positions. This change provided greater variety in the programs that could be solved by the students. It also afforded the student experience in using a three position address to denote an address greater than 999.

A second need was for the student to be exposed to the power of the 1401. This need necessitated making available for his use the complete instruction set for a 1401 card system rather than simply the 14 instructions then available. Included in this group of additional instructions were some to provide greater ease in handling problems which were commercial in nature. For example the move characters and edit instruction, probably the most powerful instruction on a 1401, was implemented in full power. Now the user could specify the format in which he wanted information displayed. In the edit-word he could denote the location of the decimal point, any commas, the desire to have leading zeros suppressed or asterisks inserted in their place (a feature known as check protection), the insertion of a dollar sign, and the use of a CR symbol to denote a negative value. The zero-suppression routine, likewise, was replaced by a single instruction. The other instructions added were zero and add, zero and subtract, the multiple-output instructions, and store A and B address registers to facilitate subroutine linkages.

In order to allow printer operations, a control-carriage instruction was included. Finally, for compatability with the 1401, the select stacker instruction was included. Simulation of this instruction is of course physically impossible with a 1620, therefore it is treated as a no-op instruction unless there is an address included with it, in which case the appropriate branch is taken.

We expanded the usefulness of the Branch on Indicator by allowing it to check for the Last Card, Channel 9, and Channel 12 situations.

In order to teach the 1401 indirect addressing scheme, it was necessary to implement the feature known as index registers. By storing appropriate values in the index registers, effective addresses could be generated without modification of the actual addresses in core. The results are faster execution and more efficient use of storage. An additional feature of the 1401 allows for the CPU to utilize the contents of the A or B address registers, rather than obtaining new addresses from core. This feature, like the index registers, results in better use of core and faster program execution.

As a test for this new simulator, one of the programs chosen was the Autocoder Pre-List program. It worked, and so did some 500 student problems run against the system.

The final goal--that of increased Through-put--was achieved in many ways, some of which we have already pointed out. In addition to these, I completely rewrote twelve of the fourteen instruction simulation routines. For instance, new algorithms had to be developed for addition and subtraction since the high-order position of the fields did not necessarily have to be numeric. We also used,wherever possible, the indirect addressing feature of the 1620. The net result is considerably greater Through-put over the 141 simulator. This was especially gratifying when you remember that, in addition to simulating the 1401 instruction, we were also maintaining the current status of the A and B registers.

In addition to the ability to execute programs, both simulators have four routines providing the following features:

1) A formatted dump of the 1401 Storage area
2) The ability to alter the contents of the 1401 storage area
3) The ability to designate at execution time the address from which execution is to commence
4) The ability to initialize the 1401 storage area to blanks. The formatted core dump includes displaying the contents of the various registers and the op-code that was last being executed.

All of these functions are under the control of the operator and are initiated through the use of the 1620 console typewriter.

As all of you are very much aware, the development of a computer is composed of two phases. The first of these, which we have discussed up to this point, is the development of the hardware, but, in order to utilize effectively the elements of the hardware, it is necessary to develop appropriate software. Now let's look at what we have provided in the way of software.

As you remember, there was provided in the original 141 system an SPS Assembler. It of course provided mnemonic op-codes corresponding to the 14 instructions of the 141 simulator plus the pseudo-op-codes necessary in a symbolic language. The highest address it could assemble was 999. It made no provision for index register specification. One of the interesting features of it was that it allowed one-pass assembly only for programs containing fewer than 100 source cards. Finally the label table had a capacity of ninety entries.

Next we see a 141 Autocoder assembler that was developed by two of my second-year students. Though written independently of the 141 SPS Assembler, their program provided many similar features.

Several minor and two major modifications allowed the 141 assembler to be expanded into the 1041 SPS Assembler. One major change allowed the address generation algorithm to specify four-position addresses in three positions and to include any index register associated with that address. A second major change made better use of the available 1620 core storage in order to process much larger programs in one pass--250 to 300 in 20K or up to 700 to 900 on a 40K machine. This was accomplished by storing only the amount of information necessary to produce the source card image on output. The writers of the 141 assembler stored columns one through 55. I had discovered after analyzing many source decks that storing ten to fifteen columns was more than adequate. At long last the 1620 user has for his use, for all practical purposes, a 1401 computer system.

I hope many of you are asking yourselves what is necessary for you to use this system. Needed is a 1620 with indirect addressing and with the additional instruction set. These two features could be eliminated at the cost of slower execution. I will be glad to talk with any of you about other modifications that you might desire. We plan to submit the 1041 system shortly to the 1620 Users' Group Library.

Alas, as I began to breathe a sigh of relief, we recieved conformation that our school's new 360 System is in the process of being installed today. In anticipation of this installation several weeks ago, the question was asked as to when I plan to have a 1401 simulator available for our 360. I categorically denied even the idea of such a project, but I must confess to you that the idea of having a 4K simulator with actual stacker select capability has me already mentally drawing the flow charts.

INSTRUCTIONS	1401	141	1041
BRANCH ON INDICATOR			
CHANNEL 9	X		X
12	X		X
LAST CARD	X		X
UNEQUAL	X	X	X
EQUAL	X	X	X
LOW	0	X	X
HIGH	0	X	X
BRANCH WORD MARK			
OR ZONE	X		X
COMPARE	X	X	X
NO OPERATION	X	X	X
READ	X	X	X
WRITE	X	X	X
WRITE & READ	X		X
PUNCH	X	X	X
READ & PUNCH	X		X
WRITE & PUNCH	X		X
WRITE & READ & PUNCH	X		X
SELECT STACKER	X		X*
CONTROL CARRIAGE	X		X
STORAGE		1000	2000
CHAIN OF ADDRESSES	X		X
INDEX REGISTERS	0		X

LEGEND:
X=STANDARD
0=OPTIONAL
S=SUBROUTINE

Attachment Number 1

SUMMARY OF FEATURES

INSTRUCTIONS	1401	141	1041
ADD	X	X	X
SUBTRACT	X	X	X
MULTIPLY	O/S	S	S
DIVIDE	O/S	S	S
ZERO & ADD	X		X
ZERO & SUBTRACT	X		X
MOVE CHARACTER TO A OR B WORD MARK	X	X	X
MOVE CHARACTER TO ✱	O		X
MOVE AND SUPPRESS ZEROS	X	S	X
MOVE NUMERIC	X		X
MOVE ZONE	X		X
MOVE CHARACTER & EDIT	X	S	X
LOAD TO A-FIELD WORD MARK	X	X	X
SET WORD MARK	X	X	X
CLEAR WORD MARK	X	X	X
CLEAR STORAGE	X	X	X
STORE A-ADDRESS REGISTER	O		X
STORE B-ADDRESS REGISTER	O		X
BRANCH	X	X	X

SESSION NUMBER F.2.5

SPEAKERS
DAVE DYE (PID)
JOHN KEITH
JIM STANSBURY, CHAIRMAN
GAYE BABER

DISCUSSION

MR. DYE DESCRIBED THE OPERATION OF PID, POINTING OUT THE PROCEDURE FOR HANDLING ORDERS AND TELLING HOW THE USER CAN HELP IMPROVE THE SERVICES. JOHN KEITH DISCUSSED THE SHIPMENT ANALYSIS QUESTIONNAIRE, PURPOSE AND USE OF THE QUESTIONNAIRE, AND A SAMPLE COPY IS INCLUDED IN THE REFERENCE MANUAL. JIM STANSBURY DISCUSSED THE STANDARDS FOR THE 1130/1800/360 CONTRIBUTED PROGRAM LIBRARY. THESE STANDARDS WILL BE ISSUED ON THE FIRST UPDATE OF THE REFERENCE MANUAL.

Cincinnati - COMMON

Sponsor:

Administrative Division, Program Library Project

Subject:

IBM Program Information Department -- Program Library Procedures

Speaker:

D. R. Dye

Company Represented:

IBM

Speaker's Address:

IBM Corporation
Program Information Department
40 Saw Mill River Road
Hawthorne, New York 10532

914 -- 592-5790

Number of Pages:

4

A presentation concerning the IBM Program Information Department (PID) was made at Cincinnati COMMON. Slides were used in conjunction with the verbal presentation which was educational in nature. An open discussion period was included.

Although the presentation was formatted somewhat differently than the following narrative, it answered these same questions about PID that you (as a user) might ask.

WHAT IS THE IBM PROGRAM INFORMATION DEPARTMENT (PID)?

PID is the central control point for availability and distribution of IBM Programming Systems, IBM Application Programs and Contributed Programs. PID serves IBM customers in the United States from an extensive facility in Hawthorne, New York. Customers of the IBM World Trade Corporation are serviced from similar facilities in Paris, Rio de Janeiro, Toronto, Tokyo and Sydney.

PID's growth from its inception in the late 1950's has been dramatic. From an elementary exchange service for a few customer and IBM authored programs, PID has become a carefully organized operation which distributes hundreds of thousands of programs to users each year.

WHAT DOES PID MEAN TO ME AS A USER?

The availability of almost 3000 programs for IBM equipment means that users can often save considerable programming, procedural and systems design effort that would be required if they started from scratch. Often the economic payoff on an installation can be improved and the effective utilization can be increased and/or realized sooner.

PID can take a load off your back by serving as the distribution agency for your programs through the Contributed Program Library.

PID also means that, as an IBM customer, you have a dependable, accurate and timely source of IBM authored programs to support your system installation. When an improvement is incorporated or a bug is fixed, in an IBM program, users registered with PID automatically receive the modification or a notice of availability of the release.

HOW DO I KNOW WHAT PROGRAMS ARE AVAILABLE FROM PID?

Abstracts of all programs available from PID are published in the "IBM Catalogs of Programs". Copies are available through your local IBM office. In addition, IBM sales and systems engineering representatives receive automatically, timely fact sheets called Programming Announcements (P-Letters) which give information on the availability and features of IBM Programming Systems and IBM Application Programs. In addition, a Memo to Users is sent to each customer on the PID file announcing the availability of a new release.

Once you have ordered an IBM program and registered with PID as a user, you will automatically receive direct announcements of new releases of that program.

HOW DO I ORDER A PROGRAM FROM PID?

Normal program orders should be submitted using Program Order forms available from local IBM offices or special prepunched order cards supplied to you by PID. By a normal order we mean one that will be processed under PID's standard in-house processing cycle of up to ten working days. Transit time to and from PID is in addition to the ten working days in-house.

Orders that require faster handling should be submitted through your IBM representative after he has contacted PID by phone to make arrangements for special handling.

It is a good practice to consult your IBM representative before submitting an order. Certain information must be provided on the form and sometimes magnetic volumes must be submitted with the order form. Without all needed information and magnetic volumes, PID cannot complete the processing of your order. The IBM representative can help you in determining if the order submittal requirements have been met.

HOW IS THE QUALITY OF PID DISTRIBUTIONS ASSURED?

Extensive use is made of special hardware designed and built for PID. Also, sampling, chain copying, programmed comparisons and assigned personal accountability techniques are used.

The result is that the quality level of program materials distributed by PID is very high.

One particularly interesting piece of special purpose gear used for this quality control is the IBM 7299 Tape Copy and Compare System. This system automatically performs a bit for bit compare, in core, of written output against input and checks record counts and control labels. This assures the readability and completeness of users' magnetic tapes onto which programs have been copied.

If you receive a magnetic tape bearing an external label stating that the tape was "verified readable at PID" but you experience difficulty with it as input on your own system, we suggest that you try to read it on another drive and even another system, if possible. If a problem is still encountered, your CE should be notified so that he may determine the precise nature of the reading problem.

WHAT CAN I DO TO USE PID MORE EFFECTIVELY?

A few simple checks when submitting an order will insure that you get the program you need when you need it.

1. Be sure that the order form is complete. These items are needed by PID to control the accuracy of the order process:

   a. IBM customer number and branch office number (your local IBM representative will fill in this information for you if requested).
   b. Type of material required.
   c. Track mode and recording density.
   d. Full return address including ZIP code and "Attention of".

2. Be sure to submit a magnetic volume to PID if it is required for the program being ordered. The program abstract in the "IBM Catalogs of Programs" always indicates if a volume must be submitted. All magnetic volumes should carry external labels identifying the sender and the program being ordered.

3. Send the order form and the magnetic volume to PID as one package.

Plan ahead -- allow adequate time for shipping to PID, up to ten workdays at PID for processing and up to four days for return shipment. If faster service is needed, have your local IBM systems engineer or sales representative call PID.

Report packaging and distribution problems to your local IBM representative immediately if a replacement is needed. The Program Distribution Questionnaire, included with every shipment from PID, is a convenient means of reporting on the condition of program materials. Please use it.

SESSION NUMBER F.2.7

SPEAKERS
NONE - GENERAL DISCUSSION

DISCUSSION
C.O.S. - COMPATIBILITY OPERATING SYSTEM.
SOME 20 PERSON PARTICIPATED IN A GENERAL DISCUSSION OF C.O.S.
VARIOUS LEVELS OF C.O.S. EXPERIENCE LED TO A GOOD DISCUSSION.
LENGTH OF THE MEETING WAS ABOUT 1 HOUR 10 MINUTES.

SESSION NUMBER F.3.1
SPEAKERS

JAMES N. FISHER, THE BADGER CO., INC. ON STORAGE & RETRIEVAL OF PERMANET FILES FOR THE 1620/1311 MONITOR SYSTEM

Storage & Retrieval of Permanent Files for the 1620/1311 Monitor System

by

James N. Fisher

The Badger Co., Inc.

Cambridge, Mass.

466 THE BADGER COMPANY, Inc.

A method of permanently loading of information (either in the form of numeric data or Hollerith-type Format headings) has been found which fills a gap inherent in the IBM 1620/1311 Monitor System. This method consists of three steps in which: (1) a Fortran program is written to read the data and temporarily store it in Working Storage (this is done through the usage of the RECØRD statement), (2) a set of three cards designed to write a 100-digit indentification tag preceding the data files is then read into the computer, and finally (3) a *DLØAD operation is performed which is used to relocate the information from Working Storage into a permanent disk storage address.

The data may be used at any time by copying it into Working Storage and then calling for it whenever necessary. This is accomplished in two steps: (1) a *DCØPY operation used to copy the permanent data files into a working storage area and (2) a FETCH statement used to call in the data by a main program.

The program is set up for a Monitor system with Working Storage beginning at 00219. If the Working Storage area has been redefined this value must be altered accordingly.

