

***** DRAFT *****

A-link CIO Adapter

Preliminary External Reference Specification



Roseville Networks Division

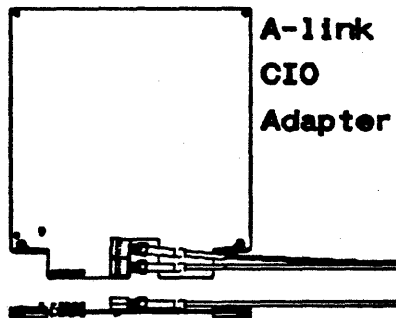
Location Code: 52-5528

Project Number: 27111A

February 11, 1987

This Revision: Feb 12, 1987

Tom Keaveny 1-786-5620



*** HP Confidential ***

Copyright © 1987 HEWLETT-PACKARD COMPANY

PREFACE

The HP27111A Alink CIO Adapter External Reference Specification document is still in a preliminary version, and, to some extent, incomplete.

The intent of the document is to provide a source of information for internal HP customers to aid them in understanding the HP27111A for their particular application, whether it be product marketing support or diagnostic integration.

All information in this document is *SUBJECT TO CHANGE*. Any changes in the HP27111A's external specifications will be noted for those who are on the document distribution list. The update will appear as either errata or a revised ERS depending upon the nature and extent of the modifications.

At release, the following information should still be considered to be *VERY PRELIMINARY* and are currently under review:

Section 4: Electrical Specifications
Section 6: Field Diagnostics
Section 7: Theory of Operation

Comments and questions should be directed to:

Tom Keaveny
Roseville Networks Division
8000 Foothills Blvd.
Roseville, CA 95678
Mailstop: R3NF

1-786-5620

PREFACE (continued)

CURRENT DISTRIBUTION

Distribution of the current version of the HP27111A ERS/IMS is to those listed below under the "distribution" heading. Those under the "notification" heading will receive an HPDESK message indicating that the document is available.

Further copies of this document may be made available by contacting:

Tom Keaveny
Roseville Networks Division
8000 Foothills Blvd.
Roseville, CA 95678
Mailstop: R3NF
HPDESK: HP5200/02

Distribution:

| | |
|-------------------|-----------|
| ABE, STEVE | HP5200/04 |
| ALEXANDER, TOM | HP4700/UX |
| BOULDIN, DANNY | HP0100/20 |
| BROMLEY, DAVE | HP1900/01 |
| BROWN, HAL | HPG200/11 |
| BROWN, JERRY | HP5200/07 |
| BUCKTHAL, ROGER | HP4800/03 |
| BURGER, STEPHEN | HPG200/13 |
| CANTU, IGNACIO | HP5200/04 |
| CHANG, PAUL | HPG200/13 |
| CHEUNG, NANCY | HP5200/07 |
| CONRY, KEVIN | HPG200/05 |
| CRAIG, BARBARA | HPG200/XX |
| DUNN, PAUL | HP1900/UX |
| ESPLIN, GREG | HP4800/03 |
| GREEN, JUDY | HP5200/07 |
| HOLLAND, ED | HPG200/14 |
| HUGINS, CHRIS | HPG200/13 |
| JAHR, STEVE | HPG200/05 |
| JENKINS, LYNN | HP5200/04 |
| LECHTENBERG, DAVE | HP4000/UX |
| MATTHEWS, RANDY | HP4800/03 |
| MEANS, DAVID | HP4700/UX |
| MEIER, (hpfc11) | HP4000/UX |
| MILLER, DAVID | HPG200/08 |
| NAKAMURA, TOM | HP5200/07 |
| NELSON, MARVIN | HP4800/03 |
| NISHIME, KAZ | HPG200/13 |
| OUYE, DARRYL | HPG200/13 |
| PEPIN, ROGER | HPG200/04 |
| SANDERS, JIM | HP4800/10 |
| SCHREMPP, MIKE | HPG200/14 |
| SILL, GALE | HP5200/07 |
| SKOKAN, STAN | HP1900/01 |
| SONTAG, JOHN | HP4700/UX |
| SORENSEN, HANS | HP0100/20 |
| SPOHN, NORRAE | HP4800/03 |
| TANI, JON | HP1900/UX |

VOGELSBERG, GARY

HP4800/MK

Notification:

| | |
|-------------------|-----------|
| BORG, MIKE | HPG200/04 |
| BRUBAKER, RAY | HP5200/02 |
| CAMPBELL, DUNCAN | HP5200/07 |
| CHATTERTON, CRAIG | HPG200/14 |
| CHIOCHIOS, JIM | HPG200/13 |
| CONNER, DON | HP5200/07 |
| CORCORAN, JOHN | HP1900/01 |
| GOULD, RON | HP0080/01 |
| GUBITZ, GARY | HP5200/07 |
| HARRIS, DAVE | HP5200/07 |
| HO, GARY | HP4700/UX |
| HODAPP, MARK | HP4000/UX |
| HUNT, CRAIG | HP2414/04 |
| KENYON, DAVE | HP5200/07 |
| KIZZIER, CHRIS | HPG200/11 |
| KONDOFF, AL | HPG200/08 |
| KUNTZ, DAVID | HP5200/07 |
| LIU, MARTIN | HP4700/UX |
| LUTGARDO, ALBERTO | HPG200/08 |
| MCANALLY, GARY | HP5200/02 |
| MCCONNELL, KAY | HP5200/04 |
| MORRIS, HOWARD | HPG200/08 |
| PEARSON, RICH | HPG200/13 |
| SCHOMMER, NICK | HP0100/06 |
| SNOW, DAVE | HPG200/13 |
| STEADMAN, HOWARD | HP4700/L3 |
| SWISLEY, RANDI | HP5200/07 |
| WALKER, NATE | HP0100/06 |
| WILKES, JOHN | HP1900/UX |
| WU, FRANCIS | HP0100/06 |

FUTURE RELEASE DISTRUBUTION

Details on revised ERS/IMS releases will be made through HPDESK, to both those on the "distribution" or "notification" (CC) lists.

To inquire about or receive an ERS/IMS simply reply to the HPDESK message detailing the release as a form of acknowledgement.

INTRODUCTION

SECTION

1

The A-link is a high speed serial fiber optic data link which may be used by the host in conjunction with high speed disks or disk clusters as well as for host to host data transfers. Target "absolute" speed of the link itself is initially 80 megabits, with intended growth path to 120 megabits. In terms of byte transfers, and accounting for link efficiency, the previous figures translate to transfer rates of 5.0 Mbytes and 7.5 Mbytes, respectively.

Physically, the card consists of two interfaces: backplane and frontplane. The A-link card will support existing CIO backplanes, and will also have the facilities to support an enhanced CIO backplane which implements additional error and fault detection functions. The frontplane has both an optical transmitter and receiver, providing a full duplex communication link to the external device. This remote device may be located as far as 1 km from the host in standard operation.

Functionally, the card is capable of processing up to 64 current outstanding requests from the host. The card is also capable of responding to request from the remote device. In general, the A-link card may also serve as either a master or a slave to the remote node, depending upon the configuration and the application.

Actual data transmission from card to remote node is controlled by hierarchical levels of protocol, much of which is implemented in hardware to achieve high performance. From the channel or backplane the card responds to the Channel I/O protocol implementation of CS-80 (essentially levels 4-0) and reformats the data according to A-link protocol (levels 3-0).

The card is equipped with a full set of diagnostic/self test code and associated circuitry.

SECTIONAL OVERVIEW

The topics covered in the following sections are as follows:

- | | |
|-------------------------------------|--|
| INTRODUCTION (this section!) | Brief description of card and its function followed by one or two statements highlighting what will be covered in each major section of the document |
| REFERENCE | A listing of helpful documents in better understanding the A-link adapter along with a list of conventions and a glossary of commonly used terms |
| SPECIFICATIONS | A list of electrical, physical and functional specifications as seen by the backplane (CIO) and front plane (A-link) users. |
| FUNCTIONAL OVERVIEW | A functional block diagram description of the CIO Adapter is given followed by an example of how each block operates during a disc read. |
| FIRMWARE DESCRIPTION | A description of firmware as seen by the host system and the |

Introduction

remote device. Supported requests and programming modes are identified.

USER INTEGRATION

A description of how the A-link CIO Adapter is integrated into a host system and how it is intended to be maintained

THEORY of OPERATION

A description of the internal circuitual behaviour of the A-link CIO adapter.

CHANGES and UPDATES

This document now serves as both an ERS and an Internal Maintenance Specification (IMS). With this change, some of the technical information deleted in the last release has been added again.

Three more releases are planned for this document.

(1) Production Prototype ERS/IMS - (2) Manufacturing Release (MR1 rev) ERS/IMS;- (3) Manufacturing Release (MR2 rev) ERS/IMS - *if necessary*

If there is any information not present in this document that is required by any of those on the distribution list, now is the time to make your requests!

Updates and Changes for this release:

| | |
|---|----------------------|
| <i>NEW</i> Theory of Operation Specifications | (section 7) |
| <i>NEW</i> Appendix Material | (appendices) |
| <i>REVISED</i> | (sections 1,2,4,5,6) |

REQUIRED DOCUMENTS

The following documents are required to exist on the document account or must be accessible in order for the ERS to be printed:

| | |
|--------------------|--------------------|
| ALBOX000.TAK.ALINK | ALBOX412 |
| ALBOX001.TAK.ALINK | ALBOX501 |
| ALBOX002.TAK.ALINK | ALBOX502 |
| ALBOX003.TAK.ALINK | ALBOX503 |
| ALBOX004.TAK.ALINK | ALBOX504 |
| ALBOX005.TAK.ALINK | ALBOX601 |
| ALBOX006.TAK.ALINK | ALBOX602 |
| ALBOX007.TAK.ALINK | ALBOX603 |
| ALBOX008.TAK.ALINK | ALCIOCOD |
| ALBOX009.TAK.ALINK | ALCIOPIN.TAK.ALINK |
| ALBOX010.TAK.ALINK | ALEQDEV.TAK.ALINK |
| ALBOX011.TAK.ALINK | ALFIG999 |
| ALBOX012.TAK.ALINK | ALFIGURE.TAK.ALINK |
| ALBOX013.TAK.ALINK | ALI186BW.TAK.ALINK |
| ALBOX014.TAK.ALINK | ALI186CS.TAK.ALINK |
| ALBOX015.TAK.ALINK | ALI186CT.TAK.ALINK |
| ALBOX016.TAK.ALINK | ALI186GC.TAK.ALINK |
| ALBOX017.TAK.ALINK | ALI186GS.TAK.ALINK |
| ALBOX018.TAK.ALINK | ALI186IN.TAK.ALINK |
| ALBOX019.TAK.ALINK | ALI186PO.TAK.ALINK |
| ALBOX020.TAK.ALINK | ALIAD2LD |
| ALBOX021.TAK.ALINK | ALIARBSG |
| ALBOX022.TAK.ALINK | ALIARRST |
| ALBOX023.TAK.ALINK | ALIAUTO.TAK.ALINK |
| ALBOX024.TAK.ALINK | ALIBARST |
| ALBOX025.TAK.ALINK | ALIBOOL |
| ALBOX026.TAK.ALINK | ALIBYT86.TAK.ALINK |
| ALBOX027.TAK.ALINK | ALIBYTSW.TAK.ALINK |
| ALBOX028.TAK.ALINK | ALICIOPA |
| ALBOX029.TAK.ALINK | ALICIORG.TAK.ALINK |
| ALBOX030.TAK.ALINK | ALICIOSG |
| ALBOX101 | ALIDCDC |
| ALBOX201 | ALIDCLBK.TAK.ALINK |
| ALBOX202 | ALIDCLS2.TAK.ALINK |
| ALBOX302 | ALIDCLST |
| ALBOX401 | ALIERCOD.TAK.ALINK |
| ALBOX402 | ALIFORST |
| ALBOX403 | ALIGRDTB.TAK.ALINK |
| ALBOX404 | ALIGWRTB.TAK.ALINK |
| ALBOX405 | ALIHYST |
| ALBOX406 | ALILEDMP.TAK.ALINK |
| ALBOX407 | ALIMCHOP.TAK.ALINK |
| ALBOX408 | ALIMSAPP.TAK.ALINK |
| ALBOX409 | ALIMSARO |
| ALBOX410 | ALIMSBA0 |
| ALBOX411 | ALIMSFOO |

Introduction

ALIMSMIO
ALIMSPC0
ALIMSPR0
ALIMSRG2.TAK.ALINK
ALINCLUD.TAK.ALINK
ALINK001.TAK.ALINK
ALINK002.TAK.ALINK
ALINK003.TAK.ALINK
ALINK004.TAK.ALINK
ALINK005.DOC.ALINK
ALINK006.TAK.ALINK
ALINK007.TAK.ALINK
ALINKACC.TAK.ALINK
ALINKDI2.TAK.ALINK
ALINKFIG
ALINKFLC.TAK.ALINK
ALINKIMP.TAK.ALINK
ALINKMAT.TAK.ALINK
ALINKPRE.TAK.ALINK
ALINKREQ.TAK.ALINK
ALINKSTC.TAK.ALINK
ALINKSTI
ALINKSTI.TAK.ALINK
ALIPAUT2
ALIPAUTO.TAK.ALINK
ALIPBFHD.TAK.ALINK
ALIPCBLK.TAK.ALINK
ALIPCRST
ALIPINIT.TAK.ALINK
ALIPP1PR
ALIPRREG.TAK.ALINK
ALIPRRST
ALIPSTAT
ALIPWRSH.TAK.ALINK
ALIR000
ALIR001
ALIR002
ALIR003
ALIR004
ALIR005
ALIR006
ALIR007
ALIR008
ALIR009
ALIR010
ALIR011
ALIR012
ALIR013
ALIR014
ALIR015
ALIR016
ALIR017
ALIR018
ALIR019

ALIR020
ALIR022
ALIR023
ALIR024
ALIR025
ALIR026
ALIR027
ALIR028
ALIR029
ALIR030
ALIR031
ALIR032
ALIR033
ALIR034
ALIR035
ALIRAMCT
ALIRXCHR
ALITXCHR
APPENDX1.IN.MANU
ARBLP4S1.TAK.ALINK
ARBLP4X2
ARBNT04
AXTCLKEN
AXTDMARD
AXTDMAWR
AXTGRD
AXTGWR
AXTLATCH
AXTLINK
AXTPCNTL
AXTPRORD
AXTRAMRD
AXTRAMWR
AXTRLR
AXTROM
BITS.TAK.ALINK
BLOCFIGS
BULLETS.IN.MANU
BYTES.TAK.ALINK
CALAYOUT
CDSTAT.DOC.ALINK
CHANHINT.TAK.ALINK
CHANOP.TAK.ALINK
CIOLBK.DOC
COVER1.IN.MANU
DEVIDY.DOC.ALINK
DEVLBK.DOC
END2COL
END2COL.TABLE.MANU
END3COL.TABLE.MANU
END4COL.TABLE.MANU
END5COL.MANU.ALINK
END5COL.TAK.ALINK
ENDBULL.IN.MANU

ENDLCOV. IN. MANU
ENDNOTE. IN. MANU
ENDPREF. IN. MANU
ENDVER1. TABLE. MANU
ENDVER2. TABLE. MANU
ENDVER3. TABLE. MANU
ENDVER4. TABLE. MANU
ENDVER40. TAK. ALINK
ENDVER5. TABLE. MANU
ENDVER6. TABLE. MANU
ENDVER7. TABLE. MANU
ENDVER8. TABLE. MANU
ENDVER9. TABLE. MANU
EXTLBK. DOC
FIBERKEY. TAK. ALINK
FLOWFIG1
FLOWFIGS
FORMAT. IN. MANU
IDYBLK. DOC. ALINK
IMAGE. IN. MANU
INTLBK. DOC
LABCOV. IN. MANU
LEVEL1. IN. MANU
LEVEL1X. IN. MANU
LEVEL2. IN. MANU
LEVEL3. IN. MANU
LEVEL3X. IN. MANU
LEVEL4. IN. MANU
M4LESS2. TAK. ALINK
MANSTART. IN. MANU
MEMPAL7. TAK. ALINK
MID2COL. TABLE. MANU
MID3COL. TABLE. MANU
MID4COL. TABLE. MANU

MID5COL. MANU. ALINK
MID5COL. TAK. ALINK
NOTE. IN. MANU
PASSCTL5
PFWPAL6. TAK. ALINK
PREFACE. IN. MANU
PREFCNTD. IN. MANU
PWRFAIL5. TAK. ALINK
REQBLK. DOC. ALINK
RTSAES. DOC. ALINK
SANS3COL. TABLE. MANU
SANS4COL. TABLE. MANU
SANS5CL2. TAK. ALINK
SANS5COL. MANU. ALINK
SECTION1. IN. MANU
TAB2COL. TABLE. MANU
TAB3COL. TABLE. MANU
TAB4COL. TABLE. MANU
TAB5COL. MANU. ALINK
TAB5COL. TAK. ALINK
TXSTAT. DOC. ALINK
VER1. TABLE. MANU
VER2. TABLE. MANU
VER3. TABLE. MANU
VER4. TABLE. MANU
VER40. TAK. ALINK
VER5. TABLE. MANU
VER6. TABLE. MANU
VER7. TABLE. MANU
VER8. TABLE. MANU
VER9. TABLE. MANU
WARNING. IN. MANU
WTCAEK. DOC. ALINK

Introduction

The following is a description of the locations of the various illustrations in the document

Only illustrations contained in boxes are noted...

| Include | Figure_File | Figure | draw or hpdraw file | Figure_File | Figure |
|----------|-------------|------------------|------------------------|----------------------|--------|
| albox000 | alfig999 | board_blank_ind | alfig101 | | |
| albox001 | alfig999 | equal_mode | alfig300 | | |
| albox002 | alfig999 | nmi_jumper_block | alfig301 | | |
| | alfig999 | alink_board | alfig303 | alfig004:alink_board | |
| albox003 | alfig999 | integrity | alfig302 | alinklib:crd_block | |
| albox016 | alinkfig | rtr_rts | alinklib | | |
| | alinkfig | rts_rtr | alinklib | | |
| albox017 | alinkfig | switch_rd | alinklib | | |
| albox014 | alinkfig | switch_wd | alinklib | | |
| albox004 | alinkfig | crd_block | alinklib | | |
| albox007 | alinkfig | pc_block | alinklib | | |
| albox008 | alinkfig | fo_block | alinklib | | |
| albox009 | alinkfig | proc_block | alinklib | | |
| albox005 | alinkfig | ba_block | alinklib | | |
| albox015 | alinkfig | wd | alinklib | | |
| albox018 | alinkfig | rd | alinklib | | |
| albox013 | alinkfig | clc | alinklib | | |
| albox006 | alinkfig | dma_block | alinklib | | |
| albox010 | alinkfig | arbiter | alinklib | | |
| albox011 | alinkfig | chan_prog | alinklib | | |
| albox012 | alinkfig | link_prog | alinklib | | |
| albox019 | alinkflo | remcon_non | diagrams | | |
| albox020 | alinkflo | remcon_non_2 | diagrams | | |
| albox021 | alinkflo | selftest | diagrams | | |
| albox022 | alinkflo | rawmode | diagrams | | |
| albox023 | alinkflo | power_chek | diagrams | | |
| albox024 | alinkflo | cabling | diagrams | | |
| albox025 | alinkflo | led_check | diagrams | | |
| albox026 | alinkflo | fluxchk_i | diagrams | | |
| albox027 | alinkflo | fluxchk_o | diagrams | | |
| albox028 | alinkflo | ext_rmtlbk | diagrams | | |
| albox029 | alinkflo | ext_piglbk | diagrams | | |
| albox030 | alinkflo | chnng_mode | diagrams | | |
| albox101 | timefig? | t_link | timing | | |
| albox201 | timefig? | t_pro_rd | timing | | |
| albox202 | timefig? | t_pro_wr | timing | | |
| albox302 | blocfigs | i_source | blocdiag | | |
| albox401 | timefig? | t_latch | timing | | |
| albox402 | timefig? | t_rom | timing | | |
| albox403 | timefig? | t_ram_rd | timing | | |
| albox404 | timefig? | t_ram_wr | timing | | |
| albox405 | timefig? | t_grd | timing | | |
| albox406 | timefig? | t_gwr | timing | | |
| albox407 | timefig? | t_dma_wr | timing | | |
| albox408 | timefig? | t_dma_rd | timing | | |
| albox409 | timefig? | t_clken | timing | | |

| | | |
|----------|---------------------|----------|
| albox410 | timefig? t_pas_cnt1 | timing |
| albox411 | timefig? t_pas_inta | timing |
| albox412 | timefig? t_rlr | timing |
| albox501 | timefig? t_cp_rd_2 | timing |
| albox502 | timefig? t_cp_wr_2 | timing |
| albox503 | timefig? t_pas_m_rd | timing |
| albox504 | timefig? t_pas_m_wr | timing |
| albox601 | blocfigs b_dc_buck | blocdiag |
| albox602 | blocfigs b_dc_dc | blocdiag |
| albox603 | blocfigs b_dc_dc_i | blocdiag |

The **REFERENCE** section provides references, both for supplemental reading and for commonly used terms to aid the reader in better understanding the A-Link CIO Adapter product.

RELATED DOCUMENTS

A-Link Protocol Document
by Greg Dolkas, Randy Haagens

HP-CIO Standard Document
by Roseville Networks Division

A-Link Protocol Chip (PRONTO) Hardware Description Document
by Bill Martin, Kurt Chan, Vince Cavanna

Passport VLSI External Reference Specifications (ERS)
by Mike Leclere

Jupiter External Reference Specifications
by HP Labs

Passport Programmer's Guide
by Bill Haffey

Channel I/O A-link Preliminary Support Plan
by Nancy Cheung

Amux Hardware ERS
by Earl Bergquist

Amux Diagnostics
by Marvin Nelson

Eagle/Amux CS/80 INSTRUCTION SET Programming Manual
by NorRae Spohn

CONVENTIONS

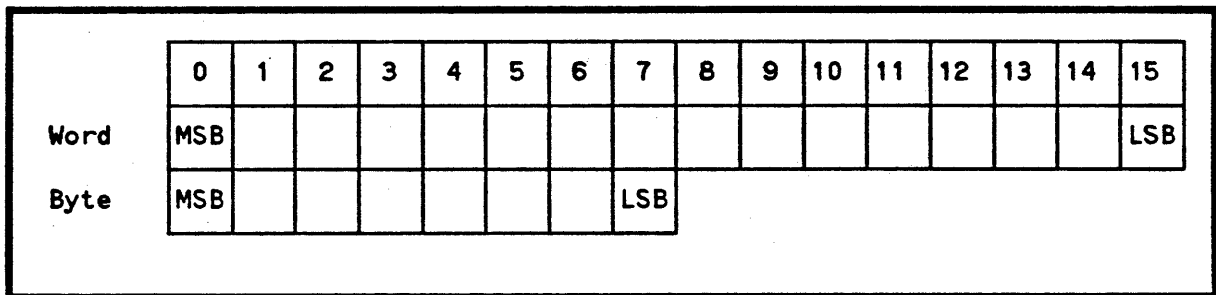
This document has defined standards for interpreting bit numbering, byte numbering, and logic notation as described in the following.

Bit and Byte Notations

Bit Numbering.

Bit numbering follows the rule that the most significant bit (MSB) of any unit is assigned bit position 0. Bits are numbered sequentially from most to least significant.

Any exceptions to this Bit Numbering scheme will be explicitly notated!

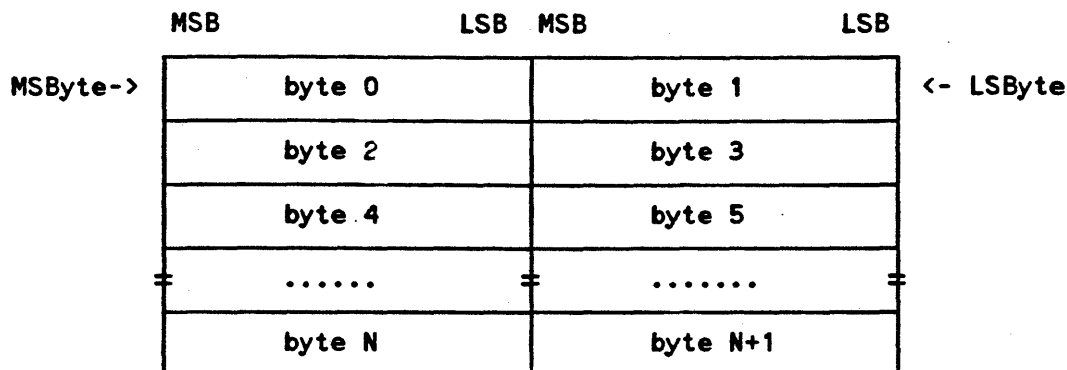


Byte vs. Word.

- Byte* 8 bit unit of information
- word* 16 bit unit of information
- upper byte* most significant 8 bits of a word unit
 -OR-
 most significant byte (MSByte) of a word unit.
- lower byte* least significant 8 bits of a word unit
 -OR-
 least significant byte (LSByte) of a word unit.

Byte Numbering.

The same rule holds true for bytes of data arranged in block form The most significant byte (MSByte) is numbered 0 and the remaining bytes are numbered sequentially from most to least significant.

**Logic Notation****Assertion vs. Deassertion.**

Assertion of a signal indicates that a signal has gone from a false to a true state.

Deassertion of a signal indicates that a signal has gone from a true to a false state.

Assertion vs. Deassertion.

All signals referred to in the hardware descriptions will contain a suffix that indicates the logical "L" or "H" level at which they are asserted.

signal+ indicates that the signal is asserted at a logical H level

signal- indicates that the signal is asserted at a logical L level

Logical 0 vs. Logical 1.

Use of logical "0" and "1" states ordinarily indicates whether a signal is asserted or deasserted

logic 0 is equivalent to Deasserted

logic 1 is equivalent to Asserted

Reference

Active vs. Passive.

Output signals (or busses of signals) are either in an active or passive state.

active indicates that the output is driven to an asserted or deasserted state

passive indicates that the output is neither asserted nor deasserted (often the case when an output has been placed in a *high impedance* state)

Schematic Notation

Some of the figures contain logic symbols following the anticipated enhancements to ANSI Y32.14. No attempt will be made to explain the nature of the symbols used here, but a good reference is Texas Instruments' Bipolar Microcomputer Components Data Book mentioned in the Reference section.

SIGNAL NAMING.

All *LOGIC* signals mentioned in the theory of operation section follow the format:

| | | |
|------------------|--|---|
| Logic Signal ::= | <SIGNAL NAME> <BUS VALUES(optional)> <POLARITY> </Logic Signal(optional for multimode signals)> | AD AD[0:15] AD[0:15]+ BREQ+/DTACK- |
| Bus Values ::= | <[><MOST #(opt)<:LEAST #(opt)><]> | [][5][5:7] |
| Polarity ::= | <{+,-}> | POS_TRUE+ NEG_TRUE- |

All *ANALOG* signal names follow the format:

| | | |
|-------------------|---------------|-----|
| Analog Signal ::= | <SIGNAL NAME> | VCC |
|-------------------|---------------|-----|

BOOLEAN OPERATORS.

The basic boolean operator notations are shown in the following table.

| | |
|-------------------|---|
| OR | + |
| OR (Abel Format) | # |
| AND | * |
| NOT | ! |
| NOT (Abel Format) | / |
| XOR | % |

[alibool]

REFERENCE DESIGNATORS.

All reference designators are printed in *Elite Italic*.

E.g., *U20, U15-8, C23*.

GLOSSARY

| | |
|-------------------------------|---|
| CS-80 | Command Set 80. A command set used to access and control mass storage devices |
| FRONTPLANE | the optical fiber interface, configuration jumpers, and status indicators |
| BACKPLANE | the channel and support circuitry interface |
| NMI | A Non Maskable Interrupt which is optionally driven by the A-Link adapter to the channel across the backplane |
| DMA | direct memory access |
| PAL | Programmable Array Logic |
| CHANNEL | a physical entity that consists of the I/O channel and I/O channel adapter, consisting of data, control and addressing busses |
| VIRTUAL CIRCUIT COMMUNICATION | the process of multiplexing many circuit operations onto a smaller number of available circuits, creating the appearance to anything requesting a circuit that a "dedicated" rather than a "virtual" circuit exists |
| CONTROL BYTE | a byte used to identify layer 1 CONTROL information, typically indicating such status as link idle, frame end delimiter, and variable length frame follows |
| INFORMATION BYTE | a byte used to identify layer 1 DATA information |
| DEVICE | the entity that is connected to the remote end of the optical fiber |
| LOGCHANNEL | a virtual data path that is established between the I/O channel adapter and an I/O card |
| SUBCHANNEL | a path to/from an I/O card, through the I/O channel, and the I/O channel adapter |
| JUPITER | high speed parallel/serial converter for layer 1 |
| PASSPORT | backplane adapter |
| PRONTO | protocol controller for layer 2 |
| LAYERED ARCHITECTURE | hierarchical architecture that describes a method of implementing data transactions across a data communications link |
| FRAME | Alink's layer 2 message unit |

| | |
|---------------------------|--|
| INFORMATION FRAME | layer 2 data frame containing layer 3 data or control information |
| CONTROL FRAME | layer 2 control frame |
| HEADER <i>Layer 2</i> | accompanying every frame, the header communicates protocol control and status information back and forth across the link |
| HEADER <i>Layer 3</i> | accompanying every level 3 message segment, level 3 headers contain virtual circuit number, message type, and other level 3 control information |
| CRC | cyclic redundancy check |
| HP-CIO | Hewlett Packard Channel-I/O |
| MESSAGE | basic unit of information passed by level 3 |
| EOF | level 2 control code indicating end of frame |
| VLF | level 2 control code indicating a Variable Length Information Frame |
| IDLE CODE | level 2 control code indicating link is idle |
| BYTE SYNCHRONIZATION CODE | level 2 control code indicating a start of frame |
| DATA CODE | level 2 code type indicating that the following code is of type data |
| LOOPBACK | mode of operation of the CIO Alink card in which front and backplane are disabled and the receiver and transmitter at each end are tied together for testing |
| END OF FRAME | level 1 control byte indicating end of frame |
| IDLE STATE | level 1 control byte indicating that the link is idle |
| | [add infinitum ...] |

FUNCTIONAL OVERVIEW

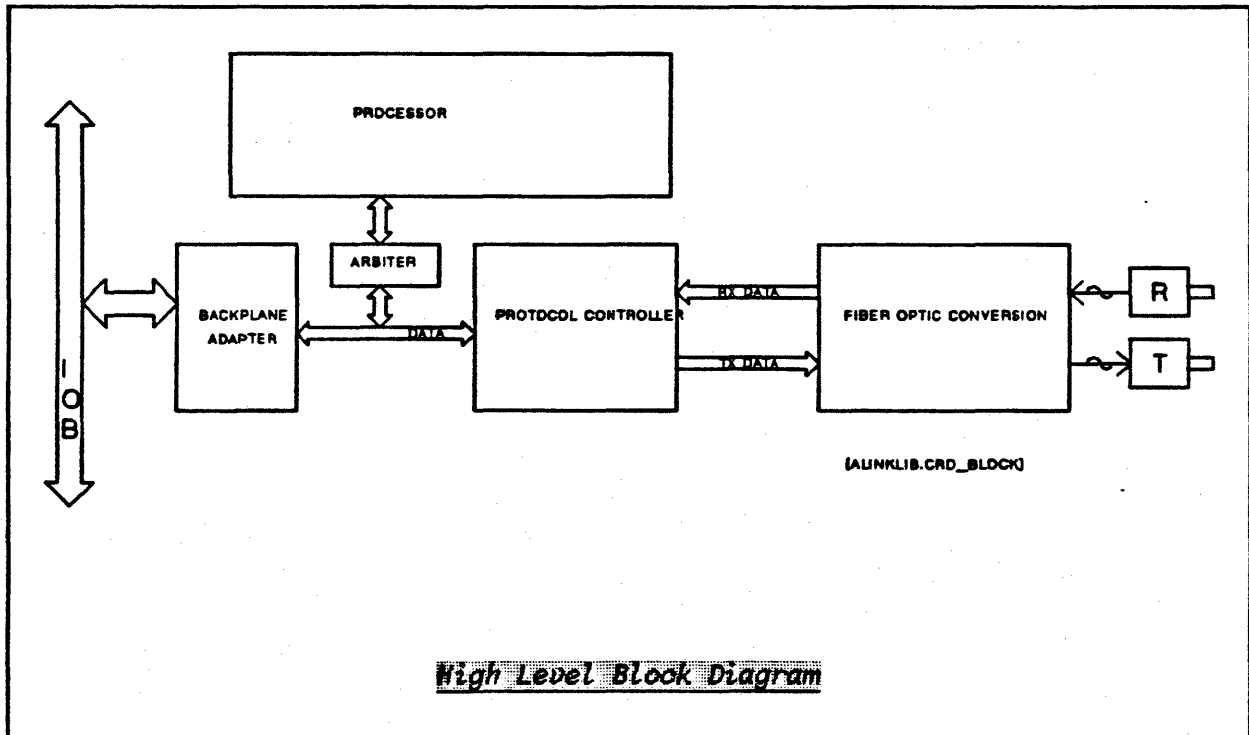
SECTION

3

The functional overview section describes the major functional blocks that comprise the HP27111A as well as presenting a typical scenario for data read and data write operations. Each functional block description highlights the features of the particular block. The scenario examples point out the interaction of the blocks as data moves between frontplane and backplane.

HIGH LEVEL BLOCK DIAGRAM DESCRIPTION

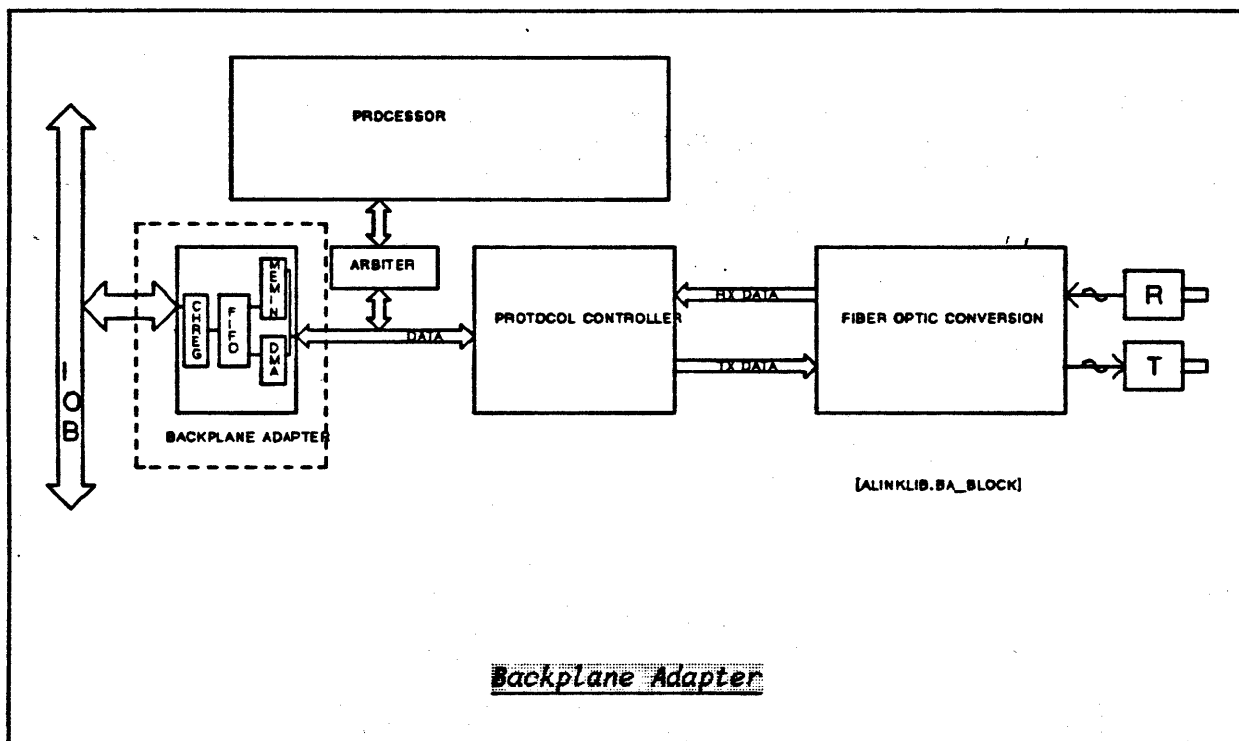
The HP-CIO version of A-link can be divided into four functional areas. The areas consist of the Backplane Adapter(BA), Processor(PB), Protocol Control(PC), and Fiber Optic Conversion(FOC). A simplified black box outline of the relations between these functions is shown in Figure 3.0. Most I/O transactions between backplane and the Fiber Optic frontplane occur by taking the direct route from Backplane Adapter to Protocol Control to the actual Fiber Optic Conversion; most Processor functions can be performed concurrently and independently of the actual data flow.



Backplane Adapter

The Backplane Adapter serves as A-link's interface to the HP-CIO backplane and is the mechanism by which data, control and status information is passed to and from the card across the backplane. To the backplane, the adapter appears as a register oriented device that is selectable through the backplane's addressing mechanism, dependent upon which type of backplane operation is being performed. To increase data throughput, a DMA type "bursting" feature is provided for data transfers to allow the backplane to source or to sink successive elements of data once the adapter's data register has been selected.

Internally, the backplane adapter appears as a memory list based control device and a high speed DMA machine. By memory list based control device, it is meant that the backplane adapter appears as a peripheral that acts upon lists of tasks that are stored in various locations in memory. The backplane adapter is also capable of interrupting the Processor block when certain conditions are detected.



Channel operations.

Channel operations provide the most primitive programatic access to the internal operation of the CIO A-link Adapter from the host computer's perspective. In CIO syntax these are known as level 1 operations.

The nine Channel operations accompanied by a brief description are given below

chanop.tak.alink

| OPERATION | DESCRIPTION |
|------------------------|--|
| Read Data | used to transfer Data from A-Link card to CIO channel through the current active Subchannel |
| Write Data | used to transfer Data from CIO channel to A-Link card through the current active Subchannel |
| Write Order | used to control Subchannels and to connect, control, and destroy Logchannels |
| Write Command | used mainly to connect and destroy Subchannels. |
| Read Status | used mainly to return physical status of A-Link card and its local Subchannel |
| Write Control | used to control the state of the Backplane Adapter |
| Read Sense | used to determine status of the Backplane Adapter and its physical interface to the channel |
| Poll Service Request | A-link responds when it is capable of processing an order or data on its current active subchannel |
| Poll Attention Request | A-link responds when it has been programmed to request a new command from the channel and is able to do so |

Subchannel interactions.

A Subchannel interaction is a series of related channel operations (often referred to as sequences) for a particular subchannel. A Subchannel establishes a path from CIO to the actual Link I/O Data path. A subchannel is prerequisite to transferring data between CIO and the A-link.

A typical subchannel sequence would be the actual establishment of a subchannel on the A-link CIO Adapter for later use by the host in transferring data. This interaction is referred to as a Connect

Functional Overview

Subchannel sequence. The following psuedo code illustrates the sequence of Channel operations that compose the sequence.

```
PROCEDURE Connect_Subchannel( Subchannel_Number : Subchannel_type ) ;
BEGIN
  { form the subchannel connect command code }
  Subchannel_connect := Subchannel_connect + Subchannel_Number ;
  { get sense of I/O card }
  CHANNEL_OPERATION(Read_Sense,A-link_card,Sense_Variable) ;
  IF Sense_Variable[Ready_For_Command] THEN
    BEGIN
      { send the Connect command }
      CHANNEL_OPERATION(Write_Command,A-link_card,Subchannel_Connect);
    END
  ELSE
    BEGIN
      { request notification when the card empties its command register ! }
      CHANNEL_OPERATION(Write_Control,A-link_Card,Request_Attention) ;
      Delay_Until_Attention_Requested ;
      { verify ready for command }
      CHANNEL_OPERATION(Read_Sense,A-link_card,Sense_Variable) ;
      { send the Connect command }
      CHANNEL_OPERATION(Write_Command,A-link_card,Subchannel_Connect);
    END ;
  END ;
END ;
```

Certain Subchannel Interactions are handled automatically by the Backplane Adapter. Remaining interactions are passed on to the Processor Block for further action. The *Connect Subchannel* interaction is handled automatically, for example. *Connect Logchannel* transactions, on the other hand, always require the intervention of the Processor Block.

Supported Subchannel Interactions and Transactions are detailed in the Firmware Description Section. (see Supported Commands and Orders)

Logchannel transactions.

Logchannel transactions are composed of series of subchannel interactions. They provide the A-link CIO Adapter with its multiplexing capabilities. A-link, itself has only one physical duplex link to the outside world. Logchannels are logical replications of this physical link and are structured such that they appear to the host CPU as individual physical A-link connections. They are, in a sense, *virtual data paths* between the Channel and the A-link card.

For more details on Logchannel transactions, consult the Channel I/O Handbook.

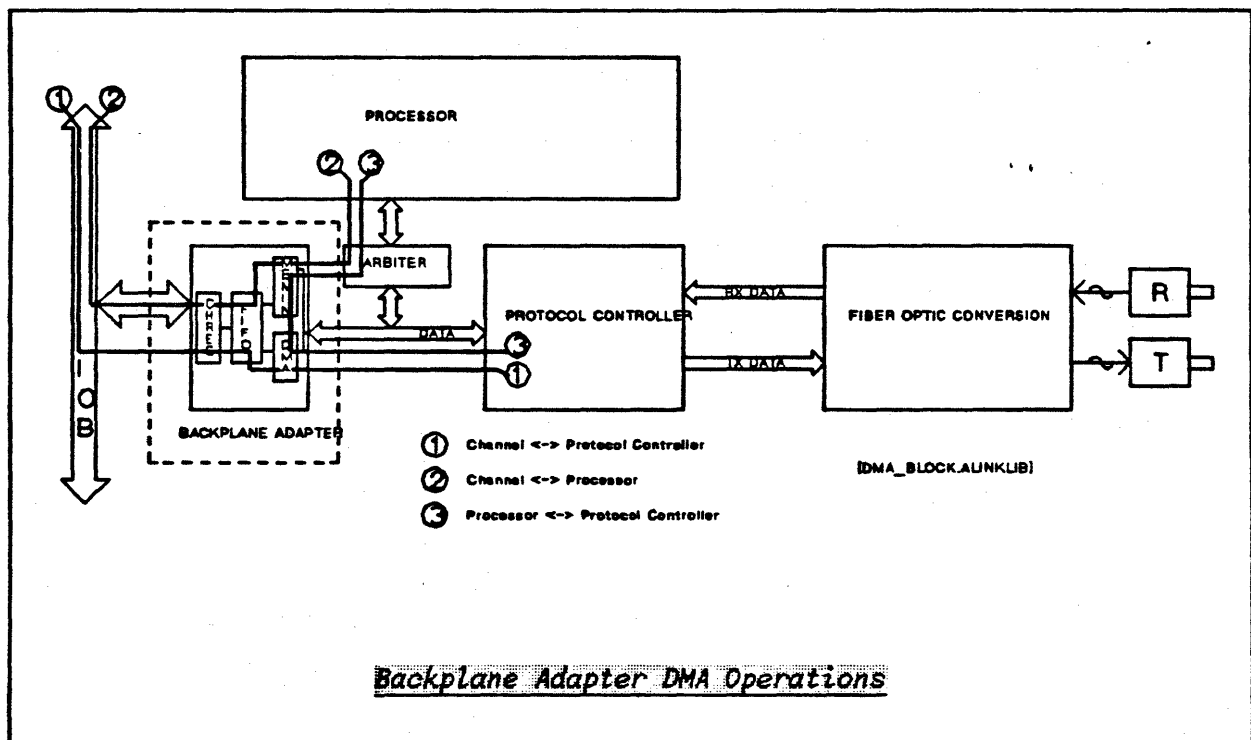
Processor communications.

The Backplane Adapter block communicates to the Processor block through Memory located within the Processor block. Status, data and control information is posted in various lists in Memory according to a predefined data structure.

The Backplane Adapter block interrogates these data structures to determine, for example, which subchannel interactions must be passed on to the Processor block, and which to process automatically. It also determines the *source* and *destination* for data transactions it is instructed to perform.

DMA operations.

DMA operations are used to transfer data between any of the ports accessible to the Backplane Adapter. These three ports are illustrated below.



As mentioned before, the *source* and *destination* ports are determined by examining the lists stored in Memory.

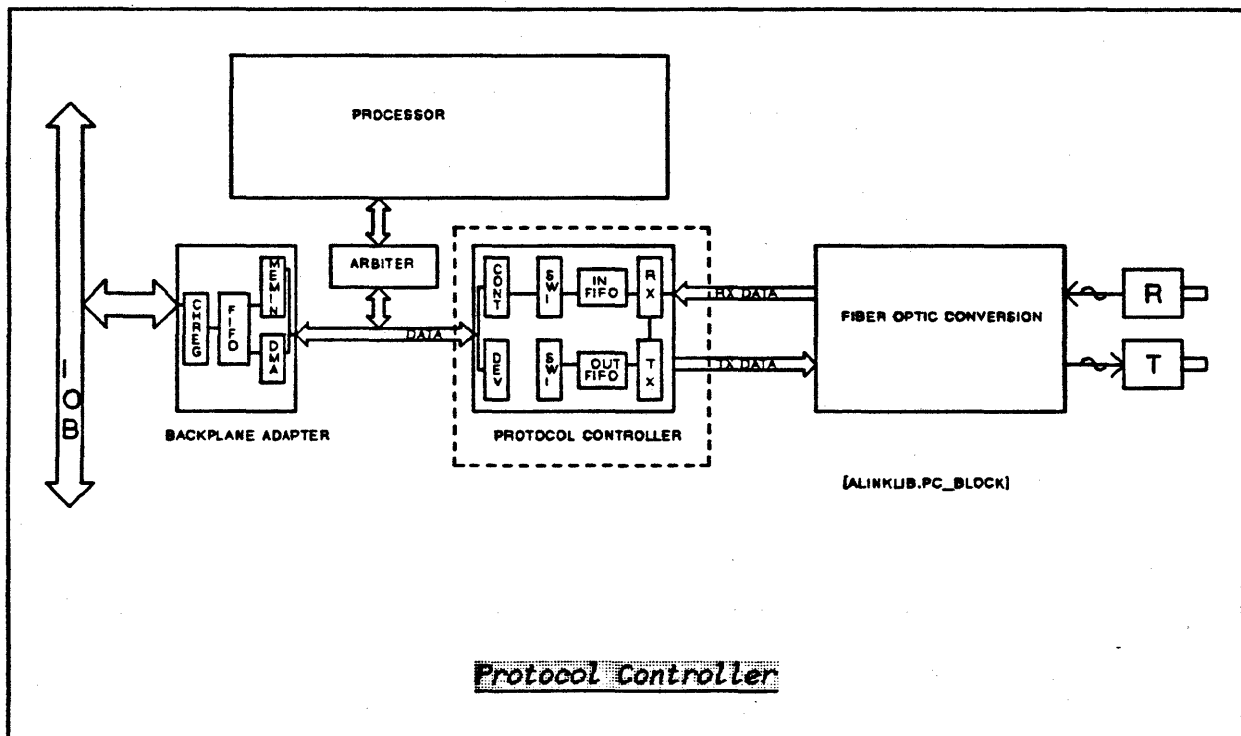
An internal half-duplex FIFO is used to buffer data and help pace data transfers.

Protocol Controller

The Protocol Control section of the A-link card is responsible for transforming data to and from the A-link Layer 2 Protocol format, dependent upon direction of data flow. It consists of a bi-directional half duplex and a full duplex interface which form four ports: Control, Device, Receive and Transmit. A special function, known as the "switch" manages the half duplex interface which is shared by the Control and Device port. Outbound data passes from either the Control or Device port to the Transmit port, where it is formatted. Inbound data, from the Receive port, is stripped of Layer 2 formatting and passes to the Control or Device port. Both inbound and outbound data paths contain a FIFO.

The actual protocol control takes place in the management of data that is sent and received by the Transmit and Receive ports. In the case of data in the Outbound FIFO, the Protocol Control block guarantees that any data loaded into the Outbound FIFO will not be overwritten until it has been correctly received by the remote node, and furthermore, that the Inbound FIFO of the remote node will not be overrun.

For inbound data, the Protocol Control block guarantees that all data has been received in the order it was loaded into the remote's Outbound FIFO, and that any attempts at overrun will generate a Protocol error.



Control Port.

The Control port is used by the Processor block to access control registers internal to the Protocol Control block and to transfer data to and from the link. The registers are used to control and monitor the behaviour of the link and data transfers active on it. Error conditions and recovery measures are managed through this path.

Data transfers via the Control port usually consist of *Header Messages* which are follow the A-link Layer 3 protocol. These Headers are used to setup and execute data transfers between the local A-link CIO Adapter and the remote device.

Device port.

The Device port is a high speed direct access port which handles the majority of I/O traffic on the link. It provides a direct data path to the Inbound or Outbound internal FIFOs depending upon the direction of data transfer.

Status indicators are provided to the Backplane Adapter block to indicate whether the Device port is ready to send or receive data as well as means of identifying when a particular stream of data has reached a termination condition.

Switch.

The Switch is a special feature of the Protocol Control block that allows the Inbound and Outbound data paths to be either manually or automatically pointed to the Device port (Backplane Adapter) or Control port (Processor). Each FIFO (Inbound and Outbound) has its own independent *switch*. When the Switch is operated manually, the Processor explicitly establishes the connection to the I/O data paths by pointing the switch to one of the ports. In automatic mode, each switch toggles between Control and Device port depending upon information previously loaded into the Control port registers by the Processor.

The Switch only affects data transfer paths; it does not inhibit Control port register accesses.

Transmit port.

The Transmit port is used by the Protocol Control block to send data from the Outbound FIFO to the Fiber Optic Conversion block. It also supplies state information to the remote node to allow management of the link (Layer 2 Protocol). Data and state information is packaged in units known as *Control and Information Frames*.

Control Frames contain the current state of the Inbound FIFO as well as link control commands.

Information Frames consist of formatted buffers obtained from the Outbound FIFO. The formatting basically involves prefacing the buffer of data with an *Information Header* which identifies the particular buffer to be sent, and then appending a CRC to the buffer to protect the data buffer's contents.

Two main factors influence whether the Protocol Control block will send an Information Frame out the Transmit port: the ability of the remote node to accept another buffer and the presence of a buffer to be sent. The Protocol Control block can determine the state of the remote node's Inbound FIFO from data received through the Receive port. Control Frames are sent if the block is incapable of sending an Information Frame.

Acknowledgement of outstanding buffers, and transmission retry is also caused by examination of the state of the remote node's Inbound FIFO.

A detailed description of the A-link layer 2 Protocol is found in the A-link Protocol Document.

Functional Overview

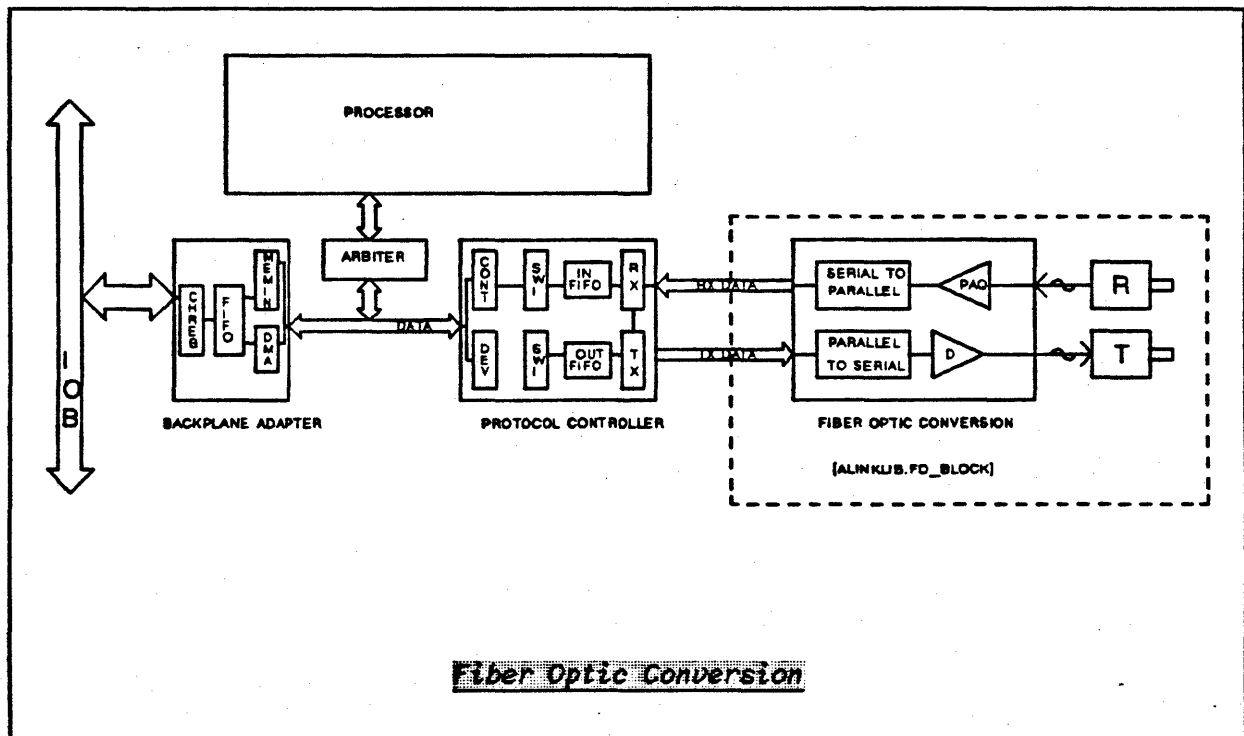
Receive port.

The Receive port takes parallel data from the Fiber Optic Conversion block, processes it, verifies its integrity, and then extracts the essential A-link Protocol Layer 2 information from it. Valid data extracted from *Information Frames* is loaded into the Inbound FIFO. Remote node state information is extracted from *Information and Control Frames* and directs behaviour of the Transmit port as described previously.

The Receive port also monitors the state of the Fiber Optic Conversion block's receiver and provides this information to the Processor block via the Control port.

Fiber Optic Conversion

The Fiber Optic Conversion block provides the conversion between byte-parallel and serial data streams. The block consists of two independent serial to parallel and parallel to serial converters that link the optic transmitter and receiver to the Protocol Control block's Receive and Transmit ports.



Optical receiver.

The Optical receiver converts flux on the input fiber into an analog waveform proportional to the intensity of the incoming flux.

Post Amplifier and Quantizer.

The analog signal from the Optical Receiver must be conditioned before information may be extracted from it. The Post Amplifier and Quantizer block amplifies the low-level analog signal and quantizes it to boolean logical levels. The quantized signal is then passed on to the Serial to Parallel Conversion block.

Serial to parallel conversion.

The Serial to parallel conversion block locks onto the incoming serial waveform and decodes and extracts the original layer 2 synchronization, control and data information, reconstructing it into a parallel format which is then sent to the Protocol Control block.

Illegal codes are flagged when the pattern is transferred to the Protocol Control block. Other status provided includes the presence of activity on the fiber, and the ability to lock onto the serial data stream, all of which are monitored by the Protocol Control block.

Parallel to serial conversion.

Parallel to serial conversion takes parallel data from the Protocol Control block, encodes it, and then serializes the data to prepare it for transmission on the optical fiber.

The encoding operation is required to accommodate some of the electrical characteristics of the analog and optical circuitry and media.

Driver.

The Driver takes the serialized output from the Parallel to Serial Converter and conditions it to meet the input requirements of the Optical Transmitter and the power requirements of the attached fiber.

Optical Transmitter.

The Optical Transmitter transforms the output of the Driver into optical flux energy which then may drive the outbound optic fiber.

Processor

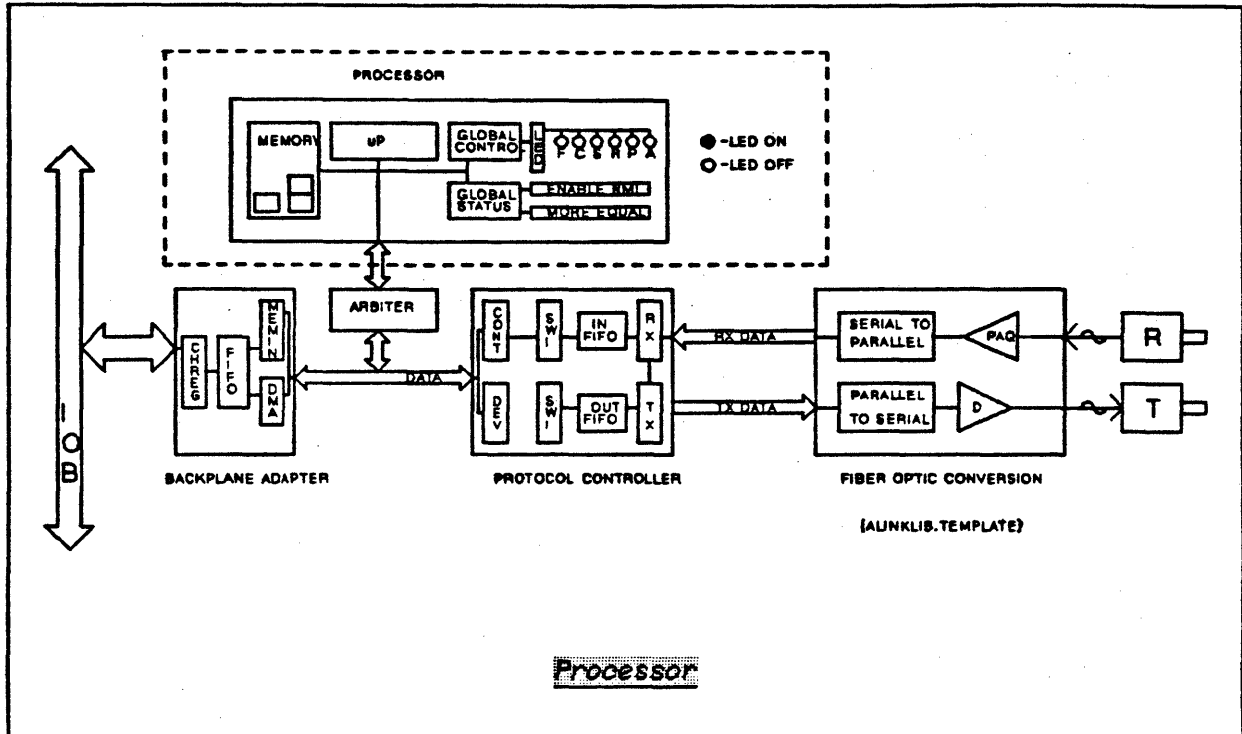
The Processor block's main responsibility is to transform Channel requests (channel operations, subchannel and logchannel transactions) into A-link protocol format; it is essentially the A-link Protocol Layer 3 engine.

Channel requests are managed in conjunction with the Backplane Adapter. The Processor block initiates and processes requests to the remote node that involve the transfer of data through the Protocol Control

Functional Overview

block. Once the Processor block has determined that a requested transfer may begin it gives the go-ahead to the Backplane Adapter and Protocol Control block, and continues its own processing independently.

The Processor block also responds to requests from remote nodes that appear at the Protocol Control block and maintains error, status and control operation of the link.



Link Control/Layer 3 engine.

Link Control/Layer 3 engine transforms a Channel read or write request into the series of steps necessary to communicate to the remote node. Basically, this involves obtaining one of the link's resources and then sending the necessary commands across this resource in order to transfer data.

Where the Channel views the A-link card as multiple I/O devices using Logchannels, A-link uses the concept of *Virtual Circuits* in multiplexing it's front-plane resources.

All data that is transferred on the link must be preceded by this Virtual Circuit identifier in what is known as a *Layer 3 Header*. Information within this Header also indicates what sort of use is intended for the Virtual Circuit resource.

Further details on types and formats of Headers is found in the section on Firmware Description.

Microprocessor/80186.

The Microprocessor/80186 gives the Processor block its processing power. It has access to the Protocol Control block's Control port, the Backplane Adapter through Memory, and Configuration and Status

registers which control and indicate the A-link adapters current state and mode of operation. Firmware, described later in this document, is executed by the 80186.

The translation of Channel request to Command Set 80 (CS-80) format is done by the 80186.

Requests originating from the remote node are handled by the Microprocessor.

The Microprocessor is also responsible for performing the on-card diagnostics and monitoring link behavior characteristics.

Memory.

Memory is the basic communications path between the Processor block and the Backplane Adapter. Certain types of Channel requests and Channel data messages are passed on to the Processor block through the Backplane Adapter into memory. The Processor responds by building lists of tasks for the Backplane Adapter to process. These lists allow the Processor to control the flow of data from the Channel to the DMA interface of the BA.

Memory is also used to maintain the state of outstanding Virtual Circuits. As each stage of the A-link Protocol or phase of a CS-80 transaction is completed, the state of the Virtual Circuit executing that process is updated.

Global Status/Control LED's.

The Global Status/Control LED's block is used by A-link to obtain information regarding card configuration, hardware revision codes, to display link status through LED's and to provide control over modes of operation.

During normal operation, the LED's provide a quick indication of link activity(a), signal quality(s), remote (r), and proper configuration(c).

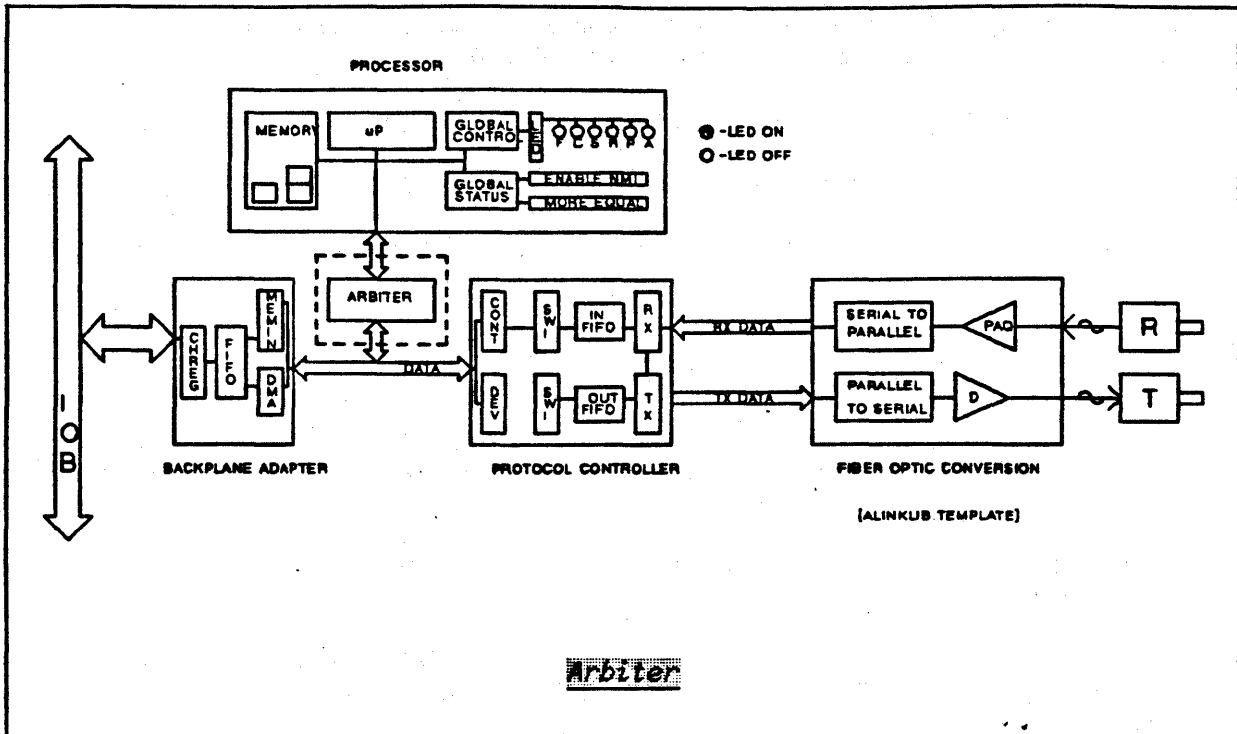
The LED's come in one of two colors, green indicating normal activity, red indicating an error condition.

Details on the significance of each LED are found in the section on Specifications.

Arbiter

The Arbiter block isolates the Processor block's data path from the link data communication's path between the Backplane Adapter and the Protocol Controller. The Arbiter then controls any data accesses that must span both busses, allowing either the Backplane Adapter or the Processor to be effective *master* of both busses.

Functional Overview



Processor to Protocol Controller Control Port.

All I/O operations between the microprocessor and the control port must be initiated through the arbiter. During these operations the Backplane Adapter is prevented from accessing either the Protocol Controller or Processor Memory. It should be noted that card requests that require a great deal of interaction between Processor and Protocol Controller may impede other pending link operations.

Backplane Adapter to Processor Memory.

All Backplane Adapter accesses to Processor Memory must be initiated through the arbiter. During these operations the microprocessor may only execute internally queued instructions (if any are present). Once again card requests that require significant interaction between Backplane Adapter and Processor/Memory may impede other pending link operations. Also, in this case, Processor background processes are slowed.

SAMPLE READ TRANSACTION

Following, is a sample read transaction in which data is transferred from the remote device to the channel. The program sequence for the backplane and frontplane are described and then the effect these sequences have on the A-link CIO adapter is illustrated.

The read transaction performed is a standard *CS-80 Read* sequence with the normal *Command*, *Execution*, and *Report* phases. In brief review of the phases of this sequence, the *Command* phase directs the disc to retrieve data from a particular disc location, the data is transmitted during the *Execution* phase, and the status of the read is returned in the *Report* phase.

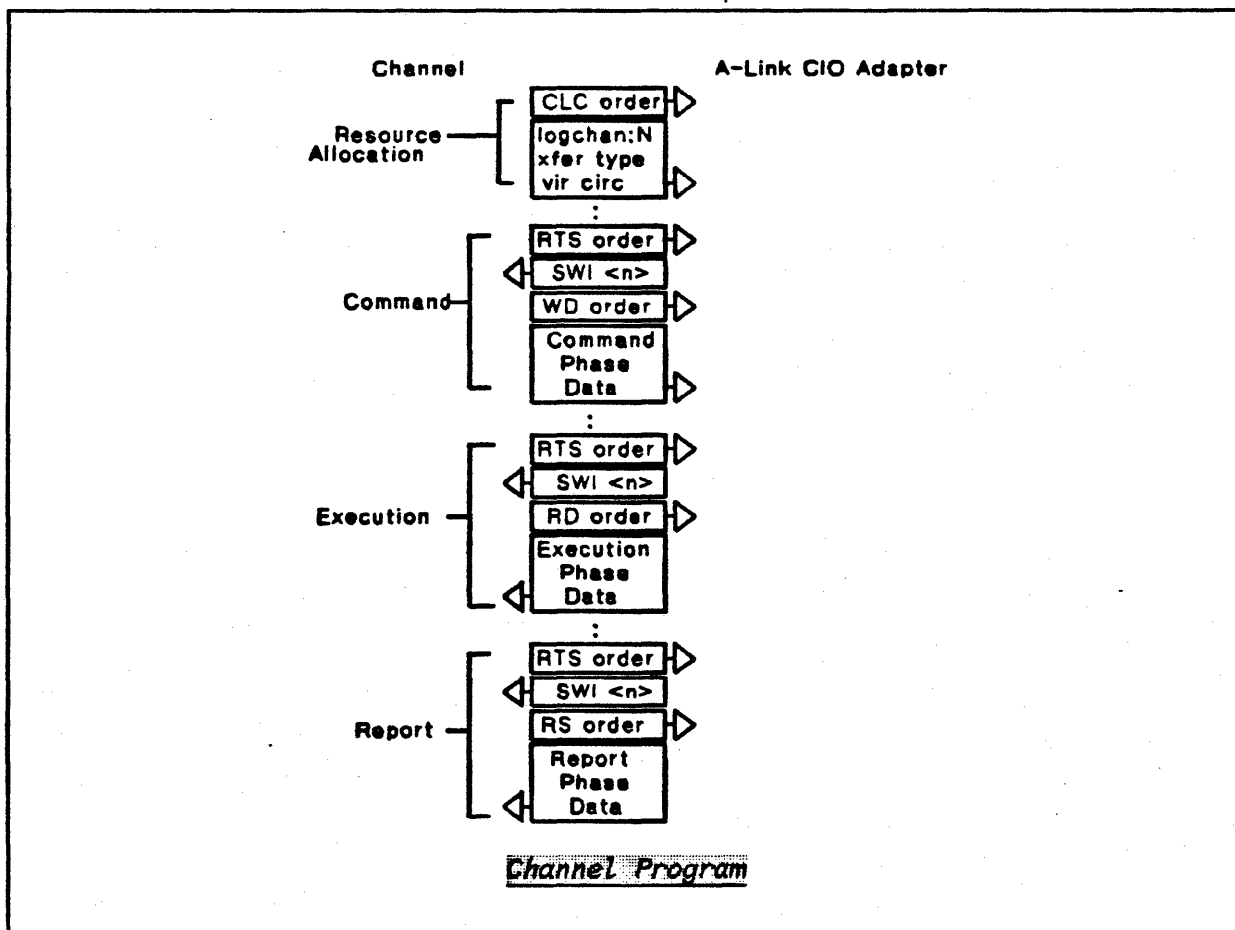
It should be noted that both the card and the link have been initialized prior to the read transaction and that a subchannel path is actively connected on the card.

Also, for the sake of simplicity, only one transaction is assumed to be in effect and the response time of both the channel and the remote device are optimum. (i.e., this is a *STRAIGHTFORWARD* explanation...)

Backplane Program Sequence

The *Backplane Program Sequence* consists of obtaining one of the A-link adapter's logchannel resources followed by the three phases of the CS-80 transaction: *Command*, *Execution*, *Report*.

In each of the CS-80 phases of the sequence, the channel program determines that the card is ready to proceed and then sends an order after which data is either sent to or received from the card.



Functional Overview

Resource Allocation.

The resource that the channel establishes with the card is known as a **Logical Channel** or **Logchannel** as described earlier.

The **Logchannel** is established by the channel performing a **Connect Logchannel Interaction** upon the card. Basically, this interaction consists of a **Connect Logchannel Order** followed by data that indicates the type of **CS-80** transaction to be performed (in our example, a *READ*) as well as the number of the logchannel and the **A-link** virtual circuit that the transaction will be associated with.

Command Phase.

In the **Command Phase**, the channel program determines that it may proceed with its desired logchannel by issuing a **Read Transparent Status Order (RTS)**. Since the card is ready to proceed, it responds with a **Switch to Logchannel #n (SWI)** status message. In **CIO** terminology this is also known as a logchannel break and **Activation**.

The channel program then sends a **Write Data Order** followed by a buffer of data that contains the set of **CS-80** commands that are to be executed by the disc. In this case, the data specifies a disc read as well as disc address and control parameters. The end of this data buffer is tagged to indicate transmission has completed.

Execution Phase.

During the **Execution Phase**, the channel program again issues a **RTS Order** and obtains a status message of **SWI** and passes through the logchannel break.

Since we are performing a *READ* operation on the disc, the channel program sends a **Read Data Order** to the card in order to retrieve the requested data.

Report Phase.

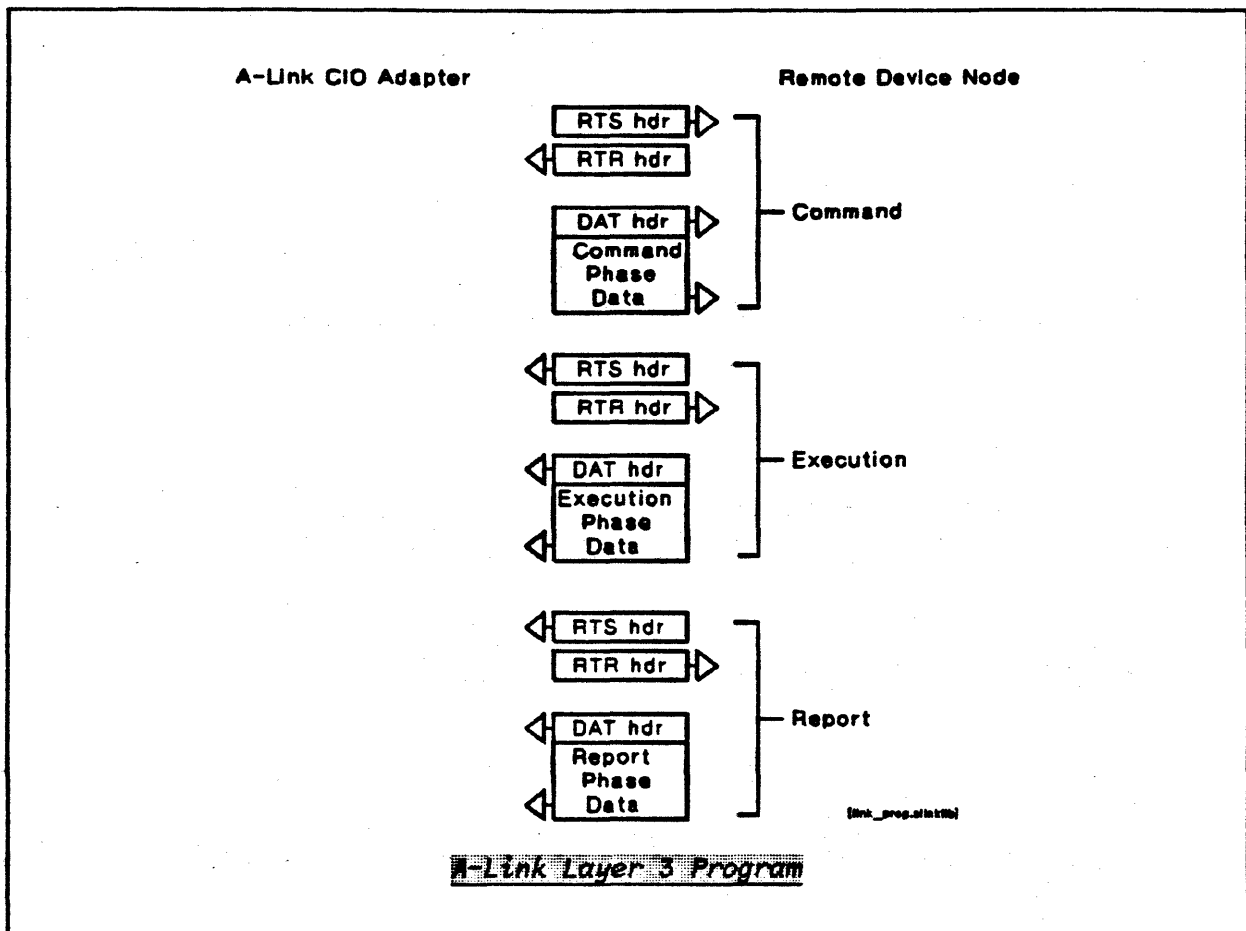
Finally, during the **Report Phase**, the channel program proceeds through one more logchannel break, and then sends a **Read Status order (RS)** to the card in order to verify the results of the **CS-80** transaction.

The **A-link CIO Adapter** is now ready to proceed with further requests.

Frontplane Program Sequence

The **Frontplane Program Sequence** uses the **A-link Layer 3** protocol to structure the three phases of the **CS-80** transaction.

During each phase of a CS-80 transaction, the source of data (local card or remote device) requests the use of a particular virtual circuit. Once the receiver of data indicates that it is ready, the data for that particular phase is transferred.



Resource Allocation.

The frontplane effectively performs a Resource allocation during each phase of the CS-80 transaction. This resource allocation consists of a Request to Send(RTS) / Ready to Receive(RTR) header exchange between source and sink of data.

The RTS header is a standard A-link layer 3 header, and thus indicates the nature of data to follow as well as the virtual circuit to associate the data with.

The RTR header indicates that the data sink is ready to receive a block of data for a particular virtual circuit.

Functional Overview

Command Phase.

In the Command Phase, the A-link CIO Adapter sends an RTS header onto the link indicating that it is about to initiate a CS-80 transaction.

When the remote device returns an RTR header, the Adapter sends out a DAT header followed by the data that forms the CS-80 Command.

Execution Phase.

In the Execution Phase for a *READ* from the remote device, the remote will respond with an RTS header indicating that it is ready to send the execution data. The card will respond with an RTR header which causes the remote device to send a DAT header followed by the data retrieved from the disc.

Report Phase.

In the Report Phase, the remote device initiates an RTS/RTR header exchange, and the card receives a DAT header followed by data that describes the success of the just completed Read.

Functional Block Interaction

The Functional Block Interaction on the A-link CIO Adapter ties the frontplane and backplane program sequences together.

Each of the major functional blocks described previously (BA,PC,PROC,FO,ARBITER) plays a specific role in the process.

To illustrate this block interaction, the data flow during the various phases of the Read Transaction is presented, as well as the cause and effect nature of the action.

Resource Allocation.

When the Channel sends the CLC order to the adapter, the CLC data travels from the I/O Backplane (IOB) to the BA's internal FIFO. The BA discovers that the CLC does not match any of the entries in the lists provided by the PROC block, and therefore gains access to PROC Memory through the Arbiter and transfers the CLC data into it.

The uP is alerted of this new data, parses it, and adds the Logchannel and Virtual Circuit pair to its list of requests. Since we are assuming there are no pending requests, the Processor builds up an RTS Header which includes the new Virtual Circuit. This initial RTS is used to mark the beginning of the *Command Phase*.

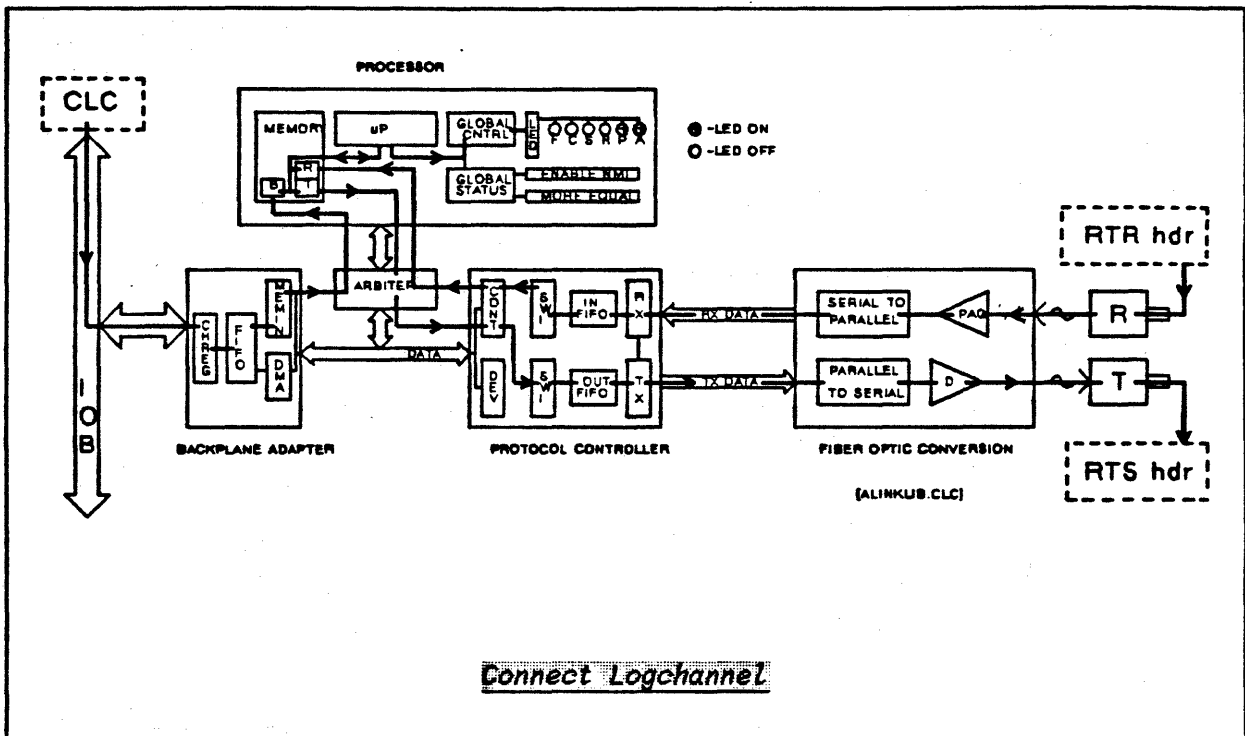
The uP issues a request to the Arbiter to access the Protocol Controller, and proceeds to send the RTS Header out through the Control Port. The RTS passes through the outbound FIFO, has control and CRC information added to it, and is then sent to the Tx portion of the FO block.

The FO Block serializes the data, converts it into a photon stream and sends it out onto the fiber.

The action of sending the RTS Header onto the link (actually, any header) causes the Activity lamp on the adapter to light.

When the RTR Header appears on the fiber, it is converted back to parallel form by the FO Block Rx section and passed on to the Receive Port of the PC block. The PC determines that the data is valid and is the expected buffer in the current sequence. The RTR Header is then placed in the Inbound FIFO.

The PROC retrieves the RTR Header from the PC, checks its Virtual Circuit and thus determines that the remote device is ready to proceed with the *Command Phase*



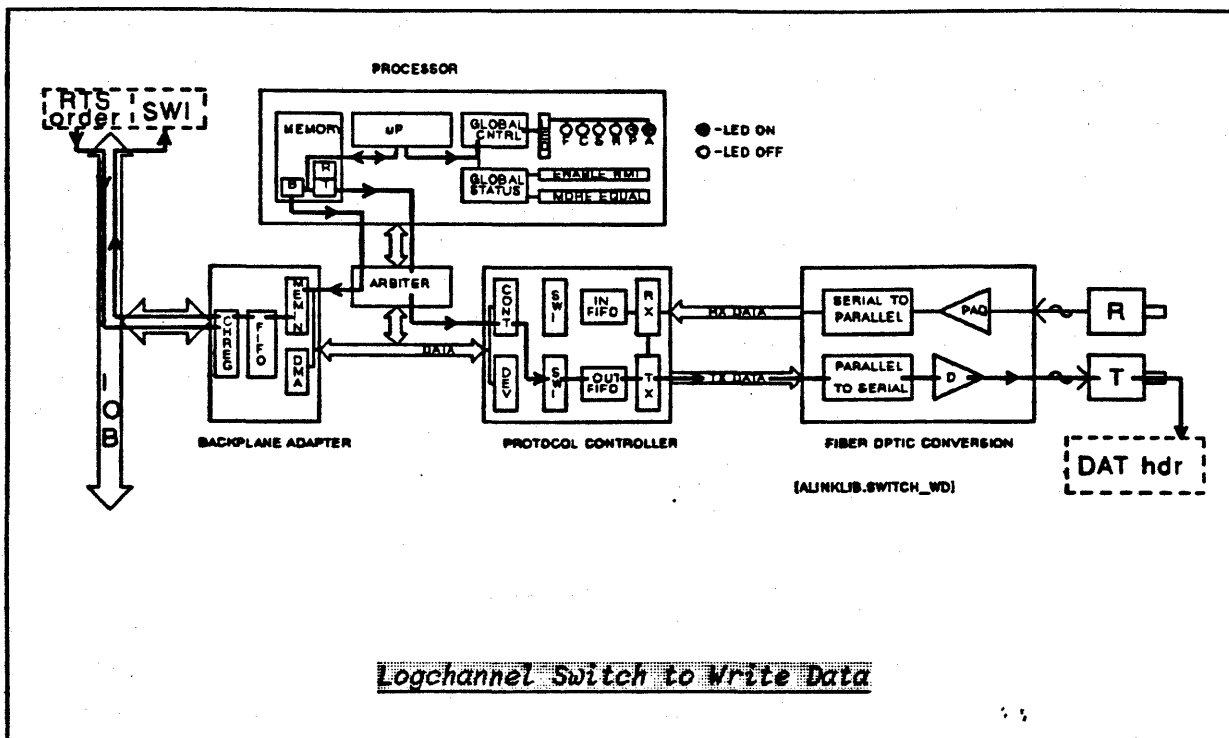
Command Phase.

