$2.95

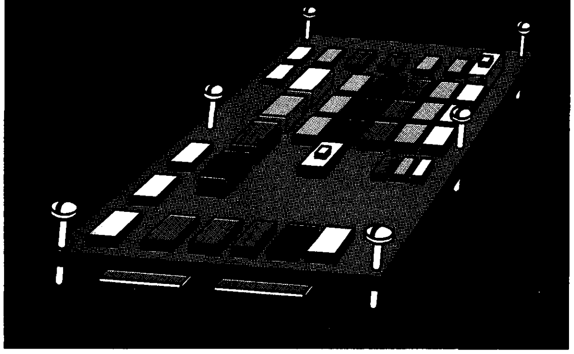
TECHNOLOGY LEADERSHIP

●

BIPOLAR MICROPROCESSOR

●

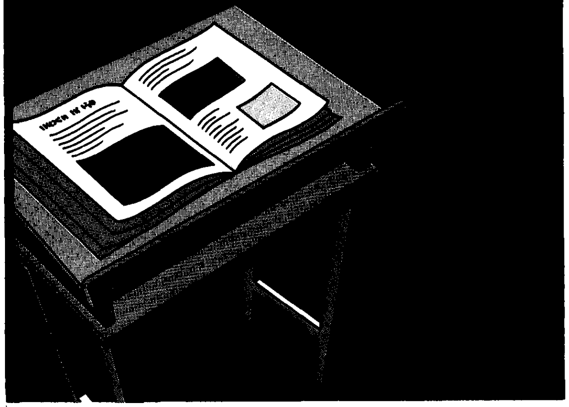
INCREASE YOUR PERFORMANCE WITH OUR 8080 EMULATOR.



Expand your present 8080A based system, yet use all of the software you've developed, with the Signetics 8080A Emulator. Built with the 3001/3002 bipolar bit slice microprocessor, the emulator executes all 8080 instructions at speeds from 2 to 9 times faster than the 8080, uses a single 5 volt supply, a single phase clock, and is microprogram expandable.

The emulator kit comes complete with all parts needed to construct this bipolar replacement for the 8080A, 8228, 8224 and 8212, including preprogrammed PROMS and P.C. board. The accompanying manual tells how to build it, how to use it, and gives a thorough tutorial description of the design and theory of operation. With this $299 kit you can have 8080A emulation in 6 hours. **Order 3000KT8080SK.**

SEND FOR THE BOOK ON MICROPROCESSING.



Reach for this book anytime you need information on bipolar microprocessors. The following pages have everything here for quick easy reference to data sheets on:

- Bit Slice Microprocessor
- Sequencers
- Microcontroller
- Selected Interface Products

with selection guides and summary data sheets for:

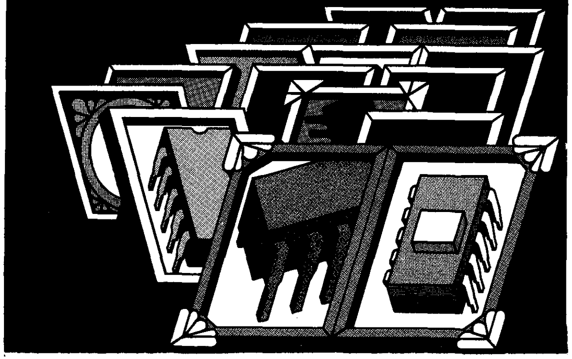
- Memory Products (including FPLA)
- Analog Interface Products
- Interface Circuits
- System Logic

and references and data sheets on:

- Development Systems
- Development Software

and data sheets on kits and application notes.
Yes, it's all here.

THE INDUSTRY'S LEADING FAMILY OF BIPOLAR MEMORIES TO PICK FROM.



Signetics is the total memory supplier. To meet the needs of microprocessor users and system designers, Signetics offers a complete line of memory products. Here are a few examples:

- Bipolar and Static MOS RAMS
  Bipolar 8 to 1024 bits, access times to 35ns
  MOS Static 256x4 to 4096x1,* access time to 85ns
- Dynamic MOS RAMS
  To 16Kx1
- Bipolar and MOS ROMS and PROMS
  Bipolar and MOS Static ROMS to 16K bits
  Bipolar PROMS to 16K bits*
  MOS EROMS* to 8 K bits
  Character Generators
- Bipolar FPLA/PLA
- Bipolar CAMS and Register Files

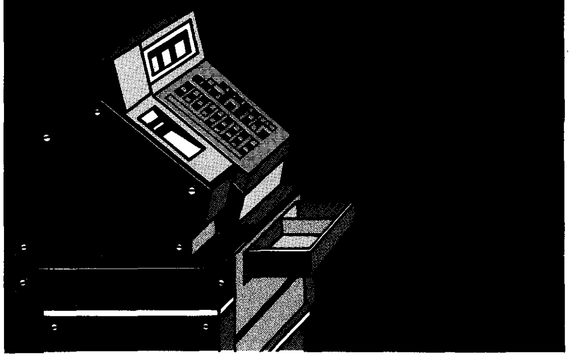
Selection guides are provided in the Bipolar Microprocessor book. For more information send for the full list of total MOS and bipolar memory line.

*Available 1st quarter 1977

DREAM UP YOUR OWN CONTROLLERS WITH THE 8X300 DESIGNER'S KIT.

Complete with the 8X300, clock crystal, 4 I/O ports, 256 bytes of working storage, 512 words of program storage, and P.C. board with wire wrap area, the kit is ready to be used as a basic starter system for your own controller design. There are 450 empty PROM locations that can be used to hold the program for your controller using this, the industry's first high speed (Schottky) complete 8-bit microprocessor. The remaining 62 locations contain diagnostics which can be used to check out the board or monitor the 8X300 through Single Step and Instruction Jam features of this kit. This $299 kit is a fast, inexpensive way to get started with the 8X300. **Order 8X300KT100SK.**

ONE-STOP SHOPPING FOR BIPOLAR AND ANALOG MICROPROCESSING INTERFACE PRODUCTS.



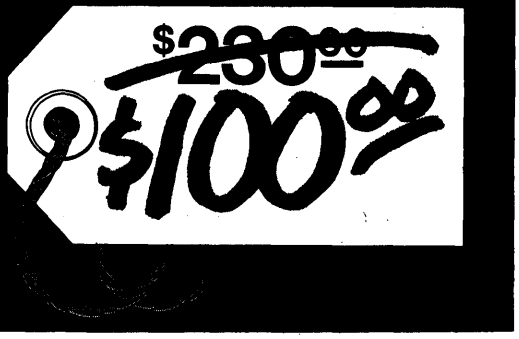
Making your microprocessor system do its job requires interface to the outside world. Signetics Analog Interface products link your microprocessor to displays and sensors. Here are a few examples:

- Peripheral Interface
  Drivers to 80 volts (DS3611 series—UDN5711 series)
  Line Receivers and Drivers
- Display Interface
  Display Decoder/Drivers to 100 volts—DM8880/-1
  NE584, NE585, NE582
- D/A Converters to 8 bits
  MC1408-8, NE5008/9
- Comparators
  NE521/522, LM111/211/311, LM119/219/319, LM139/239/339, LM193/293/393
- Timers
  NE553/554/555/556

plus an assortment of popular voltage regulators, phase locked loops, amplifiers and other specialized circuits.

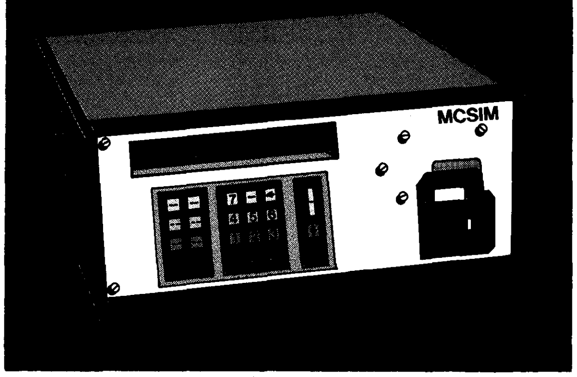
Selector guides are provided in the Bipolar Microprocessor book. For more information send for Signetics Analog manual for data sheets and application notes.

INTRODUCTORY BIPOLAR MICROPROCESSING DESIGNER'S KIT



An easy way to get a complete set of parts for a bit slice microprocessor is with this kit of parts. An 8-bit processor can be constructed using the four 3002's with microcontrol provided by the 3001 and two 82S114's contained in the kit. In addition you get two 8T26A bus transceivers and an 8T31 bidirectional I/O ports to round out the parts complement. For your start in bit slice microprocessors, get this $230 value with data book for $100. **Order 3000KT100SK.**

OUR BOOK DESCRIBES SMS McSIM AND OTHER DESIGN SUPPORT SYSTEMS.



Development of complicated microprocessor based systems and controllers can be difficult, but with the help of appropriate support systems the work can be considerably eased. Signetics presents its own as well as commercially available support products for the new Bipolar Microprocessor book. Send for one on your company letterhead today. We'll tell you about the 8X300 Microcontroller Simulator, PROM Programmers, FPLA Programmers, the Cross Assembler and other aids for system development that will be invaluable to you in both time and money.

OUR MICROPROCESSING ANSWERMEN KNOW ALL.

Signetics "Answermen" are there to solve your problems not just sell you a microprocessor. Microprocessor Application Specialists, 9 in all, located in strategic parts of the country serve customers in all corners of the continental United States. Well versed in Signetics MOS, bipolar bit slice and bipolar microcontroller microprocessors through regular intensive factory training, these "Answermen" have no limit in technology they may draw upon to provide you with a cost effective approach to your task. They may suggest the 2650, the 3002, the 2901-1 or the 8X300 in combination with System Logic, Interface and support circuits in an architecture that can do the job with the least cost. And for more difficult problems, the "Answermen" may draw on the factory staff of application experts. Call an "Answerman" for a discussion of your specific needs or for a seminar on the Signetics product line. "Answermen" are located in:

Sunnyvale
Irvine
Minneapolis
Boston
New York
Philadelphia
Los Angeles
Pompano Beach
Dallas

TABLE OF CONTENTS

CHAPTER I
BIPOLAR BIT
SLICE FAMILY

BIT-SLICE MICROPROCESSOR SERIES

Microcontrol and Arithmetic Units

The introduction of the Signetics Bit-Slice Microprocessors has brought new levels of high performance to microprocessor applications not previously possible with MOS technology. Combining the Schottky bipolar microprocessors with industry standard memory and support circuits, microinstruction cycle times of 100ns are possible.

In the majority of cases, the choice of a bipolar microprocessor slice, as opposed to an MOS device, is based on speed or flexibility of microprogramming. Starting with these characteristics, the design of the Signetics slice microprocessors has been optimized around the following objectives:

- Fast cycle time
- All memory and support chips are industry standard
- Cooler operation
- Lower total system cost

Furthermore, systems built with large-scale integrated circuits are much smaller and require less power than equivalent systems using medium and/or small scale integrated circuits.

Typically, slice microprocessors are employed in the realization of the Central Processing Unit (CPU) of a computer or for implementing dedicated smart controllers. The generalized and simplified structure of a CPU or "Smart" controller can be typically classified into 3 distinct but interactively related functional sections. These sections are generally referred to as the Processing section, the Control section, and the I/O and Memory Interface section. A simplified block diagram of a CPU is illustrated in Figure 1.

The major functions of the Processing section are to:

- provide data transfer paths;
- manipulate data through logic and arithmetic operations;
- provide storage facilities such as a register file; and
- generate necessary status flags based on the kind of operation performed by the ALU.

A SIMPLIFIED CPU ORGANIZATION

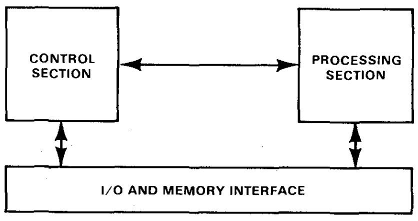


Figure 1

The major functions of the Control section are to:

- initiate memory or I/O operations;
- decode macroinstructions;
- control the manipulation and transfer of data;
- test status conditions; and
- sample and respond to interrupts.

The major functions of the I/O and Memory Interface section are to:

- multiplex data to the proper destination;
- provide bus driving/receiving capability; and
- provide latching capability.

With state-of-the-art bipolar Schottky technology, high-performance microprocessors are designed to perform functions of the Processing section. Due to the limitation on the number of pins and chip size, the overall Processing section is partitioned into several functionally equivalent slices. In today's bipolar microprocessor market, 2-bit and 4-bit slice architecture predominates. Each architecture type has its uniqueness but, in general, a slice contains a group of general purpose registers, an accumulator, special-purpose register(s) ALU and related status flags. All of these elements constitute the Processing section of a CPU. The flexibility of slice components allows the designer to construct a processing section of any desired width as required by his application.

The Control section of the CPU is more complex in design. Typically this section includes the macroinstruction decode logic, test-branch decode, microprogram sequencing logic, and the control store where the microprogram resides. Aside from the microprogram, the remaining portion of the Control section (macroinstruction decode and test-branch decode and sequencing logic), does not lend itself to efficient partitioning into vertical slices. This is due to the random nature of the logic usually found in the Control section. However, horizontal functional grouping is possible. For example, the macroinstruction decode and test-branch decode logic can now be replaced by the FPLA (Field Programmable Logic Array); the random logic traditionally needed to implement the microprogram sequencing can now be replaced by the Microprogram Control Unit; and, of course, the microprogram can be stored in high density PROMs or ROMs. Since the designer must define his own microstructure, the slice microprocessors permit fundamental optimizations to be made. With slice hardware, the designer may have no macroinstructions at all, placing all of the program in PROM for dedicated control applications. Or he may define, as required, any number of macroinstructions selected specifically for his particular processor purpose. Various minicomputers and several MOS microprocessors have been emulated using slice hardware.

The I/O and Memory Interface section consists mainly of I/O ports, high power bus drivers, receivers, and some temporary register storage facilities. Bidirectional and tri-state devices are the most popular logic elements for implementing this interface structure.

Figure 2 shows an LSI approach to the implementation of the same generalized CPU structure indicated earlier.

The devices presented in this chapter represent Signetics line of slice microprocessor components. Included is the popular 3000 series Microprogram Control Unit and the 2-bit slice Central Processing Element. These Signetics devices feature improved performance specifications over 3000 series components available on the general market. Moreover, the unique Signetics XL plastic package design results in significantly cooler operation of the chip than was previously possible with other plastic package designs. This section also features the 8X02 Control Store Sequencer. This device may be used with any TTL compatible slice processing elements and features extreme ease of use. The 8 simple, yet powerful, instructions permit subroutining and looping (using internal stack), unrestricted jumping, unrestricted conditional branching and conditional instruction skipping.

AN LSI IMPLEMENTATION OF A CPU

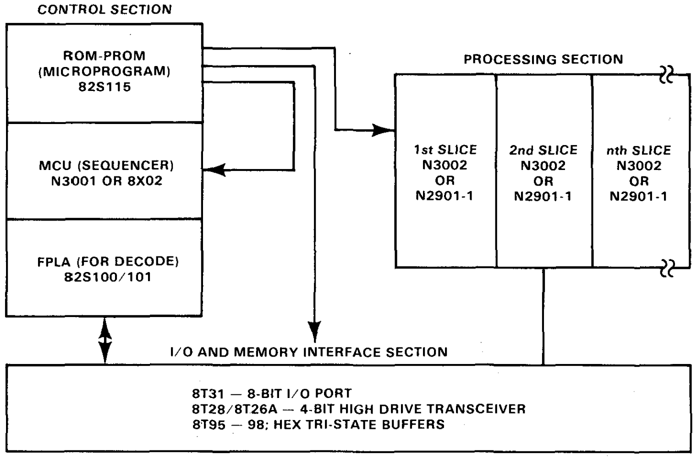

CONTROL SECTION
ROM-PROM (MICROPROGRAM) 82S115
MCU (SEQUENCER) N3001 OR 8X02
FPLA (FOR DECODE) 82S100/101
PROCESSING SECTION
1st SLICE N3002 OR N2901-1
2nd SLICE N3002 OR N2901-1
nth SLICE N3002 OR N2901-1
I/O AND MEMORY INTERFACE SECTION
8T31 — 8-BIT I/O PORT
8T28/8T26A — 4-BIT HIGH DRIVE TRANSCEIVER
8T95 — 98: HEX TRI-STATE BUFFERS

Figure 2

CONTROL STORE SEQUENCER 8X02

OBJECTIVE SPECIFICATION 8X02-XL,I

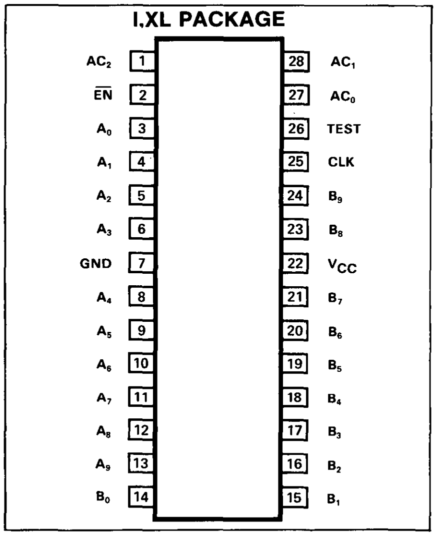
DESCRIPTION

The Signetics 8X02 is a Low-Power Schottky LSI device intended for use in high performance microprogrammed systems to control the fetch sequence of microinstructions. When combined with standard ROM or PROM, the 8X02 forms a powerful microprogrammed control section for computers, controllers, or sequenced logic.

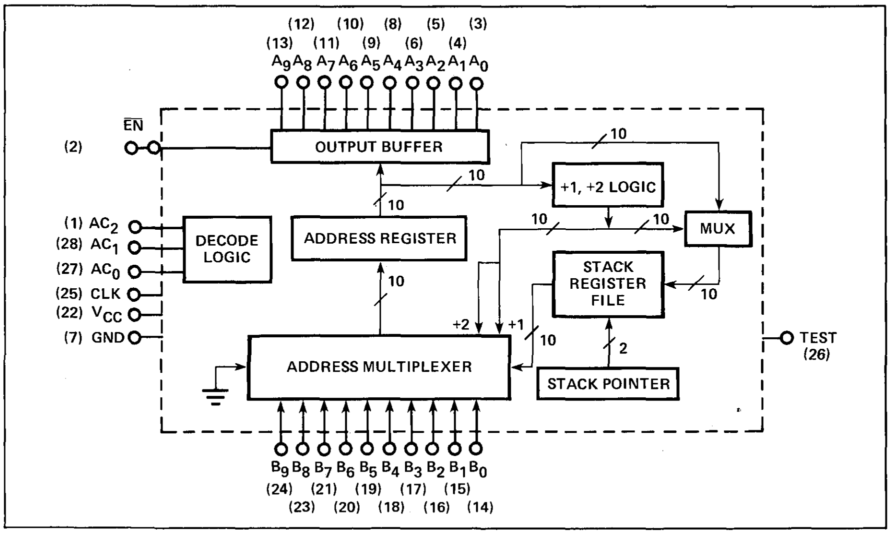
FEATURES

- **Low power Schottky process**
- **77ns cycle time (typ)**
- **1024 microinstruction addressability**
- **N-way branch**
- **4-level stack register file (LIFO type)**
- **Automatic push/pop stack operation**
- **"Test & skip" operation on test input line**
- **3-bit command code**
- **Tri-state buffered outputs**
- **Auto-reset to address 0 during power-up**
- **Conditional branching, pop stack, & push stack**

PIN CONFIGURATION



BLOCK DIAGRAM



PIN DESCRIPTION

PIN	SYMBOL	NAME AND FUNCTION	TYPE
5-6 8-13	A_0-A_9	Microprogram Address outputs	Three-state Active high
1,28,27	AC_0-AC_2	Next Address Control Function inputs All addressing control functions are selected by these command lines.	Active high
14-21 23-24	B_0-B_9	Branch Address inputs Determines the next address of an N-way branch when used with the BRANCH TO SUBROUTINE (BSR) or BRANCH ON TEST (BRT) command.	Active high
2	$\overline{EN}$	Enable input When in the low state, the Microprogram Address outputs are enabled.	Active low
25	CLK	Clock input All registers are triggered on the low-to-high transition of the clock.	
26	TEST	Test input Used in conjunction with four NEXT ADDRESS CONTROL FUNCTION commands to effect conditional skips, branches, and stack operations.	Active high
7	GND	Ground	
22	V_{CC}	+5 Volt supply	

FUNCTIONAL DESCRIPTION

The Signetics 8X02 Control Store Sequencer is an LSI device using Low Power Schottky technology and is intended for use in high performance microprogrammed applications. When used alone, the 8X02 is capable of addressing up to 1K words of microprogram. This may be expanded to any microprogram size by conventional paging techniques.

The Address Register consists of 10 D-type, edge-triggered flip-flops with a common clock. A new address is entered into the Address Register on the low-to-high transition of the clock. The next address to be entered into the Address Register is supplied via the Address Multiplexer.

The Address Multiplexer is a 5-input device that is used to select either the branch input, +1 adder, +2 adder, stack register file, or ground (all zeros) as the source of the next microinstruction address. The proper multiplexer channel is automatically selected via the Decode Logic according to the Address Control Function Input and Test Input line.

The +1, +2 logic is used to increment the present contents of the Address Register by 1 or 2, depending on the function input command. Thus, the next address to the Control Store ROM/PROM may be either the current address plus 1 (N+1) or the current address plus 2 (N+2). If the same Microprogram Address is to be used on successive occasions, the clock to the 8X02 must simply be disabled; therefore, no new address is loaded into the Address Register.

The Stack File Register is used to provide a return address linkage whenever a subroutine or loop is executed. The 4X10 stack operates in a last-in, first-out (LIFO) mode, with the stack pointer always pointing to the next address to be read. Operation of the stack pointer is automatically controlled by the Address Control Function Inputs. Since the stack is 4 words deep, up to 4 loops and/or subroutines may be nested.

The branch input is a 10-bit field of direct inputs to the multiplexer which can be selected as the next control store address. Using the appropriate branch command, an N-way branch is possible where N is the address of any microinstruction within the 1024 word microcode page. Likewise, the RESET command is a special case of an N-way branch in which the multiplexer selects an all zeros input, forcing the next microinstruction address to be zero.

The Test Input line is used in conjunction with the conditional execution of 4 Address Control Function commands. When the Test Input is false (low), the sequencer simply increments to the next address (N+1). When it is true (high), the sequencer executes a branch as defined by the input command, thereby transferring control to another portion of the microprogram.

All Address Output lines of the 8X02 are three-state buffered outputs with a common enable line ($\overline{EN}$). When the Enable line is high, all outputs are placed in a high-impedance state, and external access to the control store ROM/PROM is possible. This allows a preprogrammed set of microinstructions to be executed from external or built-in test equipment (BITE), vectored interrupts, and Writable Control Store if implemented.

NEXT ADDRESS CONTROL FUNCTION

MNEMONIC	DESCRIPTION	FUNCTION AC_2 1 0	TEST	NEXT ADDRESS	STACK	STACK POINTER
TSK	Test & skip	0 0 0	False True	Current + 1 Current + 2	N.C. N.C.	N.C. N.C.
INC	Increment	0 0 1	X	Current + 1	N.C.	N.C.
BLT	Branch to loop if test input true	0 1 0	False True	Current + 1 Stack reg file	X POP (read)	Decr Decr
POP	POP stack	0 1 1	X	Stack reg file	POP (read)	Decr
BSR	Branch to subroutine if test input true	1 0 0	False True	Current + 1 Branch address	N.C. PUSH (Curr + 1)	N.C. Incr
PLP	Push for looping	1 0 1	X	Current + 1	PUSH (Curr Addr)	Incr
BRT	Branch if test input true	1 1 0	False True	Current + 1 Branch address	N.C. N.C.	N.C. N.C.
RST	Set microprogram address output to zero	1 1 1	X	All O's	N.C.	N.C.

X = Don't care
N.C. = No change

FUNCTIONAL DESCRIPTION

The following is a description of each of the eight Next Address Control Functions (AC_2-AC_0)

MNEMONIC	FUNCTION DESCRIPTION
TSK	AC_{2-0} = 000: TEST & SKIP Perform test on TEST INPUT LINE. If test is FALSE (LOW): Next Address = Current Address + 1 Stack Pointer unchanged If test is TRUE (HIGH) Next Address = Current Address + 2 (i.e. Skip next microinstruction) Stack Pointer unchanged
INC	AC_{2-0} = 001: INCREMENT Next Address = Current Address + 1 Stack Pointer unchanged
BLT	AC_{2-0} = 010: BRANCH TO LOOP IF TEST CONDITION TRUE. Perform test on TEST INPUT LINE. If test is FALSE (LOW): Next Address = Current Address + 1 Stack Pointer decremented by 1 If test is TRUE (HIGH): Next Address = Address from Stack Register File (POP) Stack Pointer decremented by 1
POP	AC_{2-0} = 011: POP STACK Next Address = Address from Stack Register File (POP) Stack Pointer decremented by 1
BSR	AC_{2-0} = 100: BRANCH TO SUBROUTINE IF TEST CONDITION TRUE. Perform test on TEST INPUT LINE. If test is FALSE (LOW): Next Address = Current Address + 1 Stack Pointer unchanged If test is TRUE (HIGH): Next Address = Branch Address Input (B_{0-9}) Stack Pointer incremented by 1 PUSH (write) Current Address + 1 → Stack Register File
PLP	AC_{2-0} = 101: PUSH FOR LOOPING Next Address = Current Address + 1 Stack Pointer incremented by 1 PUSH (write) Current Address → Stack Register File
BRT	AC_{2-0} = 110: BRANCH ON TEST CONDITION TRUE Perform test on TEST INPUT LINE. If test is FALSE (LOW): Next Address = Current Address + 1 Stack Pointer unchanged If test is TRUE (HIGH): Next Address = Branch Address Input (B_{0-9}) Stack Pointer unchanged
RST	AC_{2-0} = 111: RESET TO ZERO Next Address = 0 Stack Pointer unchanged

ABSOLUTE MAXIMUM RATINGS

PARAMETER		RATING	UNIT
V_{CC}	Power supply voltage	+7	Vdc
V_{IN}	Input voltage	+5.5	Vdc
V_O	Off-State output voltage	+5.5	Vdc
T_A	Operating temperature range	0° to +70°	°C
T_{stg}	Storage temperature range	-65° to +150°	°C

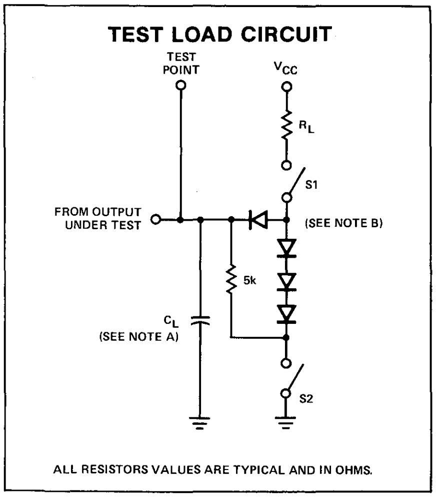
DC ELECTRICAL CHARACTERISTICS 0°C ⩽ +70°C, 4.75V, VCC ⩽ 5.25V

PARAMETER		TEST CONDITIONS	LIMITS			UNIT
			Min	Typ[1]	Max	
V_{IH}	High level input voltage		2			V
V_{IL}	Low level input voltage				0.8	V
V_I	Input clamp voltage	V_{CC} = 4.75V, I_I = -18mA			-1.5	V
V_{OH}	High level output voltage	V_{CC} = 4.75V, I_{OH} = -2.6mA	2.4			V
V_{OL}	Low level output voltage	V_{CC} = 4.75V, I_{OL} = 8mA			0.5	V
I_I	Input current at maximum Input voltage	V_{CC} = 5.25V, V_I = 5.5V			100	μA
I_{IH}	High level input current AC_2-AC_0, $\overline{EN}$, TEST B_9-B_0 CLK	V_{CC} = 5.25V, V_I = 2.7V			 40 20 60	 μA μA μA
I_{IL}	Low level input current AC_2-AC_0, $\overline{EN}$, TEST B_9-B_0 CLK	V_{CC} = 5.25V, V_I = 0.4V			 -0.72 -0.36 -1.08	 mA mA mA
I_{OS}	Short-circuit output current	V_{CC} = 5.25V	-20		-100	mA
I_{OZH}	High-Z state output current	V_{OUT} = 2.7V			20	μA
I_{OZL}	High-Z state output current	V_{OUT} = 0.4V			-20	μA
I_{CC}	Supply current	V_{CC} = 5.25V		130	155	mA

NOTE

1. All typical values are at V_{CC} = 5V, T_A = 25°C.

PARAMETER MEASUREMENT INFORMATION

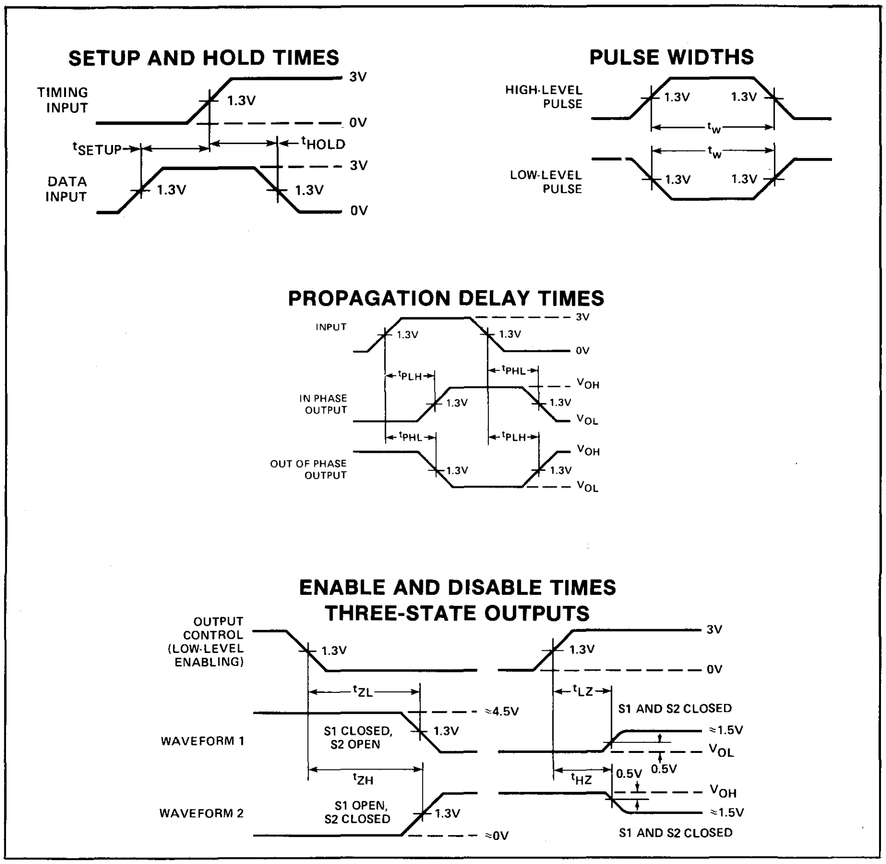


TEST LOAD CIRCUIT

ALL RESISTORS VALUES ARE TYPICAL AND IN OHMS.

NOTES

A. C_L includes probe and jig capacitance.
B. All diodes are 1N916 or 1N3064.
C. R_L = 2k, C = 15pF.



SETUP AND HOLD TIMES

PULSE WIDTHS

PROPAGATION DELAY TIMES

ENABLE AND DISABLE TIMES THREE-STATE OUTPUTS

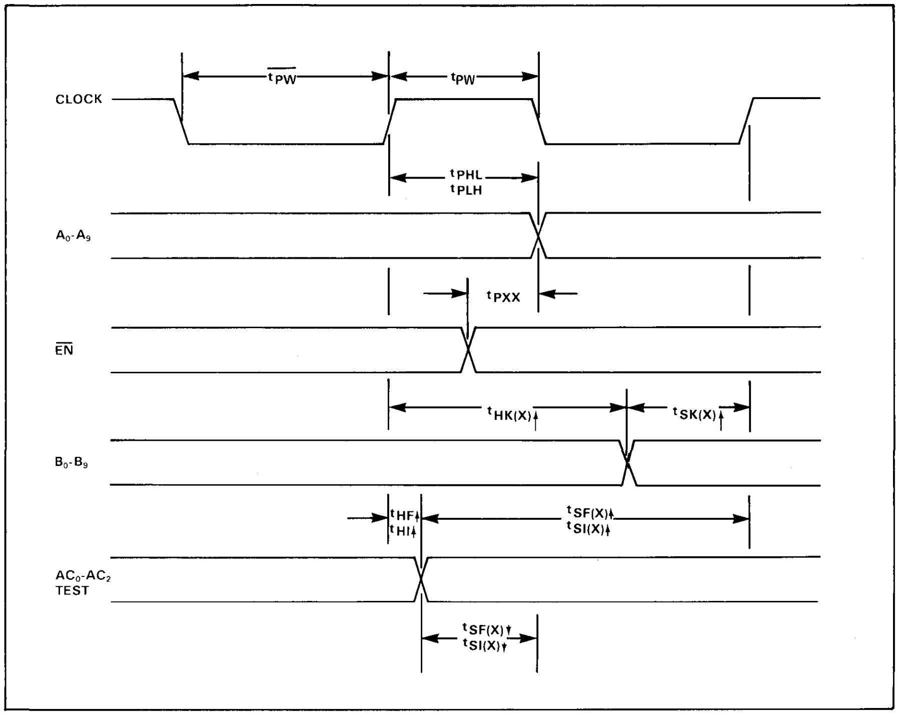
AC ELECTRICAL CHARACTERISTICS TA = 0°C to +70°C, VCC = 5.0V ± 5%

PARAMETER	FROM INPUT	TO OUTPUT	LIMITS			UNIT
			Min	Typ[1]	Max	
t_{cy} Cycle time				77		ns
t_{pw} Clock pulse width high				32		ns
t_{pw} Clock pulse width low				45		ns
Enable delay						
t_{PLZ} Low-to-high-Z	EN	A_9-A_0		12		ns
t_{PHZ} High-to-high-Z				16		ns
t_{PZL} High-Z-to-low				14		ns
t_{PZH} High-Z-to-high				8		ns
Propagation Delay						
t_{PHL} High-to-low	CLK	A_9-A_0		32		
t_{PLH} Low-to-high				20		ns
Set-up and Hold times With respect to CLK (↑)						
t_{SF} Set-up time high	Control, Data	AC_2-AC_0		70		ns
t_{HF} Hold time high				-6		ns
t_{SF} Set-up time high	Control, Data	B_9-B_0		22		ns
t_{HF} Hold time high				-15		ns
t_{SI} Set-up time high	Control, Data	Test		70		ns
t_{HI} Hold time high				-6		ns
t_{SF} Set-up time low	Control, Data	AC_2-AC_0		70		ns
t_{HF} Hold time low				8		ns
t_{SK} Set-up time low	Control, Data	B_9-B_0		24		ns
t_{HK} Hold time low				-14		ns
t_{SI} Set-up time low	Control, Data	Test		70		ns
t_{HI} Hold time low				8		ns
t_S Set-up time high	Function	BRT/BSR		50		ns
t_S Set-up time low				37		ns
t_S Set-up time high	Function	TSK		52		ns
t_S Set-up time low				70		ns
t_S Set-up time high	Function	INC		32		ns
t_S Set-up time high	Function	RST		27		ns
t_S Set-up time high	Function	POP/BLT		70		ns
With respect to CLK (↓)						
t_{SF} Set-up time high	Control, Data	AC_2-AC_0		23		ns
t_{SI} Set-up time high	Control, Data	Test		23		ns
t_{SF} Set-up time low	Control, Data	AC_2-AC_0		22		ns
t_{SI} Set-up time low	Control, Data	Test		22		ns

NOTE

[1] Typical values are for TA = 25°C, VCC = 5.0V.

VOLTAGE WAVEFORMS



INTRODUCTION

The introduction of the Signetics Series 3000 Bipolar Microprocessor Chip Set has brought new levels of high performance to microprocessor applications not previously possible with MOS technology. Combining the Schottky bipolar N3001 Microprogram Control Unit (MCU) and N3002 Central Processing Element (CPE) with industry standard memory and support circuits, microinstruction cycle times of 100ns are possible.

In the majority of cases, the choice of a bipolar microprocessor slice, as opposed to an MOS device, is based on speed or flexibility of microprogramming. Starting with these characteristics, the design of the Signetics Series 3000 Microprocessor has been optimized around the following objectives:

- Fast cycle time
- All memory and support chips are industry standard
- Cooler operation
- Lower total system cost

Furthermore, systems built with large-scale integrated circuits are much smaller and require less power than equivalent systems using medium and/or small scale integrated circuits.

The 2 components of the Series 3000 chip set, when combined with industry standard memory and peripheral circuits, allows the design engineer to construct high-performance processors and/or controllers with a minimum amount of auxillary logic. Features such as the multiple independent address and data buses, tri-state logic, and separate output enable lines eliminate the need for time-multiplexing of buses and associated hardware.

Each Central Processing Element represents a complete 2-bit slice through the data processing section of a computer. Several CPEs may be connected in parallel to form a processor of any desired word length. The Microprogram Control Unit controls the sequence in which microinstructions are fetched from the microprogram memory (ROM/PROM), with these microinstructions controlling the step-by-step operation of the processor.

Each CPE contains a 2-bit slice of 5 independent buses. Although they can be used in a variety of ways, typical connections are:

Input M-bus:	Carries data from external memory
Input I-bus:	Carries data from input/output device
Input K-bus:	Used for microprogram mask or literal (constant) value input
Output A-bus:	Connected to CPE Memory Address Register
Output D-bus:	Connected to CPE accumulator.

As the CPEs are paralleled together, all buses, data paths, and registers are correspondingly expanded.

The microfunction input bus (F-bus) controls the internal operation of the CPE, selecting both the operands and the operation to be executed upon them. The arithmetic logic unit (ALU), controlled by the microfunction decoder, is capable of over 40 Boolean and binary operations as outlined in the Function Description section of the N3002 data sheet. Standard carry look-ahead outputs (X and Y) are generated by the CPE for use with industry standard devices such as the 74S182.

A typical processor configuration is shown in Figure 1. It should be remembered that in working with slice-oriented microprocessors, the final configuration may be varied to enhance speed, reduce component count, or increase data-processing capability. One method of maximizing a processor's performance is called pipelining. To accomplish this, a group of D-type flip-flops or latches (such as the 74174 Hex D-type Flip-Flop) are connected to the microprogram memory outputs (excluding the address control field AC_0-AC_6) to buffer the current microinstruction and allow the MCU to overlap the fetch of the next instruction with the execution of the current one. The time saved in pipelining operations is the shorter of either the address set-up time to the microprogram memory (ROM/PROM) or the access time of the ROM/PROM. A convenient way of implementing pipelining is to use ROMs with on-board latches, such as the Signetics 82S115.

MICROCOMPUTER BLOCK DIAGRAM

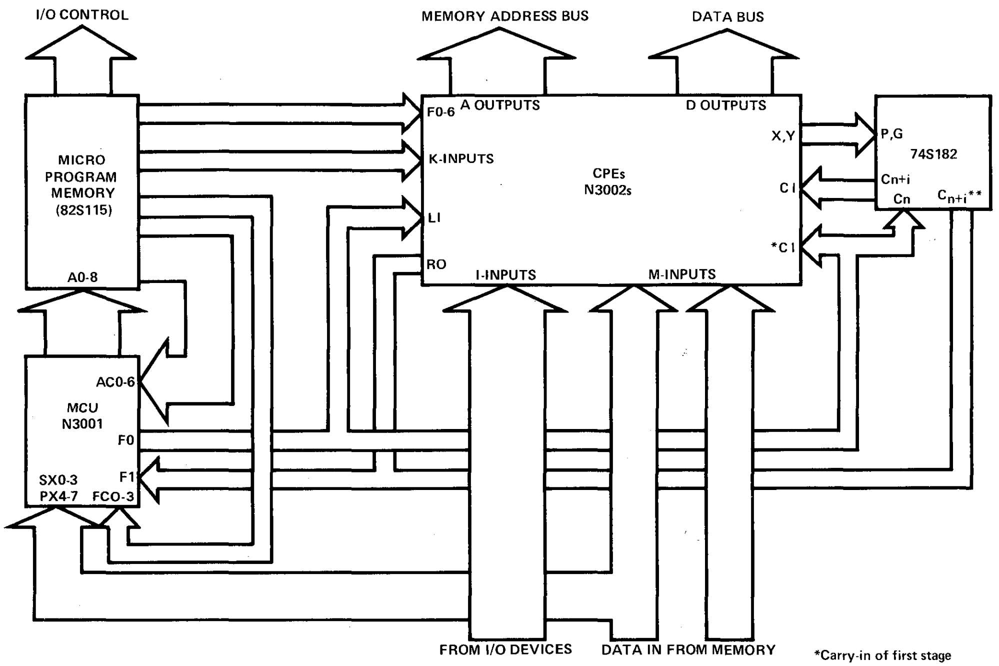


Figure 1

*Carry-in of first stage
**Carry-out of last stage

Figure 2 shows a typical microinstruction format using the 82S114 PROMs contained in the Signetics 3000 Microprocessor Designer's Evaluation Kit. Although this particular example is for a 48-bit word (6 PROMs), the allocation of bits for the mask (K-bus) and optional processor functions depends on the specific application of the system and the trade offs which the designer wishes to make.

In using the K-bus, it should be kept in mind that the K inputs are always ANDed with the B-multiplexer outputs into the ALU. Bit masking, frequently done in computer control systems, can be performed with the mask supplied to the K-bus directly from the microinstruction.

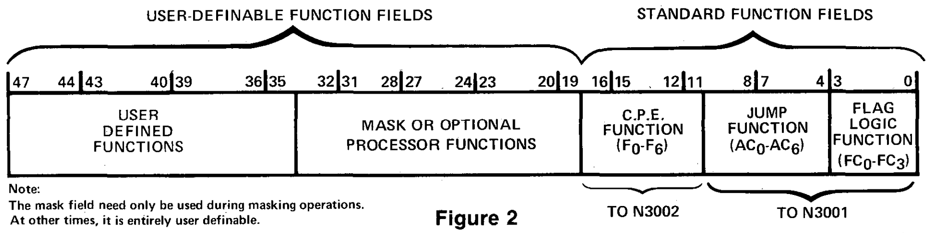
By placing the K-bus in either the all-one or all-zero condition (done with a single control bit in the microinstruction), the accumulator will either be selected or deselected, respectively, in a given operation. This feature nearly doubles the amount of microfunctions in the CPE. A description of these various microfunctions can be found in the N3002 data sheet under the heading Function Description by referring to the K-bus conditions of all-ones (11) and all-zeros (00).

The MCU controls the sequence in which microinstructions are fetched from the microprogram memory (ROM/PROM). In its classical form, the MCU would use a next-address field in each microinstruction. However, the N3001 uses a modified classical approach in which the microinstruction field specifies conditional tests on the MCU bus inputs and registers. The next-address logic of the MCU also makes extensive use of a row/column addressing scheme, whereby the next address is defined by a 5-bit row address and 4-bit column address. Thus, from a particular address location, it is possible to jump unconditionally to any location within that row or column, or conditionally to other specified locations in one operation. Using this method, the processor functions can be executed in parallel with program branches.

As an example of this flexibility, let us assume a disk controller is being designed. As part of the sequence logic, 3 bits of the disk drive status word must be tested and all 3 must be true in order to proceed with the particular sequencing operation. In any sequencing operation using a status word for conditional branch information, there are innumerable combinations of bits which must be tested throughout the sequencing operation. Using discrete logic techniques, this would involve several levels of gating. However, the entire operation can be done in two microinstructions. First, the mask (K-bus) field in the microinstruction format is encoded with a one for each corresponding status bit to be tested and a zero for each bit to be discarded. The status word is input via the I-bus and ANDed with the K-bus mask using the CPE microfunction operation from F-Group 2, R-Group III. Assuming we are using low-true logic (true = 0 volts), we now test the result, which is located in the accumulator AC, for all zeros using the CPE microfunction operation from F-Group 5, R-Group III. Depending on the zero/non-zero status of AC, a one or zero will be loaded into the carryout CO bit. This bit can now be used as a condition for the next address jump calculation within the N3001 MCU. If the AC was zero (status word was true), we will jump to the next address within our controller sequence. If the AC was non-zero (status word not true), then a jump would be made back to the beginning of this 2-microinstruction loop and the test sequence repeated until the status word (all 3 bits) is true.

Figure 3 shows a typical timing diagram for a system operating in the non-pipelined mode. Keep in mind that the maximum clock rate is dependent upon the total of propagation delay times plus required set-up times. It is at the designer's discretion to resolve the speed versus complexity trade-offs.

TYPICAL MICROINSTRUCTION FORMAT


USER-DEFINABLE FUNCTION FIELDS
STANDARD FUNCTION FIELDS
47 44 43 40 39 36 35 32 31 28 27 24 23 20 19 16 15 12 11 8 7 4 3 0
USER DEFINED FUNCTIONS
MASK OR OPTIONAL PROCESSOR FUNCTIONS
C.P.E. FUNCTION (F_0-F_6)
JUMP FUNCTION (AC_0-AC_6)
FLAG LOGIC FUNCTION (FC_0-FC_3)
TO N3002
TO N3001

Note:
The mask field need only be used during masking operations.
At other times, it is entirely user definable.

Figure 2

SYSTEM TIMING—NON-PIPELINED CONFIGURATION

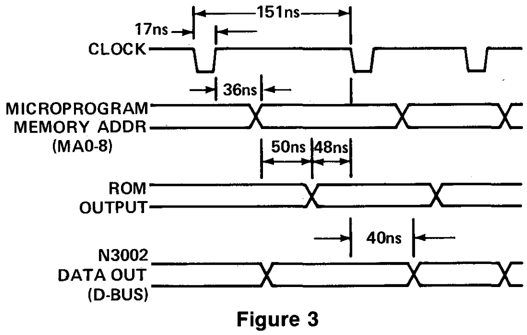


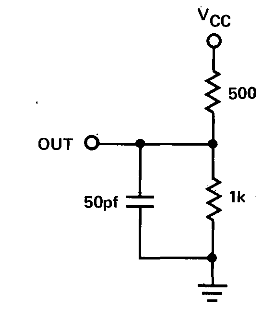
Figure 3

ABSOLUTE MAXIMUM RATINGS*

	N3001/N3002	S3001/S3002
Temperature under bias	0°C to +70°C	-55°C to +125°C
Storage temperature	-60°C to +160°C	-65°C to +150°C
All output and supply voltages	-0.5V to +7V	-0.5V to +7V
All input voltages	-1.0V to +5.5V	-1.0V to +5.5V
Output currents	100mA	100mA

*Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of the specification is not implied. Exposure to absolute maximum ratings for extended periods may effect device reliability.

PARAMETER MEASUREMENT INFORMATION



NOTE: All resistor values are typical and in ohms.
TEST CONDITIONS:
Input pulse amplitude of 2.5 volts
Input rise and fall times of 5ns between 1 volt and 2 volts
Output load of 10mA and 50pF
Speed measurements are taken at the 1.5 volt level

DC ELECTRICAL CHARACTERISTICS

N3001/N3002 T_A = 0°C to +70°C
S3001/S3002 T_A = -55°C to +125°C

PARAMETER	TEST CONDITIONS	N3001/N3002			S3001/S3002			UNIT
		Min	Typ	Max	Min	Typ	Max	
V_{IL} Low level input voltage	V_{CC} = 5.0V			0.8			0.8	V
V_{IH} High level input voltage	V_{CC} = 5.0V	2.0			2.0			V
V_{IC} Input clamp voltage	V_{CC} = 4.75V, I_C = -5mA		-0.55	-1.0		-0.8	-1.2	V
V_{OL} Low level output voltage	V_{CC} = 4.75V, I_{OL} =10mA		0.35	0.45		0.35	0.45	V
V_{OH} High level output voltage	V_{CC} = 4.75V, I_{OH} = -1mA mA_0 - mA_8, ISE, FO		2.4	3.0		2.4	3.0	V
I_F Input load current N3001	V_{CC} = 5.25V, V_F = 0.45V							
	CLK input		-0.21	-0.75		-0.21	-0.75	mA
	EN input		-0.12	-0.50		-0.12	-0.50	mA
	All other inputs		-0.05	-0.25		-0.05	-0.25	mA
I_R Input leakage current N3001	V_{CC} = 5.25V, V_R = 5.25V					V_R = 5.5V		
	CLK input			120			120	µA
	EN input			80			80	µA
	All other inputs			40			40	µA
I_{OS} Short circuit output current	V_{CC} = 5.0V mA_0 - mA_8, ISE, FO	-15	-28	-60	-15	-28	-60	mA
I_O (off) Off-state output current	V_{CC} = 5.25V							
	PR_0 - PR_2, mA_0 - mA_2, FO							
	V_{OUT} = 0.45V			-100			-100	µA
	mA_0 - mA_8, FO					V_{OUT} = 5.5V		
	V_{OUT} = 5.25V			+100			+100	µA
I_{CC} Power supply current N3001	V_{CC} = 5.25V^2		170	240		170	250	mA
Power supply current N3002			145	190		145	210	
I_F Input Load Current N3002	V_{CC} = 5.25V, V_F = 0.45v							
	F_0-F_6, CLK, K_0, K_1, EA, ED		-0.05	-0.25		-0.05	-0.25	mA
	I_0, I_1, M_0, M_1, LI		-0.85	-1.5		-0.85	-1.5	mA
	CI		-2.3	-4.0		-2.3	-4.0	mA
I_R Input Leakage Current N3002	V_{CC} = 5.25V, V_R = 5.25V							
	F_0-F_6, CLK, K_0, K_1, EA, ED			40			40	µA
	I_0, I_1, M_0, M_1, LI			60			60	µA
	CI			180			180	µA

NOTES
1. SN3001 typical values are for T_A = 25°C, V_{CC} = 5.0V
2. SN3002 EN input grounded, all other inputs and outputs open.
SN3002 CLK input grounded, other inputs open.

MICROPROGRAM CONTROL UNIT S/N3001

N3001-I

DESCRIPTION

The N3001 MCU is 1 element of a bipolar microcomputer set. When used with the S/N3002, 54/74S182, ROM or PROM memory, a powerful microprogrammed computer can be implemented.

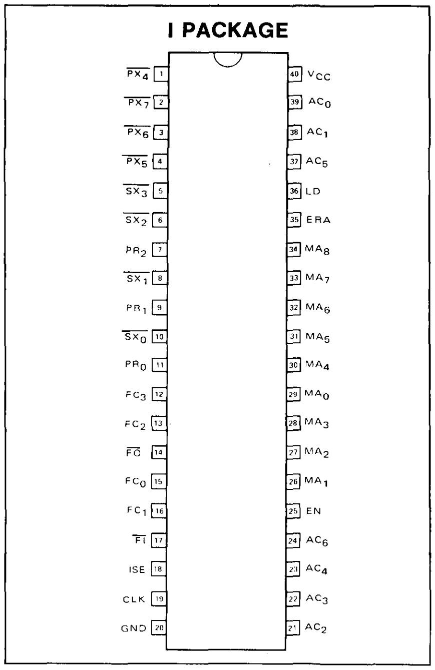
The 3001 MCU controls the fetch sequence of microinstructions from the microprogram memory. Functions performed by the 3001 include:

- Maintenance of microprogram address register
- Selection of next microinstruction address
- Decoding and testing of data supplied via several input buses
- Saving and testing of carry output data from the central processing (CP) array
- Control of carry/shift input data to the CP array
- Control of microprogram interrupts

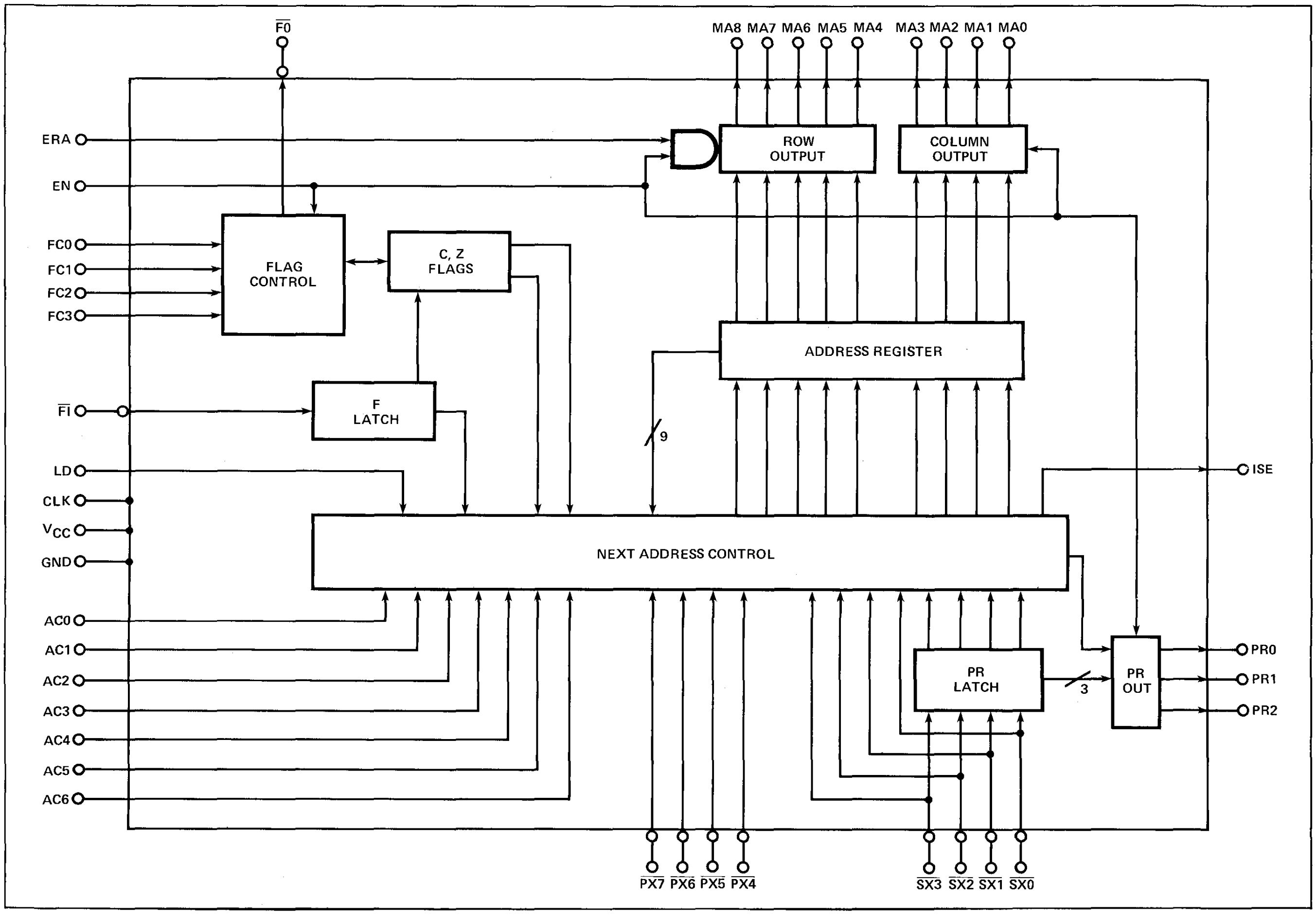
FEATURES

- **Schottky TTL process**
- **45ns cycle time (typ.)**
- **Direct addressing of standard bipolar PROM or ROM**
- **512 microinstruction addressability**
- **Advanced organization:**
  - **9-bit microprogram address register and bus organized to address memory by row and column**
  - **4-bit program latch**
  - **2-flag registers**
- **11 address control functions:**
  - **3 jump and test latch function**
  - **16 way jump and test instruction**
- **8 flag control functions:**
  - **4 flag input functions**
  - **4 flag output functions**

PIN CONFIGURATION


I PACKAGE

BLOCK DIAGRAM



PIN DESCRIPTION

PIN	SYMBOL	NAME AND FUNCTION	TYPE
1-4	$\overline{PX_4}$-$\overline{PX_7}$	Primary Instruction Bus Inputs Data on the primary instruction bus is tested by the JPX function to determine the next microprogram address.	Active low
5,6,8,10	$\overline{SX_0}$-$\overline{SX_3}$	Secondary Instruction Bus Inputs Data on the secondary instruction bus is synchronously loaded into the PR-latch while the data on the PX-bus is being tested (JPX). During a subsequent cycle, the contents of the PR-latch may be tested by the JPR, JLL, or JRL functions to determine the next microprogram address.	Active low
7,9,11	PR_0-PR_2	PR-Latch Outputs The PR-latch outputs (SX_0-SX_2) are synchronously enabled by the JCE function. They can be used to modify microinstructions at the outputs of the microprogram memory or to provide additional control lines.	Open Collector
12,13 15,16	FC_0-FC_3	Flag Logic Control Inputs The flat logic control inputs are used to cross-switch the flags (C and Z) with the flag logic input (FI) and the flag logic output (FO).	Active high
14	$\overline{FO}$	Flag Logic Output The outputs of the flags (C and Z) are multiplexed internally to form the common flag logic output. The output may also be forced to a logical 0 or logical 1.	Active low Three-state
17	$\overline{FI}$	Flag Logic Input The flag logic input is demultiplexed internally and applied to the inputs of the flags (C and Z). Note: The flag input data is saved in the F-latch when the clock input (CLK) is low.	Active low
18	ISE	Interrupt Strobe Enable Output The interrupt strobe enable output goes to logical 1 when one of the JZR functions are selected (see Functional Description). It can be used to provide the strobe signal required by interrupt circuits.	Active high
19	CLK	Clock Input	
20	GND	Ground	
21-24 37-39	AC_0-AC_6	Next Address Control Function Inputs All jump functions are selected by these control lines.	Active high
25	EN	Enable Input When in the high state, the enable input enables the microprogram address, PR-latch and flag outputs.	
26-29	MA_0-MA_3	Microprogram Column Address Outputs	Three-state
30-34	MA_4-MA_8	Microprogram Row Address Outputs	Three-state
35	ERA	Enable Row Address Input When in the low state, the enable row address input independently disables the microprogram row address outputs. It can be used to facilitate the implementation of priority interrupt systems.	Active high
36	LD	Microprogram Address Load Input When the active high state, the microprogram address load input inhibits all jump functions and synchronously loads the data on the instruction buses into the microprogram address register. However, it does not inhibit the operation of the PR-latch or the generation of the interrupt strobe enable.	Active high
40	V_{CC}	+5 Volt supply	

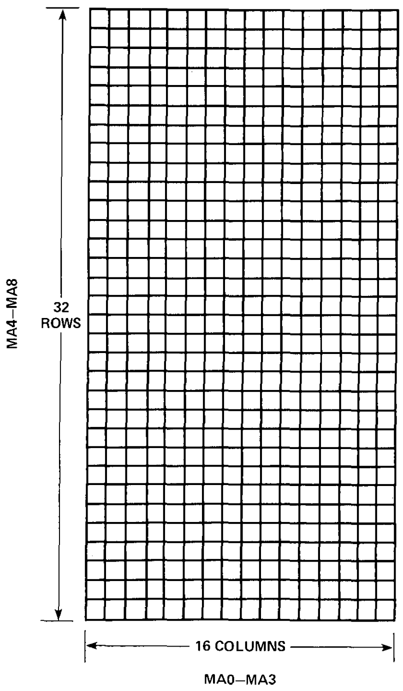
THEORY OF OPERATION

The MCU controls the sequence of microinstructions in the microprogram memory. The MCU simultaneously controls 2 flip-flops (C, Z) which are interactive with the carry-in and carry-out logic of an array of CPEs.

The functional control of the MCU provides both unconditional jumps to new memory locations and jumps which are dependent on the state of MCU flags or the state of the "PR" latch. Each instruction has a "jump set" associated with it. This "jump set" is the total group of memory locations which can be addressed by that instruction.

The MCU utilizes a two-dimensional addressing scheme in the microprogram memory. Microprogram memory is organized as 32 rows and 16 columns for a total of 512 words. Word length is variable according to application. Address is accomplished by a 9-bit address organized as a 5-bit row and 4-bit column address.

ADDRESSING ORGANIZATION



FUNCTIONAL DESCRIPTION

The following is a description of each of the eleven address control functions. The symbols shown below are used to specify row and column addresses.

SYMBOL	MEANING
row_n	5-bit next row address where n is the decimal row address.
col_n	4-bit next column address where n is the decimal column address.

Unconditional Address Control (Jump) Functions

The jump functions use the current microprogram address (i.e., the contents of the microprogram address register prior to the rising edge of the clock) and several bits from the address control inputs (AC0-AC6) to generate the next microprogram address.

Flag Conditional Address Control (Jump Test) Functions

The jump/test flag functions use the current microprogram address, the contents of the selected flag or latch, and several bits from the address control function to generate the next microprogram address.

JUMP FUNCTION TABLE

MNEMONIC	FUNCTION DESCRIPTION
JCC	Jump in current column. AC_0-AC_4 are used to select 1 of 32 row addresses in the current column, specified by MA_0-MA_3, as the next address.
JZR	Jump to zero row. AC_0-AC_3 are used to select 1 of 16 column addresses in row_0, as the next address.
JCR	Jump in current row. AC_0-AC_3 are used to select 1 of 16 addresses in the current row, specified by MA_4-MA_8, as the next address.
JCE	Jump in current column/row group and enable PR-latch outputs. AC_0-AC_2 are used to select 1 of 8 row addresses in the current row group, specified by MA_7-MA_8, as the next row address. The current column is specified by MA_0-MA_3. The PR-latch outputs are asynchronously enabled.

JUMP/TEST FUNCTION TABLE

MNEMONIC	FUNCTION DESCRIPTION
JFL	Jump/test F-latch. AC_0-AC_3 are used to select 1 of 16 row addresses in the current row group, specified by MA_8, as the next row address. If the current column group, specified by MA_3, is col_0-col_7, the F-latch is used to select col_2 or col_3 as the next column address. If MA_3 specifies column group col_8-col_{15}, the F-latch is used to select col_{10} or col_{11} as the next column address.
JCF	Jump/test C-flag. AC_0-AC_2 are used to select 1 of 8 row addresses in the current row group, specified by MA_7 and MA_8, as the next row address. If the current column group specified by MA_3 is col_0-col_7, the C-flag is used to select col_2 or col_3 as the next column address. If MA_3 specifies column group col_8-col_{15}, the C-flag is used to select col_{10} or col_{11} as the next column address.
JZF	Jump/test Z-flag. Identical to the JCF function described above, except that the Z-flag, rather than the C-flag, is used to select the next column address.
JPR	Jump/test PR-latch. AC_0-AC_2 are used to select 1 of 8 row addresses in the current row group, specified by MA_7 and MA_8, as the next row address. The 4 PR-latch bits are used to select 1 of 16 possible column addresses as the next column address.
JLL	Jump/test leftmost PR-latch bits. AC_0-AC_2 are used to select 1 of 8 row addresses in the current row group, specified by MA_7 and MA_8, as the next row address. PR_2 and PR_3 are used to select 1 of 4 column addresses in col_4 through col_7 as the next column address.
JRL	Jump/test rightmost PR-latch bits. AC_0 and AC_1 are used to select 1 of 4 high-order row addresses in the current row group, specified by MA_7 and MA_8, as the next row address. PR_0 and PR_1 are used to select 1 of 4 possible column addresses in col_{12} through col_{16} as the next column address.
JPX	Jump/test PX-bus and load PR-latch. AC_0 and AC_1 are used to select 1 of 4 row addresses in the current row group, specified by MA_6-MA_8, as the next row address. PX_4-PX_7 are used to select 1 of 16 possible column addresses as the next column address. SX_0-SX_3 data is locked in the PR-latch at the rising edge of the clock.

PX-Bus and PR-Latch Conditional Address Control (Jump/Test) Functions

The PX-bus jump/test function uses the data on the primary instruction bus (PX$_4$-PX$_7$), the current microprogram address, and several selection bits from the address control function to generate the next microprogram address. The PR-latch jump/test functions use the data held in the PR-latch, the current microprogram address, and several selection bits from the address control function to generate the next microprogram address.

Flag Control Functions

The flag control functions of the MCU are selected by the 4 input lines designated FC$_0$-FC$_3$. Function code formats are given in "Flag Control Function summary."

The following is a detailed description of each of the 8 flag control functions.

Flag Input Control Functions

The flag input control functions select which flag or flags will be set to the current value of the flag input ($\overline{FI}$) line.

Data on $\overline{FI}$ is stored in the F-latch when the clock is low. The content of the F-latch is loaded into the C and/or Z flag on the rising edge of the clock.

Flag Output Control Functions

The flag output control functions select the value to which the flag output ($\overline{FO}$) line will be forced.

FLAG CONTROL FUNCTION TABLE

MNEMONIC	FUNCTION DESCRIPTION
SCZ	Set C-flag and Z-flag to FI. The C-flag and the Z-flag are both set to the value of FI.
STZ	Set Z-flag to FI. The Z-flag is set to the value of FI. The C-flag is unaffected.
STC	Set C-flag to FI. The C-flag is set to the value of FI. The Z-flag is unaffected.
HCZ	Hold C-flag and Z-flag. The values in the C-flag and Z-flag are unaffected.

FLAG OUTPUT CONTROL FUNCTION TABLE

MNEMONIC	FUNCTION DESCRIPTION
FF0	Force FO to 0. FO is forced to the value of logical 0.
FFC	Force FO to C. FO is forced to the value of the C-flag.
FFZ	Force FO to Z. FO is forced to the value of the Z-flag.
FF1	Force FO to 1. FO is forced to the value of logical 1.

FLAG CONTROL FUNCTION SUMMARY

TYPE	MNEMONIC	DESCRIPTION	FC$_1$	0
Flag Input	SCZ	Set C-flag and Z-flag to f	0	0
	STZ	Set Z-flag to f	0	1
	STC	Set C-flag to f	1	0
	HCZ	Hold C-flag and Z-flag	1	1

TYPE	MNEMONIC	DESCRIPTION	FC$_3$	2
Flag Output	FF0	Force FO to 0	0	0
	FFC	Force FO to C-flag	1	0
	FFZ	Force FO to Z-flag	0	1
	FF1	Force FO to 1	1	1

LOAD FUNCTION	NEXT ROW					NEXT COL			
LD	MA$_8$	7	6	5	4	MA$_3$	2	1	0
0	See Address Control Function Summary								
1	0	X$_3$	X$_2$	X$_1$	X$_0$	X$_7$	X$_6$	X$_5$	X$_4$

NOTE
f = Contents of the F-latch xn = Data on PX- or SX-bus line n (active low)

ADDRESS CONTROL FUNCTION SUMMARY

MNEMONIC	DESCRIPTION	FUNCTION							NEXT ROW					NEXT COL			
		AC$_6$	5	4	3	2	1	0	MA$_8$	7	6	5	4	MA$_3$	2	1	0
JCC	Jump in current column	0	0	d_4	d_3	d_2	d_1	d_0	d_4	d_3	d_2	d_1	d_0	m_3	m_2	m_1	m_0
JZR	Jump to zero row	0	1	0	d_3	d_2	d_1	d_0	0	0	0	0	0	d_3	d_2	d_1	d_0
JCR	Jump in current row	0	1	1	d_3	d_2	d_1	d_0	m_8	m_7	m_6	m_5	m_4	d_3	d_2	d_1	d_0
JCE	Jump in column/enable	1	1	1	0	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	m_3	m_2	m_1	m_0
JFL	Jump/test F-latch	1	0	0	d_3	d_2	d_1	d_0	m_8	d_3	d_2	d_1	d_0	m_3	0	1	f
JCF	Jump/test Z-flag	1	0	1	1	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	m_3	0	1	c
JPR	Jump/test PR-latch	1	1	0	0	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	m_3	0	1	z
JLL	Jump/test left PR bits	1	1	0	1	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	p_3	p_2	p_1	p_0
JRL	Jump/test right PR bits	1	1	1	1	1	d_1	d_0	m_8	m_7	1	d_1	d_0	0	1	p_3	p_2
JPX	Jump/test PX-bus	1	1	1	1	0	d_1	d_0	m_8	m_7	m_6	d_1	d_0	x_7	x_6	x_5	x_4

NOTE

dn = Data on address control line n
mn = Data in microprogram address register bit n

Pn = Data in PR-latch bit n
xn = Data on PX-bus line n (active low)
f,c,z = Contents of F-latch, C-flag, or Z-flag, respectively

STROBE FUNCTIONS

The load function of the MCU is controlled by the input line designated LD. If the LD line is active high at the rising edge of the clock, the data on the primary and secondary instruction buses, PX_4-PX_7 and SX_0-SX_3, *is loaded into the microprogram address* register. PX_4-PX_7 are loaded into MA_0-MA_7 and SX_0-SX_3 are loaded into MA_4-MA_7. The high-order bit of the microprogram address register MA_8 is set to a logical 0. The bits from the primary instruction bus select 1 of 16 possible column addresses. Likewise, the bits from the secondary instruction bus select 1 of the first 16 row addresses.

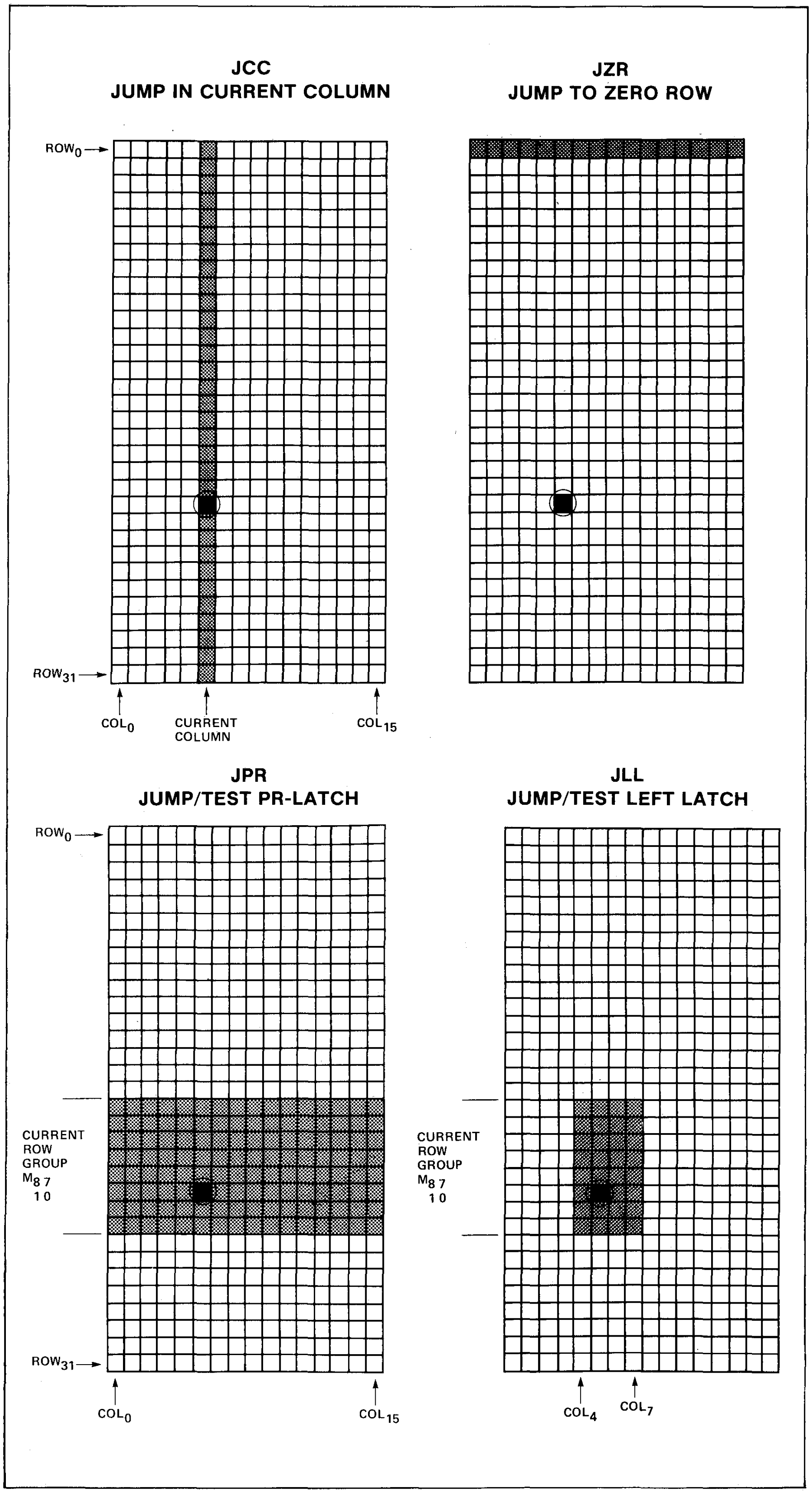
The MCU generates an interrupt strobe enable on the output line designated ISE. The line is placed in the active high state whenever a JZR to col_{15} is selected as the address control function. Generally, the start of a macroinstruction fetch sequence is situated at row_0 and col_{15} so the interrupt control may be enabled at the beginning of the fetch/execute cycle. The interrupt control responds to the interrupt by pulling the enable row address (ERA) input line low to override the selected next row address from the MCU. Then by gating an alternative next row address on to the row address lines of the microprogram memory, the microprogram may be forced to enter an interrupt handling routine. The alternative row address placed on the microprogram memory address lines does not alter the contents of the microprogram address register. Therefore, subsequent jump functions will utilize the row address in the register, and not the alternative row address, to determine the next microprogram address.

Note, the load function always overrides the address control function on AC_0-AC_6. It does not, however, override the latch enable or load sub-functions of the JCE or JPX instruction, respectively. In addition, it does not inhibit the interrupt strobe enable or any of the flag control functions.

JUMP SET DIAGRAMS

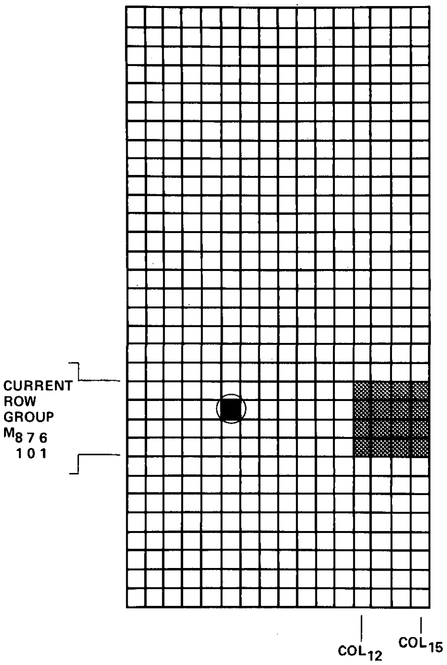
The following 10 diagrams illustrate the jump set for each of the 11 jump and jump/test functions of the MCU. Location 341 indicated by the circled square, represents 1 current row (row_{21}) and current column (col_5) address. The dark boxes indicate the microprogram locations that may be selected by the particular function as the next address.

JUMP SET DIAGRAMS

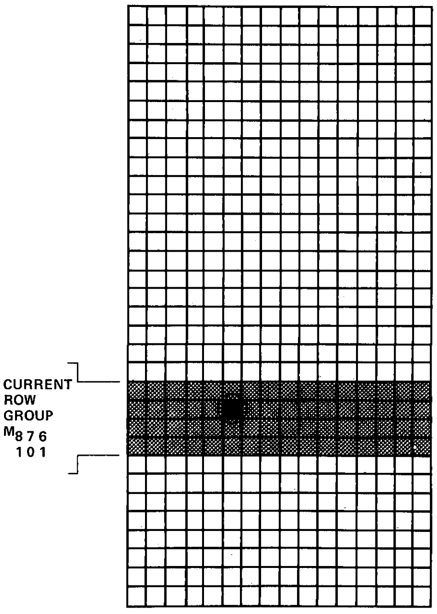


JUMP SET DIAGRAMS (Cont'd)

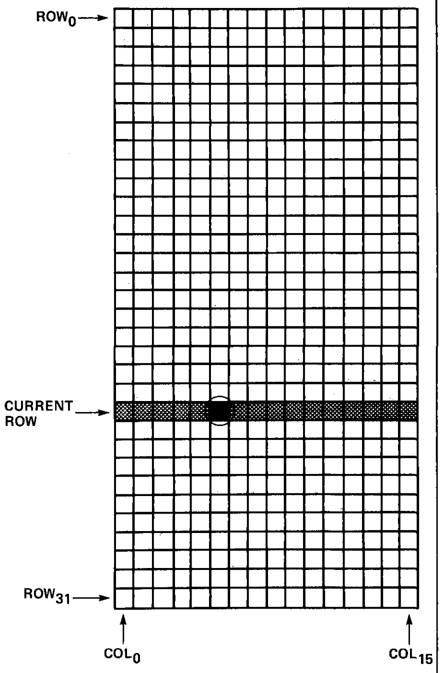
JRL
JUMP/TEST RIGHT LATCH



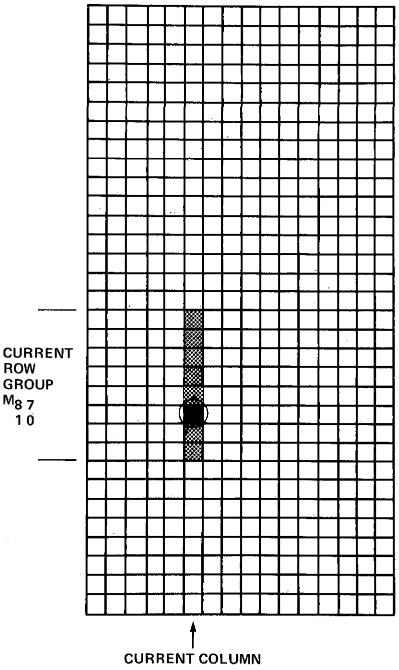
JPX
JUMP/TEST PX-BUS



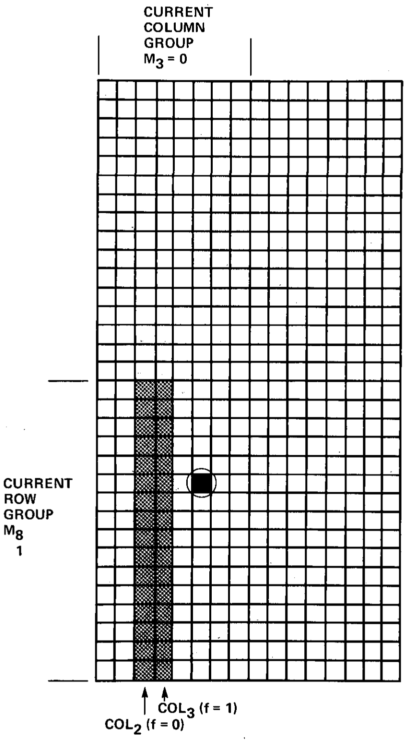
JCR
JUMP IN CURRENT ROW



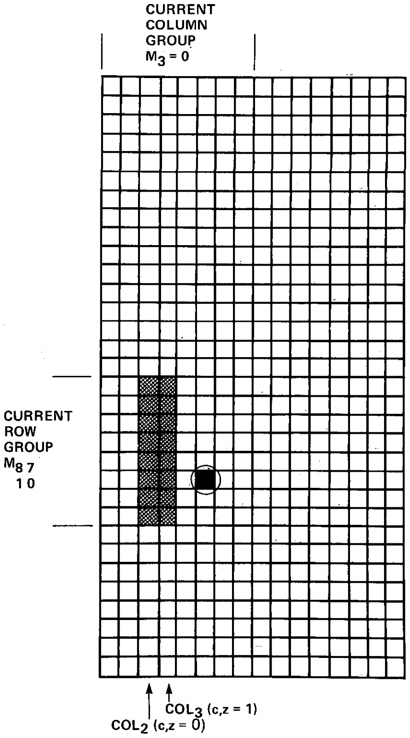
JCE
JUMP COLUMN/ENABLE



JFL
JUMP/TEST F-LATCH



JCF, JZF
JUMP/TEST C-FLAG
JUMP/TEST Z-FLAG



N3001 T_A = 0°C to +70°C, V_{CC} = 5.0V, ± 5% N3001-I

AC ELECTRICAL CHARACTERISTICS S3001 T_A = -55°C to +125°C, V_{CC} = 5.0V ± 10%

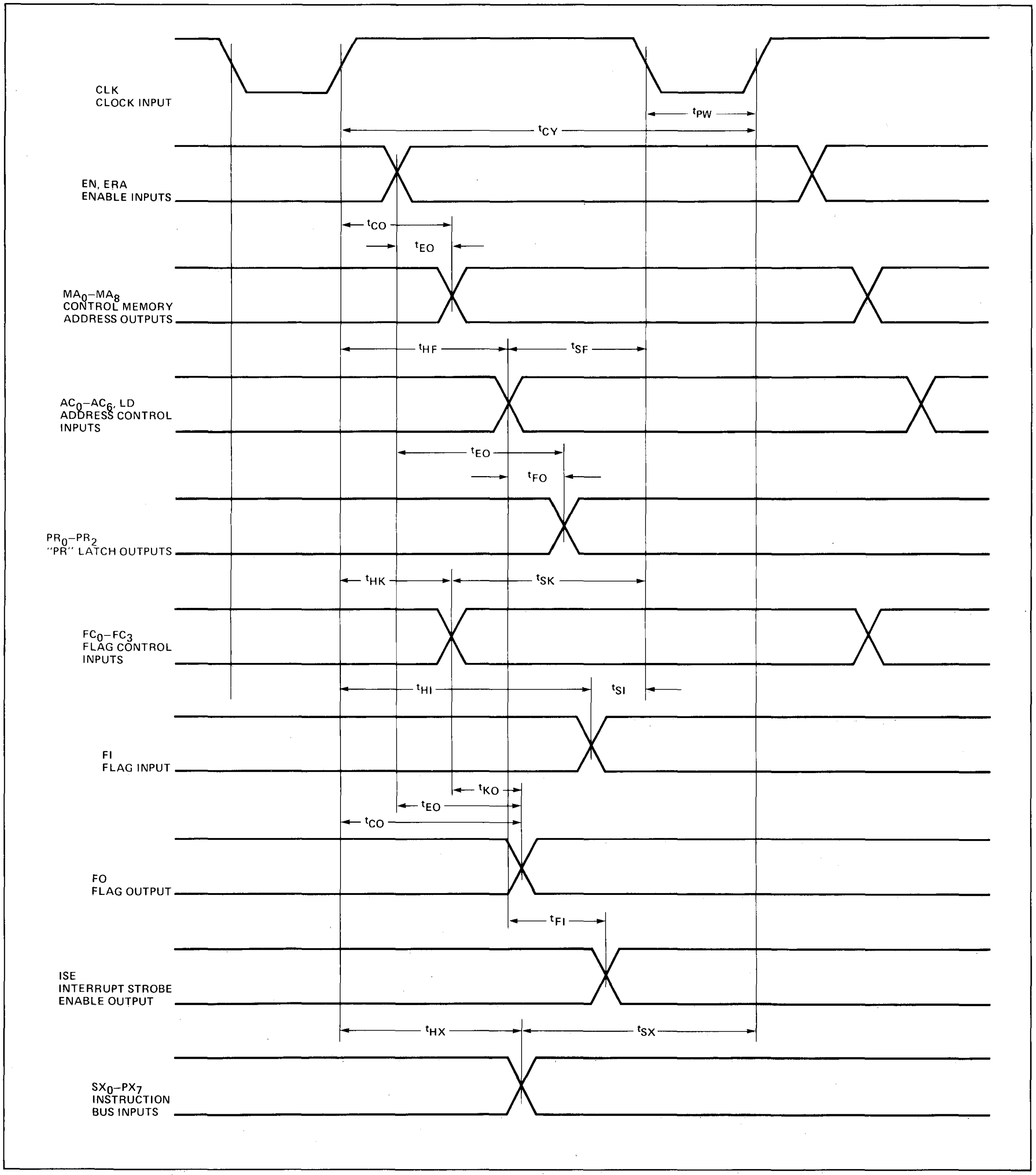
PARAMETER		N3001			S3001			UNIT
		Min	Typ[1]	Max	Min	Typ[1]	Max	
t_{CY}	Cycle Time[2]	60	45		95	45		ns
t_{PW}	Clock Pulse Width	17	10		40	10		ns
	Control and Data Input Set-Up Times:							
t_{SF}	LD, AC_0-AC_6 (Set to "1"/"0")	20	3/14		20	3/14		ns
t_{SK}	FC_0, FC_1	7	5		10	5		ns
t_{SX}	PX_4-PX_7 (Set to "1"/"0")	28	4/13		35	4/13		ns
t_{SI}	FI (Set to "1"/"0")	12	-6/0		15	-6/10		ns
t_{SX}	SX_0-SX_3	15	5		35	5		ns
	Control and Data Input Hold Times:							
t_{HF}	LD, AC_0-AC_6 (Hold to "1"/"0")	4	-3/-14		5	-3/-14		ns
t_{HK}	FC_0, FC_1	4	-5		10	-5		ns
t_{HX}	PX_4-PX_7 (Hold to "1"/"0")	0	-4/-13		25	-4/-13		ns
t_{HI}	FI (Hold to "1"/"0")	16	6.5/0		22	6.5/0		ns
t_{HX}	SX_0-SX_3	0	-5		25	-5		ns
t_{CO}	Propagation Delay from Clock Input (CLK) to Outputs (mA_0-mA_8, FO) (tPHL/tPLH)		17/24	36	10	17/24	45	ns
t_{KO}	Propagation Delay from Control Inputs FC_2 and FC_3 to Flag Out (FO)		13	24		13	50	ns
t_{FO}	Propagation Delay from Control Inputs AC_0-AC_6 to Latch Outputs (PR_0-PR_2)		21	32		21	50	ns
t_{EO}	Propagation Delay from Enable Inputs EN and ERA to Outputs (mA_0-mA_8, FO, PR_0-PR_2)		17	26		17	35	ns
t_{FI}	Propagation Delay from Control Inputs AC_0-AC_6 to Interrupt Strobe Enable Output (ISE)		20	32		20	40	ns

NOTE

[1] Typical values are for TA = 25°C and 5.0 supply voltage.

[2] S3001: tCY = tWP + tSF + tCO

VOLTAGE WAVEFORMS



CENTRAL PROCESSING ELEMENT S/N3002

S3002-I • N3002-XL,I

DESCRIPTION

The N3002 Central Processing Element (CPE) is one part of a bipolar microcomputer set. The N3002 is organized as a 2-bit slice and performs the logical and arithmetic functions required by microinstructions. A system with any number of bits in a data word can be implemented by using multiple N3002s, the N3001 microcomputer control unit, the N74S182 carry look-ahead unit and ROM or PROM memory.