STORING DATA FILES ON DISK

I. Loading Information into Working Storage

A Fortran program is written by the programmer to read the desired information into the Working Storage Area. A program to read in an alphameric set of headings is shown as follows:

```
≠≠JØB       5
≠≠FØRX
               DEFINE DISK (n1, n3)
               DIMENSIØN YH (n1, n2)
               DØ  2  N=1, n4
             2 YH  (N) = 0.0
               READ 1001, N1, N2
               READ 1002, (( YH (J, K), J=1, N1), K=1, N2)
               N=2
               RECØRD (N)  (( YH (J, K), J=1, N1), K=1, N2)
          1001 FØRMAT (2I4)
          1002 FØRMAT (n1A4)
               END
≠≠≠≠
```

Where: n_1 = number of individual items to be written per data record=N1

n_2 = number of data records to be stored = N2

$n_3 \geq n_2 + 1$

$n_4 \leq n_1 \cdot n_2$

This program, since it is to be used for alphameric headings, uses an A-Format in Statement number 1002 to read in data. For numeric information the programmer can use either E-, F-, or I-Format instead. Note that in the third instruction after Statement 2, N is set equal to 2. This is done in order to leave the first sector of working storage available for the 100-digit identification tag of Section II.

Also: The maximum number of sectors/data record = 2. This is determined by the value of f and k (See Service Manual, File No. 1620-36, Form C26-5739-3).

II. Writing Identification Tag

Next, the following three cards are loaded into the computer, with the Working Storage area defined as beginning at $\bar{0}0219$ in Card 1.

Cd. 1. $36001000050036001800050034000500070138000500070248 1\bar{0}0219\bar{0}01\bar{0}0100$

Cd. 2. $\bar{9}87897\bar{1}0\bar{0}0100\bar{1}0100\bar{1}2000\bar{0}8\bar{0}4\bar{0}0106\bar{\bar{5}}9998$

Cd. 3. Blank Card

Note: Some of the inputs in Cd. 2 are not necessary. However, it is strongly suggested that they be used exactly as presented in this paper.

III. Permanently Loading of Data

Finally, the data is loaded into a designated disk address. This is accomplished by means of the following set of cards:

```
##JØB    5
##DUP              Col. 21                          Col. 49
*DLØAD  name   1002191n5n5n5n5n51 n6n6n6n6n6         DI
####
```

$1002191n_5n_5n_5n_5n_51\ n_6n_6n_6n_6n_6$

Where name = name assigned to the data file by the programmer (6 characters or less).

n_5 = 00219 + n_7 - 1

n_6 = available sector address - able to contain n_7 data records. This may be determined from a DIP* Listing of the Monitor System.

n_7 = N_2 if there is one sector/data record; otherwise use $2(N_2)$ for two sectors/data record. See note for Section I.

Note: Both n_5 and n_6 are five-digit numbers which must be right-justified.

UTILIZING DATA FILES

I. Copying Data Into Working Storage

When the data files are desired, they must first be made available to a program by copying them into Working Storage. Copying of information from a permanent disk address into Working Storage is accomplished in the following manner:

##JØB   5   Col 21

##DUP

*DCØPY name $1n_6n_6n_6n_6n_6 1n_8n_8n_8n_8n_8 100219$

####

Where name = same as that in previous section.

* a DIP Listing consists of a map of all programs, their corresponding disk locations, plus all available sector locations. This may be obtained through the USER'S Group. The IBM Program Library

Number is 1.6.141.

$n_8 = n_6 + n_7 - 1$ and must also be a five-digit right-justified number.

Note: It is strongly suggested that the programmer carefully check the *DCØPY and *DLØAD cards. Errors in these cards have been known to render all disk-loaded programs completely useless.

II. Operations with Data File

The program reads in the data files by means of the FETCH statement. The record number is set equal to 2, as was indicated in Step I of the Storing Data Files on Disk Section. For a detailed explanation as to the proper usage of the FETCH and RECØRD statements, see page 106 of the IBM 1620 Monitor II System Reference Manual - File No. 1620-36, Form C26-5739-3. In order to fetch and utilize this stored information, the program must contain a DEFINE DISK statement, in which the value of n_1 must remain unchanged, while all n_2 and n_3 terms may now be less than or equal to those corresponding to Step I of the first section.

A sample segment of a program used to fetch the alphanumeric information of Step I, Section I and complement it with computed values is presented below:

```
N = 2
FETCH (N) ((YH (J,K), J=1, N1), K=1, N2)
```

```
        ........................................
        ........................................
        ........................................
        ........................................
        ........................................
        ........................................
        PUNCH 1003, ((YH (J,K), J=1, N1), (ZØ (N,K), N=1, 3), K=1, N2)
1003    FØRMAT (n1 A4, F10.0, 2F13.4)
```

Where: ZØ (N,K) = the computed values used for output by the program.

The following output was obtained from such a sample segment:

```
DIAMETER,                   FEET      1.0000    2.0000    3.0000
TRAY RING,                  INCHES    4.0000    5.0000    6.0000
DOWNCOMER WIDTH,            INCHES    7.0000    8.0000    9.0000
PATH LENGTH,                INCHES   10.0000   11.0000   12.0000
DC-TRAY CLEARANCE,          INCHES   13.0000   14.0000   15.0000
INLET WEIR HEIGHT,          INCHES   16.0000   17.0000   18.0000
OUTLET WEIR HEIGHT,         INCHES   19.0000   20.0000   21.0000
HOLE DIAMETER,              INCHES   22.0000   23.0000   24.0000
HOLE PITCH, TRIANGULAR,     INCHES   25.0000   26.0000   27.0000
```

At the moment, permanently-loaded information may only be used by a disk-loaded program. This may be improved upon in the future.

The advantage of permanently storing data files on the disk is that a programmer now may be able to limit the number of FØRMAT statements appearing in his program. Further, independent programs may now be allowed to use the same records in places, thus further reducing the amount of storage required for some programs. Finally, long chains of alphameric headings complemented by output data may now be permanently stored as data files. Caution, however, should be used in experimenting with the Storage-System, for as mentioned in Step I, Section II, some errors may result in the destruction of disk-loaded programs.

James N. Fisher

J. N. Fisher

SESSION NUMBER F.3.4

SPEAKERS
    NO SCHEDULED SPEAKERS.

DISCUSSION
    PLANS WERE MADE FOR THE 360 AGENDA AT THE SAN FRANCISCO MEETING. THE GOOD AND BAD RESULTS OF THE SESSIONS AT THIS MEETING WERE EVALUATED, AND WE DECIDED WHAT PLANNED PRESENTATIONS WERE DESIRED AT THE NEXT MEETING.

SESSION NUMBER F.3.6

SPEAKERS

GEORGE J. REYNOLDS, GENERAL MOTORS INSTITUTE ON COMPUTER
TIME STUDY ANALYSIS FOR WORK MEASUREMENT

COMPUTER TIME STUDY ANALYSIS FOR WORK MEASUREMENT

George J. Reynolds
Assistant Professor of Industrial Engineering
General Motors Institute
Flint, Michigan

ABSTRACT

This article describes the format for the observation, recording and calculations of work standards. In addition, a computer program for the IBM 1620 Model II 40K is included which provides a more efficient calculation time for each time study. This in turn reduces cost per work standard.

COMPUTER TIME STUDY ANALYSIS FOR WORK MEASUREMENT

George J. Reynolds
Ronald W. Cox

The determination of a fair day's work has progressed from haphazard methods to rather scientific methods. One scientific approach to this is called "Time Study." Time study can be explained as a technique for the measurement of the time factor in the utilization of men, materials, tools, and equipment. In most industrial organizations, time study is one of the common techniques used for setting standards which are used as a basis for determining a fair day's work. In many organizations, the department doing time study work is called Industrial Engineering.

Taking a time study consists of breaking down the operation into parts and recording the time taken for each of these parts by means of a stop watch. It also consists of recording job data, and setting the standard for the job. This standard is expressed in time per unit and is considered representative of a fair day's work. A time study primarily records the cycle time of the operation.

In properly recording a time study there are certain data which must be clearly set down so that anyone referring to the study at a later date may know all the facts surrounding it. Some such items are obvious in their purposes - as for example, the part name. Others are more in the nature of manufacturing data.

The following is a list of those items which are most frequently found on time studies:

Date of Study. This establishes the day, month and year in which the study was made. Under certain conditions it might be necessary to indicate the hour or the shift.

Effective Date. This establishes the day, month and year on which the study became effective. It is highly important that this date tie into the routing date.

Material. This item should adequately describe the material used. The kind, the specification number, the dimensions and any other facts required for identification must be set down. This is quite important since the study is limited to the specified material, and any change in the material may automatically call for a new study.

Part Name. Carry here the full part name as it is carried on the blueprint.

Part Number. Accurately record the part number as it appears on the print.

Operator Number and Name. Aside from identifying the man upon whom the study was madethe very act of learning the workman's name - if done in a tactful and friendly way - can help the time study analyst in selling himself as well as letting him know the man. This is important in establishing a better attitude toward the study on the part of the operator.

Department Number. To show where the study was made.

Machine Number and Name. This item should indicate by name, model, and inventory number the specific machine studied to identify it since no two machines, even though they be of the same make and model, perform identically.

Cutting Compound. Show the kinds of cutting compound used such as oil, soap, etc., and wherever available refer to the specification number. The compound frequently has a very direct effect upon the cutting time and the correct recording of this item is an essential.

Operation Number and Name. A code to identify the operation and to tie it up with the routing.

Speeds and Feeds. A machine time study that does not carry complete and accurate speed and feed data has little or no value since it is such data that tells us how effectively the machine was performing during the study.

Tools and Tool Number. Here identify whatever die, tool or fixtures are used in the study by number. Where necessary a brief description may be used as for example: Carboloy tips, special diamond dresser, abrasive wheel specification, etc. Like material data, speeds and feeds, and cutting compound, this identification of tools and tool number definitely limits and clarifies the study and a careful and accurate recording is important since any change would qualify the operation for a recheck.

Production Per Hour. When you have completed the study through the development of the standard, calculate the number of pieces by dividing 1.0 hour by the standard time per unit.

Male or Female. Indicate here the sex of the operator. This information may be called for in labor discussions relative to whether a job is correctly classified as male or female.

Allowances. This space provides for entering certain allowances for items not covered in the elemental detail of the study.

Total Time of the Study. Here is recorded the starting and stopping time of the study, or the total overall time of the study. This data may be used as a check against the accuracy of the study.

Name of the Observer. The time study analyst's name should be signed here.

Reason for Study. This space is set aside for such general remarks as the time study analyst may wish to make, for example, "Rechecked Standard."

Approvals. Here is recorded the foreman's signature and all the necessary approvals required by management.

Work Place. A sketch to work place, piece part showing operation or area worked on, etc.

The more completely and accurately we record on our studies all the data relative to the job, the more valuable our studies become, both to us and to those functional areas which will call on us every day for information. As time passes, we very quickly forget the details surrounding the taking of a study and when it becomes necessary to discuss it, a month or year or two years later, we very often cannot recall the whole picture to mind. Therefore, it is for this reason that the true value of full and correct job data is necessary.

The actual mechanics of determining the standard time are not complex, but must be thoroughly understood and correctly done in order to arrive at a basis for getting an accurate work standard. The generally accepted method is:

1. Obtain time data from observation by gathering stop watch readings, element frequency, foreign elements, performance, and allowances.
2. Calculate leveled average time for each element.
3. Determine allowed time per piece.
4. Calculate standard time.

After completing the recording of stop watch readings, earmarking irregular elemental times, and recording elemental frequencies, then individual representative time values must be determined for each element. These times for each element come from the recorded snapback or continuous stop watch readings. This procedure can be explained by means of a sample continuous recorded study (Figure 1, page 11). The first recorded time from the starting of the watch until the end of the first element is .07 minutes; so this figure would be recorded opposite (I) in element #1. Then the time for element #2 can be obtained by subtracting the continuous reading of element #1 from that of element #2. Thus, .38 - .07 = .31 and this figure is recorded opposite (I) in element #2 and so on, until all times are obtained. These times, then, will reflect the standardized method and conditions under which the operation is to be performed and should be averaged in order to determine the leveled average time for each element.

This leveled average should be calculated by dividing the total of all time intervals, which are not earmarked, by the number of such intervals. For example, the completed study of the first element in Figure 1, page 11, covers a total number of 10 observations for the cycles. With one of these recorded times earmarked, the leveled average time should be determined by dividing by 9 the total time consumed by the remaining 9 time intervals. In this manner, the time for this regular element will reflect the basic established method for the operator in the performance of this particular part of the operation.

Earmarked irregularities are called foreign elements. Where the irregularity is timed and recorded with a regular element reading, the foreign element can be calculated by subtracting the leveled average time from the recorded watch reading.

After this time value has been calculated, the prorating can be done the same as if the irregularity had been timed and recorded separately. These calculations with respective frequencies should be plainly shown in the provided space on the time study form. In addition to foreign elements, the details of delays for allowances should be filled in. These will tend to vary depending on company policy.

If company policy involves performance rating, the next step is to determine the operator performance by comparing the operator to the average normal operator concept. The objective of this rating is to establish an allowed time which is representative of the average normal worker. This rating should be expressed as a percentage above or below 100% which represents the average normal operator.

Next is to determine the allowed time per piece for each element of the study. This is done by multiplying the leveled average time by the performance and the elemental frequency for each element.

Finally, this method of calculation involves the summation of all allowed times and all detail of delay allowances. After this is completed, subtract the total time for allowances on a per shift basis from the total shift time to determine the minutes available for work. Next, the minutes available for work time is divided by the summation of the allowed times plus delays on a per piece basis. This yields standard pieces per shift. Divide this number by eight hours which gives standard pieces per hour. Divide the standard pieces per hour into one hour and the final work standard is expressed in hours per piece.

All of the calculations for leveled average time, allowed time per piece, foreign elements, detail of delays and the setting of the final work standard can be accomplished by the IBM 1620 Model II, 40k computer. In order for the computer to accomplish this, the time study analyst must prepare his studies in a language the computer understands. Therefore, it is required that all data be in decimals whereby the key punch operator can take this data directly from the time study observation sheet (Figure 1). The IBM cards should be prepared in the following manner:

FIRST READ STATEMENT (.5F7.0)

a. Part Number

b. Operation Number

c. Type of Study Code

11 = "A" Continuous

12 = "A" Snap Back

13 = "B" Continuous

14 = "B" Snap Back

15 = "C" Continuous

15 = "C" Snap Back

d. Number of elements in the study.

e. Number of columns of element readings.