To the adapter the Command Phase has two basic steps: the Switch and the actual Command Data Transfer.

During the Switch, the adapter prepares the link and the channel for the transfer of command data. In the case of the link, the adapter sends out a DAT Header onto the link via the Arbiter and PC, which indicates to the remote device that the actual Command Data is to follow. Just before sending out the DAT header, the Processor placed the PC in *Transmit Automatic Mode*. This special mode will cause the PC to SWITCH from the Control Port to the Device Port once the header has passed through the Control Port.

For the channel, the Processor programs the Backplane Adapter to respond to a Read Transparent Status (RTS) Order with a SWITCH to Logchannel <n>. It also tells the BA to expect a Write Data order to follow and to transfer the incoming data to the PC.

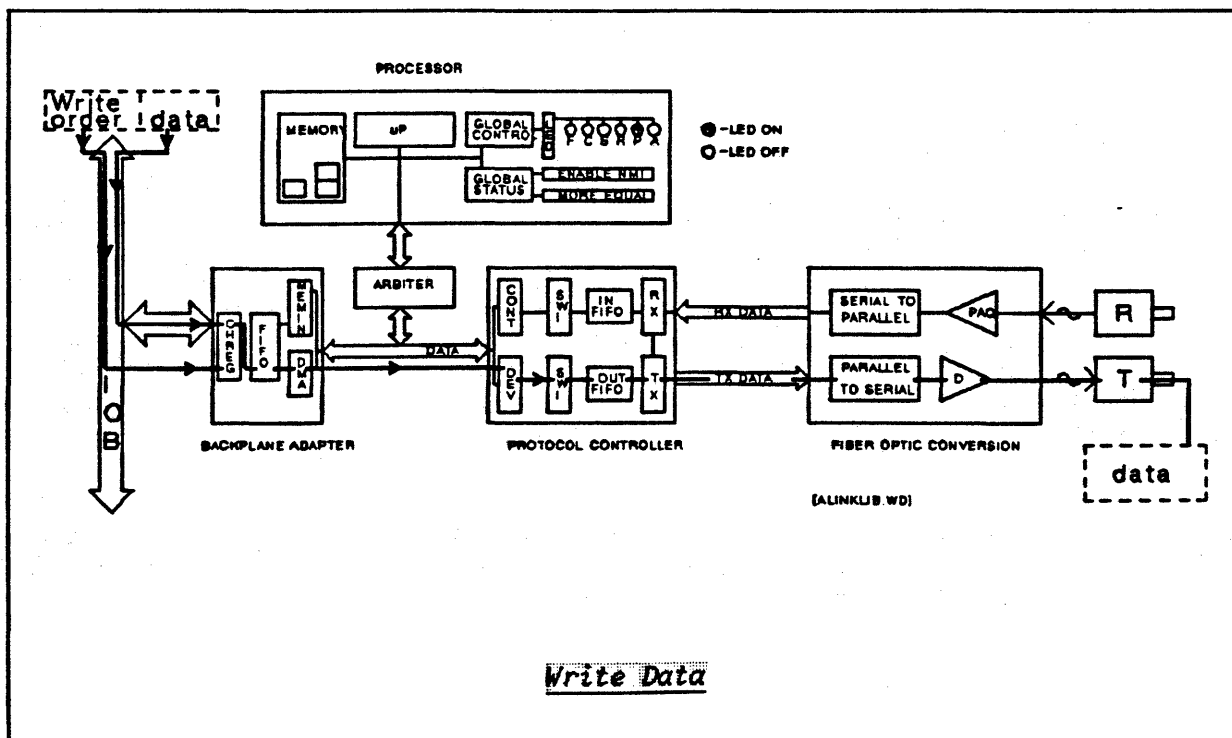
Functional Overview



During the Command Data Transfer, the channel sends the Write Data order, which the BA accepts, and then lets the incoming data (the CS-80 Command) pass through it and into the PC where it is readied for transfer onto the link.

When the BA encounters a word tagged with an end delimiter, it passes this final transfer to the PC and then waits for the next list of events to be supplied by the Processor.

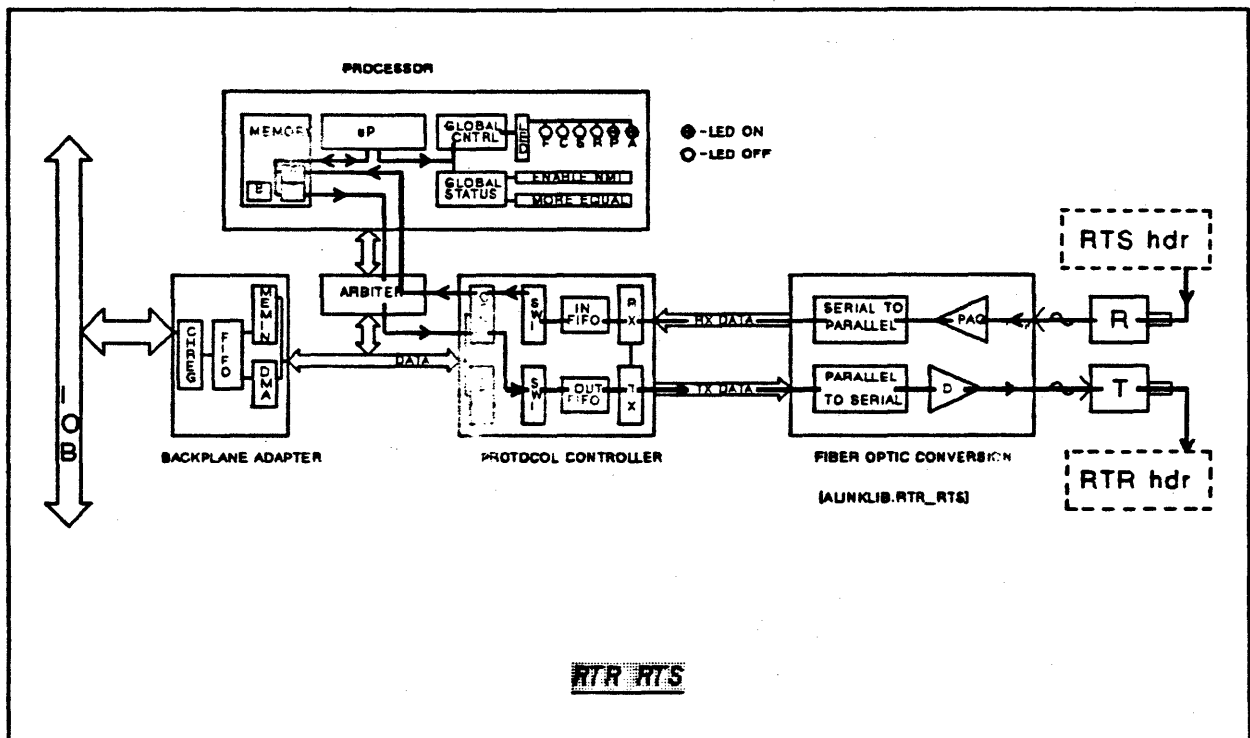
When the PC encounters the last word(byte), it flags this event by marking a bit in the level 2 header and points its Transmit Switch back to the Control Port.



Execution Phase.

Before the actual Execution Phase data is transferred, the adapter waits for the remote device to request use of the desired virtual circuit by an RTS Header.

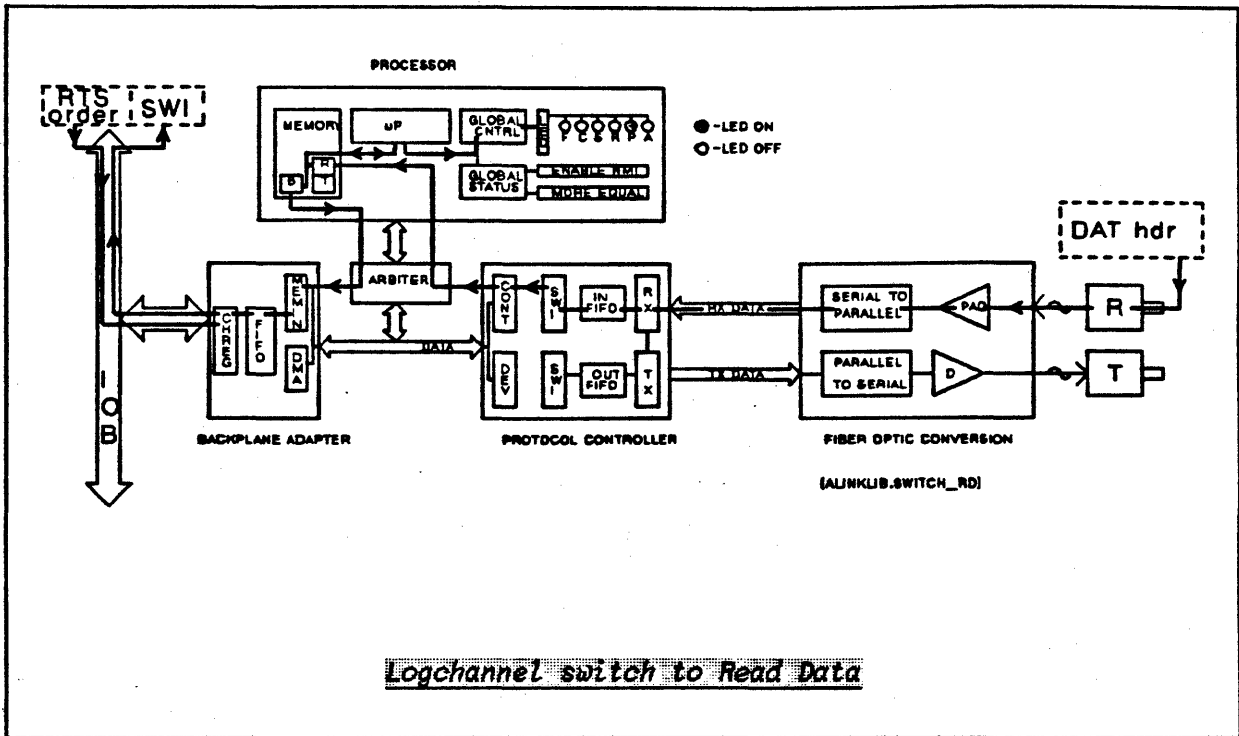
When the Processor sends out the acknowledging RTR Header, it proceeds through the Arbiter to the Protocol Controller, as before, and flashes the Activity light.



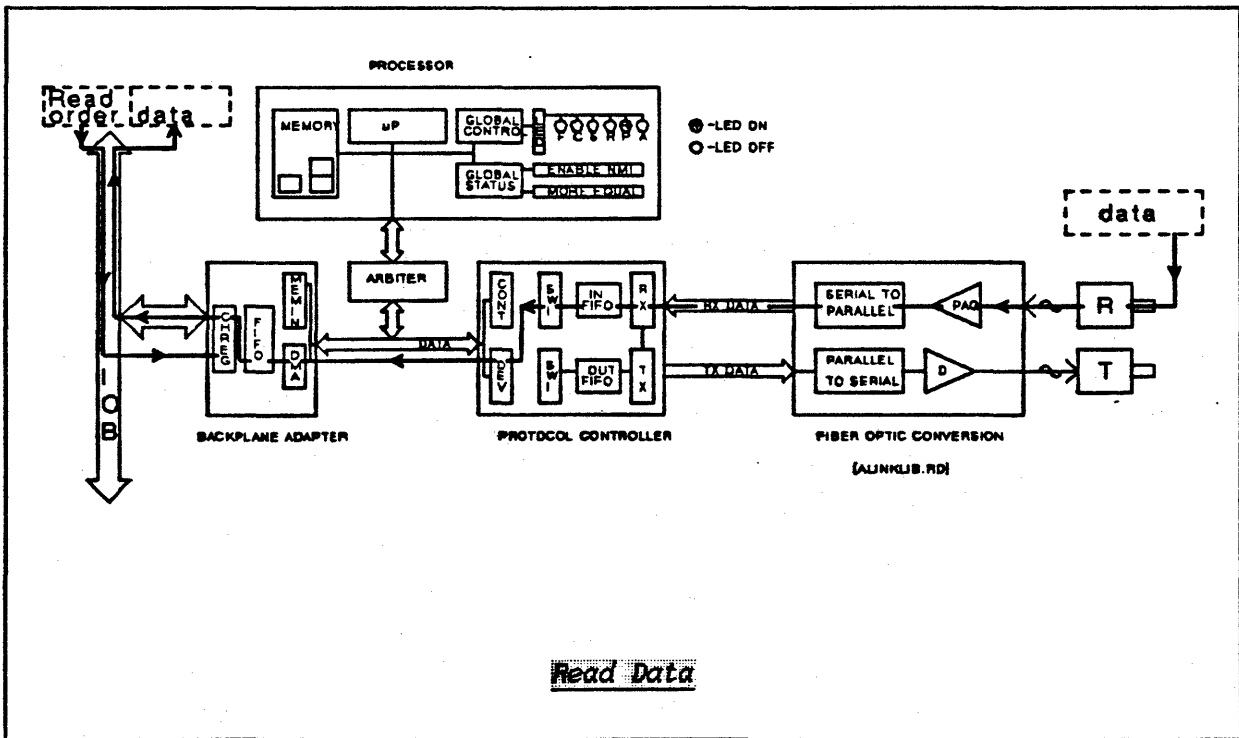
Once the RTR Header has been sent, the Processor provides a new list of events for the Backplane Adapter to follow. It programs the BA to wait for the Protocol Controller to indicate that it has data, respond with a SWI <n> to the next Read Transparent Status Order, and then wait for a Read Data order, at which time it will transfer data from the Protocol Controller to the Channel.

Meanwhile, the Processor places the Protocol Controller in *Receive Automatic Mode*. This is similar to the Transmit Auto mode, in that once the DAT header is received, the SWITCH points to the Device port, and the remaining incoming data will transfer through this port.

Functional Overview



Assuming that the adapter has received the DAT header, the PC will now indicate that it has data at the Device Port. The BA will proceed through the RTS Order and start passing the *Execution Phase Data* from the PC to the channel until it detects an end delimiter.



Report Phase.

The adapter interaction during **Report Phase** is almost exactly the same as during the **Execution Phase**.

The main difference, is that the Processor builds a slightly different list of events for the Backplane Adapter to follow. Instead of looking for a Read Data (RD) Order, the Backplane Adapter looks for a Read Status (RS) Order before allowing *Report Phase Data* to travel from the Protocol Controller to the channel.

Three types of specifications will be covered in this section: **Functional, Electrical, and Physical.**

Functional specifications are concerned with card performance and the basic end user interface.

Electrical specifications outline power and timing requirements of the card when installed in the host system and connected to a remote device.

Physical specifications describe the card in terms of size, connectivity and operating environment.

FUNCTIONAL

The A-Link CIO Adapter is a high speed full duplex serial interface capable of transmitting over distances up to 1 Kilometer to a remote device. To product has two basic interfaces: the **CIO Backplane** and the **A-Link Frontplane**. Given these two interfaces, the card is capable of operation in any of several Configurations, depending upon the requirements of the host channel and the remote device. The A-Link card also has several features that allow it to maintain a high level of data integrity, including local **Self-Test** and a set of **Status Indicators** that are visible on the card and available for backplane interrogation.

- A-link Protocol Frontplane
- Channel I/O Protocol Backplane
- up to 64 active Logchannels
- up to 64 active Virtual Circuits
- up to 5 Mbyte frontplane
- up to 5 Mbyte backplane (*10 Mbyte for < 128 byte bursts*)
- optional Channel NMI capability
- host to host, host to disc capabilities
- parity and CRC data protection
- future support for secondary power
- full duplex optical interface
- LED status indicators
- programmable loopback
- on card self test
- link performance and status monitoring capabilities

SPECIFICATIONS

Communication Path Configurations

Currently, the A-link CIO Adapter supports only the point to point configuration with an A-MUX Disc Cluster.

Within this configuration, two basic communication paths exist: **Host to Disc**, and (*at remote device's option*) **Host to Host**.

Special diagnostic communication paths, such as *Loopback*, will be described later.

Host to Disc.

The **Host to Disc** configuration connects the A-link CIO Adapter to any of the discs in an A-MUX Disc Cluster (or any disc controller with an A-link interface and CS-80 capability). The host SPU can access any of the discs in the cluster according to the virtual circuit mapping as specified by the disc cluster.

Host to Host.

The **Host to Host** configuration connects the A-link CIO Adapter to another SPU A-link adapter through the A-MUX Cluster, assuming that more than one SPU has an A-link connection to the remote device.

Frontplane (A-Link)

The A-Link CIO Adapter Frontplane consists of a full duplex fiber optic connection used to transmit and receive data. This provides a high speed interface which is controlled by A-Link Protocol.

A-Link Protocol.

The A-link Protocol defines an I/O communications link between host System processing unit(SPU) and a remote device, either another SPU or an I/O device controller, such as a disc controller.

The A-link CIO Adapter supports and implements all four layers of the A-link Protocol.

For a detailed description of these layers, refer to the A-Link Protocol document.

For a description of the layer 3 format see the section on Firmware description.

Virtual Circuit Capabilities.

The A-link CIO Adapter is capable of communicating over any of 65536 virtual circuits. One of these circuits, 0FFFFH is reserved for link control independent of the remaining 65535 circuits.

The number of active virtual circuits is limited to 64. This number represents the combination of virtual circuit requests from both the local host SPU and any requests for access to a virtual circuit from the remote device.

Transfer Rates.

Both the A-link card's receiver and transmitter are capable of operating at 80 Mbaud, which translates to approximately 5 Megabytes. This 5 Megabyte figure accounts for the control and information mix present on the link.

Actual 5 Megabyte performance will be limited due to the half duplex nature of the A-link adapter's backplane, loading of the channel, and the ability of the remote device to sink and source data at the described rate.

Link ID.

The Link ID of the A-link CIO Adapter is type 0. It may communicate with any remote device that supports type 0 access.

Backplane

The A-link card uses Channel I/O (CIO) protocol to communicate with the host SPU across its backplane. The card supports logchannel multiplexing. Logchannel accesses to the link should be setup in a CS-80 structure.

The A-Link CIO Adapter supports the CIO Standard Document layers 0 through 3. It is considered to be a *NON Level 1 Device Adapter*. Though the card agrees with the CIO Standard not all Level 2 and Level 3 operations are supported.

Channel I/O compatibility.

The A-Link CIO Adapter is compatible with channels built to either CIO Standard Version 1.5 or 2.0(draft).

Performance differences between these two standards should be solely due to the host SPU's channel implementation.

Feature differences are mainly the lack of parity protection on data and control on the 1.5 backplane. The adapter cannot determine backplane bus faults in this case.

SPECIFICATIONS

Supported Orders and Commands.

The A-link CIO Adapter supports the following commands and orders:

Commands [CSC, DSC, RSC, ETA]

Orders [CLC, WC, RD, WD, RS{,d}, IDY, RTS, WTC, DIS]

Details on support are found in the section on Firmware Description.

Transfer Rates.

For a typical block of data (=2048 bytes), the adapter is capable of an average transfer rate of 5 Megabytes.

The A-link adapter has an internal FIFO that may be active in either the input or output direction. The FIFO length is 128 bytes. Burst mode operation ($n \leq 128$ bytes) will equal 2 times the IOSB rate for the particular channel. Thus, for a 200 nanosecond IOSB a 10 Megabyte burst rate over 128 bytes is achievable.

Logchannel Capabilities.

The adapter supports up to 64 connected logchannels. Each of these logchannels may optionally have a virtual circuit associated with it.

For a disc interface, this implies that a combination of 64 outstanding Read and Write requests are possible.

Configurable Options

The A-Link CIO Adapter has three basic configurable options: Equal mode, Channel NMI, and Power Fail Support.

Equal mode.

Equal mode is a jumperable option that places the A-link card in either the *LESS Equal* or *MORE Equal* state.

This option is used to maintain deadlock avoidance with the remote device.

More Equal mode allows the card to proceed with a write operation to the link when all of its resources are ready (i.e., *data is available*).

Less Equal mode prevents the card from proceeding with sending data onto the link until the remote device indicates that it is capable of proceeding.

Channel NMI Enable.

Channel NMI Enable is a jumperable option that allows the card to drive the NMI line on the CIO backplane.

When Channel NMI Enable- is *TRUE* the card is capable of driving the NMI line.

When Channel NMI Enable- is *FALSE* the card cannot drive the NMI line.

The card can detect the state of the option and report that status to the host SPU through the channel.

Power Fail Support.

The A-link card provides optional power fail support by the secondary power available jumper. If a backplane supports secondary power and the associated channel signals, the card will enter a power down mode during a *Power Fail Warning* and will maintain its on card memory should Primary Power fail and Secondary Power remain.

Note that the state of the link will be lost during a Primary Power fail, and that pending transactions may need to be aborted.

Note that this circuitry will probably not be tested extensively since none of the current channels implement secondary power or power fail.

Status Indicators

The A-Link card has two basic report mechanisms of card operational status: LED indicators, and Backplane Status Requests.

LED indicators.

The A-Link CIO Adapter has six LED indicators that are used to communicate two types of status: Operational Status during normal operations or a Self Test Failure Code (STFC)

The significance of the LED indicators is determined by the state of the Failed Self Test indicator as shown in the following tables.

SPECIFICATIONS

alinksti

| ID | COLOR | DEFINITION | STATE | DESCRIPTION |
|----|-------|--|-------|--|
| F | red | Failed selftest | on | Card failed selftest. The C,S, and R LED's indicate the type of self test failure |
| | | | off | card passed self test |
| C | red | Configuration error | on | More equal jumper is in wrong position (F must be OFF) |
| | | | off | Card is correctly configured (F must be OFF) |
| S | red | Signal not present -OR- unacceptable quality | on | one or more of the following are true: <ul style="list-style-type: none"> ■ No transitions detected on link ■ Phase lock receiver could not acquire signal ■ Link error rate exceeds maximum threshold (F must be OFF) |
| | | | off | signal is of acceptable quality (F must be OFF) |
| R | red | Remote not responding | on | Remote fails to respond to a periodic request for identification (F must be OFF) |
| | | | off | Remote has identified itself (F must be OFF) |

alinksti

| ID | COLOR | DEFINITION | STATE | DESCRIPTION |
|----|-------|------------------------------------|-------|---|
| P | green | Passed self test, card oPerational | on | card has passed self test and is operational <i>(F must be OFF)</i> |
| | | | blink | Firmware awaiting for backplane to issue an initial connect subchannel command <i>(F must be OFF)</i> |
| | | | off | Firmware detected a link Protocol Error or determined that the pca was no longer functional <i>(F must be OFF)</i> |
| A | green | Activity | on | Indicates that a header has just been transmitted <i>(F must be OFF)</i> |
| | | | off | no headers are currently begin transmitted <i>(F must be OFF)</i> |

SPECIFICATIONS

alinkstc

| CSR leds | AREAS COVERED | FAILURE |
|----------|--|------------------------|
| 111 * | Processor | ROM Test |
| 110 | Processor | 186 Test |
| 101 | Processor | RAM test |
| 100 | Processor Arbiter Protocol controller | PRONTO test |
| 011 | Processor Arbiter Protocol Controller Fiber Optic Conversion | JUPITER Test |
| 010 | Processor Arbiter Backplane Adapter | PASSPORT Test |
| 001 | Processor Arbiter Protocol Controller Fiber Optic Conversion Backplane Adapter | DMA Transmit Path Test |
| 000 | Processor Arbiter Protocol Controller Fiber Optic Conversion Backplane Adapter | DMA Receive Path Test |

*NOTE - 1- ON, 0- OFF

Backplane Status Report.

The states of all of the above status indicators are available to the backplane. This information is obtained through the logchannel request to **Read Card Information** with the subfunction **Global Status**.

This logchannel request will yield the following information -

- state of **More Equal** jumper
- state of **NMI Enable** jumper
- current state of the link (*up,sick, or down*)
- current mode of operation

* Preliminary Version *

- fiber optic receive circuitry currently Out of Lock
- fiber optic receiver inactive
- copies of the current Activity, Signal, Remote, and Configuration LED states
- cumulative link errors since last adapter reset
- elapsed time since last adapter reset
- link error rate for the previous hour

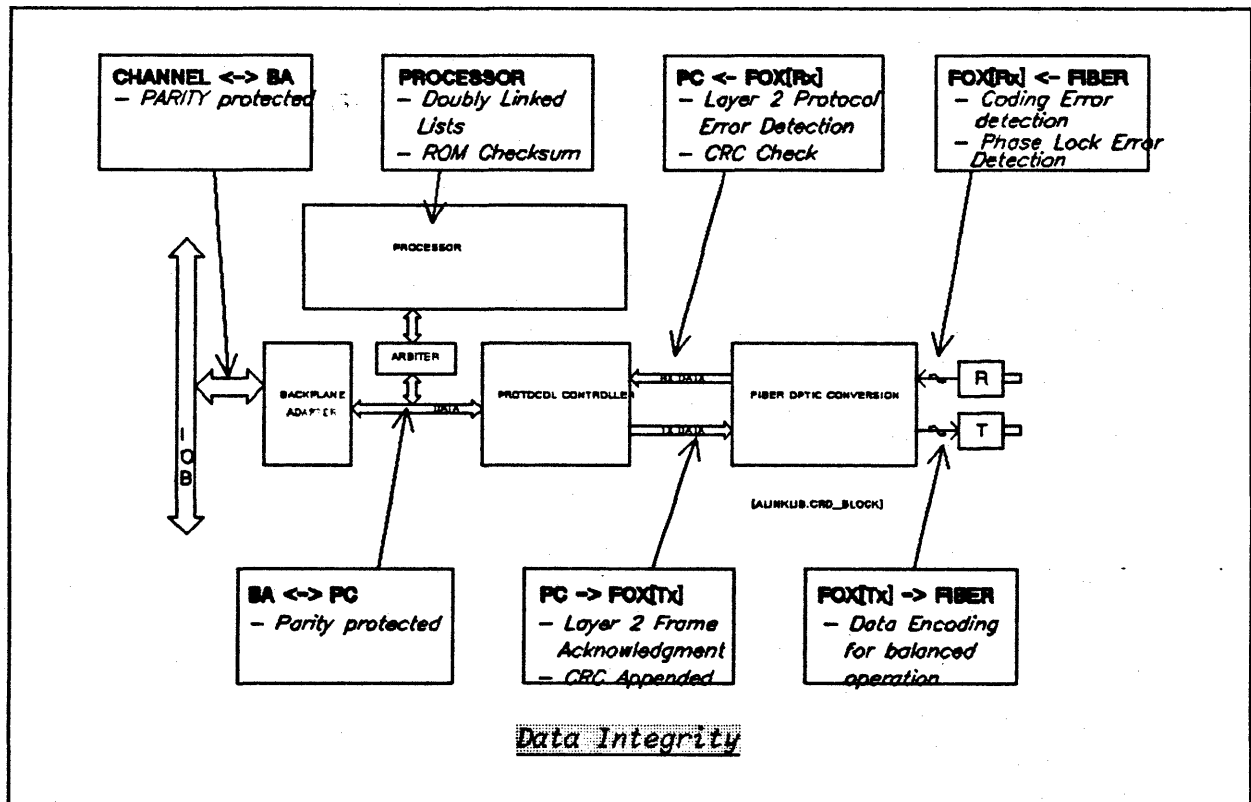
For more details on these status fields refer to the *Firmware Description* Section.

Data Integrity

The A-Link adapter employs a number of mechanisms to insure the data integrity during operation.

Data is protected over the following paths channel to Backplane Adapter, Backplane Adapter to Protocol Controller, Protocol Controller to Fiber Optic Conversion, and Fiber Optic Conversion to Fiber.

The Processor Block also implements data protection schemes.



SPECIFICATIONS

Channel to Backplane Adapter.

Data from Channel to Backplane Adapter is **PARITY** protected. One parity bit is assigned to each byte of the word wide channel interface.

Parity provides 100% Single bit error detection.

Backplane Adapter to Protocol Controller.

Data from Backplane Adapter to Protocol Controller is also **PARITY** protected.

Parity provides 100% Single bit error detection.

Protocol Controller to Fiber Optic Conversion.

Data from Protocol Controller to Fiber Optic Conversion has a number of safeguards. In the *Transmit Direction*, a CRC is appended to each frame that is transferred onto the link.

In the *Receive Direction*, validity of the CRC is checked and a special sequence of control codes must be detected. The Protocol Controller also monitors status lines provided by the Fiber Optic Conversion block and rejects any frames where error status is found.

For details on the polynomial used to generate the CRC, refer to the PRONTO ERS or the A-link Standard.

Fiber Optic Conversion to Fiber.

The Fiber Optic Conversion block uses a special encoding scheme to maintain balanced levels on the optical link.

The Optical Receiver section can detect illegal codes found in the incoming bit stream and also reports on its ability to lock on to the stream.

For more details on the encoding and decoding scheme consult the A-link Standard document.

Processor.

The Processor block uses a CRC method to verify **ROM** checksums stored in the actual ROM. While the RAM area does not have any parity protection, some measure of RAM validity is obtained through the use of **Doubly Linked Lists** when accessing data structures.

Self Test

The A-Link CIO Adapter Self Test verifies operational capabilities of ~90% of the card resident circuitry.

Successful completion of self test is indicated by both the LED status indicators and the assertion of the PST bit in the Read Sense register from the CIO backplane.

The card has four basic varieties of self-test: **Power On**, **Reset**, **Addressed**, and **External Fiber**.

Power On.

The **Power On** test performs a 1 second test on the adapter and establishes the minimum capability of the adapter to communicate to both front plane and back plane.

Reset.

The **Reset** test performs a 3 second test on the adapter. A more comprehensive card RAM and data path test are performed at this level.

This test is invoked by asserting the RST- signal to the card for a minimum of 1 microsecond.

Addressed.

The **Addressed** test performs a 15 second test on the adapter. It is invoked through the **Addressed Device Clear** capability of the CIO backplane.

This test exercises the local data paths in an internal loop between the various functional blocks.

External Fiber.

The **External Fiber** test is a special diagnostic mode used only for field verification and manufacturing. It provides a means of verifying the operation of the analog fiber optic conversion circuitry that is inaccessible in the normal self test modes.

It requires the installation of an **Optical Loopback Fiber** (*8120-xxxx*) or its equivalent) or an **Optical Coupling Device** (*xxxx-xxxx*) or its equivalent).

ELECTRICAL

The A-Link CIO Adapter is designed for use in systems that implement a Channel I/O Backplane and an A-link optical Frontplane.

SPECIFICATIONS

Backplane

Power.

Estimated Maximum Power Consumption: 14.6 Watts
16.2 Watts (2 sigma)

Figures and absolute specifications will be included in this document when parametric information from the NMOS-III designs which are used to implement the card become available.

Loading.

The A-link CIO Adapter meets all Loading specification as detailed in the CIO Standard.

Figures and absolute specifications will be included in this document when parametric information from the PASSPORT NMOS-III design is available.

Timing.

The A-link CIO Adapter meets all timing specification as detailed in the CIO Standard.

Figures and absolute specifications will be included in this document when parametric information from the PASSPORT NMOS-III design is available.

Frontplane

The A-Link CIO Adapter Frontplane consists of an Optical Transmitter and Optical Receiver pair.

Optical Receiver.

The Optical Receiver specs will go here.

Preliminary specifications:

| | |
|----------------|-------------|
| Wavelength: | 820 nm |
| Baudrate: | 80 Megabaud |
| Minimum Power: | -27 dbM |
| Maximum Power: | -9 dbM |

Optical Transmitter.

The Optical Transmitter specs will go here.

Power figures here are for the output power *coupled* into a 100 μm fiber cable.

Preliminary specifications:

| | |
|---------------------|-------------|
| Wavelength: | 820 nm |
| Baudrate: | 80 Megabaud |
| Output Power (min): | -21 dbM |
| Output Power (max): | -9 dbM |

Fiber.

Recommended Fiber characteristics will go here.

Recommended Fiber for use is HFBR-BY0030 30m Fiber optic cable

Preliminary specifications:

| | |
|---------------------|---------------------|
| Cable Attenuation: | 8 dB/km |
| Numerical Aperture: | 0.3 @ length > 300m |
| Maximum Length: | 500 m |
| Core Diameter : | 100 μm |
| Cladding Diamter: | 140 μm |
| Bandwidth (min): | 40 Mhz |

RECEIVED SIGNAL PROBE TAP.

The Received Signal Probe Tap is located to the left of the Front Panel LEDs on the HP27111A. It is used as a test monitoring point mainly for manufacturing purposes. It is intended for 50 ohm cables equipped with subminiature probes. The monitoring device should be isolated from the cable by a blocking capacitor.

The signal presented represents the serial bit stream received from the front panel optical receiver connection.

| | |
|---------------------------|---|
| Signal Output: | 70 mV |
| Frequency: | 40 Mhz |
| Recommended Probe: | HP 10027A 1:1 Dual Purpose 50-ohm Miniature Probe |
| Recommended Blocking Cap: | HP 10240B Blocking CAP |

SPECIFICATIONS

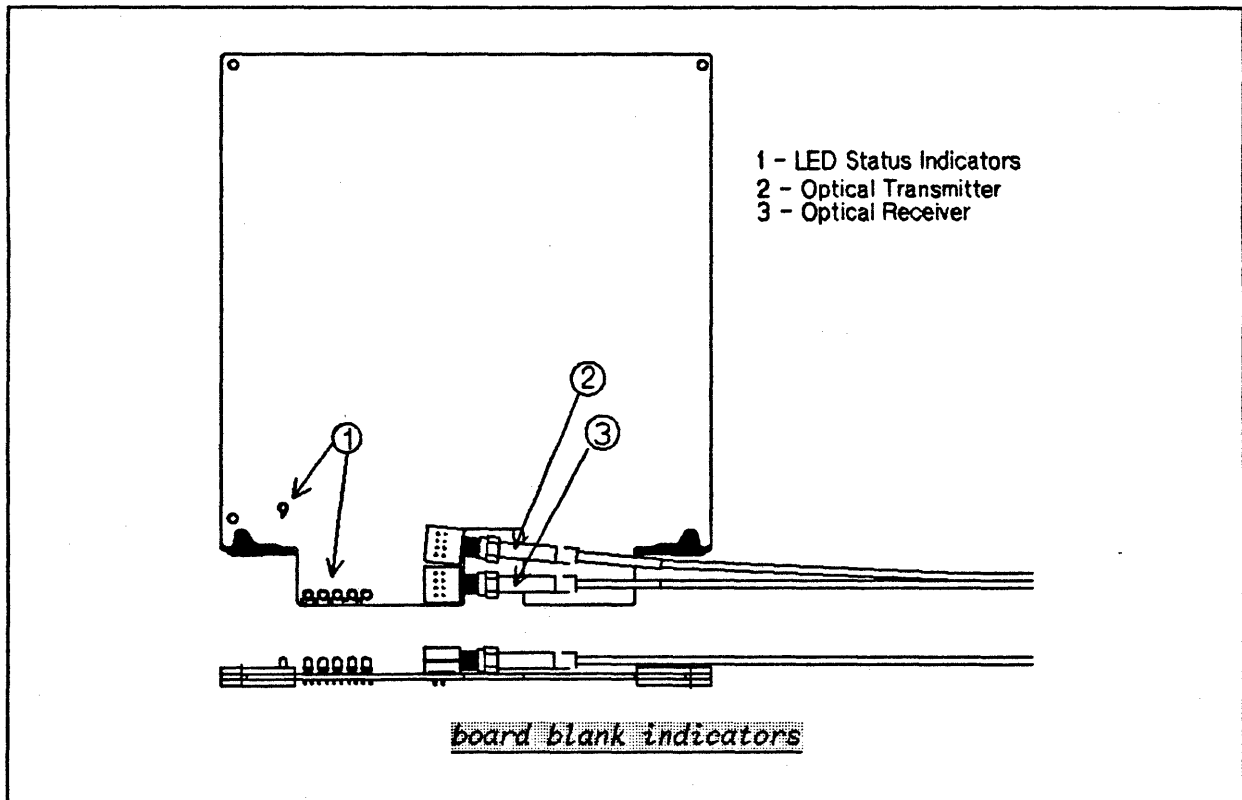
PHYSICAL

The A-Link CIO Adapter is a 6 layer Printed Circuit Assembly (PCA) which is compatible with CIO card cages and connectors and provides two mechanical connections for optical input and output.

Dimensions

| | |
|------------------|--------------------------------|
| Length: | 193.04mm (7.600in) |
| Width: | 171.45mm (6.750in) |
| Thickness: | 16.38mm (0.645in) |
| Weight: | 0.26kg (0.564lb) [preliminary] |
| Shipping Weight: | 1.36kg (3.000lb) [preliminary] |

Unlike most CIO PCAs, the A-Link PCA makes use of the traditional external connector area for both its optical connectors and for optical circuitry.



Connections

The A-link CIO Adapter PCA has connections to both **Frontplane** and **Backplane** as well as a set of **Jumperable Options** and **Status Indicators**.

Frontplane.

Connection to the front plane is made through two SMA style connectors *part-number 00000-00000*.

The location of the receiver and transmitter are shown in the board blank illustration.

Backplane.

The backplane connector is an AMP 102584-2 female 80 pin model which is compatible with the AMP 1025867-8 male connector which is located on the male backplane motherboard.

The pinout of the connector is as follows:

SPECIFICATIONS

alciopin

| PIN | Mnemonic | PIN | Mnemonic |
|-----|----------|-----|----------|
| A1 | FGND | B1 | FGND |
| A2 | DB14- * | B2 | DB15- |
| A3 | DB12- | B3 | DB13- |
| A4 | GND | B4 | GND |
| A5 | DB10- | B5 | DB11- |
| A6 | DB8- | B6 | DB9- |
| A7 | GND | B7 | GND |
| A8 | DB6- | B8 | DB7- |
| A9 | DB4- | B9 | DB5- |
| A10 | GND | B10 | GND |
| A11 | DB2- | B11 | DB3- |
| A12 | DB0- | B12 | DB1- |
| A13 | GND | B13 | GND |
| A14 | AD2- | B14 | AD3- |
| A15 | AD0- | B15 | AD1- |
| A16 | GND | B16 | GND |
| A17 | DOUT- | B17 | UAD- |
| A18 | BPO- | B18 | BP1- |
| A19 | CEND- | B19 | CBYT- |
| A20 | SYNC- | B20 | POLL- |
| A21 | GND | B21 | GND |
| A22 | CCLK+ | B22 | IOSB- |
| A23 | GND | B23 | GND |
| A24 | BR- | B24 | ARQ- |
| A25 | DBYT- | B25 | DEND- |
| A26 | MYAD- | B26 | RST- |
| A27 | GND | B27 | GDN |
| A28 | ... | B28 | SYS DEP |
| A29 | ... | B29 | ... |
| A30 | ... | B30 | AP- |
| A31 | CDP[0]- | B31 | CDP[1]- |
| A32 | PFW- | B32 | NMI- |
| A33 | PPON+ | B33 | SPON+ |
| A34 | GND | B34 | GND |
| A35 | AC- | B35 | AC- |
| A36 | AC+ | B36 | AC+ |
| A37 | -12 | B37 | -12 |
| A38 | +12 | B38 | +12 |
| A39 | +5S | B39 | +5S |
| A40 | +5P | B40 | +5P |

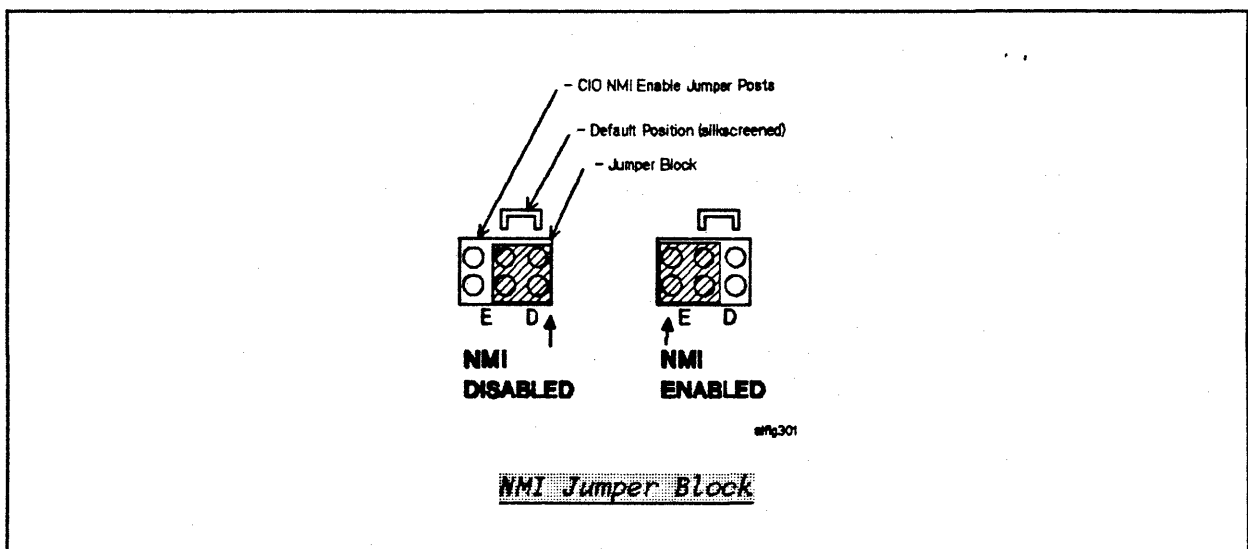
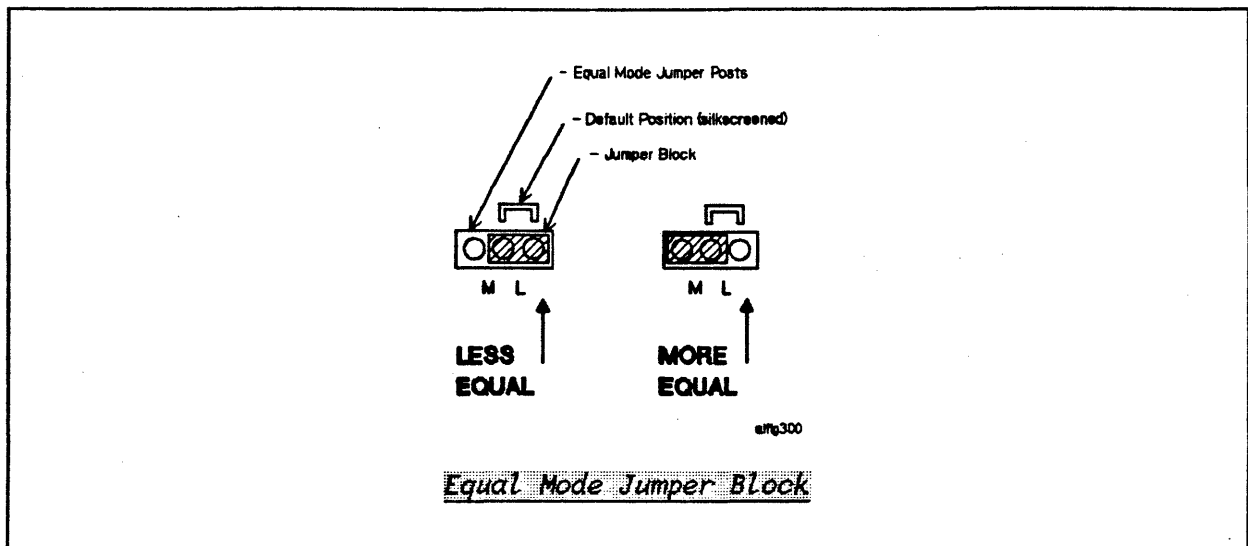
* NOTE - Bit numbering on the CIO connector treats bit 0 as the LSB!

Jumperable Options.

There are two Jumperable options on the A-link CIO Adapter: Equal Mode *and* Enable NMI.

* Preliminary Version *

The orientation of the jumpers for the modes of operation are shown in the following illustrations:



Note, that each jumper post area has silkscreened *one-character legends* to describe the jumper setting:

- M- More equal
- L- Less equal
- D- NMI Disabled
- E- NMI Enabled

A *default position* indicator is also provided. As shown above the default positions of the jumpers are:

Less Equal
NMI Disabled

A missing jumper will result in the default setting.

SPECIFICATIONS

Status Indicators.

The six status indicators are located as shown on the A-Link CIO Adapter. Each LED has a one character mnemonic to aid in identification.

- | | | |
|-----|-----------------------------------|---------|
| ■ F | - Failed Selftest | (RED) |
| ■ C | - Configuration error | (RED) |
| ■ S | - Signal quality marginal | (RED) |
| ■ R | - Remote not responding | (RED) |
| ■ P | - Passed selftest and oPerational | (GREEN) |
| ■ A | - Activity on the link | (GREEN) |

A lit lamp indicates that the adapter detects the associated state or event.

The lamps have a minimum luminous rating of 1.5 millicandles, and have an off-axis viewing angle of +/- 30°.

Operating Environment

The A-link CIO Adapter meets class B environmental specs.

Operational temperature is from 0° to 70°C.

Air flow rate and cooling requirements will be included here.

This section describes the control firmware resident on the HP-CIO adapter as seen by both the host and by the device at the other end of the link. The characteristics of the device at the end of the link are specifically not included in this section; commands *to* the card are described while commands *through* the card are not.

Also described are the HP-CIO channel program requirements of the adapter, and information indicating the performance to be expected in the execution of each step in a channel program.

It is assumed the reader is familiar with the A-link Level 3 protocol. If not, see "A-link Protocol Standard".

NOTE

This document includes support for both the "Normal" and "Negotiated" mode of A-link. It is expected that only the Normal mode will be used, and so the Negotiated mode will be left for implementation as time permits. If anyone has a need for Negotiated mode at first release; *please* inform the author.

Changes from last revision

Event Polling removed after no responses in favor of retaining it were received.

The *C* bit of request block was removed. The *d* bit of the RS order is sufficient to determine if chaining is to be used.

ERT

Logical channel error trap (ERT) and the associated protocol references have been removed. Instead of error trapping, the adapter will source or sink 1 byte of data for each data order until the RS order is encountered.

Logical channel end-of-data (EOD) will be reported if either the data is terminated early by the device or some error-trap event occurs during the request.

Firmware Description

Loopback

Request 6 subfunctions 0,1, and 2 have been replaced with request 1 subfunction 253. Requests 4 and 5, subfunction 246, have been replaced by request 4 subfunction 253 (which has the same channel format as request 1 subfunction 253). Request 6 subfunction 2 now performs a link resync and also insures that the optical components are enabled (ie, can be used as 'Leave Jupiter Loopback state'). A new request 6 subfunction 253 performs what used to be called 'internal loopback' and has the same functionality as the new requests which 'enter Jupiter loopback', 'read device loopback', and 'resynchronize link' executed in sequence.

Subfunction 254 and 255

Subfunctions 254 and 255 have been reversed for all requests. Now, subfunction 255 applies to raw link access requests.

Status bytes

The status byte values have been changed. One byte has been added to transaction status (a byte count). New status's indicating invalid frame type and invalid data tag recieved, have been added.

There is now the possibility of zero length status from a CS/80 style transaction due to implementation of a new status protocol. The zero length status message has the same meaning as a CS/80 QSTAT of zero.

CHANNEL PROGRAMMING

The resident firmware uses the Logchannel Protocol in its transactions over the backplane. Some of this protocol is implemented within the backplane interfacing circuitry on the card, specifically the *Passport* chip, and any limitations imposed by *Passport* apply to the adapter as a whole. The rest of the Logchannel protocol is either implemented by the card resident processor, or not implemented (i.e. not supported) at all.

Supported Commands and Orders

The following HP-CIO commands and orders are supported by the HP-CIO device adapter:

Commands:

CSC
DSC
RSC

Orders:

CLC
 WC
 RD
 WD
 RS[,d]
 IDY
 RTS
 WTC
 DIS

| |
|-------------|
| NOTE |
|-------------|

The following descriptions of orders and their effects are for information only. The adapter's unit of interaction with the host is a transaction, which is roughly equivalent to a request to the adapter's system driver. The adapter requires that all transactions follow certain sequences of orders, which are described later. Enhancements for other sequencing, i.e. different types of driver requests, should be made with the author.

CSC: Connect__subchannel Command

The CSC command is used at power-up to connect a subchannel to the adapter. This command must also be used after any other action which disconnects the subchannel, i.e. the DIS order, the DSC command, and the adapter resets RST and DCL (addressed device clear).

DSC: Destroy__subchannel Command

The DSC command tells the adapter to quit whatever it is doing on the HP-CIO bus and deallocate its subchannel. The adapter responds to DSC with SCD (subchannel destroyed) when this is done. If the DSC causes the adapter to abort a transfer from or to the link, the adapter will bit bucket the transfer in progress and will resynchronize the link in some cases (the cases may not be determinable by the host and are therefore not detailed here). All effected link transactions will advance to their status phase and will report *host aborted transaction indirectly (DSC or VC reset)*.

In addition to abortion of link data transfer, the host (HP-CIO) data transfer will be terminated via the HP-CIO DEND signal on the next data byte handshake (possibly sourcing a 'garbage' data byte) before SCD acknowledgement is given for the DSC.

RSC: Resume__subchannel Command

The RSC command is used to cause the adapter to request an order in the (previously connected) subchannel. This is used to nullify a previous Pause Order.

Firmware Description

NOTE

RSC is handled quickly by the Passport chip if it occurs when the subchannel is paused, however, if the RSC is redundant, (ie, occurs when the subchannel is unpaused) then the device adapter's microprocessor will service an interrupt and invoke significantly more overhead in handling RSC.

CLC: Connect_logical_channel Order

The CLC order is used to initiate a transaction on the adapter, i.e. each transaction begins with a CLC order and its associated data block.

The format of the data block, called *initiation data* by the HP-CIO Standard, and the *request block* by this document, is as follows:

| | | | | |
|------------------------|---|---|---|---------|
| 7 | 6 | 5 | 4 | 3 - 0 |
| Logical channel (high) | | | | |
| Number (LCN) (low) | | | | |
| 0 | B | 0 | 0 | Request |
| Request Subfunction | | | | |
| Virtual Circuit (high) | | | | |
| Number (low) | | | | |
| Length of (highest) | | | | |
| "Execution" | | | | |
| Message | | | | |
| in bytes (low) | | | | |
| Data Tag for some SFNs | | | | |

reqblk

The various fields in the Request Block are defined as follows:

Logical Channel Number is the 16 bit number assigned by the HP-CIO channel to this transaction. This number is used by the channel and adapter to identify this transaction in all future communications regarding this transaction and therefore must be unique for each pending (ie, concurrent) transaction. The only exception to the requirement of uniqueness is for the *Reset Virtual Circuit* request.

B bit, if set, specifies that RD and WD orders used in this transaction are to be used in HP-CIO *Blocked* mode, with a blocking factor of 8192 bytes. All parts of a request which contains B=1 are performed in the "Normal" mode.

If the B bit is clear, RD and WD orders are not blocked across the backplane, and are transferred across A-link in Negotiated mode. This sends the entire data part from the link to the host in ONE data block. The other parts of the transaction (RS orders), if any, are still sent in Normal mode.

Since not all devices support both modes, it is up to the programmer to set this bit correctly.

Request (Rq) code tells the card what type of transaction is being initiated. Valid choices are:

- 1 = Read data from Device
- 2 = Write data to Device
- 4 = Read status information from card
- 5 = Write configuration data to card
- 6 = Control card

Subfunction field further describes the request to be performed. The valid parameters are defined with the detailed description of each request.

Virtual Circuit Number is the A-link level 3 virtual circuit number. The adapter makes the association between the A-link VC number and the HP-CIO LCN whenever necessary. With the exception of a request to reset a virtual circuit, virtual circuits numbers must be unique for each pending (ie, concurrent) transaction that uses a virtual circuit.

Length of Execution Data is a 32 bit integer giving the length of the main data portion of the request. The length is given unsigned, and in bytes. Transfers making use of this field are so indicated in the detailed request description.

Data Tag is used with some requests to specify which type of data is to be transferred.

WC: Write__control Order

The WC order is used to extend a channel program to include two or more adapter requests. The data block of the WC is identical to the Request block from the CLC order *from the Request field on down*. It does *not* include the Logchannel Number. The WC order is used after any RS order which does not have the "d" bit set.

Any sequence of data transfers can be accomplished by chaining together a sufficient quantity of more atomic requests (e.g. Request 1 & 2, subfunction 0). For requests which are already implemented in firmware, however, it is always faster to use the canned request, than to build it from its component pieces.

RD: Read__data Order

The RD order is used to read the data previously defined by the Request field in the CLC data block from the card to the host. This may be data from the device off the link, or from status memory on the card itself.

Firmware Description

The RD order is used either in normal or blocked mode as defined by the B bit in the request field. Blocking, if used, is done with the number of bytes as specified in the Data blocking field in the IDY block. Blocking may only be used for data which originates on the link.

If blocking is used, the HP-CIO block mode bits S1S2 must be set to 1,0 such that the channel sequence for blocking is of the following form:

<< Logchannel Break >>

RD or WD < link data >

where the channel sequence is repeated until the channel count has expired or the card indicates end of data. *Logchannel Break* concludes with the channel requesting transparent status via the RTS order and the card responding with a SWITCH to the logical channel associated with the transaction in progress (end of data is also reported via the data returned to the channel after an RTS).

WD: Write_data Order

The WD order is used to write the data previously defined by the Request field in the CLC data block to the card from the host. This is data which will be sent to the device over the link, or to the card as configuration information.

The B bit in the request field of the CLC data block is used to specify data blocking for "Execution" messages. It is used in roughly the same manner as for the RD order, that is, blocking may only be used on data to be sent to the link. "Control" messages are never blocked.

If blocking is used, the HP-CIO block mode bits S1S2 must be set to 1,0.

RS: Read_status Order

The RS order is used to read a block of status data from the card or the link. The format of the block is dependent on which type of request was specified in the request field of the CLC block, and whether or not an error occurred during the request. However the first two bytes of block are the same for all conditions.

The "d" bit in the RS order is used to indicate the end of the channel program (i.e. the end of the overall request). If set the Logical Channel Number will be disconnected at the conclusion of the order. If the bit is not set, a WC order is assumed to follow and the same logical channel will be used for the request indicated in the *logical channel-less* Request block.

Transaction Status. The RS order receives a status message from the adapter regarding how the transaction went. This status message is as follows (one of the following for any given request):

Transaction Status=0

| | |
|---|---|
| C | 0 |
| count | |
| [count] bytes from device, where 0 <= count <= 32 | |

Transaction Status=
1,3 thru 7,9,12 or 13

| | |
|-----------|--------------------|
| C | 1,3 thru 7,9,12,13 |
| count = 0 | |

Transaction Status=10 or 11

| | |
|------------------|----------|
| C | 10 or 11 |
| count = 7 | |
| frame type | |
| frame attributes | |
| MSbyte of parm. | |
| 2nd byte parm. | |
| 3rd byte parm. | |
| LSbyte of parm | |
| frame tag | |

Transaction Status=2

| | |
|------------|---|
| C | 2 |
| count = 4 | |
| IMS byte 0 | |
| IMS byte 1 | |
| IMS byte 2 | |
| IMS byte 3 | |

Transaction Status=8

| | |
|-------------------|---|
| C | 8 |
| count = 4 | |
| MSbyte of residue | |
| 2nd byte residue | |
| 3rd byte residue | |
| LSbyte of residue | |

txstat

C = Adapter/Device configuration is incorrect

Transaction status:

- 0 = no error
- 1 = reserved (formerly link down before request started)
- 2 = device signaled error (IMS)
- 3 = illegal adapter request code
- 4 = illegal adapter request subfunction
- 5 = incompatible link handshake (RTS vs RQS)
- 6 = request aborted by link resynchronization
- 7 = request aborted by Virtual Circuit reset
- 8 = residue count: looped back less then requested length of data
- 9 = requested while in incompatible state
- 10 = message sequencing error in received link header
- 11 = bad tag received in link header
- 12 = host aborted transaction indirectly (DSC or VC reset)
- 13 = short request block length: missing parameters

The adapter automatically requests the device's Identity when the link comes up to determine what deadlock avoidance algorithm is being used. This is compared to the adapter's configuration. If there is an incompatibility, the "C" bit will be set, and remain set until either the adapter or device's configuration

Firmware Description

is changed (normally a manual operation). *The adapter should not be used to transfer link information to or from the host if this bit is set. Link Deadlock could occur.*

Transaction status 0 indicates that no errors in the transfer of data were detected by the adapter. Further status information may be available from the device; this is device dependent. For Request 1, subfunction 1 and Request 2, subfunctions 1 and 2, the Transaction status 0 will be followed by from 0 to 32 bytes of device dependent status.

Status 2 indicates that, while executing the request, the adapter received an IMS message from the device which was directed at this specific request (Virtual circuit) and the request was advanced to the status phase. The contents of the IMS parameter are returned in the status block (status block count = 4).

Status 3 and 4 are generated by the adapter if the request or subfunction fields are illegal. This should never be generated under normal (debugged) operation.

Status 5 is generated if the adapter and device end up with incompatible handshake messages, for example, an RTS as received from the device and a non-blocked read request was posted to the adapter from the host. The device gets an IMS <Link Protocol Error> to abort the transaction in addition to the host receiving this error status.

Transaction status 6 is reported if a link resync was performed at any time during the transaction. This can happen due to a fatal condition being detected at the device (power fail, parity error, etc). All outstanding transactions associated with virtual circuits are aborted in this way when the link resynchronizes.

NOTE

This status will also be reported for link transactions that are present when various of the control requests (request code 6) are requested. Subfunctions 2, 253, 254, and 255 all perform link resynchronization as part of their function and will therefore generate this status for pending link transactions.

Transaction status 7 is reported if a virtual circuit reset was received from the device.

Transaction status 8 is reported along with a four byte count if a loopback performed by the loopback destination is less than the length requested by the host. The four byte residue count is given as a count of untransferred data bytes.

Transaction status 9 is generated by the adapter if the request is issued while in the wrong card state, ie, either the request is raw mode and the card is not in the raw mode state, or the request is for normal operation and the card is in the raw mode state. This should never be generated under normal (debugged) operation.

Transaction status 10 is reported if an unrecognized or unexpected (ie, out of order) frame type was received. The body of the frame is returned in the status block after the adapter sends an IMS to the virtual circuit and receives the IMS acknowledgement. This should never be generated under normal (debugged) operation.

Transaction status 11 is reported if an unrecognized or unexpected data tag is received. The body of the frame is returned in the status block after the adapter sends an IMS to the virtual circuit and receives the IMS acknowledgement. This should never be generated under normal (debugged) operation.

Transaction status 12 is the result of a logical channel transaction being aborted due to subchannel destruction or a *Reset virtual circuit* request (request 6, subfunction 4) is issued to a virtual circuit associated with another active logical channel. The device will have participated in the IMS acknowledgement prior to reporting this status.

Transaction status 13 is caused by receipt of an insufficient number of parameters in the request block of a transaction (ie, no subfunction feild). This status should not occur in a debugged system. Extra parameters received in the request block will be discarded and will *not* be flagged as an exception.

Status 1 through 13 are reported after an immediate switch to the status phase is performed (internally on the adapter) while processing the request. Status 2 is followed by the 4 bytes from the IMS message which was received by the adapter and Status 8 is followed by a 4 byte residue count, Status 10 and 11 return a 7 byte frame image (of the defective frame that generated the status). Upon entry into any of these error states, the associated request(s) will be advanced 'artificially' to the status phase, ie, no special channel protocol is necessary to handle these exceptions.

IDY: Identify Order

The IDY order returns the Adapter's Identification Message, which appears as follows:

| | | | | | | | |
|----------------------------|---|-------------|---|---|---|---------|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Card ID = 8 | | | | | | | |
| Firmware ID = 0 | | | | | | | |
| Firmware revision | | | | | | | |
| Date code = ??? | | | | | | | |
| MPX=01 | | HW revision | | | | Mode=10 | |
| # of active ports - 1 = 64 | | | | | | | |
| # of requests/port - 1 = 0 | | | | | | | |
| Data blocking | | | | | | | |
| 8192 bytes/block | | | | | | | |

idyblk

This block shows the card's Identification code, 8. Since this document describes the first firmware set to be used on this adapter hardware, the firmware ID is 0. The Date Code will be filled in later. The adapter uses the Logchannel multiplexing scheme, and can process all orders in either byte or word mode.

The adapter can support up to 64 requests pending on the adapter at the same time. Since the concept of a "port" does not fit exactly with A-link, the # of active ports, and # requests per port parameters are not exactly correct. The closest analogy is to make each virtual circuit a port, making 64 ports, and one

Firmware Description

request per port. The adapter requires that each concurrent request to the link have a different virtual circuit.

RTS: Read__transparent__status Order

The RTS order is used to control the Logchannel Multiplexing of the HP-CIO channel. Its usage here is in accordance with that standard.

This adapter will report the following transparent status messages: idle (IDL), switch (SWI), end-of-data (EOD) for early termination of blocked interactions, logical channel destroyed (LCD) as acknowledgement of a WTC order with the destroy logical channel (DLC) request, and asynchronous event sensed (AES) for reporting enabled interrupts to the host.

The adapter does *not* make use of the error trap (ERT) transparent status.

WTC: Write__transparent__control Order

The WTC order is used to control the Logchannel Multiplexing of the HP-CIO channel. Its usage here is in accordance with that standard.

The messages supported are resume logical channel (RLC), end-of-data (EOD) for early termination of host-sourced blocked data, destroy logical channel (DLC), and asynchronous event acknowledge (AEK) to reenable host interrupts.

DIS: Disconnect Order

The DIS order is used to remove either the subchannel connected to the adapter, or the logchannel used by a request pending on the adapter, in accordance with the HP-CIO standard. It is not expected that this order will ever be used.

Channel Program Structure

The channel programs used by this adapter consist, typically, of three orders. These are the CLC order to initiate a transaction, either a RD or WD order to transfer data between the device and the host, and an RS,d order to return status of transaction from either the card and device's point of view. Between the CLC and R/WD order a Logchannel Break must exist, as specified by the HP-CIO standard. Another Logchannel Break must also be inserted between the R/WD order and the RS,d order if the status message is coming from the device. Inserting a Break here is allowed for card-only status messages, but will slow the transaction slightly.

Chaining

An expansion of logical channel Channel programs can be used by replacing all but the last RS,d order with RS and all but the first CLC order with WC orders (and removing the LCN parameters from the *initiation data* block of the WC request blocks). There is no logical channel break between the RS orders and the WC orders. This method can be used to chain an arbitrary number of requests together to execute serially in the same logical channel.

All transaction which have multiple data transfers require a Logchannel Break before each transfer.

Performance Est.

Device adapter attributable overhead has been empirically measured at about 1.2 milliseconds for request 1 subfunction 1 (CS/80 read) of one block of data (8192 bytes) from the device and 16 bytes of command to the device. This number is based on preliminary traces taken of prototype hardware and emulated devices. The number is expected to drop below 1 millisecond.

Details of this analysis are in the process of being documented and will be presented here or appendicized later.

The time needed to transfer data over the link is not included because this is dependent on the transfer rate of the device at the other end. This information alone is not sufficient to determine sequential time to process one CS/80 read.

Card overhead which is not in series with a transfer (i.e. hidden) is not included. And the amount that can be hidden has been conservatively based on current empirical data of a prototype device.

CARD REQUEST PROCESSING

This section details the use and operation of the *Request* and *Subfunction* fields in the CLC data block.

Since the adapter sits on the half-duplex HP-CIO backplane, the A-link protocol restriction for half duplex implementations applies to all writes to the link. This restriction, described in more detail in the "A-link Protocol Standard" document, requires that no DATA messages be sent over the link if any RTR or PTS messages are outstanding. The restriction is implemented entirely by the adapter, and in no way restricts channel programming.

In the following descriptions the term "Device" is used to mean "the thing at the other end of the A-link cable", be it a peripheral or another computer.

Read Device Data (Rq = 1)

The Read Device Data request is used to read data messages from the link into the host. The Subfunction field in the CLC block defines the type of request which will be performed:

Firmware Description

| Subfunction | Request type |
|-------------|---|
| 0 | Single data transfer from device (Read blocked) |
| 1 | Triplet ("CS-80") type transaction (Write, Read blocked, Read) |
| 253 | Read Device Loopback |
| 254 | Read Device Identity |
| 255 | Raw link read |
| all others | not defined |

Rq = 1, SF = 0

This request reads one message from the link into the host. The message may be broken into many message segments, if specified by the B bit in the request field. This request is intended mainly for direct host to host communications, but has other applications.

The channel program for this type of request is roughly as follows:

```
CLC
  < Request block >

<< Logchannel Break >>

RD, may be blocked
  < message from link >

<< Logchannel Break optional >>

RS[,d]
  < transaction status from adapter only >
```

B bit set. When this request is received (i.e. when the Request block has been given to the card) AND an A-link RTS message has been received for the same virtual circuit and data type tag specified in the Request block, an A-link RTR message will be generated to the link. The data type tag in that message will be set to the *Data Type* field in the request block, and the length field set to the value specified in the *Data Length* field. The request being posted to the card, and the RTS message from the link may occur in either order. An RTR will not be generated until both have been received. An RTS received without a read request can cause an interrupt to the host, if desired. See Write Card Configuration.

B bit clear. If the B bit is clear, this request uses the Negotiated mode of communication. When both the request is present and a RQS has been received from the device, the adapter will send a PTS with a length field equal to the minimum of the length in the RQS and the length given in the request block.

Data part. When an A-link DATA message is received with a Data Tag equal to the data type requested, the data from that message will be transferred from the link into the host. If the B bit in the request field is *set*, and the message has not ended, more messages can be accepted at a later time. If the B bit is clear, the (one) message received should have EOM asserted with it..

The status block for subfunction 0 is from the adapter only.

Rq = 1, SF = 1

This request is used for transactions formatted as triplets, for example, CS-80 requests. The first transfer is a write to the device, the second and third are reads. The first and last transfers are limited in length to one message segment (See Data blocking parameter of IDY block for byte count), while the middle read may be "unlimited" ($2^{32}-1$ bytes) in length.

The channel program for this type of request is roughly as follows. Note that although CS-80 terminology is used here, any transaction which uses a triplet format may be used. The adapter does not interpret any part of the data being transferred over the link.

```

CLC
  < Request block >

<< Logchannel Break >>

WD, may not be blocked
  < CS-80 command message >

<< Logchannel Break >>

RD, may be blocked
  < CS-80 Execution message >

<< Logchannel Break >>

RS[,d]
  < CS-80 Report message >

```

The adapter will issue an RTS message for the first transfer (the WDs data block) at its earliest convenience. The data type tag in that RTS message is set to [0]. When an RTR message is received for the correct virtual circuit and data type the adapter signals the backplane to switch to that logical channel and the data block will be sent. The length field is not used since the adapter has no idea how long this message is.

B bit set. The adapter will then wait for an RTS message from the device, signaling that the device is ready to send the "Execution" message. The adapter will then send an RTR message to the device to allow it to send the RDs data. The Length field in the RTR is an echo of the value given in the RTS, as long as it is not greater than that given in the request block. If this field is not supplied (zero) the transfer will be made without length fields, one message segment at a time. An EOM from the device, or an HP-CIO EOD transparent control message will terminate the transfer. If the length gets exhausted before an EOM is seen the adapter will wait for another RTS from the device, and then proceed as before.

Firmware Description

B bit clear. Requests with the B bit clear use the Negotiated mode for the transfer of the Execution data. For this type of transfer the adapter waits until it sees a RQS is received, then responds with a PTS containing a length field which is the minimum of the RQS length and the length given in the request block.

The data type tag is [1] for this transfer.

Report phase.

After this transfer, the adapter will wait for another RTS message to be received, this one for the RS order's data block. When it is received, an RTR message will be sent with a length field of the value given in the RTS and with a data type tag of [2]. The device can then send the DATA block with the device's Report message. When this is received, the backplane is switched to that logical channel, and the transfer made into the host.

An alternative protocol for the status phase is for the device to report abbreviated status (ABS) only. This message is intended to communicate the analogy of a Transaction status of 0 from the card except it is from the device. This status is transferred to the host status block as a zero length device status.

In the normal case the RS block consists of a Transaction status of 0 (from the adapter) followed by data from the device. In case of an error detected within the device, the RS block may contain information regarding the reason; this is handled entirely by the device. If the device detects an error serious enough to abort the transaction, an IMS message generated by the device will cause the adapter to immediately enter an internal status-reporting state and will sink or source bytes of data for WD and RD orders until the RS order is received after which the status block will consist of Transaction status 2 followed by 4 bytes of IMS parameter.

Rq = 1, SF = 253

This request supports several forms of link loopback. The returned data must be compared and verified by the host.

The virtual circuit given in the request block is used for the loopback test. The count must also be provided.

The adapter does not interpret any part of the data being transferred over the link.

This test is non-destructive of card state, and may be used concurrently with other requests.

The channel program for this type of request is roughly as follows.

```

CLC
  < Request block >

<< Logchannel Break >>

WD, may not be blocked
  < device loopback data source >

<< Logchannel Break >>

RD, may not be blocked
  < returned loopback data from device >

<< Logchannel Break >>

RS[,d]
  < status >

```

The adapter will issue a RQL (request to loopback) message for the first transfer (the WDs data block) at its earliest convenience. When a PTL (permission to loopback) message is received for the correct virtual circuit the adapter signals the backplane to switch to that logical channel and the WD data block will be sent. The length field is used but may be overrun by the host if the device gives permission to loopback with a lower length than requested.

After this transfer, the adapter will wait for the RTL (returned loopback data) message to be received. When RTL is received on the correct virtual circuit, the adapter signals the backplane to switch to that logical channel and the data block will be received by the host in the RD block.

The RS block will contain status for the request. If the length of loopback granted by the device is less than requested by the host, the residue will be appended to the RS block data and the status will be 8, indicating a residue count. Note, this status does not indicate any device fault: devices may support a limited loopback buffer size. To date, the minimum length supported is 256 bytes.

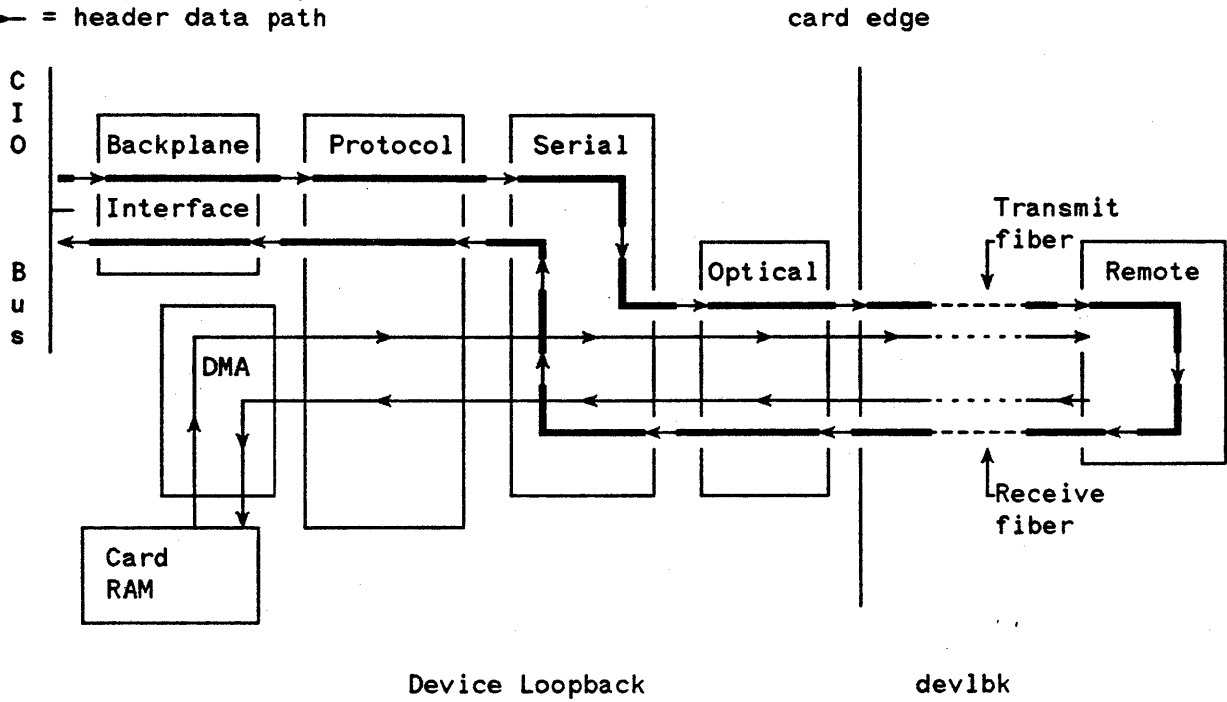
Loopback variations. Device loopback can be used for loopback to one of: 1) the well known virtual circuit (FFFF hex), 2) any virtual circuit, 3) the adapter's own circuitry using a loopback fiber, and 4) the adapter's own circuitry minus the optical components of the adapter.

Loopback types 1 and 2 are expected to be used as part of normal operation whereas types 3 and 4 would only be used as part of a diagnostic strategy. Diagrams for the adapter circuitry covered by each loopback follows. Note a fifth type of loopback, CIO loopback is covered under request 4, subfunction 253 (CIO loopback) and has the same channel program format but the four type covered here are all implemented with variations of this request.



Firmware Description

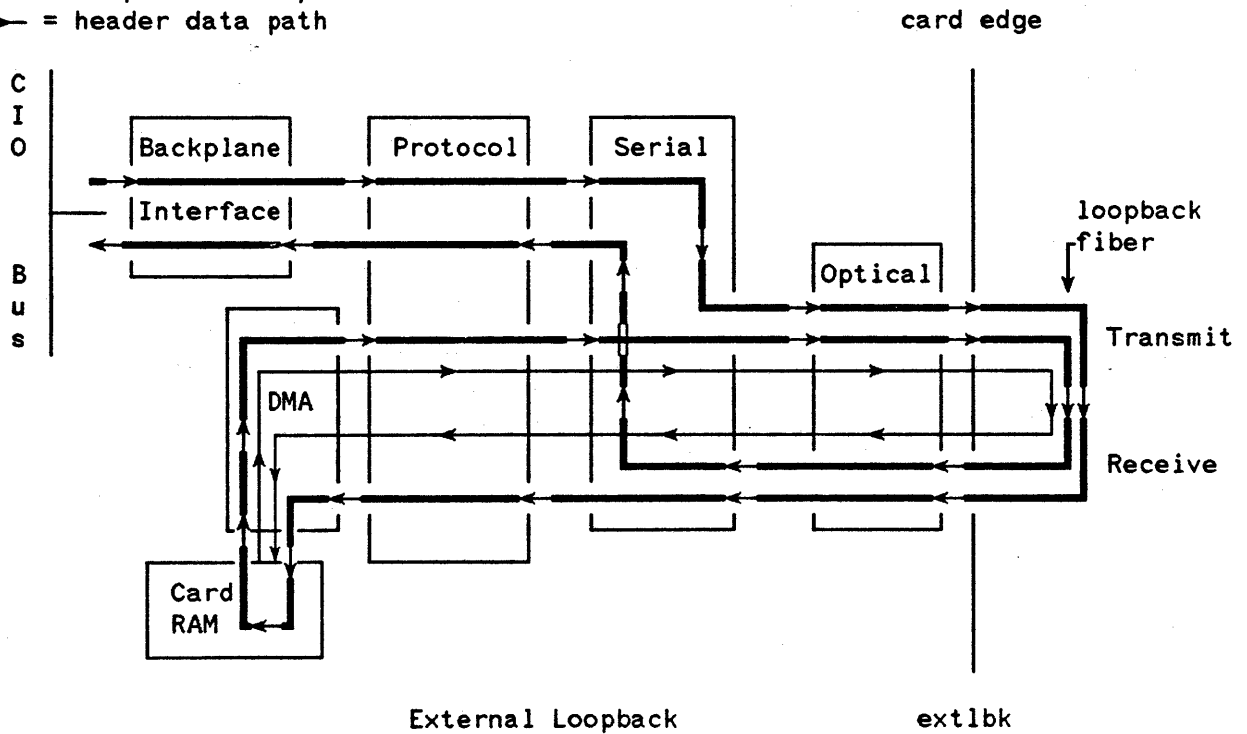
Types 1 and 2. These loopbacks only differ in the virtual circuit specified.

 = loopback data path
 = header data path

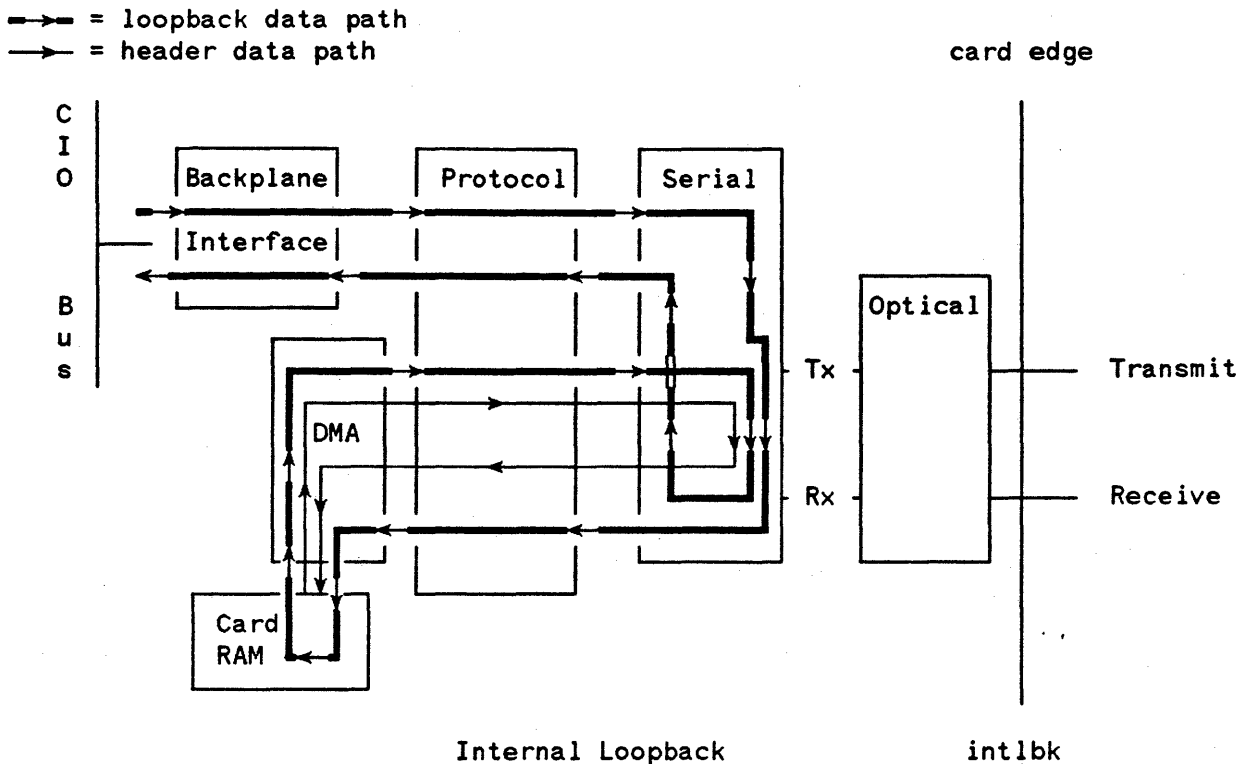


Type 3. Loopback through the adapter's circuitry is accomplished by first connecting a single fiber from the transmitter terminal to the receive terminal on the adapter edge (a.k.a. loopback fiber).

 = loopback data path
 = header data path



Type 4. Loopback through the adapter's circuitry without a fiber is accomplished by first issuing request 6 subfunction 252, enter loopback mode.



Rq = 1, SF = 254

This request reads the Identity block contained within the device itself, as addressed by the virtual circuit number in the request block. It does *not* read the *adapter's* Identity block; this is returned via the HP-CIO IDY order.

The channel program for this request is as follows:

```

CLC
  < request block >

<< Logchannel Break >>

RD
  < 4 byte Identity message >

<< Logchannel Break optional >>

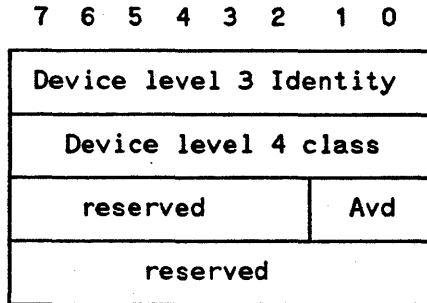
RS[,d]
  < transaction status >
  
```

The device's Identity block, returned with the RD order, comes from the parameter field in the A-link Identity message. The Avd field, which specifies the device's deadlock avoidance scheme is saved by the

Firmware Description

adapter for comparison with its own configuration. The result of this check is available in the "C" bit in the transaction status (returned with the RS order; see Read Subfunction 0). This request can be used to update this bit if one of the ends of the link is changed. (Note, this is also done automatically from time to time.) This is the only time the adapter looks at information passed between the link and host.

The information is passed to the host in the following format:



devidy

The Avd field is defined by the "A-link Protocol Standard" document. The level 3 Identity and level 4 Class fields are defined by the device to which the adapter is connected. Some devices have defined their identity as follows:

Level 3 Identity

- 0 = Host SPU interface (e.g. this adapter)
- 1 = A-mux Disk and peripheral multiplexer

Level 4 Class

- 0 = unknown (e.g. Host interface)
- 1 = Pseudo device (e.g. A-mux)
- 2 = CS-80 device (e.g. disk drive)

Rq = 1, SF = 255

This request is intended for device link protocol development, factory defect analysis and possibly diagnostics. If the adapter is in the raw link access state (request 6, subfunction 255) then the request will read whatever segment arrives next on the link. If the adapter is not in the raw link access state then the request will enter the status phase with a Transaction status of 9, 'Adapter in incompatible state'.

The data returned by this request is unformatted and therefore typically has a CLS (layer 3) header prefix. Refer to the A-link protocol standard for details.

Write Device Data (Rq = 2)

Write Device Data works roughly like Read Device Data, but in the other direction. The subfunctions are defined as follows:

| Subfunction | Request type |
|-------------|--|
| 0 | Single data transfer to device (write blocked) |
| 1 | Triplet ("CS-80") type transaction (write, write blocked, read) |
| 2 | Doublet (A-mux & CS-80 misc) (write blocked, read) |
| 255 | Raw link write |
| all others | Not defined |

Rq = 2, SF = 0

This request transmits one message to the link from the host. The message may be broken into many message segments, if specified by the B bit in the request field. This request is intended to be used for direct host to host communications.

The channel program for this type of request consists of:

```

CLC
  < Request block >

<< Logchannel Break >>

WD, may be blocked
  < message to link >

<< Logchannel Break optional >>

RS[,d]
  < transaction status from adapter only >

```

B bit set. When this request is received an A-link RTS message will be generated to the link. The data type tag field in the message will be set to the *Data Type* field in the request block, and the length field set to the value specified in the *Data Length* field.

When an A-link RTR message is received from the link with the same data type tag transmission will begin. A-link DATA message segments will be sent until either the entire message is sent, or until the length field in the RTR message is used up. If the RTR length is used up before the message has ended another RTS will be sent. The requested length in this RTS will be set to the original value less any data which has been sent.

B bit clear. If the B bit is clear, the Negotiated mode of transfer will be used. The adapter sends a RQS message with the length field set to the length given in the request block, then waits for a PTS from the device. When received, the adapter will send the entire data block to the device in one message, with EOM asserted. Note that the entire block is sent regardless of the length field in the PTS from the device,

Firmware Description

due to limitations of the hardware on the adapter. If the device gets more data than it wanted it can flush the excess.

The status block for subfunction 0 is from the adapter only.

Rq = 2, SF = 1

This request is used for write transactions formatted as triplets, for example, CS-80 requests. The first and second transfers are to the device, and the last a read from the device. The first and last transfers are limited to single A-link message segments, while the middle write may be of "unlimited" length.