FEATURES

- **45ns cycle time (typ)**
- **Easy expansion to multiple of 2 bits**
- **11 general purpose registers**
- **Full function accumulator**
- **Useful functions include:**
  - **2's complement arithmetic**
  - **Logical AND, OR, NOT, exclusive-NOR**
  - **Increment, decrement**
  - **Shift left/shift right**
  - **Bit testing and zero detection**
  - **Carry look-ahead generation**
  - **Masking via K-bus**
  - **Conditioned clocking allowing non-destructive testing of data in accumulator and scratchpad**
- **3 input buses**
- **2 output buses**
- **Control bus**

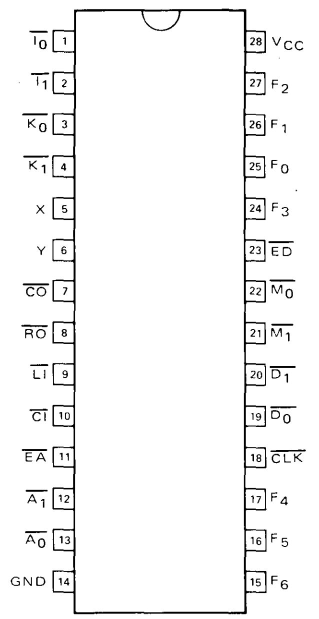
FUNCTION TRUTH TABLE

FUNCTION GROUP	F_6	F_5	F_4
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

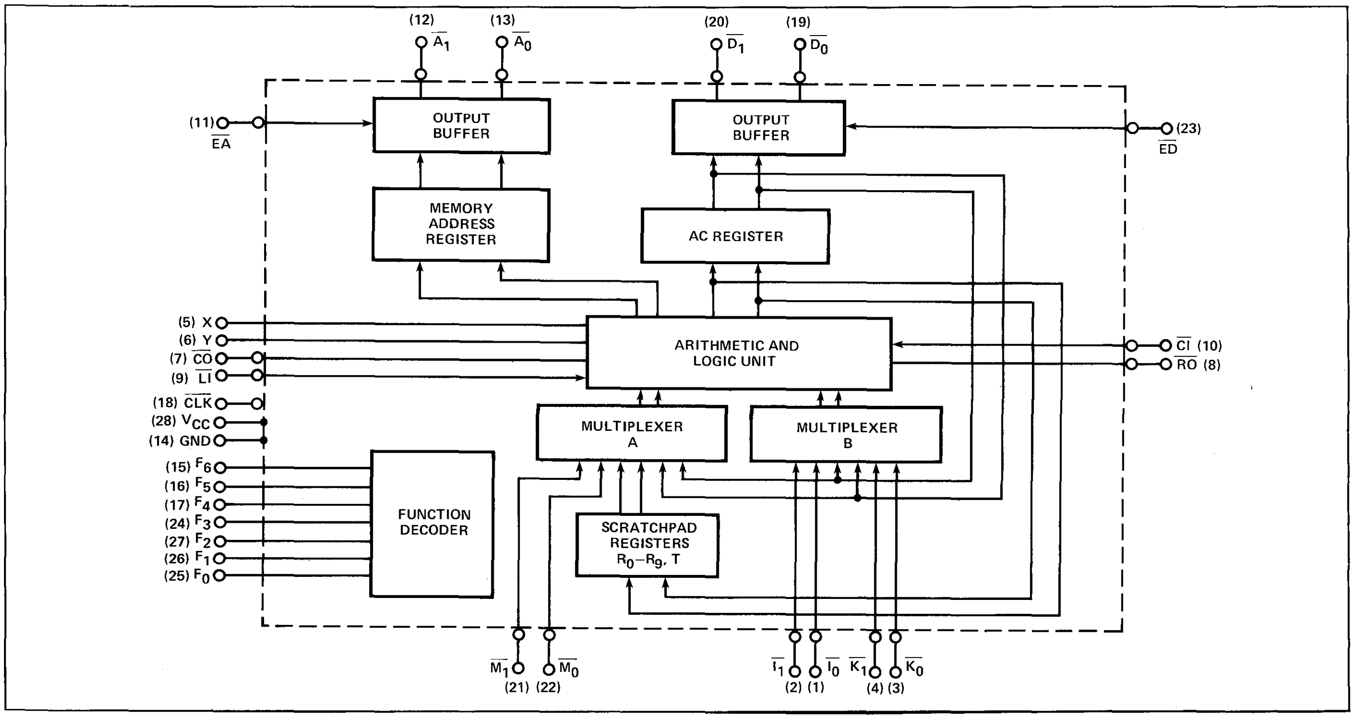
REGISTER GROUP	REGISTER	F_3	F_2	F_1	F_0
I	R_0	0	0	0	0
	R_1	0	0	0	1
	R_2	0	0	1	0
	R_3	0	0	1	1
	R_4	0	1	0	0
	R_5	0	1	0	1
	R_6	0	1	1	0
	R_7	0	1	1	1
	R_8	1	0	0	0
	R_9	1	0	0	1
	T	1	1	0	0
	AC	1	1	0	1
II	T	1	0	1	0
	AC	1	0	1	1
III	T	1	1	1	0
	AC	1	1	1	1

PIN CONFIGURATION

XL,I PACKAGE



BLOCK DIAGRAM



PIN DESCRIPTION

PIN	SYMBOL	NAME AND FUNCTION	TYPE
1, 2	$\overline{I_0}$-$\overline{I_1}$	External Bus Inputs The external bus inputs provide a separate input port for external input devices.	Active low
3, 4	$\overline{K_0}$-$\overline{K_1}$	Mask Bus Inputs The mask bus inputs provide a separate input port from the microprogram memory, to allow mask or constant entry.	Active low
5, 6	X, Y	Standard Carry Look-Ahead Cascade Outputs The cascade outputs allow high speed arithmetic operations to be performed when they are used in conjunction with the 74S182 Look-Ahead Carry Generator	Active high
7	$\overline{CO}$	Ripple Carry Out The ripple carry output is only disabled during shift right operations.	Active low Three-state
8	$\overline{RO}$	Shift Right Output The shift right output is only enabled during shift right operations.	Active low Three-state
9	$\overline{LI}$	Shift Right Input	Active low
10	$\overline{CI}$	Carry Input	Active low
11	$\overline{EA}$	Memory Address Enable Input When in the low state, the memory address enable input enables the memory address outputs (A_0-A_1).	Active low
12-13	$\overline{A_0}$-$\overline{A_1}$	Memory Address Bus Outputs The memory address bus outputs are the buffered outputs of the memory address register (MAR).	Active low Three-state
14	GND	Ground	
14-17, 24-27	F_0-F_6	Micro-Function Bus Inputs The micro-function bus inputs control ALU function and register selection.	Active high
18	$\overline{CLK}$	Clock Input	
19-20	$\overline{D_0}$-$\overline{D_1}$	Memory Data Bus Outputs The memory data bus outputs are the buffered outputs of the full function accumulator register (AC).	Active low Three-state
21-22	$\overline{M_0}$-$\overline{M_1}$	Memory Data Bus Inputs The memory data bus inputs provide a separate input port for memory data.	Active low
23	$\overline{ED}$	Memory Data Enable Input When in the low state, the memory data enable input enables the memory data outputs (D_0-D_1).	Active low
28	V_{CC}	+5 Volt Supply	

SYSTEM DESCRIPTION

Microfunction Decoder and K-Bus

Basic microfunctions are controlled by a 7-bit bus (F_0-F_6) which is organized into 2 groups. The higher 3 bits (F_4-F_6) are designated as F-Group and the lower 4 bits (F_0-F_3) are designated as the R-Group. The F-Group specifies the type of operation to be performed and the R-Group specifies the registers involved.

The F-Bus instructs the microfunction decoder to:
- Select ALU functions to be performed
- Generate scratchpad register address
- Control A and B multiplexer

The resulting microfunction action can be:
- Data transfer
- Shift operations
- Increment and decrement
- Initialize stack
- Test for zero conditions
- 2's complement addition and subtraction
- Bit masking
- Maintain program counter

A and B Multiplexers

A and B multiplexers select the proper 2 operands to the ALU.

A multiplexer selects inputs from one of the following:
- M-bus (data from main memory)
- Scratchpad registers
- Accumulator

B multiplexer selects inputs from one of the following:
- I-bus (data from external I/O devices)
- Accumulator
- K-bus (literal or masking information from micro-program memory)

Scratchpad Registers

- Contains 11 registers (R_0-R_9, T)
- Scratchpad register outputs are multiplexed to the ALU via the A multiplexer
- Used to store intermediate results from arithmetic/logic operations
- Can be used as program counter

Arithmetic/Logic Unit (ALU)

The ALU performs the arithmetic and logic operations of the CPE.

Arithmetic operations are:
- 2's complement addition
- Incrementing
- Decrementing
- Shift left
- Shift right

Logical operations are:
- Transfer
- AND
- Inclusive-OR
- Exclusive-NOR
- Logic complement

ALU operation results are then stored in the accumulator and/or scratchpad registers. For easy expansion to larger arrays, carry look-ahead outputs (X and Y) and cascading shift inputs (LI, RO) are provided.

Accumulator

- Stores results from ALU operations
- The output of accumulator is multiplexed into ALU via the A and B multiplexer as one of the operands

Input Buses

M-bus: Data bus from main memory

- Accepts 2 bits of data from main memory into CPE
- Is multiplexed into the ALU via the A multiplexer

I-bus: Data bus from input/output devices

- Accepts 2 bits of data from external input/output devices into CPE
- Is multiplexed into the ALU via the B multiplexer

K-bus: A special feature of the N3002 CPE

- During arithmetic operations, the K-bus can be used to **mask** portions of the field being operated on
- Select or remove accumulator from operation by placing K-bus in all "1" or all "0" state respectively
- During non-arithmetic operation, the carry circuit can be used in conjunction with the K-bus for word-wise-OR operation for bit testing
- Supply literal or constant data to CPE

Output Buses

A-bus and Memory Address Register

- Main memory address is stored in the memory address register (MAR)
- Main memory is addressed via the A-bus
- MAR and A-bus may also be used to generate device address when executing I/OO instruc-instructions
- A-bus has Tri-State outputs

D-bus: Data bus from CPE to main memory or to I/O devices

- Sends buffered accumulator outputs to main memory or the external I/O devices
- D-bus has Tri-State outputs

FUNCTION DESCRIPTION

F GROUP	R GROUP	K BUS	NAME	EQUATION	DESCRIPTION
0	I	XX	—	$R_n + (AC \wedge K) + CI \rightarrow R_n, AC$	**Logically AND AC with the K-bus.** Add the result to R_n and carry input (CI). Deposit the sum in AC and R_n.
		OO	ILR	$R_n + CI \rightarrow R\ , AC$	Conditionally increment R_n and load the result in AC. Used to load AC from R_n or to increment R_n and load a copy of the result in AC.
		11	ALR	$AC + R_n + CI \rightarrow R_n, AC$	Add AC and CI to R_n and load the result in AC. Used to add AC to a register. If R_n is AC, then AC is shifted left one bit position.
0	II	XX	—	$M + (AC \wedge K) + CI \rightarrow AT$	**Logically AND AC with the K-bus.** Add the result to CI and the M-bus. Deposit the sum in AC or T.
		OO	ACM	$M + CI \rightarrow AT$	Add CI to M-bus. Load the result in AC or T, as specified. Used to load memory data in the specified register, or to load incremented memory data in the specified register.
		11	AMA	$M + AC + CI \rightarrow AT$	Add the M-bus to AC and CI, and load the result in AC or T, as specified. Used to add memory data or incremented memory data to AC and store the sum in the specified register.
0	III	XX	—	$AT_L \wedge (\overline{I_L \wedge K_L}) \rightarrow RO$ $LI \vee [(I_H \wedge K_H) \wedge AT_H] \rightarrow AT_H$ $[AT_L \wedge (I_L \wedge K_L)]$ $[AT_H \vee (I_H \wedge K_H)] \rightarrow AT_L$	None
		OO	SRA	$AT_L \rightarrow RO$ $AT_H \rightarrow AT_L$ $L_I \rightarrow AT_H$	Shift AC or T, as specified, right one bit position. Place the previous low order bit value on RO and fill the high order bit from the data on LI. Used to shift or rotate AC or T right one bit.
1	I	XX	—	$K \vee R_n \rightarrow MAR$ $R_n + K + CI \rightarrow R_n$	**Logically OR R_n with the K-bus.** Deposit the result in MAR. Add the K-bus to Rn and CI. Deposit the result in R_n.
		OO	LMI	$Rn \rightarrow MAR, \quad Rn + CI \rightarrow Rn$	Load MAR from R_n. Conditionally increment R_n. Used to maintain a macro-instruction program counter.
		11	DSM	$11 \rightarrow MAR, \quad Rn - 1 + CI \rightarrow Rn$	Set MAR to all ones. Conditionally decrement R_n by one. Used to force MAR to its highest address and to decrement Rn.
1	II	XX	—	$KVM \rightarrow MAR$ $M + K + CI \rightarrow AT$	**Logically OR the M-bus with the K-Bus.** Deposit the result in MAR. Add the K-bus to the M-bus and CI. Deposit the sum in AC or T.
		OO	LMM	$M \rightarrow MAR, \quad M + CI \rightarrow AT$	Load MAR from the M-bus. Add CI to the M-bus. Deposit the result in AC or T. Used to load the address register with memory data for macro-instructions using indirect addressing.
		11	LDM	$11 \rightarrow MAR$ $M - 1 + CI \rightarrow AT$	Set MAR to all ones. Subtract one from the M-bus. Add CI to the difference and deposit the result in AC or T, as specified. Used to load decremented memory data in AC or T.

FUNCTION DESCRIPTION (Cont'd)

F GROUP	R GROUP	K BUS	NAME	EQUATION	DESCRIPTION
1	III	XX	—	$(\overline{AT} \vee K) + (AT \wedge K) + CI \rightarrow AT$	**Logically OR the K-bus with the complement of AC or T, as specified.** Add the result to the logical AND of specified register with the K-bus. Add the sum to CI. Deposit the result in the specified register.
		OO	CIA	$\overline{AT} + CI \rightarrow AT$	Add CI to the complement of AC or T, as specified. Deposit the result in the specified register. Used to form the 1's or 2's complement of AC or T.
		11	DCA	$\overline{AT} - 1 + CI \rightarrow AT$	Subtract one from AC or T, as specified. Add CI to the difference and deposit the sum in the specified register. Used to decrement AC or T.
2	I	XX	—	$(AC \wedge K) - 1 + CI \rightarrow R_n$	**Logically AND the K-bus with AC.** Subtract one from the result and add the difference to CI. Deposit the sum in R_n.
		OO	CSR	$CI - 1 \rightarrow R_n$ (See Note 1)	Subtract one from CI and deposit the difference in Rn. Used to conditionally clear or set R_n to all 0's or 1's, respectively.
		11	SDR	$AC - 1 + CI \rightarrow R_n$ (See Note 1)	Subtract one from AC and add the difference to CI. Deposit the sum in R_n. Used to store AC in R_n or to store the decremented value of AC in R_n.
2	II	XX	—	$(AC \wedge K) - 1 + CI \rightarrow AT$ (See Note 1)	**Logically AND the K-bus with AC.** Subtract one from the result and add the difference to CI. Deposit the sum in AC or T, as specified.
		OO	CSA	$CI - 1 \rightarrow AT$ (See Note 1)	Subtract one from CI and deposit the difference in AC or T. Used to conditionally clear or set AC or T.
		11	SDA	$AC - 1 + CI \rightarrow AT$ (See Note 1)	Subtract one from AC and add the difference to CI. Deposit the sum in AC or T. Used to store AC in T, or decrement AC, or store the decremented value of AC in T.
2	III	XX	—	$(I \wedge K) - 1 + CI \rightarrow AT$ (See Note 1)	**Logically AND the data of the K-bus with the data on the I-bus.** Subtract one from the result and add the difference to CI. Deposit the sum in AC or T, as specified.
		OO	CSA	$CI - 1 \rightarrow AT$	Subtract one from CI and deposit the difference in AC or T. Used to conditionally clear or set AC or T.
		11	LDI	$I - 1 + CI \rightarrow AT$	Subtract one from the data on the I-bus and add the difference to CI. Deposit the sum in AC or T, as specified. Used to load input bus data or decremented input bus data in the specified register.
3	I	XX	—	$R_n + (AC \wedge K) + CI \rightarrow R_n$	**Logically AND AC with the K-bus.** Add R_n and CI to the result. Deposit the sum in R_n.
		OO	INR	$R_n + CI \rightarrow R_n$	Add CI to R_n and deposit the sum in R_n. Used to increment R_n.
		11	ADR	$AC + R_n + CI \rightarrow R_n$	Add AC to R_n. Add the result to CI and deposit the sum in R_n. Used to add the accumulator to a register or to add the incremented value of the accumulator to a register.
3	II	XX	—	$M + (AC \wedge K) + CI \rightarrow AT$	**Logically AND AC with the K-bus.** Add the result to CI and the M-bus. Deposit the sum in AC or T.
		OO	ACM	$M + CI \rightarrow AT$	Add CI to M-bus. Load the result in AC or T, as specified. Used to load memory data in the specified register, or to load incremented memory data in the specified register.
		11	AMA	$M + AC + CI \rightarrow AT$	Add the M-bus to AC and CI, and load the result in AC or T, as specified. Used to add memory data or incremented memory data to AC and store the sum in the specified register.

FUNCTION DESCRIPTION (Cont'd)

F GROUP	R GROUP	K BUS	NAME	EQUATION	DESCRIPTION
3	III	XX	—	$AT + (I \wedge K) + CI \rightarrow AT$	**Logically AND the K-bus with the I-bus.** Add CI and the contents of AC or T, as specified, to the result. Deposit the sum in the specified register.
		OO	INA	$AT + CI \rightarrow AT$	Conditionally increment AC or T. Used to increment AC or T.
		11	AIA	$I + AT + CI \rightarrow AT$	Add the I-bus to AC or T. Add CI to the result and deposit the sum in the specified register. Used to add input data or incremented input data to the specified register.
4	I	XX	—	$CI \vee (R_n \wedge AC \wedge K) \rightarrow CO$ $R_n \wedge (AC \wedge K) \rightarrow R_n$	**Logically AND the K-bus with AC.** Logically AND the result with the contents of R_n. Deposit the final result in R_n. Logically OR the value of CI with the word-wise OR of the bits of the final result. Place the value of the carry OR on the carry output (CO) line.
		OO	CLR	$CI \rightarrow CO,\ O \rightarrow R_n$	Clear R_n to all O's. Force CO to CI. Used to clear a register and force CO to CI.
		11	ANR	$CI \vee (R_n \wedge AC) \rightarrow CO$ $R_n \wedge AC \rightarrow R_n$	Logically AND AC with R_n. Deposit the result in R_n. Force CO to one if the result is non-zero. Used to AND the accumulator with a register and test for a zero result.
4	II	XX	—	$CI \vee (M \wedge AC \wedge K) \rightarrow CO$ $M \wedge (AC \wedge K) \rightarrow AT$	**Logically AND the K-bus with AC.** Logically AND the result with the M-bus. Deposit the final result in AC or T. Logically OR the value of CI with the word-wise OR of the bits of the final result. Place the value of the carry OR on CO.
		OO	CLA	$CI \rightarrow CO,\ O \rightarrow AT$	Clear AC or T, as specified, to all O's. Force CO to CI. Used to clear the specified register and force CO to CI.
		11	ANM	$CI \vee (M \wedge AC) \rightarrow CO$ $M \wedge AC \rightarrow AT$	Logically AND the M-bus with AC. Deposit the result in AC or T. Force CO to one if the result is non-zero. Used to AND M-bus data to the accumulator and test for a zero result.
4	III	XX	—	$CI \vee (AT \wedge 1 \wedge K) \rightarrow CO$ $AT \wedge (I \wedge K) \rightarrow AT$	**Logically AND the I-bus with the K-bus.** Logically AND the result with AC or T. Deposit the final result in the specified register. Logically OR CI with the word-wise OR of the final result. Place the value of the carry OR on CO.
		OO	CLA	$CI \rightarrow CO,\ O \rightarrow AT$	Clear AC or T, as specified, to all O's. Force CO to CI. Used to clear the specified register and force CO to CI.
		11	ANI	$CI \vee (AT \wedge I) \rightarrow CO$ $AT \wedge 1 \rightarrow AT$	Logically AND the I-bus with AC or T, as specified. Deposit the result in the specified register. Force CO to one if the result is non-zero. Used to AND the I-bus to the accumulator and test for a zero result.
5	I	XX	—	$CI \vee (R_n \wedge K) \rightarrow CO$ $K \wedge R_n \rightarrow R_n$	**Logically AND the K-bus with R_n.** Deposit the result in Rn. Logically OR CI with the word-wise OR of the result. Place the value of the carry OR on CO.
		OO	CLR	$CI \rightarrow CO,\ O \rightarrow R_n$	Clear Rn to all O's. Force CO to CI. Used to clear a register and force CO to CI.
		11	TZR	$CI \vee R_n \rightarrow CO$ $R_n \rightarrow R_n$	Force CO to one if Rn is non-zero. Used to test a register for zero. Also used to AND K-bus data with a register for masking and, optionally, testing for a zero result.
5	II	XX	—	$CI \vee (M \wedge K) \rightarrow CO$ $K \wedge M \rightarrow AT$	**Logically AND the K-bus with the M-bus.** Deposit the result in AC or T, as specified. Logically OR CI with the word-wise OR of the result. Place the value of the carry OR on CO.
		OO	CLA	$CI \rightarrow CO,\ O \rightarrow AT$	Clear AC or T, as specified, to all O's. Force CO to CI. Used to clear the specified register and force CO to CI.
		11	LTM	$CI \vee M \rightarrow CO$ $M \rightarrow AT$	Load AC or T, as specified, from the M-bus. Force CO to one if the result is non-zero. Used to load the specified register from memory and test for a zero result. Also used to AND the K-bus with the M-bus for masking and, optionally, testing for a zero result.

FUNCTION DESCRIPTION (Cont'd)

F GROUP	R GROUP	K BUS	NAME	EQUATION	DESCRIPTION
5	III	XX	—	$CI \vee (AT \wedge K) \rightarrow CO$ $K \wedge AT \rightarrow AT$	Logically AND the K-bus with AC or T, as specified. Deposit the result in the specified register. Logically OR CI with the word-wise OR of the result. Place the value of the carry OR on CO.
		OO	CLA	$CI \rightarrow CO,\ O \rightarrow AT$	Clear AC or T, as specified, to all O's. Force CO to CI. Used to clear the specified register and force CO to CI.
		11	TZA	$CI \vee AT \rightarrow CO$ $AT \rightarrow AT$	Force CO to one if AC or T, as specified, is non-zero. Used to test the specified register for zero. Also used to AND the K-bus to the specified register for masking and, optionally, testing for a zero result.
6	I	XX	—	$CI \vee (AC \wedge K) \rightarrow CO$ $R_n \vee (AC \wedge K) \rightarrow R_n$	Logically OR CI with the word-wise OR of the logical AND of AC and the K-bus. Place the result of the carry OR on CO. Logically OR R_n with the logical AND of AC and the K-bus. Deposit the result in R_n.
		OO	NOP	$CI \rightarrow CO,\ R_n \rightarrow R_n$	Force CO to CI. Used as a null operation or to force CO to CI.
		11	ORR	$CI \vee AC \rightarrow CO$ $R_n \vee AC \rightarrow R_n$	Force CO to one if AC is non-zero. Logically OR AC with R_n. Deposit the result in R_n. Used to OR the accumulator to a register and, optionally, test the previous accumulator value for zero.
6	II	XX	—	$CI \vee (AC \wedge K) \rightarrow CO$ $M \vee (AC \wedge K) \rightarrow AT$	Logically OR CI with the word-wise OR of the logical AND of AC and the K-bus. Place the carry OR on CO. Logically OR the M-bus, with the logical AND of AC and the K-bus. Deposit the final result in AC or T.
		OO	LMF	$CI \rightarrow CO,\ M \rightarrow AT$	Load AC or T, as specified, from the M-bus. Force CO to CI. Used to load the specified register with memory data and force CO to CI.
		11	ORM	$CI \vee AC \rightarrow CO$ $M \vee AC \rightarrow AT$	Force CO to one if AC is non-zero. Logically OR the M-bus with AC. Deposit the result in AC or T, as specified. Used to OR M-bus with the AC and, optionally, test the previous value of AC for zero.
6	III	XX	—	$CI \vee (I \wedge K) \rightarrow CO$ $AT \vee (I \wedge I) \rightarrow AT$	Logical OR CI with the word-wise OR of the logical AND of the I-bus and the K-bus. Place the carry OR on CO. Logically AND the K-bus with the I-bus. Logically OR the result with AC or T, as specified. Deposit the final result in the specified register.
		OO	NOP	$CI \rightarrow CO,\ AT \rightarrow AT$	Force CO to CI. Used as a null operation or to force CO to CI.
		11	ORI	$CI \vee I \rightarrow CO$ $I \vee AT \rightarrow AT$	Force CO to one if the data on the I-bus is non-zero. Logically OR the I-bus to AC or T, as specified. Deposit the result in the specified register. Used to OR I-bus data with the specified register and, optionally, test the I-bus data for zero.
7	I	XX	—	$CI \vee (R_n \wedge AC \wedge K) \rightarrow CO$ $R_n \overline{\oplus} (AC \wedge K) \rightarrow R_n$	Logically OR CI with the word-wise OR of the logical AND of R_n and AC and the K-bus. Place the carry OR on CO. Logically AND the K-bus with AC. Exclusive-NOR the result with R_n. Deposit the final result in R_n.
		OO	CMR	$CI \rightarrow CO,\ Rn \rightarrow Rn$	Complement the contents of R_n. Force CO to CI.
		11	XNR	$CI \vee (R_n \wedge AC) \rightarrow CO$ $R_n \overline{\oplus} AC \rightarrow R_n$	Force CO to one if the logical AND of AC and R n is non-zero. Exclusive-NOR AC with R_n. Deposit the result in R_n. Used to exclusive-NOR the accumulator with a register.

FUNCTION DESCRIPTION (Cont'd)

F GROUP	R GROUP	K BUS	NAME	EQUATION	DESCRIPTION
7	II	XX	—	$CI \vee (M \wedge AC \wedge K) \rightarrow CO$ $M \overline{\oplus} (AC \wedge K) \rightarrow AT$	Logically OR CI with the word-wise OR of the logical AND of AC and the K-bus and M-bus. Place the carry OR on CO. Logically AND the K-bus with AC. Exclusive NOR the result with the M-bus. Deposit the final result in AC or T.
		OO	LCM	$CI \rightarrow CO, \ \overline{M} \rightarrow AT$	Load the complement of the M-bus into AC or T, as specified. Force CO to CI.
		11	XNM	$CI \vee (M \wedge AC) \rightarrow CO$ $M \overline{\oplus} AC \rightarrow AT$	Force CO to one if the logical AND of AC and the M-bus is non-zero. Exclusive-NOR AC with the M-bus. Deposit the result in AC or T, as specified. Used to exclusive-NOR memory data with the accumulator.
7	III	XX	—	$CI \vee (AT \wedge I \wedge K) \rightarrow CO$ $AT \overline{\oplus} (I \quad K) \rightarrow AT$	Logically OR CI with the word-wise OR of the logical AND of the specified register and the I-bus and K-bus. Place the carry OR on CO. Logically AND the K-bus with the I-bus. Exclusive-NOR the result with AC or T, as specified. Deposit the final result in the specified register.
		OO	CMA	$CI \rightarrow CO, \ \overline{AT} \rightarrow AT$	Complement AC or T, as specified. Force CO to CI.
		11	XNI	$CI \vee (AT \wedge I) \rightarrow CO$ $I \overline{\oplus} AT \rightarrow AT$	Force CO to one if the logical AND of the specified register and the I-bus is non-zero. Exclusive-NOR AC with the I-bus. Deposit the result in AC or T, as specified. Used to exclusive-NOR input data with the accumulator.

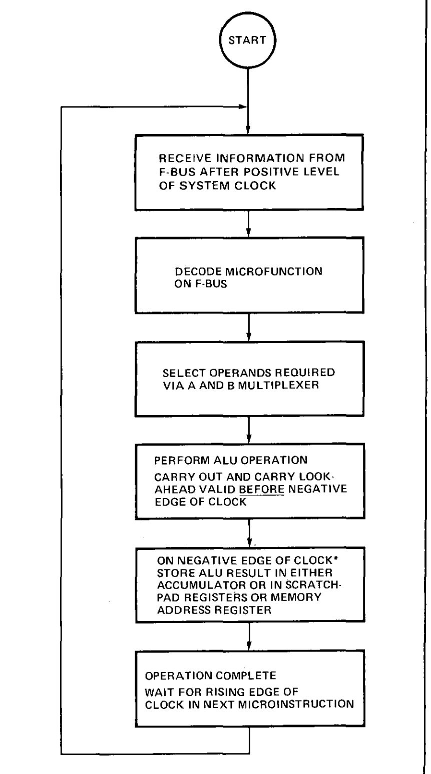
FUNCTION DESCRIPTION KEY

SYMBOL	MEANING
I,K,M	Data on the I, K, and M buses, respectively
CI,LI	Data on the carry input and left input, respectively
CO,RO	Data on the carry output and right output, respectively
Rn	Contents of register n including T and AC (R-Group I)
AC	Contents of the accumulator
AT	Contents of AC or T, as specified
MAR	Contents of the memory address register
L,H	As subscripts, designate low and high order bit, respectively
+	2's complement addition
-	2's complement subtraction
∧	Logical AND
∨	Logical OR
⊕	Exclusive-NOR
→	Deposit into

NOTE

1. 2's complement arithmetic adds 111 . . . 11 to perform subtraction of 000 . . . 01.

MICROCYCLE TIMING SEQUENCE


START
RECEIVE INFORMATION FROM F-BUS AFTER POSITIVE LEVEL OF SYSTEM CLOCK
DECODE MICROFUNCTION ON F-BUS
SELECT OPERANDS REQUIRED VIA A AND B MULTIPLEXER
PERFORM ALU OPERATION CARRY OUT AND CARRY LOOK-AHEAD VALID BEFORE NEGATIVE EDGE OF CLOCK
ON NEGATIVE EDGE OF CLOCK* STORE ALU RESULT IN EITHER ACCUMULATOR OR IN SCRATCH-PAD REGISTERS OR MEMORY ADDRESS REGISTER
OPERATION COMPLETE WAIT FOR RISING EDGE OF CLOCK IN NEXT MICROINSTRUCTION

*SYSTEM CLOCK MAY BE OMITTED THROUGH EXTERNAL GATING IN ORDER TO USE CARRY, SHIFT OR CARRY LOOK-AHEAD CIRCUIT TO PERFORM NON-DESTRUCTIVE TEST ON DATA IN ACCUMULATOR OR IN SCRATCHPAD REGISTERS.

AC ELECTRICAL CHARACTERISTICS N3001 = T_A = 0°C to +70°C, V_{CC} = 5V ± 5%

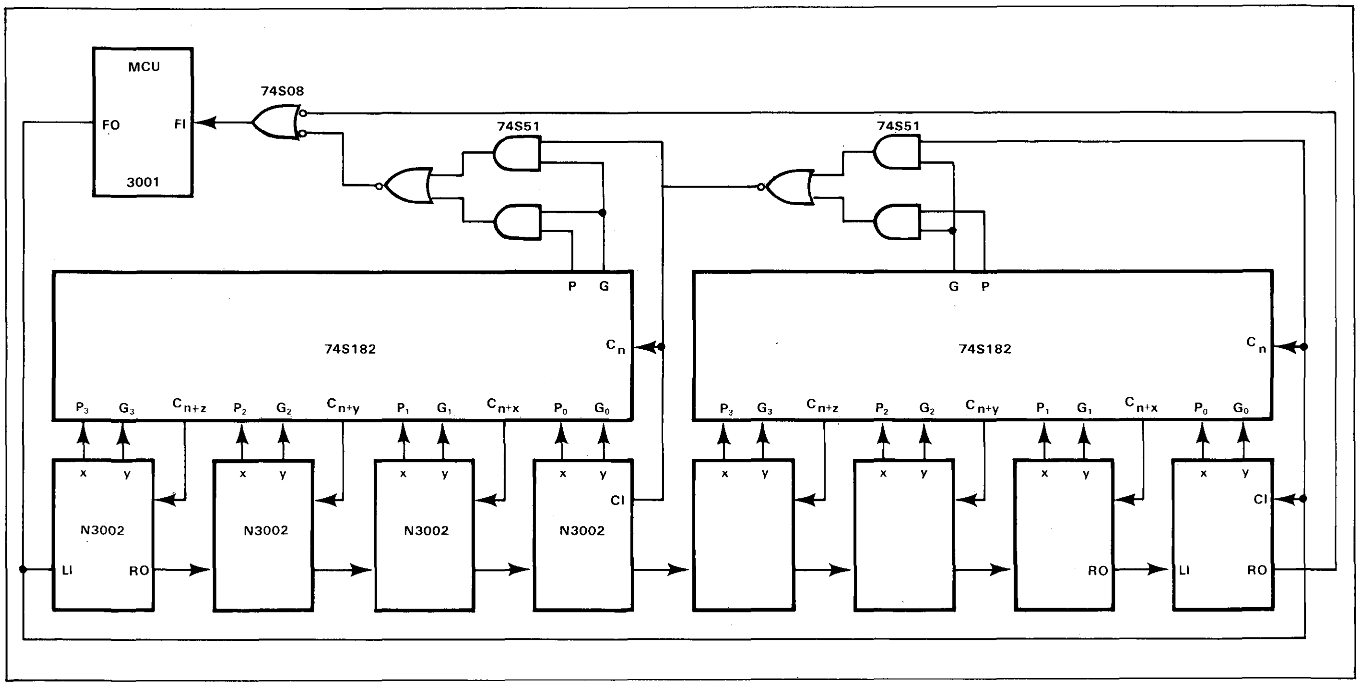
S3001 = T_A = -55°C to +125°C, V_{CC} = 5V ± 10%

PARAMETER	N3002			S3002			UNIT
	Min	Typ[1]	Max	Min	Typ[1]	Max	
tCY Clock Cycle Time	70	45		120	45		ns
tWP Clock Pulse Width	17	10		42	10		ns
tFS Function Input Set-Up Time (F_0 through F_6)	48	-23 → 35		70	-23 → 35		ns
Data Set-Up Time:							
tDS I_0, I_1, M_0, M_1, K_0, K_1	40	12 → 29		60	12 → 29		ns
tSS LI, CI	21	0 → 7		30	0 → 7		ns
Data and Function Hold Time:							
tFH F_0 through F_6	4	0		5	0		ns
tDH 1_0, I_1, M_0, M_1, K_0, K_1	4	-28 → -11		5	-28 → -11		ns
tSH LI, CI	12	-7 → 0		15	-7 → 0		ns
Propagation Delay to X, Y, RO from:							
tXF Any Function Input		28	52		28	65	ns
tXD Any Data Input		16 → 20	33		16 → 20	65	ns
tXT Trailing Edge of CLK		33	48		33	75	ns
tXL Leading Edge of CLK	13	18 → 40	70	13	18 → 40	90	ns
Propagation Delay to CO from:							
tCL Leading Edge of CLK	16	24 → 44	70		24 → 44	90	ns
tCT Trailing Edge of CLK		30 → 40	56		30 → 40	100	ns
tCF Any Function Input		25 → 35	52		25 → 35	75	ns
tCD Any Data Input		17 → 23	55		17 → 23	65	ns
tCC CI (Ripple Carry)		9 → 13	20		9 → 13	30	ns
Propagation Delay to A_0, A_1, D_0, D_1 from:							
tDL Leading Edge of CLK		17 → 25	40		17 → 25	75	ns
tDE Enable Input ED, EA		10 → 12	20		10 → 12	35	ns

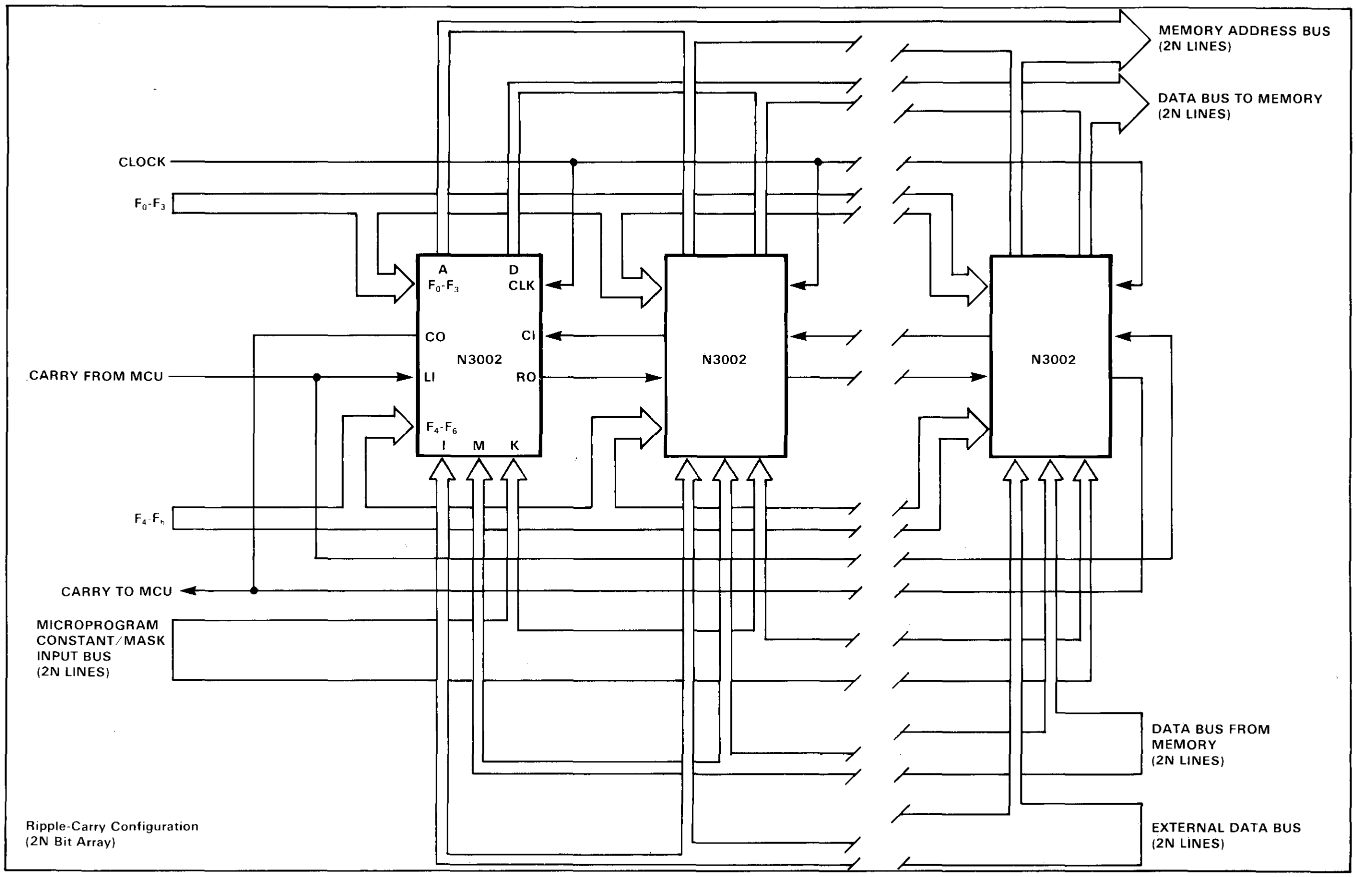
NOTE

1. Typical values are for TA = 25°C and typical supply voltage.

CARRY LOOK-AHEAD CONFIGURATION

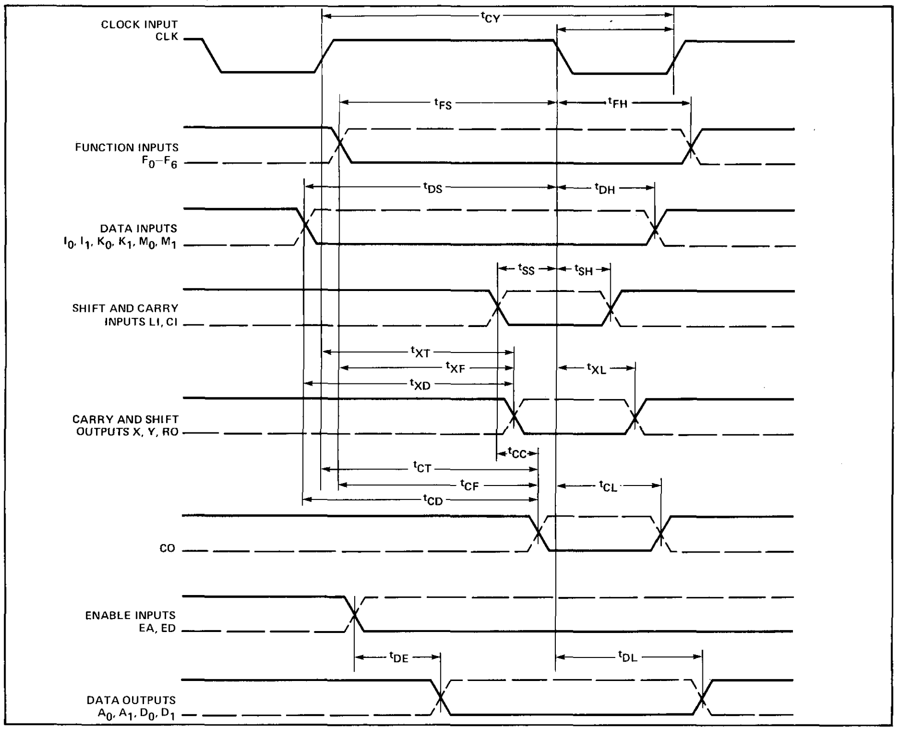

MCU
FO
FI
3001
74S08
74S51
74S51
P G
G P
74S182
74S182
C_n
C_n
P_3 G_3 C_{n+z} P_2 G_2 C_{n+y} P_1 G_1 C_{n+x} P_0 G_0
P_3 G_3 C_{n+z} P_2 G_2 C_{n+y} P_1 G_1 C_{n+x} P_0 G_0
x y
N3002
N3002
N3002
N3002
CI
CI
LI
RO
RO
LI
RO

TYPICAL CONFIGURATIONS



Ripple-Carry Configuration
(2N Bit Array)

PARAMETER MEASUREMENT INFORMATION



DESIGNER'S DEVELOPMENT KIT 3000 KT1000

DESCRIPTION

The Signetics Series 3000 Bipolar Microprocessor Chip Set provides new levels of high performance to microprocessor applications not previously possible with MOS technology. Combining the Schottky bipolar N3001 Microprogram Control Unit (MCU) and N3002 Central Processing Element (CPE) with industry standard memory and support circuits, microinstruction cycle times of 100ns are possible.

In the majority of cases, the choice of a bipolar microprocessor slice, as opposed to an MOS device, is based on speed or flexibility of microprogramming. Starting with these characteristics, the design of the Signetics Series 3000 Microprocessor has been optimized around the following objectives:

- Fast cycle time
- All memory and support chips are industry standard
- Cooler operation
- Lower total system cost

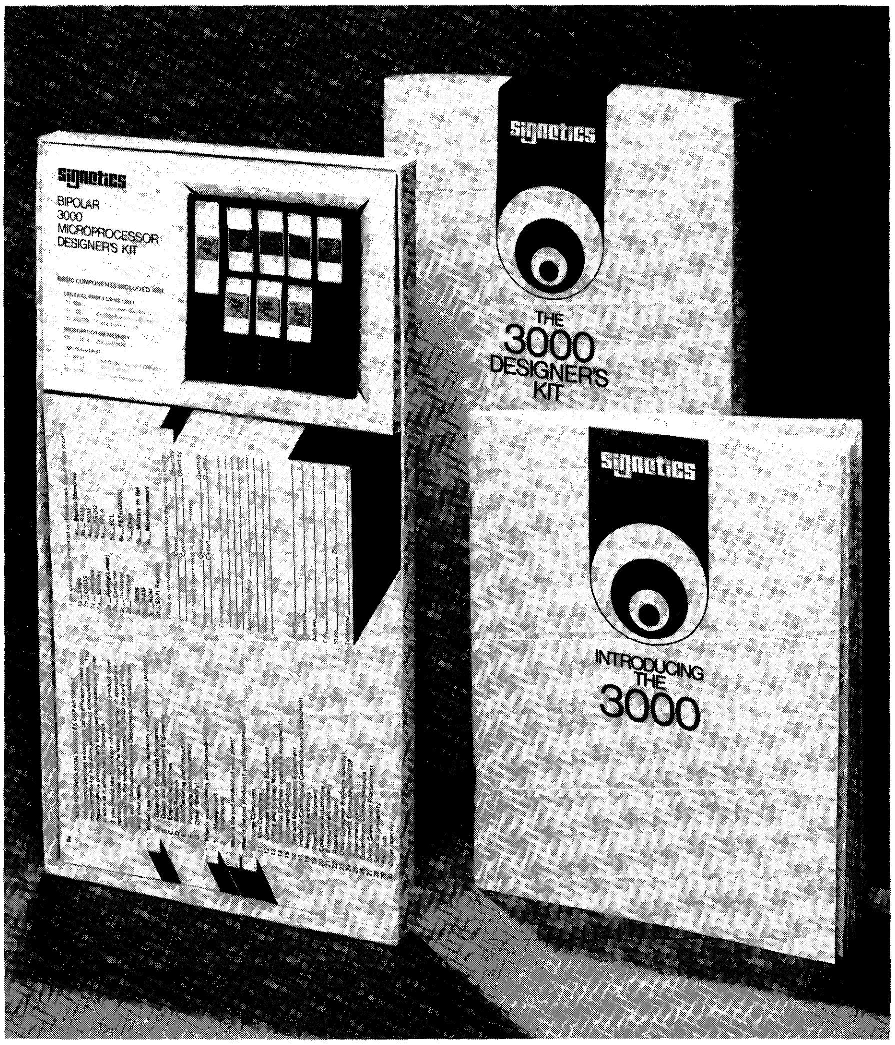
Systems built with large-scale integrated circuits are much smaller and require less power than equivalent systems using medium and/or small scale integrated circuits.

The 2 components of the Series 3000 chip set, when combined with industry standard memory and peripheral circuits, allows the design engineer to construct high-performance processors and/or controllers with a minimum amount of auxiliary logic. Features such as the multiple independent address and data buses, tri-state logic, and separate output enable lines eliminate the need for time-multiplexing of buses and associated hardware.

Each Central Processing Element represents a complete 2-bit slice through the data processing section of a computer. Several CPEs may be connected in parallel to form a processor of any desired word length. The Microprogram Control Unit controls the sequence in which microinstructions are fetched from the microprogram memory (ROM/PROM), with these microinstructions controlling the step-by-step operation of the processor.

Each CPE contains a 2-bit slice of 5 independent buses. Although they can be used in a variety of ways, typical connections are:

Input M-bus:	Carries data from external memory
Input I-bus:	Carries data from input/output device
Input K-bus:	Used for microprogram mask or literal (constant) value input
Output A-bus:	Connected to CPE Memory Address Register
Output D-bus:	Connected to CPE accumulator.



As the CPEs are paralleled together, all buses, data paths, and registers are correspondingly expanded.

The microfunction input bus (F-bus) controls the internal operation of the CPE, selecting both the operands and the operation to be executed upon them. The arithmetic logic unit (ALU), controlled by the microfunction decoder, is capable of over 40 Boolean and binary operations as outlined in the FUNCTION DESCRIPTION section of the N3002 data sheet. Standard carry look-ahead outputs (X and Y) are generated by the CPE for use with industry standard devices such as the 74S182.

FEATURES

- **Bipolar Schottky Technology**
- **Multiple Input/Output Bus Structure**
- **Fastest Microprocessor Available**
- **512 Microinstruction Addressibility**
- **Full Function Accumulator**

COST

$100.00 (Total Value = $230)

AVAILABILITY

Immediate delivery for Signetics Rep. or Distributors.

CONTENTS

1 ea—N3001 Microprogram Control Unit
4 ea—N3002 Central Processing Element
1 ea—74S182 Look-Ahead Carry
3 ea—82S114 256 × 8 PROM
1 ea—8T31 Bidirectional I/O Port
2 ea—8T26A Quad Bus Transceiver
1 ea—Introductory Manual

BLOCK DIAGRAM

MICROCOMPUTER BLOCK DIAGRAM

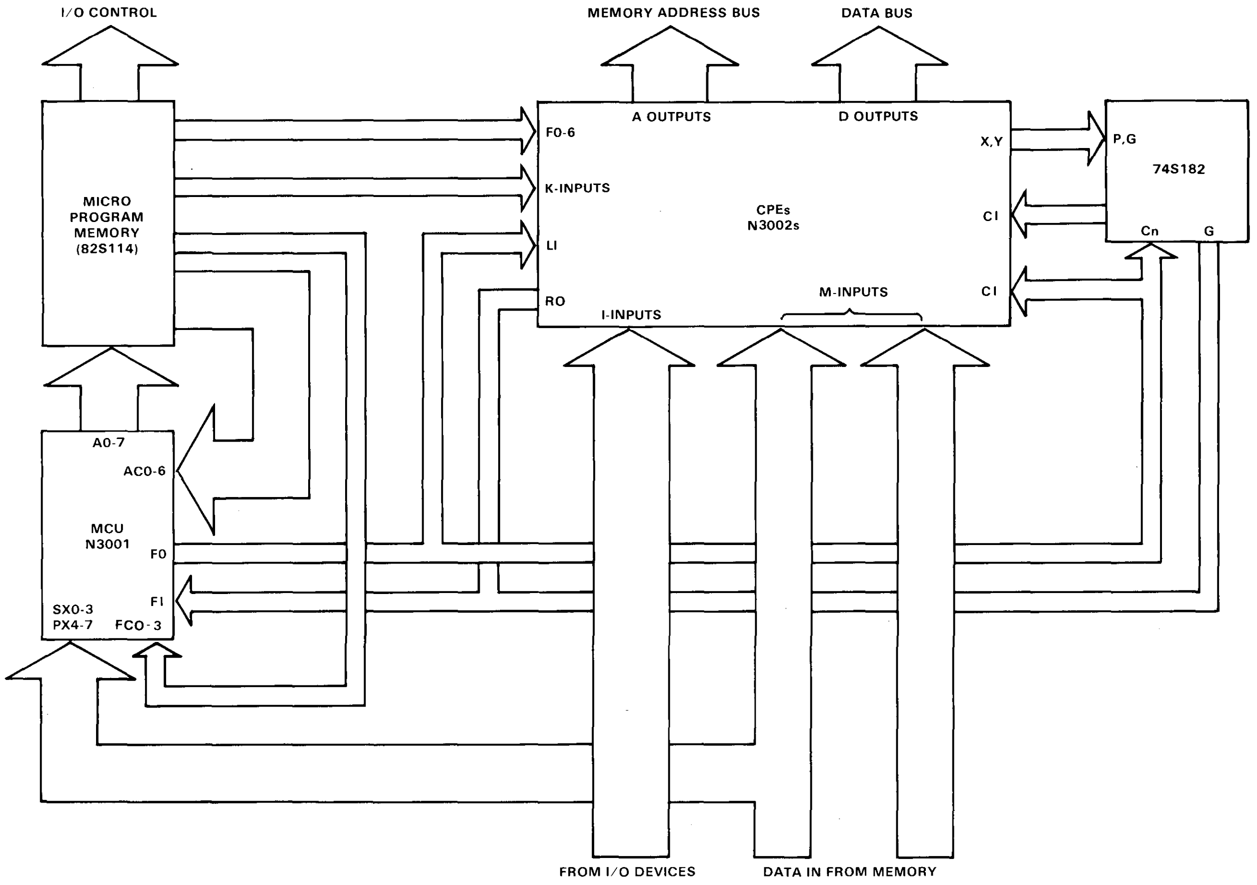


Figure 1

COMPATIBLE PRODUCTS

82S100, 82S101 FPLA

- Field programmable (Ni-Cr Link)
- Input variables—16
- Output functions—8
- Product terms—48
- Address access time—50ns
- Tri-state (82S100) or open collector (82S101) outputs
- 28-pin ceramic dip

82S115/123/129 PROMs

- Schottky TTL technology
- Single +5V power supply
- 32 × 8 organization (82S123)
- 256 × 8 organization (82S129)
- 512 × 8 organization (82S115)
- Field programmable (Nichrome)
- On-chip storage latches (82S115 only)
- Low current pnp inputs
- Tri-state outputs
- 35ns typical access time
- Standard 24-pin DIP (82S115)
- Standard 16-pin DIP (82S123, 82S129)

82S25/82S116/82S11 RAMs

- Schottky TTL technology
- 16 × 4 organization (82S25)
- 256 × 1 organization (82S116)
- 1024 × 1 organization (82S11)
- On-chip address decoding
- 16-pin ceramic dip

8T26A/8T28 Quad Transceiver

- Schottky TTL technology
- 4 pairs of bus drivers/receivers
- Separate drive and receive enable lines
- Tri-state outputs
- Low current pnp inputs
- High fan out-driver sinks 40mA
- 20ns maximum propagation delay
- Standard 16-pin DIP

8T31 8-Bit Bidirectional Port

- Schottky TTL technology
- 2 independent bidirectional buses
- 8-bit latch register
- Independent read, write controls for each bus
- Bus A overrides if a write conflict occurs
- Register can be addressed as a memory location
- via Bus B Master Enable
- 30ns maximum propagation delay
- Low input current: 500μA
- High fan out-sinks 20mA
- Standard 24-pin DIP

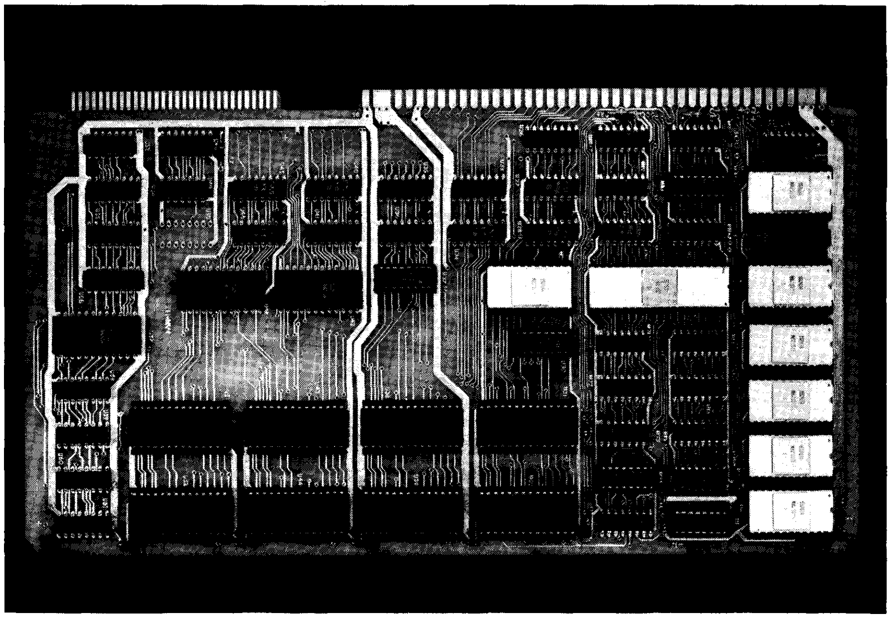
BIPOLAR EMULATION KIT FOR THE SERIES 3000 AN 8080 SYSTEM EMULATOR

3000KT8080SK

DESCRIPTION

The 8080 Emulation Kit is a microprogrammable microprocessor utilizing Schottky LSI components to implement an emulation of an Intel 8080A microcomputer system. The emulation is functionally equivalent to a microprocessor system incorporating the following Intel devices: 8080A, 8228, 8224 and 8212. The kit provides the standard address, data, status and control buses as defined in the Intel 8080 Microcomputer System Manual. Since the kit uses bipolar LSI elements, the emulator lacks the two-phase non-overlapping clock. Furthermore, those signals emanating from the 8080 during SYNC time are not provided, but rather the useful status signals provided by the 8228 system controller are implemented. The emulation also provides an extension of the 8228 operation during multi-byte interrupts. This is realized by allowing any 8080 program branch instruction to be inserted during interrupts rather than restricting multi-byte instructions to CALL during interrupts. Finally, a nonstandard status signal, RTRAP, is provided which indicates that the present instruction is a reserved or undefined instruction. After this indication, the processor will enter the normal HALT routine and await an interrupt. (Intel 8080A operation during undefined instructions is undefined.) Thus all 12 of the unused instructions in the 8080 instruction set are reserved for future instruction set expansion. These unused codes may be used at any time to extend the usual instruction set without requiring any reprogramming of the bipolar PROMs used for microprogram memory. Finally, the emulator is fully static so that the clock may be adjusted from a typical cycle time of 150ns to dc.

The kit contains all the parts necessary to construct the emulator and includes preprogrammed PROMs. The kit is designed to be assembled by a skilled technician in about 8 hours.



FEATURES

- **Full emulation of 8080A system**
- **Speed increase by factor of 2 to 9.2 over 8080A system**
- ***Static operation; microcycle time dc to 150ns***
- **Operation from single +5V supply**
- **Executes all 8080 instructions**
- **Hardware multiply and divide**
- **Microprogram expandable**
- **Includes single phase clock**
- **Full vectored interrupt to any location within 64K memory**

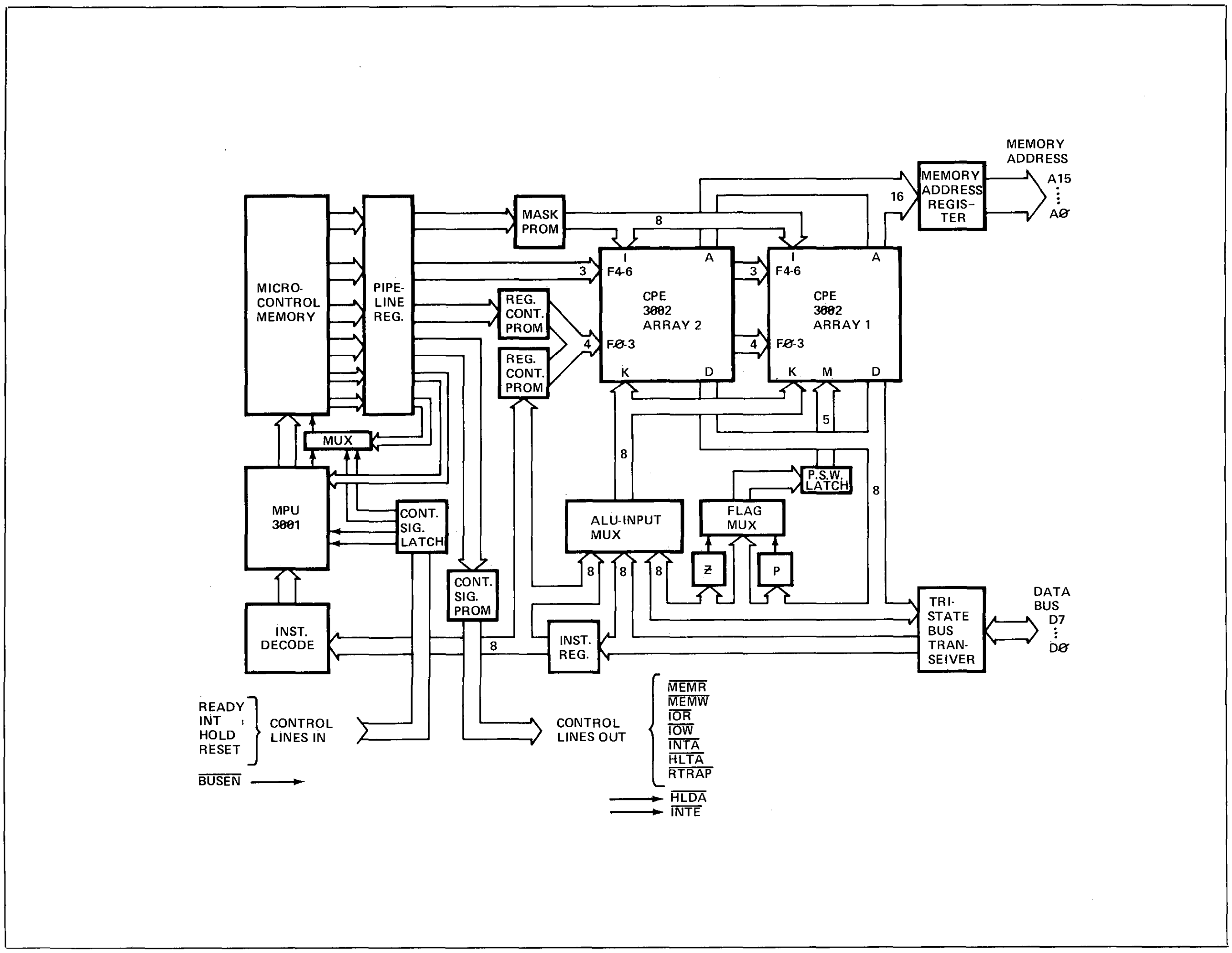
Part Number:	3000KT8080SK
Cost:	$299.00
Availability:	Immediate delivery from Signetics, Rep or Distributors

KIT CONTENTS

1 each	N74123
1 each	N3001
8 each	N3002
7 each	N82S115
1 each	N82S23
2 each	N82S123
2 each	N82S126
3 each	N8263
3 each	N74S182
1 each	N74S280
2 each	N7475
1 each	DM8613
11 each	N74S174
2 each	N8T28
3 each	N8T97
1 each	N74S153
2 each	N74S157
1 each	N7400
1 each	N74S02
3 each	N74S04
1 each	N74S08
1 each	N74S10
1 each	N74S133
2 each	Resistor networks 1K, 16-pin
1 each	P.C. board
1 each	Manual
1 each	Schematic
1 each	Set of microprogram listings
Plus:	Over 25 miscellaneous resistors, capacitors and other parts

BIPOLAR EMULATION KIT FOR THE SERIES 3000
AN 8080A SYSTEM EMULATOR — 3000KT8080SK

BLOCK DIAGRAM



INSTRUCTION EXECUTION TIMES (150ns microinstruction cycle time, 150ns RAM access time)

INSTRUCTION	CYCLES	EXECUTION TIME (μs)	INSTRUCTION	CYCLES	EXECUTION TIME (μs)
LXI	7	1.05	CMC	4	.60
PUSH	9	1.35	DAA	10	1.50
PUSH PSW	9	1.35	SHLD	11	1.65
POP	9	1.35	LHLD	12	1.80
POP PSW	8	1.20	EI	2	.30
STA	9	1.35	DI	2	.30
LDA	8	1.20	NOP	2	.30
XCHG	7	1.05	MUL	26	3.90
XTHL	13	1.95	MOV r1, r2	3	.45
SPHL	3	.45	MOV M, r	6	.90
PCHL	6	.90	MOV , M	6	.90
DAD	4	.60	HLT	4	.60
STAX	6	.90	MVI r	4	.60
LDAX	5	.75	MVI M	7	1.05
INX	2	.30	INR	3	.45
DCX	3	.45	DCR	4	.60
CMA	2	.30	INR M, DCR M	8	1.20
STC	3	.45	Arithmetic reg	4	.60

INSTRUCTION	CYCLES	EXECUTION TIME (μs)
Arithmetic mem	7	1.05
Arithmetic immed.	4	.60
RLC	4	.60
RRC	3	.45
RAL	4	.60
RAR	3	.45
JMP	6	.90
JMP[1]	4	.60
CALL	13	1.95
CALL[1]	4	.60
RET	8	1.20
RET[1]	2	.30
RST	13	1.95
IN	8	1.20
OUT	8	1.20

	Min	Typ[2]	Max
DIV	4.80	6.60	8.40

NOTES

1. Conditional branch with condition not met.
2. Depends upon value of divisor.

MICROASSEMBLER

DESCRIPTION

The Signetics Microassembler is a complete software package designed to support general slice system architectures which employ Signetics bipolar microprocessor components. It provides a flexible microassembly language required to describe the numerous configurations in which bipolar microprocessor chips are used. The program allows for the entry of specific device model parameters and hardware configurations which permits the symbolic assembly language to relate directly to the user's system. In particular, the microassembler provides primary support for the 3002 and 2901 central processing units and the 8X02 Control Store Sequencer through intrinsic field definitions for these devices. Through proper modification of the models used in the microassembler, it supports the 3001 Microprogram Control Unit as well as the 8X300 Fixed Instruction Bipolar Microprocessor.

In keeping with the varied nature of slice microprocessor systems, the microassembler provides for complete flexibility in adapting to the user's system definition. The microassembler provides for variable microinstruction formats as defined by the user. Moreover, op codes may be user-defined. The following operators are supported by the microassembler:

A+B	Add A and B
A-B	Add the complement of B to A
-A	Complement A
A*B	Multiply A and B
A/B	Compute the quotient of A divided by B
A.MOD.B	Compute the remainder of A divided by B
A.SHL.B	Shift A left B bits (supply zeroes to emptied bit positions)
A.SHR.B	Shift A right B bits
.NOT.A	Invert all bits of A
A.AND.B	Logical and of bits in A and B
A.OR.B	Logical inclusive or of bits in A and B
A.XOR.B	Logical exclusive or of bits in A and B

The microassembler itself is adaptable to convenient implementation by the user. The basic microassembler is written in standard ANSI FORTRANIV. Therefore, the program may be run on virtually any machine of sufficient memory size which has the requisite FORTRAN compiler. Or, if desired, the program may be accessed via TYMSHARE, GE, or NCSS Timesharing Services.

The outputs of the Signetics Microassembler will provide for complete system development and documentation. The microassembly program is a two-pass assembler. Upon assembly, a program listing is obtained which contains error messages relative to the success of the assembly process. This output is used in the debug stage of program development to produce code of the correct format. In the second pass, the complete program listing including source and object information directly provides the documentation required for the system firmware. A second output is also provided which is intended for the production of punched paper tape. The paper tape is designed for: 1) the production of PROMs which are to contain the object microprogram; 2) entry into ROM simulators for system checkout; and 3) entry into other microprocessor development systems.

CHAPTER 2
BIPOLAR FIXED INSTRUCTION MICROPROCESSOR

INTRODUCTION

A Microcomputer Designed for Control

The 8X300 is a microcomputer designed for control. It features:

Execution Speed

- 250ns instruction execution time
- Direct address capability—up to 8192 16-bit words of program memory
- Eight 8-bit general purpose registers
- Simultaneous data transfer and data edit in a single instruction cycle time
- *n-way branch or n-entry table lookup in 2* instruction cycle times
- 8X300 instructions operate with equal speed on 1-bit, 2-bit, 3-bit, 4-bit, 5-bit, 6-bit, 7-bit, or 8-bit data formats

The 8X300 instruction set features control-oriented instructions which directly access variable length input/output and internal data fields. These instructions provide very high performance for moving and interpreting data. This makes the 8X300 ideal in switching, controlling, and editing applications.

Direct Processing of External Data

The 8X300 I/O system is treated as a set of internal registers. Therefore data from external devices may be processed (tested, shifted, added to, etc.) without first moving them to internal storage. In fact, the entire concept is to treat data at the I/O interface no differently than internal data. This concept extends to the software which allows variables at the input/output system to be named and treated in the same way as data in storage.

Separate Program Storage and Data Storage

The storage concept of the 8X300 is to separate program storage from data storage. Program storage is implemented in read-only memory in recognition of the fact that programs for control applications are fixed and dedicated. The benefits of using read-only memory are that great speeds may be obtained at lower cost than if read/write memory were used, and that program instructions reside in a non-volatile medium and cannot be altered by system power failures.

8X300 Architecture

Figure 2 of the 8X300 data sheet illustrates the 8X300 architecture. The 8X300 contains an Arithmetic Logic Unit (ALU), Program Counter, and an Address Register. Eight 8-bit general purpose registers are also provided, including 7 working registers and an auxiliary register which performs as a working register and also provides an implied operand for many instructions. The 8X300 registers are shown in Figure 2 of the 8X300 data sheet and are summarized below:

Control Registers include:

- Instruction—A 16-bit register containing the current instruction
- Program Storage Address Register (AR)—A 13-bit register containing the address of the current instruction being accessed from Program Storage
- Program Counter (PC)—A 13-bit register containing the address of the next instruction to be read from Program Storage

Data Registers include:

- Working Registers (WR)—Seven 8-bit registers for data storage
- Overflow (OVF)—A 1-bit register that retains the most significant bit position carry from ALU addition operation. Arithmetically treated as 2°
- Auxiliary (AUX)—An 8-bit register. Source of implied operand for arithmetic and logical instructions. May be used as a working register.

A crystal external to the CPU may be used to generate the CPU system clock. The CPU executes 8 instruction types.

INTERPRETER 8X300

8X300-I

DESCRIPTION

The Signetics 8X300 Interpreter is a monolithic, high-speed microprocessor implemented with bipolar Schottky technology. As the central processing unit, CPU, it allows 16-bit instructions to be fetched, decoded and executed in 250ns. A 250ns instruction cycle requires maximum memory access of 65ns, and maximum I/O device access of 35ns.

Interpreter instructions operate on 8-bit, parallel data. Logic is distributed along the data path within the Interpreter. Input data can be rotated and masked before being subject to an arithmetic or logical operation; and output data can be shifted and merged with the input data, before being output to external logic. This allows 1- to 8-bit I/O and data memory fields to be accessed and processed in a single instruction cycle.

PROGRAM STORAGE INTERFACE

Program Storage is typically connected to the A0-A12 (A12 is least significant bit) and I0-I15 signal lines. An address output on A0-A12 identifies one 16-bit instruction word in program storage. The instruction word is subsequently input on I0-I15 and defines the interpreter operations which are to follow.

The Signetics 82S115 PROM, or any TTL compatible memory, may be used for program storage.

I/O DEVICES INTERFACE

An 8-bit I/O bus, called the Interface Vector (IV) data bus, is used by the Interpreter to communicate with 2 fields of I/O devices. The complementary $\overline{LB}$ and $\overline{RB}$ signals identify which field of the I/O devices is selected.

Both I/O data and I/O address information can be output on the IV bus. The SC and WC signals are typically used to distinguish between I/O data and I/O address information as follows:

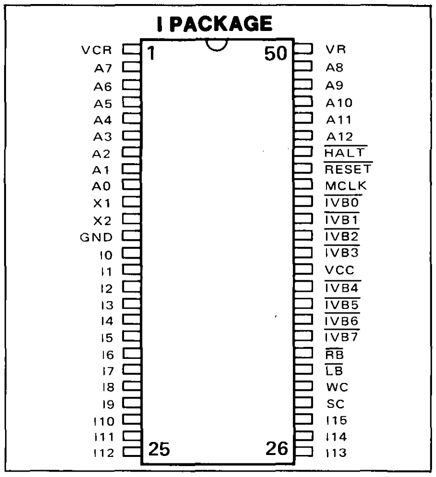
SC	WC	
1	0	I/O address is being output on IV bus
0	1	I/O data is being output on IV bus
0	0	I/O data is expected on the IV, bus, as input to the Interpreter
1	1	Not generated by the Interpreter

The Signetics 82SXXX series RAM, and the 8T32/33 may be attached to the IV bus.

FEATURES

- **185ns instruction decode and execute delay (with Signetics 8T32/33 I/O port)**
- **Eight 8-bit working registers**
- **Single instruction access to 1-bit, 2-bit, 3-bit . . . or 8-bit field on I/O bus**
- **Separate instruction address, instruction, and I/O data busses**
- **On-chip oscillator**
- **Bipolar Schottky technology**
- **TTL inputs and outputs**
- **Tri-state output on I/O data bus**
- **+5 volt operation from 0° to 70° C**

PIN CONFIGURATION



PIN DESCRIPTION

PIN	SYMBOL	NAME AND FUNCTION	TYPE
2-9, 45-49	A0-A12:	Instruction address lines. A high level equals "1." These outputs directly address up to 8192 words of program storage. A12 is least significant bit.	Active high
13-28	I0-I15:	Instruction lines. A high level equals "1." Receives instructions from Program Storage. I_{15} is least significant bit.	Active high
33-36, 38-41	$\overline{IVB0}$-$\overline{IVB7}$	Interface Vector (IV) Bus. A low level equals "1." Bidirectional tri-state lines to communicate with I/O devices. $\overline{IVB7}$ is least significant bit.	Three-state Active low
42	$\overline{MCLK}$:	Master Clock. Output to clock I/O devices, and/or provide synchronization for external logic	
30	WC:	Write Command. High level output indicates data is being output on the IV Bus.	Active high
29	SC:	Select Command. High level output indicates that an address is being output on the IV Bus.	Active high
31	$\overline{LB}$:	Left Bank. Low level output to enable one of two sets of I/O devices ($\overline{LB}$ is the complement of RB).	Active low
32	$\overline{RB}$:	Right Bank. Low level output to enable one of two sets of I/O devices ($\overline{RB}$ is the complement of $\overline{LB}$).	Active low
44	$\overline{HALT}$:	Low level is input to stop the Interpreter.	Active low
43	$\overline{RESET}$:	Low level is input to initialize the Interpreter.	Active low
10-11	X1, X2:	Inputs for an external frequency determining crystal. May also be interfaced to logic or test equipment.	
50	VR	Reference voltage to Pass Transistor.	
1	VCR	Regulated output voltage from Pass Transistor.	
37	VCC:	5V power connection.	
12	GND:	Ground.	

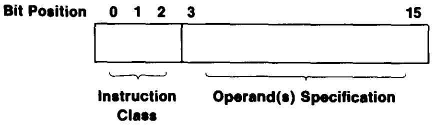
INSTRUCTION CYCLE

Each interpreter operation is executed in 1 instruction cycle, which may be as short as 250ns. The Interpreter generates MCLK to synchronize external logic to the instruction cycle. Instruction cycles are subdivided into quarter cycles. MCLK is an output during the last quarter cycle.

During the third quarter cycle of an instruction, an address is output on A0-A12, identifying the location in program storage of the next instruction word. This instruction word defines the next instruction, which must be input on I0-I15 during the first quarter cycle of the next instruction cycle (see Table 1).

Instruction Set Summary

The 16-bit instruction word input on I0-I15 is decoded by the instruction decode logic to implement events that are to occur during the remainder of the instruction cycle. Generally the 16-bit instruction word is decoded as follows:



A detailed usage of the 13 "operand(s) specification" bits is given in following sections.

TYPICAL SYSTEM CONFIGURATION

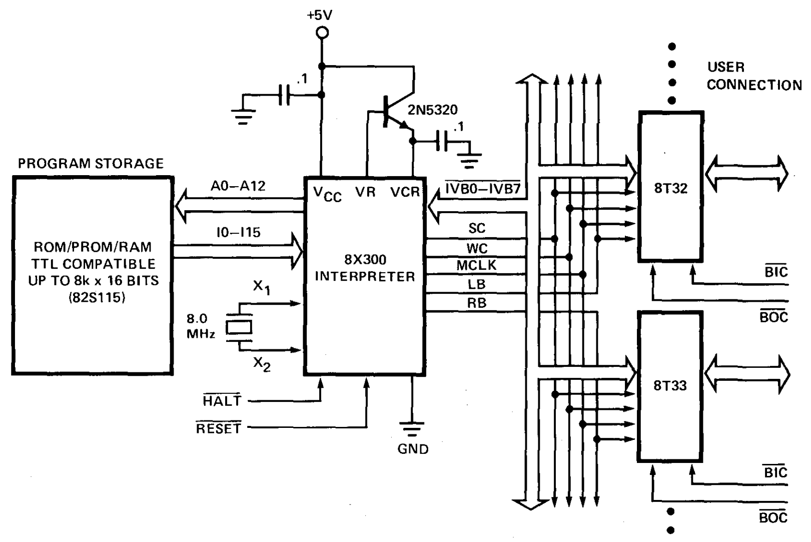


Figure 1

INST. AND IV BUS DATA INPUT	DATA PROCESSING	ADDR. AND IV BUS CHANGING	ADDR. AND IV BUS DATA VALID MCLK = HIGH
← ¼ cycle →	← ¼ cycle →	← ¼ cycle →	← ¼ cycle →

Table 1 INSTRUCTION CYCLE

Three operation code bits allow for 8 instruction classes. The 8 instruction classes are summarized in Table 2. Each entry is referred to as an "instruction class" because the unique architecture of the Interpreter allows a number of powerful variations to be specified by the 13 operand(s) specification bits. A complete description of instruction formats and some instruction examples are provided in the Application Notes.

Data Processing

The Interpreter architecture includes eight 8-bit working registers, an arithmetic logic unit (ALU), an overflow register, and the 8-bit IV Bus. Internal 8-bit data paths connect the registers and IV Bus to the ALU inputs, and the ALU output to the registers and IV Bus. Data processing logic is distributed along these internal 8-bit data paths. Rotate and mask logic precedes the ALU on the data entry path. Shift and merge logic precedes the ALU on the data entry path. Shift and merge logic follows the ALU on the data output path. All 4 sets of logic can operate on 8 data bits in a single instruction cycle. (See Figure 2)

When less than 8 bits of data are specified for output to the IV bus by the ALU, the data field (shifted if necessary) is inserted into the prior contents of the IV bus latches. The IV bus latches contain data input at the start of an instruction. This data in the IV bus latches will be specified in the instruction as a) IV bus source data or b) data from an automatic read when the IV bus is specified as a destination. Therefore, IV bus bit positions outside an inserted bit field are unmodified.

Data Addressing

Sources and destinations of data are specified using a 5-bit octal number, as shown in Table 2. The source and/or destination of data to be operated upon is specified in a single instruction word.

Referring to Table 3, the Auxiliary register (address 00) is the implied source of the second argument for ADD, AND or XOR operations.

IVL and IVR are write-only registers used only as a destination. They have addresses and are treated as registers, but in reality they do not exist. When IVL is specified as a destination or the D field = 20-27$_8$, then LB = 'low', RB = 'high' are generated; when IVR is specified as a destination or the D field = 30-37$_8$, then RB = low, LB = 'high' are generated.

When IVL or IVR is specified as the destination in an instruction, SC is also activated and data is placed on the IV bus. If IVL or IVR is specified as a source of data, the source data is all zeroes.

INSTRUCTION SEQUENCE CONTROL

The Address Register and Program Counter are used to generate addresses for accessing an instruction. The Address Register is used to form the instruction address, and in all but 3 instructions (XEC, NZT, and JMP) the address is copied into the Program Counter. The instruction address is formed in 1 of 3 ways:

1. For all instructions but the JMP, XEC, and a satisfied NZT, the Program Counter is incremented by 1 and placed in the Address Register.
2. For the JMP instruction, the full 13-bit address field from the JMP instruction is placed into the Address Register and copied into the Program Counter.
3. For the XEC and NZT instructions, the high order 5- or 8-bits of the Program Counter are combined with 8- or 5-lower-order bits of ALU output (XEC or NZT) and placed in the Address Register. For the NZT instruction, it is also copied into the Program Counter.

INTERPRETER ARCHITECTURE

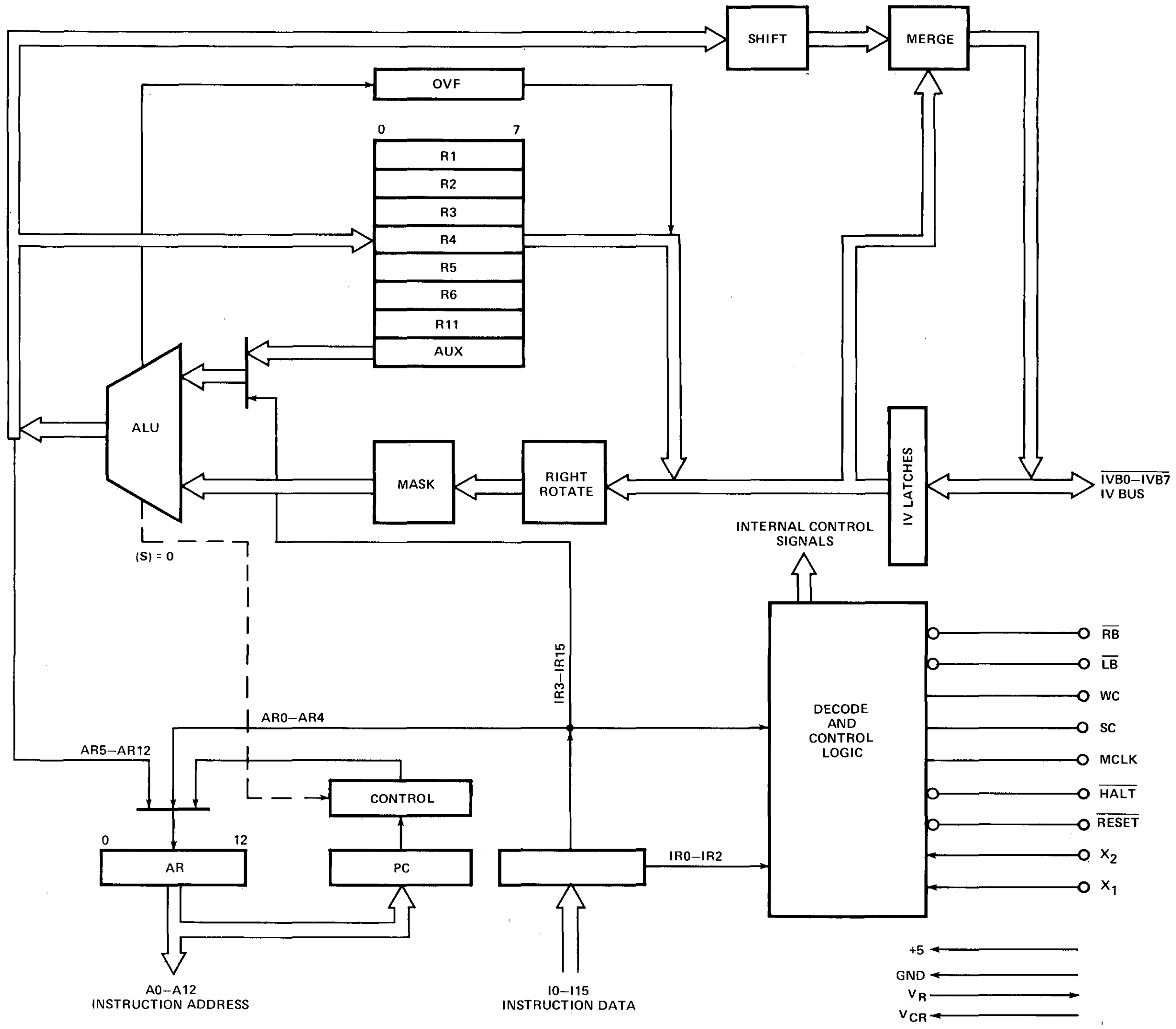


Figure 2

INSTRUCTION MNEMONIC	OP CODE	FORMATS	DESCRIPTION	I/O CONTROL SIGNALS	→ INSTRUCTION CYCLE →	
					Instruction Input and Data Processing	Address/IV Bus Output
MOVE	0	Register to Register 0 2 3 7 8 10 11 15 0 | S | R | D $S\neq 07,17,20\text{-}37_8$ $D\neq 10,20\text{-}37_8$	(S) → D Move contents of register specified by S to register specified by D. Right rotate contents of register S by R places before operation.	SC = WC = $\overline{\text{LB}}$/RB = $\overline{\text{LB}}$/RB =	0 0 X X	1 if D = 07,17 0 1 if D = 17 0 if D = 07
		IV Bus to Register: 0 2 3 7 8 10 11 15 0 | S | L | D $S=20\text{-}37_8$ $D\neq 10,20\text{-}37_8$	Move right rotated IV bus (source) data specified by S to register specified by D. L specifies the length of source data with most significant bits set to zero.	SC = WC = $\overline{\text{LB}}$/RB = $\overline{\text{LB}}$/RB =	0 0 0 if S = 20-27 1 if S = 30-37	1 if D = 07,17 0 1 if D = 17 0 if D = 07
		Register to IV Bus: 0 2 3 7 8 10 11 15 0 | S | L | D $S\neq 07,17,20\text{-}37_8$ $D=20\text{-}37_8$	Move contents of register specified by S to the IV bus. Before placement on IV bus, data is shifted as specified by D, and L bits merged with destination IV bus data.	SC = WC = $\overline{\text{LB}}$/RB = $\overline{\text{LB}}$/RB =	0 0 0 if D = 20-27 1 if D = 30-37	0 1 0 if D = 20-27 1 if D = 30-37
		IV Bus to IV Bus: 0 2 3 7 8 10 11 15 0 | S | L | D $S=20\text{-}37_8$ $D=20\text{-}37_8$	Move right rotated IV bus data (sources) specified by S to the IV bus. Before placement on IV bus, data is shifted or specified by D and merged with original source data. L specifies the length of source data to be operated on.	SC = WC = $\overline{\text{LB}}$/RB = $\overline{\text{LB}}$/RB =	0 0 0 if S = 20-27 1 if S = 30-37	0 1 0 if D = 20-27 1 if D = 30-37
ADD	1	SAME AS MOVE	(S) plus (AUX) → D Same as MOVE but contents of AUX added to the source data. If carry from most significant bit then OVF = 1, otherwise OVF = 0		SAME AS MOVE	
AND	2	SAME AS MOVE	(S)∧(AUX) → D Same as MOVE but contents of AUX ANDed with source data.		SAME AS MOVE	
XOR	3	SAME AS MOVE	(S) ⊕ (AUX) → D Same as MOVE but contents of AUX exclusive ORed with source data.		SAME AS MOVE	
XEC	4	Register Immediate: 0 2 3 7 8 15 4 | S | I $S\neq 07,17,20\text{-}37_8$ $I=000\text{-}377_8$	Execute instruction at current page address offset by I + (S). Execute the instruction at the address determined by concatenating 5 high order bits of PC with the 8 bit sum of I and register specified by S. PC is not incremented.	SC = WC = $\overline{\text{LB}}$/RB	0 0 x	0 0 x
		IV Bus Immediate: 0 2 3 7 8 10 11 15 4 | S | L | I $S=20\text{-}37_8$ $I=00\text{-}37_8$	Execute the instruction at the address determined by concatenating 8 high order bits of PC with the 5 bit sum of I and rotated IV bus data (source) specified by S. R/L specifies length of source data with most significant bits set to zero. PC is not incremented.	SC = WC = $\overline{\text{LB}}$/RB = $\overline{\text{LB}}$/RB =	0 0 0 if S = 20-27 1 if S = 30-37	0 0 x x

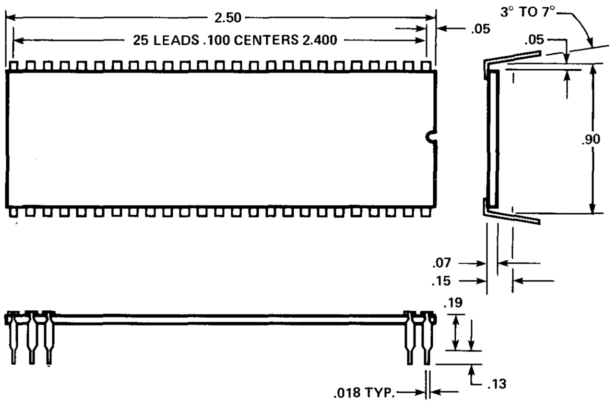
Table 2 INSTRUCTION SET SUMMARY

INSTRUCTION MNEMONIC	OP CODE	FORMATS	DESCRIPTION	I/O CONTROL SIGNALS	← INSTRUCTION CYCLE →	
					Instruction Input and Data Processing	Address/IV Bus Output
NZT	5	Register Immediate: 0 2 3 7 8 15 5 S I S≠07,17,20-37$_8$ I=000-377$_8$	If (S) = 0, jump to current page address offset by I; if S = 0, PC + 1 → PC If contents of register specified by S is non zero then transfer to address determined by concatenating 5 high order bits of PC with I; if contents of register specified by S is zero, increment PC.	SC = WC = $\overline{LB}$/RB =	0 0 x	0 0 x
		IV Bus Immediate: 0 2 3 7 8 10 11 15 5 S L I S=20-37$_8$ I=00-37$_8$	If right rotated IV bus data (source) is Non Zero then Transfer to address determined by concatenating 8 high order bits of PC with I; if contents of register specified by S is zero, increment PC.	SC = WC = $\overline{LB}$/RB = $\overline{LB}$/RB =	0 0 0 if S = 20-27 1 if S = 30-37	0 0 x x
XMIT	6	Register Immediate: 0 2 3 7 8 15 6 D I D ≠ 10, 20-37$_8$ I=000-377$_8$	Transmit I → D Transmit and store 8 bit binary pattern I to register specified by D.	SC = WC = $\overline{LB}$/RB = $\overline{LB}$/RD =	0 0 x x	1 if D = 07,17 0 1 if D = 17 0 if D = 07
		IV BUS IMMEDIATE 0 2 3 7 8 10 11 15 6 D L I D=20-37$_8$ I=00-37$_8$	Transmit binary pattern I to IV bus. Before placement on IV bus, literal I is shifted as specified by D and L bits merged with existing IV bus data.	SC = WC = $\overline{LB}$/RB = $\overline{LB}$/RB =	0 0 0 if D = 20-27 1 if D = 30-37	0 1 0 if D = 20-27 1 if D = 30-37
JMP	7	Address Immediate: 0 2 3 15 7 A A=00000-17777$_8$	Jump to Program Address A Jump to program storage address A. A is stored in the address register (AR).	SC = WC = $\overline{LB}$/RB =	0 0 x	0 0 x

NOTE
1. RB is complement of LB.
2. "0" = Low voltage
"1" = High voltage
x = Don't care

Table 2 **INSTRUCTION SET SUMMARY** (Cont'd)

INTERPRETER PACKAGE DIMENSIONS



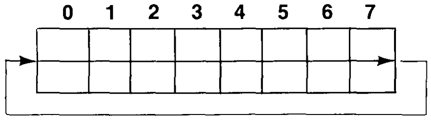
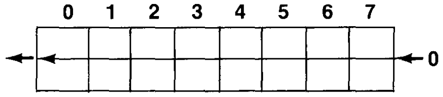
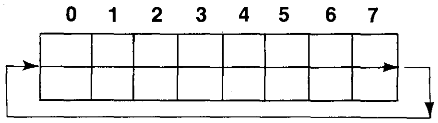
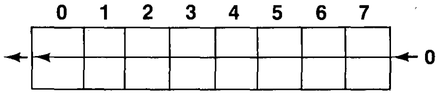
S AND/OR D FIELD SPECIFICATION (OCTAL)	SOURCE/DESTINATION
00	Auxiliary Register (AUX)
01 to 06	Work registers (R1 to R6) respectively
07	IVL write-only "register" (destination only)
10	Overflow status (OVF)—source only
11	Working register (R11)
17	IVR write-only "register" (destination only)
2N (N = 0,1,2, 3,4,5,6,7)	a. If a source, IV bus data right rotated (7—N) bits and masked (specified by R/L). $\overline{LB}$ = 'low' and $\overline{RB}$ = 'high' generated. **IV Bus Source Data**  b. If a destination, IV bus data left shifted (7—N) bits and merged (specified by $\overline{R/L}$). LB = 'low' and $\overline{RB}$ = 'high' generated. **IV Bus Destination Data** 
3N (N = 0,1,2, 3,4,5,6,7)	a. If a source, IV bus data right rotated (7—N) bits and masked (specified by R/L). $\overline{LB}$ = 'high' and $\overline{RB}$ = 'low' generated. **IV Bus Source Data**  b. If a destination, IV bus data left shifted (7—N) bits and merged (specified by R/L). $\overline{LB}$ = 'high' and $\overline{RB}$ = 'low' generated. **IV Bus Destination Data** 

Table 3 DATA SOURCE DESTINATION ADDRESS

INTERPRETER INTERNAL REGISTERS

Programmable Registers (all 8 bits):

AUX— General working register. Contains second term for arithmetic or logical operations.