SECOND READ STATEMENT (10F8.2)

Recording of Element Reading

a. Keypunch individual stop watch time values in vertical order regardless of the type of study.

b. For any missing time values or those disregarded because of some assignable cause code as "zero."

c. In the case of a continuous type of study where a value has been coded as "zero" add 1000 to next acceptable reading.

d. If any of the readings are earmarked as a foreign element code as follows:

Type	"A"	add	900	to	the	value
"	"B"	"	800	"	"	"
"	"C"	"	700	"	"	"
"	"D"	"	600	"	"	"
"	"E"	"	500	"	"	"

Note: Add zero if necessary for a total of five items.

THIRD READ STATEMENT (10F8.2)

Element Performance Evaluation and Element Frequency

The performance and element frequency must be key punched consecutively for each element respectively.

FOURTH READ STATEMENT (15F5.2)

a. Foreign element frequency

b. Detail of delay allowance

   1. Minutes/shift items

   2. Minutes/piece items

The frequency and detail of delay allowance items are key punched consecutively, i.e., frequency, minutes/shift item, minutes/piece item as Line 1 horizontally, etc. If any categories do not have five items in the particular study, key punch zero as the missing items until a total of five items each have been listed.

FIFTH READ STATEMENT (7A4)

a. Name card.

b. Continuous run - if more than one time study is run at the same time, a 1 is key punched in column 30 to separate the studies.

For example, with the data from the time study shown in Figure 1, page 11 as input to the IBM 1620, the computer would give the following output for all the necessary calculations involved in stop watch time study analysis.

```
                    TIME STUDY OBSERVATION OUTPUT

NAME            RON COX
PART NUMBER       88128
OPER NUMBER       12352.
TYPE OF STUDY        11.
                                          ELEMENT 1
LEVELED AVERAGE                           .0733
MINIMUM PIECE ALLOWED TIME                .0366

                                          ELEMENT 2
LEVELED AVERAGE                           .3320
MINIMUM PIECE ALLOWED TIME                .1411

                                          ELEMENT 3
LEVELED AVERAGE                           .7270
MINIMUM PIECE ALLOWED TIME                .3453

                                          ELEMENT 4
LEVELED AVERAGE                           .1300
MINIMUM PIECE ALLOWED TIME                .1300

                                          ELEMENT 5
LEVELED AVERAGE                           .5114
MINIMUM PIECE ALLOWED TIME                .4347

                                          ELEMENT 6
LEVELED AVERAGE                           .0430
MINIMUM PIECE ALLOWED TIME                .0204

                                          ELEMENT 7
LEVELED AVERAGE                           .2566
MINIMUM PIECE ALLOWED TIME                .1219

                                          ELEMENT 8
LEVELED AVERAGE                          1.5260
MINIMUM PIECE ALLOWED TIME                .2746

                                          DELAY MIN/PIECE

A =       .0106
B =       .0258
C =       .0282
D =      0.0000
ALLOWED CYCLE TIME/PIECE                 1.5048
ATTAINABLE HOURLY PRODUCTION            29.871
TOTAL DELAY (MIN/SHIFT)                   34.
TOTAL DELAY (MIN/PIECE)                  .064
MIN AVAILABLE FOR WORK                   446.
STND PIECES/SHIFT                      284.155
STND PIECES/HOUR                        35.519
STND HOUR/PIECES                         .0281
```

11

NO.	ELEMENT DESCRIPTION LEFT SIDE	RIGHT SIDE	ELE. NO.	1	2	3	4	5	6	7	8	9	10	AVER.	LEVEL-ED AVER.	PER-FORM-ANCE EVAL.	FREQ.	MINS/PC ALLOWED TIME
	Element #1		I				A											
			C	.07	3.57	7.17	11.25	14.82	18.75	22.38	26.03	29.68	33.69			100	1/2	
	Element #2		I															
			C	.38	3.92	7.52	11.57	15.13	19.07	22.72	26.38	30.02	34.02			85	1/2	
	Element #3		I															
			C	1.10	4.62	8.26	12.30	15.88	19.82	23.44	27.09	30.75	34.74			95	1/2	
	Element #4		I															
			C	1.22	4.76	8.41	12.41	16.00	19.97	23.58	27.22	X	X			100	1/1	
	Element #5		I					B										
			C	1.71	5.29	8.88	12.93	16.88	20.47	24.10	27.75	31.77	35.54			85	1/1	487
	Element #6		I															
			C	1.74	5.34	8.91	12.96	16.93	20.53	24.16	27.80	31.81	35.57			95	1/2	
	Element #7		I			C												
			C	2.01	5.58	9.52	13.22	17.19	20.79	24.41	28.05	32.07	35.83			95	1/2	
	Element #8		I															
			C	3.51	7.09	11.07	14.76	18.68	22.31	25.94	29.60	33.61	37.36			90	1/5	
			I															
			C															
			I															
			C															

CODE		FOREIGN ELEMENTS
A	10	Fumble
B	15	Stuck Part
C	12	Drop Part
	—	
	—	

DETAIL OF DELAYS: MIN/SHIFT		MIN/PC	
PERSONAL	24		
P & CU	10		
TOTAL		TOTAL	

ATTAINABLE HOURLY PRODUCTION = $\frac{\text{60 MINUTES}}{\text{(ALLOWED CYCLE TIME)}}$ OR ....

ALLOWED CYCLE TIME/PC =

FINAL CALCULATION OF STANDARD:

480 MIN. − (ALLOWED DELAY / MIN SHIFT) .... = (MIN AVAILABLE FOR WORK) .... ÷ (ALLOWED CYCLE TIME + DELAYS IN MIN/PC) .... = (STD PCS /SHIFT) .... ÷ 8 = (STD PCS/HR) ....

STANDARD HOURS/PIECE ....

Figure 1

IE-7

SESSION NUMBER F.3.7

SPEAKERS
NONE - SQUAWK SESSION

DISCUSSION
PREPARED A LIST OF RECOMMENDATIONS FOR IBM. DICK PHILLIPS, OF IBM, WAS WITH US.

SPECIAL ADDED ATTRACTIONS

IBM 1130 SYSTEM

IBM QUICKTRAN DEMONSTRATION

IBM QUIKTRAN DEMONSTRATION
Parlor J, 4th Floor, Netherland Hilton Hotel

QUIKTRAN is a time-sharing data processing system that brings the power of a modern large-scale computer directly to the desk of the engineer and scientist, the mathematician, the business and financial planner - anyone who solves problems with numbers.

The QUIKTRAN system uses the FORTRAN language and has available a library of widely used programs.

To show you the QUIKTRAN system in operation, we have arranged the following <u>DEMONSTRATIONS:</u>

1) Common library programs including:
   a) ROOT, a program which finds the nth root of m by Newton's iteration method.
   b) QUAD, which finds the real and imaginery roots of the equation ax^2+bx+c.
2) Programs written especially for the COMMON meeting designed to demonstrate the conversationality of QUIKTRAN.

Demonstrations will be given:

Tuesday, September 5	6:00 p.m. - 9:00 p.m.
Wednesday, September 6	9:00 a.m. - 9:00 p.m.
Thursday, September 7	9:00 a.m. - 6:00 p.m.
Friday, September 8	9:00 a.m. - 5:00 p.m.

IBM 1130 SYSTEM

An IBM 1130 has been provided for the COMMON Meeting. It will be available from Wednesday (Sept. 6) through Friday (Sept. 8), to all members for testing or demonstration purposes.

The 1130 consists of a 1442 Card Read/Punch, an 1132 Printer, and 8K disk. Two key punches, disk packs, and a System Reference Library are also available for your use. The following programs have been loaded on disk packs:

Monitor (Mod. level 4)
Commercial Subroutine Package
Scientific Subroutine Package
Statistical System
Work Measurement Aids
Structural Engineering System Solver (STRESS)
Mechanical Design System
Numerical Surface Techniques
Critical Path Method/PERT
Other Demo Packages & Utilities

The equipment is located in Parlor I of the Netherland Hilton. You may sign up to reserve time for testing or demonstration. Please feel free to stop by any time; there will always be someone there to assist you.

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

ABNER, J R
5904 SEWELL RD
PENSACOLA, FLA 32504

ALLBRITTON, E J
&3 BOX 191
CLARDSVILLE, MO 63336

ALLEN, J E
1206 MULBERRY ST
DES MOINES, IOWA 50309

ALVAREZ, J J
IBM GLENDALE LAB DEPT 265
ENDICOTT, N.Y. 13760

ANDERSON, B K
20 SOUTH ROAD
SOUTHINGTON, CONN

ARMBRUSTER, L F

MONTGOMERY, W VA 25136

ARNTSON, W M
1700 W. THIRD AVE.
FLINT, MICH 48502

ARTHUR, A J
3270 CABRILLO AVE
SANTA, CALIF 95051

ATER, G T
35 PAGE RD
CHILLICOTHE, OHIO 45601

AUSTIN, L B
1700 W. THIRD AVE

FLINT, MICH 48502

BABER, G M
3000 SPOUT RUN PARKWAY
ARLINGTON, VA 22201

BAILIE, D L
455 C STREET
WASHOUGAL, WASH 98671

BAILEY, D C
MONTEREY AND COTTLE ROAD
SAN JOSE, CALIF

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

BAKER, L H
1206 MULBERRY ST
DES MOINES, IOWA 50308

BALL, M J
159 IDLE HOUR DRIVE
LEXINGTON, KY 40502

BALLENTINE, J D
547 PAIGIE STREET
SCHENECTADY, N.Y. 12307

BARMORE, D R
4780 SNOW DRIVE
SAN JOSE, CALIF 95111

BARNEY, P L
740 S. ALABAMA
INDIANAPOLIS, IND 46206

BARR, S
222 BROADWEL
NEW YORK, N.Y. 10038

BENEDICT, D E
2000 FORRER BLVD
DAYTON, OHIO 45401

BERGER, D E
2605 REYNOLDS CR.
HUNTSVILLE, ALA 35810

BERNARD, R J
606 JODI DR
HAMMOND, LA 70401

BERWICK, J M
MAIN
HAWKESBURY, ONT CANADA

BICKFORD, P A

GREENCASTLE, IND 46135

BLACKNEY, W C
TS&D, ARC, 2020 BLDG DOW CHM
MIDLAND, MICH

BLATCHLEY, C G
300 BENT RD
WYNCOTE, PA 19095

493

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
BLISS, R J
      5830 WESTHENRIETTA RD
      HENRIETTA, N.Y.          14606

BOBAY, J P
      1000 5TH ST
      COLUMBUS, IND            47201

BOND, W F
      656 STORM AVE
      BROOKHAVEN, MISS         39601

BOSCHAN, C
      144 EAST 24 ST
      NEW YORK, N.Y.           10010

BRADY, J J
      4620 FOREST AVE
      NORWOOD, OHIO            45212

BRANAGAN, R E
      2442 TRENTON AVENUE
      MONTREAL,                QUEBEC

BRASKAMP, B
      1111 CONNECTICUT AVE N.W.
      WASHINGTON, D.C.         20036

BRECHBILL, D O
      625 CLEVELAND
      COLUMBUS, OHIO           43215

BRENNER, R L
      P.O. BOX 561
      BURLINGTON, IOWA         52601

BRENNAN, R D
      2670 HANOVER
      ALTO, CALIF

BRIGGS, D R
      4306 BEDFORD
      MIDLAND, TEX             79701

BROWN, L W
      P.O. BOX 2328
      MOBILE, ALA              36601

BUCKLEU, R E
      1414 BEECH ST. S.E.
      DECATUR, ALA             35601
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

BUESKING, C W
4617 TAYLOR AVE
EVANSVILLE, IND 47715

BUFE, O E
2300 CHESTER AVE
CLEVELAND, OHIO 44114

BURGESON, J W
220 EAST UNION ST
WHEATON, ILL 60187

BURGGRABE, W F
1400 SOUTH THIRD ST
ST. LOUIS, MO 63166

BURNS, R A
ALGOMA STEEL CORP LTD
SAULT, ONT RIE CANADA

BURROWS, W A

PITTSBURGH, PA 15225

BYTHER, T E
31 OAK ST
OLD TOWN, ME 04468

CAMPBELL, M
2240 SOUTH LONE PINE
SPRINGFIELD, MO 65804

CAPLAN, F L
215 ROSE HILL AVE
NEW ROCHELLE, N.Y. 10804

CARLSON, D R
5707 LINDENWOOD LANE
FAIRFIELD, OHIO

CARLSON, D M
327 S. FOURTH AVE
ANN ARBOR, MICH 48108

CASTELLAN, N J
DEPT OF PSYCHOLOGY I U
BLOOMINGTON, IND 47401

CEELY, F F
6206 BREN MAR DRIVE
ALEXANDRIA, VA 22312

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
CHAIKIN, A J
     9300 GEORGE PALMER HWY
     LANHAM, MD                20034

CIPRIETTI, B J
     34 IPSWICH PLACE
     HAMILTON, ONT             CANADA

CLARKE, J R
     449 W. 5TH ST
     CHILLICOTHE, OHIO         45601

CLARK, L A
     3520 W. ROUNTREE
     SPRINGFIELD, MO           65804

CLARK, W H
     4 EDGEBROOK ROAD
     BINGHAMTON, N.Y.          13903

CLARK, A G
     132 DAVIS STREET
     PAINTED, N.Y.             14870

CLEGG, J B
     346 COMPTON RD
     CINCINNATI, OHIO          45215

CLOSMAN, S
     112 EAST POST ROAD
     WHITE, N.Y.