The channel program consists of:

```
CLC
  < Request block >

<< Logchannel Break >>

WD, may not be blocked
  < CS-80 command message >

<< Logchannel Break >>

WD, may be blocked
  < CS-80 execution message >

<< Logchannel Break >>

RS[,d]
  < CS-80 report message >>
```

The adapter will issue an A-link RTS message for the first transfer (the first WD data block) at its earliest convenience. The data type tag in that RTS message is set to [0]. When an RTR message is received for the correct virtual circuit and data type the adapter will switch the HP-CIO channel to the corresponding logical channel, and the "command" message will be sent over the link.

B bit set. The adapter will then send an RTS message over the link for the "execution" message. The length field in the RTS message contains the length field from the CLC block, and a data type tag of [1]. A corresponding RTR message is waited for, and when it arrives the HP-CIO channel is switched to the appropriate logical channel.

The adapter then sends as many DATA message segments as are needed, up to the length specified by the RTR message length field. If the RTR length is used up before the message is completely sent, another RTS message will be sent for the difference.

B bit clear. If the B bit is not set the adapter uses the Negotiated mode for transferring the Execution message. The adapter sends a RQS message giving the length set in the request block. Upon receipt of a PTS message from the device, the adapter switches the HP-CIO channel to the appropriate logchannel, and sends the entire data block in one piece.

After the "Execution" message is sent the adapter waits for an RTS message from the device for the "Report" message. When this is seen, the adapter sends an RTR message with no length field and a data tag of [2].

The report phase is performed as in request 1 subfunction 1.

Rq = 2, SF = 2

This request is used for several CS-80 and A-mux operations, including Host to Host communications, and Configure Clears. There are two data transfers, a write followed by a read. The write message may be of any type, as specified by the Data Tag field in the Request block. The read is always of type [2] (CS-80 "Report").

The channel program consists of the following:

```

CLC
  < Request block >

<< Logchannel Break >>

WD, may be blocked
  < message to link >

<< Logchannel break >>

RS[,d]
  < status message from device >

```

B bit set. Requests with the B bit set will issue an RTS for the write when the request is posted to the card. The length field is set to the length given in the request block, and the data type tag is set to whatever is specified in the Data Tag field in the request block.

When an RTR is received from the device, the adapter will begin sending message segments until either the entire message is sent, or the length field in the RTR is "used up". If the RTR length expires first another RTS will be sent for the difference.

B bit clear. Requests with the B bit clear will issue a RQS for the data, with a length field set to the size specified in the request block. When a PTS is received the adapter will send the entire data block in a single message to the device. The data will be flagged with EOM, and have the Data Tag specified in the Request block.

The report phase is performed as in request 1 subfunction 1.

Rq = 2, SF = 255

This request is intended for device link protocol development, factory defect analysis and possibly diagnostics. If the adapter is in the raw link access state (request 6, subfunction 255) then the request will dump the sourced data onto the link.

Firmware Description

If the adapter is not in the raw link access state then the adapter will enter the status phase (ie, source 1 byte of 'garbage' and waiting for the RS order) and will report Transaction status 9, 'Adapter in incompatible state'.

The data sent by this request is unformatted and therefore will typically need a CLS (layer 3) header prefixed to the data block. Refer to the A-link protocol standard for details.

Read Card Information (Rq = 4)

The Read Card Information request reads data from the card local memory into the host. There are NO interactions with the link caused by this request. As the information read is independent of any particular virtual circuit, only the Logchannel number, Request, and Subfunction fields are required in the request block.

The information read is dependent on the subfunction specified in the request block:

| Subfunction | Request type |
|-------------|----------------|
| 22 | Interrupt mask |
| 33 | Global Status |
| 250 | Read Card RAM |
| 253 | CIO loopback |

The channel program for the above subfunctions, except 253, consists of:

```
CLC
  < Request block >

<< Logchannel Break >>

RD
  < card data >

<< Logchannel Break optional >>

RS[,d]
  < transaction status >
```

Rq = 4, SF = 22

This request returns the interrupt mask set by Write Card Configuration subfunction 22. There are no side effects caused by this request.

Rq = 4, SF = 33

This request reads 12 bytes giving the status of the card, as follows:

| | | | | | | | | | |
|--------|------------------------|-----|---|---|---------|-----|-----|------|--|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Byte 0 | OFL | ACT | | > | Link St | JLB | Raw | | |
| 1 | | NMI | P | A | C | S | R | F | |
| 2 | | | | | | | | high | |
| 3 | Cumulative Link Errors | | | | | | | | |
| 4 | | | | | | | | | |
| 5 | | | | | | | | low | |
| 6 | | | | | | | | high | |
| 7 | Elapsed Time (seconds) | | | | | | | | |
| 8 | | | | | | | | | |
| 9 | | | | | | | | low | |
| 10 | | | | | | | | high | |
| 11 | Error Rate | | | | | | | low | |

cdstat

OFL, if set, indicates that the two Jupiter chips are out of lock, and therefore cannot communicate. This bit may be transient in nature.

ACT, if clear, indicates that the Jupiter chip is reporting Link Not Active. This is usually because the link isn't connected to anything or the remote is powered down.

>, if set, indicates that the "More Equal" configuration jumper is in the "1" position.

Link St (link status), if 0 the link is down or (virtually) disconnected; if 1 the nominal error rate threshold has been equaled or exceeded: the link is 'sick' (see Error Rate below); if 2 the link is operating normally.

JLB, if set, the link is in internal loopback mode (Jupiter LoopBack). This will occur while processing the internal loopback transaction (request 6, subfunction 0) or while in the internal loopback state (see request 6, subfunction 252)

Raw, if set, the card is in raw mode. This mode is used in conjunction with requests 1 and 2, subfunction 255: raw link read and write. Raw mode is provided for diagnostic and development and should not be set in an operating system.

NMI, if set, the HP-CIO backplane signal NMI is enabled to be asserted in the event of a link reset by the remote (or by the card itself when the fiber is installed as a loopback).

Firmware Description

P, if not set, indicates the card has detected a fatal error and has attempted to report 'dead or dying' to the host. This is a copy of the state of the *Passed* (or *Operational*) LED on the card front edge.

A, if set, indicates that the card is currently sending or receiving a data transmission. This is a copy of the state of the *Activity* LED on the card front edge.

C, if set, indicates that the adapter has determined that the present adapter/device configuration is incorrect, for example, that there are two "more" or "less" equal ends. The adapter cannot be used until this condition is corrected, which usually involves manually changing a switch on ONE of the two ends of the link. This bit is a copy of the *Configuration Error* LED on the card front edge.

Cumulative Link Errors is a 32 bit unsigned integer which is a count of the number of link errors (CRC errors, coding violations, etc) which have been detected by the protocol chip since the last time the adapter was reset.

S, if set, indicates that the adapter has determined that the link is not performing up to normal expectations, either in the number of CRC errors, Link out-of-lock transitions, or that it is simply not working (ACT=0). This is a copy of the *Signal Error* LED on the card front edge.

R, if set, indicates that the Device is Not Responding to requests. Successful communication with the device, either because of host requests, or the card's periodic checking of Identity, keep this bit off. This is a copy of the *Remote Not Responding* LED on the card front edge.

F, if set, indicates card malfunction detected in self test. This transaction should return a clear F. This is a copy of the HP-CIO self test *Failed* LED on the card front edge.

Elapsed Time is a 32 bit unsigned integer giving the number of seconds which have elapsed since the last time the adapter was reset.

Error Rate is the average error rate over the previous hour. Each hour (60 minutes) the number of link errors which occurred during that hour is stored in this field. This gives a measure of link quality, and can be monitored from time to time to detect trends. The average value for this field with a 1km link should be 144. A value greater than [tbd] puts the adapter into "Sick" state, and will post an interrupt to the host, if enabled. On power up this field is initialized to zero, which will remain there for the first hour. The field is a 16 bit unsigned integer; if more than 65535 errors occur, 65535 will be reported.

Rq = 4, SF = 250

This request reads the contents of a selected portion of card RAM, into the host. The CLC request block requires all but the data tag fields to be specified although the fields are used differently than for other requests as is shown below.

WARNING

This request is an auxiliary diagnostic intended for factory use only. Any unauthorized use of this transaction will cause unpredictable results.

The Virtual Circuit parameter is used as the *segment address* and should be set to 0's. The upper two bytes of the length field are used as the 16 bit *offset address*. The lower two bytes of the length field are used as the number of bytes to transfer in the RD block.

* Preliminary Version *

With a segment address of 0, the upper 8 of 16 bits of the byte count are masked and an after-masked value of 0 will be converted to a count of 256.

Rq = 4, SF = 253

CIO loopback has the same channel program format as request 1, subfunction 253, and all the aspects of the request correspond except that data is only transferred between the host and adapter RAM and will be limited to 256 bytes by the adapter. See read device loopback, request 1, subfunction 253 for details.

```

CLC
  < Request block >

<< Logchannel Break >>

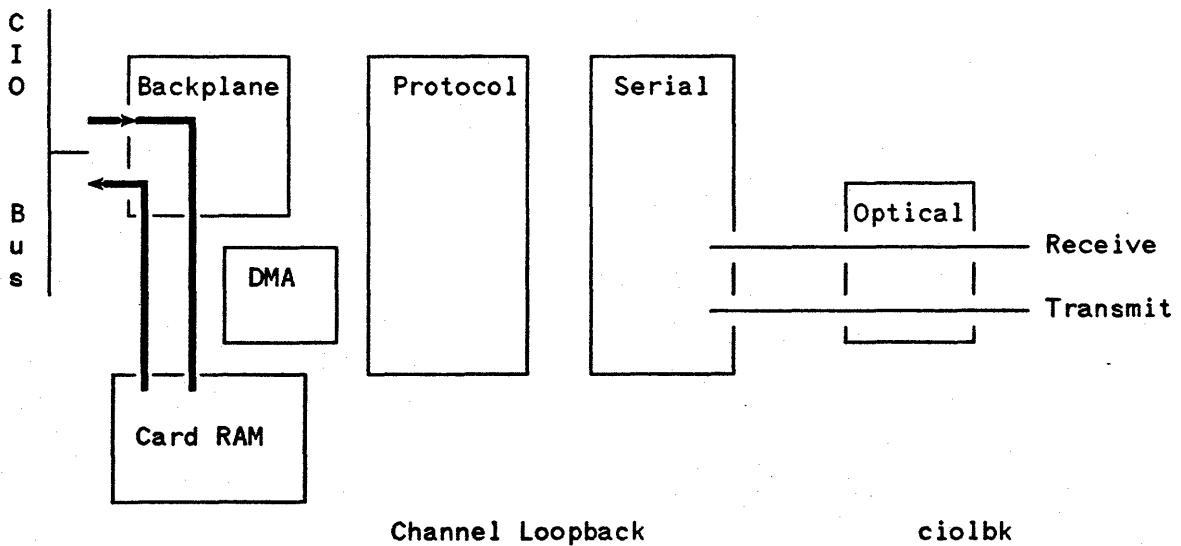
WD, may not be blocked
  < loopback data source >

<< Logchannel Break >>

RD, may not be blocked
  < returned loopback RAM from adapter, maximum of 256 bytes >

<< Logchannel Break >>

RS[,d]
  < status >
    
```



Write Card Configuration (Rq = 5)

This request, Request code 5, is used to write configuration information to the card. The subfunction field determines which information is to be written, as follows:

| Subfunction | Request type |
|-------------|--------------------|
| 22 | Set interrupt Mask |
| 250 | RAM |

The channel program consists of:

```

CLC
  < Request block >

<< Logchannel Break >>

WD
  < configuration info or RAM data >

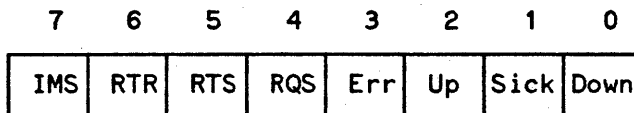
<< Logchannel Break optional >>

RS[,d]
  < transaction status >
    
```

Rq = 5, SF = 22

This request sets the interrupt mask. A "1" in a bit's position *enables* that interrupt. The default at power-on is all zeros.

The interrupt mask is as follows:



aemask

IMS, if set, allows an IMS message received on a virtual circuit for which there is no outstanding request of any type pending on the adapter to cause an interrupt to the host.

RTR, if set, allows an RTR message received for which there is no outstanding write request pending on the adapter to interrupt the host.

RTS, if set, allows an RTS message received for which there is no outstanding Blocked read request pending on the adapter to interrupt the host.

RQS, if set, allows an RQS message received for which there is no outstanding Non-blocked read request pending on the adapter to interrupt the host.

Err, if set, allows the adapter to interrupt the host whenever a link error occurs. Only one interrupt is generated if multiple errors occur before the interrupt is read. This interrupt could be used by proactive diagnostics to log the exact timing of link errors for later possible correlation checks.

Up, if set, allows the adapter to interrupt the host when the link state changes from either down or sick to the up state. Only one interrupt is generated if multiple changes occur before the interrupt is read.

Sick, if set, allows the adapter to interrupt the host when the link state goes from down or up to sick. In the sick state the adapter is able to communicate with the device, but at a marginal level due to link errors or signal condition. This condition should be corrected as soon as possible, since in this state the chances for an undetected error are relatively high (one error per xxxx years).

Down, if set, allows the adapter to interrupt the host when the link state goes from up or sick to down. In the down state the adapter is not able to communicate with the device in any way. All requests to send or receive data over the link are rejected by the adapter.

More information on interrupts is available under *Event Processing*.

Rq = 5, SF = 250

This request writes the contents of a selected portion of card RAM, from the host. The CLC request block requires all but the data tag feilds to be specified although the feilds are used differently than for other requests as is shown below.

WARNING

This request is an auxiliary diagnostic intended for factory use only. Any unauthorized use of this transaction *will* cause unpredictable results.

The Virtual Circuit parameter is used as the *segment address* and should be set to 0's. The upper two bytes of the length feild are used as the 16 bit *offset address*. The lower two bytes of the length feild are used as the number of bytes to transfer in the WD block.

With a segment address of 0, the upper 8 of 16 bits of the byte count are masked and an after-masked value of 0 will be converted to a count of 256.

Control Card (Rq = 6)

These requests send certain control commands to the adapter itself. There may be activity generated on the link, if the adapter is commanded to do so.

The channel program for all subfunctions except 253, consists of the following:

Firmware Description

```
CLC
  < Request block >

<< Logchannel Break >>

RS[,d]
  < transaction status >
```

The command to be executed is defined by the subfunction field in the Request block, as follows:

| Subfunction | Command |
|-------------|---|
| 2 | Resynchronize Link (Leave Jupiter Loopback state) |
| 3 | Remote Link Reset |
| 4 | Reset Virtual Circuit |
| 250 | Execute code at given vector |
| 252 | Enter Jupiter Loopback state |
| 253 | Rq 6, SF 252 - Rq 1, SF 253 - Rq 6, SF 2 |
| 254 | Leave raw link access state |
| 255 | Enter raw link access state |

Rq = 6, SF = 2

This request can be used to 'go online' by enabling the optics and resynchronizing link activity. Execution of this request concurrently with requests performing link transfers will result in those transfers immediately entering an internal status-reporting state and the channel will be advanced to the report phase indicating 'request aborted by link resynchronization', transaction status 6.

Execution of this request is not necessary except in a diagnostic application where spontaneous Resync under host control is desired or operation after being in the Jupiter loopback state is desired without resetting the adapter.

Rq = 6, SF = 3

This request causes the Remote Link Reset bit into the A-link protocol chip to be pulsed. This causes the corresponding pin on the remote end to pulse, and do whatever that means. It is thought that an RLR to a device would cause a reset of that device or its controller. An RLR to another host causes an NMI or some similar "grab its neck and shake" action, e.g. forced cold load.

Currently, the AMUX CS/80 device multiplexer (Alink compatible) will perform an equivalent to power-on reset to all multiplexed devices upon receipt of RLR.

Rq = 6, SF = 4

This request causes the adapter to send a "Reset DLS" message to the virtual circuit given in the request block. This causes the addressed VC to be reset to its default or power-up state, for example, a "Reset Clear" to a CS-80 device.

If the adapter sends or receives a Reset DLS message it will abort any pending request, and remove (forget) any pending asynchronous interrupts on that particular virtual circuit. Aborted requests report a status = 12, 'host aborted transaction indirectly by host (DSC or VC reset)'.

If the reset affects any pending requests on other virtual circuits within the same physical device it is up to the host to abort (via HP-CIO DLC) these requests on the adapter since the adapter has no knowledge of virtual circuit usage.

Rq = 6, SF = 250

This request allows a subroutine to be called anywhere within the address space of the adapter's microprocessor. The CLC request block requires all but the data tag feilds and the lower 16 bits of the count feild to be specified although the feilds used are used differently than for other requests as is shown below.

WARNING

This request is an auxiliary diagnostic intended for factory use only. Any unauthorized use of this transaction will cause unpredictable results.

The Virtual Circuit parameter is used as the *segment address*. The upper two bytes of the length feild are used as the 16 bit *offset address*.

The ensuing channel sequence is usually to enter the status phase once the desired routine is completed, however, the routine itself may be written to follow any channel programming that is desired (especially since the routine may have been downloaded).

The function of the request is to do an inter-segment CALL to the vector specified by the request block parameters.

Rq = 6, SF = 252

This request changes the adapter and link state into an internal loopback (a.k.a. Jupiter Loopback). All transactions will attempt to use the adapter even though the loopback is enabled. This transaction could be used to test the HP-CIO NMI jumper by following the transaction with a link reset (request 6, subfunction 3).

To leave the internal loopback state, use request 6, subfunction 2: re-synchronize link. When entering the loopback state with this request, an internal link resynchronization occurs, so it is not necessary to use request 6, subfunction 2 for the purpose of resynchronizing while in loopback status (thus it is used to exit the loopback state). The card can use raw mode while in the internal loopback state.

Firmware Description

This request is included to aid in development and diagnostics and is not expected to be used as part of any normal path.

Rq = 6, SF = 253

This request has the same channel program as request 1, subfunction 253 (read device loopback). This request performs the loopback as described in 'read device loopback' under the description of loopback type 4 loopback.

```
CLC
  < Request block >

  << Logchannel Break >>

  WD, may not be blocked
  < adapter loopback data source >

  << Logchannel Break >>

  RD, may not be blocked
  < returned loopback data from adapter >

  << Logchannel Break >>

  RS[,d]
  < status >
```

This duplicity of functionality was requested so that this one request would perform what would take three requests using the 'read device loopback' request. This request first enters the Jupiter loopback state and resynchronizes (the same function as request 6 subfunction 252); then a read device loopback is performed (the same function as request 1 subfunction 253) and since the adapter is looped back to itself, the maximum loopback capability without a loopback fiber is exercised. Finally, the link is brought back online and resynchronized (same function as request 6 subfunction 2).

See diagram Internal Loopback under description from request 1 subfunction 253.

Rq = 6, SF = 254

This request undoes subfunction 255. This request is used to return the adapter to normal operation following raw link access.

Rq = 6, SF = 255

This request changes the adapter state such that raw link reads and writes may be accomplished (requests 1 and 2, subfunction 255).

All activity on the link should be terminated by the host BEFORE this request is given, as it may lose incoming and outgoing ALINK headers because of the asynchronous nature of the mode change.

Non raw mode transfer requests will immediately enter an internal status-reporting state and will advance the channel to the status phase reporting transaction status = 7 if such requests are processed while in the raw transfer state.

This request is included to aid in development and diagnostics and is not expected to be used as part of any normal path. This request is not to be used to implement protocols but may be used to test the protocols provided and access exception handling on the remote side of the link.

Request Abort

If at any time the host wishes to stop (abort) an in-process transaction, it may send a Destroy LogChannel (DLC) transparent control message to the adapter, giving the Logchannel Number (NOT the Virtual Circuit number) of the transaction to be aborted. The adapter, if the request is still active and is a link transfer, will send an IMS <Cancel> message to the device. When the IMS is acknowledged, the adapter will send a Logchannel Destroyed (LCD) transparent message to the HP-CIO channel. No action will be taken if the requested Logchannel does not exist, either because it just completed, or it never existed.

The DLC abort process is "nice". It only affects the virtual circuit to which it was directed, and notifies the device that the host wishes the request to be terminated. The device, if it determines necessary, may continue the transaction to a comfortable stopping point before acknowledging the Cancel. This may be necessary to maintain device data integrity.

If the DLC does not (or cannot) succeed the host may issue the more severe abort, Destroy SubChannel, DSC command. This may be needed if the host cannot proceed with the channel program (channel detected an error fetching data from host memory, for example), or if neither the DLC has been acknowledged nor the request completed normally after a reasonable timeout (device hung). Upon receiving DSC, the adapter will immediately terminate the current backplane activity, if any (by DENDING a possible data transfer in progress), and return a status of SubChannel Destroyed, SCD. The host must be very certain that the DSC abort is necessary; if the adapter must abort an in-process transfer to or from the link to comply with the abort, the link will be cleared (resynchronized) to prevent the device from using the truncated data, and to re-establish level-3 synchronization. *All transactions pending on the link, except for the one with data terminated by the DSC, will be immediately put in an internal status-reporting state and will advance the channel to enter a status phase and report transaction status = 6 if this happens, just as if a resynchronize were received from the device end of the link.* The transaction which was terminated with DSC (in the event of DSC in mid-data transfer) will report transaction status 12, *host aborted transaction indirectly (DSC or VC reset)* if it is not terminated first by a DLC.

EVENT PROCESSING

Events can be generated to the A-link adapter from two sources, changes in the state of the link itself, or unexpected messages received from the link. These events can be passed to the host through interrupts.

When an event is detected by the adapter it is queued behind other events which are waiting to be sent to the host. This queuing occurs only if the event is enabled. An event is not queued if it has already

Firmware Description

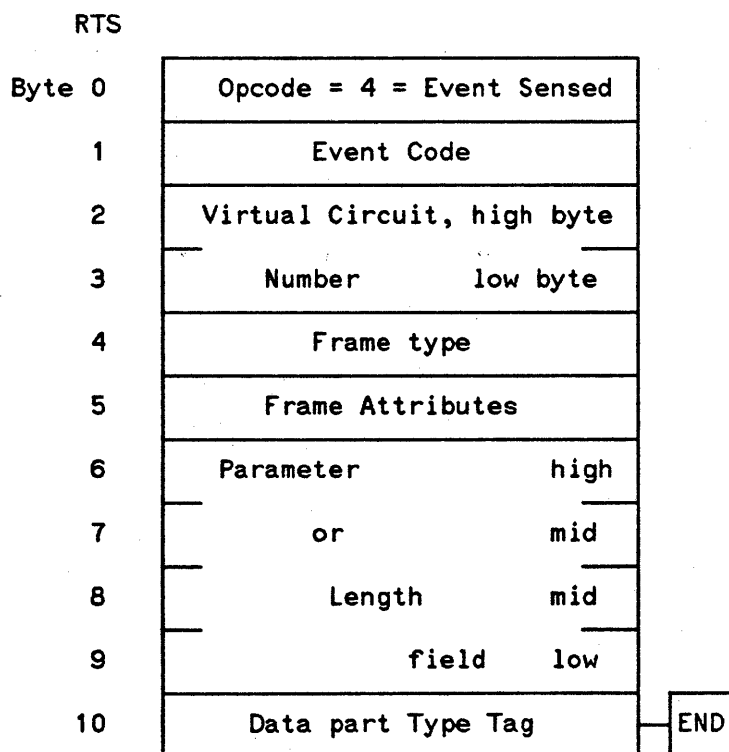
happened, or if the "First" bit is not set in the Frame Attributes field. (If two "link up" events occur, only the first will be queued, and the second lost.) Message type events (e.g. RTS, RTR) are independent for each virtual circuit, so one RTS message event, for example, can be pending for each virtual circuit.

Receiving an RTS, RTR, or RQS event with the "first" bit set in the Frame Attributes field causes a virtual circuit state table to be allocated to hold the message contents for future requests. This causes two effects; the maximum number of these events cannot be greater than the number of virtual circuits the adapter can handle concurrently, and if the message events are not eventually followed by a read or write request to "use up" the event, the state table will remain allocated, reducing the number of "real" requests which the adapter can handle. IMS events do not allocate a state table, nor do RTR, RTS, or RQS messages which do not have the First bit set.

Interrupt Operation

Any enabled event is reported to the host via an HP-CIO Asynchronous Event Sensed transparent status message. Events are enabled via Request 5, Subfunction 22, and any events which were "pending" when the enable for that event was received will generate an interrupt.

The Event Block is formatted as follows:



rtsaes

The event code field is defined as follows:

- 0 = not used
- 1 = asynchronous message received; see Rq 5, SF 22
- 2 = unsolicited message received and discarded
- 3 = link is now up

- 4 = link is now sick
- 5 = link is now down
- 6 = link error

Event code 1 indicates that a message was received on a virtual circuit for which there was no outstanding request. If the message was an RTS, RTR, or RQS then the "First" bit was set in the Frame Attributes field. IMS messages are also reported with this event code.

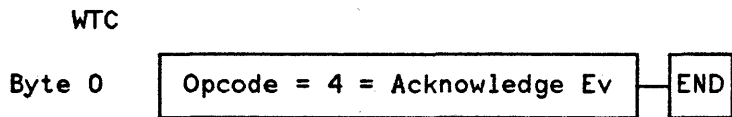
If the interrupt was caused by an RTS, RTR, or RQS message with the First bit set in the Frame Attributes field, that fact will be remembered by the card for the next write or read on that virtual circuit.

Event code 2 is used when an RTS, RTR, or RQS message is received on a virtual circuit for which there is no outstanding request, and which does not have the "first" bit set in the Frame Attributes field of the message. Since First = 0 specifies that this is part of the body of some request, the lack of a pending request on the adapter indicates that the message is from a request which should not exist. The message is forwarded to the host for information purposes and discarded.

For event codes 3-6 only the first two bytes are returned; the event applies to the entire adapter, not just a specific virtual circuit. Event 0 will never occur.

Once an event is reported to the host in this way, all further events are disabled until that event is acknowledged. This prevents the card from flooding the host with events.

The acknowledgment is performed by sending an Asynchronous Event Acknowledge transparent control message to the card. The control message is formatted as follows:



wtcaek

The **USER INTEGRATION** section describes how the A-Link CIO adapter is integrated into a host system and how it is intended to be maintained.

Four major areas are covered in this section: **Installation, Performance Verification, Serviceability, and Upgrade.**

INSTALLATION

The installation of the A-link CIO Adapter requires the following to be present:

- A host system that contains a CIO backplane and an available card slot
- An A-Link CIO Adapter
- A duplex fiber optic cable (HP-27111A-XXXXX)

Installation of the A-link CIO Adapter involves four basic steps:

- Verify that the appropriate A-link driver software is installed
- Select the appropriate **Card Configuration**
- Remove power from the system and insert the A-Link Adapter into the CIO Cardcage
- Attach the fiber optic connectors
- Restore power to the system

Software installation

Refer to the appropriate software installation manual for the particular operating systems and associated hardware.

Selecting Card Configuration

There is one configurable options on the A-link CIO Adapter Version HP27111A: *Equal Mode* A second option, *Channel NMI Enable*, is not available on this version. A third option, *Secondary Power Available* may eventually be supported, but will not be detailed here.

USER INTEGRATION

Initial releases of the A-link CIO Adapter are intended for attachment to the A-MUX Disc Cluster. The A-MUX requires that the Equal Mode jumper be placed in the LESS EQUAL position. The adapter will be shipped with the jumper in this position. Also, because this is the adapter's default position, the card will continue to operate with a disc device if the jumper is not actually present.

NOTE

Software development systems which are using a pair of A-link CIO Adapters connected together should follow the following rules:

| Development Type | card A | card B |
|--|-----------------------------------|-----------------------------------|
| Card to Card (disc emulator mode) A=Normal card B=Disc Emulator | LESS EQUAL | MORE EQUAL |
| Card to Card (SPU to SPU mode) | LESS EQUAL ----- MORE EQUAL | MORE EQUAL ----- LESS EQUAL |

----- -or- -----

[aleqdev]

Backplane Installation

To prepare for installation, first disconnect power to the backplane that the A-link adapter is to be loaded into.

WARNING

Attempt to install an A-link CIO Adapter in a powered backplane or system is a quick way to dramatically prove the validity of OHM's law and the general lack of resistance of the human body to such proofs. Thus, the appropriate legal warnings and whatnots will go in this spot

Backplane Installation of the A-Link CIO Adapter is simply done by grasping the extractor handles of the adapter and then aligning the board with the card guides.

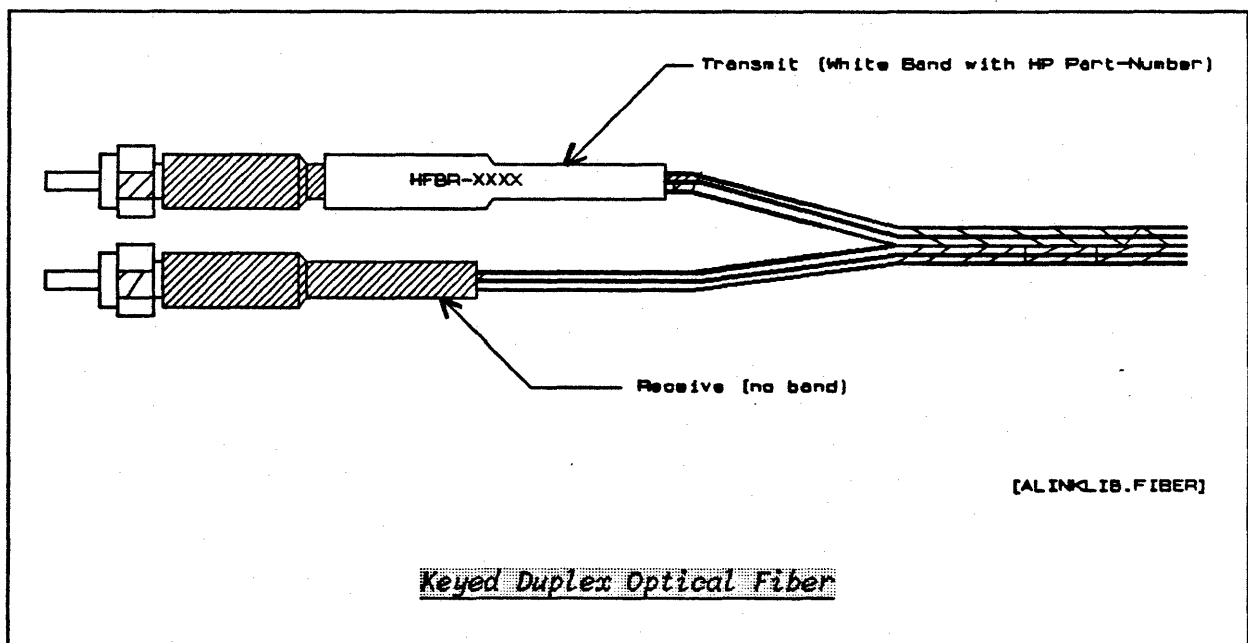
(We will have a photograph illustrating this, one day)

Using the extractor handles for support, firmly insert the card all the way into the card cage. Something will click (somewhere) when the board is properly inserted.

(If something "cracks" instead of "clicks"...)

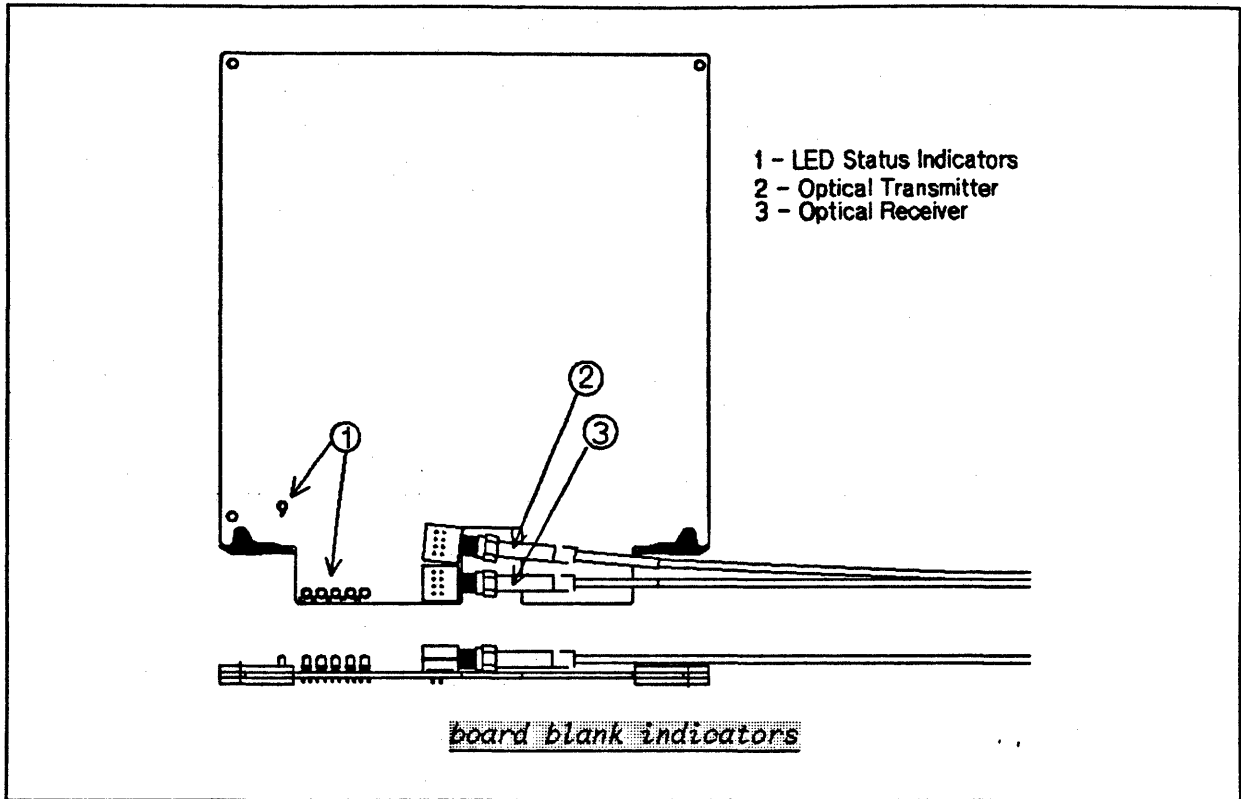
Optical Connection

The standard Fiber Cable (HP-27111A-XXXXX) is color-keyed for connection to the A-link CIO Adapter. Each half of one end of the duplex fiber has a colored band around it. The band is located near the actual optical connector on each fiber. The white-banded fiber always connects to the optical Transmitter.



The location of the Transmit (Tx) and Receive (Rx) optical connectors is shown in the following illustration.

USER INTEGRATION



The barrel housings of the optical connectors are also color-keyed. The transmitter housing is light-grey; the receiver housing is dark-grey. Correct connection is described in the table below:

| Connection | Fiber Band | Optical Barrel Color |
|------------------|------------|----------------------|
| Transmitter (Tx) | White Band | Light Grey |
| Receiver (Rx) | none | Dark Grey |

[fiberkey]

To attach one of the fiber connectors to the board, grasp the metal housing of the fiber connector and insert it into the appropriate connector (Tx or Rx) and rotate the metal housing *CLOCKWISE*. To remove a fiber connector from the board grasp the metal housing and rotate it *COUNTER-CLOCKWISE* until the connector pulls easily away with little force.

Once the card has been properly connected power may be returned to the host system.

PERFORMANCE VERIFICATION

Two types of performance verification are supported by the A-link CIO Adapter: **Hardware and Software.**

Hardware Verification Selftest

During hardware verification, the adapter's data paths are exercised and ###% of "stuck at" faults are detected and appropriate failure codes are returned. Further verification may be accomplished by optional front plane and backplane tests (under SPU/channel control).

Basic.

The **Basic** selftest is invoked following a Channel Reset and provides ###% node coverage of the adapter. Failure status is noted by the [F] frontplane LED and to the backplane by testing Read sense[PST]. The backplane channel interface and the frontplane fiber optic and analog support circuitry are not normally tested during the **Basic** self test.

Extended.

The **Extended** selftest is invoked by an *Addressed Device Clear* (Write_Control[DCL],Write_Control[DEN]) and provides the same node coverage as the basic selftest . The only difference between the two tests is that a more comprehensive RAM test is performed.

Failure Codes.

The adapter **Failure Codes** may be noted on the front panel LED's.

At the time this ERS is published there is no means of reporting the failure code information to the host SPU through the CIO channel.

USER INTEGRATION

alinkstc

| CSR leds | AREAS COVERED | FAILURE |
|-----------------|--|------------------------|
| 111 * | Processor | ROM Test |
| 110 | Processor | 186 Test |
| 101 | Processor | RAM test |
| 100 | Processor Arbiter Protocol controller | PRONTO test |
| 011 | Processor Arbiter Protocol Controller Fiber Optic Conversion | JUPITER Test |
| 010 | Processor Arbiter Backplane Adapter | PASSPORT Test |
| 001 | Processor Arbiter Protocol Controller Fiber Optic Conversion Backplane Adapter | DMA Transmit Path Test |
| 000 | Processor Arbiter Protocol Controller Fiber Optic Conversion Backplane Adapter | DMA Receive Path Test |

***NOTE - 1- ON, 0- OFF**

External Loopback.

Further testing may be accomplished by connecting the A-link adapter's optic transmitter and receiver together. This may be done by connecting the remote ends of the fiber with a special coupling device (HP#xxxxxx) or by removing the local fiber and connecting the local receiver and transmitter with a standard 30 meter fiber (HP#8120-xxxx).

External Loopback covers an additional ### of the adapter circuitry, specifically the frontplane analog and optics. A response to a card status request should indicate *Remote Present, Signal Acceptable, and a Configuration Error*

The A-link adapter cannot determine that a loopback cable has been connected. It will, however, normally determine that a *Configuration Error* is present and will permit only diagnostic activity to occur on the fiber front plane.

This test is performed by issuing a series of requests to the adapter. (see Firmware Description section).

Remote Device Loopback.

Remote Device Loopback provides a means to test the integrity of the A-link from local adapter through the remote device. This test is performed by issuing a series of requests to the adapter. (see Firmware Description section).

Functional Verification

Functional Verification of the A-link subsystem once the HP27111A has successfully completed selftest is performed continuously by the card through **Well Known Virtual Circuit Operations** in which the remote device is monitored. Further verification of the remote device may be accomplished by sending the appropriate commands to the remote via the HP27111A.

Well Known Virtual Circuit Operations.

The Well Known Virtual Circuit (WKA) is a special virtual circuit reserved for link control and status. WKA operations are performed initially by the HP27111A following a link resynchronization event (such as successful poweron or selftest) until the remote device responds with the correct identification procedure.

Failure of the remote to respond will cause the HP27111A to light the No Remote [R] LED, and to indicate the No Remote state in the card status block. The LED will remain lit until the remote device responds correctly.

Once the link has initialized, the HP27111A periodically interrogates the remote device, at approximately one minute intervals. If the device does not respond, the [R] LED will light and again remain lit until the remote device responds correctly.

The Well Known Virtual Circuit may also be used for remote device data transfer operations, however, it is advised that WKA be used mainly for device loopback. Also, on certain Alink sub-system devices, the Well Known Virtual Circuit may follow a different data path than other Virtual Circuit requests. The Remote Device manual should be consulted.

If an HP27111A is acting as the Remote Device, it will process all WKA requests locally on the card *unless the card is in RAW mode.*

USER INTEGRATION

Remote Device Verification.

Remote Device Verification consists of whatever tests are necessary to verify performance of the remote device, aside from remote device loopback. The Remote Device Manual should be consulted for more information on these matters.

SERVICEABILITY

The A-link CIO Adapter has a set of tools and troubleshooting techniques to maintain a high level of serviceability, when necessary, in the field. The board contains a minimum of field replaceable parts, thus the key to serviceability is providing an efficient means of determining when an adapter or fiber connecting cable is incapable of performing to specification. Following is a description of diagnostic tools along with an outline of a possible Troubleshooting plan that utilizes them.

Diagnostic Tools

The A-link CIO Adapter can be serviced with a basic diagnostic tool set. The tool set would include both hardware and software. The goal is to *MINIMIZE* the amount of additional hardware that a support person must have on hand, by providing various means of communicating card status.

Firmware Interface.

The Firmware Interface provides numerous hooks for diagnostics and field support. While the Requests and Status responses are detailed in the section on firmware description section, the most used card requests are listed below.

For more details on the structure of these card requests, refer to the FIRMWARE DESCRIPTION section

Card Requests

| Request Type | Rq | SF | Description |
|---|----|-----|--|
| Read Device Loopback | 1 | 253 | Initiates the loopback of data messages from the host at various levels of looping. |
| Raw Link Read | 1 | 255 | Reads next data received from link. May include A-link Layer 3 headers as well as layer 3 data. |
| Raw link Write | 2 | 255 | Sends data directly onto link. Useful for stimulating remote device in diagnostic mode. |
| Read Global Status | 4 | 33 | Returns copy of LED States, Link State, and Error Statistics |
| Read Card RAM | 4 | 250 | Reads buffer of data from card RAM. Useful in dump of miscellaneous card state. |
| CIO Loopback | 4 | 253 | Initiates the loopback of data from the Channel through card RAM and back to the Channel. |
| Write Card RAM | 5 | 250 | Writes a buffer of data into card RAM. Useful for forcing the card into particular states or to download code. |
| Resynchronize Link (leave Jupiter Loopback) | 6 | 2 | Resynchronizes Link and automatically forces exit from Jupiter Loopback state. |
| Execute at Given Vector | 6 | 250 | Causes card to begin execution at specified address vector. |
| Enter Jupiter Loopback | 6 | 252 | Causes card to enter Jupiter Loopback state. All data is looped internal to card. Optical interface is placed offline. |
| Read Device Loopback (Jupiter Loopback Shorthand Version) | 6 | 253 | Read Device Loopback type#4. Automatically performs Jupiter Loopback state control operations. |
| Enter Raw link access state | 6 | 254 | Allows channel direct I/O access to link through Raw link read and write requests. Adapter suspends A-link layer 3 management. |
| Leave Raw link access state | 6 | 255 | Returns to normal operational mode. HP27111A manages A-link Layer 3 interface. |

In addition to the card level interface, diagnostics can also interrogate the primitive CIO channel interface registers. In particular, the following registers provide the following useful testing features:

chanhint

| OPERATION | BIT(s) | DESCRIPTION |
|---------------|--------|--|
| Read Sense | PRE | when asserted, indicates that an I/O device (e. g., A-link) is present in the addressed slot |
| Read Sense | PST | when asserted, indicates that the I/O device currently addressed has successfully passed its local selftest |
| Read Sense | ARQ | when asserted, indicates that the I/O device has new status present its Status register |
| Write Control | DCL | when asserted, places the addressed I/O device in a reset state (Device Clear) |
| Write Control | DEN | when asserted following the assertion of Write Control[DCL], removes the addressed I/O device from reset state (Device ENable) |

NOTE

Accessing the Read Sense register in the period between the initiation of some type of adapter reset (*Channel Reset, Power On, Addressed Device Clear*) and the completion of its associated self-test interval will yield indeterminate data!

Event Detection. As detailed in the firmware section, the A-link CIO adapter is capable of detecting and reporting a number of asynchronous events that may be monitored by diagnostic applications.

Further details on EVENT DETECTION are found in the FIRMWARE DESCRIPTION

Link State. A change in link state status will cause the adapter to record and optionally post this status in a transparent status message to the backplane. In particular this mechanism is useful for detecting the link entering the *SICK* or *DOWN* states.

Immediate Status (IMS). The Immediate Status (IMS) mechanism allows the remote device to inform the host of events occurring at the remote. In particular, error conditions may be reported through this mechanism. IMS is ordinarily reported through an Asynchronous Event Sensed transparent status message.

For more details on IMS format consult the ERS or Manual for the remote device in question

Status Indicators.

The A-link CIO adapter provides a means of status indication which is extremely useful during diagnostics: local LED's. The LED state is available by performing a Global Status request.

A discussion of the status indicators follows:

[F] Failed Self Test LED. This LED indicates the response of the adapter to the last self-test that was initiated and is also used to interpret the remaining *RED* LEDs. This LED conforms to the CIO Standard for selftest lamps.

- LED *ON* at selftest initiation
- remains *ON* during self test
- if *ON* after selftest interval completes, it indicates a selftest failure. if *OFF*, card has successfully completed selftest
- LED should then remain in its current state until the next selftest

[S] Signal LED. This LED indicates that the adapter is having difficulty extracting data from the received *Signal*. The state of this LED is updated periodically by the card. Thus the intensity of the LED provides a quick indication of the quality of the incoming signal. Since the A-link adapter has a programmable link error detection threshold, and that error detection is a condition for lighting the *Signal* LED, varying this threshold will also cause an intensity change.

Total absence of a signal will cause the LED to light at full intensity.

[C] Configuration LED. This LED indicates that it is unable to proceed with communications with the remote device because its equal mode jumper is in the wrong position.

Note, that when the device is configured for external loopback through an optical fiber, *it must report a CONFIGURATION ERROR.*

[R] Remote LED. This LED indicates that the remote device fails to respond to periodic requests for identification following a resynchronization of the link. The requests for identification take the form of A-link layer 3 Request Identity (RQI) headers which expect a response of an A-link layer 3 Identify (IDY) header.

The RQI headers are sent out at an interval of 1 request/minute and are useful in determining when a remote device is present on the link but is not yet communicative.

[P] Passed and Operational LED. This LED is lit to indicate that the card has passed self test and is operational. It ordinarily remains lit until an error condition on the card causes it to go offline. Error conditions that will cause this are **Link Protocol Errors, Link Data Parity Errors and Data Structure Faults.**

Link Protocol Errors occur when the card receives a frame of information that violates the A-link layer 2 protocol.

USER INTEGRATION

Link Data Parity Errors occur when one of the NMOS devices that implement the Backplane Adapter and Protocol Controller blocks reads in data that is corrupt. Since the link data path is in question at this point, the card will turn off its operational LED before attempting to report a Dead or Dying (DOD) message to the host.

Data Structure Faults occur when an inconsistency is discovered in the internal data structures that the local processor uses to manage data transactions on the link.

This LED will blink following any form of card reset until a *Connect Subchannel Command* is received from the SPU through CIO.

[A] Activity LED. This LED is used to indicate that the card has just sent out A-link layer 3 header information on to the link. Since this is a random event, depending upon card requests for link access, the intensity of the LED is again an indication of the occurrence of this state.

Even when no card requests from the channel are pending on the link, the card will turn the LED on whenever it sends out its periodic RQI to the remote device.

Optical Tool kit.

The **Optical Tool Kit** is the only additional hardware set necessary for support and maintenance of the A-link CIO Adapter. It consists of the following Items:

- Fiber Optic Connector Assembly Tooling Kit (HFBR-0100)
- Optical Power Meter
- Optical Power Source

Tooling Kit.

I'll have to play with one of these one day...

Optical Power Meter. is used to monitor flux levels at the fiber. Currently the **Photodyne 11XE** power meter and Amphenol SMA connector are recommended. Typically, the fiber's SMA connector is threaded onto the power meter's adaptor and the power (in dbM) is observed after 5 seconds (to allow settling).

Optical Power Transmitter. is used to drive a fixed amount of flux onto a fiber. Currently the **Photodyne 8XE-B** Optical Transmitter with Amphenol SMA receptacle is recommended. Typically, the fiber's SMA connector is threaded onto the power transmitter's adaptor in order to drive the fiber. For the 8XE, the display indicates the amount of current that is driving the source LED. This current is adjustable with a small flathead screwdriver and should be set to ##mA.

Remote Device Indications.

Physical access to the Remote Device to which the A-link CIO Adapter is connected to is a useful tool in diagnosing the state of the Adapter and the link. Comparison of the LED states between adapter and remote device is a simple process to determine appropriate service responses.

Indicators on both devices (local and remote) also allow the end user to do preliminary diagnostics of his own and provide useful information to the response center which may save a service call.

Field Troubleshooting

NOTE

THE FOLLOWING TROUBLESHOOTING OUTLINE IS ONLY A SUGGESTION AND MAY NOT EXIST IN ITS DESCRIBED STATE AT CARD RELEASE. SOME SYSTEMS MAY NOT IMPLEMENT ALL FEATURES

Field Troubleshooting can be broken up into six areas dependent upon two factors:

- whether the procedure *MUST* be performed on-site
- the impact of the test on the system containing the HP27111A

Criteria for the two factors are as follows:

alinkacc

| Access Type | Description |
|-------------|---|
| ON-Site | Requires physical, visual access to the system, adapter, fiber, and remote device. May require measurement and modifications to the previous. |
| Console | Requires system access through a terminal connection which may either be at a remote service center OR ON-Site. Does not require any visual checks or physical modifications. |

USER INTEGRATION

alinkimp

| Impact | Description |
|-----------------------|--|
| NON-Disruptive | Any tests or procedures that do not interfere with pending activity either within the system or on the card and link itself. |
| Disruptive | Any tests or procedures which will cause all current activity on the card to be <i>ABORTED</i> , but do not interfere with any other part of the SPU. |
| DESTRUCTIVE | Any tests or procedures which force the SPU & Channel into an initialization state, such as <i>Power Cycling</i> or <i>Front Panel Resets</i> , thereby destroying all current operations (and getting the customer edgy...) |

Which yields the following test matrix:

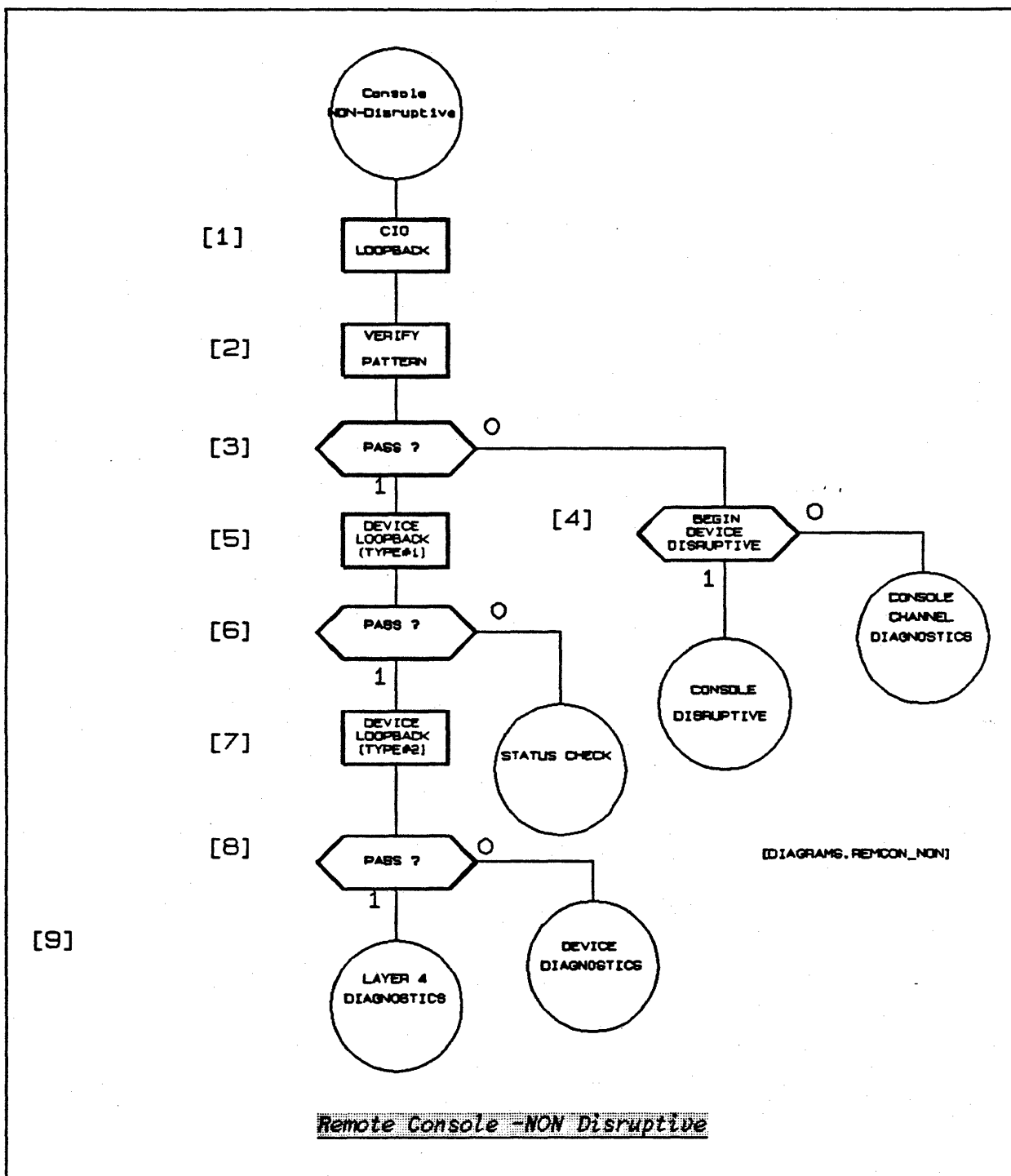
alinkmat

| | ON-Site | Console |
|-----------------------|--|---|
| NON-Disruptive | Power Check Cable Connections LED Check Remote Indicators | Channel Interface Channel Loopback Card & Link Status Device Loopback Type #1 <i>(Well Known Address)</i> Device Loopback Type #2 <i>(NON-Well Known Address)</i> |
| Disruptive | Flux Check Device Loopback Type#3 <i>(Self Loopback Fiber)</i> | DCL Selftest Raw Mode Communication Device Loopback Type #4 <i>(Internal Jupiter)</i> |
| DESTRUCTIVE | Destructive Flux Check Equal Mode Reconfiguration Device Loopback Type#3 <i>(Self Loopback Fiber)</i> | pound on the reset button... |

Console - Non Disruptive.

This set of procedures attempts to determine the general state of the card, link, and remote interface. When a possible error is found, the procedure tree suggests a course of action.

During the following examples, it is assumed that the necessary logchannel breaks and logchannels switches are occurring at the appropriate time and these details are omitted.



USER INTEGRATION

Channel Loopback is used to loop a buffer of data from the channel to the HP27111A RAM and then back to the channel. It is a good exercise of the channel interface.

[1] CIO Loopback. First we check the ability of the card to communicate across the CIO backplane. A CIO Loopback request is sent to the card. Next, a 256 byte pattern buffer is sent to the card with a Write Data (WD) Order. The pattern buffer is returned with the next Read Data (RD) Order.

[2] Verify Pattern. The contents of the buffer just received are compared against the contents of the pattern that was sent. If the patterns match and the size of the buffer returned was 256, then the test was successful.

[3] Pass?. If the channel communication path appears to be solid, the test will proceed.

If not (PASS=0), either the channel or the HP27111A channel interface is assumed to be suspect.

[4] Begin Console Disruptive?. At this point it is recommended that the card be reinitialized and perform a self-test (Begin=1).

If not (Begin=0), a test of the CPU-Channel path is advised.

[5] Device Loopback Type #1 (WKA). This path attempts to determine to what extent the system can communicate to the remote device. The first step is to issue a Device Loopback request with *Virtual Circuit=OFFFh* (WKA) specified in the CLC block. A WD order and pattern buffer is sent by the channel. The next RD order should cause the card to return the data.

[6] Pass ?. The success of the loopback request is determined by examining the data returned via

the RD order as well as the status in the RS block, and checking the transaction status field. Ordinarily a status of 0 indicates no error.

If successful (Pass=1) the card will attempt to communicate across a non-WKA path.

If failed (pass=0), further information on the state of the link will be read.

[7] Device Loopback Type #2 (non-WKA). At this point in the test, the HP27111A is pretty much vindicated, and the integrity of the device is in question. The next step is a Device Loopback request with *Virtual Circuit <> OFFFh* specified in the CLC block is issued. Non-WKA loopback is a more extensive exercise of the remote device. Restrictions on the number used for the Virtual Circuit may be imposed by the remote device. The order sequence following the CLC is the same as for Type #1 Loopback.

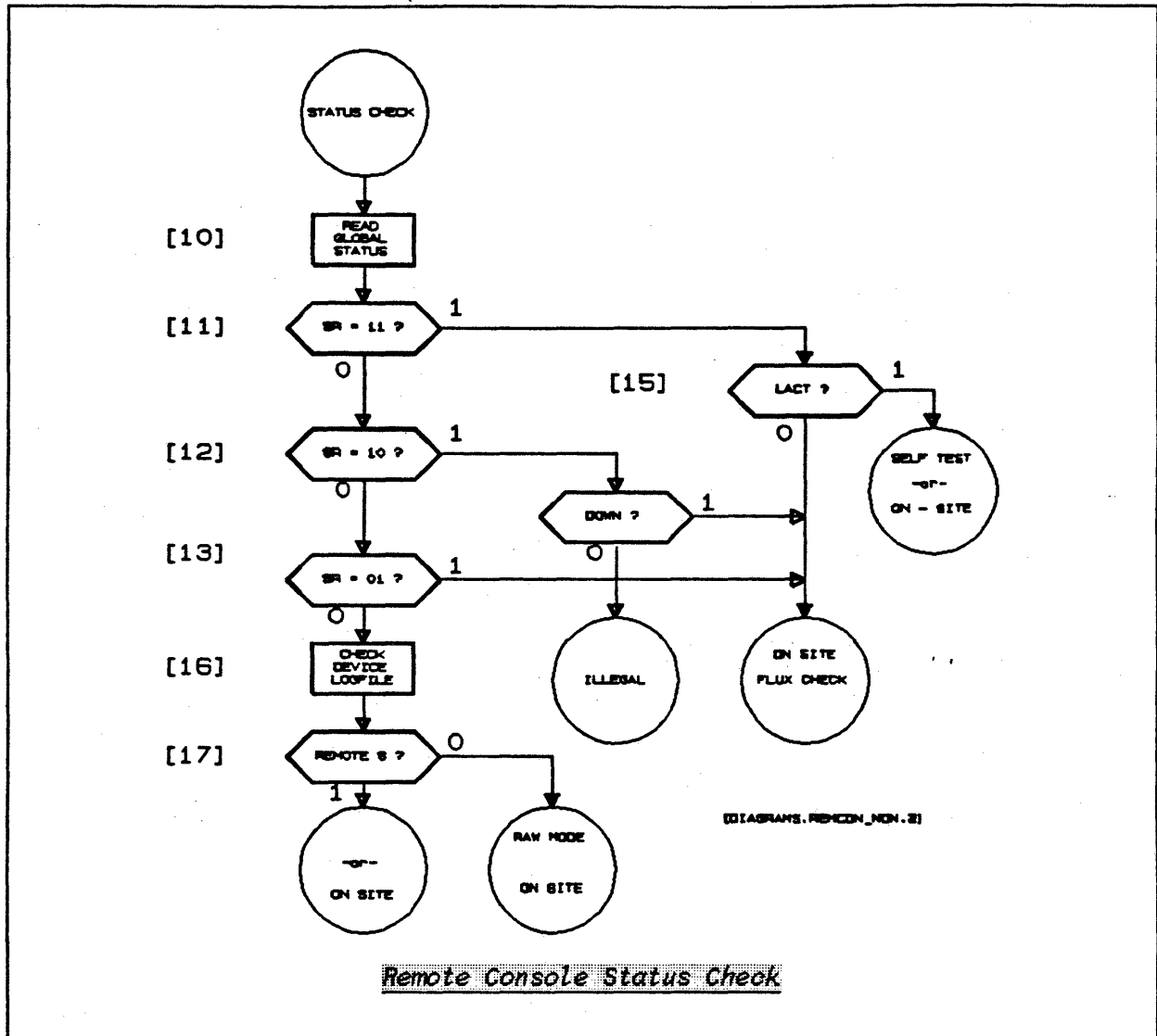
[8] Pass ?. Once again, the data returned is checked and the transaction status in the RS status message is verified.

If successful (Pass=1), problems must be at a higher software level. If the remote device is an A-Mux, it is suggested that a level 4 CS-80 loopback be invoked.

If failed (Pass=0), diagnostics should be invoked on the device. The HP27111A has a very low probability of being the FRU at this point.

[9] (omitted).

step 9 has been omitted



[10] Read Global Status. This path attempts to determine the state of the link. A Global Status request is issued, and the contents of the status block returned are examined, beginning with the marginal Signal [S] and Remote not present [R] bits.

[11] SR=11 ?. When true (1), the A-link connection is in question. A check will be made to try to further isolate the error.

When false(0), consider the other cases.

[12] SR=10 ?. When true (1), the remote device seems to be present, but the signal is marginal. In the card's perspective, the link is probably *Down* due to the excessive error rate. The link state should be checked.

When false(0), consider the other cases.

[13] SR=01 ?. When true (1), signal quality is acceptable, but the remote device is not responding. The pathway between local transmitter and remote device is suspect.

When false(0), and considering that the Device Loopback request likely timed out at the host, the problem area is probably the pathway between the local transmitter and the remote

USER INTEGRATION

device's receiver. The best guess at this point is to check the logfiles for indications that the remote is experiencing a high link error rate.

[14] Down ?. Checks for excessive link error rate, which would preclude any link activity.

When true (DOWN=1), the HP27111A will refuse to perform any device data transfer requests, but, as noted earlier, the card still has been able to communicate with the remote device. The optic communication path between the remote device transmitter and local receiver is in question here, and On-site tests are the next best alternative.

When false (DOWN=0), the card is providing inconsistent information and Roseville Networks Division should be called.

[15] LACT ?. Checks for presence of an optical signal on the receive link.

When true (LACT=1), transitions are present on the link, but of questioned signal quality as noted earlier. The next options are either disruptive or On-site testing.

When false (LACT=0), no transitions are detected. The pathway between remote device and local receiver is questionable. On-site or disruptive testing are necessary at this point.

[16] Check Device Logfile. The system diagnostic should check the remote device link error log for excessive error status reports. The status reports are generated through the Asynchronous Event [AES] reporting mechanism. Excessive link errors are the equivalent of the remote device asserting its own "S" indicator.

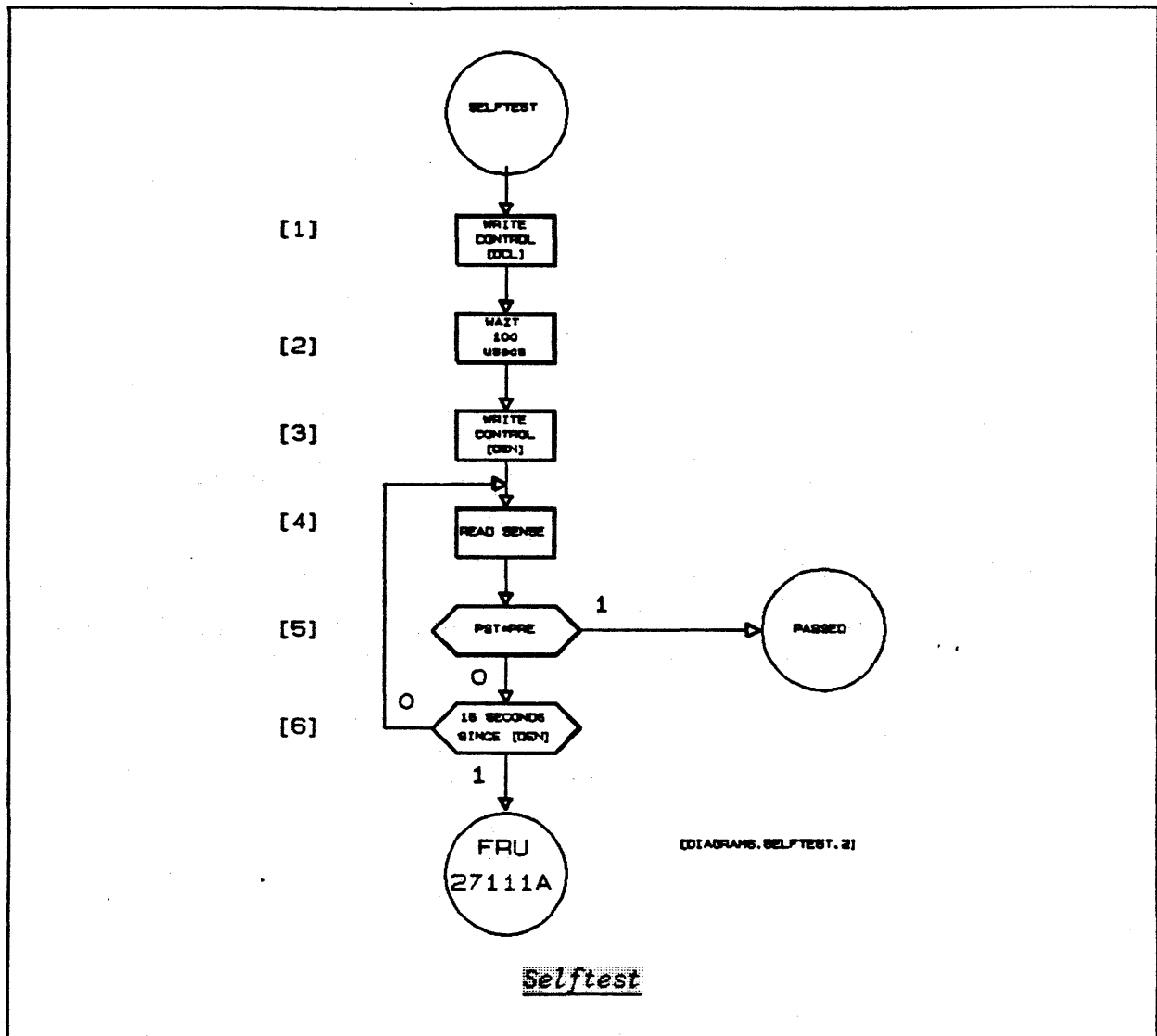
[17] Device S ?. When remote device signal quality is poor (Remote S=1), the next appropriate step is to perform a self test on the HP27111A and then proceed to On-site diagnostics.

When entire link signal quality appears acceptable (Remote S=0), the card believes the remote is present (R=0), but will not perform a Device Loopback, the best bet is to enter Raw Communications mode and use any raw remote diagnostics that are available. If none exist, then perform a self test on the HP27111A and then proceed to On-site diagnostics.

Console - Disruptive.

This set of procedures attempts to isolate the FRU at the expense of any system activity that is presently associated with the card. They should typically be run when either the local HP27111A or the remote device is suspect.

DCL Selftest. uses Addressed Device Clear to invoke the HP27111A's Extended Selftest. Selftest failure should cause the card to be replaced.



[1] Write Control[DCL]. The card should be reset using the CIO Addressed Device Clear mechanism. This will cause the card to perform the extended self test. The results of the selftest can be determined by checking the Read Sense register. The Addressed Device Clear selftest is initiated by a Write Control[DCL] operation. This places the adapter in a reset state.

[2] Wait 100 microseconds. The card must remain reset for at least 100 microseconds to assure that all circuitry has been reinitialized.

To re-enable the card, a Write Control[DEN] operation takes place. Once this operation

occurs, the adapter will begin its extended test of on-card circuitry.

To determine if the card has actually passed selftest the contents of the sense register must be read.

When the card has successfully passed self-test, the Passed Self Test (PST) and Present (PRE) sense bits will be set (PST*PRE = 1). At the least, the PRE bit should be set.

If the card has passed selftest then it is now ready to be connected to the backplane and proceed with further tests. If not then the elapsed time since the release of reset should be checked.

USER INTEGRATION

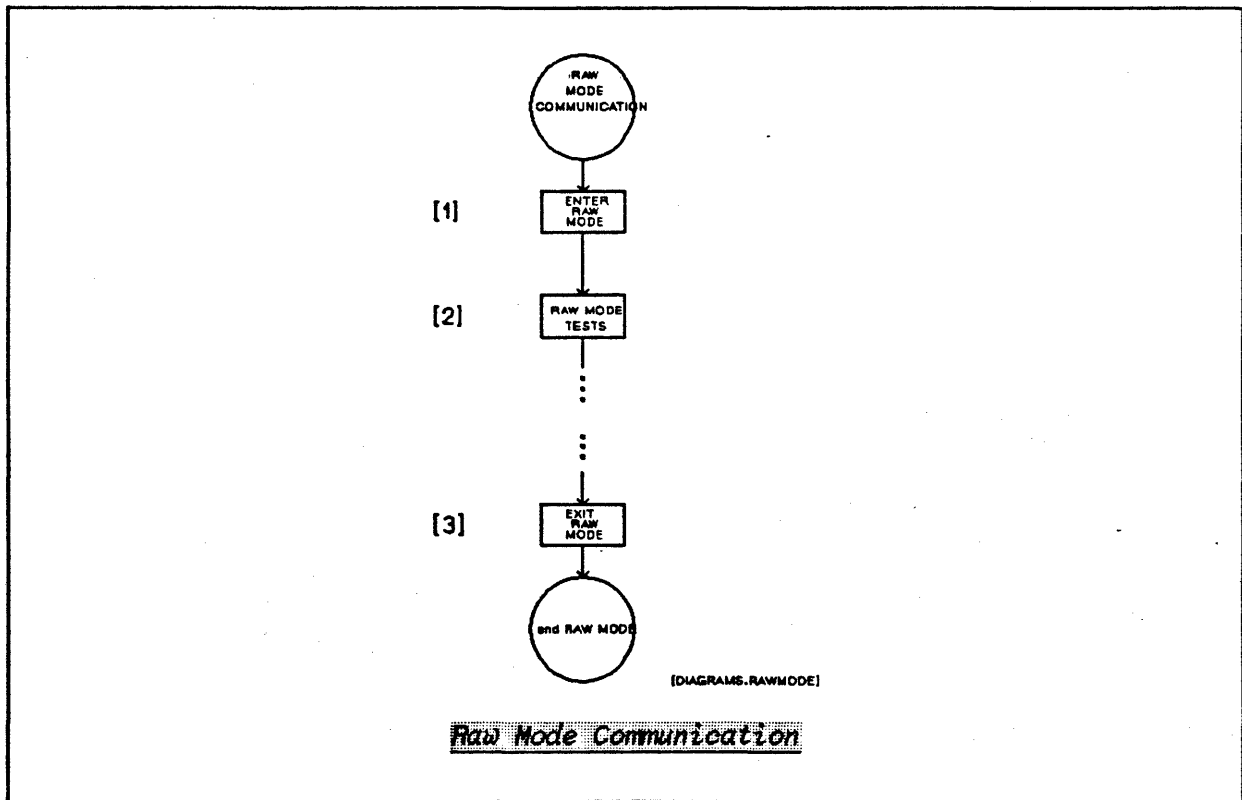
The card should respond with the passed selftest indicator within fifteen seconds of the Write Control[DEN] operation. If this amount of time has not elapsed then the Read Sense operation should be retried.

If the card has not responded affirmatively after 15 seconds, it should be considered the FRU.

NOTE

Future firmware enhancements of the HP27111A may allow certain diagnostic procedures to continue even after a card fails the Device Clear test. Further information on this capability is obtainable from the author.

Raw Mode Communication. is a special feature of the HP27111A which allows the SPU to directly control data and headers that are sent out on to the link. The following procedure shows how to invoke this mode of operation. Specifics of the actual data used in the Raw Mode interaction will have to be provided by the diagnostic package for the remote device and are *NOT* part of the HP27111A product.



[1] Enter Raw Mode. Raw mode is entered by issuing an Enable Raw Mode request to the HP27111A.

[2] Raw Mode Tests. Appropriate tests for the remote device would be performed here. They would consist of Raw Mode Read and Raw Mode Write requests issued to the adapter. Basically, the SPU has direct control over the A-link Layer 3 data transfer during Raw Mode.

For more information on A-link layer 3, consult the A-link Specification.

[3] Exit Raw Mode. Once testing has completed, the SPU should issue a Disable Raw Communication Mode request to the adapter to return the adapter to it's normal, CS-80, state.

Console - DESTRUCTIVE.

The current diagnostic tree has no direct entry into this procedural area.. However, it is the discretion of the diagnostician to calmly invoke a reset through the console...

ON-Site - Non Disruptive.

These procedures are mainly visual checks and measurements to verify that the A-link subsystem has been correctly configured. Some procedures require a set of optical test tools as well as access to the HP27111A and remote device.

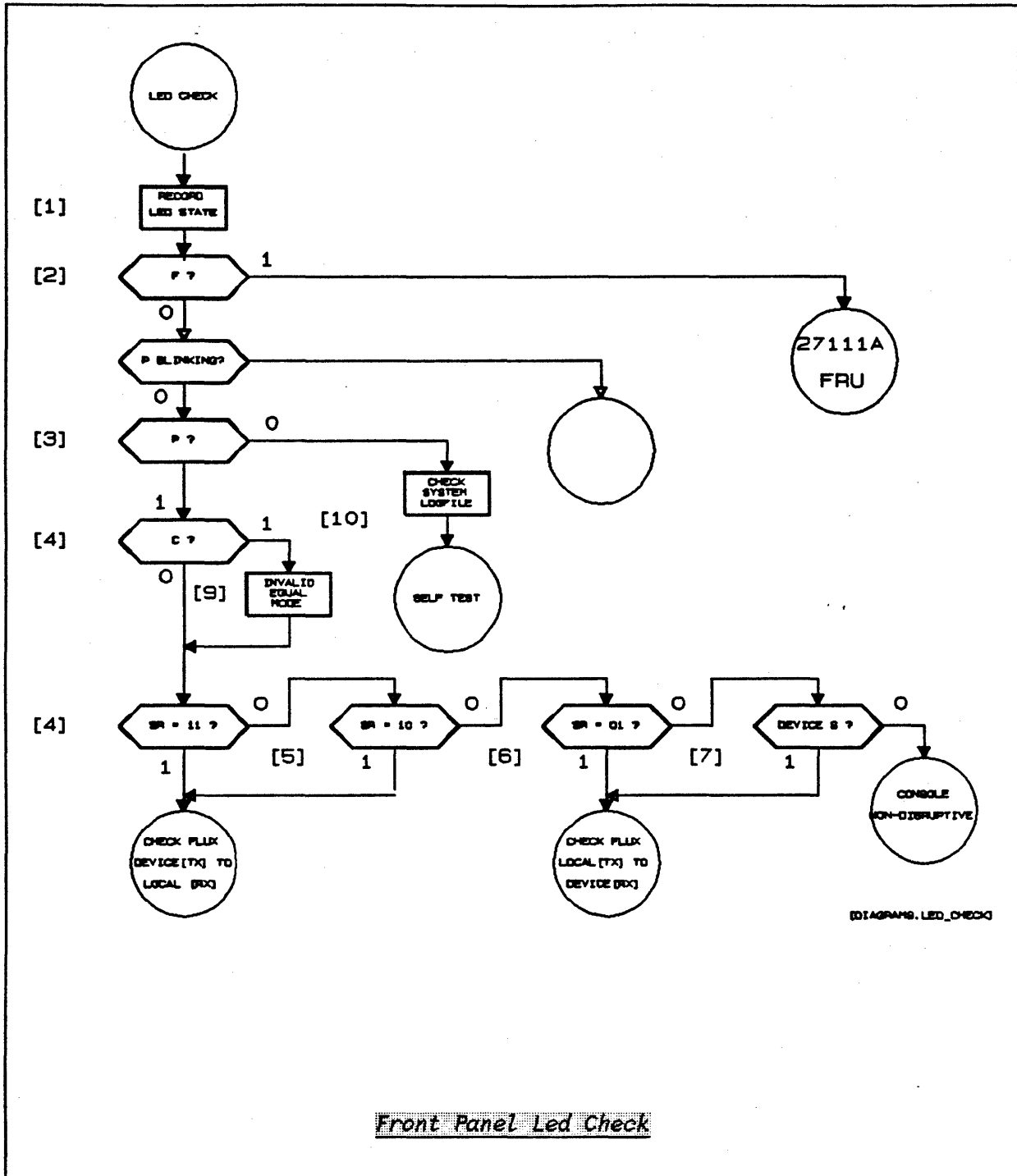
Power Check. verify that SPU and remote device are powered.

Cable Connections. Verify that both the HP27111A and remote device have their receivers and transmitters appropriately connected.

| Connection | Fiber Band | Optical Barrel Color |
|------------------|------------|----------------------|
| Transmitter (Tx) | White Band | Light Grey |
| Receiver (Rx) | none | Dark Grey |

[fiberkey]

LED Check. The LED Check's main use is in guiding the ON-Site Diagnostic person to which fiber's to check and whether an optical loopback connector should be installed.



Front Panel Led Check

[0] Record LED State. The state of the LED's should be recorded at this point for reference sake. The record should indicate the relative intensity of the LED's.

[1] F ?. If failed ([F]led=1), then the card failed self-test and should be replaced.

If the [P]led is blinking, it indicates that the card has not yet received a Connect Subchannel Command from the backplane. The primary suspect at this point is the connection between

the host SPU and the adapter's backplane interface.

[2] P ?. If Passed and oPerational ([P]led =1), then the remaining LED's indicate the link state.

If (P=0), the card is considered dead and the system logfile should be checked to determine what may have caused the fault.

[3] C ?. When ([C]led=1), the local card has the equal mode jumper in the wrong position. Since changing the state of this jumper requires a power cycle, this procedure should be performed after all other intended procedures have been completed.

If (C=0), either the card is correctly configured, or the remote device status has not been read to determine whether the configuration is correct.

[4] SR=11 ?. When true (1), the A-link connection is in question. In particular, a problem seems to lie between the remote device transmitter and the local receiver pathway. The

best step at this point is to perform a Device Tx to Local Rx Flux check.

[5] SR=10 ?. When true (1), the remote seems to be present, but the signal is marginal. The best step at this point is to perform a Device Tx to Local Rx flux check.

[6] SR=01 ?. When true (1), signal quality is acceptable, but the remote is not responding. The pathway between local transmitter and remote device is suspect. If possible, the remote device received signal status should be checked.

[7] Device S ?. All remote devices should have a Marginal Signal Quality indicator.

When true (1), the link signal quality as observed at the Remote Device's receiver is marginal, and a Local Tx to Device Rx flux check should be performed.

When false (0), the signal quality for the entire link appears acceptable. A Console NON-Disruptive test is recommended at this point.

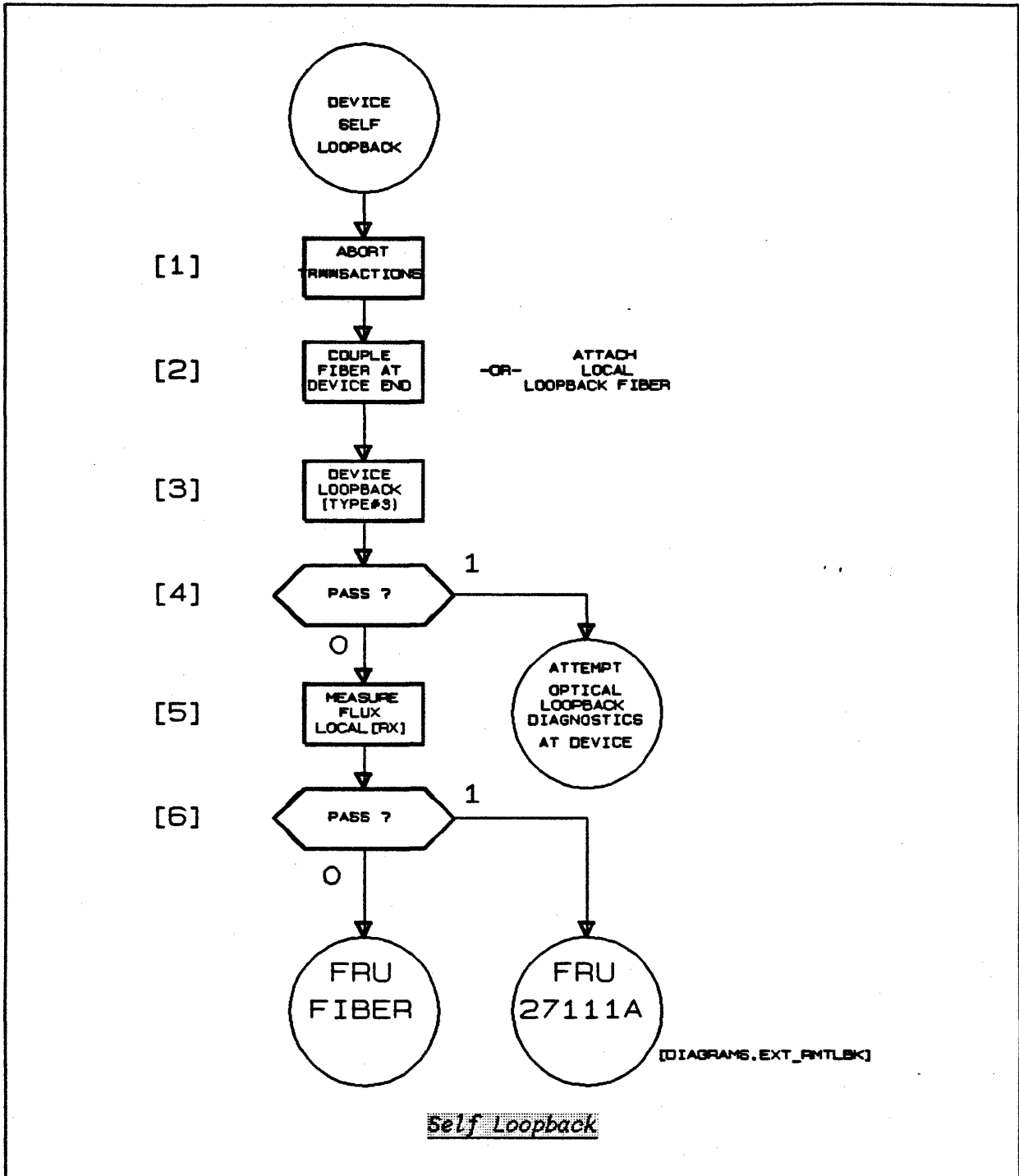
Device Indicators. verifies the state of any available remote device indicators. Appropriate response to error indications should be determined by consulting the diagnostic manual for the remote device.

ON-Site - Disruptive.

In some cases it may be possible to perform tests on the card without removing power from the channel that contains the card.

Device Loopback: Self connection. This is the traditional loopback state in which the HP27111A has its local Transmitter and Receiver connected together by an external fiber. It uses Device Loopback Type# 3.

USER INTEGRATION



[1] Abort transactions. All current transactions should be aborted

[2] Couple Remote Fiber. The remote end of the duplex fiber should be coupled together using the fiber coupler (HP-#####-#####). If a fiber coupler is not available, any fiber may be used that connects the HP27111A's local Transmitter and Receiver together.

| |
|-------------|
| NOTE |
|-------------|

The length of the fiber from local Transmitter to local Receiver must not exceed 500 meters!

[3] Device Loopback Type#3. A Device Loopback Type#3 request should be issued to the adapter. The appropriate **WD** and **RD** orders should follow. Any virtual circuit may be used for this operation, since the card is effectively offline.

[4] Pass ?. The success of the loopback request is determined by examining the data returned via the **RD** order as well as the status in the **RS** block, and checking the transaction status field. Ordinarily a status of 0 indicates no error.

If (pass=1), the HP27111A and the fiber appear solid, and suspicion is transferred to the remote device. If the remote device is capable of performing its own external optical loopback this should be done.

If (pass=0), then either the HP27111A or the fiber has failed.