R1 — General working register

R2 — General working register

R3 — General working register

R4 — General working register

R5 — General working register

R6 — General working register

R11 — General working register

Other Registers:

Address Register (AR)
— A 13-bit register containing the address of the current instruction.

OVF — The least-significant bit of this register is used to reflect overflow status resulting from the most recent ADD operation (see Instruction Set Summary).

Program Counter (PC)
— Normally contains the address of the current instruction and is incremented to obtain the next instruction address.

Instruction Register (IR)
— Holds the 16-bit instruction word currently being executed.

Table 4

SYSTEM DESIGN USING THE INTERPRETER

Designing hardware around the 8X300 Interpreter reduces to selecting a program storage devicer (ROM, PROM, etc.), selecting I/O devices (IV byte, multiplexers, RAM, etc.), selecting clock mode (system driven or crystal controlled) and interfacing the Interpreter to these components, as shown in Figure 3.

System Clock

The Interpreter has an integrated oscillator which generates all necessary clock signals. The oscillator is designed to connect directly to a series resonant quartz crystal via pins X1 and X2. The crystal resonant frequency, f, is related to the desired cycle time, T, by the relationship f = 2/T. For a 250ns system, f = 8.00MHz.

In lower speed applications where the cycle time need not be precisely controlled, a capacitor may be connected between X1 and X2 to drive the oscillator. Approximate capacitor values are given in Table 6. If cycle time is to be varied, X1 and X2 should be driven from complementary outputs of a pulse generator. Figure 4 shows a typical configuration. For systems where the Interpreter is to be driven from a master clock, the X1 and X2 lines may be interfaced to TTL logic as shown in Figure 5.

Type:	Fundamental mode, series resonant
Impedance at Fundamental:	35 ohms maximum
Impedance at harmonics and spurs:	50 ohms minimum

Table 5 CRYSTAL CHARACTERISTICS

EXAMPLE OF CONTROL SYSTEM

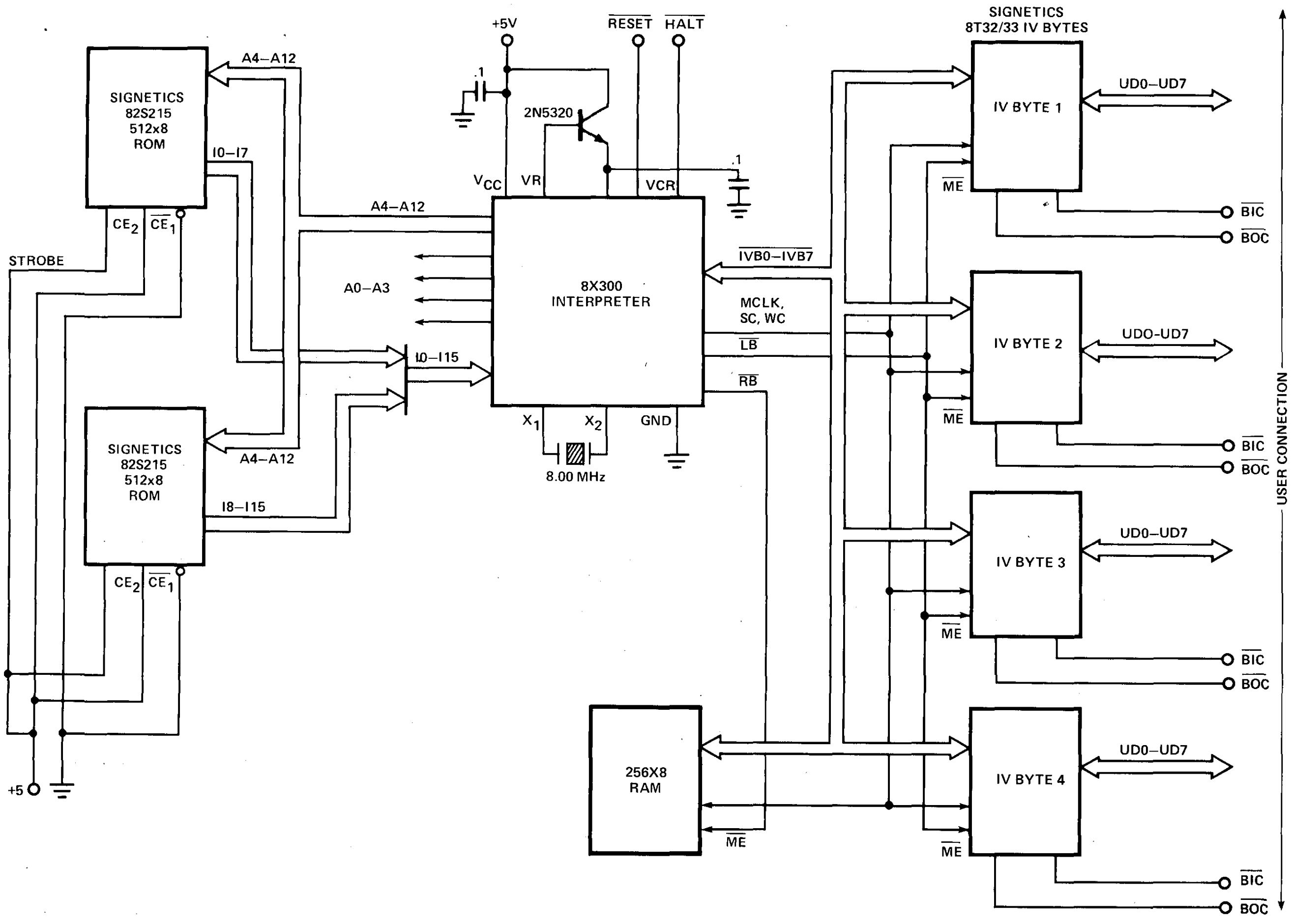


Figure 3

Cx,pF	CYCLE TIME
100	300ns
200	500ns
500	1.1μs
1000	2.0μs

Table 6 CLOCK CAPACITOR VALVES

Halt, Reset Signals

$\overline{\text{HALT}}$:

A low level at the $\overline{\text{HALT}}$ input stops internal operation of the Interpreter at the start of the next instruction after $\overline{\text{HALT}}$ is applied (quarter cycle after MCLK). Since $\overline{\text{HALT}}$ is sampled at the start of each instruction cycle it is possible to prevent a cycle by applying $\overline{\text{HALT}}$ early in that cycle. $\overline{\text{HALT}}$ does not inhibit MCLK or affect any internal registers. Normal operation begins with the next complete cycle after the $\overline{\text{HALT}}$ input goes high.

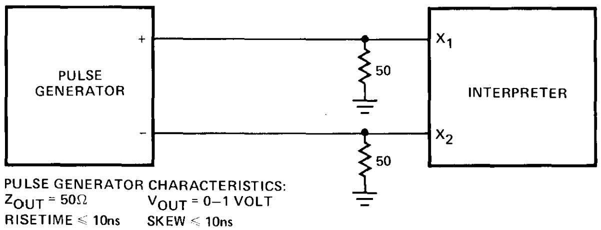
$\overline{\text{RESET}}$:

A low level at the $\overline{\text{RESET}}$ input sets the program counter and address register to zero. While $\overline{\text{RESET}}$ is low MCLK is inhibited. If $\overline{\text{RESET}}$ is applied during the last 2 quarter cycles, the MCLK during that cycle may be shortened. $\overline{\text{RESET}}$ should be applied for 1 full instruction cycle time to assure proper operation. When $\overline{\text{RESET}}$ input goes high an MCLK occurs prior to the resumption of normal processing. $\overline{\text{RESET}}$ does not affect the other internal registers.

EXAMPLE:

A specific example of a control system, using the 8X300 Interpreter—four 8T32/33 IV Bytes, and two 82S215 ROMs is shown in Figure 3. Only 8 components are required to build this system which contains 512 words of program storage, 32 TTL I/O connection points, and operates at a 250ns instruction cycle time.

CLOCKING WITH A PULSE GENERATOR

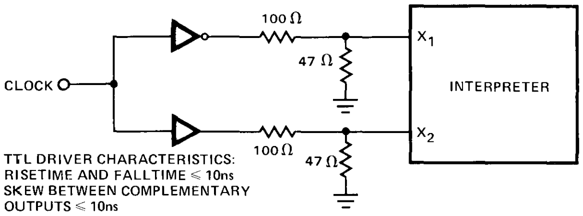


PULSE GENERATOR CHARACTERISTICS:
$Z_{OUT} = 50\Omega$ $V_{OUT} = 0\text{–}1$ VOLT
RISETIME ≤ 10ns SKEW ≤ 10ns
COMPLEMENTARY OUTPUTS

NOTE: ALL RESISTORS VALUES ARE TYPICAL AND IN OHMS.

Figure 4

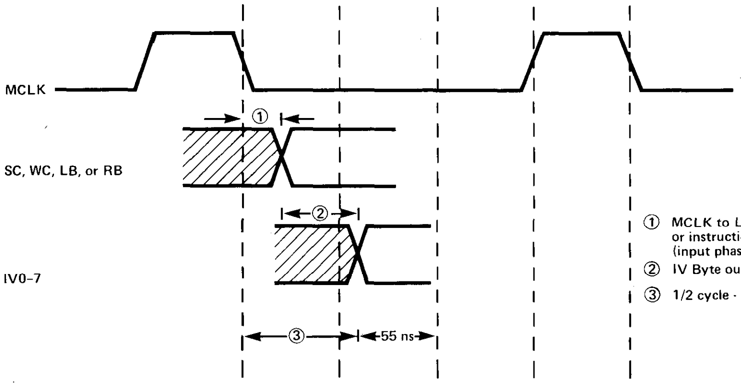
CLOCKING WITH TTL



TTL DRIVER CHARACTERISTICS:
RISETIME AND FALLTIME ≤ 10ns
SKEW BETWEEN COMPLEMENTARY OUTPUTS ≤ 10ns

Figure 5

SYSTEM INSTRUCTION CYCLE TIME



① MCLK to LB/RB (input phase) or instruction to LB/RB (input phase).
② IV Byte output enable (T_{OE}).
③ 1/2 cycle - 55ns.

Figure 6

SYSTEM TIMING

In systems with fast instruction cycle times, most Interpreter delays are strictly determined by internal gate propagation delays. Since some events are constrained to occur in certain quarter cycles, as system cycle times become slower, the delays will appear to increase due to gating with internal clocks. In the table of AC Electrical Characteristics, 2 columns are used: 1 to denote times which occur due to internal clock intervention and 1 to denote minimum delays for fast cycle times.

When using Signetics 8T32/33/35/36 IV Bytes, selection of instruction cycle time involves calculating the maximum program storage access time. Assuming the instruction is available when MCLK falls, the I/O control lines are stable 35ns later. Signetics IV Bytes require another 35ns to disable a previously selected byte and enable the desired byte (assumes a change in bank signals). A 10ns margin has been added to the IV Byte enable for this evaluation to reflect the fact that most systems will have more capacitive loading than the 50pF test condition in the IV Byte specification and to allow for line delays.

The system instruction cycle time for·normal systems such as shown in Figure 7 is determined by Interpreter propagation delays, program storage access time, and IV Byte output enable times. Instruction cycle time is normally constrained by one or more of the following conditions:

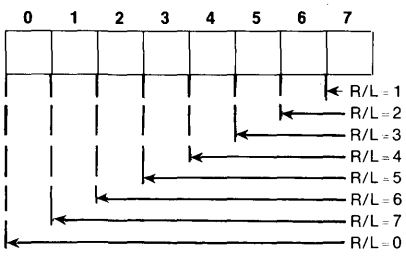
1. Instruction to LB/RB (input phase) and IV Byte output enable:
   $T_{OE} \leq \frac{1}{2}$ cycle – 55ns (Figure 6).

2. Program storage access time and instruction to LB/RB (input phase) and IV Byte output enable and IV data (input phase) to address ≤ instruction cycle time (Figure 7).
3. Program storage access time and instruction to address ≤ instruction cycle time (Figure 8).

The first constraint can be used to determine the minimum cycle time. Using the inequality $35ns + 35ns \leq \frac{1}{2}$ cycle - 55ns implies $\frac{1}{2}$ cycle ≥ 125ns or an instruction time of 250ns.

Program storage access time for a 250ns instruction cycle can be calculated from the second constraint. Noting that the specification for IV data (input phase) to address is 115ns: Program storage access time + 35ns + 35ns + 115ns ≤ 250ns implies program storage access time ≤ 65ns.

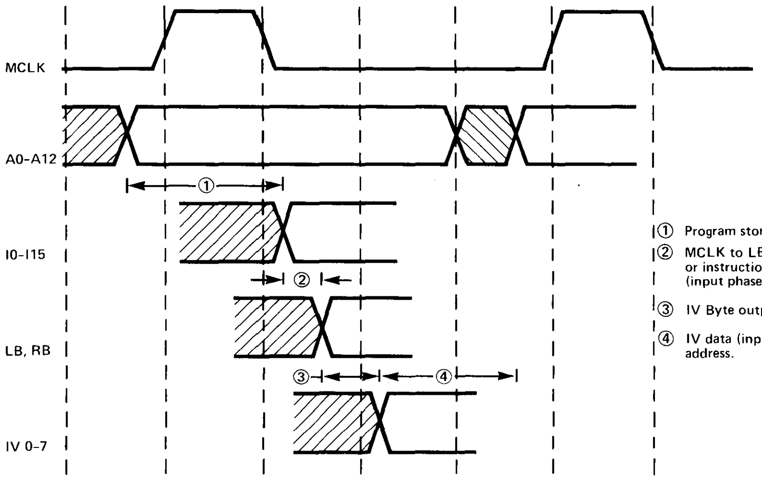
The third constraint can be used to verify the maximum program storage access time. Noting that the specification for instruction to address is 185ns: Program storage access time + 185ns ≤ 250ns confirms that program storage access time 65ns is satisfactory.



IV BUS DATA
LENGTH SPECIFICATION

Figure 9

SYSTEM INSTRUCTION CYCLE TIME



① Program storage access time.
② MCLK to LB/RB (input phase) or instruction to LB/RB (input phase).
③ IV Byte output enable (T_{OE}).
④ IV data (input phase) to address.

Figure 7

SYSTEM INSTRUCTION CYCLE TIME

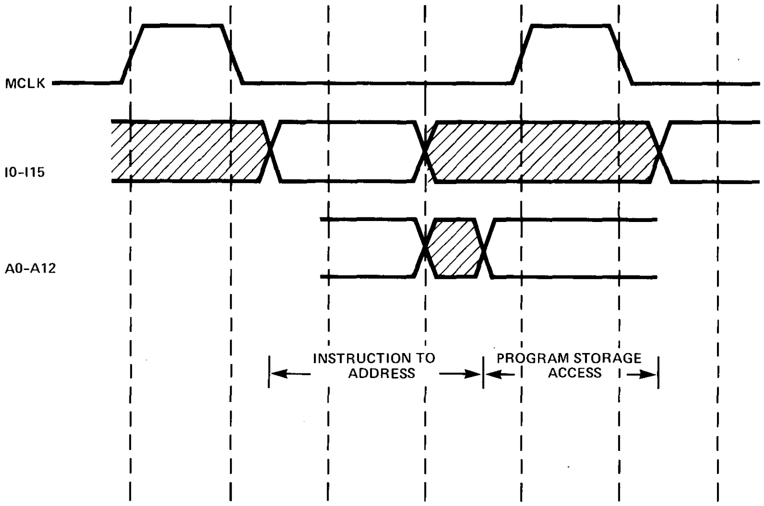


Figure 8

TYPICAL INSTRUCTION CYCLE TIMING

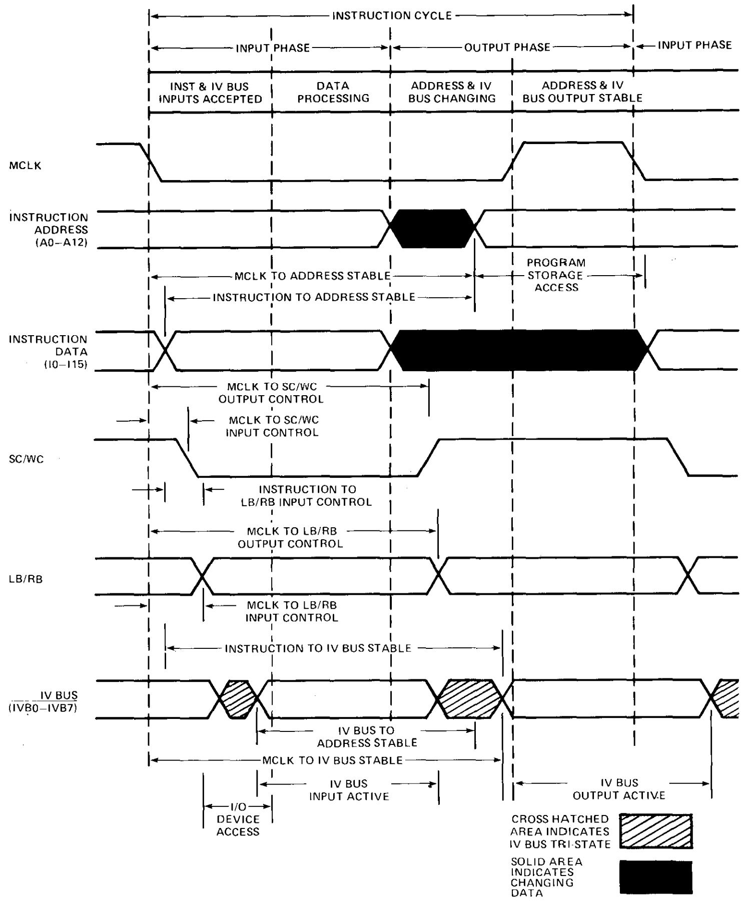

INSTRUCTION CYCLE
INPUT PHASE
OUTPUT PHASE
INPUT PHASE
INST & IV BUS INPUTS ACCEPTED
DATA PROCESSING
ADDRESS & IV BUS CHANGING
ADDRESS & IV BUS OUTPUT STABLE
MCLK
INSTRUCTION ADDRESS (A0–A12)
MCLK TO ADDRESS STABLE
PROGRAM STORAGE ACCESS
INSTRUCTION TO ADDRESS STABLE
INSTRUCTION DATA (I0–I15)
MCLK TO SC/WC OUTPUT CONTROL
MCLK TO SC/WC INPUT CONTROL
SC/WC
INSTRUCTION TO LB/RB INPUT CONTROL
MCLK TO LB/RB OUTPUT CONTROL
LB/RB
MCLK TO LB/RB INPUT CONTROL
INSTRUCTION TO IV BUS STABLE
IV BUS (IVB0–IVB7)
IV BUS TO ADDRESS STABLE
MCLK TO IV BUS STABLE
IV BUS INPUT ACTIVE
I/O DEVICE ACCESS
IV BUS OUTPUT ACTIVE
CROSS HATCHED AREA INDICATES IV BUS TRI-STATE
SOLID AREA INDICATES CHANGING DATA

Figure 10

ABSOLUTE MAXIMUM RATINGS

Supply Voltage V_{CC}	7V
Logic Input Voltage	5.5V
Crystal Input Voltage	2V

AC ELECTRICAL CHARACTERISTICS V_{CC} = 5V ± 5% and 0°C $\leqslant T_A < 70$°C

DELAY DESCRIPTION	PROPAGATION DELAY TIME	CYCLE TIME LIMIT
X1 falling edge to MCLK (driven from external pulse generator)	75ns	
MCLK to SC/WC falling edge (input phase)	25ns	
MCLK to SC/WC rising edge (output phase)		½ cycle + 25ns
MCLK to LB/RB (input phase)	35ns	
Instruction to LB/RB output (input phase)	35ns	
MCLK to LB/RB (output phase)		¼ cycle + 35ns
MCLK to IV data (output phase)	185ns	½ cycle + 60ns
IV data (input phase) to IV data (output phase)	115ns	
Instruction to Address	185ns	½ cycle + 40ns
MCLK to Address	185ns	½ cycle + 40ns
IV data (input phase) to Address	115ns	
MCLK to IV data (input phase)		½ cycle – 55ns
MCLK to Halt falling edge to prevent current cycle		¼ cycle – 40ns
Reset rising edge to first MCLK		0 to 1 cycle

NOTE

1. Reference to MCLK is to the falling edge when loaded with 300pF.
2. Loading on Address lines is 150pF.

DC ELECTRICAL CHARACTERISTICS

PARAMETER		TEST CONDITIONS	LIMITS			UNIT
			Min	Typ	Max	
V_{IH}	High level input voltage					
	X1,X2		.6			V
	All others		2			V
V_{IL}	Low level input voltage					
	X1,X2				.4	V
	All others				.8	V
V_{CL}	Input clamp voltage (Note 1)	V_{CC} = 4.75V I_I = -10mA			-1.5	V
I_{IH}	High level input current					
	X1,X2	V_{CC} = 5.25V V_{IH} = .6V		2700		μA
	All others	V_{CC} = 5.25V V_{IH} = 4.5V		<1	50	μA
I_{IL}	Low level input current					
	X1,X2	V_{CC} = 5.25V V_{IL} = .4V		-2500		μA
	$\overline{\text{IVBO}}$-7	V_{CC} = 5.25V V_{IL} = .4V		-140	-200	μA
	IO-I15	V_{CC} = 5.25V V_{IL} = .4V		-880	-1600	μA
	$\overline{\text{HALT}}$, $\overline{\text{RESET}}$	V_{CC} = 5.25V V_{IL} = .4V		-230	-400	μA
V_{OL}	Low level output voltage					
	A0-A12	V_{CC} = 4.75V I_{OL} = 4.25mA		.35	.55	V
	All others	V_{CC} = 4.75V I_{OL} = 16mA		.35	.55	V
V_{OH}	High level output voltage	V_{CC} = 4.75V I_{OH} = 3mA	2.4			V
I_{OS}	Short circuit output current (Note 2)	V_{CC} = 5.25V	-30		-140	mA
V_{CC}	Supply voltage		4.75	5	5.25	V
I_{CC}	Supply current	V_{CC} = 5.25V		300	450	mA
I_{REG}	Regulator control	V_{CC} = 5.0V	-14		-21	mA
I_{CR}	Regulator current (Note 3)	V_{CR} = 0			290	mA
V_{CR}	Regulator voltage (Note 3)	V_{REG} = 0V	2.2		3.2	V

NOTES

1. Crystal inputs X1 and X2 do not have clamp diodes.
2. Only one output may be grounded at a time.
3. (Limits apply for V_{CC} = 5V ± 5% and 0°C < T_A < 70°C unless specified otherwise.)

DESCRIPTION

The MicroController Cross Assembly Program (MCCAP) is a program designed to translate symbolic instructions into object code that can be executed by the 8X300. This program will run on most computers that have a Fortran compiler with a computer word length of at least 16 bits and a random access I/O capability; it can be run on most minicomputers as well as large scale computers.

The Assembler is written in Fortran. This provides compatibility with most computer systems and makes it transportable. The program is modular and uses a minimum of memory. However, it may be operated in an overlay mode if necessary.

FEATURES

- **9 track EBCDIC tape**
- **Cross Assembler (written in Fortran)**
- **Flexible Object Format MCSIM
ROM Simulator
ROM Production Format**
- **Symbolic Addressing**
- **Forward References**
- **Expression Evaluation**
- **Symbolic I/O Representation**

The Assembler is a two-pass program that issues helpful error messages and produces an easily read program listing. An object module may be loaded into the SMS MicroController Simulator (MCSIM), the SMS ROM Simulator 1000A, or used to produce ROMs or PROMs.

The Assembler features symbolic addressing, forward references and expression evaluation. It also has the capability to symbolically represent the Interface Vector (IV). In addition, the Assembler is capable of expressing data in several number systems as well as in ASCII character codes.

MCCAP Output

The output from a MCCAP compilation includes an assembler listing and an object module. During pass 2 of the assembly process, a program listing is produced. The listing displays all information pertaining to the assembled program. This includes the assembled octal instructions, the user's original source code and error messages. The listing may be used as a documentation tool through the inclusion of comments and remarks which describe the function of a particular program segment. The main purpose of the listing, however, is to convey all pertinent information about the assembled program, i.e., the memory addresses and their contents.

The object module is also produced during pass 2. This is a machine readable computer output produced on paper tape. The output module contains the specifications necessary for loading the memory of the SMS.*

Microcontroller Simulator (MCSIM), for loading the memory of ROM Simulator, or for producing ROMs or PROMs. The object module can be produced in MCSIM, ROM Simulator or BNPF format.

*Scientific Micro System
520 Clyde Avenue
Mountain View, CA 94043

8-BIT LATCHED ADDRESSABLE BIDIRECTIONAL I/O PORT 8T32/8T33/8T35/8T36

8T32/8T33/8T35/8T36-NA,F

TYPES

8T32 Tri-State, Synchronous User Port
8T33 Open Collector, Synchronous User Port
8T35 Open Collector, Asynchronous User Port
8T36 Tri-State, Asynchronous User Port

DESCRIPTION

The Interface Vector (IV) Byte is an 8-bit bidirectional data register designed to function as an I/O interface element in microprocessor systems. It contains 8 data latches accessible from either a microprocessor (IV) port or a user port. Separate I/O control is provided for each port. The 2 ports operate independently, except when both are attempting to input data into the IV Byte. In this case, the user port has priority.

A unique feature of the 8T32/33/35/36 IV Byte is the way in which it is addressed. Each IV Byte has an 8-bit, field programmable address, which is used to enable the microprocessor port. When the SC control signal is high, data at the microprocessor port is treated as an address. If the address matches the IV Byte's internally programmed address, the microprocessor port is enabled, allowing data transfer through it.

The port remains enabled until an address which does not match is presented, at which time the port is disabled (data transfer is inhibited). A Master Enable input ($\overline{ME}$) can serve as a ninth address bit, allowing 512 IV Bytes to be individually selected on a bus, without decoding. The user port is accessible at all times, independent of whether or not the microprocessor port is selected.

A unique feature of this family is its ability to start up in a predetermined state. If the clock is maintained at a voltage less than .8V until the power supply reaches 3.5V, the user port will always be all logic 1 levels, while the IV port will be all logic 0 levels.

ORDERING

The 8T32/33/35/36 may be ordered in preaddressed form. To order a preaddressed IV Byte, use the following part number format:

N8TYY-XXX P

−P= F Ceramic package
NA Plastic package

—XXX= Any address from 000 through 255 (decimal) - 256 available addresses

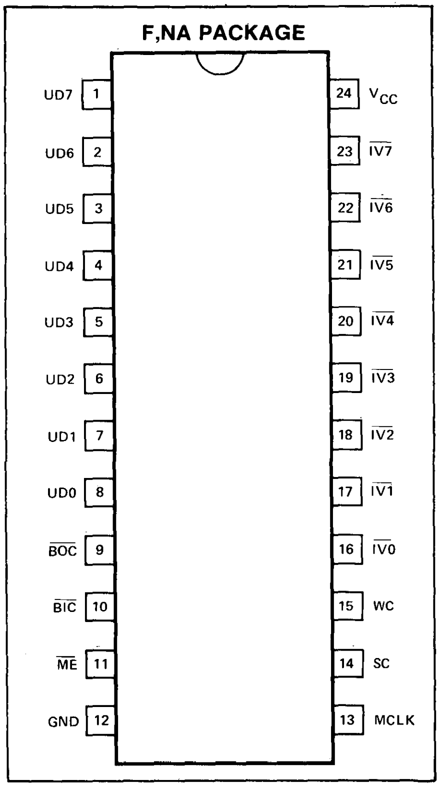
—YY= IV Byte version (32, 33, 35, 36)

A stock of 8T32s and 8T36s with addresses 1 through 10 will be maintained. A small quantity of addresses 11 through 50 will also be available with a longer lead time.

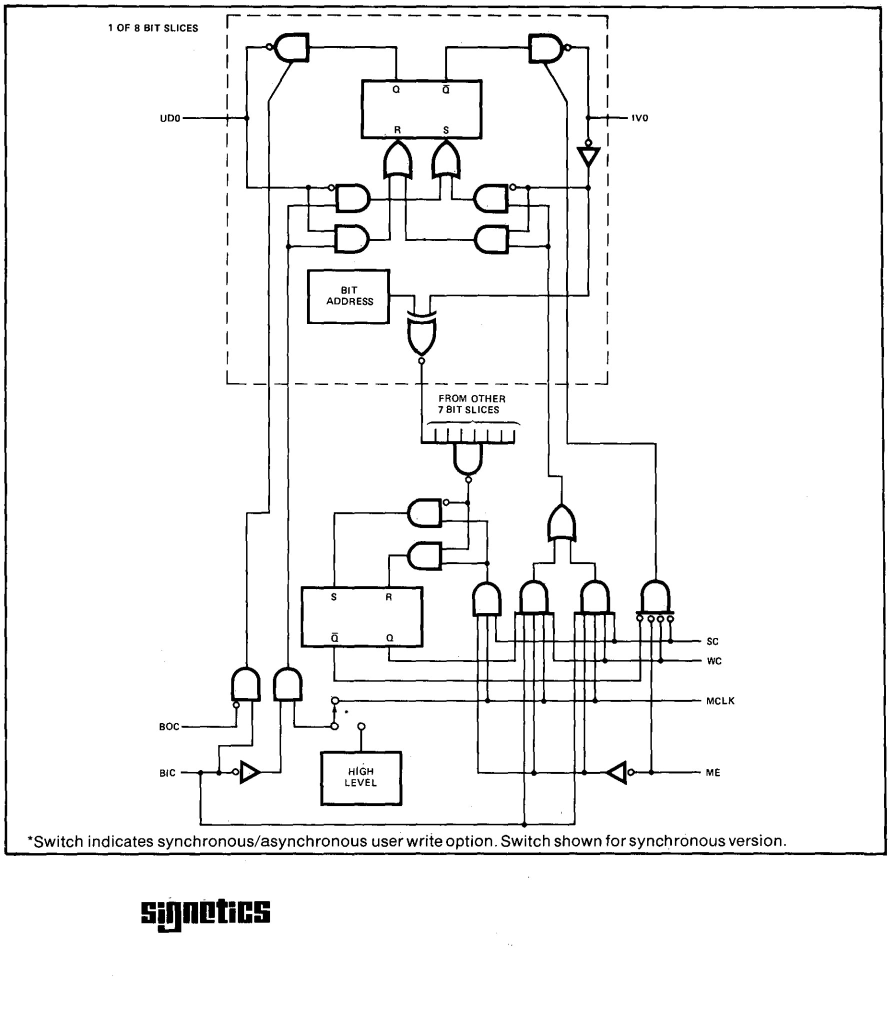
FEATURES

- **A field-programmable address allows 1 of 512 IV Bytes on a bus to be selected, without decoders.**
- **Each byte has 2 ports, one to the user, the other to a microprocessor. IV Bytes are completely bidirectional.**
- **Ports are independent, with the user port having priority for data entry.**
- **A selected IV Byte de-selects itself when another IV Byte address is sensed.**
- **User data input available as synchronous (8T32, 8T33) or as asynchronous (8T35, 8T36) function.**
- **The User Data Bus is available with tri-state (8T32, 8T36) or open collector (8T33, 8T35) outputs.**
- **At power up, the IV Byte is not selected and the user port outputs are high.**
- **Tri-state TTL outputs for high drive capability.**
- **Directly compatible with the 8X300 Interpreter.**
- **Operates from a single 5V power supply over a temperature range of 0° C to 70° C.**

PIN CONFIGURATION



BLOCK DIAGRAM



*Switch indicates synchronous/asynchronous user write option. Switch shown for synchronous version.

PIN DESCRIPTION

PIN	SYMBOL	NAME AND FUNCTION	TYPE
1-8	$\overline{\text{UD0}}$-$\overline{\text{UD7}}$:	User Data I/O Lines. Bidirectional data lines to communicate with user's equipment. Either tri-state or open collector outputs are available.	Active high
16-23	$\overline{\text{IV0}}$-$\overline{\text{IV7}}$:	Interface Vector (IV) Bus. Bidirectional data lines to communicate with controlling digital system (microprocessor).	Active low three-state
10	$\overline{\text{BIC}}$:	Byte Input Control. User input to control writing into the IV Byte from the User Data Lines.	Active low
9	$\overline{\text{BOC}}$:	Byte Output Control. User input to control reading from the IV Byte onto the User Data Lines.	Active low
11	$\overline{\text{ME}}$:	Master Enable. System input to enable or disable all other system inputs and outputs. It has no effect on user inputs and outputs.	Active low
15	WC:	Write Command. When WC is high and SC is low, IV Byte, if selected, stores contents of IV0-IV7 as data.	Active high
14	SC:	Select Command. When SC is high and WC is low, data on IV0-IV7 is interpreted as an address. IV Byte selects itself if its address is identical to IV bus data; it de-selects itself otherwise.	Active high
13	MCLK:	Master Clock. Input to strobe data into the latches. See function tables for details.	Active high
24	VCC:	5V power connection.	
12	GND:	Ground.	

$\overline{\text{BIC}}$	$\overline{\text{BOC}}$	MCLK	USER DATA BUS FUNCTION	
			8T32, 8T33	8T35, 8T36
H	L	X	Output Data	Output Data
L	X	H	Input Data	Input Data
L	X	L	Inactive	Input Data
H	H	X	Inactive	Inactive

H = High Level L = Low Level X = Don't care

Table 1 USER PORT CONTROL FUNCTION

$\overline{\text{ME}}$	SC	WC	MCLK	$\overline{\text{BIC}}$	STATUS LATCH	IV BUS FUNCTION
L	L	L	X	X	SET	Output Data
L	L	H	H	H	SET	Input Data
L	H	L	H	X	X	Input Address
L	H	H	H	L	X	Input Address
L	H	H	H	H	X	Input Data and Address
L	X	H	L	X	X	Inactive
L	H	X	L	X	X	Inactive
L	L	H	H	L	X	Inactive
L	L	X	X	X	Not Set	Inactive
H	X	X	X	X	X	Inactive

Table 2 MICROPROCESSOR PORT CONTROL FUNCTION

USER DATA BUS CONTROL

The activity of the User Data Bus is controlled by the BIC and BOC inputs as shown in Table 1.

For the 8T32 and 8T33, User Data Input is a synchronous function with MCLK. A low level on the BIC input allows data on the User Data Bus to be written into the Data Latches only if MCLK is at a high level. For the 8T35 and 8T36, User Data Input is an asynchronous function. A low level on the BIC input allows data on the User Data Bus to be latched regardless of the level of the MCLK input. Note that when 8T35 or 8T36 IV Bytes are used with the 8X300 Interpreter care must be taken to insure that the IV Bus is stable when it is being read by the 8X300 Interpreter.

To avoid conflicts at the Data Latches, input from the Microprocessor Port is inhibited when BIC is at a low level. Under all other conditions the 2 ports operate independently.

INTERFACE VECTOR BUS CONTROL

As is shown in Table 2, the activity of the microprocessor port (IV Bus) is controlled by the ME, SC, WC and BIC inputs, as well as the state of an internal status latch. BIC is included to show user port priority over the microprocessor port for data input.

Each IV Byte's status latch stores the result of the most recent IV Byte select; it is set when the IV Byte's internal address matches the IV Bus. It is cleared when an address that differs from the internal address is presented on the IV Bus.

In normal operation, the state of the status latch acts like a master enable; the microprocessor port can transfer data only when the status latch is set.

When SC and WC are both high, data on the IV Bus is accepted as data, whether or not the IV Byte was selected. The data is also interpreted as an address. The IV Byte sets its select status if its address matches the data read when SC and WC were both high; it resets its select status otherwise.

BUS OPERATION

Data written into the IV Byte from one port will appear inverted when read from the other port. Data written into the IV Byte from one port will not be inverted when read from the same port.

AC ELECTRICAL CHARACTERISTICS

PARAMETER	INPUT	TEST CONDITION	LIMITS			UNIT
			Min	Typ	Max	
t_{PD} User data delay (Note 1)	UDX MCLK* BIC†	C_L = 50pF		25 45 40	38 61 55	ns
t_{OE} User output enable	BOC	C_L = 50pF	18	26	47	ns
t_{OD} User output disable	BIC BOC	C_L = 50pF	18 16	28 23	35 33	ns
t_{PD} IV data delay (Note 1)	IVBX MCLK	C_L = 50pF		38 48	53 61	ns
t_{OE} IV output enable	ME SC WC	C_L = 50pF	14	19	25	ns
t_{OD} IV output disable	ME SC WC	C_L = 50pF	13	17	32	ns
t_W Minimum pulse width	MCLK BIC†		40 35			ns
t_{SETUP} Minimum setup time	UD□ BIC* IVX ME SC WC	(Note 2)	15 25 55 30 30 30			ns
t_{HOLD} Minimum hold time	UDX□ BIC* IVX ME SC SC	(Note 2)	25 10 10 5 5 5			ns

* Applies for 8T32 and 8T33 only.

† Applies for 8T35 and 8T36 only.

□ Times are referenced to MCLK for 8T32 and 8T33, and are referenced to BIC for 8T35 and 8T36.

NOTES:

1. Data delays referenced to the clock are valid only if the input data is stable at the arrival of the clock and the hold time requirement is met.
2. Set up and hold times given are for "normal" operation. BIC setup and hold times are for a user write operation. SC setup and hold times are for an IV Byte select operation. WC setup and hold times are for an IV Bus write operation. ME setup and hold times are for both IV write and select operations.

DC ELECTRICAL CHARACTERISTICS V_{CC} = 5V ± 5%, 0°C ⩽ T_A ⩽ 70°C unless otherwise specified

PARAMETER		TEST CONDITIONS	LIMITS			UNIT
			Min	Typ	Max	
	Input voltage					V
V_{IH}	High		2.0			
V_{IL}	Low				.8	
V_{IC}	Clamp	I_I = -5mA			-1	
	Output voltage	V_{CC} = 4.75V				V
V_{OH}	High		2.4			
V_{OL}	Low				.55	
	Input current[3]	V_{CC} = 5.25V				µA
I_{IH}	High	V_{IH} = 5.25V		<10	100	
I_{IL}	Low	V_{IL} = .5V		-350	-550	
	Output current[4]					mA
I_{OS}	Short circuit	V_{CC} = 4.75V				
	UD bus		10			
	IV bus		20			
I_{CC}	V_{CC} supply current	V_{CC} = 5.25V		100	150	mA

PROGRAMMING SPECIFICATIONS[5]

PARAMETER		TEST CONDITIONS	LIMITS			UNITS
			Min	Typ	Max	
V_{CCP}	Programming supply voltage					
	Address		7.5		8.0	V
	Protect			0		V
I_{CCP}	Programming supply current	V_{CCP} = 8.0V			250	mA
	Max time V_{CCP} > 5.25V				1.0	s
	Programming voltage					
	Address		17.5		18.0	V
	Protect		13.5		14.0	V
	Programming current					
	Address				75	mA
	Protect				150	mA
	Programming pulse rise time					
	Address		.1		1	µs
	Protect		100			µs
	Programming pulse width		.5		1	ms

NOTES

[3] The input current includes the tri-state/open collector leakage current of the output driver on the data lines.

[4] Only one output may be shorted at a time.

[5] If all programming can be done in less than 1 second, VCC may remain at 7.75V for the entire programming cycle.

ADDRESS PROGRAMMING

The IV Byte is manufactured such that an address of all high levels (> 2V) on the IV Data Bus inputs matches the Byte's internal address. To program a bit so a low-level input (< 0.8V) matches, the following procedure should be used:

1. Set all control inputs to their inactive state (BIC = BOC = ME = V_{CC}, SC = WC = MCLK = GND). Leave all IV Data Bus I/O pins open.
2. Raise V_{CC} to 7.75V ± .25V.
3. After V_{CC} has stabilized, apply a single programming pulse to the User Data Bus bit where a low-level match is desired. The voltage should be limited to 18V; the current should be limited to 75mA. Apply the pulse as shown in Figure 1.
4. Return V_{CC} to 0V. (Note 6).
5. Repeat this procedure for each bit where a low-level match is desired.
6. Verify that the proper address is programmed by setting the Byte's status latch (IV0-IV7 = desired address, ME = WC = L, SC = MCLK = H). If the proper address has been programmed, data presented at the IV Bus will appear inverted on the User Bus outputs. (Use normal V_{CC} and input voltage for verification.)

After the desired address has been programmed, a second procedure must be followed to isolate the address circuitry. The procedure is:

1. Set V_{CC} and all control inputs to 0V. (V_{CC} = BIC = BOC = ME = SC = WC = MCLK = 0V). Leave all IV Data Bus I/O pins open.
2. Apply a protect programming pulse to every User Data Bus pin, one at a time. The voltage should be limited to 14V; the current should be limited to 150mA. Apply the pulse as shown in Figure 2.
3. Verify that the address circuitry is isolated by applying 7V to each User Data Bus pin and measuring less than 1mA of input current. The conditions should be the same as in step 1 above. The rise time on the verification voltage must be slower than 100μs.

ABSOLUTE MAXIMUM RATINGS

PARAMETER		RATING	UNIT
V_{CC}	Power supply voltage	-0.5 to +7	Vdc
V_{IN}	Input voltage	-0.5 to +5.5	Vdc
V_O	Off-state output voltage	-0.5 to +5.5	Vdc
T_A	Operating temperature range	-55 to +125	°C
T_{stg}	Storage temperature range	-65 to +150	°C

ADDRESS PROGRAMMING PULSE

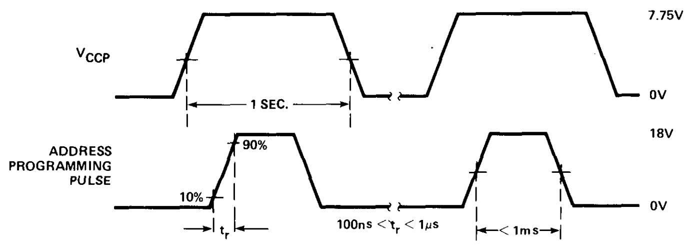


Figure 1

PROTECT PROGRAMMING PULSE

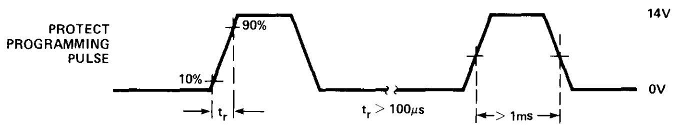
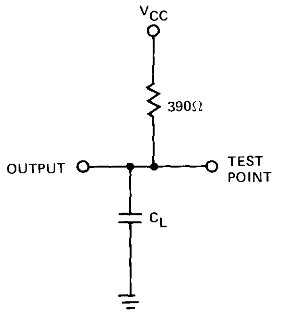


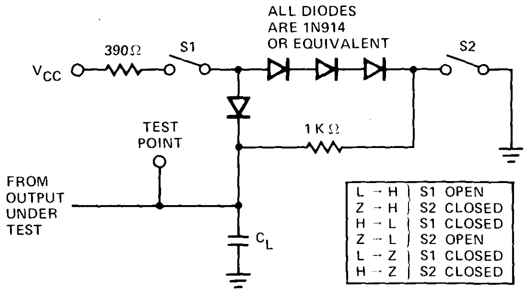
Figure 2

PARAMETER MEASUREMENT INFORMATION

LOAD CIRCUIT FOR OPEN COLLECTOR OUTPUTS

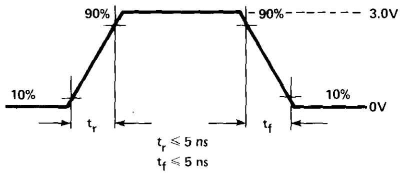


LOAD CIRCUIT FOR TRI-STATE OUTPUTS

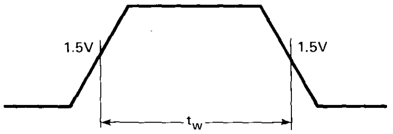


NOTE: C_L includes fixture capacitance.

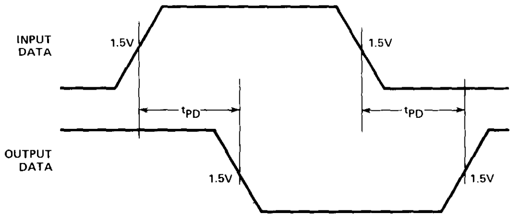
INPUT WAVEFORM



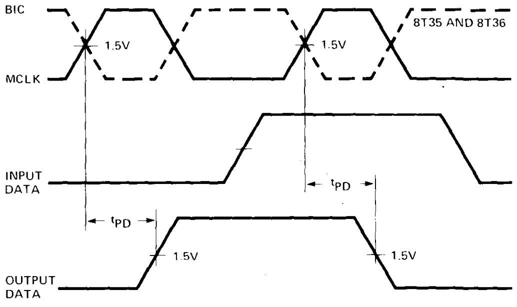
CLOCK PULSE WIDTH



DATA DELAY TIMES
Input Data Reference

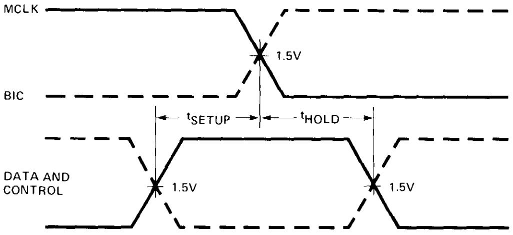


DATA DELAY TIMES
Clock Referenced

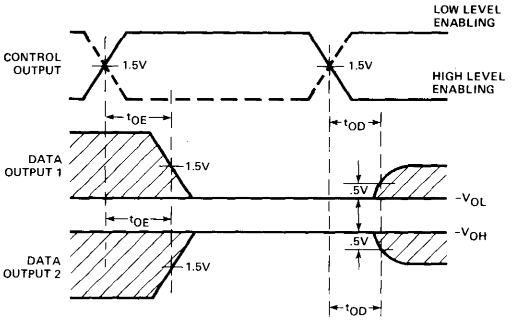


PARAMETER MEASUREMENT INFORMATION (Cont'd)

SETUP AND HOLD TIMES



OUTPUT ENABLE AND DISABLE TIMES
(Tri-State Outputs)



WAVEFORM #1 IS FOR AN OUTPUT WITH INTERNAL CONDITIONS SUCH THAT THE OUTPUT IS LOW WHEN THE TRI-STATE DRIVER IS ENABLED. WAVEFORM #2 IS FOR THE OPPOSITE CONDITION.

APPLICATIONS

Figure 3 shows some of the various ways to use the IV Byte in a system. By controlling the BIC and BOC lines, the Bytes may be used for the input and output of data, control, and status signals. IV Byte 1 functions bidirectionally for data transfer and IV Byte 2 provides a similar function for discrete status and control lines. IV Bytes 3 and 4 serve as dedicated output and input ports, respectively.

IV BYTE APPLICATIONS

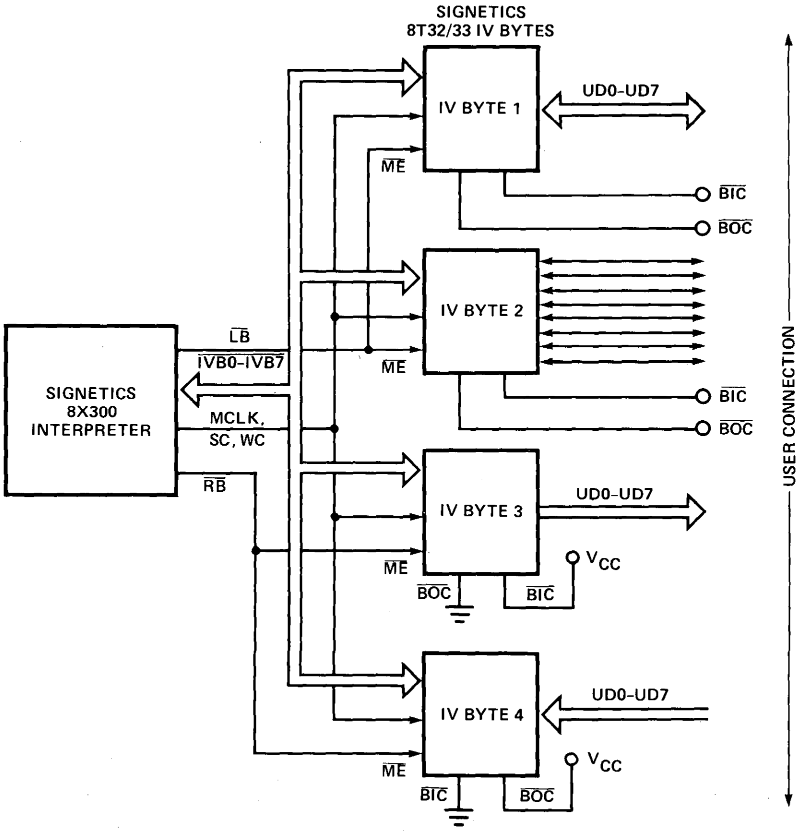


Figure 3

BUS EXPANDER — 8T39

8T39-I,XL

DESCRIPTION

The Bus Expander is specifically designed to increase the I/O capability of 8X300 systems previously limited by fanout considerations. The bus expander serves as a buffer between the 8X300 and blocks of I/O devices. Each bus expander can buffer a block of 16 I/O ports while only adding a single load to the 8X300.

FEATURES

- **15ns propagation delay**
- **Bidirectional**
- **Three-state outputs on both ports**
- **Pre-programmed address range**

APPLICATIONS

The 8T39 Bus Expander is designed to be used with the 8X300 microprocessor to allow increased I/O capability in those systems previously limited by fanout considerations. Figure 1 shows a typical arrangement of the bus expander in an 8X300 system. Each expander services I/O ports whose address is within the range of the expander. Other I/O ports or working storage may be directly connected to the bus as shown.

The bus expander is not limited to use with the 8X300, but may be applied in any system which uses a combined address/data bus.

8T39 ADDRESSING

During normal operation of the 8X300 when an I/O port address is being sent on the IV Bus (SC is high), the I/O port will examine all eight bits of the IV Bus for an address compare. Since the 8T39 is used to buffer blocks of I/O ports, only the four most significant bits are examined by the 8T39 for an address compare.

Note that redundant addresses are not programmed into separate devices. Rather, a discrete device (such as the 8T39-03) may be wired for any address requiring two 1 bits and two 0 bits in the address. The various address ranges for this same device are obtained by permuting the high order (DI0 and DO0 are MSB) data lines accordingly. Both input and output lines must be redefined in order to maintain data and address integrity on the extended bus. Table 1 summarizes the 8T39 addressing.

ORDERING INFORMATION

The Bus Expander is ordered by specifying the following part number:

```
N8T39-XXP
 │   └──P = { I  - Ceramic Package
 │            XL - Epoxy Package
 └── Address Range As Determined
     From Table 1
```

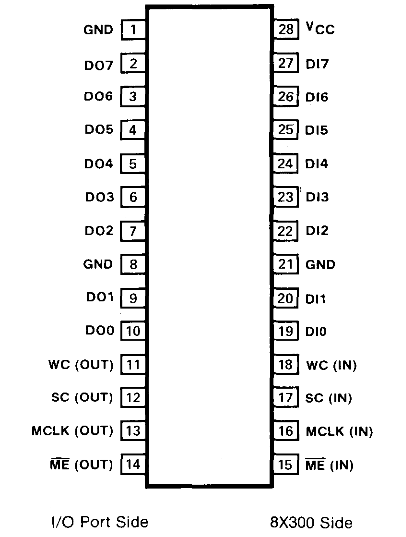
Address functions are specified with the convention that bit 0 is the MSB and bit 7 is the LSB. The DI bus address decoding is active low.

FUNCTIONAL DESCRIPTION

The Bus Expander contains eight sets of non-inverting bidirectional tri-state drivers for the bus data bits, four non-inverting unidirectional drivers for I/O port control, and necessary control logic. The control logic is required to maintain the proper directional transfer of bus data as dictated by the states of the I/O port control signals and the currently enabled I/O port. Each bus expander is programmed during manufacturing to respond to a specific block of I/O port addresses. Only I/O ports with addresses in the range of a given bus expander may be connected to that expander. A bus expander may be used on either left bank or right bank. Multiple expanders on the same bank must have different address ranges; however, expanders with the same address range can be connected if they are on different banks. Systems may be configured with I/O ports connected directly to the 8X300, as well as I/O ports connected through a bus expander; however, no unbuffered I/O port may have an address within the span of a bus expander on the same bank.

PIN CONFIGURATION

I,XL PACKAGE



PIN DESIGNATION

PIN NO.	SYMBOL	NAME & FUNCTION	TYPE
2-7,9,10	DO0-DO7	I/O port data bus	Active low, three-state
11	WC(OUT)	Write control output	Active high
12	SC(OUT)	Select control output	Active high
13	MCLK(OUT)	Master clock output	Acitve high
14	$\overline{ME}$(OUT)	Master enable output	Active low
15	$\overline{ME}$(IN)	Master enable input	Active low
16	MCLK	Master clock input	Active high
17	SC(IN)	Select control input	Active high
18	WC(IN)	Write control input	Active high
19,20,22-27	DI0-DI7	Microcontroller data bus	Active low, three-state
1,8,21	GND	Ground	Active low, three-state
28	VCC	+5 volt supply	

ABSOLUTE MAXIMUM RATINGS

PARAMETER		RATING	UNIT
V_{CC}	Power supply voltage	-0.5 to +7	Vdc
V_{IN}	Input voltage	-0.5 to +5.5	Vdc
V_O	Off-state output voltage	-0.5 to +5.5	Vdc
T_A	Operating temperature range	0 to +70	°C
T_{stg}	Storage temperature range	-65 to +150	°C

Addition of bus expanders may impact system cycle time due to the added delay in the data path. For the purposes of calculating allowable cycle time as described in the 8X300 data sheet, the bus expander delays may be considered additive to the I/O port delays so that a buffered I/O port simply appears as a slower I/O port.

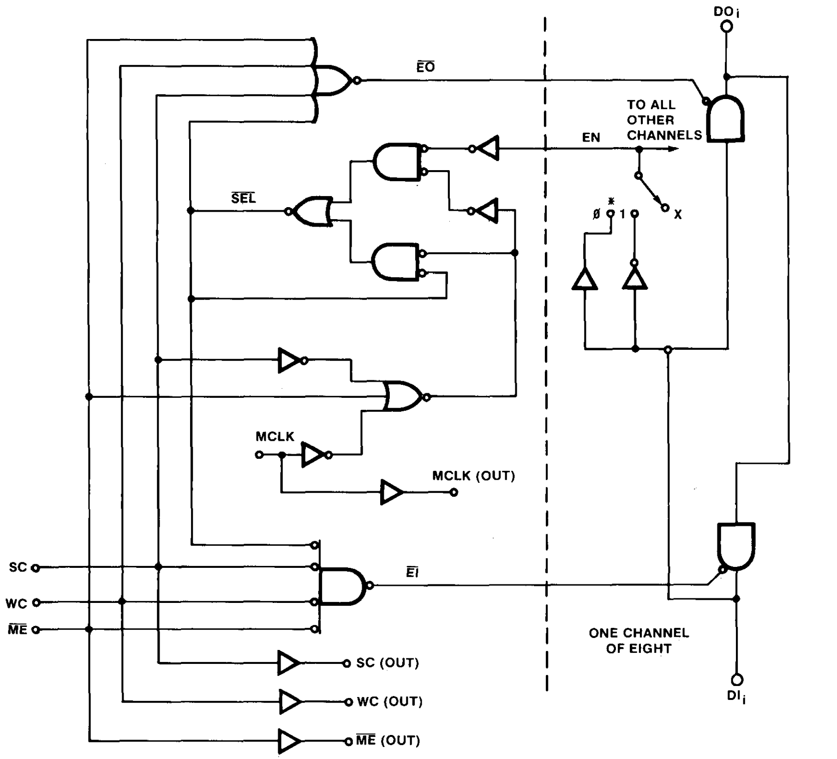
TRUTH TABLE

$\overline{\text{ME}}$	SC	WC	MCLK	SELECT LATCH	DATA TRANSFER DIRECTION	ADDRESS* COMPARISON
L	L	L	X	Set	DI Bus ← DO Bus	No
L	L	L	X	Not set	DI Bus → DO Bus	No
L	L	H	X	X	DI Bus → DO Bus	No
L	H	X	L	X	DI Bus → DO Bus	No
L	H	X	H	X	DI Bus → DO Bus	Yes
H	X	X	X	X	DI Bus → DO Bus	No

NOTES

*When an address comparison is made, the select latch is set if the data on the DI Bus is within the manufactured address range of the IV Bus Expander. Otherwise, the select latch is cleared.

BLOCK DIAGRAM



NOTES

*Selection made during manufacture

X = Don't care

PART TYPE	ADDRESS PATTERN MSB(0) LSB(7)	ADDRESS BLOCKS (Decimal)
8T39-00	0000XXXX	0-15
8T39-01	0001XXXX	16-31, 32-47, 64-79, 128-143
8T39-03	0011XXXX	48-63, 80-95, 144-159, 96-111, 160-175, 192-207
8T39-07	0111XXXX	112-127, 176-191, 208-223, 224-239
8T39-17	1111XXXX	240-255

Table 1 8T39 ADDRESSING SUMMARY

TYPICAL APPLICATION USING 2 BUS EXPANDERS TO GIVE 33 I/O PORTS PLUS WORKING STORAGE

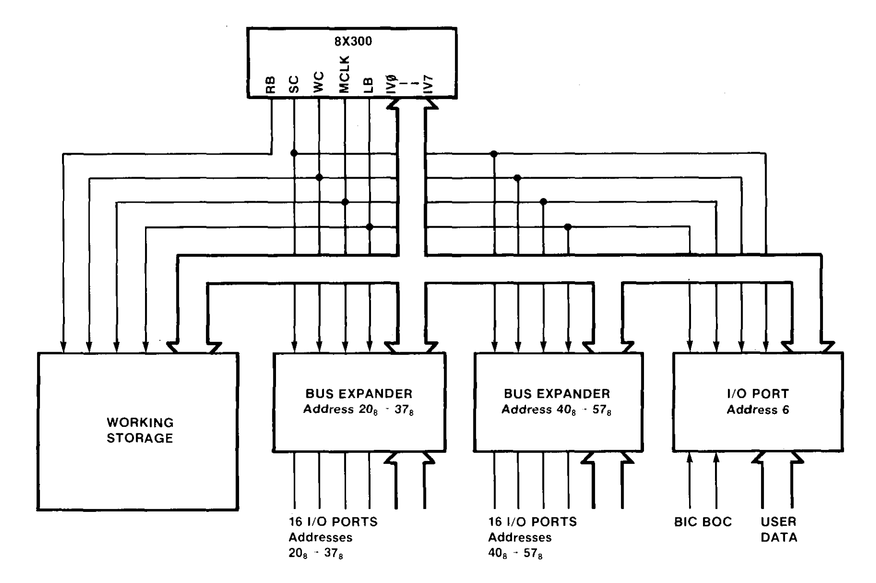
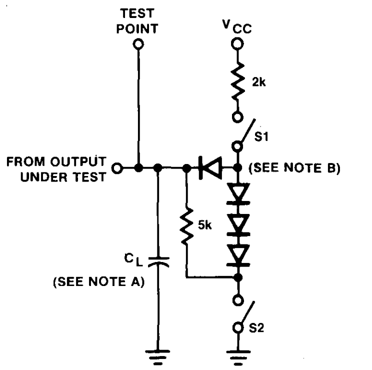


Figure 1

TEST LOAD CIRCUIT



Type for All resistors values are typical and in ohms.

NOTES

A. C_L includes probe and jig capacitance.
B. All diodes are 1N916 or 1N3064.

DC ELECTRICAL CHARACTERISTICS V_{CC} = 5V ± 5%, 0°C ⩽ T_A ⩽70°C

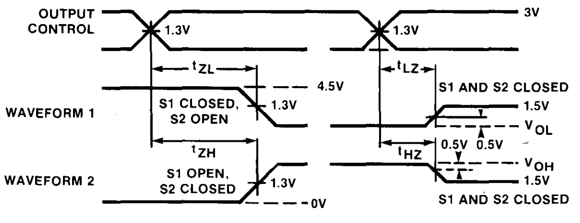
PARAMETER		TEST CONDITIONS	LIMITS			UNIT
			Min	Typ	Max	
	Input voltage					V
V_{IL}	Low				.8	
V_{IH}	High		2.0			
V_{IC}	Clamp				-1	
	Output voltage	V_{CC} = 4.75V				V
V_{OL}	Low	I_{OL} = 16mA			.55	
V_{OH}	High	I_{OH} = -3.2mA	2.4			
	Input current	V_{CC} = 5.25V				uA
I_{IL}	Low	V_{IL} = .5V			-250	
I_{IH}	High	V_{IH} = 5.25V		< 10	100	
I_{OS}	Short circuit output current	V_{CC} = 4.75V	-40			mA
I_{CC}	Supply current	V_{CC} = 5.25V			200	mA

AC ELECTRICAL CHARACTERISTICS V_{CC} = 5V ± 5%, 0°C ⩽ T_A ⩽ 70°C, C_L = 300pF

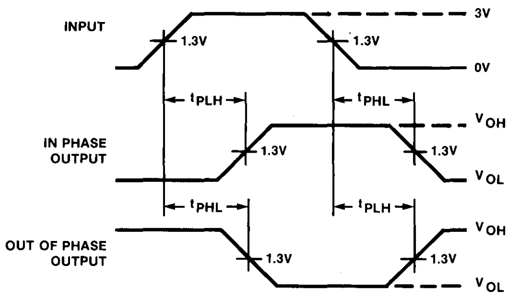
PARAMETER		TO	FROM	TEST CONDITIONS	LIMITS			UNIT
					Min	Typ	Max	
	Path delay	DOX	DIX					ns
tpd	Data	DIX	DOX				15	
tpd	Control	$\overline{ME}$ (out) MCLK (out) SC (out WC (out)	$\overline{ME}$ (in) MCLK (in) SC (in) WC (in)				15	

VOLTAGE WAVEFORMS

CONTROL PATH DELAY
(THREE-STATE OUTPUTS



DATA PATH DELAY TIMES



DESIGNER'S EVALUATION KIT FOR FIXED INSTRUCTION BIPOLAR MICROPROCESSOR 8X300KT100SK

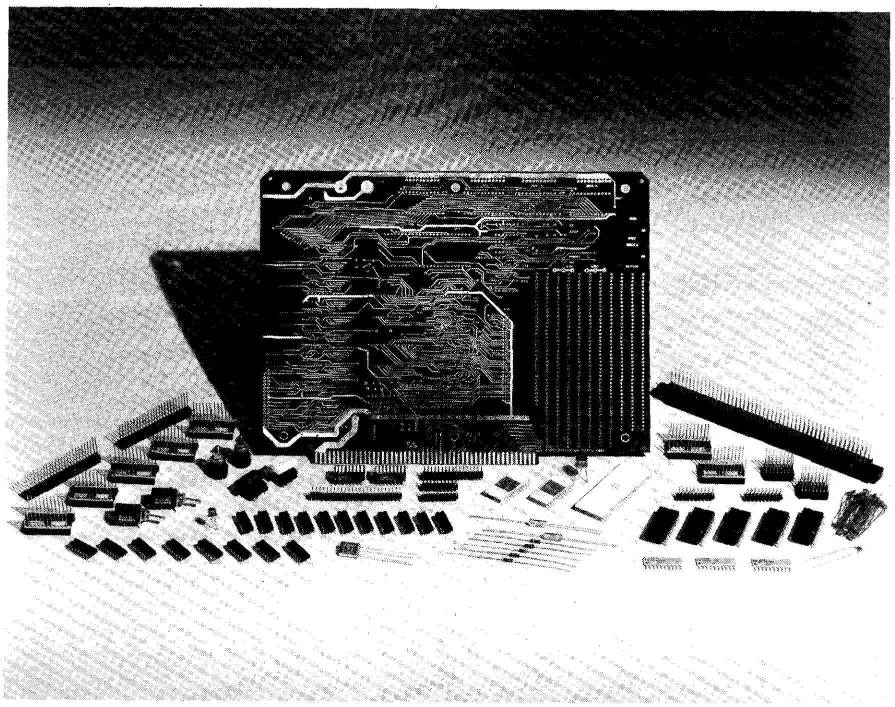
DESCRIPTION

The Signetics 8X300 Fixed Instruction Bipolar Microprocessor provides new levels of high performance to microprocessor applications not previously possible with MOS technology.

In the majority of cases, the choice of a bipolar microprocessor slice, as opposed to an MOS device, is based on speed. The 8X300 processor, combined with high-speed memory and I/O devices, is capable of executing all instruction in 250ns.

The 8X300 is optimized for control and data movement applications. It has a 13-bit address bus for selecting instructions from program storage and a separate input bus for entering a 16-bit instruction words. Data handling and I/O device addressing are accomplished via the 8-bit Interface Vector (IV) bus. The IV bus is supported by four additional control lines and a clock.

The unique features of the 8X300 IV bus and instruction set permit 8-bit parallel data to be rotated or masked before undergoing arithmetic or logical operations. Then, the data may be shifted and merged into any set of from 1 to 8 contiguous bits at the destination. The entire process of input, shifting, processing and output is done in 1 instruction cycle time. The 250ns cycle time makes the 8X300 ideally suited for high-speed applications.



The evaluation board contains all the elements which a designer needs to judge the suitability of the 8X300 for his systems applications. Included with the 8X300 are 4 I/O ports for external device interface, 256 bytes of temporary (working) data storage, and 512 words of program storage, all properly connected to the 8X300 to allow immediate exercising of the board. For this purpose, the PROMs are preprogramed with the I/O control, RAM control, and RAM integrity diagnostic programs. With the remaining PROM space, the designer may enter his own benchmark, test, or development routines.

The board design allows complete flexibility in access to the address, instruction, and IV busses as well as all controls and signals of the 8X300. The IV bus, I/O port user connection, clock signals control lines, address bus and instruction bus are wired to output pins, the board edge connector and flat cable connectors.

The board layout permits variations and/or expansions of the basic design. In addition to the access to all signals for transfer off the board, a wire wrap area is provided so that the designer may add to the board circuitry as he desires. The addition may include memory, additional interfaces, or special circuits which meet specific user requirements.

Controls are also provided for diagnostic and instructional purposes by allowing various operating modes. In the WAIT mode, the program may be single stepped for ease of checkout. The one-shot instruction jamming allows control of the program start location, changes of program flow, changing or examining the internal registers, or testing of simple sequences. The repeated instruction jamming provides a means of repetitive execution of an instruction so that the I/O bus and the control lines may be examined without software changes. In both of these jam cases, the jammed instruction is selected by board-mounted switches.

1 ea— 8T31 (Bidirectional I/O Port)
2 ea— 8T26A (Quad Bus Transceiver)
4 ea— 74157 (Quad 2-Input Data Selector)
2 ea— 7474 (Dual D Flip Flop)
2 ea— 7400 (Quad Nand Gate)
1 ea— 7427 (3-Input NOR Gate)
1 ea— P.C. Board
Misc. Parts
1 ea— Introductory Manual, assembly instructions, code listings and schematics

FEATURES

- **250ns CPU with Crystal**
- **4 I/O Ports (32 Lines)**
- **256 Bytes Data Storage**
- **512 Words Program Storage**
- **Run/Wait Control**
- **Single Step**
- **Instruction Jamming**
  One Shot Instruction Jam
  Repeated Jam
- **All Buses to Output Pins**
- **Firmware Diagnostics**
- **Wire-Wrap Area**
- **Edge Connector**
- **Flat Cable Connectors**
- **Wire Wrap Posts for Bus Lines**

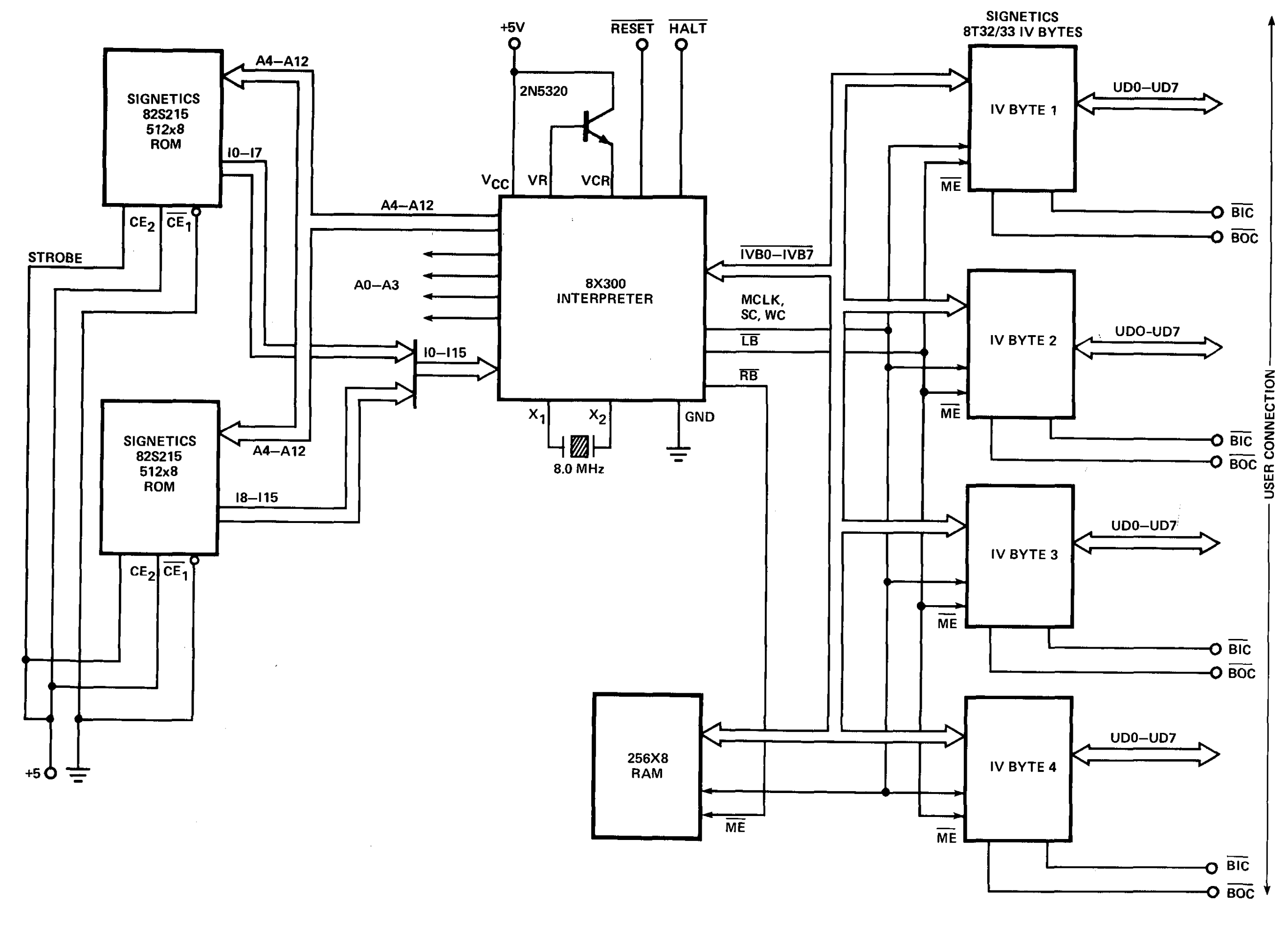
COST: $299 (Total Value = $504)

AVAILABILITY

Immediate delivery from Signetics Rep. or Distributors.

CONTENTS

1 ea— 8X300
8 ea— 82S116 (256 x 1 RAM)
2 ea— 82S115 (512 x 8 PROM)
4 ea— 8T32 Addressable Bidirectional I/O Port



8X300 KIT CONFIGURATION

Auxiliary Circuits

The 8X300 can be used with any bipolar (or TTL-compatible) ICs. It can directly address 8192 program instruction locations and up to 512 I/O ports. The memory paging feature may be employed for larger working storage. Typical auxiliary circuits include:

Program Storage		82S115 (512x8 PROM)
I/O	8T32/33	(8-Bit Synchronous Bidirectional I/O Port)
	8T35/36	(8-Bit Asynchronous Bidirectional I/O Port)
	8T31	(8-Bit Bidirectional I/O Port)
	8T39	(Quad Bus Extender)
Working Storage		82S116 RAM

The Microcontroller Cross Assembly Program (MCCAP) is a program designed to translate symbolic instructions into object code that can be executed by the 8X300. This program will run on most computers that have a FORTRAN compiler with a computer word length of at least 16 bits and a random access I/O capability; it can be run on most minicomputers as well as large scale computers.

The Assembler is written in FORTRAN which provides compatibility with most computer systems and makes it transportable. The program is modular and uses a minimum of memory. However, it may be operated in an overlay mode if necessary.

The Assembler is a 2-pass program that issues helpful error messages and produces an easily read program listing. During the first pass, the labels are examined and placed in a symbol table. Certain errors may be detected during this pass and will be displayed on the output listing.

In the second pass, the object code is completed, symbolic addresses are resolved, and a listing and object module are produced. Certain errors not detected during the first pass may be detected and displayed on the listing.

The Assembler features symbolic addressing, forward references and expression evaluation. It also has the capability to symbolically represent the Interface Vector (IV). In addition, the Assembler is capable of expressing data in several number systems as well as in ASCII character codes.

MCCAP is distributed in 1 of 4 standard forms. These distribution forms are described below.

9-Track EBCDIC Magnetic Tape

This is standard IBM compatible magnetic tape recorded in 9-track format. The tape is recorded with 1 source card image per record.

Density:	800 BPI
Record Length:	80 Bytes
Block Size:	80 Bytes (1 record/block)

9-Track ASCII Magnetic Tape

This form is the same as 9-track EBCDIC tape except that ASCII character codes are used with the parity bit a zero.

7-Track BCD Magnetic Tape

This is standard IBM compatible magnetic tape recorded in 7-track format. The tape is recorded in even parity BCD with 1 source card image per record.

Density:	800 BPI
Record Length:	80 Characters
Block Size:	80 Characters (1 record /block)

029 Punch Cards

Standard 80 column punched cards, punched in 029 (EBCDIC) punch codes.

CHAPTER 3
STANDARD SUPPORT CIRCUITS

INTRODUCTION

In addition to the dedicated support circuits available for the various Signetics microprocessors, Signetics offers a complete line of standard circuits to complete the design of a microcomputer system.

A complete line of Schottky-clamped TTL, read/write memory arrays is offered. All feature open collector or tri-state output options for optimization of word expansion in bused organizations. Memory expansion is further enhanced by full on-chip address decoding, chip enable function and PNP input transistors which reduce input loading requirements. All devices offer high performance read access and write cycle times making these devices ideally suited in high speed memory applications such as "cache," buffers, scratch pads, writable control store, main store, etc.

Signetics offers the industry's broadest line of bipolar high performance ROMs, PROMs and FPLAs. The PROMs and FPLAs are field programmable, which means that custom patterns are immediately available by following the provided fusing procedures. Signetics PROMs are supplied with all outputs at logical 0. Outputs are programmed to a logic 1 at any specified address by fusing a Ni-Cr link matrix. All bipolar ROMs, PROMs and FPLAs are fully TTL compatible, and include on-chip decoding and chip enable functions for ease of memory expansion. Tri-state and open collector output functions are available, and low input currents reduce input buffer requirements. Most Signetics PROMs and FPLAs also have pin and performance compatible ROMs and PLAs, offering the user the ultimate in flexibility and cost reduction.

Signetics n-channel MOS products include a complete family of 1K static RAMs and 8K static ROMs. These feature TTL compatible inputs and outputs, and require only a single +5V power supply. A variety of 4K dynamic RAMs is also available for system configurations requiring large amounts of read/write memory.

The 8T series of interface devices includes display drivers, bus drivers, input/output ports, level converters, and special purpose devices. A complete line of standard and low power Schottky (LS) 74 series devices is available in addition to the 8200 series of MSI devices. Many devices from the Signetics analog product line are also suitable for use in microcomputer systems. These include voltage regulators, operational amplifiers, comparators and timers.

This chapter provides product line summaries and data for a representative selection of Signetics standard support circuits. Further information can be found in the Signetics Data Manual.

BIPOLAR MEMORIES PRODUCT SUMMARY

DEVICE	DESCRIPTION	CONFIGURATION	OUTPUT**	MAXIMUM TAA (ns)	TEMPERATURE RANGE*
CAMS					
8220	8-Bit CAM	4 x 2	OC	40	C
10155	16-Bit CAM	8 x 2	OE	13	C
SAMS					
82S12	32-Bit SAM	8 x 4	OC	35	C
82S112	32-Bit SAM	8 x 4	TS	35	C
RAMS					
82S21	64-Bit RAM	32 x 2	OC	50	C
82S25	64-Bit RAM	16 x 4	OC	50	M,C
54/74S89	64-Bit RAM	16 x 4	OC	50	M,C
54/74S189	64-Bit RAM	16 x 4	TS	35	M,C
3101A	64-Bit RAM	16 x 4	OC	35	M,C
82S16	256-Bit RAM	256 x 1	TS	50	M,C
82S17	256-Bit RAM	256 x 1	OC	50	M,C
82S116	256-Bit RAM	256 x 1	TS	40	C
82S117	256-Bit RAM	256 x 1	OC	40	C
54/74S200	256-Bit RAM	256 x 1	TS	50	M,C
54/74S201	256-Bit RAM	256 x 1	TS	50	M,C
54/74S301	256-Bit RAM	256 x 1	OC	50	M,C
82S09	576-Bit RAM	64 x 9	OC	45	M,C
10144	256-Bit RAM	256 x 1	OE	30	C
82S10	1024-Bit RAM	1024 x 1	OC	45	M,C
82S11	1024-Bit RAM	1024 x 1	TS	45	M,C
93415A	1024-Bit RAM	1024 x 1	OC	45	M,C
93425A	1024-Bit RAM	1024 x 1	TS	45	M,C
ROMS					
82S226	1024-Bit ROM	256 x 4	OC	50	M,C
82S229	1024-Bit ROM	256 x 4	TS	50	M,C
82S230	2048-Bit ROM	512 x 4	OC	50	M,C
82S231	2048-Bit ROM	512 x 4	TS	50	M,C
82S214	2048-Bit ROM	256 x 8	TS	60	M,C
8228	4096-Bit ROM	1024 x 4	TTL	75	C
82S215	4096-Bit ROM	512 x 8	TS	60	M,C
82S280	8096-Bit ROM	1024 x 8	OC	125	M,C
82S281	8096-Bit ROM	1024 x 8	TS	125	M,C
82S290	16,192-Bit ROM	2048 x 8	OC		M,C
82S291	16,192-Bit ROM	2048 x 8	TS		M,C
PROMS					
82S23	256-Bit PROM	32 x 8	OC	50	M,C
82S123	256-Bit PROM	32 x 8	TS	50	M,C
10139	256-Bit PROM	32 x 8	OE	20	C
82S27	1024-Bit PROM	256 x 4	OC	40	C
82S126	1024-Bit PROM	256 x 4	OC	50	M,C
82S129	1024-Bit PROM	256 x 4	TS	50	M,C
10149	1024-Bit PROM	256 x 4	OE	17	C
82S114	2048-Bit PROM	256 x 8	TS	60	M,C
82S130	2048-Bit PROM	512 x 4	OC	50	M,C
82S131	2048-Bit PROM	512 x 4	TS	50	M,C
82S115	4096-Bit PROM	512 x 8	TS	60	M,C
82S136	4096-Bit PROM	1024 x 4	OC	60	M,C
82S137	4096-Bit PROM	1024 x 4	TS	60	M,C
82S184	8192-Bit ROM	2048 x 4	OC	100	M,C
82S185	8192-Bit ROM	2048 x 4	TS	100	M,C
FPLA					
82S100	FPLA	16 x 48 x 8	TS	50	M,C
82S101	FPLA	16 x 48 x 8	OC	50	M,C

*TEMPERATURE RANGE
C = Commercial (0°C to +75°C)
M = Military (-55°C to +125°C)
All ECL 10,000 (-30°C to +85°C)

**OUTPUT
TS = Tri-State
OC = Open Collector
OE = Open Emitter

INTERFACE

	LOGIC			
DEVICE	DESCRIPTION	Commercial	Military	Data Book Page Ref.
8T04	7-Segment Decoder/Driver	●	●	17
8T05	7-Segment Decoder/Transistor Driver	●	●	20
8T06	7-Segment Decoder/Display Driver	●	●	23
8T09	Tri-State Quad Bus Driver	●	●	26
8T10	Tri-State Quad D-Type Bus Flip-Flop	●	●	29
8T13	Dual Line Driver	●	●	33
8T14	Triple Line Receiver with Hysteresis	●	●	35
8T15	Dual Communications EIA/MIL Line Driver	●		38
8T16	Dual Communications EIA/MIL Line Receiver	●		40
8T18	Dual 2-Input NAND Gate (High Voltage to TTL Interface)	●	●	43
8T20	Bidirectional One Shot	●		45
8T22	Retriggerable One Shot	●		49
8T23	Dual Line Driver for IBM 360/370 Interface (75123)	●		51
8T24	Triple Line Receiver for IBM 360/370 Interface (75124)	●		53
8T25	Tri-State Dual MOS Sense Amplifier/Latch	●		56
8T26A	Tri-State Quad Bus Receiver	●		59
8T27	Quad Inverting Bus Driver/Receiver	DEV		N/A
8T28	Tri-State Quad Bus Receiver	●		59
8T30	Dual TTL/DTL to MOS Transceiver/Port Controller	●		62
8T31	8-Bit Bidirectional I/O Port	●		71
8T32	Interface Vector (IV) Byte	●		71
8T33	Interface Vector (IV) Byte	●		71
8T34	Quad Bus Transceiver (DM8834) (Tri-State Outputs)	●		78
8T35	Asynch. Programmable 8-Bit I/O Port, o/c	DEV	●	N/A
8T36	Asynch. Programmable 8-Bit I/O Port (Tri-State)	●		N/A
8T37	Hex Bus Receiver with Hysteresis—Schmitt Trigger	●	●	80
8T38	Quad Bus Transceiver (Open Collector) (DM8838)	●		82
8T80	Quad 2-Input NAND Interface Gate	●	●	83
8T90	Hex Inverter Interface Element	●	●	84
8T93	High Speed Hex Inverter (PNP Inputs)	●		85
8T94	High Speed Hex Inverter (Open Collector) (PNP Inputs)	●		86
8T95	High Speed Hex Buffers/Inverters (74365/DM8095)	●		86
8T96	High Speed Hex Buffers/Inverters (74366/DM8096)	●		87
8T97	High Speed Hex Buffers/Inverters (74367/DM8097)	●		87
8T98	High Speed Hex Buffers/Inverters (74368/DM8098)	●		87
8T363	Dual Zero Crossing Detector	●		94
8T380	Quad Bus Receiver with Hysteresis—Schmitt Trigger	●		97

INTERFACE

ANALOG				
DEVICE	DESCRIPTION	Commercial	Military	Data Book Page Ref.
	PERIPHERAL INTERFACE			
75450B	Dual Peripheral Driver	●		121
75451B	Dual Peripheral Driver	●		126
75452B	Dual Peripheral Driver	●		128
75453B	Dual Peripheral Driver	●		130
75454B	Dual Peripheral Driver	●		132
MC1488	Quad Line Driver	●		110
MC1489/1489A	Quad Line Receiver	●		112
75S107	Dual Line Receiver	●		114
74S108	Dual Line Receiver	●		117
DM7820/8820	Dual Differential Line Receiver	●	●	103
DM7820A/8820A	Dual Differential Line Receiver	●	●	N/A
DM7830/8830	Dual Differential Line Driver	●	●	105
	INTERFACE DISPLAY			
DM8880	Display Decoder Driver	●		107
NE584/585	Gas Discharge Segment & Digit Driver	● ●		101
NE582	LED Digit Driver	●		99
	MEMORY INTERFACE			
3207A	MOS Clock Driver	●		134
3207A-1	MOS Clock Driver	●		136
7520	Dual Core Memory Sense Amp	●		144
7521	Dual Core Memory Sense Amp	●		144
7522	Dual Core Memory Sense Amp	●		144
7523	Dual Core Memory Sense Amp	●		144
7524	Dual Core Memory Sense Amp	●		144
7525	Dual Core Memory Sense Amp	●		144
75S207	Dual MOS Sense Amp	●		138
75S208	Dual MOS Sense Amp	●		141
75315	Core Memory Driver	●		N/A
75324	Core Memory Driver	●		151
75325/55325	Memory Driver	●		156
75361A	MOS Clock Driver	●		170

DC ELECTRICAL CHARACTERISTICS

	INPUT VOLTAGE															OUTPUT VOLTAGE					
PARAMETER	V_{IL} (V) LOW LEVEL			V_{IH} (V) HIGH LEVEL			V_{IC} CLAMP VOLTAGE			VOLTAGE RATING			VT_L (mV)[20] LOW LEVEL THRESHOLD VOLTAGE			VT_H (mV)[20] HIGH LEVEL THRESHOLD VOLTAGE			VOL (V)[7] LOW LEVEL		
TEST CONDITIONS							V_{CC} = MIN I_{IN} = -12mA			V_{IN} = 10mA			V_{CC} = MIN V_{IN} = 0.8V I_{OL} = -400μA			V_{CC} = MAX V_{IN} = 0.8V I_{OH} = 16mA			V_{CC} = MIN		
	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max
8T26A		N/A			N/A				-1.0		N/A		0.85					2		Driver I_{OL}=48mA Receiver I_{OL}=20mA	0.5 0.5
8T28		N/A			N/A				-1.0		N/A		0.85					2		Driver I_{OL}=48mA Receiver I_{OL}=20mA	0.5 0.5
8T31		N/A			N/A		I_{IN} = -5mA		-1		N/A			N/A			N/A			I_{OL}=20mA	0.55

DC ELECTRICAL CHARACTERISTICS (Cont'd)

				INPUT CURRENT						OUTPUT CURRENT						POWER SUPPLY						
PARAMETER	V_{OH} (V) HIGH LEVEL			I_{IL} (mA) LOW LEVEL			I_{IH} (μA) HIGH LEVEL			I_{CBO} (μA) LEAKAGE CURRENT			I_{OS} SHORT CIRCUIT CURRENT			I_{CC} POWER/CURRENT CONSUMPTION (mW/mA)				I_{CC} (mA) V_{IN} = 2.0V		
TEST CONDITIONS	V_{CC} = MIN I_{OH} = -160μA			V_{CC} = MAX V_{IN} = 0.4V			V_{CC} = MAX V_{IN} = 4.5V			V_{IN} = 2.0V			V_{CC} = MAX V_{IN} = 0V V_{OUT} = 0V			V_{CC} = MAX V_{IN} = 0V MILITARY		COMMERCIAL		V_{CC} = MAX		
	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Typ	Max	Typ	Max	Min	Typ	Max
8T26A		Driver			Driver			Driver/Receiver			HIGH Level			Driver		N/A			457/87		N/A	
		I_{OL} = -10mA			LOW Level				25		V_{OUT} = 2.4V		50		-150							
	2.4					-200						100		Receiver								
		Receiver			LOW Level (Disabled)						LOW Level		-30		-75							
		I_{OL} = -100μA				-25					V_{OUT} = 0.5V											
	3.5				Receiver							-100										
		I_{OL} = -2.0mA				-200					HIGH Level			Driver		N/A			578/100		N/A	
	2.4										V_{OUT} = 2.4V		-50		-150							
8T28		Driver			Driver			Driver/Receiver				100		Receiver								
		I_{OL} = -10mA			LOW Level				25		LOW Level		-30		-75							
	2.4					-200					V_{OUT} = 0.5V											
		Receiver			LOW Level (Disabled)							-100										
		I_{OL} = -100μA				-25																
	3.5				Receiver					-10			-20		-200	N/A		N/A				150
		I_{OL} = -2.0mA				-200																
	2.4										N/A		10	UD Bus		N/A		N/A			100	150
8T31		I_{OL} = 3.2mA			V_{IN} = 0.55V			V_{IN} = 5.5V						IV Bus								
	2.4					-500		100					20									

NOTES

1. All voltage measurements are referenced to the ground terminal. Terminals not specifically referenced are left electrically open.
2. All measurements are taken with ground pin tied to zero volts.
3. Positive current is defined as into the terminal referenced.
4. Precautionary measures should be taken to insure current limiting in accordance with absolute maximum ratings.
5. Measurements apply to each gate element independently.
6. Output source current is supplied through a resistor to ground.
7. Output sink current is supplied through a resistor to V_{CC}.
8. Connect an external 1K 1% resistor to the output for this test.
9. Not more than one output should be shorted at one time.
10. Previous condition is a high level output state.
11. Previous condition is a low level output state.
12. Test each driver separately.
13. For more electrical specifications see data sheet.
14. I_{CC} is dependent upon loading. I_{CC} limit specified is for no-load test condition for both drivers.
15. With forced output current of 240μA, the output voltage must not exceed 0.15V.