COLE, C T
     106 KATAHDIN DR
     POLAND, OHIO              44514

CONROD, R L
     P.O. BOX 208
     BEDFORD, MASS             01730

COOPER, T
     USDAARSBSSARCBLDG226
     BELTSVILLE, MD            20705

CORDING, B L
     2705 DELLWOOD DR
     ORLANDO, FLA              32806

CORNELL, R L
     4137 BEARD AVENUE SOUTH
     MINNEAPOLIS, MINN         55410
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

COTTON, B W P.O. DRAWER G GRAPEVINE, TEX	76051
COX , R C 1429 KEED AVENUE BATON, LA	70806
CRAFT, J C 2109 NORRIS RD N.W. HUNTSVILLE, ALA	35810
CRUMB, H F 33 LIBERTY ST NEW YORK, N.Y.	10045
CUMMINGS, L R FEDERALSBURG, MD	21632
CUNNINGHAM, A J 25 MAIN ST ANSONIA, CONN	06401
DAGENFIELD, R L 1214 JAEGER ST COLUMBUS, OHIO	43206
DANZEISEN, L A WOODVILLE ROAD TOLEDO, OHIO	43601
DEAKIN, G R 302 MOUNTAIN DRIVE PEARISBURG, VA	24134
DECK, J C 661 CHESTNUT STREET VALPARAISO, IND.	46383
DEGENNARO, M G 175 OLD COUNTRY RD HICKSVILLE, N.Y.	11801
DELONG, W 629 SIBLEY STREET HAMMOND, IND	46320
DENING, J W 3451 MCHENRY AVENUE CINCINNATI, OHIO	45225

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
DEUTSCH, E
     COMPUTER CENTRE
     GENESEO, N.Y.            14454

DEWEY, G C
     11444 LACKLAND ROAD
     ST. LOUIS, MO            63141

DICOSTANZO, J A
     P.O. BOX 390
     POUGHKEEPSIE, N.Y.       12602

DONALDSON, D L
     1570 WOODMAN DR 15
     DAYTON, OHIO             45432

DONEGAN, A W
     107 SMITHFIELD DRIVE
     ENDICOTT, N.Y.           13760

DONNELLY, M J
     906 BELGIAN AVENUE
     BALTIMORE, MD            21218

DOUCETTE, J E
     109 CHANDLER STREET
     BOSTON, MASS             02116

DOWD, C K
     3711 LOCHEARN DRIVE
     BALTIMORE, MD            21207

DRAY, R D
     47 PAGE ROAD
     CHILLICOTHE, OHIO        45601

DUNSMORE, D A
     414 WALNUT STREET
     CINCINNATI, OHIO         45202

DUQUETTE, D J
     THIRD AVE & FORDHAM ROAD
     BRONZ, N.Y.              10458

DWYER, J R
     9321 E. 84TH TERR
     RAYTOWN, MO              64138

DYE . D R
     40 SAWMILL RIVER ROAD
     HAWTHORNE, N.Y.
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

EATON, T J
400 WASHINGTON AVENUE
ST. LOUIS, MO 63102

EDWARDS, R A
112 E. POST ROAD
WHITE, N.Y. 10601

ELWELL, W G
7030 STARR
LINCOLN, NEBR 68505

ENYEDY, G
AUBURN ROAD
PAINESVILLE, OHIO 44077

FAERBER, R B
1266 AVALON DRIVE
SAN JOSE, CAL 95125

FALCONELLO, P
THIRD AVE & FORDHAM ROAD
BRONZ, N.Y. 10458

FANUELE, V L
12 WILDWOOD DRIVE
WAPPINGERS, N.Y. 12590

FELICE, L
1341 BALCOM AVENUE
NEW YORK, N.Y. 10461

FELLER, G G
1402 10TH AVENUE S.E.
ROCHESTER, MINN 55901

FINCH, D G
3316 CROSS COUNTRY DRIVE
WILMINGTON, DEL 19803

FISHER, J N
363 THIRD STREET
CAMBRIDGE, MASS 02148

FITZPATRICK, E D
ILLINOIS STATE UNIVERSITY
NORMAL, ILL 61761

FLEMING, C O
P.O. BOX 34380
DALLAS, TEX 75234

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

FODOR, J E		
	ENGINEERING BUILDING	
	MADISON, WIS	
FOERSTER, C S		
	112 VALLECITOS WAY	
	LOS CATOS, CAL	95030
FOLLAND, G H		
	MILFORD, MICH	48042
FOLTZ, T V		
	4818 AVONDALE DRIVE	
	FT. WAYNE, IND	46806
FORSTROM, R W		
	19401 DORAL COURT	
	YORBA, CAL	
FORTUNE, F C		
	19394 GULFSTREAM DRIVE	
	TEQUESTA, FLA	
FOWLER, W G		
	3730 MEADOWBROOK DRIVE	
	ZANESVILLE, OHIO	43701
FRASER, W C		
	36 DOBIE	
	MONTREAL, QUE	
FRASER, W I		
	21 ETON COURT	
	CAMLACHIE, ONT	CANADA
FULLAN, D J		
	112 E. POST ROAD	
	WHITE, N.Y.	
GABBERT, D A		
	2117 INDIANA STREET	
	PARKERSBURG, W. VA	
GABRIEL, R F		
	SOUTH ORANGE AVENUE	
	SOUTH, N.J.	07079
GANATRA, J K		
	458 FOX HILL DRIVE	
	BLOOMFIELD, MICH	48013

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

GARDNER, D S
211 WALNUT STREET
RIDGEWOOD, N.J. 07450

GARGANO, H M
1901 CHAPMAN AVENUE
ROCKVILLE, MD 20852

GEIGER, A J
2186 MIDDLEHURST DRIVE
COLUMBUS, OHIO 43201

GENTILE, J F
AUBURN ROAD
PAINESVILLE, OHIO 44077

GERMANN,
SOUTH ORANGE AVENUE
SOUTH, N.J. 07079

GIBSON, S
3117 MILFORD AVE.
BALTIMORE, MD 21207

GILMARTIN, W R
126 STRATFORD DRIVE
IRWIN, PA 15642

GINGERICH, D F
5011 W. 26TH STREET
TOPEKA, KANS 66614

GLENN, J S
40 STARK STREET
NASHUA, N.H. 03060

GLOSTER, A S
P.O. BOX 117
OAK RIDGE, TENN 37830

GOESCH, G W
17304 ZINA AVENUE
LOS GATOS, CAL 95030

GOLDMAN, N
111 CUMMINGTON STREET
BOSTON, MASS 02215

OLDBERG, M
THIRD AVE & FORDHAM ROAD
BRONZ, N.Y. 10458

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
GOLIER, R T
     TWO GATEWAY CENTER
     PITTSBURGH, PA           15222

GOODRUM, D L
     RR 1 BOX 460
     WATEVLIET, MICH

GOSSETT, E C
     37 ALLEN
     ALLENDALE, N.J.          04701

GRAY, W C
     11444 LACKLAND ROAD
     ST. LOUIS, MO            63141

GREEN, J
     JAMAICAWAY TOWER 151
     BOSTON, MASS             02130

GREEN, D M
     7991 COLONY DRIVE
     ALGONAC, MICH

GREEN, H A
     194 CECIL STREET
     SARNIA, ONT              CANADA

GRIBBEN, C J
     25 MONUMENT CIRCLE
     INDIANAPOLIS, IND        46200

GROFT, G E
     COUNTRY CLUB MANOR APT J2
     YORK, OA                 17403

GWILLIAM, J C
     16 ELM STREET
     NORWICH, N.Y.            13815

HAGUE, M T
     31 KENT
     SCARSDALE, N.Y.

HAMANT, W E
     8302 MAYFAIR
     CINCINNATI, OHIO         45216

HAMILTON, T K
     315 GRAHAM AVENUE
     COLUMBUS, OHIO           43203
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
HAMPEL, C R
     222 E. CENTRAL PKWY
     CINCINNATI, OHIO        45202

HARBRON, T R
     ANDERSON COLLEGE
     ANDERSON, IND           46011

HART, T M
     23 GARRAHAN
     WILKES-BARRE, PA        18702

HAYWARD, A P
     435 6TH AVENUE
     PITTSBURGH, PA          15219

HEETDERKS, J W
     3001 MILLER P.O.BOX 218
     DEARBORN, MICH

HERTEL, E S
     29 DUNN AVENUE
     MANGATUCK, CONN         06770

HERWITZ, P S
     LAKEVIEW AVENUE W.
     PEEKSKILL, N.Y.         10566

HICKMAN, G A
     222 E. CENTRAL PKWY
     CINCINNATI, OHIO        45202

HILLIER, W J
     1616 WALNUT STREET
     PHILADELPHIA, PA        19103

HILL, W H
     6715 BLVD EAST
     GUTTENBERG, N.J.        07093

HOFFMAN, B A
     P.O. BOX 3621
     PORTLAND, ORE           97208

HOFFERT, E R
     1705 URBANA ROAD
     CLEVELAND, OHIO         44112

HOFFMAN, L L
     GUGGENHEIM LABS
     PRINCETON, N.J.         08240
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
HOKE, W E
     104 SMITHFIELD DRIVE
     ENDICOTT, N.Y.

HORTON, M H
     4800 OAK GROVE DRIVE
     PASADENA, CAL                91103

HUGH, G M
     53 HOLMESDALE CRES
     TORONTO, ONT                 CANADA

HUMPHREY, M E
     803 PAINTER AVENUE
     NATRONA, PA   S              15065

HUSSEIN, H A
     415 E. 64TH STREET
     NEW YORK, N.Y.               10021

HYMAN, I M
     LUKENS STEEL COMPANY
     COATESVILLE, PA              19320

IBSEN, J K
     5985 N.W. 62ND AVENUE
     DES MOINES, IOWA             50324

IMBERTSON, J R
     2189 DOSWELL AVENUE
     ST. PAUL, MINN               55108

IMLAY, C E
     1045 RICHEY ROAD
     ZANESVILLE, OHIO             43701

INGRAM, B C
     1018 ROSETREE LANE
     CINCINNATI, OHIO             45243

JAEGER, R B
     RD 1
     EMPORIUM, PA                 15834

JAGTIANI, H J
     P.O. BOX 2-AC
     RICHMOND, VA                 23205

JEFFUS, S E
     5626 MAYNARD
     FORT, TEX                    79906
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
JENSEN, E L
     4660 SO. 60TH AVENUE
     OMAHA, NEBR                 68117

JINKS, W V
     22 E. 7TH STREET
     CHILLICOTHE, OHIO           45601

JOHNSON, J D
     6023 PENN AVENUE SO.
     MINNEAPOLIS, MINN           55419

JOHNSON, B M
     6126 THOLE ROAD
     CINCINNATI, OHIO            45230

JOHNSON, D R
     227 W. 10TH
     HINSDALE, ILL               60521

JOHNSTON, W H
     25 MAIN STREET
     ANSONIA, CONN               06401

JONAS, C R
     2667 N. UNION ROAD
     MIDLAND, MICH               48640

JONES, J L
     3214 RADIANCE ROAD
     LOUISVILLE, KY              40220

JONES, L W
     P.O. BOX 3621
     PORTLAND, ORE               97208

JONES, R L
     1514 WOODCLIFFE AVENUE
     BALTIMORE, MD               21228

JONES, H V
     524 1/2 S. UNIVERSITY BLVD
     NORMAN, OKLA                73069

KEESLING, M A
     5721 SOUTH KIMBARK
     CHICAGO, ILL                60637

KEESLING, J W
     5721 SOUTH KIMBARK
     CHICAGO, ILL                60637
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
KEITH, J H
     10820 S.W. 52 DRIVE
     MIAMI, FLA

KELLEY, J R
     618 SOUTH MICHIGAN BLVD
     CHICAGO, ILL             60605

KEMP, E V
     4820 URBANA ROAD
     SPRINGFIELD, OHIO        45502

KENNY, A D
     401 EAST 74TH STREET
     NEW YORK, N.Y.           10021

KERR, H B
     BOX 21A TTV
     COOKEVILLE, TENN         38501

KERR, M C
     BOX 21A TTV
     COOKEVILLE, TENN         38501

KINDRED, A R
     2707 RUTGERS AVENUE
     BRADENTON, FLA           33505

KISSNER, J R
     55 E. WASHINGTON
     HAGERSTOWN, MD           21740

KLEIN, D R
     145 LOWRYS LANE
     ROSEMONT, PA             19010

KOEPSELL, P L
     SOUTH DAK STATE UNIV.
     BROOKINGS, S.D.          57006

KOERING, L O
     3725 FRAZIER ROAD
     ENDWELL, N.Y.

KOLLER, E B
     570 ST. JOHNS ROAD
     POINTE, QUE              CANADA

KRAMER, P H
     901 EVERNIA STREET
     WEST, FLA  EACH          33401
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
KROENCKE, T H
     29-01 BORDEN AVENUE
     LONG, N.Y.                11101

KUHN, R E
     RD 2 BOX 94-C
     JERSEY, PA                17740

LACHNIET, W M
     HAMILTON STANDARD
     WINDSOR, CONN

LAFON, J W
     825 RIO GRANDE BLVD N.W.
     ALBUQUERQUE, N.M.         87104

LAFONTAINE, J C
     5824 NEVADA AVENUE NO.
     MINNEAPOLIS, MINN         55428

LAING, C D
     567 TURNER DRIVE
     BURLINGTON, ONT           CANADA

LAMPTON, G B
     161 FAIRWAY CIRCLE
     ROCK, S.C.                29730

LAMPNE, D J
     16 RANCH TRAIL ROAD
     WILLIAMSVILLE, N.Y.       14221

LANDWEHR, M E
     MONTEREY & COTTLE ROADS
     SAN JOSE, CAL             95030

LANE, S E
     3405 N.W. 40TH
     OKLAHOMA, OKLA            73102

LANE, W G
     2194 NORTH AVENUE
     CHICO, CAL                95926

LAPORTE, D R
     BOX 112 BAY ROAD
     BELCHERTOWN, MASS         01007

LAROCQUE, J G
     MAIN
     HAWKESBURY, ONT           CANADA
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

LEACH, U H	
11101 INWOOD AVENUE	
SILVER, MD	20902
LEBLANC, M A	
6980-14TH AVE. APT 4	
MONTREAL, QUE	CANADA
LEGER, R	
2442 TRENTON AVENUE	
MONTREAL, QUE	CANADA
LEHNER, M F	
5049 TRUESDALE AVENUE	
BALTIMORE, MD	21206
LENNON, R D	
1833 WANNINGER	
CINCINNATI, OHIO	45230
LESTER, G	
1232 CHATEAU DRIVE	
SAN JOSE, CAL	95120
LEWIS, E	
786 ENGLEWOOD	
BUFFALO, N.Y.	14223
LIGON, H H	
LOTT, TEX	76656
LITTLE, J C	
723 NOTTINGHAM ROAD	
BALTIMORE, MD	21229
LIVEZEY, W C	
1608 WALNUT STREET	
PHILADELPHIA, PA	19103
LOGUE, W E	
DIETZ ROAD	
WARREN, OHIO	44482
LONERGAN, P	
LOUIS, J Y	
175 OLD COUNTRY ROAD	
HICKSVILLE, N.Y.	11801

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
LUDWIG, D A
     251 NORTH CASSINGHAM ROAD
     COLUMBUS, OHIO           43209

LUKINS, J C
     3519 WILLOWOOD DRIVE
     LEXINGTON, KY            40502

LUNEBURG, I
     RFD1
     SOUTHBRIDGE, MASS        01550

LUNGER, G C
     RD. 2
     COLUMBIANA, OHIO         44408

LYDIKSEN, H W
     2 TOWNS ROAD
     LEVITTOWN, PA            19056

LYNCH, S A
     8028 VAN BUREN
     MUNSTER, IND             46321

LYON, K W
     1357 HILLCREST
     CINCINNATI, OHIO         45224

MACNAUGHTON, B B
     5050 POPLAR
     MEMPHIS, TENN            38117

MAGEE, R H
     2131 BUETER ROAD
     FT. WAYNE, IND           46803

MALLON, R
     94 PERRY STREET
     HARRISONBURG, VA         22801

MANGOLD, D R
     2830 VICTORY PARKWAY
     CINCINNATI, OHIO         45206

MANIKOWSKI, P R
     112 E. POST ROAD
     WHITE, N.Y.