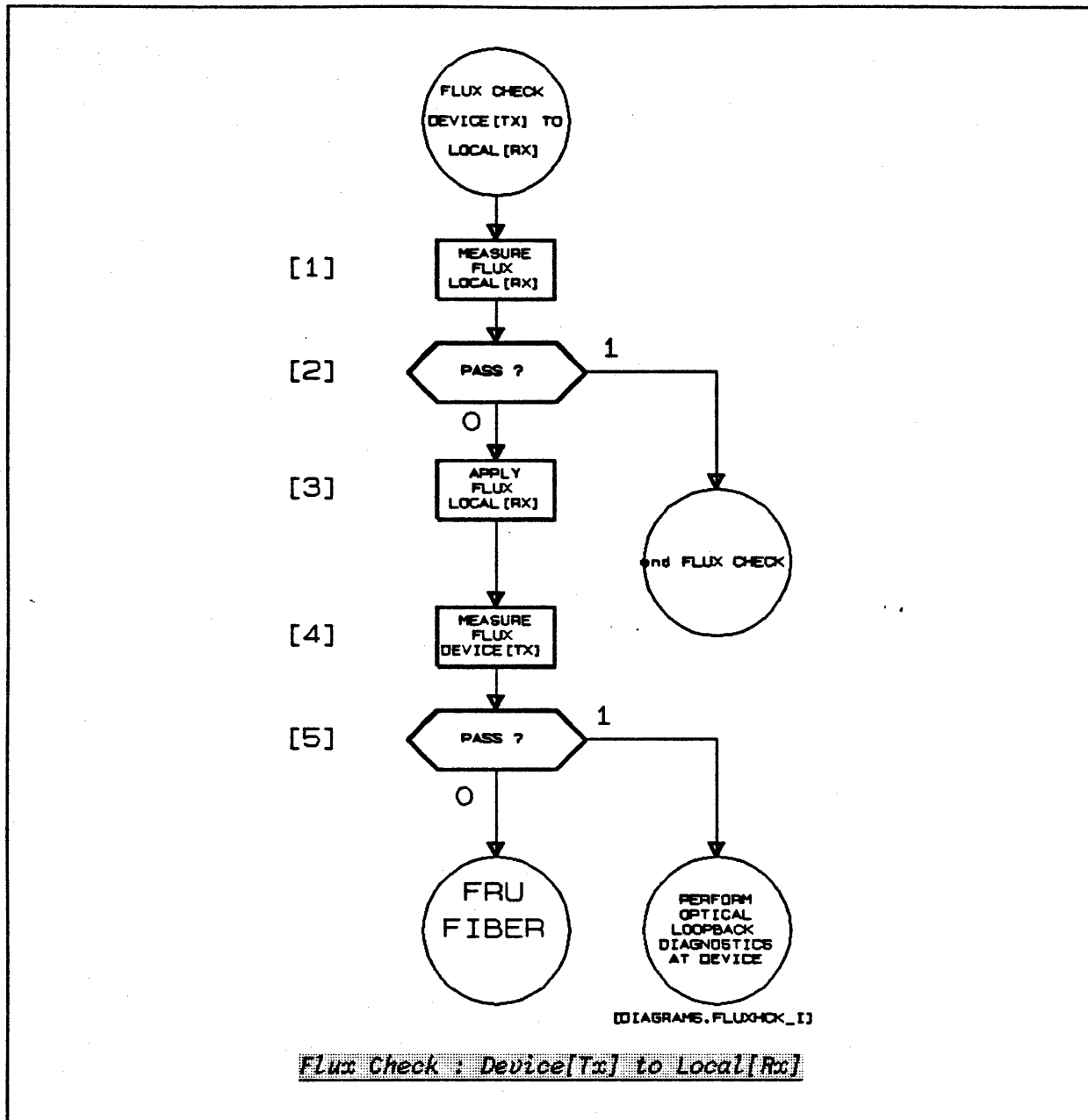
[5] Measure Flux:Local Rx. A measurement of flux at the local Rx is performed. This flux originated at the local transmitter, since we are in a fiber loopback mode.

[6] Pass ?. If (Pass=1), then the HP27111A is the FRU, and the failure should be attributed to the Optical Receiver Path.

If (Pass=0), then the Fiber is the FRU.

Flux Check. measures the flux levels, when possible, on the optical fiber.

USER INTEGRATION



Flux Check: Device Tx to Local Rx:

If (Pass=0), then the path from the Device Tx is in question.

[1] **Measure Flux: Local Rx.** The flux at the Local Rx fiber should be measured to see whether it falls within the acceptable range according to specifications.

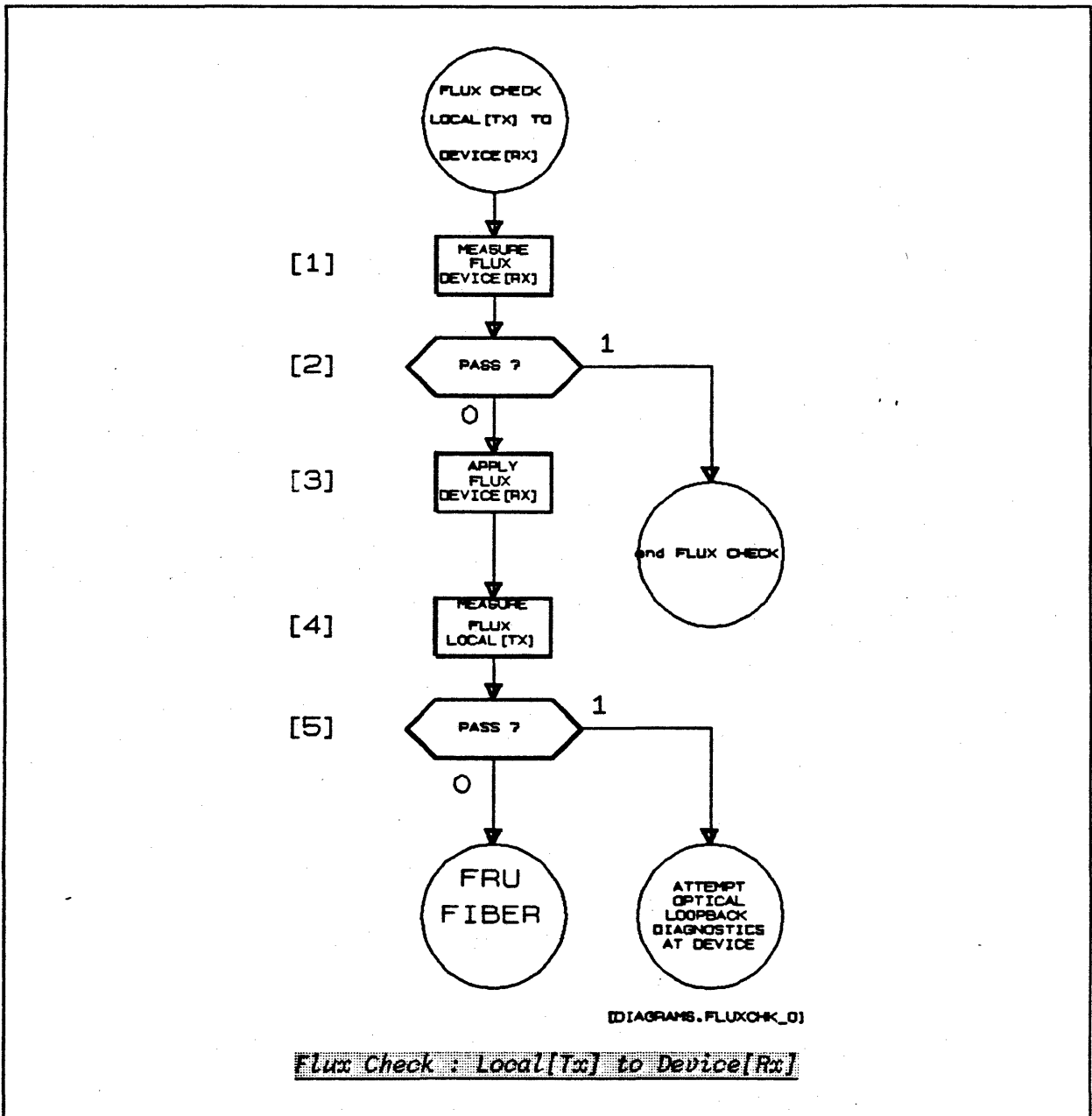
[3] **Apply Flux: Local Rx.** To save a little time, the portable optical source is used to apply flux to the local Rx.

[2] **Pass ?.** If (Pass=1), then the flux check is complete. The next steps would be console non-disruptive tests, followed by device loopback tests of type #3 or #4.

[4] Measure Flux: Device Tx. The flux at the Device Tx fiber should be measured to see whether it falls within the acceptable range according to specifications.

[5] Pass ?. If (Pass=1), then the Remote Device is in question, and any loopback tests that can be performed on the optical ports should be done at this time.

If (Pass=0), then the Fiber is the FRU.



Flux Check: Local Tx to Device Rx

[1] Measure Flux: Device Rx. The flux at the Device Rx fiber should be measured to see whether it falls within the acceptable range according to specifications.

USER INTEGRATION

[2] **Pass ?.** If (Pass=1), then the flux check is complete. The next steps would be console non-disruptive tests, followed by device loopback tests of type #3 or #4.

If (Pass=0), then the path from the Local Tx is in question.

[3] **Apply Flux: Device Rx.** To save a little time, the portable optical source is used to apply flux to the Device Rx.

[4] **Measure Flux: Local Tx.** The flux at the Local Tx fiber should be measured to see whether it falls within the acceptable range according to specifications.

[5] **Pass ?.** If (Pass=1), then the Remote Device is in question, and any loopback tests that can be performed on the optical ports should be done at this time.

If (Pass=0), then the Fiber is the FRU.

ON-Site - *DESTRUCTIVE.*

This set of procedures should be treated as "last resorts", in that they require powering down of the system. In particular, changing the Equal Mode jumper should be delayed as long as possible in the procedure as to access the jumper the card must be removed from the channel's card cage.

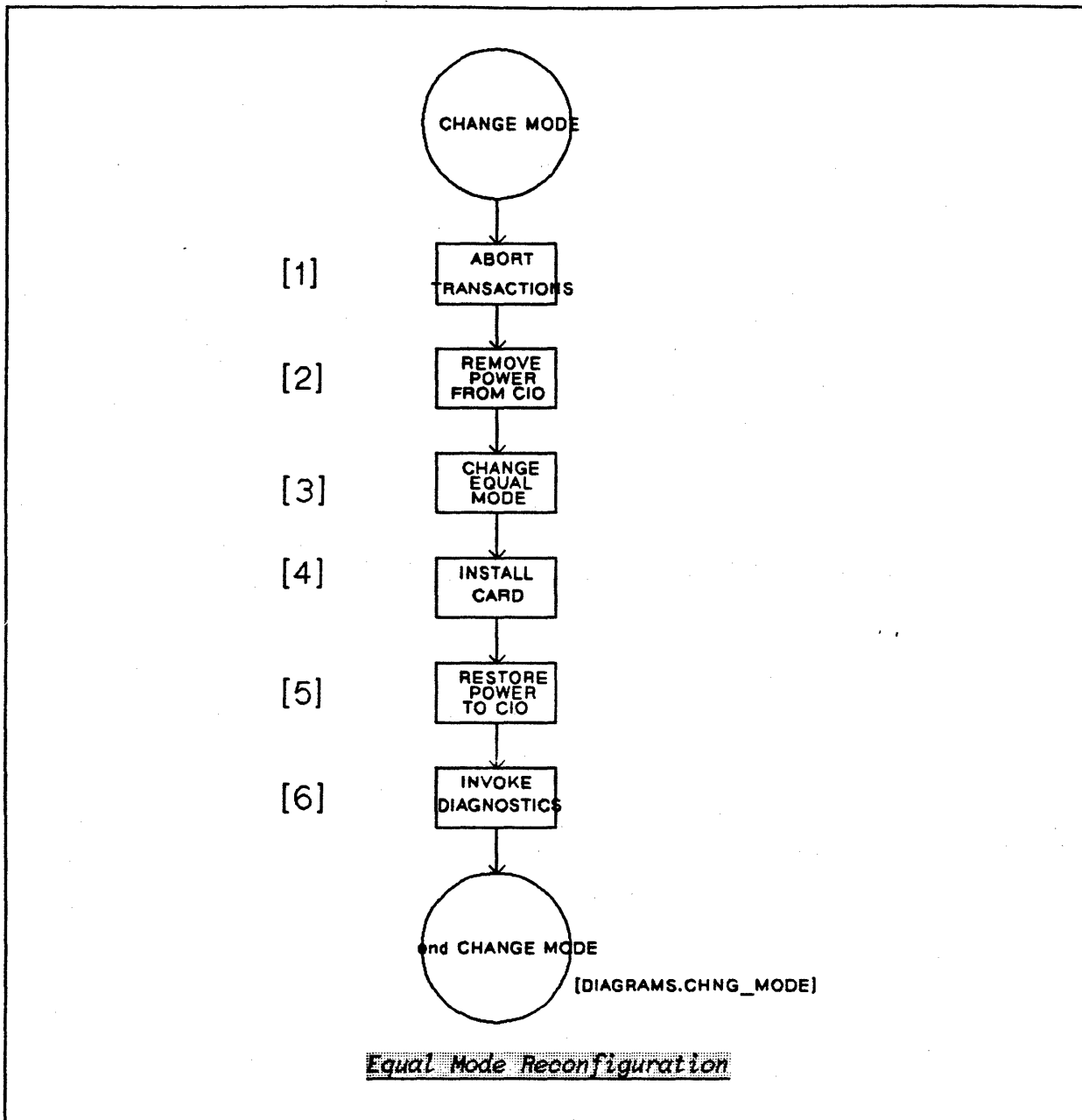
Device Loopback: Type #3. In some cases, it may be impossible to place the HP27111A in the appropriate loopback configuration without removing power from the system. When the loopback fiber has been installed and power has been returned to the CIO channel, the Device Loopback Type #3 procedure should be followed.

Also, if it has been determined at an earlier test that the Equal Mode jumper needs to be adjusted, then this is an appropriate time to do so.

Flux Check - *DESTRUCTIVE.* is performed when the fiber connectors on the HP27111A are not accessible unless the adapter is removed from the card cage.

The check follows the same procedure as the *Disruptive Flux Check!*

Equal Mode Reconfiguration. is performed when certain tests indicate that the Equal Mode jumper is in the wrong position.



Equal Mode Reconfiguration

[1] Abort transactions. All current transactions should be aborted

[2] Remove Power from CIO.. Power should be removed from the CIO card cage.

[3] Change Equal Mode. The Equal Mode Jumper should be set in its opposite position at this point.

[4] Install card. The card should be re-installed into the CIO card cage.

[5] Power system. Power to the CIO channel should be restored at this time.

USER INTEGRATION

[6] Invoke Diagnostics.

UPGRADE

The only planned field upgrade feature would be firmware revisions and secondary power support.

Firmware Revisions

Initial releases of the HP27111A will have socketed PROMs that may be field replaceable.

The firmware revision (and hardware revision) are provided in the adapter's CIO Identification Message.

Hardware Revisions

The HP27111A provides a hardware revision code which it uses to determine firmware compatibility and which may be used by the host to determine driver compatibility.

Initial releases of the HP27111A will have secondary power circuitry disabled. It has not been determined at this time whether this will be a field upgradeable option.

Link type

The HP27111A is compatible with any A-link Protocol device that supports link type O. The link type is determined by examining bits [0:1] of the first byte of a layer 2 control frame header.

THEORY OF OPERATION

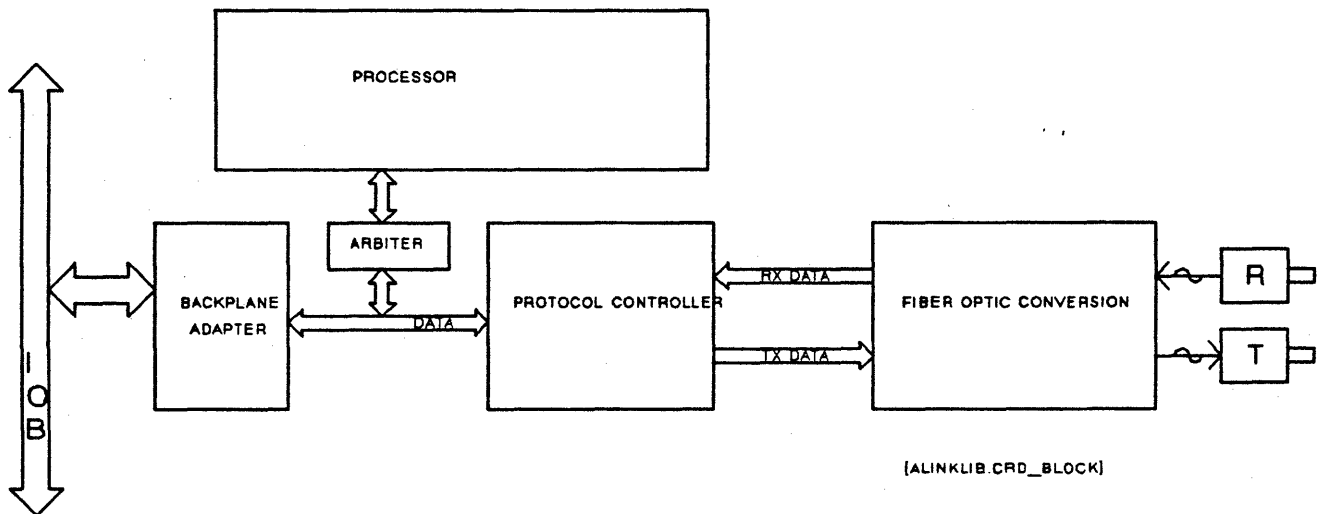
SECTION

7

The Theory of Operation section describes the hardware used to implement the functional blocks of the HP27111A. Each of the major functional blocks is described in terms of the hardware sub-blocks that comprise them.

The five major functional blocks are:

- Backplane Adapter
- Protocol Controller
- Fiber Optic Conversion
- Processor
- Arbiter



NOTE

This Theory of Operation describes the operation of the LP4/PP versions of the HP27111A as of Thu, Jun 19, 1986, 4:33 PM. This document is also a good approximation of the LP3 behaviour.

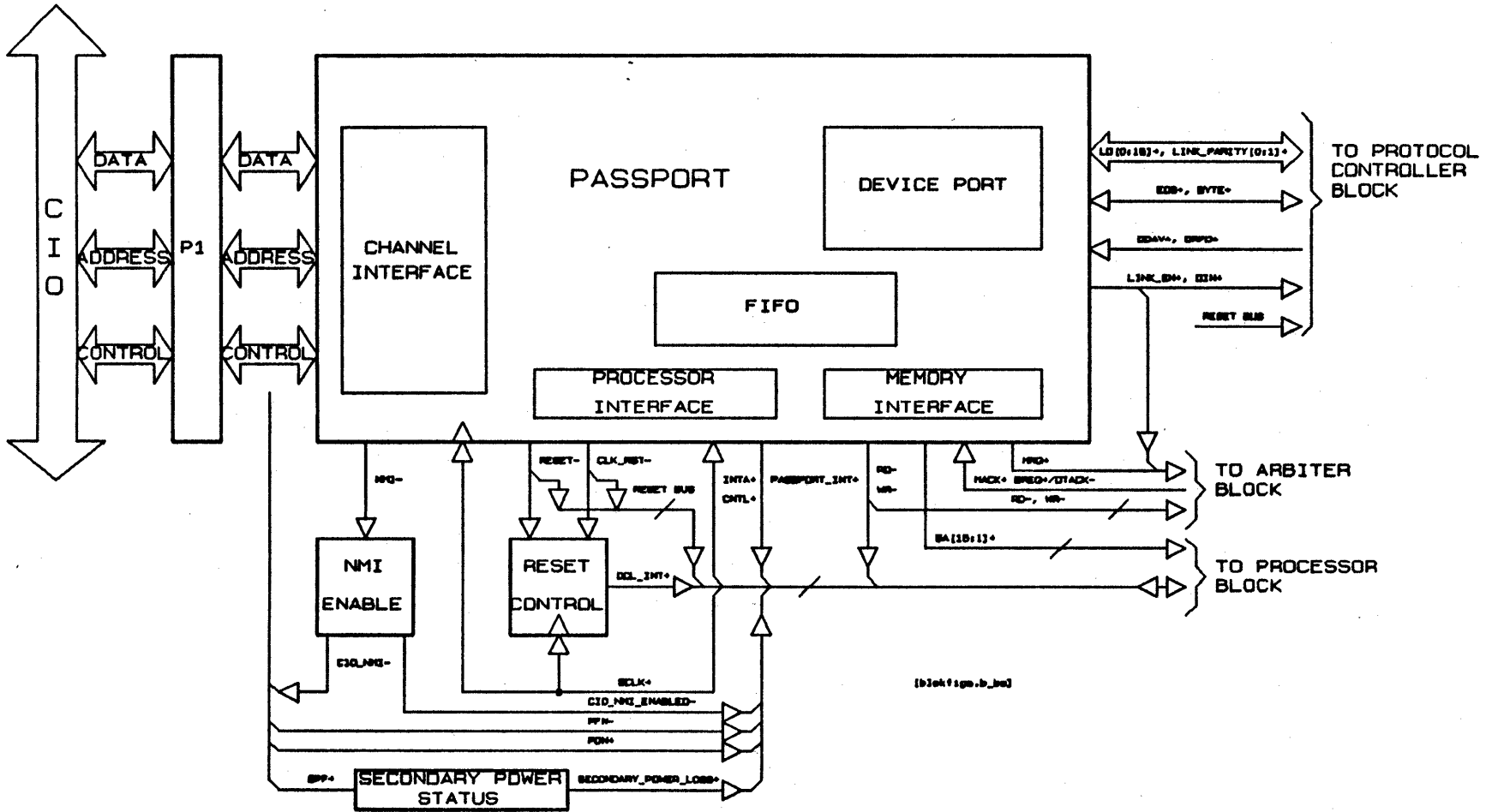
BACKPLANE ADAPTER/Passport

File: [ALIMSBA0.TAK.ALINK]

The Backplane Adapter serves as a communications path controller between the Processor, Protocol Controller and CIO Channel. It consists of 8 major sub-blocks:

- Channel Interface
- Device Port
- FIFO
- Memory Interface
- Diagnostic Interface Port
- Processor Interface
- Reset Control Circuitry
- Channel NMI Enable

The first six sub-blocks are contained within one IC, which is known as PASSPORT {U14}. The remaining sub-blocks are implemented discretely.



Channel Interface

The Channel Interface is the connection between the HP27111A and the Channel I/O (CIO) bus. The CIO bus consists of data, address, and control signals which are used to access various registers located within the PASSPORT IC.

The CIO bus is accessed via an 80 pin connector {P1} which has the following pinout assignments:

| PIN | MNEUMONIC | PIN | MNEUMONIC |
|-----|-----------|-----|-----------|
| A1 | FGND | B1 | FGND |
| A2 | DB14- * | B2 | DB15- |
| A3 | DB12- | B3 | DB13- |
| A4 | GND | B4 | GND |
| A5 | DB10- | B5 | DB11- |
| A6 | DB8- | B6 | DB9- |
| A7 | GND | B7 | GND |
| A8 | DB6- | B8 | DB7- |
| A9 | DB4- | B9 | DB5- |
| A10 | GND | B10 | GND |
| A11 | DB2- | B11 | DB3- |
| A12 | DB0- | B12 | DB1- |
| A13 | GND | B13 | GND |
| A14 | AD2- | B14 | AD3- |
| A15 | AD0- | B15 | AD1- |
| A16 | GND | B16 | GND |
| A17 | DOU- | B17 | UAD- |
| A18 | BP0- | B18 | BP1- |
| A19 | CE- | B19 | CBYT- |
| A20 | SYNC- | B20 | POLL- |

| PIN | MNEUMONIC | PIN | MNEUMONIC |
|-----|-----------|------|-----------|
| A21 | GND | B21 | GND |
| A22 | CCLK+ | B22 | IOSB- |
| A23 | GND | B23 | GND |
| A24 | BR- | B24 | ARQ- |
| A25 | DBYT- | B25 | DEND- |
| A26 | MYAD- | B26 | RST- |
| A27 | GND | B27 | GND |
| A28 | SYS DEP | B28 | SYS DEP |
| A29 | SYS DEP | B29 | SYS DEP |
| A30 | SYS DEP | B30 | AP- |
| A31 | CDPO- | B31, | CDP1- |
| A32 | PFW- | B32 | NMI- |
| A33 | PON+ | B33 | SPF+ |
| A34 | GND | B34 | GND |
| A35 | AC- | B35 | AC- |
| A36 | AC+ | B36 | AC+ |
| A37 | -12 | B37 | -12 |
| A38 | +12 | B38 | +12 |
| A39 | +5S | B39 | +5S |
| A40 | +5P | B40 | +5P |

[aliciopa]

NOTE

* : The CIO bit naming convention places bit 0 as the least significant bit. Internal to the HP27111A, bit 0 is the most significant bit. Therefore a translation must be made between CIO and internal representations. Thus, CIO signal DB15- is equivalent to internal signal DB[0]-, DB14- is equivalent to internal signal DB[1]-, and so on.

The non-power supply signals have the following significance with the HP27111A:
Channel Interface Signals

| MNEMONIC | NAME | TYPE | FUNCTION |
|-----------|------------------------|-------|---|
| AD[0:3]- | Address | I | Addresses physical adapter or adapter subchannel |
| ARQ- | Attention Request | OC | Asserted during asynchronous interrupts |
| BP[0:1]- | Bus Primitive | I | Used to define operation type |
| BR- | Burst Request | OC | Indicates ready for another data transfer |
| CBYT- | Channel Byte | IO | Data Width Indicator (8 vs. 16) for Channel Write Data |
| CEND- | Channel End | IO | Indicates last data transfer for Channel Read or Write Data |
| DB[0:15]- | Data Bus | IO * | 16-bit data bus |
| DBYT- | Device Byte | O | Data Width Indicator (8 vs. 16) for Channel Read Data operation |
| DEND- | Device End | O | Last data for current Channel Read Data operation |
| DOUT- | Data Out | I | Defines direction of channel data flow |
| IOSB- | I/O Strobe | I | Data strobe for bus operations |
| MYAD- | My Address | OC | Indicates AD[0:3]- matches either the HP27111A physical or subchannel address |
| NMI- | Non Maskable Interrupt | OC ** | direct interrupt to system processing unit |
| PFW- | Power Fail Warning | I | asserted 5 msecs before power fail |
| POLL- | Poll | I | Will cause HP27111A to respond via the Data Bus if the physical adapter needs attention or the subchannel needs service |

Channel Interface Signals

| MNEMONIC | NAME | TYPE | FUNCTION |
|-----------|------------------------|------|--|
| PON+ | Primary Power On | I | Power up and stable |
| RST- | Channel Reset | I | Resets HP27111A and on deassertion invokes selftest |
| SPF+ | Secondary Power Failed | I | Immediately after a reset condition will indicate whether the secondary power supply was valid |
| SYNC- | Synchronizer | I | Indicates beginning of channel read/write bus operation |
| UAD- | Unary Address | I | Used for physical adapter address assignment |
| CDP[0:1]- | Channel Data Parity | IO | Data Bus Parity (ODD) |
| AP- | Address Parity | I | Parity coverage for address and bus primitives |

[aliciosg] LEGEND:

- I = Input Signal
- O = Output Signal
- OC = Open Drain Output Signal
- * : DB[0:15]- also has an open collector mode during Poll cycles
- ** : NMI- is optionally enabled on the HP27111A.

CONTROL AND ADDRESS.

The channel interface Control and Address busses consist of a number of signals.

For more detail on the behaviour and protocol of the CIO bus refer to the Channel I/O Bus Specification Manual.

Registers supported by the HP27111A are shown in the following table.

| SYNC- | POLL- | DOUT- | BP[1:0]- | AD[3:0]- | OPERATION |
|-------|-------|-------|----------|------------|--------------------------------|
| 0 | 1 | X | 00 | X | Poll Service Request Group 0 |
| 0 | 1 | X | 01 | X | Poll Service Request Group 1 |
| 0 | 1 | X | 10 | X | Poll Attention Request Group 0 |
| 0 | 1 | X | 11 | X | Poll Attention Request Group 1 |
| 1 | 0 | 0 | 00 | subchannel | Read Data |
| 1 | 0 | 0 | 01 | physical | Read Sense |
| 1 | 0 | 0 | 10 | | no op |
| 1 | 0 | 0 | 11 | physical | Read Status |
| 1 | 0 | 1 | 00 | subchannel | Write Data |
| 1 | 0 | 1 | 01 | physical | Write Control |
| 1 | 0 | 1 | 10 | subchannel | Write Order |
| 1 | 0 | 1 | 11 | physical | Write Command |

[aliciorg]

Which have the following effect on the HP27111A

Channel Operations Table

| OPERATION | DESCRIPTION |
|---------------|---|
| Read Data | used to transfer Data from A-Link card to CIO channel through the current active Subchannel |
| Write Data | used to transfer Data from CIO channel to A-Link card through the current active Subchannel |
| Write Order | used to control Subchannels and to connect, control, and destroy Logchannels |
| Write Command | used mainly to connect and destroy Subchannels. |
| Read Status | used mainly to return physical status of A-Link card and its local Subchannel |

Channel Operations Table (continued)

| OPERATION | DESCRIPTION |
|------------------------|--|
| Write Control | used to control the state of the Backplane Adapter |
| Read Sense | used to determine status of the Backplane Adapter and its physical interface to the channel |
| Poll Service Request | A-link responds when it is capable of processing an order or data on its current active subchannel |
| Poll Attention Request | A-link responds when it has been programmed to request a new command from the channel and is able to do so |

The address field **AD[0:3]-** shows two types of references: Physical and Subchannel. This means that if **AD[0:3]-** matches the value of PASSPORT's internal Subchannel or Physical address then the operation is valid. PASSPORT indicates this validity to CIO by asserting **MYAD-**.

The Physical address is first "learned" when **UAD-** is asserted and a Write Data operation is decoded by PASSPORT. The state of **AD[0:3]-** is copied into an internal Physical Address register. This value will be saved until a Power On or Channel Reset occurs.

The Subchannel address is maintained internally by PASSPORT according to firmware directives issued by the Processor.

Note that all "Data" transfers require that a Subchannel be present.

Although not shown in the above table, **IOSB-** acts as a clocking pulse that PASSPORT uses to determine that the bus operation has occurred.

CONTROL AND ADDRESS PARITY.

PASSPORT has a programmable "parity checking mode" in which it will check and react to parity errors on the Control, Address or Data busses. Parity checking mode is enabled by a Channel Write Control[**PEN**] operation. The Control and Address busses are then parity protected by the **AP-** bit. If the **AP-** bit together with the Control and Address inputs do not form ODD parity, PASSPORT enters a *Channel Control Parity Error* condition and not assert **MYAD-**. If the **AD[0:3]-** bus matches either the physical or subchannel address for the card and the error condition is detected, PASSPORT will also log a *Channel Control Parity Error* in the Processor Control Block and interrupt the Processor.

DATA BUS.

The channel interface data bus consists of a 16 bit data bus DB[0:15]-, two parity bits CDP[0:1]-, and four auxiliary lines {CEND-, CBYT-, DEND-, DBYT- }.

The data bus is used for normal I/O communication with the backplane as well as poll cycle operations according to CIO protocol. Data from Read Sense and Read Status operations will be driven onto the lower byte of the data bus, DB[7:0]. Data for Write operations to the Command, Control and Order registers is also taken from DB[7:0]-.

The auxiliary fields are used in conjunction with read and write operations with the Data register. The two end delimiters, DEND-, and CEND-, indicate the termination of a data transfer in response to an *Order* that required the data register to be read or written. DEND- is only asserted by PASSPORT during read operations. CEND- may be asserted by the channel during either write operations or read operations, with the latter case indicating the channel is terminating the data transfer rather than the device.

Transfers with the Data register may be either *packed* or *unpacked*. During packed transfers the byte flag signals, DBYT- and CBYT-, are used to indicate whether a byte or word is being driven onto the data bus by the HP27111A. DBYT- will be asserted concurrently with DEND- and only the upper byte DB[0:7]- will be valid. CBYT- must only be received concurrent with CEND- and only the upper byte DB[0:7]- will be accepted. In *unpacked* transfers, all Data register operations use the lower byte of the data bus DB[8:15]-. CBYT- must be asserted for *all* unpacked data transfers.

DATA PARITY.

PASSPORT will generate ODD parity on all data bus read operations (except for poll cycles) and drive CDP[0:1]- to appropriate levels. If "parity check mode" has been enabled, PASSPORT will also check for ODD parity on data during Channel Write operations to all registers *EXCEPT FOR THE WRITE CONTROL REGISTER!* Detection of a parity error will cause PASSPORT to interrupt the Processor and indicate a Channel Data Parity Error in its control block status.

FIFO

The FIFO is, as its name infers, a First In First Out memory buffer that effectively serves as a data path between the Channel and the card. The FIFO stores the state of the data bus and parity bits as well as the auxiliary byte and end delimiters. The FIFO is 24 locations deep in either packed or unpacked mode, and each location holds either a byte or a word.

While the FIFO is bidirectional, it can only be active in one direction. That is, the FIFO cannot interleave reads and writes until a logical breakpoint in the data transfer occurs.

Data within the FIFO can be transferred between the channel and either the Protocol Controller or the Processor.

The FIFO is entirely contained within the PASSPORT IC.

Device Port

The Device Port is PASSPORT's interface to the Protocol Controller. It consists of a 16 bit data bus, 2 parity bits, 2 auxiliary bits and 4 control lines.

DATA BUS.

The Device Port Data Bus is 16 bits wide and can be viewed as an upper and lower byte. Each of these bytes is protected by a parity bit. PASSPORT is programmed at initialization to generate and check for ODD parity. The Data bus is also used in the Memory interface.

The Auxiliary Bits, EOS+ U14-P11 and BYTE+ U14-L13 are used to indicate data termination and odd data count respectively. These bits are asserted concurrently with data. The EOS+ and BYTE+ bits correspond to the Channel's (DEND-, CEND-) and (DBYT-, CBYT-) data signals.

BYTE+ will only be asserted concurrently with EOS+, and the odd byte will be located in the upper byte of the data transfer, LD[0:7]+.

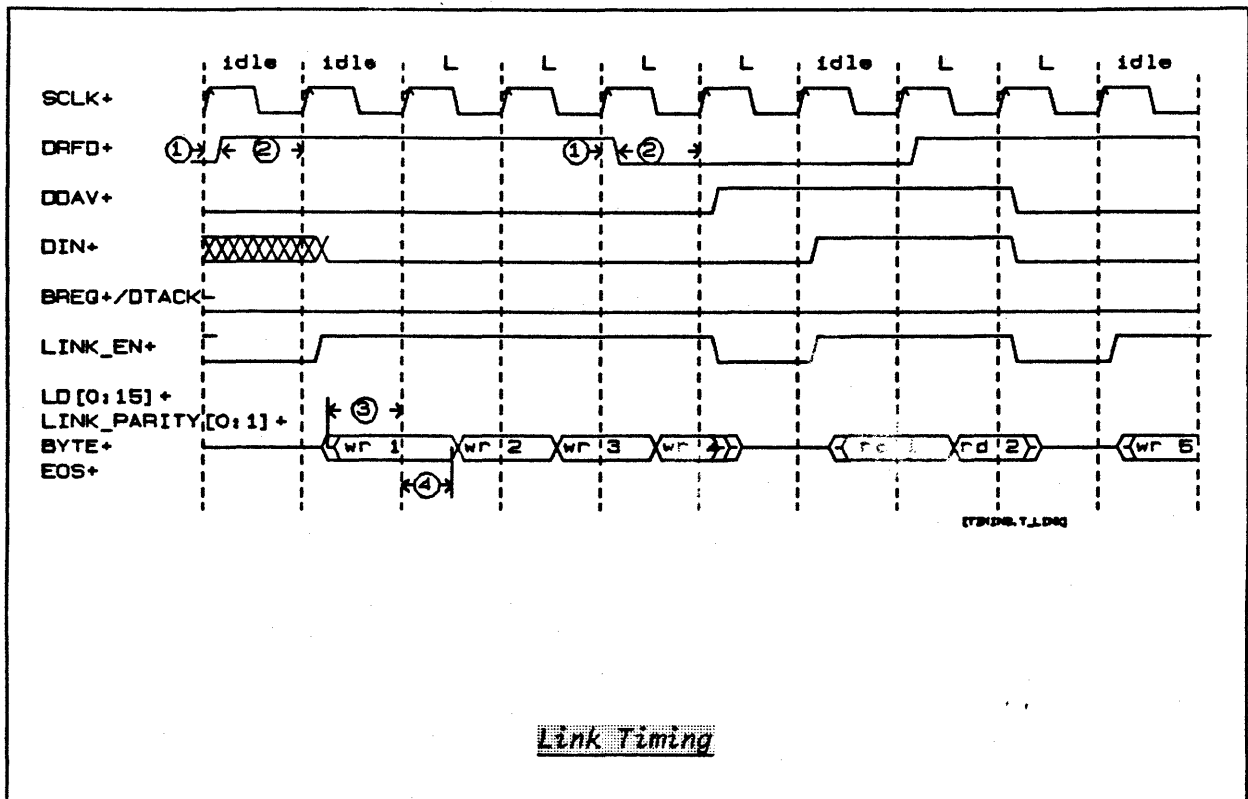
CONTROL BUS.

PASSPORT uses four control lines to transfer data to the Protocol Controller: DRFD+, DDAV+, LINK_EN+, and DIN+ {U14-D14, D13, F12, F13}. The first two signals, DRFD+ and DDAV+, are short hand for Device Ready for Data and Device Data Available, respectively. PASSPORT samples these signals to determine whether the Protocol Controller is ready to perform a read or write transaction. LINK_EN+ is the Link Enable signal which is asserted to indicate that PASSPORT wishes to perform a data transfer. The direction of the transfer is specified by the DIN+ signal. When asserted, DIN+ indicates a Read operation; when deasserted, a Write operation.

Assuming that PASSPORT has been programmed (by the Processor) to perform a Device Port transfer, when the appropriate "Ready" signal appears, PASSPORT will drive its LINK_EN+ and DIN+ lines to indicate to the Protocol Controller that an active transfer is to occur. The transfer will occur, if, on the succeeding clock edge, the Protocol Controller's ready signal is still present.

CLOCKING.

All Device Port activity is relative to SCLK+ {U14-K3}. The clock is provided by the Processor Block.



Link Timing Paramters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|-------------------|---|-----|-----|
| 1 | t_{CNTL_HOLD} | hold of any of the following <i>control</i> signals (DEV_RFD+, DEV_DAV+, DIN+, LINK_EN+) to rising edge of SCLK+. | 0 | 31 |
| 2 | t_{CNTL_SETUP} | setup of any of the following control signals (DEV_RFD+, DEV_DAV+, DIN+, LINK_EN+) to rising edge of SCLK+. | 87 | ... |
| 3 | T_{DATA_SETU} | setup of DATA to rising edge of SCLK+ | 7 | ... |
| 4 | t_{DATA_HOLD} | hold of DATA from rising edge of SCLK+ | 18 | ... |

Processor Interface

The Processor Interface is used by PASSPORT to determine when it should examine memory for new information and also to inform the Processor that status information is available. The interface is also used to supply power status indication that is taken from the CIO backplane.

Theory of Operation

PASSPORT CONTROL SIGNALS.

The interface from PASSPORT's perspective consists of three signals: CNTL+, INTA+, and PASSPORT_INT+ {U14-D12, C14, U16-7}.

When PASSPORT detects that CNTL+ is asserted it will pick up a block of memory for processing. PASSPORT will process this block depending upon whether this is the first assertion of CNTL+ since a reset condition. In either case, as long as CNTL+ is asserted, PASSPORT will attempt to process the block.

If PASSPORT completes the task or CNTL+ is deasserted, it will post status information in memory and assert the interrupt signal INTR-. INTR- is an *open-drain* output for PASSPORT {U14-E14} which is pulled to a deasserted state by U35-1. INTR- in turn, is buffered by an inverter {U16-13} and forms the positive true signal PASSPORT_INT+ {U16-7} which is presented to the Processor Block.

PASSPORT_INT+ remains asserted until PASSPORT determines that the interrupt has been acknowledged by finding INTA+ asserted. PASSPORT expects that INTA+ will then deassert completing the Interrupt Handshake.

PRIMARY POWER STATUS.

Two Primary Power Status signals are driven by the Backplane Adapter to the Processor Block: PON+ and PFW+.

PON+ is the same power indicator signal from the CIO backplane and is merely passed on to the Processor block domain.

PFW+ {U16-2} is a buffered {U16-18} and inverted version of PFW- which is also taken from the CIO backplane.

SECONDARY POWER STATUS.

Secondary Power Status is also presented to the Processor block in the form of the SECONDARY_POWER_LOSS+ signal. The assertion of this signal indicates that secondary power was lost during the last recovery from a Primary Power Failure.

The behaviour of this signal is controlled by the Secondary Power Jumper which is loaded between {U13-3} and {U13-6}. When the jumper is present, SECONDARY_POWER_LOSS+ is driven by a buffered version of SPF+ from the CIO backplane. Two inverting ALS240 buffers from {U16} form a non-inverting buffer pair, with SPF+ driving input {U16-6} and the output {U16-5} driving SECONDARY_POWER_LOSS+.

When the jumper is absent, the signal is always asserted by virtue of the 10Kohm resistor {U35-2} that pulls it to approximately +5 Volts.

| |
|-------------|
| NOTE |
|-------------|

The Secondary Power Status Jumper is set to the *OPEN* position on the HP27111A. Loss of Secondary Power is *always* assumed.

CLOCKING.

All Processor Interface activity is relative to SCLK+. The clock is provided by the Processor Block.

Memory Interface

PASSPORT communicates to the Processor using a shared memory scheme where task lists are received and processed, and status blocks are posted as the tasks are completed.

The hardware interface consists of a 16 bit address, of which 13 are used in this pca, the data portion of the Device Data Bus, and a Memory Control bus.

The programmatic interface consists of an Initialization Block, Command Block, Autolist Items and Buffers.

A detailed description of these is available in the PASSPORT ERS and PASSPORT Programming Manual.

ADDRESS BUS.

PASSPORT's Memory Address bus is a 16 bit bus, SA[15:0] of which only 13 bits, SA[13:1] are used on this pca. These 13 bits address any of 8K words of data in the RAM shared by the Backplane Adapter and Processor blocks. This represents RAM addresses 0 - 3FFEh.

If the card has been configured for expansion RAM, address bits SA[15:14] are also used, and the address range increases to 0 - 0FFFEh or 32K words.

The remaining address line, SA[0] is unused during PASSPORT Memory Accesses and is indeterminate.

DATA BUS.

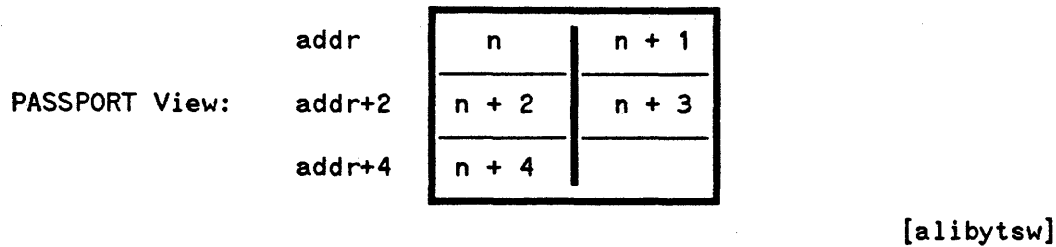
PASSPORT accesses memory data in WORD MODE only. Data structures within memory are accessed normally and treated as a 16 bit word with LD[0:7]+ representing the most significant byte.

Data Structures are stored in memory at increasing addresses. For example, an 8 word data structure starting at address 100H would have its 8th word stored at address 10EH.

BYTE ARRAY Data Structure transfers between PASSPORT and the CIO Channel interpret the data as shown in the following diagram.

Theory of Operation

Also, as seen from the diagram, when PASSPORT transfers an *ODD* number of bytes, the Odd Byte is assigned to the most significant byte.



NOTE

The Processor block must be careful when using data structures with **BYTE ARRAY** formats with the Backplane Adapter. This is because the Processor block accesses the individual bytes within a **BYTE ARRAY** in a different order when using sequential byte manipulation instructions.

Consult the description of the Processor block for further details.

MEMORY CONTROL.

Because the Memory area of the HP27111A is shared, PASSPORT must first request access from the Processor via the Memory Control and Arbiter state machines {U15, U17, U23}.

PASSPORT has four output and two input lines used in Memory Control.

MRQ+ {U14-F14} is asserted by Passport to indicate that it wishes to access memory

DIN+ indicates the direction of the transfer, asserting for a Memory Read

RD- {U14-G13} is the Memory Read strobe, asserted when PASSPORT is ready to have data driven onto its data bus

WR- {U14-H14} is the Memory Write strobe, asserted when PASSPORT has valid data driven onto the data bus

MACK+ {U14-E13} is the Memory Acknowledge line, indicating to PASSPORT its memory request has been granted

BREQ+/DTACK- {U14-E12} is the Data Transfer Acknowledge line, during Memory transfers, indicating that the current Memory operation may complete.

When PASSPORT wants to access memory, it will first assert its **MRQ+** line along with the **DIN+** line, indicating the type of memory access, read or write, that it intends to perform.. If memory is capable of being accessed the **MACK+** signal will be asserted. At this point PASSPORT has control over all RAM

within the Processor block. The presence of MACK+ causes PASSPORT to actively drive the SA[]+ address bus. PASSPORT then drives RD- or WR- to initiate the memory access and then waits for the data transfer acknowledge signal DTACK- to indicate that data is now setup. PASSPORT will then deassert MRQ+ and RD- or WR- to complete the transfer.

RD-s, WR-, as well as the SA[]+ bus are normally in the high impedance state. They remain at high-impedance during a memory request until MACK+ is asserted. These control signals then return to their high-impedance state once the memory transfer is complete.

The LD[]+ bus will only be placed in high-impedance once the memory request cycle has begun, and will be driven when MACK+ is asserted only if a Write Memory cycle is active.

Further details on memory transfers are found in the Arbiter description.

CLOCKING.

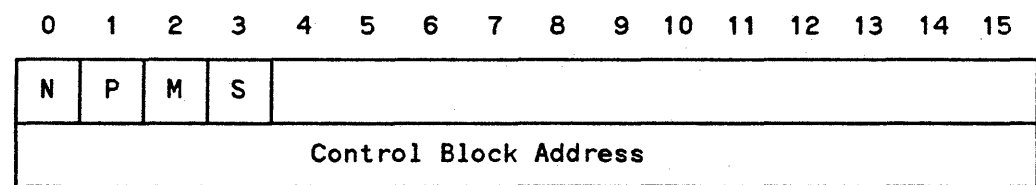
All Memory Interface activity is relative to SCLK+. The clock is provided by the Processor Block.

INITIALIZATION BLOCK.

PASSPORT must be initialized each time it is reset by the channel. It performs this by fetching an area of memory known as the initialization block upon detecting the first assertion of CNTL+ following the reset condition. The initialization block provides information to PASSPORT about the nature of the devices it will interface to.

The location of this block of memory is at 0FFF0H. Since the upper two address lines are unused on the HP27111A this corresponds to RAM word address 03FF0H, assuming that 16Kbytes of memory is present.

PASSPORT fetches the data words located at locations 03FF0H and 03FF2H and parses them according to the following format:



where,

| Init-bit | Function |
|----------|-------------------------------|
| N | 8 bit Peripheral Interface |
| P | Peripheral Parity Check On |
| M | Memory Parity Check On |
| S | Swap Data Buffer Memory Bytes |

[alipinit]

Theory of Operation

The HP27111A sets these data words to 04000H and 0100H, respectively, indicating that the Device Port has Parity, and that the location of the Control Block is at address 100H.

After PASSPORT has fetched these two words, it will immediately post status information in the Control Block area, and assert **PASSPORT_INT+**.

CONTROL BLOCK.

The Control Block is used by the Processor to issue commands to PASSPORT and for PASSPORT to post various types of status. The Control Block is 7 words long and has the following format:

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|----------------------|---|----------|---|---|-------------------------|---|---|----------------------|---|----|----|----|----|----|----|
| A | | Pcommand | | | CIO Status (Pcommand=2) | | | | | | | | | | |
| Auto Item Pointer #1 | | | | | | | | | | | | | | H | |
| Auto Item Pointer #2 | | | | | | | | | | | | | | H | |
| Auto Item Pointer #3 | | | | | | | | | | | | | | H | |
| Auto Item Pointer #4 | | | | | | | | | | | | | | H | |
| Status Byte #1 | | | | | | | | CIO Command received | | | | | | | |
| Status Byte #2 | | | | | | | | CIO Order received | | | | | | | |

where,

| Pcommand | Function |
|----------|---------------------------------------|
| 0 | set PST |
| 1 | set NMI |
| 2 | set CIO Device Status from Data[8:15] |
| 3 | pause |
| 4 | unpause |
| 5 | destroy subchannel |
| 6 | listen to backplane |
| 7 | flush transfer |

[alipcbk]

| | | | | | | | | |
|---------------|----|----|----|----|----|----|-----|------------|
| Status Byte 1 | RC | CF | PM | NA | OV | PC | PD | PP |
| Status Byte 2 | RO | AR | PA | | | | WHY | [alipstat] |

| | |
|-----|---|
| AR | ARQ |
| CF | Command Failed |
| NA | NMI Acknowledge from CIO |
| OV | Overflowed Buffer List |
| PA | Paused |
| PC | Parity Error on CIO Control |
| PD | Parity Error on CIO Data |
| PM | Parity Error on Memory |
| PP | Parity Error on Pronto Data |
| RC | Received Command (present in CIO Command Received Byte) |
| RO | Received Order (present in CIO Order Received Byte) |
| WHY | Why Reset 0 : PON+ deasserted 1 : RST- asserted 2,3 : Addressed Device Clear |

| CIO Bit coding summary | | | | | | | | | | | | | | | | |
|----------------------------------|---|------|----------|---|------|-------------------------------|---|------|--------------|---|--------------|------|-----|---|-----------|------|
| data bits for level 1 primitives | | | | | | logchannel card message bytes | | | | | | | | | | |
| CIO orders | | | commands | | | dev status | | | WTC messages | | RST messages | | | | | |
| IDY | 0 | 1000 | DSC | 0 | subc | RFC | 0 | 0000 | 0 | | IDL | 0 | | | | |
| DLD | 1 | b00c | AEK | 1 | evno | AES | 1 | evno | RLC | 1 | lchi | lclo | SWI | 1 | lchi | lclo |
| PAU | 2 | 0000 | CSC | 2 | subc | DPE | 2 | 0000 | EOD | 2 | lchi | lclo | EOD | 2 | lchi | lclo |
| DIS | 3 | 0000 | | 3 | | SCD | 3 | subc | DLC | 3 | lchi | lclo | CD | 3 | lchi | lclo |
| RS | 4 | b00d | RSC | 4 | subc | | 4 | | AEK | 4 | ak0..ak14 | | AES | 4 | ak0..ak14 | |
| WC | 5 | b000 | ETA | 5 | subc | SCR | 5 | clas | | | | | ERT | 5 | lchi | lclo |
| RD | 6 | b000 | | | | DOD | 6 | diag | | | | | | | | |
| WD | 7 | b000 | | | | PER | 7 | diag | | | | | | | | |
| RTS | 8 | 1000 | | | | ERT | 8 | subc | | | | | | | | |
| WTC | 9 | 1000 | | | | | | | | | | | | | | |
| RDS | A | b00c | | | | | | | | | | | | | | |
| CLC | B | 1000 | | | | | | | | | | | | | | |

[alciodoc]

Every time PASSPORT detects the assertion of CNTL+ it will sample the first 5 words of the control block. Depending on its state PASSPORT will post status on its own accord and assert PASSPORT_INT+ to indicate this, or it can be requested to post status (still a valid mode for A-Link?) when it detects the deassertion of CNTL+, where again the assertion of PASSPORT_INT+ indicates new status posted. Whenever PASSPORT posts status it will write to the last 6 words of the Control BLock (102H-10CH).

Further information on Control Block function can be found in the PASSPORT Programmer's guide and the HP27111A Firmware IMS.

Theory of Operation

AUTO LIST ITEMS.

PASSPORT's data transfer tasks are obtained from Auto List Items which are indicated in the Control Block. The Auto List Items indicate to PASSPORT the source and destination for data transfer and the triggering events which may initiate a transfer.

| | | | | | | | | | | | | | | | |
|----------------------------|---|--------|---|---|------------------------|---|---|----------------------------|---|----|----|----|----|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| RWO | | BWO | | F | CIO order/command | | | | | | | | | | |
| PUP | T | opcode | | | CIO order/command mask | | | | | | | | | | |
| Buffer Header List Pointer | | | | | | | | | | | | | | | |
| | | | | | | | | CIO order/command received | | | | | | | |
| Next Auto Item Pointer | | | | | | | | | | | | | | | |

[alipauto]

| RWO | | Opcode | |
|-----|-------------------------------|--------|------------------------------|
| 0 | don't wait | 0 | no-op |
| 4 | wait for CIO command | 2 | Pronto -> Memory |
| 5 | wait for CIO order | 3 | Memory -> Pronto |
| 6 | wait for DAV | 4 | CIO -> Memory |
| 7 | wait for RFD | 5 | Memory -> CIO |
| | | 6 | CIO -> Pronto |
| | | 7 | Pronto -> CIO |
| BWO | | PUP | (CIO Pause Control) |
| 1.0 | use "b" from last order | 0 | leave pause state alone |
| 0.b | force a value for "b" | 1 | unpause after executing |
| F | (Force CIO Pause Control) | 2 | pause after executing opcode |
| 0 | Leave pause flip-flop alone | | |
| 1 | Force unpausing during wait | | |
| T | | | |
| 0 | Continue Listening | | |
| 1 | go Idle after completing item | | |

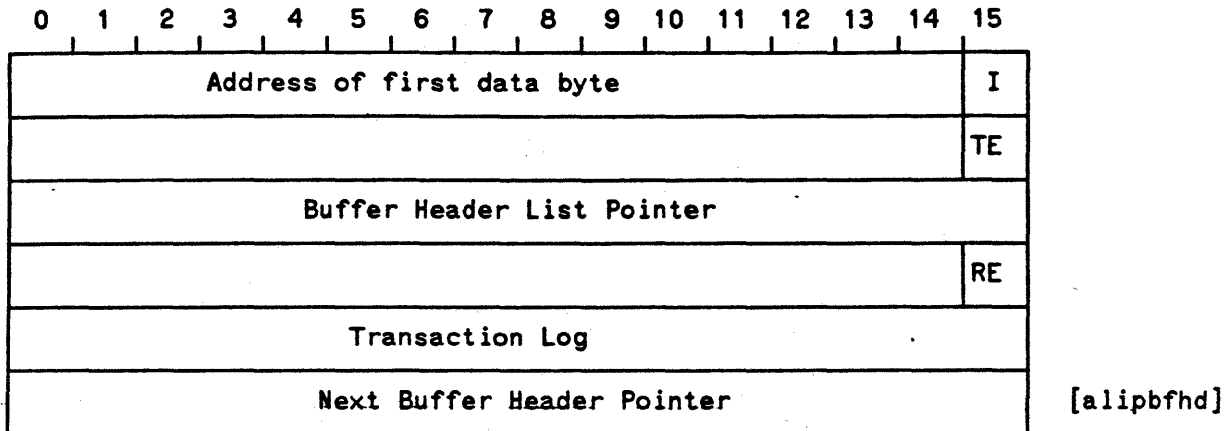
[alipaut2]

Refer to the PASSPORT Programmer's guide and the HP27111A Firmware IMS for more information on the Auto Item lists.

BUFFERS.

Buffers are used when it is necessary to transfer data between card memory and either the CIO Channel or the Protocol Controller. Each buffer consists of a Buffer Header and an actual Data Buffer.

The Buffer Header provides the address of the Data Buffer as well as size, log, and delimiter tag fields. The size field indicates the contiguous size of the Data Buffer and is an input to PASSPORT. The Log is a transaction byte count indicating how many bytes of data were read from or written to the Data Buffer. The send and receive delimiter tag fields are either tested or posted by PASSPORT and indicate whether an EOS delimiter should be appended to outbound data or whether the inbound data buffer was terminated by EOS.



Further information on Buffer functions can be found in the PASSPORT Programmer's guide and the HP27111A Firmware IMS.

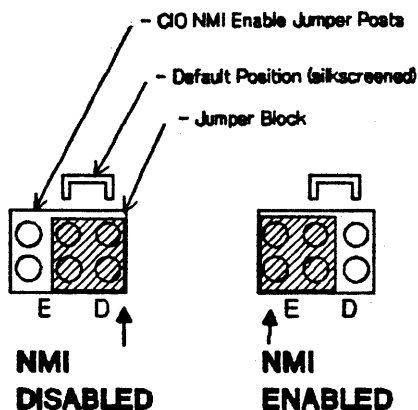
Channel NMI Enable

The ability of the HP27111A to assert the NMI- signal on the CIO backplane is controlled by the NMI Enable Jumper *J1*. When the Jumper is set in the NMI_Enabled position, the NMI- signal from PASSPORT is connected to the CIO_NMI- signal. This jumper position also causes *U35-3* to be pulled to GND thus asserting CIO_NMI_ENABLED-.

When the jumper is set in the NMI_Disabled position, indicated by DIS on the schematic, CIO_NMI- is a no-connect on the card and CIO_NMI_ENABLED- is pulled high to a deasserted state.

CIO_NMI_ENABLED- is driven to the Processor block for sampling.

Theory of Operation



afig301

NOTE

This jumper is not loaded on the HP27111A, forcing the card to always default to NMI *disabled*.

Reset Control

The HP27111A is reset through PASSPORT with three levels of severity: Power On, Channel, and Device Clear. PASSPORT, in turn generates two types of reset pulses which are then distributed to the remainder of the HP27111A, RESET- and CLK__RST- {U14-N4, P3}.

Ordinarily, a Power On reset occurs when power is initially applied to the channel and the HP27111A, in order to initialize the card. A Channel reset may occur at any time and resets all the cards in a particular channel. Device Clear is a selective reset of the HP27111A.

The cause of the latest reset is posted by PASSPORT in the control block following the initialization sequence.

POWER ON RESET.

A Power On Reset is generated when PON+ deasserts.

This form of reset causes both CLK__RST- and RESET- to assert.

All card state information is lost unless the card is configured for secondary power usage and secondary power was not lost.

A typical Power On sequence from the Backplane Adapter's perspective is as follows. When power is initially applied, PON+ is deasserted. When power is stable and PON+ is still deasserted, PASSPORT will assert both CLK__RST- and RESET- to the rest of the card. Both of these signals will remain asserted

as long as PON+ is deasserted. When PON+ asserts, CLK_RST- will deassert approximately 2 SCLK+ cycles after this event. RESET- will remain asserted for approximately 512 more SCLK+ cycles and then will deassert.

While PON+ is deasserted, and power is present and stable, all external pins on PASSPORT will be in the high-impedance state with the the exception, of course, of CLK_RST- and RESET-.

When both CLK_RST- and RESET- have deasserted, the Backplane Adapter is in the following state:

Backplane Adapter Power On Status

| SIGNAL STATE | SIGNAL |
|----------------|--|
| Asserted | (none) |
| Deasserted | Channel signals: ARQ-, NMI-, MYAD-, BR- Internal signals: DIN+, LINK_EN+, PASSPORT_INT+, MRQ+, RESET-, CLK_RST- |
| High Impedance | Channel signals: DEND-, DBYT-, P_CDPO-, P_CDP1-, P_DB[0:15]- Internal signals: LD[0:15]+, SA[15:1]+, BYTE+, END+, RD-, WR-, LINK_PARITY[0:1]+ |

The Backplane Adapter will remain in this state until the Channel or selftest firmware controlled by the Processor programs it to behave differently.

The Backplane Adapter will not respond to the Channel until the Channel has taught it a *physical address* (see control and address).

NOTE

The HP27111A revision code "0" will lose *ALL* card state information regardless of presence of secondary power. The necessary hardware is present to recover in revision "0". The necessary firmware is not!

CHANNEL RESET.

A Channel Reset is caused when RST- is asserted.

This form of reset causes both CLK_RST- and RESET- to assert.

All card state information is lost during this reset.

Theory of Operation

Channel Reset behaviour for the Backplane Adapter is essentially the same as with the Power On reset. When RST- is asserted by the Channel, PASSPORT will assert both CLK_RST- and RESET- to the rest of the card. Both of these signals will remain asserted as long as RST- is asserted. When RST- deasserts, CLK_RST- will deassert approximately 2 SCLK+ cycles after this event. RESET- will remain asserted for approximately 512 more SCLK+ cycles and then will deassert.

While RST- is asserted, and power is present and stable, all external pins on PASSPORT will be in the high-impedance state with the exception, of course, of CLK_RST- and RESET-.

When both CLK_RST- and RESET- have both deasserted, the Backplane Adapter is in the following state: All of the Backplane Adapter signals will remain in their "power on" state with the exception of the Device Clear Machine which enters the <IDLE> state.

As before, the Backplane Adapter will remain in this state until the Channel or selftest firmware controlled by the Processor programs it to behave differently.

DEVICE CLEAR.

A Device Clear is caused when a Channel Write Control[DCL] operation is issued to the address corresponding to the card's physical address.

This form of reset causes only RESET- to assert.

The Device Clear remains in effect until a Channel Write Control[DEN] is issued to the address corresponding to the card's physical address. RESET- will deassert 512 SCLK+ cycles after the Channel operation.

The Backplane Adapter's behaviour during a Device Clear is very similar to both the Power On and Channel reset cases. The two exceptions are the preserving of the card's physical address, and the state of the Device Clear Machine.

The card's physical address, stored in PASSPORT, remains valid through the duration of the Device Clear. Thus the card remains capable of channel communication after the Device Clear, and cannot be "retaught" a new physical address.

The Device Clear machine differentiates the Device Clear from the other types of resets and relays this state to the rest of the card.

DEVICE CLEAR MACHINE.

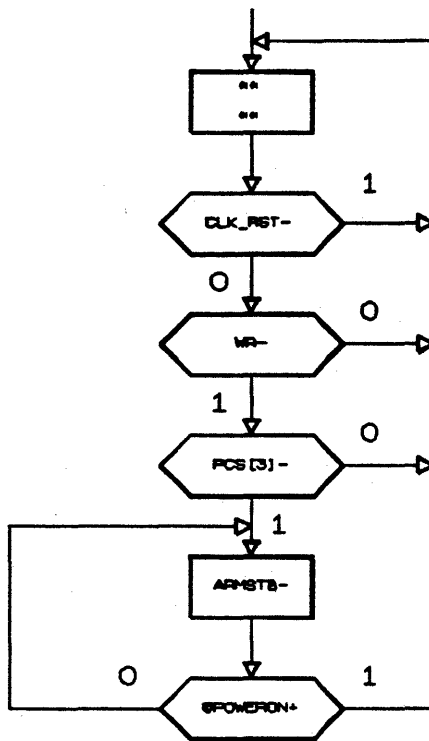
The Device Clear sequence is detected externally by the Device Clear Machine which will assert DCL_INT+ to the 80186 Processor. Basically, the machine looks for an instance when RESET- is asserted without an accompanying CLK_RST-.

The Device Clear machine is implemented using part of a 16R4 PAL {U17}. In describing the state behaviour of the Device Clear Machine, the following group of signals is treated as a vector:

Device Clear State: [DCL_INT+, CLR_WAIT-, Q1+]

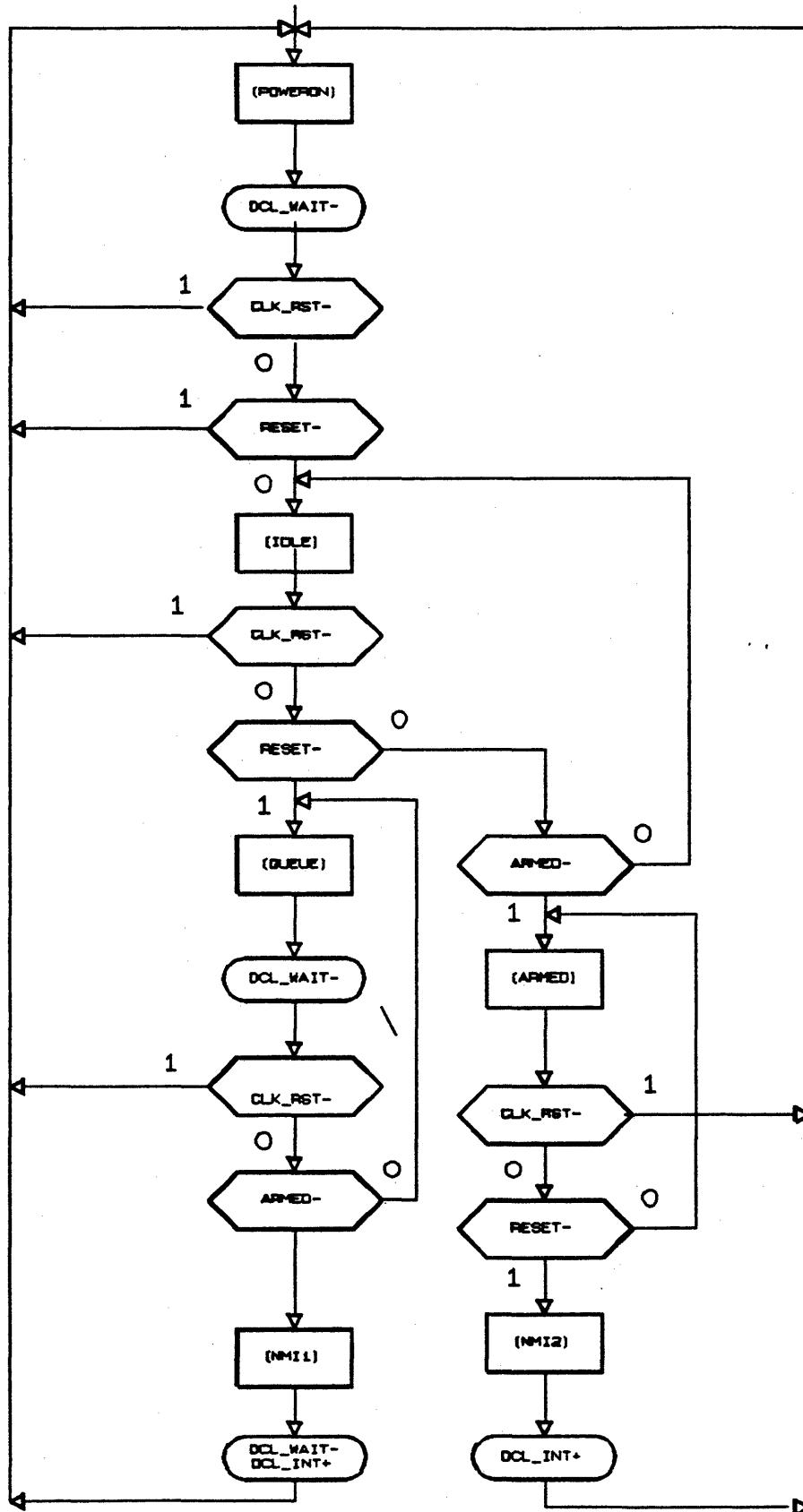
Q1+ is a state output of {U17} that has no other connections.

Flow charts to illustrate the Device Clear Machine's behaviour follows:



[DIAGRAMS, ARMSTB]

Theory of Operation



[DIAGRAMS.DCL_NMI]

ARMED- is asserted during a Processor Write to the memory area addressed by **PCS[3]-**, as shown in the flow chart. **DCL_INT+** cannot be asserted until **ARMED-** has been asserted. Note that **ARMED-** is placed in a deasserted state by the main Device Clear Machine entering the **<POWERON>** state, indicated by the internal signal **@POWERON+**. Thus any type of reset disarms the Device Clear Mode, including a Device Clear itself. This is because the Processor block needs to check its ability to respond to a DCL each time it is either soft or hard reset.

The main Device Clear State Machine behaviour is then as follows:

Device Clear State:**<POWERON>**
Outputs:**[CLR_WAIT-]**

The DCL machine samples the states of **PASSPORT's** two reset control lines **CLK_RST-** and **RESET-**. When both reset control lines are deasserted, the machine will enter the **<IDLE>** state.

Device Clear State:**<IDLE>**
Outputs:**[]**

The DCL machine now looks for one of three events: a Device Clear Reset, a non-Device Clear Reset, or the Processor indicating that the Device Clear process is armed.

A non-Device Clear Reset is recognized by the assertion of **CLK_RST-** and will cause the machine to re-enter the **<POWERON>** state. A Device Clear Reset is recognized by the assertion of **RESET-** without **CLK_RST-** and causes the machine to **<QUEUE>** the Device Clear status. Assertion of **ARMED-** will cause the machine to enter the **<ARMED>** state.

If none of these events occur, the machine will remain in the **<IDLE>** state.

Device Clear State:**<QUEUE>**
Outputs:**[CLR_WAIT-]**

The machine is now waiting to inform the Processor block that a DCL has occurred. It cannot do this, however until it detects the assertion of **ARMED-** indicating that the Processor can process the device clear.

The machine will remain in the **<QUEUE>** state until **ARMED-** is asserted or a non-Device Clear Reset occurs, entering the **<NMI1>** and **<POWERON>** states, respectively.

Device Clear State:**<NMI1>**
Outputs:**[DCL_INT+, CLR_WAIT-]**

DCL_INT+ is asserted. The machine will always return to the **<POWERON>** state at this point, and wait for the Device Clear to complete.

Device Clear State:**<ARMED>**
Outputs:**[]**

The Processor block is now ready to handle a Device Clear if one occurs. At this point the machine then looks for a Device Clear Reset or a non-Device Clear Reset. A non-Device Clear Reset will cause the

Theory of Operation

machine to return to the <POWERON> state. A Device Clear Reset will cause the machine to enter the <NMI2> state.

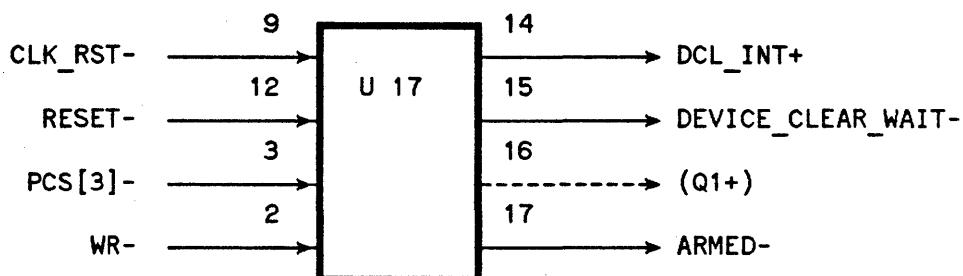
If no reset conditions are present, the machine loops in the <ARMED> state. This state is the normal operating state of the machine when the HP27111A is operational.

Device Clear State:<NMI2>
Outputs:[DCL_INT+]

DCL_INT+ is asserted. The machine will always return to the <POWERON> state at this point, and wait for the Device Clear to complete.

The state of the Device Clear machine is communicated to the Processor block via the CLR_WAIT- and DCL_INT+ signals. As noted in the state machine description, CLR_WAIT- is asserted when the adapter is in a Device Clear condition. When CLR_WAIT- is deasserted, the adapter is in a normal operational state. Thus, if a DCL_INT+ does occur, the Processor block needs to sample CLR_WAIT- and wait until the signal deasserts before attempting to access any devices that are reset by the Device Clear. The nature of the machine is such that DCL_INT+ is asserted for only one state.

The Device Clear machine is implemented using part of a 16R4 PAL {U17}. The table below shows the input and state pin assignments.



| STATE | [DCL_INT+,DEVICE_CLEAR_WAIT-,Q1+] | U17[14,15,16] |
|---------|-----------------------------------|---------------|
| POWERON | . T . | 000 (0) |
| IDLE | . . T | 011 (3) |
| ARMED | . . . | 010 (2) |
| QUEUE | . T . | 001 (1) |
| NMI1 | T T T | 101 (5) |
| NMI2 | T . . | 110 (6) |

[alidclbk]

MODE INITIALIZATION.

When PASSPORT is reset, it performs a mode initialization operation to determine how it should operate, once the reset has been removed. The INTR- line is used to set the operating mode. It is sampled by

PASSPORT one SCLK+ cycle prior to the deassertion of RESET-. For proper operation this signal should be at a logical *HIGH* level at this time. Failure of this condition causes PASSPORT to initialize incorrectly and report to the Channel via the sense register that it is a CIO Level 1 card.

Diagnostic Interface Port

The NMOS-III methodology provides a **Diagnostic Interface Port** which is ordinarily used only in wafer and packaged part verification. The port provides a method of serially scanning in and out vector information for NMOS-III chips.

The HP27111A is set up such that at some future time, defect analysis equipment may be used to access this test port on PASSPORT (as well as PRONTO).

| |
|-------------|
| NOTE |
|-------------|

While the capability to use this feature is presented, it will not be tested and is not guaranteed. More information on the function can be obtained by consulting the NMOS-III Test Methodology Guidelines.

The port consists of 5 inputs and 1 output. Four of these inputs are shared with the PRONTO chip. The other unique input is the Diagnostic Port select line.

COMMON INPUTS.

The four inputs that are shared with the PRONTO chip are STB-, REF-, DIN- and DSL. These lines are pulled up through 10k resistors to +5 volts.

STB- {U14-N2} is pulled up through U35-9. It is used to strobe in or out a bit of serial data.

DIN- {U14-N1} is pulled up through U35-10. It is the serial data input line to the diagnostic port.

REF- and DSL {U14-L3,M1} are tied together and pulled up through U35-11. They would ordinarily be unused during any sort of board test.

DIAGNOSTIC PORT SELECT.

PPORT_DIP_SEL- {U14-M3} is pulled up through U35-8. It is used to select the port for an operation.

Theory of Operation

DIAGNOSTIC OUTPUT.

The diagnostic ports output for PASSPORT can be monitored at *U14-M2*.

Power References

In addition to the nominal +5V and GND voltage connections, PASSPORT's NMOS-III process requires two additional reference voltages: VDL and VBG.

VDL. VDL is the differential latch voltage and is set to a nominal voltage of 2.7V.

VBG.

VBG is a "backgate bias" voltage that is applied to the substrate. It is set to a nominal voltage of -2V.

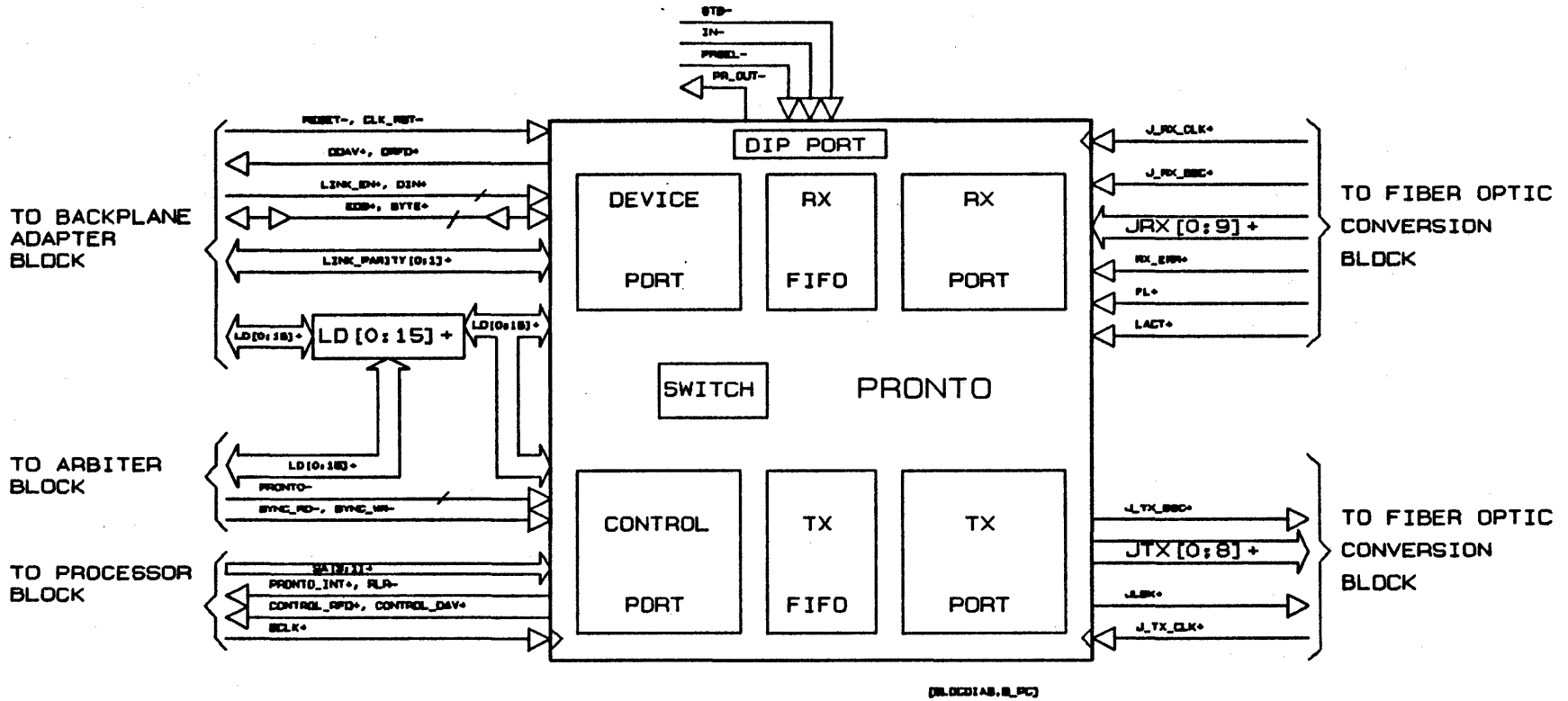
PROTOCOL CONTROLLER

File: [ALIMSPCO.TAK.ALINK]

The Protocol Controller functional block is implemented in hardware by a single IC -- PRONTO {U34}. This IC is a multi-port device and consists of the following, internal sub-blocks:

- Control Port
- Device Port
- TX Fifo
- Receive FIFO
- Switch
- Transmit Port
- Receive Port

Of the above ports, the Transmit and Receive Ports' have individual data busses, while the Control and Device Ports share a common data bus.



Control Port

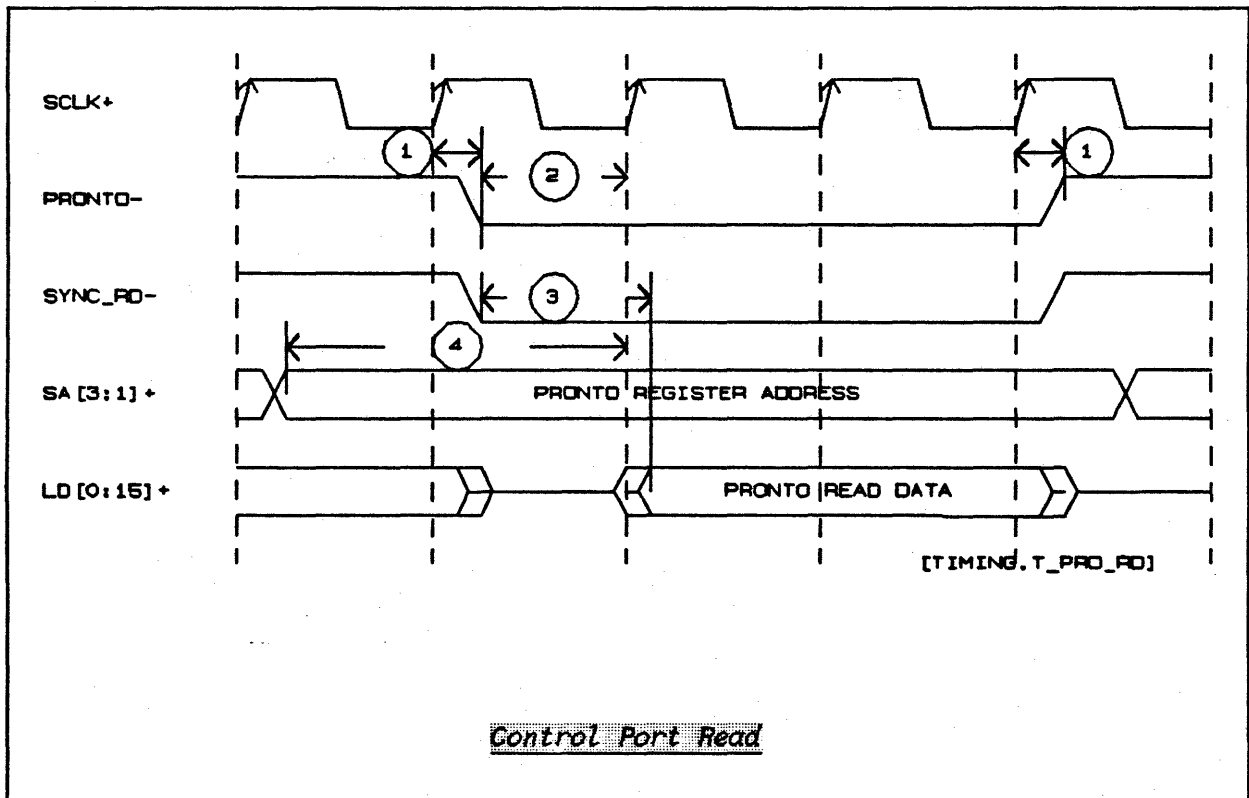
The Control Port is PRONTO's interface to the Processor. It consists of control, address, and the Link Data bus. The Control Port gives the Processor access to various configuration and status registers within PRONTO as well as providing hardware status lines for use by the Processor.

CONTROL AND ADDRESS.

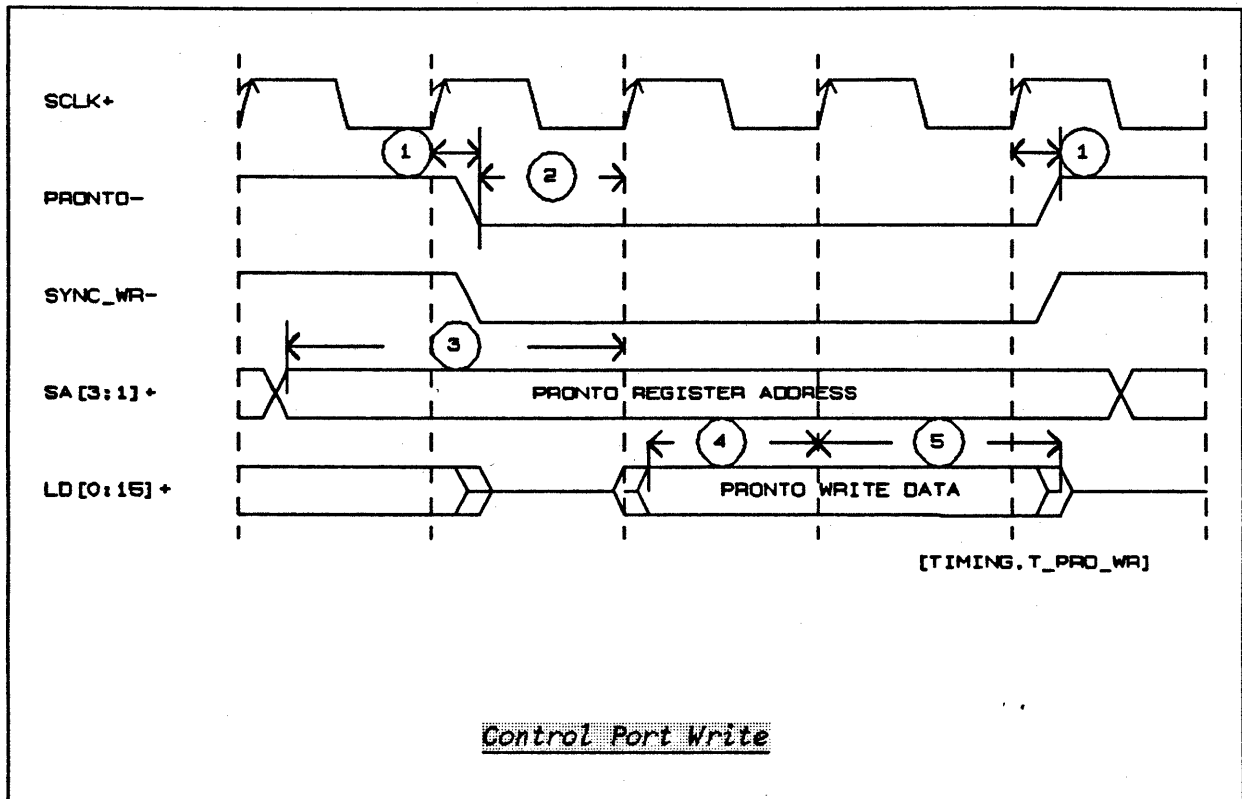
The Control and Address interface is a standard registered I/O device form. It consists of a select line **SELECT-** {U34-C14}, address bus **SA[3:1]+** {U34-B10,A10,B09}, and two I/O strobes **SYNC_RD-** {U34-C13} and **SYNC_WR-** {U34-B14}.