DUAL COMMUNICATIONS EIA/MIL LINE DRIVER 8T15

8T15 A,F

DESCRIPTION

The 8T15 Dual Communications Line Driver provides line driving capability for data transmission between Data Communication and Terminal Equipment. The device meets or exceeds the requirements of EIA Standard RS-232B and C, MIL STD-188B and CCITT V 24.

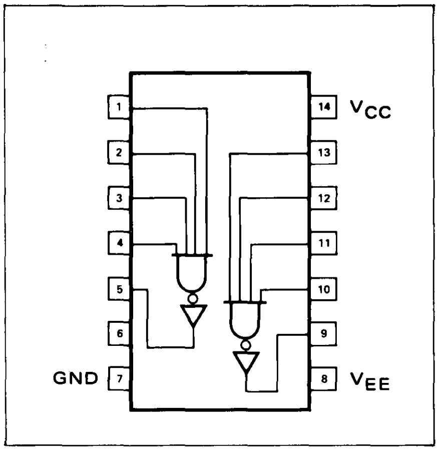
This dual 4-input NAND driver will accept standard TTL logic level inputs and will drive interface lines with nominal data levels of +6V and −6V. Output slew rate may be adjusted by attaching an external capacitor from the output terminal to ground. The outputs are protected against damage caused by accidental shorting to as high as ±25V.

ABSOLUTE MAXIMUM RATINGS*

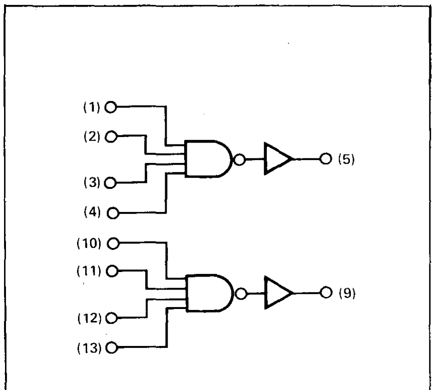
Input Voltage	+5.5V
Output Voltage	±25V
V_{CC}	+15V
V_{EE}	−15V
Storage Temperature	−65°C to +150°C
Operating Temperature	0°C to +75°C

* Limiting values above which serviceability may be impaired.

PIN CONFIGURATION



LOGIC DIAGRAM



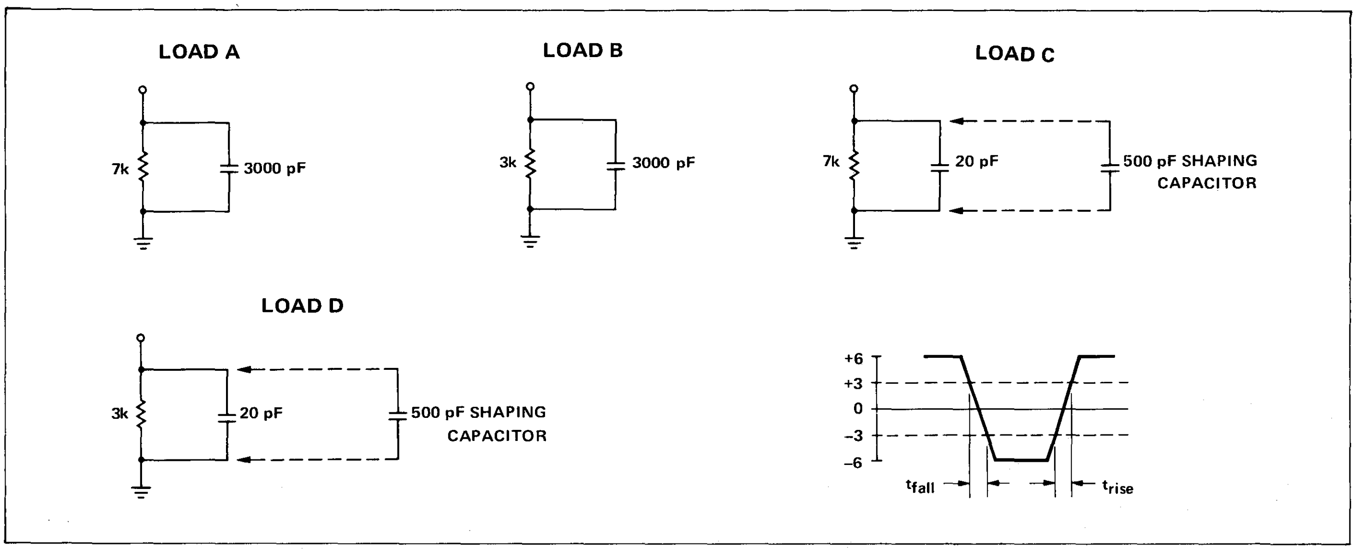
VEE = (8) GND = (7)
VCC = (14) ( ) = Denotes Pin Numbers

T_A = 25°C, V_{CC} = +12.0V, V_{EE} = −12.0V

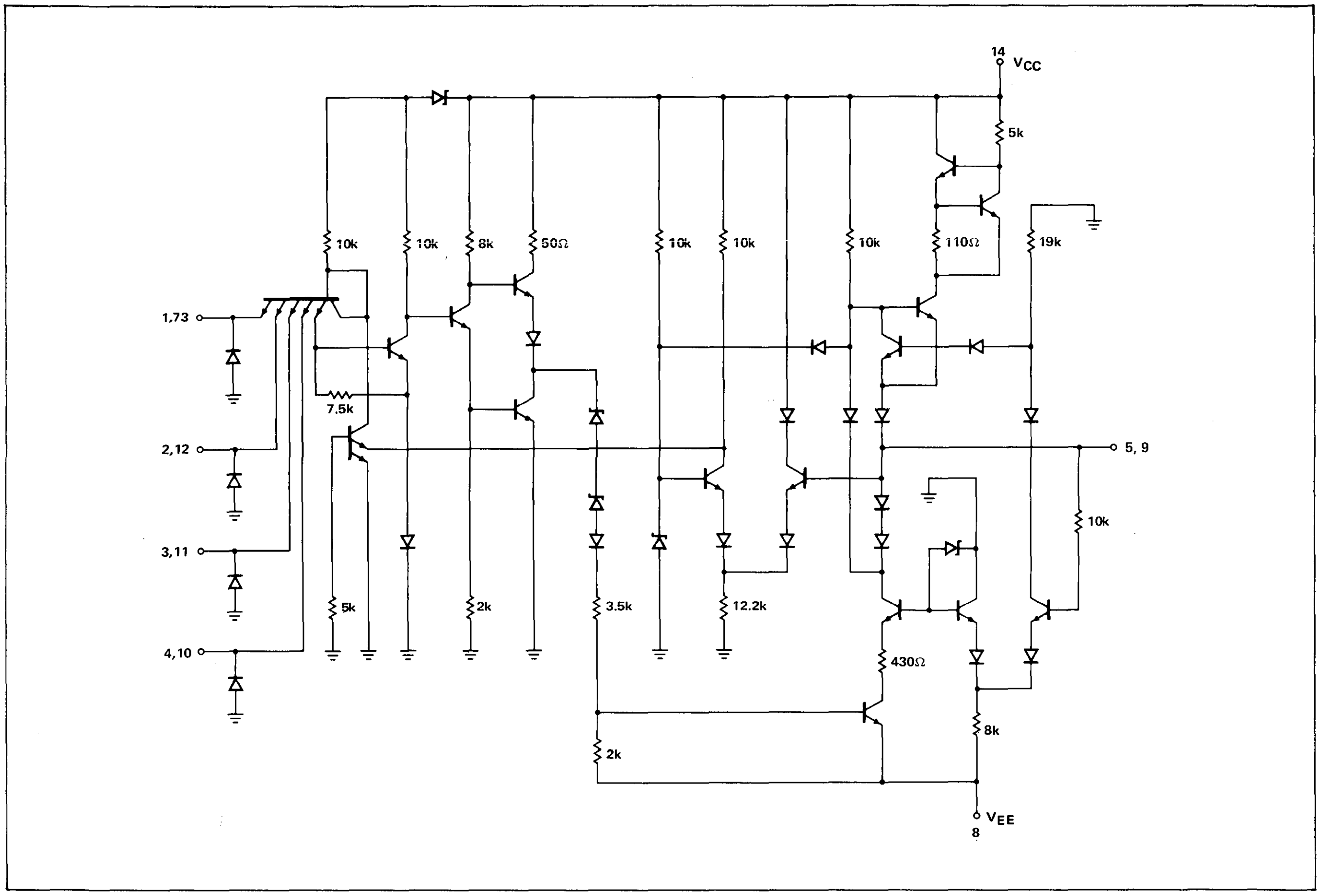
CHARACTERISTICS	TEST CONDITIONS			UNIT	LIMITS		
	INPUTS		OUTPUTS		MIN	TYP	MAX
	DRIVEN	OTHER					
Output Rise Time[1]			Load A	μs			4
Output Fall Time[1]			Load B	μs			4
Output Rise Time[1]			Load C	ns	200		
Output Fall Time[1]			Load D	ns	200		
Current from Positive Supply[2]				mA			16
Current from Negative Supply[2]				mA			28
Output Impedance							
(Power on)	0.0V		−3.5± 1mA	ohms		95	
(Power on)	2.0V		+3.5± 1mA	ohms		95	
(Power off)			±2V	ohms	300	2.5M	

[1] Rise and fall times are measured between the +3V and −3V points on the output waveform.
[2] V_{CC} = +12.6V, V_{EE} = −12.6V

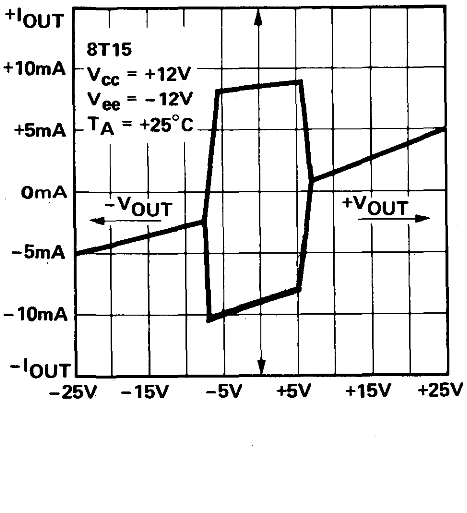
AC TEST FIGURES & WAVEFORMS



SCHEMATIC DIAGRAM

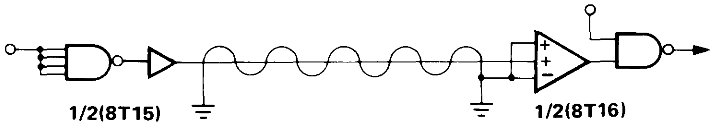


TYPICAL OUTPUT CHARACTERISTIC CURVE

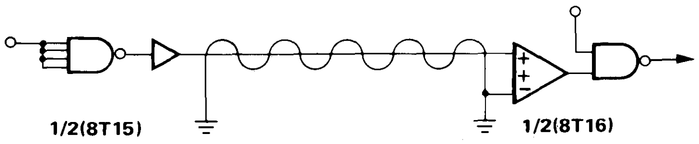


TYPICAL APPLICATIONS

HIGH DIFFERENTIAL NOISE IMMUNITY (EIA + INPUT)



HIGH COMMON MODE NOISE IMMUNITY (MIL + INPUT)



DUAL COMMUNICATIONS EIA/MIL LINE RECEIVER WITH HYSTERESIS 8T16

8T16 A,F

DESCRIPTION

The 8T16 Dual Communications Line Receiver provides receiving capability for data lines between Data Communication and Terminal Equipment. The device meets or exceeds the requirements of EIA Standard RS-232B and C, MIL-STD-188B and CCITT V24 and operates from a single 5 volt power supply.

The receivers accept single (EIA) or double ended (MIL) inputs and are provided with an output strobing control. Both EIA and MIL input standards are accommodated.

When using the EIA input terminal (with the Hysteresis terminal open), input voltage threshold levels are typically +2V and −2V with a guaranteed minimum Hysteresis of 2.4V. By grounding the "Hysteresis" terminal, the EIA input voltage threshold levels may be shifted to typically +1.0V and +2.1V with a minimum guaranteed Hysteresis of 0.75V. (Note that when using the EIA inputs, the MIL inputs — both positive and negative — must be grounded).

The MIL input voltage threshold levels are typically +0.6V and −0.6V with a minimum guaranteed Hysteresis of 0.7V. A MIL negative terminal is provided on each receiver per specification MIL-STD-188B to provide for common mode noise rejection.

Each receiver includes a strobe input so that:

a. A "1" on the strobe input allows data transfer.
b. A "0" on the strobe input holds the output high.

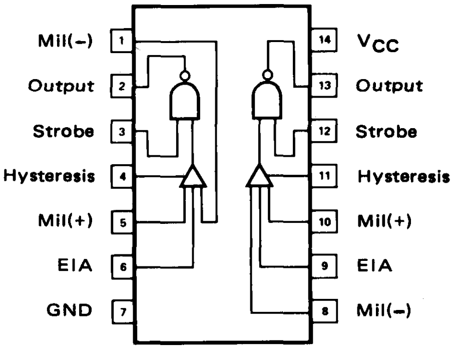
ABSOLUTE MAXIMUM RATINGS*

Input Voltage (EIA and MIL)	±25V
V_{CC}	+7.0V
Storage Temperature	−65°C to +175° C
Operating Temperature	0° C to +75° C

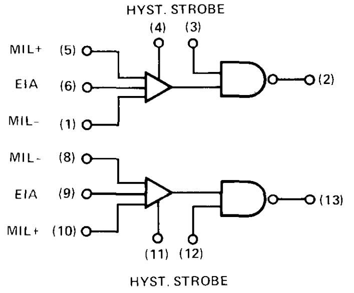
* Limiting values above which serviceability may be impaired.

PIN CONFIGURATION

A,F PACKAGE

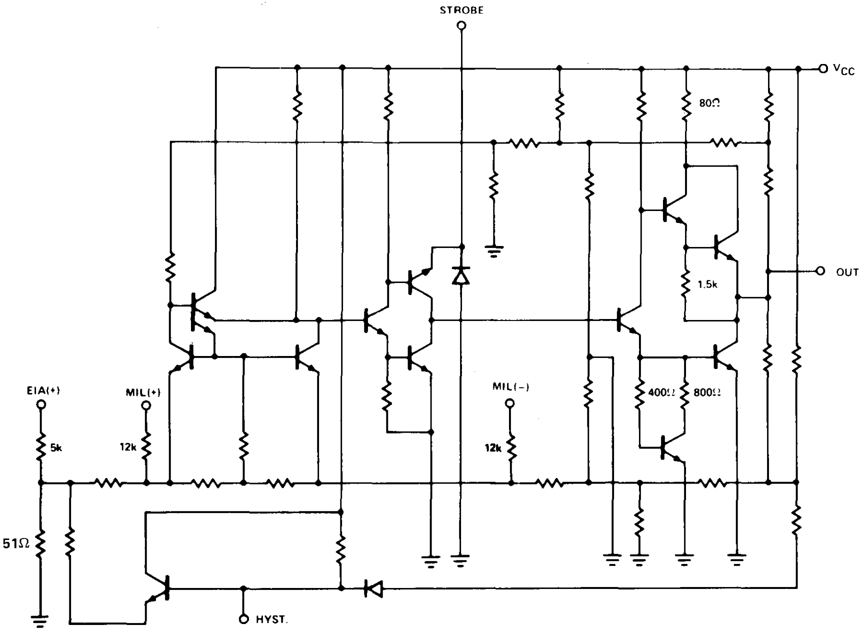


LOGIC DIAGRAM



V_{CC} = (14)
GND = (7)
( ) = Denotes Pin Numbers

SCHEMATIC DIAGRAM



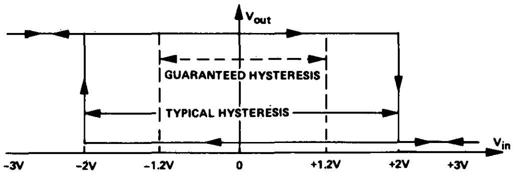
T_A = 25° C and V_{CC} = 5.0V

PARAMETER	TEST CONDITIONS				LIMITS			UNITS
	INPUTS				MIN	TYP	MAX	
	EIA	MIL(+)	MIL(−)	STROBE				
Input Resistance (EIA)	±25V	0.0V	0.0V		3	5	7	kΩ
Input Resistance (MIL)	0.0V	±25V	0.0V		7.5	11.4		kΩ
Propagation Delay				5.00V		100	150	ns
Signal Switching Acceptance				5.00V	20			kHz

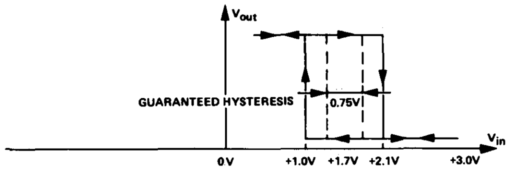
1. This test guarantees transfer of signals of up to 20kHz. Connect 1000pF between the output terminal and ground.

HYSTERESIS CURVES

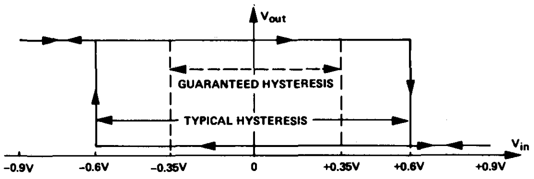
EIA — "HYSTERESIS" OPEN



EIA — "HYSTERESIS" GROUNDED

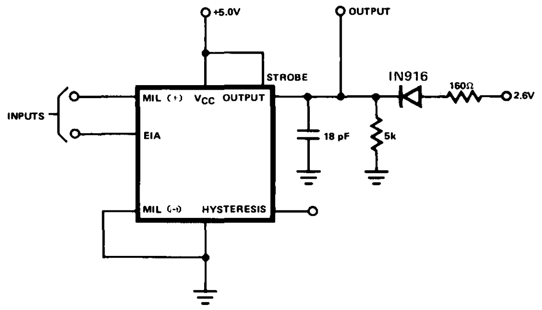


MIL — HYSTERESIS

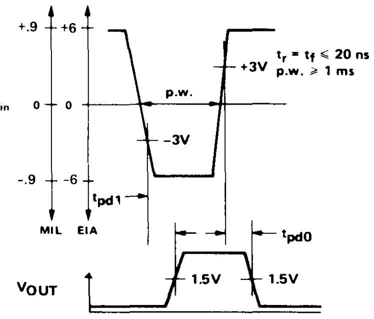


* V_{in} IS REFERENCED TO THE MIL (-) INPUT TERMINAL

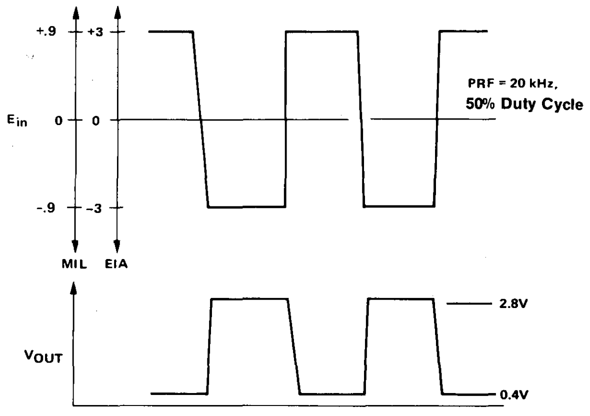
AC TEST FIGURE AND WAVEFORMS



PROPAGATION DELAY

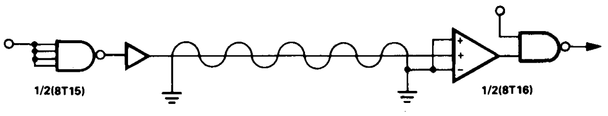


SIGNAL SWITCHING ACCEPTANCE

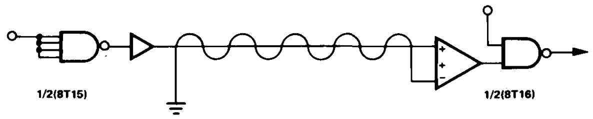


TYPICAL APPLICATIONS

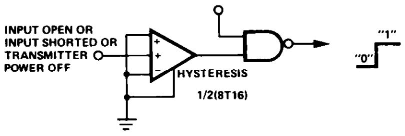
HIGH DIFFERENTIAL NOISE IMMUNITY (EIA + INPUT)



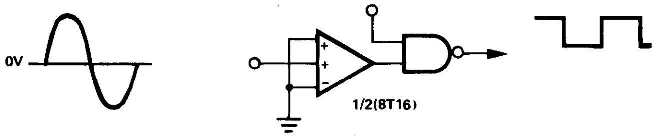
HIGH COMMON MODE NOISE IMMUNITY (MIL + INPUT)



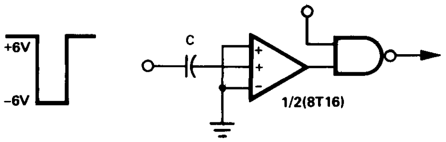
EIA FAIL-SAFE OPERATION



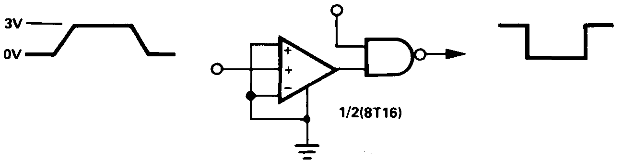
SINE TO SQUARE WAVE CONVERTER



AC COUPLED OPERATIONS



SCHMITT TRIGGER



TRI-STATE QUAD BUS TRANSCEIVER — 8T26A/8T28

8T26A-B,F • 8T28-B,F

DESCRIPTION

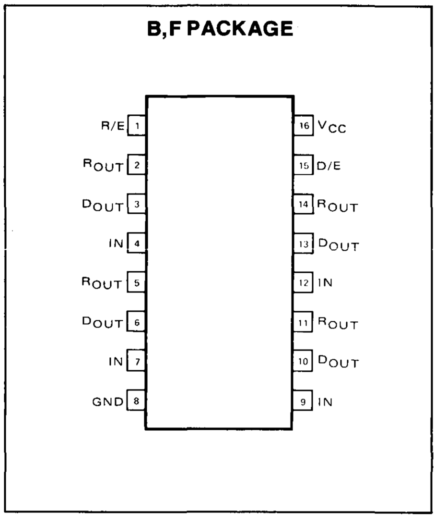
The 8T26A/28 consists of four pairs of Tri-State logic elements configured as Quad Bus Drivers/Receivers along with separate buffered receiver enable and driver enable lines. This single IC Quad Transceiver design distinguishes the 8T26A/28 from conventional multi-IC implementations. In addition, the 8T26/28's ultra high speed while driving heavy bus capacitance (300pF) makes these devices particularly suitable for memory systems and bidirectional data buses.

Both the Driver and Receiver gates have Tri-State outputs and low-current PNP inputs. Tri-State outputs provide the high switching speeds of totempole TTL circuits while offering the bus capability of open collector gates. PNP inputs reduce input loading to 200μA maximum.

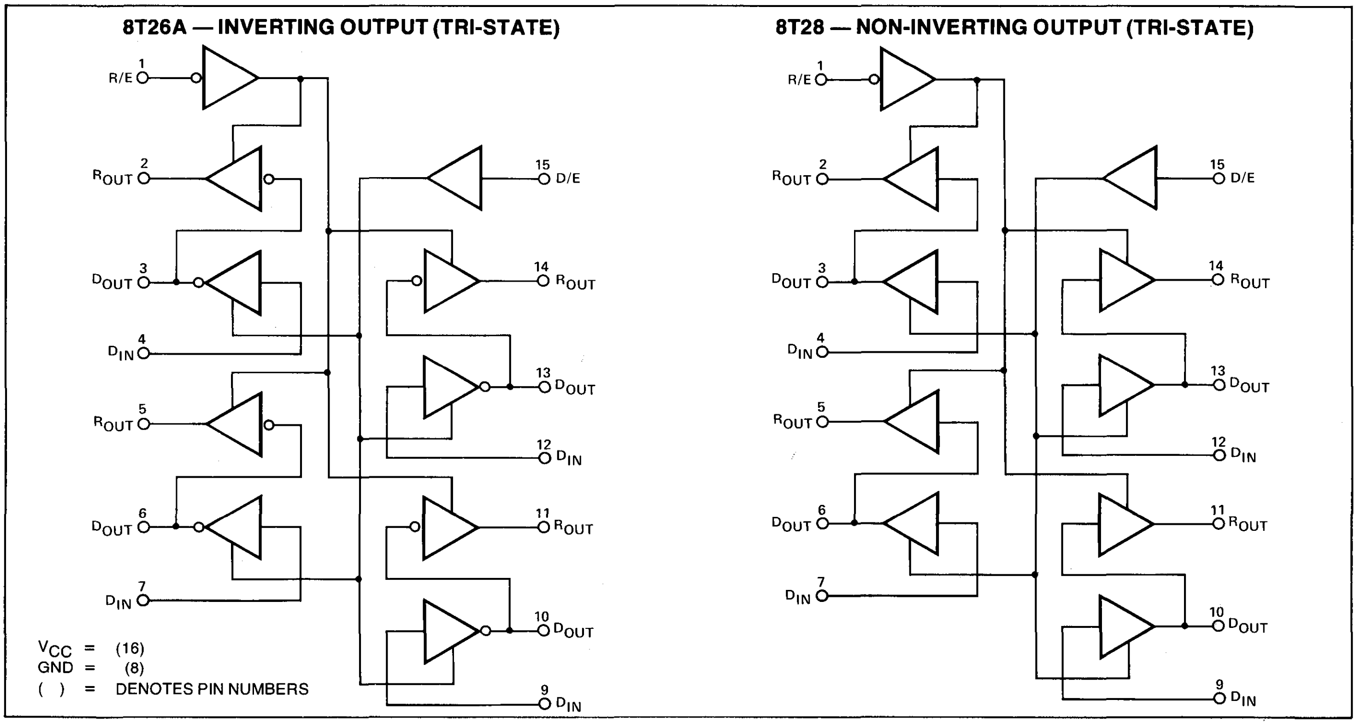
APPLICATIONS

- **Half-duplex data transmission**
- **Memory interface buffers**
- **Data routing in bus oriented systems**
- **High current drivers**
- **MOS/CMOS-to-TTL interface**

PIN CONFIGURATION



LOGIC DIAGRAM

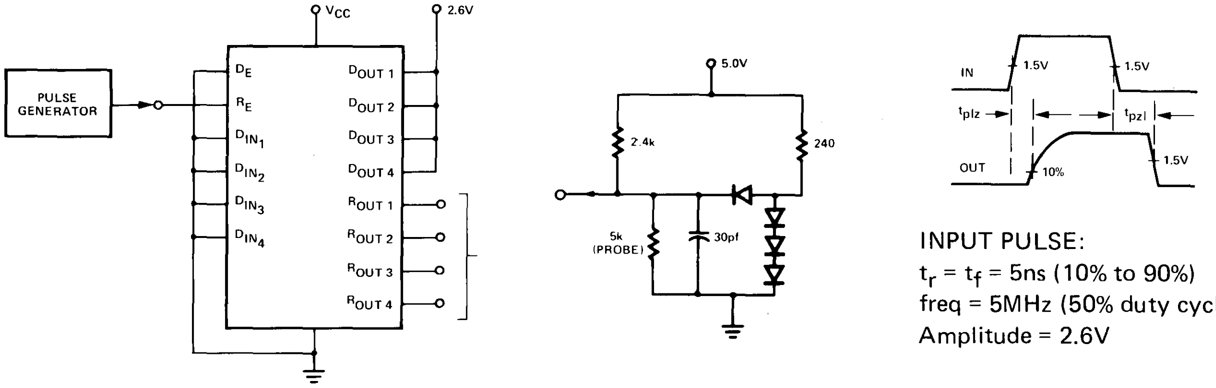


SWITCHING CHARACTERISTICS

PARAMETER		TEST CONDITIONS	8T26A	8T28	UNIT
			Max	Max	
Propagation Delay t_{ON}	D_{OUT} to R_{OUT}	C_L = 30pF, Note 9	14	17	ns
t_{OFF}	D_{OUT} to R_{OUT}		14	17	
t_{ON}	D_{IN} to D_{OUT}	C_L = 300pF, Note 9	14	17	ns
t_{OFF}	D_{IN} to D_{OUT}		14	17	
Data Enable to Data Output t_{PZL}	High Z to O	C_L = 300pF, Note 9	25	28	ns
t_{PLZ}	O to High Z		20	23	
Receiver Enable to Receiver Output t_{PZL}	High Z to O	C_L = 30pF, Note 9	20	23	ns
t_{PLZ}	O to High Z		15	18	

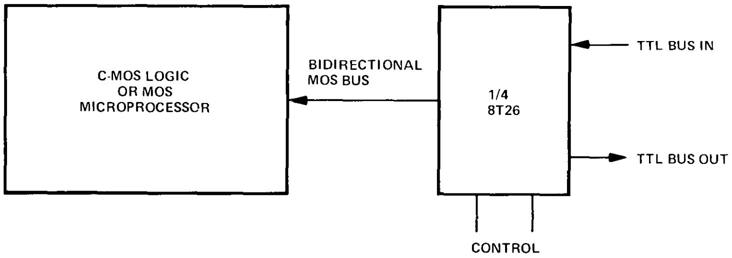
BLOCK DIAGRAM

PROPAGATION DELAY (RECEIVE ENABLE TO RECEIVE OUTPUT)



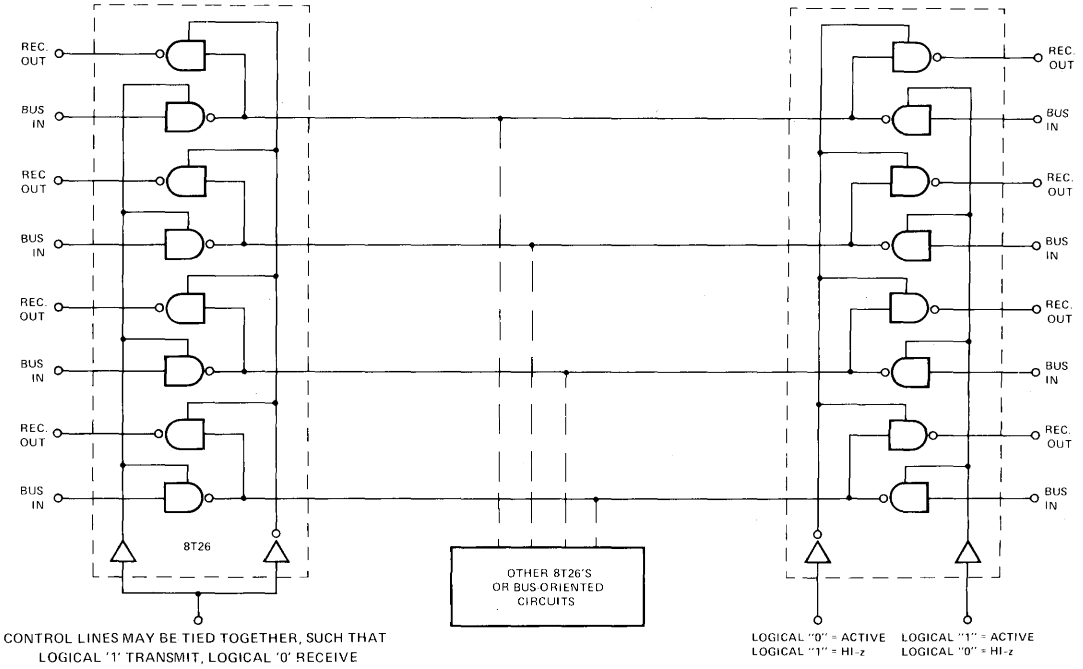
INPUT PULSE:
$t_r = t_f$ = 5ns (10% to 90%)
freq = 5MHz (50% duty cycle)
Amplitude = 2.6V

BIDIRECTIONAL MOS - CMOS TO TTL INTERFACE



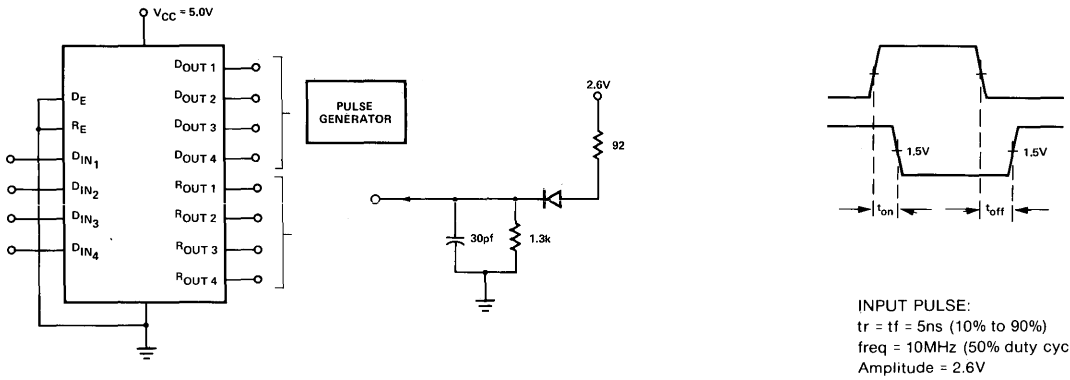
TYPICAL APPLICATIONS

BIDIRECTIONAL DATA BUS



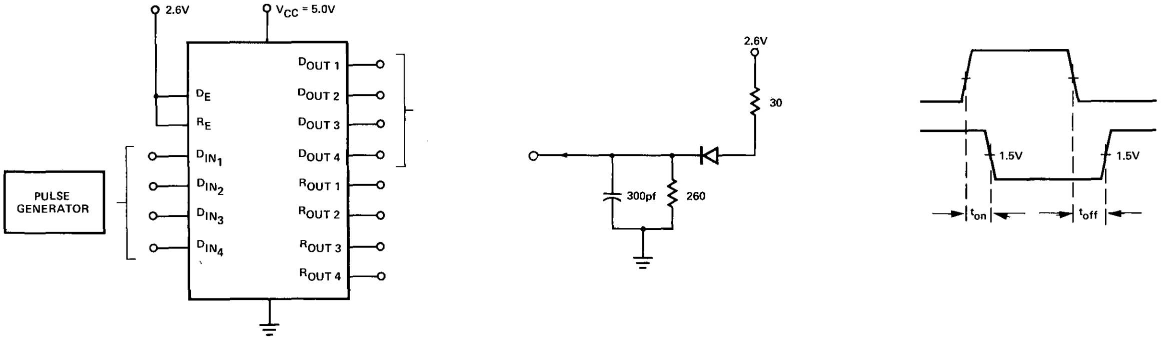
AC TEST CIRCUITS AND WAVEFORMS

PROPAGATION DELAY (DOUT TO ROUT)

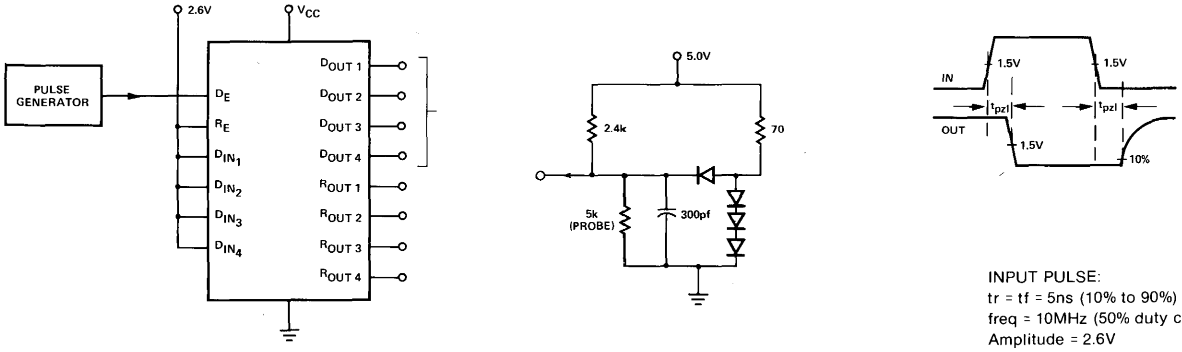


INPUT PULSE:
tr = tf = 5ns (10% to 90%)
freq = 10MHz (50% duty cycle)
Amplitude = 2.6V

PROPAGATION DELAY (DIN TO DOUT)



PROPAGATION DELAY (DATA ENABLE TO DATA OUTPUT)



INPUT PULSE:
tr = tf = 5ns (10% to 90%)
freq = 10MHz (50% duty cycle)
Amplitude = 2.6V

8-BIT BIDIRECTIONAL I/O PORT 8T31

OBJECTIVE SPECIFICATION

8T31 N,F

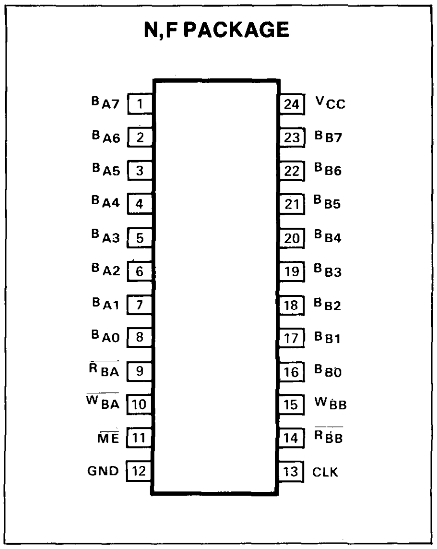
DESCRIPTION

The 8T31 8-bit Bidirectional I/O Port is designed to function as a general purpose I/O interface element in minicomputers, microcomputers and other bus oriented digital systems. It consists of 8 clocked latches with two sets of bidirectional inputs/outputs, Bus A (B_{A0}-B_{A7}) and Bus B (B_{B0}-B_{B7}). Each Bus has a write control line and a read control line. The two buses operate independently except for the case where the user is attempting to write data in from each bus simultaneously. In that case, the data on Bus A will be written into the latches while Bus B will be forced into a high impedance state. Data written into one Bus will appear inverted at the other Bus.

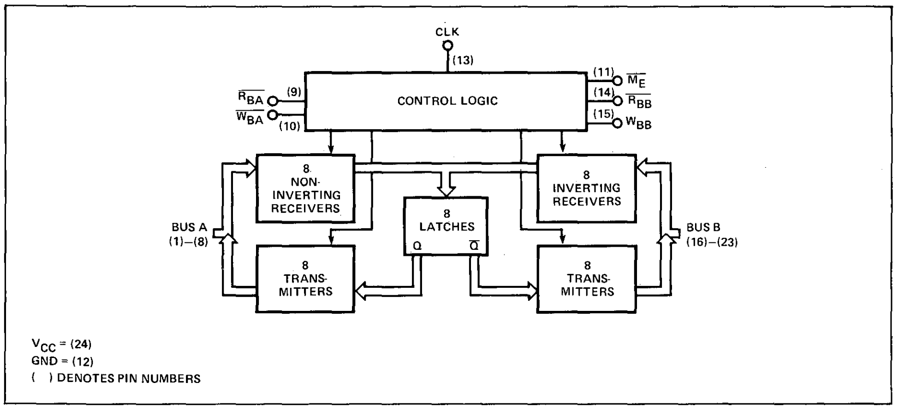
A master enable (M_E) is provided that enables or disables Bus B regardless of the state of the other inputs.

A unique feature of the 8T31 is its ability to start up in a predetermined state. If the clock is maintained at a voltage less than .8V until the power supply reaches 3.5V, Bus A will always be all logic 1 levels, while Bus B will be all logic 0 levels.

PIN CONFIGURATION



BLOCK DIAGRAM

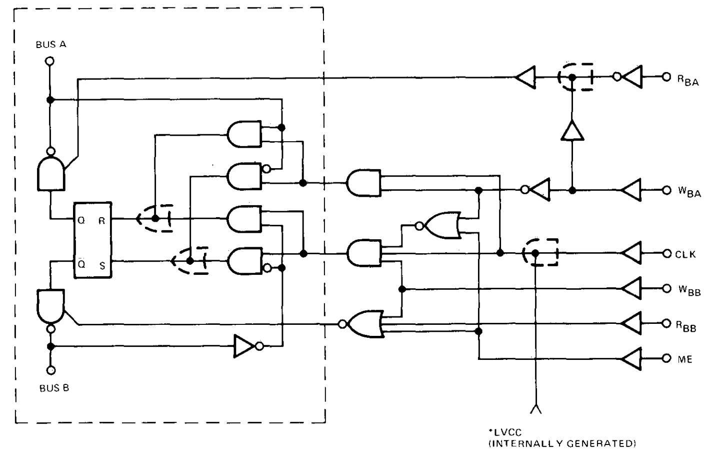


FUNCTION TABLE

BUS A			
$\overline{R_{BA}}$	$\overline{W_{BA}}$	CLK	
X	0	1	WRITE (INPUT)
0	1	X	READ (OUTPUT)
1	1	X	HI-Z

BUS B					
$\overline{R_{BB}}$	$\overline{W_{BB}}$	W_{BA}	CLK	$\overline{ME}$	
X	X	X	X	1	HI-Z
1	0	X	X	0	HI-Z
X	1	0	X	0	HI-Z
0	0	X	X	0	READ (OUTPUT)
X	1	1	1	0	WRITE (INPUT)

SCHEMATIC



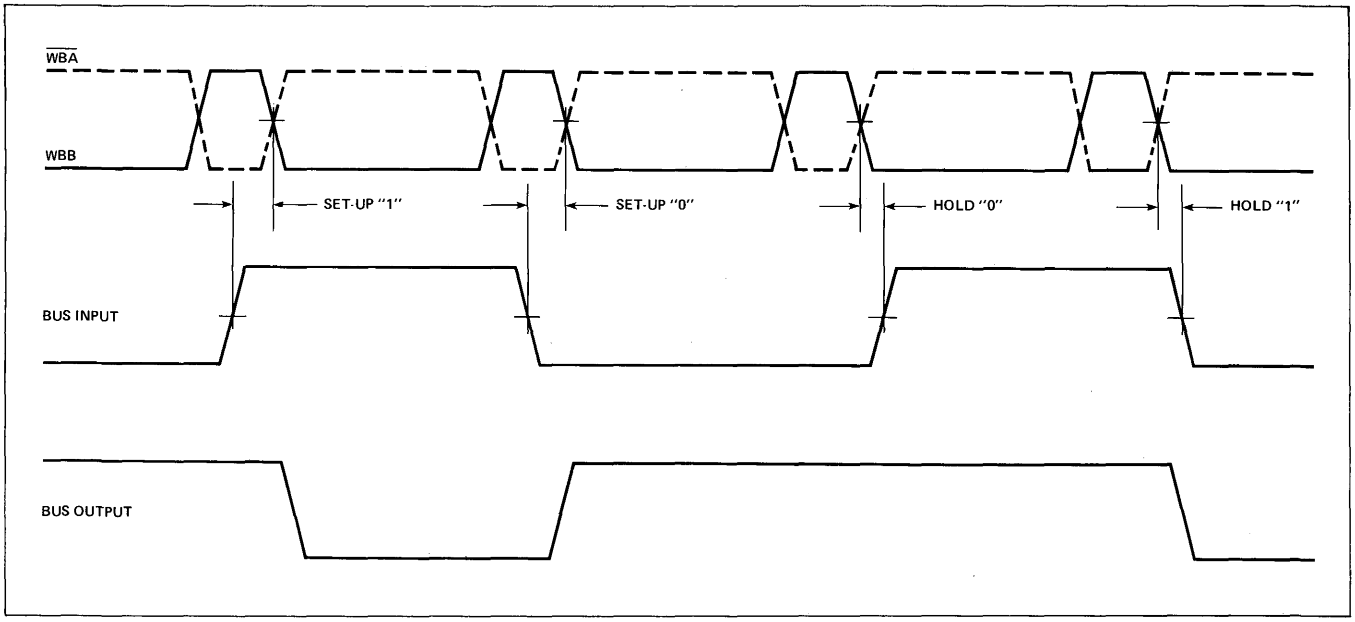
NOTE: CIRCUIT INSIDE DOTTED LINE IS FOR ONE BIT ONLY.
* LOW VOLTAGE CONTROL CIRCUIT

SWITCHING CHARACTERISTICS

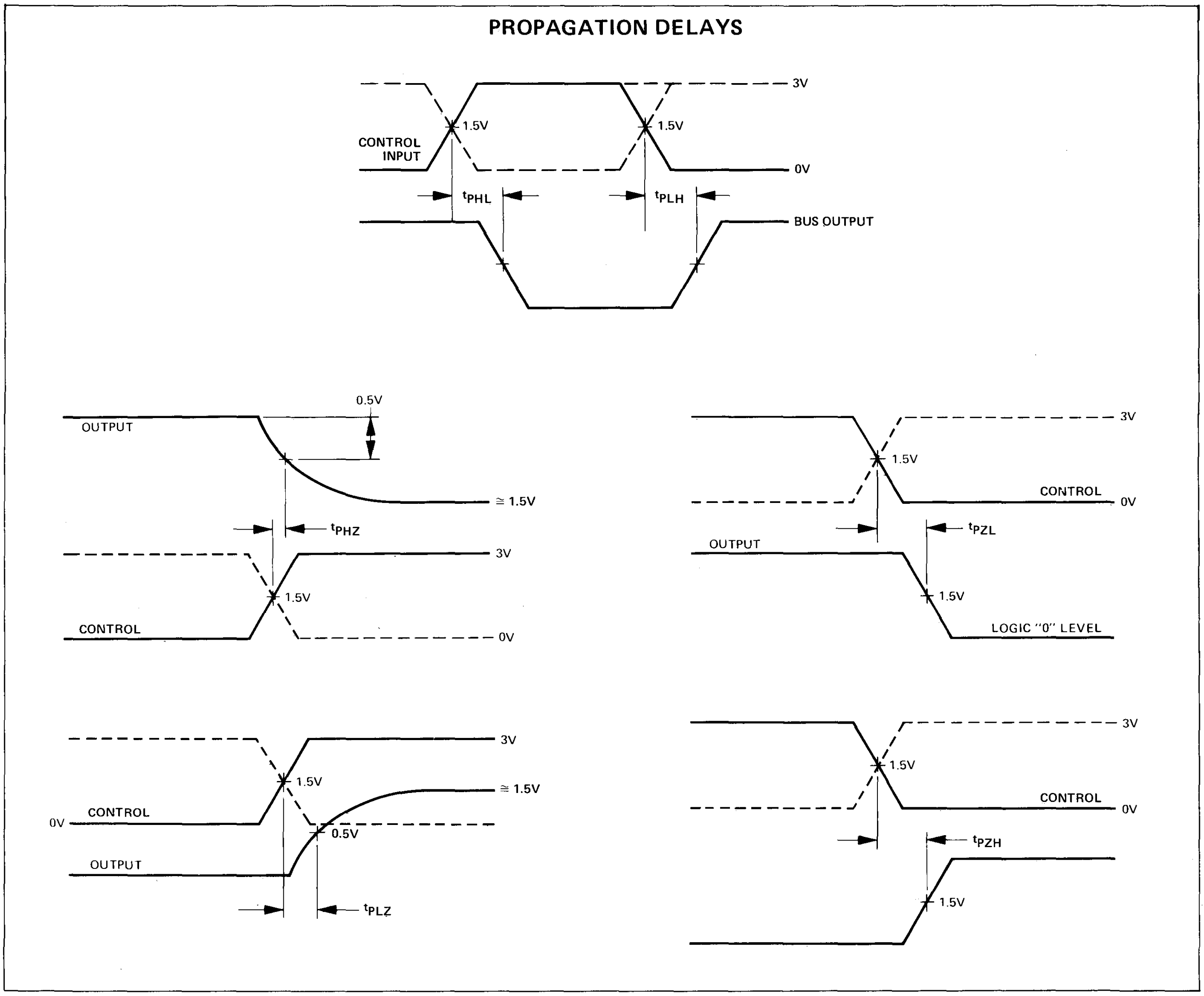
PARAMETER		TEST CONDITIONS	LIMITS			UNIT
			Min	Typ	Max	
t_{ZL} t_{ZH} t_{ZL} t_{ZH} t_{LZ} t_{HZ}	Propagation Delay From Read (RBB), Write (WBB) and Master Enable (ME) to Bus B	C_L = 300pF C_L = 300pF C_L = 30pF C_L = 30pF C_L = 30pF C_L = 30pF		27 29 17 14 13 17	45 50 30 25 20 30	ns ns ns ns ns ns
t_{SETUP} t_{HOLD1} t_{HOLD0}	Bus A Data Setup and Hold Times		0 10 25	-10 4 16		ns ns ns
t_{SETUP} t_{HOLD}	Bus A Write Setup and Hold Times		30 0	20 -30		ns ns
t_{SETUP} t_{HOLD}	Bus B Data Setup and Hold Times		* 0			ns ns
C_{IN}	Input Capacitances Control Data	V_{IN} = 0V V_{IN} = 0V V_{IN} = 3V			6 12 9	pF pF pF

*The Bus B Data Setup Time is equal to the clock pulse width.

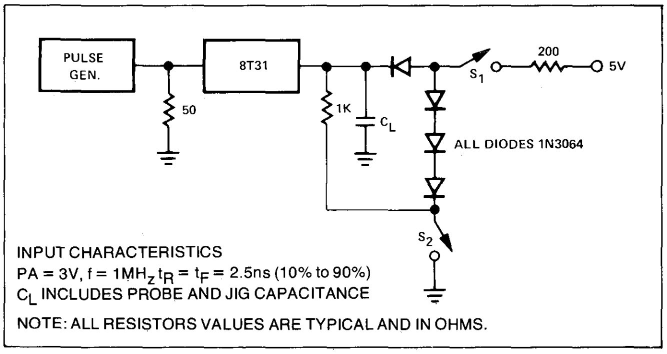
CLOCK



AC WAVEFORMS



AC TEST CIRCUIT



INPUT CHARACTERISTICS
PA = 3V, f = 1MH$_z$ t_R = t_F = 2.5ns (10% to 90%)
C_L INCLUDES PROBE AND JIG CAPACITANCE

NOTE: ALL RESISTORS VALUES ARE TYPICAL AND IN OHMS.

TEST TABLE

	S_1	S_2
t_{PHL}	Closed	Closed
t_{PLH}	Closed	Closed
t_{PL}	Closed	Closed
t_{PHZ}	Closed	Closed
t_{PZL}	Closed	Open
t_{PZH}	Open	Closed

QUAD BUS TRANSCEIVER WITH TRI-STATE OUTPUTS 8T34

8T34A

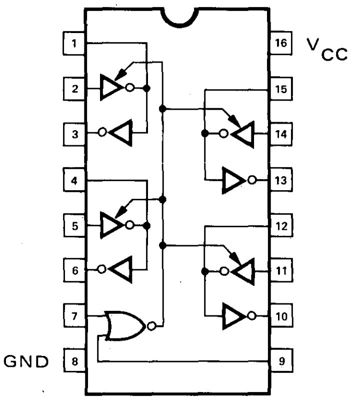
DESCRIPTION

The 8T34 is a quad transceiver with a common two input driver disable control. Tri-state driver outputs together with low input current requirements for the receivers offer extreme versatility in bus organized data transmission systems. The data busses may be terminated or unterminated.

Drivers in the third output state (Hi-Z) load the bus only with negligible current. The receiver input current is low, allowing at least 100 driver/receiver pairs to utilize a single bus. The receiver incorporates hysteresis to provide maximum noise immunity. In addition the receiver does not load the bus with V_{CC} = 0V as it may be the case when peripherals drive a common I/O bus and are shut off.

PIN CONFIGURATION

A PACKAGE



TRUTH TABLE

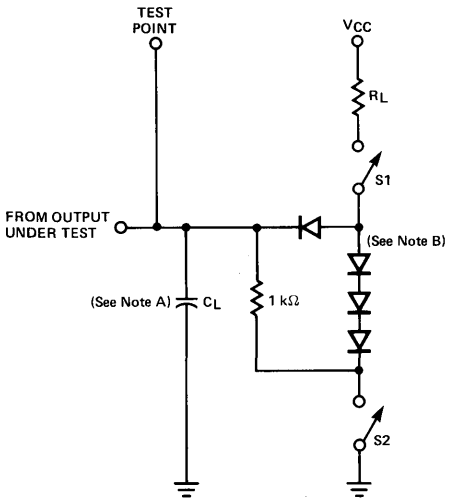
MODE	DISABLE	DISABLE	DRIVER	BUS	RECEIVER
	A	B	IN		OUT
RECEIVE	1	X	X	1	0
RECEIVE	X	1	X	0	1
DRIVE	0	0	1	0	1
DRIVE	0	0	0	1	0

ELECTRICAL CHARACTERISTICS (T_A = +25° C, V_{CC} = 5.0V)

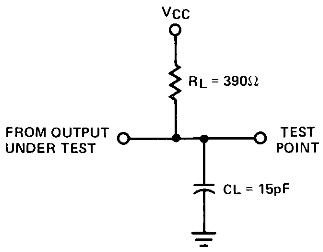
	PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{HZ}	Disable to Bus	Load 1, C_L = 15pF Waveform 4	8	15	30	ns
t_{LZ}	Disable to Bus	Load 1, C_L = 15pF Waveform 3	3	9	30	ns
t_{ZH}	Disable to Bus	Load 1, C_L = 50pF Waveform 3	5	10	30	ns
t_{ZL}	Disable to Bus	Load 1, C_L = 50pF Waveform 4	8	18	30	ns
t_{PHL}	Driver to Bus	Load 3	4	9	20	ns
t_{PLH}	Driver to Bus	Waveform 5	3	6	15	ns
t_{PHL}	Bus to Receiver	Load 2	5	14	25	ns
t_{PLH}	Bus to Receiver	Waveform 6	12	27	40	ns

SWITCHING PARAMETER MEASUREMENT INFORMATION

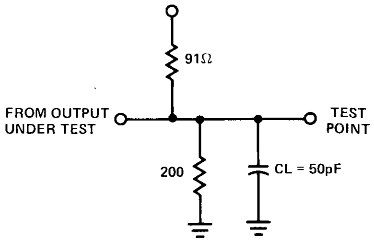
LOAD CIRCUIT FOR TRI-STATE OUTPUTS



LOAD 1



LOAD 2



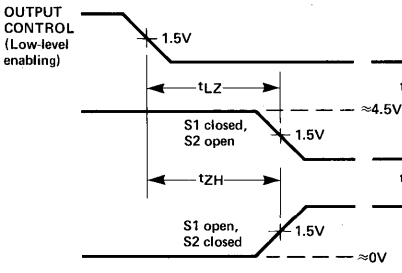
LOAD 3

NOTES:

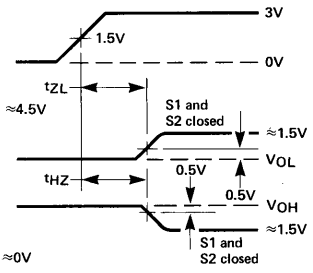
A. C_L includes probe and jig capacitance

B. Pin diodes are IN3064

VOLTAGE WAVEFORMS
ENABLE AND DISABLE TIMES, TRI-STATE OUTPUTS

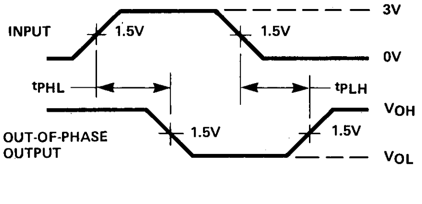


WAVEFORM 3

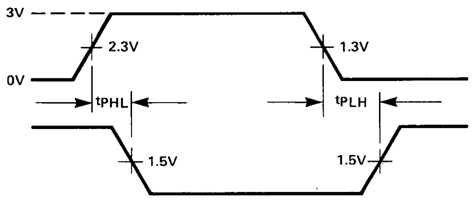


WAVEFORM 4

VOLTAGE WAVEFORMS PROPAGATION DELAY TIMES



WAVEFORM 5



WAVEFORM 6

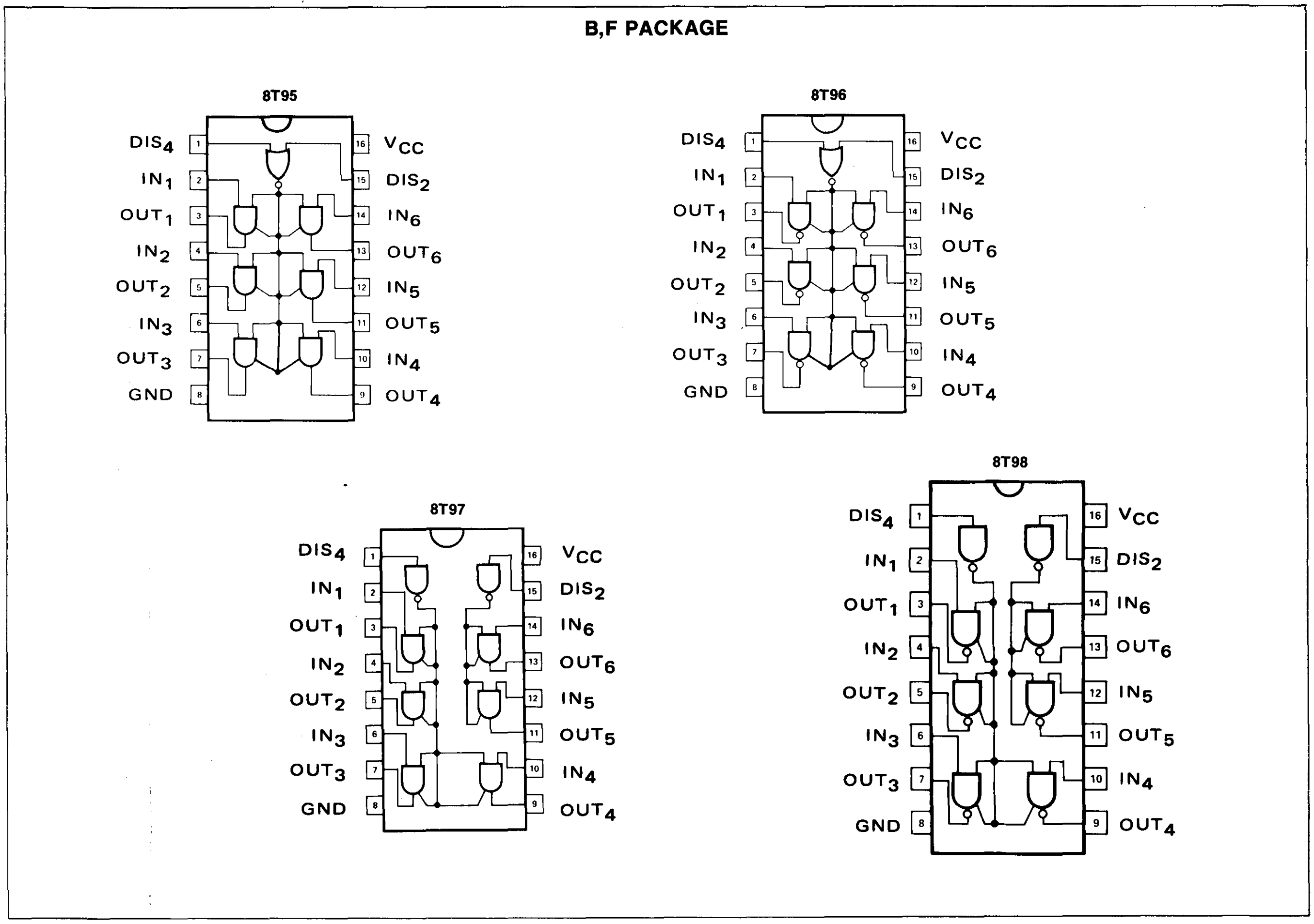
HIGH SPEED HEX TRI-STATE BUFFERS 8T95/8T96
HIGH SPEED HEX TRI-STATE INVERTERS 8T97/8T98

8T95-B,F • 8T96-B,F • 8T97-B,F • 8T98-B,F

DESCRIPTION

Each of the Tri-State Bus Interface Elements described herein has low current PNP inputs and is designed with Schottky TTL technology for ultra high speed. The devices are used to convert TTL/DTL or MOS/CMOS to tri-state TTL Bus levels. For maximum systems flexibility the 8T95 and 8T97 do so without logic inversion, whereas, the 8T96 and 8T98 provide the logical complement of the input. The 8T95 and 8T96 feature a common control line for all six devices, whereas, the 8T97 and 8T98 have control lines for four devices from one input and two from another input.

PIN CONFIGURATIONS



B,F PACKAGE

TRUTH TABLES

8T95			
DISABLE DIS_1	INPUT DIS_2	INPUT	OUTPUT
0	0	0	0
0	0	1	1
0	1	x	H-z
1	0	x	H-z
1	1	x	H-z

8T96			
DISABLE DIS_1	INPUT DIS_2	INPUT	OUTPUT
0	0	0	1
0	0	1	0
0	1	x	H-z
1	0	x	H-z
1	1	x	H-z

TRUTH TABLES (Cont'd)

8T97			
DISABLE DIS_4	INPUT DIS_2	INPUT	OUTPUT
0	0	0	0
0	0	1	1
x	1	x	H-z*
1	x	x	H-z**

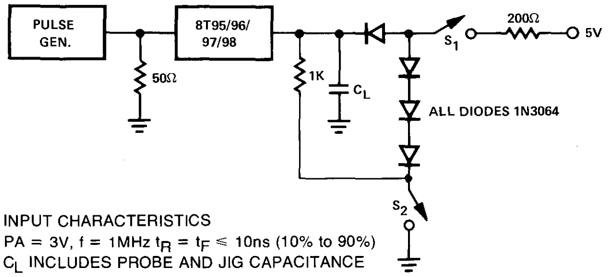
8T98			
DISABLE DIS_4	INPUT DIS_2	INPUT	OUTPUT
0	0	0	1
0	0	1	0
x	1	x	H-z*
1	x	x	H-z**

* Output 5-6 only ** Output 1-4 only x = Irrelevant

AC ELECTRICAL CHARACTERISTICS T_A = 25° C and V_{CC} = 5.0V

	PARAMETER	TEST CONDITIONS	LIMITS						UNITS
			MIN		TYP		MAX		
			95/97	96/98	95/97	96/98	95/97	96/98	
	Propagation Delays (All Devices)								
t_{on}	Data Inputs to	See AC Test Figures	3	3	9	6	13	10	ns
t_{off}	Data Outputs		3	4	7	7	12	11	ns
	Disable to Outputs								
t_{PIH}	Logic "1" to High Z		3	3	5	6	10	10	ns
t_{POH}	Logic "0" to High Z		3	5	6	10	12	16	ns
t_{PHI}	High Z to Logic "1"		8	7	19	15	25	22	ns
t_{PHO}	High Z to Logic "0"		12	11	14	18	25	24	ns

AC TEST CIRCUIT



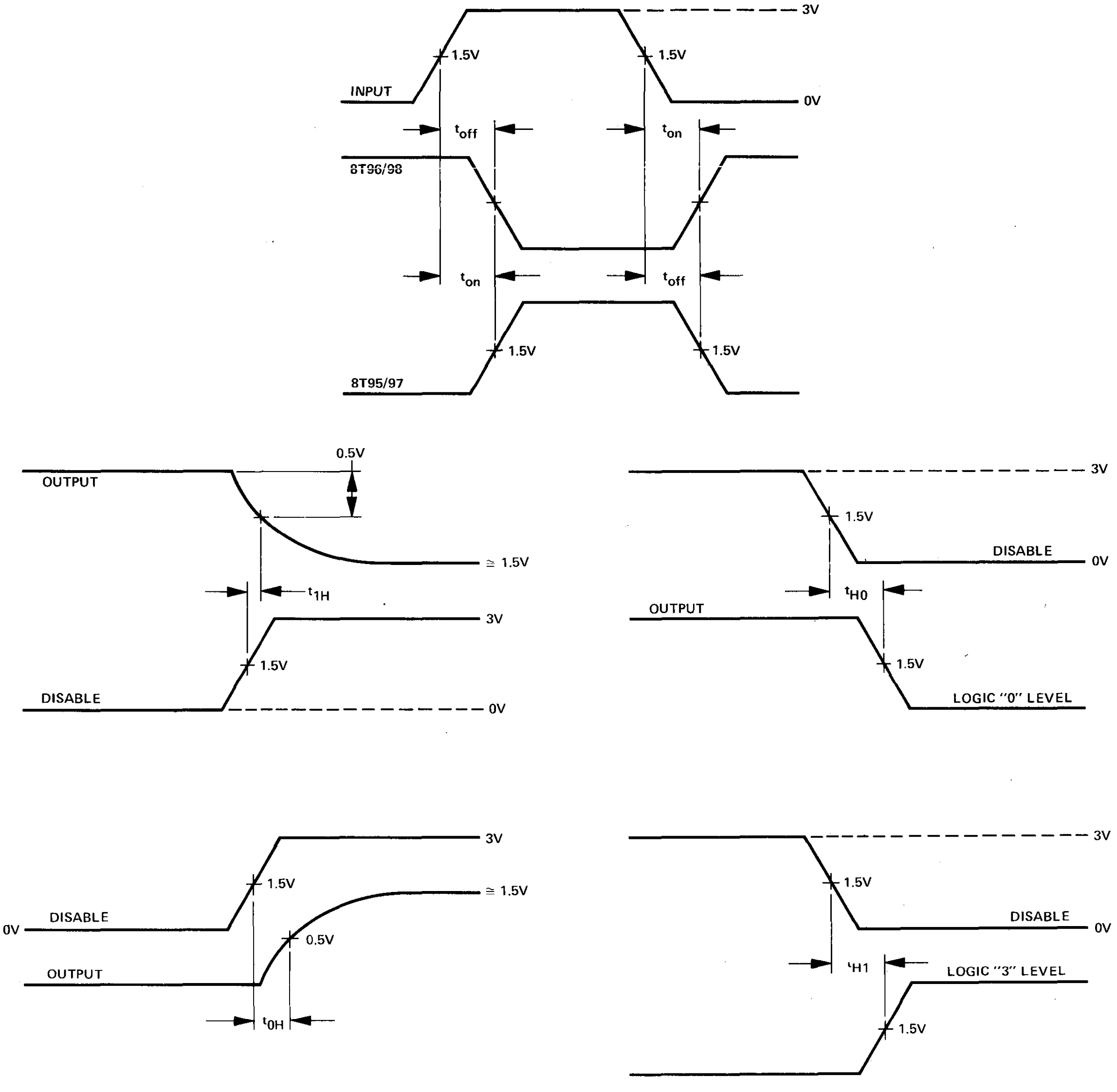
INPUT CHARACTERISTICS
PA = 3V, f = 1MHz t_R = t_F ≤ 10ns (10% to 90%)
C_L INCLUDES PROBE AND JIG CAPACITANCE

TRUTH TABLE

	S_1	S_2	C_L
t_{on}	Closed	Closed	50pF
t_{off}	Closed	Closed	50pF
t_{0H}	Closed	Closed	5pF
t_{1H}	Closed	Closed	5pF
t_{H0}	Closed	Open	50pF
t_{H1}	Open	Closed	50pF

8T95-B,F • 8T96-B,F • 8T97-B,F • 8T98-B,F

PROPAGATION DELAYS



LOGIC

54/74 SERIES TTL

DEVICE	DESCRIPTION	Commercial 7400	Military 5400	Commercial 74H	Military 54H	Commercial 74LS	Military 54LS	Commercial 74S	Military 54S	Data Book Page Ref.
54/7400	Quad 2-Input NAND Gate	●	●	●	●	●	●	●	●	53
54/7401	Quad 2-Input NAND Gate with o/c	●	●	●	●	●	●			53
54/7402	Quad 2-Input NOR Gate	●	●			●	●	●	●	54
54/7403	Quad 2-Input NAND Gate with o/c	●	●			●	●	●	●	55
54/7404	Hex Inverter	●	●	●	●	●	●	●	●	55
54/7405	Hex Inverter with o/c	●	●	●	●	●	●	●	●	56
54/7406	Hex Inverter with Buffer/Driver with o/c	●	●							56
54/7407	Hex Buffer/Driver with o/c	●	●							57
54/7408	Quad 2-Input AND Gate	●	●	●	●	●	●	●	●	57
54/7409	Quad 2-Input AND Gate with o/c	●	●			●	●	●	●	58
54/7410	Triple 3-Input NAND Gate	●	●	●	●	●	●	●	●	58
54/7411	Triple 3-Input NAND Gate	●	●	●	●	●	●	●	●	59
54/7412	Triple 3-Input NAND Gate with o/c					●	●			59
54/7413	Dual NAND Schmitt Trigger	●	●			●	●			60
54/7414	Hex Schmitt Trigger	●	●			●	●			60
54/7415	Triple 3-Input AND Gate with o/c					●	●	●	●	61
54/7416	Hex Inverter Buffer/Driver with o/c	●	●							62
54/7417	Hex Buffer/Driver with o/c	●	●							62
54/7420	Dual 4-Input NAND Gate	●	●	●	●	●	●	●	●	63
54/7421	Dual 4-Input AND Gate	●	●	●	●	●	●			63
54/7422	Dual 4-Input NAND Gate with o/c			●	●	●	●	●	●	64
54/7426	Quad 2-Input NAND Gate with o/c	●	●			●	●			64
54/7427	Triple 3-Input NOR Gate	●	●			●	●			65
54/7428	Quad 2-Input NOR Buffer	●	●			●	●			65
54/7430	8-Input NAND Gate	●	●	●	●	●	●			66
54/7432	Quad 2-Input OR Gate	●	●			●	●	●	●	66
54/7433	Quad 2-Input NOR Buffer	●	●			●	●			67
54/7437	Quad 2-Input NAND Buffer	●	●			●	●	●	●	67
54/7438	Quad 2-Input NAND Buffer with o/c	●	●			●	●	●	●	68
54/7439	Quad 2-Input NAND Buffer	●	●							68
54/7440	Dual 4-Input NAND Buffer	●	●	●	●	●	●	●	●	69
54/7442	BCD-to-Decimal Decoder	●	●			●	●			69
54/7443	Excess 3-to-Decimal Decoder	●	●							70
54/7444	Excess 3-Gray-to-Decimal Decoder	●	●							71
54/7445	BCD-to-Decimal Decoder/Driver with o/c	●	●							72
54/7446A	BCD-to-7 Segment Decoder/Driver	●	●							73
54/7447A	BCD-to-7 Segment Decoder/Driver	●	●							74
54/7448	BCD-to-7 Segment Decoder/Driver	●	●							76
54/7450	Expandable Dual 2-Wide 2-Input AOI	●	●	●	●					77
54/7451	Dual 2-Wide 2-Input AOI Gate	●	●	●	●	●	●	●	●	77
54/7452	Expandable 4-Wide 2-2-2-3 Input AND/OR			●	●					78
54/7453	4-Wide 2-Input AOI Gate (Expandable)	●	●	●	●					79
54/7454	4-Wide 2-Input AOI Gate	●	●	●	●	●	●			80
54/7455	2-Wide 4-Input AOI Gate			●	●	●	●			81
54/7460	Dual 4-Input Expander	●	●	●	●					81
54/7461	Triple 3-Input Expander			●	●					82
54/7462	3-2-2-3 Input AND/OR Expander			●	●					82
54/7464	4-2-3-2 Input AOI Gate							●	●	83
54/7465	4-2-3-2 Input AOI Gate							●	●	83
54/7470	J-K Flip-Flop	●	●							84
54/7471	J-K Master-Slave Flip-Flop with AND/OR Inputs			●	●					84
54/7452	J-K Master-Slave Flip-Flop	●	●	●	●					85
54/7473	Dual J-K Master-Slave Flip-Flop	●	●	●	●	●	●			86
54/7474	Dual D-Type Edge-Triggered Flip-Flop	●	●	●	●	●	●	●	●	87
54/7475	Quad Bistable Latch	●	●			●	●			88
54/7476	Dual J-K Master-Slave Flip-Flop	●	●	●	●	●	●			90
54/7477	Quad Bistable Latch		●							91
54/7478	Dual J-K Negative Edge-Triggered Flip-Flop					●	●			91
54/7480	Gated Full Adder	●	●							92
54/7483	4-Bit Binary Full Adder	●	●							93
54/7483A	4-Bit Binary Full Adder	●	●			●				93

54/74 SERIES TTL

DEVICE	DESCRIPTION	Commercial 7400	Military 5400	Commercial 74H	Military 54H	Commercial 74LS	Military 54LS	Commercial 74S	Military 54S	Data Book Page Ref.
54/7485	4-Bit Magnitude Comparator	●	●			●	●	●	●	95
54/7486	Quad 2-Input Exclusive-OR Gate	●	●			●	●	●	●	98
54/7490	Decade Counter	●	●			●	●			100
54/7491	8-Bit Shift Register	●	●							102
54/7492	Divide-By-Twelve Counter	●	●			●	●			102
54/7493	4-Bit Binary Counter	●	●			●	●			104
54/7494	4-Bit Shift Register (PISO)	●	●							105
54/7495A	4-Bit Left-Right Shift Register	●	●			●	●			N/A
54/7495B	4-Bit Left-Right Shift Register					●				106
54/7496	5-Bit Shift Register	●	●			●	●			108
54/74100	4-Bit Bistable Latch (Dual)	●	●							110
54/74101	J-K Negative Edge-Triggered Flip-Flop			●	●					111
54/74102	J-K Negative Edge-Triggered Flip-Flop			●	●					112
54/74103	Dual J-K Negative Edge-Triggered Flip-Flop			●	●					114
54/74106	Dual J-K Negative Edge-Triggered Flip-Flop			●	●					115
54/74107	Dual J-K Master-Slave Flip-Flop	●	●			●	●			116
54/74108	Dual J-K Negative Edge-Triggered Flip-Flop			●	●					117
54/74109	Dual J-K Positive Edge-Triggered Flip-Flop	●	●			●	●			118
54/74112	Dual J-K Negative Edge-Triggered Flip-Flop					●	●	●	●	119
54/74113	Dual J-K Negative Edge-Triggered Flip-Flop					●	●	●	●	121
54/74114	Dual J-K Negative Edge-Triggered Flip-Flop					●	●	●	●	122
54/74116	Dual 4-Bit Latch with Clear	●	●							123
54/74121	Monostable Multivibrator	●	●							124
54/74122	Retriggerable Monostable Multivibrator	●								128
54/74123	Retriggerable Monostable Multivibrator	●	●							129
54/74123A	Retriggerable Monostable Multivibrator	●	●							129
54/74125	Quad Bus Buffer Gate w/Tri-State Outputs	●	●							130
54/74126	Quad Bus Buffer Gate w/Tri-State Outputs	●	●							131
54/74128	Quad 2-Input NOR Buffer	●	●							131
54/74132	Quad Schmitt Trigger	●	●			●	●			132
54/74133	13-Input NAND Gate							●	●	132
54/74134	12-Input NAND Gate w/Tri-State Outputs							●	●	133
54/74135	Quad Exclusive-OR/NOR Gate							●		133
54/74136	Quad Exclusive-OR with o/c					●	●			134
54/74138	3-to-8 Line Decoder/Demux					●	●	●	●	134
54/74139	Dual 2-to-4 Line Decoder/Demux					●	●	●	●	135
54/74140	Dual 14-Input NAND Line Driver							●	●	N/A
54/74145	BCD-to-Decimal Decoder/Drive with o/c	●	●			●	●			136
54/74147	10-Line to 4-Line Priority Encoder	●	●							137
54/74148	8-Line to 3-Line Priority Encoder	●	●							138
54/74150	16-Line to 1-Line Mux	●	●							140
54/74151	8-Line to 1-Line Mux	●	●			●	●	●	●	142
54/74152	8-Line to 1-Line Data Selector/Mux	●	●							143
54/74153	Dual 4-Line to 1-Line Mux	●	●			●	●	●	●	145
54/74154	4-Line to 16-Line Decoder/Demux	●	●							146
54/74155	Dual 2-Line to 4-Line Decoder/Demux	●	●							147
54/74156	2-Line to 4-Line Decoder/Demux	●	●							148
54/74157	Quad 2-Input Data Selector (Non-Inv.)	●	●			●	●	●	●	149
54/74158	Quad 2-Input Data Selector (Inv.)	●	●			●	●	●		150
54/74160	Synchronous 4-Bit Decoder Counter	●	●			●				151
54/74161	Synchronous 4-Bit Binary Counter	●	●			●	●			153
54/74162	Synchronous 4-Bit Decade Counter	●	●			●	●			156
54/74163	Synchronous 4-Bit Binary Counter	●	●			●	●			160
54/74164	8-Bit Parallel-Out Serial Shift Register	●	●			●	●			162
54/74165	Parallel-Load 8-Bit Shift Register	●	●							164
54/74166	8-Bit Shift Register	●	●							167
54/74170	4 X 4 Register File	●	●			●	●			169
54/74172	16-Bit Multiple Port Register File							●		172
54/74173	Quad D-Type Flip-Flop (Tri-State) (8T10)	JUNE								N/A
54/74174	Hex D-Type Flip-Flop with Clear	●	●			●	●	●		174

LOGIC

54/74 SERIES TTL

DEVICE	DESCRIPTION	Commercial 7400	Military 5400	Commercial 74H	Military 54H	Commercial 74LS	Military 54LS	Commercial 74S	Military 54S	Data Book Page Ref.
54/74174	Hex D-Type Flip-Flop with Clear	●	●			●	●	●		174
54/74175	Quad D-Type Edge-Triggered Flip-Flop	●	●			●	●	●		175
54/74176	Presettable Decade Counter/Latch (8280)	●								176
54/74177	Presettable Binary Counter/Latch (8281)	●								176
54/74178	4-Bit Parallel Access Shift Register (8270)	●						DEV		177
54/74179	4-Bit Parallel Shift Register (8271)	●						DEV		177
54/74180	8-Bit Odd/Even Parity Checker	●	●							178
54/74181	4-Bit Arithmetic Logic Unit	●	●			●	●	●	●	178
54/74182	Look-Ahead Carry Generator	●	●					●	●	183
54/74190	Synchronous Up/Down BCD Counter	●	●			●	●			184
54/74191	Synchronous Up/Down Binary Counter	●	●			●	●			187
54/74192	Synchronous Decade Up/Down Counter	●	●			●	●			190
54/74193	Synchronous 4-Bit Binary Up/Down Counter	●	●			●	●			192
54/74194	4-Bit Bidirectional Universal Shift Reg	●	●			●	DEV	●	DEV	195
54/74195	4-Bit Parallel-Access Shift Register	●	●			●	●	●		197
54/74196	Presettable Decade Counter/Latch (8290)	●				DEV		DEV		200
54/74197	Presettable Binary Counter/Latch (8291)	●				DEV		DEV		201
54/74198	8-Bit Shift Register	●	●							203
54/74199	8-Bit Shift Register	●	●							206
54/74221	Dual Monostable Multivibrator	●	●			●	●			210
54/74251	Data Selector/Mux with Tri-State Outputs					●	●	●		212
54/74253	Dual 4-Line to 1-Line Data Selector/Mux					●	●	●	●	214
54/74257	Quad 2-Line to 1-Line Data Selector/Mux					●	●	●		215
54/74258	Quad 2-Line to 1-Line Data Selector/Mux					●	●	●	●	216
54/74260	Dual 5-Input NOR Gate					●	●	●	●	217
54/74261	2's Complement Multiplier					●	●			218
54/74266	Quad Exclusive-NOR Gate					●	●			220
54/74279	Quad S-R Latch	●	●							221
54/74280	9-Bit Odd/Even Parity Generator/Checker							●	●	221
54/74283	4-Bit Adder					●				222
54/74290	Decade Counter					●	●			223
54/74293	4-Bit Binary Counter					●	●			225
54/74295A	4-Bit Right-Shift Left-Shift Register					●				227
54/74298	Quad 2-Input Mux with Storage	●	●			●				228
54/74375	Quad Latch					●				N/A
54/74386	Exclusive-OR Gate					●				230
54/74670	4 X 4 Register File (Tri-State)					●	●			230

LOGIC

8200 SERIES TTL/MSI

		STANDARD 8200		SCHOTTKY 82S		Data Book Page Ref.
DEVICE	DESCRIPTION	Commercial	Military	Commercial	Military	
8200	Dual 5-Bit Buffer Register	●	●			255
8201	Dual 5-Bit Buffer Register with D Inputs	●	●			255
8202	10-Bit Buffer Register	●	●			255
8203	10-Bit Buffer Register with D Inputs	●	●			255
~~8230~~	~~8-Input Digital Multiplexer~~	~~●~~	~~●~~	~~●~~		~~257~~
8231	8-Input Digital Multiplexer	●	●	●		259
8232	8-Input Digital Multiplexer	●	●	●		259
8233	2-Input 4-Bit Digital Multiplexer	●	●	●		262
8234	2-Input 4-Bit Digital Multiplexer	●	●	●		262
8235	2-Input 4-Bit Digital Multiplexer	●	●			262
8241	Quad Exclusive-OR Gate	●	●	●		264
8242	Quad Exclusive-NOR Gate	●	●	●		264
8243	8-Bit Position Scaler	●	●			267
8250	Binary-to-Octal Decoder	●	●	●		271
8251	BCD-to-Decimal Decoder	●	●			271
8252	BCD-to-Decimal Decoder	●	●	●		271
8260	Arithmetic Logic Unit	●	●			275
8261	Fast Carry Extender	●	●			278
8262	9-Bit Parity Generator and Checker	●	●	●		280
8263	3-Input 4-Bit Digital Multiplexer	●	●			282
8264	3-Input 4-Bit Digital Multiplexer	●	●			282
8266	2-Input 4-Bit Digital Multiplexer	●	●	●		285
8267	2-Input 4-Bit Digital Multiplexer	●	●	●		285
8268	Gated Full Adder	●	●			288
8269	4-Bit Comparator	●	●			291
8270	4-Bit Shift Register	●	●	●		292
8271	4-Bit Shift Register	●	●	●		292
8273	10-Bit Serial-In, Parallel-Out Shift Register	●	●			297
8274	10-Bit Parallel-In, Serial-Out Shift Register	●	●			299
8275	Quad Bistable Latch	●	●			301
8276	8-Bit Serial Shift Register	●	●			302
8277	Dual 8-Bit Shift Register	●	●			304
8280	Presettable Decade Counter	●	●			306
8281	Presettable Binary Counter	●	●			306
8282	BCD Arithmetic Unit			●		310
8283	BCD Adder			●		314
8284	Binary Up/Down Counter	●	●			318
8285	Decade Up/Down Counter	●	●			318
8288	Divide-by-Twelve Counter	●	●			321
8290	Presettable High Speed Decade Counter	●	●	●		323
8291	Presettable High Speed Binary Counter	●	●	●		323
8292	Presettable Low Power Decade Counter	●	●			328
8293	Presettable Low Power Binary Counter	●	●			328

LOGIC FUNCTION SELECTOR GUIDES

Use these charts to quickly identify the most suitable devices to meet your system needs. The following charts group together similar function circuits in Signetics TTL families.