MANLEY, R A
     1 FEDERAL STREET
     YONKERS, N.Y.            10702
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
MANN, E F
     172 GATE HOUSE TRAIL
     HENRIETTA, N.Y.          14467

MAPPUS, J H
     BOX 5207
     NORTH, S.C. STON         29406

MARKS, M
     112 E. POST ROAD
     WHITE, N.Y.              10601

MARKULIN, T
     1701 N STREET
     ENDICOTT, N.Y.

MARTIN, D J
     3512 VALLEY TRAIL
     CHATTANOOGA, TENN        37405

MARTIN, W E
     USN UNDERWATER SOUND LAB
     NEW YORK, N.Y.           09560

MASKIELL, F M
     149 DEMAR BLVD
     CANONSBURG, PA           15317

MATELCER, F W
     920 N 14TH
     DEKALB, ILL              60115

MATHEWS, W M
     308 MARSHALL DRIVE
     SHILLINGTON, PA          19607

MATHIASON, L J
     617 W. 5TH STREET
     CHILLICOTHE, OHIO        45601

MATTATALL, G L
     47 FRASER COURT
     NEWCASTLE, N.B.

MATTHIS, F
     669 BROUGHTON ROAD
     BETHEL, PA               15102

MATTHEISS, P K
     42-6 REVERE ROAD
     DREXEL, PA               19026
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
MAUDLIN, C E
     TEXAS WOMANS UNIVERSITY
     DENTON, TEX                76204

MCCALL, E H
     2676 ROTH PLACE
     WHITE, MINN AKE            55110

MCILVAIN, D R
     1528 WALNUT STREET
     PHILADELPHIA, PA           19102

MCKAY, C D
     25 GROSVENOR COURT
     KINGSTON, ONT              CANADA

MCLANGHLIN, E E
     2345 IRIS
     LAKEWOOD, COLO             80215

MCMINN, C S
     14 HILLVIEW DRIVE
     NORWICH, N.Y.              13815

MCMEILL, D W
     P.O. BOX 34380
     DALLAS, TEX                75234

MCPHILLIPS, T J
     719 US POST OFFICE & CH
     CINCINNATI, OHIO           45202

MCCUSKER, P A
     4254 CARPENTER AVENUE
     NEW YORK, N.Y.             10466

MELUSKEY, J T
     920 OLDE HICKORY ROAD
     LANCASTER, PA              17601

MENEGHELLI, H A
     3414 KENDALL CIRCLE
     CUYAHOGA, OHIO             44221

METEER, R C
     252 COUNTY CENTER ROAD
     WHITE, N.Y.                10603

MEURER, R F
     327 WALMAR DRIVE
     BAY VILLAGE, OHIO
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
MICHAEL, L A
     52 BYPASS SO.
     LAFAYETTE, IND          47905

MICHALOWSKI, E W
     293 HAMILTON AVENUE
     TONAWANDA, N.Y.         14150

MICKEL, F B
     1730 LYTER DRIVE
     JOHNSTOWN, PA           15905

MIKO, D E
     32 MARK
     ST. MARYS, PA           15857

MILLER, S R

     CHINA, CAL              93555

MILLS, M E
     2043 SUTTON AVENUE
     CINCINNATI, OHIO        45230

MILLER, G C
     5917 SANDHURST LANE 222
     DALLAS, TEX             75206

MILLER, H W
     630 RIDGEWAY COURT
     MONROE, OHIO            45050

MODELL, D J
     300 UNION COMMERSE BLDG.
     CLEVELAND, OHIO         44115

MONJEAU, G P
     31 PASTURE LANE
     POUGHKEEPSIE, N.Y.      12603

MOORE, J C
     402 ORANGE STREET
     MADISON, FLA            32340

MORTON, J R
     1939 E. FIRST STREET
     DAYTON, OHIO            45401

MOSCHETTI, R J
     4TH STREET
     JEANNETTE, PA           15644
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

MUELLER, J D HIGHWAY 52 NORTH ROCHESTER, MINN	55901
MULGREW, I D 12700 KERCHEVAL DETROIT, MICH	48215
NARDONE, A A 166 HIGH STREET WESTERLY, R.I.	02891
NEMETH, L E 54 ROSSITER AVENUE PHOENIXVILLE, PA	19460
NORTON, W A 600 N. 18TH STREET BIRMINGHAM, ALA	35202
NOVAK, M J 834 EPPLEY AVENUE ZANESVILLE, OHIO	43701
ODESKY, R I 1536 ALEXANDRIA LEXINGTON, KY	40504
OKEEFFE, W H HOWARD STREET FRANKLIN, PA	16301
OLDE, G L 3412 BELLEFONTE DRIVE LEXINGTON, KY	40502
ORLOFF, M J 4141 EASTERN AVENUE S.E. GRAND, MICH	49508
OWEN, J J P.O. BOX 570 SAVANNAH, GA	31402
PANELLA, D B 1069 LINDEN AVENUE AKRON, OHIO	44310
PARISIAN, J E 522 W. CHEMUNG STREET PAINTED, N.Y.	14870

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

PARKER, C D

MILFORD, MICH 48042

PASSICK, J E
1001 BROAD STREET
JOHNSTOWN, PA 15907

PAULSEN, J E
1200 17TH STREET N.W.
WASHINGTON, D.C. 20036

PEARSON, L W
1001 JEFFERSON
OXFORD, MISS 38655

PEDERSEN, P M
124 RIVERSIDE DRIVE
HOPEWELL, VA 23860

PEDIN, P S
2000 FORRER BLVD
DAYTON, OHIO 45401

PERFETTE, B
669 EVERGREEN DRIVE
TONOWANDA, N.Y. 14150

PETERS, C D
527 RIVERSIDE DRIVE 6A
NEW YORK, N.Y. 10027

PHILLIPS, R W
CLOVE ROAD
VERBANK, N.Y.

PITEL, R
159 SCENIC DRIVE
HORSEHEADS, N.Y. 18450

POLISHOOK, B H
105 WAVERLY ROAD
SCARTDALE, N.Y.

PONIKVAR, H E
DIETZ ROAD
WARREN, OHIO 44482

POPOVICH, G G
4 HILLSIDE COURT
ENDICOTT, N.Y. 13760

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

PORTER, C B
ILLINOIS STATE UNIVERSITY
NORMAL, ILL 61761

POWELL, J O
1327 CATHERWOOD DRIVE
SOUTH, IND 46614

PRATT, R L
7500 OLD XENIA PIKE
DAYTON, OHIO 45432

QUAYLE, B B
6924 RAVENSCROFT
ST. LOUIS, MO 63123

QUO , P C
63 CROMWELL ROAD
CINCINNATI, OHIO 45218

RAAB, P V
1201 S. SECOND STREET
MILWAUKEE, WISC 53204

RANDALL, R F
826 SOUTH TRAVIS
SHERMAN, TEX 75090

RANDALL, D C
18835 KINGS ROAD
HOMEWOOD, ILL 60430

RAPP, K L
RR 1
WAVERLY, OHIO 45601

RAVER, R E
MINK HOLLOW ROAD
HIGHLAND, MD 20777

REDDING, D M
4701 MARBURG AVENUE
CINCINNATI, OHIO 45209

REDINGER, S J
3539 GLENEDGE LANE
CINCINNATI, OHIO 45213

REDINGER, L A
3539 GLENEDGE LANE
CINCINNATI, OHIO 45213

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
REES, W A
     2310 FOXLEY ROAD
     TIMONIUM, MD              21093

REINHARDT, B N
     24 COLLEGE HAVEN
     ANDERSON, IND             46012

REYNOLDS, G J
     1700 WEST THIRD AVENUE
     FLINT, MICH               48502

REYNOLDS, J M
     COMPUTER CENTER W GEORGIA CO
     CARROLLTON, GA            30117

RIBERA, V R
     1809 HARDY DRIVE
     EDMOND, OKLA              73034

RISPOLI, L M
     P.O. BOX 5207
     NORTH, S.C. STON          29406

RISSOLO, L R
     3326 CROSSCOUNTRY DRIVE
     WILMINGTON, DEL           19803

RITTER, T L

     MOUNT, OHIO               43050

RODANTE, F C
     250 CONSTITUTION PLAZA
     HARTFORD, CONN            06103

RODENBECK, R
     1400 SOUTH THIRD STREET
     ST. LOUIS, MO             63166

ROESSING, K W
     4028 IMPALA DRIVE
     PITTSBURGH, PA            15239

ROESCH, R A
     5225 TROOST AVENUE
     KANSAS, MO

ROHR, N R
     RT. 1
     WASHINGTON, W VA          26181
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

ROSS, R D
COMPUTER CENTER
UNIVERSITY, MISS 38677

ROTH, R E
COMPUTER CENTRE
GENESEO, N.Y. 14454

ROTH, R W
TAYLOR UNIVERSITY
UPLAND, IND 46989

ROTHSCHILD, R C
32 W. GRIMSBY
BUFFALO, N.Y. 14223

ROWE, L V
5 BEECHNUT DRIVE
WEST, OHIO 45383

ROWLAND, L W
436 SUMMIT DRIVE
CHILLICOTHE, OHIO 45601

SAK , T L
P.O. BOX 5536
HOUSTON, TEX 77034

SAMUELS, L B
1271 AVE OF AMERICAS
NEW YORK, N.Y. 10020

SANDBERG, A A
3504 W. ADAMS STREET
BELLWOOD, ILL 60104

SAUNDERS, A F
80 RANGE HILL DRIVE
ROCKVILLE, CONN 06066

SCARAMOZZINO, P J
297 BOSTON AVENUE
MEDFORD, MASS 02155

SCEARCE, W A
2440 GRINSTEAD DRIVE
LOUISVILLE, KY 40204

SCHEMMER, J A
4660 SO. 60TH AVENUE
OMAHA, NEBR 68117

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

SCHILDER, M
2077 SIERRA ROAD
PLYMOUTH, PA G 19462

SCHODITSCH, G F
1700 SOUTH SECOND STREET
ST. LOUIS, MO 63177

SCHROEDER, L R
73 COMO AVENUE
BUFFALO, N.Y. 14220

SCHULTZ, T E
316 ANITA DRIVE
WINSTON-SALEM, N.C. 27104

SCOTT, R E
298 LINCOLNIA ROAD
ALEXANDRIA, VA 22304

SEITZ, L J
2480 WEST 70TH AVENUE
DENVER, COLO 80221

SELSMEYER, W T
8 RIVERVIEW ROAD
APALACHIN, N.Y. 13732

SERDENGECTI, S
3821 LYNOAK
CLAREMONT, CAL 91711

SEROUSSI, S F
P.O. BOX 208
BEDFORD, MASS 01730

SHAFFER, M R
675 GARDEN PKWY
CIRCLEVILLE, OHIO 43113

SHAFF, P H
1960 SEYMOUR AVENUE
CINCINNATI, OHIO 45237

SHARP, E A
CREIGHTON UNIVERSITY
OMAHA, NEBR 68131

SHEPHERD, K M
1211 FEDERAL AVENUE
ZANESVILLE, OHIO 43701

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
SHERMER, D A
     275 WINCHESTER AVENUE
     NEW HAVEN, CONN            06504

SHIRER, F D
     P.O. BOX 1212
     HOUSTON, TEX               77024

SHOUSE, J W
     596 SOUTH 10TH
     SAN JOSE, CAL              95112

SIMMERMACHER, W R
     6 ROSE LANE
     CHAPPAOVA, N.Y.            10514

SIMS, D L
     3516 W. SHANDON
     MIDLAND, TEX               79701

SLUDER, L R
     175 OLD COUNTRY ROAD
     HICKSVILLE, N.Y.           11801

SMITH, A P
     112 EAST POST ROAD
     WHITE, N.Y.

SMITH, R L
     710 AIRFIELD LANE
     MIDLAND, MICH              48640

SNAILER, R J
     53 LACE LANE
     WESTBURY, N.Y.             11590

SOLE, W E
     1466 EGMOND DRIVE
     SARNIA, ONT                CANADA

SONNENBERG, L K
     1770 RADCLIFFE ROAD
     DAYTON, OHIO               45406

SOUSLEY, J E
     6838 KIRKDALE DRIVE
     FORT, IND                  46805

STACKE, W B
     241 6TH AVENUE
     NEW YORK, N.Y.             10014
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
STANSBURY, J C
     2 PARK AVENUE
     NEW YORK, N.Y.          10016

STAPLETON, W L
     1150 EGLINTON AVENUE EAST
     DON MILLS, ONT          CANADA

STAUTNER, J F
     112 EAST POST ROAD
     WHITE, N.Y.             10601

STEELE, L
     18 ANTHONY DRIVE
     POUGHKEEPSIE, N.Y.      12601

STEGINA, F J
     475 ELM STREET
     WEST, CONN              06516

STEIN, T W
     30 BRANFORD ROAD
     HASTINGS-ON-HUDSON, N.Y

STEWART, W B
     ROSE STREET
     LEXINGTON, KY

STRAUSS, W T
     1901 CHAPMAN AVENUE
     ROCKVILLE, MD           20852

STRITE, R S
     2628 WOODLEY PLACE
     FALLS, VA               22046

STROH, G B
     1727 HOLLYWOOD
     GROSSE, MICH  WOODS     48236

SUM  , B C
     FISHKILL PARK APTS 2B
     FISHKILL, N.Y.