Asserting **SELECT-** and **SYNC_RD-** causes data addressed by **SA[3:1]+** to be driven onto the Link Data Bus. In the same fashion, the assertion of **SELECT-** and **SYNC_WR-** causes the current contents of the Link_Data Bus to be stored into the PRONTO register addressed by **SA[3:1]+**. During a write operation, PRONTO samples data on the second rising edge of **SCLK+** following the assertion of **SCLK+**.

TIMING.



Theory of Operation



Control Port Access Timing Paramters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|--------------------------|--|-----|-----|
| 1 | $t_{\text{CNTL_HOLD}}$ | hold of any of the following <i>control</i> signals (PRONTO-, SYNC_RD-, SYNC_WR-) to rising edge of SCLK+. | 0 | 31 |
| 2 | $t_{\text{CNTL_SETUP}}$ | setup of any of the following control signals (PRONTO-, SYNC_RD-, SYNC_WR-) to rising edge of SCLK+. | 87 | ... |
| 3 | $T_{\text{SA_SETUP}}$ | setup of SA[3:1]+ to first rising edge of SCLK+ after SELECT- has asserted | 87 | ... |
| 4 | $t_{\text{RD_DATA}}$ | delay of LD[[]]+ from assertion of SYNC_RD- and SELECT- | 0 | 88 |
| 5 | $t_{\text{LD_SETUP}}$ | setup of LD[[]]+ to second rising edge of SCLK+ after SELECT- has asserted | 7 | .. |
| 6 | $t_{\text{LD_HOLD}}$ | hold of LD[[]]+ from second rising edge of SCLK+ after SELECT- has asserted | 18 | .. |

DATA.

During Control Port accesses, only the 16 bit LD[0:15]+ bus has any meaning. There is no Parity on the Control Port, and the auxiliary data bits only have meaning for Device Port transfers.

HARDWARE STATUS LINES.

PRONTO has a number of Hardware Status Lines that are used by the Processor to detect various states of the PRONTO interface.

The PRONTO_INT+ {U34-A13} line indicates that an interrupt condition exists within PRONTO that the Processor has not masked. This line will remain asserted until either (a) the interrupt condition is cleared, or (b) the interrupt condition is masked by the Processor.

The CONTROL_RFD+ {U34-A11} line indicates to the Processor that PRONTO can accept a word of data written to its Write Data Register.

The CONTROL_DAV+ {U34-B11} signal, when asserted, indicates that PRONTO can provide a word of data read from its Read Data Register.

The RLR- {U34-A12} signal is asserted for 16 system clock cycles when PRONTO detects that an RLR (Remote Link Reset) has appeared at its Receive Port and that the appropriate RRA (Remote Reset Acknowledge) has been sent out the Transmit Port.

REGISTER SET.

The PRONTO Register Set consists of 5 Read and 8 Write registers.

Pronto Register Set

| SA[3:1] | Read Operation | Write Operation |
|---------|------------------|-----------------------|
| 0 | Rx FIFO Data | Tx FIFO Data |
| 1 | Interrupt Status | Interrupt Acknowledge |
| 2 | Interrupt Mask | Interrupt Mask |
| 3 | Link Error Count | DMA Control |
| 4 | Pronto Status | Configuration |
| 5 | undefined | Rx Header Size |
| 6 | undefined | Virtual Circuit Match |
| 7 | undefined | Tx Header Size |

[aliprreg]

More details on the function of the individual registers can be obtained from the PRONTO ERS.

Device Port

The Device Port is PRONTO's interface to the Backplane Adapter. It consists of a 16 bit data bus, 2 parity bits, 2 auxiliary bits and 4 control lines.

DATA BUS.

The Device Port Data Bus is 16 bits wide and can be viewed of as an upper and lower byte. (LD[0:7]+ and LD[8:15]+). Each of these bytes is protected by a parity bit, LINK__PARITY[0]+ {U34-N14} for the upper byte and LINK__PARITY[1]+ {U34-G13} for the lower byte. PRONTO will always generate ODD parity and may be optionally programmed to check for ODD parity. If a parity error is detected, PRONTO's Interrupt Status Register will log this event.

The Auxiliary Bits, EOS+ and BYTE+ {U34-F14, M13} are used to indicate data termination and odd data count respectively. These bits are asserted concurrently with data. On Device Port Write, the EOS+ signal indicates that the current data should be delimited by an Layer 2 EOS flag when sent out on to the link. On a Device Port Read, PRONTO will assert EOS+ if the current data being transferred was delimited by EOS+.

The BYTE+ signal can only be asserted when EOS+ is true. It indicates that only the upper byte of the data bus contains valid information.

CONTROL BUS.

PRONTO uses five control lines to transfer data to and from the Backplane Adapter: DRFD+, DDAV+, LINK__EN+, DIN+ and DEVRDY. {U34-E14, F13, D13, D14, E13}

DRFD+ (Device Ready for Data) is asserted when the Processor has programmed PRONTO to allow Device Port Writes and there is space available in PRONTO's internal Transmit FIFO for data.

DDAV+ (Device Data Available) is asserted when the Processor has programmed PRONTO to allow Device Port Reads and there is data present in PRONTO's internal Receive FIFO.

The LINK__EN+ and DIN+ signals are used by PRONTO to determine when the Backplane Adapter has completed a transfer. If the Backplane Adapter request matches PRONTO's capability, a transfer occurs and PRONTO updates its internal FIFOs. If the Backplane Adapter makes a request and PRONTO is unable to complete the transfer, PRONTO ignores the request. The assertion of LINK__EN+ during a Device Port Read also causes PRONTO to drive the data bus as well as the auxiliary bits.

DEVRDY+ when asserted with LINK__EN+ indicates to PRONTO that a Device Port Transfer has completed in the direction specified by DIN+ as long as the appropriate Data Ready signal was asserted. This signal is pulled up by U35-15, thus LINK__EN+ paces the transfer of data.

Switch

The Switch is PRONTO's mechanism of automatically (or manually) alternating transfers between the Control and Device Ports. Each of PRONTO's internal FIFO's contain a Switch. The behaviour of the switch is initially setup through PRONTO's register set. The position of the switch determines which of the Ports may perform a transfer and which of the FIFO ready lines (DRFD+,DDAV+,CONTROL_RFD+,CONTROL_DAV+) are asserted.

MANUAL MODE.

When PRONTO is in Manual mode, the state of the switch is controlled directly through the PRONTO DMA Control register. The Processor can explicitly control which ports will generate the RFD and DAV signals. The bits which control this function are PRONTO DMA Control[TX_DNC,RX_DNC]. The "DNC" stands for Device+/Control-, and writing a "1" to this register's bit position would allow a FIFO, if it had been previously enabled, to perform Device Port transfers.

AUTOMATIC MODE.

In automatic mode, DMA capabilities are switched between Control and Device Port according to the following table.

Automatic Mode Behaviour

| FIFO | DEVICE PORT => CONTROL PORT | CONTROL PORT => DEVICE PORT |
|------|--------------------------------------|--|
| Tx | Device Port Write with EOS+ asserted | Tx Header Size count expiration. Switch moves on last write of the transmit buffer. |
| Rx | Device Port Read with EOS+ asserted. | Rx Header Size count expiration with a Virtual Circuit Match. Switch moves on last read of the header. |

The only exception to the above table is that if PRONTO has been configured in software to send out an EOS at the expiration of the Tx Header Size count, the Tx FIFO switch will not move.

Automatic mode is enabled by writing Pronto DMA Control[TxAUT] or Pronto DMA Control[RxAUT] according to restrictions outlined in the PRONTO ERS.

Rx FIFO

The Rx (Receive) FIFO is a 128 byte deep memory that is word wide on the Control/Device side and byte wide on the Receive side. Valid data from the Fiber Optic Conversion block is loaded into the Rx FIFO through the Receive port where it may be unloaded by either the Control or Device Port depending upon which port has been programmed for access via the Rx Switch. If the Rx FIFO has been enabled by

Theory of Operation

the Processor and data is present, PRONTO will assert DDAV+ or CONTROL__DAV+ dependent on the Rx Switch state. The Rx FIFO is enabled by writing Pronto DMA Control[ERX].

One exception to the above is when PRONTO has been programmed by the Processor to Halt upon detection of the EOS auxiliary flag. In this case, when either the Control or Device port reads a data word that was delimited by the EOS flag, the DAV signals will deassert, irregardless of FIFO state, until the Receive EOS condition is cleared.

Tx FIFO

The Tx (Transmit) FIFO is a 128 byte deep memory that is used to queue data from the Link Data bus for transfer out the Transmit Port to the Fiber Optic Conversion circuitry. To the Device and Control ports it appears as a word wide (16 bit) interface with two Parity bits. Since the Control Port does not have parity, PRONTO automatically generates it when the Control Port writes to the Tx FIFO.

To the Transmit Port the FIFO appears as a set of eight bytewise buffers each having as many as 16 bytes within them.

The Tx FIFO must be enabled by the Processor before it will indicate that it can accept data. The FIFO is enabled by writing DMA Control[ETX]. When enabled, the FIFO will indicate that it is Ready For Data (RFD) by asserting either DRFD+ or CONTROL__RFD+, dependent on the state of the Tx Switch.

Tx Port

The Tx Port is used by PRONTO to communicate to the Jupiter Transmitter (JTX) circuitry. It consists of a transmit clock which is used to transfer both data and control signals according to the A-link Layer 2 Link Protocol.

TRANSMIT CLOCK.

The Transmit Clock, J__TX__CLK+ {U34-K02} is a 6.66 Mhz signal which is received by the Tx Port. Data and Control signals are updated by the Tx Port on the rising edge of J__TX__CLK+.

CONTROL.

The Tx Port provides two control signals for Jupiter and the Fiber Optic circuitry: J__TX__BSC+ and JLBK+. {U34-P09, N08}

J__TX__BSC+ acts a byte synchronization signal and is used to delimit the layer 2 control and information frames.

JLBK+ is the Jupiter Loopback Control signal and when asserted causes the Jupiter Receiver to takes its data from the local Jupiter Transmitter rather than the Fiber Optic Receiver.

DATA.

The Data bus, JTX[0:8]+ is effectively an 8 bit data bus with a 1 bit type field. When JTX[0]+ is asserted the remaining data bits identify a layer 2 control character. When JTX[0]+ is deasserted, the remaining data bits are treated as purely data.

| JTX[0]+ | JTX[1:8] | Character Type |
|---------|------------|--|
| 0 | 00h - 0FFh | Data Character |
| 1 | DDh | EOF or End of Frame Control Character |
| 1 | EEh | IDL or Idle Control Character |
| 1 | FFh | VLF or Variable Length Frame Control Character |

[alixchr]

Transmit Character Code Mapping**LOOPBACK MODE.**

Before PRONTO enters or exits Loopback mode by asserting JLBK+, it will send the EOF control character indicating the current frame is completed followed by 16 control characters. JLBK+ will assert and 16 more IDL control characters will transfer. Finally, J_TX_BSC+ is asserted and the transfer of Layer 2 "Control" or "Information" frames is resumed.

LAYER 2 OPERATION.

For a detailed description of Layer 2 operation consult the A-link Protocol Standard.

Rx Port

The Rx Port is used by PRONTO to communicate with the Jupiter Receiver (JRX) circuitry. It consists of a receive clock which is used to transfer both data and control signals according to the A-link Layer 2 Link Protocol.

PRONTO uses the Rx Port to parse incoming Information ("I") and Control ("C") frames, according to layer 2 protocol.

Theory of Operation

RECEIVE CLOCK.

The Receive Clock, **J_RX_CLK+ {U34-H01}** is a 6.66Mhz clock which is used by the Rx Port to sample incoming data and control signals. All signals sampled by the Rx Port are relative to the rising edge of the clock.

CONTROL.

The Control interface from the Jupiter Rx to the Rx Port consists of a set of four signals (**J_RX_BSC+,FL+,RX_ERR+,LACT+**) which are used by the Rx Port to delimit the incoming byte stream, determine the data integrity, and determine the current signal quality of the Receive Optical signal.

J_RX_BSC+ {U34-B06} acts as a Receive Byte Synchronization signal and is used by the Rx Port's internal parsing machines to locate the beginning of layer 2 "I" and "C" frames. It corresponds to the transmitter's Transmit Byte Synchronization signal.

FL+ {U34-B06} is used to determine the state of Jupiter Rx's phase lock loop. When asserted, it indicates that the loop is in lock. Every time **FL+** deasserts, PRONTO ignores the current frame and waits 16 receive clock cycles before it resynchronizes its layer 2 operations by looking for the assertion of **J_RX_BSC+**.

RX_ERR+ {U34-B07} is used to determine that the current data being sampled could not be correctly received by the Jupiter Receiver. Upon detection of **RX_ERR+**, PRONTO will ignore the current frame it is parsing and resynchronize itself at layer 2. It will also increment its internal Link Error Counter.

LACT+ {U34-A07} is the Link Active status signal. PRONTO's internal parser will operate as long as this signal is asserted. A Link Inactive state will cause the parser to wait until **LACT+** reasserts. The internal parser will usually generate at least one Link Error on each transition of the **LACT+** signal.

DATA.

The Receive Data bus, **JRX[0:9]** is a 10 bit data bus with a 2 bit type field **JRX[0,5]** and an 8 bit data field **JRX[1:4,6:9]**. The type field is used to distinguish three types of layer 2 characters: Control, Data, and Illegal as shown in the following table.

Illegal characters will cause the parser to ignore the current frame and increment the Link Error Counter.

| JRX[0,5]+ | JRX[1:4,5:8] | Character Type |
|-----------|--------------|--|
| 0 0 | 00h - 0FFh | Data Character |
| 0 1 | xxh | Illegal |
| 1 0 | xxh | Illegal |
| 1 1 | DDh | EOF or End of Frame Control Character |
| 1 1 | EEh | IDL or Idle Control Character |
| 1 1 | FFh | VLF or Variable Length Frame Control Character |

[alirxchr]

Receive Character Code Mapping

CONTROL PORT MONITORING OF LINK STATUS.

Certain Rx Port Link Status information is available for monitoring via Control Port register accesses. In particular, Link Error Count, Link State Change, Jupiter Rx Out of Lock, and Link Active status are available.

Link Error Count is the total of all link error events since the last read of the Link Error Count register or the acknowledgement of PRONTO's Link Error Interrupt (LERR). It is obtained by reading the Link Error Counter register.

Link State Change is an interrupt that is generated anytime there is a transition of the FL+ or the LACT+ signals. It is found in the Interrupt Status register.

Jupiter Rx Out of Lock, is the inverse and slightly delayed version of the FL+ signal. It is found as the PRONTO Status[OFL+] bit.

Finally, a slightly delayed version of the Link Active state (LACT+) is available as the PRONTO Status[ACT+] bit.

Reset Behaviour

The Protocol Controller is reset every time RESET- asserts. In addition, PRONTO's internal clock generators are restarted each time CLK__RST- is asserted.

POWER ON.

During Power On Resets both RESET- and CLK__RST- are initially asserted.

Theory of Operation

If the system clock, SCLK+ is active and periodic. The Protocol Controller will enter its reset state on the leading edge of SCLK+ following the assertion of RESET-.

The resultant state of the Protocol Controller's external signals is shown on the following table.

Protocol Controller Reset State

| SIGNAL STATE | SIGNAL NAME |
|----------------|--|
| Asserted | (none) |
| Deasserted | PRONTO_INT+, CONTROL_RFD+, CONTROL_DAV+, DEV_RFD+, DEV_DAV+, RLR-, JLBK+, JTX_BSC+ |
| High Impedance | BYTE+, END+, LINK_PARITY[0:1]+, LD[0:15]+ |
| Unknown | JTX[0:8] |

Internal to PRONTO, the following occur. The Interrupt Mask, Link Error Count, Tx Header, VC Match, and Rx Header registers are cleared. Both FIFO's are disabled. Both Switches point to the Control Port and are placed in Manual Mode. The Interrupt Status register is undefined.

PRONTO cannot respond to Processor operations until RESET- deasserts. Attempts to access PRONTO via the Arbiter while RESET- is asserted will produce indeterminate results.

Approximately 6 SCLK+ cycles after RESET- deasserts, the Transmit Port will begin to send out a repeating 6 character sequence of data as well as asserting J_TX_BSC- every 6th rising edge of J_TX_CLK+.

The Receive Port will begin to process data at this point, also.

The Protocol Controller's interface to the Processor and Backplane Adapter will remain in the state produced by RESET- until the Processor has programmed and initialized PRONTO.

The only exception to this is that if a Remote Link Reset is received at PRONTO's Rx Port, RLR- will assert as usual.

CHANNEL RESET.

The Channel Reset behaviour for the Protocol Controller is the same as with Power On Reset.

DEVICE CLEAR.

The Device Clear behaviour for the Protocol Controller is the same as Power On Reset with the following exception. PRONTO's internal clock generation circuits are not reset since CLK_RST- is not asserted.

Diagnostic Interface Port

PRONTO contains the same Diagnostic Interface Port that PASSPORT does. At the time of this writing it was unclear whether the multiple clock operation of PRONTO would prevent the port from being used.

The port consists of 5 inputs and 1 output. Four of these inputs are shared with the PASSPORT chip. The other unique input is the Diagnostic Port select line.

COMMON INPUTS.

The four inputs that are shared with the PASSPORT chip are STB-, REF-, DIN- and DSL (U14-N1,L2,M2,L1). These lines are pulled up through 10k resistors to +5 volts.

These lines have the same function as described in the subsection on *Diagnostic Interface Port* in the *Backlane Adapter* section.

DIAGNOSTIC PORT SELECT.

PRONTO_DIP_SEL- {U34-N2} is pulled up through U35-5. It is used to select the port for an operation.

DIAGNOSTIC OUTPUT.

The diagnostic ports output for PRONTO can be monitored at U34-M1.

FIBER OPTIC CONVERSION

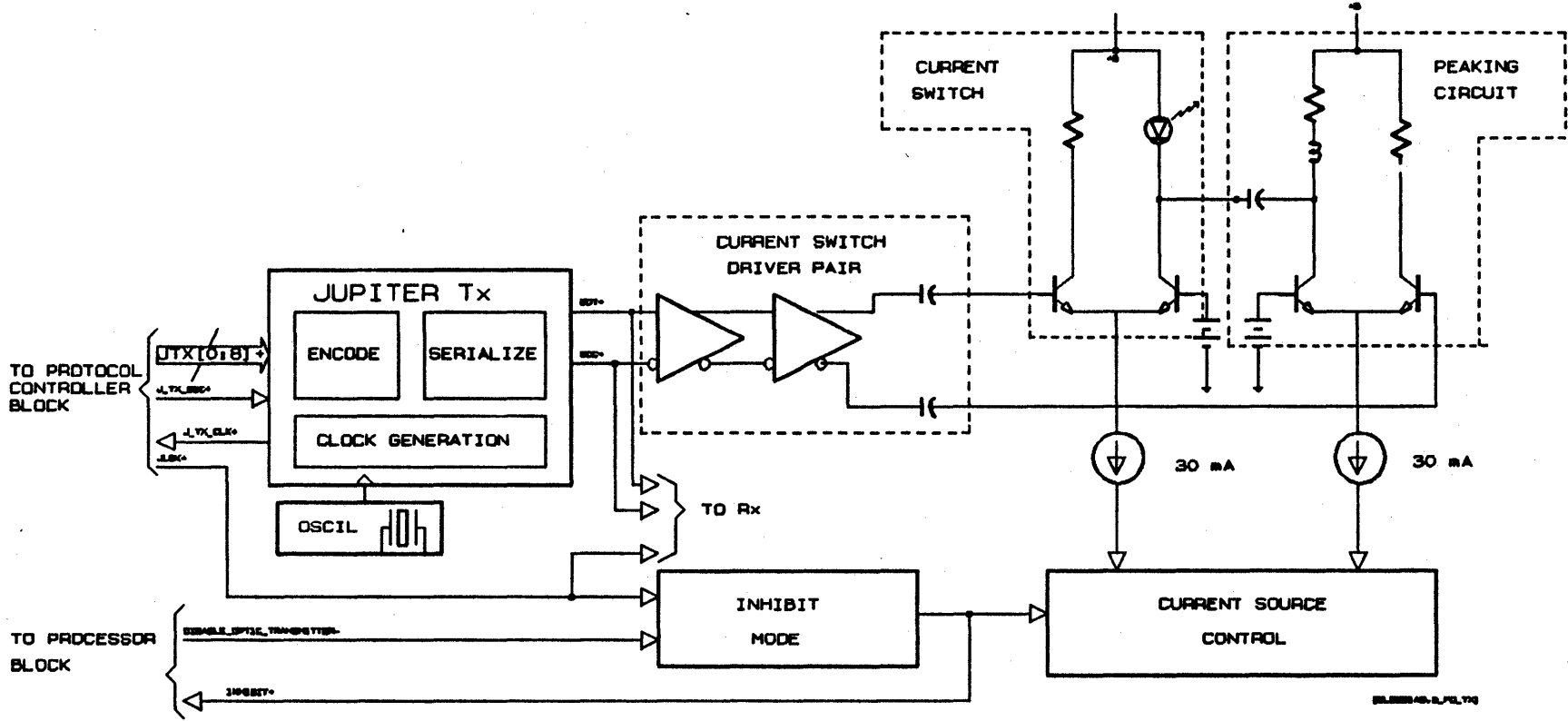
File: [ALIMSFOO.TAK.ALINK]

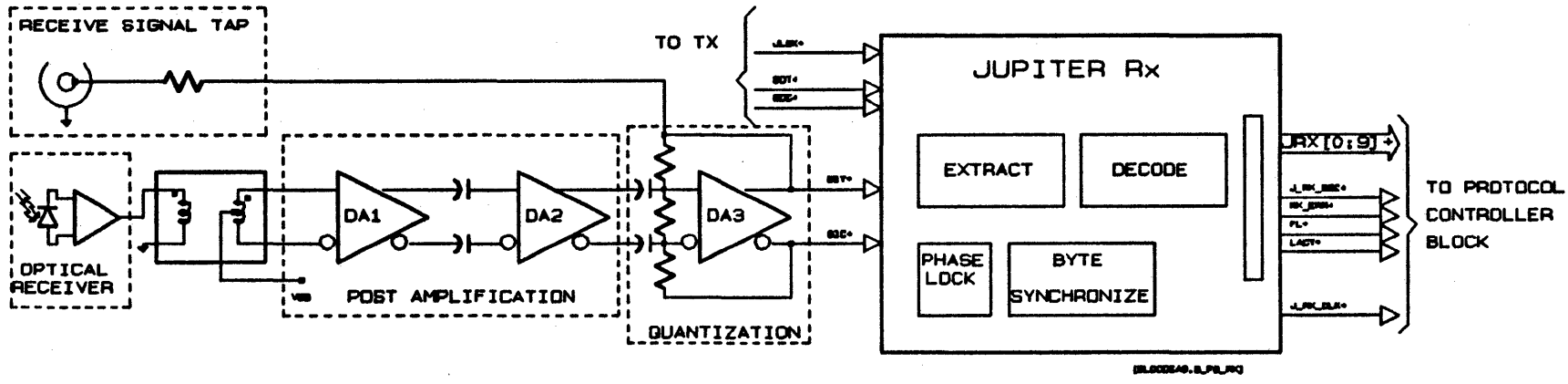
The Fiber Optic Conversion block provides the conversion between the parallel data format of the Protocol Controller and the serial data format of the Optical Interface.

Thus, there are two main data paths in the Fiber Optic Conversion (FOX) block: Transmit and Receive.

In the Transmit data path, parallel data is taken from the Protocol Controller's Tx Port, serialized by the Jupiter Tx block, and converted to light energy for transmission onto the fiber by the Optical Transmitter block.

In the Receive data path, the incoming light variations are converted to electrical pulses, amplified into a serial bit stream by the Optical Receiver block, and then converted to a parallel format by the Jupiter Rx block for further processing by the Protocol Controller.





Parallel to Serial (Jupiter Tx)

The Parallel to Serial block takes parallel data from the Protocol Controller's Tx Port and encodes and serializes it for conditioning by the Optical Transmitter Driver circuitry.

The block consists of the Jupiter Tx {U43} integrated circuit, which contains a Parallel Port, Data Encoding, and Serializing functions, as well as various discrete components to generate the Transmit Clocks.

TRANSMIT CLOCK GENERATION.

Jupiter Tx has two main clocks: the J_TX_CLK+ parallel clock and a 40 Mhz internal serial clock.

Both of the clocks are derived from the output of Y2 which is a 6.66Mhz crystal canned oscillator. The oscillator's output Y2-8 is connected to the External Byte Clock input of Jupiter Tx U43-G8. Jupiter is programmed to use this clock source input by tying U43-H7 to GND.

J_TX_CLK+ is essentially a delayed version of Y2-8, and is used by Jupiter Tx to sample data relative to its rising edge.

The 40 Mhz serial clock is generated through internal phase lock loop circuitry which is set by R25, L3, C42 and C43.

The ferrite bead FB1 is used to prevent high frequency noise from coupling into Jupiter Tx's phase lock loop control input U43-H3.

PARALLEL TRANSFER.

Jupiter Tx samples the J_TX_BSC+ {U43-A8} and JTX[+] bus signals on every rising edge of J_TX_CLK+ {U43-A7}. Note that Jupiter operates on a 10 bit data word, even though Pronto's Tx Port is only 9 bits wide, since JTX[0]+ is tied to both U43-A7 and U43-D7. J_TX_BSC+ when asserted overrides JTX[+] and is flagged for use by the Data Encoder.

DATA ENCODING.

All data is encoded before it is serialized and sent out the fiber. The reason for the encoding process is that it assures that the serial bit stream transferred onto the link has no or minimal DC offset. DC offset makes it difficult for a receiver at the other end of the link to quantize the incoming data stream, since either the positive or negative delta of the waveform is reduced as the offset increases.

A "5 to 6" balanced coding technique is used to minimize this offset. Each 10 bit data word is split into two different 5 bit nibbles and each nibble is transformed into a new 6 bit representation. Thus each 10 bit data word yields a new 12 bit word which will be serialized and placed on the link.

One special case for this coding technique is the Byte Synchronization signal J_TX_BSC+. This signal will cause the encoder to generate one of two special alternating 12 bit BSC (Byte Sync) control character codes.

Theory of Operation

Details on the coding scheme can be found in the Jupiter ERS or in the Alink Protocol document.

SERIALIZING.

The 12 bit encoded data word is then shifted out at an 80MBaud rate and appears as an emitter coupled logic differential signal pair SOT+, SOC- {U43-C2, C1}. This differential pair is sent both to the Optical Transmitter Driver block and the Jupiter Receive Serial to Parallel block (for use during loopback tests).

R17, R54 are used for ECL load termination to -5.2 Volts.

Optical Transmitter Driver

The Optical Transmitter section consists of:

- Transmit LED
- Current Switch Driver
- LED Current Switch
- LED Peaking Circuit
- Current Source Generator
- Inhibit Mode Control

TRANSMIT LED.

The Transmit LED is an HFBR-1402 with various and sundry specifications that will be included here at one time or another. Needless to say it is capable of 80Mbaud operation.

The Transmit LED emits photons when current passes from its anode to cathode. The optical power produced is an approximately linear function of the input current.

Current flowing through the LED will be considered to be at a logic "1" and asserted state; absence of current, the logic "0" or deasserted state. Thus, the presence of optical flux is a logic "1" state and the absence of flux, the logic "0" state.

When the transmitter is disabled the state of the Transmit LED is defined to be off or deasserted.

Actually, a small amount of current is flowing through the Transmit LED at all times. This current is provided by the path from the LED cathode through R46 to ground. This biasing current is necessary to allow the fast switching of the diode.

CURRENT SWITCH DRIVER.

The Current Switch Driver is used to buffer the differential serial output pair, SOT+, SOC- from the Jupiter Transmitter and drive the LED switch and peaking circuitry.

Two stages of an MC10216 ECL Driver {U47} are used to perform this function. SOT+ and SOC- drive the first stage at U47-13, 12. The outputs of the first stage, U47-15, 14 drive the second stage at U47-3, 2 to form DSOT+ and DSOC-.

This differential output pair is isolated from the LED current switches by C54, C55 for D.C. biasing purposes.

Resistor network R22 is used to terminate and provide loading for the ECL output drivers of U47. The common of the resistor network R22-1 is tied to VEE (-5.2V).

Capacitors C40 and C38 are used for supply bypassing.

LED CURRENT SWITCH.

Current is selectively driven through the Transmit LED by means of a differential Current Switch. Transistors U55[Q7] and U55[Q8] form a differential pair, with the collector of U55[Q7] loaded by the Transmit LED. The emitters of the differential pair are tied to a constant 30 mA current source through resistors R30, R49. Both bases are biased to approximately 2.5 Volts with resistor network R34-2, 3. The base of U55[Q8] is effectively driven by the Current Switch Driver's complimentary output DSOC- through isolation capacitor C54. The other base is not driven, resulting in *single-ended* operation.

DSOC- has very little D.C. offset, by virtue of the Jupiter Transmitter's balanced coding scheme. It's voltage swing is approximately 0.7 volts. The net effect of this is that isolation and coupling capacitor C54 tends to produce a 0.7 volts peak to peak signal about the 2.5 volts bias point at the base of U55[Q8].

When DSOC- asserts, the negative voltage swing is tracked by C54, lowering the voltage at the base of U55[Q7]-2. This causes the base of U55[Q8] to drop to approximately 2.15 volts. As the 30mA current source attempts to track this change by lowering the potential of the junction of R30, R29, the LED driver transistor U55[Q7] begins to turn on. The emitter resistor junction cannot swing to a low enough potential to keep U55[Q8] on so all of the current is supplied by U55[Q7]. At this point the emitter resistor junction is at approximately 1.60 Volts.

When DSOC- deasserts, the voltage swing, in this case, positive, is again tracked by C54. As the base voltage rises towards 2.85 volts, U55[Q8] begins to turn on again. As U99[Q8] begins to supply more of the 30mA current, the emitter resistor junction potential begins to rise, until it reaches 1.95 Volts. At this point, U55[Q7] is completely off.

Capacitor C56 filters noise at the base of U55[Q7] and maintains a stable bias potential, when the transistor switches. Without this transistor, the base bias voltage could wander due to parasitic capacitances both within the transistor and on the printed circuit board.

R43 loads U55[Q8].

LED PEAKING CIRCUIT.

The HP27111A also employs a peaking circuit to improve the quality of the optical waveform. The peaking circuit draws additional current through the LED when it is initially turned on, and actually charges up the cathode when the LED is initially turned off.

This action tends to sharpen the turn on and turn off response of the LED.

Theory of Operation

The circuit which performs the peaking function is another differential current switch $U56[Q5, Q6]$ which operates in parallel with the regular current switch. The collector of $Q6$ has a special load consisting of $R45$ and $L8$. It is coupled into the LED's cathode node via $C63$.

current switch consisting of $U56[Q5]$ and $U56[Q6]$ which operate in parallel with the regular current switch. The collector of $U56[Q6]$ has a special load consisting of $R45$ and $L8$. It is coupled into the LED's cathode node via $C63$. The emitters of the differential pair are tied to a second constant 30 mA current source through resistors $R35$, $R36$. Both bases are biased to approximately 2.5 Volts with resistor network $R34-4$, 5 . The base of $U56[Q6]$ is effectively driven by the Current Switch Driver's true output DSOT+ through isolation capacitor $C55$. The other base is not driven, resulting in *single-ended* operation.

DSOT+ has the same A.C. and D.C. characteristics of DSOC- described earlier. Thus, $C57$ tends to produce a 0.7 volts peak to peak signal about the 2.5 volts bias point at the base of $U56[Q6]$.

The basic behaviour of the peaking current switch is the same as with the LED driver current switch. When DSOT+ asserts, the positive voltage swing is tracked by $C55$, raising the voltage at the base of $U56[Q6]$. This causes the base of $U56[Q6]$ to rise to approximately 2.85 volts. As the 30mA current source attempts to track this change by raising the potential of the junction of $R35$, $R36$, transistor $U56[Q5]$ begins to turn off, its base being held at a fixed potential. Thus, all 30mA must be supplied by $U56[Q6]$. At this point the emitter resistor junction is at approximately 1.95 Volts.

As $U56[Q6]$'s collector begins to sink current, inductor $L8$ initially presents a high impedance current path. To satisfy the collector's demands, capacitor $C63$ provides an alternate lower impedance path. The current demands are then supplied by the cathode of the Transmit LED. As time passes, $L8$ begins to conduct current, and $C63$ supplies less current.

The net effect of this is a current pulse at the Transmit LED on the asserting edge.

Similarly, on the deassertion of DSOT+, the negative voltage swing is tracked by $C55$, lowering the voltage at the base of $U56[Q6]$. This causes the base of $U56[Q6]$ to lower to approximately 2.15 volts. Again, the 30mA current source attempts to track this change by lowering the potential of the junction of $R35$, $R36$, turning $U56[Q5]$ on and $U56[Q6]$ off.

$U56[Q6]$ turns off quickly. However, $L8$ wants to continue providing current. This current travels to the cathode of the Transmit LED via $C63$. Since $U55[Q7]$ is off at this time, current is injected into the LED. From a charge standpoint, electrons are removed from the cathode of the Transmit LED to $C63$ to balance the charge that was withdrawn to satisfy the needs of the inductor.

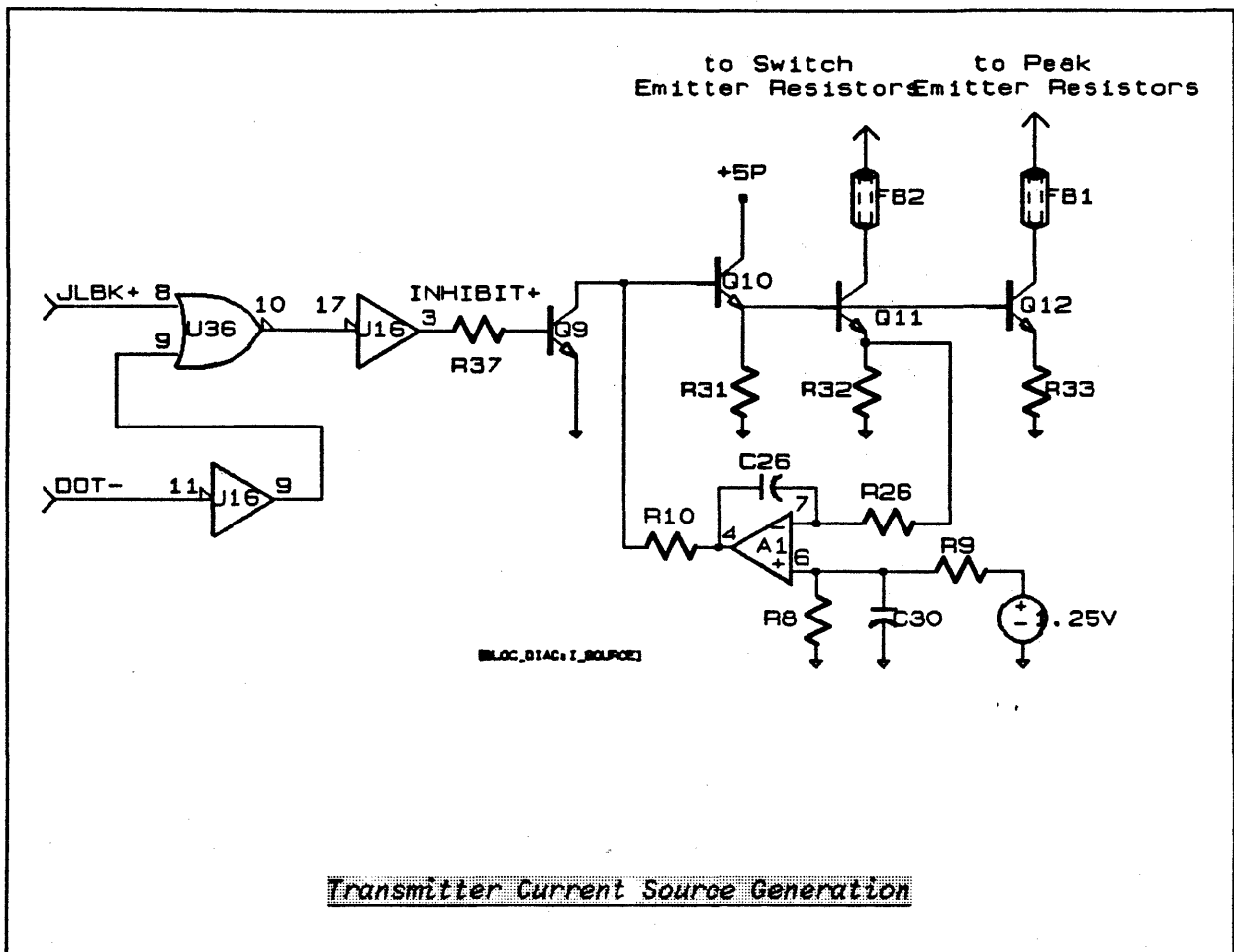
This effectively causes a negative "peak" in the current waveform at the LED's cathode, and causes the cathode to charge up.

Capacitor $C57$ filters noise at the base of $U56[Q5]$ and maintains a stable bias potential, when the transistor switches. Without this transistor, the base bias voltage could wander due to parasitic capacitances both within the transistor and on the printed circuit board.

Resistor $R45$ is used for damping as well as loading $U56[Q6]$. $R44$ loads $U56[Q5]$.

CURRENT SOURCE GENERATION.

The Current Source Generation circuitry is used to set the level of operating current for the Optical Transmitter Driver.



Referring to the block diagram, in normal operation, INHIBIT+ is deasserted and the output voltage of U16-3 is less than 0.4 volts. Therefore U56[Q9] is off and may be ignored. The opamp U42[A1] and transistors U55[Q10], [Q11] form a closed loop which determines the amount of controlling base current that the current sources formed by U55[Q11] and U56[Q12] are provided.

The level of current flowing through U55[Q11] is determined by sense resistor R21 which provides a sense voltage proportional to the current. The current flowing through U74[Q12] is the same magnitude as U99[Q11] since both transistor bases are common and R32 and R33 are equal.

The sense voltage is compared against a 0.6 volt reference established by voltage divider R8, R9 at the negative and positive terminals, respectively of opamp U51[A1]. A 0.6 volt reference is obtained because, R8 and R9 are equal in magnitude and the main reference U42-8 which they are dividing down is set to approximately 1.2 volts.

If the negative terminal is lower in voltage, a positive differential exists and the opamp output U42[A1]-4 increases. This causes current to flow through R10 and into the base of U55[Q10]. U55[Q10] is connected as an emitter follower and the amplified base current flows out the emitter and across R18. When the voltage across R31 begins to approach 0.6 volts, current also begins to flow into the base of U55[Q11] (and U56[Q12]). This causes the voltage across the sense resistor R23 to increase, which in turn, decreases the voltage differential, tending to stabilize the loop.

Theory of Operation

If the sense voltage is too high, the opamp lowers its output voltage, decreasing the amount of bias current to the transistors and causes the sense voltage to decrease.

The loop will stabilize when the sense voltage is approximately 0.6 volts which corresponds to a *U55[Q11]* collector current of 30 milliamps.

Capacitor *C26* is a compensation capacitor to assure opamp stability.

Capacitor *C52* is used to filter out high frequency variations in the biasing voltage applied to the bases of *U55[Q11]* and *U56[Q12]*.

Compliance of the current sources is further aided by the presence of two ferrite beads, *FB3*, *FB2*, which are placed between the collectors of the current sources and the emitter resistor junctions. The beads have the property of *dissipating* (rather than storing) energy at high frequencies. The beads tend to isolate the current sources from switching effects at their loads.

INHIBIT MODE.

The Optical Transmit circuitry is placed in Inhibit Mode whenever INHIBIT+ is asserted. INHIBIT+ is asserted when either *JLBK+* or *DISABLE__OPTIC__TRANSMITTER-* (abbreviated as *DOT-* in the diagram) are asserted. The INHIBIT+ function is implemented by an AS02 {*U36*} and two inverters from the ALS240 {*U16*}.

When INHIBIT+ is asserted, sufficient current flows from *U16-3* through bias resistor *R37* to turn on *U56[Q9]* and place it in saturation. This forces the collector of *U56[Q9]* to lower its voltage to less than 0.5 volts. This causes *U55[Q10]* and then *U55[Q11]* and *U56[Q12]* to turn off.

Opamp *U42[A1]* will sense this condition and raise its output voltage. Since *U56[Q9]* is in saturation, however, it will sink all the current flowing through *R10*, and the current sources will remain disabled.

Deasserting INHIBIT+ will cause a large amount of current to flow into the *U55[Q10]*, *U55[Q11]* path, raising the sense voltage and the opamp will regulate the differential as previously explained.

Optical Receiver and Amplifier

The Optical Receiver and Amplifier section is used to take the optical signal from the fiber, convert it to an electrical waveform, and then amplify and quantize it for data extraction by the Jupiter Rx block.

It consists of:

- Optical Receiver
- Isolation Transformer
- Post Amplifier
- Quantizer
- Receive Signal Tap

OPTICAL RECEIVER.

The Optical Receiver *P3* uses the HFBR-2406 photodiode amplifier combination. This circuit converts light energy coupled into the unit into a low-level electrical waveform which appears at the pin 2 of the part.

The passive network of *C59*, *C61*, *C65*, *R40*, *R41* and *L6* provides a filtered version of the -5.2 volt supply which is used by the HFBR-2406's internal pre-amplifier.

The receiver unit requires that a 100 micron fiber be used in conjunction with an SMA type connector. Optimal operation requires that the SMA connector be securely attached to the barrel of the HFBR-2406.

The receiver outputs a positive voltage proportional to the signal strength. Higher output indicates more flux.

ISOLATION TRANSFORMER.

The Isolation Transformer *T1* is used to provide a controlled isolated impedance to the Optical Receiver and at the same time produce a differential signal from the single-ended output of the Receiver.

T1 is a "five port" pulse transformer. Its primary coil has two taps. Its secondary coil has two taps that provide a 1:1 winding ratio as well as a third center tap.

The output of the Receiver is capacitively coupled (*C66*) to the "dotted" tap of *T1*'s primary coil. The other tap of the coil is grounded. Positive excursions at the receiver, indicating increasing flux, cause the *T1-5* to be positive with respect to ground. Negative excursions at the receiver, indicating decreasing flux, cause *T1-5* to be negative with respect to ground.

The secondary's main taps *T1-2,6* are connected to *R51*. Since the turns ratio is 1:1, the Receiver sees a load of approximately 511 ohms. This value was selected as the optimal loading value for the receiver.

The secondary's center tap *T1-4* is connected to the bias voltage of the receiver section ECL amplifier, *U68-11*. This causes the secondary's main taps to swing around the nominal ECL bias voltage, yielding a differential ECL signal at the pulse transformer's secondary. A positive differential indicates presence of flux, with the "dotted" tap assumed to be the positive terminal.

POST AMPLIFICATION.

The differential output of the Transformer must be amplified and quantized before it can be processed by Jupiter's Receiver. The post amplification is accomplished by using two stages of ECL MC10216 {*U68*} differential drivers which are operated in or near their linear region as amplifiers. Each driver-amplifier stage has a voltage gain of approximately 5 so the two stage circuit's gain is on the order of 25.

Differentials from the Isolation Transformer appearing at *U68-10, 9* are amplified, and output at *U68-7, 6* to the next ECL stage. The amplified differential at *U68-5, 4* is amplified further and output at *U68-3, 2*.

Resistor network *R52* is used to terminate the ECL outputs of *U68*.

Theory of Operation

R42, *C60* and *C70* are used to filter the -5.2 Volt supply and prevent coupling to other analog stages through the -5.2 Volt plane.

QUANTIZATION.

The Quantizer uses hysteresis techniques to prevent the output of the Optical Receiver block from chattering in the absence of a signal. This chattering is normally the result of noise injected into the various stages of amplification. Hysteresis is set such that the noise level is usually lower than the switching threshold.

The output of the second driver-amplifier stage is A.C. coupled into the differential input of a third ECL driver-amplifier. Resistors *R48*, *R49*, and *R50* are used to provide the necessary feedback for hysteresis.

In the absence of activity, the outputs of *U68[DA3]* will tend to lock in either a positive or negative difference, due to *R48*, *R49*, and *R50*. In effect, these resistors set up a dual tapped voltage divider of the output differential, such that a scaled and matching differential appears at the driver-amplifier's inputs, which tends to lock the amplifier in a given output state.

The actual amount of signal fed back to the input pair may be calculated as

$$\begin{aligned} V+ &= \frac{R_{33} + R_{35}}{R_{33} + R_{34} + R_{35}} * (V_{sit} - V_{sic}) \\ &= 0.64 * (V_{sit} - V_{sic}) \end{aligned}$$

$$\begin{aligned} V- &= \frac{R_{33}}{R_{33} + R_{34} + R_{35}} * (V_{sit} - V_{sic}) \\ &= 0.36 * (V_{sit} - V_{sic}) \end{aligned}$$

where *V+* and *V-* represent the voltages at the inputs to the quantizing amplifier.

With an output difference of approximately 0.7 volts this leaves an input differential of ~ ±200 mV. In order to change *U68[DA3]*'s output state, a sufficient opposite differential must be applied to *U68[DA3]*'s inputs. Coupling capacitors *C68* and *C69* provide a low enough impedance on signal transitions from the Post Amplifier such that deltas greater than 180mV cause the output to change.

The differential outputs *U68-15*, *14* source terminated by *R38*, *R39* then form the serial input data pair {*SIT+*,*SIC-*} and are sent to Jupiter Rx for further processing.

RECEIVE SIGNAL TAP.

The Serial Input signal can be monitored by external measurement equipment through use of the Receive Signal Tap connector *J3*. This connector provides a subminiature single ended tap which may be used by a variety of probes.

To minimize the effects of cable loading on the internal signals, *R53* is used to isolate the probe tap.

Serial to Parallel (Jupiter Rx)

The transformation of incoming serial data to a parallel mode for transfer to the Protocol Controller's Tx Port is performed by the Serial to Parallel block. The block consists of the Jupiter Rx integrated circuit and a passive phase lock loop tuning network.

SERIAL DATA EXTRACTION.

The Jupiter Rx selects from two possible sources of incoming data according to the state of the *JLBK+* signal. When *JLBK+* is deasserted, which is the normal operating mode, data is extracted from the serial differential stream appearing at {*SIT+*,*SIC-*}. During internal loopback, when *JLBK+* is asserted, data is taken from the Jupiter Tx's {*SOT+*,*SOC-*} pair.

The first thing that the Jupiter Rx does is sense whether there are any transitions on the selected differential input pair. Presence of transitions is indicated by the assertion of *LACT+*.

If there are transitions on the selected differential input pair, the Jupiter Rx takes this bit stream and attempts to extract a baud clock from it using phase-locked loop techniques. The nominal frequency of this baud clock is 80 Mhz. The ability of the phase-lock loop to lock onto the incoming signal is indicated by the assertion of *FL+*. In other words it is a measure of the "bit synchronization" ability of the circuit. It should be noted that the *FL+* status signal has no meaning if *LACT+* is deasserted.

RECEIVE PHASE LOCK.

Jupiter Rx has two main clocks: the *J_RX_CLK+* parallel clock and a 80 Mhz internal serial clock.

The 80 Mhz internal clock is derived from the input serial data stream. The locking window of frequency range is set by tuning components *R28*, *C52*, *L4*, *C48* and *C49*.

J_RX_CLK+ is "divide by 12" version of the 80Mhz clock. All data on the *JRX* Parallel Port is valid on the rising edge of this clock.

The ferrite bead *FB2* is used to prevent high frequency noise from coupling into Jupiter Rx's phase lock loop control input *U44-H3*.

Theory of Operation

BYTE SYNCHRONIZATION.

Once bit synchronization is achieved, state machines internal to the Jupiter Rx attempt to achieve Byte Synchronization, that is the alignment of the incoming bit stream on their normal 12 bit boundaries. The machines accomplish this by scanning the incoming bit stream for the BSC control character. Once the BSC character has been aligned, the output of the Character Decoder should be valid.

CHARACTER DECODING.

The Character Decoding circuitry reverses the effect of the "5 to 6" bit transformation by the Jupiter Tx circuitry. It is essentially a ROM decoder and presents its decoded output to the JRX Parallel Port.

The outputs of the Decoding ROM are considered to be valid whenever an aligned 6 bit nibble is placed at its inputs. If an invalid code is detected at this time, the `RX_ERR+` will assert coincident with the invalid data at the JRX Parallel Port.

The Decoding ROM also detects the special case in which a BSC character has been received.

JRX PARALLEL PORT.

The JRX Parallel Port connects directly with the Protocol Controller's Rx Port with the signals described in that section.

The data (`JRX[0:9]+`), synchronization (`J_RX_BSC+`) and integrity (`RX_ERR+`) signals are updated on the rising edge of each `J_RX_CLK+` from the output of Character Decoder.

Also updated are the states of the `FL+` and `LACT+` lines.

As noted before, the function $(FL+) * (LACT+) * (\sim RX_ERR+)$ must be true if the data or byte sync character is to be considered valid at the rising edge of `J_RX_CLK+`.

Reset Behaviour

The Fiber Optic Conversion Block exhibits the same behaviour for all types of Reset conditions. Since `DISABLE_OPTIC_TRANSMITTER-` is asserted during all Reset conditions, the Optical Transmitter is disabled and no flux is sent on through the fiber. The Jupiter Transmitter continues to operate during the Reset. However, the data supplied to it by the Protocol Controller is indeterminate, and therefore the serial output of the Transmitter is also indeterminate.

The Fiber Optic Receive path from the Optical Receiver to the Jupiter Receiver Parallel Port continues to operate as normal. It is unaffected by Reset inasmuch as it will continue to extract whatever data is received at its optical input.

Both parallel clocks, `J_TX_CLK+` and `J_RX_CLK+` remain operational and periodic during reset.

Fiber Optic Conversion Reset State

| SIGNAL STATE | SIGNAL NAME |
|--------------|---|
| Asserted | INHIBIT+ |
| Deasserted | (none) |
| Unknown | JRX[0:9]+, SOT+, SOC-, SIT+, SIC-, LACT+, RX_ERR+, J_RX_BSC+, FL+ |
| Periodic | at 6.66 Mhz: J_TX_CLK+, J_RX_CLK+/- |

PROCESSOR

File: [ALIMSPRO.TAK.ALINK]

The Processor block consists of a relatively straightforward microprocessor architecture, based upon the 80186 16-bit microprocessor. Thus it can be thought of as being built around three major busses: Address, Data and Control. The Address and Data busses, as well as a subset of the Control bus are accessible by the Backplane Adapter through arbitration, although the Backplane Adapter can only access RAM via these busses.

In addition to the three major busses, the Processor block has two auxiliary busses, a Global Control and Global Status bus, which are used to control and sense various card activities.

The major subblocks of the Processor Block are

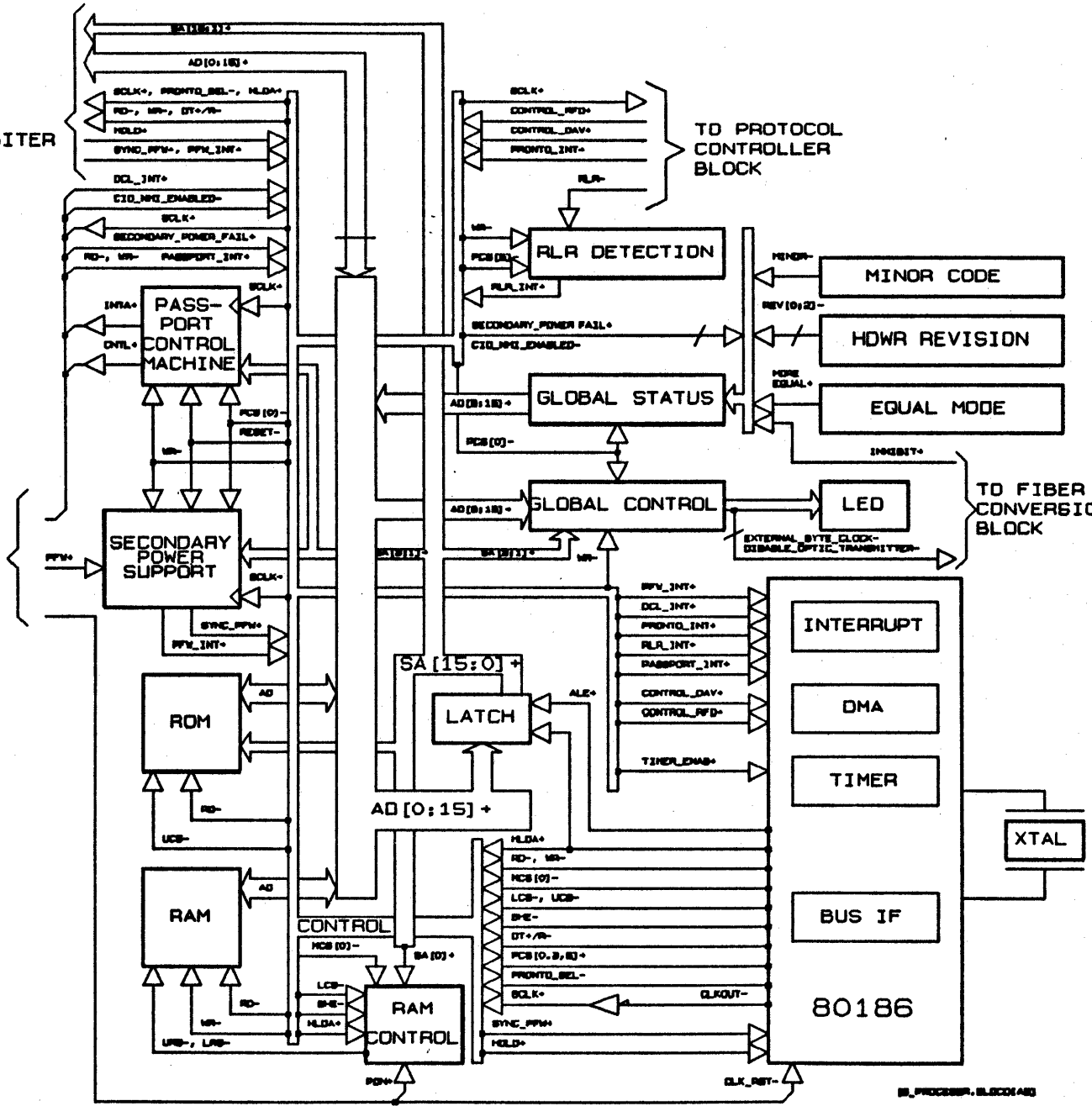
- iapx80186
- Integrated Bus Interface
- RAM Control Logic
- Address Latches
- RAM
- Program Memory
- Global Control and Status Registers
- Integrated DMA
- Integrated Interrupt Controller
- Integrated Timer
- Passport Control Machine
- RLR Detection
- System Clocking
- Reset and Device Clear

TO BACKPLANE ADAPTER BLOCK

TO ARBITER BLOCK

TO PROTOCOL CONTROLLER BLOCK

TO FIBER OPTIC CONVERSION BLOCK



iapx80186

The 80186 {U20} is an 8Mhz 16-bit microprocessor with integrated peripherals. Its major interfaces consists of a multiplexed address-data bus AD[0:15]+ and a control bus. Its integrated peripheral set, discussed in succeeding subsections consists of a timer, interrupt, DMA and bus controller.

NOTE

Certain *CRITICAL* timing parameters require that an 8-Mhz processor be used in the HP27111A application. Upgrading to a faster 80186, while maintaining the same clock frequency, should only be made after a complete review of all timing specifications. In particular, any timing specification of an upgrade part which shows less than a 5 nanosecond delay from the leading edge of CLKOUT- could be cause for concern.

ADDRESS AND DATA.

The Address-Data bus, AD[0:15]+, is the main processor communications bus. It is used to communicate with all external devices. Because the bus is multiplexed, external latching circuits are used to capture and maintain the address (see Address Latches). The bus always represents an address on the first state of any bus cycle, which is indicated by the assertion of ALE+

The Processor block is a memory-mapped architecture, i.e., all devices, internal or external can be accessed at certain predefined memory locations. Decoding of these addresses is handled internally by the Bus Interface unit and externally by the RAM Control.

CONTROL INTERFACE.

The 80186 has a fairly standard set of control signals. The signals can be divided into N general bus classes: Device Control, Arbitration, DMA , Timer, and Interrupt.

Device Control signals are used to access devices on the 80186 bus during read and write operations. They also define a sequence of processor states that can be termed a Bus Cycle.

Arbitration signals are used to negotiate sharing of the major Processor busses by Passport and the 80186.

DMA, or Direct Memory Access signals are used by the 80186 to determine when an external device, in this case, the Protocol Controller (PRONTO) is ready for a Data transfer. These signals are processed by the Integrated DMA controller.

The Timer bus, is used by the 80186 to enable its internal Integrated Timer.

The Interrupt bus is used by the 80186 to determine when an external event has occurred so that it may break normal program flow and respond to the event using its Integrated Interrupt Controller.

The functions of the signals that make up these busses are described in the following table:

80186 Control Signals

| 80186 Pin | Alink Signal | FUNCTION |
|-----------|--------------|---|
| ALE+ | ALE+ | Address Latch Enable |
| WR- | WR- | Write Strobe |
| RD- | RD- | Read Strobe |
| BHE- | BHE- | Bus High Enable is used to determine byte vs. word memory accesses |
| DT/R- | DT/R- | Direction line indicating whether the 80186 will drive or receive data on the current bus cycle |
| HOLD+ | HOLD+ | Indicates Passport is requesting access to shared memory |
| HLDA+ | HLDA+ | Indicates that the 80186 has granted Passport memory access |
| TEST- | SYNC_PFW+ | Allows 80186 to test the state of the synchronized PFW line |

[ali186ct]

System Clock Generation

The system clocks are derived using the 80186's internal clock generating circuitry. A 16 Mhz crystal {Y7} is operated in a fundamental mode parallel resonant fashion inside a tank formed by two 18 pF capacitors {C9, C10}. The resulting 16 Mhz waveform is divided by 2 to form an 8 Mhz output, CLKOUT-. All device pin timings for the 80186 are specified relative to this clock.

CLKOUT-'s cycle is defined by a leading, *falling edge*. The majority of the remaining circuitry on the HP27111A requires a leading, *rising edge*. Therefore the sense of the clock is inverted by an AS02 {U36-6} before being distributed to the rest of the card as SCLK+ {U36-4}. The remaining input of the NOR gate is driven by an ALS240 inverter {U16-4, 16} whose input is pulled high through resistor U35-6. If {U16-4} is pulled low by external test equipment, SCLK+ will be forced into a deasserted state.

NOTE

Any attempt to increase the System Clock frequency will require review of all state machine interfaces on the card. At this printing of the IMS it appears that NMOS-III external timing precludes upgrade to a higher crystal frequency.

Integrated Chip Select

The 80186 contains an internal Chip Select unit. This unit is programmed to decode the internal address bus into a number of select lines and to associate with each set of select lines a particular number of wait states.

The HP27111A makes use of six of these select lines: UCS-, LCS-, MCS[1]-, PCS[0]-, PCS[4]- (PRONTO_SEL-) , and PCS[5]-. PCS[4:5]- are programmed for 2 internal wait states, while the others are set to 0 internal wait states. The external Ready control lines (ARDY+,SRDY+) are sampled only upon reset initialization as defined by the 80186, and are pulled up so that they are always asserted.

For the current card the range of addresses that will cause the select lines to assert are as follows:

Chip Select Ranges

| Signal | ADDRESS RANGE | ADDRESSED DEVICE |
|---------|------------------|----------------------|
| UCS- | 0F8000h-0FFFFFFh | ROM |
| LCS- | 0-7FFFh | RAM |
| MCS[1]- | 18000h-1FFFFh | EXPANSION RAM |
| PCS[0]- | 0F000h-0F07Fh | GLOBAL REGISTERS |
| PCS[3]- | 0F180h-0F1FFh | DEVICE CLEAR MACHINE |
| PCS[4]- | 0F200h-0F27Fh | PRONTO |
| PCS[5]- | 0F280h-0F2FFh | RLR ACKNOWLEDGE |

[ali186cs]

These same select lines address individual devices or functions according to the following table. Also included in the table is the area addressed by the 80186's internal peripherals, even though an external chip select strobe is not generated.

| Address | Peripheral Addressed | Access Mode |
|-----------------|------------------------------|-------------|
| 0000- 3FFF | RAM | Read/Write |
| 4000- 7FFF- | EXPANSION RAM | Read/Write |
| F000 | GLOBAL STATUS | Read |
| F000 | GLOBAL CONTROL | Write |
| F002 | PFW ACKNOWLEDGE | Write |
| F004 | PFW GENERATE | Write |
| F008 | SET CNTL+ | Write |
| F00A | CLEAR CNTL+ | Write |
| F00C | SET INTA+ | Write |
| F00E | CLEAR INTA+ | Write |
| F180 | DCL ACKNOWLEDGE | Write |
| F200 | PRONTO RX FIFO DATA | Read |
| F200 | PRONTO TX FIFO DATA | Write |
| F202 | PRONTO INTERRUPT STATUS | Read |
| F202 | PRONTO INTERRUPT ACKNOWLEDGE | Write |
| F204 | PRONTO INTERRUPT MASK | Read/Write |
| F206 | PRONTO LINK ERROR COUNT | Read |
| F206 | PRONTO DMA CONTROL | Write |
| F208 | PRONTO STATUS | Read |
| F208 | PRONTO CONFIGURATION | Write |
| F20A | PRONTO RX HEADER SIZE | Write |
| F20C | PRONTO VIRTUAL CIRCUIT MATCH | Write |
| F20E | PRONTO TX HEADER SIZE | Write |
| F280 | RLR ACKNOWLEDGE | Write |
| FF00- FFFF | 80186 INTEGRATED PERIPHERALS | Read/Write |
| 18000- 1FFFF | EXPANSION RAM | Read/Write |
| F0000- F7FFF | EXPANSION ROM | Read |
| F8000- FFFFF | ROM | Read |

[alimsrg2]

PROGRAMMING NOTES.**RAM Control Logic**

The RAM Control Logic subblock uses control signals from the 80186 to generate the RAM select strobes. It also monitors the state of primary power and is used to maintain RAM state during failure of primary power.

Theory of Operation

RAM SELECTS.

The actual RAM selects (UPPER_RAM_SEL- and LOWER_RAM_SEL-) are generated inside U17 using the PAL16R4's combinational outputs. Now, the 80186 is capable of either byte or word addressing according to the following table:

| RAM Select Decode Table | | | | | |
|-------------------------|---------|------|--------|------|------|
| LCS- | MCS[1]- | BHE- | SA[0]+ | URS- | LRS- |
| 0 | 0 | X | X | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 |

[ali186bw].

Note that in addition to the LCS- line, the MCS[1]- lines may be used to access RAM. This is to facilitate the use of Expansion RAM in future revisions of the HP27111A.

PRIMARY POWER MONITORING.

The RAM Control PAL also monitors the state of the Primary Power On indicator from the CIO backplane (PON+). As long as this signal is asserted, the RAM Select decode occurs as defined in the previous table. If PON+ deasserts, however, and Secondary Power is still present, the select will lines will deassert until PON+ reasserts.

| RAM Select Decode Table with Power Monitor | | | | | | |
|--|------|---------|------|--------|------|------|
| PON+ | LCS- | MCS[1]- | BHE- | SA[0]+ | URS- | LRS- |
| 0 | X | X | X | X | 0 | 0 |
| 1 | 0 | 0 | X | X | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 |

[ali186po]

BYTE VS. WORD.

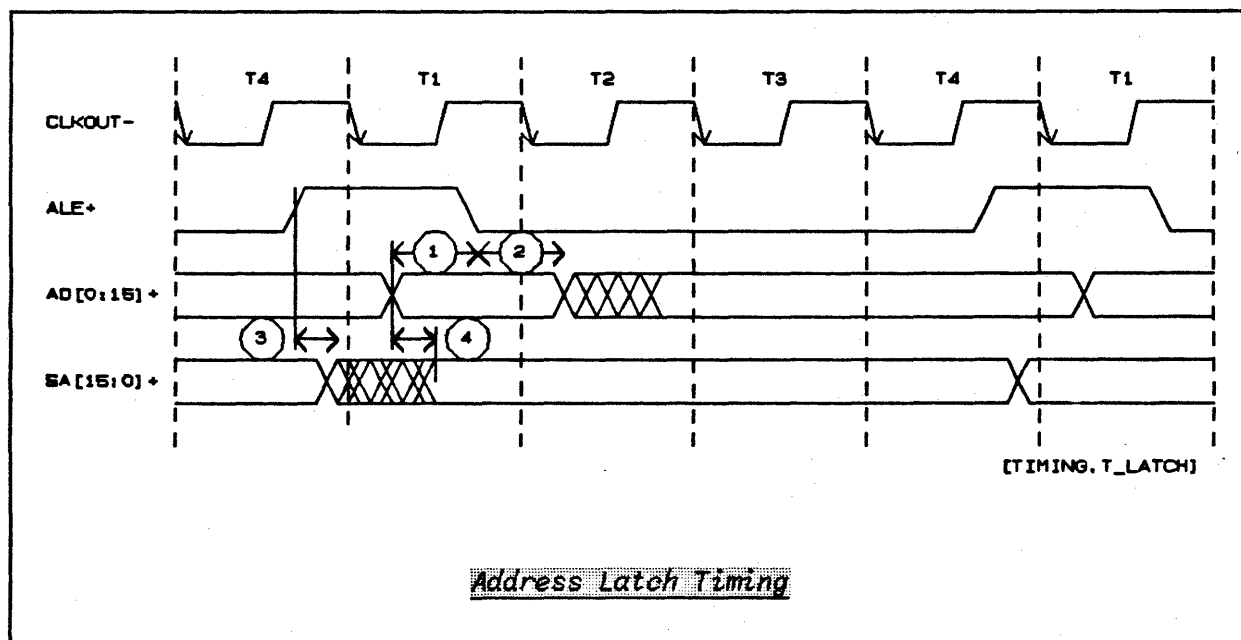
RAM may be accessed by the 80186 in either Byte or Word units. When a Word is accessed whose address falls on an even address, both of the RAM selects are asserted.

When a Byte is accessed, or a Word that falls on an odd address boundary, only one of the RAM selects will be asserted at a time. Byte accesses to *even* addresses select the Upper Ram. Byte accesses to *odd* addresses select the Lower Ram. SA[0]+ is asserted during Odd Byte accesses.

Address Latches

The Address Latch subblock latches the contents of the AD[0:15]+ bus during the 80186's address cycle and generates the SA[15:0]+ address bus. The subblock is implemented using two ALS573 latches {U27,U38}. The latch control input is connected to the ALE+ line. Therefore new data is loaded into the latch whenever ALE+ is asserted and held upon ALE+ deasserting.

The tri-state control signal is connected to HLDA+. Thus, the latches will only drive the bus when HLDA+ is deasserted, i.e., when the 80186 has not granted the bus to Passport.

TIMING.

Address Latch Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|---------------------|--|-----|-----|
| 1 | t _{AD_ALE} | setup of AD[+] to falling edge of ALE+ | 10 | ... |
| 2 | t _{ALE_AD} | hold of SA[+] from falling edge of ALE+ | 7 | ... |
| 3 | t _{ALE_SA} | hold of SA[+] from assertion (rising edge) of ALE+ | 8 | 20 |
| 4 | t _{AD_SA} | delay from AD[+] to SA[+] | 0 | 14 |

Program Memory

Program Memory on the card consists of 2 16Kx8 250 nanosecond ROMs or, optionally PROMs {U30,U29}, providing 32K bytes of code space. The card also supports 8Kx8 and 32Kx8 sizes for future or OEM versions. U30 drives the upper byte AD[0:7]+, while U29 drives the lower byte AD[8:15].

Program Memory is accessed when UCS- asserts and actively drives the data bus when RD- asserts.

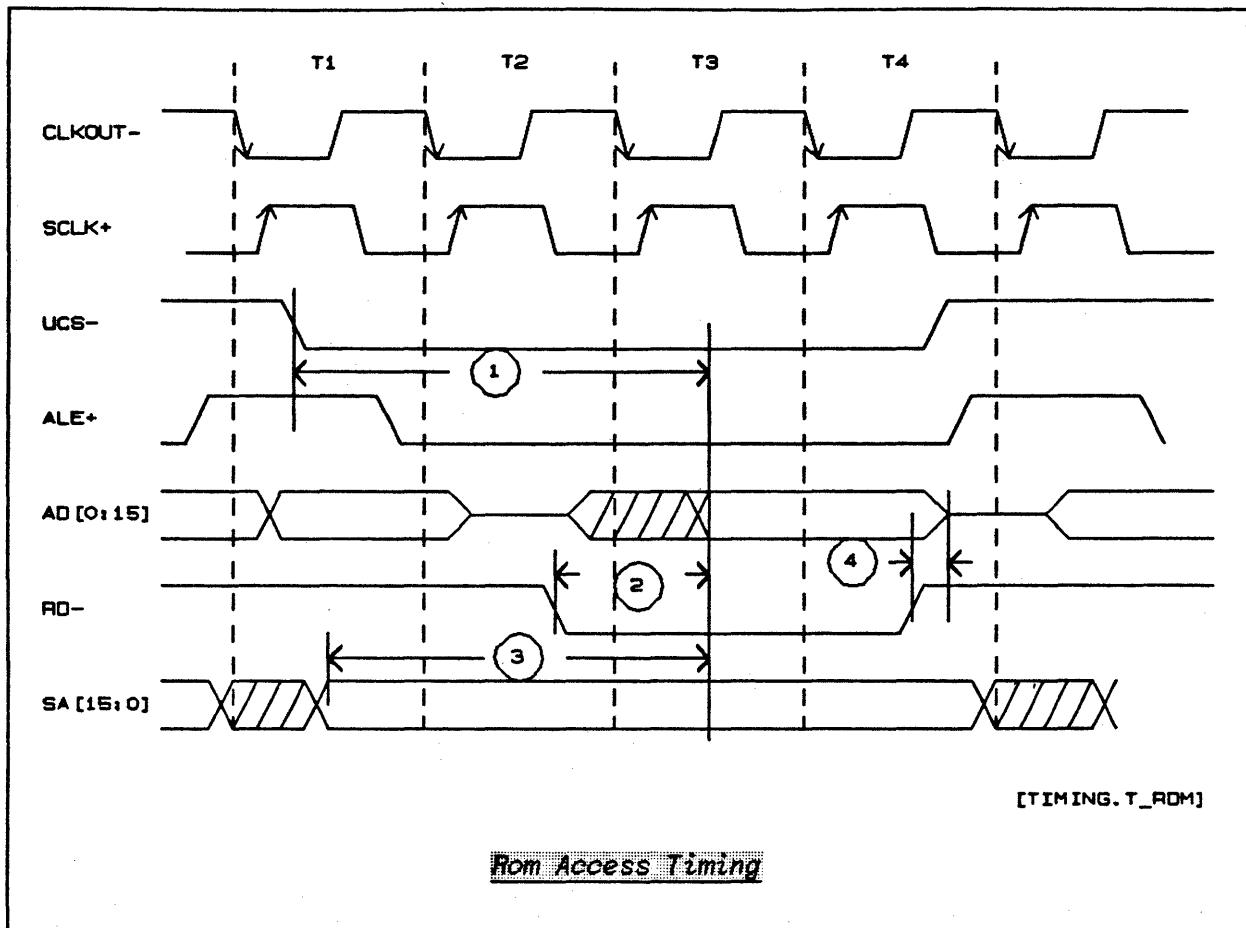
Note that neither SA[0]+ nor BHE- are used to determine which ROM is selected. This is because the 80186 effectively does word accesses on all instruction fetches and discards an extra byte if it is not needed.

EXPANSION ROM.

- Program Memory may be changed by replacing the 16K x 8 memories with either 8K or 32K memories. Using the 8K memories is a simple switch since the necessary address line, SA[14]+, is not used.

To expand to 32K memory however, requires that a jumper or zero-ohm resistor be inserted between U48-1 and U48-8. Inserting this jumper connects the SA[15]+ address bit to the most significant address bit of the 32K memory.

TIMING.



from Access Timing

RAM Read Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|---------------|---|-----|-----|
| 1 | t_{UCS_AD} | delay from assertion of UCS- to AD[+] valid | .. | 289 |
| 2 | t_{RD_AD} | delay from assertion of RD- to AD[+] valid | .. | 160 |
| 3 | t_{SA_AD} | delay from valid SA[+] to AD[+] valid | .. | 284 |
| 4 | t_{AD_T4} | setup of AD[+] to leading edge <T4> CLKOUT- | 20 | ... |
| 5 | t_{T4_AD} | hold of AD[+] to leading edge <T4> CLKOUT- | 10 | ... |
| 6 | t_{AD_RD} | hold of AD[+] from deassertion of RD- | 0 | .. |

RAM

RAM consists of two 8K x 8 150 nanosecond static RAMs {U49,U50}, expandable to 32K x 8 static RAMs.

Each RAM is individually accessed by its own select line. UPPER_RAM_SEL- when asserted selects the most significant byte AD[0:7]+ or {U50}. LOWER_RAM_SEL- when asserted selects the least significant byte, AD[8:15] or {U49}. When 8K RAMs are used the additional select line {U49,U50-26} is pulled high to an asserted state through U37-15.

Because only 16Kbytes out of a possible 64Kbytes are present, there is replication of physical memory such that multiple addresses identify the same location.

The various addresses for any RAM memory location can be calculated as

$$\text{BASE_ADDRESS} + \text{I} * 4000\text{H} + \text{J} * 18000\text{H}$$

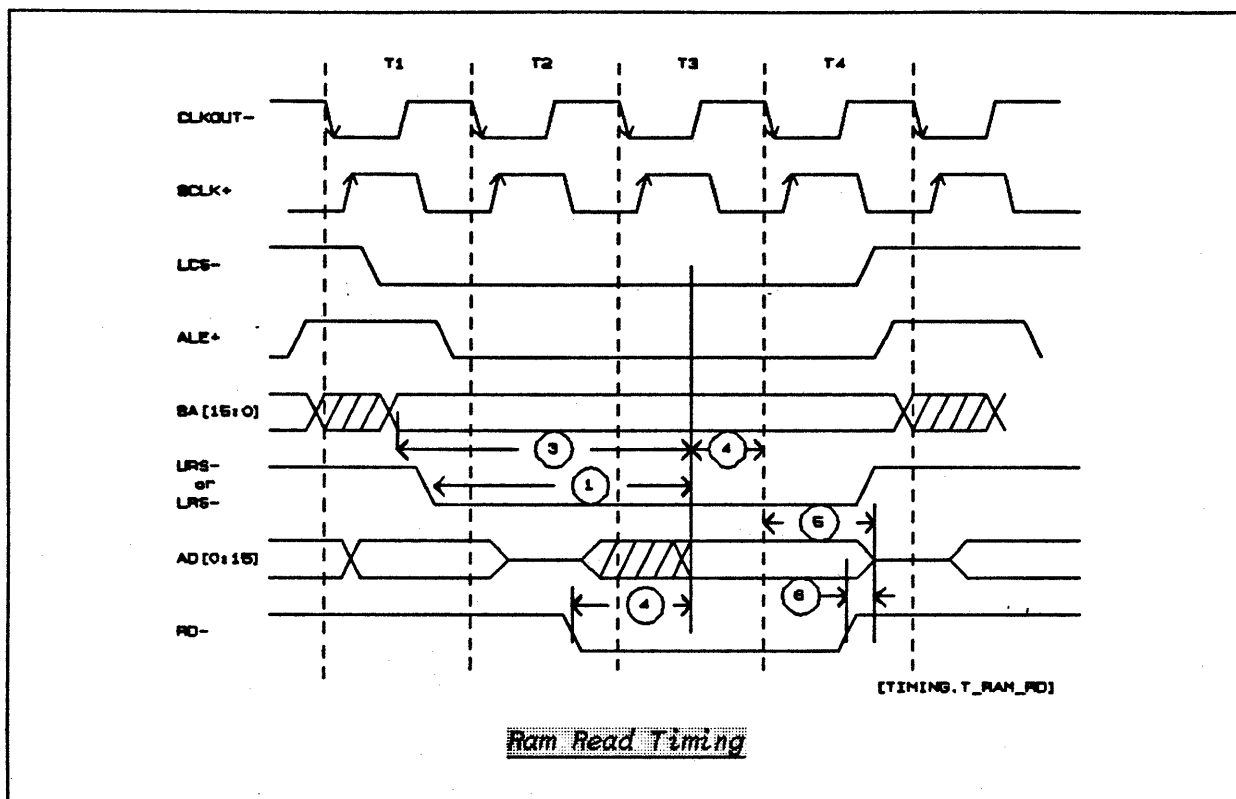
where, I and J range from 0..1 .

EXPANSION RAM.

RAM may be expanded by replacing the 8K memories by 32K memories, and connecting an additional address line by jumpering U48-2 to U48-7. This allows SA[14]+ to drive one of the additional RAM address lines located at U49, U50-26.

Note, however, the card architecture is such that the 64K bytes must be broken into two separate 32K byte spaces. The first 32 Kbytes is accessed from SA[15:0]+ = 0-7FFFH and the second 32 Kbytes is accessed from SA[15:0]+ = 18000H-1FFFFH.

TIMING - MEMORY READ.

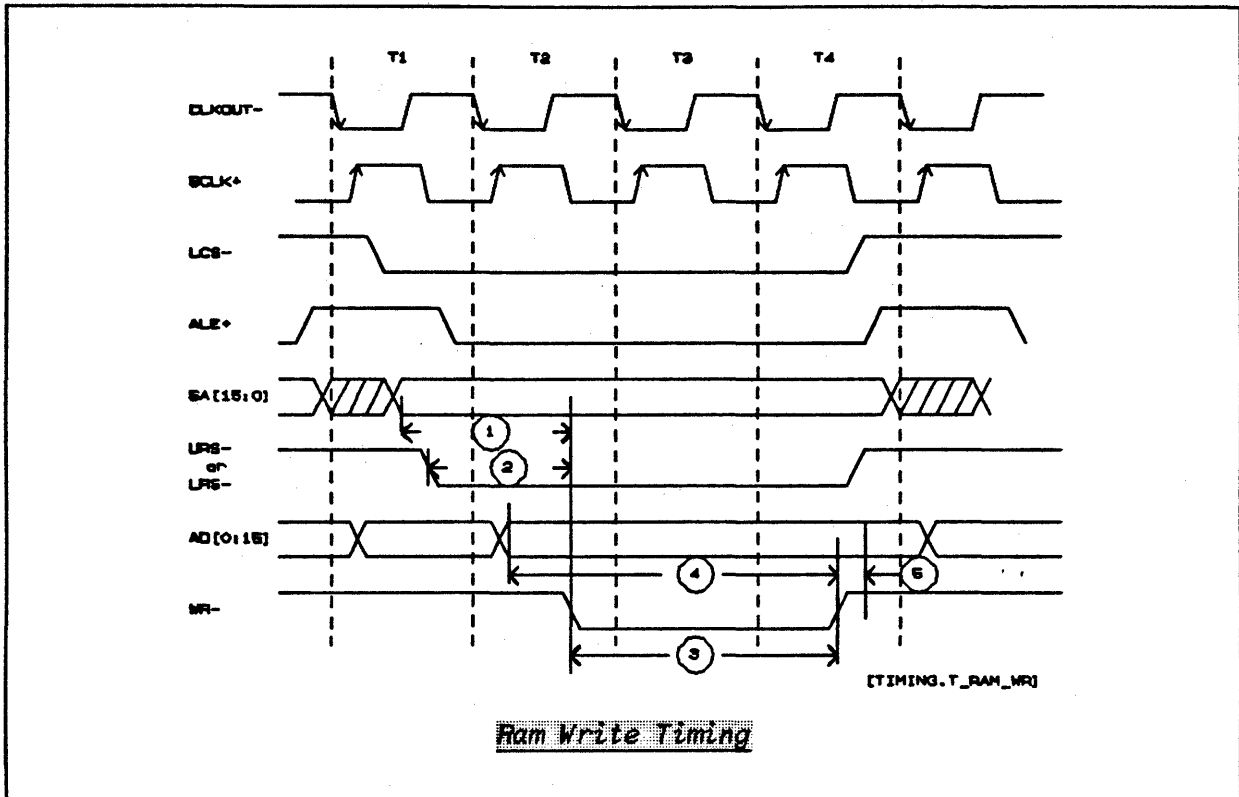


RAM Read Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|--------------|---|-----|-----|
| 1 | t_{RS_AD} | delay from assertion of either URS- or LRS- to AD[0:15] | .. | 249 |
| 2 | t_{RD_AD} | delay from assertion of RD- to AD[0:15] valid | .. | 160 |
| 3 | t_{SA_AD} | delay from valid SA[15:0] to AD[0:15] valid | .. | 284 |
| 4 | t_{AD_T4} | setup of AD[0:15] to leading edge <T4> CLKOUT- | 20 | ... |
| 5 | t_{T4_AD} | hold of AD[0:15] to leading edge <T4> CLKOUT- | 10 | ... |
| 6 | t_{AD_RD} | hold of AD[0:15] from deassertion of RD- | 0 | .. |

Theory of Operation

TIMING - MEMORY WRITE.



Ram Write Timing

Write Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|--------------------|---|-----|-----|
| 1 | t _{SA_WR} | setup from valid SA[15:0] to WR- asserted | 64 | ... |
| 2 | t _{RS_WR} | setup from assertion of either URS- or LRS- to assertion of WR- | 19 | .. |
| 3 | t _{SA_AD} | pulse width of WR- | 210 | ... |
| 4 | t _{AD_RD} | setup of AD[15:0] to deasserting (clocking) edge of WR- | 255 | .. |
| 5 | t _{WR_AD} | hold of AD[15:0] from deasserting (clocking) edge of WR- | 0 | .. |

Global Status Register

The Global Status Register is used by the Processor to detect certain card state and configuration information. The register *U18*, an ALS541, actively drives the lower byte of the data bus AD[8:15]+ when RD- and PCSO- are asserted, satisfying the output enable "AND" condition { *U18-7 and U18-19* }.

FORMAT.