ARITHMETIC UNITS/MICROPROCESSOR CPUs

FUNCTION	DEVICE	NUMBER OF BITS	COMPLEMENT INPUTS	SUM OUTPUT	CARRY OUTPUT	CARRY LOOK AHEAD CIRCUIT	CARRY IN
ALU (BINARY)	8260 74181 74S181 74LS181	4 4 4 4					
ALU (BCD)	82S82	4					
GATED FULL ADDER (BINARY)	8268 7480 7483 74S83 74S283	2 2 4 4 4	X X	X X X X X	X X X X X	 X X X	 X
GATED FULL ADDER (BCD)	82S83	4		X	X	X	X
LOOK-AHEAD CARRY	8261 (Extender) 74182 74S182	4 4 4					
2's COMPLEMENT MULTIPLIER	74LS261	2x4					

DATA SELECTOR/MULTIPLEXER

FUNCTION	DEVICE	NUMBER OF OUTPUTS	NON-INVERTING OUTPUT	INVERTING OUTPUT	INVERTING/NON-INVERTING OUTPUT	CLOCKED	HOLD MODE	OPEN COLLECTOR OUTPUT	TRI-STATE OUTPUT	STANDARD OUTPUT	ENCODED MODE CONTROL	OUTPUT INHIBIT (0)	OUTPUT INHIBIT (1)	INPUT STROBE	SELECT LINE	OUTPUT ENABLE	LATCHED OUTPUTS	LATCHED	OPEN EMITTER OUTPUTS	DATA COMPLEMENT SELECT
SINGLE 8-INPUT MULTIPLEXER	8230/82S30	2			X					X					X	X				
	8231/82S31	2			X			X					X		X	X				
	8232/82S32	2			X					X		X			X	X				
	74151	2			X					X			X	X	X					
	74LS151	2			X					X			X	X	X					
	74S151	2			X					X			X	X	X					
	74152	1		X						X			X		X					
	74LS251	2			X				X				X	X	X					
	74S251	2			X				X				X	X	X					
	9312	2			X									X	X					
SINGLE 16-INPUT MULTIPLEXER	74150	1		X										X	X					
DUAL 4-INPUT MULTIPLEXER	74153	2	X							X	X	X		X						
	74LS153	2	X							X	X	X		X						
	74S153	2	X							X	X	X		X						
	74LS253	2	X						X						X	X				
	74S253	2	X						X						X	X				
	9309	2			X					X	X									
QUAD 2-INPUT MULTIPLEXER	8233/82S33	4	X							X	X	X								
	8234/82S34	4		X				X			X		X							
	8235	4			X			X			X		X							
	8266/82S66	4			X					X	X		X							
	8267/82S67	4			X			X			X		X							
	74157/74S157	4	X							X			X	X	X					
	74158/74S158	4		X						X			X	X	X					
	74S257	4	X						X						X	X				
	74S258	4		X					X						X	X				
	74298 (w/storage)	4	X			↓	X			X					X		X			
QUAD 3-INPUT MULTIPLEXER	8263	4			X					X	X									X
	8264	4			X			X			X		X			X				X

COUNTERS

FUNCTION	DEVICE	NUMBER OF STAGES	UP COUNTER	DOWN COUNTER	UP/DOWN COUNTER	ASYNCHRONOUS PRESET	SYNCHRONOUS PRESET	ASYNCHRONOUS RESET (0)	ASYNCHRONOUS RESET (9)	ASYNCHRONOUS RESET (15)	SYNCHRONOUS RESET (0)	CLOCK TRIGGERING	SINGLE UP/DOWN CONTROL	SEPARATE UP/DOWN CLOCK	COUNT ENABLE	CARRY IN	CARRY OUT	BORROW OUT	ENCODED MODE CONTROL	DECODED OUTPUT	CLOCK FREQUENCY (MHz)		
																					Min	Typ	Max
RIPPLE BINARY COUNTER	8281/74197	4	X			X		X				↓									20	25	
	8291	4	X			X		X				↓									40	60	
	82S91	4	X			X		X				↓									85	100	
	8293	4	X			X		X				↓									5	10	
	7493	4	X					X				↑								X	10	18	
	74LS93	4	X					X				↑								X			32
	74LS197	4	X			X		X				↓									15		
RIPPLE DECADE COUNTER	8280/74176	4	X			X		X				↓									20		
	8290	4	X			X		X				↓									40	60	
	82S90	4	X			X		X				↓									85	100	
	8292	4	X			X		X				↓									5		
	7490	4	X					X	X			↓									10		
	74LS90	4	X					X	X			↓									30		
	74LS196	4	X			X		X				↓									30		
	74S196	4	X			X		X				↓									85	100	
RIPPLE DIVIDE BY 12 COUNTER	8288	4	X			X		X				↓									20	25	
	7492	4	X					X				↓							X		10		
	74LS92	4	X					X				↓							X		16		
SYNCHRONOUS BINARY COUNTER	8284	4			X			X		X		↓	X		X	X	X				20	30	
	74161	4	X				X	X				↓			X			X					25
	74LS161	4	X				X	X				↑			X			X					25
	74163	4	X				X				X	↑			X			X					25
	74LS163	4	X				X				X	↑			X			X					25
	74191	4			X	X						↑	X		X	X	X	X			20	25	
	74LS191	4			X	X						↑	X		X	X	X	X					20
	74193	4			X	X		X				↑		X			X	X			25	32	
	74LS193	4			X	X		X				↑		X			X	X					25
SYNCHRONOUS DECADE COUNTER	8285	4			X	X		X	X			↓	X		X	X	X				20	30	
	74160	4	X				X	X				↑			X		X						25
	74LS160	4	X																				
	74162	4	X				X				X	↑			X		X						25
	74LS162	4	X				X				X	↑			X		X						25
	74190	4			X	X						↑	X		X	X	X	X					20
	74LS190	4			X	X						↑	X		X	X	X	X			20	25	
	74192	4			X	X		X				↑		X			X	X					25
	74LS192	4			X	X		X				↑		X			X	X					25

DECODERS/DEMULTIPLEXERS

FUNCTION	DEVICE	OUTPUT SINK (mA)	OUTPUT SOURCES (mA)	HIGH OUTPUT VOLTAGE	OPEN COLLECTOR OUTPUTS	STANDARD OUTPUTS	ACTIVE LOW OUTPUTS	INHIBIT INVALID INPUTS	INPUT STROBE/ENABLE	OPEN EMITTER OUTPUTS	INVERTED OUTPUTS	RESISTOR PULLUP OUTPUTS	RIPPLE BLANKING INPUT	BLANKING INPUT	LAMP TEST	COMMON ANODE LED	COMMON CATHODE LED	LAMPS
BCD TO 7-SEGMENT DECODER/DRIVER	8T04	40			X								X	X	X	X		
	8T05		2.3									X	X	X	X		X	
	8T06	40			X								X	X	X		X	
	7446	20		30	X								X	X	X			
	7447	20		15	X								X	X	X	X		X
	7448		6.4									X	X	X	X		X	
BINARY TO OCTAL DECODER/ DEMULTIPLEXER	8250/82S50					X	X	X										
2-LINE TO 4-LINE DECODER/ DEMULTIPLEXER	74LS139 (Dual)					X	X		X									
	74S139 (Dual)					X	X		X									
	74155 (Dual)					X	X		X									
	74156 (Dual)				X		X		X									
3-LINE TO 8-LINE DECODER/ DEMULTIPLEXER	74LS138					X	X		X									
	74S138					X	X		X									
4-LINE TO 16-LINE DECODER/ DEMULTIPLEXER	74154					X	X	X	X									
EXCESS 3-TO-DECIMAL DECODER	7443					X	X	X										
EXCESS 3-GRAY-TO DECIMAL DECODER	7444					X	X	X										

ENCODERS

FUNCTION	DEVICE	NUMBER OF OUTPUTS	NON-INVERTING OUTPUTS	INVERTING OUTPUTS	TRI-STATE OUTPUTS	STANDARD OUTPUTS	OPEN COLLECTOR OUTPUTS	GROUP SELECT OUTPUT	INPUT ENABLE	OUTPUT ENABLE	OUTPUT INHIBIT (0)	OUTPUT INHIBIT (1)	OUTPUT INHIBIT (Hi-Z)	OPEN EMITTER OUTPUT
8-LINE TO 3-LINE PRIORITY ENCODER	82148	3		X	X			X	X	X			X	
	74148	3		X				X	X	X		X		
10-LINE TO 4-LINE PRIORITY ENCODER	82147	4		X	X					X			X	
	74147	4		X		X								

PARITY GENERATORS

FUNCTION	DEVICE	NUMBER OF BITS	EVEN PARITY OUTPUT	ODD PARITY OUTPUT	OUTPUT INHIBIT	ODD/EVEN SELECT	ODD/EVEN OUTPUT	
PARITY GENERATOR/ CHECKER	8262/82S62	9	X	X	X			
	74180	8	X	X		X		
	74S280	9	X	X				

COMPARATORS

FUNCTION	DEVICE	NUMBER OF BITS	ENCODED OUTPUT	CASCADING INPUT	A < B INPUT	A = B INPUT	A > B INPUT	A > B OUTPUT	A = B OUTPUT	A < B OUTPUT	OUTPUT ENABLE	INHIBIT OUTPUT (0)
COMPARATOR	8269	4	X	X								
	7485	4		X	X	X	X	X	X	X		
	74S85	4		X	X	X	X	X	X	X		
	9324	5						X	X	X	X	X

FLIP-FLOPS

FUNCTION	DEVICE	INVERTING OUTPUT	NON-INVERTING OUTPUT	CLOCK TRIGGERING	PRESET	RESET/CLEAR (SEPARATE)	AND-GATED INPUT	AND-OR GATED INPUT	COMMON RESET/CLEAR	OR-GATED INPUT	CLOCK ENABLE	GATED OUTPUTS	SEPARATE CLOCK	COMMON CLOCK	POLARITY INPUT	CLOCK FREQUENCY (MHz)		
																Min	Typ	Max
D-TYPE FLIP-FLOP	7474 (Dual)	X	X	↑	X	X										15	25	
	74LS74 (Dual)	X	X	↑	X	X										25	33	
	74S74 (Dual)	X	X	↑	X	X											70	
	74174 (Hex)		X	↑					X					X		25	35	
	74LS174 (Hex)		X	↑					X					X		30	40	
	74S174 (Hex)		X	↑					X					X		75	110	
	74175 (Quad)	X	X	↑					X					X				25
	74LS175 (Quad)	X	X	↑					X					X		30	40	
	74S175 (Quad)	X	X	↑					X					X		75	110	
SINGLE J-K FLIP-FLOP	7470	X	X	↑	X	X	X						X			15	35	
	74H71	X	X	H	X			X					X			25	30	
	7472	X	X	H	X	X	X						X			15	20	
	74H72	X	X	H	X	X	X						X			25	30	
	74H101	X	X	↓	X			X					X			40	50	
	74H102	X	X	↓	X	X	X									40	50	
DUAL J-K FLIP-FLOP	7473	X	X	H		X							X			15	20	
	74H73	X	X	H		X							X			25	30	
	74LS73	X	X	↓		X							X					30
	7476	X	X	H	X	X							X			15	20	
	74H76	X	X	H	X	X							X			25	30	
	74LS76	X	X	↓	X	X							X					30
	74LS78	X	X	↓	X				X					X		30	45	
	74H103	X	X	↓		X							X			40	50	
	74H106	X	X	↓	X	X							X			40	50	
	74107	X	X	H		X							X			15	20	
	74LS107	X	X	↓		X							X			30	45	
	74H108	X	X	↓	X				X					X		40	50	
	74109 (J/K)	X	X	↑	X	X							X			25	33	
	74LS109 (J/K)	X	X	↑	X	X							X					25
	74LS112	X	X	↓	X	X							X			30	45	
	74S112	X	X	↓	X	X							X			80	125	
	74LS113	X	X	↓	X								X					30
	74S113	X	X	↓	X											80	125	
	74LS114	X	X	↓	X				X					X		30	45	
	74S114	X	X	↓	X				X					X		80	125	

REGISTERS

FUNCTION	DEVICE	NUMBER OF BITS	INVERTED OUTPUTS	NON-INVERTED OUTPUTS	CLOCK TRIGGERING	ASYNCHRONOUS RESET	ASYNCHRONOUS LOAD	SYNCHRONOUS LOAD	HOLD MODE	RIGHT SHIFT	LEFT SHIFT	ENCODED MODE CONTROL	MULTIPLE CLOCKS	SEPARATE READ/WRITE ADDRESS	READ ENABLE	WRITE ENABLE	CLOCK FREQUENCY (MHz) Min	Typ	Max
REGISTER FILE																			
	74170	4x4		X										X	X	X		10	15
	74172 (Multiport)	8x2		X										X	X	X			20
	74S174 (Multiport)	8x2		X										X	X	X			20
GENERAL						↑													
PURPOSE	8270/74178	4		X	↓			X		X		X				15	22		
SHIFT	82S70/74S178	4		X	↓		X		X	X		X					40	60	
REGISTERS	8271/74179	4		X	↓	X	X		X	X		X					15	22	
	82S71/74S179	4		X	↓	X	X		X	X		X					40	60	
	7495	4		X	↓			X		X	X	X					25	36	
	7496	5		X	↑	X	X			X							10		
	74194	4		X	↑	X	X		X	X	X	X					25	36	
	74S194	4		X	↑	X	X		X	X	X						70	105	
	74195	4		X	↑	X	X		X	X		X					30	39	
	74S195	4		X	↑	X	X		X	X		X							70
	74198	8		X	↑	X		X	X	X	X	X					25	35	
	74199	8		X	↑	X		X	X	X		X					25	35	
	9300	4		X	↑	X		X	X	X		X					30	38	
PARALLEL-IN/																			
PARALLEL-OUT	8200 (Dual)	5		X	↓												15	35	
REGISTERS	8201 (Dual)	5	X		↓												15	35	
	8202	10		X	↓												15	35	
	8203	10	X		↓												15	35	
PARALLEL-IN/																			
SERIAL-OUT	8274	10		X	↓	X	X		X	X		X					25	30	
SHIFT	7494	4			↑	X	X			X							10		
REGISTERS	74165	8	X	X	↑		X		X	X		X					20	26	
	74166 (Serial-in also)	8		X	↑	X		X	X	X		X					25	35	
SERIAL-IN/																			
SERIAL-OUT	8276	8	X	X	↑				X	X							15	20	
SHIFT	8277/9328 (Dual)	8	X	X		X				X			X				15	20	
REGISTERS	7491	8			↑					X							10	18	
SERIAL-IN/																			
PARALLEL-OUT	8273	10		X	↑	X			X				X						
SHIFT	74164	8		X	↑	X											25	36	
REGISTERS																			

LATCHES

FUNCTION	DEVICE	NUMBER OF BITS	R/S INPUT	D INPUT	ENABLE (INPUT)	TRIGGER	NON-INVERTING OUTPUT	INVERTING OUTPUT	RESET	CLOCK	DISABLE (OUTPUT)
LATCHES	8T10	4			X	↑	X		X	X	X
	8275	4		X	X	↑	X				
	7475	4		X		↑	X	X		X	
	74100 (Dual)	4		X		H	X			X	
	74116 (Dual)	4		X	X	L	X		X		
	74174/74S174	6		X		↑	X		X	X	
	74175/74S175	4		X		↑	X	X	X	X	
	74279	4	X								
ADDRESSABLE LATCH	9334	8		X	X	L	X		X		

DECODERS/DRIVERS

FUNCTION	DEVICE	OUTPUT SINK (mA)	OUTPUT SOURCE (mA)	HIGH OUTPUT VOLTAGE (V)	OPEN COLLECTOR OUTPUTS	STANDARD OUTPUTS	ACTIVE LOW OUTPUTS	INHIBIT INVALID INPUTS	INPUT STROBE/ENABLE	ACTIVE HIGH INPUTS
BCD-TO-DECIMAL DECODER	8251					X	X			
	8252/82S52					X	X	X		
	7441	7	2	70	X			X		
	7442					X	X	X		
	9301					X	X			
BCD-TO-DECIMAL DECODER/DRIVER	7445	80	.25	30	X		X	X		
	74145	80	.25	15	X		X	X		

INTRODUCTION

Recent trends in system design are focusing on significantly lower costs and greater reliability while simultaneously maintaining or improving system performance. Consequently, second generation as well as new system designs demand alternatives to conventional logic realizations which substantially reduce the system manufacturing costs. Signetics' System Logic Family provides this alternative by employing state-of-the-art technology to produce Large-Scale-Integrated, system oriented building blocks. By replacing a relatively large number of conventional logic circuits with a few System Logic components, system costs are reduced in proportion to system printed circuit board area, total number of circuits and power requirements. Moreover, system reliability is increased in direct proportion to the decrease in the number of integrated circuits within the system.

SYSTEM LOGIC FAMILY

Signetics' System Logic offers the designer a dual technology logic family to achieve cost objectives. First, where speed is a primary objective, low power Schottky TTL technology is employed to yield devices which feature high speed, low power, and medium density. Therefore, significant reduction in the number of high speed logic components is possible without any sacrifice in performance. The distinct advantages of low power Schottky TTL give it one of the best technologies for high performance LSI design. The particular features of LS are:

- Comparable propagation delay to standard TTL7-10ns/gate average
- Low power dissipation—2mW per gate typical at 50% duty cycle
- Low speed power product—19pj typical
- 45MHz typical maximum J-K flip-flop clock frequency
- High fan-out capability—22 unit load—LS input current requirement is ¼ that of standard TTL (0.36mA/input). Outputs are capable of sinking 8mA.
- High logic density—70 gates/mm^2 of silicon
- Higher system reliability due to reduced power density.

Secondly, while lower power Schottky offers increased density at high speeds, substantial component reduction is achieved with devices employing Integrated Injection Logic (I^{2}L). The very high density and low power of I^{2}L makes it extremely attractive for use in realizing relatively large system building blocks. System level functions may each be accomplished at 10MHz speeds by a single I^{2}L integrated circuit. I^{2}L features are:

- Speed—comparable to T^{2}L (10-20ns propagation delay)
- Power—1-2 orders lower than any technology power down mode
- Density—40% smaller than PMOS—comparable to NMOS
- Interface—capability of mixing T^{2}L, ECL circuits on the same chip.

The System Logic series of products are specially designed to aid system logic designers in the design of high performance, cost effective systems with a minimal number of parts. This is achieved through a line of standard products which are designed with the following objectives:

- **Large Scale Building Blocks**—devices are partitioned by function with high-level complexity and sophistication. These one chip building blocks are equivalent to 400 to 1500 elementary logic gates.
- **Highest Performance Possible**—devices in each category are optimized in performance according to their requirements. Devices requiring highest speed are designed using low power Schottky TTL and devices requiring medium speed are designed using I^{2}L to minimize device power dissipation and maximize logic density.
- **Off-the-shelf items**—system logic devices are designed to allow general purpose usage. Devices in this series can be used as stand-alone items to enhance performance on a certain system design or to utilize several components within the family to design a minimal cost system with minimum number of components, such as in microprocessor based systems.

SIGNETICS CAPABILITIES

Signetics is a leader in the development of bipolar LSI logic circuits using both LS and I^{2}L technologies. Our capabilities are demonstrated by an 8-bit fixed Instruction Microprocessor that Signetics is presently manufacturing with low power Schottky technology.

This circuit, the 8X300 Interpreter, contains the equivalent of 700 logic gates on a 250X250 mil chip. This capability is being used to develop the System Logic Family and produce the substantial cost reduction offered by LSI without paying a penalty in performance.

Signetics is a forerunner in I^{2}L technology. Signetics has been researching I^{2}L as a standard product technology for over three years. Presently, Signetics possesses one of the fastest I^{2}L processes in the industry. Recognizing the vast potential of I^{2}L technology, Signetics is heavily committed to develop this high-speed line of LSI products using this technology. Signetics' I^{2}L process is basically the same process as its low power Schottky process that utilizes a thin expitaxial layer for speed improvement. Since this process is a highly reliable standard process in Signetics, the confidence level of producing such LSI circuits is extremely high. In addition to its own development activities, Signetics has access to N.V. Philips' vast development resources for continued development and enhancement of its present capabilities. This massive investment will result in continued improvement in speed/power performance of I^{2}L products.

Table 1 shows expected trends in I^{2}L speed-/power curves during the next five years. Clearly I^{2}L will play a major role in the innovation of faster and possibly more complex System Logic functions in the support of a continued exploitation of LSI circuitry in systems design.

ADVANTAGES

The advantages of using LSI in system design are manyfold. First, note that the cost of an LSI chip is no more than the sum cost of the circuits it replaces, consequently there is a direct saving in manufacturing costs as a result of a net reduction in the number of parts. Moreover an LSI system offers economic advantages over an SSI or MSI approach besides lowering direct manufacturing costs. Some important considerations are:

- Power supply costs are reduced because logic cells internal to the device require less drive capability and consequently consume less power.
- Field repair costs are reduced because reliability is higher with an LSI implementation. This higher reliability is achieved because IC failure rates are largely proportional to number of devices rather than device complexity.
- Another factor to be considered is that the development cost of printed circuit boards will decrease because of smaller number of ICs and that to some extent, this reduction offsets the cost of layout on the LSI devices.
- Applications requiring medium speed performance are often forced to use lower density, high speed semiconductor devices due to lack of availability of medium speed LSI devices. I^{2}L has changed this picture significantly. With the capability of controlling the amount of device injection current into an LSI circuit, the device can be controlled to operate at the desired speed and power for that particular application; thus, power consumption is minimized.

USING STANDARD LS PROCESS		
Average Propagation Delay	10-20ns at 200μA/Gate Injection Current	
Density (Dual Layer Metal)	Shift Register etc.	320 gates/mm^2
	Random Logic	—150 gates/mm^2
	Single Inverter	—1.5 mil^2
Chip Complexity	2000 gates at maximum speed (Random Logic)	
	4000 gates at maximum speed (Regular Arrays)	
Speed-Power	3pj @ maximum speed	
	0.35pj @ < 1MHz	

Table 1 SIGNETICS I^{2}L CAPABILITY

SYSTEM LOGIC PRODUCT FAMILY

Signetics' System Logic series of products can be categorized into five major groups according to their function. Grouping is as follows:

- Communications
- Memory
- Microprocessors
- Peripheral I/O
- Miscellaneous

Circuits within each group are interrelated in function, thus insuring compatibility both in electrical characteristics and in technology.

Figure 1 illustrates the type of products in the System Logic series. Products with part numbers assigned are either released or under development. Devices with no part numbers assigned are on the product plan. New products will be added to the family tree at regular intervals. A brief description of the System Logic devices is given in the pages that follow.

SYSTEM LOGIC PRODUCT STATUS

New Products

IN PRODUCTION

8X01—Cyclic Redundancy Check Generator/Checker
—150 Gates at 10MHz (Typical)
—I²L Technology

8X02—Control Store Sequencer
—300 Gates
—LS Technology

8X08—Frequency Synthesizer

PLANNED

8X05—Direct Memory Access Control Unit
—1200 Gates at 10MHz
—I²L Technology

8X06—64X8 FIFO
—3000 Gates at 10MHz
—I²L Technology

8X07—64X9 FIFO
—3000 Gates at 10MHz
—I²L Technology

8X —16X8 LIFO
—600 Fates at 10MHz
—I²L Technology

8X —Multiplier
—I²L Technology

8X —Dual Port Register File
—LS Technology

8X —Pipe Lined I/O Port
—LS Technology

8X —Peripheral Interface Unit
—I²L Technology

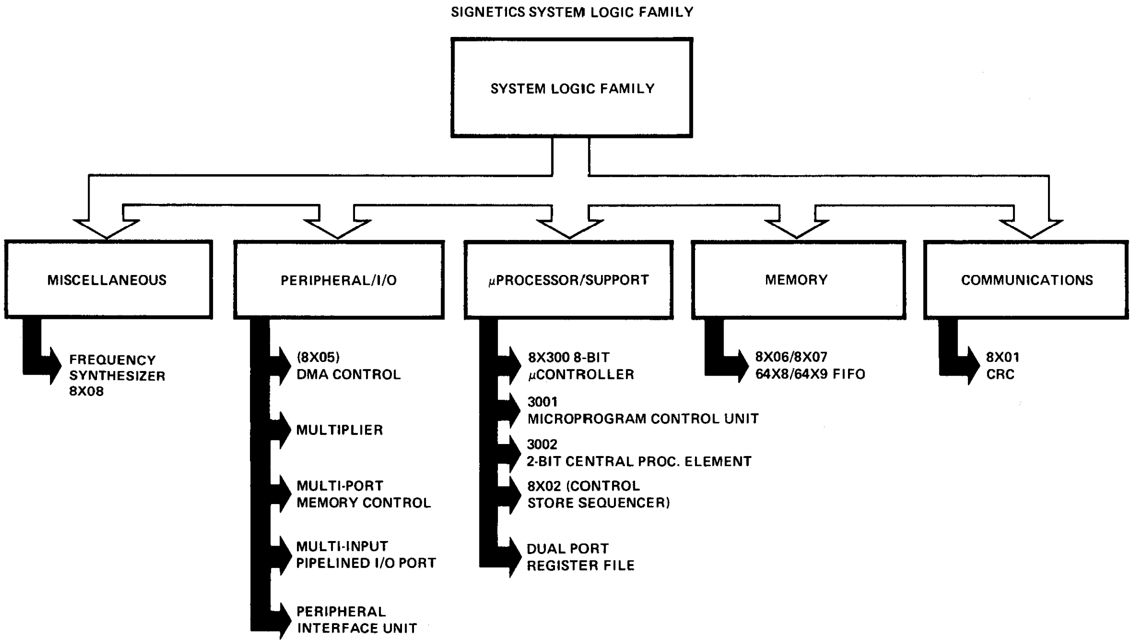

SIGNETICS SYSTEM LOGIC FAMILY
SYSTEM LOGIC FAMILY
MISCELLANEOUS
PERIPHERAL/I/O
μPROCESSOR/SUPPORT
MEMORY
COMMUNICATIONS
FREQUENCY SYNTHESIZER 8X08
(8X05) DMA CONTROL
MULTIPLIER
MULTI-PORT MEMORY CONTROL
MULTI-INPUT PIPELINED I/O PORT
PERIPHERAL INTERFACE UNIT
8X300 8-BIT μCONTROLLER
3001 MICROPROGRAM CONTROL UNIT
3002 2-BIT CENTRAL PROC. ELEMENT
8X02 (CONTROL STORE SEQUENCER)
DUAL PORT REGISTER FILE
8X06/8X07 64X8/64X9 FIFO
8X01 CRC

Figure 1

CRC GENERATOR/CHECKER 8X01

OBJECTIVE SPECIFICATION

8X01-A,F

DESCRIPTION

The CRC Generator/Checker circuit is used to provide an error detection capability for serial digital data handling system. The serial data stream is divided by a selected polynomial and the division remainder is transmitted at the end of the data stream, as a Cyclic Redundancy check character (CRCC). When the data is received, the same calculation is performed. If the received message is error-free, the calculated remainder should satisfy a predetermined pattern. In most cases, the remainder is zero except in the case where Synchronous Data Link Control type protocols are used whereby the correct remainder is checked for 1111000010111000 (x^0 - x^{15}).

8 polynomials are provided and can be selected via a 3-bit control bus. Popular polynomials such as CRC-16 and CCITT are implemented. Polynomials can be programmed to start with either all zeros or all ones.

Automatic right justification for polynomials of degree less than 16 is provided.

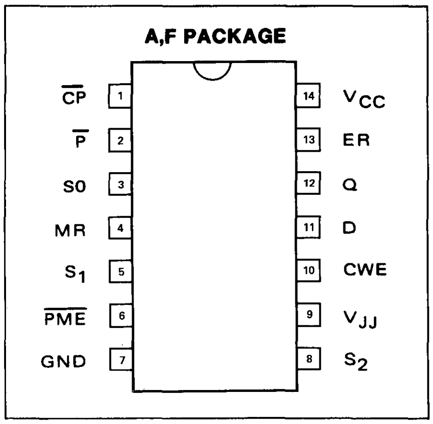
FEATURES

- **I^2L technology**
- **TTL inputs/outputs**
- **10MHz (max) data rate**
- **Total power dissipation = 175mw (max)**
- **Vcc = 5.0V**
- **VJJ = 1.0V**
- **Separate preset and reset controls**
- **SDLC specified pattern match**
- **Automatic right justification**

TYPICAL APPLICATIONS

- **Floppy and other disc systems**
- **Digital cassette and cartridge systems**
- **Data communication systems**

PIN CONFIGURATION



RECOMMENDED OPERATING CONDITIONS

PARAMETER		LIMITS			UNIT
		MIN	TYP	MAX	
V_{CC}	Supply voltage	4.75	5.0	5.25	V
I_{JJ}	Supply current	40		100	mA

FUNCTIONAL DESCRIPTION

The CRC Generator/Checker circuit provides a means of detecting errors in a serial data communications environment. A binary message can be interpreted as a binary polynomial H(x). This polynomial can be divided by a generator polynomial P(x) such that H(x) = P(x) Q(x) + R(x) whereby Q(x) is the quotient and P(x) is the remainder. During transmission, the remainder is appended to the end of the message as check bits. For a given message, a unique remainder is generated. Hardware implementation of division is simply a feedback shift register with Exclusive-OR gating. Subtraction and addition in modulo 2 is implemented by the Exclusive-OR function. The number of shift register stages is equal to the degree of the divisor polynomial.

Table 1 shows the polynomials implemented in the CRC circuit. Each polynomial can be selected via the 3-bit polynomial control inputs S_0, S_1 and S_2. To generate the check bits, the data stream is entered via the Data (D) input, using the high to low transition of the Clock (CP) input. This data is gated with the most significant output (Q) of the register, and controls the exclusive OR gates. The Check Word Enable (CWE) must be held high while the data is being entered. After the last data bit is entered, the CWE is brought low and the check bits are shifted out of the register and appended to the data bits using external gating.

To check an incoming message for errors, both the data and check bits are entered through the D input with the CWE input held high. The 8X01 is not in the data path, but only monitors the message. The Error output becomes valid after the last check bit has been entered into the 8X01 by a high to low transition of CP. If no detectable errors have occurred during the data transmission, the resultant internal register bits are all low and the Error output (ER) is low. If a detectable error has occurred, ER is high. ER remains valid until the next high to low transition of CP or until the device has been preset or reset. PME must be high if ER output is used to reflect all zero result.

For data communications using the Synchronous Data Link Control protocol (SDLC), the 8X01 is first preset to all ones before any accumulation is done. This applies to both transmitter and receiver.

A special pattern of 1111000010111000 (x^0 - x^{15}) is used in place of all zeros during receiving for valid message check. PME is incorporated to select this option. If PME is low during the last bit time of the message, ER output is low if result matches this special pattern. When ER is high, error has occurred.

A high level on the Master Reset (MR) input asynchronously clears the register. A low level on the Preset (P) input asynchronously sets the entire register if the control code inputs specify a 16-bit polynomial; in the case of the 12 or 8-bit check polynomials only the most significant 12 or 8 register bits are set and the remaining bits are cleared.

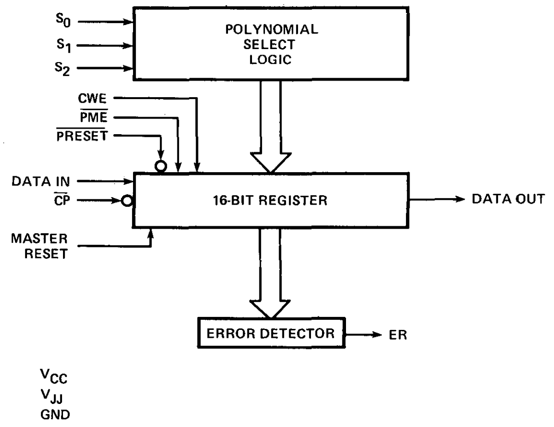
PIN DESIGNATIONS

PIN NO.	FUNCTION
S_0, S_1, S_2	Polynominal Select inputs
D	Data input
$\overline{\text{CP}}$	Clock (operates on high to low transition) input
CWE	Check Word Enable
$\overline{\text{P}}$	Preset (active low) input
MR	Master Reset (active high) input
Q	Data output
ER	Error (active high) output
$\overline{\text{PME}}$	Pattern match enable (active low)

TRUTH TABLE

SELECT CODE			POLYNOMIAL	REMARKS
S_2	S_1	S_0		
L	L	L	$X^{16}+X^{15}+X^2+1$	CRC-16
L	L	H	$X^{16}+X^{14}+X+1$	CRC-16 REVERSE
L	H	L	$X^{16}+X^{15}+X^{13}+X^7+X^4+X^2+X^1+1$	
L	H	H	$X^{12}+X^{11}+X^3+X^2+X+1$	CRC-12
H	L	L	$X^8+X^7+X^5+X^4+X+1$	
H	L	H	X^8+1	LRC-8
H	H	L	$X^{16}+X^{12}+X^5+1$	CRC-CCITT
H	H	H	$X^{16}+X^{11}+X^4+1$	CRC-CCITT REVERSE

LOGIC DIAGRAM

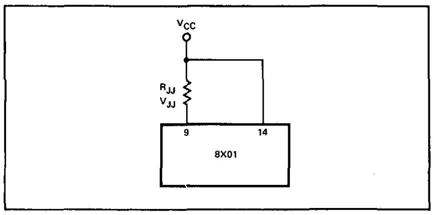


DC CHARACTERISTICS OVER OPERATING TEMPERATURE RANGE

(Unless Otherwise Noted)

PARAMETER		TEST CONDITIONS	LIMITS			UNIT
			MIN	TYP	MAX	
V_{IH}	Input high voltage		2.0			V
V_{IL}	Input low voltage				0.8	V
V_{IC}	Input clamp diode voltage	Vcc = MIN, I_{IN} = 18mA			-1.5	V
V_{OH}	Output high voltage	Vcc = MIN, I_{OH} = 400μA	2.7			V
V_{OL}	Output low voltage	Vcc = MIN, I_{OL} = 8mA			0.5	V
I_{IH}	Max. input current	Vcc = MAX			0.1	mA
I_{IH}	Input high current	VCC = MAX, VIN = 2.7V			20	μA
I_{IL}	Input low current	VCC = MAX, VIN = 0.4V			-0.36	mA
I_{OS}	Output short circuit current	VCC = MAX, VOUT = 0V	-10		-42	mA
I_{JJ}	Injection current	Vcc = MAX, Inputs open		60	100	mA
I_{CC}	Supply current	Vcc = MAX, inputs open		10	18	mA

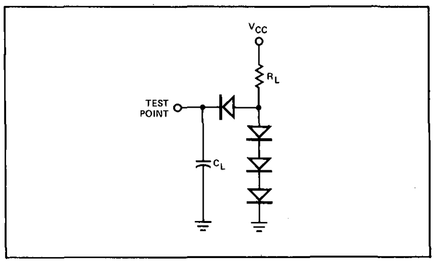
I²L INJECTOR CURRENT SOURCE



$$R_{JJ} = \frac{V_{CC} - V_{JJ}}{I_{jj}} = \frac{(5.0 - 0.8)\,V}{60mA}$$

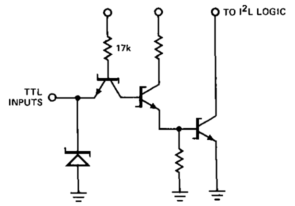
$$= \frac{4.20V}{60mA} = 700\Omega$$

TEST CIRCUIT

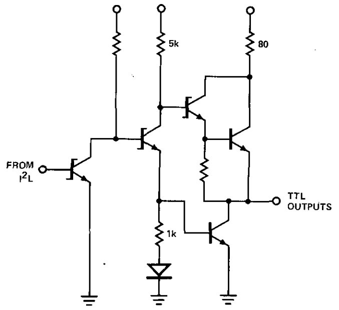


INPUT/OUTPUT CIRCUITS

INPUT STRUCTURE



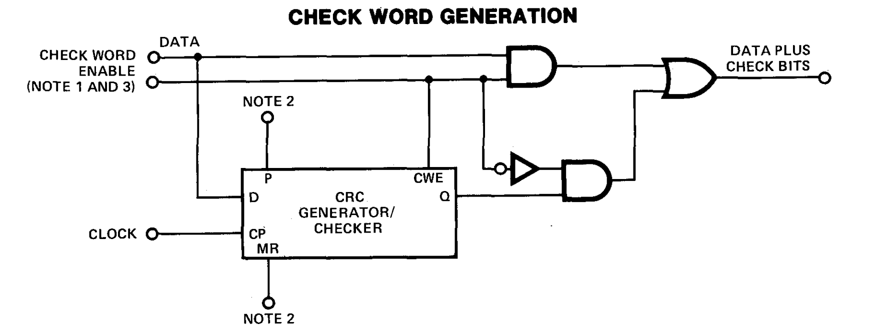
OUTPUT STRUCTURE



NOTE: ALL RESISTORS VALUES ARE TYPICAL AND IN OHMS.

SWITCHING CHARACTERISTICS $T_A = 25°C, I_{JJ} = 60mA$

PARAMETER		TEST CONDITIONS		LIMITS V_{CC}=5V			LIMITS V_{CC}=4.5V			UNITS
				MIN	TYP	MAX	MIN	TYP	MAX	
$t_wCP(L)$	Clock pulse width (low)	Fig. 2		30						ns
t_sD	Setup time, Data to Clock	Fig. 5			60	75				ns
t_sCWE	Setup time, CWE to Clock	Fig. 5			45	65				ns
t_n	Hold time, Data, CWE to Clock	Fig. 5	C_L = 15pF		0					ns
$t_wP(L)$	Preset pulse width (low)	Fig. 3		40						ns
$t_wMR(H)$	Master reset pulse width (high)	Fig. 5		40						ns
t_{REC}	Recovery time, MR, Preset to Clock	Fig. 3,4			60	90				ns
f_{max}	Maximum clock frequency							8	10	MHz
	propagation delay									
t_{PLH},t_{PHL}	Clock, MR, Preset to Data output	Fig. 2,3,4	C_L = 15pf					80	95	ns
	propagation delay									
t_{PLH},t_{PHL}	Clock, MR, Preset to Error Output		C_L = 15pf					110	125	ns



NOTES:
1. Check word Enable is HIGH while data is being clocked, LOW during transmission of check bits.
2. 8X01 must be reset or preset before each computation
3. CRC check bits are generated and appended to data bits.

Figure 2

VOLTAGE WAVEFORMS

PROPAGATION DELAYS $\overline{CP}$ TO Q AND $\overline{CP}$ TO ER

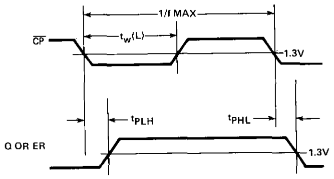


Figure 3

PROPAGATION DELAYS, $\overline{P}$ TO Q AND ER PLUS RECOVERY TIME P TO $\overline{CP}$

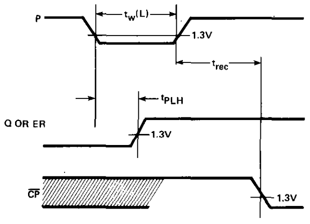


Figure 4

PROPAGATION DELAYS, MR TO Q AND ER PLUS RECOVERY TIME, MR TO $\overline{CP}$

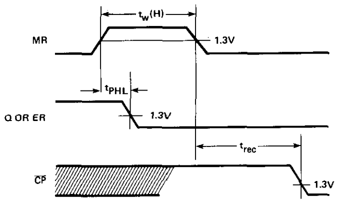


Figure 5

SET UP AND HOLD TIMES D TO $\overline{CP}$ and CWE TO $\overline{CP}$

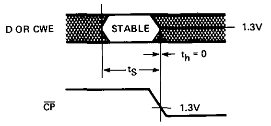


Figure 6

DESCRIPTION

This LSI integrated circuit performs the digital control functions required for generating AM/FM radio frequency local oscillator signals using digital phase locked loop techniques. By the use of low power Schottky and ECL technologies on the same substrate it is possible to operate at 80MHz input frequencies with an average system power of 1.6mW per gate typical.

PHASE LOCKED LOOP PRINCIPLES

Digital phase locked loops are comprised of 4 basic building blocks: A fixed reference frequency generator (crystal oscillator and divider), a phase comparator, a voltage controlled oscillator (VCO) and a programmable counter (÷N).

In cases where very high frequencies must be generated, a fixed prescaler (÷M) is employed to divide the local oscillator frequency down to a frequency compatible with the programmable counter. F_{out} from the VCO is divided down by the prescaler and programmable counters and compared to the reference frequency by the phase detector. If $\frac{F_{out}}{MN}$ is not equal to F_{ref} in phase and frequency, the phase detector generates a signal which causes the VCO frequency to increase or decrease until $F_{ref} = \frac{F_{out}}{MN}$, when this occurs, the local oscillator is essentially as stable as the crystal reference oscillator.

The local oscillator frequency (F_{out}) is changed by programming a different number into the programmable counter. The distance between discrete frequencies or the channel spacing is determined by the reference frequency.

FEATURES

- **80MHz input frequency**
- **ECL prescaler**
- **LS process**
- **Single 5V supply**
- **Power dissipation—600mW (max)**
- **External components—**
  - **1 crystal**
  - **2 capacitors**

For the AM/FM circuit, up to 200 channels are possible with selectable channel spacing of 10kHz for AM operation and 2000 channels at 100kHz for FM operation.

AM/FM Frequency Synthesizer Circuit Description

The frequency synthesizer circuit logic diagram is shown below. Following is a description of each of the major blocks.

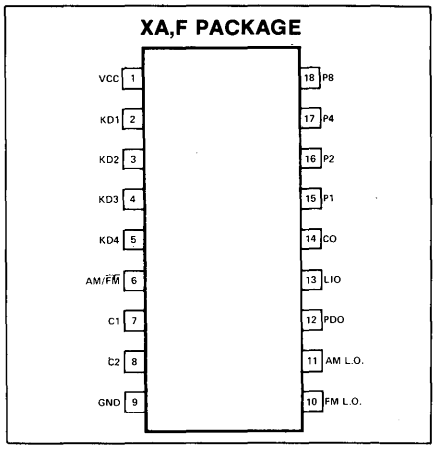
Programmable Counter

The programmable counter consists of 3 stages of decade counter plus a divide by 1 or 0 counter to divide by numbers up to 1999. BCD programming data is presented to the dividers in parallel form, one digit at a time. Parallel data is strobed into internal latches via strobe signals; one strobe for each digit. A ÷ 5 80MHz ECL prescaler precedes the programmable counter for FM operation. This prescaler plus an external 160MHz ÷ 2 flip-flop provide a ÷ 10 160MHz prescaler (÷M) function to scale the programmable counter input frequency down to 16MHz maximum. A logic control circuit bypasses the ÷M prescaler and the first decade counter for AM operation. By this technique, the channel spacing is programable to 10kHz for AM operation and 100kHz for FM operation.

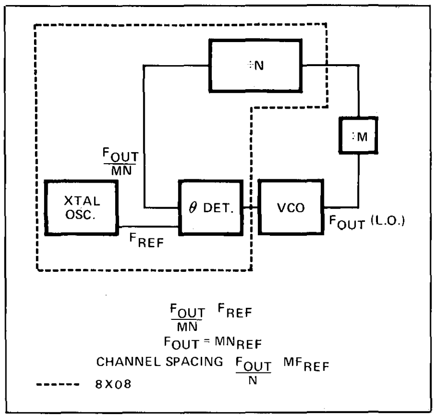
VCO

An externally provided integrator and voltage controlled oscillator must be provided to perform the complete frequency synthesizer function. The integrator converts the pulses that come from the phase detector into a dc signal that controls the output frequency of the voltage controlled oscillator. It is in the integrator part of the circuit that the critical loop constants are determined. The voltage controlled oscillator is normally a LC tuned oscillator with varactor diode tuning that is controlled by the dc signals from the integrator. In this case, two are required; one for the AM band and one for the FM band. The FM oscillator output must be +5V ECL compatible while the AM oscillator must be TTL compatible.

PIN CONFIGURATION



PHASE LOCKED LOOP BLOCK DIAGRAM

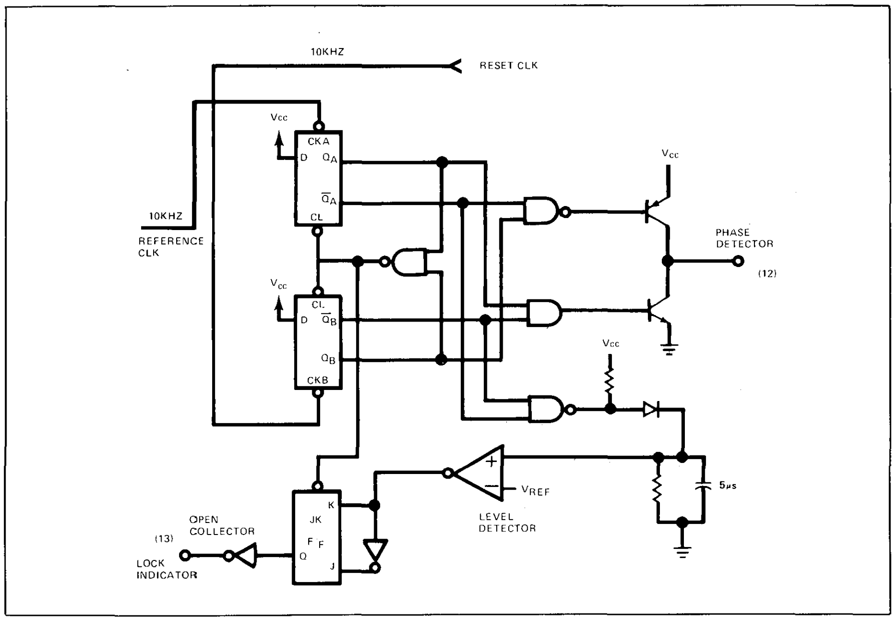


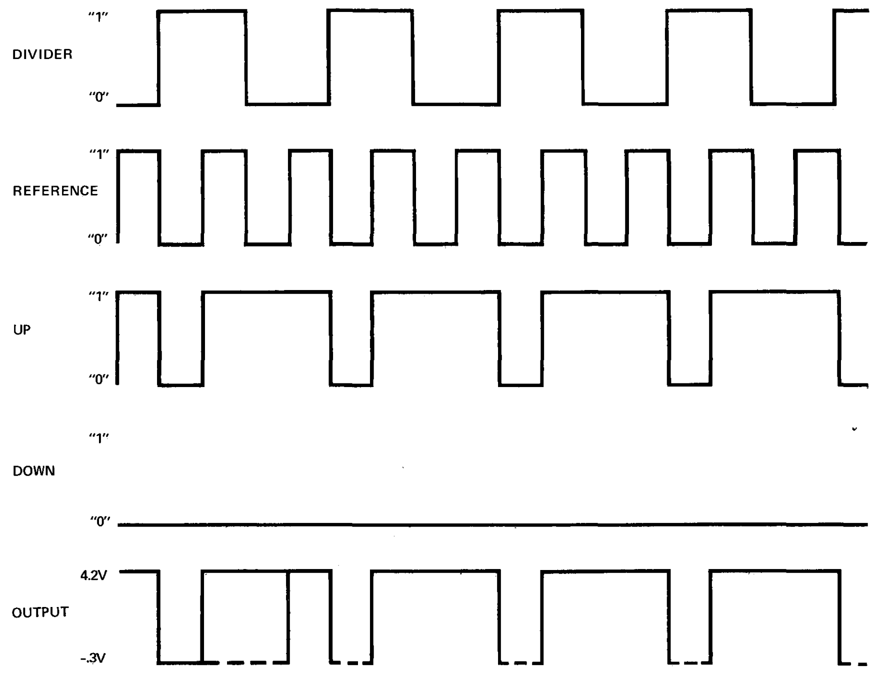
Phase Detector Circuit

The phase detector is a digital edge-detecting device that provides an output three-state signal that is in a high impedance state when the 2 input signals are equal in phase and/or frequency. The output of the phase detector is a series of pulses that swing from the high impedance state to .3V typical or from the high impedance state to 4.2V typical. If the positive edge of the divider input leads the reference, the pulses will go to 4.2V. If it lags they will go to .3V.

The width of the output pulses is a function of the time between the positive edges (phase) of the 2 signals. An example of the operation of the device is shown where the reference signal is twice the frequency of the divider signal and has a phase lead of 270°. The output pulses are converted to a dc signal by integrating amplifiers causing the output frequency of the voltage controlled oscillator to increase or decrease (increase in this case) until the divider output and the reference output are equal in phase and frequency.

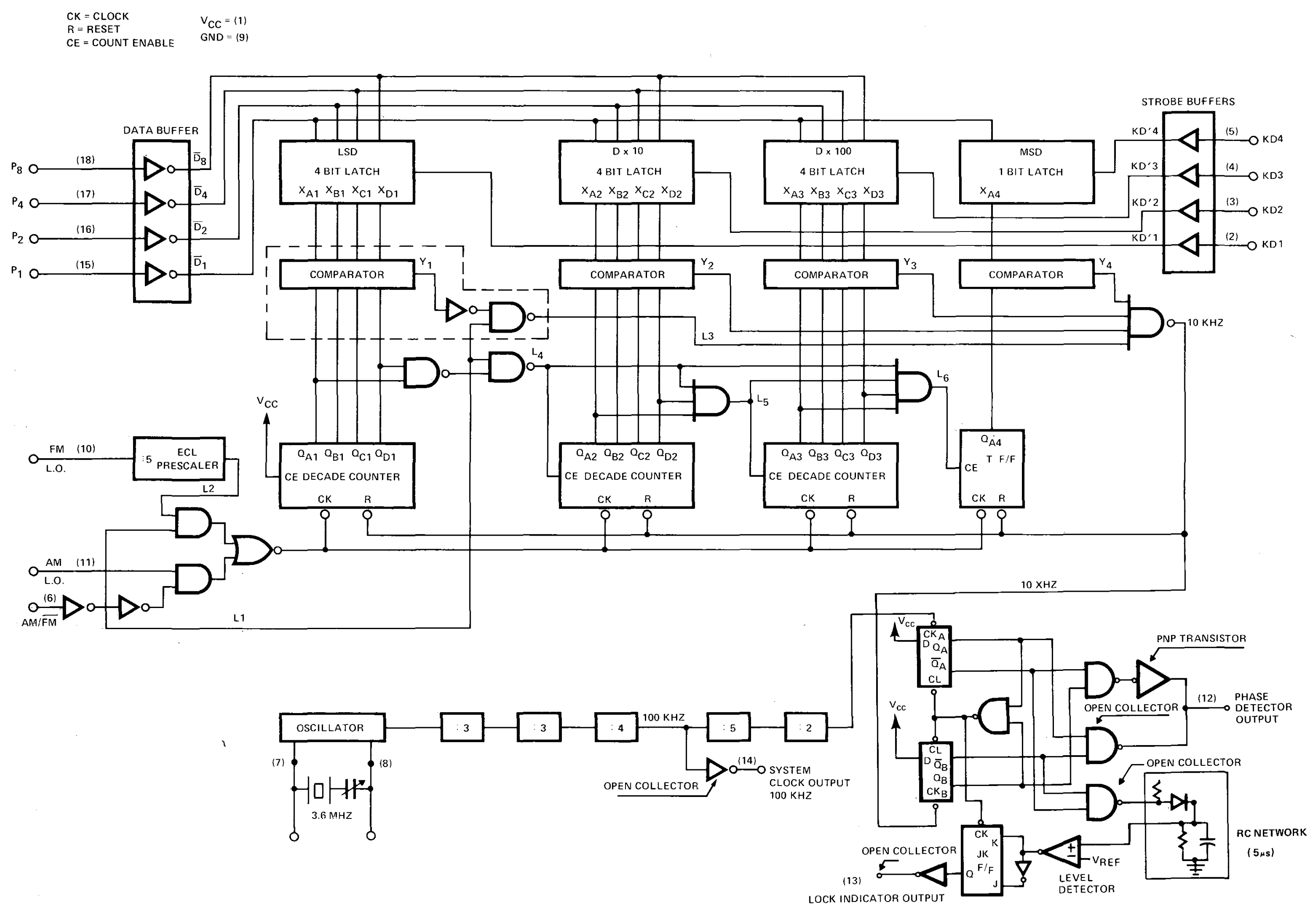
PHASE DETECTOR CIRCUIT





NOTE: — — — IS HIGH IMPEDANCE STATE.

LOGIC DIAGRAM



RECOMMENDED OPERATING CONDITIONS

PARAMETER		LIMITS			UNIT
		Min	Typ	Max	
T_A	Operating free air temperature	-40		+85	°C
V_{CC}	Supply voltage	4.75	5.0	5.25	V
	Max AM local oscillator input operating frequency (Pin 11)		20		MHz
	Max FM local oscillator input operating frequency (Pin 10)		100	80	MHz
	Maximum reference frequency oscillator operating frequency		10		MHz

ELECTRICAL CHARACTERISTICS OVER RECOMMENDED OPERATING CONDITIONS

PARAMETER		TEST CONDITIONS	LIMITS			UNITS
			Min	Typ	Max	
V_{IH}	High level input voltage					
	P, K_D, AM/$\overline{FM}$ inputs		5.25			V
	AM L.O. input		2			V
	FM L.O. input		4.1		5.25	V
V_{IL}	Low level input voltage					
	P, K_D, AM/$\overline{FM}$ inputs				3.75	V
	AM L.O. input				0.8	V
	FM L.O. input				3.3	V
V_I	Input current at maximum input voltage					
	P, K_D, AM/$\overline{FM}$ inputs	V_{CC} = max, V_I = 16V			200	μA
		V_{CC} = max, V_I = 5.25V			40	μA
	AM L.O. input (with 5kΩ pullup to V_{CC})	V_{CC} = max, V_I = 5.25V			200	μA
	FM L.O. input	V_{CC} = max, V_I = 5.25V			400	μA

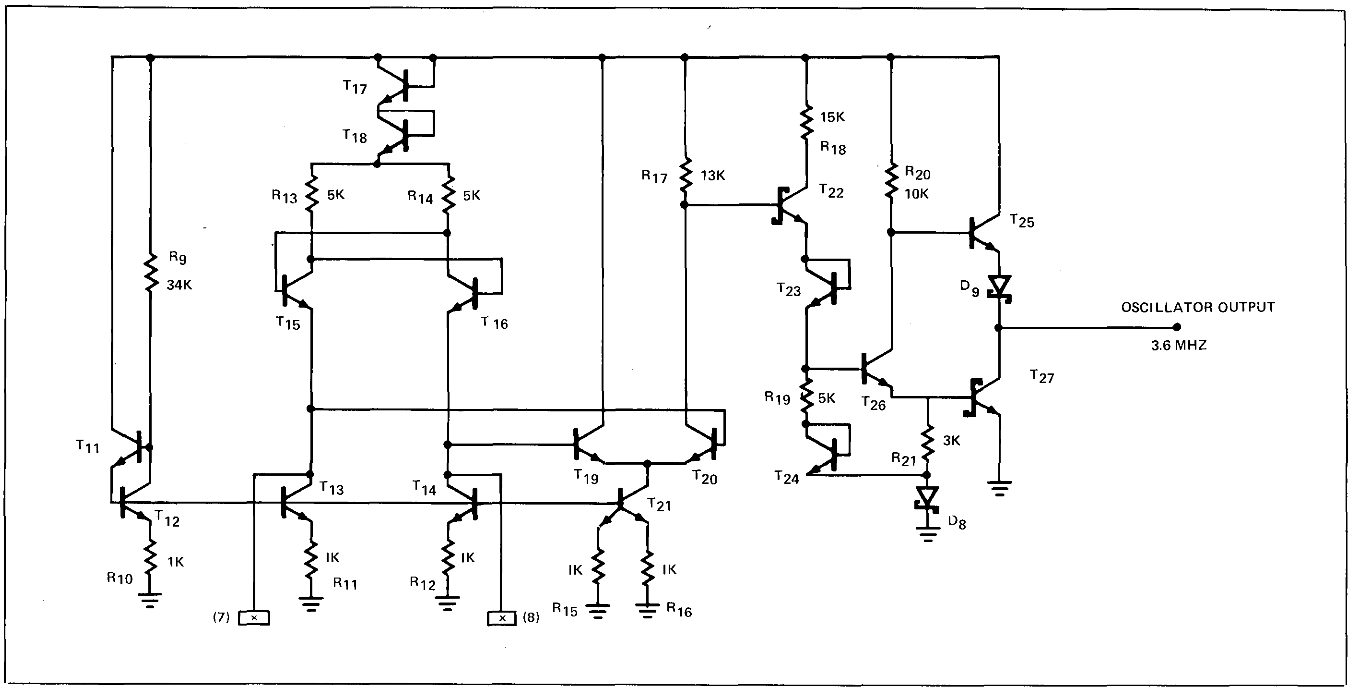
DC ELECTRICAL CHARACTERISTICS

PARAMETER		TEST CONDITIONS	LIMITS			UNITS
			Min	Typ	Max	
I_{IL}	Low level inputs current					
	P, K_D, AM/$\overline{FM}$ inputs	V_{CC} = max, V_I = 3.75V			-40	μA
	AM L.O. input (with 5kΩ pullup to V_{CC})	V_{CC} = max, V_I = 0.4V	-.7		-1.6	mA
	FM L.O. input	V_{CC} = max, V_I = 0.4V			-40	μA
V_{OL}	Low level output voltage					
	System clock output	V_{CC} = min, I_{OL} = 16mA			0.8	V
	Lock indicator output	V_{CC} = min, I_{OL} = 16mA			0.8	V
	Phase detector output	V_{CC} = min, I_{OL} = 40μA			0.5	V
V_{OH}	High level output voltage					
	Phase detector output	V_{CC} = min, I_{OH} = -40μA	V_{CC}-0.5V			
	High level output current					
	System clock output	V_{CC} = min, V_{OH} = 16V			250	μA
	Lock indicator output	V_{CC} = min, V_{OH} = 16V			250	μA
I_{CC}	Supply current	V_{CC} = max			130	mA

Crystal Oscillator Circuit

In this circuit, the cross-coupled transistor pair form a bistable circuit. The crystal provides positive feedback between the emitters of T_{15} and T_{16} which causes the circuit to oscillate at the crystal frequency.

CRYSTAL OSCILLATOR CIRCUIT



ANALOG

DEVICE	DESCRIPTION	Commercial	Military /883B	Data Book Page Ref.
	CONSUMER/COMMUNICATION CIRCUITS			
NE/SE540	Power Driver	●		233
NE541	Power Driver	●		N/A
NE542	Dual Preamp	●		241
NE543	Servo Amplifier	●		245
NE544	Servo Amplifier	●		N/A
NE546	AM Radio	●		247
NE570/571	Analog Compandor	●		N/A
NE/SE5596	Balanced Modulator/Demodulator	●		305
µA758	Stereo Decoder	●		292
ULN2111	FM Detector/Limiter	●		307
ULN2208	FM Gain Block	●		315
ULN2209	FM Gain Block	●		318
TBA120S/u	TV Sound IF	●		321
TBA1440	TV Video IF	●		324
TCA440	AM Radio	●		N/A
TBA327/395/396	TV PAL Chroma Set	●		N/A
CA3089	FM IF System	●		N/A
PA239	Dual Preamp	●		307
LM381/381A	Dual Preamp	●		296
LM382	Dual Preamp	●		299
LM387	Dual Preamp	●		302
MC1496-1596	Balanced Modulator/Demodulator	●		305
	PHASE LOCKED LOOPS			
NE/SE560	Phase Locked Loop	●		257
NE/SE561	Phase Locked Loop	●		262
NE/SE562	Phase Locked Loop	●		267
NE/SE564	Phase Locked Loop	●		N/A
NE/SE565	Phase Locked Loop	●		274
NE/SE566	Function Generator	●		279
NE/SE567	Tone Decoder PLL	●	●	292
	OP AMPS			
NE/SE531	High Slew Rate Op Amp	●		42
NE/SE/SA532	Dual Op Amp	●	●	47
NE/SE535	High Slew Rate Op Amp	●		50
NE/SU536	FET Input Op Amp	●		51
MC1456/1556	High Performance Op Amp	●		85
MC1458/1558	Dual Op Amp	●	●	87
µA709/709C	Op Amp	●	●	89
µA740/740C	FET Input Op Amp	●		91
µA741/741C	General Purpose Op Amp	●	●	93
µA747/747C	Dual Op Amp	●	●	96
µA748/748C	General Purpose Op Amp	●	●	100
LM101/201	High Performance Op Amp	●	●	56
LM101A/201A/301A	High Performance Op Amp	●	●	60
LM107/207/307	General Purpose Op Amp	●	●	69
LM108/208/308	Precision Op Amp	●	●	72,76
LM108A/208A/308A	Precision Op Amp	●	●	80
LM124/224/324	Quad Op Amp	●		82
SA1458	Dual Op Amp	●		87
SA534	Quad Op Amp	●		103
SA709	Op Amp	●		89
SA741	General Purpose Op Amp	●		93
SA747	Dual Op Amp	●		96
	TIMERS			
NE/SE/SA553	Quad Timer	●		155
NE/SE/SA554	Quad Timer	●		155
NE/SE/SA555	Timer	●	●	158
NE/SE/SA556	Dual Timer	●		162
NE557	Unijunction Oscillator	●		N/A

ANALOG

DEVICE	DESCRIPTION	Commercial	Military /883B	Data Book Page Ref.
	MOSFET-RF (D-MOS)			204,
SD200/201	Single Gate UHF	●		206
SD202/203	Single UHF	●		206
SD300/301	Dual Gate UHF	●		212
SD303	Dual Gate UHF	●		212
SD304	Dual Gate VHF	●		212
SD305	Dual Gate VHF Mixer	●		218
SD306	Dual Gate VHF Amp	●		221
SD307	Dual Gate UHF Mixer	●		N/A
SD308	Dual Gate UHF Amp	●		N/A
SD6000	VHF Mixer/Amp IC	●		228
	MOSFET-ANALOG/DIGITAL SWITCHES (D-MOS)			
SD210/211	Switch/Driver	●		209
SD212/213	Switch	●		209
SD214/215	Switch	●		209
SD5000/5001	Quad Switch Array IC	●		224
SD5100/5101	Quad Multiplexer IC	●		224
SD5200	Quad Switch Driver IC	●		224
SD5300	8x2 Crosspoint Switch	●		N/A
	VIDEO AMPLIFIERS			
NE/SE501	Video Amp	●		166
NE/SE592	Video Amp	●		167
µA733/733C	Video Amp	●	●	172
	DIFFERENTIAL AMPLIFIERS			
NE/SE510	Dual Differential Amp	●	●	106
NE/SE511	Dual Differential Amp	●	●	107
NE/SE515	Differential Amp	●	●	109
	VOLTAGE REGULATORS			
LM109/209/309	5V Power Voltage Regulator	●		147
NE/SE550	Precision Voltage Regulator	●		112
µA723/723C	Precision Voltage Regulator	●	●	116
SA723	Precision Voltage Regulator	●		
7805/06/08/12/15/18/24	Three Terminal Positive Voltage Regulators	●		122
78L02/05/06/12/15	Three-Terminal Positive Voltage Regulators	●		130
78M05/06/08/12/15/20/24	Three-Terminal Positive Voltage Regulators	●		139
LM340-05/06/08/12/15/18/24	Voltage Regulators	●		151
LM341-05/06/08/12/15/18/24	Voltage Regulators	●		N/A
	COMPARATORS			
NE521/522	High Speed Dual Differential Comparator/Sense Amp	●		177
NE/SE526	Analog Voltage Comparator	●		183
NE/SE527	Analog Voltage Comparator	●		184
NE/SE529	Analog Voltage Comparator	●		187
µA710/710C	Differential Voltage Comparator	●	●	202
µA711/711C	Dual Comparator	●	●	203
LM111/211/311	Comparator	●		190, 192
LM119/219/319	Dual Comparator	●		194, 195
LM139/239/339	Quad Comparator	●	●	197
LM139A/239A/339A	Quad Comparator	●	●	198
MC3302	Quad Comparator	●		200
SA539	Quad Comparator	●		N/A

CHAPTER 4
APPLICATIONS

INTRODUCTION

The ability of the semiconductor industry to manufacture complete general purpose processors on single chips represents a significant technological advance which should prove to be of great benefit to digital systems manufacturers. In terms of chip size and density of transistors, the processors are simply extensions of the continually evolving MOS technology. But in terms of function provided, a significant threshold has been crossed.

By allowing designers to convert from hardware logic to programmed logic, the integrated processor provides several important advantages.

1. Logic functions may be implemented in memory bits instead of logic gates. The user then has greater access to the advantages of memory circuits. Memories use patterned circuitry and thus provide greater density and therefore greater economy.
2. Random logic implementations of complex functions are highly specialized and cannot be used in other applications. They are not often used in large volume. Programmed logic, on the other hand, relies on general purpose processor and memory circuits that are used in many applications. Thus, economies of volume are available for both the user and the manufacturer.
3. Because the functional specialization resides in the user's program rather than the hardware logic, changes, corrections and additions can be much easier to make and can be accomplished in a much shorter time.
4. With the programmed logic approach it is often possible to add new features and create new products simply by writing new programs.
5. The design cycle of a system using programmed logic can be significantly shorter than a similar system that attempts to use custom random logic. The debugging cycle is also greatly compressed.

A general purpose processor designed to implement programmed logic has many characteristics that allow it to do conventional computer operations as well. Many applications will specialize in programmed logic or in data processing, but some will take advantage of both areas. In a line printer application, for example, a processor can act primarily as a controller handling the housekeeping duties, control sequencing and data interfacing for the printer. It also might buffer the data or do some code conversions, but that is not its primary duty. On the other hand, in a text editing intelligent terminal, the processor is mainly concerned with data manipulation since it handles code translations, display paging, insertions, deletions, line justification, hyphenation, etc.

A point-of-sale type of terminal represents an application that combines both control and data processing activities for the processor. Coordinating the activities of the various devices and displays that make up the terminal is an important part of the job, as are the calculations that are essential to the operation of the machine.

Because of the diversity of application areas, a single microprocessor type cannot effectively service all applications. Signetics offers a variety of microprocessor families, each of which has specific benefits in the various application areas.

This chapter contains hardware and software application memos for the 8X300, 3000 families. Additional application memos are in preparation. The services of a field applications engineer or the factory microprocessor applications staff are also available.

A GUIDE TO THE SELECTION OF SUPPORT COMPONENTS FOR THE SERIES 3000 MICROPROCESSOR

INTRODUCTION

The Signetics Series 3000 Bipolar Microprocessor product line consists of the N3001 MCU (Microprogram Control Unit) and the N3002 CPE (Central Processing Element). These two devices, using low-power Schottky technology, can be readily and efficiently interfaced with all other industry standard components, including the 7400, 74S, 74LS and Signetics 82S and 8T families.

Figure 1 is a generalized functional block diagram of a typical Bipolar Microprocessor based system. This applications memo categorizes various components that can be used to implement each of the major functional blocks shown in Figure 1.

THE PROCESSING SECTION

The Processing Section of a computer, or "smart" controller, as shown in Figure 2, provides facilities for the transfer of data, logical and arithmetic manipulation of data, and temporary storage of data, address and status information. Cache memories are frequently used to enhance system performance by providing immediately available information and data to the CPU at high speed.

Signetics devices that can be used to implement the Processing Section of the CPU or controller are listed below:

ALU, General Purpose Registers, and Carry-Look-Ahead circuits

DEVICE	DESCRIPTION
N3002	Central Processing Element
74S182	Carry-Look-Ahead Generator

High Speed Bipolar Cache Memories

DEVICE	DESCRIPTION
3101A	RAM (16 words by 4 bits)
82S25	Write-While-Read RAM (32 words by 2 bits)
82S25	RAM (16 words by 4 bits)
82S09	RAM (64 words by 9 bits)
82S116/117	RAM (256 words by 1 bit)
74S200/201	RAM (256 words by 1 bit)
74S301	RAM (256 words by 1 bit)
93415A	RAM (1024 words by 1 bit)
93425A	RAM (1024 words by 1 bit)
82S10/11	RAM (1024 words by 1 bit)

BLOCK DIAGRAM OF TYPICAL BIPOLAR MICROPROCESSOR SYSTEM

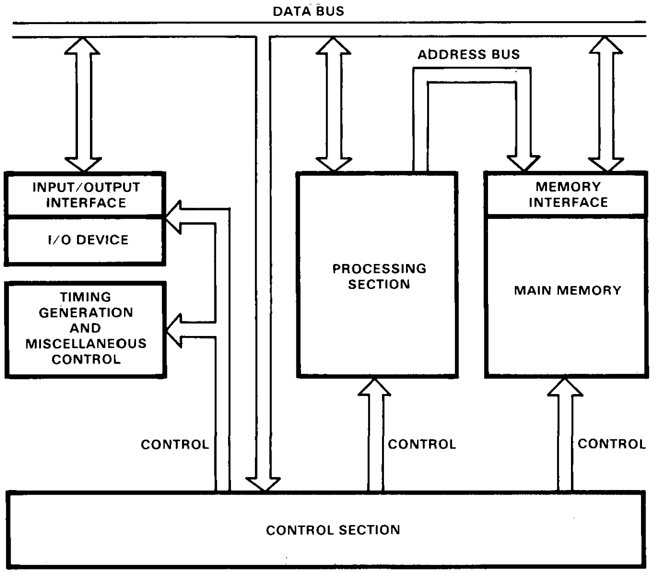


Figure 1

THE PROCESSING SECTION

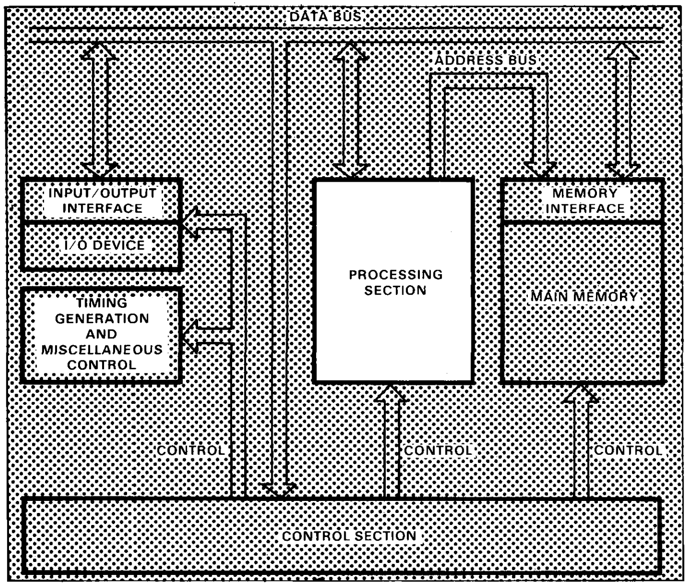


Figure 2

THE CONTROL SECTION

The Control Section logic as shown in Figure 3, handles most, if not all, of the control functions. These functions are typified by operations such as decoding macro-instructions, testing of hardware or program status, initializing memory and I/O operations, manipulating data, and sampling and responding to external and internal interrupts. These control functions are implemented by executing a single microinstruction or a series of microinstructions. Through microprogramming, a structured form of control can be realized.

Macro Instruction Decoding

DEVICE	DESCRIPTION
82S100/101	Field Programmable Logic Array (FPLA)

NOTE

In addition to the FPLA, all ROMs and PROMs listed below under Conditional Test and Branch Decoding and Microprogram Memory can also be used.

Conditional Test and Branch Decoding

DEVICE	DESCRIPTION
82S123	PROM (32 words by 8 bits)
82S23	PROM (32 words by 8 bits)
82S229	PROM (256 words by 4 bits)
82S226	ROM (256 words by 4 bits)
82S129	PROM (256 words by 4 bits)
82S126	PROM (256 words by 4 bits)
82S27	PROM (256 words by 4 bits)

Microprogram Sequencing

DEVICE	DESCRIPTION
N3001	Microprogram Control Unit (512-word addressability)
8X02	Control Store Sequencer (1024-word addressability)

NOTE

In addition, all ROMs and PROMs listed in Conditional Test and Branch Decoding can also be used.

THE CONTROL SECTION

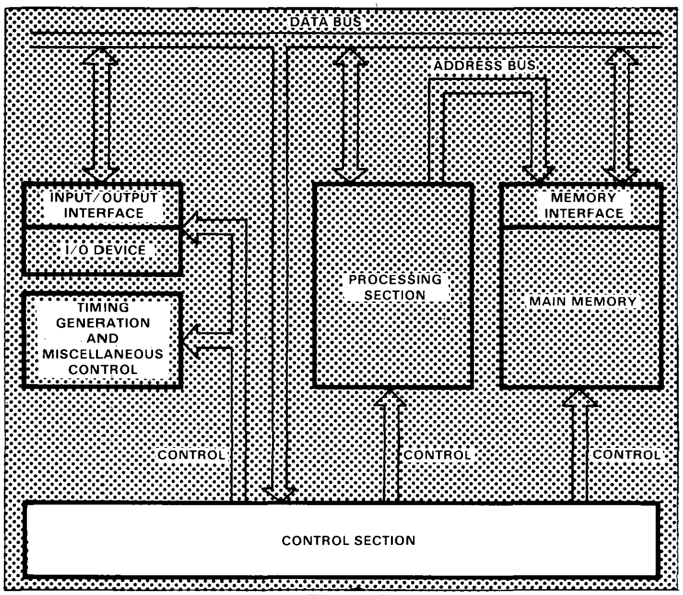


Figure 3

Microprogram Memory

DEVICE	DESCRIPTION
82S130	PROM (512 words by 4 bits)
82S131	PROM (512 words by 4 bits)
82S114	PROM with output latches (256 words by 8 bits)
82S214	ROM with output latches (256 words by 8 bits
82S215	ROM with output latches (512 words by 8 bits)
82S230/231	ROM (512 words by 4 bits)
8228	ROM (1024 words by 4 bits)
82S115	PROM with output latches (512 words by 8 bits)
82S136/137	PROM (1024 words by 4 bits)
82S236/237	ROM (1024 words by 4 bits)
82S280/281	ROM (1024 words by 8 bits)
82S184/185	PROM (2048 words by 4 bits)
82S284/285	ROM (2048 words by 4 bits)

I/O INTERFACE LOGIC

The are many different types of bus structures (See Figure 4). To save hardware and the number of signal lines, the use of a bidirectional bus is an excellent solution for handling the data bus problem, while a tristate bus structure is ideal for the address bus. In many instances, when systems of different logic families need to be interfaced with each other, logic level translators are required.

Bus buffers and drivers

DEVICE	DESCRIPTION
8T26A	Tri-state, inverting quad bus transceiver.
8T28	Tri-state, non-inverting quad bus transceiver.
8T31	Tri-state 8-bit bidirectional I/O port.
8T09	Tri-state quad bus driver (40mA), with individual enable/disable
8T10	Tri-state quad D-type bus (FF) with high drive capability
8T13	Dual-line Driver
8T14	Triple line receiver with hysteresis
8T15	Dual Communications EIA/MIL line driver
8T16	Dual Communication EIA/MIL line receiver with hysteresis.
8T23	Dual-line driver
8T24	Triple-line receiver with hysteresis
8T38	Quad bus transceiver
8T95	Tri-state, non-inverting hex buffer
8T96	Tri-state, non-inverting hex buffer
8T97	Tri-state, inverting hex buffer
8T98	Tri-state, inverting hex buffer
8T100	Quad differential line drivers
8T110	Quad differential line receivers

I/O INTERFACE LOGIC

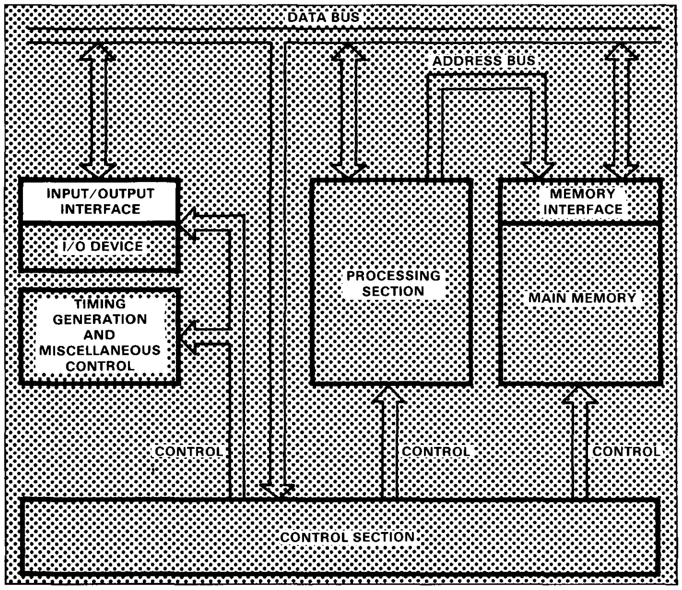


Figure 4

Logic level translators

DEVICE	DESCRIPTION
10124	Quad TTL-to-ECL driver
10125	Quad ECL-to-TTL receiver
8T80	Quad gate TTL to High-voltage
8T90	Hex buffer TTL to High-voltage
8T25	MOS-to TTL translator
8T18	High-voltage to TTL

MEMORY INTERFACE AND MAIN MEMORY

When dynamic MOS memory devices are used as main memory, see Figure 5, a memory refresh scheme will be needed. Usually sense amplifiers, address and data buffers, and parity generators and checkers are considered as part of the memory interface logic.

Sense amplifiers and drivers

DEVICE	DESCRIPTION
7520	Dual core-memory sense amplifiers
7521	Dual core-memory sense amplifiers
7522	Dual core-memory sense amplifiers
7523	Dual core-memory sense amplifiers
7524	Dual core-memory sense amplifiers
7525	Dual core-memory sense amplifiers
3207A-1	Quad TTL-MOS clock drivers
3207A	Quad TTL-MOS clock drivers
8T09	Tri-state quad bus driver
8T380	Tri-state bus receiver, with:
8T25	Tri-state dual sense amplifier/latch

Parity generator and checker

DEVICE	DESCRIPTION
8262	9-bit parity
74180	8-bit parity

Memory refresh logic

DEVICE	DESCRIPTION
8281	Binary counter
8291	Binary counter
74123	Dual one-shot
82S33	Quad 2-to-1 multiplexer
82S34	Quad 2-to-1 multiplexer

MOS Memory

DEVICE	DESCRIPTION
1103	1K (1024X1) RAM
1103-1	1K (1024X1) RAM
2602-1	1K (256X4) RAM, static
2606	1K (256X4) RAM, static
2102	1K (1024X1) RAM
2680	4K (4096X1) RAM
2660	4K (4096X1) RAM

MEMORY INTERFACE/MAIN MEMORY

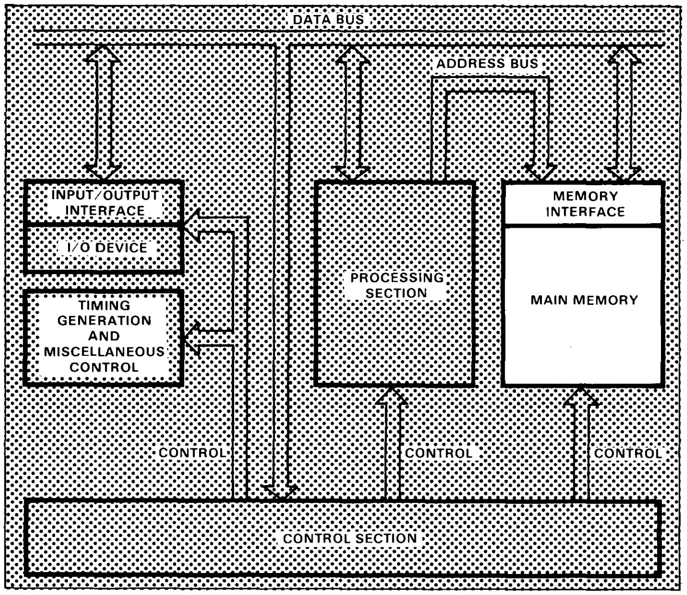


Figure 5

TIMING GENERATION/MISCELLANEOUS CONTROL

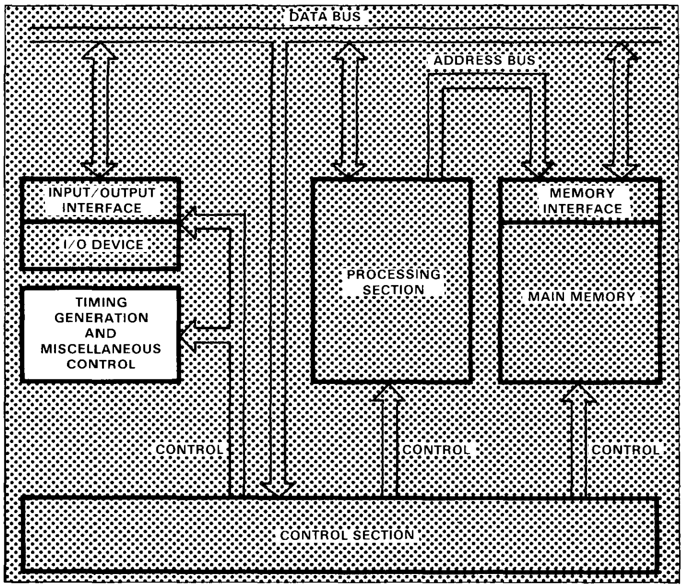


Figure 6

TIMING GENERATION AND MISCELLANEOUS CONTROL

Most timing generation requirements as shown in Figure 6 can be met by using a one-shot (retriggerable monostable multivibrator). If a multiple-phase clocking system is required, additional shift registers may be used.

DEVICE	DESCRIPTION
74123	Dual-monostable multivibrator
9602	Dual-monostable multivibrator
74S194	4-bit bidirectional universal shift register
74S195	4-bit parallel-access shift registers
74S178	4-bit shift register
74S179	4-bit shift register

INTRODUCTION

The Signetics Series 3000 Schottky Bipolar Microprocessor Chip Set has brought new levels of high performance and flexibility to microprocessor applications not previously possible with MOS technology. Combining the Schottky Bipolar N3001 Microprogram Control Unit (MCU) and the N3002 Central Processing Element (CPE) with industry standard memory and support circuits microinstruction cycle times of 100ns are possible, when using a pipelined architecture.

In the majority of cases, the choice of a bipolar microprocessor slice, as opposed to an MOS device, is based on speed and flexibility of microprogramming. Starting with these characteristics, the design of the Signetics Series 3000 Microprocessor has been optimized to achieve the following objectives:

- Logic replacement capability
- Higher performance in terms of speed
- Industry standard memory and support chips
- Cooler operation
- Lower total system cost

Furthermore, systems built with large-scale integrated circuits are much smaller in size and require less power than equivalent systems using medium and/or small-scale integrated circuits. Therefore, the end product is more cost effective and competitive in the market place.

The two components of the Series 3000 chip set, namely, the N3001 MCU and the N3002 CPE, when combined with industry standard memory and peripheral circuits, allows the design engineer to construct high-performance processors and/or controllers with a minimum amount of auxiliary logic. Features such as the multiple independent address and data buses, tri-state logic, and separate output enable lines eliminate the need for time-multiplexing of buses and associated hardware.

DESCRIPTION OF THE N3002 CPE

Each N3002 Central Processing Element (CPE) represents a complete 2-bit slice of the data processing section of a computer. Several CPEs may be connected in parallel to form a processor of any desired word length.

A block diagram of the N3002 CPE is shown in Figure 1.

Each CPE contains a 2-bit slice of five independent buses. Although these buses may be used in a variety of ways, typical connections are:

BUS	FUNCTION
Input M	Carries data from external memory.
Input I	Carries data from the input/ output device.
Input K	Used for microprogrammed mask or literal (constant) value input.
Output A	Connected to the CPE Memory Address Register as a Memory Address Bus.
Output D	Connected to the CPE accumulator as a data bus.

N3002 CPE BLOCK DIAGRAM

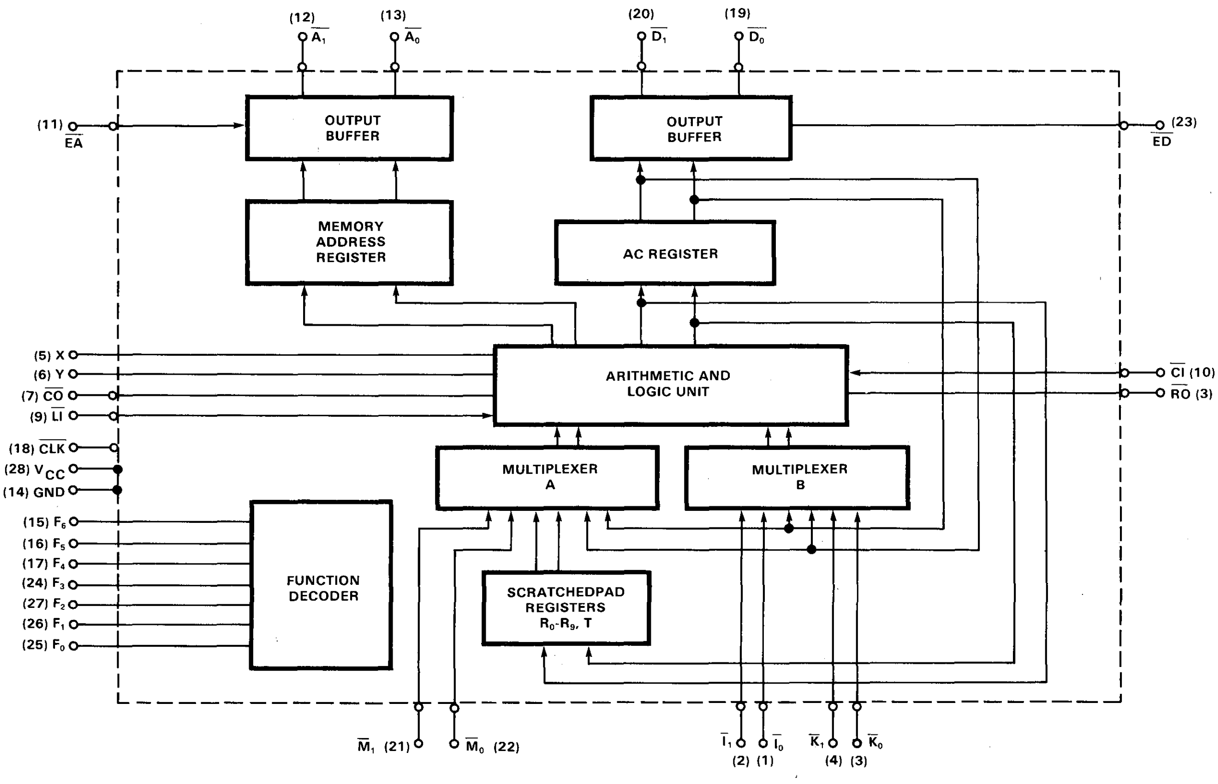


Figure 1

The microfuntion input bus (F-bus), supplied by the microprogram, controls the internal operation of the CPE, selecting both the operands and the operation to be executed upon them. The arithmetic logic unit (ALU), controlled by the microfunction decoder, is capable of over 40 Boolean and binary operations as outlined in the Function Description section of the N3002 data sheet.

Standard carry propagate and generate outputs (X and Y) are provided in the CPE for use with industry standard devices such as the Look-Ahead Carry Generator 74S182.

A summary of all possible operations as defined by the F_{0-6} bus is given below. These operations are classified as Logical, Control, Arithmetic, Move and Shift functions.