SWAIN, P J
     620 CHESTER STREET
     MONTREAL, QUE           CANADA

TAYLOR, J S
     7500 OLD XENIA PIKE
     DAYTON, OHIO            45432
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
TENNISON, R D
     112 E. POST ROAD
     WHITE, N.Y.                10601

THOMSON, W K
     627 SALEM AVENUE
     DAYTON, OHIO               45406

THOMSON, G W
     1600 WEST 8 MILE ROAD
     FERNDALE, MICH             48220

TILTON, B A
     1828 NICHOLASVILLE ROAD
     LEXINGTON, KY              40503

TRACY, M G
     RED 3, ROCK ROAD
     MANSFIELD, OHIO            44903

TRAINER, F E
     2830 VICTORY PARKWAY
     CINCINNATI, OHIO           45206

TUNNEY, J L
     627 SALEM AVENUE
     DAYTON, OHIO               45406

TVEDT, R G
     MPHS ST UNIV ADM BLD 122
     MEMPHIS, TENN              38111

VAN NESS, V V
     2 ROCKLEDGE ROAD
     PLEASANTVILLE, N.Y.        10570

VAUGHAN, N C
     3424 WILSHIRE BLVD
     LOS ANGELES, CAL           90005

VERBRUGGE, M G
     RR 4
     RENSSELAER, IND            47978

VERITY, A F
     114 MEYER AVENUE
     VALLEY, N.Y.               11580

VOGEL, F
     52 BYPASS SO.
     LAFAYETTE, IND             47905
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

WADSWORTH, M A
600 NORTH SECOND STREET
HARRISBURG, PA 63 17105

WALKER, J P
4948 POWELL ROAD
DAYTON, OHIO 45424

WALKER, R P
3418 KNOX STREET
ST. JOSEPH, MICH 49085

WALL, D D
39 EVERGREEN CIRCLE
PRINCETON, N.J. 08540

WARNER, G
8033 LINDEN
MUNSTER, IND 46321

WARTAN, W A
600 S.W. 4TH STREET
BIRMINGHAM, ALA 35211

WATKINS, J B
7 EAST LAKEVIEW DR APT 1
CINCINNATI, OHIO 45237

WATKINS, C A
1600 W. 8 MILE ROAD
FERNDALE, MICH 48220

WEAVER, S E
120 BRIARWOOD WAY
LOS GATOS, CAL 95030

WEBER, D L
1400 SHERIDAN ROAD
NORTH, ILL O 60064

WEGNER, R D
1300 E. STREET N.W.
WASHINGTON, D.C.

WEHE, H W
RD 3
LIGONIER, PA 15658

WEISS, J H

ANGOLA, IND 46703

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

WERNICKE, G K	
4561 SEVILLE DRIVE	
ENGLEWOOD, OHIO	45322
WERNER, R T	
3201 BOUDINOT AVENUE	
CINCINNATI, OHIO	45211
WEST, R W	
107 SEMINARY AVENUE	
BINGHAMTON, N.Y.	13905
WESTERHAM, E A	
1042 N. JEFFERSON	
OTTWMUA, IOWA	52501
WESTON, D J	
SARNIA, ONT	CANADA
WHELAN, L	
BOX 2408	
GARY, IND	
WIGDAHL, A B	
1201 S. SECOND STREET	
MILWAUKEE, WISC	53204
WILLIAMSON, R T	
2500 STEPHEN ROAD	
LOUISVILLE, KY	40214
WILLARD, L B	
WAHPETON, N.D.	58075
WILSON, R J	
6350 MAPLE DRIVE	
INDIANAPOLIS, IND	
WINDHAM, C E	
213 SIVLEY	
OXFORD, MISS	38655
WOLF, G L	
6684 LANDERWOOD LANE	
SAN JOSE, CAL	95120
WOOD, E J	
549 W. WASHINGTON BLVD	
CHICAGO, ILL	60606

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

WOOD, P C
30 GLEVELLYN ROAD
LOWELL, MASS 01852

WOODROW, P J
50 WASHINGTON ROAD
PRINCETON, N.J. 08540

WRIGHT, J R
3110 SO. 27TH STREET
LA CROSSE, WISC 54601

WRIGHT, E J
MILTON TURNPIKE
MILTON, N.Y. 12547

WYNN, W E
15 ASH
PARK, ILL 60466

YAMANAKA, J H
11815 S.W. FONNER STREET
TIGARD, ORE 97223

YANKOVICH, J M
2314 HENDERSON
BETHLEHEM, PA 18017

YOUNGBERG, R W
85 AUBURN LANE
EAST, N.Y. H 11732

YOUSSEF, K E
1441 SMITHFIELD STREET
PITTSBURGH, PA 15222

ZACHLIN, A C
3720 GROSVENOR
CLEVELAND, OHIO 44118

ZIELINSKI, E T
7753 WESTWIND LANE
CINCINNATI, OHIO 45242

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

PANELLA, D B
    1069 LINDEN AVENUE
    AKRON, OHIO 44310

LAFON, J W
    825 RIO GRANDE BLVD N.W.
    ALBUQUERQUE, N.M. 87104

CEELY, F F
    6206 BREN MAR DRIVE
    ALEXANDRIA, VA 22312

SCOTT, R E
    298 LINCOLNIA ROAD
    ALEXANDRIA, VA 22304

GREEN, D M
    7991 COLONY DRIVE
    ALGONAC, MICH

GOSSETT, E C
    37 ALLEN
    ALLENDALE, N.J. 04701

BRENNAN, R D
    2670 HANOVER
    ALTO, CALIF

HARBRON, T R
    ANDERSON COLLEGE
    ANDERSON, IND 46011

REINHARDT, B N
    24 COLLEGE HAVEN
    ANDERSON, IND 46012

WEISS, J H

    ANGOLA, IND 46703

CARLSON, D M
    327 S. FOURTH AVE
    ANN ARBOR, MICH 48108

CUNNINGHAM, A J
    25 MAIN ST
    ANSONIA, CONN 06401

JOHNSTON, W H
    25 MAIN STREET
    ANSONIA, CONN 06401

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
SELSMEYER, W T
     8 RIVERVIEW ROAD
     APALACHIN, N.Y.          13732

BABER, G M
     3000 SPOUT RUN PARKWAY
     ARLINGTON, VA            22201

DONNELLY, M J
     906 BELGIAN AVENUE
     BALTIMORE, MD            21218

DOWD, C K
     3711 LOCHEARN DRIVE
     BALTIMORE, MD            21207

GIBSON, S
     3117 MILFORD AVE.
     BALTIMORE, MD            21207

JONES, R L
     1514 WOODCLIFFE AVENUE
     BALTIMORE, MD            21228

LEHNER, M F
     5049 TRUESDALE AVENUE
     BALTIMORE, MD            21206

LITTLE, J C
     723 NOTTINGHAM ROAD
     BALTIMORE, MD            21229

COX , R C
     1429 KEED AVENUE
     BATON, LA                70806

MEURER, R F
     327 WALMAR DRIVE
     BAY VILLAGE, OHIO

CONROD, R L
     P.O. BOX 208
     BEDFORD, MASS            01730

SEROUSSI, S F
     P.O. BOX 208
     BEDFORD, MASS            01730

LAPORTE, D R
     BOX 112 BAY ROAD
     BELCHERTOWN, MASS        01007
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

SANDBERG, A A
3504 W. ADAMS STREET
BELLWOOD, ILL 60104

COOPER, T
USDAARSBSSARCBLDG226
BELTSVILLE, MD 20705

MATTHIS, F
669 BROUGHTON ROAD
BETHEL, PA 15102

YANKOVICH, J M
2314 HENDERSON
BETHLEHEM, PA 18017

CLARK, W H
4 EDGEBROOK ROAD
BINGHAMTON, N.Y. 13903

WEST, R W
107 SEMINARY AVENUE
BINGHAMTON, N.Y. 13905

NORTON, W A
600 N. 18TH STREET
BIRMINGHAM, ALA 35202

WARTAN, W A
600 S.W. 4TH STREET
BIRMINGHAM, ALA 35211

GANATRA, J K
458 FOX HILL DRIVE
BLOOMFIELD, MICH 48013

CASTELLAN, N J
DEPT OF PSYCHOLOGY I U
BLOOMINGTON, IND 47401

DOUCETTE, J E
109 CHANDLER STREET
BOSTON, MASS 02116

GOLDMAN, N
111 CUMMINGTON STREET
BOSTON, MASS 02215

GREEN, J
JAMAICAWAY TOWER 151
BOSTON, MASS 02130

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

KINDRED, A R
2707 RUTGERS AVENUE
BRADENTON, FLA 33505

DUQUETTE, D J
THIRD AVE & FORDHAM ROAD
BRONZ, N.Y. 10458

FALCONELLO, P
THIRD AVE & FORDHAM ROAD
BRONZ, N.Y. 10458

OLDBERG, M
THIRD AVE & FORDHAM ROAD
BRONZ, N.Y. 10458

BOND, W F
656 STORM AVE
BROOKHAVEN, MISS 39601

KOEPSELL, P L
SOUTH DAK STATE UNIV.
BROOKINGS, S.D. 57006

LEWIS, E
786 ENGLEWOOD
BUFFALO, N.Y. 14223

ROTHSCHILD, R C
32 W. GRIMSBY
BUFFALO, N.Y. 14223

SCHROEDER, L R
73 COMO AVENUE
BUFFALO, N.Y. 14220

BRENNER, R L
P.O. BOX 561
BURLINGTON, IOWA 52601

LAING, C D
567 TURNER DRIVE
BURLINGTON, ONT CANADA

FISHER, J N
363 THIRD STREET
CAMBRIDGE, MASS 02148

FRASER, W I
21 ETON COURT
CAMLACHIE, ONT CANADA

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
MASKIELL, F M
     149 DEMAR BLVD
     CANONSBURG, PA          15317

REYNOLDS, J M
     COMPUTER CENTER W GEORGIA CO
     CARROLLTON, GA          30117

SIMMERMACHER, W R
     6 ROSE LANE
     CHAPPAOVA, N.Y.         10514

MARTIN, D J
     3512 VALLEY TRAIL
     CHATTANOOGA, TENN       37405

KEESLING, M A
     5721 SOUTH KIMBARK
     CHICAGO, ILL            60637

KEESLING, J W
     5721 SOUTH KIMBARK
     CHICAGO, ILL            60637

KELLEY, J R
     618 SOUTH MICHIGAN BLVD
     CHICAGO, ILL            60605

WOOD, E J
     549 W. WASHINGTON BLVD
     CHICAGO, ILL            60606

LANE, W G
     2194 NORTH AVENUE
     CHICO, CAL              95926

ATER, G T
     35 PAGE RD
     CHILLICOTHE, OHIO       45601

CLARKE, J R
     449 W. 5TH ST
     CHILLICOTHE, OHIO       45601

DRAY, R D
     47 PAGE ROAD
     CHILLICOTHE, OHIO       45601

JINKS, W V
     22 E. 7TH STREET
     CHILLICOTHE, OHIO       45601
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

MATHIASON, L J 617 W. 5TH STREET CHILLICOTHE, OHIO	45601
ROWLAND, L W 436 SUMMIT DRIVE CHILLICOTHE, OHIO	45601
MILLER, S R CHINA, CAL	93555
CLEGG, J B 346 COMPTON RD CINCINNATI, OHIO	45215
DENING, J W 3451 MCHENRY AVENUE CINCINNATI, OHIO	45225
DUNSMORE, D A 414 WALNUT STREET CINCINNATI, OHIO	45202
HAMANT, W E 8302 MAYFAIR CINCINNATI, OHIO	45216
HAMPEL, C R 222 E. CENTRAL PKWY CINCINNATI, OHIO	45202
HICKMAN, G A 222 E. CENTRAL PKWY CINCINNATI, OHIO	45202
INGRAM, B C 1018 ROSETREE LANE CINCINNATI, OHIO	45243
JOHNSON, B M 6126 THOLE ROAD CINCINNATI, OHIO	45230
LENNON, R D 1833 WANNINGER CINCINNATI, OHIO	45230
LYON, K W 1357 HILLCREST CINCINNATI, OHIO	45224

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
MANGOLD, D R
     2830 VICTORY PARKWAY
     CINCINNATI, OHIO         45206

MCPHILLIPS, T J
     719 US POST OFFICE & CH
     CINCINNATI, OHIO         45202

MILLS, M E
     2043 SUTTON AVENUE
     CINCINNATI, OHIO         45230

QUO , P C
     63 CROMWELL ROAD
     CINCINNATI, OHIO         45218

REDDING, D M
     4701 MARBURG AVENUE
     CINCINNATI, OHIO         45209

REDINGER, S J
     3539 GLENEDGE LANE
     CINCINNATI, OHIO         45213

REDINGER, L A
     3539 GLENEDGE LANE
     CINCINNATI, OHIO         45213

SHAFF, P H
     1960 SEYMOUR AVENUE
     CINCINNATI, OHIO         45237

TRAINER, F E
     2830 VICTORY PARKWAY
     CINCINNATI, OHIO         45206

WATKINS, J B
     7 EAST LAKEVIEW DR APT 1
     CINCINNATI, OHIO         45237

WERNER, R T
     3201 BOUDINOT AVENUE
     CINCINNATI, OHIO         45211

ZIELINSKI, E T
     7753 WESTWIND LANE
     CINCINNATI, OHIO         45242

SHAFFER, M R
     675 GARDEN PKWY
     CIRCLEVILLE, OHIO        43113
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
ALLBRITTON, E J
     &3 BOX 191
     CLARDSVILLE, MO           63336

SERDENGECTI, S
     3821 LYNOAK
     CLAREMONT, CAL            91711

BUFE, O E
     2300 CHESTER AVE
     CLEVELAND, OHIO           44114

HOFFERT, E R
     1705 URBANA ROAD
     CLEVELAND, OHIO           44112

MODELL, D J
     300 UNION COMMERSE BLDG.
     CLEVELAND, OHIO           44115

ZACHLIN, A C
     3720 GROSVENOR
     CLEVELAND, OHIO           44118

HYMAN, I M
     LUKENS STEEL COMPANY
     COATESVILLE, PA           19320

LUNGER, G C
     RD. 2
     COLUMBIANA, OHIO          44408

BOBAY, J P
     1000 5TH ST
     COLUMBUS, IND             47201

BRECHBILL, D O
     625 CLEVELAND
     COLUMBUS, OHIO            43215

DAGENFIELD, R L
     1214 JAEGER ST
     COLUMBUS, OHIO            43206

GEIGER, A J
     2186 MIDDLEHURST DRIVE
     COLUMBUS, OHIO            43201

HAMILTON, T K
     315 GRAHAM AVENUE
     COLUMBUS, OHIO            43203
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
LUDWIG, D A
     251 NORTH CASSINGHAM ROAD
     COLUMBUS, OHIO           43209

KERR, H B
     BOX 21A TTV
     COOKEVILLE, TENN         38501

KERR, M C
     BOX 21A TTV
     COOKEVILLE, TENN         38501

MENEGHELLI, H A
     3414 KENDALL CIRCLE
     CUYAHOGA, OHIO           44221

FLEMING, C O
     P.O. BOX 34380
     DALLAS, TEX              75234

MCMEILL, D W
     P.O. BOX 34380
     DALLAS, TEX              75234

MILLER, G C
     5917 SANDHURST LANE 222
     DALLAS, TEX              75206

BENEDICT, D E
     2000 FORRER BLVD
     DAYTON, OHIO             45401

DONALDSON, D L
     1570 WOODMAN DR 15
     DAYTON, OHIO             45432

MORTON, J R
     1939 E. FIRST STREET
     DAYTON, OHIO             45401

PEDIN, P S
     2000 FORRER BLVD
     DAYTON, OHIO             45401

PRATT, R L
     7500 OLD XENIA PIKE
     DAYTON, OHIO             45432

SONNENBERG, L K
     1770 RADCLIFFE ROAD
     DAYTON, OHIO             45406
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