The format of the status register is as follows:

| | | | | | | | | |
|-----------|---------|------|------|------|---------|---------|------|------|
| U18 pin : | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| AD[] : | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Mnemonic: | REV[2]- | INH+ | NME- | EQM- | REV[1]- | REV[0]- | SPL+ | MIN- |

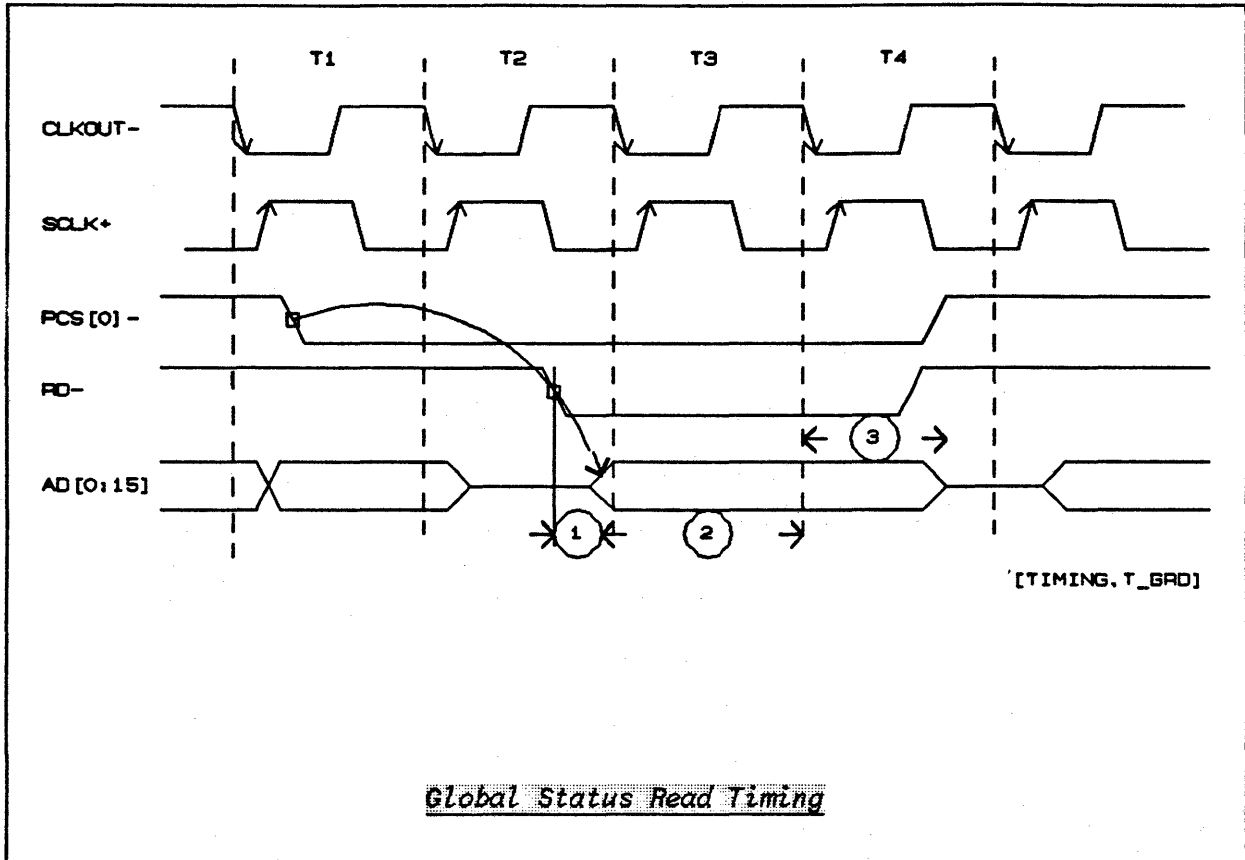
[ali186gs]

Global Status Table

| MNEUMONIC | SIGNAL NAME | STATE | MEANING |
|-----------|-----------------------|-------|---|
| MIN- | MINOR- | T | Display both Minor and Major error codes upon self test failure |
| | | F | Display only Major error code upon self test failure |
| SPF+ | SECONDARY_POWER_LOSS+ | T | Secondary Power was lost. Valid for #usec after card reset. |
| | | F | Secondary Power was present. Valid for #msec after card reset. |
| REV[0:2]- | REV[0:2]- | 0-7 | Indicates current PCA revision number, incrementing upon revision. |
| EQM- | EQUAL_MODE- | T | EQUAL MODE Jumper is in the <i>MORE EQUAL</i> position. |
| | | F | EQUAL MODE Jumper is in the <i>LESS EQUAL</i> position. |
| NME- | CIO_NMI_ENABLED- | T | NMI jumper is in position to allow card to generate and NMI to CIO. |
| | | F | Card cannot assert the NMI- line on the CIO backplane. |
| INH+ | INHIBIT+ | T | Optical Transmitter is disabled due to either Jupiter Loopback Mode or setting of DIS- bit in the Global Control Register |
| | | F | Optical Transmitter is enabled. |

[align=right]

TIMING.



Global Status Read Timing

Global Read Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|----------------|--|-----|-----|
| 1 | t_{GRD_AD} | delay from asserted PCS0-, RD- to valid AD[+] | 0 | 25 |
| 2 | t_{AD_SCLK} | setup of valid AD[+] to rising edge of <T4> SCLK+ | 20 | ... |
| 3 | t_{SCLK_AD} | hold of valid AD[+] from rising edge of <T4> SCLK+ | 40 | ... |

Global Control Register

The Global Control Register {U26} is used to change the state of the LEDs and control modes of operation of the Optical Transmitter Section. The register is implemented using an ALS273. The clear control input {U26-7} is connected to RESET-, thus any reset places all outputs in the electrical low state. Since all of the signals driven by the register are negative true in sense, all outputs of the Global Control Register will be true after a logic reset.

The Global Control Register is updated on the rising edge of GLOBAL_CONTROL_WRITE- through the clock input {U26-11}. This negative true pulse is generated by {U23-19} when the following conditions are true: PCS[0]-, WR-, SA[3:1]=0.

FORMAT.

| | | | | | | | | |
|------------|-----|------|----|----|----|----|----|----|
| U28 pin : | 19 | 16 | 15 | 12 | 9 | 6 | 5 | 2 |
| AD[] bit : | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Mnemonic: | res | DOT- | P- | A- | C- | S- | R- | F- |

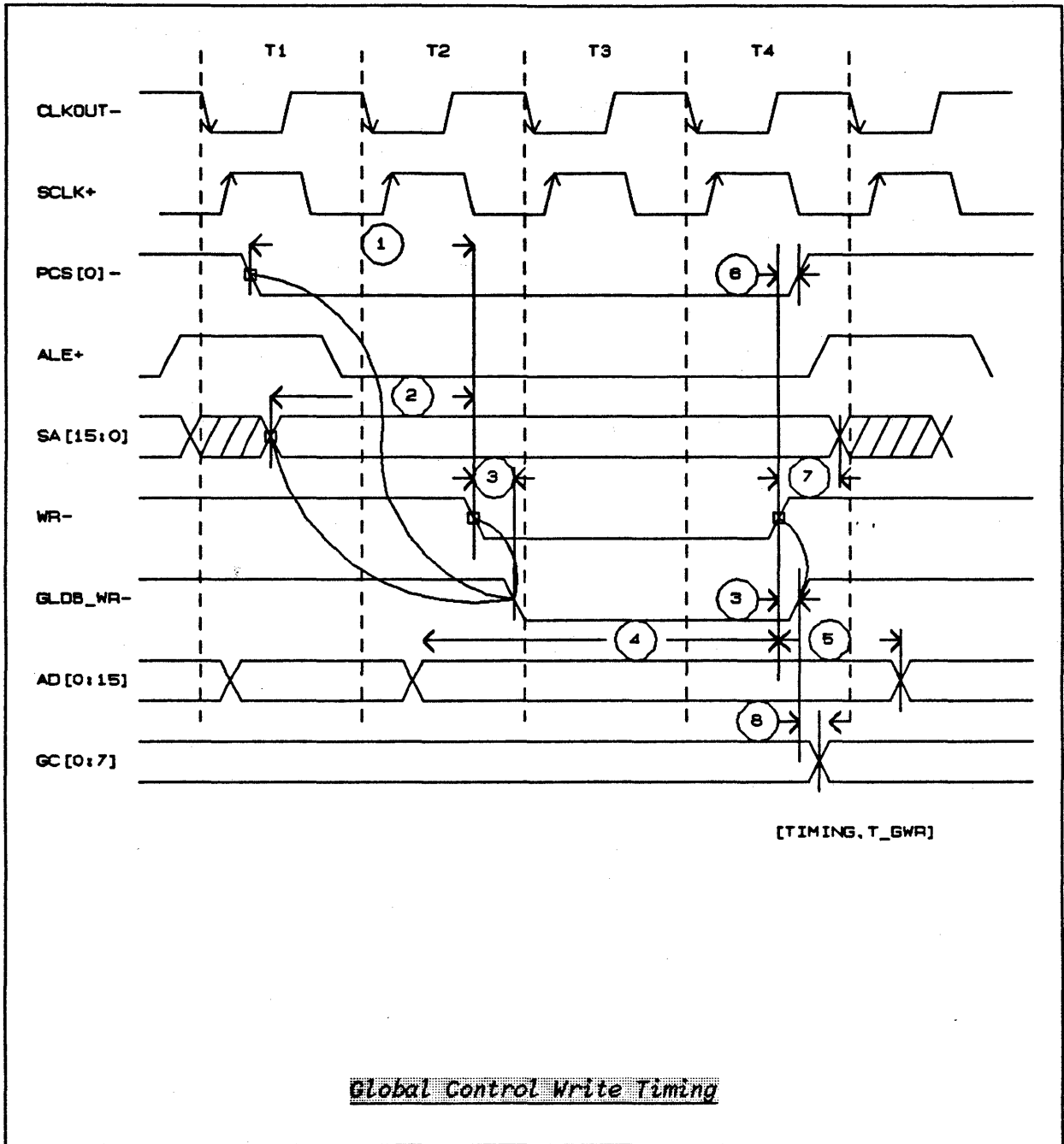
[ali186gc]

Global Control Table

| MNEUMONIC | SIGNAL NAME | STATE | FUNCTION |
|-----------|----------------------------|-------|-----------------------------|
| res | RESERVED | | Reserved for future use |
| DOT- | DISABLE OPTIC TRANSMITTER- | T | Disable Optical Transmitter |
| | | F | Allow Optical Transmission |
| P- | PASSED_SELF_TEST- | T | Turn [P] led ON |
| | | F | Turn [P] led OFF |
| A- | ACTIVITY- | T | turn [A] led ON |
| | | F | turn [A] led OFF |
| C- | CONFIGURATION ERROR- | T | turn [C] led ON |
| | | F | turn [C] led OFF |
| S- | SIGNAL_ERROR- | T | turn [S] led ON |
| | | F | turn [S] led OFF |
| R- | REMOTE_ERROR- | T | turn [R] led ON |
| | | F | turn [R] led OFF |
| F- | SELF_TEST_FAILED- | T | turn [F] led ON |
| | | F | turn [F] led OFF |

[align=right]

TIMING.



Global WRite Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|----------|--|-----|-----|
| 1 | tPCSO_WR | setup of PCSO- to asserted WR- | 35 | ... |
| 2 | tSA_WR | setup of valid SA[]+ to asserted WR- | 35 | ... |
| 3 | tWR_GWR | delay from WR- to valid GLOBAL_CONTROL_WRITE- | 0 | 35 |
| 4 | tAD_WR | setup of AD[]+ to clocking edge of WR- | 10 | ... |
| 5 | tWR_AD | hold of AD[]+ from clocking edge of WR- | 35 | ... |
| 6 | tWR_PCS0 | hold of PCSO- from clocking edge of WR- | 0 | ... |
| 7 | tWR_SA | hold of SA[]+ from clocking edge of WR- | 0 | ... |
| 8 | tGWR_GC | delay from clocking edge of GLOBAL_CONTROL_WRITE- to valid output of Global Control Register U23 | 3 | 15 |

Integrated DMA

The 80186's Integrated DMA unit is used to provide a high-speed data path between the Protocol Controller and the Processor that may operate in parallel with normal processor activity. The DMA unit provides two channels for transfer operations, channels 0 and 1, which are used for read and write DMA operations respectively. Once a particular DMA channel has been activated programatically, it begins to monitor the appropriate DMA request line. When a request appears, the DMA unit acquires control of the 80186's external busses, and performs the transfer operation.

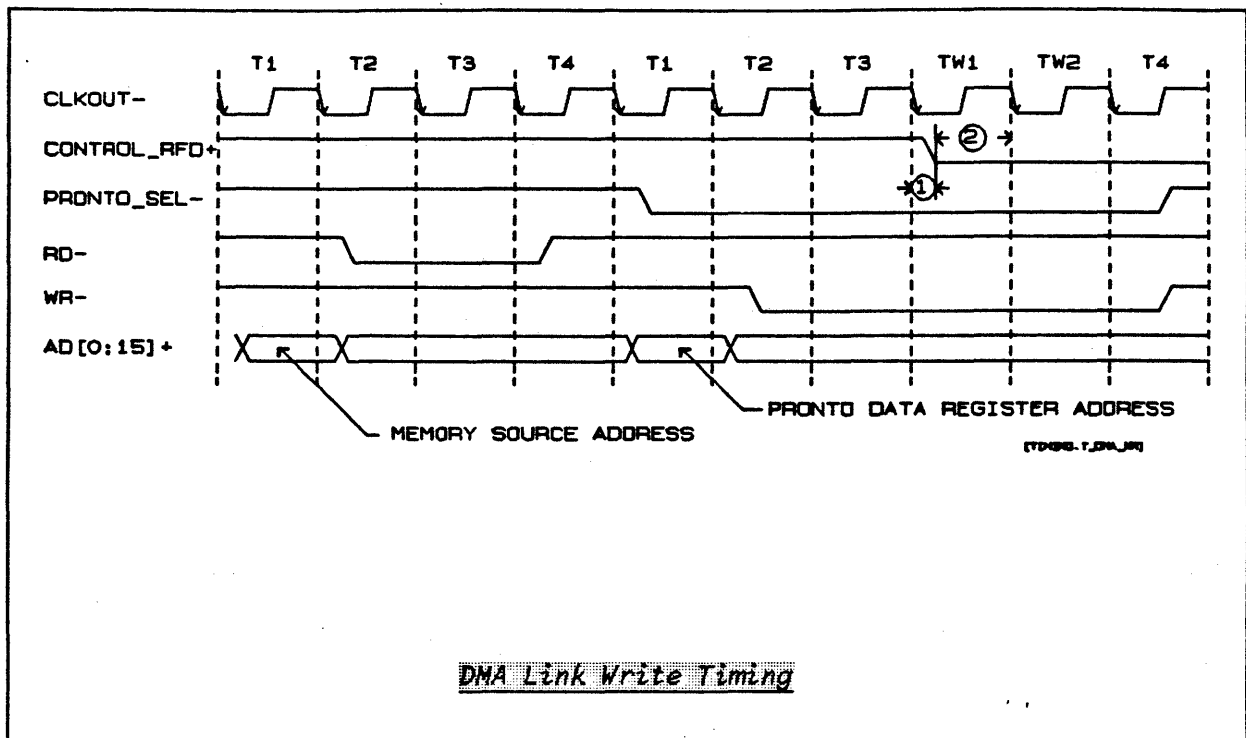
The 80186 treats all DMA operations *atomically*. Data read from the source is temporarily stored in an internal holding register during the "fetch" portion of the DMA cycle. The second "deposit" cycle is run immediately afterwards. Only then can the 80186 perform an instruction fetch, operand fetch, operand store, or grant control of its busses to the Backplane Adapter via the Arbiter.

PROGRAMMING NOTES.

The 80186 must be programmed for *Destination Synchronized* on DMA Channel 1 for communication with the Protocol Controller (PRONTO). DMA Channel 0 must be programmed for *Source Synchronized* transfers when communicating with PRONTO.

DMA LINK WRITE.

DMA Write's to the Link are performed when the DMA unit detects that the CONTROL_RFD+ {U20-19} line has asserted and DMA channel 1 is currently enabled. Data is fetched from memory at a location pointed to by channel 1's source address pointer and then written out to the Protocol Controller on the following bus cycle. The DMA unit samples the CONTROL_RFD+ line again during the write to see whether the Protocol Controller wishes another word of data. If CONTROL_RFD+ is false, the unit will wait until CONTROL_RFD+ asserts again and proceed with additional transfers. This process continues until the DMA unit exhausts its transfer count or the firmware stops the transfer in progress.



DMA Link Write Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|------------------|---|-----|-----|
| 1 | t_{CLKO_CDAV} | hold of CONTROL_DAV+ from falling edge of CLKOUT- | 0 | 100 |
| 2 | t_{CDAV_CLKO} | setup of CONTROL_DAV+ to falling edge of CLKOUT- | 25 | ... |

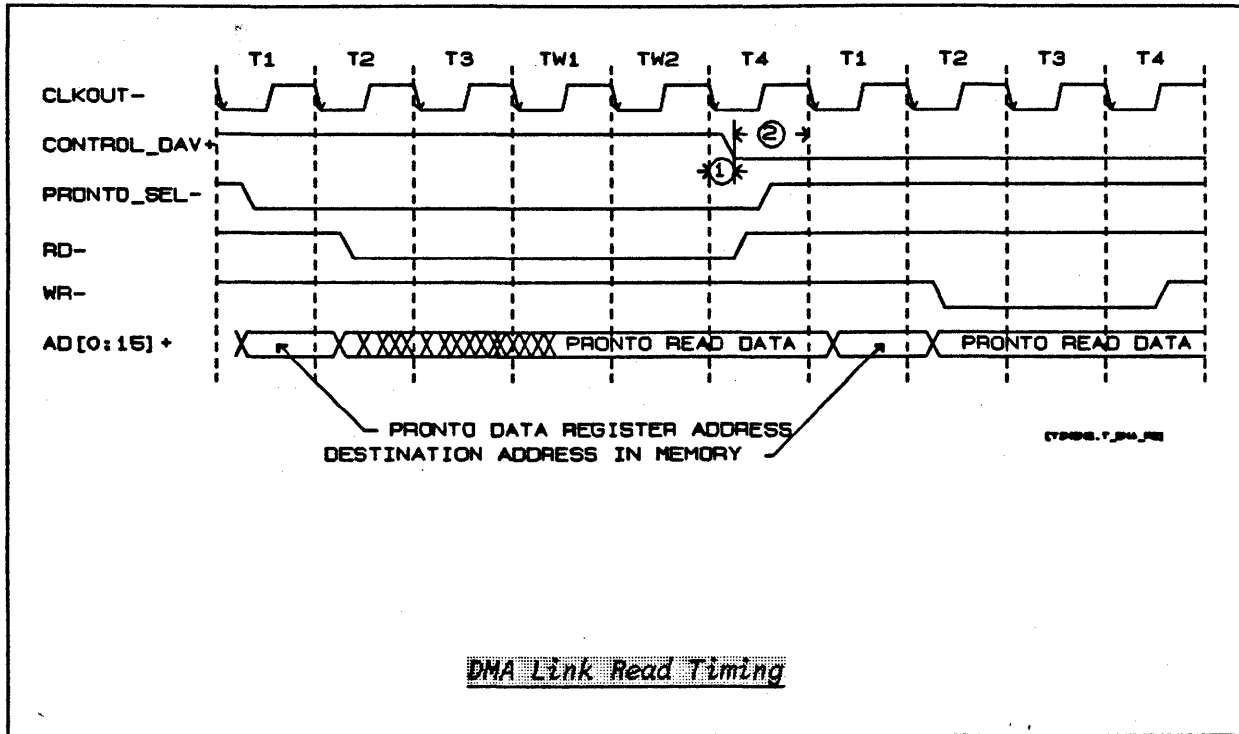
With Protocol Controller accesses programmed for two wait states, the 80186 will start sampling CONTROL_DRFD+ on the leading edge of state <TW2> of the "deposit" cycle to determine whether consecutive DMA Write cycles should be run.

If CONTROL_DRFD+ is not asserted on the leading edge of <TW2>, the processor will run a normal bus cycle, if one has been pending, or possibly transfer bus control to the Backplane Adapter if the Arbiter request met a desired setup time.

DMA LINK READ.

DMA Link Read's operate similarly to Link Write's except that the transfer is from the Protocol Controller to RAM. Once Channel 0 has been programmed and enabled, it samples CONTROL_DAV+ {U20-18} until it detects an assertion. The DMA Unit then acquires the 80186's external bus and reads a word of data from the Protocol Controller. On the following bus cycle, the data word is written to a location in memory determined by the contents of the destination address register.

Theory of Operation



DMA Link Read Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|-------------------------|---|-----|-----|
| 1 | $t_{\text{CLKO_CRFD}}$ | hold of CONTROL_RFD+ from falling edge of CLKOUT- | 0 | 100 |
| 2 | $t_{\text{CRFD_CLKO}}$ | setup of CONTROL_RFD+ to falling edge of CLKOUT- | 25 | ... |

The 80186 samples CONTROL_DAV+ on the leading edge of <T1> of the deposit cycle. This is true regardless of the number of wait states PRONTO may introduce in the data fetch portion of the DMA cycle.

Integrated Interrupt Controller

The Integrated Interrupt Controller is used by the 80186 to monitor and respond to transitions on the Interrupt bus. The controller processes both internal and external interrupt requests. There are five external interrupts processed by the interrupt controller (PASSPORT_INT+, RLR_INT+, PRONTO_INT+, PFW_INT+ and DCL_INT+) and five internal interrupts (TIMER[0:2], DMA_Channel[0:1]). All of these interrupts are maskable (i.e., ignorable) with the exception of DCL_INT+.

| 80186 Pin | Alink Signal Name | Vector Address |
|-------------------------------|-------------------|----------------|
| NMI | DCL_INT+ | 8H |
| INT0 | PASSPORT_INT+ | 48H |
| INT1 | PRONTO_INT+ | 52H |
| INT2 | RLR_INT+ | 56H |
| INT3 | PFW_INT+ | 60H |
| Internal Peripheral Interrupt | | Vector Address |
| Timer[0] | | 32H |
| Timer[1] | | 72H |
| Timer[2] | | 76H |
| DMA Channel[0] | | 40H |
| DMA Channel[1] | | 44H |

[ali186in]

Integrated Interrupt Controller Table

INTEGRATED REGISTER SET.

The Integrated Interrupt Controller has a number of internal registers which may be accessed by the 80186. A complete listing of the registers and their functions can be obtained by referring to the 80186 Programmer's Manual.

GENERAL INTERRUPT PROCESSING.

All interrupts are handled by the Interrupt Controller in the same basic manner. Once an interrupt condition occurs (internal or external), if the interrupt is unmasked and interrupt processing has been enabled by firmware then the Interrupt Controller gains control of the processor and begins to respond.

First, the current program counter, flags, and code segment register are stored in RAM. Second, a new program counter and code segment value are fetched from RAM. Each interrupt has a unique RAM storage location which is referred to as its "interrupt vector location". Once the new values have been fetched, program control transfers to the service routine originating at the address indicated by the interrupt vector contents fetched from RAM.

Normally, at this point, firmware responds in whatever way it deems necessary, acknowledges the interrupt condition, and then clears the state of the interrupt controller by writing to the EOI register. Finally, An IRET instruction is executed, and the original program counter, flags and code segment are restored and the program continues from the point at which it was interrupted.

PROGRAMMING NOTES.

All of the interrupts used on the HP27111A should be programmed for

Cascade : FALSE

Nested : FALSE

Level : TRUE

The current state of any of the interrupts can be determined by examining the Integrated Interrupt Request Register at internal address 0FF2Eh.

PASSPORT INTERRUPT.

A Passport Interrupt is generated whenever Passport has been told to interrupt upon completion of certain tasks, receives unexpected information from the CIO channel, or when it detects an abnormal state such as a parity error.

Externally, **PASSPORT__INT+** is connected to {U20-45} and handled internally by Interrupt Register Set[0]. It should be programmed for Level Triggered mode. Writing to the Passport__IntAck Register will cause **PASSPORT__INT+** to deassert.

PRONTO INTERRUPT.

A Pronto Interrupt is generated whenever Pronto has detected a particular condition and firmware has allowed that condition to assert **PRONTO__INT+**.

Externally, **PRONTO__INT+** is connected to {U20-44} and is handled internally by Interrupt Register Set[1]. It should be programmed for Level Mode [LTM]. Because this line is the logical-OR of many interrupt conditions, the PRONTO ERS Manual should be consulted as to what sequence and combination of Pronto register accesses will cause **PRONTO__INT+** to deassert.

In general, writing 0000H to the Pronto Mask Register should always cause the external signal to deassert, but will not acknowledge the interrupt condition.

RLR INTERRUPT.

The RLR or Remote Link Reset is asserted by the RLR Detect subblock when it detects the Protocol Controller's assertion of the RLR- pin. It indicates that the Remote Device is attempting to Clear the state of the local HP27111A. Externally **RLR__INT+** is connected to {U20-42} and handled internally by Interrupt Register Set[2]. The external line may be deasserted by writing to the Clear RLR Register (see RLR Detect).

Note, that reception of an RLR is also flagged inside the Protocol Controller's Interrupt Status register.

PFW INTERRUPT.

A Power Fail Warning Interrupt is generated whenever the Power Fail Warning machine detects the repeated assertion of the CIO channel's PFW- line.

Externally, PFW__INT+ is connected to {U20-41} and internally handled by Interrupt Register Set[3]. It cannot be explicitly cleared by firmware; only the Power Fail Warning Machine can deassert it.

PFW__INT+ can be asserted manually, for testing purposes by writing to the PFW Generate register (see subsection on Secondary Power Support dealing with Power Fail Warning).

DCL INTERRUPT.

The Device Clear Interrupt is asserted whenever the Device Clear Machine determines that CIO is performing an Addressed Device Clear (DCL). DCL__INT+ is connected to the 80186's NMI or Non-Maskable-Interrupt line {U20-46}.

A non-maskable status is given to this interrupt because a Device Clear state in itself *DOES NOT* reset the 80186. Rather, the 80186 performs a "soft" reset by programatically clearing its internal state registers and after the Device Clear is complete, performs an extensive self-test on the rest of the HP27111A.

When DCL__INT+ asserts, the 80186 detects this state change and will process the interrupt on the next available bus cycle. This interrupt should force the 80186 to execute its Device Clear Interrupt Handler (see programming notes). Since DCL__INT+ is a pulse, there is no need for the 80186 to externally acknowledge this interrupt.

Integrated Timer

The 80186 Integrated Timer consists of three multi-purpose timer-counter modules [0:2]. It serves three purposes on the HP27111A: Real-Time Clock, Selftest Timer, and Device Clear Handling.

REAL-TIME CLOCK.

Firmware makes use of Timer[2] as a real-time clock in order to perform its link statistic calculations. In this mode of operation, Timer Module[2] runs independently of the other two timer modules.

SELFTEST TIMER.

During selftest, Timers[0,2] are programmed to allow certain of the selftest routines to timeout if they are wait to long for a certain event and return to the main selftest program.

DEVICE CLEAR WAIT TIMER.

The Device Clear Wait Timer uses Timer[0] as an event detector while firmware is waiting for the Device Clear state to complete. The DEVICE__CLEAR__WAIT- line {U17-17} is connected to the Timer Channel[0] input pin {U20-20} on the 80186. The Timer Channel[0] Input pin on the 80186 acts as a gate for Timer[0]. The timer can count only when the pin is at an electrical High level. Therefore,

Theory of Operation

Timer[0] cannot increment until the `DEVICE__CLEAR__WAIT-` line is deasserted. Therefore, the 80186 can determine whether the Device Clear state has completed by observing that Timer[0] advances.

PROGRAMMING NOTES.

Passport Control

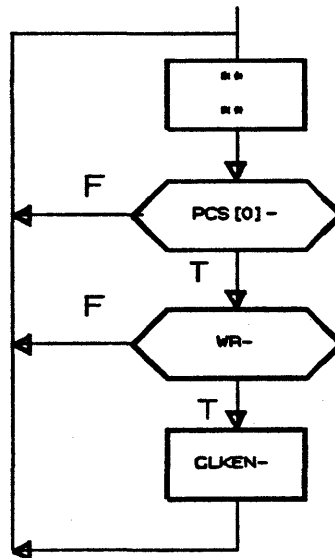
The Processor Block controls PASSPORT by passing data structures back and forth in RAM using handshake lines to indicate the presence of valid data.

The Passport Control Machine maintains the states of the `CNTL+` and `INTA+` signals to Passport. Each of these signals appears as a pair of addressed registers to the processor, one register to set the state, and the other to clear it. Signal state is only updated during register writes; register reads have no effect.

The hardware that implements this machine is a portion of a 16R6 Pal located at `U23`. There are actually three small machines present here, one machine each for the `CNTL+` and `INTA+` lines, and a third "Clock Enable" machine which determines when the other machines may sample the address lines to see whether a state change is necessary.

CLOCK ENABLE.

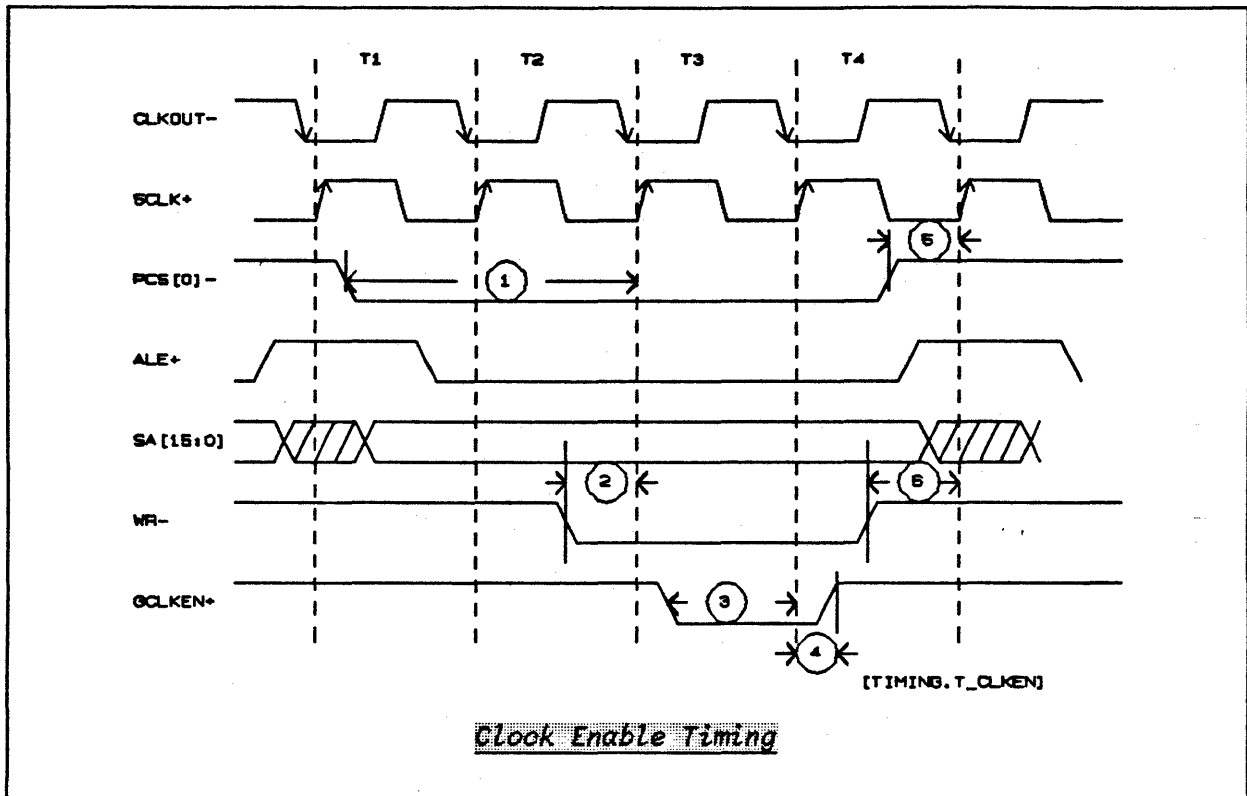
As shown in the accompanying flow diagram, the "clock enable" state machine remains in an idle state until the processor writes to the Global Register area. The `CLKEN+` {`U23-16`} signal then asserts in the state corresponding to Processor bus state `<T3>`. `CLKEN+` will always deassert during bus state `<T4>`, returning to an idle state, and begins sampling again at the leading edge of `<T1>`.



[diagrams.clken]

The reason for this special clocking scheme is due to the timing behaviour of the 80186 `WR-` line. `WR-`

can deassert a minimum of 5 nsec after the leading edge of CLKOUT- when entering bus state T4. Because the delay from CLKOUT- to SCLK+ is on the order of 4.5 nsec, not accounting for printed circuit delay effects, attempting to sample WR- with SCLK+ when entering T4 may setup a metastable condition. The state machine, as designed, will never sample WR- during this window.



Global Control Clock Enable Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|------------------|--|-----|-----|
| 1 | t_{PCS0_T3} | setup of PCS0- to leading edge of <T3> SCLK+ | 25 | ... |
| 2 | t_{WR_T3} | setup of WR- to leading edge of <T3> SCLK+ | 25 | ... |
| 3 | t_{GCKE_SCLK} | setup of GCLKEN- to leading edge of SCLK+ | 100 | ... |
| 4 | t_{SCLK_GCKE} | hold of GCLKEN- from leading edge of SCLK+ | 0 | 25 |
| 5 | t_{PCS0_T1} | setup of deasserting PCS0- to leading edge of <T1> SCLK+ | 25 | ... |
| 6 | t_{WR_T1} | setup of deasserting WR- to leading edge of <T1> SCLK+ | 25 | ... |

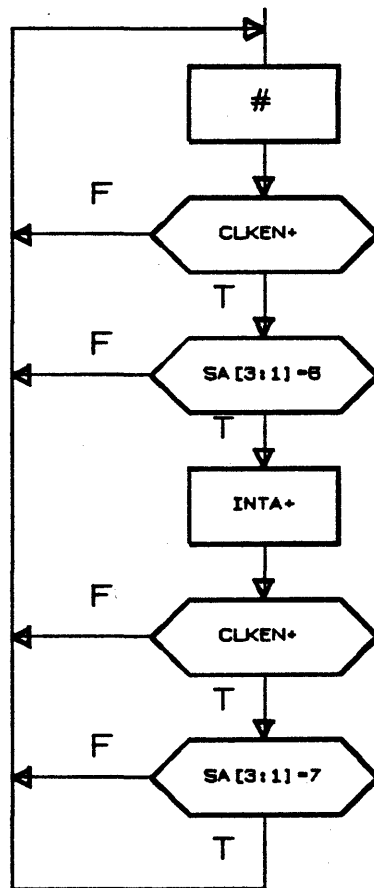
Theory of Operation

INTA+.

The INTA+ {U23-14} signal machine is effectively a synchronous R-S flip-flop.

Logic Reset places the machine with INTA+ in a de-asserted state. When CLKEN+ is true, the state of address bits SA[3:1]+ are sampled. If SA[3:1]+ =6, the machine enters the INTA+ asserted state. If CLKEN+ is false, or any other combination of SA[3:1]+ is detected, the machine remains in the deasserted state.

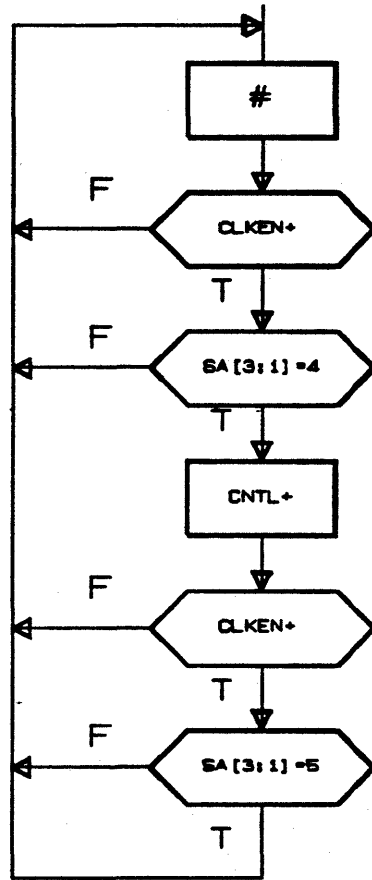
When INTA+ is asserted, the machine then waits for the condition that will force the deasserted state. This condition is CLKEN+ and SA[3:1]+ =7. Again, all other condition, with the exception of logic reset, will leave the machine in the asserted state.



[diagrams.inta]

CNTL+.

CNTL+ {U23-15} behaves exactly like the INTA+ machine, except that the set and reset address conditions for SA[3:1]+ are 4 and 5 respectively.

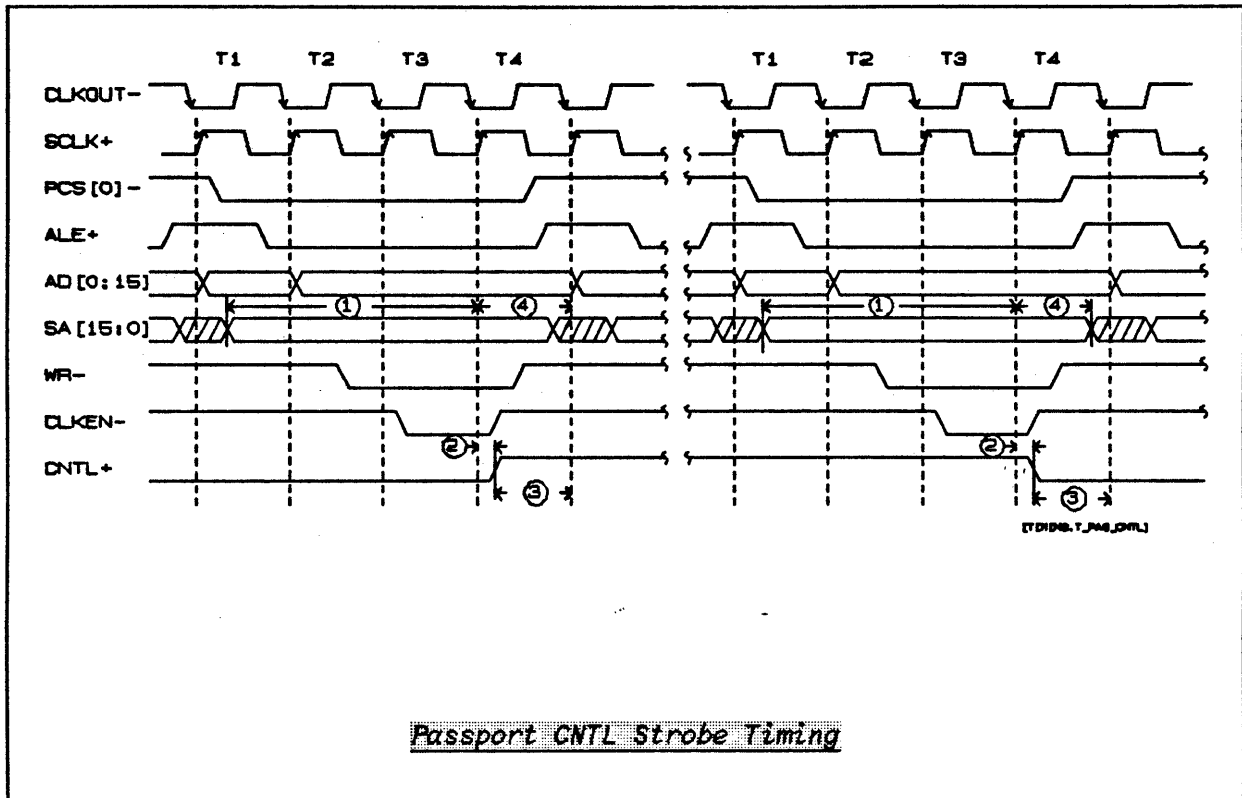


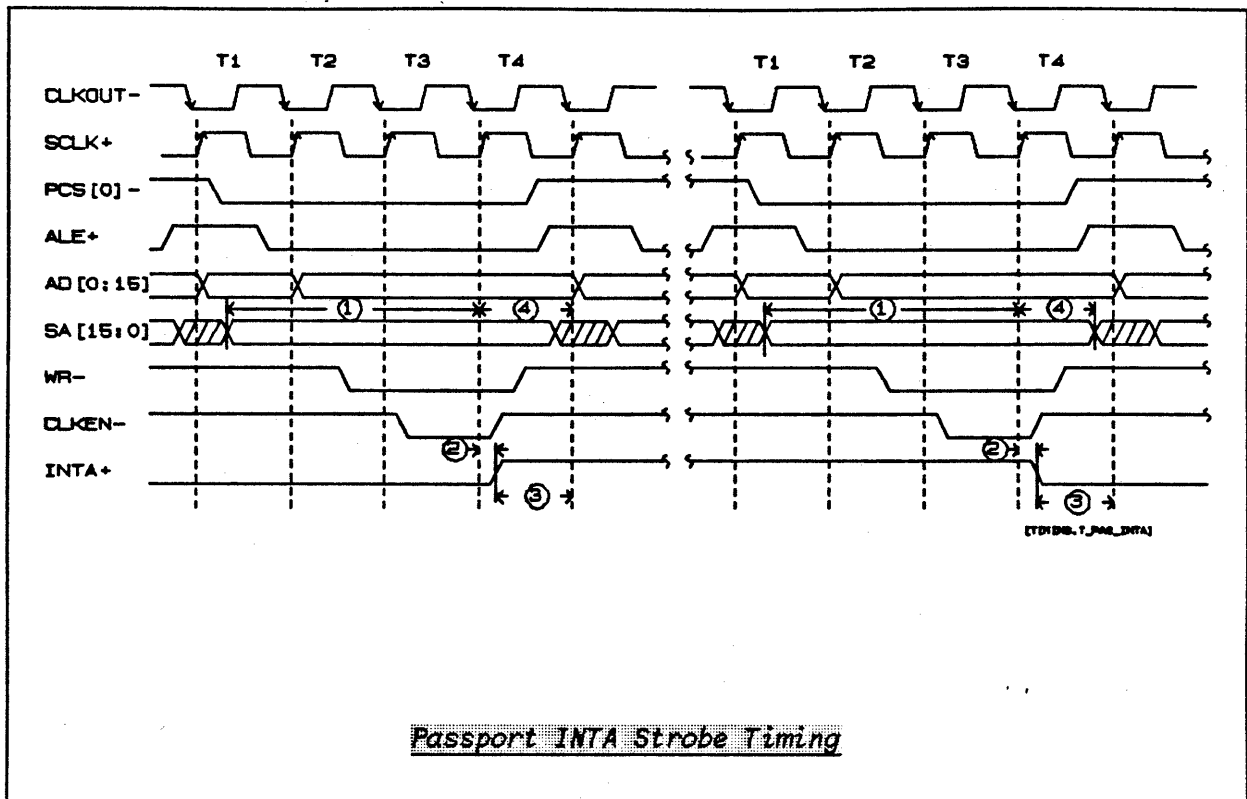
[diagram.cnt1]

Theory of Operation

TIMING.

The timing for accesses to the Passport Control machine is shown following. The control register states are updated on processor bus cycle T4 for valid accesses. Passport requires that control inputs be valid at least 90 nsecs before the rising edge of SCLK+, which is satisfied by the given circuit.





Passport Control Machine Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|---------------|---|-----|-----|
| 1 | t_{SA_T4} | setup of SA[+] to leading edge of <T4> SCLK+ | 25 | ... |
| 2 | t_{T4_P} | delay from leading edge of <T4> SCLK+ to valid control (INTA+, CNTL+) | 0 | 25 |
| 3 | t_{P_SCLK} | setup of control (INTA+, CNTL+) to any leading edge SCLK+ | 100 | ... |
| 4 | t_{T4_SA} | hold of SA[+] from leading edge <T4> SCLK+ | 0 | ... |

RAM DATA STRUCTURE FORMAT.

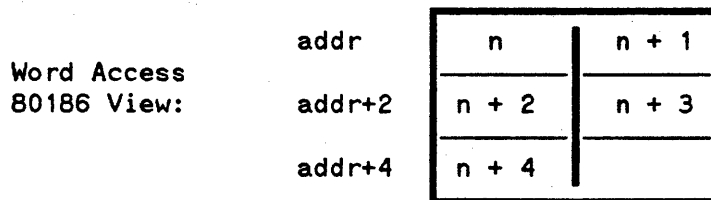
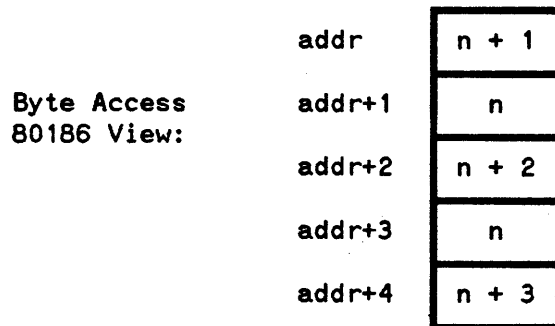
The RAM Data Structure Format is important from the Processor block's view because the Processor block can access data in either *byte* or *word* format and PASSPORT can only use *WORD MODE*.

All Data Structures consisting of lists or arrays of WORDS are operated on normally by the 80186, with the data structure beginning at the low address.

Theory of Operation

All Data Structures consisting of lists or arrays of BYTES, however, are *NOT* stored on sequential byte addresses. This is a limitation of the Backplane Adapter.

An illustration of an odd array of 5 bytes is shown:



[alibyt86]

Secondary Power Support

When secondary power is available, and the Alink CIO Adapter is properly configured, the adapter is capable of some secondary power support. Secondary power support consists of four areas -

- Secondary Power Configuration
- Power Fail Warning Detection
- Memory Protection
- Power Fail Recovery

NOTE

The Power Fail Warning feature is not currently implemented in firmware on the HP27111A at the writing of this document. The hardware and layout necessary to support this function IS present. Therefore the following section should act as a guide to future designers-programmers attempting to implement this feature!!!!

SECONDARY POWER CONFIGURATION.

The "POWER Option Shunt" {U13} is used to configure the card to take power for certain I.C.'s from either the channel's primary or secondary power sources, according to the following table:

| Primary Shunt | Secondary Shunt | Power State |
|---------------|-----------------|------------------|
| Open | Open | illegal |
| Open | Closed | Secondary Active |
| Closed | Open | Primary Only |
| Closed | Closed | illegal |

[alipwrsh]

Primary Power (+5P from the CIO Backplane) feeds U13-1. Secondary Power (+5S from the CIO Backplane) feed U13-2. The opposite sides of the shunt, U13-7,8 are wired together. The Primary Shunt then connects U13-1,8; the Secondary Shunt, U13-2,7. When the secondary power option is selected, the following I.C.'s are supplied power from the secondary power supply: U17, U49, U50. Secondary power is brought into the board via the channel connector pins P1-A39 and P1-B39.

When the primary power option is selected, all I.C.s are directly or indirectly powered by the Primary power supply.

POWER FAIL WARNING DETECTION.

Whether or not a card has been configured for secondary power support, it can still detect the onset and termination of a Power Fail state. Detection of the onset of a possible Power Fail is performed by the Power Fail Warning machine located in U23.

This machine samples the PFW+ signal, which is an inversion of the PFW- signal from the CIO backplane. This signal is asserted at least 10 msec before an actual power fail would occur. Because this signal is asynchronous with respect to SCLK+, the machine checks for two consecutive assertions of PFW+ before the warning is considered to be valid. The synchronized version of PFW+ is SYNC_PFW+ {U23-17}.

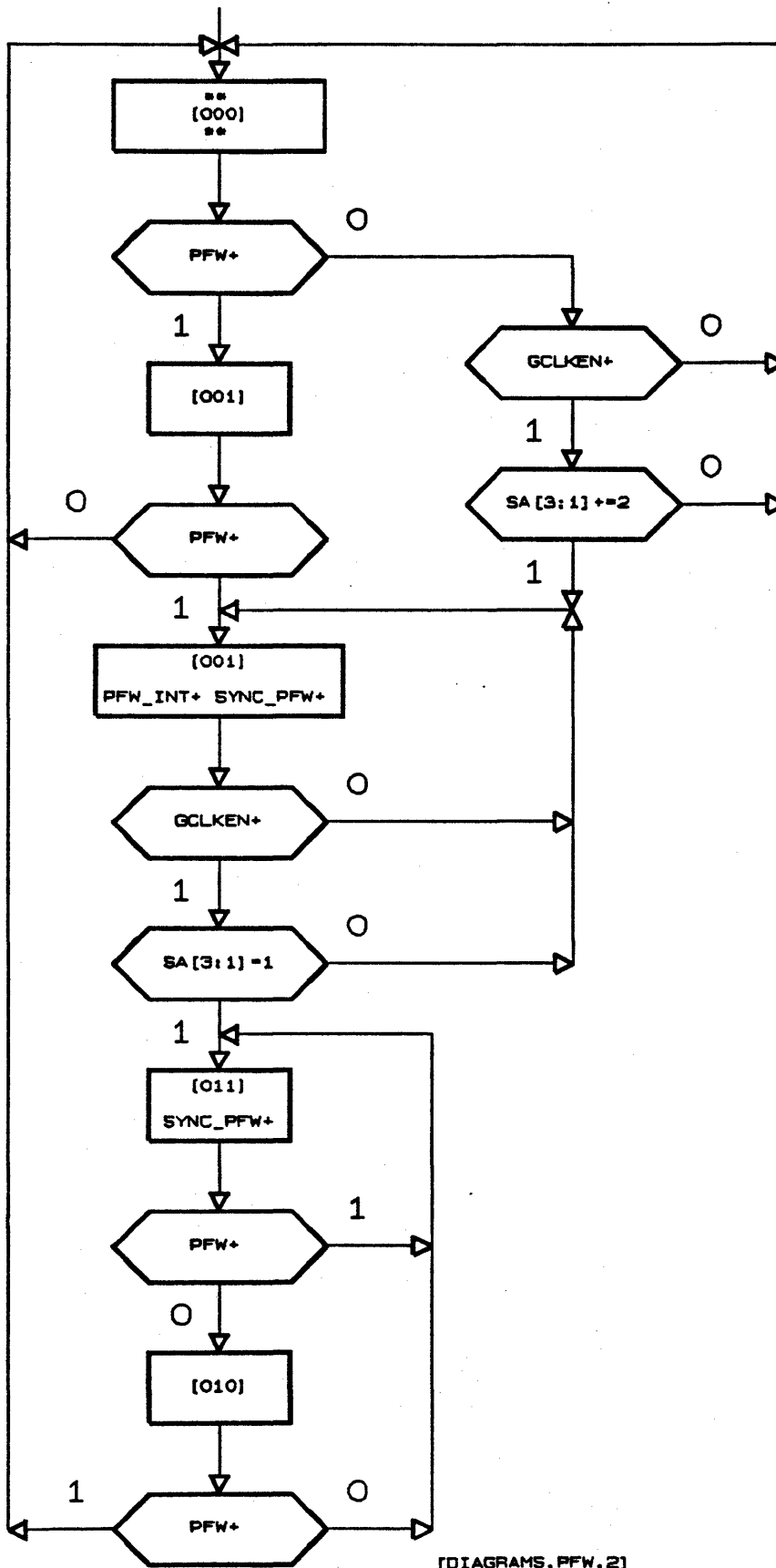
When a warning is present, i.e., SYNC_PFW- is asserted, PFW_INT+ is asserted (to interrupt the processor), and the machine waits for the processor to acknowledge this event before checking for termination of the warning. This is called the WARNING state, and at this time firmware should clean up and save whatever state information is necessary into RAM.

Acknowledgement of the PFW_INT+ is accomplished by writing to the Global Control Register set with SA[3:1]+ = 1. The CLKEN+ signal is used to qualify the address as with the Passport Control Machine.

When the processor acknowledges the warning, it should then execute the TEST instruction. This instruction forces the processor to suspend operation while its TEST pin (U20-47) is a high voltage level. This corresponds to SYNC_PFW+ asserting, which means that the processor is idled until the power fail warning terminates or power actually fails.

Once the warning state has been acknowledged, the machine again samples the PFW+ signal, this time looking for consecutive deassertion which will return the machine to its idle state, SYNC_PFW+ will deassert, and the processor can proceed.

Theory of Operation



[DIAGRAMS.PFW.2]

As noted in the flow chart, it is possible for the 80186 to manually cause PFW_INT+ to assert. The generation is caused by writing to the Global Register with SA[3:1] = 2.

A reset causes the Power Fail Warning Machine to enter the idle state.

MEMORY PROTECTION.

Memory Protection for RAM is provided by the Memory Control PAL {U17} which forces the RAM chip select signals URS- and LRS- to a deasserted state whenever the primary power status signal PON+ is false. As long as the RAM select signals are deasserted, transitions on the WR- line during an actual power fail would have no effect. These signals will remain deasserted as long as PON+ is false and secondary power is present.

POWER FAIL RECOVERY.

Power Fail Recovery will be accomplished during the Processor's Reset Handling routine (sometimes known as selftest...). The plan would be something like this:

```

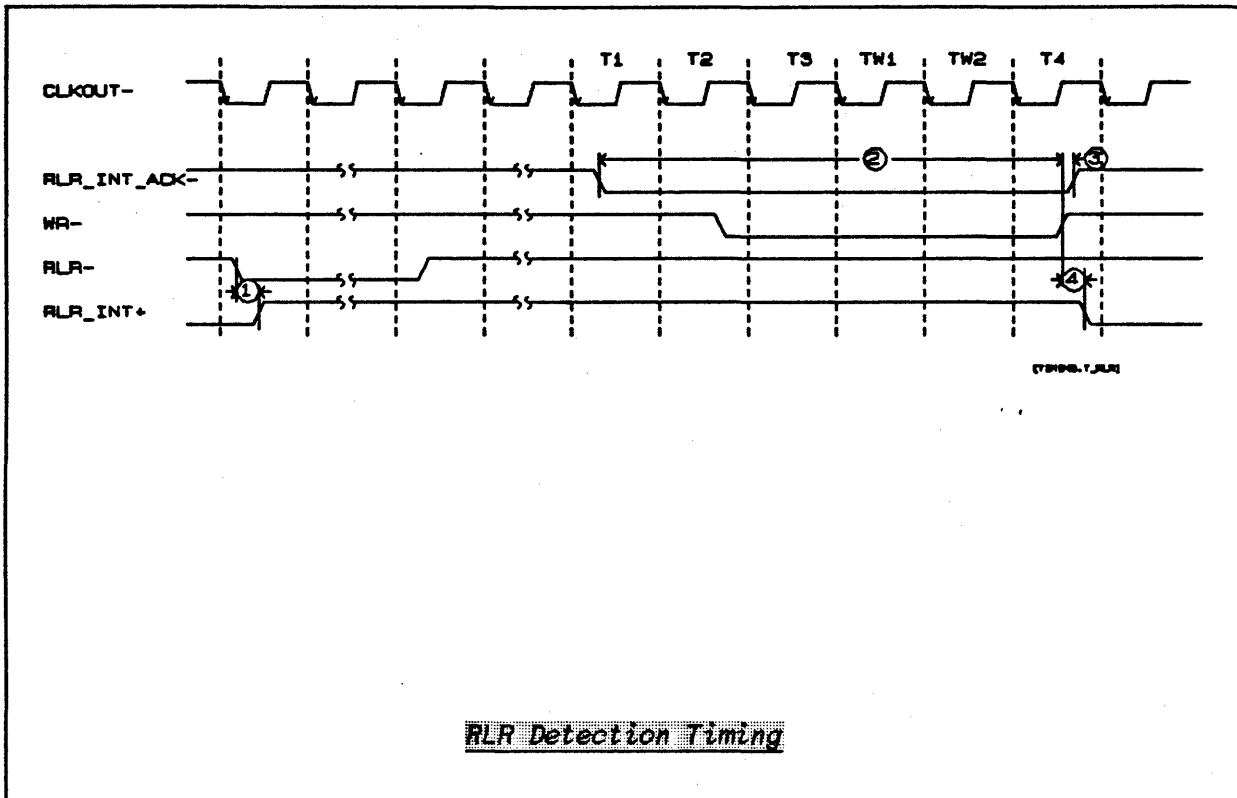
-if initializing from hard reset
- test global status register first
- if global status functional then
- check secondary__power__fail condition in global status register
- if not present then
- {maybe ram is good!}
- wake up Passport
- check Reason__for__Reset condition
- if Power__ON then
- check RAM for valid RAM state flags
- if valid__RAM then
- report to channel that we are O.K.???
- go to main code
- else
- { we couldn't recover cleanly }
- do selftest
- endif
- else {start from scratch}
- do selftest
- endif
- else {start from scratch -- memory is phooey}
- do selftest
- endif
- else
- do selftest {and report the failure}
- endif
-else
- wait for DCL to complete
- do selftest
- do auxiliary selftest
-endif

```

RLR Detection

Assertion of the RLR- pin by Pronto is detected by U22-5 which forces the J-K flip-flop into the set state. This will cause RLR_INT+ {U22-6} to assert, which can be detected by the processor.

RLR_INT+ is cleared by WR- to the register area decoded by PCS[5]- {U20-31} This will cause RLR_INT_ACK- to be asserted when WR- makes a rising transition satisfying the K- clearing condition at U22-3.



Remote Link Reset Detect Timing Parameters

| NOTE | NAME | DESCRIPTION | MIN | MAX |
|------|-----------------------|---|-----|-----|
| 1 | t _{RLR_IRLR} | delay from assertion of RLR- to RLR_INT+ asserted | 3 | 13 |
| 2 | t _{ARLR_WR} | setup of RLR_INT_ACK- to rising edge of WR- | 15 | ... |
| 3 | t _{WR_ARLR} | hold of RLR_INT_ACK- from rising edge of WR- | 0 | ... |
| 4 | t _{WR_XRLR} | delay from WR- to RLR_INT+ deasserted | 7 | 18 |

LED's

The LED's (CR2 through CR7) are driven by the Main Global Control Register. The anodes of each of the LED's are connected to the primary power plane. Each of the LED's has an integrated current limiting resistor, and the cathode-resistor path is tied to individual pins on U26 according to the following table:

| | | | | | | |
|------------|-------|-------|-----|-----|-----|-----|
| U26 pin : | 15 | 12 | 9 | 6 | 5 | 2 |
| AD[] bit : | 10 | 11 | 12 | 13 | 14 | 15 |
| Mnemonic: | P- | A- | C- | S- | R- | F- |
| LED | CR3 | CR2 | CR6 | CR5 | CR4 | CR7 |
| Color | Green | Green | Red | Red | Red | Red |

[aliledmp]

The appendix contains information about status provided by the LED's for this version of the HP27111A. Since this information is controlled by firmware, it is subject to change.

Hardware Revision Code

A three bit revision code, REV[0:2]- is provided to identify future revisions of the HP27111A by firmware. Each of these lines are pulled up through resistors U37-4, 5 and -6, respectively. The pca has provision for the loading of three 0 ohm jumpers, R18, R19, R20, which can be used to selectively pull each revision bit to ground, asserting that bit. The first release of the HP27111A will not load any of the revision jumpers and a revision code of "0" will be reported.

The state of the revision code is found by firmware by reading the Global Status Register and testing the appropriate bit fields.

Equal Mode Jumper

The Equal Mode Jumper is a three position post connector {J2} which is used to select the behaviour of the HP27111A when arbitration of the link resources is necessary. The state of the equal mode jumper is provided to the Global Status register by MORE_EQUAL- {J2-1}, and the jumper can be in either the *Less Equal+* or *More Equal-* position.

When the jumper is in the *Less Equal* position, {J2-2, 3} are shorted together, and {J2-1} is allowed to be pulled high through resistor U37-7. This leaves MORE_EQUAL- deasserted.

When the jumper is in the *More Equal* position, {J2-2, 1} are shorted together, and U37-7 is pulled to GND. This leaves MORE_EQUAL- asserted.

Theory of Operation

If the jumper is missing, {J2-7} is pulled high, and MORE__EQUAL- is false.

Minor Code Report

The Minor Code Report option is intended to be used during the manufacturing test process to force the HP27111A to provide additional status information upon selftest failure.

Normally, the HP27111A will only display a *Major* error code upon selftest failure. This code indicates the phase of selftest being performed when an error was detected.

If the HP27111A fails self-test, it checks the state of MINOR- to see whether it should display additional *Minor* codes to supplement the *Major* code. If MINOR- is true then additional codes will be displayed.

MINOR- may be asserted by board test equipment by pulling {U18-9} low. Normally this line is pulled up through resistor RDB9 to a false state.

Reset Behaviour

POWER ON AND CHANNEL RESET.

The Processor block responds in the same basic fashion for both Power On and Channel Resets.

The 80186 is reset by the assertion of CLK__RST-. Internally, all of its registers are cleared, and all of its integrated internal peripherals are placed in their reset state.

The Global Control Register is reset by the assertion of RESET-. This should cause all of the LEDs to light.

The general state of the Processor block following either Power On or Channel Reset is as follows:

Processor Power On State

| SIGNAL STATE | SIGNAL NAME |
|----------------|--|
| Asserted | SELF_TEST_FAILED-, REMOTE_ERROR-, SIGNAL_ERROR-, CONFIGURATION_ERROR-, ACTIVITY-, PASSED_SELF_TEST-, DISABLE_OPTIC_TRANSMITTER- |
| Deasserted | ALE+, UCS-, LCS-, MCS1-, PCS0-, PCS3-, PRONTO_SEL-, RLR_INT_ACK-, RLR_INT-, HLDA+, GLOBAL_CONTROL_WRITE-, PFW_INT+, INTA+, CNTL+, UPPER_RAM_SEL-, LOWER_RAM_SEL-, SYNCPFW- |
| High Impedance | AD[0:15]+, SA[15:0]+, RD-, WR-, BHE- |
| Periodic | At 8Mhz: SCLK+, CLKOUT-, U36-1/ |

Since CLK_RST- deasserts before RESET-, the 80186 will attempt to initialize before the rest of the card is ready. It does this by attempting to read from address 0FFFF?H, which contains the entry point address for the reset routine. When the 80186 is reset, it samples its external ready lines, which are ordinarily pulled to an asserted state through U37-1. It then proceeds with its reset initialization routine.

NOTE

The 80186 can be prevented from proceeding through its initialization code by holding U37-1 deasserted after Power On or Channel Reset. This feature may be used by technicians during failure analysis.

During Power On and Channel Reset's, CLK_RST- deasserts before RESET- does. Firmware prevents the 80186 from proceeding beyond preliminary initialization by monitoring the state of the CLR_WAIT- line which is connected to the Timer 0 input {U20-20}. When the timer indicates that the wait condition is no longer present the processor proceeds with its booting and selftest code.

Therefore, once RESET- has deasserted, the 80186 will proceed through its basic initialization code and then begin performing a self test on the card.

DEVICE CLEAR.

During a Device Clear, the Global Control Register is reset as before, lighting all the LEDs. Rather than resetting the processor via CLK_RST- the 80186 detects DCL_INT+ asserting at its NMI input.

The 80186 fetches the Device Clear Handler routine entry point from RAM and then begins to execute the Device Clear Handler code.

Theory of Operation

The 80186 at this point performs a similar wait function as with the other forms of reset. It programs its internal timers to detect the deassertion of `CLR_WAIT-` and waits until this occurs.

When `CLR_WAIT-` deasserts, the 80186 then proceeds to execute code which simulates the effects of being reset via the hardware reset pin (`CLK_RST-`).

After the 80186 init code executes, the 80186 proceeds on to selftest the card.

ARBITER

File: [ALIMSARO.TAK.ALINK]

The Arbiter block controls access to the two major data paths on the HP27111A, the Link Data (LD[0:15]+) and System Data (AD[0:15]) busses. Normally, the Link Data bus is used for communication between Protocol Controller and Backplane Adapter, while, in parallel, the Processor is using the System Data bus. Operations that require the Backplane Adapter to access Processor memory, or that the Processor access the Protocol Controller are processed by the Arbiter.

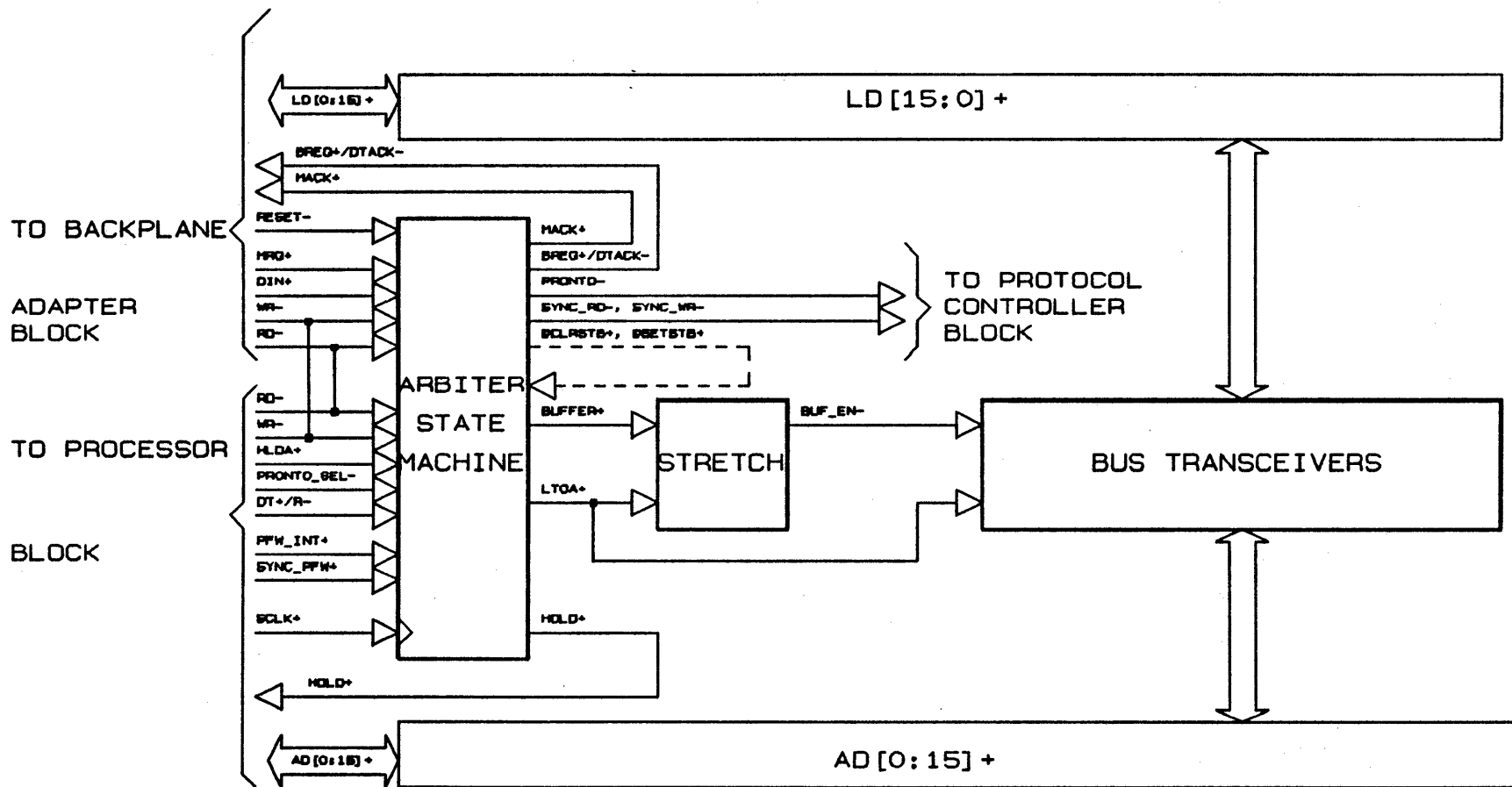
Thus, the Arbiter handles two types of requests:

- Backplane Adapter request (to access memory)
- Processor request (to access Protocol Controller)

The Arbiter arbitrarily (being an arbiter, of course) gives priority to Processor requests.

The hardware to handle these requests consists of:

- Arbiter Pals
- Bus Transceivers
- Pulse Stretcher



[BLOCK1A8_8_ARBITER]

Arbiter PALS

CONTROL SIGNALS.

The following signals are used by the arbiter to control the access of the LD[] and AD[] busses.

Arbiter Signals

| SIGNAL | DRIVEN BY | RECEIVED BY | FUNCTION |
|----------------|-----------|----------------|--|
| UPDATE- | ARBITER | --- | When asserted, indicates that the Arbiter is still updating its bus request status. When deasserted, the Arbiter is actively processing a bus request. |
| BREQ+ / DTACK- | ARBITER | PASSPORT | Bus Request/Data Transfer Acknowledge. A multiplexed Arbiter output. When MACK+ is false, BREQ+ is asserted to force PASSPORT to relinquish the LD[] bus on the following clock state. When MACK+ is true, DTACK- acts as a data transfer acknowledge signal. |
| PRONTO- | ARBITER | PRONTO | Pronto Select. Asserted by Arbiter to initiate a Control Port access with Pronto. |
| BUFFER+ | ARBITER | BUS XCVR LOGIC | Buffer Activate. Assertion causes the isolation bus transceivers to be come enabled, enabling the data path between the LD[] and AD[] busses. |
| MACK+ | ARBITER | PASSPORT | Memory Request Acknowledge. Asserted when the Arbiter has determined that the 80186 has released its busses for access by PASSPORT. in response to a Passport Memory Request. |
| RESET- | PASSPORT | ARBITER | Logic reset. When asserted, causes the Arbiter to enter its idle state. |
| DIN+ | PASSPORT | ARBITER | Data In. When asserted in conjunction with a Passport Memory Request indicates that a Read operation is desired; when deasserted, a Write operation. |

Arbiter Signals (cont.)

| SIGNAL | DRIVEN BY | RECEIVED BY | FUNCTION |
|-----------------------|--------------------|--------------------|--|
| MRQ+ | PASSPORT | ARBITER | Memory Request. When asserted, generates HOLD+ to obtain access to 80186 busses. |
| HOLD+ | ARBITER | 80186 | Hold. Assertion indicates that the Arbiter wants access to the 80186 busses. |
| HLDA+ | 80186 | ARBITER | Hold Acknowledge. Assertion indicates to Arbiter that the 80186 has tri-stated its external busses and that RAM is accessible. |
| LTOA+ | ARBITER | BUS XCVR LOGIC | LD[] to AD[]. Establishes the direction of data flow between the two busses. When asserted, LTOA+ will allow data to flow from LD[] to AD[]. |
| SYNC_RD- | ARBITER | PRONTO | Synchronous Read. Asserted with SELECT- during Pronto Control Port Read operations. |
| SYNC_WR- | ARBITER | PRONTO | Synchronous Write. Asserted with SELECT- during Pronto Control Port Write operations. |
| DT/R- | 80186 | ARBITER | Data Transmit. Asserted by the 80186 during Write operations, and deasserted during Read operations. |
| PRONTO-SEL- (PCS[4]-) | 80186 | ARBITER | 80186 Pronto Select Strobe. Asserted by 80186 to identify a Read or Write operation as intended for Pronto. |
| RD- | 80186 and PASSPORT | MEMORY and ARBITER | Read Strobe. Asserted during Read operations |
| WR- | 80186 and PASSPORT | MEMORY and ARBITER | Write Strobe. Asserted during Write operations. |
| STRETCH+ | ARBITER | BUS XCVR LOGIC | Stretch Enable. Used to stretch by a half clock cycle the amount of time the bus transceivers are enabled. |

Arbiter Signals (cont.)

| SIGNAL | DRIVEN BY | RECEIVED BY | FUNCTION |
|---------|--------------------|--------------------|--|
| BUF_EN- | BUS XCVR LOGIC | BUS XCVR | Buffer Enable. Enables the Bus Isolation Transceivers {U25,U26}. |
| FLOAT- | test equip | ARBITER | Float Synchronous Outputs. Assertion causes the following outputs to enter the high-impedance state: LTOA+, SELECT-, BREQ+, MACK+, SYNC_WR, SYNC_RD, BUFFER+, UPDATE-. |
| @SETSTB | ARBITER (internal) | ARBITER (internal) | Set Pronto Strobe. Causes Arbiter to generate SYNC_RD- or SYNC_WR- to Pronto. |
| @CLRSTB | ARBITER (internal) | ARBITER (internal) | Clear Pronto Strobe. Forces the deassertion of either SYNC_RD- or SYNC_WR-. |

[aliarbsg]

In describing the state behaviour of the main state machine the following group of arbiter signals is treated as a vector:

[UPDATE,BREQ,SELECT,BUFFER,MACK]+

Within the vector, all signals are considered to be positive true and each vector position is given binary weighting, with the most significant signal position to the left.

For example if the following levels for the vector were measured on the card

[UPDATE,BREQ,SELECT,BUFFER,MACK]+ = [LLHLL]

The state would be identified as vector #4.

The following would be identified as vector #26:

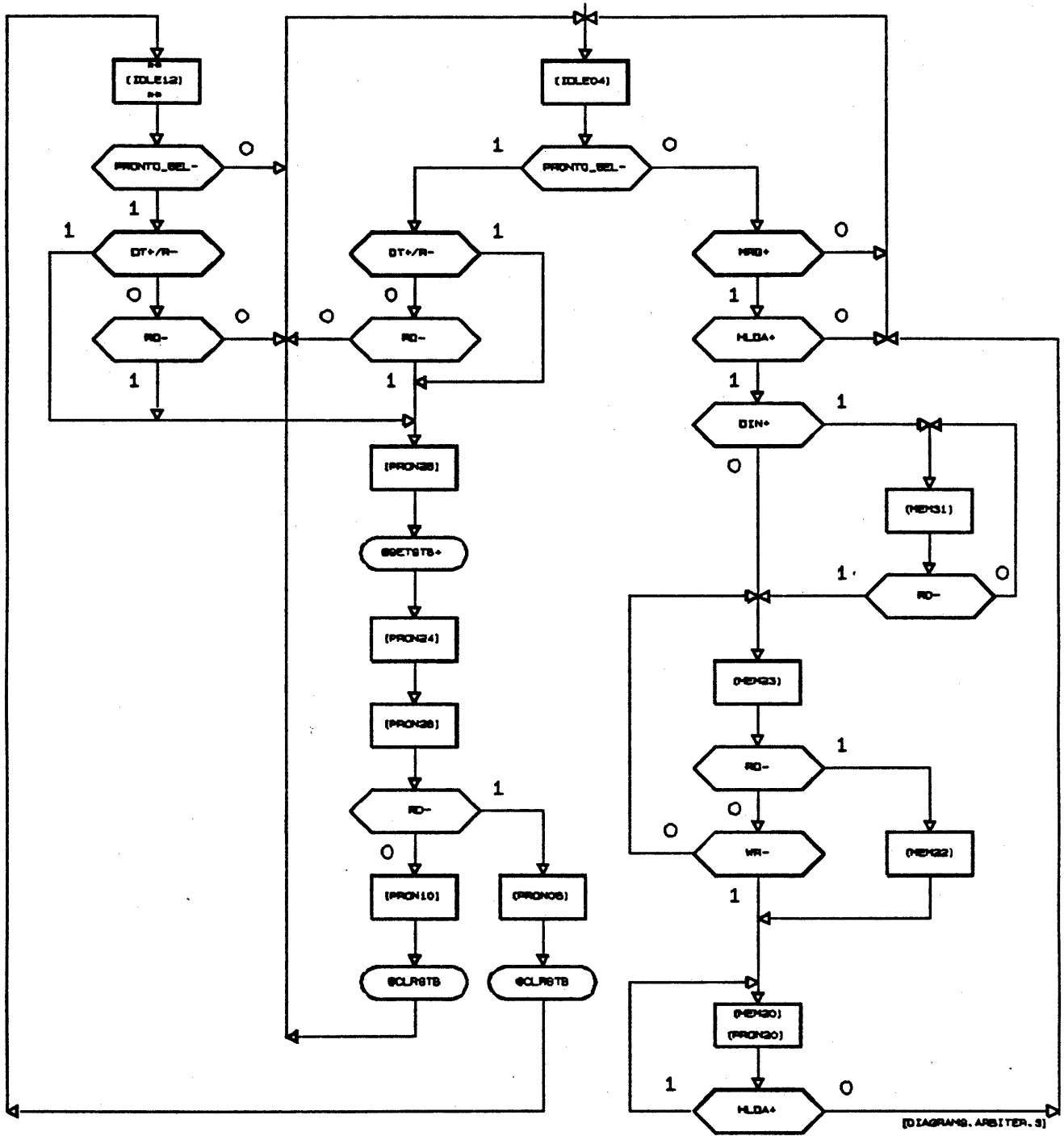
[UPDATE,BREQ,SELECT,BUFFER,MACK]+ = [HHLHL] = vector #26.

and so on.

Main Arbiter:FLOW CHART.

The Arbiter State Machine is implemented according to the following flow chart.

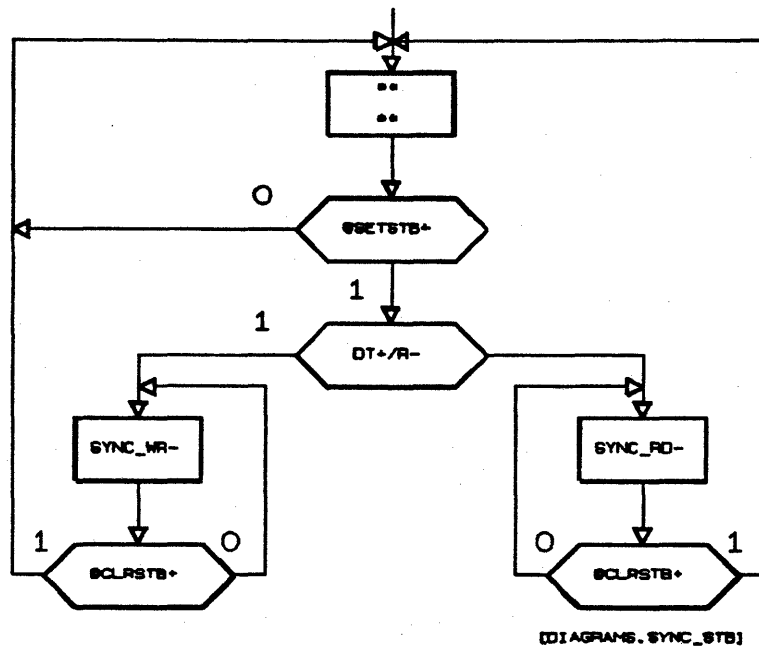
Theory of Operation



READ WRITE GENERATION:FLOW CHART.

The read and write strobes to Pronto, SYNC_RD- and SYNC_WR- {U15-12, 13} are essentially generated by the main arbiter, but for conceptual reasons, a separate flow chart is used to describe their state behaviour.

As noted in the control signal table, the signal pair @SET_STROBE+ and @CLR_STROBE+ are internal decodings of the main arbiter's state.



Bus Transceivers

The arbiter uses a pair of 8 bit bus transceivers {U25,U26} to isolate the AD[]+ and LD[]+ busses. The transceivers have direction and enable control lines LTOA+ {U15-16} and BUF_EN- {U36-13}, respectively.

Transceiver function according to control input is shown in the following table:

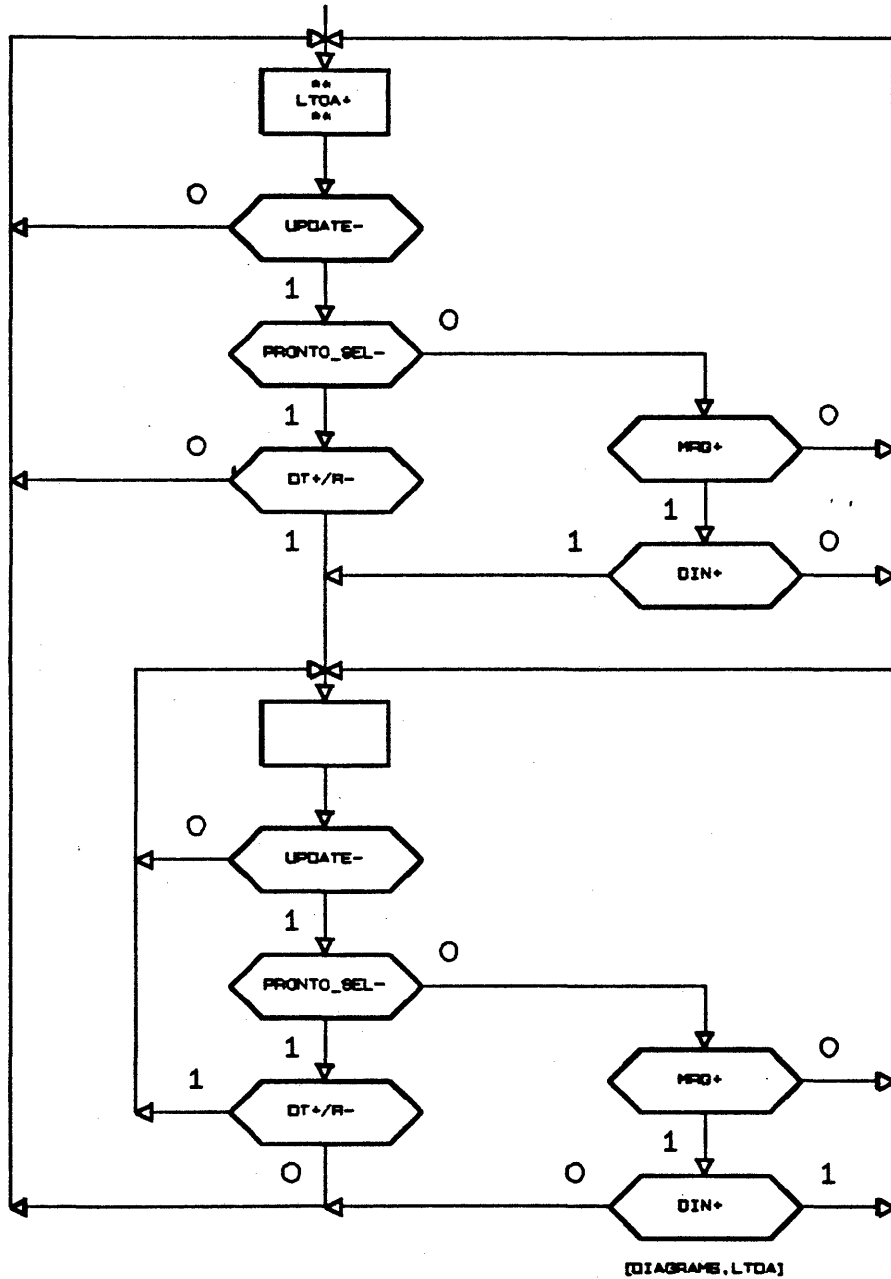
| BUF_EN- | LTOA+ | Data Flow Direction |
|---------|-------|---------------------|
| 0 | X | isolated |
| 1 | 0 | AD[] to LD[] |
| 1 | 1 | LD[] to AD[] |

[aliad2ld]

arbiter bus transceiver behaviour

DIRECTION MACHINE.

The Direction state machine controls the state of the LTOA+ {U15-16} signal, which in turn, determines the direction of bus transfer between the LD[+] and AD[+] busses. The state machine operates such that the state of LTOA+ may only change when UPDATE- is asserted, and either the Processor or Passport is attempting to gain control of the bus (PRONTO_SEL- or MRQ+ asserted).



Pulse Stretcher

To satisfy various data timing requirements, the arbiter makes use of a pulse stretching circuit to selectively lengthen the time in which the bus transceivers are enabled, by asserting STRETCH+ {U21-10}.

The stretch function is implemented using a JK flip-flop in a D flip-flop mode which is clocked by a delayed and inverted version of SCLK+ {U36-1}. This corresponds to clocking {U21} a short delay after the falling edge of SCLK+, or, effectively, mid-cycle.

When the direction of data transfer is from AD[] to LD[] the stretcher is held inactive since LTOA+ in a false state holds the JK reset pin {U21-15} asserted. This holds STRETCH+ deasserted.

When the direction of data transfer is from LD[] to AD[] the stretcher is active since LTOA+ is true. STRETCH+ then follows the state of BUFFER+ {U15-17} since the flip-flop has been placed in a D-flip-flop orientation by connecting the J and K- inputs together {U21-14,U21-13}.

STRETCH+ will continue to follow BUFFER+ with a half state delay until LTOA+ is false, at which point STRETCH+ will always deassert.

Arbiter State Machine:Processor Request

When the Processor makes a request to the Arbiter to access the Protocol Controller, the Arbiter responds by forcing the Backplane Adapter to relinquish the LD[]+ bus, and then sequences through the proper control strobes for Control Port access of the Protocol Controller. After the access has completed, the Arbiter then decides whether to allow the Backplane Adapter to proceed again.

As noted in the flow chart, the arbiter takes two slightly different paths depending upon whether the Processor is performing a read or a write operation.

PROCESSOR READ PRONTO.

The 80186 has been programmed to insert two wait states into its normal 4 cycle bus operation when accessing Pronto. This corresponds to a bus cycle progression of <T1>,<T2>,<T3>,<TW1>,<TW2>,<T4>. It will sample data on the leading edge of bus state <T4>, which is the beginning of the sixth bus cycle.

The function of the

With this in mind, from the arbiter's perspective the state sequence for a Processor Read from Pronto is as follows:

Arbiter State:<IDLE04> 80186:<T1>

The arbiter has been idling in this state waiting for the assertion of PRONTO_SEL- {U15-4} (or MRQ+ {U15-8} from Passport).