MNEMONIC	F GROUP	R GROUP	K BUS	EQUATIONS	FUNCTIONS
INSTRUCTION TYPE: LOGICAL OPERATIONS					
ANR	4	I	ALL 1	$R_n \wedge AC \rightarrow R_n$ $CI \vee (R_n \wedge AC) \rightarrow CO$	And AC with register and test result for zero.
ANM	4	II	ALL 1	$M \wedge AC \rightarrow AT$ $CI \vee (M \wedge AC) \rightarrow CO$	And M bus with AC and test result for zero.
ANI	4	III	ALL 1	$AT \wedge I \rightarrow AT$ $CI \vee (AT \wedge I) \rightarrow CO$	And I bus with AC (ORT) and test result for zero.
TZR	5	I	ALL 1	$CI \vee R_n \rightarrow CO$ $R_n \rightarrow R_n$	Test reg for zero.
	5	I	K	$CI \vee (R_n \wedge K) \rightarrow CO$ $K \wedge R_n \rightarrow R_n$	Mask reg with K bus.
LTM	5	II	ALL 1	$CI \vee M \rightarrow CO$ $M \rightarrow AT$	Load M bus in AT and test for zero.
	5	II	K	$CI \vee (M \wedge K) \rightarrow CO$ $K \wedge M \rightarrow AT$	Mask and test result for zero.
TZA	5	III	ALL 1	$CI \vee AT \rightarrow CO$ $AT \rightarrow AT$	Test reg for zero.
	5	III	K	$CI \vee (AT \wedge K) \rightarrow CO$ $K \wedge AT \rightarrow AT$	Mask and test result for zero.
ORR	6	I	ALL 1	$CI \vee AC \rightarrow CO$ $R_n \vee AC \rightarrow R_n$	Or AC to reg, and test previous AC value for zero.
ORM	6	II	ALL 1	$CI \vee AC \rightarrow CO$ $M \vee AC \rightarrow AT$	Or M bus to AC and test previous AC value for zero.
ORI	6	III	ALL 1	$CI \vee I \rightarrow CO$ $I \vee AT \rightarrow AT$	OR I bus to AT and test I bus data for zero.
XNR	7	I	ALL 1	$CI \vee (R_n \wedge AC) \rightarrow CO$ $R_n \overline{\oplus} AC \rightarrow R_n$	Exclusive NOR AC with R_n; force CO to 1 if ($R_n \wedge AC$) is non-zero.
XNM	7	II	ALL 1	$CI \vee (M \wedge AC) \rightarrow CO$ $M \overline{\oplus} AC \rightarrow AT$	Exclusive-NOR M bus with AC; force CO to 1 if ($M \wedge AC$) is non-zero.
XNI	7	III	ALL 1	$CI \vee (AT \wedge I) \rightarrow CO$ $I \overline{\oplus} AT \rightarrow AT$	Exclusive-NOR I bus with AC or T reg; force CO to 1 if ($AT \wedge I$) is non-zero.
INSTRUCTION TYPE: CONTROL OPERATIONS					
DSM	1	I	ALL 1	$11 \rightarrow MAR$ $R_n - 1 + CI \rightarrow R_n$	Set MAR to all one's. Decrement R_n conditionally.
LDM	1	II	ALL 1	$11 \rightarrow MAR$ $M - 1 + CI \rightarrow AT$	Set MAR to all one's. Load conditionally decremented M bus in AC or T reg.
CLR	4	I	ALL 0	$00 \rightarrow R_n$ $CI \rightarrow CO$	Clear reg and force CO to CI.
CLA	4	II	ALL 0	$00 \rightarrow AT$ $CI \rightarrow CO$	Clear AC or T and force CO to CI.
NOP	6	I	ALL 0	$R_n \rightarrow R_n$ $CI \rightarrow CO$	Null operation and force CO to CI.
CSR	2	I	ALL 0	$CI - 1 \rightarrow R_n$	Conditionally clear or set R_n to all O's or 1's, respectively.
CSA	2	II	ALL 0	$CI - 1 \rightarrow AT$	Conditionally clear or set AC or T reg to all 0's or 1's, respectively.

Table 1 N3002 INSTRUCTION SUMMARY

MNEMONIC	F GROUP	R GROUP	K BUS	EQUATION	FUNCTIONS
INSTRUCTION TYPE: ARITHMETIC OPERATIONS					
ILR	0	I	ALL 0	$R_n + CI \rightarrow R_n$, AC	Load AC from R_n or to increment R_n and load result in R_n, AC.
ALR	0	I	ALL 1	$AC + R_n + CI \rightarrow R_n$, AC	Add AC to R_n or $R_n + 1$.
AMA	0	II	ALL 0	$M + CI \rightarrow AT$	Load or increment M bus to AC or T reg.
	0	II	ALL 1	$M + AC + CI \rightarrow AT$	Add M bus or the incremented M bus to AC (May be used for indexing).
LMI	1	I	ALL 0	$R_n \rightarrow MAR$ $R_n + CI \rightarrow R_n$	Load MAR with R_n; or increment R_n. Used to update and maintain program counter.
DSM	1	I	ALL 1	$11 \rightarrow MAR$ $R_n - 1 + CI \rightarrow R_n$	Force MAR to all 1's. Conditionally decrement R_n.
LMM	1	II	ALL 0	$M \rightarrow MAR$ $M + CI \rightarrow AT$	Load MAR with M bus; add 1 to M bus with result stored in AC or T reg. Used for indirect Addressing.
LDM	1	II	ALL 1	$11 \rightarrow MAR$ $M - 1 + CI \rightarrow AT$	Force MAR to all 1's. Load M bus or decremented M bus to AC or T reg.
CIA	1	III	ALL 0	$AT + CI \rightarrow AT$	Add CI to complemented value of AT. Used to obtain 1's or 2's complement of AC or T reg.
DCA	1	III	ALL 1	$AT - 1 + CI \rightarrow AT$	Decrement at by 1 conditionally.
SDR	2	I	ALL 1	$AC - 1 + CI \rightarrow R_n$	Conditionally decrement or transfer AC to R_n.
SDA	2	II	ALL 1	$AC - 1 + CI \rightarrow AT$	Conditionally decrement AC by 1, or transfer AC to AC or T reg.
LDI	2	III	ALL 1	$I - 1 + CI \rightarrow AT$	Conditionally decrement I bus by 1 or transfer I bus to AC or T reg.
INR	3	I	ALL 0	$R_n + CI \rightarrow R_n$	Conditionally increment R_n by 1 or transfer R_n to R_n.
ADR	3	I	ALL 1	$AC + R_n + CI \rightarrow R_n$	Add AC to R_n if CI is false. Sum + 1 if CI is true.
INA	3	III	ALL 0	$AT + CI \rightarrow AT$	Conditionally increment AC or T reg by 1.
AIA	3	III	ALL 1	$I + AT + CI \rightarrow AT$	Add I bus to Ac or T reg if CI is false; Sum + 1 if CI is true.
INSTRUCTION TYPE: MOVE OPERATIONS					
LMI	1	I	ALL 0	$R_n \rightarrow MAR$ $R_n + CI \rightarrow R_n$	Move R_n to MAR, conditionally increment R_n. Can be used to maintain program counter.
LMM	1	II	ALL 0	$M \rightarrow MAR$ $M + CI \rightarrow AT$	Move M bus data to MAR; Increment M bus by 1.
SDA	2	II	ALL 1	$AC - 1 + CI \rightarrow AT$	If CI is true, move AC to AC or T reg.
LDI	2	III	ALL 1	$I - 1 + CI \rightarrow AT$	If CI is true, move I bus data to AC or T reg.
ACM	3	II	ALL 0	$M + CI \rightarrow AT$	If CI is false, move M bus data to AC or T reg.
LTM	5	II	ALL 1	$M \rightarrow AT$ $CI \quad M \rightarrow CO$	Move M bus to AC or T reg and test for zero.
LMF	6	II	ALL 0	$CI \rightarrow CO$ $M \rightarrow AT$	Move M bus to AC or T reg; force CO to CI.
CMR	7	I	ALL 0	$CI \rightarrow CO$ $\overline{R_n} \rightarrow R_n$	Complement R_n; forCE CO to CI.
LCM	7	II	ALL 0	$CI \rightarrow CO$ $\overline{M} \rightarrow AT$	Load AC or T reg with complemented M bus; force CO to CI
CMA	7	III	ALL 0	$CI \rightarrow CO$ $\overline{AT} \rightarrow AT$	Complement AC or T reg; force CO to CI.

Table 1 N3002 INSTRUCTION SUMMARY (Cont'd)

MNEMONIC	F GROUP	R GROUP	K BUS	EQUATION	FUNCTIONS
INSTRUCTION TYPE: SHIFT OPERATIONS					
SRA	0	III	ALL 0	$AT_L \rightarrow RO$ $AT_H \rightarrow AT_L$ $LI \rightarrow AT_H$	Shift or rotate right 1 BIT.
ALR	0	I	ALL 1	$AC + R_n + CI \rightarrow$ R_n, AC	If (R_n) = (AC), the operation is equivalent to shift left by 1 bit.

Table 1 N3002 INSTRUCTION SUMMARY (Cont'd)

DESCRIPTION OF THE N3001 MCU

The Microprogram Control Unit (MCU) controls the sequence in which microinstructions are acessed from the micropro-garm (or control program) memory (ROM, PROM, or bipolar RAM in the case of writable control store). Each microinstruction, in turn, controls the overall operation of the CPU and I/O functions. Since the MCU has a 9-bit address bus, it can access up to 512 words of control program. Additional addressing capability can be implemented via the firmware-controlled page register. Other features include:

- Schottky TTL Process
- 45ns cycle time (typical)
- Direct addressing of standard bipolar PROM or ROM
- advanced Organization:
  The 9-bit microprogram address register and bus are organized to address memory by row and column.
- 4-bit program Latch
- 2 flag registers
- Eleven address control functions:
  3 jump and test latch functions
  16 way jump and test instructions
- Eight flag control functions:
  4 flag input functions
  4 flag output functions

A block diagram of the 3001 MCU is shown in Figure 2.

The MCU uses a two-dimensional addressing scheme in the microprogram memory. This memory, consisting of 32 rows and 16 columns, will accommodate a total of 512 words. The word length is variable, and will depend on the application. The 32-by-16 matrix is shown in Figure 3.

The MCU is controlled directly by the microinstruction via the AC_{0-6} bus and FC_{0-3} bus. These two control buses define a group of jump/test functions to formulate the next microinstruction address. Details on these jump/test functions are outlined in the Function Description section of the N3001 data sheet. A brief summary is given below:

3001 MCU BLOCK DIAGRAM

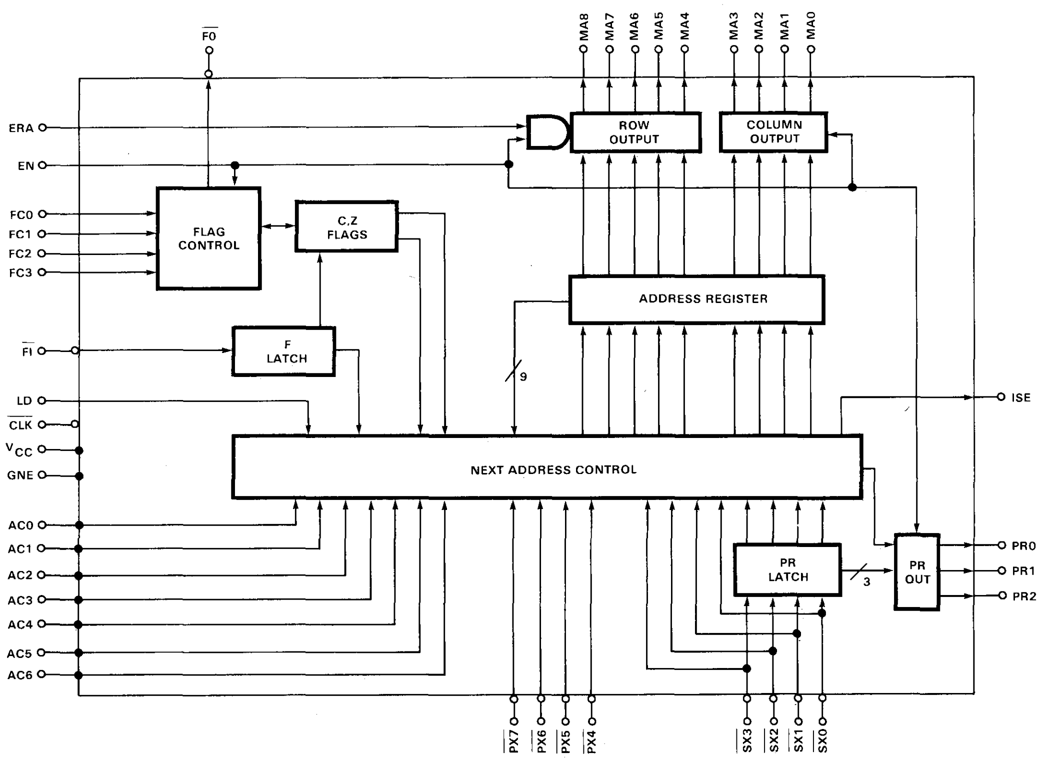


Figure 2

ROW AND COLUMN ADDRESS MATRIX

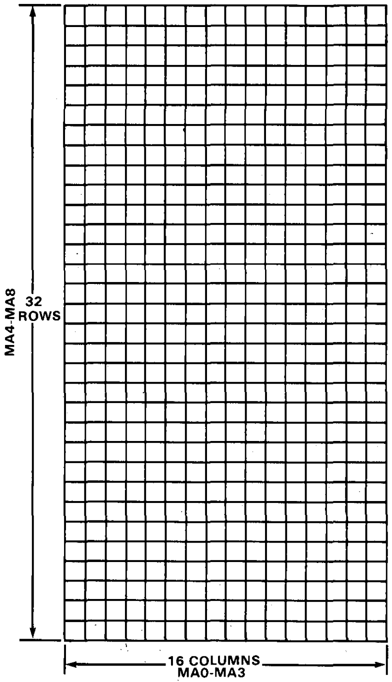
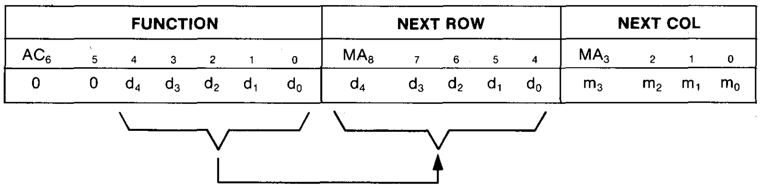


Figure 3

Unconditional Address Control Functions

MNEMONIC	FUNCTION
JCC	Jump in current column. AC_0 through AC_4 are used to select 1 of 32 row addresses in the current column, specified by MA_0 through MA_3, as the next address. See Figure 4.
JZR	Jump to zero row. AC_0 through AC_3 are used to select 1 of 16 column addresses in row $_0$, as the next address. See Figure 5.
JCR	Jump in current row. AC_0 through AC_3 are used to select 1 of 16 addresses in the current row, specified by MA_4 through MA_8, as the next address. See Figure 6.
JCE	Jump in current column-/row group and enable PR-latch outputs. AC_0 through AC_2 are used to select 1 of 8 row addresses in the current row group, specified by MA_7 and MA_8 as the next row address. The current column is specified by MA_0 through MA_3. The PR-latch outputs are asynchronously enabled. See Figure 7.

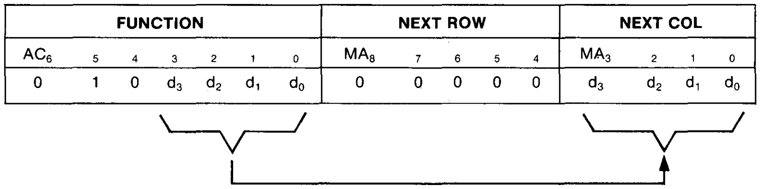
JCC (JUMP IN CURRENT COLUMN)



FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
0	0	d_4	d_3	d_2	d_1	d_0	d_4	d_3	d_2	d_1	d_0	m_3	m_2	m_1	m_0

Figure 4

JZR (JUMP TO ZERO ROW)



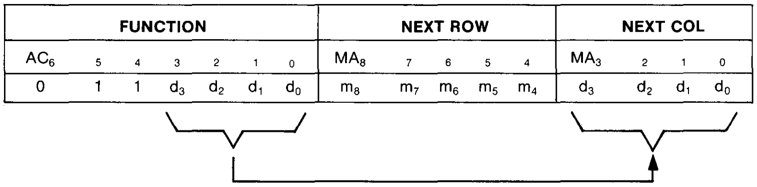
FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
0	1	0	d_3	d_2	d_1	d_0	0	0	0	0	0	d_3	d_2	d_1	d_0

NOTE

When $d_3 d_2 d_1 d_0$ = 1111, the ISE signal will be active. This signal can be used to communicate with the interrupt priority logic or some other type of logic.

Figure 5

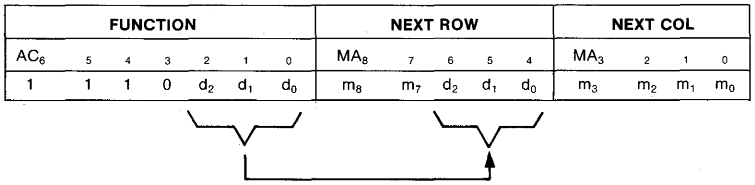
JCR (JUMP IN CURRENT ROW)



FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
0	1	1	d_3	d_2	d_1	d_0	m_8	m_7	m_6	m_5	m_4	d_3	d_2	d_1	d_0

Figure 6

JCE (JUMP IN CURRENT COLUMN/ROW 6RNP AND ENABLE PR-LATCH OUTPUTS



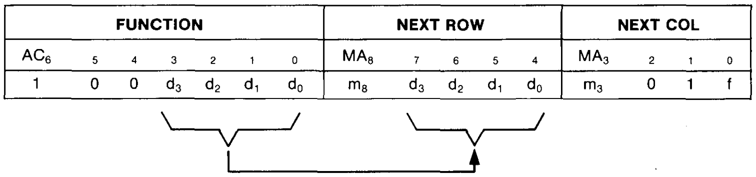
FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
1	1	1	0	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	m_3	m_2	m_1	m_0

Figure 7

Jump/Test on Flag Functions

MNEMONIC	FUNCTION
JFL	Jump/test F-latch. AC_0 through AC_3 are used to select 1 of 16 row addresses in the current row group, specified by MA_8, as the next row address. If the current column group, specified by MA_3, is col_0 through col_7, the F-latch is used to select col_2 or col_3 as the next column address. If MA_3 specifies column group col_8 through col_{15}, the F-latch is used to select col_{10} or col_{11} as the next column address. See Figure 8.
JCF	Jump/test C-flag. AC_0 through AC_2 are used to select 1 of 8 row adresses in the current row group, specified by MA_7 and MA_8, as the next row adress. If the current column group specified by MA_8 is col_0 through col_7, the C-flag is used to select col_2 or col_3 as the next column address. If MA_3 specifies column group col_8 through col_{15}, the C-flag is used to select col_{10} or col_{11} as the next column address. See Figure 9.
JZF	Jump/test Z-flag. Identical to the JCT function described above, except that the Z-flag, rather than the C-flag, is used to select the next column address. See Figure 10.

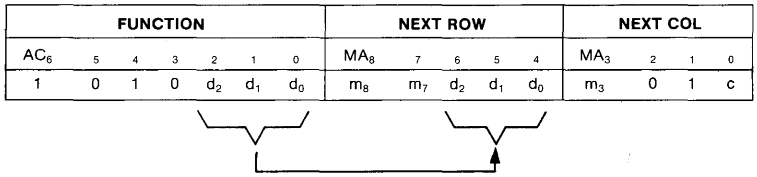
JFL (JUMP/TEST F LATCH)



FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
1	0	0	d_3	d_2	d_1	d_0	m_8	d_3	d_2	d_1	d_0	m_3	0	1	f

Figure 8

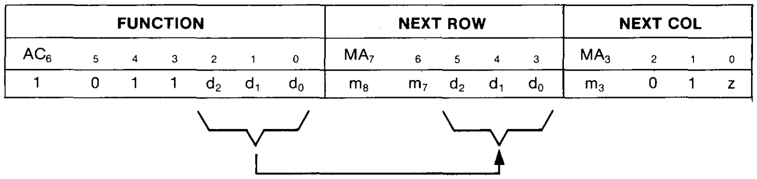
JCF (JUMP/TEST C FLAG)



FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
1	0	1	0	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	m_3	0	1	c

Figure 9

JZF (JUMP/TEST Z FLAG)



FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_7	6	5	4	3	MA_3	2	1	0
1	0	1	1	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	m_3	0	1	z

Figure 10

Flag Output Control Function (C, Z)

MNEMONIC	FC_3	FC_2	FUNCTION DESCRIPTION
FFO	0	0	Force FO to O. FO is forced to the value of logical O.
FFC	0	1	Force FO to C. FO is forced to the value of the C-flag.
FFZ	1	0	Force FO to Z. FO is forced to the value of the Z-flag.
FF1	1	1	Force FO to 1. FO is forced to the value of logical 1.

Flag Input Control Functions (C, Z)

MNEMONIC	FC_1	FC_0	FUNCTION DESCRIPTION
SCZ	0	0	Set C-flag and Z-flag to F1. The C-flag and the Z-flag are both set to the value of F1.
STZ	0	1	Set Z-flag to F1. The Z-flag is set to the value of F1. The C-flag is unaffected.
STC	1	0	Set C-flag to F1. The C-flag is set to the value of F1. The Z-flag is unaffected.
HCZ	1	1	Hold C-flag and Z-flag. The value in the C-flag and Z-flag are unaffected.

Jump/Test On PX Bus and PR Latch Functions

MNEMONIC	FUNCTION
JPR	Jump/test PR-latch. AC_0 through AC_2 are used to select 1 or 8 row addresses in the current row group, specified by MA_7 and MA_8, as the next row address. The four PR-latch bits are used to select 1 of 16 possible column addresses as the next column address. See Figure 11.
JLL	Jump/test left-most PR-latch bits. AC_0 through AC_2 are used to select 1 of 8 row addresses in the current row group, specified by MA_7 and MA_8, as the next row address. PR_2 and PR_3 are used to column addresses in col_4 through col_7 as the next column address. See Figure 12.
JRL	Jump/test right-most PR-latch bits. AC_0 and AC_1 are used to select 1 of 4 high-order row addresses in the current row group, specified by MA_7 and MA_8, as the next row address. PR_0 and PR_1 are used to select 1 of 4 possible column addresses in col_{12} through col_{15} as the next column address. See Figure 13.
JPX	Jump/test PX-bus and load PR-latch. AC_0 and AC_1 are used to select 1 of 4 row addresses in the current row group, specified by MA_6 through MA_8, as the next row address. PX_4 through PX_7 are used to select 1 of 16 possible column addresses as the next column address. SX_0 through SX_3 data is locked in the PR-latch at the rising edge of the clock. See Figure 14.

JPR (JUMP/TEST PR LATCH)

FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
1	1	0	0	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	p_3	p_2	p_1	p_0

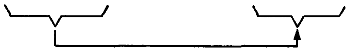


Figure 11

JLL (JUMP/TEST LEFT-MOST PR LATCH BITS)

FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
1	1	0	1	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	0	1	p_3	p_2

Figure 12

JRL (JUMP/TEST RIGHT-MOST PR LATCH BITS)

FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
1	1	1	1	1	d_1	d_0	m_8	m_7	1	d_1	d_0	1	1	p_1	p_0

Figure 13

JPX (JUMP/TEST PX BUS AND LOAD PR LATCH)

FUNCTION							NEXT ROW					NEXT COL			
AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
1	1	1	1	0	d_1	d_0	m_8	m_7	m_6	d_1	d_0	x_7	x_6	x_5	x_4

Figure 14

TYPICAL APPLICATIONS FOR THE SIGNETICS SERIES 3000

Because of its high performance and flexibility features, the Signetics Series 3000 is ideally suited for use in the following types of applications:

- Smart peripheral controllers
  Disk, magnetic tape, line printers, etc.
- Data communications
  Front end communications processors
  Communication controllers
  Intelligent terminals
- Business Applications
  Automated accounting computer
  Point of sale systems
  Automatic banking systems
- Computer emulation
- General purpose minicomputer
- Scientific/medical lab automation
  Blood analyzing systems
  Patient monitoring systems
  Instrument data acquisition systems
- Industrial and process control
  Machine tool control
  Assembly line flow control
  Batch mixing and weighing control
- Airborne vehicle applications
  Navigation control
  Weapon firing control

SERIES 3000 DESIGN EXAMPLES USING THE CARRY LOOK-AHEAD GENERATOR (74S182)

Example 1: The processing section of a 16-bit word CPU (data loop).

The processing section of a computer consists of data manipulation logic, storage devices, counters, and data transfer paths. The CPE was designed to perform the functions of the processing section of a computer.

To obtain low cost and high performance design, parallel carry look-ahead can be implemented using the multi-source industry standard part, 74S182. For a 16-bit CPU configuration, 8 CPE arrays, one MCU, and a 2 stage carry look-ahead generator can be connected as shown in Figure 15. Note that the three state gate 8T95 is disabled by the detection of a Shift Right operation (SRA); hence, its output to the FI input of the 3001 is placed in a high impedance state. The SRA operation is detected by the 6-input NAND function. The data, address and function buses are not shown in Figure 15, and the following logic state definition is observed:

N3002	K bus	Active Low
	I bus	Active Low
	M bus	Active Low
	D bus	Active Low
	A bus	Active Low
	LI, RO, CI	Active Low
	X, Y	Active High
N3001	FI, FO	Active Low
74S182	P_0-P_3 (X_{0-3})	Active High
	G_0-G_3 (Y_{0-3})	Active High
	P, G (X, Y)	Active High
	C_n	Acitve Low
	C_{n+x}, C_{n+y}, C_{n+z}	Active Low

Depending on the availability of spare gates, the same processing section (or data loop) can be implemented as shown in Figure 15. The active state of all data buses and other signals is similar to the example shown in Figure 15. Note that the Shift Right operation again inhibits the Carry via the 74S08 gate. The 74S05 is used due to its open collector type output which can be connected to the FI input of the N3001. The pull up is 470 Ω to +5V.

OTHER LOGIC	CARRY LOOK-AHEAD 74S182	TYP-ICAL (ns)	MAX (ns)
—	1st Stage	7	10.5
—	2nd Stage	4.5	7
8T95	—	6	7.5
N3002 T_{XD}	—	18	33

TOTAL DELAY THROUGH DATA LOOP 35.5 (TYP), 58ns (MAX)

Table 2 PROPAGATION DELAY ANALYSIS FOR FIGURE 15, EXAMPLE A

The propagation delay through the two stages of 74S182 is analyzed as follows:

OTHERS	LOOK-AHEAD	TYP-ICAL	MAX
	1st stage	7ns	10.5ns
74S08		7ns	7.5ns
	2nd stage	4.5ns	7.0ns
74S05		5ns	7.5ns
74S05		5ns	7.5ns
N3002 T_{XD}		18ns	33.0ns
Total Delay		46.5ns	73.0ns

Example 2: The control section of a 16-Bit Word CPU (control loop)

The control section of a microprogrammed processor usually performs the following functions:

- Decodes machine instructions
- Controls ALU functions (in N3002)
- Controls data paths
- Updates CPU hardware status
- Initiates I/O or memory operations
- Other miscellaneous control functions

The N3001 MCU, ROM/PROM (Signetics 82S115, 82S114, etc.) and minimum additional logic (such as gates, FF's or the Signetics' FPLA) can be used to implement the control section of the CPU. A simplified logic diagram of the control section is shown in Figure 16.

For a microprogrammed CPU, one of the most important design efforts is in the definition of the microinstruction (microword). *In general, the microinstruction consists of* a number of control fields to handle the following functions:

- Initiation of main memory operations
- Initiation of I/O operations
- Instruction decoding
- Manipulation data transfer
- Testing machine status
- Responding to and processing interrupts
- Address sequencing

When all the possible and necessary control functions are defined, then the microword can be structured. The following structure is an attempt to define a 32-bit microword that fulfills the above-mentioned control functions (Figure 17).

When defining the microword, it is essential to compact each field as much as possible without sacrificing efficiency. To illustrate this principle, a format field is supplied which defines multiple usage of the lower order 9 bits, K < 8 > and K < 7:0 > of the microword. The formats are classified as shown in Table 3.

CONTROL SECTION OF CPU

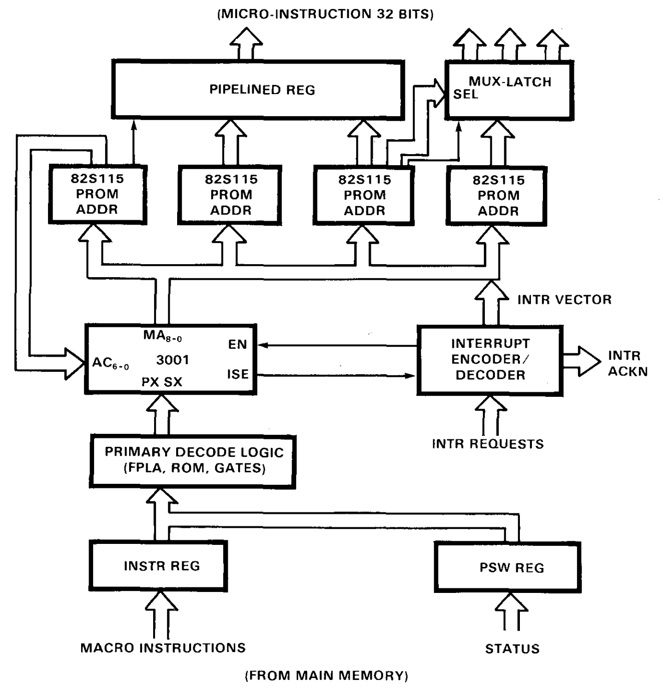
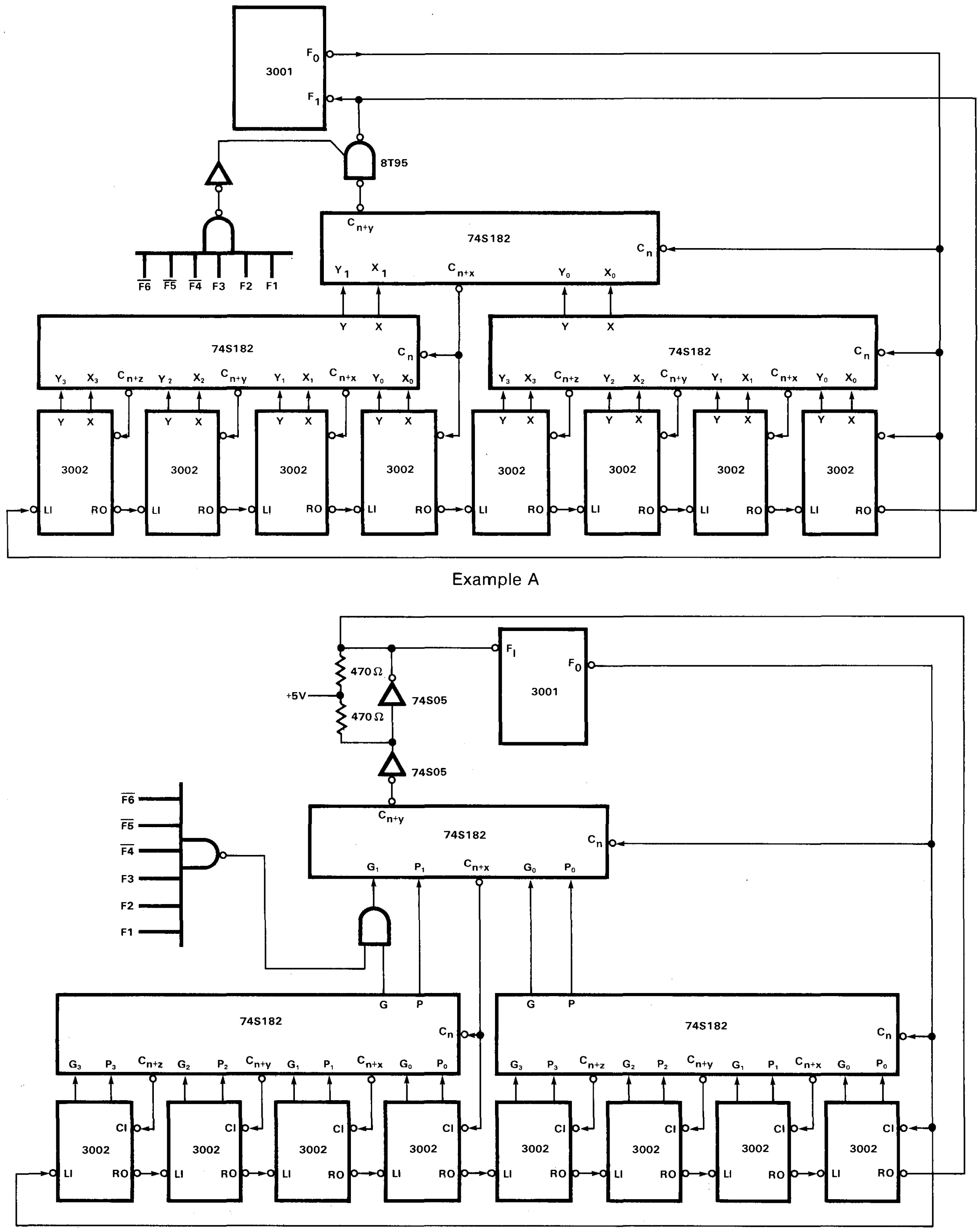


Figure 16

16-BIT WORD CPU (DATA LOOP)


3001
F_0
F_1
8T95
74S182
C_{n+y}
C_n
Y_1 X_1 C_{n+x} Y_0 X_0
$\overline{F6}$ $\overline{F5}$ $\overline{F4}$ F3 F2 F1
Y X
Y_3 X_3 C_{n+z} Y_2 X_2 C_{n+y} Y_1 X_1 C_{n+x} Y_0 X_0
3002
LI RO
470Ω
+5V
74S05
F_I
G_1 P_1 C_{n+x} G_0 P_0
G P
G_3 P_3 C_{n+z} G_2 P_2 C_{n+y} G_1 P_1 C_{n+x} G_0 P_0
CI

Example A

Example B

Figure 15

The charts illustrated in Figures 18 and 19 further define and usage of the K < 7:0 > field:

(A) When FMT < 1:0 > = 00, the K < 7:0 > field is used to control the Processor status logic.
(B) When FMT < 1:0 > = 01 or 11, the K < 7:0 > field is used to control the I/O—Memory operations.

Note that the K bits in this example are used in four different ways depending on the format of that microinstruction. They are used to initiate I/O operations, memory operations, control of the processor status word, and to provide constants/masks on the K bus. Refer to Figure 20 for a possible hardware implementation of the logic. This part of the definition and the DCDR < 2:0 > field may vary from machine to machine and the user can exert considerable design flexibility and freedom here. On the other hand, the AC < 6:0 >, FC < 3:0 > and F < 6:0 > fields are absolutely necessary to control the MCU and CPE arrays.

The next consideration in designing the control section is how to enhance performance. Three basic approaches can be employed here: (no attempt is made to explore other possibilities).

1. Use high performance parts, such as Signetics FPLA (35ns typical), 4K PROM (82S115, with 35ns typical), N3001 MCU (45ns typical).
2. Improve architecture by using a pipelined register to latch up the microinstruction. With this arrangement, simultaneous execution (of the current microinstruction) and fetching (of the next microinstruction) can be performed. The Signetics PROMs, 82S114 and 82S115, have output latches built-in, and they can be used as pipelined registers. Note that in this example, the format field FMT < 1:0 > and the AC < 6:0 > field should not be buffered, since the former is used to select the section of the pipeline register to be loaded and the latter is used to sequence the microprogram address register (setting up the next microinstruction address).
3. Use a horizontally oriented microword structure. In a horizontal structure, the microword itself consists of a large number of bits (sometimes more than 100), and a fewer total number of microinstructions. The purpose for having such a wide word is the higher degree of parallelism in operation that can be achieved in the same machine cycle.

MICROPROGRAM EXAMPLES

The following examples demonstrate how a macroinstruction may be implemented by executing a sequence of preprogrammed microinstructions. To make the examples more comprehensive, a number of assumptions were made, and they are included in the general description associated with each of the examples.

MICROWORD STRUCTURE

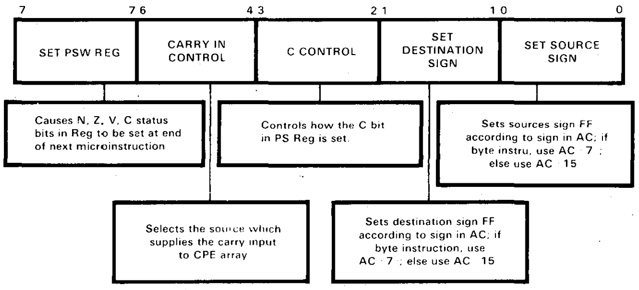

7
7 6
4 3
2 1
1 0
0
SET PSW REG
CARRY IN CONTROL
C CONTROL
SET DESTINATION SIGN
SET SOURCE SIGN
Causes N, Z, V, C status bits in Reg to be set at end of next microinstruction
Controls how the C bit in PS Reg is set.
Sets sources sign FF according to sign in AC; if byte instru, use AC 7 ; else use AC 15
Selects the source which supplies the carry input to CPE array
Sets destination sign FF according to sign in AC; if byte instruction, use AC 7 ; else use AC 15

Figure 17

FORMAT FIELD	FUNCTION OF K < 8 > AND K < 7:0 >
0 0	K < 7:0 > controls PSW logic. K < 8 > masks upper byte of K bus for CPEt130
0 1	K < 7:0 > controls I/O and memory. K < 8 > masks both bytes of K bus for CPE
1 0	K < 7:0 > masks lower byte of K bus for CPE K < 8 > masks upper byte of K bus for CPE
1 1	K < 7:0 > controls I/O and memory K < 8 > masks upper byte of K bus for CPE

Table 3 FORMAT FIELDS

CONTROL OF THE PROCESSOR STATUS LOGIC

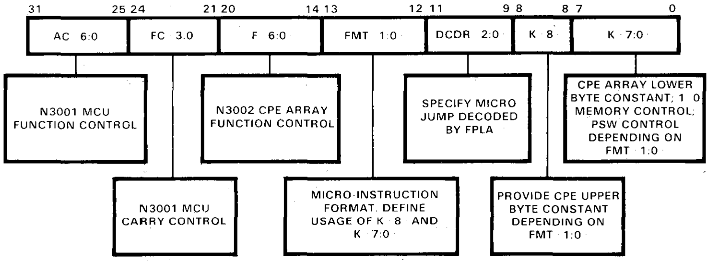

31
25 24
21 20
14 13
12 11
9 8
8 7
0
AC 6:0
FC 3.0
F 6:0
FMT 1:0
DCDR 2:0
K 8
K 7:0
N3001 MCU FUNCTION CONTROL
N3002 CPE ARRAY FUNCTION CONTROL
SPECIFY MICRO JUMP DECODED BY FPLA
CPE ARRAY LOWER BYTE CONSTANT; 1 0 MEMORY CONTROL; PSW CONTROL DEPENDING ON FMT 1:0
N3001 MCU CARRY CONTROL
MICRO-INSTRUCTION FORMAT. DEFINE USAGE OF K 8 AND K 7:0
PROVIDE CPE UPPER BYTE CONSTANT DEPENDING ON FMT 1:0

Figure 18

CONTROL OF THE I/O-MEMORY OPERATIONS

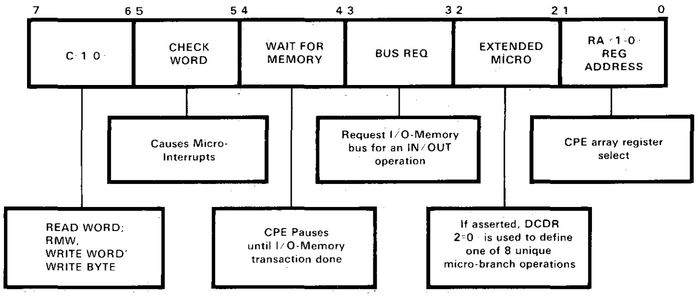

7
6 5
5 4
4 3
3 2
2 1
0
C 1 0
CHECK WORD
WAIT FOR MEMORY
BUS REQ
EXTENDED MICRO
RA 1 0 REG ADDRESS
Causes Micro-Interrupts
Request I/O-Memory bus for an IN/OUT operation
CPE array register select
READ WORD; RMW, WRITE WORD WRITE BYTE
CPE Pauses until I/O-Memory transaction done
If asserted, DCDR 2:0 is used to define one of 8 unique micro-branch operations

Figure 19

Macro Instruction

Move Word

15 10	9 7	6 4	3 2	1 0
OP CODE	RS	RD	AMS	AMD

RS = Source Register (0-7)
RD = Destination Register (0-7)
AMS = Address Mode of Source Operand
AMD = Address Mode of Destination Operand

Address modes are user definable. But for this example let's consider the following assumptions:

AMS = Register direct; i.e., the source operand is contained in RS.

AMD = Memory indirect; i.e., the effective operand address is formed by adding the contents of the designated RD to the base address. The base address is the location following the instruction. (In this case it is the same as the program counter R_7).

OPERATIONS
(Source) → (Destination)

STATUS: No change

DESCRIPTION
Basic microprogram involves the following types of operations:

1. Fetch macro instruction, update P.C.
2. Decode macro instruction into classes by FPLA.
3. Form Destination Address.
4. Form Source Operand Address.
5. Execution
6. Allow for interrupts at the end of execution.

Load Immediate Byte

15 11	10 8	7 0
OP CODE	R_n	IMM

OPERATIONS
(IMM_{7-0}) → Right-hand byte of R_n; Left-hand byte of R_n undisturbed.

R_n: Refers to R_0-R_7 in N3002; R_7 is also used as a Program counter.

STATUS: No change

DESCRIPTION
The microprogram implementation of this macro instruction involves the following types of operations:

1. Fetch the macro instruction from Main Memory; Wait for Memory until the instruction arrives at the Instruction register of the CPU.
2. Decode instruction by FPLA into classes.
3. Formulate operand addressing and execution.

MICROWORD FORMAT CONTROL LOGIC

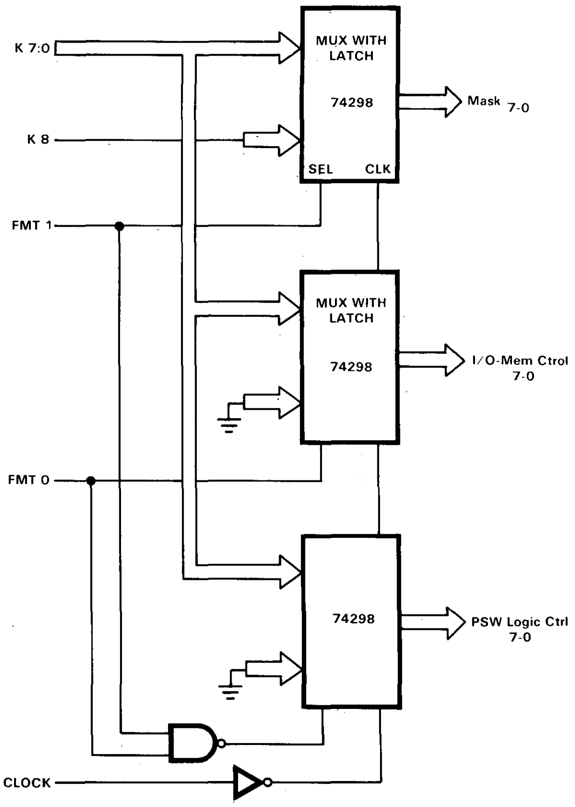
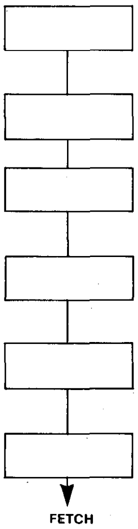


Figure 20

MOVE WORD OPERATION



SUMMARY: Fetch instruction; update program counter; AC = JCC; FC = HCZ, FF1; F = LMI; FMT = 1; K < 8 > = 0; K < 7:0 > = Mem Request, Read Mem, Wait for Mem, new instruction.

SUMMARY: Decode; M-Bus → ACCUM; AC = JPX; FC = HCZ, FF0; F = ACM; FMT = 2; K < 8 > = 0; K < 7:0 > = 0.

SUMMARY: Form destination address; PC → ACCUM, PC; AC = JCR; FC = HCZ, FF0; F = ILR; FMT = 2 K < 8 > = 0; K < 7:0 > = 0.

SUMMARY: Form destination address continued; RD + ACCUM → RD, ACCUM; AC = JPR; FC = HCZ, FF0; F = ALR; FMT = 1; K < 8 > = 1; K < 7:0 > = use RA < 1:0 > to select RD.

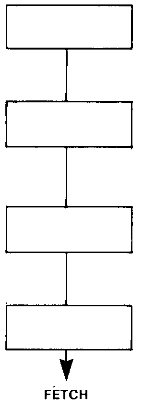
SUMMARY: Form source address; get source operand Rs → ACCUM; AC = JCR; FC - HCZ, FF0; F = ILR; FMT = 1; K < 8 > = 0; K < 7:0 > = use RA < 1:0 > to select Rs.

SUMMARY: Store operand into memory; Jump to Fetch AC = JZR; FC = HCZ, FFO; F = LMI; FMT = 1; K < 8 > = 0; K < 7:0 > = use RA < 1:0 > to select RD; Write Mem, Mem Request, enable interrupts.

Figure 21

4. Store results in destination.
5. During the above process, the program counter must be updated and be ready for fetching the next macro instruction.
6. Allow for interrupt at the end of instruction execution.

LOAD IMMEDIATE BYTE OPERATION



SUMMARY: Fetch Instr.; Update Program Counter; AC = JRC; FC = HCZ, FF1; F = LMI; FMT = 1 (Memory Control Mode); K < 8 > = 0 (both Bytes); K < 7:0 > = Read Memory, New instruction. Mem Req. Wait for Memory.

SUMMARY: Decode, M-Bus → AC; AC = JPX: FC = HCZ, FF0; F = ACM; FMT = 2 (Mask Mode); K < 8 > = 0; K < 7:0 > = 0.

SUMMARY: Execution; AC (00FF) → AC; Mask out right byte of (IMM) AC = JCC: FC = HCZ, FF1; F = (select F2 and RI); FMT = 2 (Mask Mode); K < 8 > = 0; K < 7:0 > = FF.

SUMMARY: Execution and store results; Jump to fetch; AC = JZR; FC = HCZ, FF0; F = ORR: FMT = 1 (Mask and I/O Control Mode); K < 8 > = 1 (both bytes); K < 7:0 > = Use RA_{0-1} to access internal Rn Enable interrupts.

Figure 22

APPENDIX A

The following symbols and their meanings apply to the N3002 instruction summary:

SYMBOL	MEANING
I,K,M	Data on the I, K, and M buses, respectively
CI,LI	Data on the carry input and left input, respectively
CO,RO	Data on the carry output and right output, respectively
R_n	Contents of register n including T and AC (R-Group I)
AC	Contents of the accumulator
AT	Contents of AC or T, as specified
MAR	Contents of the memory address register
L,H	As subscripts, designate low and high order bit, respectively
+	2's complement addition
−	2's complement subtraction
∧	Logical AND
∨	Logical OR
+	Exclusive-NOR
→	Deposit into

APPENDIX B

The following table summarizes the Jump/Test instructions and symbology for the N3001.

MNEMONIC	DESCRIPTION	FUNCTION							NEXT ROW					NEXT COL			
		AC_6	5	4	3	2	1	0	MA_8	7	6	5	4	MA_3	2	1	0
JCC	Jump in current column	0	0	d_4	d_3	d_2	d_1	d_0	d_4	d_3	d_2	d_1	d_0	m_3	m_2	m_1	m_0
JZR	Jump to zero row	0	1	0	d_3	d_2	d_1	d_0	0	0	0	0	0	d_3	d_2	d_1	d_0
JCR	Jump in current row	0	1	1	d_3	d_2	d_1	d_0	m_8	m_7	m_6	m_5	m_4	d_3	d_2	d_1	d_0
JCE	Jump in column/enable	1	1	1	0	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	m_3	m_2	m_1	m_0
JFL	Jump/test F-latch	1	0	0	d_3	d_2	d_1	d_0	m_8	d_3	d_2	d_1	d_0	m_3	0	1	f
JCF	Jump/test C-flag	1	0	1	0	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	m_3	0	1	c
JZF	Jump/test Z-flag	1	0	1	1	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	m_3	0	1	z
JPR	Jump/test PR-latch	1	1	0	0	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	p_3	p_2	p_1	p_0
JLL	Jump/test left PR bits	1	1	0	1	d_2	d_1	d_0	m_8	m_7	d_2	d_1	d_0	0	1	p_3	p_2
JRL	Jump/test right PR bits	1	1	1	1	1	d_1	d_0	m_8	m_7	1	d_1	d_0	1	1	p_1	p_0
JPX	Jump/test PX-bus	1	1	1	1	0	d_1	d_0	m_8	m_7	m_6	d_1	d_0	x_7	x_6	x_5	x_4

NOTE

d_n = Data on address control line n
m_n = Data in microprogram address register bit n
p_n = Data in PR-latch bit n
x_n = Data on PX-bus line n (active LOW)
f,c,z = Contents of F-latch, C-flag, or Z-flag, respectively

HOW TO DESIGN WITH THE CONTROL STORE SEQUENCER 8X02 MP2

INTRODUCTION

The basic structure of a high performance Central Processing Unit (CPU) or a "Smart" controller can be typically classified into two distinct but interactively related functional sections. One section is generally referred to as the Processing Section and the other the Control Section.

With the state of the art in bipolar Schottky technology, high-performance microprocessors are designed to perform functions of the Processing Section. Due to the limitation on pin numbers and chip size, the overall Processing Section is partitioned into several functionally equivalent slices. In today's bipolar microprocessor market, 2-bit and 4-bit slice architecture predominates. Each architecture type has its uniqueness but, in general, a slice contains a group of general-purpose registers, an accumulator, special-purpose register(s), ALU and related status flags. All of these elements constitute the Processing Section of a CPU.

The Control Section of the CPU is more complex in design. Typically this section includes the macro-instruction decode logic, test-branch decode, microprogram sequencing logic and the control store where the microprogram resides. Aside from the microprogram, the remaining portion of the Control Section (macro instruction decode, test-branch decode and sequencing), does not lend itself to efficient partitioning into vertical slices. This is due to the random nature of the logic usually found in the Control Section. However, horizontal functional grouping is possible. For example, the macro-instruction decode and test-branch decode logic can now be replaced by the Signetics FPLA (Field Programmable Logic Array); the random logic traditionally needed to implement the microprogram sequencing can now be replaced by the Signetics Control Store Sequencer; and, of course, the microprogram can be stored in a high-density PROM or ROM such as the 82S115.

GENERAL DESCRIPTION

The Signetics Control Store Sequencer is a low-power Schottky LSI, designed for use in high-performance, microprogrammed systems. The basic function of this device is to set up the microprogram address from which the microinstruction is fetched. All microinstructions are assumed to reside in the Control Store (ROMs, PROMs, or RAMs, in the case of Writable Control Store).

The fundamental philosophy behind the design of the sequencer evolved around three points.

1. To design an LSI that can handle most of the essential sequencing functions normally required for efficient microprogramming.
2. To design an LSI that is easy to use.
3. To design an LSI in a 28-pin DIP and yet maintain a high addressability of 1024 words.

Additional address requirement can be easily met by providing external page registers, which can be either entirely or partially controlled via the microprogram.

FUNCTIONAL DESCRIPTION

The Control Sequencer architecture is shown in Figure 1. The address register is a 10-bit D-type flip-flop which holds the current address. The register changes state when the Clock is in a low-to-high transition (edge-triggered). The address register can be loaded with different address sources under control of the 3 address control lines (AC_{2-0}) and 1 test input line. These sources are:

- All 0s for reset
- Current address + 1 for simple increment
- Current address + 2 for skip
- 10-bit branch address from outside
- Stack register file output

There is a 4-level Stack Register File and a 2-bit Stack Pointer, both of which respond automatically to operations requiring a Push (Write to Stack Register File) or Pop (Read Stack Register File).

The file is organized as a 4 word by 10-bit matrix and operates as a LIFO. The Stack Pointer operates as an up/down counter.

A cross section of the activities that take place within various logic elements is shown in Table 1. The AC_{2-0} and the Test Input are the variables in the chart. This chart may be used to gain a better understanding of the device so that its capability can be fully utilized.

Following is a detailed description of all the possible functions performed by the Control Store Sequencer. Note that the mnemonics are in parenthesis, and the logic state is defined as True = 5V and False = 0V.

AC_{2-0} = 000
Test and Skip (TSK)

Perform test on Test Input line.

If test is False:
Next address = current address + 1. Stack Pointer unchanged.

If test is True:
Next address = current address + 2, i.e., skip next microinstructions. Stack Pointer unchanged.

This function is used to facilitate transfer of control based on the result of a test on the Test Input line.

AC_{2-0} = 001
Increment (INC)

Next address = current address + 1.

This function is used to serially sequence the address register by 1. This simple function eliminates the need for providing 10 external address lines to do a Branch to next address.

CONTROL STORE SEQUENCER ARCHITECTURE

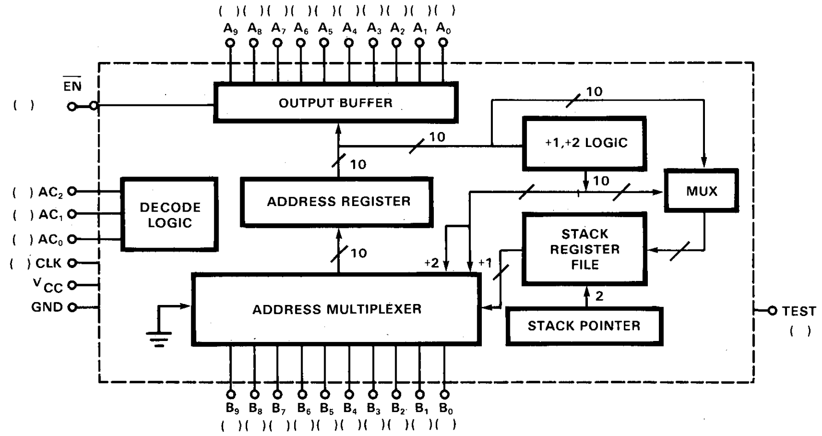


Figure 1

MNEMONIC	DESCRIPTION	FUNCTION $AC_{2\ 1\ 0}$	TEST	NEXT ADDRESS	STACK	STACK POINTER
TSK	Test and skip	0 0 0	False True	Current + 1 Current + 2	N.C. N.C.	N.C. N.C.
INC	Increment	0 0 1	X	Current + 1	N.C.	N.C.
BLT	Branch to loop if test input true	0 1 0	False True	Current + 1 Stack Register File	X Pop (Read)	Decrement Decrement
POP	Pop stack	0 1 1	X	Stack Register File	Pop (Read)	Decrement
BSR	Branch to subroutine if test input true	1 0 0	False True	Current + 1 Branch Address	N.C. Push (Current + 1)	N.C. Increment
PLP	Push for looping	1 0 1	X	Current + 1	Push (Current Address)	Increment
BRT	Branch if test input true	1 1 0	False True	Current + 1 Branch Address	N.C. N.C.	N.C. N.C.
RST	Set microprogram address output to zero	1 1 1	X	All O's	N.C.	N.C.

X = Don't care
N.C. = No change

Table 1 NEXT ADDRESS CONTROL FUNCTION

AC_{2-0} = 010 Branch to Loop if Test Condition True (BLT)

Perform test on Test Input line.

If test is True:
Next address = address from register file (POP). Stack Pointer decremented by 1.

If test is False:
Next address = current address + 1. Stack Pointer decremented by 1.

This function is used as the last microinstruction of a loop (assuming that the beginning microinstruction is a Push for Looping AC_{2-0} = 101). By means of this function, the loop is re-executed or exited depending on the result of the test on the Test Input line. If the test is True, the loop will be re-executed by using the address supplied by the Stack Register File. If the test is False, the control exits the loop by moving to the next address. In either case, the Stack Pointer is kept current automatically.

AC_{2-0} = 011 Branch to Subroutine if Test Input True (BSR)

If test is True:
Next address = branch address (B_{9-0}). Push current address + 1 → Stack Register File. Stack Pointer incremented by 1.

If test is False:
Next address = current address + 1. No push on stack. Stack Pointer unchanged.

This function facilitates the transfer of control based on the result of the test on the Test Input line. If the test is False, no branch will take place and the next instruction will be executed. If the test is True, the address register is loaded with B_{9-0} (Branch Address) lines and, at the meantime, the current address + 1 is written or pushed into the Stack Register File. The latter condition allows branching to a micro-subroutine whose beginning address is supplied by B_{9-0} and, at the meantime, the return address is saved in the Stack Register File.

AC_{2-0} = 101 Push for Looping (PLP)

Next address = current address + 1. Stack Pointer incremented by 1. Push (current address) → Stack Register File.

This function is generally used as the first microinstruction of a program loop. The current address is saved in the Stack Register File. This function works hand in hand with the BLT function.

AC_{2-0} = 110 Branch if Test Condition True (BRT)

If test is True:
Next address = branch address (B_{9-0}).

If test is False:
Next address = current address + 1.

This function is used to facilitate transfer of control based on the result of the test on the Test Input line. If the test is True, the next address is supplied by the B_{9-0} lines. If the test is False, the control proceeds to the next address.

AC_{2-0} = 111 Reset to 0 (RST)

Next address = 0, for reset.

This function is used to reset the address to all 0's. The state of the B_{9-0} lines has no bearing on the next address setup.

The following additional functions can be performed by the Sequencer, although they are not related to the state of the AC_{2-0} and Test Input:

1. The device will power up to a known state, which is all 0's. This feature can be used to initiate the "power on reset" subroutine.
2. When the external clock is inhibited, all internal register contents are undisturbed. This is a means of retaining the current address (and therefore executing the current microinstruction) for timing delay purposes, where the unit time is equal to the microinstruction cycle time. The clock inhibit signal can be supplied directly by the micro-code or it could be the status condition of certain control logic.
3. The three state output buffers can be disabled (placed in a high-impedance state) when an external address source is used to access the microprogram. This external address can be a microp-interrupt vector, which directly accesses the starting microinstruction of the interrupt handling subroutine. If Address 0 is to be reserved for initialization, then the micro-interrupt vector can be asserted on any address lines other than A_0, such as $A_1A_2A_3$ for 8 levels of interrupts.

HOW TO DESIGN WITH THE 8X02

The 8X02 is totally compatible with all bipolar TTL logic elements. A typical hardware setup is shown in Figure 2. This example generally represents the control loop of a 16-bit CPU. The various major block functions partitioned by the dotted line boxes can be described as follows:

1. One FPLA is used to decode the macroinstruction.
2. One FPLA is used to decode the hardware and program status condition.
3. A multiplexer is used to channel one of eight conditions to the Test Input of the 8X02. The multiplexer select control is directly supplied by the microcode. These conditions may vary from system to system. The multiplexer approach is a simple way to accommodate the multitude of conditions that need to be tested. Note that to force a 1 and 0 can be accomplished by tying the inputs to 5V and 0V respectively.
4. The 8X02 is used to sequence the microprogram.
5. The eight 82S115 PROMs are used to implement a 1K-word by 32-bit microprogram. Additional PROMs may be added as required. The address output signals (A_9-A_0) of the 8X02 can drive up to 8mA.

To control the 8X02 as it is configured in Figure 2, the firmware basically has to provide fields for:

$AC_{2\text{-}0}$: 3 bits for address control
ACK INH: 1 bit for clock inhibit
$S_{2\text{-}0}$: 3 bits for multiplexer select. In a simpler design, a 1-bit field connected directly to the Test Input pin of 8X02 may satisfy the design requirement.

MICROPROGRAMMING CONSIDERATIONS

During the design phase of the firmware (or microprogram), the firmware engineer may find it necessary to allocate certain addresses in the microprogram to handle specific functions which are hardware dependent. For example:

1. One address each may be assigned as the entry point for subroutine handling, depending on the way that the interrupt vector is connected to the address bus ($A_{9\text{-}0}$).
2. Address 0 may be assigned to handle system initialization functions.
3. A convenient number of addresses may be allocated to take care of memory fetch functions as well as sampling (enabling) interrupts.

8X02 AS PART OF CONTROL LOOP CONFIGURATION

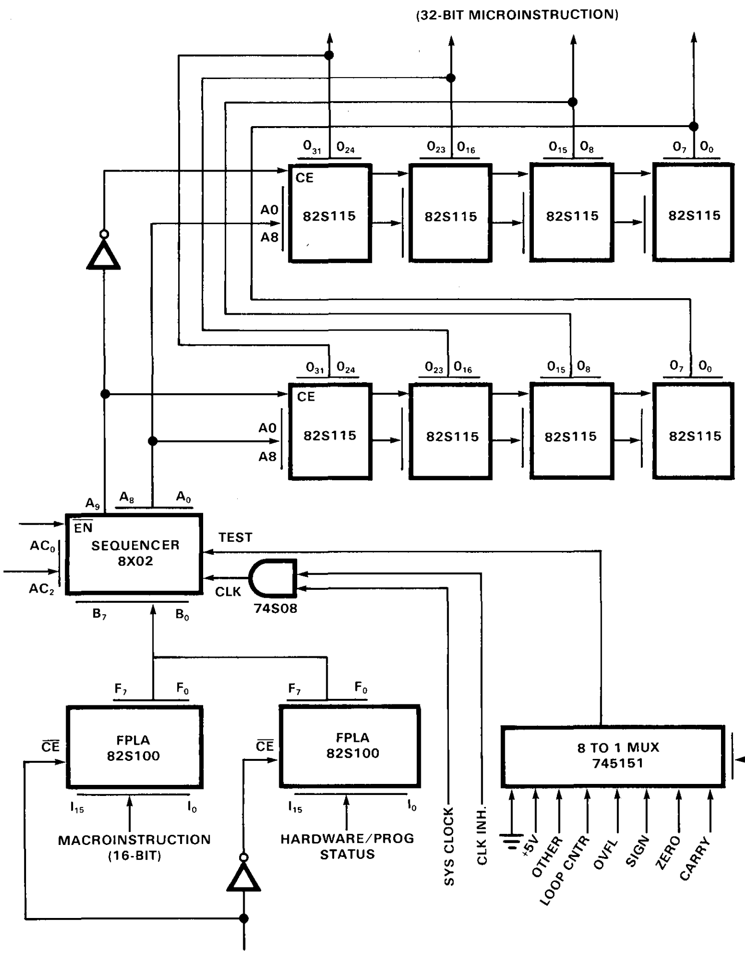
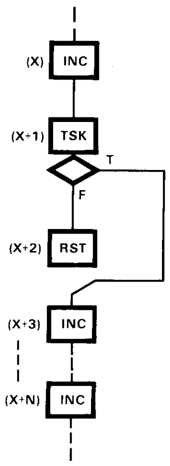


Figure 2

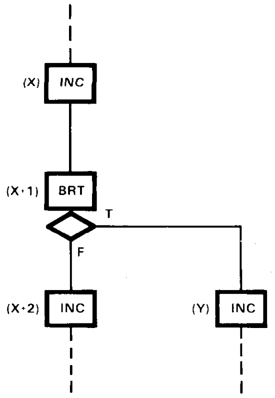
TEST AND SKIP PROGRAMMING TECHNIQUE



1. The TSK instruction is used to facilitate transfer of controls.
2. When executing the TSK instruction, the Test Input is checked and, if the Test is True, skip the next instruction and go to address (X + 3). If the Test is False, go to the next address (X + 2).
3. The RST instruction is used to bring the control to address 0 where micro-interrupts can be enabled.

Figure 3

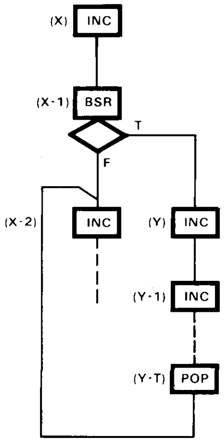
CONDITIONAL BRANCHING TECHNIQUE



1. N-WAY BRANCH within the same 1024 word page is possible.
2. When the Test condition is True, the branch will be taken. The branch address is supplied via B_{9-0}. In the example it is (Y).
3. When the Test condition is False, the control will proceed to the next instruction at address (X + 2).

Figure 4

SUBROUTINE NESTING TECHNIQUE



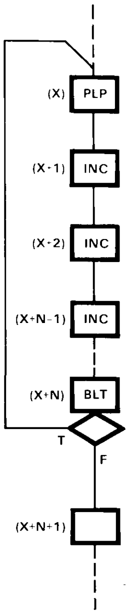
In this subroutine the beginning address (Y) must be presented to B_{9-0} inputs during the BSR instruction. If the Branch is taken, the return address (X + 2) will be saved in the Stack Register File. When the subroutine is done, issue a POP instruction to return the main program to address (X + 2).

NOTE

Addresses are shown in parenthesis and instructions are shown inside the blocks.

Figure 5

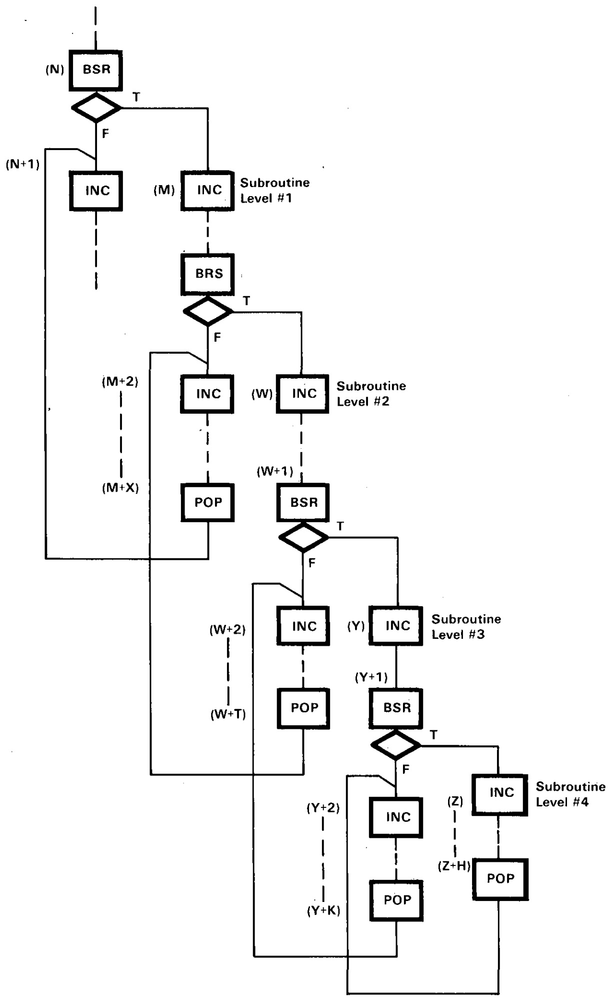
PROGRAM LOOPING TECHNIQUE



1. The first instruction of the loop must be a PLP. During the execution of a PLP, the sequencer pushes the current address (X) into the Stack Register File.
2. The last instruction of the loop must be a BLT instruction. When the BLT is executed, the sequencer checks the "Test Input" which is normally connected to a loop count overflow signal. If the loop counter does not overflow, the loop will be re-executed. If the loop counter does overflow, the next instruction will be automatically accessed and executed.

Figure 6

4-LEVEL SUBROUTINE NESTING TECHNIQUE



When applying the subroutine nesting technique, the following can be used as guideline:

1. Use the BSR instruction to branch to the subroutine if the Test Input of the sequencer is high.
2. Use a POP instruction to return from a subroutine.
3. Caution must be exercised to avoid stack overflow or underflow.
4. A 10-bit address (beginning address of subroutine) must be supplied to the B_{9-0} during BSR instruction.

NOTE

Addresses are shown in parenthesis and the instructions inside blocks of flowchart.

Figure 7

DESCRIPTION

Typical interfaces to the 8X300 employ the 8T32/33 or 8T35/36 bidirectional I/O ports. These devices provide a single connnection between the 8X300 and the user status control and data lines. Each interface is denoted as an Interface Vector and is field programmed to a specific address.

ADDRESSING DATA ON THE INTERFACE VECTOR

The Interface Vector is comprised of general purpose 8-bit I/O registers called Interface Vector (IV) Bytes. The IV registers serve to select IV bytes. In order for an instruction to access (read or write) an IV byte, the address of that byte must be output to the IVL or IVR registers.

Thus, two instructions are required to operate on an Interface Vector byte:

XMIT ADDRESS, IVL MACHINE INSTRUCTION

Each of the two IV registers (IVL and IVR) may be set to select an IV byte, therefore two I/O ports may be active at one time—one on the Right Bank (IVR) and one on the Left Bank (IVL). Data may be input and output in one instruction following the selection of IV bytes:

```
XMIT     ADDRESS1,IVL
XMIT     ADDRESS2,IVR
ADD      LB, RB
```

Once the IV byte is selected (addressed) it will remain selected until another address is output to the same IV register. Since an IV register (IVL, IVR) can be used only as a destination field of an instruction, any instruction sending data to IVL or IVR can be used to select an IV byte.

From the user's standpoint, however, all IV byte outputs can be read by an external device regardless of whether they are selected or not.

The address range of IVL and IVR is 0-255$_{10}$.

ELECTRICAL CHARACTERISTICS OF THE INTERFACE VECTOR

Each IV byte consists of 8 storage latches which hold data transferred between the Interpreter and the User System, 8 tri-state input/output lines and 2 input/output control lines, called Byte Input Control (BIC) and Byte Output Control (BOC) as shown in Figure 1. The control lines functions are summarized in Table 1. Table 2 contains a summary of the electrical characteristics of the IV byte.

IV BYTE PROVIDING DYNAMICALLY DEFINED DATA FLOW

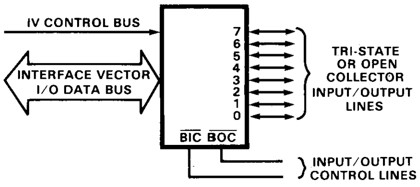

IV CONTROL BUS
INTERFACE VECTOR
I/O DATA BUS
7
6
5
4
3
2
1
0
TRI-STATE OR OPEN COLLECTOR INPUT/OUTPUT LINES
BIC BOC
INPUT/OUTPUT CONTROL LINES

Figure 1

IV BYTE WIRED FOR USER OUTPUT ONLY

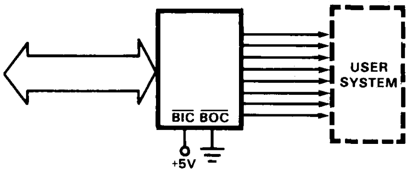


Figure 2

IV BYTE WIRED FOR INPUT ONLY

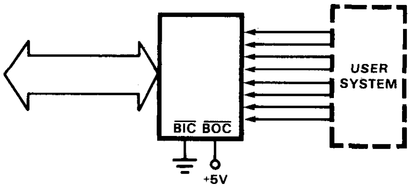


Figure 3

Working storage consisting of RAM may be connected to either or both left and right I/O banks. An example of such an arrangement is shown in Figure 4. Paging may be added to the memory to extend the addressability.

CONTROL LINES		FUNCTION
BOC (low true)	BIC (low true)	
H	H	8 I/O lines in high impedance state—disable
L	H	8 I/O lines in output mode—8 bit storage latch data available in the output lines.
X	L	8 I/O lines in input mode—data can be read by Interpreter

Table 1 FUNCTIONS OF THE BIC AND BOC LINES

PARAMETER		TEST CONDITIONS	LIMITS			UNIT
			Min	Typ	Max	
V_{IH}	High level input voltage		2		5.5	V
V_{IL}	Low level input voltage		-1		0.8	V
V_{IC}	Input clamp voltage	I_{IL} = -5mA			-1	V
V_{OH}	High level output voltage	I_{OH} = 1mA	2.4			V
V_{OL}	Low level output voltage	I_{OL} = -16mA			0.5	V
I_{IH}	High level input current[1]	V_{IH} = 5.5V			100	uA
I_{IL}	Low level input current[1]	V_{IL} = 0.50V			-800	uA
I_{OS}	Output short circuit current	V_{OL} = 0V	-20		-200	mA
C_{IN}	Data input capacitance	V_{IL} = 0V			12	pF

NOTE

[1] Input current is always present regardless of the state of BIC and BOC.

Table 2 IV BYTE TERMINAL ELECTRICAL CHARACTERISTICS

8X300 TYPICAL SYSTEM CONFIGURATION WITH WORKING STORAGE

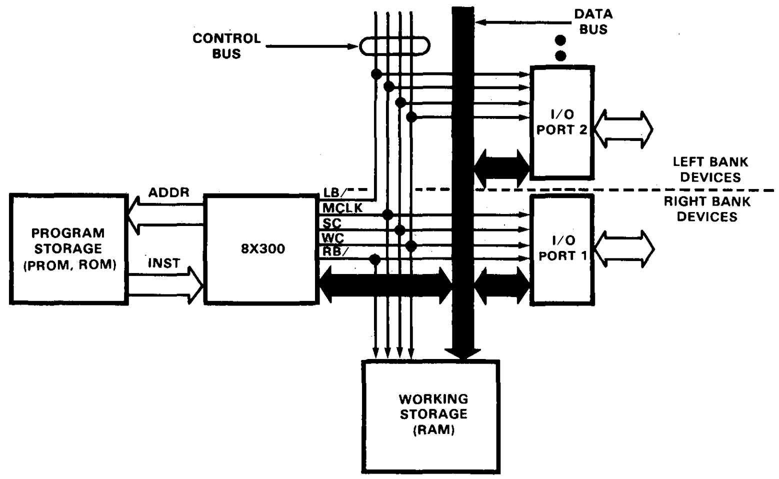


Figure 4

8X300 MICROPROCESSOR APPLICATIONS MEMO

FLOPPY DISC INTERFACE

DESCRIPTION

The 8X300 controls a floppy disc drive with a minimal amount of additional circuitry. In this example, byte assembly and disassembly are performed by the program ("bit banging") to reduce interface circuitry. Addition of such circuitry would increase hardware costs and decrease significantly peak processor utilization.

Data is transferred to and from the floppy disc via I/O driver routines. These I/O driver routines provide a standard software interface to a floppy disc and require 180 words of program storage. When not transferring data to and from the disc, the 8X300 is available to service other devices such as keyboards, displays or data communication lines. Figure 1 illustrates the system.

DESIGN APPROACH

Data bytes are assembled or disassembled by sensing a clock, inputing data bit or generating clock, and outputing a data bit. Preamble patterns, track address, and other disc format requirements are implemented by programming. Disc drive head must be stepped to the desired track before data transfer is initiated. Disc drive status is monitored to determine any error conditions. A four step procedure is followed to implement the design:

1. Analyze interface. Analyze floppy disc relative to: Number of control/data lines, timing and data rates associated with each control/data line, and electrical characteristics of each control/data line. Determine any supplemental circuits needed for electrical compatibility (see Table 1).
2. Perform functional analysis. The functions to be programmed and any which require supplemental logic are determined. In this case, supplemental logic is utilized to process the 250ns pulses associated with DATA, CLOCK and WR DATA. Optional logic for byte assembly and disassembly also are shown. The programmed functions are:
   a. Byte assembly/disassembly
   b. Generate preamble, track address, timing and sector synchronization
   c. Sense clock and disc status
   d. Step head to desired track
3. Define the program to process input and to generate output (see Figure 2).
4. Determine 8X300 configuration (see Table 2).

FLOPPY DISC INTERFACE

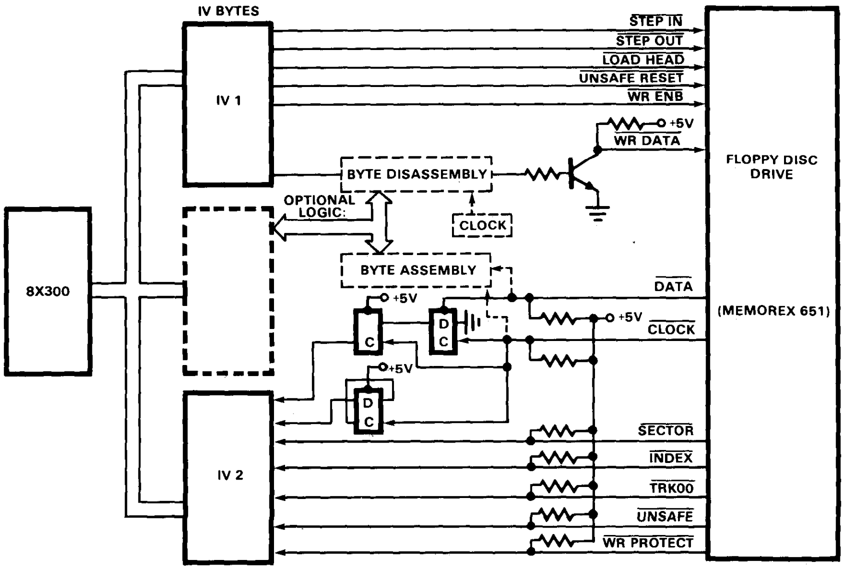


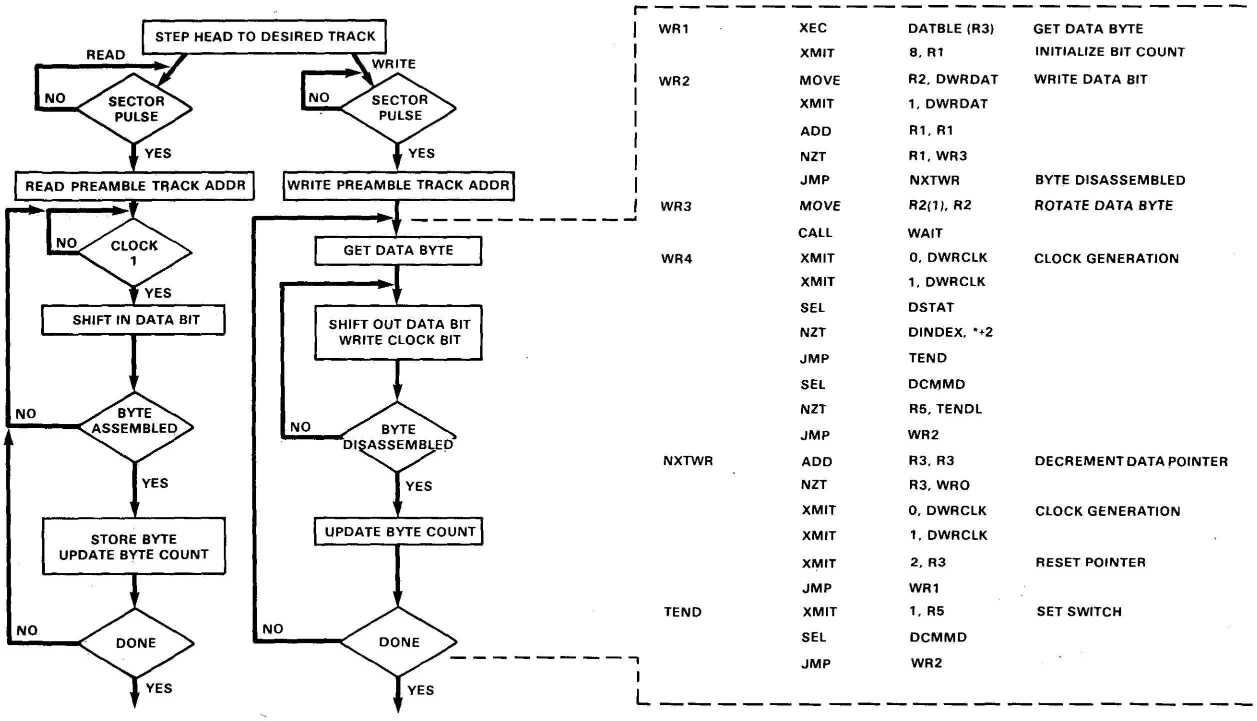
Figure 1

SIGNAL NAME	DATA RATE-1	SIGNAL DURATION	ELECTRICAL CHARACTERISTICS	# IV BITS	INTERFACE REQUIRED	FUNCTION
STEP IN	20ms	.01-10ms	TTL with pullup	1		Step head 1 track in
STEP OUT	20ms	.01-10ms	TTL with pullup	1		Step head 1 track out
LOAD HEAD	Level		TTL with pullup	1		Load head
UNSAFE RESET	Level		TTL with pullup	1		Clears unsafe condition
WR ENB	Level		TTL with pullup	1		Enables write operation
WR DATA	2µs	.25µs	50mA current	1	2R,T	Data/clock to disc
SECTOR	5ms	1ms	OC output	1	R	Sector indicated
INDEX	160ms	1ms	OC output	1	R	Begin of track indicator
TRK00	Level		OC output	1	R	Head on track 00
UNSAFE	Level		OC output	1	R	Unsafe condition indicator
WR PROTECT	Level		OC output	1	R	Write protected disc
DATA	4µs	.25µs	OC output	1	R,2FF	Data from disc
CLOCK	4µs	.25µs	OC output	1	R,FF	Clock from disc

R = Resistor
T = Transistor
FF = Flip-Flop

Table 1 INTERFACE ANALYSIS

FUNCTIONAL ANALYSIS



```
WR1     XEC     DATBLE (R3)   GET DATA BYTE
        XMIT    8, R1         INITIALIZE BIT COUNT
WR2     MOVE    R2, DWRDAT    WRITE DATA BIT
        XMIT    1, DWRDAT
        ADD     R1, R1
        NZT     R1, WR3
        JMP     NXTWR         BYTE DISASSEMBLED
WR3     MOVE    R2(1), R2     ROTATE DATA BYTE
        CALL    WAIT
WR4     XMIT    0, DWRCLK     CLOCK GENERATION
        XMIT    1, DWRCLK
        SEL     DSTAT
        NZT     DINDEX, *+2
        JMP     TEND
        SEL     DCMMD
        NZT     R5, TENDL
        JMP     WR2
NXTWR   ADD     R3, R3        DECREMENT DATA POINTER
        NZT     R3, WRO
        XMIT    0, DWRCLK     CLOCK GENERATION
        XMIT    1, DWRCLK
        XMIT    2, R3         RESET POINTER
        JMP     WR1
TEND    XMIT    1, R5         SET SWITCH
        SEL     DCMMD
        JMP     WR2
```

Through put: (a) Peak data rate: 250K bits/sec. (b) Peak processor utilization: 97.5%, including byte assembly/disassembly. (c) Peak processor utilization: 12.2%, with external byte assembly/disassembly.

Figure 2

ROM/PROM FOR PROGRAM STORAGE		WORKING STORAGE FOR DATA BUFFERS	IV BYTES FOR INPUT/OUTPUT INTERFACE
Input Driver	76 words	256 Bytes	6 IV bits for output
Output Driver	74 words		7 IV bits for input
Head Step Driver	30 words		Total: 2 IV bytes
Total	180 words		

Table 2 8X300 CONFIGURATION

TELETYPE MULTIPLEXER

DESCRIPTION

The 8X300 is easily interfaced to a teletype or similar asynchronous device. Processor utilization is less than .1%, even when used in a character assembly mode.

A single 8X300 can be used as a multiplexer for many low speed asynchronous devices. For example, the 8X300 can be used as a front end multiplexer for a large computer system. Figure 3 illustrates the system.

DESIGN APPROACH

A basic teletype I/O driver routine receives, transmits and echoes a character. Character assembly/disassembly is implemented by sensing start bit, sampling data bit and generating output bit timing. A four-step procedure is followed to implement the design:

1. Analyze interface. Analyze teletype relative to: Number of control/data lines, timing and data rates associated with each control/data line, electrical characteristics of each control/data line, and determine any supplemental circuits needed for electrical compatibility (see Table 3).
2. Perform functional analysis. The functions to be programmed and any which require supplemental logic are determined. In this case, no supplemental logic is required. The programmed functions are:
   a. Character assembly/disassembly
   b. Sense start bit
   c. Generate bit timing and simultaneous character echo
3. Define the program to process input and to generate output (see Figure 4).
4. Determine 8X300 configuration (see Table 4).

TELETYPE MULTIPLEXER

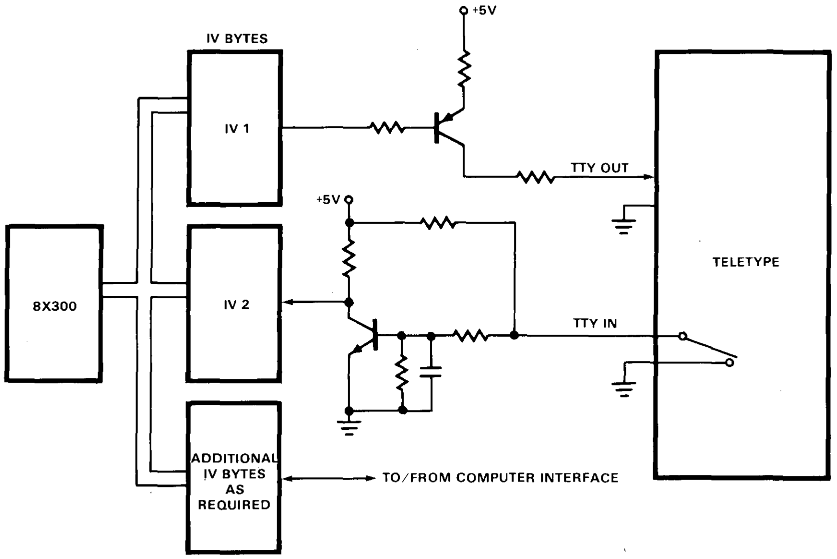


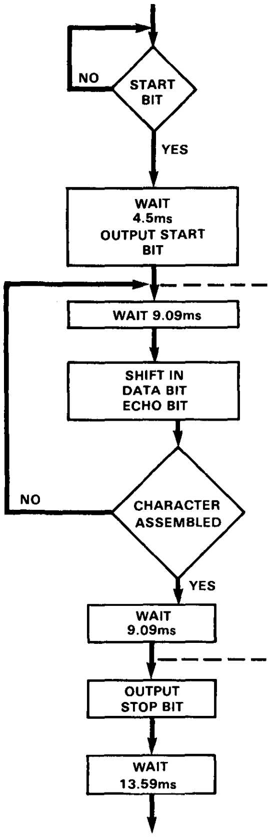
Figure 3

SIGNAL NAME	DATA RATE	SIGNAL DURATION	ELECTRICAL CHARACTERISTICS	# IV BITS	INTERFACE REQUIRED	FUNCTION
TTY OUT	9.09ms	9.09ms	20mA current	1	3R,T	Data to TTY printer
TTY IN	9.09ms	9.09ms	20mA current	1	4R,T,C	Data from TTY keyboard

R = Resistor
T = Transistor
C = Capacitor

Table 3 INTERFACE ANALYSIS

FUNCTIONAL ANALYSIS



```
DELAY1   XMIT    220,R1       WAIT 9 MSEC
         CALL    DELAY
         XMIT    2, IVL       SHIFT IN DATA BIT
         MOVE    TTYIN,AUX
         XOR     R5,R5
         XMIT    1,IVL        ECHO DATA BIT
         XMIT    1,AUX
         XOR     R5,TTYOUT

         MOVE    R5(1),R5

         ADD     R3,R3        BYTE ASSEMBLED?
         NZT     R3,DELAY1
         XMIT    220,R1
         CALL    DELAY
```

Through put: (a) Peak data rate: 220 bits/sec. (b) Peak Processor utilization: .05%

Figure 4

ROM/PROM FOR PROGRAM STORAGE	WORKING STORAGE FOR DATA BUFFERS	IV BYTES FOR INPUT/OUTPUT INTERFACE
Teletype driver .... 49 words Delay routine .... 10 words Total .... 59 words	2 bytes per Teletype	1 IV bit, for output, per Teletype 1 IV bit for input, per Teletype Total: 2 IV bytes per 8 Teletypes

Table 4 8X300 CONFIGURATION

8X300 MICROPROCESSOR APPLICATIONS MEMO

DATA CONCENTRATOR

DESCRIPTION

The 8X300 multiplexes multiple low speed terminals. It buffers the data in its working storage for efficient transmission over common carrier or other data link facilities. Single inquiry/response terminals are interfaced to a single half-duplex synchronous line via a Universal Asynchronous Receive-/Transmit (UART) interface. This eliminates cabling to each terminal. The 8X300 transfers inquiry and response messages between terminals and a remote computer data base via a data communications line. Various communication data rates are accommodated by simple program modification. Figure 5 illustrates the system.