TAYLOR, J S
7500 OLD XENIA PIKE
DAYTON, OHIO 45432

THOMSON, W K
627 SALEM AVENUE
DAYTON, OHIO 45406

TUNNEY, J L
627 SALEM AVENUE
DAYTON, OHIO 45406

WALKER, J P
4948 POWELL ROAD
DAYTON, OHIO 45424

HEETDERKS, J W
3001 MILLER P.O.BOX 218
DEARBORN, MICH

BUCKLEU, R E
1414 BEECH ST. S.E.
DECATUR, ALA 35601

MATELCER, F W
920 N 14TH
DEKALB, ILL 60115

MAUDLIN, C E
TEXAS WOMANS UNIVERSITY
DENTON, TEX 76204

SEITZ, L J
2480 WEST 70TH AVENUE
DENVER, COLO 80221

ALLEN, J E
1206 MULBERRY ST
DES MOINES, IOWA 50309

BAKER, L H
1206 MULBERRY ST
DES MOINES, IOWA 50308

IBSEN, J K
5985 N.W. 62ND AVENUE
DES MOINES, IOWA 50324

MULGREW, I D
12700 KERCHEVAL
DETROIT, MICH 48215

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI OHIO

```
STAPLETON, W L
     1150 EGLINTON AVENUE EAST
     DON MILLS, ONT           CANADA

MATTHEISS, P K
     42-6 REVERE ROAD
     DREXEL, PA               19026

YOUNGBERG, R W
     85 AUBURN LANE
     EAST, N.Y. H             11732

RIBERA, V R
     1809 HARDY DRIVE
     EDMOND, OKLA             73034

JAEGER, R B
     RD 1
     EMPORIUM, PA             15834

ALVAREZ, J J
     IBM GLENDALE LAB DEPT 265
     ENDICOTT, N.Y.           13760

DONEGAN, A W
     107 SMITHFIELD DRIVE
     ENDICOTT, N.Y.           13760

HOKE, W E
     104 SMITHFIELD DRIVE
     ENDICOTT, N.Y.

MARKULIN, T
     1701 N STREET
     ENDICOTT, N.Y.

POPOVICH, G G
     4 HILLSIDE COURT
     ENDICOTT, N.Y.           13760

KOERING, L O
     3725 FRAZIER ROAD
     ENDWELL, N.Y.

WERNICKE, G K
     4561 SEVILLE DRIVE
     ENGLEWOOD, OHIO          45322

BUESKING, C W
     4617 TAYLOR AVE
     EVANSVILLE, IND          47715
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

CARLSON, D R	
5707 LINDENWOOD LANE	
FAIRFIELD, OHIO	
STRITE, R S	
2628 WOODLEY PLACE	
FALLS, VA	22046
CUMMINGS, L R	
FEDERALSBURG, MD	21632
THOMSON, G W	
1600 WEST 8 MILE ROAD	
FERNDALE, MICH	48220
WATKINS, C A	
1600 W. 8 MILE ROAD	
FERNDALE, MICH	48220
SUM , B C	
FISHKILL PARK APTS 2B	
FISHKILL, N.Y.	
ARNTSON, W M	
1700 W. THIRD AVE.	
FLINT, MICH	48502
AUSTIN, L B	
1700 W. THIRD AVE	
FLINT, MICH	48502
REYNOLDS, G J	
1700 WEST THIRD AVENUE	
FLINT, MICH	48502
JEFFUS, S E	
5626 MAYNARD	
FORT, TEX	79906
SOUSLEY, J E	
6838 KIRKDALE DRIVE	
FORT, IND	46805
OKEEFFE, W H	
HOWARD STREET	
FRANKLIN, PA	16301
FOLTZ, T V	
4818 AVONDALE DRIVE	
FT. WAYNE, IND	46806

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

MAGEE, R H
2131 BUETER ROAD
FT. WAYNE, IND 46803

WHELAN, L
BOX 2408
GARY, IND

DEUTSCH, E
COMPUTER CENTRE
GENESEO, N.Y. 14454

ROTH, R E
COMPUTER CENTRE
GENESEO, N.Y. 14454

ORLOFF, M J
4141 EASTERN AVENUE S.E.
GRAND, MICH 49508

COTTON, B W
P.O. DRAWER G
GRAPEVINE, TEX 76051

BICKFORD, P A

GREENCASTLE, IND 46135

STROH, G B
1727 HOLLYWOOD
GROSSE, MICH WOODS 48236

HILL, W H
6715 BLVD EAST
GUTTENBERG, N.J. 07093

KISSNER, J R
55 E. WASHINGTON
HAGERSTOWN, MD 21740

CIPRIETTI, B J
34 IPSWICH PLACE
HAMILTON, ONT CANADA

DELONG, W
629 SIBLEY STREET
HAMMOND, IND 46320

BERNARD, R J
606 JODI DR
HAMMOND, LA 70401

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

WADSWORTH, M A
600 NORTH SECOND STREET
HARRISBURG, PA 63 17105

MALLON, R
94 PERRY STREET
HARRISONBURG, VA 22801

RODANTE, F C
250 CONSTITUTION PLAZA
HARTFORD, CONN 06103

STEIN, T W
30 BRANFORD ROAD
HASTINGS-ON-HUDSON, N.Y

BERWICK, J M
MAIN
HAWKESBURY, ONT CANADA

LAROCQUE, J G
MAIN
HAWKESBURY, ONT CANADA

DYE , D R
40 SAWMILL RIVER ROAD
HAWTHORNE, N.Y.

BLISS, R J
5830 WESTHENRIETTA RD
HENRIETTA, N.Y. 14606

MANN, E F
172 GATE HOUSE TRAIL
HENRIETTA, N.Y. 14467

DEGENNARO, M G
175 OLD COUNTRY RD
HICKSVILLE, N.Y. 11801

LOUIS, J Y
175 OLD COUNTRY ROAD
HICKSVILLE, N.Y. 11801

SLUDER, L R
175 OLD COUNTRY ROAD
HICKSVILLE. N.Y. 11801

RAVER, R E
MINK HOLLOW ROAD
HIGHLAND, MD 20777

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

JOHNSON, D R	227 W. 10TH HINSDALE, ILL	60521
RANDALL, D C	18835 KINGS ROAD HOMEWOOD, ILL	60430
PEDERSEN, P M	124 RIVERSIDE DRIVE HOPEWELL, VA	23860
PITEL, R	159 SCENIC DRIVE HORSEHEADS, N.Y.	18450
SAK , T L	P.O. BOX 5536 HOUSTON, TEX	77034
SHIRER, F D	P.O. BOX 1212 HOUSTON, TEX	77024
BERGER, D E	2605 REYNOLDS CR. HUNTSVILLE, ALA	35810
CRAFT, J C	2109 NORRIS RD N.W. HUNTSVILLE, ALA	35810
BARNEY, P L	740 S. ALABAMA INDIANAPOLIS, IND	46206
GRIBBEN, C J	25 MONUMENT CIRCLE INDIANAPOLIS, IND	46200
WILSON, R J	6350 MAPLE DRIVE INDIANAPOLIS, IND	
GILMARTIN, W R	126 STRATFORD DRIVE IRWIN, PA	15642
MOSCHETTI, R J	4TH STREET JEANNETTE, PA	15644

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
KUHN, R E
     RD 2 BOX 94-C
     JERSEY, PA              17740

MICKEL, F B
     1730 LYTER DRIVE
     JOHNSTOWN, PA           15905

PASSICK, J E
     1001 BROAD STREET
     JOHNSTOWN, PA           1590[illegible]

ROESCH, R A
     5225 TROOST AVENUE
     KANSAS, MO

MCKAY, C D
     25 GROSVENOR COURT
     KINGSTON, ONT           CANADA

WRIGHT, J R
     3110 SO. 27TH STREET
     LA CROSSE, WISC         54601

MICHAEL, L A
     52 BYPASS SO.
     LAFAYETTE, IND          47905

VOGEL, F
     52 BYPASS SO.
     LAFAYETTE, IND          47905

MCLANGHLIN, E E
     2345 IRIS
     LAKEWOOD, COLO          80215

MELUSKEY, J T
     920 OLDE HICKORY ROAD
     LANCASTER, PA           17601

CHAIKIN, A J
     9300 GEORGE PALMER HWY
     LANHAM, MD              20034

LYDIKSEN, H W
     2 TOWNS ROAD
     LEVITTOWN, PA           19056

BALL, M J
     159 IDLE HOUR DRIVE
     LEXINGTON, KY           40502
```

540

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
LUKINS, J C
      3519 WILLOWOOD DRIVE
      LEXINGTON, KY               40502

ODESKY, R I
      1536 ALEXANDRIA
      LEXINGTON, KY               40504

OLDE, G L
      3412 BELLEFONTE DRIVE
      LEXINGTON, KY               40502

STEWART, W B
      ROSE STREET
      LEXINGTON, KY

TILTON, B A
      1828 NICHOLASVILLE ROAD
      LEXINGTON, KY               40503

WEHE, H W
      RD 3
      LIGONIER, PA                15658

ELWELL, W G
      7030 STARR
      LINCOLN, NEBR               68505

KROENCKE, T H
      29-01 BORDEN AVENUE
      LONG, N.Y.                  11101

VAUGHAN, N C
      3424 WILSHIRE BLVD
      LOS ANGELES, CAL            90005

FOERSTER, C S
      112 VALLECITOS WAY
      LOS CATOS, CAL              95030

GOESCH, G W
      17304 ZINA AVENUE
      LOS GATOS, CAL              95030

WEAVER, S E
      120 BRIARWOOD WAY
      LOS GATOS, CAL              95030

LIGON, H H

      LOTT, TEX                   76656
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
JONES, J L
     3214 RADIANCE ROAD
     LOUISVILLE, KY          40220

SCEARCE, W A
     2440 GRINSTEAD DRIVE
     LOUISVILLE, KY          40204

WILLIAMSON, R T
     2500 STEPHEN ROAD
     LOUISVILLE, KY          40214

WOOD, P C
     30 GLEVELLYN ROAD
     LOWELL, MASS            01852

MOORE, J C
     402 ORANGE STREET
     MADISON, FLA            32340

FODOR, J E
     ENGINEERING BUILDING
     MADISON, WIS

HERTEL, E S
     29 DUNN AVENUE
     MANGATUCK, CONN         06770

TRACY, M G
     RED 3, ROCK ROAD
     MANSFIELD, OHIO         44903

SCARAMOZZINO, P J
     297 BOSTON AVENUE
     MEDFORD, MASS           02155

MACNAUGHTON, B B
     5050 POPLAR
     MEMPHIS, TENN           38117

TVEDT, R G
     MPHS ST UNIV ADM BLD 122
     MEMPHIS, TENN           38111

KEITH, J H
     10820 S.W. 52 DRIVE
     MIAMI, FLA

BLACKNEY, W C
     TS&D, ARC, 2020 BLDG DOW CHM
     MIDLAND, MICH
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
JONAS, C R
          2667 N. UNION ROAD
          MIDLAND, MICH                48640

SMITH, R L
          710 AIRFIELD LANE
          MIDLAND, MICH                48640

BRIGGS, D R
          4306 BEDFORD
          MIDLAND, TEX                 79701

SIMS, D L
          3516 W. SHANDON
          MIDLAND, TEX                 79701

FOLLAND, G H

          MILFORD, MICH                48042

PARKER, C D

          MILFORD, MICH                48042

WRIGHT, E J
          MILTON TURNPIKE
          MILTON, N.Y.                 12547

RAAB, P V
          1201 S. SECOND STREET
          MILWAUKEE, WISC              53204

WIGDAHL, A B
          1201 S. SECOND STREET
          MILWAUKEE, WISC              53204

CORNELL, R L
          4137 BEARD AVENUE SOUTH
          MINNEAPOLIS, MINN            55410

JOHNSON, J D
          6023 PENN AVENUE SO.
          MINNEAPOLIS, MINN            55419

LAFONTAINE, J C
          5824 NEVADA AVENUE NO.
          MINNEAPOLIS, MINN            55428

BROWN, L W
          P.O. BOX 2328
          MOBILE, ALA                  36601
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

MILLER, H W
630 RIDGEWAY COURT
MONROE, OHIO 45050

ARMBRUSTER, L F

MONTGOMERY, W VA 25136

BRANAGAN, R E
2442 TRENTON AVENUE
MONTREAL, QUEBEC

FRASER, W C
36 DOBIE
MONTREAL, QUE

LEGER, R
2442 TRENTON AVENUE
MONTREAL, QUE CANADA

SWAIN, P J
620 CHESTER STREET
MONTREAL, QUE CANADA

LEBLANC, M A
6980-14TH AVE. APT 4
MONTREAL, QUE CANADA

RITTER, T L

MOUNT, OHIO 43050

LYNCH, S A
8028 VAN BUREN
MUNSTER, IND 46321

WARNER, G
8033 LINDEN
MUNSTER, IND 46321

GLENN, J S
40 STARK STREET
NASHUA, N.H. 03060

HUMPHREY, M E
803 PAINTER AVENUE
NATRONA, PA S 15065

SHERMER, D A
275 WINCHESTER AVENUE
NEW HAVEN, CONN 06504

LIST OF REGISTRANTS
SEPTEMBER COMMON MEETING
SEPTEMBER 6-7-8, 1967
CINCINNATI, OHIO

CAPLAN, F L 215 ROSE HILL AVE NEW ROCHELLE, N.Y.	10804
BARR, S 222 BROADWEL NEW YORK, N.Y.	10038
BOSCHAN, C 144 EAST 24 ST NEW YORK, N.Y.	10010
CRUMB, H F 33 LIBERTY ST NEW YORK, N.Y.	10045
FELICE, L 1341 BALCOM AVENUE NEW YORK, N.Y.	10461
HUSSEIN, H A 415 E. 64TH STREET NEW YORK, N.Y.	10021
KENNY, A D 401 EAST 74TH STREET NEW YORK, N.Y.	10021
MARTIN, W E USN UNDERWATER SOUND LAB NEW YORK, N.Y.	09560
MCCUSKER, P A 4254 CARPENTER AVENUE NEW YORK, N.Y.	10466
PETERS, C D 527 RIVERSIDE DRIVE 6A NEW YORK, N.Y.	10027
SAMUELS, L B 1271 AVE OF AMERICAS NEW YORK, N.Y.	10020
STACKE, W B 241 6TH AVENUE NEW YORK, N.Y.	10014
STANSBURY, J C 2 PARK AVENUE NEW YORK, N.Y.	10016

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
MATTATALL, G L
     47 FRASER COURT
     NEWCASTLE, N.B.