When PRONTO_SEL- is asserted the arbiter checks the sense of DT/R- {U15-5} to see whether a read or write operation is intended. For a read, DT/R-=LOW and the arbiter waits for the assertion of RD- {U15-6}, which will occur in the next state.

Arbiter State:<IDLE04> 80186:<T2>

On the leading edge of this state, the Direction

Theory of Operation

Machine determines that the 80186 has a read request pending and asserts LTOA+

The 80186 asserts RD- during <T2>, which is detected by the arbiter and causes it to change state.

Arbiter State:<PRON28> 80186:<T3>

On the leading edge of this state, the Direction Machine determines that the 80186 still has a read request pending and continues to assert LTOA+.

At this point, the arbiter has deasserted its UPDATE- {U15-15} signal, indicating it is active. It also issues a bus request to the PASSPORT for the link bus by asserting BREQ+ {U15-18}, which guarantees that the bus will be available on the next state.

Arbiter State:<PRON24> 80186:<TW1>

The arbiter then asserts PRONTO- and SYNC_RD- {U15-14, 12} to PRONTO to begin the read.

Arbiter State:<PRON26> 80186:<TW2>

Data from Pronto becomes valid on the LD[] bus during this state.

The arbiter asserts BUFFER+ which causes BUF_EN- {U36-13} to assert. BUF_EN- being true turns on bus transceivers U25 and U26 allowing data on the LD[]+ bus to be driven onto the AD[] bus.

The assertion of BUFFER+ is also detected by the pulse stretcher {U21} on the rising edge of {U36-1}, causing STRETCH+ {U21-10} to assert.

Arbiter State:<PRON08> 80186:<T4>

The 80186 samples data on the leading edge of this bus cycle.

The arbiter deasserts BUFFER+ but BUF_EN- remains asserted for another half clock cycle due to STRETCH+. This satisfies the 80186 data hold time specification.

Arbiter State:<IDLE12> 80186:<T1>

The arbiter deasserts SELECT- and SYNC_RD-. Pronto requires two states to recover before processing another Device Port access. Therefore the arbiter continues to assert BREQ+.

Since the 80186 is in its <T1> state, it is possible that another arbiter request is present. If a *Pronto Write* request is present, the arbiter jumps into the Write Request path at <PRON28>, otherwise the arbiter

returns to the <IDLE04> state.

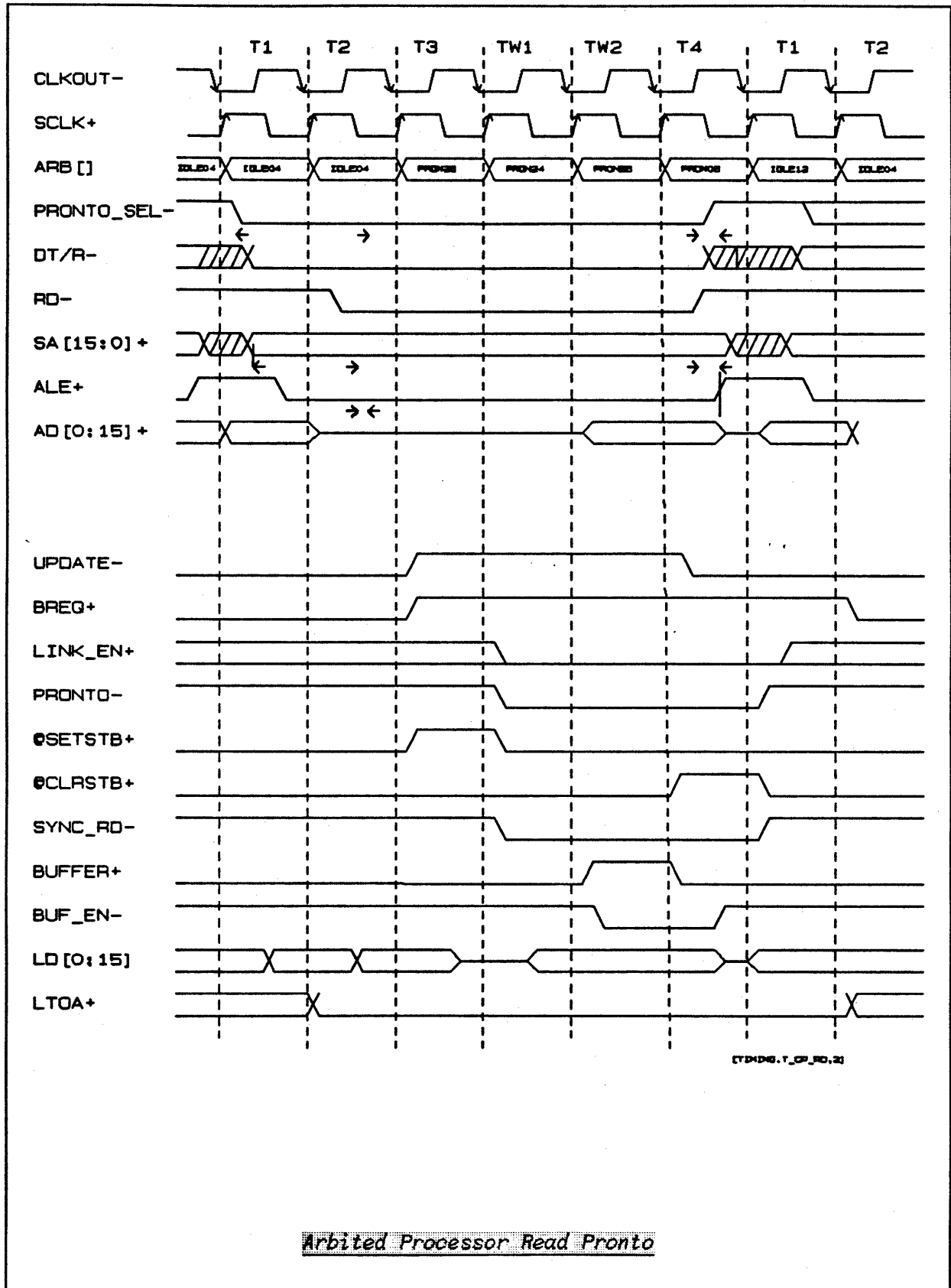
The arbiter verifies this by checking the sense of **PRONTO_SEL-** and **DT/R-** to see whether an operation is intended, and if so, whether it is a read or a write.

Arbiter State:<IDLE04> 80186:<T2>

BREQ+ has been deasserted, which will allow the Backplane Adapter to begin a Device Port transfer if one is able to be made.

The general behaviour of the arbiter in this state has been previously described earlier in the read request discussion if a *Pronto Read* request is present. If no request is present, the arbiter will continue to remain in the idle state until either the Processor or the Backplane Adapter does issue a request.

Theory of Operation



PROCESSOR WRITE PRONTO.

Again, the 80186 has been internally programmed to insert two wait states into its bus cycle for a Write Pronto Register operation. Data is presented on the AD[+] bus for sampling sometime during bus state <T2>.

The arbiter state sequence is as follows:

Arbiter State:<IDLE04> 80186:<T1>

The arbiter has been idling in this state waiting of the assertion of PRONTO_SEL-.

When PRONTO_SEL- is asserted the arbiter checks the sense of DT/R- to see whether a read or write operation is intended. Since this is a Write operation, DT/R-=HIGH and the arbiter immediately proceeds to the next state.

Arbiter State:<PRON28> 80186:<T2>

On the leading edge of this state, the Direction machine deasserts LTOA+, setting up the bus transceivers for an 80186 write operation.

The arbiter deasserts its UPDATE- signal, indicating it is active. It also asserts BREQ+, indicating to Passport that the arbiter wants the LD[] bus on the next clock cycle.

Arbiter State:<PRON24> 80186:<T3>

The arbiter then asserts and PRONTO- and SYNC_WR- {U15-13} to PRONTO to begin the Pronto Register Write.

Arbiter State:<PRON26> 80186:<TW1>

The arbiter asserts BUFFER+, which causes BUF_EN- to assert, enabling bus transceivers U25 and U26. Data from the AD[+] bus is driven onto the LD[+] bus. The pulse stretcher is disabled since LTOA+ is false.

Arbiter State:<PRON10> 80186:<TW2>

Data on the LD[+] bus is sampled by Pronto on the leading edge of this state.

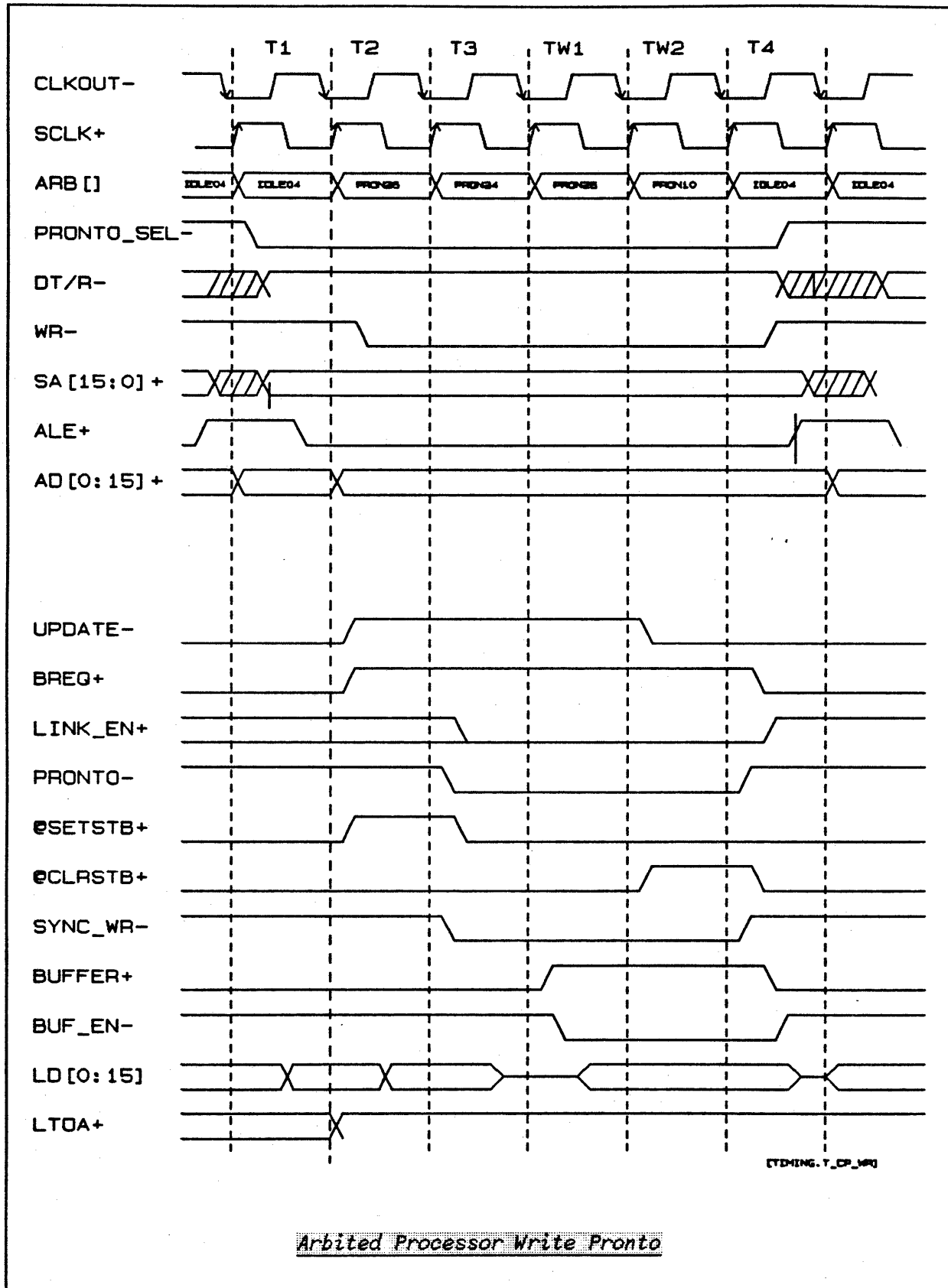
Arbiter State:<IDLE04> 80186:<T4>

The arbiter deasserts BUFFER+ and therefore BUF_EN- (since there is no stretching here) as well as SYNC_WR- and SELECT-, completing the Pronto access. BREQ+ is deasserted, indicating to Passport that the bus request has completed.

Arbiter State:<IDLE04> 80186:<T1>

The arbiter remains in the idle state, waiting for its next request.

Theory of Operation



CONSECUTIVE REQUESTS.

The arbiter will always grant 80186 access to the LD[+] bus without any holdoff over any number of consecutive machine cycle requests.

The arbiter is setup such that no more than two consecutive Processor requests can occur without the Backplane Adapter getting at least one clock cycle in which to transfer data across the LD[] bus.

Even under complete loading by the Processor, the link data bus is still capable of at least 1 transfer every 6 clock cycles, which results in a worst case (*and unrealistic*) link throughput of 2.7 Mbytes. Since firmware cannot keep the arbiter that busy, the effect on throughput should be minimal.

Arbiter State Machine:Backplane Adapter Request

The arbiter also processes Passport's access requests for RAM. The Passport request mechanism is a bit more complicated than the 80186 request path because there are more interlocked handshakes involved.

Before examining the state machine, it is necessary to understand how the 80186 allows external devices to access its address data and control busses.

80186 AD[] BUS REQUEST MECHANISM.

The 80186 uses a Hold/Hold Acknowledge handshake to transfer control of its external busses. The assertion of HOLD+ {U17-12} causes the 80186 to relinquish its busses at the next convenient opportunity, usually within 2 to 3 clock cycles. To indicate that this has occurred the 80186 then asserts HLDA+ {U20-51 + U15-3} to acknowledge that the hold request has been granted. HLDA+ will remain asserted until the holding condition is removed by the deassertion of HOLD+.

Ordinarily all Passport Memory Requests will cause HOLD+ to be asserted. The exception to this is when the Power Fail Warning machine has entered the <Warning Acknowledged> state. Since memory is hands off at this point, {U17} ignores requests until the power fail terminates.

RAM CONTROL.

Once HLDA+ has been asserted the RAM Control Pal U17-6 asserts both of the RAM select lines, UPPER_RAM_SEL-, LOWER_RAM_SEL-. The RAM Control PAL ignores any Processor information until HLDA+ has been deasserted.

The revised RAM Control truth table is shown below.

| RAM Select Decode Table with Power Monitor, Arbiting | | | | | | | |
|--|------|------|---------|------|--------|------|------|
| PON+ | HLDA | LCS- | MCS[1]- | BHE- | SA[0]+ | URS- | LRS- |
| 0 | X | X | X | X | X | 0 | 0 |
| 1 | 0 | 0 | 0 | X | X | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | X | X | X | X | 1 | 1 |

[aliramct]

Secondly, both RD- and WR- are pulled up through 10K resistors (U37-3,8) to +5. This prevents glitches on these lines from occurring as the Processor and Backplane Adapter exchange control of the RAM, since both of them place these lines in a high-impedance state before releasing control of the bus.

PASSPORT READ MEMORY.

As mentioned in the description of the Backplane Adapter, Passport's memory operations can be considered to be a 4+ cycle operation, with the additional cycles due to wait states.

In the case of a Memory Read, there are a variable number of wait states after MRQ+ is asserted until the 80186 completes its acknowledge as well as 1 fixed wait state that is always asserted after <P3>. Thus the Memory Read bus cycle sequence from Passport's perspective would appear as <P1>,[variable # of <Pw1>],<P2>,<P3>,<PW>,<P4>.

In terms of the main state machine states A Passport Read Memory operation appears as the following arbiter state sequence:

Arbiter state:<IDLE04> Passport:<P1> The arbiter detects the assertion of MRQ+. HLDA+ is false, however, so the arbiter remains in its current state.

Since the arbiter is still in the UPDATE- state, the direction machine detects MRQ+ and DIN+ {U15-7} and deasserts LTOA+.

Arbiter state:<IDLE04> Passport:<P1w> The arbiter remains in the IDLE04 state until it sees the 80186 assert HLDA+. Passport is in a bus wait state. Depending upon when MRQ+ was asserted, the arbiter may have to wait from 2 to 5 clock cycles until the HLDA+ signal is returned.

The eventual assertion of HLDA+ also asynchronously causes two other events to occur. First, the address latches {U27,U18} will be disabled, tri-stating the address bus SA[+] . Secondly, the two RAM select lines {URS-,LRS-} will be asserted. This behaviour will be maintained until HLDA+ is removed.

Arbiter state:< MEM31> Passport:<P2>

The arbiter asserts MACK+ {U15-19} which will be detected by Passport on the leading edge of the next cycle, indicating that Passport may begin its read operation. The assertion of MACK+ also allows Passport to drive the SA[] bus.

The data path between the LD[+] and AD[+] busses is linked by the assertion of BUFFER+ and thus, BUF_EN- , which, since LTOA+ is false, will cause data from the AD[+] bus to appear on the LD[+] bus.

Because the Memory data path is not quite fast enough to satisfy Passport's timing specs, a memory wait state is setup by de-asserting DTACK- {U15-18}.

UPDATE- is now false, and the direction machine locks into the state with LTOA+ false.

Arbiter state:< MEM31> Passport:<P3>

Passport asserts RD- and data from the RAMs begins to propagate to the LD[] bus. The arbiter holds DTACK- deasserted, forcing Passport to stretch the read another cycle.

Arbiter state:< MEM23> Passport:<P3w>

DTACK- is asserted by the arbiter, allowing Passport to terminate the read.

Arbiter state:< MEM22> Passport:<P4>

Passport samples data from the LD[+] bus on the leading edge of this cycle. Passport then deasserts MRQ+ and RD-, and the arbiter deasserts MACK+ completing the read operation.

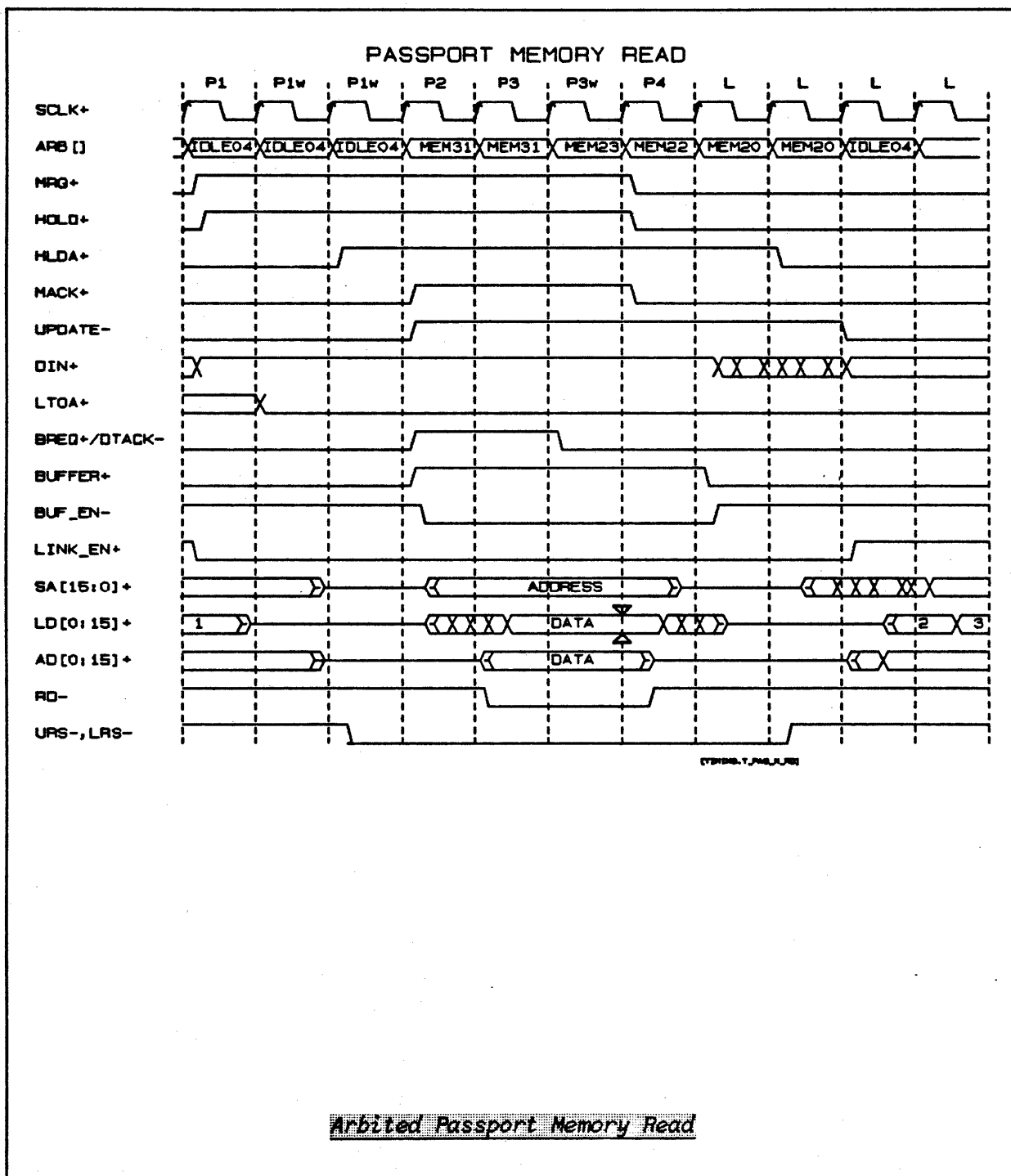
Arbiter state:< MEM20> Passport:< >

The arbiter disables the AD[+] to LD[+] data path by deasserting BUFFER+. It now waits for the HLDA+ signal to be deasserted by the 80186, which will allow it to complete the arbitration.

When HLDA+ deasserts, the address latches are re-enabled and the RAM select lines are deasserted.

Theory of Operation

Arbiter state: <IDLE04> Passport: < > UPDATE- is asserted again, indicating that the arbiter is capable of processing another request.



PASSPORT WRITE MEMORY.

In the case of a Memory Write, there are a variable number of wait states after MRQ+ is asserted until the 80186 completes its acknowledge. There are no memory wait states for a write operation. Thus the Memory Write bus cycle sequence from Passport's perspective would appear as <P1>,[variable # of <Pw1>],<P2>,<P3>,<P4>.

In terms of the main state machine states A Passport Write Memory operation appears as the following arbiter state sequence:

Arbiter state:<IDLE04> Passport:<P1>

The arbiter detects the assertion of MRQ+. HLDA+ is false, however, so the arbiter remains in its current state.

Since the arbiter is still in the UPDATE- state, the direction machine detects MRQ+ asserted and DIN+ deasserted and asserts LTOA+, since data will be transferred from LD[]+ to AD[]+.

Arbiter state:<IDLE04> Passport:<P1w>

The arbiter remains in the IDLE04 state until it sees the 80186 assert HLDA+. Passport is in a bus wait state. Depending upon when MRQ+ was asserted, the arbiter may have to wait from 2 to 5 clock cycles until the HLDA+ signal is returned.

The eventual assertion of HLDA+ also asynchronously causes two other events to occur. First, the address latches {U27,U38} will be disabled, tri-stating the address bus SA[]+. Secondly, the two RAM select lines {URS-,LRS-} will be asserted. This behaviour will be maintained until HLDA+ is removed.

Arbiter state:< MEM23> Passport:<P2>

The arbiter asserts MACK+ which will be detected by Passport on the leading edge of the next cycle, indicating that Passport may begin its read operation. The assertion of MACK+ also allows Passport to drive the SA[] bus.

The data path between the LD[] and AD[] busses is linked by the assertion of BUFFER+ and thus, BUF_EN-, which, since LTOA+ is true, will cause data from the LD[]+ bus to appear on the AD[]+ bus.

Since we need to insure that data is going to be valid on the trailing edge of Passport's write pulse, STRETCH+ is asserted midway through this cycle.

Memory is fast enough to process the Passport write access, so DTACK- is asserted.

Theory of Operation

UPDATE- is now false, and the direction machine locks into the state with **LTOA+** true.

Arbiter state:< MEM23> Passport:<P3>

Passport asserts **WR-** {U15-9}.

Arbiter state:< MEM20> Passport:<P4>

Passport deasserts **MRQ+** and **WR-**, and the arbiter deasserts **MACK+** completing the write operation.

The arbiter deasserts **BUFFER+**. However **STRETCH+** remains asserted for another half cycle, keeping **BUF_EN-** asserted until mid-cycle. At that point the bus transceivers are disabled, and the **LD[]** and **AD[]** busses are isolated.

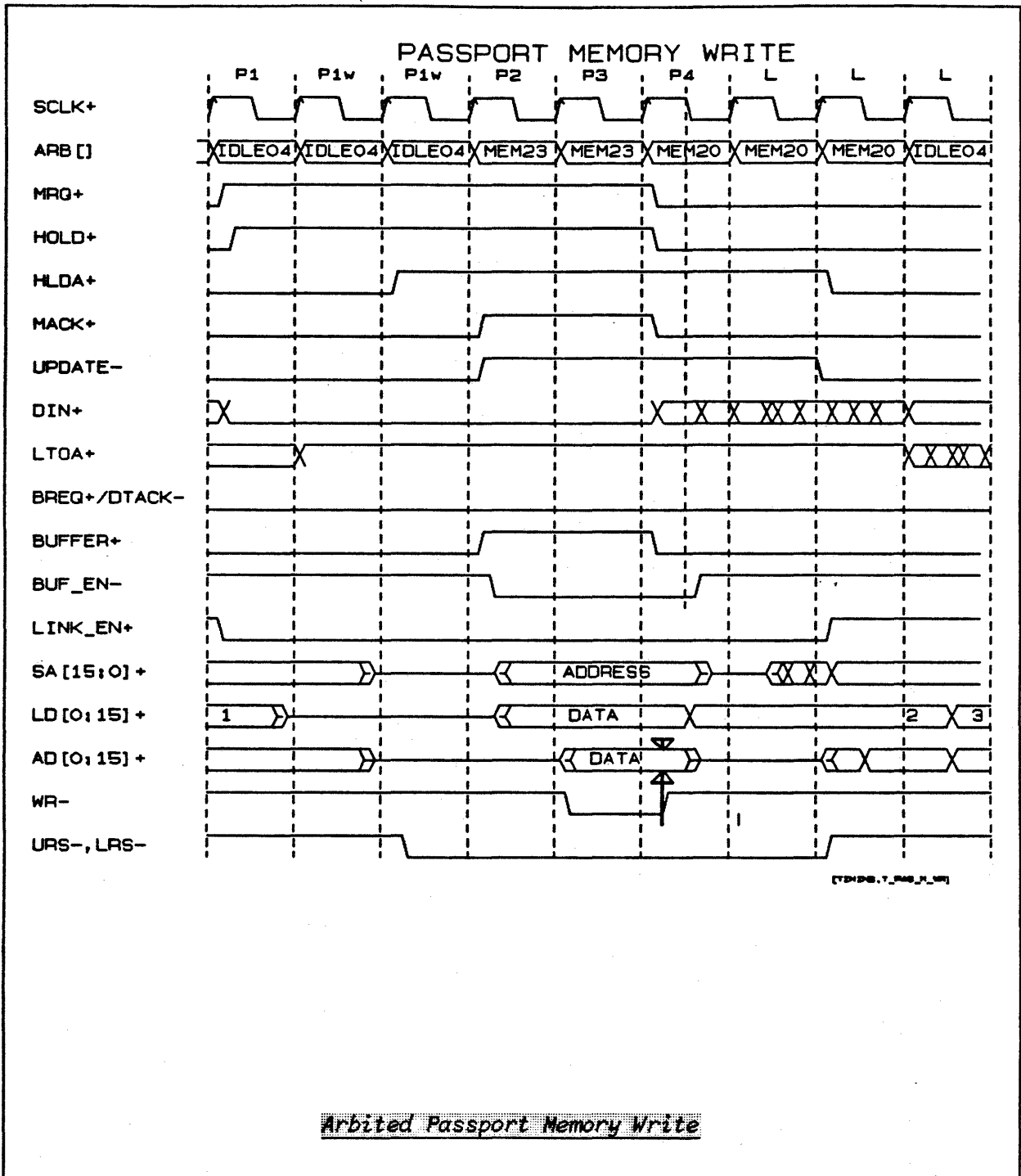
It now waits for the **HLDA+** signal to be deasserted by the 80186, which will allow it to complete the arbitration.

Arbiter state:< MEM20> Passport:< >

When **HLDA+** deasserts, the address latches are re-enabled and the RAM select lines are deasserted. The arbiter then returns to its idle state.

Arbiter state:<IDLE04> Passport:< >

UPDATE- is asserted again, indicating that the arbiter is capable of processing another request.



CONSECUTIVE REQUESTS.

During some modes of operation, Passport will issue continuous requests to the Arbiter for memory access. The arbitration circuitry is designed to assure that neither Passport or the 80186 can monopolize the bus. In particular if Passport is in a mode where it wishes to constantly access the bus, it will never be granted consecutive accesses unless the 80186 has no need of the bus.

Theory of Operation

The reason that this works is that Passport has at least three idle states between every memory access. This idle time is measured as the number of states from the deassertion to the reassertion of MRQ+. If the 80186 had need of running a machine cycle it will initiate one on the third state after MRQ+ deasserts. It will then acknowledge this request to the arbiter after the machine cycle completes, allowing the arbiter to assert MACK+ to Passport and effectively interleaving the requests.

Arbitration of multiple requests

It is possible for the card to enter a state in which both Passport and the Processor desire to access each other's busses. In this case, the Processor request always receives precedence.

There are two cases of interest here. The Processor and Passport request arrive simultaneously or the Passport request arrives while the arbiter has granted control to the Processor.

SIMULTANEOUS REQUESTS.

The arbiter always gives precedence to Processor requests when both appear simultaneously. This is in accord with the 80186 internal bus arbitration behaviour which says that the 80186 must finish its current access before granting its bus.

In this case, Passport would have to wait as many as 8 states before being granted access to the bus by the assertion of MACK+.

PASSPORT REQUESTS WHILE ARBITER UNDER PROCESSOR CONTROL.

The arbiter ignores any new requests when it is not in states <IDLE04>, <IDLE12>. Therefore Passport will be held off until the Processor completes the current, and possibly a second bus cycle.

If MRQ+ asserted during 80186 bus state <T4>, the 80186 will run another machine cycle of 4 to 6 clock cycles, otherwise the 80186 will grant the bus on the next available machine cycle. The arbiter will then see both MRQ+ and UPDATE- and generate MACK+.

The result is that Passport will have to wait anywhere between 3 and 8 clock cycles before MACK+ is returned.

Reset Behaviour

The Arbiter Block exhibits the same behaviour for all types of Reset conditions. This is because the Arbiter Block is entirely reset by the assertion of RESET-, which is generated for all forms of Reset. The main arbiter enters the <IDLE12> state, and the following signals are asserted or deasserted according to the following table.

Arbiter Power On State

| SIGNAL NAME | SIGNAL STATE |
|-------------|---|
| Asserted | UPDATE-, BREQ+ |
| Deasserted | MACK+, SELECT-, BUFFER+, STRETCH+, BUF_EN-, LTOA+, SYNC_RD-, SYNC_WR-, HOLD+ |

Basically, the Arbiter is incapable of processing any requests as long as any Reset Condition is present.

Once RESET- has been deasserted, there should be no incoming requests from either the Backplane Adapter or the Processor, and the arbiter should enter <IDLE04> on the next clock cycle.

Power Fail Warning Behaviour

If the adapter enters the Power Fail Wait state (PowerFailState = <010>), due to the assertion of PFW- by the channel, the arbiter will inhibit the assertion of HOLD+ in the presence of MRQ+ until the Power Fail Detecting machine returns to its idle state. Thus, the arbiter will not process and Passport requests during the Power Fail Wait state.

MISCELLANEOUS CIRCUITRY

Channel Supplied Power

The HP27111A has connection for all four D.C. supply voltages from the CIO Backplane. They are Primary Power, Secondary Power, -12 Volt Reference and 12 Volt Reference.

PRIMARY POWER.

Primary Power comes from the +5P connections on CIO {P1-A40, B40}. Before it is distribute through the board it passes through a 4 Amp fuse, F1. At this point it is immediately bypassed by a 33 uF capacitor C6.

It is used to directly power most of the circuitry on the pca.

A number of individual ceramic bypass capacitors are distributed throughout the card. The 0.01 capacitors include C23 thru C37, C44, C46, C47, C49, C50

SECONDARY POWER.

Secondary Power comes from the +5S connections on CIO {P1-A39, B39}. Before it is distributed to the card, it passes through a 0.5 Amp fuse F4 after which it is immediately bypassed by a 22 uF capacitor C11.

The Secondary Power trace is then routed to the Power Shunt Jumper U13, described in the Processor block. There are three capacitors, C8, C24, C25, that are used to bypass the three components {U17, U49, U50} connected to the internal secondary power bus.

12 VOLT REFERENCE.

The 12 Volt Reference comes from the +12 connections on CIO {P1-A38, B38}. Before it is distributed to the card, it passes through a 0.5 Amp fuse F2 after which it is immediately bypassed by a 6.8 uF capacitor C14.

The 12 Volt Reference is used to power only the VDL Reference {Q7} regulator and the operational amplifier portion of the DC-DC Converter {U42}, where it is bypassed by C29.

-12 VOLT REFERENCE.

The -12 Volt Reference comes from the -12 connections on CIO {P1-A37, B37}. Before it is distributed to the card, it passes through a 0.5 Amp fuse F3 after which it is immediately bypassed by a 6.8 uF capacitor C7.

The sole purpose of this reference is for use with the VBG Reference {Q2}.

GROUND RETURN.

The following backplane pins are used for Ground Return on the HP27111A:
P1-A4, A7, A10, A13, A16, A21, A23, A27, A34, B4, B7, B10, B13, B16, B21, B23, B27, B34.

FRAME GROUND.

The Frame Ground backplane connections {*P1-A1, B1*} are used as an Electrostatic Discharge return path by the pca. The unused pins of the optical connectors, *P2, P3* are connected to Frame Ground.

NMOS Voltage Supplies

VDL REFERENCE.

The VDL reference is used by PASSPORT and PRONTO as a clock reference voltage. Its nominal value must be 2.76 Volts and the maximal current draw is 15 mA. The reference is derived by using an LM317L three-terminal voltage regulator {Q1}.

The input {*Q1-3*} is connected to the +12 volt backplane supply. The output and common connections {*Q1-2, 1*} are connected through resistor divider *R1, R3* to GND. Since the LM317L regulates such that the output to common differential is 1.25 Volts, an equation can be derived to select *R1, R3* such that {*Q1*}s output is 2.76 Volts relative to ground.

Resistor *R2* is used for load regulation. The LM317L specifies operation for a minimum load of 10 mA, which for a 2.76 Volt reference is 276 ohms. When *R3* is taken in parallel with the series connection of *R1, R3* it introduces a minimum load of approximately 184 ohms to the regulator satisfying the loading condition.

Total current required by the load regulating resistors and the actual NMOS-III loads is ~30 mA which is significantly less than the maximum specified rating of the LM317L, which is 100 mA.

C2 and *C1* act as bypass and stability capacitors for the regulator.

VBG REFERENCE.

VBG is a backgate bias reference voltage which is applied to the NMOS-III devices' substrates. It requires a nominal voltage of -2.0 Volts and a bias supply current of 2 mA.

The reference is derived from the -12 Volt backplane supply by an LM337 three-terminal negative voltage regulator {Q2}. The input {*Q2-3*} is connected to the -12 volt backplane supply. The output and common connections {*Q2-2, 1*} are connected through resistor divider *R4, R6* to GND. Since the LM337

Theory of Operation

regulates such that the output to common differential is -1.25 Volts, an equation can be derived to select R_6 , R_4 such that $\{Q2\}$'s output is -2.03 Volts relative to ground.

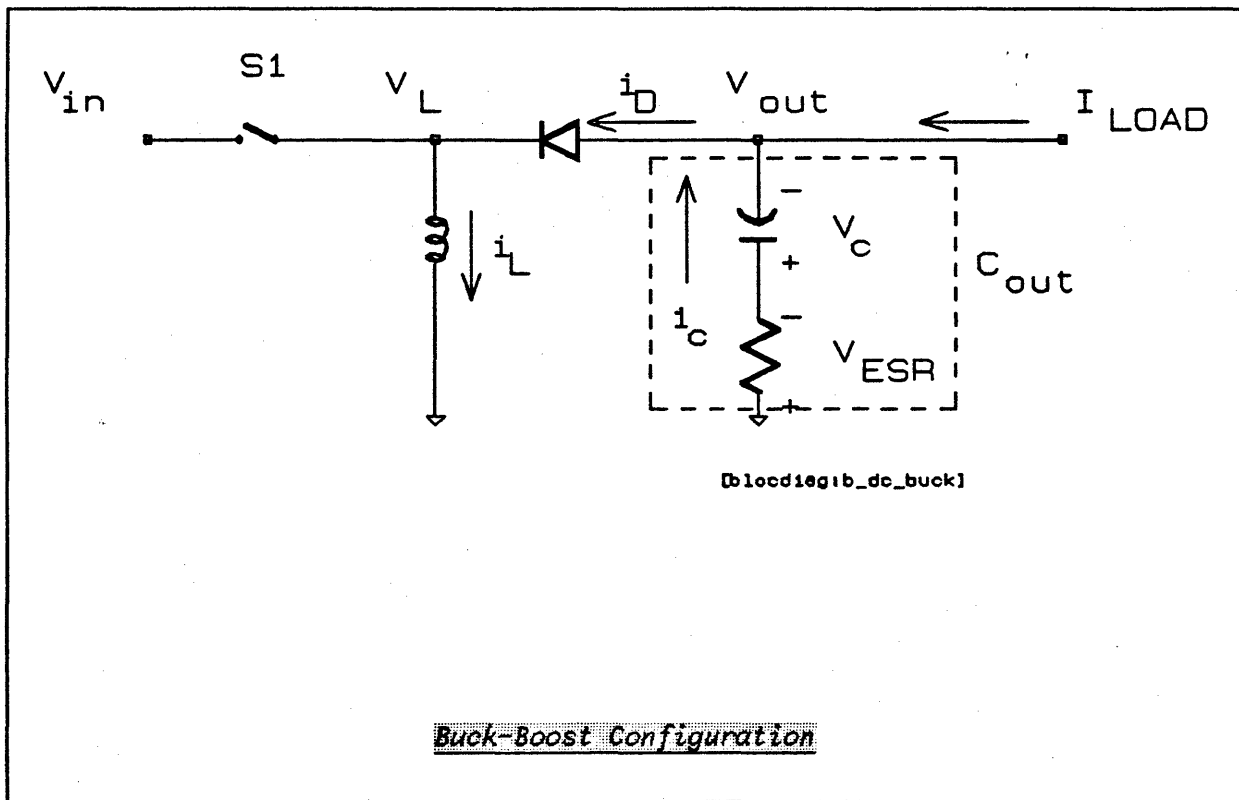
Resistor R_5 is used for load regulation. The LM337L specifies operation for a minimum load of 10 mA, which for a -2.03 Volt reference is approximately 200 ohms. When R_5 is taken in parallel with the series connection of R_4 , R_6 it introduces a minimum load of approximately 160 ohms to the regulator, satisfying the loading condition.

Total current required by the load regulating resistors and the actual NMOS-III loads is ~15 mA which is significantly less than the maximum specified rating of the LM337L, which is 1.5 A.

C_4 and C_3 act as bypass and stability capacitors for the regulator.

DC-DC Converter Basic Operation

This switching regulator uses what is known as a Buck-boost configuration. Its functional diagram is shown in the figure below.

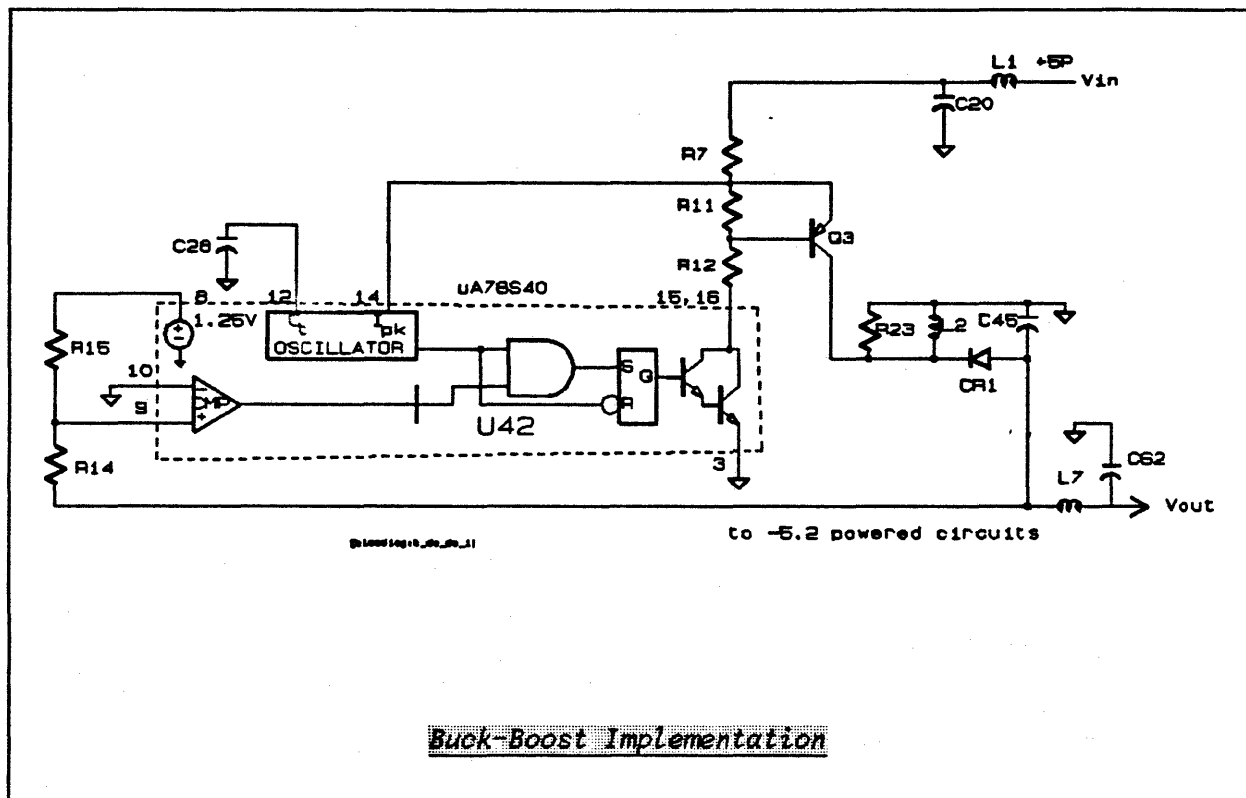


The (constant) load current delivers charge to C_{out} . This makes V_{out} more positive. This effect can be offset, in order to keep V_{out} constant, by conducting charge away thru the diode. The current thru the diode is determined by the current thru the inductor which depends on the switching HISTORY of S_1 . The duty cycle and frequency of S_1 is controlled by the uA78S40 U42 such as to keep V_{out} constant.

For small loads, the regulator accomplishes the required charge balance by delivering triangular current pulses of large amplitude but short duration during some cycles and small amplitude but long duration during other cycles. The various circuit waveforms are then unsynchronized to the local oscillator and it is difficult to take scope pictures.

For moderate to large load currents a current limiting resistor causes synchronization to occur as will be explained later, and scope pictures are then feasible.

The circuitual implementation of the Buck-boost configuration is shown below.



In this case, the role of the switch, S1, is performed by the PNP transistor Q3, and R7 is the current limiting resistor. The output charge storage capacitor, C_{out} is formed by C45.

ARCHITECTURE OF THE UA78S40.

The functional blocks relevant to this design are described below.

Current-Controlled Oscillator. Generates the gating signals used to control the on/off condition of the transistor power switch. The oscillator frequency is set by a single external capacitor and may be varied over a range of 100 Hz to 100KHz. The oscillator duty cycle is internally fixed at 6:1, but may be modified by the current-limiting circuit.

Theory of Operation

Temperature-Compensated, Current-Limiting Circuit. Senses the switching transistor current across an external resistor and may modify the oscillator on-time, which in turn limits the peak current. This provides protection for the switching transistor and power diode. It also decreases the range of inductor current possible and in so doing favors the mode of operation that provides for some conduction on every oscillator cycle. The nominal activation voltage is 300mv, and the peak current can be programmed by a single resistor *R7*.

Temperature Compensated Voltage Reference. A 1.25v temperature-compensated, band-gap voltage source provides a stable reference to which the output voltage is compared.

High-Gain Differential Comparator. Used to inhibit the basic gating signal generated by the oscillator from turning on the transistor switch when the output voltage is too high. The common-mode input range extends from ground to 1.5V less than V_{cc} .

Transistor Switch. A darlington switch that can switch 1.5A peak current in 500ns.

High-Gain Amplifier. Uncommitted op-amp (*not shown*) similar to the uA741 but with a power output stage that can sink 35mA and source 150mA. The input stage has also been modified to include ground in its common-mode range. This op-amp is used in the FOX transmitter circuit.

OPERATION OF THE uA78S40.

The oscillator runs continuously at a rate controlled by the selected external timing cap, *C28*, the internal charging/discharging current sources, and the internally set oscillator threshold voltages and their hysteresis. During the positive going ramp of voltage on the timing capacitor, the output flip-flop will be set, and consequently, the output switch transistor will be turned on, and current will begin to flow in the switch transistor provided that the comparator output is high (logic one). If the comparator output is low (logic zero) during the positive ramp of the oscillator, the output flip-flop will not be set and current will not flow in the switch transistor. Note that the oscillator will continue to ramp up and down even if the output flip-flop and switch are inhibited by the comparator. If the comparator changes from low to high during the positive-going ramp of the oscillator, the flip-flop will be immediately set, initiating a "partial" on interval for the output switch transistor. The flip-flop is always reset during the negative-going ramp of the oscillator. Note also that once set during a positive-going ramp of the oscillator (and comparator "high"), the flip-flop will remain set until the negative-going ramp portion of the oscillator cycle. Thus, the comparator change from low to high can initiate a switch "on" condition but not terminate it. A comparator output low state (normally indicating an output voltage greater than the set point) can inhibit a portion of one cycle, an entire cycle, more than one cycle, or more than one cycle plus a portion of one cycle. Thus, the uA78s40 may be considered a variable pulse-width, variable frequency modulator. The current limit function is internally implemented by the injection of increased charging current into the timing capacitor, causing rapid attainment of the positive-going oscillator threshold. The oscillator frequency thus increases with the load. Therefore, operation during short circuit conditions consists of a very short (but finite) "on" interval followed by a normal minimum "off" interval provided by the negative-going ramp of the oscillator.

MODES OF OPERATION.

The regulator has several operating modes. Which one is invoked depends on the output loading as described below.

MODERATE load. See figure[waveforms for moderate load] Assume that the regulator is in regulation, that V_{out} is $-5.2v$, and V_{in} is equal to $5v$. $S1$ is initially open ($Q3$ is off) and the diode $CR1$ is not conducting. The current in the inductor $L2$ is therefore zero (no energy stored). The voltage is also zero. When switch $S1$ closes ($Q3$ is on), the diode becomes reverse biased so that the current thru $L2$ is entirely from V_{in} . The voltage across $L2$ is approximately equal to V_{in} and its current, i_L therefore increases from zero linearly with time at the rate of V_{in}/L . The energy stored in the inductor therefore increases.

The opening/closing of $S1$ is controlled by a feedback loop that senses the amount of deviation of the output voltage from its desired value.

When $S1$ opens, $L2$ resists a change in the value and direction of its current. v_L goes negative and the diode becomes forward biased and conducts the full inductor current. The voltage across $L2$ is approximately equal to V_{out} (less one diode drop). The inductor current therefore decreases at the rate V_{out}/L . The inductor current reaches zero before $S1$ turns on. The inductor is then floating and its voltage goes to zero with an underdamped second order response (oscillatory).

In this mode waveforms are periodic even though the oscillator is free-running and its output would not normally be synchronized to the switching of $S1$. The reason is that there is sufficient current limiting taking place to modify every period of the timing capacitor voltage. The oscillator on time is shortened until it equals the off time at the designed for current. If the comparator demands more current into $L2$ "too late" during the on cycle of the oscillator, that cycle will be shortened because current limiting will take place and the timing cap will charge faster. The phase difference becomes smaller. On the other hand, if the comparator demands more current "too early" the phase is not adjusted. In steady state the comparator will demand more current at just the "right" time. Without current limiting it would be possible to compensate for the change in charge in C_{out} caused by the load current by drawing larger currents from C_{out} but for a shorter time. This can be accomplished by skipping oscillator cycles (without turning $S1$ on) and allowing the current in $L2$ to increase until it is large enough to supply the required charge during the (always enforced) oscillator off time. But because the current amplitude is limited, beyond a certain value of load current the only way for the regulator to meet the charge demands is to conduct on every cycle.

VERY LIGHT Loading. In this mode the current does not limit on every cycle. The waveform of the inductor current is unsynchronized to the oscillator. That is $S1$ does not turn on for the same amount of time during each oscillator cycle. In fact during some cycles $S1$ remains off during the entire period. Difficult to get pictures because waveforms non-periodic.

HEAVY Loading. See figure[Waveforms for Heavy Loading] Essentially the same mode of operation as with MODERATE loading. The difference is that the inductor current never reaches zero. That is, $S1$ turns on and inductor current starts to increase before it reaches zero. The inductor voltage never becomes zero (no oscillatory decay to zero). The current thru the output capacitor is such that the output voltage can turn around by itself. That is, the capacitor current decreases to zero with a finite slope. The capacitor voltage can therefore change its direction by itself and rise above $-5.2v$ thus triggering the comparator.

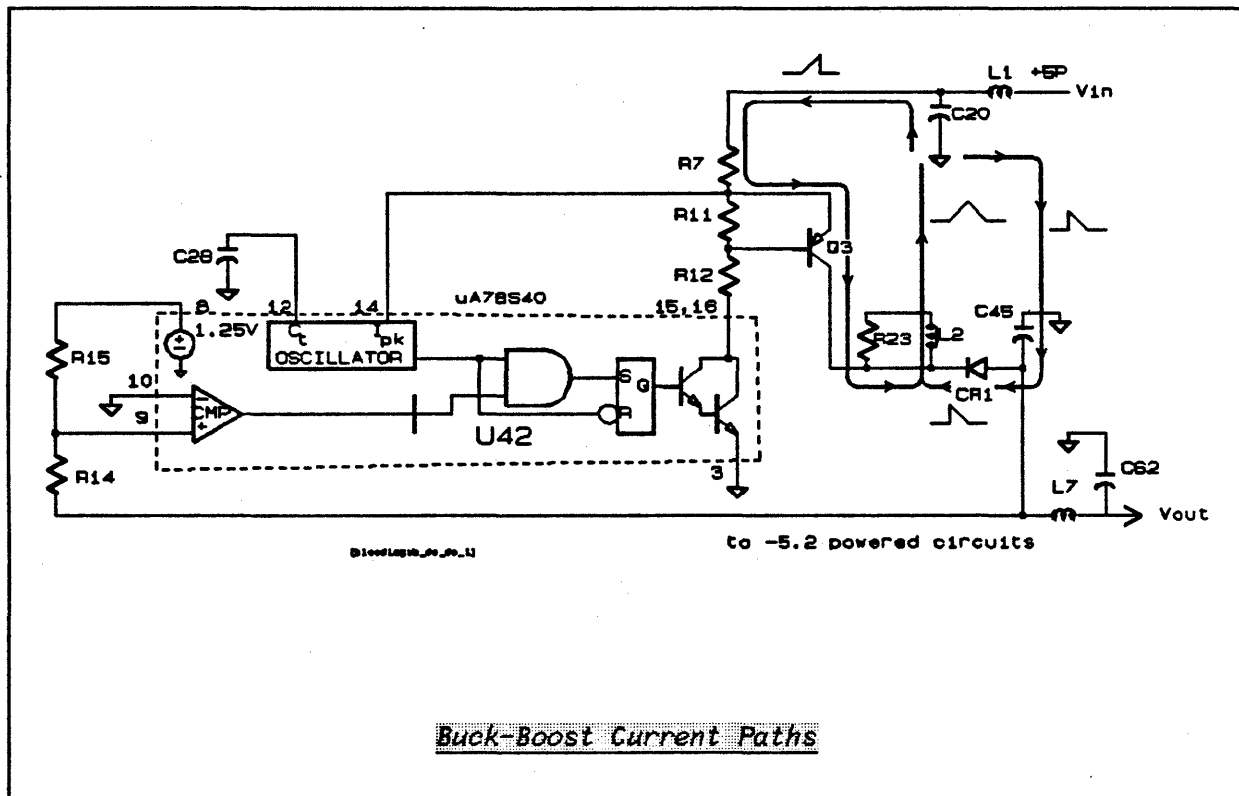
This converter is designed so that at the rated load current the inductor current just reaches zero before it starts rising again.

Theory of Operation

OVERLOAD. See figure [Waveforms for Overload] Essentially the same mode of operation as with **HEAVY LOADING.** The difference is that the current thru the output capacitor decreases to zero with an infinite slope. In this case the capacitor voltage cannot change direction by itself. The only way for the voltage to change direction is for the comparator to demand it. This can only happen if the capacitor voltage is more positive than -5.2v . This means that the capacitor voltage must **ALWAYS** be above -5.2v . The comparator will **ALWAYS** demand more current. The regulator is then operating open loop. Even though the system is out of regulation the output voltage drops very slowly with increased output loading.

LAYOUT CONSIDERATIONS.

The figure below shows the paths taken by the current pulses of largest magnitude.



The layout is designed to prevent the large currents from flowing thru ground impedances that are in common with the low-level sections of the circuit. To this end there exists a single point ground at the low end of the input filter capacitors.

The area of the loops that carry currents w/ large slopes has also been minimized in order to minimize radiation. This was done this by dedicating entire sections of planes as the return paths so that regardless of where the signal flows, the return, at high frequencies, will be on the plane immediately underneath.

To this effect the following components are grounded separately via planes to the single point ground: Inductor and its damping resistor, pin 3 of the 78S40 (the emitter of the internal switching transistor), output filter capacitor.

The current thru the inductor does not have a steep slope and therefore the fact that the inductor is axial and does not contain the flux within its core is not expected to be a problem.

Note that the AC ripple currents recirculate within the DC-DC converter.

SPECIFICATIONS.

Input voltage: 5v, 10% tolerance.

Output voltage: -5.2v, 10% tolerance.

Output current: 450mA

Efficiency: 50%

Output ripple: 180mv peak-peak

Input ripple: 180mv peak-peak

Minimum switching frequency: 20Mhz.

SIGNAL NAME LIST

APPENDIX

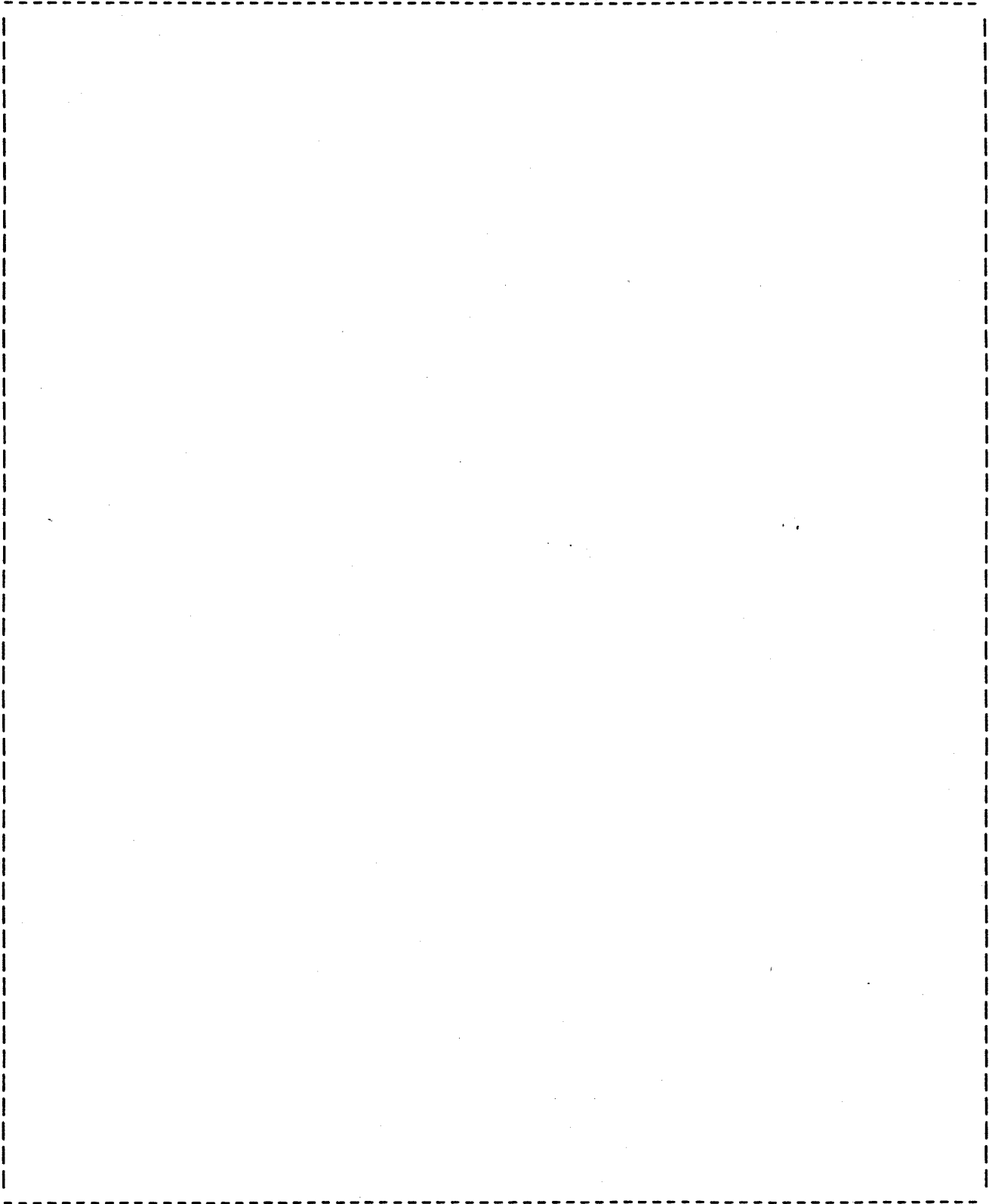
A

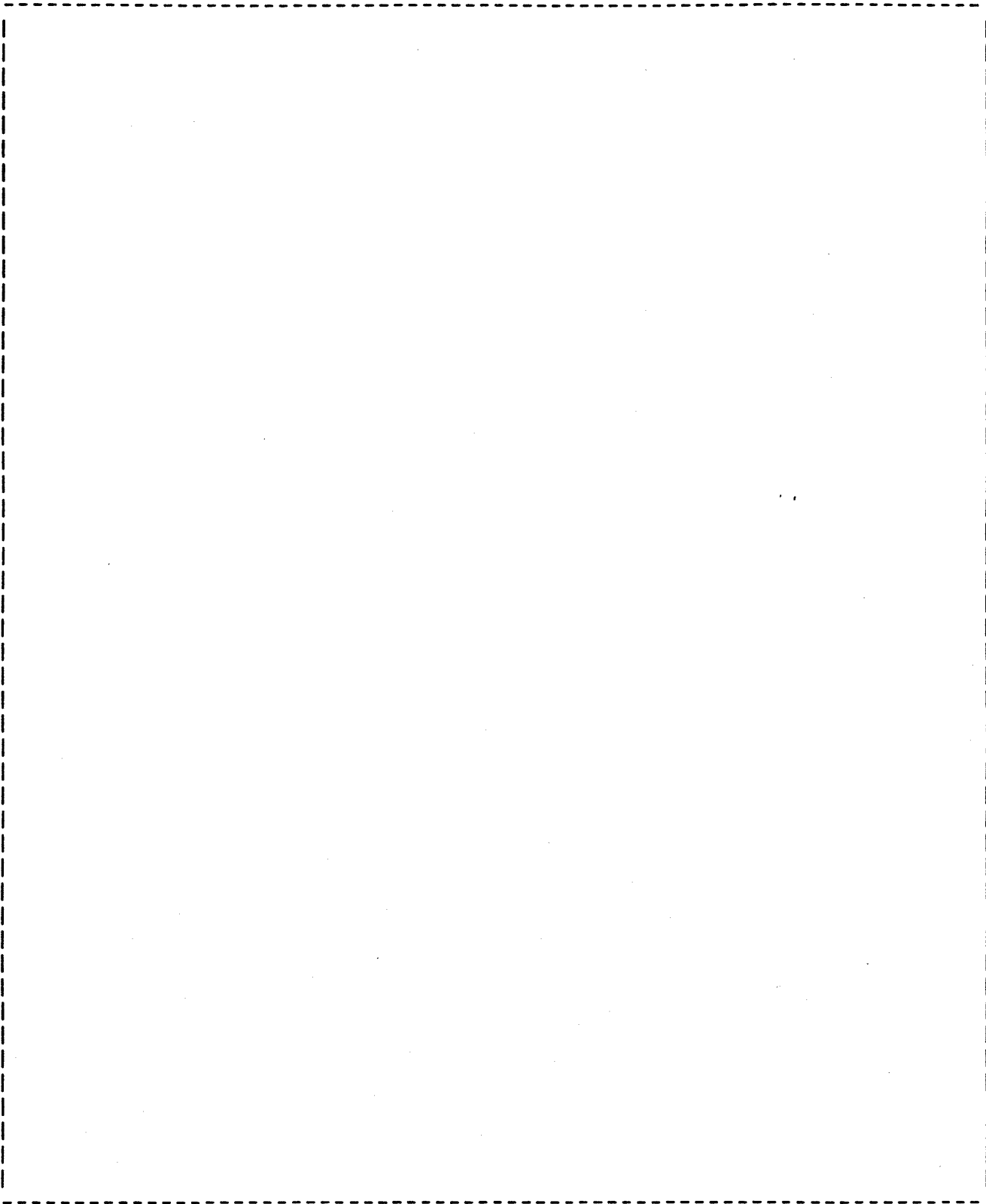
A signal name list will be provided from either the CALAY or 3065 test system dump.

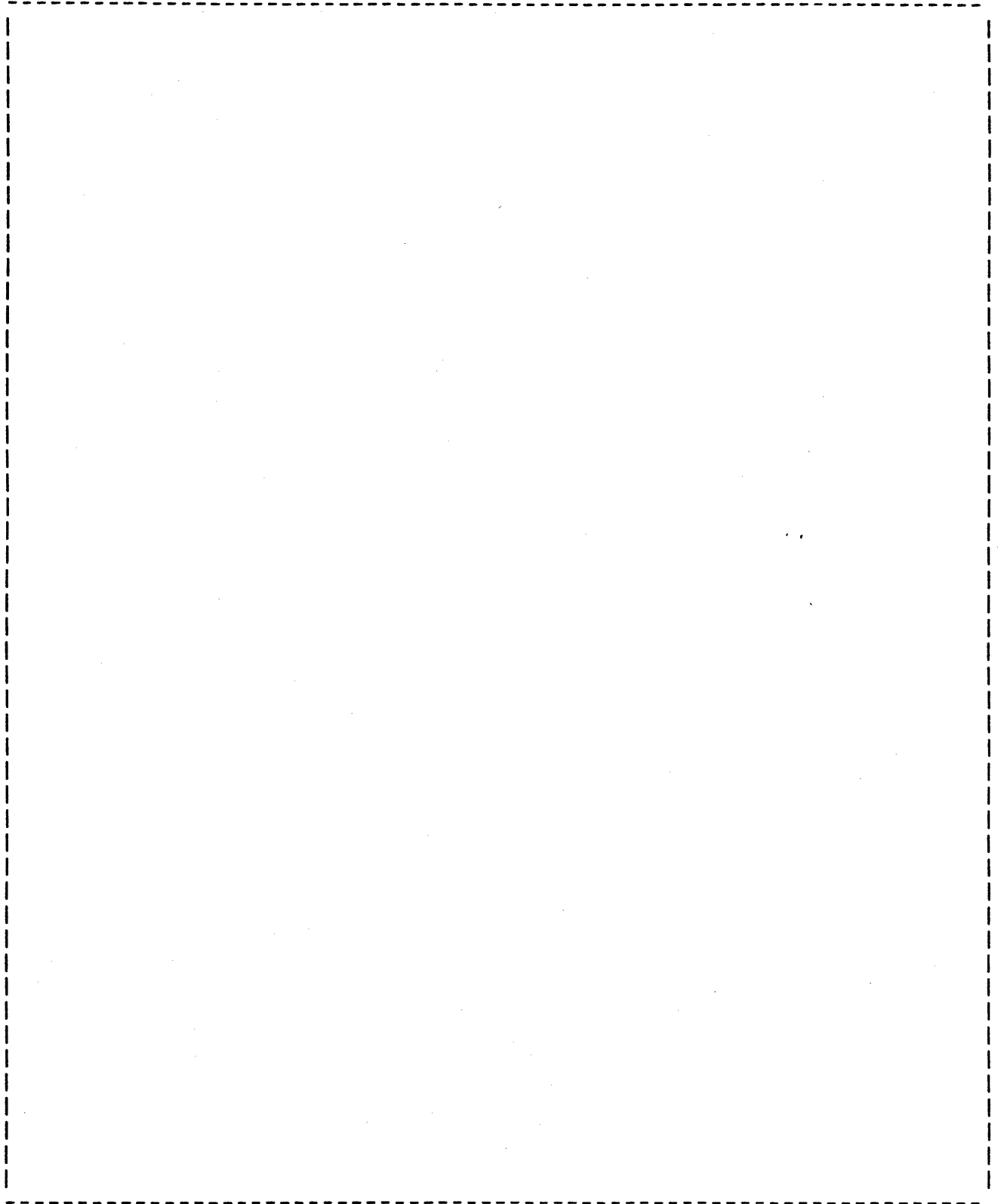
SCHEMATICS

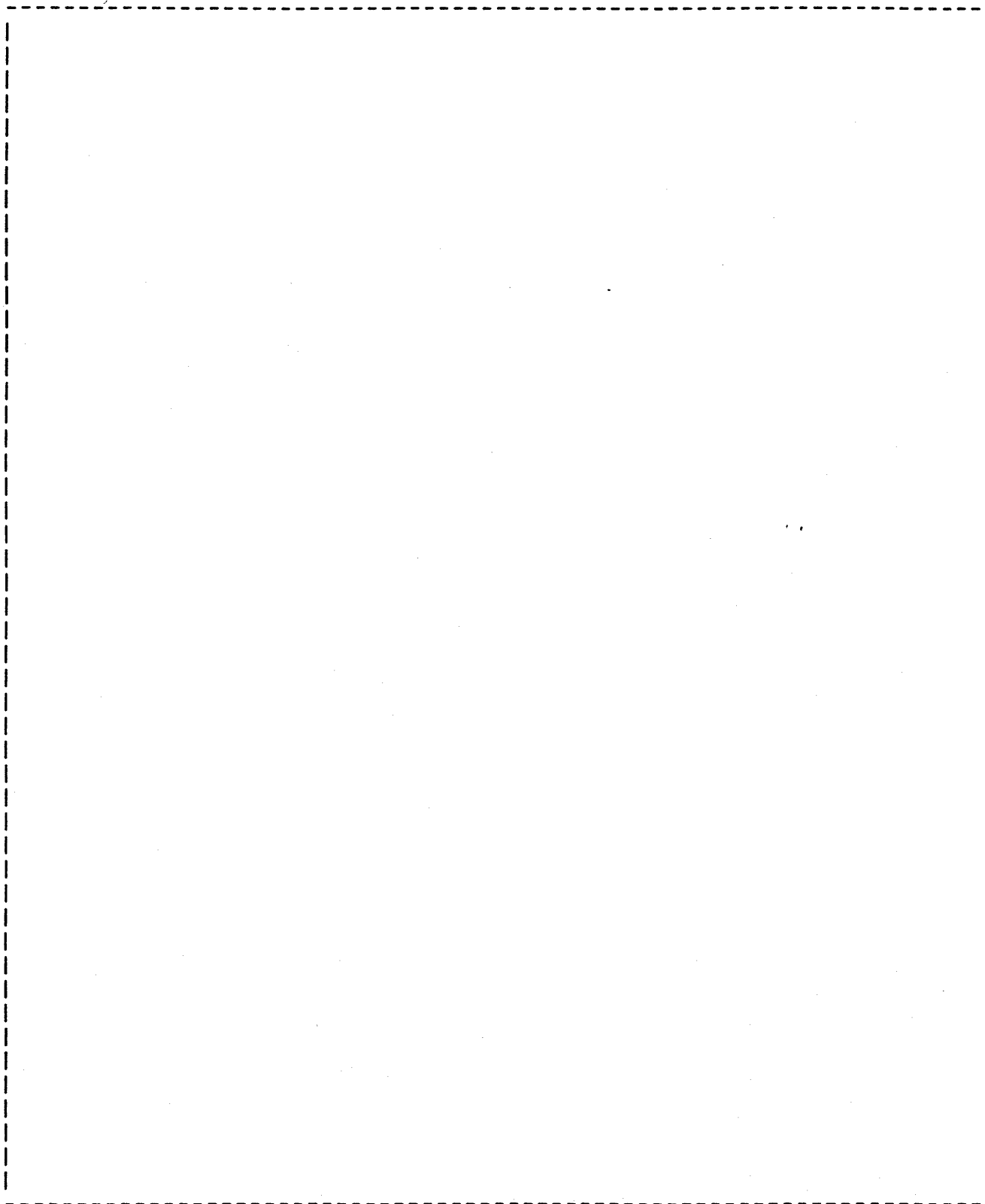
APPENDIX

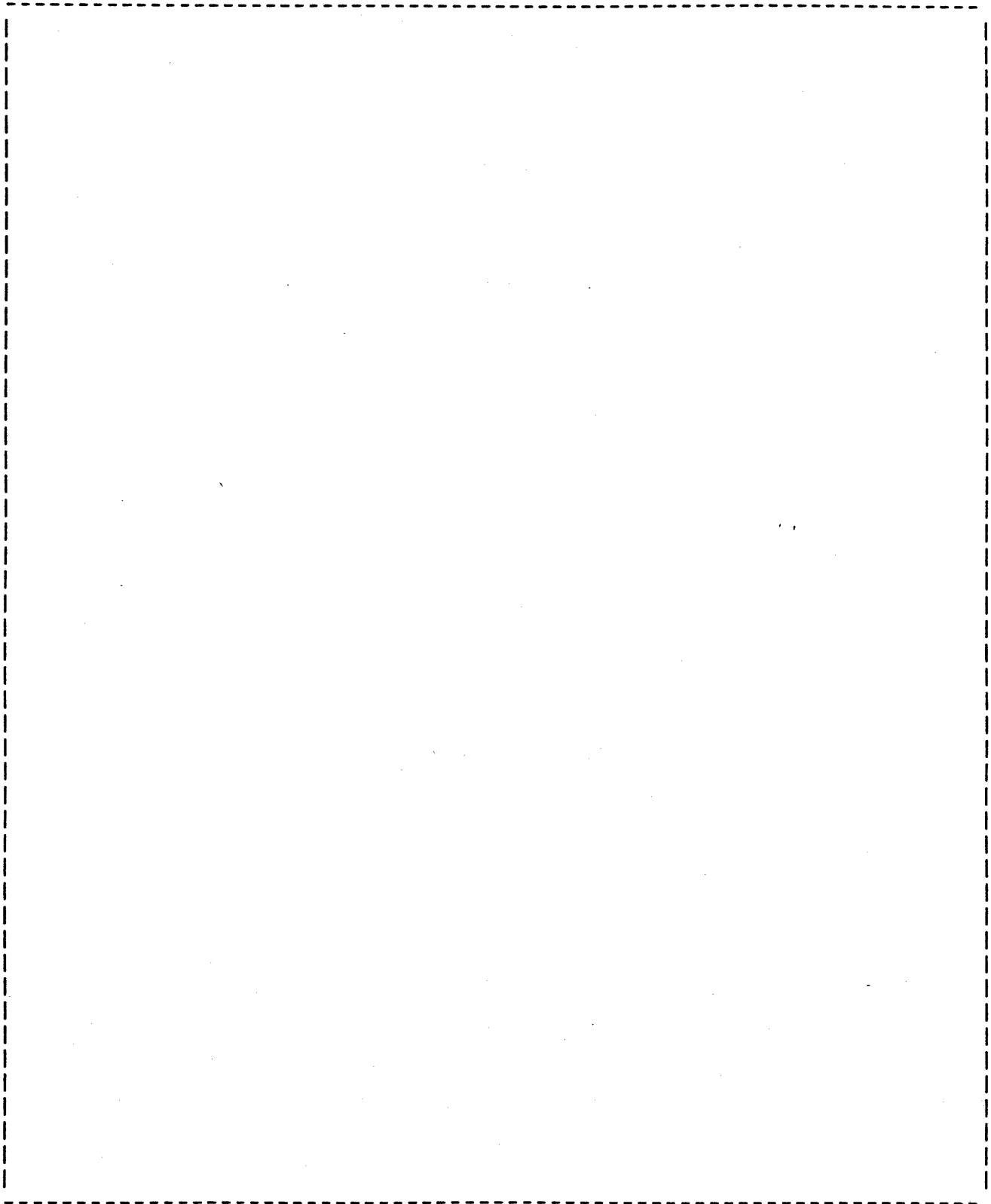
B

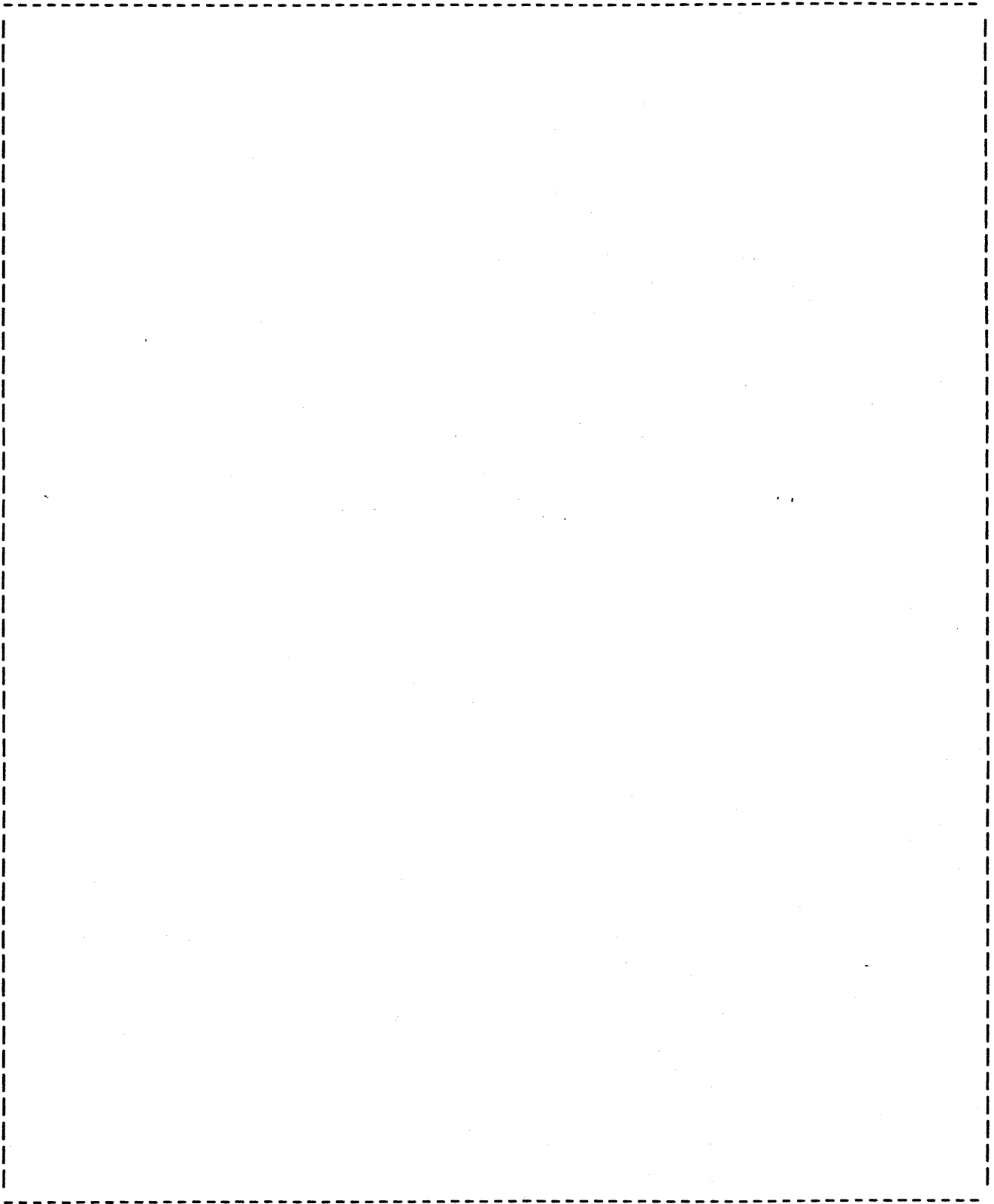












PARTS LIST

APPENDIX

C

MLDB LIST - ASSY:ALINK
 UPDATED BY:GRACE EARL

(FIBER OPTIC CIO)
 ENGINEER:TOM KEAVENY

DATED:01/12/87
 PRINTED:JAN 23, 1987

| STOCK NUMER | QTY | DESCRIPTION-16 | REFERENCE DESG/TEXT |
|-------------|------|-------------------|---|
| 0160-3879 | 16.0 | CAP .01UF100VCER | C32,C38,C40,C53,C54,C55,C56,C57 C58,C64,C65,C66,C67,C68,C69,C70 |
| 0160-4387 | 1.0 | CAP 47PF 5% 200V | C63 |
| 0160-4788 | 3.0 | CAP 18PF 5% 100V | C12,C13,C42 |
| 0160-4789 | 1.0 | CAP 15PF 5% 100V | C52 |
| 0160-4822 | 1.0 | CAP 1000PF 5% CER | C28 |
| 0160-4833 | 1.0 | CAP .022UF 10% | C43 |
| 0160-4835 | 7.0 | CAP .1UF 10% | C1,C2,C5,C16,C19,C39,C48 |
| 0160-6500 | 26.0 | CAP .01UF 10% CER | C23,C24,C25,C26,C27,C29,C30,C31, C33,C34,C35,C36,C37,C44,C46,C47, C49,C50 C8, C9, C10,C15,C17,C18,C21,C22, |
| 0180-0116 | 2.0 | CAP 6.8UF 35V TA | C7,C14 |
| 0180-0228 | .0 | CAP 22UF 15V TA | C11 |
| 0180-0229 | 2.0 | CAP 33UF 10V TA | C6,C62 |
| 0180-0291 | 2.0 | CAP 1.0UF 35V TA | C3,C4 |
| 0180-2208 | 2.0 | CAP 220UF 10V TA | C18,C45 |
| 0180-2690 | 5.0 | C-F 3.3UF 15V TA | C41,C51,C59,C60,C61 |
| 0403-0527 | 2.0 | EXTRACT HANDLE | New extractor handles |
| 0410-1519 | 1.0 | XTAL 16 MHZ | Y1 |
| 0698-0084 | 1.0 | RES 2.15K 1%.125 | R10 |
| 0698-3430 | 2.0 | RES 21.5 1/8W | R32,R33 |
| 0698-3438 | 1.0 | RES 147 1% .125W | R4 |
| 0698-3441 | 1.0 | RES 215 1% .125W | R11 |
| 0698-3442 | 3.0 | RES 237 1% .125W | R3,R5,R6 |
| 0698-3443 | 2.0 | RES 287 1% .125W | R1,R2 |
| 0698-3446 | 2.0 | RES 383 1% .125W | R16,R17 |
| 0698-5137 | 1.0 | RES 46.4 1% .25W | R12 |
| 0698-7188 | 2.0 | RES 10 1% .05W F | R25,R28 |
| 0698-7205 | 2.0 | RES 51.1 1% .05W | R38,R39 |
| 0698-7220 | 1.0 | RES 215 1% .05W | R50 |
| 0698-7223 | 2.0 | RES 287 1% .05W | R48,R49 |
| 0698-7229 | 2.0 | RES 511 1% .05W | R31,R51 |
| 0698-7236 | 4.0 | RES 1K 1% .05W F | R8,R9,R24,R53 |
| 0698-7239 | 1.0 | RES 1.33K | R15 |
| 0698-8823 | 4.0 | RES 8.25 .125W | R29,R30,R35,R36 |
| 0699-0556 | 1.0 | RES 5.11 1% | R42 |
| 0699-0905 | 2.0 | RESISTOR | R40,R41 |
| 0699-1578 | 1.0 | RES 10 .2W | R47 |
| 0757-0200 | 1.0 | RES 5.62K 1%.125 | R14 |
| 0757-0280 | 1.0 | RES 1.0K 1%.125W | R23 |

Ever so preliminary...

Parts List

| | | | |
|-----------|-----|------------------|-------------|
| 0757-0395 | 3.0 | RES 56.2 1% .12W | R43,R44,R45 |
| 0757-0438 | 3.0 | RES 5.11K 1%.125 | R27,R37,R46 |
| 0757-0442 | 1.0 | RES 10K 1% .125W | R26 |

MLDB LIST - ASSY:ALINK
 UPDATED BY:GRACE EARL

(FIBER OPTIC CIO)
 ENGINEER:TOM KEAVENY

DATED:01/12/87
 PRINTED:JAN 23, 1987

| STOCK NUMER | QTY | DESCRIPTION-16 | REFERENCE DESG/TEXT |
|-------------|-----|------------------|-----------------------------|
| 0811-3605 | 1.0 | RES .1 5% .5W PW | R7 |
| 0811-3716 | .0 | 0 OHM RESISTOR | R18,R19,R20 |
| 1005-0055 | 1.0 | FIBER OPT XMITR | P2 TRANSMITTER |
| 1005-0056 | 1.0 | FIBER OPTIC RCVR | P3 RECEIVER |
| 1200-0567 | 2.0 | SKT 28DIP DS | U29,U30 |
| 1250-1737 | 1.0 | SCKT TEST PT | J3 TEST POINT FOX RECEIVER |
| 1251-4670 | 1.0 | POST 1X3 (.100) | J2 (EQL) |
| 1251-7276 | 1.0 | CONN-POST-TP-SKT | P1 BACKPLANE CONNECTOR |
| 1258-0141 | 1.0 | JUMPER REM | J2 (EQL) |
| 1480-0116 | 2.0 | PIN GRV .062X.25 | FOR EXTRACTOR HANDLES |
| 1810-0270 | 1.0 | RES NET 9X680 | R52 |
| 1810-0286 | 2.0 | RESNET15X10KDIP | U35,U37 |
| 1810-0339 | 1.0 | RESNET 270X7 SIP | R22 |
| 1810-0933 | 1.0 | PRG 4 SHUNT DIP | U13 POWER SHUNT NETWORK |
| 1813-0520 | 1.0 | 6.660MHZ CAN | Y2 |
| 1818-3198 | 2.0 | HM6264P-15 | U49,U50 |
| 1820-1320 | 2.0 | IC 10216 CERAMIC | U47,U68 |
| 1820-2724 | 2.0 | IC SN74ALS573N | U27,38 |
| 1820-2758 | 1.0 | IC 74ALS109 P | U22 |
| 1820-2951 | 1.0 | IC 74ALS240 P | U16 |
| 1820-3121 | 2.0 | IC 74ALS245 P | U25,U26 |
| 1820-3318 | 1.0 | IC 74ALS273 P | U28 |
| 1820-3707 | 1.0 | IC-74ALS541 | U18 |
| 1820-3729 | 1.0 | 74AS02 | U36 |
| 1820-4559 | 1.0 | PROG PAL16R4LDC | U17 MEMORY CONTROL PAL |
| 1820-4560 | 1.0 | PROG PAL16R6LDC | U23 PFW CONTROL PAL |
| 1820-4561 | 1.0 | PROG PAL16R8 | U15 ARBITER PAL |
| 1820-4621 | 1.0 | iAPX-186 uP | U20 |
| 1826-0276 | 1.0 | IC 78L05A +5 REG | Q4 |
| 1826-0603 | 1.0 | UA 78S40 | U42 |
| 1826-0772 | 1.0 | IC (LM317L)TO-92 | Q1 |
| 1826-1099 | 1.0 | LM337L NEG V REG | Q2 |
| 1853-0398 | 1.0 | XSTR PNP SI | Q3 (MJE-210) |
| 1858-0100 | 2.0 | XSTR ARY 14P-DIP | U55,U56 for FOX TRANSMITTER |
| 1901-0782 | 1.0 | DIO-1N5821 | CR1 |
| 1990-0968 | 1.0 | LED, RED | CR7 |
| 1990-1025 | 3.0 | LED HLMP-1600 | CR4,5,6 |
| 1990-1027 | 2.0 | LED HLMP-1640 | CR2,3 |
| 1DQ3-0001 | 1.0 | JUPITER TRANSMTR | U43 |
| 1DQ4-0001 | 1.0 | JUPITER RECEIVER | U44 |
| 1FH6-0001 | 1.0 | PRONTO CHIP 1FH6 | U34 |
| 1FH7-0001 | 1.0 | PASSPORT 1FH7 | U14 |
| 2110-0712 | 1.0 | FUSE 4A 125V | F1 |
| 2110-0716 | 2.0 | FUSE .5A 125V | F2,F3,(F4 opt.) |

Ever so preliminary...