DESIGN APPROACH

The 8X300 polls each terminal requesting an input character or signaling an output character. Each character is transferred over a high speed (9600 baud) synchronous line whose data rate determines the scan time of the 8X300 multiplexing program. The multiplexer program formats polling messages, maintains status, generates and checks the Longitudinal Redundancy Character, performs character recognition, and buffers characters. Additional driver programs are required to communicate with the full duplex data communications line to/from a remote computer data base. A four step procedure is followed to implement the design:

1. Analyze interface. Analyze UART relative to: Number of control/data lines, timing and data rates associated with each control/data line, electrical characteristics of each control/data line and determine any supplemental circuits needed for electrical compatibility (see Table 5).
2. Perform functional analysis. The functions to be programmed and any which require supplemental logic are determined. In this case, no supplemental logic is required. The programmed functions are:
   a. Maintain current line status
   b. Generate synchronization pattern, poll command, sense character synch
   c. Resynchronize with clock and monitor modem and UART status
3. Define the program to process input and to generate output (see Figure 6).
4. Determine the 8X300 configuration (see Table 6).

DATA CONCENTRATOR

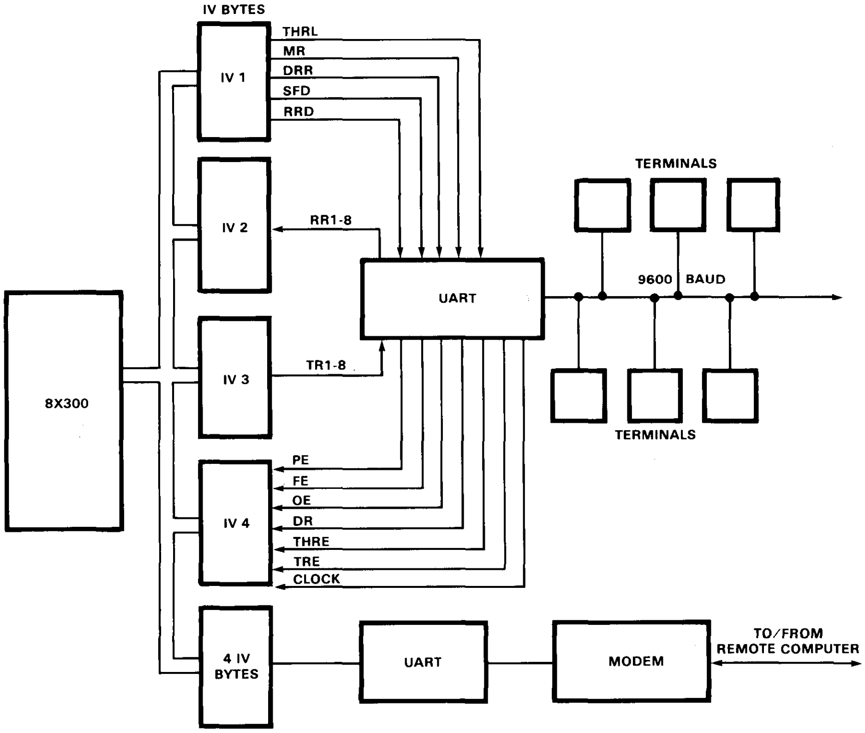
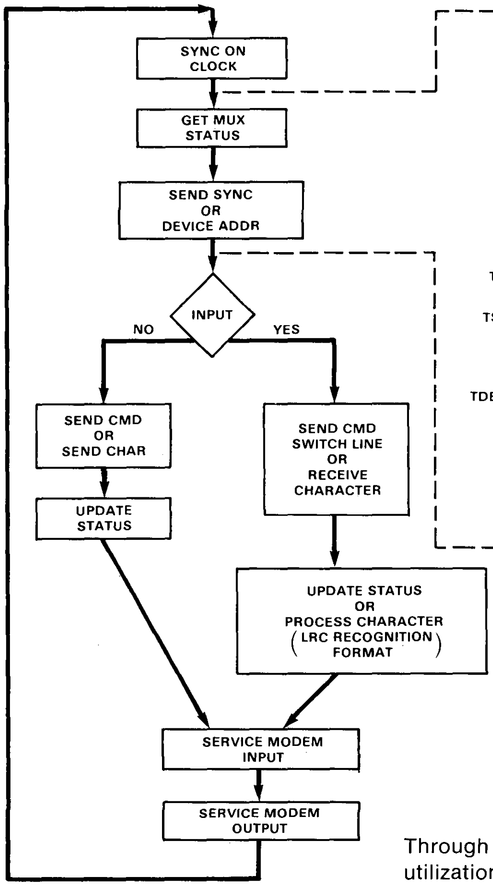


Figure 5

SIGNAL NAME	DATA RATE	SIGNAL DURATION	ELECTRICAL CHARACTERISTICS	# IV BITS	INTERFACE REQUIRED	FUNCTION
TR1-8	1.041ms	1.2-10μs	TTL	8	-	Output data
THRL	1.041ms	1.2-10μs	TTL	1	-	Load output data
MR	level		TTL	1	-	Master reset
DRR	level		TTL	1	-	Data received reset
SFD	level		TTL	1	-	Status flag disable
RRD	level		TTL	1	-	Receiver Register disable
RR1-8	1.041ms	1.041ms	TTL	8	-	Received data
PE	level		TTL	1	-	Parity error
FE	level		TTL	1	-	Frame error
OE	level		TTL	1	-	Over run error
DR	level		TTL	1	-	Data received flag
THRE	level		TTL	1	-	XMTR holding reg. empty
TRE	level		TTL	1	-	Transmitter register empty
CLOCK	1.041ms	1.041ms	TTL	1	-	Data rate clock

Table 5 INTERFACE ANALYSIS

FUNCTIONAL ANALYSIS



```
            SEL     DEVADR              GET MUX STATUS
            MOVE    DEVADR, R2
            SEL     STATE
            MOVE    STATE, R3
            XEC     JOB (R3), 8         SELECT NEXT
            JMP     MODEMI              JOB

JOB         JMP     TSYNC
            JMP     TDEVADR
            JMP     TCMD
            JMP     RSYNC/CHAR

TSYNC       XMIT    SYNC R4           ⎫
            CALL    TRANSMIT          ⎪  TRANSMIT SYNC
TSYNC1      SEL     STATE             ⎬  CHARACTER AND
            XMIT    1, AUX            ⎪  UPDATE MUX
            ADD     STATE, STATE      ⎪  STATUS
            JMP     JOB-1             ⎭
TDEVADR     SEL     DEVADR            ⎫
            MOVE    DEVADR, R4        ⎪
            CALL    TRANSMIT          ⎪  TRANSMIT DEVICE
            SEL     STATE             ⎬  ADDRESS AND
            XMIT    1, AUX            ⎪  UPDATE MUX
            ADD     STATE, STATE      ⎪  STATUS
            JMP     JOB-1             ⎭
```

Through put: (a) Peak data rate: 2880 characters/sec. (b) Peak processor utilization 30% including modem servicing.

Figure 6

ROM/PROM FOR PROGRAM STORAGE	WORKING STORAGE FOR DATA BUFFERS	IV BYTES FOR INPUT/OUTPUT INTERFACE
Multiplexer driver .... 156 words Character processing .... 100 words Total .... 256 words	32 bytes	13 IV bits for output per UART 15 IV bits for input per UART Total: 4 IV bytes per UART

Table 6 8X300 CONFIGURATION

8X300 MICROPROCESSOR APPLICATIONS MEMO

REMOTE ALPHANUMERIC TERMINAL CONTROLLER

DESCRIPTION

The 8X300 interfaces to simple keyboard/display devices with a minimal amount of interface circuitry. The display may be buffered or the 8X300 system can supply buffering and refresh. In this example, the personality of the keyboard/display terminal is programmed into program storage to implement various editing and format functions. A single 8X300 can be used to control a local cluster of alphanumeric terminals since the processor utilization for a single terminal is very low. Messages to and from each terminal are transferred to a remote computer (interface not shown). Figure 7 illustrates the system.

DESIGN APPROACH

A terminal driver routing inputs and buffers messages in working storage. The driver also performs character and line deletion functions and implements a flicker free display of the message. A special set of control characters are used to terminate a message and forward the message. A four step procedure is followed to implement the design:

1. Analyze interface. Analyze keyboard and display relative to: Number of control and data lines, timing and data rates associated with each control/data line, electrical characteristics of each control/data line and determine supplemental circuits needed for electrical compatibility. Here the interfaces are completely compatible electrically (see Table 7).
2. Perform function analysis. The functions to be programmed and any which require supplemental logic are determined. In this case, no supplemental logic is required. The programmed functions are:
   a. Store a message input from keyboard
   b. Update display to produce flicker free output
   c. Implement character delete, line delete editing functions
   d. Recognize end of message control character.
3. Define the program to process input and to generate output (see Figure 8).
4. Determine the 8X300 configuration (see Table 8).

REMOTE ALPHANUMERIC TERMINAL COUNTER

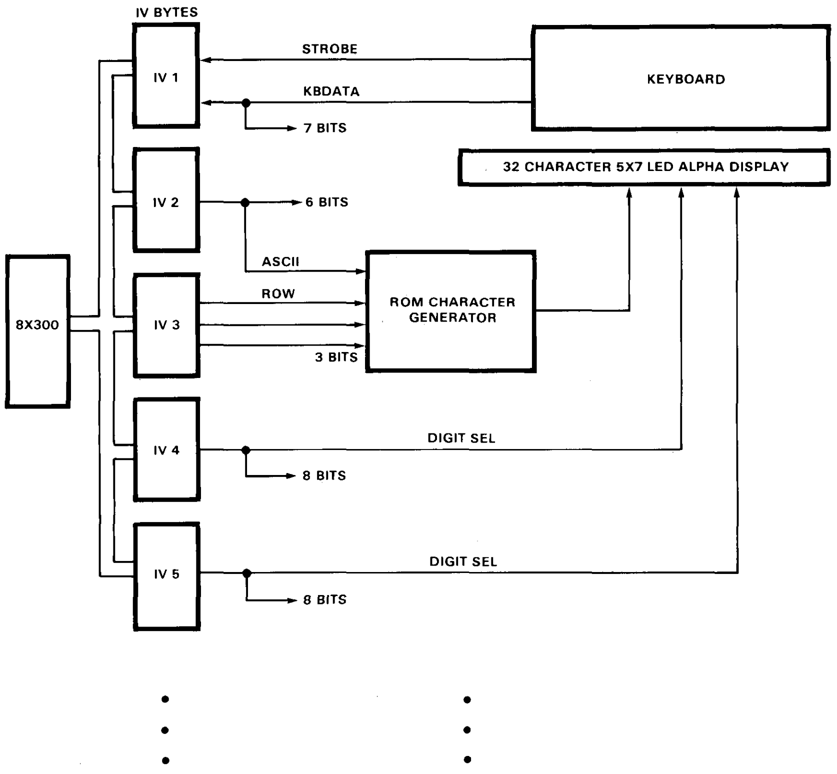
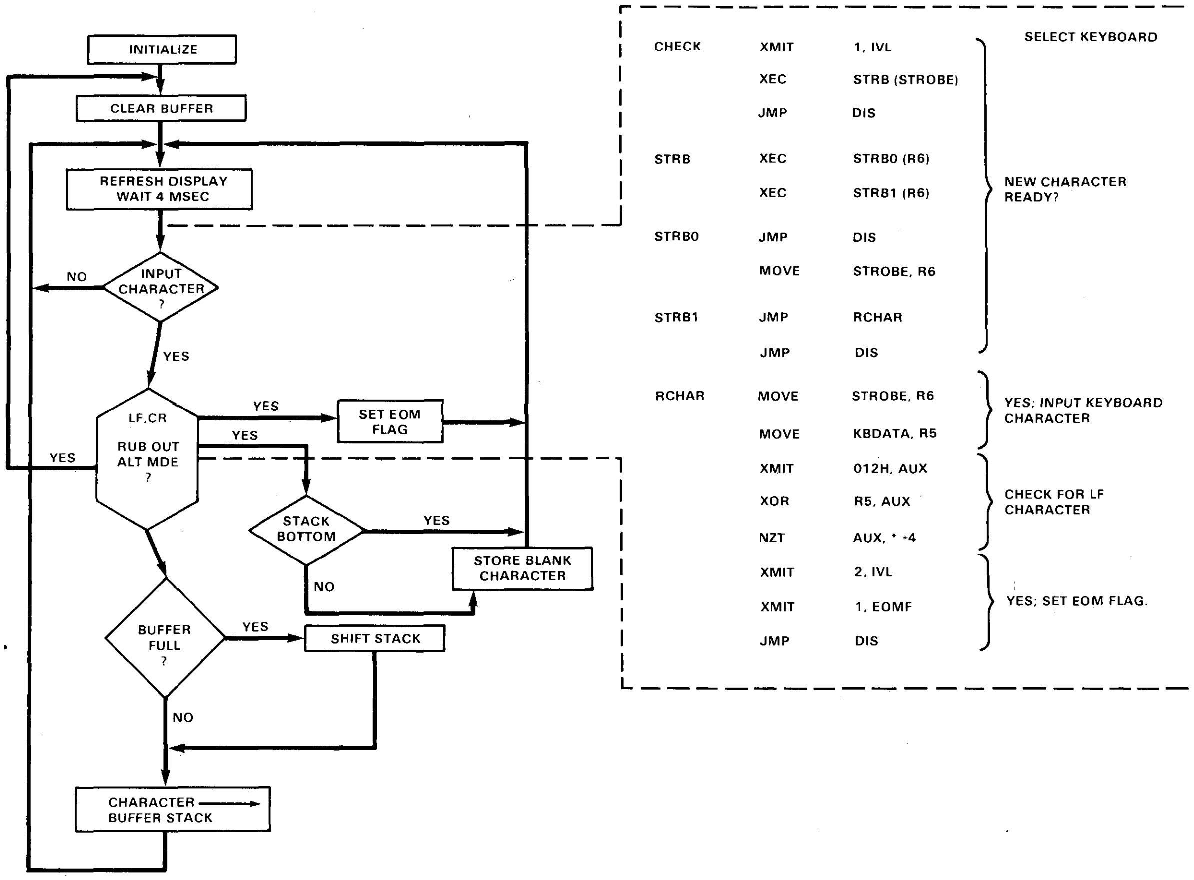


Figure 7

SIGNAL NAME	DATA RATE	SIGNAL DURATION	ELECTRICAL CHARACTERISTICS	# IV BITS	INTERFACE REQUIRED	FUNCTION
STROBE	level	4msec (min)	TTL	1	-	Input Character ready
KBDATA	level	4msec (min)	TTL	7	-	Keyboard input character
ASCII	level	16.6msec (max) 200ns (min)	TTL	6	-	Select character
ROW	level		TTL	3	-	Select row of digit
DIGIT SEL	level		TTL	32	-	Select digit for display

Table 7 INTERFACE ANALYSIS

FUNCTIONAL ANALYSIS



```
CHECK   XMIT   1, IVL           SELECT KEYBOARD
        XEC    STRB (STROBE)
        JMP    DIS
STRB    XEC    STRB0 (R6)
        XEC    STRB1 (R6)       NEW CHARACTER
STRB0   JMP    DIS              READY?
        MOVE   STROBE, R6
STRB1   JMP    RCHAR
        JMP    DIS
RCHAR   MOVE   STROBE, R6       YES; INPUT KEYBOARD
        MOVE   KBDATA, R5       CHARACTER
        XMIT   012H, AUX
        XOR    R5, AUX          CHECK FOR LF
        NZT    AUX, *+4         CHARACTER
        XMIT   2, IVL
        XMIT   1, EOMF          YES; SET EOM FLAG.
        JMP    DIS
```

Figure 8

ROM/PROM FOR PROGRAM STORAGE	WORKING STORAGE FOR DATA BUFFERS	IV BYTES FOR INPUT/OUTPUT INTERFACE
Keyboard/driver .... 140 words	32 bytes per display	41 IV bits for output per display 8 IV bits for input per display Total: 7 IV bytes per display

Table 8 8X300 CONFIGURATION

8X300 MICROPROCESSOR APPLICATIONS MEMO

COMPUTER I/O BUS EMULATOR

DESCRIPTION

The 8X300 system emulates a Microdata 1600 I/O bus. Microdata I/O bus compatible peripherals may then be easily connected to and controlled by a standard 8X300 system. A Microdata I/O bus driver program provides a standard software interface to peripheral devices and requires only 27 words of program storage. Figure 9 illustrates the system.

DESIGN APPROACH

Data bytes are transferred to and from the I/O bus in accordance with Microdata I/O bus specifications. Control signal timing and data transfer sequences are generated by programming. A four step procedure is followed to implement the design:

1. Analyze interface. Analyze Microdata I/O bus relative to: Number of control/data lines, timing and data rates associated with each control/data line, electrical characteristics of each control/data line and determine supplemental circuits needed for electrical compatibility (see Table 9).
2. Perform functional analysis. The functions to be programmed and any which require supplemental logic are determined. In this case, no supplemental logic is required. The programmed functions are:
   a. Transfer bytes in and out
   b. Generate control signal timing and data transfer sequences
3. Define the program to process input and to generate output (see Figure 10).
4. Determine the 8X300 configuration (see Table 10).

COMPUTER I/O BUS EMULATOR

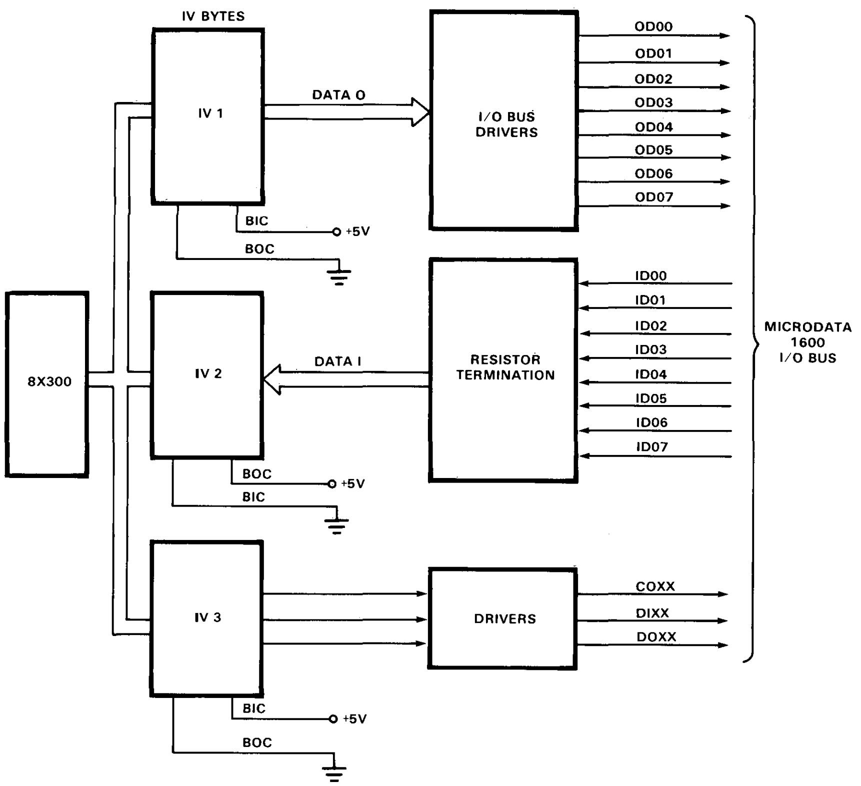


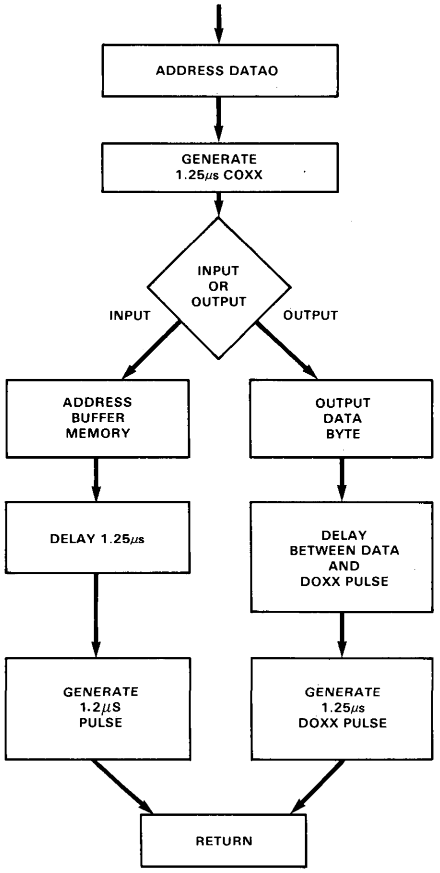
Figure 9

SIGNAL NAME	DATA RATE	SIGNAL DURATION	ELECTRICAL CHARACTERISTICS	# IV BITS	INTERFACE REQUIRED	FUNCTION
OD00-07	level		open collector	8	8D	Data/address from computer
ID00-07	level		TTL	8	8R	Data to computer
COXX	4μs	1.25μs	open collector	1	D	Control output timing
DIXX	4μs	1.25μs	open collector	1	D	Data input timing
DOXX	4μs	.75-1.25μs	open collector	1	D	Data output timing

D = Open collector driver
R = Resistors

Table 9 INTERFACE ANALYSIS

FUNCTIONAL ANALYSIS



```
MICROIO  XMIT     1, IVL
         MOVE     R1, DATAO
         XMIT     3,IVL
------------------------------------------
         XMIT     1, COXX       PULSE ON
         XEC      TAB (R3)
         NOP
         NOP
         XMIT     0, COXX       PULSE OFF
------------------------------------------
         MOVE     R2, IVR
         NZT      R3, INPUT
------------------------------------------
OUTPUT   MOVE     WS, DATAO
------------------------------------------
         NOP
         XMIT     3, IVL
------------------------------------------
         XMIT     1, DOXX       PULSE ON
         NOP
         NOP
         NOP
         XMIT     0, DOXX       PULSE OFF
------------------------------------------
TAB      XMIT     1, IVL
         XMIT     2, IVL
```

Figure 10

ROM/PROM FOR PROGRAM STORAGE	WORKING STORAGE FOR DATA BUFFERS	IV BYES FOR INPUT/OUTPUT INTERFACE
I/O Driver .... 27 words	Depends on peripheral	11 IV bits for output 8 IV bits for input Total: 3 IV bytes per peripheral

Table 10 8X300 CONFIGURATION

8X300 MICROPROCESSOR APPLICATIONS MEMO

INTERFACE TO EXTERNAL READ/WRITE MEMORY

DESCRIPTION

The 8X300 controls the storage, retrieval and processing of large blocks of data. Data is stored in a large capacity (up to 64K bytes) *read/write* RAM *external to the* 8X300 system. The memory is assembled from widely available n-channel (n-MOS) static or dynamic RAM circuits. Minimal interface circuitry is required to connect the 8X300 Interface Vector bytes to the address, data and control lines of the external memory. Figure 11 illustrates the system.

DESIGN APPROACH

Data bytes are read from or written into memory through a single IV type. Two additional IV bytes are used as a 16-bit address register to the external memory. 16 bits provide an address range of 65K bytes. The read/write control signals to the memory require two IV bits. Instruction sequences are used for memory read and memory write operations to implement 1 to 2 microsecond memory access times. A four step procedure is followed to implement the design:

1. Analyze interface. Analyze n-MOS RAM circuits relative to: Number of control/data lines, timing and data rates associated with each control/data line, electrical characteristics of each control/data line and determine any supplemental circuits needed for electrical compatibility (see Table 11).
2. Perform functional analysis. The functions to be programmed and any which require supplemental logic are determined. The programmed functions are:
   a. Store memory address in IV bytes ADRHI, ADRLO.
   b. Set appropriate read/write control bits
   c. Wait for memory operation complete
3. Define the program to process input and to generate output.
   a. GET instruction sequence to read memory location addressed by contents of IV bytes ADRHI, ADRLO (see Figure 12).
   b. PUT instruction sequence to write data into the memory location addressed by the contents of IV bytes ADRHI, ADRLO (see Figure 13).
4. Determine the 8X300 configuration (see Table 12).

INTERFACE TO EXTERNAL READ/WRITE MEMORY

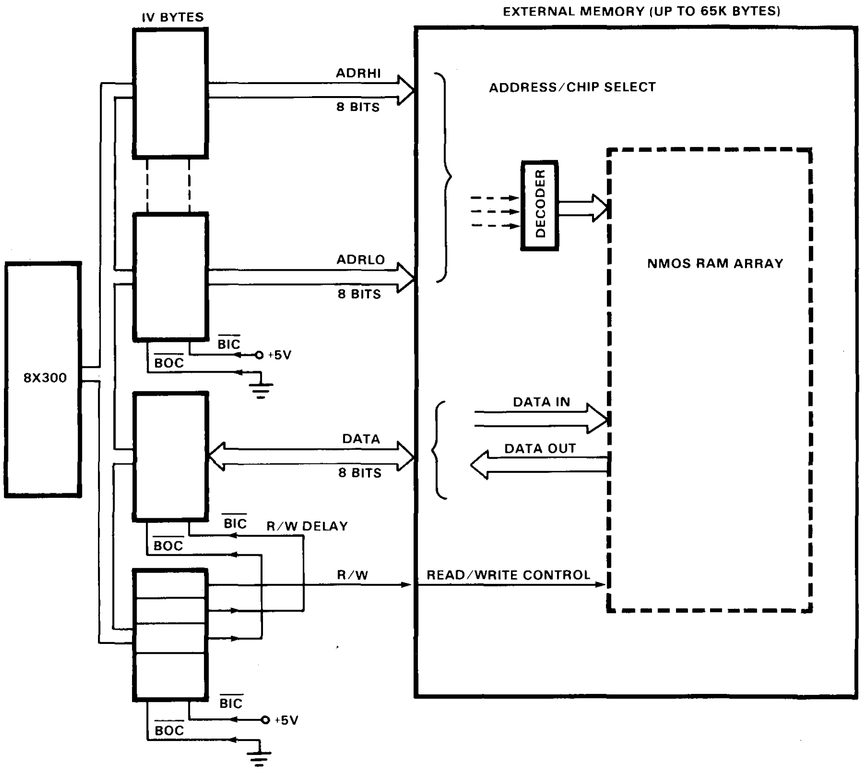
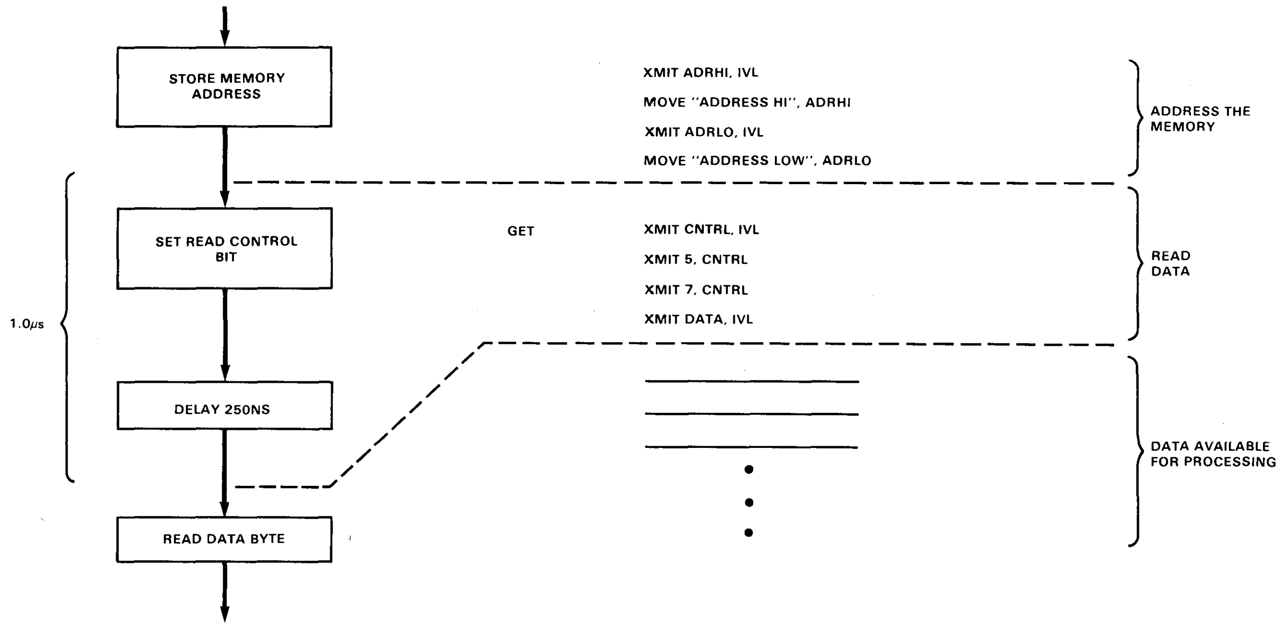


Figure 11

SIGNAL NAME	DATA RATE	SIGNAL DURATION	ELECTRICAL CHARACTERISTICS	# IV BITS	INTERFACE REQUIRED	FUNCTION
ADRHI	Level		TTL	8	* none	Most significant byte. Memory address, and chip select input
ADRLO	Level		TTL	8	* none	Least significant byte memory address
DATA	Level		TTL	8	* none	Memory data
R/W	500ns (min)	>250ns	TTL	1	* none	Memory read/write control
R/W DELAY	500ns (min)	>500ns	TTL	1	* none	Data enable delay during memory write

Table 11 INTERFACE ANALYSIS

GET INSTRUCTION SEQUENCE

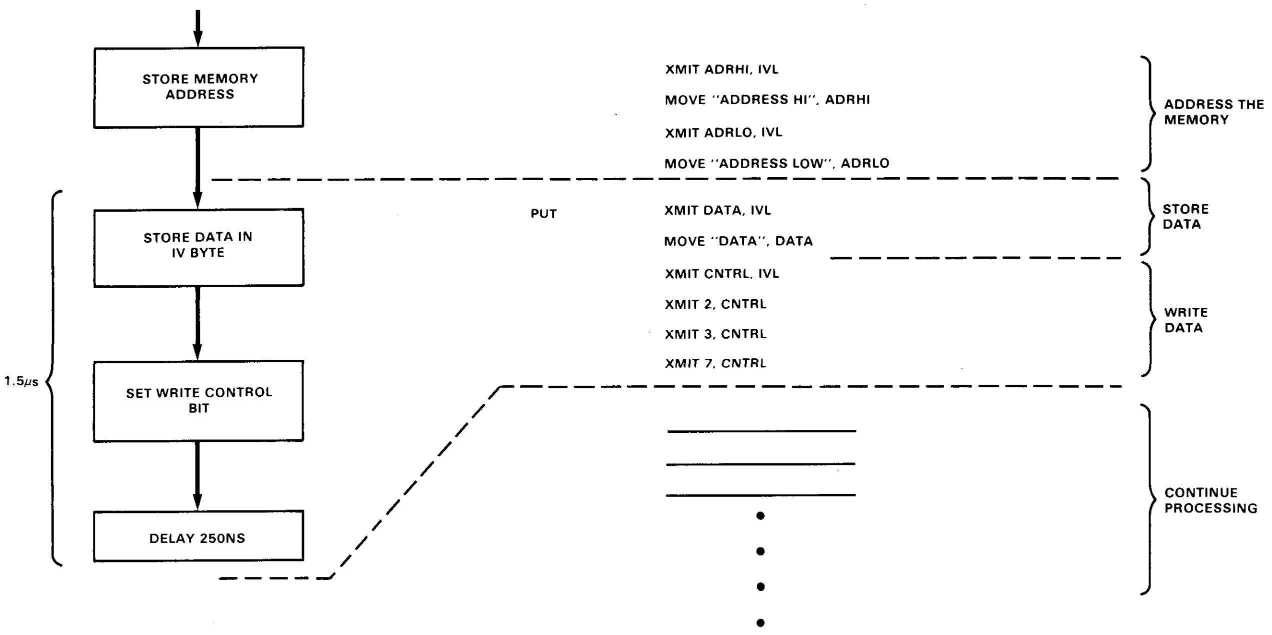


```
XMIT ADRHI, IVL
MOVE "ADDRESS HI", ADRHI
XMIT ADRLO, IVL
MOVE "ADDRESS LOW", ADRLO

XMIT CNTRL, IVL
XMIT 5, CNTRL
XMIT 7, CNTRL
XMIT DATA, IVL
```

Figure 12

PUT INSTRUCTION SEQUENCE



```
            XMIT ADRHI, IVL
            MOVE "ADDRESS HI", ADRHI
            XMIT ADRLO, IVL
            MOVE "ADDRESS LOW", ADRLO
PUT         XMIT DATA, IVL
            MOVE "DATA", DATA
            XMIT CNTRL, IVL
            XMIT 2, CNTRL
            XMIT 3, CNTRL
            XMIT 7, CNTRL
```

Throughput:
Single byte transfer time

Memory read:	833k bytes per second
Memory write:	555k bytes per second

For multiple time transfers, throughput is reduced by the time to loop and update addresses and byte counts to approximately:

Memory read:	250k bytes per second
Memory write:	200k bytes per second

Figure 13

ROM/PROM FOR PROGRAM STORAGE	WORKING STORAGE	IV BYTES FOR INPUT/OUTPUT INTERFACE
GET sequence .... 4 words PUT sequence .... 6 words	None	18 IV bits for output 8 IV bits for input and output Total: 4 IV bytes

Table 12 8X300 CONFIGURATION

256 WAY BRANCH

DESCRIPTION

Many data communication applications require conversion of one code structure to another. The 8X300's Execute instruction provides a fast and efficient method of performing this conversion.

A single Execute instruction can provide up to a 256 way branch based on a byte stored in a register.

This assumes one of the 256 values does not occur during operation of the Execute table. This is easily prevented by testing for one of the values before entering the table, thereby completing the 256 way branch. The example in Figure 14 details how the test for R1 equal to zero is performed first (NZT). If zero, the appropriate conversion value is loaded into R3 (XMIT). If not zero, then the Execute table determines which of the other 255 combinations is in R1 and loads the appropriate conversion value in R3.

The 256 way branch requires 260 words of program storage and 1.0 microseconds maximum to execute. The Execute table and the Execute instruction must all be located with one 256 byte page where the first instruction address contains zeros in the 8 least significant bits. The other four instructions may be placed anywhere within the 8X300's address space.

8X300 EXECUTE INSTRUCTION

```
CONVERT:      NZT     R1, PAGE            "BRANCH IF R1 ≠ 0.
              XMIT    VALUE, R3           "SET R3 BASED ON R1 = 0.
              JMP     NEXT
   PAGE:      JMP     TABLE

              XMIT    VALUE, R3           "MUST BE AT EVEN PAGE ADDRESS
               |
               |     255 XMIT INSTRUCTIONS SET
               |     R3 TO A VALUE BASED ON
               |     CODE COMBINATION IN R1.
               ↓
  TABLE:      XEC .   TABLE (R1)          "TABLE = 377.
   NEXT:       •
               •
               •

PROGRAM STORAGE—260 WORDS
MAXIMUM EXECUTABLE INSTRUCTIONS—4; 1.0µS
```

NOTE

Value is the appropriate conversion code for each code combination in R1.

Figure 14

8X300 MICROPROCESSOR APPLICATIONS MEMO

FAST IV SELECT

DESCRIPTION

The fast IV select is implemented by adding bits to the instruction word, in increments of 4 or 8 bits. This technique allows IV bytes and working storage to be selected within the same instruction where it is used. This can save important processor time by saving one instruction cycle for each select instruction. It eliminates the need for the IV select instruction. It trades fewer instruction cycle times for hardware. It also trades 16-bit select instructions for 4 to 8-bit select fields, thus saving 8 to 12 bits of program storage for every select instruction saved. To some extent, this reduces the cost impact of a larger instruction word. The technique can be used on both IV and buffer storage (including working storage). When used on IV, a decoder is used following an address hold latch to select one IV per address combination. Buffer storage does not require the decoder, instead it utilizes the address directly.

The fast select IV can be used on the same system with normal select IV since all the fast select IV contains the same address. The Master Enable (ME) input of each fast select IV is enabled by the AND of Bank Select (LB, RB) and the single line decode.

Due to memory access delays, the clock used to latch the fast select address is delayed with a couple of inverter delays to assure address validity. On large systems, there are extra delays which may require the address to be programmed in the instruction prior to its usage. Then a double set of address hold latches are used so the address will appear sufficiently early.

FAST IV SELECT CONFIGURATION FOR SMALL SYSTEMS

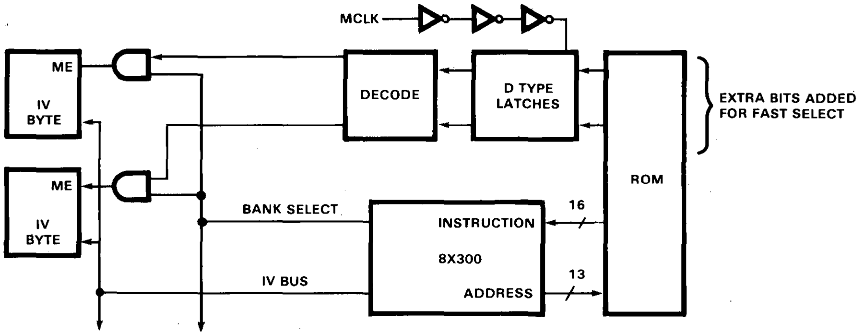


Figure 15

CONFIGURATION FOR LARGE SYSTEMS

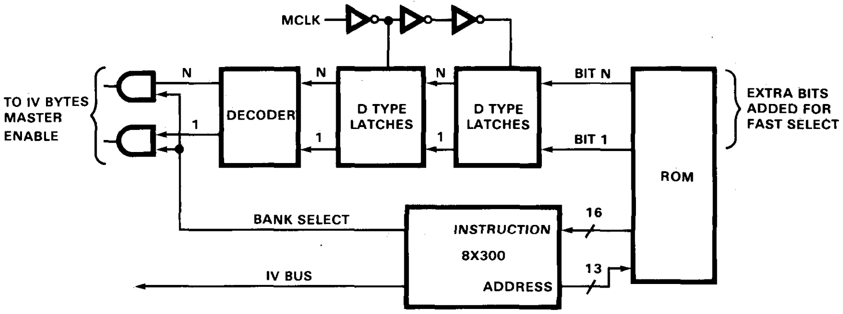


Figure 16

DESCRIPTION

The 8X300 has a repertoire of 8 instruction classes which allow the user to test input status lines, set or reset output control lines, and perform high speed input/output data transfers. All instructions are 16 bits in length. Each instruction is fetched, decoded and executed completely in 250ns.

Data is represented as an 8-bit byte; bit positions are numbered from left to right, with the least significant bit in position 7.

0	1	2	3	4	5	6	7
MSB							LSB

Within the Interpreter, all operations are performed on 8-bit bytes. The Interpreter performs 8-bit, unsigned 2's complement complement arithmetic.

INSTRUCTION FORMATS

The general 8X300 instruction format is:

0 1 2	3 4 5 6 7 8 9 10 11 12 13 14 15
Op Code	Operand(s)

Table 1 contains a summary of the 8X300 instruction set and description of the operand fields.

All instructions are specified by a 3-bit Operation (Op) Code field. The operand may consist of the following fields: Source (S) field, Destination (D) field, Rotate/Length (R/L) field, Immediate (I) Operand field, and (Program Storage) Address (A) field.

The instructions are divided into 5 format types based on the Op Code and the form of the Operand(s) as shown in Figure 1.

OPERATION	FORMAT	RESULT	NOTES
MOVE	Type I Type II	Content of data field addressed by S, R/L replaces data in field specified by D, R/L.	If S and D both are register addresses then R L specifies a right rotate of R/L places applied to the register specified by S.
ADD		Sum of AUX and data specified by S, R/L replaces data in field specified by D, R/L.	
AND		Logical AND of AUX and data specified by S, R/L replaces data in field specified by D, R/L.	
XOR		Logical exclusive OR of AUX and data specified by S, R/L replaces data in field specified by D, R/L.	
XMIT	Type II Type IV	The literal value I replaces the data in the field specified by S, L.	If S is IV or WS address then I limited to range 00-37. Otherwise I limited to range 000-377. If S specifies an IV or WS address then I is limited to the range 00-37. I is limited to the range 000-377 otherwise. The offset operation is performed by reducing the value of PC to the nearest multiple of 32 (if I : 00-37) or 256 (if I : 000-377) and adding the offset.
NZT		If the data in the field specified by S, L equals zero, perform the next instruction in sequence. If the data specified by S,L is not equal to zero, execute the instruction at address determined by using the literal I as an offset to the Program Counter.	
XEC		Perform the instruction at address determined by applying the sum of the literal I and the data specified by S,L as an offset to the Program Counter. If that instruction does not transfer control, the program sequence will continue from the XEC instruction location.	
JMP	Type V	The literal value I replaces contents of the Program Counter.	I limited to the range 00000-07777.

Table 1 8X300 INSTRUCTION SET

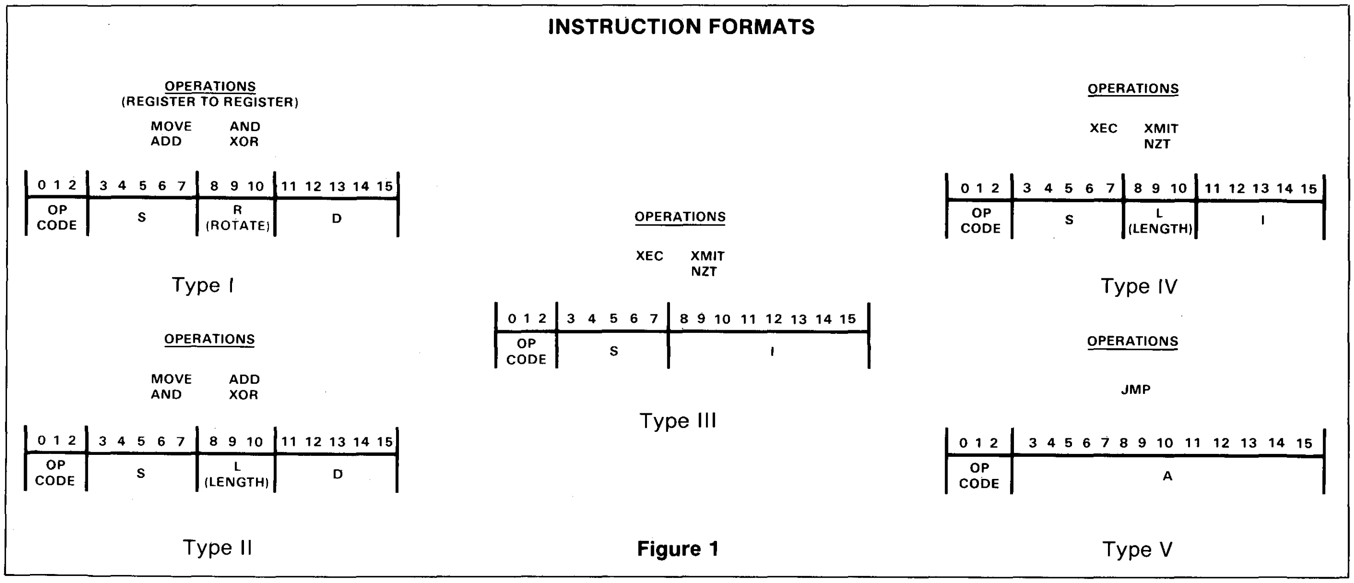


Figure 1

INSTRUCTION FIELDS

Op Code Field (3-Bit Field)

The Op Code field is used to specify 1 of 8 8X300 instructions as shown in Table 2.

S,D Fields (5-Bit Fields)

The S and D fields specify the source and destination of data for the operation defined by the Op Code field. The Auxiliary Register is the implied source for the instructions ADD, AND and XOR which require two source fields. That is, instructions of the form:

ADD X, Y

imply a third operand, say Z, located in the Auxiliary Register so that the operation which takes place is actually X + Z, with the result stored in Y. This powerful capability means that 3 operands are referenced in 250ns.

The S and/or D fields may specify a register, or a 1 to 8-bit I/O field, or a 1 to 8-bit Working Storage field. S and D field value assignments in octal are shown in Table 3.

R/L Field (3-Bit Field)

The R/L field performs one of two functions, specifying either a field length (L) or a right rotation (R). The function it specifies for a given instruction depends upon the contents of the S and D fields:

A. When both S and D specify registers, the R/L field is used to specify a right rotation of the data specified by the S field. (Rotation occurs on the bus and not in the source register.) The register source data is right rotated within one instruction cycle time independent of the number (0 to 7) of bit positions specified in the R/L field.

B. When either or both the S and D fields specify an IV or Working Storage data field, the R/L field is used to specify the length of the data field (within the byte) accessed, as shown in Figure 2.

I Field (5/8-Bit Field)

The I field is used to load a literal value (a binary value contained in the instruction into a register, IV or Working Storage data field or to modify the low order bits of the Program Counter.

The length of the I field is based on the S field in XEC, NZT, and XMIT instructions.

A. When S specifies a register, the literal I is an 8-bit field (Type III format).

B. When S specifies an IV or Working Storage data field, the literal I is a 5-bit field (Type IV format).

A Field (13-Bit Field)

The A field is a 13-bit Program Storage address field. This allows the 8X300 to directly address 8192 instructions.

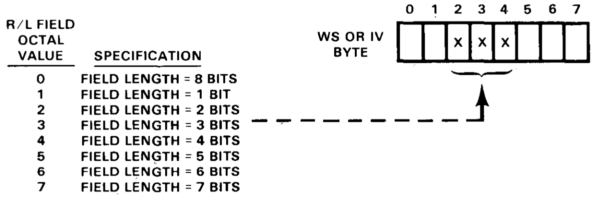
REGISTER OPERATIONS

When a register is specified as the source, and an IV or Working Storage field as the destination, the least significant bits of the operations (MOVE, ADD, AND, XOR) are merged with the original destination data. The least significant bits of the result are stored in the IV or Working Storage data field specified by the D and R/L fields in the instruction.

OP CODE OCTAL VALUE	INSTRUCTION		RESULT
0	MOVE	S,R/L,D	(S) → D
1	ADD	S,R/L,D	(S) plus (AUX) → D
2	AND	S,R/L,D	(S) ∧ (AUX) → D
3	XOR	S,R/L,D	(S)⊕(AUX) → D
4	XEC	I,R/L,S or I,S	Execute instruction at current PC offset by I + (S)
5	NZT	I,R/L,S or I,S	Jump to current PC offset by I if (S) ≠ 0
6	XMIT	I,R/L,D or I,D	Transmit literal I → D
7	JMP	A	Jump to program location A

Table 2 INSTRUCTION SET SUMMARY

R/L FIELD



R/L FIELD OCTAL VALUE	SPECIFICATION
0	FIELD LENGTH = 8 BITS
1	FIELD LENGTH = 1 BIT
2	FIELD LENGTH = 2 BITS
3	FIELD LENGTH = 3 BITS
4	FIELD LENGTH = 4 BITS
5	FIELD LENGTH = 5 BITS
6	FIELD LENGTH = 6 BITS
7	FIELD LENGTH = 7 BITS

Figure 2

When an IV or Working Storage field of 1 to 8 bits is specified as the source, and a register as the destination, the 8-bit result of the operations (MOVE, ADD, AND, XOR) is stored in the register. The operations ADD, AND, XOR actually use the IV or Working Storage data field (1 to 8 bits) with leading zeros to obtain 8-bit source data for use with the 8-bit AUX data during the operation.

Because IVL and IVR are write-only pseudo registers, they can be specified as destination fields only (see Table 3). Operations involving IVL and IVR as sources are not possible. For example, it is not possible to increment IVR or IVL in a single instruction, and the contents of IVL or IVR cannot be transferred to a working register, IV byte, or Working Storage location.

The OVF (Overflow) Register can only be used as a source field; it is set or reset only by the ADD instruction.

INSTRUCTION DESCRIPTIONS

The following instruction descriptions employ MCCAP (the 8X300 Cross Assembly Program) programming notation. This notation varies somewhat from the instruction descriptions provided in Tables 1 and 3. Thus, for example, explicit L field definition, as shown in Table 1 and Table 3, is not required by MCCAP instructions; MCCAP creates appropriate variable field addresses from the information contained in the Data Declaration statements provided by the programmer at the beginning of his program.

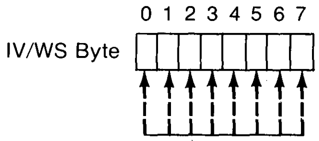
The 8X300 instruction set is described below with examples shown in Figures 3 through 10.

0_8-17_8 is used to specify 1 of 7 working registers (R1-R6, R11), Auxiliary Register, Overflow Register, IVL and IVR write-only registers.

OCTAL VALUE		OCTAL VALUE	
00	Auxiliary Register (AUX)	10	OVF-Overflow register-Used as an S (source) field only.
01	R1	11	R11
02	R2	12	Unassigned
03	R3	13	Unassigned
04	R4	14	Unassigned
05	R5	15	Unassigned
06	R6	16	Unassigned
07	IVL Register-IV Byte address write-only register-Specified only in D field in all instructions	17	IVR Register-Working Storage address write-only register-Specified only in D field in all instructions

a. Register Specification

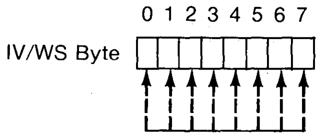
20-$_8$-27_8 is used to specify the least significant bit of a variable length field within the IV/WS Byte previously selected by the IVL register. The length of the field is determined by R/L.



OCTAL VALUE	
20	Field within previously selected IV/WS Byte; position of LSB = 0
21	Field within previously selected IV/WS Byte; position of LSB = 1
22	Field within previously selected IV/WS Byte; position of LSB = 2
23	Field within previously selected IV/WS Byte; position of LSB = 3
24	Field within previously selected IV/WS Byte; position of LSB = 4
25	Field within previously selected IV/WS Byte; position of LSB = 5
26	Field within previously selected IV/WS Byte; position of LSB = 6
27	Field within previously selected IV/WS Byte; position of LSB = 7

b. Left Bank Field Specification

30_8-37_8 is used to specify the least significant bit of a variable length field within the IV/WS Byte previously selected by the IVR Register. The length of the field is determined by R/L.



OCTAL VALUE	
30	Field within previously selected IV/WS Byte; position of LSB = 0
31	Field within previously selected IV/WS Byte; position of LSB = 1
32	Field within previously selected IV/WS Byte; position of LSB = 2
33	Field within previously selected IV/WS Byte; position of LSB = 3
34	Field within previously selected IV/WS Byte; position of LSB = 4
35	Field within previously selected IV/WS Byte; position of LSB = 5
36	Field within previously selected IV/WS Byte; position of LSB = 6
37	Field within previously selected IV/WS Byte; position of LSB = 7

c. Right Bank Field Specification

Table 3 S AND D FIELD SPECIFICATIONS

MOVE S,D or MOVE S(R),D

Format: Type I, Type II

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	Source					R/L			Destination				

Operation: (S)→(D)

Description

Move data. The contents of S are transferred to D; the contents of S are unaffected. If both S and D are registers, R/L specifies a right rotate of the source data before the move. Otherwise, R/L is implicit and specifies the length of the source and/or destination IV/WS field. If the MOVE is between an IV byte and a Working Storage byte, an 8-bit field must always be moved.

Example

Store the least significant 3 bits of register 5 (R5) in bits 4, 5 and 6 of the IV byte previously addressed by the IVL register.

ADD S,D or ADD S(R),D

Format: Type I, Type II

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	1	Source					R/L			Destination				

Operation

(S) plus (AUX) = D; set OVF if carry from most significant bit (bit 0) occurs.

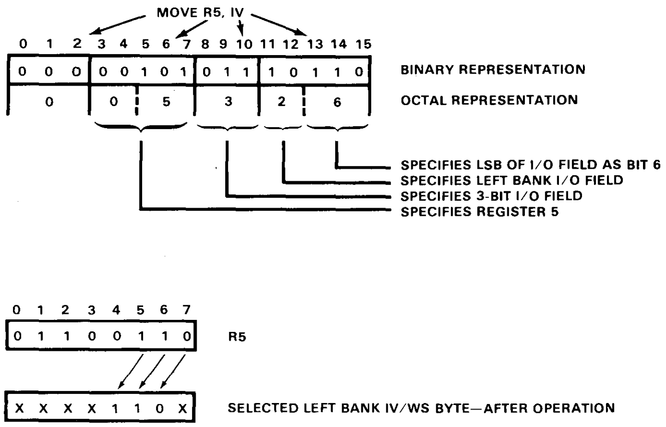
Description

Unsigned 2's complement 8-bit addition. The contents of S are added to the contents of the Auxiliary Register (which is the implied source). The result is stored in D; OVF is updated. If both S and D are registers, R/L specifies a right rotate of the source (S) data before the operation. Otherwise, R/L is implicit and specifies the length of the source and/or destination IV/WS fields. S and AUX are unaffected unless specified as the destination.

Example

Add the contents of R1 (rotated 4 places) to AUX and store the result in R3.

MOVE S,D or MOVE S(R),D



NOTE: X'S IN THE IV BYTE DENOTE BITS UNAFFECTED BY THE MOVE OPERATION.

Figure 3

ADD S,D or ADD S(R),D

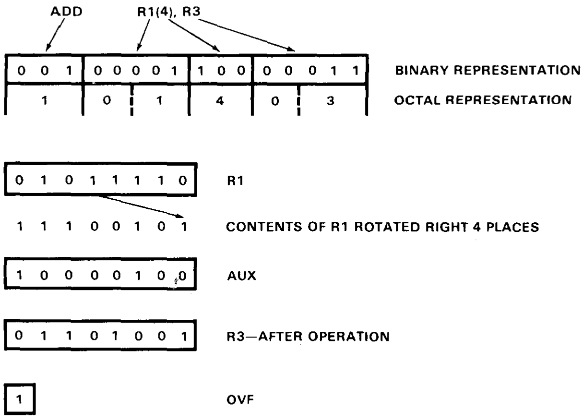


Figure 4

AND S,D or AND S(R),D

Format: Type I, Type II

0 1 2	3 4 5 6 7	8 9 10	11 12 13 14 15
0 1 0	Source	R/L	Destination

Operation: (S) ∧ (AUX)→D

Description

Logical AND. The AND of the source field and the Auxiliary Register is stored into the destination. If both S and D are registers, R/L specifies a right rotate of the source (S) data before the AND operation. Otherwise R/L is implicit and specifies the length of the source and/or destination IV/WS fields. S and AUX are unaffected unless specified as a destination.

Example

Store the AND of the selected right bank byte and AUX in R4. The right bank data field is called WSBCD and is 4 bits long and located in bits 2, 3, 4 and 5.

XOR S,D or XOR S(R),D

Format: Type I, Type II

0 1 2	3 4 5 6 7	8 9 10	11 12 13 14 15
0 1 1	Source	R/L	Destination

Operation: (S) ⊕ (AUX)→D

Description

Exclusive OR. The exclusive OR of the source field and the Auxiliary Register is stored in the destination. If both S and D are registers, R/L specifies a right rotate of the source (S) data before the XOR operation. Otherwise R/L is implicit and specifies the length of the source and/or destination IV/WS fields. S and AUX are unaffected unless specified as a destination.

Example

Replace the selected IV byte field with the XOR of the field and AUX. The IV byte field is called STATUS and is 5 bits in length and located in bits 3, 4, 5, 6 and 7 of LB.

AND S,D or AND S(R),D

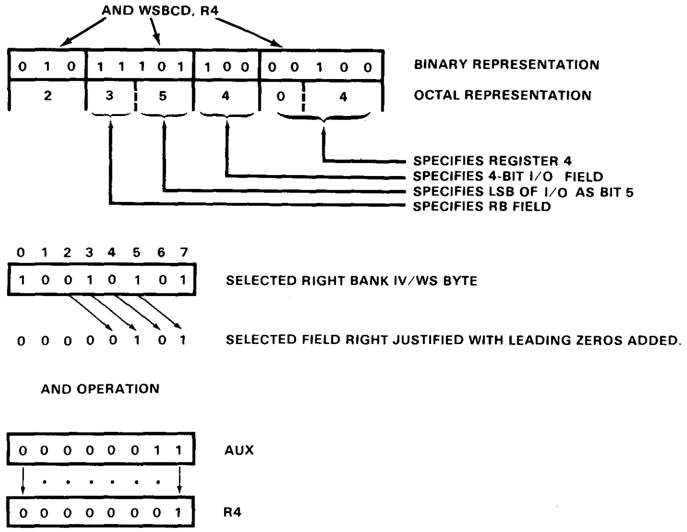


Figure 5

XOR S,D or XOR S(R),D

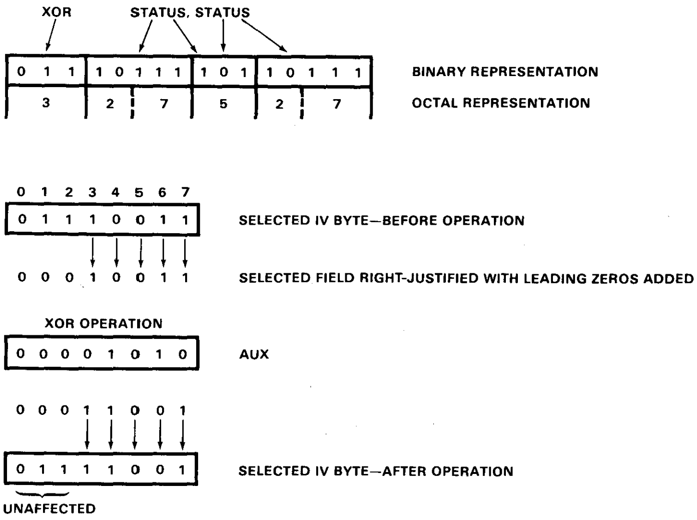


Figure 6

XEC I(S) or XEC I(S,L)

Format:

0 1 2	3 4 5 6 7	8 9 10 11 12 13 14 15
1 0 0	Source	I Field

Type III

0 1 2	3 4 5 6 7	8 9 10	11 12 13 14 15
1 0 0	Source	R L	I Field

Type IV

Operation

Execute instruction at the address specified by the Address Register with lower 5 or 8 bits replaced by (S) + I.

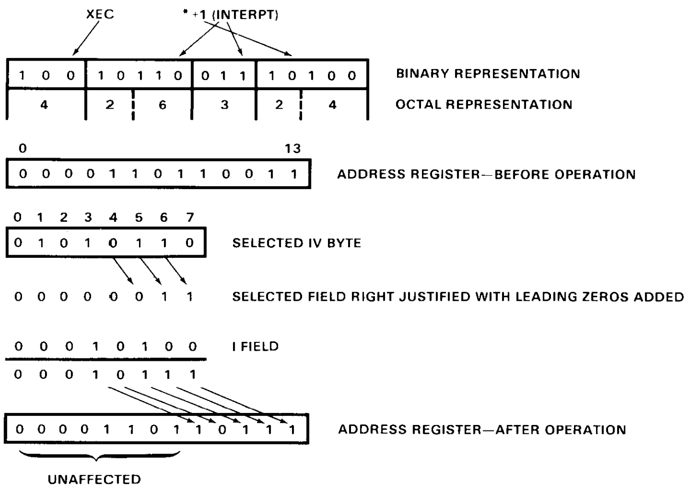
Description

Execute the instruction at the address determined by replacing the low order bits of the Address Register (AR) with the low order bits of the sum of the literal I and the contents of the source field. If S is a register, the low order 8 bits of AR are replaced; if S is an IV or Working Storage field, the low order 5 bits of AR are replaced, resulting in an execute range of 256 and 32 respectively. The Program Counter is not affected unless the instruction executed is a JMP or NZT (whose branch is taken).

Example

Execute one of n JMPs in a table of JMP instructions determined by the value of the selected IV byte field. The table follows immediately after the XEC instruction and the IV field is called INTERPT and is a 3-bit field located in bits 4, 5 and 6.

XEC I(D)



ADDRESS	INSTRUCTION	
0000110110011	XEC *+1 (INTERPT)	
0000110110100	JMP A1	
•		
0000110110111	JMP A4	JMP TABLE
•		
0000110111011	JMP A8	

JMP A4 IS EXECUTED BECAUSE IV FIELD INTERPT = 3

Figure 7

XMIT I,D or XMIT I,D,L

Format:

0 1 2	3 4 5 6 7	8 9 10 11 12 13 14 15
1 1 0	Destination	I Field

Type III

0 1 2	3 4 5 6 7	8 9 10	11 12 13 14 15
1 1 0	Destination	R L	Destination

Type IV

Operation: I→(D)

Description

Transmit literal. The literal field I is stored in D. If D is a register, an 8-bit field is transferred; if D is an IV or Working Storage field, up to a 5-bit field is transferred.

Example

Store the bit pattern 110 in the selected Working Storage field. The field name is VALUE and is located in bits 3, 4 and 5.

XMIT I,D

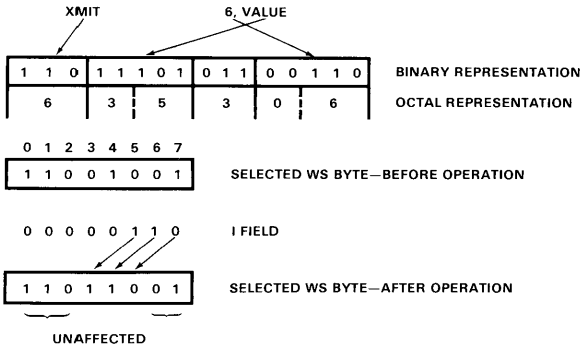
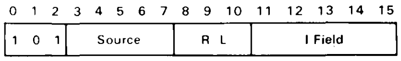


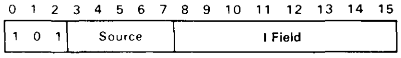
Figure 8

NZT S,I

Format:



Type III



Type IV

Operation

Non-Zero Transfer. If (S) ≠ 0, PC offset by I → PC; otherwise PC + 1 → PC.

Description

If the data specified by the S field is non-zero, replace the low order bits of the Program Counter with I. Otherwise, processing continues with the next instruction in sequence. If S is a register, the low order 8 bits of the PC are replaced; if S is an IV or Working Storage field, the low order 5 bits of the PC are replaced, resulting in an NZT range of 256 and 32 respectively.

Example

Jump to Program Address ALPHA if the selected IV byte field is non-zero. The field name is OVERFLO and it is a 1-bit field located in bit 3.

NZT S,I

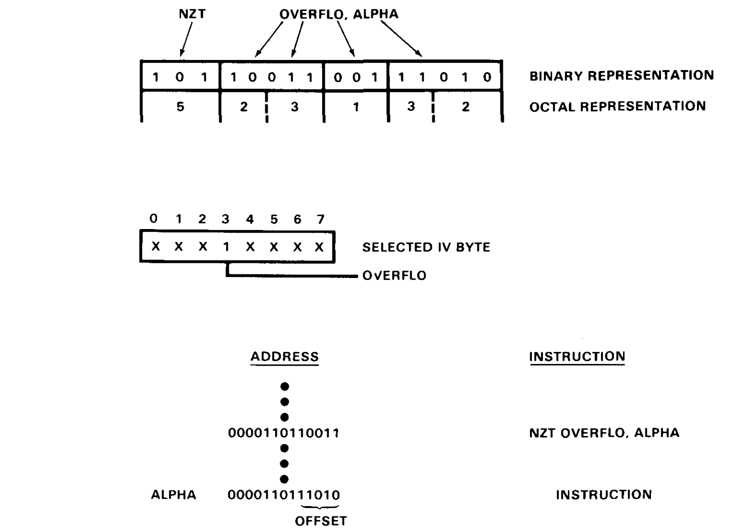
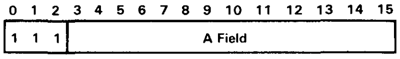


Figure 9

JMP A

Format: Type V



Operation: A → PC

Description

The literal value A is placed in the Program Counter and Address Register, and processing continues at location A. A has a range of 0-17777$_8$ in current systems (0-8191).

Example

Jump to location ALPHA (0000101110001)

JMP A

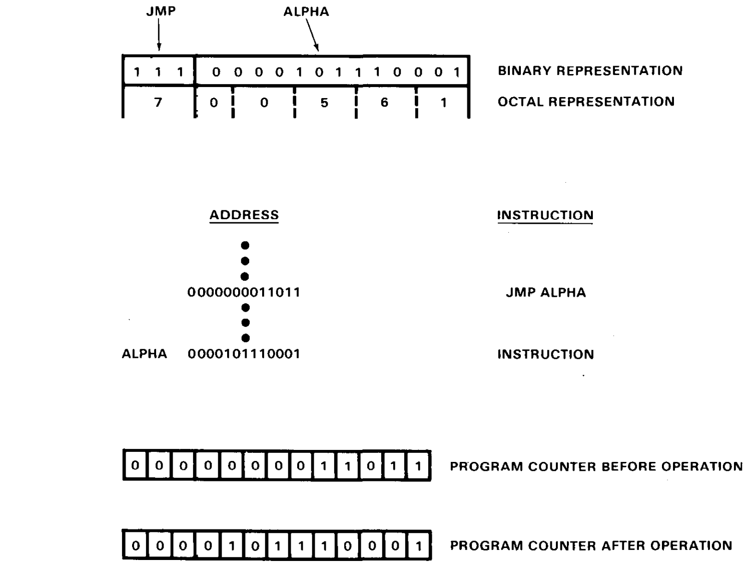


Figure 10

DESCRIPTION

The 8X300 Cross Assembly Program, MCCAP, provides a programming language which allows the user to write programs for the 8X300 in symbolic terms. MCCAP translates the user's symbolic instructions into machine-oriented binary instructions. For example, the jump instruction, JMP, to a user defined position, say ALPHA, in program storage is coded as:

JMP ALPHA

and is translated by MCCAP into the following 16-bit word (see Figure 1).

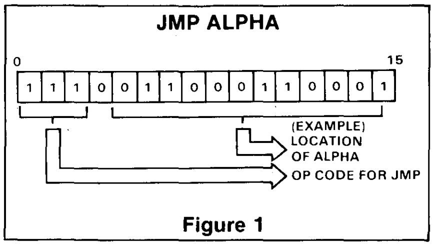


Figure 1

MCCAP allocates the 8X300 program storage and assigns Interface Vector and Working Storage address to symbols as declared in the user's program.

The ability to define data of the Interface Vector as symbolic variables is a powerful feature of MCCAP. Interface Vector variables may be operated on directly using the same instructions as those for variables in Working Storage and for the working registers.

The Assembler Declaration statements of MCCAP allow the programmer to define symbolic variable names for data elements tailored to his application. Individual bits and sequences of bits in Working Storage and on the Interface Vector may be named and operated upon directly by 8X300 instructions.

In addition to simplifying the language and bookkeeping of the program, MCCAP provides program segmentation and communication between segments; i.e., the main program and any subprograms. If a sequence of code appears more than once in a program, it can be written as a separate program segment, a subprogram, and called into execution whenever that subprogram's function is required. Program segmentation also permits the construction of a program in logically discrete units. These segments need not be written sequentially or even by the same person. The various program segments provide a function description, or block diagram, of the application. Communication between segments means that control and data can be transferred in both directions. MCCAP automatically generates the code for subprogram entry and exit mechanisms when the appropriate CALL and RTN statements are invoked.

MCCAP SOURCE PROGRAM

```
                    MICROCONTROLLER SYMBOLIC ASSEMBLER VER 1.0

1680                     *
1681                     *
1682 01544                          PROC    RDCMMD
1683                     *
1684 01544  6 07003                 SEL     IVRESP        FDC RESPONSE BYTE
1685 01545  6 20101                 XMIT    UR, BCTRL     ESTABLISH USER READ ONLY
1686 01546  6 07002                 SEL     IVDATA        HOLDS COMMAND BYTE
1687 01547  0 27305                 MOVE    FUNC, R5      FUNCTION CODE
1688 01550  0 24306                 MOVE    DADDR, R6     DISK ADDRESS
1689 01551  0 21202                 MOVE    BUFF, R2      BUFFER FUNCTION CODE
1690 01552  6 07003                 SEL     IVRESP
1691 01553  6 25100                 XMIT    0, DONE       SHOW COMMAND IN PROGRESS
1692 01554  6 20100                 XMIT    UW, BCTRL     RESTORE USER WRITE
1693 01555  6 27101                 XMIT    1, XFR        SIGNAL USER FDC ACCEPTED BYTE
1694 01556  6 07001                 SEL     IVCTRL        USER CONTROL BYTE
1695 01557  5 26117                 NZT     CMMD,*        WAIT FOR CMMD TO GO LOW
1696 01560  6 07003                 SEL     IVRESP        FDC RESPONSE BYTE
1697 01561  6 27100                 XMIT    0, XFR        LOWER XFR SIGNAL
1698 01562  6 07001                 SEL     IVCTRL        USER CTRL BYTE
1699 01563  4 26123                 XEC     *(CMMD), 2    WAIT FOR NEXT COMMAND SIGNAL
1700 01564  6 07003                 SEL     IVRESP        SECOND COMMAND BYTE AVAILABLE
1701 01565  6 20101                 XMIT    UR, BCTRL     SET IVDATA TO USER READ ONLY
1702 01566  6 07002                 SEL     IVDATA        2ND COMMAND BYTE
1703 01567  0 27704                 MOVE    TRACK, R4     TRACK ADDRESS
1704 01570  0 27503                 MOVE    SECT, R3      SECTOR ADDRESS
1705 01571  6 07003                 SEL     IVRESP        FDC RESPONSE BYTE
1706 01572  6 27101                 XMIT    1, XFR        SIGNAL USER
1707 01573  6 20100                 XMIT    UW, BCTRL     RESTORE USER WRITE
1708 01574  6 07001                 SEL     IVCTRL
1709 01575  5 26135                 NZT     CMMD,*        WAIT FOR CMMD TO GO LOW
1710 01576  6 07003                 SEL     IVRESP        FDC RESPONSE BYTE
1711 01577  6 27100                 XMIT    0, XFR        LOWER XFR SIGNAL
1712                     *
1713 01600  7 01652                 RTN                   RETURN
1714                                END     RDCMMD
1715                     *
```

Figure 2

MCCAP OUTPUT

The output from a MCCAP compilation includes an assembler listing and an object module. During pass two of the assembly process, a program listing is produced. The listing displays all information pertaining to the assembled program. This includes the assembled octal instructions, the user's original source code and error messages. The listing may be used as a documentation tool through the inclusion of comments and remarks which describe the function of a particular program segment. The main purpose of the listing, however, is to convey all pertinent information about the assembled program, i.e., the memory addresses and their contents.

The object module is also produced during pass two. This is a machine-readable computer output produced on paper tape. The output module contains the specifications necessary for loading the memory of the Microcontroller Simulator (MCSIM), for loading the memory of the SMS ROM Simulator, or for producing ROMs or PROMs. The object module can be produced in MCSIM, ROM Simulator or BNPF format.

An example of a MCCAP source program is shown in Figure 2.

PROGRAM STRUCTURE

Program Segments

A MCCAP program consists of one or more program segments. Program segments are the logically discrete units, such as the main program and subprograms, which comprise a user's complete program. Program segments consist of sequences of program statements. The first program segment must be the main program. The main program names the overall program and is where execution begins. All other segments are subprograms; each subprogram must be named. Control and data can be passed in both directions between segments. No segment may call itself, or one of its callers, or the main program. Program segments take the form as shown in Figure 3.

The Assembler Declaration statements define variables and constants. They must precede the use of the declared variables and constants in the Executable Statements in a program. The Executable Statements are those which result in the generation of one or more executable machine instructions.

Subprograms

Subprograms are program segments which perform a specific function. A major reason for using subprograms is that they reduce programming and debugging labor when a specific function is required to be executed at more than one point in a program. By

PROGRAM SEGMENTS

PROGRAM STATEMENT
DECLARATION STATEMENT(S)
•
•
•
EXECUTABLE STATEMENT(S)
•
•
SUBPROGRAMS
•
END STATEMENT

a. Main Program Form

PROCEDURE STATEMENT
DECLARATION STATEMENT(S)
•
•
•
EXECUTABLE STATEMENT(S)
•
•
•
END STATEMENT

b. Subprogram Form

Figure 3

creating the required function as a subprogram, the statements associated with that function may be coded once and executed at many different points in a program. Figure 3 illustrates an example.

The program structure in Figure 3 causes the code associated with PROC WAIT to be executed three times within PROG MANY-WAIT. This is accomplished even though the statements associated with PROC WAIT are coded only once, rather than three times.

Subprogram Calls and Returns

For user-provided procedures, a jump to the associated procedure and a return link are created for each procedure reference. The instructions to accomplish this result in subprogram entry time. The instructions to accomplish subprogram exit result in exit time. The user may utilize the MCCAP procedure mechanism for linking calling programs with called programs or he may create his own instructions to do so. The following describes the linkage mechanism and timing for MCCAP user procedures.

Linkage between called and calling programs is achieved through the generation of an indexed "return jump" table, the length of which corresponds to the number of different times in the program that the subprograms are called. This table is generated automatically by MCCAP when procedure CALL and RTN statements are invoked. For each procedure reference, MCCAP creates two statements in the calling program. Thus, the time required for the subprogram entry is 0.5 microseconds. The subprogram return mechanism requires the execution of two instructions or 0.5 microseconds. These times do not include saving and restoring of the working registers. The total time to save all working registers is 3.5 microseconds, the same time to restore all registers. Saving of all working registers is normally not necessary, but worst case calculations for entry and exit time below do include this time. Thus, subprogram exit and entry times are:

$$0.5\mu s \leqslant \text{Entry Time} \leqslant 4.0\mu s$$
$$0.5\mu s \leqslant \text{Exit Time} \leqslant 4.0\mu s$$

Details of the code required for procedure CALL and RTN are provided in the Programming Examples section. See Figures 21 and 22.

Macros

A macro is a sequence of instructions that can be inserted in the assembly source text by encoding a single instruction. The macro is defined only once and may then be invoked any number of times in the program. This facility simplifies the coding of programs, reduces the chance of errors, and makes programs easier to change.

A macro definition consists of a heading, a body and a terminator. This definition must precede any call on the macro. In MCCAP, the heading consists of the MACRO statement which marks the beginning of the macro and names it. The body of the macro is made up of those MCCAP statements which will be inserted into the source code in place of the macro call. The terminator consists of an ENDM statement which marks the physical end of the macro definition.

MCCAP Statements

The MCCAP language consists of thirty statements categorized as follows:

Assembler Directive Statements
Assembler Declaration Statements
Communication Statements
Macro Statements
Machine Statements

The following lists the statements in each category, describes their use, and provides examples. Detailed use of the instructions including rules of syntax and parameter restrictions are described in the MCCAP Reference Manual.

Assembler Directive Statements

Assembler Directive statements define program structure and control the assembler outputs. They do not result in the generation of 8X300 executable code. There are twelve Assembler Directive statements:

PROG Statement
PROC Statement
ENTRY Statement
END Statement
ORG Statement
OBJ Statement
IF Statement
ENDIF Statement
LIST Statement
NLIST Statement
EJCT Statement
SPAC Statement

PROG Statement

Use
Defines the names and marks the beginning of a main program.

Example: PROG PROCESS

PROC Statement

Use
Defines the names and marks the beginning of a subprogram.

Example: PROC WAIT

ENTRY Statement

Use
Defines the name and marks the location of a secondary entry point to a subprogram.

Example: ENTRY POINT 2

END Statement

Use
Terminates a program segment or a complete program.

Examples: END SUB1
END MAIN

ORG Statement

Use
Sets the program counter to the value specified in the operand field.

Example: ORG 200

OBJ Statement

Use
To specify the format of the object module.

Examples: OBJ R
OBJ M
OBJ N

NOTE
"R" indicates the ROM Simulator format. "M" indicates the Microcontroller Simulator format. "N" indicates BNPF format.

IF Statement

Use
To mark the beginning of a sequence of code, which may or may not be assembled depending on the value of an expression.

Examples: IF VAL
IF X + Y

ENDIF Statement

Use

To mark the end of sequence of code, which is to be conditionally assembled. In the case of nested IF statements, an ENDIF is paired with the most recent IF.

Example: ENDIF

LIST Statement

Use

To select and control output of a MCCAP assembly.

Example: LIST S,O,M,I

NLIST Statement

Use

To suppress elements of the output from a MCCAP assembly.

Example: NLIST O,M,I

EJCT Statement

Use

To cause the output listing to be advanced to the next page.

Example: EJCT

SPAC Statement

Use

To insert blank lines into the output listing. The number of lines inserted is indicated in the operand field.

Example: SPAC 3

Assembler Declaration Statement

Assembler Declaration statements define and describe the data, constants and variables, in a program or subprogram. There are four Assembler Declaration statements:

EQU Statement
SET Statement
LIV Statement
RIV Statement

EQU Statement

Use

To define a fixed constant.

Examples: FIVE EQU 5
ON EQU 1

SET Statement

Use

To define and assign a value to a constant, which may later be assigned a new value by another SET statement.

Example: OFF SET 0

LIV Statement

Use

To define and assign symbolic names to variables, usually IV bytes, located on the left bank of the Interface Vector.

Example: LITE LIV 23,2,1

NOTE

The effect of the above example is to define a variable whose name is LITE. It is located in a byte whose address is 23. The right-most bit of LITE is bit 2 and its length is 1 bit.

RIV Statement

Use

To define and assign symbolic names to variables, usually in Working Storage, located on the right bank of the Interface Vector.

Example: DATA RIV 200,6,3

NOTE

The effect of the above example is to define a variable whose name is DATA. It is located in a byte whose address is 200. The right-most bit DATA is bit 6 of the byte and its length is 3 bits.

Communication Statements

Communication statements are executable statements which provide the mechanism for main program to subprogram linkage. They provide the means by which subprograms are called and returned from. There are two kinds of Communication statements:

CALL Statement
RTN Statement

CALL Statement

Use

To transfer control from a calling program to the called subprogram. The CALL statement causes the generation of two 8X300 instructions.

Examples: CALL WAIT
CALL SINE

NOTE

The above are valid statements to be coded into the program if WAIT and SINE have been defined in PROC statements. The effect of invoking these statements is to transfer execution control to the procedures WAIT and SINE respectively.

RTN Statement

Use

To transfer control from a called subprogram to a calling program.

Example: RTN

Macro Statements

Macro statements provide the mechanism for defining macros and for inserting them into the source code. There are three Macro statements:

MACRO Statement
ENDM Statement
MACRO CALL Statement

MACRO Statement

Use

To mark the beginning of a macro definition. The MACRO statement forms the heading of the macro definition.

Examples: MAC1 MACRO
MAC2 MACRO A,B,C

NOTE

The second example would mark the beginning of a macro called MAC2. The "A,B,C" represents a formal parameter list. These parameters, used in writing the macro body, will be replaced by the actual parameters listed in the MACRO CALL statement.

ENDM Statement

Use

To mark the end of a macro definition. The ENDM statement forms the terminator of the macro definition.

Example: ENDM

MACRO CALL Statement

Use

To indicate where a macro is to be inserted into the source code and to specify any actual parameters needed by the macro.

Example: MAC2 DATA, INPUT, RESULT

NOTE

There is no single macro call statement. Any macro name which has been defined as such may be coded as if it were a valid MCCAP statement. The macro name is coded in the operation field and the actual parameters are placed in the operand field.

Machine Statement

Machine statements are the MCCAP symbolic representations of the 8X300 executable statements. Machine statements have a one to one correspondence to 8X300 instructions. Each Machine statement results in the generation of a single 8X300 instruction. There are eight Machine statements:

MOVE Statement
ADD Statement
AND Statement
XOR Statement
XMIT Statement
XEC Statement
NZT Statement
JMP Statement

MOVE Statement

Use

To copy the contents of a specified register, WS variable or IV variable into a specified register, WS or IV. Defined in Instruction Descriptions.

Examples:
```
MOVE R1(6);R6
MOVE X,Y
```

NOTE

The first example illustrates a six place right rotate of R1's data before it is moved to R6. The contents of R1 are not affected. The second example may be a Working Storage or Interface Vector variable move, depending on the way X and Y are defined in Declaration Statements.

ADD Statement

Use

To add the contents of a specified register, WS variable, or IV variable to the contents of the AUX register and place the result in a specified register, WS variable or IV variable.

Examples:
```
ADD R1(3),R2
ADD DATA,OUTPUT
```

NOTE

The first example illustrates a three place right rotate of R1's data before the addition is carried out. Under certain conditions a rotate may be used to multiply the specified operand by a power of 2 before the addition is done. The contents of R1 are nc. affected. The second example suggests that the contents of WS variable have been added to the contents of the AUX register and the result placed in an IV variable, making the result immediately available to the user's system.

AND Statement

Use

To compute the logical AND of the contents of a specified register, WS variable or IV variable and the contents of the AUX register. The logical result is placed in a specified register, WS variable or IV variable. In actual practice, the AND statement is often used to mask out undesired bits of a register.

Examples:
```
AND R2,R2
AND R3(1),R5
AND X,Y
```

NOTE

The first example illustrates the use of an AND statement in what might be a masking operation. If the AUX register contains 00001111 then this statement sets the 4 high order *bits of R2 to 0 no matter what they were originally. The 4* low order bits of R2 would be unaffected.

The second example illustrates a one place rotate to the right of R3's data before the AND is carried out. The contents of R3 are not affected. In the third example, X and Y may be parts of the same WS or IV byte, or one may be a WS byte and the other an IV byte.

XOR Statement

Use

To compute the logical exclusive OR of the contents of a specified register, WS variable or IV variable and the contents of the AUX register, and place the result in a specified register, WS variable or IV variable. In practice, the XOR statement is often used to complement a value and to perform comparisons.

Examples:
```
XOR R6,R11
XOR R1(7),R4
XOR X,Y
```

NOTE

The first example illustrates the use of an XOR statement in what might be a complementing operation. If the AUX register contains all 1's then the execution of this statement results in the complement of the contents of R6 replacing the contents of R11. The second and third examples are of the same form as the second and third examples of the AND statement.

XMIT Statement

Use

To transmit or load literal values into registers, WS variables or IV variables.

Examples:
```
XMIT DATA,IVR
XMIT OUTPUT,IVL
XMIT -11,AUX
XMIT -00001011B,AUX
XMIT -13H,AUX
```

NOTE

The first example selects a previously declared WS variable by transmitting its address to the IVR register. The second example selects a previously declared IV variable by transmitting its address to the IVL register. The last three examples all result in the generation of the same machine code. They all load the AUX register with -11_{10}. In the first case, the programmer has written the number in base 10. In the second case, the programmer has written the number in binary and has indicated this by placing a B after the number. In the third case, the number has been written in octal as indicated by an H after the number.

XEC Statement

Use

To select and execute one instruction out of a list of instructions in program memory as determined by the value of a data variable, and then continue the sequential execution of the program beginning with the statement immediately following the XEC unless the selected instruction is a JMP or NZT statement.

Examples:
```
                  XEC JTABLE(R1),3
JTABLE      JMP   GR8ERTHAN
                  JMP LESSTHAN
                  JMP EQUALTO

                  XEC SEND(INPUT),4
                  "NEXT INSTRUCTION"
                  "NEXT INSTRUCTION"
                   .           .
                   .           .
SEND              XMIT 11011011B,AUX
                  XMIT 11111111B,AUX
                  XMIT 10101010B,AUX
                  XMIT 00000000B,AUX
```

NOTE

In the first example, the execution of the program will transferred to one of three labeled instructions on the basis of whether register R1 contains 0, 1 or 2. In the second example, the XEC statement causes the execution of a statement which transmits a special bit pattern to the AUX register in response to an input signal which is either 0, 1, 2 or 3. After the pattern is transmitted, the execution of the program continues with the next instruction after the XEC.

NZT Statement

Use

To carry out a conditional branch on the basis of whether or not a register, WS variable, or IV variable is zero or non-zero.

Examples:
```
NZT R1,*+2
NZT SIGN,NEG
```

NOTE

In the first example, if the contents of R1 are non-zero, then program execution will continue with the instruction, whose address is the sum of the address of the NZT statement and 2. If the contents of R1 are 0, the program execution continues with the next instruction after the NZT statement. In the second example, if the contents of a WS or IV variable called SIGN is non-zero, then program execution will continue beginning with the instruction whose address is NEG. Otherwise execution continues with the next instruction after the NZT statement.

JMP Statement

Use

To transfer execution of the program to the statement whose address is the operand of the JMP statement.

Examples:
```
JMP START
JMP *-2
```

NOTE

In the first example, execution of the program continues sequentially beginning with the instruction labeled START. In the second example, program execution continues beginning with the instruction whose address is the JMP instruction's address minus 2.

SEL Statement

Use

Select a variable in Working Storage or on the Interface Vector, so that subsequent machine instructions may reference that variable.

Examples:
```
SEL DATA
SEL OUTPUT
```

NOTE

It is the programmer's responsibility to assure that the proper page has been addressed before calling the SEL *statement if the variable may be in Working Storage. The* SEL statement causes a single instruction, XMIT, to be assembled into the user program. The operand of the XMIT instruction is the byte address of the named variable (argument of the reference) as it has been allocated in Working Storage or on the Interface Vector.

PROGRAMMING EXAMPLES

This section contains programming examples which demonstrate how the 8X300's instructions can be assembled to perform some simple, commonly required functions. These examples are written as program fragments. They are not complete programs as the Data Declaration and Directive statements have been omitted. Otherwise, they follow standard MCCAP conventions.

Looping

Looping is terminated by incrementing a counter and testing for zero. Register R1 is used as counter register and is loaded with a negative number so that the program counts up to zero. Figure 4 illustrates the process.

LOOPING

```
          XMIT      NEG,R1
                    Load negative loop count.
ALPHA     •••
           •
           •        Loop start.
           •
          XMIT      1,AUX
                    Store increment value in AUX register
                    which is an implicit operand of ADD
                    instruction.
          ADD       R1,R1
                    Increment counter register. Add con-
                    tents of AUX to contents of R1 and
                    store the sum in R1.
          NZT       R1,ALPHA
                    Test contents of R1 for zero. If zero,
                    execute next sequential instruction,
                    otherwise, jump to ALPHA and conti-
                    nue execution from there.
           •
           •
           •

TIME: 750 nanoseconds
```

Figure 4

Inclusive-OR (8 Bits)

Generate inclusive-OR of the contents of R1 and R2. Store the logical result in R3. Although the 8X300 does not have an OR instruction, it can be quickly implemented by making use of the fact that (A + B) + (A B) is logically equivalent to A ⊕ B.

INCLUSIVE-OR

```
MOVE     R2,AUX     Load one of the operands into AUX
                    register so that it can be used as the
                    implicit operand of XOR and AND
                    instructions.
XOR      R1,R3      Take exclusive OR of AUX and R1.
                    Store result in R3.
AND      R1,AUX     Take AND of AUX and R1. Place re-
                    sults in AUX.
XOR      R3,R3      Take exclusive OR of AUX (A · B) and
                    R3 (A + B). Store result in R3. R3 now
                    contains inclusive OR of R1 and R2.
TIME: 1.0 microseconds
```

Figure 5

Two's Complement (8-Bits)

Generate the two's complement of the contents of R2. Store the result in R3. Assume that R2 does not contain 200_8.

TWO'S COMPLEMENT

```
XMIT     -1,AUX     Load AUX in preparation for XOR.
XOR      R2,R3      1's complement of R2 is now in R3.
XMIT     1,AUX      Load AUX in preparation for ADD.
ADD      R3,R3      2's complement of R2 is now in R3.
TIME: 1.0 microseconds
```

Figure 6

8-Bit Subtract

Subtract the contents of R2 from the contents of R1 by taking the two's complement of R2 and adding R1. Store the difference in R3.