FITZPATRICK, E D
     ILLINOIS STATE UNIVERSITY
     NORMAL, ILL                61761

PORTER, C B
     ILLINOIS STATE UNIVERSITY
     NORMAL, ILL                61761

JONES, H V
     524 1/2 S. UNIVERSITY BLVD
     NORMAN, OKLA               73069

MAPPUS, J H
     BOX 5207
     NORTH, S.C. STON           29406

RISPOLI, L M
     P.O. BOX 5207
     NORTH, S.C. STON           29406

WEBER, D L
     1400 SHERIDAN ROAD
     NORTH, ILL  O              60064

GWILLIAM, J C
     16 ELM STREET
     NORWICH, N.Y.              13815

MCMINN, C S
     14 HILLVIEW DRIVE
     NORWICH, N.Y.              13815

BRADY, J J
     4620 FOREST AVE
     NORWOOD, OHIO              45212

GLOSTER, A S
     P.O. BOX 117
     OAK RIDGE, TENN            37830

LANE, S E
     3405 N.W. 40TH
     OKLAHOMA, OKLA             73102

BYTHER, T E
     31 OAK ST
     OLD TOWN, ME               04468
```

541

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

JENSEN, E L 4660 SO. 60TH AVENUE OMAHA, NEBR	68117
SCHEMMER, J A 4660 SO. 60TH AVENUE OMAHA, NEBR	68117
SHARP, E A CREIGHTON UNIVERSITY OMAHA, NEBR	68131
CORDING, B L 2705 DELLWOOD DR ORLANDO, FLA	32806
WESTERHAM, E A 1042 N. JEFFERSON OTTWMUA, IOWA	52501
PEARSON, L W 1001 JEFFERSON OXFORD, MISS	38655
WINDHAM, C E 213 SIVLEY OXFORD, MISS	38655
ENYEDY, G AUBURN ROAD PAINESVILLE, OHIO	44077
GENTILE, J F AUBURN ROAD PAINESVILLE, OHIO	44077
CLARK, A G 132 DAVIS STREET PAINTED, N.Y.	14870
PARISIAN, J E 522 W. CHEMUNG STREET PAINTED, N.Y.	14870
WYNN, W E 15 ASH PARK, ILL	60466
GABBERT, D A 2117 INDIANA STREET PARKERSBURG, W. VA	

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
HORTON, M H
     4800 OAK GROVE DRIVE
     PASADENA, CAL            91103

DEAKIN, G R
     302 MOUNTAIN DRIVE
     PEARISBURG, VA           24134

HERWITZ, P S
     LAKEVIEW AVENUE W.
     PEEKSKILL, N.Y.          10566

ABNER, J R
     5904 SEWELL RD
     PENSACOLA, FLA           32504

HILLIER, W J
     1616 WALNUT STREET
     PHILADELPHIA, PA         19103

LIVEZEY, W C
     1608 WALNUT STREET
     PHILADELPHIA, PA         19103

MCILVAIN, D R
     1528 WALNUT STREET
     PHILADELPHIA, PA         19102

NEMETH, L E
     54 ROSSITER AVENUE
     PHOENIXVILLE, PA         19460

BURROWS, W A

     PITTSBURGH, PA           15225

GOLIER, R T
     TWO GATEWAY CENTER
     PITTSBURGH, PA           15222

HAYWARD, A P
     435 6TH AVENUE
     PITTSBURGH, PA           15219

ROESSING, K W
     4028 IMPALA DRIVE
     PITTSBURGH, PA           15239

YOUSSEF, K E
     1441 SMITHFIELD STREET
     PITTSBURGH, PA           15222
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

VAN NESS, V V
2 ROCKLEDGE ROAD
PLEASANTVILLE, N.Y. 10570

SCHILDER, M
2077 SIERRA ROAD
PLYMOUTH, PA G 19462

KOLLER, E B
570 ST. JOHNS ROAD
POINTE, QUE CANADA

COLE, C T
106 KATAHDIN DR
POLAND, OHIO 44514

HOFFMAN, B A
P.O. BOX 3621
PORTLAND, ORE 97208

JONES, L W
P.O. BOX 3621
PORTLAND, ORE 97208

DICOSTANZO, J A
P.O. BOX 390
POUGHKEEPSIE, N.Y. 12602

MONJEAU, G P
31 PASTURE LANE
POUGHKEEPSIE, N.Y. 12603

STEELE, L
18 ANTHONY DRIVE
POUGHKEEPSIE, N.Y. 12601

HOFFMAN, L L
GUGGENHEIM LABS
PRINCETON, N.J. 08240

WALL, D D
39 EVERGREEN CIRCLE
PRINCETON, N.J. 08540

WOODROW, P J
50 WASHINGTON ROAD
PRINCETON, N.J. 08540

DWYER, J R
9321 E. 84TH TERR
RAYTOWN, MO 64138

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

VERBRUGGE, M G
RR 4
RENSSELAER, IND 47978

JAGTIANI, H J
P.O. BOX 2-AC
RICHMOND, VA 23205

GARDNER, D S
211 WALNUT STREET
RIDGEWOOD, N.J. 07450

FELLER, G G
1402 10TH AVENUE S.E.
ROCHESTER, MINN 55901

MUELLER, J D
HIGHWAY 52 NORTH
ROCHESTER, MINN 55901

LAMPTON, G B
161 FAIRWAY CIRCLE
ROCK, S.C. 29730

SAUNDERS, A F
80 RANGE HILL DRIVE
ROCKVILLE, CONN 06066

GARGANO, H M
1901 CHAPMAN AVENUE
ROCKVILLE, MD 20852

STRAUSS, W T
1901 CHAPMAN AVENUE
ROCKVILLE, MD 20852

KLEIN, D R
145 LOWRYS LANE
ROSEMONT, PA 19010

FAERBER, R B
1266 AVALON DRIVE
SAN JOSE, CAL 95125

LANDWEHR, M E
MONTEREY & COTTLE ROADS
SAN JOSE, CAL 95030

LESTER, G
1232 CHATEAU DRIVE
SAN JOSE, CAL 95120

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

SHOUSE, J W
596 SOUTH 10TH
SAN JOSE, CAL 95112

WOLF, G L
6684 LANDERWOOD LANE
SAN JOSE, CAL 95120

BAILEY, D C
MONTEREY AND COTTLE ROAD
SAN JOSE, CALIF

BARMORE, D R
4780 SNOW DRIVE
SAN JOSE, CALIF 95111

ARTHUR, A J
3270 CABRILLO AVE
SANTA, CALIF 95051

GREEN, H A
194 CECIL STREET
SARNIA, ONT CANADA

SOLE, W E
1466 EGMOND DRIVE
SARNIA, ONT CANADA

WESTON, D J

SARNIA, ONT CANADA

BURNS, R A
ALGOMA STEEL CORP LTD
SAULT, ONT RIE CANADA

OWEN, J J
P.O. BOX 570
SAVANNAH, GA 31402

HAGUE, M T
31 KENT
SCARSDALE, N.Y.

POLISHOOK, B H
105 WAVERLY ROAD
SCARTDALE, N.Y.

BALLENTINE, J D
547 PAIGIE STREET
SCHENECTADY, N.Y. 12307

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

RANDALL, R F 826 SOUTH TRAVIS SHERMAN, TEX	75090
MATHEWS, W M 308 MARSHALL DRIVE SHILLINGTON, PA	19607
LEACH, U H 11101 INWOOD AVENUE SILVER, MD	20902
POWELL, J O 1327 CATHERWOOD DRIVE SOUTH, IND	46614
GABRIEL, R F SOUTH ORANGE AVENUE SOUTH, N.J.	07079
GERMANN, SOUTH ORANGE AVENUE SOUTH, N.J.	07079
LUNEBURG, I RFD1 SOUTHBRIDGE, MASS	01550
ANDERSON, B K 20 SOUTH ROAD SOUTHINGTON, CONN	
CAMPBELL, M 2240 SOUTH LONE PINE SPRINGFIELD, MO	65804
CLARK, L A 3520 W. ROUNTREE SPRINGFIELD, MO	65804
KEMP, E V 4820 URBANA ROAD SPRINGFIELD, OHIO	45502
WALKER, R P 3418 KNOX STREET ST. JOSEPH, MICH	49085
BURGGRABE, W F 1400 SOUTH THIRD ST ST. LOUIS, MO	63166

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

DEWEY, G C
    11444 LACKLAND ROAD
    ST. LOUIS, MO 63141

EATON, T J
    400 WASHINGTON AVENUE
    ST. LOUIS, MO 63102

GRAY, W C
    11444 LACKLAND ROAD
    ST. LOUIS, MC 63141

QUAYLE, B B
    6924 RAVENSCROFT
    ST. LOUIS, MO 63123

RODENBECK, R
    1400 SOUTH THIRD STREET
    ST. LOUIS, MO 63166

SCHODITSCH, G F
    1700 SOUTH SECOND STREET
    ST. LOUIS, MO 63177

MIKO, D E
    32 MARK
    ST. MARYS, PA 15857

IMBERTSON, J R
    2189 DOSWELL AVENUE
    ST. PAUL, MINN 55108

FORTUNE, F C
    19394 GULFSTREAM DRIVE
    TEQUESTA, FLA

YAMANAKA, J H
    11815 S.W. FONNER STREET
    TIGARD, ORE 97223

REES, W A
    2310 FOXLEY ROAD
    TIMONIUM, MD 21093

DANZEISEN, L A
    WOODVILLE ROAD
    TOLEDO, OHIO 43601

MICHALOWSKI, E W
    293 HAMILTON AVENUE
    TONAWANDA, N.Y. 14150

553

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
PERFETTE, B
     669 EVERGREEN DRIVE
     TONOWANDA, N.Y.          14150

GINGERICH, D F
     5011 W. 26TH STREET
     TOPEKA, KANS             66614

HUGH, G M
     53 HOLMESDALE CRES
     TORONTO, ONT             CANADA

ROSS, R D
     COMPUTER CENTER
     UNIVERSITY, MISS         38677

ROTH, R W
     TAYLOR UNIVERSITY
     UPLAND, IND              46989

VERITY, A F
     114 MEYER AVENUE
     VALLEY, N.Y.             11580

DECK, J C
     661 CHESTNUT STREET
     VALPARAISO, IND.         46383

PHILLIPS, R W
     CLOVE ROAD
     VERBANK, N.Y.

WILLARD, L B

     WAHPETON, N.D.           58075

FANUELE, V L
     12 WILDWOOD DRIVE
     WAPPINGERS, N.Y.         12590

LOGUE, W E
     DIETZ ROAD
     WARREN, OHIO             44482

PONIKVAR, H E
     DIETZ ROAD
     WARREN, OHIO             44482

BRASKAMP, B
     1111 CONNECTICUT AVE N.W.
     WASHINGTON, D.C.         20036
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

```
PAULSEN, J E
     1200 17TH STREET N.W.
     WASHINGTON, D.C.           20036

WEGNER, R D
     1300 E. STREET N.W.
     WASHINGTON, D.C.

ROHR, N R
     RT. 1
     WASHINGTON, W VA           26181

BAILIE, D L
     455 C STREET
     WASHOUGAL, WASH            98671

GOODRUM, D L
     RR 1 BOX 460
     WATEVLIET, MICH

RAPP, K L
     RR 1
     WAVERLY, OHIO              45601

STEGINA, F J
     475 ELM STREET
     WEST, CONN                 06516

ROWE, L V
     5 BEECHNUT DRIVE
     WEST, OHIO                 45383

KRAMER, P H
     901 EVERNIA STREET
     WEST, FLA  EACH            33401

SNAILER, R J
     53 LACE LANE
     WESTBURY, N.Y.             11590

NARDONE, A A
     166 HIGH STREET
     WESTERLY, R.I.             02891

BURGESON, J W
     220 EAST UNION ST
     WHEATON, ILL               60187

MCCALL, E H
     2676 ROTH PLACE
     WHITE, MINN AKE            55110
```

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

CLOSMAN, S
112 EAST POST ROAD
WHITE, N.Y.

EDWARDS, R A
112 E. POST ROAD
WHITE, N.Y. 10601

FULLAN, D J
112 E. POST ROAD
WHITE, N.Y.

MANIKOWSKI, P R
112 E. POST ROAD
WHITE, N.Y.

MARKS, M
112 E. POST ROAD
WHITE, N.Y. 10601

METEER, R C
252 COUNTY CENTER ROAD
WHITE, N.Y. 10603

SMITH, A P
112 EAST POST ROAD
WHITE, N.Y.

STAUTNER, J F
112 EAST POST ROAD
WHITE, N.Y. 10601

TENNISON, R D
112 E. POST ROAD
WHITE, N.Y. 10601

HART, T M
23 GARRAHAN
WILKES-BARRE, PA 18702

LAMPNE, D J
16 RANCH TRAIL ROAD
WILLIAMSVILLE, N.Y. 14221

FINCH, D G
3316 CROSS COUNTRY DRIVE
WILMINGTON, DEL 19803

RISSOLO, L R
3326 CROSSCOUNTRY DRIVE
WILMINGTON, DEL 19803

LIST OF REGISTRANTS

SEPTEMBER COMMON MEETING

SEPTEMBER 6-7-8, 1967

CINCINNATI, OHIO

LACHNIET, W M
HAMILTON STANDARD
WINDSOR, CONN

SCHULTZ, T E
316 ANITA DRIVE
WINSTON-SALEM, N.C.   27104

BLATCHLEY, C G
300 BENT RD
WYNCOTE, PA   19095

MANLEY, R A
1 FEDERAL STREET
YONKERS, N.Y.   10702

FORSTROM, R W
19401 DORAL COURT
YORBA, CAL

GROFT, G E
COUNTRY CLUB MANOR APT J2
YORK, OA   17403

FOWLER, W G
3730 MEADOWBROOK DRIVE
ZANESVILLE, OHIO   43701

IMLAY, C E
1045 RICHEY ROAD
ZANESVILLE, OHIO   43701

NOVAK, M J
834 EPPLEY AVENUE
ZANESVILLE, OHIO   43701

SHEPHERD, K M
1211 FEDERAL AVENUE
ZANESVILLE, OHIO   43701