MLDB LIST - ASSY:ALINK
 UPDATED BY:GRACE EARL

(FIBER OPTIC CIO)
 ENGINEER:TOM KEAVENY

DATED:01/12/87
 PRINTED:JAN 23, 1987

| STOCK NUMER | QTY | DESCRIPTION-16 | REFERENCE DESG/TEXT |
|----------------|-----|-----------------|----------------------|
| 27111-81001 | 1.0 | 2764 EPROM 8KX8 | U29 PROGRAMMED EPROM |
| 27111-81002 | 1.0 | 2764 EPROM 8KX8 | U30 PROGRAMMED EPROM |
| 5180-7231 | 1.0 | ALINK PC BOARD | BOARD BLANK |
| 9100-3314 | 1.0 | INDCTR 150NH 5% | L4 |
| 9100-3334 | 2.0 | INDCTR 25UH | L1,L7 |
| 9135-0068 | 1.0 | INDCTR 33NH | L8 |
| 9140-0308 | 1.0 | IDCTR .12UH 5% | L3 |
| 9140-0452 | 2.0 | INDCTR 2.7UH 5% | L5,L6 |
| 9140-0835 | 1.0 | TRANSFORMER | T1 |
| 9140-1161 | 1.0 | 50 UH; L; | L2 |
| 9170-1366 | 2.0 | FERRITE BEAD A | FB3,FB4 |
| 9170-1380 | 2.0 | FERRITE BEAD B | FB1,FB2 |
| RESISTOR-SIP A | 1.0 | RESISTOR SIP | R34 |

MLDB LIST - ASSY:ALINK
 UPDATED BY:GRACE EARL

(FIBER OPTIC CIO)
 ENGINEER:TOM KEAVENY

DATED:01/12/87
 PRINTED:JAN 23, 1987

=====
 MATERIAL LIST ANALYSIS --> SUMMARY
 =====

Total number of parts req'd to build assembly: 188
 Total number of different part numbers: 97
 Total cost of the parts required: \$ 290.54
 Material Variance: 10%
 Estimated total MFG labor: \$.00
 MFG Labor Variance/OVHD Rate: 627%
 Est. MFG Cost (matl & var/labor & OVHD): \$ 319.60

Theo. failure rate (RECAP) by part count: 1.33573%/khrs
 74865hrs

Total number of failures in 3mo. warranty (2000hrs): .027

Estimated number of failures/year -- first year: .075
 (1yr = 8000hrs) second year: .053
 third year: .043

Failure Rate based on QA2985 data where available: 1.0248%/khrs
 97583hrs

Failure Rate based on CSY field data where available: %/khrs
 hrs

Of all part info, oldest data last updated: 82/07/16

Parts List

POTENTIAL PROBLEMS FOUND ---> SUMMARY INCOMPLETE?

At least one of the parts is missing cost information!
At least 1 component is missing a theoretical failure rate!
There's at least one NON-PERFERRED/NON-CLASSIFIED part on this list!

=====
User Notes:
ALINK CIO ADAPTER PRODUCTION PROTOTYPE MATERIAL LIST

STATE MACHINE SOURCE FILES

APPENDIX

D

The following procedure was used to generate the PLAs for the Programmable Logic Arrays used on the HP27111A.

A state table description was first written in CALS format. CALS (Computer Aided Logic Synthesis) is a PLA compiler.

The output of the CALS program was reformatted into what is known as "PALASM" format (i.e., PAL assembler). This is an older format which is attributable (supposedly) to Monolithic Memories Inc.

The PALASM formatted equations were once again reformatted into ABEL (Data I/O) format for use with DATA I/O based programming systems.

For more information on ABEL consult the DATA I/O ABEL Manual. For more information on CALS consult the CALS manual.

CALS SOURCE FILES

Arbiter PAL: U15

\$SEQMAC 040002 [arb1p4s1.tak] Sync Read&Write Machine {6/13/86}

RESET DTR ONSTB OFFSTB

SYNCWR SYNCRD

*I RESET/1XXX

 IDLE/0X0X

 GOREAD/001X

 GOWRITE/011X

 WAIT/0XX0

 RETURN/0XX1

*O

*S XREAD/01

 XWRITE/10

 XIDLE/00

*TABLE

 XIDLE RESET/XIDLE/
 IDLE/XIDLE/
 GOREAD/XREAD/
 GOWRITE/XWRITE/.

 XREAD RESET/XIDLE/
 WAIT/XREAD/
 RETURN/XIDLE/.

 XWRITE RESET/XIDLE/
 WAIT/XWRITE/
 RETURN/XIDLE/.

*END

\$LFSR

\$PRINTE

\$SEQMAC 060001 [arb1p4s1.tak] A to B machine

RESET PROSEL MRQ UPDATE DTR RW

ATOB

*I RESET/1XXXXX

 NRESET/0XXXXX

 PCHNGAB/01X10X

 PCHNGBA/01X11X

 PNOCHNG/01X0XX

 MCHNGAB/0011X1

 MCHNGBA/0011X0

 MNOCHNG/0010XX

 BUSIDLE/000XXX

*O

*S ATOB/0

Ever so preliminary...

```

BTOA/1
*TABLE
  ATOB  PNOCHNG/ATOB/
        MNOCHNG/ATOB/
        RESET/ATOB/
        PCHNGAB/ATOB/
        MCHNGAB/ATOB/
        PCHNGBA/BTOA/
        MCHNGBA/BTOA/
        BUSIDLE/ATOB/.

  BTOA  PNOCHNG/BTOA/
        MNOCHNG/BTOA/
        RESET/ATOB/
        PCHNGAB/ATOB/
        MCHNGAB/ATOB/
        PCHNGBA/BTOA/
        MCHNGBA/BTOA/
        BUSIDLE/BTOA/.

*END
$LFSR
$PRINTE
$SEQMAC 080205 [arb1p4s1.tak] main arbiter {10/9/86}
  RESET PROSEL MRQ  HLDA  WRITE  READ  RW  DTR
  OFFSTB ONSTB
  UPDATE BREQ PRONTO BUFFER MACK
*I  RESET/1XXXXXXXX
    NTRESET/0XXXXXXXX
    PROIDLE/00XXXXXXXX
    PROWRIT/01XXXXXX1
    PROREAD/01XXXXXX0
    PRORDWT/01XXX0X0
    PRORDGO/01XXX1X0
    RDCYCLE/0XXXX1XX
    WRCYCLE/0XXXX0XX
      ATOB/0XXXXXXXX
      BTOA/0XXXXXXXX
      IDLE/000XXXXXX
    MEMREQ/0010XXXX
    MWRGRNT/0011XX0X
    MRDGRNT/0011XX1X
    MRQRDWT/0XXXX0XX
    MRQWRWT/0XXX00XX
    MRQWRGO/0XXX10XX
    MRQRDGO/0XXXX1XX
    HLDLOOP/0XX1XXXX
    HLDDONE/0XX0XXXX
      ANY/XXXXXXXXXX
*S  IDLE04/11011
    PRON28/00011
    PRON18/01101
    PRON08/10111
    PRON26/00101
    PRON24/00111

```

State Machine Source Files

```

PRON16/01111
IDLE12/10011
PRON10/10101
PRON20/01011
MEM22/01001
MEM23/01000
MEM20/01011
MEM31/00000
*O SETSTB/01
  CLRSTB/10
  NORMAL/00
*TABLE
  IDLE04      IDLE/IDLE04/NORMAL
              PROWRIT/PRON28/NORMAL
              PRORDWT/IDLE04/NORMAL
              PRORDGO/PRON28/NORMAL
              MEMREQ/IDLE04/NORMAL
              MRDGRNT/MEM31/NORMAL
              MWRGRNT/MEM23/NORMAL
              RESET/IDLE12/NORMAL.

  IDLE12      PROWRIT/PRON28/NORMAL
              PRORDWT/IDLE04/NORMAL
              PRORDGO/PRON28/NORMAL
              PROIDLE/IDLE04/NORMAL
              RESET/IDLE12/NORMAL.

  PRON28      NTRESET/PRON24/SETSTB
              RESET/IDLE12/NORMAL.

  PRON24      NTRESET/PRON26/NORMAL
              RESET/IDLE12/NORMAL.

  PRON26      WRCYCLE/PRON10/NORMAL
              RDCYCLE/PRON08/NORMAL
              RESET/IDLE12/NORMAL.

  PRON10      NTRESET/IDLE04/CLRSTB
              RESET/IDLE12/NORMAL.

  PRON08      NTRESET/IDLE12/CLRSTB
              RESET/IDLE12/NORMAL.

  MEM31       MRQRDWT/MEM31/NORMAL
              MRQRDGO/MEM23/NORMAL
              RESET/IDLE12/NORMAL.

  MEM23       MRQWRWT/MEM23/NORMAL
              MRQWRGO/MEM20/NORMAL
              MRQRDGO/MEM22/NORMAL
              RESET/IDLE12/NORMAL.

  MEM22       NTRESET/MEM20/NORMAL
              RESET/IDLE12/NORMAL.

```

Ever so preliminary...

MEM20 HLDLOOP/MEM20/NORMAL
 HLDDONE/IDLE04/NORMAL
 RESET/IDLE12/NORMAL.

*END
\$LFSR
\$PRINTE
\$EOJ

State Machine Source Files

\$EQUATN 080400 [arbnxt03.tak] plug'n'chug

UPDATE RESET BREQ PRONTO MACK SYNCWR DTR SYNCRD OFFSTB ONSTB DSYNCWR DSYNCRD
DSYNCWR = -RESET*-OFFSTB*SYNCWR + -RESET*DTR*ONSTB*-SYNCWR*-SYNCRD.

DSYNCRD = -RESET*-OFFSTB*SYNCRD + -RESET*-DTR*ONSTB*-SYNCWR*-SYNCRD.

OFFSTB = -RESET*UPDATE*PRONTO.

ONSTB = -RESET*-UPDATE*-BREQ*-PRONTO*MACK.

\$LFSR

\$PRINTE

\$EOJ

Device Clear/Memory Control PAL: U17

```
$SEQMAC 030103 DCL/NMI state machine
CLKRST LOGRST ARMED
NMI
DCLINT WAIT Q1
```

[alidcls2]

*I

```
  RESET/1XX
  CLEAR/01X
  QUIET/00X
  ARMNMI1/001
  ARMNMI2/011
  ARMWT1/000
  ARMWT2/010
```

*O

```
  N/1
  DCL/0
```

*S

```
  POWERON/111
  IDLE/100
  ARMED/101
  QUEUE/110
  NMI1/010
  NMI2/001
```

*TABLE

```
  POWERON  RESET/POWERON/N
           CLEAR/POWERON/N
           QUIET/IDLE/N.

  IDLE     RESET/POWERON/N
           CLEAR/QUEUE/N
           ARMWT1/IDLE/N
           ARMNMI1/ARMED/N.

  QUEUE    RESET/POWERON/N
           ARMWT1/QUEUE/N
           ARMWT2/QUEUE/N
           ARMNMI1/NMI1/N
           ARMNMI2/NMI1/N.

  ARMED    RESET/POWERON/N
           QUIET/ARMED/N
           CLEAR/NMI2/N.

  NMI2     RESET/POWERON/DCL
           QUIET/POWERON/DCL
           CLEAR/POWERON/DCL.

  NMI1     RESET/POWERON/DCL
           QUIET/POWERON/DCL
           CLEAR/POWERON/DCL.
```


State Machine Source Files

```
*END
$LFSR
$PRINTE
$SEQMAC 040001 ARMSTB machine      [alidcls2]
CLKRST PCS3 WR POWERON
ARMED
*I
  ANY/XXXX
  RESET/1XXX
  NTRST/OXX0
  NOPCS/00XX
  NOWR/010X
  ARMIT/011X
  PWRON/OXX1
*O
*S
  IDLE/0
  ARM/1
*TABLE

  IDLE  RESET/IDLE/
        NOPCS/IDLE/
        NOWR/IDLE/
        ARMIT/ARM/.

  ARM   RESET/IDLE/
        PWRON/IDLE/
        NTRST/ARM/.
```

```
*END
$LFSR
$PRINTE
$EOJ
{
```

Modified: Wed, Jul 2, 1986, 3:44 PM
Reason: Combined Global and Clken into one signal -- GCLKEN ;
Added "A0" to literal term combinations ;

Modified: Mon, Jul 7, 1986, 11:46 AM
Reason: Changed GCLKEN to a new signal ARMSTB, because the GCLKEN
signal cannot be asserted during logic reset. The ARMSTB
signal is generated in the same fashion as GCLKEN but
is reset only on CLKRST ;
Removed "A0" from literal term combinations ;

Modified: Fri, Jul 25, 1986, 12:21 PM
Reason: After discussion with Mike Leclere and analysis of the
state transitions defined by CALS it was found that
the asynchronous behaviour of the LOGIC RESET- signal
could cause disfunction of the machine.

Additional states were added to allow only single bit
transitions in areas where the onset of LOGIC RESET-
is critical

```

case STATE of
  POWER_ON : begin { stay here until reset activity is complete }
    if ~CLK_RST*~LOGIC_RESET then STATE := IDLE
    else STATE := POWER_ON
    end ;
  IDLE : begin { watch for a Logic Reset without a Clock Reset !}
    if CLK_RST then STATE := POWER_ON else
    if LOGIC_RESET then STATE := QUEUE
    if ~LOGIC_RESET*ARMSTB then STATE := ARMED
    end ;
  ARMED: begin { watch for a Logic Reset without a Clock Reset !}
    if CLK_RST then STATE := POWER_ON else
    if LOGIC_RESET then STATE := NMI1
    if ~LOGIC_RESET then STATE := ARMED
    end ;
  QUEUE : begin { queue up the DCL until the processor is ready }
    if CLK_RST then STATE := POWER_ON else
    if ARMSTB then STATE := NMI2
    else STATE := QUEUE
    end ;
  NMI1 : begin { pulse the NMI line and then Restart the machine }
    STATE := POWER_ON
    end ;
  NMI2 : begin { pulse the NMI line and then Restart the machine }
    STATE := POWER_ON
    end ;
}

```

Power Fail/Passport Control PAL: U23

\$SEQMAC 030001 Clock Enable Strobe

[source=PASSCTL5]

RST WRITE PCS0
GCLKEN

*I RESET/1XX
NOWRI/00X
NOSEL/010
WRITE/011

*S ENAB/1
IDLE/0

*TABLE

IDLE RESET/IDLE/
NOWRI/IDLE/
NOSEL/IDLE/
WRITE/ENAB/.

ENAB RESET/IDLE/
WRITE/IDLE/
NOSEL/IDLE/
NOWRI/IDLE/.

*END

\$NRMINM

\$LFSR

\$PRINTE

\$SEQMAC 050001 Passport Control Strobes

[source=PASSCTL5]

RST GCLKEN A3 A2 A1
NCNTL

*I RESET/1XXXX
CTLON/01100
CTLOFF/01101
AX1XMIS/01X1X
AOXXMIS/010XX
AXX1MIS/01XX1
AXXOMIS/01XX0
ENMISS/00XXX

*O

*S CNTL/0
NOCNTL/1

*TABLE

NOCNTL RESET/NOCNTL/
CTLOFF/NOCNTL/
AX1XMIS/NOCNTL/
AOXXMIS/NOCNTL/
CTLON/CNTL/
ENMISS/NOCNTL/ .

CNTL RESET/NOCNTL/
CTLOFF/NOCNTL/
AX1XMIS/CNTL/
AOXXMIS/CNTL/

ENMISS/CNTL/
CTLON/CNTL/ .

```

*END
$NRMINM
$LFSR
$PRINTE
$SEQMAC 050001 Passport INTERRUPT ACKNOWLEDGE STROBE           [source=PASSCTL5]
RST      GCLKEN      A3      A2      A1
NINTA
*I  RESET/1XXXX
    ACKON/01110
    ACKOFF/01111
    AXOXMIS/01XOX
    AOXXMIS/010XX
    ENMISS/00XXX
*S  INTA/0
    NOINTA/1
*TABLE
  INTA      RESET/NOINTA/
            ACKOFF/NOINTA/
            AXOXMIS/INTA/
            AOXXMIS/INTA/
            ENMISS/INTA/
            ACKON/INTA/.

  NOINTA   RESET/NOINTA/
            ACKOFF/NOINTA/
            AXOXMIS/NOINTA/
            AOXXMIS/NOINTA/
            ENMISS/NOINTA/
            ACKON/INTA/.

*END
$NRMINM
$LFSR
$PRINTE
$EOJ

```

State Machine Source Files

\$SEQMAC 060203 POWER FAIL Synchronizer
 RST PFW CLKEN A3 A2 A1
 SYNC_PFW INT
 NQ2 NQ1 NQ0

[source=PWRFAIL5]

```

*I WARNING/01XXXX
  NORMAL/00XXXX
  RESET/1XXXXX
  ACK/0X1001
  GEN/00101X
  MISS2/000XXX
  MISS3/00X1XX
  MISS4/00XXOX
  A1XXMIS/0X11XX
  AX1XMIS/0X1X1X
  AXXOMIS/0X1XX0
  ENMISS/0X0XXX
*O IDLE/11
  INTWARN/00
  WARN/01
*S IDLE/111
  SYNCUP/110
  WARN/000
  SYNCOUT/101
  ACKWARN/100
*TABLE
  IDLE WARNING/SYNCUP/IDLE
        RESET/IDLE/IDLE
        GEN/WARN/IDLE
        MISS2/IDLE/IDLE
        MISS3/IDLE/IDLE
        MISS4/IDLE/IDLE
  SYNCUP WARNING/WARN/IDLE
        RESET/IDLE/IDLE
        NORMAL/IDLE/IDLE
  WARN RESET/IDLE/INTWARN
        ACK/ACKWARN/INTWARN
        A1XXMIS/WARN/INTWARN
        AX1XMIS/WARN/INTWARN
        AXXOMIS/WARN/INTWARN
        ENMISS/WARN/INTWARN.
  ACKWARN WARNING/ACKWARN/WARN
        RESET/IDLE/WARN
        NORMAL/SYNCOUT/WARN.
  SYNCOUT WARNING/ACKWARN/WARN
        RESET/IDLE/WARN
        NORMAL/IDLE/WARN
  
```

```

*END
$NRMINM
$LFSR
$PRINTE
$EOJ
  
```

{ Thu, Jul 10, 1986, 6:55 PM -> Changed sense of SYNC_PFW to internal
 Negative True and external Positive True.

Ever so preliminary...

}

PALASM SOURCE EQUATIONS

The following source equations were generated from the CALS PLA generator running on MPE/V.

Arbiter PAL: U15

PAL16R8
ARBITER PAL U15
[arblp4x2.tak.alink]

CLK /RESET HLDA /PROSEL DTR /READ DIN MRQ /WRITE GND
EN /SYNCRD /SYNCWR /PRONTO /UPDATE /LtoA /BUFFER /BREQ /MACK VCC

```
;  
; Cals source = [arblp4s1][arbnxt04]  
; Cals output = [calsarb4][calsar4s]  
; Wed, Oct 8, 1986, 6:36 PM  
; Thu, Oct 9, 1986, 8:52 AM  
;
```

```
LtoA := /RESET*/PROSEL*/MRQ*LtoA  
      + /RESET*/PROSEL*MRQ*UPDATE*DIN  
      + /RESET*/UPDATE*LtoA  
      + /RESET*PROSEL*UPDATE*DTR
```

```
SYNCWR := /RESET*/PRONTO*SYNCWR +  
          /UPDATE*/RESET*/BREQ*/PRONTO*MACK*DTR*/SYNCRD +  
          /UPDATE*/RESET*SYNCWR
```

```
SYNCRD := /UPDATE*/RESET*/BREQ*/PRONTO*MACK*/SYNCWR*/DTR +  
          /UPDATE*/RESET*SYNCRD +  
          /RESET*/PRONTO*SYNCRD
```

```
UPDATE := RESET +  
          /PROSEL*/HLDA*UPDATE +  
          PRONTO*/BUFFER +  
          /PROSEL*UPDATE*/BREQ +  
          /PROSEL*/MRQ*UPDATE +  
          PROSEL*/READ*/DTR*UPDATE +  
          UPDATE*PRONTO +  
          /HLDA*/UPDATE*BREQ*BUFFER
```

```
BREQ := /RESET*/PROSEL*/DIN*BREQ +  
        /RESET*/PROSEL*/MRQ*UPDATE*/PRONTO +  
        /RESET*PROSEL*/READ*/DTR*UPDATE*/PRONTO +  
        /RESET*/PROSEL*/HLDA*UPDATE*/PRONTO +  
        /RESET*READ*/MACK +  
        /RESET*/PROSEL*UPDATE*/BREQ*/PRONTO +
```

```
    /RESET*UPDATE*/BUFFER +  
    /RESET*/UPDATE*BREQ  
  
PRONTO := /RESET*/UPDATE*/BREQ*MACK  
  
BUFFER := /HLDA*UPDATE +  
          PROSEL*UPDATE +  
          /MRQ*UPDATE +  
          WRITE*/READ*/UPDATE*BREQ +  
          READ*PRONTO*/BUFFER +  
          /UPDATE*/PRONTO*MACK +  
          UPDATE*/BREQ +  
          RESET  
  
MACK   := READ*BREQ*/MACK +  
          WRITE*BREQ*/BUFFER +  
          /UPDATE*MACK +  
          /BREQ*MACK +  
          /HLDA*UPDATE +  
          PROSEL*UPDATE +  
          /MRQ*UPDATE +  
          RESET
```


Device Clear/Memory Control PAL: U17

PAL16R4

Memory Pal & Power on State Machine Source U17
 Fri, Jul 25, 1986, 5:01 PM [mempal7.tak]

SCLK /WR /PCS3 SAO /BHE HLDA /LCS PON /CLKRST GND
 EN /LOGRST /LRS /DCLINT /WAIT /Q1 /ARMED /URS /MCS1 VCC

```

;
; Cals source: [alidcls2.tak.alink]
; Cals output: [calsdcl2.tak.alink]
;
    
```

```

; Lower RAM Strobe
;
LRS = LCS*PON*/SAO*/HLDA ; 80186 Lower Byte 0-7FFFH
+ MCS1*PON*/SAO*/HLDA ; 80186 Lower Byte 18000H - 1FFFFH
+ PON*HLDA ; All Passport accesses
    
```

```

; Upper RAM Strobe
;
URS = LCS*PON*BHE*/HLDA ; 80186 Upper Byte 0-7FFFH
+ MCS1*PON*BHE*/HLDA ; 80186 Upper Byte 18000H - 1FFFFH
+ PON*HLDA ; All Passport accesses
    
```

```

DCLINT:= /DCLINT +
        CLKRST +
        /LOGRST*Q1 +
        /ARMED*WAIT+
        /LOGRST*/ARMED +
        WAIT*Q1 +
        /WAIT*/Q1
    
```

```

WAIT := LOGRST*/Q1 +
        WAIT*/Q1 +
        LOGRST*WAIT+
        /DCLINT +
        CLKRST
    
```

```

Q1 := /LOGRST*ARMED*/WAIT+
        /WAIT*Q1 +
        LOGRST*Q1 +
        /DCLINT +
        CLKRST
    
```

```

ARMED := /CLKRST*ARMED*/DCLINT +
        /CLKRST*ARMED*/WAIT +
    
```

```
/CLKRST*ARMED*/Q1 +  
/CLKRST*PCS3*WR*/ARMED
```

Power Fail/Passport Control PAL: U23

PAL16R6

U12 Power fail and hold request & cntl machine
[PFWPAL6.TAK]

CLK /RST /WRITE PFW A2 /PCSO A1 MRQ A3 GND
EN /HOLD /PFW_INT /INT_ACK /CNTL /GCLKEN /SYNC_PFW /NQO /GLOBWR VCC

```

;
; Cals source = [pwrfail5.tak][passct15.tak]
; Cals output = [calspwr5.tak][calspas5.tak]
;
; Thu, Jul 10, 1986, 7:04 PM : made SYNC_PFW positive true
; Wed, Oct 22, 1986, 3:31 PM : keep MRQ deasserted during Reset
;

```

```

GLOBWR = PCSO*WRITE*/A1*/A2*/A3 ; Global Write Strobe

GCLKEN := /RST*/GCLKEN*WRITE*PCSO ; Clock Enable Pulse

HOLD = /MRQ ; Allow Passport to Request RAM
      + /PFW_INT*SYNC_PFW ; If not waiting for end of Warn
      + RST ; And we are not being reset.

CNTL := /GCLKEN*CNTL ;
      + A2*CNTL
      + /A3*CNTL
      + GCLKEN*A3*/A2*A1
      + RST

INT_ACK := /GCLKEN*INT_ACK
      + GCLKEN*A3*A2*A1
      + /A2*INT_ACK
      + /A3*INT_ACK
      + RST

PFW_INT := RST +
      /PFW*SYNC_PFW*/NQO +
      GCLKEN*/A3*/A2*A1*/PFW_INT +
      /PFW*/A2*SYNC_PFW +
      /PFW*A3*SYNC_PFW +
      /PFW*/GCLKEN*SYNC_PFW +
      PFW_INT*/SYNC_PFW +
      PFW*NQO

SYNC_PFW := /PFW*/SYNC_PFW*NQO +
      /PFW*/A2*NQO +
      /PFW*/GCLKEN*NQO +
      /PFW*A3*NQO +
      PFW*SYNC_PFW*NQO +

```

```
          RST +  
          /PFW*SYNC_PFW*/NQ0  
NQ0 := /PFW*PFW_INT*/NQ0 +  
       /PFW*/SYNC_PFW*NQ0 +  
       /PFW*/A2*NQ0 +  
       /PFW*/GCLKEN*NQ0 +  
       /PFW*A3*NQ0 +  
       RST
```

ABEL FORMATTED EQUATIONS

The following equations were transformed from PALASM format to ABEL format using the TOABEL utility on a DOS based Vectra.

Arbiter PAL: U15

ABEL(tm) Version 2.02a - Document Generator
PAL16R8
ARBITER PAL U15 hp part# : 1820-4561
[arblp4x2.tak.alink]

Page 1
05-Feb-87 09:26 AM

-Translated by TOABEL-
Equations for Module _u15

Device P16R8

Reduced Equations:

```
LTOA := !(LTOA & !MRQ & PROSEL & RESET
# DIN & MRQ & PROSEL & RESET & !UPDATE
# !LTOA & RESET & UPDATE
# DTR & !PROSEL & RESET & !UPDATE);
```

```
SYNCWR := !(PRONTO & RESET & !SYNCWR
# BREQ & DTR & !MACK & PRONTO & RESET & SYNCRD & UPDATE
# RESET & !SYNCWR & UPDATE);
```

```
SYNCRD := !(BREQ & !DTR & !MACK & PRONTO & RESET & SYNCWR &
UPDATE
# RESET & !SYNCRD & UPDATE
# PRONTO & RESET & !SYNCRD);
```

```
UPDATE := !(RESET
# !HLDA & PROSEL & !UPDATE
# BUFFER & !PRONTO
# BREQ & PROSEL & !UPDATE
# !MRQ & PROSEL & !UPDATE
# !DTR & !PROSEL & READ & !UPDATE
# !PRONTO & !UPDATE
# !BREQ & !BUFFER & !HLDA & UPDATE);
```

```
BREQ := !(BREQ & !DIN & PROSEL & RESET
# !MRQ & PRONTO & PROSEL & RESET & !UPDATE
```

```
# !DTR & PRONTO & !PROSEL & READ & RESET & !UPDATE  
# !HLDA & PRONTO & PROSEL & RESET & !UPDATE  
# MACK & !READ & RESET  
# BREQ & PRONTO & PROSEL & RESET & !UPDATE  
# BUFFER & RESET & !UPDATE  
# !BREQ & RESET & UPDATE);
```

```
PRONTO := !(BREQ & !MACK & RESET & UPDATE);
```

```
BUFFER := !(HLDA & !UPDATE  
# !PROSEL & !UPDATE  
# !MRQ & !UPDATE  
# !BREQ & READ & UPDATE & !WRITE  
# BUFFER & !PRONTO & !READ
```

ABEL(tm) Version 2.02a - Document Generator
PAL16R8
ARBITER PAL U15
[arblp4x2.tak.alink]

05-Feb-87 09:26 AM

-Translated by TOABEL-
Equations for Module _u15

Device P16R8

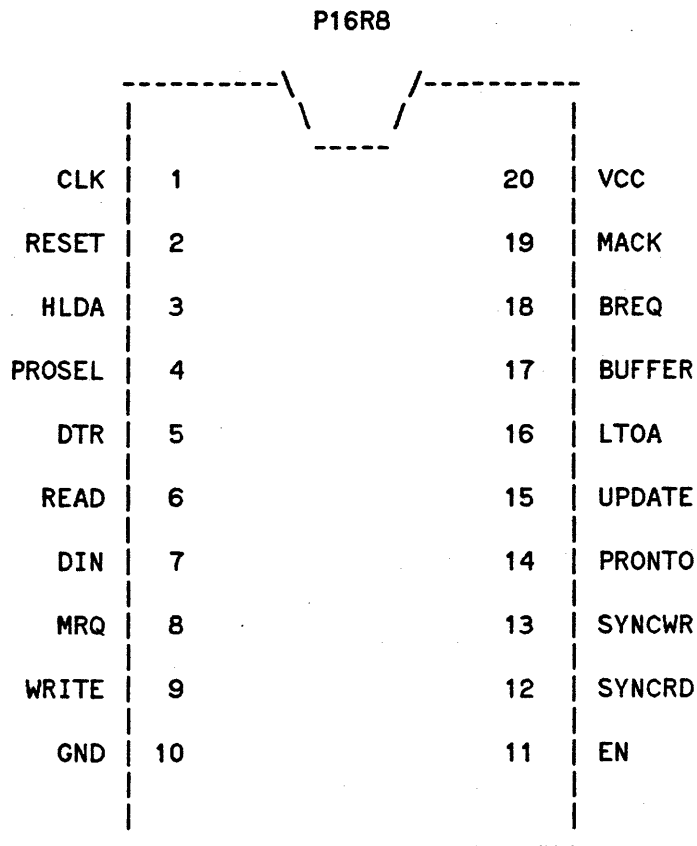
```
# !MACK & PRONTO & UPDATE  
# BREQ & !UPDATE  
# !RESET);
```

```
MACK := !( !BREQ & MACK & !READ  
# !BREQ & BUFFER & !WRITE  
# !MACK & UPDATE  
# BREQ & !MACK  
# !HLDA & !UPDATE  
# !PROSEL & !UPDATE  
# !MRQ & !UPDATE  
# !RESET);
```

ABEL(tm) Version 2.02a - Document Generator
 PAL16R8
 ARBITER PAL U15
 [arblp4x2.tak.alink]

-Translated by TOABEL-
 Chip diagram for Module _u15

Device P16R8



ABEL(tm) Version 2.02a - Document Generator

05-Feb-87 09:26 AM

PAL16R8

ARBITER PAL U15

[arblp4x2.tak.alink]

-Translated by TOABEL-
for Module _u15

Device P16R8

Device Type: P16R8

Terms Used: 43 out of 64

| Pin # | Name | Terms | | Term Type | Pin Type |
|-------|--------|-------|-----|-----------|----------|
| | | Used | Max | | |
| 1 | CLK | -- | -- | --- | Clock |
| 2 | RESET | -- | -- | --- | Input |
| 3 | HLDA | -- | -- | --- | Input |
| 4 | PROSEL | -- | -- | --- | Input |
| 5 | DTR | -- | -- | --- | Input |
| 6 | READ | -- | -- | --- | Input |
| 7 | DIN | -- | -- | --- | Input |
| 8 | MRQ | -- | -- | --- | Input |
| 9 | WRITE | -- | -- | --- | Input |
| 10 | GND | -- | -- | --- | GND |
| 11 | EN | -- | -- | --- | Enable |
| 12 | SYNCRD | 3 | 8 | Normal | Output |
| 13 | SYNCWR | 3 | 8 | Normal | Output |
| 14 | PRONTO | 1 | 8 | Normal | Output |
| 15 | UPDATE | 8 | 8 | Normal | Output |
| 16 | LTOA | 4 | 8 | Normal | Output |
| 17 | BUFFER | 8 | 8 | Normal | Output |
| 18 | BREQ | 8 | 8 | Normal | Output |
| 19 | MACK | 8 | 8 | Normal | Output |
| 20 | VCC | -- | -- | --- | VCC |

end of module _u15

Device Clear/Memory Control PAL: U17

ABEL(tm) Version 2.02a - Document Generator
PAL16R4

Page 1
05-Feb-87 09:29 AM

Memory Pal & Power on State Machine Source U17 hp part# : 1820-4559
Fri, Jul 25, 1986, 5:01 PM [mempal7.tak]
-Translated by TOABEL-
Equations for Module _u17

Device P16R4

Reduced Equations:

```
LRS = !(HLDA & LCS & PON & SAO
      # HLDA & MCS1 & PON & SAO
      # HLDA & PON);
```

```
URS = !(BHE & HLDA & LCS & PON
      # BHE & HLDA & MCS1 & PON
      # HLDA & PON);
```

```
DCLINT := !(DCLINT
            # CLKRST
            # LOGRST & Q1
            # ARMED & WAIT
            # ARMED & LOGRST
            # Q1 & WAIT
            # Q1 & WAIT);
```

```
WAIT := !(LOGRST & Q1
         # Q1 & WAIT
         # LOGRST & WAIT
         # DCLINT
         # CLKRST);
```

```
Q1 := !(ARMED & LOGRST & WAIT
       # Q1 & WAIT
       # LOGRST & Q1
       # DCLINT
       # CLKRST);
```

```
ARMED := !(ARMED & CLKRST & DCLINT
          # ARMED & CLKRST & WAIT
```

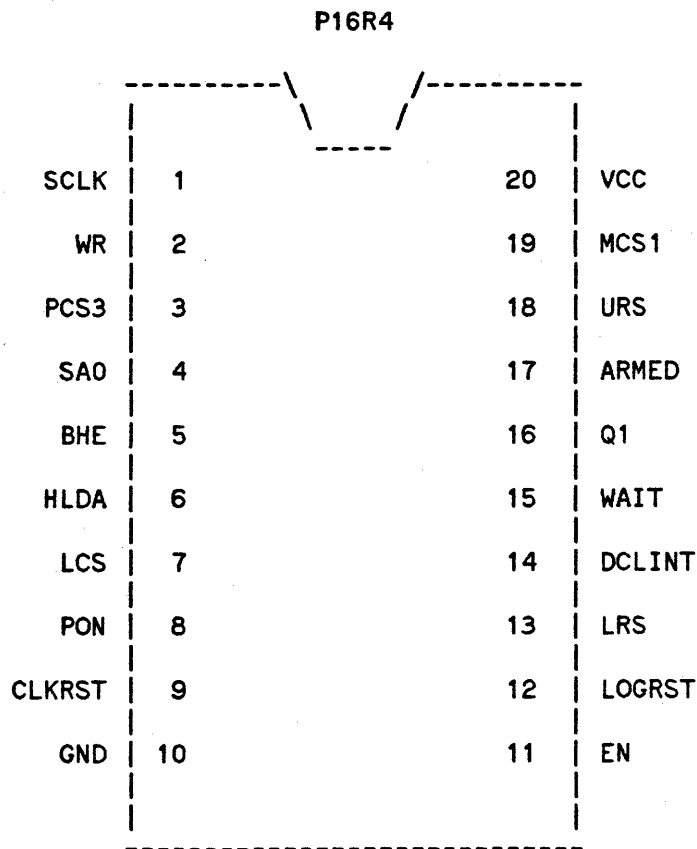
State Machine Source Files

```
# !ARMED & CLKRST & Q1  
# ARMED & CLKRST & !PCS3 & !WR);
```

ABEL(tm) Version 2.02a - Document Generator
PAL16R4

Memory Pal & Power on State Machine Source U17
Fri, Jul 25, 1986, 5:01 PM [mempal7.tak]
-Translated by TOABEL-
Chip diagram for Module _u17

Device P16R4



ABEL(tm) Version 2.02a - Document Generator
 PAL16R4

05-Feb-87 09:29 AM

Memory Pal & Power on State Machine Source U17
 Fri, Jul 25, 1986, 5:01 PM [mempal7.tak]
 -Translated by TOABEL-
 for Module _u17

Device P16R4

Device Type: P16R4

Terms Used: 29 out of 64

| Pin # | Name | Terms Used | Max | Term Type | Pin Type |
|-------|--------|------------|-----|-----------|----------|
| 1 | SCLK | -- | -- | --- | Clock |
| 2 | WR | -- | -- | --- | Input |
| 3 | PCS3 | -- | -- | --- | Input |
| 4 | SA0 | -- | -- | --- | Input |
| 5 | BHE | -- | -- | --- | Input |
| 6 | HLDA | -- | -- | --- | Input |
| 7 | LCS | -- | -- | --- | Input |
| 8 | PON | -- | -- | --- | Input |
| 9 | CLKRST | -- | -- | --- | Input |
| 10 | GND | -- | -- | --- | GND |
| 11 | EN | -- | -- | --- | Enable |
| 12 | LOGRST | 0 | 7 | Normal | I/O |
| 13 | LRS | 3 | 7 | Normal | I/O |
| 14 | DCLINT | 7 | 8 | Normal | Output |
| 15 | WAIT | 5 | 8 | Normal | Output |
| 16 | Q1 | 5 | 8 | Normal | Output |
| 17 | ARMED | 4 | 8 | Normal | Output |
| 18 | URS | 3 | 7 | Normal | I/O |
| 19 | MCS1 | 0 | 7 | Normal | I/O |
| 20 | VCC | -- | -- | --- | VCC |

end of module _u17

Power Fail/Passport Control PAL: U23

Page 1

ABEL(tm) Version 2.02a - Document Generator
PAL16R6

05-Feb-87 09:28 AM

U12 Power fail and hold request & cntl machine. hp part# : 1820-4560
[PFWPAL6.TAK]-Translated by TOABEL-
Equations for Module _u23

Device P16R6

Reduced Equations:

GLOBWR = !(A1 & A2 & A3 & PCS0 & WRITE);

GCLKEN := !(GCLKEN & PCS0 & RST & WRITE);

HOLD = !(MRQ # PFW_INT & SYNC_PFW # RST);

CNTL := !(CNTL & GCLKEN
A2 & CNTL
A3 & CNTL
A1 & A2 & A3 & GCLKEN
RST);INT_ACK := !(GCLKEN & INT_ACK
A1 & A2 & A3 & GCLKEN
A2 & INT_ACK
A3 & INT_ACK
RST);PFW_INT := !(RST
NQ0 & PFW & SYNC_PFW
A1 & A2 & A3 & GCLKEN & PFW_INT
A2 & PFW & SYNC_PFW
A3 & PFW & SYNC_PFW
GCLKEN & PFW & SYNC_PFW
PFW_INT & SYNC_PFW
NQ0 & PFW);SYNC_PFW := !(NQ0 & PFW & SYNC_PFW
A2 & NQ0 & PFW
GCLKEN & NQ0 & PFW
A3 & NQ0 & PFW
NQ0 & PFW & SYNC_PFW

State Machine Source Files

```
# !RST  
# NQO & !PFW & !SYNC_PFW);
```

```
NQO := !(NQO & !PFW & !PFW INT  
# !NQO & !PFW & SYNC_PFW  
# !A2 & !NQO & !PFW  
# GCLKEN & !NQO & !PFW  
# A3 & !NQO & !PFW
```

ABEL(tm) Version 2.02a - Document Generator
PAL16R6
U12 Power fail and hold request & cntl machine
[PFWPAL6.TAK]

-Translated by TOABEL-
Equations for Module _u23

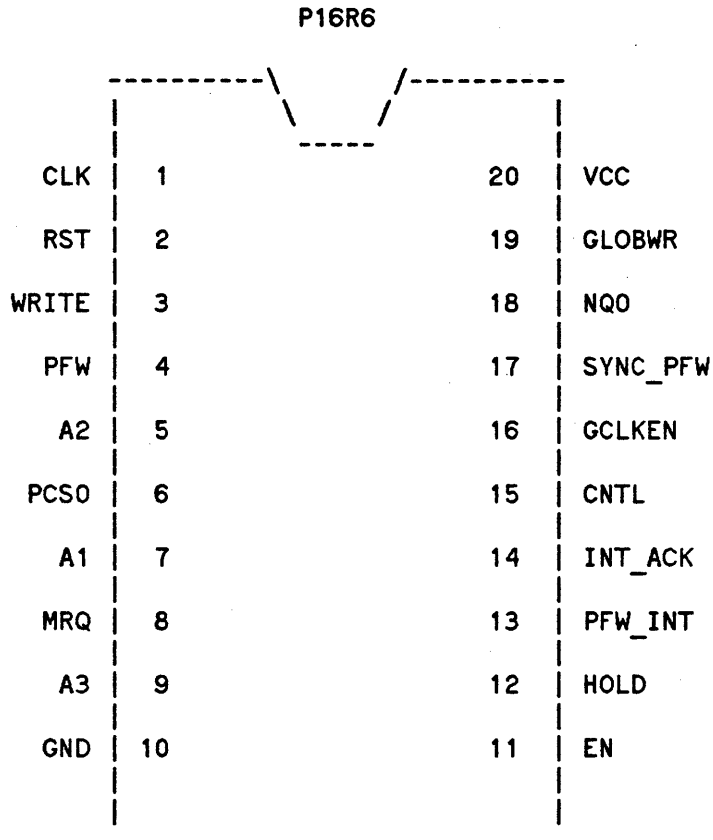
Device P16R6

!RST);

ABEL(tm) Version 2.02a - Document Generator
PAL16R6
U12 Power fail and hold request & cntl machine
[PFWPAL6.TAK]

-Translated by TOABEL-
Chip diagram for Module _u23

Device P16R6



ABEL(tm) Version 2.02a - Document Generator
 PAL16R6
 U12 Power fail and hold request & cntl machine
 [PFWPAL6.TAK]

-Translated by TOABEL-
 for Module _u23

Device P16R6

Device Type: P16R6

Terms Used: 38 out of 64

| Pin # | Name | Terms | | Term Type | Pin Type |
|-------|----------|-------|-----|-----------|----------|
| | | Used | Max | | |
| 1 | CLK | -- | -- | --- | Clock |
| 2 | RST | -- | -- | --- | Input |
| 3 | WRITE | -- | -- | --- | Input |
| 4 | PFW | -- | -- | --- | Input |
| 5 | A2 | -- | -- | --- | Input |
| 6 | PCS0 | -- | -- | --- | Input |
| 7 | A1 | -- | -- | --- | Input |
| 8 | MRQ | -- | -- | --- | Input |
| 9 | A3 | -- | -- | --- | Input |
| 10 | GND | -- | -- | --- | GND |
| 11 | EN | -- | -- | --- | Enable |
| 12 | HOLD | 3 | 7 | Normal | I/O |
| 13 | PFW_INT | 8 | 8 | Normal | Output |
| 14 | INT_ACK | 5 | 8 | Normal | Output |
| 15 | CNTL | 5 | 8 | Normal | Output |
| 16 | GCLKEN | 1 | 8 | Normal | Output |
| 17 | SYNC_PFW | 7 | 8 | Normal | Output |
| 18 | NQO | 6 | 8 | Normal | Output |
| 19 | GLOBWR | 1 | 7 | Normal | I/O |
| 20 | VCC | -- | -- | --- | VCC |

end of module _u23

LED STATUS: CARD OPERATIONAL

APPENDIX

E

The following table indicates the status provided by the front panel LEDs when the HP27111A has successfully passed its internal selftest.

alinksti

| ID | COLOR | DEFINITION | STATE | DESCRIPTION |
|----|-------|--|-------|--|
| F | red | Failed selftest | on | Card failed selftest. The C,S, and R LED's indicate the type of self test failure |
| | | | off | card passed self test |
| C | red | Configuration error | on | More equal jumper is in wrong position (F must be OFF) |
| | | | off | Card is correctly configured (F must be OFF) |
| S | red | Signal not present -OR- unacceptable quality | on | one or more of the following are true: <ul style="list-style-type: none"> ■ No transitions detected on link ■ Phase lock receiver could not acquire signal ■ Link error rate exceeds maximum threshold (F must be OFF) |
| | | | off | signal is of acceptable quality (F must be OFF) |
| R | red | Remote not responding | on | Remote fails to respond to a periodic request for identification (F must be OFF) |
| | | | off | Remote has identified itself (F must be OFF) |

alinksti

| ID | COLOR | DEFINITION | STATE | DESCRIPTION |
|----------|-------|---------------------------------------|--------------|---|
| P | green | Passed self test, card oPerational | on | card has passed self test and is operational (F must be OFF) |
| | | | blink | Firmware awaiting for backplane to issue an initial connect subchannel command (F must be OFF) |
| | | | off | Firmware detected a link Protocol Error or determined that the pca was no longer functional (F must be OFF) |
| A | green | Activity | on | Indicates that a header has just been transmitted (F must be OFF) |
| | | | off | no headers are currently begin transmitted (F must be OFF) |

LED STATUS: SELF TEST FAILURE COD

The minor error codes are reported when the card fails one of its selftests and the card is capable of detecting the assertion of MINOR-. The 3 or 4 octal digit minor code is constantly repeated on a 3 or 4 digit cycle.

Error Codes for HP27111A

LED Format::

| F | CSR | PA | |
|---|-----|----|--|
| * | MMM | .. | :: MMM represents the Major Failure Code [0..7] |
| * | mmm | .* | :: mmm represents the First Minor Code Digit [0..7] |
| * | mmm | *. | :: mmm represents the Second Minor Code Digit [0..7] |

Where * :: LED is ON!!!!
. :: LED is off

=====
Minor Code Format::

| Second Minor Code | Error Description |
|------------------------|----------------------------------|
| | |
| v | v |
| <45> | Fails something or Another... |
| | |
| First Minor Code | |
| ***** | |
| Major Failure Code = 7 | Boot Rom |
| ***** | |
| ***** | |
| Major Failure Code = 6 | Processor |
| ***** | |
| ***** | |
| Major Failure Code = 5 | RAM Test |
| ***** | |
| ***** | |
| Major Failure Code = 4 | Basic Pronto Test |
| ***** | |
| Minor Code | Description |
| ----- | |
| <01> | "Pronto I/O Fault: Fails LD[15]" |
| <02> | "Pronto I/O Fault: Fails LD[14]" |
| <03> | "Pronto I/O Fault: Fails LD[13]" |
| <04> | "Pronto I/O Fault: Fails LD[12]" |

LED Status: Self Test Failure Codes

```

<05> "Pronto I/O Fault: Fails LD[11]"
<06> "Pronto I/O Fault: Fails LD[10]"
<07> "Pronto I/O Fault: Fails LD[9]"
<10> "Pronto I/O Fault: Fails LD[8]"
<11> "Pronto I/O Fault: Fails LD[7]"
<12> "Pronto I/O Fault: Fails LD[6]"
<13> "Pronto I/O Fault: Fails LD[5]"
<14> "Pronto I/O Fault: Fails LD[4]"
<15> "Pronto I/O Fault: Fails LD[3]"
<16> "Pronto I/O Fault: Fails LD[2]"
<17> "Pronto I/O Fault: Fails LD[1]"
<20> "Pronto I/O Fault: Fails LD[0]"
<21> "Soft Reset Failure : Mask <> 0"
<22> "Soft Reset Failure: Pronto Status Bad"

```

Major Failure Code = 3 Jupiter Pronto Link Test

| Minor Code | Description |
|------------|---|
| ----- | ----- |
| <01> | "Soft Reset Failure: Extraneous Interrupts" |
| <02> | "Expected Link Active" |
| <03> | "Expected Frequency Lock" |
| <04> | "RLR timeout" |
| <05> | "RRA timeout" |
| <06> | "Pronto Interrupt Pin" |
| <07> | "Pronto ISR Timeout" |
| <10> | "Failed to Clear 186 Internal Int for Pronto" |
| <11> | "Missing RLR Interrupt Status" |
| <12> | "Unmasked RLR fails to gen Interrupt" |
| <13> | "RLR Mask Failure" |
| <14> | "Fail to Clear RLR" |
| <15> | "Missing RRA Interrupt Status" |
| <16> | "Unmasked RRA fails to gen Interrupt" |
| <17> | "RRA Mask Failure" |
| <20> | "Fail to Clear RRA" |
| <21> | "Missing External RLR int" |
| <22> | "Fail RLR Isr" |
| <23> | "Internal RLR Clear" |
| <24> | "External RLR Clear" |
| <25> | "Wait RSN" |
| <26> | "Missing RSN Interrupt Status" |
| <27> | "Unmasked RSN fails to gen Interrupt" |
| <30> | "RSN Mask Failure" |
| <31> | "Fail to Clear RSN" |
| <32> | "Wait RSA" |
| <33> | "Missing RSA Interrupt Status" |
| <34> | "Unmasked RSA fails to gen Interrupt" |
| <35> | "RSA Mask Failure" |
| <36> | "Fail to Clear RSA" |
| <37> | "No RFD" |
| <40> | "Unexpected DAV" |
| <41> | "Missing DAV" |
| <42> | "Missing RXE Interrupt Status" |
| <43> | "Unmasked RXE fails to gen Interrupt" |

- <44> "RXE Mask Failure"
- <45> "Fail to Clear RXE"
- <46> "Timeout on Send Buffer:Halt on RXE Test"
- <47> "Timeout on Receiving First Buffer:Halt on RXE Test"
- <50> "No Halt on RXE"
- <51> "Fail to Recover DAV after Halt on RXE"
- <52> "Unexpected MMV"
- <53> "Timeout on Receiving Buffer:MMV due to VC Test"
- <54> "Missing MMV Interrupt Status"
- <55> "Unmasked MMV fails to gen Interrupt"
- <56> "MMV Mask Failure"
- <57> "Fail to Clear MMV"
- <60> "Timeout on Receiving Buffer:MMV due to TYP Test"
- <61> "Missing MMV Interrupt Status"
- <62> "Unmasked MMV fails to gen Interrupt"
- <63> "MMV Mask Failure"
- <64> "Fail to Clear MMV"
- <65> "Underflow on RFD"
- <66> "Overflow on RFD"
- <67> "Resync failed to clear DAV"
- <70> "Resync Failed to assert RFD"
- <71> "Timeout on Sendbuf"
- <72> "Timeout on Getbuf"
- <73> "Data Compare"

 Major Failure Code = 2 Passport Test

| Minor Code | Description |
|------------|--|
| ----- | ----- |
| <01> | "No Interrupt on Passport Init" |
| <02> | "Passport ISR failed" |
| <03> | "Passport Init Interrupt Ack fail" |
| <04> | "Passport Init Control Block Error" |
| <05> | "Pause Flag Set Fail" |
| <06> | "Pause Flag Clear Fail" |
| <07> | "Timeout on Autoitem 1 (wait on DRFD)" |
| <10> | "Next Autoitem <> Autoitem2" |
| <11> | "Missing Pause after Autoitem1" |
| <12> | "Stop Autoitem2 Failure" |
| <13> | "Force Unpause failed" |
| <14> | "Timeout on Autoitem2 (wait on DDAV)" |
| <15> | "Next Autoitem <> Autoitem3" |
| <16> | "Missing Pause after Autoitem2" |
| <17> | "Stop Autoitem 3 Failure" |
| <20> | "Next Autoitem <> Autoitem4" |
| <21> | "Missing Pause after Autoitem3" |
| <22> | "Timeout on Autoitem4 (wait on DRFD)" |
| <23> | "Next Autoitem <> NIL" |
| <24> | "Missing UnPause Status after Autoitem4" |
| <25> | "Wait RSN" |
| <26> | "Missing RSN Interrupt Status" |
| <27> | "Unmasked RSN fails to gen Interrupt" |
| <30> | "RSN Mask Failure" |
| <31> | "Fail to Clear RSN" |

LED Status: Self Test Failure Codes

- <32> "Wait RSA"
- <33> "Missing RSA Interrupt Status"
- <34> "Unmasked RSA fails to gen Interrupt"
- <35> "RSA Mask Failure"
- <36> "Fail to Clear RSA"
- <37> "Timeout on Memory to Pronto"
- <40> "Auto List Ptr #1 <> 0"
- <41> "Buffer Header Ptr #1 <> 0"
- <42> "Memory to Device: Bad TLOG"
- <43> "Missing DAV:Pronto to Memory"
- <44> "Timeout: Pronto to Memory"
- <45> "Auto List Ptr #2 <> 0"
- <46> "Buffer Head Ptr #2 <> 0"
- <47> "Pronto to Memory: Bad TLOG"
- <50> "Missing Pronto RXE"
- <51> "Pronto to Memory: Missing EOS flag"
- <52> "Bad Data"

Major Failure Code = 1 DMA Tx Test

| Minor Code | Description |
|------------|---|
| ----- | ----- |
| <01> | "Pronto Init: Wait on DAV" |
| <02> | "Dma Tx Timeout" |
| <03> | "Missing Dma Tx Internal Interrupt" |
| <04> | "Dma Tx ISR Timeout" |
| <05> | "Dma Tx Intack Failed" |
| <06> | "Tx Switch Failed: Expected TxD" |
| <07> | "Memory To Pronto Phase" |
| <10> | "Auto List Ptr #3 <> 0" |
| <11> | "Buffer Head Ptr #3 <> 0" |
| <12> | "Bad TLOG" |
| <13> | "Tx Switch Failure: Expected ~TxD" |
| <14> | "Missing TXE (TAK test)" |
| <15> | "Missing RFD (TAK test)" |
| <16> | "Fail to auto ack TXE" |
| <17> | "Timeout on Transmit Header w/wrong VC" |
| <20> | "Tx Switched on Mismatched VC" |
| <21> | "Timeout on Transmit Header w/Wrong Type" |
| <22> | "Tx Switched on Non Data Type Header" |

Major Failure Code = 0 DMA Rx Test

| Minor Code | Description |
|------------|-------------------------------------|
| ----- | ----- |
| <01> | "Dma Rx Timeout" |
| <02> | "Invalid Header Received" |
| <03> | "Missing Internal Dma RX Interrupt" |
| <04> | "Dma Rx Timeout on ISR" |
| <05> | "Failed to Clear Dma Rx interrupt" |
| <06> | "Rx Switch Failure: Expected RXD" |
| <07> | "DMA Rx: Passport Phase" |
| <10> | "Auto List Ptr #4 <> 0" |
| <11> | "Buffer Head Ptr #4 <> 0" |

<12> "Pronto to Memory: Bad TLOG"
 <13> "Missing Pronto RXE"
 <14> "Missing EOS flag"
 <15> "Bad Data"
 <16> "Rx Switch Failure: Expected ~RxD"
 <17> "Missing DAV (RAK Test)"
 <20> "Missing RXE"
 <21> "Fail to autoack RXE"
 <22> "Timeout on Rx Header w/mismatched VC"
 <23> "Rx Switched on mismatched VC"
 <24> "Timeout on Rx Header w/non data type"
 <25> "Rx Switched on non data type"

 Auxiliary Codes: Passport or Pronto Unexepected Status during <nnA>

 <nn0> "Status:Pronto Internal Error Detected"
 <nn1> "Status:Passport Command Failed during <nn>"
 <nn2> "Status:Passport Parity Error with Memory I/O "
 <nn3> "Status:Passport Parity Error on CIO Control bus"
 <nn4> "Status:Passport Parity Error on CIO Data bus"
 <nn5> "Status:Passport Parity Error on Pronto Data bus"
 <nn6> "Status:Passport Buffer Overflow in Autolist"
 <nn7> "Status:Pronto Protocol Error Detected"

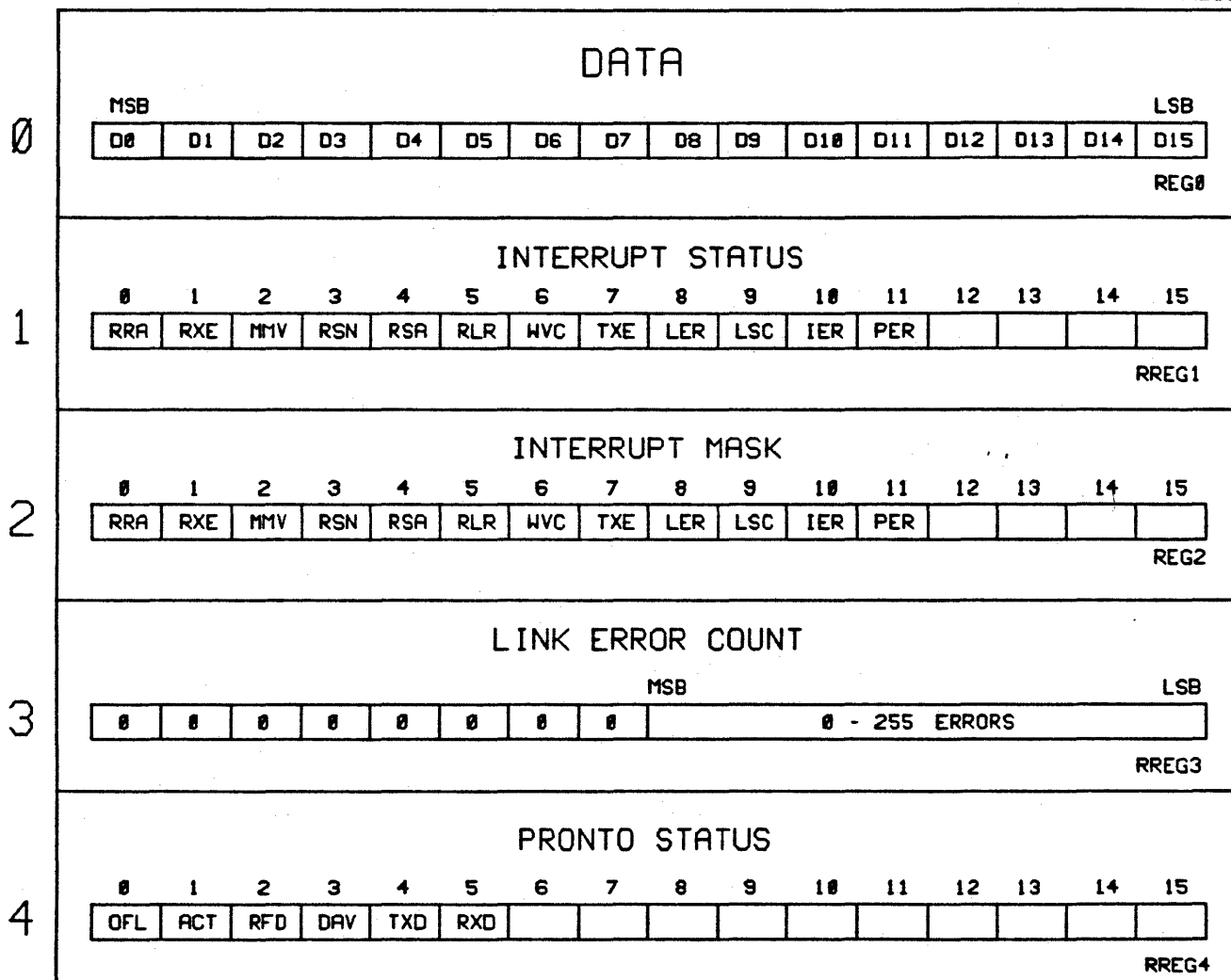
PRONTO REGISTERS

APPENDIX

G

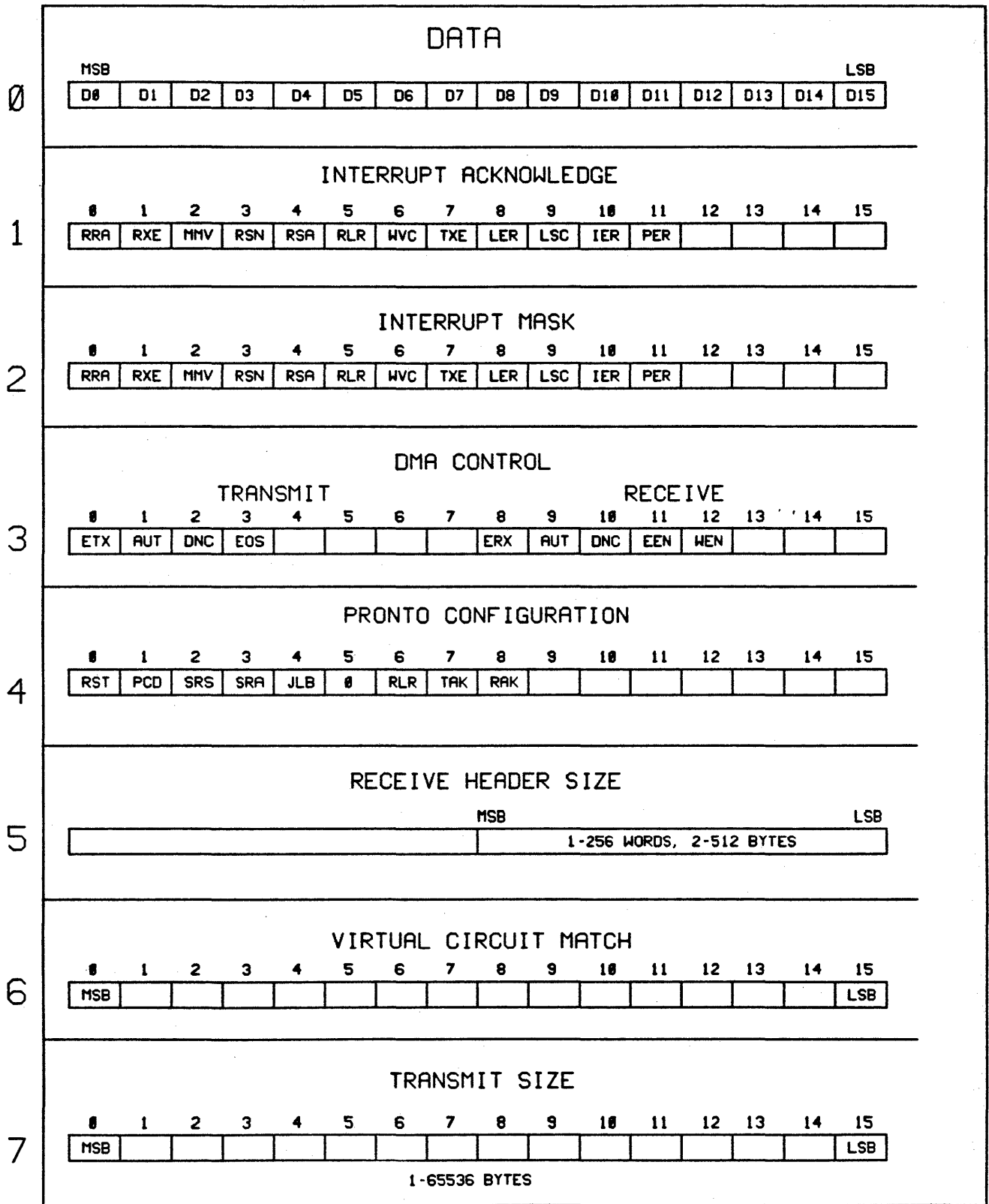
READ REGISTERS

RREGS



WRITE REGISTERS

WREGS



CHANNEL REGISTERS

APPENDIX

H

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---------------|------------------------|---|---|---|---|---|---|---|------------|-----|-----|-----|-----------|-----|-----|-----|
| CIO Register | | | | | | | | | | | | | | | | |
| Read Data | Upper Byte (word mode) | | | | | | | | Lower Byte | | | | | | | |
| Read Sense | | | | | | | | | RFC | PST | PRE | NMI | | ARE | | ARQ |
| Read Status | | | | | | | | | Opcode | | | | Extension | | | |
| Write Data | Upper Byte (word mode) | | | | | | | | Lower Byte | | | | | | | |
| Write Control | | | | | | | | | | | DCL | DEN | NMK | RQA | ARE | ARD |
| Write Order | | | | | | | | | Opcode | | | | Extension | | | |
| Write Command | | | | | | | | | Opcode | | | | Extension | | | |

[alciobit]

RFC : Ready For Command
PST : Passed SelfTest
PRE : PREsent
NMI : Non Maskable Interrupt
ARE : Attention Request Enabled
ARQ : Attention ReQuest
DCL : addressed Device CLear
DEN : addressed Device ENable
NMK : Non Maskable interrupt acKnowledge
RQA : ReQuest Attention
ARE : Attention Request Enable
ARD : Attention Request Disable

For details on the Opcode and Extension fields, refer to the Firmware Description section or the CIO Specification document.

Table of Contents

| | |
|---------------------------------------|---|
| Current Distribution | 3 |
| Future Release Distribution | 5 |

Section 1 INTRODUCTION

| | |
|-------------------------------|-----|
| Sectional Overview | 1-1 |
| Changes and Updates | 1-2 |
| Required Documents | 1-3 |

Section 2 REFERENCE

| | |
|-------------------------------------|-----|
| Related Documents | 2-1 |
| Conventions | 2-2 |
| Bit and Byte Notations | 2-2 |
| Bit Numbering | 2-2 |
| Byte vs. Word | 2-2 |
| Byte Numbering | 2-3 |
| Logic Notation | 2-3 |
| Assertion vs. Deassertion | 2-3 |
| Logical L vs. Logical H | 2-3 |
| Logical 0 vs. Logical 1 | 2-3 |
| Active vs. Passive | 2-4 |
| Schematic Notation | 2-4 |
| Signal Naming | 2-4 |
| Boolean Operators | 2-4 |
| Reference Designators | 2-5 |
| Glossary | 2-6 |

Section 3 FUNCTIONAL OVERVIEW

| | |
|--|-----|
| High Level Block Diagram Description | 3-1 |
| Backplane Adapter | 3-2 |
| Channel operations | 3-2 |
| Subchannel interactions | 3-3 |
| Logchannel transactions | 3-4 |
| Processor communications | 3-5 |
| DMA operations | 3-5 |
| Protocol Controller | 3-6 |
| Control Port | 3-6 |
| Device port | 3-7 |
| Switch | 3-7 |
| Transmit port | 3-7 |
| Receive port | 3-8 |
| Fiber Optic Conversion | 3-8 |
| Optical receiver | 3-8 |
| Post Amplifier and Quantizer | 3-9 |
| Serial to parallel conversion | 3-9 |
| Parallel to serial conversion | 3-9 |

Table of Contents

| | |
|---|------|
| Driver | 3-9 |
| Optical Transmitter | 3-9 |
| Processor | 3-9 |
| Link Control/Layer 3 engine | 3-10 |
| Microprocessor/80186 | 3-10 |
| Memory | 3-11 |
| Global Status/Control LED's | 3-11 |
| Arbiter | 3-11 |
| Processor to Protocol Controller Control Port | 3-12 |
| Backplane Adapter to Processor Memory | 3-12 |
| Sample Read Transaction | 3-12 |
| Backplane Program Sequence | 3-13 |
| Resource Allocation | 3-14 |
| Command Phase | 3-14 |
| Execution Phase. | 3-14 |
| Report Phase. | 3-14 |
| Frontplane Program Sequence. | 3-14 |
| Resource Allocation | 3-15 |
| Command Phase | 3-16 |
| Execution Phase. | 3-16 |
| Report Phase. | 3-16 |
| Functional Block Interaction | 3-16 |
| Resource Allocation | 3-16 |
| Command Phase | 3-17 |
| Execution Phase. | 3-19 |
| Report Phase. | 3-21 |

Section 4 SPECIFICATIONS

| | |
|---|------|
| Functional. | 4-1 |
| Communication Path Configurations | 4-2 |
| Host to Disc | 4-2 |
| Host to Host | 4-2 |
| Frontplane (A-Link) | 4-2 |
| A-Link Protocol | 4-2 |
| Virtual Circuit Capabilities | 4-3 |
| Transfer Rates. | 4-3 |
| Link ID. | 4-3 |
| Backplane | 4-3 |
| Channel I/O compatibility. | 4-3 |
| Supported Orders and Commands. | 4-4 |
| Transfer Rates. | 4-4 |
| Logchannel Capabilities. | 4-4 |
| Configurable Options | 4-4 |
| Equal mode | 4-4 |
| Channel NMI Enable | 4-5 |
| Power Fail Support. | 4-5 |
| Status Indicators | 4-5 |
| LED indicators | 4-5 |
| Backplane Status Report | 4-8 |
| Data Integrity. | 4-9 |
| Channel to Backplane Adapter | 4-10 |

Table of Contents

| | |
|--|------|
| Backplane Adapter to Protocol Controller | 4-10 |
| Protocol Controller to Fiber Optic Conversion. | 4-10 |
| Fiber Optic Conversion to Fiber. | 4-10 |
| Processor | 4-10 |
| Self Test | 4-10 |
| Power On. | 4-11 |
| Reset. | 4-11 |
| Addressed. | 4-11 |
| External Fiber. | 4-11 |
| Electrical | 4-11 |
| Backplane | 4-12 |
| Power | 4-12 |
| Loading. | 4-12 |
| Timing | 4-12 |
| Frontplane. | 4-12 |
| Optical Receiver | 4-12 |
| Optical Transmitter | 4-13 |
| Fiber. | 4-13 |
| Received Signal Probe Tap | 4-13 |
| Physical | 4-14 |
| Dimensions | 4-14 |
| Connections | 4-15 |
| Frontplane | 4-15 |
| Backplane | 4-15 |
| Jumperable Options | 4-16 |
| Status Indicators | 4-18 |
| Operating Environment | 4-18 |

Section 5

FIRMWARE DESCRIPTION

| | |
|--|------|
| Changes from last revision. | 5-1 |
| ERT. | 5-1 |
| Loopback. | 5-2 |
| Subfunction 254 and 255 | 5-2 |
| Status bytes | 5-2 |
| Channel Programming | 5-2 |
| Supported Commands and Orders | 5-2 |
| CSC: Connect_subchannel Command. | 5-3 |
| DSC: Destroy_subchannel Command | 5-3 |
| RSC: Resume_subchannel Command | 5-3 |
| CLC: Connect_logical_channel Order | 5-4 |
| WC: Write_control Order | 5-5 |
| RD: Read_data Order | 5-5 |
| WD: Write_data Order. | 5-6 |
| RS: Read_status Order | 5-6 |
| IDY: Identify Order | 5-9 |
| RTS: Read_transparent_status Order | 5-10 |
| WTC: Write_transparent_control Order | 5-10 |
| DIS: Disconnect Order. | 5-10 |
| Channel Program Structure | 5-10 |
| Chaining | 5-10 |
| Performance Est. | 5-11 |

Table of Contents

| | |
|--|------|
| Card Requests | 5-11 |
| Read Device Data (Rq = 1). | 5-11 |
| Rq = 1, SF = 0 | 5-12 |
| Rq = 1, SF = 1 | 5-13 |
| Rq = 1, SF = 253 | 5-14 |
| Rq = 1, SF = 254 | 5-17 |
| Rq = 1, SF = 255 | 5-18 |
| Write Device Data (Rq = 2) | 5-18 |
| Rq = 2, SF = 0 | 5-19 |
| Rq = 2, SF = 1 | 5-20 |
| Rq = 2, SF = 2 | 5-21 |
| Rq = 2, SF = 255 | 5-21 |
| Read Card Information (Rq = 4) | 5-22 |
| Rq = 4, SF = 22 | 5-22 |
| Rq = 4, SF = 33 | 5-23 |
| Rq = 4, SF = 250 | 5-24 |
| Rq = 4, SF = 253 | 5-25 |
| Write Card Configuration (Rq = 5). | 5-26 |
| Rq = 5, SF = 22 | 5-26 |
| Rq = 5, SF = 250 | 5-27 |
| Control Card (Rq = 6). | 5-27 |
| Rq = 6, SF = 2 | 5-28 |
| Rq = 6, SF = 3 | 5-28 |
| Rq = 6, SF = 4 | 5-29 |
| Rq = 6, SF = 250 | 5-29 |
| Rq = 6, SF = 252 | 5-29 |
| Rq = 6, SF = 253 | 5-30 |
| Rq = 6, SF = 254 | 5-30 |
| Rq = 6, SF = 255 | 5-30 |
| Request Abort | 5-31 |
| Event Processing | 5-31 |
| Interrupt Operation | 5-32 |

Section 6

USER INTEGRATION

| | |
|---|-----|
| Installation | 6-1 |
| Software installation | 6-1 |
| Selecting Card Configuration | 6-1 |
| Backplane Installation | 6-2 |
| Optical Connection | 6-3 |
| Performance Verification | 6-5 |
| Hardware Verification Selftest | 6-5 |
| Basic | 6-5 |
| Extended | 6-5 |
| Failure Codes | 6-5 |
| External Loopback | 6-6 |
| Remote Device Loopback | 6-7 |
| Functional Verification | 6-7 |
| Well Known Virtual Circuit Operations | 6-7 |
| Remote Device Verification | 6-8 |
| Serviceability | 6-8 |
| Diagnostic Tools | 6-8 |

Table of Contents

| | |
|---|------|
| Firmware Interface | 6-8 |
| Event Detection | 6-10 |
| Status Indicators | 6-11 |
| Optical Tool kit | 6-12 |
| Remote Device Indications | 6-13 |
| Field Troubleshooting | 6-13 |
| Console - <i>Non-Disruptive</i> | 6-14 |
| Console - <i>Disruptive</i> | 6-18 |
| Console - <i>DESTRUCTIVE</i> | 6-21 |
| ON-Site - <i>Non-Disruptive</i> | 6-21 |
| ON-Site - <i>Disruptive</i> | 6-23 |
| ON-Site - <i>DESTRUCTIVE</i> | 6-28 |
| Upgrade | 6-30 |
| Firmware Revisions | 6-30 |
| Hardware Revisions | 6-30 |
| Link type | 6-30 |

Section 7

THEORY OF OPERATION

| | |
|--------------------------------------|------|
| Backplane Adapter/Passport | 7-2 |
| Channel Interface | 7-4 |
| Control and Address | 7-6 |
| Control and Address Parity | 7-8 |
| Data Bus | 7-9 |
| Data Parity | 7-9 |
| FIFO | 7-9 |
| Device Port | 7-10 |
| Data Bus | 7-10 |
| Control Bus | 7-10 |
| Clocking | 7-10 |
| Processor Interface | 7-11 |
| PASSPORT Control Signals | 7-12 |
| Primary Power Status | 7-12 |
| Secondary Power Status | 7-12 |
| Clocking | 7-13 |
| Memory Interface | 7-13 |
| Address Bus | 7-13 |
| Data Bus | 7-13 |
| Memory Control | 7-14 |
| Clocking | 7-15 |
| Initialization Block | 7-15 |
| Control Block | 7-16 |
| Auto List Items | 7-18 |
| Buffers | 7-18 |
| Channel NMI Enable | 7-19 |
| Reset Control | 7-20 |
| Power On Reset | 7-20 |
| Channel Reset | 7-21 |
| Device Clear | 7-22 |
| Device Clear Machine | 7-22 |
| Mode Initialization | 7-26 |
| Diagnostic Interface Port | 7-27 |

Table of Contents

| | |
|--|------|
| Common Inputs | 7-27 |
| Diagnostic Port Select | 7-27 |
| Diagnostic Output | 7-28 |
| Power References | 7-28 |
| VDL | 7-28 |
| VBG | 7-28 |
| Protocol Controller | 7-29 |
| Control Port | 7-31 |
| Control and Address | 7-31 |
| Timing | 7-31 |
| Data | 7-33 |
| Hardware Status Lines | 7-33 |
| Register Set | 7-33 |
| Device Port | 7-34 |
| Data Bus | 7-34 |
| Control Bus | 7-34 |
| Switch | 7-35 |
| Manual Mode | 7-35 |
| Automatic Mode | 7-35 |
| Rx FIFO | 7-35 |
| Tx FIFO | 7-36 |
| Tx Port | 7-36 |
| Transmit Clock | 7-36 |
| Control | 7-36 |
| Data | 7-37 |
| Loopback Mode | 7-37 |
| Layer 2 Operation | 7-37 |
| Rx Port | 7-37 |
| Receive Clock | 7-38 |
| Control | 7-38 |
| Data | 7-38 |
| Control Port Monitoring of Link Status | 7-39 |
| Reset Behaviour | 7-39 |
| Power On | 7-39 |
| Channel Reset | 7-40 |
| Device Clear | 7-40 |
| Diagnostic Interface Port | 7-41 |
| Common Inputs | 7-41 |
| Diagnostic Port Select | 7-41 |
| Diagnostic Output | 7-41 |
| Fiber Optic Conversion | 7-42 |
| Parallel to Serial (Jupiter Tx) | 7-45 |
| Transmit Clock Generation | 7-45 |
| Parallel Transfer | 7-45 |
| Data Encoding | 7-45 |
| Serializing | 7-46 |
| Optical Transmitter Driver | 7-46 |
| Transmit LED | 7-46 |
| Current Switch Driver | 7-46 |
| LED Current Switch | 7-47 |
| LED Peaking Circuit | 7-47 |
| Current Source Generation | 7-48 |
| Inhibit Mode | 7-50 |
| Optical Receiver and Amplifier | 7-50 |

Table of Contents

| | |
|---|------|
| Optical Receiver | 7-51 |
| Isolation Transformer | 7-51 |
| Post Amplification | 7-51 |
| Quantization | 7-52 |
| Receive Signal Tap | 7-53 |
| Serial to Parallel (Jupiter Rx) | 7-53 |
| Serial Data Extraction | 7-53 |
| Receive Phase Lock | 7-53 |
| Byte Synchronization | 7-54 |
| Character Decoding | 7-54 |
| JRX Parallel Port | 7-54 |
| Reset Behaviour | 7-54 |
| Processor | 7-56 |
| iapx80186 | 7-58 |
| Address and Data | 7-58 |
| Control Interface | 7-58 |
| System Clock Generation | 7-59 |
| Integrated Chip Select | 7-60 |
| Programming Notes | 7-61 |
| RAM Control Logic | 7-61 |
| RAM Selects | 7-62 |
| Primary Power Monitoring | 7-62 |
| Byte vs. Word | 7-63 |
| Address Latches | 7-63 |
| Timing | 7-63 |
| Program Memory | 7-64 |
| Expansion ROM | 7-64 |
| Timing | 7-64 |
| RAM | 7-66 |
| Expansion RAM | 7-66 |
| Timing - Memory Read | 7-67 |
| Timing - Memory Write | 7-68 |
| Global Status Register | 7-69 |
| Format | 7-69 |
| Timing | 7-71 |
| Global Control Register | 7-72 |
| Format | 7-72 |
| Timing | 7-74 |
| Integrated DMA | 7-76 |
| Programming Notes | 7-76 |
| DMA Link Write | 7-76 |
| DMA Link Read | 7-77 |
| Integrated Interrupt Controller | 7-78 |
| Integrated Register Set | 7-79 |
| General Interrupt Processing | 7-79 |
| Programming Notes | 7-80 |
| Passport Interrupt | 7-80 |
| Pronto Interrupt | 7-80 |
| RLR Interrupt | 7-80 |
| PFW Interrupt | 7-80 |
| DCL Interrupt | 7-81 |
| Integrated Timer | 7-81 |
| Real-Time Clock | 7-81 |
| Selftest Timer | 7-81 |

Table of Contents

| | |
|---|-------|
| Device Clear Wait Timer | 7-81 |
| Programming Notes | 7-82 |
| Passport Control | 7-82 |
| Clock Enable | 7-82 |
| INTA+ | 7-84 |
| CNTL+ | 7-85 |
| Timing | 7-86 |
| RAM Data Structure Format | 7-87 |
| Secondary Power Support | 7-88 |
| Secondary Power Configuration | 7-89 |
| Power Fail Warning Detection | 7-89 |
| Memory Protection | 7-91 |
| Power Fail Recovery | 7-91 |
| RLR Detection | 7-92 |
| LED's | 7-93 |
| Hardware Revision Code | 7-93 |
| Equal Mode Jumper | 7-93 |
| Minor Code Report | 7-94 |
| Reset Behaviour | 7-94 |
| Power On and Channel Reset | 7-94 |
| Device Clear | 7-95 |
| Arbiter | 7-97 |
| Arbiter PALS | 7-99 |
| Control Signals | 7-99 |
| Main Arbiter:Flow Chart | 7-101 |
| Read Write Generation:FLOW CHART | 7-103 |
| Bus Transceivers | 7-103 |
| Direction Machine | 7-104 |
| Pulse Stretcher | 7-104 |
| Arbiter State Machine Processor Request | 7-105 |
| Processor Read Pronto | 7-105 |
| Processor Write Pronto | 7-109 |
| Consecutive Requests | 7-111 |
| Arbiter State Machine:Backplane Adapter Request | 7-111 |
| 80186 AD[] Bus Request Mechanism | 7-111 |
| RAM Control | 7-111 |
| Passport Read Memory | 7-112 |
| Passport Write Memory | 7-115 |
| Consecutive Requests | 7-117 |
| Arbitration of multiple requests | 7-118 |
| Simultaneous Requests | 7-118 |
| Passport Requests while Arbiter under Processor Control | 7-118 |
| Reset Behaviour | 7-118 |
| Power Fail Warning Behaviour | 7-119 |
| Miscellaneous Circuitry | 7-120 |
| Channel Supplied Power | 7-120 |
| Primary Power | 7-120 |
| Secondary Power | 7-120 |
| 12 Volt Reference | 7-120 |
| -12 Volt Reference | 7-120 |
| Ground Return | 7-121 |
| Frame Ground | 7-121 |
| NMOS Voltage Supplies | 7-121 |
| VDL Reference | 7-121 |

Table of Contents

VBG Reference 7-121
DC-DC Converter Basic Operation 7-122
 Architecture of the UA78S40. 7-123
 Operation of the UA78S40. 7-124
 Modes of Operation 7-125
 Layout Considerations. 7-126
 Specifications 7-127

Appendix A
SIGNAL NAME LIST

Appendix B
SCHEMATICS

Appendix C
PARTS LIST

Appendix D
STATE MACHINE SOURCE FILES

CALS Source Files. D-2
 Arbiter PAL: U15 D-2
 Device Clear/Memory Control PAL: U17 D-7
 Power Fail/Passport Control PAL: U23. D-10
PALASM Source Equations D-14
 Arbiter PAL: U15 D-14
 Device Clear/Memory Control PAL: U17 D-16
 Power Fail/Passport Control PAL: U23. D-18
ABEL Formatted Equations. D-20
 Arbiter PAL: U15 D-20
 Device Clear/Memory Control PAL: U17 D-25
 Power Fail/Passport Control PAL: U23. D-29

Appendix E
LED STATUS: CARD OPERATIONAL

Appendix F
LED STATUS: SELF TEST FAILURE CODES

Appendix G
PRONTO REGISTERS

Table of Contents

**Appendix H
CHANNEL REGISTERS**