8-BIT SUBTRACT

```
XMIT     -1,AUX     Perform 2's complement, R2.
XOR      R2,R3
XMIT     1,AUX
ADD      R3,AUX     2's complement of R2 is now in AUX.
ADD      R1,R3      R1-R2 is now in R3.
TIME: 1.25 microseconds
```

Figure 7

16-Bit ADD, Register to Register

Add a 16-bit value stored in R1 and R2 to a 16-bit value in R3 and R4. Store the result in R1 and R2.

16-BIT ADD, REGISTER TO REGISTER

```
MOVE     R2,AUX     Move low order byte of first operand
                    to AUX in preparation for ADD.
ADD      R4,R2      Add the low order bytes of the two
                    operands and store the result in R2.
                    R2 contains the low order byte of the
                    result.
MOVE     R1,AUX     Move high order byte of first operand
                    to AUX.
ADD      OVF,AUX    Add in possible carry from addition of
                    low order bytes.
ADD      R3,R1      Add the high order bytes plus carry
                    and place result in R1. R1 contains the
                    high order byte of the result.
TIME: 1.25 microseconds
```

Figure 8

16-Bit ADD, Memory to Memory

Add a 16-bit value in Working Storage, OPERAND1, to a 16-bit value in Working Storage, OPERAND2, and store result in Working Storage OPERAND1. H1 and L1 represent the high and low order of bytes OPERAND1. H2 and L2 represent the high and low order bytes of OPERAND2.

16-BIT ADD, MEMORY TO MEMORY

```
XMIT     L2,IVR     Transmit address of low order byte of
                    second operand to IVR.
MOVE     L2,AUX     Move low order byte to AUX.
XMIT     L1,IVR     Transmit address of low order byte of
                    first operand to IVR.
ADD      L1,L1      Add low order bytes and store result in
                    L1.
MOVE     OVF,AUX    Move possible carry from addition of
                    low order bytes to AUX register.
XMIT     H2,IVR     Add high order byte of second oper-
                    and to possible carry. Store result in
                    AUX.
ADD      H2,AUX
XMIT     H1,IVR
ADD      H1,H1      High order byte of sum is in H1. Low
                    order byte of sum is in L1.
TIME: 2.25 microseconds
```

Figure 9

Byte Assembly From Bit Serial Input

This is typical of problems associated with interfacing to serial communications lines. An 8-bit byte is assembled from bit inputs that arrive sequentially at the Interface Vector. A single bit on the Interface Vector named STROBE is used to define bit timing, and a second bit, named INPBIT, is used as the bit data interface. Figure 10 illustrates the byte assembly.

BYTE ASSEMBLY

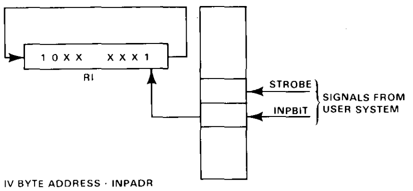


Figure 10

BYTE ASSEMBLY PROGRAM

```
XMIT          0,R1         R1 will be used as a character
                           buffer. It has been cleared.
XMIT          8,R2         R2 will be used as a bit counter.
XMIT          INPADR,IVL   Select IV Byte that contains
                           INPBIT and STROBE.
NEXT BIT NZT  STROBE,*+2   Test STROBE for data ready.
                           The MOVE instruction is exe-
                           cuted only when STROBE = 1.
JMP           *-1          Loop until STROBE = 1.
MOVE          INPBIT,AUX
XOR           R1(1),R1     Rotate R1 one place right. This
                           puts a zero in the least signifi-
                           cant bit position. Then take the
                           exclusive OR of this rotated
                           version of R1 and of AUX. Place
                           the result in R1. The least sig-
                           nificant bit of R1 will now equal
                           the latest value of INPBIT.
XMIT          -1,AUX
ADD           R2,R2        Decrement R2.
                           If R2 is not yet zero, then more
                           bits must be collected to com-
                           plete the byte being assembled.
MOVE          R1(1),R1     This instruction will only be
                           executed when 8 bits have been
                           collected. After this is done, it is
                           still necessary to rotate one
                           more time to get the last INP-
                           BIT into the high order bit posi-
                           tion of R1.
TIME: 1.8 microseconds per bit (minimum)
```

Figure 11

Rotate Left

The 8X300 has no instructions which explicitly rotate data to the left. Such an instruction would be redundant because of the circular nature of the rotate operation. For example, a rotate of two places to the left is identical to a rotate of six places to the right. The rotate n places to the left in an 8-bit register, rotate 8-n places to the right. This example illustrates a rotate of the contents of R4 three places to the left.

MOVE R4(5),R4
TIME: 250 nanoseconds

Three Way Compare

The contents of R1 are compared to the contents of R2. A branch is taken to one of three points in the program depending upon whether R1 = R2, R1 < R2, or R1 > R2.

THREE WAY COMPARE

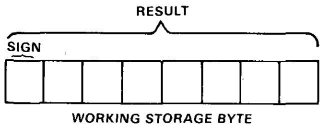


Figure 12

THREE WAY COMPARE PROGRAM

```
XMIT     RESULT,IVR        Choose a working Storage byte
                           by transmitting its address to
                           IVR register.
XMIT     -1,AUX            Load AUX with all 1's, in prep-
                           aration for complementing con-
                           tents of R2.
XOR      R2,RESULT         Store complement of R2 in RE-
                           SULT.
XMIT     1,AUX
ADD      RESULT,AUX        AUX now contains 2's comple-
                           ment of R2.
ADD      R1,RESULT         RESULT now contains R1-R2.
NZT      RESULT,NEQUAL     If RESULT ≠ 0, then R1 ≠ R2.
JMP      EQUAL
NEQUAL   NZT SIGN,LESS     Sign Bit = 1 only when R1 < R2.
GREATER  Continue
                           ●
                           ●
                           ●
EQUAL    Continue
                           ●
                           ●
                           ●
LESS     Continue
TIME: 2.0 microseconds
```

Figure 13

Interrupt Polling

Three external interrupt signals are connected to three IV bits. The three bits are scanned by the program to determine the presence of an interrupt request. A branch is taken to one of eight program locations depending upon whether any or all of the interrupt request signals are present. The IV bits associated with the interrupt requests are wired to the low order three bits of the IV byte named Control. Figures 14 and 15 illustrate the interrupt polling.

INTERRUPT POLLING

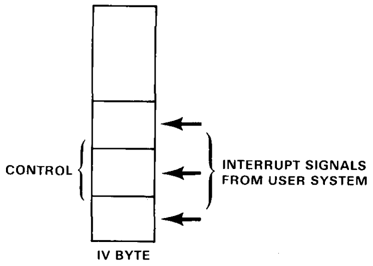


Figure 14

INTERRUPT POLLING PROGRAM

```
XMIT     CONTROL,IVL              Choose proper IV Byte
                                  by transmitting its ad-
                                  dress to IVL register.
XEC      JTABLE  (CONTROL),8      Execute the one in-
                                  struction whose ad-
                                  dress is the sum of
                                  JTABLE and the con-
                                  tents of CONTROL. The
                                  8 indicates the length of
                                  the table.
                                  ●
                                  ●
                                  ●
JTABLE   JMP   ALPHA1             Table of 8 instructions,
                                  one of which is execut-
                                  ed as a result of the XEC
                                  instruction above.
         JMP   ALPHA2
         ●
         ●
         ●
         JMP   ALPHA8
TIME: 750 nanoseconds.
```

Figure 15

Bit Pattern Detection In An I/O Field

Test input field called Input for specific bit pattern, for example: 1 0 1 1. If pattern is not found, branch to NFOUND, otherwise continue sequential execution. Figures 16 and 17 illustrate the procedure.

BIT PATTERN DETECTION

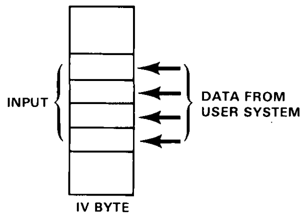


Figure 16

BIT PATTERN DETECTION PROGRAM

```
XMITI    INPUT,IVL        Choose proper IV Byte by trans-
                          mitting its address to IVL regis-
                          ter.
XMIT     1011B,AUX        Store desired bit pattern in AUX
                          register for use as implicit oper-
                          and of XOR instruction.
XOR      INPUT,AUX        Take exclusive OR of the con-
                          tents of INPUT and AUX. Store
                          the result in AUX. Now the con-
                          tents of AUX will be zero if the
                          contents of INPUT are 1011.
NZT      AUX,NFOUND       Test AUX for zero. Branch to
                          NFOUND if non-zero.
●   ●
●   ●
●   ●
NFOUND Continue
TIME: 1.0 microseconds
```

Figure 17

Control Sequence #1

Set an output bit when an input bit goes high (is set) (see Figure 18).

CONTROL SEQUENCE #1

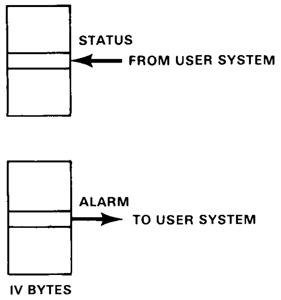


Figure 18

CONTROL SEQUENCE #1 PROGRAM

```
XMIT     STATUS,IVL       Choose input IV byte by trans-
                          mitting its address to IVL.
NZT      STATUS,*+2       Test input bit to determine
                          whether it is still zero. Skip next
                          instruction if it is not zero.
JMP      *-1              Jump to previous instruction.
XMIT     ALARM,IVL        Choose output IV byte.
XMIT     1,ALARM          Set output bit by loading ALARM
                          with 1.
TIME: 1.0 microseconds (minimum)
```

Figure 19

Control Sequence #2

Output a specific 5-bit pattern in response to a specified 3-bit input field.

Subprogram Calls and Returns

The mechanism for managing subprogram calls and returns is based on assigning a return link value to each subprogram caller; this return link value is then used, on exit from the subprogram, to index into the return jump table which returns control to the callers of the subprogram. Figure 21 is an example of a subprogram called from four different locations in the main program.

As seen from Figure 21, each subprogram (or procedure) caller is assigned a "tag" or index values ranging from 0 to 3, or a total of four index values for the four callers. Before jumping to the subprogram, the index value is placed in a previously agreed upon location, register R11 in this case. Upon exit from the subroutine, the index value stored in R11 is used as an offset to the Program Counter in order to execute the proper JMP instruction. The key to returning to the proper caller is the index jump table. Figure 22 gives a detailed description of the return operation.

CONTROL SEQUENCE #2 PROGRAM

```
        XMIT   STATUS,IVL      Choose the IV byte which receives the 3-bit input from user's system.
        MOVE   STATUS,R1       Move the 3 bits of interest from the IV byte to register R1. The 3 bits are automatical-
                               ly right justified.
        XMIT   ALARM,IVL       Choose the IV byte through which the response is sent to the user's system.
        XEC    PATTERN(R1),8   Select specific pattern from PATTERN table.
        JMP    *+9
PATTERN XMIT   A,ALARM    )
        XMIT   B,ALARM    |
        XMIT   C,ALARM    |
        •      •          |    Transmit proper pattern to output IV byte subfield by executing just one
        •      •          }    of these eight instructions. A through H represent the names associated
        •      •          |    with eight different control bit patterns.
        XMIT   H,ALARM    |
        •      •          |
        •      •          |
        •      •          )

TIME: 1.25 microseconds.
```

Figure 20

RETURN OPERATION

```
Address n     XEC*+1  This instruction results in the execu-
                      tion of the instruction located at the
                      current value of the Program counter
                      p plus 1 plus the contents of R11,
                      which is the caller index value.
Address n+1   JMP A
Address n+2   JMP B   The JMP table follows in consecutive
                      Program Storage locations following
                      SEC.
Address n+3   JMP C
Address n+4   JMP D
```

Figure 21

SUBPROGRAM CALL

```
Program
Storage
Address                           Instruction
000137_8   XMIT 0, R11       Load AUX with 0Caller # 1
000140_8   JMP SUBR          Jump to start of subprogram.
000141_8   Next Instruction
               •
               •
               •
               •
001133_8   XMIT 1, R11       Load AUX with 1Caller # 2
001134_8   JMP SUBR          Jump to start of subprogram.
001135_8   Next Instruction
               •
               • •
               •
003260_8   XMIT 2, R11       Load AUX with 2Caller # 3
003261_8   JMP SUBR          Jump to start of subprogram
003262_8   Next Instruction
               •
               •
               •
               •
003654_8   XMIT 3, R11       Load AUX with 3Caller #4
003655_8   JMP SUBR          Jump to start of subprogram
003656_8   Next Instruction
               •
               •
               •
               •

                     a. Main Program
SUBR       Machine Instructions
           JMP  TABL
               •
               •
               •
               •

Subroutine
Return
Code

TABL       XEC   *+1(R11)    Execute JMP located at
                             current PC + 1 + (R11).
           JMP   000141_8    Return to Caller #1
           JMP   001135_8    Return to Caller #2
           JMP   003262_8    Return to Caller #3
           JMP   003656_8    Return to Caller #4

                     b. Subroutine
```

Figure 22

A FAST MICROPROCESSOR FOR CONTROL APPLICATIONS 8X300

ABSTRACT

Many possible applications for microprocessors demand a very quick response to requests for action or information. While MOS microprocessors are relatively cheap, they do not generally possess the necessary speed. Although bipolar microprocessors tend to possess greater speed, they are mostly designed as general purpose devices, which means that they are not ideally suited to the requirements of a fast real-time microcomputer system. The Signetics 8X300 microprocessor has been specifically designed to fulfill this role. This article describes the architecture and instruction set of the 8X300 and, by the use of examples, explains the capabilities and applications of the device.

Considerably improved data throughout is obtained from the use of separate data and address buses for the program memory, coupled with extremely flexible I/O control. Data may be input, modified and output all in the same instruction by the use of the two independent, parallel I/O ports.

INTRODUCTION

As semiconductor technology improved, allowing greater die area with economically acceptable yield, the amount of logic that could be put on a marketable integrated circuit increased. It naturally followed that rather than provide more individual elementary gates on a single die, these gates would be interconnected to afford the user of these chips more complicated logic functions in a single package. The attractiveness of the more complex integrated circuits compelled semiconductor manufacturers to strive for increasing circuit density.

The prospect of putting an entire, although elementary, computer CPU on a single die focused attention on those fabrication processes which allowed the greatest densities. Therefore, the MOS process was the first to yield an entire microprocessor on a chip. Unfortunately, a price was paid in that the MOS processes did not produce as high a speed of logic element as the usual bipolar processes. Because of density limitations, the bipolar process could only produce the less dense parts of chip microprocessors—the bit slices.

Now, however, the improvements in bipolar technology permit the construction of single chip microprocessors with all of the performance advantages of bipolar Schottky technology. Such a circuit has been fabricated and is being produced with significantly high yield to allow commercial availability of quantity parts. The product is the 8X300 microprocessor produced by Signetics. It is the purpose of this paper to present the 8X300 by discussing the architecture and some of the key fabrication and technology features of the microprocessor. This paper concludes with a brief review of some of the present as well as potential applications of this device.

MICROPHOTOGRAPH OF THE 8X300 CHIP

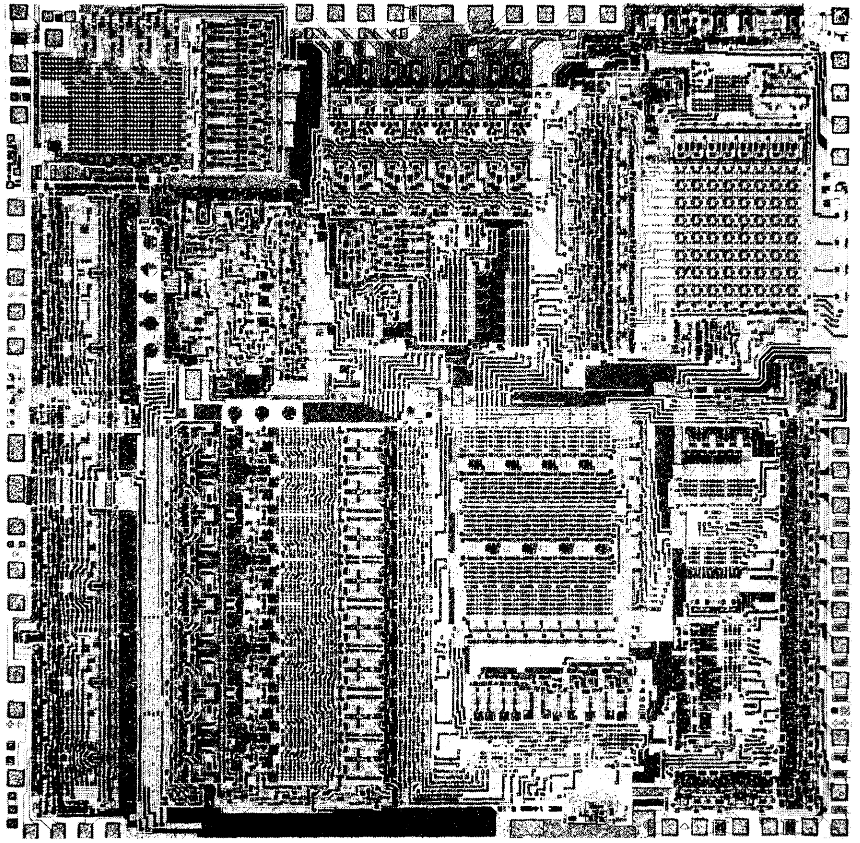


Figure 1

The 8X300 was optimized for control applications rather than for extensive numerical processing, so before the main presentation begins, it is advisable to describe the basic requirements in the envisaged application field of the 8X300.

Control here applies to a wide variety of areas and is not necessarily limited to those specific areas itemized below. The action of control may be the sole purpose of a standalone microprocessor. In such a task, the microprocessor examines statuses at a particular rate and issues command words or bits to the external circuitry to effect the function of the whole machine as it is described in the control program. Thus, the microprocessor selects specific bits defined by the program, tests the bits, and responds or directs by setting or clearing other bits. Although elementary on the surface, this task may be quite complex involving timing, interval measurement, and various forms of decision making, all at potentially high speed. Control may also take the form of bit or word manipulation and data movement such as in data concentrators, communication controllers, disk and tape controllers and similar devices. Here the data destined for storage, transfer or transmission may require alteration (for example bit packing, preamble addition or error detection/correction); consequently the control also involves calculation or data generation. Consider an industrial metal cutter required to form a complex shape as directed by some external data input. Matrix multiplications may be a very necessary part of this controller's process in order to carry out its function.

Thus, we see that controllers in this context may perform a wide variety of bit and arithmetic processing depending upon the type of controller one is discussing. The 8X300 is capable of good performance in all of these control areas.

ARCHITECTURE

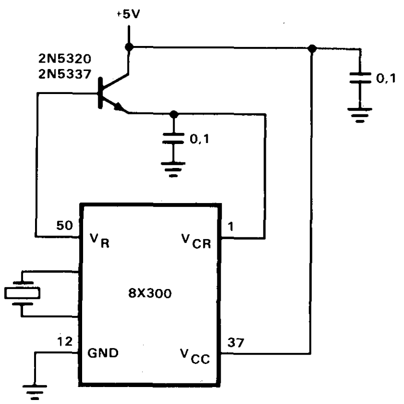
The architecture of a microprocessor is intimately connected to the technology used to produce the device, for one could define architectures which are realizable only with certain fabrication approaches. Also, a microprocessor's architecture is described by its instruction set and its input/output structure. So, in this section, the 8X300 will be examined both from the inside—technology, block diagram, etc., and from the outside—instruction set, I/O bus, timing, etc.

The 8X300 is fabricated using standard Schottky technology. Dual layer metallization is used to minimize die area, reduce capacitance and hence maximize the speed of the processor. A microphotograph of the 8X300 die is shown in Figure 1. The die measures 250 mils square and is the largest bipolar microprocessor in existence. The 8X300 is a complete processor on a single chip and, as will be seen later, results in a minimum circuit count processor system. Linear elements are also provided on the die as shown in Figure 2.

One functional entity is the clock generator circuit, which oscillates at a frequency determined by an external crystal or timing capacitor. This circuit generates all timing signals required internally by the 8X300 and externally for bus timing. Secondly, a voltage regulator in combination with an externally connected (user-provided) pass transistor, provides a stable low voltage source for the operation of selected internal segments. This voltage is approximately 3 volts and is used in areas where power conservation rather than speed is a prime concern. *(The 3 volts does not imply I²L utilization.)* Maximum current used is 450mA (300mA typical) with 150mA used in the 5 volt (V_{CC}) connection and 300mA used in the 3 volt (V_{CR}).

With the regulator, the entire processor operates from a single +5 volt supply over the commercial temperature range (0° C to

CLOCK AND POWER SUPPLY REQUIREMENTS FOR THE 8X300



● ICC(TYP) = 300mA

Figure 2

SCHEMATIC ARCHITECTURE OF THE 8X300

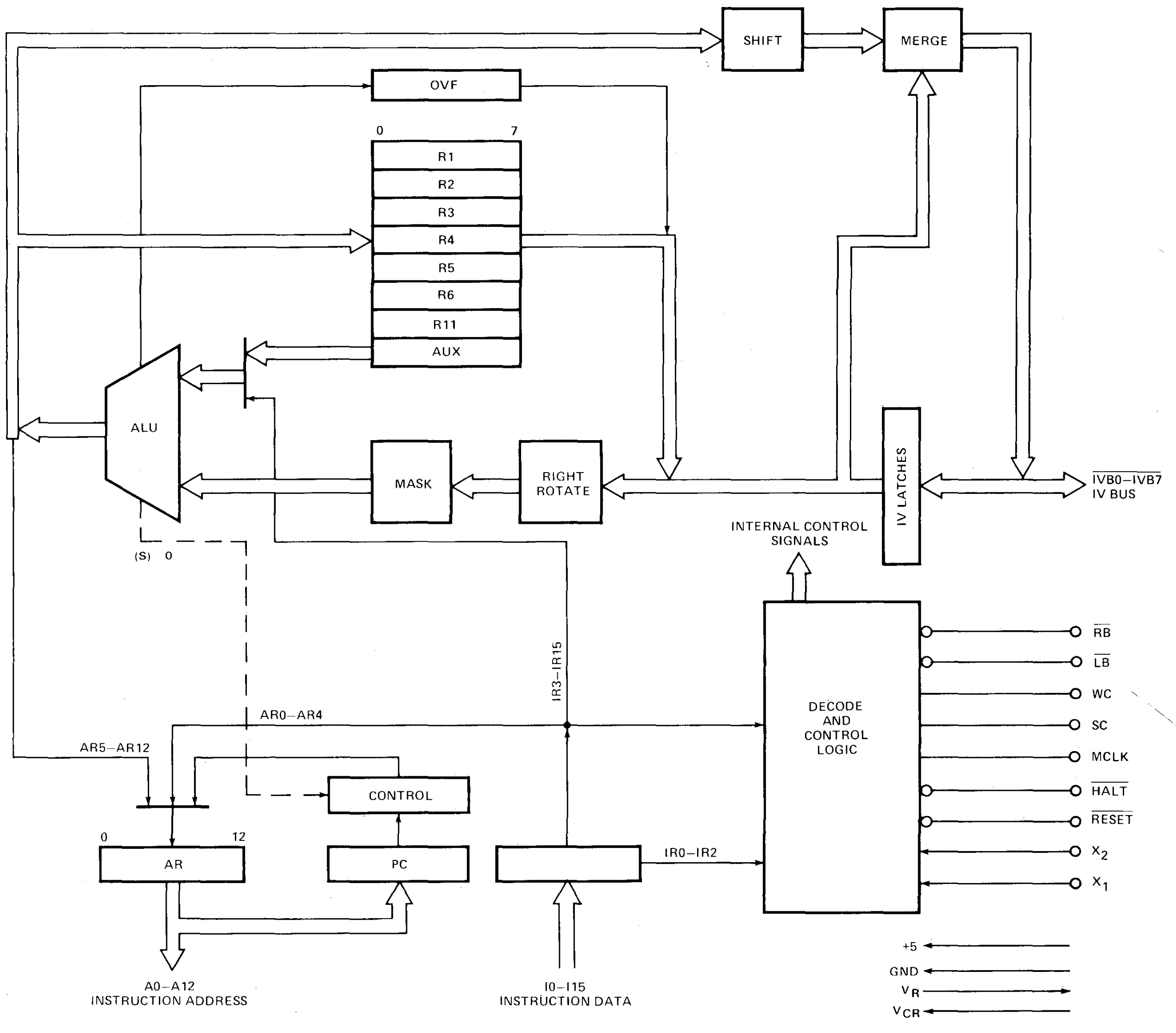


Figure 3

+70°C). The 8X300 is packaged in a 50-pin dual-in-line ceramic package.

The block diagram of the 8X300 processor is shown in Figure 3. It does not show the circuitry just described. First, note that full instruction decoding logic is provided to interpret the instruction classes and perform the indicated operation. This will be discussed in more detail later. This decoding and control logic provides all internal signals required as well as certain control lines for data input and output. These lines are RB, LB, WC, SC and MCLK. External control may be applied to hold the 8X300 in a non-processing or wait state (via halt) or force the processor to instruction address zero (reset). The processor also contains its own program counter (PC) which is automatically incremented upon instruction execution or, in certain cases, is not incremented or is loaded with a new value. Current address control, provided by the address register (AR) may be derived all or in part from the program counter, the instruction data (AR0-AR4) or from the output of the ALU (AR5-AR12). Thus, the present and future instruction to be executed may be altered through instructions or the condition of selected data.

Input/Output

Separate buses are provided for instruction address and instruction data. The current contents of the address register (AR) are presented on a 13-bit bus (A0-A12) to the program memory to fetch the 16-bit instruction word. The 8X300 possesses the capability of directly addressing 8K of program storage. The instruction word enters the processor via the instruction bus (I0-I15) and is stored in the instruction register (R).

The processing part of the 8X300 is shown in the upper half of Figure 3. The entire processor is oriented about 8-bit data manipulation; therefore interfaces to external circuitry use an 8-bit bus, designated the Interface Vector (IV) bus (IV0-IV7). For internal storage of data, eight 8-bit read/write registers are provided, designated R1-R6, R11 and AUX (auxiliary). The auxiliary register contains one of the operands that are used in two operand instructions such as ADD, AND and XOR (Exclusive-OR). A 1-bit overflow register (OVF) is provided to store the overflow resulting from add operations. The IV latch is not addressable, but stores original data brought in from the IV bus to be used in the merge operation prior to output. At the heart of the processing is the ALU which performs various arithmetic and logic operations on data. The ALU, when combined with the rotate, mask, shift and merge elements, permits unique data operations.

Before proceeding, it is essential that the IV bus concept be explained. From this, we shall go back and discuss the architecture and instruction set in greater detail. The IV bus serves both as an address and data bus and is accompanied by the bus control signals shown in Figure 4. Since the bus carries addresses as well as data, I/O ports must be enabled before data transfers may take place. This is usually accomplished by presenting an address on the bus under program control. The control line SC is used to indicate address content of the bus. When presented with an address, an I/O port either enables itself (becomes active on the bus to accept or present data) if the address presented is its own, or disables itself (becomes inactive) if the address presented does not match its own address.

Together with this, processor I/O ports have been designed which allow 1 of 512 interface vector bytes to be selected without decoders. Having two ports, one for the user and the other to the microprocessor, these IV bytes are completely bidirectional. The unique feature of these bytes is the way in which they are addressed. Each IV byte has an 8-bit field programmable address, which is used to enable the microprocessor port, allowing data transfer through it.

To effect input and output data transfer, the 8X300 IV outputs are three-state drivers. Additionally, to control external devices, the 8X300 issues the write command, WC, which indicates whether data transfers are read (into the 8X300) or write (out of the 8X300). The bus direction is entirely under control of the 8X300.

A unique feature of the 8X300 is the partitioning of the bus into two banks, designated left bank (LB) and right bank (RB). Using the LB and RB signals from the processor as master enables for the I/O ports, the processor may dynamically select ports as Figure 5 illustrates. Two I/O ports may be active during one cycle provided that they are on opposite banks. To do this,

TYPICAL SYSTEM CONFIGURATION SHOWING USER INTERFACE

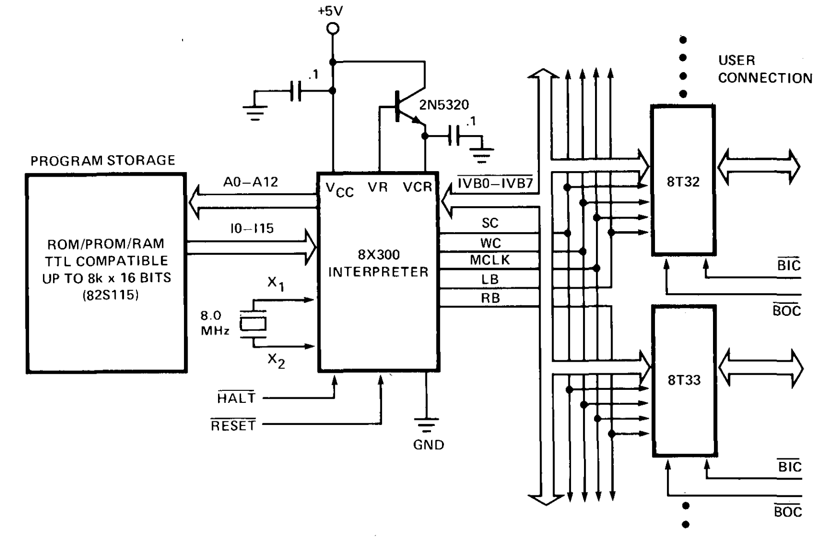


Figure 4

INSTRUCTION EXECUTION SEQUENCE AND LEFT BANK/RIGHT BANK ADDRESSING

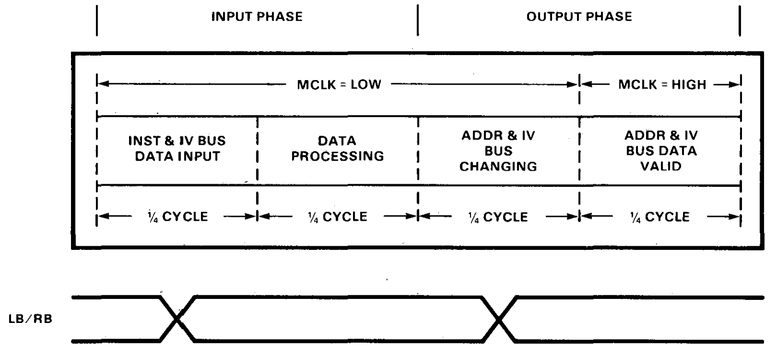


Figure 5

I/O ports recognize addresses, data or controls only when enabled by the bank signal to which they are connected. Clearly, the bank partitioning may be considered as a ninth address bit which is alterable by the processor within an instruction. (The 8X300, therefore, has 512 direct I/O port address capability.) A general data operation between two I/O ports could follow the following steps. First, an address is presented to one bank enabling a selected I/O port and disabling all others on that bank. Secondly, another address is presented to the opposite bank effecting a similar selection there. Subsequently, in one instruction cycle, the 8X300 may accept data from one port (on one bank), operate on the data and deposit the result in the other port in the second bank. If the working storage of the registers is not sufficient, additional storage can be added using an I/O port address to add another 256X8 words of RAM . See Figure 6.

WORKING STORAGE (RAM) MAY BE INCLUDED IN THE SYSTEM DESIGN

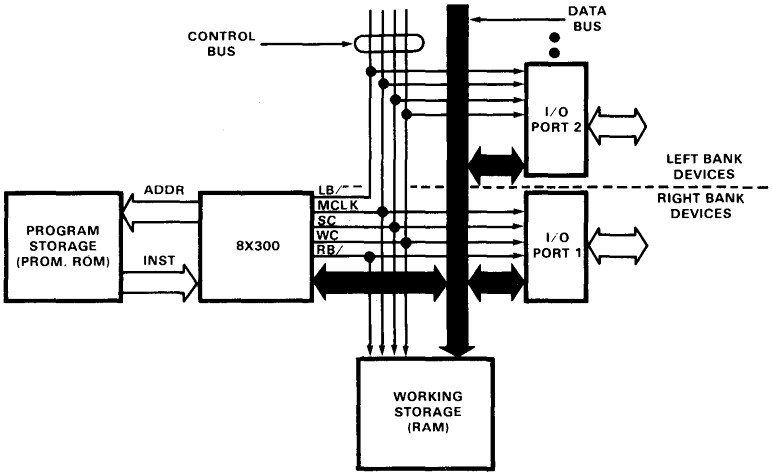


Figure 6

In order to fully appreciate the speed of the last operation, accepting data from one port and depositing it on the other, it is necessary to explore the details of the instruction cycle. Each 8X300 operation is executed in one instruction cycle which is subdivided into four quarter cycles. The quarter cycles are shown in Figure 5. The instruction address for an operation is presented at the output during the third quarter of the previous instruction cycle. With a memory of sufficient speed, the instruction is returned and accepted by the processor during the first quarter of the cycle in which that instruction is to be executed. The instruction is decoded and used to direct the operation of the processor throughout the cycle.

For data processing, the instruction cycle may be viewed as having two halves. During the first half of the cycle, data to be processed is brought into the processor and stored in the IV latch. This is accomplished during the first quarter cycle. The next quarter cycle of this first half is used to bring the data through the ALU, thereby processing the data as required by the instruction. The second half cycle is the output phase during which the data is presented to the IV bus and finally clocked into the appropriate I/O port after bus stabilization. The processor issues MCLK for this purpose.

Bank selection during input and output phases is independent, thus data may be input from the right bank and deposited in the left bank or vice-versa, or to and from the same bank if the same IV is used. Bank selection during instruction cycle phases is specified by the instruction. Therefore, the processor may input data from one port, operate on the data and return it to a second port in one instruction cycle time. Remember that instruction fetching is concurrent with data operations. The cycle time is 250ns, making the 8X300 comparable in speed on a microcycle basis, to bipolar slice systems.

Instruction Set

The power of the 8X300 architecture is embodied in the instruction set which controls the ALU, rotate, mask, shift and merge functions to provide for various data operations. Each 16-bit instruction word is subdivided into several fields. The arithmetic and logical instructions follow the format shown in Figure 7. There are eight instruction classes each with variations depending upon the operand specifications. These instructions provide for:

Arithmetic and logic operations—
  Add, And and XOR
Data movement—
  Move and XMIT (transmit)
Context alteration—
  JMP (unconditional jump), NZT (test and branch on non-zero) and XEC (execute the instruction at the address specified without program counter alteration)

FORMAT OF ARITHMETIC AND LOGIC INSTRUCTIONS

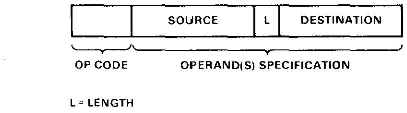


Figure 7

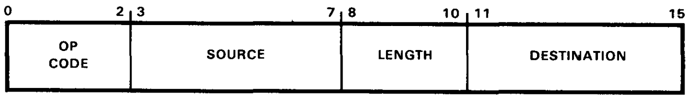
The operand fields specify the source of the data as one of the internal registers or from the IV bus as left bank or right bank, and the destination of the data as one of the internal registers, left bank or right bank or as left bank or right bank addresses. Additionally, these fields specify the length and position of the data which is to be processed. As an example, see Figure 8.

Before going through an example, some features of this instruction should be explained. The first 3 bits are used for the op-code. The 5 source bits contain two separate information groups: The first 2 bits (3 and 4) define the actual source while the next 3 bits (5, 6 and 7) define the least significant bit of the variable length field of the source. These are represented in the diagram by two digits—the first modulo 4 and the second modulo 8. In the example the first digit being 2 selects left bank I/O (right bank = 3).

The length of the source data field is specified by the length (bits 8, 9 and 10). The 5 destination bits are represented by two digits—the first modulo 4 and the second modulo 8, as the source. In Figure 8, the destination is an internal register, specified by the first digit being 0 or 1. The actual register is specified by the value of the second digit. These operand fields control the rotate, mask and shift operations as data proceeds through the microprocessor.

Rather than go through the details of the complete instruction set, it is more instructive to proceed with an example which will serve to illustrate what may be done with a single instruction. What shall be done in this

TYPICAL INSTRUCTION



- Op Code = 0 (MOVE), 1 (ADD), 2 (AND), 3 (XOR)
- Source = 20 -27_8 (Left bank), 30 - 37_8 (Right bank)
- Length = 0 - 7 (L)
- Dest. = 60 - 67_8, 11_8, 17_8
- Action: Least Significant L bits of IV source data are right justified and processed by the ALU. The 8-bit result is deposited in the destination register.

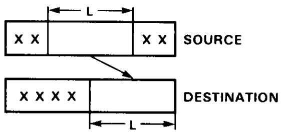


Figure 8

THE ADD OPERATION

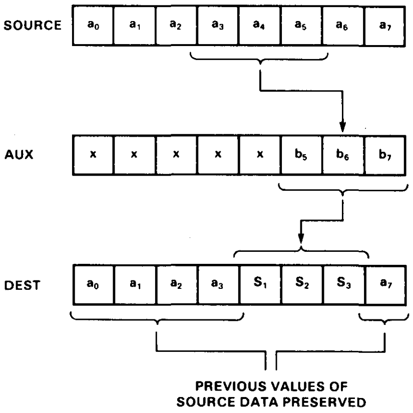


Figure 10

example is to select two I/O ports, add the contents of the AUX register to a specified segment of the source, merge the result with the original data and deposit the result at the destination.

Suppose the source of the data is in IV port, address 5 on the left bank, and the destination address is contained in internal register R3. Further suppose that the AUX register already contains the required value to be added. First, the I/O ports are selected: XMIT 5, IVL (transmit the number 5 to the bus as left bank address). MOVE R3,IVR (move contents of R3 to the bus as a right bank address). The I/O ports have now been enabled using two instructions—500ns total thus far. Now perform ADD LB, RB (add left bank to AUX and deposit in right bank port). The add instruction is shown in Figure 9 where the add operand fields specify the selection of bits throughout rotate and length (mask), and after addition specify the position of merge (shift) in the original data. Although the source, length and destination fields shown here are unique to the MOVE, ADD, AND and XOR instructions, the comments made about these fields also apply to the fields of all the other instructions. Port 5 on the left bank is assumed to contain a_0-a_7 (Figure 10) and the AUX register is assumed to contain b_0-b_7. The source field specifies selecting bits starting with a_5 and the length field specifies taking 3 bits to the left. Thus, a_5, a_4 and a_3 are masked off and right justified. Note that this requires that only contiguous bits be selected for operation. Next, the selected bits are added to the same length of bits (beginning at the right) from the AUX register.

ADD INSTRUCTION TO PERFORM THE ADDITION SHOWN IN FIGURE 10

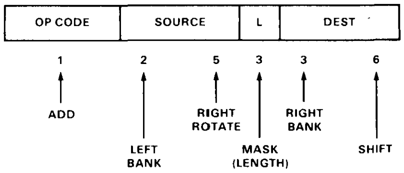


Figure 9

Thus, the sum (a_3, a_4, a_5) + (b_5, b_6, b_7) is computed producing a 3-bit sum, S_1, S_2, S_3 (and a possible overflow). The destination field of the add instruction then specifies a shift of 1 bit to the left. The shift is made and the 3 bits of the sum are merged with the original source data. Note that the same length specification (3 in the example) applies in source selection, operation and merge functions and is not alterable within one instruction cycle. The destination contains the set of bits shown in Figure 10 after the add operation. Note that the entire set of rotate, mask, add, shift and merge functions took place in one instruction cycle time.

The content of each field can be represented by a set of digits. These digits have a direct relation to the specific operations which the data undergoes as it is directed along the 8X300 internal data paths. The op code for add is 1. This is followed by a two digit source field. The source field is in fact two fields (in this particular case) in which the first digit, 2, specifies left bank, while the second digit specifies the rotate operation which is to be performed on the incoming data. The L or length field specifies the number of bits to be accepted for ALU operation. This is the mask function specification and selects a quantity of bits counting from the right. The masking operation takes place after the rotate. The destination field, like the source field, specifies the bank or internal register (right bank in this case), and for the bank destination, also specifies the shift operation.

There is one important point to note about the instruction format. Since the fields are easily represented by octal digits and since these digits have a direct relation to the function specified by the field, programming the 8X300 is very easy. Simple mnemonic representation of each of the field specifications, such as ADD for the add function, LB and RB for left bank and right bank and so forth, are easily translated into the octal representation. With this convenience, several hundred lines of program code can be easily generated by hand from the mnemonic representation. Consequently, for small tasks (i.e., less than 500 instructions), an assembler is not essential for efficient programming. A simple conversion is required to generate the actual program memory content.

The above example is typical of what can be done with the MOVE, ADD, AND and XOR instructions. However, the control functions perform differently and are worthy of further attention. Specifically, the XEC (execute) instruction is powerful in that it may be register or I/O vectored. The XEC instruction temporarily changes the contents of the address register for the one instruction cycle following the XEC while allowing subsequent control to be resumed through the program counter. In this light, XEC may be viewed as calling a single instruction subroutine. The XEC instruction performs the vectoring by concatenating the higher order program counter contents with a number determined, in part, by the contents of one of the internal registers or by the content of an I/O port. Thus, the XEC instruction may be used to sequence through a list where the list counter is an internal register, or it may be used to branch to a specific service routine based on some external status reflected in a selected I/O port.

APPLICATIONS

The 8X300 may be exploited in a variety of applications where high speed is required and where the architecture fits the particular requirement. The 8X300 may serve in disk controllers, communications data concentrators and demultiplexers, tape controllers, industrial process controllers, video controllers including entertainment and games, as well as CRT/keyboard terminals, plus a variety of other applications. Principally, the 8X300 affords its greatest service to the user in high speed, relatively sophisticated systems. For example, a low speed MOS processor might be used to control a CRT display and do so economically. However, add the requirement to do data processing for, say, graphics or color display, then the 8X300 becomes increasingly attractive. With 8-bit parallel to serial conversion, the 8X300 may easily process data and directly produce video for alphanumeric display. Generally, one may conclude that the 8X300 serves well where the control processor is required to be in the data path. In controlling computer peripherals, one alternative is to use a single 8X300 processor to control a number of peripherals, as opposed to having one lower speed, less costly processor with separate memory and auxiliary circuits in each peripheral.

Economically, the 8X300 is certainly competitive with the bit slice approach. For those who need the performance, the 8X300 affords a complete, single chip processor at a power consumption of only 1.5 watts in contrast to three to four chips for a bit slice equivalent using nearly 5 watts.

The typical system configuration for the 8X300 is shown in Figure 4. The 8X300 interfaces to the external world through a convenient number of I/O ports connected to the IV bus. Program storage is provided by a suitable ROM or PROM, but RAM could be used here also depending upon the user's application. However, in the more common control applications, the function of the processor is dedicated and, consequently, there is no need to have alterable program storage. This reasoning is also evident in the 8X300 architecture, as exemplified in Figure 4. It is clear that there is no direct connection between the program store and the I/O system, as opposed to other microprocessors (the MOS microprocessor in particular) in which instructions are fetched over the same bus on which data and I/O transfers take place.

Figure 4 also emphasizes the compact nature of the processor system. Note that the CPU and program store are realized in as few as three packages (e.g. 8X300 with two 82S115 chips). I/O ports are added as required for the particular system configuration.

Connections to the IV bus are not restricted to the 8T32 type of addressable bidirectional I/O port. Depending upon requirements, a number of devices may be employed. Working storage in the form of RAM may be interfaced directly to the IV bus with an 8T31 or other suitable device used as an address latch. This affords the user temporary storage for data and status information. ROM may also be provided in order to access fixed constants for use by the processor. Examples of such ROM include sine function look-up tables, coordinate translation constants, sensor linearization curves, etc.

Some users have objected to the overhead cost in addressing I/O ports prior to an operation. As in the example used in this paper, 500ns (two instructions) were taken up in selecting I/O ports prior to the major data operation. This is acceptable if ports continue to be accessed for a number of times and thereby reduce the addressing overhead. However, for those who see this as a limitation, there is a convenient alternative. The instruction memory may be extended such that an extra field appears as an additional bus which is applied to each I/O port. Port selection (addressing) would then be done upon instruction fetch. No latch addressable I/O ports would be used, but the normal active-on-address-decode scheme would be employed. The address field may be as wide as required to serve all system I/O ports and if necessary memory. Bus left bank and right bank partitioning would still be used, so the address field would contain two addresses, one for each bank. With this scheme, an entire operation such as described earlier, including the selection of I/O ports, could be accomplished in 250ns.

CHAPTER 5
DEVELOPMENT SYSTEMS AND PROGRAMS

INTRODUCTION

Microprocessors are considerably different from the random logic which they replace, consequently the development of microprocessor based systems is equally different. The interconnection of hardware devices is generally simplified with microprocessors and a major portion of the system function is carried out through the appropriate software or firmware. While the hardware configurations are generally variations of the basic controller or CPU, structure the firmware is tailored specifically for the system design. Thus firmware development is a significant and vital portion of a microprocessor based system designing and equipment which speeds this development and further enhances the attractiveness of microprocessors.

Once software is developed, the system development advances to the firmware hardware integration phase. This iterative process of test and software modification (or hardware modification) results in a functional system design. PROMS are then fused with the debugging programs and the system is finalized. This process is made convenient by equipment which electrically simulates the microprocessor system along with the firmware program.

This chapter describes the development systems available to the microprocessor user which expedites the two system development phases outlined above.

COMPATIBLE PRODUCTS

The following manufacturers produce equipment which can be made use of in developing systems which use Signetics components.

SIGNETICS PRODUCT	INDUSTRY STANDARD DEVELOPMENT AID
8X300	• Scientific Microsystems (SMS) 520 Clyde Ave. Mt. View 94043 Microcontroller Simulator (MCSIM)
8X300, 3000 Series, 2901 FPLA/PROMs (Programming)	• SMS ROM Simulator • Data I/O Corp. POB 308, 1297 N.W. Mall Issaquah, Wash. 98027 • Curtis Electro Devices P.O. B 4090 Mountain View, CA. 94040

CHAPTER 6
MILITARY

The Signetics Mil 38510/883 Program is organized to provide a broad selection of processing options, structured around the most commonly requested customer flows. The program is designed to provide our customers:

- Standard processing flows to help minimize the need for custom specs.
- Cost savings realized by using standard processing flows in lieu of custom flows.
- Better delivery lead times by minimizing spec negotiation time, plus allows customers to buy product off-the-shelf or in various stages of production rather than waiting for devices started specifically to custom specs.

The following explains the different processing options available to you. Special device marking clearly distinguishes the type of screening performed.

JAN QUALIFIED

JAN Qualified product is designed to give you the optimum in quality and reliability. The JAN processing level is offered as the result of the government's product standardization programs, and is monitored by the Defense Electronic Supply Center (DESC), through the use of industry-wide procedures and specifications.

JAN Qualified products are manufactured, processed and tested in a government certified facility to Mil-M 38510, and appropriate device slash sheet specifications. Design documentation, lot sampling plans, electrical test data and qualification data for each specific part type has been approved by the Defense Electronic Supply Center (DESC) and products appear on the DESC Qualified Products List (QPL-38510).

Group B testing, per Mil-Std-883 Method 5005, is performed on each six weeks of production on each slash sheet for each package type. Group C, per Mil-Std-883 Method 5005, is performed every ninety days for each microcircuit group. Group D testing, per Mil-Std-883 Method 5005, is performed every six months for each package type.

In addition to the common specs used throughout the industry for processing and testing, JAN Qualified products also possess a requirement for a standard marking to be used throughout the IC industry.

JAN PROCESSING

This option is extremely useful when the reliability and screening of a JAN device is required, however, Signetics is not listed on the QPL for the product needed. Processing is performed to Mil-Std-883 Method 5004, and product is 100% electrically tested to the appropriate JAN slash sheet.

Group B, C and D data for JAN processed and the other military processing levels which follow, consists of Group B and D testing performed per Mil-Std-883 Method 5005, every six months minimum by package type and Group C per Mil-Std-883 Method 5005, is run every ninety days on each microcircuit group.

JAN REL

Processing to this option is ideal when no JAN slash sheets are released on devices required. Product is processed to Mil-Std-883 Method 5004, and is 100% electrically tested to industry data sheets. (Specific parameters required by a customer may also be included.)

/883B

This is a lower priced version of the JAN Rel option described above. Processing is identical with the only exceptions being the dc electrical testing over the temperature range and ac electrical testing at room temperature are performed as a part of Group A instead of 100%.

MIL TEMP

If you need a Military temp. range device, but do not require all the high reliability screening performed in the other processing options, our Mil-Temp. product is ideal. Mil-Temp. parts are the standard full Mil-Temperature range product guaranteed to a 1% AQL to the Signetics data sheet parameters.

MILITARY GENERIC DATA

Signetics has a new program for those customers who require qualification data on their products. This program allows our customers to obtain reliability information without the necessity of running Groups B, C and D inspections for their particular purchase order. It provides for the customer something that has not been readily available before in the semiconductor industry in that all Military Generic Data is controlled and audited by both Government Inspection and Quality Assurance.

Signetics Military Generic Data is generated by the Military Products Division. The data is compiled from 1) JAN qualification lots, and 2) Data generated by qualification lots run for other reliability programs.

A Military Generic family is defined as consisting of die function and package type families.

A Generic die function lot qualifies a ninety day manufacturing period and a representative package type qualifies a one hundred and eight day manufacturing period.

Military Generic Data

- Allows our customers to qualify Signetics products based on existing qualification data performed at Signetics.
- Allows our customers to reduce costs and improve deliveries.
- Provides assurance that all Signetics die function families and packages meet JAN and customer reliability requirements.
- Provides and attributes summary to the customer backed by lot identity and traceability.

Generic Quality Conformance Data is supplied upon customer request in a format conforming to either Notice 2 (March 1, 1976) or Notice 1 (February 5, 1975) of Mil-Std-883A. Testing to either notice is considered equivalent, since only the test frequency has been altered. Table 2 provides the definition and qualifying manufacturing periods as outlined in Notice 1 and Notice 2 of Mil-Std-883A.

SCREEN	MIL-STD-883 METHOD	CLASS A	REQUIREMENT	CLASS B	REQUIREMENT	CLASS C	REQUIREMENT
Internal Visual (Preseal)[1]	2010	Cond A	100%	Cond B	100%	Cond B	100%
Stabilization Bake	1008 (24 hr min.)	Cond C min	100%	Cond C min	100%	Cond C min	100%
Temperature Cycling[2]	1010	Cond C	100%	Cond C	100%	Cond C	100%
Constant Acceleration	2001	Cond E min Y1 plane	100%	Cond E min Y1 plane	100%	Cond E min Y1 plane	100%
Visual Inspection[11]			100%		100%		100%
Seal[4]							
Fine Leak Gross Leak	1014	Cond A or B Cond C2	100%	Cond A or B Cond C2		Cond A or B Cond C2	
Serialization			Note 7		--		--
Critical Electrical Parameters (pre Burn-in)	Subgroup A-1 (note 4)	Read and Record	100% Note 8	Optional	Note 5	Not Required	--
Burn-in Test	1015 T_A = + 125°C	240 hours min.[10]	100%	160 hours min.	100%	Not Required	--
Critical Electrical Parameters (post Burn-in)	Subgroup A-1 (note 4)	Read and Record	100% Note 8	Not Required	--	Not Required	--
Signetics FAILURE CRITERIA		PDA 5%		PDA 10%		Not Required	--
Reverse Bias Burn-in[6]	1015.1, T_A = + 150°C t = 72 hours	Cond A or C[9]	100%	Not Required	--	Not Required	--
Critical Electrical Parameters (post burn-in)	Subgroup A-1	Read and Record	100% Note 8	Required	100%	Not Required	--
Final Electrical Test Parameters	Perform 100% go-no-go measurements of sub-group A parameters[12]	Subgroups A1, A2, A3, A4, A9, Functional tests, truth table when applicable (A7)	100%	Subgroups A1, A2, A3, A4, A9, Functional tests, truth table when applicable (A7)	100%	Subgroup A1, functional tests, truth table when applicable (A7)	100%
Radiographic Inspection[3]	2012	Yes	100%	Not Required	—	Not Required	—
Quality Conformance Inspection[8]	5005	Class A	Note 10	Class B	Note 10	Class C	Note 10
External Visual	2009	Yes	100%	Yes	100%	Yes	100%

NOTES

1. Unless otherwise specified, at the manufacturer's option, test samples for Group B, bond strength (Method 5005) may be selected randomly immediately following internal visual (Method 5004) prior to sealing.
2. For Class B and C devices, this test may be replaced with thermal shock Method 1011, Test Condition A, minimum.
3. The radiographic screen may be performed in any sequence after seal.
4. When fluorocarbon gross leak testing is utilized, Test Condition C_2 shall apply as a minimum.
5. When specified in the applicable device specification, 100% of the devices shall be tested.
6. The reverse bias burn-in is a requirement only when specified in the applicable device specification and is recommended only for certain MOS, linear or other microcircuits where surface sensitivity may be of concern. When reverse bias burn-in is not required, interim electrical parameter measurements following burn-in test are omitted. The order of performing the burn-in and the reverse bias burn-in may be inverted.
7. Class A devices shall be serialized prior to interim electrical parameter measurements.
8. Electrical parameters shall be read and recorded.
9. For Class A devices, Test Condition F of Method 1015 and 3.4.2 herein shall not apply.
10. Samples shall be selected for testing in accordance with the specific device class and lot requirements of Method 5005.
11. At the manufacturer's option, visual inspection for catastrophic failures may be conducted after each of the thermal/mechanical screens, after the sequence or after seal test. Catastrophic failures are defined as missing leads, broken packages or lids off.
12. Detailed test, conditions and limits applicable to each subgroup are given in Signetics data manual electrical characteristics table. See Table 3 for corresponding Group A tests of Mil-Std-883.

Table 1 MIL-M-38510/MIL-STD-883 PROCESSING LEVELS

MIL-STD-883A GROUP A SUBGROUP	TEST DESCRIPTION
A1	Static tests at 25°C
A2	Static tests at maximum rated operating temperature
A3	Static tests at minimum rated operating temperature
A4	Dynamic tests at 25°C*
A5	Dynamic tests at maximum rated operating temperature*
A6	Dynamic tests at minimum rated operating temperature*
A7	Functional tests at 25°C
A8	Functional tests at maximum and minimum rated operating temperatures
A9	Switching tests at 25°C
A10	Switching tests at maximum rated operating temperature
A11	Switching tests at minimum rated operating temperature

*Applicable only to Signetics Analog Products

Table 3 MIL-STD-883 GROUP A ELECTRICAL TESTS

BIPOLAR MICROPROCESSORS

PRODUCT	DESCRIPTION	AVAILABILITY	
		Dip	Flat Pack
3001	Microprogram Control Unit	I	R
3002	Central Processing Element (2-bit slice)	I	R
8X300	Interpreter/Microcontroller	I	*
2901-1	Central Processing Element (4-bit slice)	*	*

*Under development

QUALIFIED SUB-GROUPS	QUALIFIES	NOTICE 1	NOTICE 2
A	Electrical Test	n/a[1]	n/a[1]
B	Package—Same package construction lead finish and devices produced on same production line through final seal.	Data selected from devices manufactured within 6 weeks of manufacturing period.	Data selected from devices manufactured within 24 weeks of manufacturing period.
C	Die/Process—Devices representing the same process families may be used.	Data selected from the representing devices from the same microcircuit group and sealed within 12 weeks of manufacturing period.	Allows the data to be selected from the devices produced within 48 weeks of manufacturing period.
D	Package—Qualifies the same package construction and lead finished devices produced on the same production line through final seal.	n/a[2]	Data selected from the devices representing the same package construction and lead finish manufactured within the 24 weeks of manufacturing period. OR If no such data available, manufacturing period extends to 48 weeks.

NOTES

1. Group A is performed on each lot of Signetics devices.
2. Group D not offered in Mil-Std-883 Notice 1.

Table 2 DEFINITION AND QUALIFYING MANUFACTURING PERIODS FOR NOTICE 1 AND 2 MIL-STD-883

MILITARY MICROPROCESSOR SUPPORT CIRCUITS

PRODUCT	DESCRIPTION	AVAILABILITY	
		Dip	Flatpack
LOGIC			
54123	Retriggerable Monostable Multivibrator	F	W
54180	8-Bit Odd/Even Parity Checker	F	W
54298	Quad 2-Input Mux with Storage	F	W
54S182	Look-Ahead Carry Generator	*	*
54S194	4-Bit Bidirectional Shift Register	*	*
54S195	4-Bit Parallel Access Shift Register	*	*
54LS365	High Speed Hex Tri-State Buffer	F	*
54LS366	High Speed Hex Tri-State Buffer	F	*
54LS367	High Speed Hex Tri-State Buffer	F	*
54LS368	High Speed Hex Tri-State Buffer	F	*
8262	9-Bit Parity Generator Checker	F	W
8281	Presettable Binary Counter	F	W
8291	Presettable High Speed Binary Counter	F	W
9602	Dual Monostable Multivibrator	F	W
INTERFACE			
8T09	Quad Bus Driver with Tri-State Output	F	W
8T10	Quad D-Type Bus Latch (Tri-State Outputs)	F	W
8T13	Dual Line Driver	F	W
8T14	Triple Line Receiver/Schmitt Trigger	F	W
8T15	Dual Communication EIA/Mil Line Driver	*	*
8T16	Dual Communication EIA/Mil Line Receiver	*	*
8T26	Quad Bus Driver/Receiver (Tri-State)	F	*
8T28	Quad Bus Non-Inverting Driver/Receiver (Tri-State)	F	*
8T32	IV Bytes (Programmable)	I	*
8T33	IV Bytes (Programmable)	I	*
8T34	IV Bytes (Programmable)	I	*
8T35	IV Bytes (Programmable)	I	*
8T95	High Speed Hex Buffer/Inverter (Tri-State)	F	*
8T96	High Speed Hex Buffer/Inverter (Tri-State)	F	*
8T97	High Speed Hex Buffer/Inverter (Tri-State)	F	*
8T98	High Speed Hex Buffer/Inverter (Tri-State)	F	*

*Under development

MILITARY MEMORIES

BIPOLAR MEMORIES CROSS REFERENCE

DEVICE	ORGANIZATION	PACKAGE*		FAIRCHILD	HARRIS	MMI	INTERSIL	AMD	TI
PROMs									
82S23	32X8	F	R	-	7602-2	5330	5600	27S08	54S188
82S115	512X8	I	R	-	7644-2	-	-	-	
82S123	32X8	F	R	-	7603-2	5331	5610	27S09	54S288
82S126	256X4	F	R	93416	7610-2	5300	5603	27S10	54S387
82S129	256X4	F	R	93426	7611-2	5301	5623	27S11	54S287
82S130	512X4	F	R	93436	7620-2	5305	5604	-	-
82S131	512X4	F	R	93446	7621-2	5306	5624	-	-
82S136	1024X4	F,I	R	93443	7642-2	5352	5606	-	-
82S137	1024X4	F,I	R	93453	7643-2	5353	5626	-	-
82S140	512X8	I	R	93438	7640-2	5340	5605	-	-
82S141	512X8	I	R	93448	7641-2	5341	5625	-	-
82S180	1024X8	I	R	-	-	5380	-	-	-
82S181	1024X8	I	R	-	-	5381	-	-	-
82S184	2048X4	I	R	-	-	-	-	-	-
82S185	2048X4	I	R	-	-	-	-	-	-
FPLAs									
82S100	16X48X8	I	R	93459	-	82S100	-	27S100	-
82S101	16X48X8	I	R	93458	-	82S101	-	27S101	-
PLAs									
82S200	16X48X8	I	R	-	-	-	-	-	-
82S201	16X48X8	I	R	-	-	-	-	-	-
RAMs									
54S89	16X4	F	R	-	-	-	-	-	5489
54S189	16X4	F	R	-	-	-	-	-	54189
54S200	256X1	F	R	-	-	-	-	-	54S200
54S201	256X1	F	R	-	-	-	-	-	54S201
54S301	256X1	F	R	-	-	-	-	-	54S301
82S09	64X9	I	R	93419	-	-	-	-	-
82S10	1024X1	F,I	R	93415	-	-	55S08	2952	-
82S11	1024X1	F,I	R	93425	-	-	55S18	2953	-
82S16	256X1	F	R	93421	-	5531	5523	2700	-
82S17	256X1	F	R	93411	-	5530	5533	2701	-
82S25	16X4	F	R	93403	0064	5560	5501	3101	-
ROMs									
82S215	512X8								
82S223	32X8								
82S224	32X8								
82S226	256X4								
82S229	256X4								
82S230	512X4								
82S231	512X4								
82S280	1024X8								
82S281	1024X8								

*NOTE

R	BeO Flatpack
F	Cerdip
I	Ceramic DIP

MILITARY LOGIC

5400 SERIES

		54		54LS		54S		54H	
DEVICE	DESCRIPTION	F	W	F	W	F	W	F	W
	GATES								
5400	Quad 2-input NAND Gate	o	o	o	o	o	o	o	o
5401	Quad 2-Input NAND Gate with o/c	o	o	o	o	-	-	o	o
5402	Quad 2-Input NOR Gate	o	o	o	o	o	o	-	-
5403	Quad 2-Input NAND Gate with o/c	o	o	o	o	o	o	-	-
5408	Quad 2-Input AND Gate	o	o	o	o	o	o	o	o
5409	Quad 2-Input AND Gate with o/c	o	o	o	o	o	o	-	-
5410	Triple 3-Input NAND Gate	o	o	o	o	o	o	o	o
5411	Triple 3-Input NAND Gate	o	o	o	o	o	o	o	o
5412	Triple 3-Input NAND Gate with o/c	o	o	o	o	-	-	-	-
5415	Triple 3-Input AND Gate with o/c	-	-	o	o	o	o	-	-
5420	Dual 4-Input NAND Gate	o	o	o	o	o	o	o	o
5421	Dual 4-Input AND Gate	o	o	o	o	-	-	o	o
5422	Dual 4-Input NAND Gate with o/c	-	-	o	o	o	o	o	o
5426	Quad 2-Input NAND Gate with o/c	o	-	o	o	-	-	-	-
5427	Triple 3-Input NOR Gate	o	o	o	o	-	-	-	-
5430	8-Input NAND Gate	o	o	o	o	-	-	o	o
5432	Quad 2-Input OR Gate	o	o	o	o	o	o	-	-
5450	Expandable Dual 2-Wide 2-Input A0I Gate	o	o	-	-	-	-	o	o
5451	Dual 2-Wide 2-Input A0I Gate	o	o	o	o	o	o	o	o
5452	Expandable 4-Wide 2-2-2-3 Input AND-OR Gate	-	-	-	-	-	-	o	o
5453	4-Wide 2-Input A0I Gate (Expandable)	o	o	-	-	-	-	o	o
5454	4-Wide 2-Input A0I Gate	o	o	o	o	-	-	o	o
5455	2-Wide 4-Input A0I Gate	-	-	o	o	-	-	o	o
5460	Dual 4-Input Expander	o	o	-	-	-	-	o	o
5464	4-2-3-2 Input A0I Gate	-	-	-	-	o	o	-	-
5465	4-2-3-2 Input A0I Gate	-	-	-	-	o	o	-	-
5486	Quad 2-Input Exclusive-OR Gate	o	o	o	o	o	o	-	-
54133	13-Input NAND Gate	-	-	-	-	o	o	-	-
54134	12-Input NAND Gate with 3-State Outputs	-	-	-	-	o	o	-	-
54136	Quad Exclusive-OR Gate with o/c	-	-	o	o	-	-	-	-
54260	Dual 5-Input NOR Gate	-	-	o	o	o	o	-	-
54266	Quad Exclusive NOR Gate	-	-	o	o	-	-	-	-

		54		54LS		54S		54H	
DEVICE	DESCRIPTION	F	W	F	W	F	W	F	W
	HEX INVERTERS/BUFFERS								
5404	Hex Inverter	o	o	o	o	o	o	o	o
5405	Hex Inverter with o/c	o	o	o	o	o	o	o	o
5406	Hex Inverter with Buffer/ Driver with o/c	o	o	-	-	-	-	-	-
5407	Hex Buffer/Driver with o/c	o	o	-	-	-	-	-	-
5416	Hex Inverter Buffer/Driver with o/c	o	o	-	-	-	-	-	-
5417	Hex Buffer/Driver with o/c	o	o	-	-	-	-	-	-
5428	Quad 2-Input NOR Buffer	o	o	o	o	-	-	-	-
5433	Quad 2-Input NOR Buffer	o	o	o	o	-	-	-	-
5437	Quad 2-Input NAND Buffer	o	o	o	o	o	-	-	-
5438	Quad 2-Input NAND Buffer with o/c	o	o	o	o	-	-	-	-
5440	Dual 4-Input NAND Buffer	o	o	o	o	o	o	o	o
54125	Quad Bus Buffer Gate with 3-State Outputs	o	o			-	-	-	-
54126	Quad Bus Buffer Gate with 3-State Outputs	o	o			-	-	-	-
54128	Quad 2-Input NOR Buffer	o	o	-	-	-	-	-	-
54140	Dual 4-Input NAND Line/Driver	-	-	-	-	o	o	-	-
	FLIP-FLOPS								
5470	J-K Flip-Flop	o	o	-	-	-	-	-	-
5472	J-K Master-Slave Flip-Flop	o	o	-	-	-	-	o	o
5473	Dual J-K Master-Slave Flip-Flop	o	o	o	o	-	-	o	o
5474	Dual D-Type Edge-Triggered Flip-Flop	o	o	o	o	o	o	o	o
5476	Dual J-K Master-Slave Flip-Flop	o	o	o	o	-	-	o	o
5478	Dual J-K Negative Edge-Triggered Flip-Flop	-	-	o	o	-	-	-	-
54101	J-K Negative Edge-Triggered Flip-Flop	-	-	-	-	-	-	o	o
54103	Dual J-K Negative Edge-Triggered Flip-Flop	-	-	-	-	-	-	o	o
54106	Dual J-K Negative Edge-Triggered Flip-Flop	-	-	-	-	-	-	o	o
54107	Dual J-K Master-Slave Flip-Flop	o	Q	o	o	-	-	-	-
54108	Dual J-K Negative Edge-Triggered Flip-Flop	-	-	-	-	-	-	o	-
54109	Dual J-K Positive Edge-Triggered Flip-Flop	o	o	o	o	-	-	-	-
54112	Dual J-K Negative Edge-Triggered Flip-Flop	-	-	o	o	o	o	-	-
54113	Dual J-K Negative Edge-Triggered Flip-Flop	-	-	o	o	o	o	-	-
54114	Dual J-K Negative Edge-Triggered Flip-Flop	-	-	o	o	o	o	-	-

KEY

o =Available in packages indicated at head of column unless otherwise stated
- = No plans yet
Blank = Qualification in process

MILITARY LOGIC

5400 SERIES Cont'd

		54		54LS		54S		54H	
DEVICE	DESCRIPTION	F	W	F	W	F	W	F	W
	LATCHES								
5475	Quad Bistable Latch	o	o	o	o	-	-	-	-
5477	Quad Bistable Latch	-	o	-	-	-	-	-	-
54100	4-Bit Bistable Latch (Dual)	I	Q	-	-	-	-	-	-
54116	Dual 4-Bit Latch with Clear	I	-	-	-	-	-	-	-
54279	Quad S-R Latch	o	o			-	-	-	-
	SCHMITT TRIGGERS								
5413	Dual Hex Schmitt Trigger	o	o	o	o	-	-	-	-
5414	Hex Schmitt Trigger	o	o	o	o	-	-	-	-
54132	Quad Schmitt Trigger	o	o	o	o	-	-	-	-
	DECODERS								
5442	BCD-to-Decimal Decoder	o	o			-	-	-	-
5443	Excess 3-to-Decimal Decoder	o	o	-	-	-	-	-	-
5444	Excess 3-Gray-to-Decimal Decoder	o	o	-	-	-	-	-	-
5445	BCD-to-Decimal Decoder/ Driver with o/c	o	o	-	-	-	-	-	-
5446A	BCD-to-7 Segment Decoder/ Driver	o	o	-	-	-	-	-	-
5447	BCD-to-7 Segment Decoder/ Driver	o	o	-	-	-	-	-	-
5448	BCD-to-7 Segment Decoder/ Driver	o	o	-	-	-	-	-	-
54138	3-to-8 Line Decoder/Demux	-	-	o	o	-	-	-	-
54139	Dual 2-to-4 Line Decoder/Demux	-	-	o	o	o	o	-	-
54145	BCD-to-Decimal Decoder/ Driver with o/c	o	o			-	-	-	-
54154	4-Line to 16-Line Decoder/ Demux	I	Q	I	Q	-	-	-	-
54155	Dual 2-LIne to 4-Line Decoder/Demux	o	o	-	-	-	-	-	-
54156	Dual 2-Line to 4-Line Decoder/Demux	o	o	-	-	-	-	-	-
54254	4-Line to 16-Line Decoder	-	-	o	o	-	-	-	-
54261	2X4 2's Complement-Multiplier	-	-	o	o	-	-	-	-
	ENCODERS								
54147	10-Line to 4-Line Priority Encoder	o	o	-	-	-	-	-	-
54148	8-Line to 3-Line Priority Encoder	o	o	-	-	-	-	-	-

		54		54LS		54S		54H	
DEVICE	DESCRIPTION	F	W	F	W	F	W	F	W
	MONOSTABLE MULTIVIBRATORS								
54121	Monostable-Multivibrator	o	o			-	-	-	-
54122	Retriggerable Monostable Multivibrator	o	o			-	-	-	-
54123	Retriggerable Monostable Multivibrator	o	o			-	-	-	-
54222	Dual Monostable Multivibrator	o	o	o	o	-	-	-	-
	COUNTERS								
5490	Decade Counter	o	o			-	-	-	-
5492	Divide-by-Twelve Counter	o	o			-	-	-	-
5493	4-Bit Binary Counter	o	o			-	-	-	-
54160	Synchronous 4-Bit Decade Counter	o	o	-	-	-	-	-	-
54161	Synchronous 4-Bit Binary Counter	o	o			-	-	-	-
54162	Synchronous 4-Bit Decade Counter	o	o	-	-	-	-	-	-
54163	Synchronous Binary Counter	o	o	o	o	-	-	-	-
54190	Synchronous BCD Up/Down Counter	o	o			-	-	-	-
54191	Synchronous Binary Up/Down Counter	o	o	o	o	-	-	-	-
54192	Synchronous Decade Up/ Down Counter	o	o			-	-	-	-
54193	Presettable Binary Up/Down Counter	o	o			-	-	-	-
54196	Presettable Decade Counter/ Latch (8290)	-	-			-	-	-	-
54197	Presettable Binary Counter/ Latch	-	-			-	-	-	-
54290	Decade Counter	-	-			-	-	-	-
54293	4-Bit Binary Counter	-	-			-	-	-	-
	DATA SELECTORS/ MULTIPLEXERS								
54150	16-Line to 1-Line Mux	I	o	-	-	-	-	-	-
54151	8-Line to 1-Line Mux	o	o	o	o	o	o	-	-
54153	Dual 4-Line to 1-Line Mux	o	o	o	o	o	o	-	-
54157	Quad 2-Input Data Selector (non-inv.)	o	o	o	o	o	o	-	-
54158	Quad 2-Input Data Selector (inv.)	o	o	o	o	o	o	-	-
54251	Data Selector/Mux with 3-State Outputs	-	-	o	o	-	-	-	-
54253	Dual 4-Line to 1-Line Data Selector/Mux	-	-	o	o	o	o	-	-
54257	Quad 2-Line to 1-Line Data Selector/Mux	-	-			-	-	-	-
54258	Quad 2-Line to 1-Line Data Selector/Mux	-	-			-	-	-	-
54298	Quad 2-Input Mux with Storage	o	o	-	-	-	-	-	-

KEY

o =	Available in packages indicated at head of column unless otherwise stated
- =	No plans yet
Blank =	Qualification in process

MILITARY LOGIC

5400 SERIES Cont'd

		54		54LS		54S		54H	
DEVICE	DESCRIPTION	F	W	F	W	F	W	F	W
	SHIFT REGISTERS								
5491	8-Bit Register	o	o	-	-	-	-	-	-
5494	4-Bit Shift Register (PISO)	o	o	-	-	-	-	-	-
5495	4-Bit Left-Right Shift Register	o	o			-	-	-	-
5496	5-Bit Shift Register	o	o			-	-	-	-
54164	8-Bit Parallel-Out Serial Shift Register	o	-	o	o	-	-	-	-
54165	Parallel-Load 8-Bit Shift Register	o	o	-	-	-	-	-	-
54166	8-Bit Shift Register	o	o	-	-	-	-	-	-
54170	4X4 Register File	o	-	o	o	-	-	-	-
54194	4-Bit Bidirectional Universal Shift Register	o	o			-	-	-	-
54195	4-Bit Parallel-Access Shift Register	o	o			-	-	-	-
54198	8-Bit Shift Register	I	-	-	-	-	-	-	-
54199	8-Bit Shift Register	I	-	-	-	-	-	-	-
54670	4X4 Register File (3-State)	-	-	o	o	-	-	-	-
	ARITHMETIC ELEMENTS								
5480	Gated Full Adder	o	o	-	-	-	-	-	-
5483	4-Bit Binary Full Adder	o	o	o	o	-	-	-	-
5485	4-Bit Magnitude Comparator	o	o			o	o	-	-
54180	8-Bit Odd/Even Parity Generator/Checker	o	o	-	-	-	-	-	-
54181	4-Bit Arithmetic Logic Unit	I	-	I	Q	I	-	-	-
54182	Look-Ahead Carry Generator	o	o	-	-	-	-	-	-

8200 SERIES

DEVICE	DESCRIPTION	PACKAGE	
	ARITHMETIC ELEMENTS		
8243	8-Bit Position Scaler	I	Q
8260	Arithmetic Logic Unit	I	Q
8261	Fast Carry Extender	F	W
8262	9-Bit Parity Generator and Checker	F	W
8269	4-Bit Comparator	F	W
	COUNTERS		
8280	Presettable Decade Counter	F	W
8281	Presettable Binary Counter	F	W
8284	Binary Up/Down Counter	F	W
8285	Decade Up/Down Counter	F	W
8288	Divide-by-Twelve Counter	F	W
8290	Presettable High Speed Decade Counter	F	W
8291	Presettable High Speed Binary Counter	F	W
8292	Presettable Low Power Decade Counter	F	W
8293	Presettable Low Power Binary Counter	F	W
	DECODERS		
8250	Binary-to-Octal Decoder	F	W
8251	BCD-to-Decimal Decoder	F	W
8252	BCD-to-Decimal Decoder	F	W
	GATES		
8241	Quad Exclusive-OR Gate	F	W
8242	Quad Exclusive-NOR Gate	F	W
	LATCHES		
8275	Quad Bistable Latch	F	W
	MULTIPLEXERS		
8230	8-Input Digital Mutiplexer	F	W
8231	8-Input Digital Multiplexer	F	W
8232	8-Input Digital Multiplexer	F	W
8233	2-Input 4-Bit Digital Multiplexer	F	W
8234	2-Input 4-Bit Digital Multiplexer	F	W
8235	2-Input 4-Bit Digital Multiplexer	F	W
8263	3-Input 4-Bit Digital Multiplexer	I	Q
8264	3-Input 4-Bit Digital Multiplexer	I	Q
8266	2-Input 4-Bit Digital Multiplexer	F	W
8267	2-Input 4-Bit Digital Multiplexer	F	W
	REGISTERS		
8200	Dual 5-Bit Buffer Register	I	Q
8201	Dual 5-Bit Buffer Register with D Inputs	I	Q
8202	10-Bit Buffer Register	I	Q
8203	10-Bit Buffer Register with D Inputs	I	Q
8270	4-Bit Shift Register	F	W
8271	4-Bit Shift Register	F	W
8273	10-Bit Serial-In, Parallel-Out Shift Register	F	W
8274	10-Bit Parallel-In, Serial-Out Shift Register	F	W

KEY

o =	Available in packages indicated at head of column unless otherwise stated
- =	No plans yet
Blank =	Qualification in process
F =	Cerdip
W =	Cer Pack, Flat
Q =	Ceramic Flat

MILITARY ANALOG

INDUSTRY CROSS REFERENCE

DEVICE	DESCRIPTION	PACKAGE		FSC	MOT	NSC	TI	RAYTHEON
	COMPARATORS							
SE526	Analog Voltage Comparator	F	K	-	-	-	-	-
SE527	Analog Voltage Comparator	F	K	-	-	LM161	-	-
SE529	Analog Voltage Comparator	F	K	-	-	-	-	-
LM139	Quad Comparator	F	-	μA139	-	LM139	-	LM139
μA710	Differential Voltage Comparator	F	T	μA710	MC1710	LM710	SN52710	RM710
μA711	Comparator	F	K	μA711	MC1711	LM711	SN52711	RM711
	DIFFERENTIAL AMPLIFIERS							
SE510	Dual Differential Amplifier	F	-	-	-	-	-	-
SE511	Dual Differential Amplifier	F	-	-	-	-	-	-
SE515	Differential Amplifier	F	K	-	-	-	-	-
μA733	Video Amplifier	F	K	μA733	MC1733	LM733	SN52733	RM733
	OPERATIONAL AMPLIFIERS							
LM101	High Performance Op Amp	F	T	μA101	MLM101	LM101	-	LM101
LM101A	High Performance Op Amp	F	T	μA101A	MLM101A	LM101A	SN52101A	LM101A
LM107	General Purpose Op Amp	F	T	μA107	MLM107	LM107	SN52107	LM107
LM108	Precision Op Amp	F	T	μA108	-	LM108	-	LM108
LM108A	Precision Op Amp	F	T	μA108A	-	LM108A	-	LM108A
LM124	Quad Op Amp	F	-	-	-	LM124	-	LM124
LM158	Dual Op Amp	-	T	-	-	LM158	-	-
MC1558	Dual Op Amp	F	T	μA1558	MC1558	LM1558	-	RM1558
SE532	Dual Op Amp	-	T	-	-	LM158	-	-
μA709	Op Amp	F	T	μA709	MC1709	LM709	SN52709	RM709
μA709A	Op Amp	F	T	μA709A	MC1709P2	-	-	-
μA741	General Purpose Op Amp	F	T	μA741	MC1741	LM741	SN52741	RM741
μA747	Dual Op Amp	F	K	μA747	MC1747	LM747	-	RM747
μA748	General Purpose Op Amp	F	T	μA748	MC1748	LM748	SN52748	-
	PHASE LOCKED LOOPS							
SE567	Tone Decoder PLL	F	T	-	-	LM567	-	-
	LINE RECEIVERS							
DM7820	Dual Differential Line Receiver	F	-	-	-	DM7820	SN55182	-
DM7830	Dual Differential Line Receiver	F	-	-	-	DM7830	SN55183	-
	TIMERS							
SE555	Timer	F	T	MC1555	-	LM555	SN52555	RM555
	VOLTAGE REGULATOR							
μA723	Precision Voltage Regulator	F	L	μA723	MC1723	LM723	SN52723	RM723

PREFIX NOMENCLATURE

S/SE	Signetics Proprietary (-55°C to + 125°C)
MC	Motorola Second Source
LM	National Second Source
μA	Fairchild Second Source

When Signetics claims our bipolar microprocessors make your design resources go further

How can you switch to bipolar from MOS microprocessing without obsoleting your existing designs? How can you reduce hardware design time, reduce programming time, reduce debugging time? How can you try it before you buy it?

Signetics has put it all together in bipolar microprocessors featuring the industry's most complete product family, fastest 2-bit μPA (3001), first 8-bit fixed instruction processor (8X300), the 8080 emulator, several designers' kits, interface elements, bipolar memories and all using Signetics superior low power Schottky LSI technology.

● **8080 Emulator upgrades performance of existing system.** Delivers 5 times more performance, reduces micro-code writing time, operates from existing software, makes designing easy. (Available now.)

2 **A free microprocessing book.** It's filled with memory, logic, interface info and application notes. All Signetics' high performance bipolar products are listed in detail.

● **The industry's leading selection of RAM's, ROM's, PROM's, FPLA's, you name it.** Signetics has more configurations and performance levels available from stock. You get the right fit and absolute maximum efficiency in minimum time.

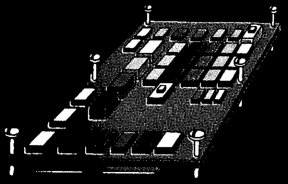
● **8X300 Designer's Kit.** We supply all the parts you need to build your own general purpose controller—you do the PROM programming, checkout prototype system, order production quantities from Signetics. It's just another example of Signetics total support.

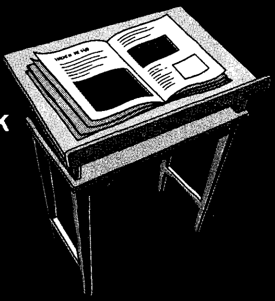
● **Long list of interface products to pick from.** Signetics has the interface products you'll need to make your bipolar microprocessing shopping easier and more convenient. Now you get everything on your list from one source.

6 **Introductory Designer's Kit. A $230.00 value, for $100.00.** You get 12 parts and 1 manual to let you work with bipolar, design with bipolar, program

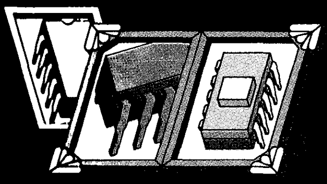
...we support it.

Increase performance with the 8080 Emulator.





2 We have the book on microprocessing.



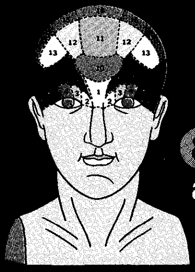
The industry's leading family of bipolar memories.

Dream up controllers with the 8X300 Designer's Kit.

One-stop shopping for interface products.

6 Introductory Designer's Kit $100.

Microprocessor book describes design support systems available.



8 Our microprocessor answermen know all.

with bipolar, and prove out all the advantages for yourself. The parts you need to actually try out Signetics' 3000 Total System.

A description of all design support systems. Read all about 'em. SMS ROM Simulator, SMS McSim Prototyping System, Microassembler, SMS McCAP Assembler and more, all in the microprocessor book.

8 Field Applications Engineers that know bipolar microprocessing. They have the industry's best support systems at hand to help you solve your design problems. They will show you exactly how Signetics' total bipolar microprocessor capabilities will make your design resources go further.

Now you know when Signetics claims to reduce your hardware design time, to reduce your programming time, and to reduce your debugging time...Signetics supports it. Start saving time in the future by mailing the coupon today.

Attach this to your letterhead for fast response.

□ Send me information on the 3000 Introductory Designer's Kit.

□ Send me information on the 8X300 Designer's Kit—to dream up my own controllers.

□ I want to read the book on bipolar microprocessing. Send it ASAP. FREE.

□ Send me info now on an emulator that delivers 5 times more performance, the 8080.

□ Have a Bipolar Microprocessing Answerman call me for an appointment.

My application is ______________________________

____

Name Title

Telephone Mail Stop

THINK signetics

811 E. Arques Ave., Sunnyvale, Ca. 94086

© 1976. a subsidiary of U.S. Philips Corporation

signetics

a subsidiary of U.S. Philips Corporation

Signetics Corporation
811 East Arques Avenue
Sunnyvale, California 94086
Telephone 408/739-7700

SIGNETICS HEADQUARTERS

811 East Arques Avenue
Sunnyvale. California 94086
Phone: (408) 739-7700

ARIZONA

Phoenix
Phone: (602) 971-2517

CALIFORNIA

Inglewood
Phone: (213) 670-1101

Irvine
Phone: (714) 833-8980
(213) 924-1668

San Diego
Phone: (714) 560-0242

Sunnyvale
Phone: (408) 736-7565

COLORADO

Parker
Phone: (303) 841-3274

FLORIDA

Pompano Beach
Phone: (305) 782-8225

ILLINOIS

Rolling Meadows
Phone: (312) 259-8300

INDIANA

Noblesville
Phone: (317) 773-6770

KANSAS

Wichita
Phone: (316) 683-6035

MASSACHUSETTS

Woburn
Phone: (617) 933-8450

MICHIGAN

Southfield
Phone: (313) 645-1232

MINNESOTA

Edina
Phone: (612) 835-7455

NEW JERSEY

Cherry Hill
Phone: (609) 665-5071

Piscataway
Phone: (201) 981-0123

NEW YORK

Wappingers Falls
Phone: (914) 297-4074

Woodbury. L.I.
Phone: (516) 364-9100

OHIO

Worthington
Phone: (614) 888-7143

TEXAS

Dallas
Phone: (214) 661-1296

REPRESENTATIVES

ARIZONA

Phoenix
Chaparral-Dorton
Phone: (602) 263-0414

CALIFORNIA

San Diego
Mesa Engineering
Phone: (714) 278-8021

Sherman Oaks
Astralonics
Phone: (213) 990-5903

CANADA

Calgary. Alberta
Philips Electronics
Industires Ltd.
Phone: (403) 243-7737

Montreal. Quebec
Philips Electronics
Industries Ltd.
Phone: (514) 342-9180

Ottawa. Ontario
Philips Electronics
Industries Ltd.
Phone: (613) 237-3131

Scarborough. Ontario
Philips Electronics
Industries Ltd.
Phone: (416) 292-5161

Vancouver. B.C.
Philips Electronics
Industries Ltd.
Phone: (604) 435-4411

COLORADO

Denver
Barnhill Five. Inc.
Phone: (303) 426-0222

CONNECTICUT

Newtown
Kanan Associates
Phone: (203) 426-8157

FLORIDA

Altamonte Springs
Semtronic Associates
Phone: (305) 831-8233

Largo
Semtronic Associates
Phone: (813) 586-1404

ILLINOIS

Chicago
L-Tec Inc.
Phone: (312) 286-1500

KANSAS

Lenexa
Buckman & Associates
Phone: (913) 492-8470

MARYLAND

Glen Burni
Microcomp. Inc.
Phone: (301) 761-4600

MASSACHUSETTS

Reading
Kanan Associates
Phone: (617) 944-8484

MICHIGAN

Bloomfield Hills
Enco Marketing
Phone: (313) 642-0203

MINNESOTA

Edina
Mel Foster Tech. Assoc.
Phone: (612) 835-2254

MISSOURI

St. Charles
Buckman & Associates
Phone: (314) 724-6690

NEW JERSEY

Haddonfield
Thomas Assoc., Inc.
Phone: (609) 854-3011

NEW MEXICO

Albuquerque
The Staley Company, Inc.
Phone: (505) 821-4310/11

NEW YORK

Ithaca
Bob Dean, Inc.
Phone: (607) 272-2187

NORTH CAROLINA

Cary
Montgomery Marketing
Phone: (919) 467-6319

OHIO

Centerville
Norm Case Associates
Phone: (513) 433-0966

Fairview Park
Norm Case Associates
Phone: (216) 333-4120

OREGON

Portland
Western Technical Sales
Phone: (503) 297-1711

TEXAS

Dallas
Cunningham Company
Phone: (214) 233-4303

Houston
Cunningham Company
Phone: (713) 461-4197

UTAH

West Bountiful
Barnhill Five, Inc.
Phone: (801) 292-8991

WASHINGTON

Bellevue
Western Technical Sales
Phone: (206) 641-3900

WISCONSIN

Greenfield
L-Tec, Inc.
Phone: (414) 545-8900

DISTRIBUTORS

ALABAMA

Huntsville
Hamilton/Avnet Electronics
Phone (205) 533-1170

ARIZONA

Phoenix
Hamilton/Avnet Electronics
Phone: (602) 275-7851

Liberty Electronics
Phone: (602) 257-1272

CALIFORNIA

Costa Mesa
Schweber Electronics
Phone: (714) 556-3880

Culver City
Hamilton Electro Sales
Phone: (213) 558-2173

El Segundo
Liberty Electronics
Phone: (213) 322-8100

Mountain View
Elmar Electronics
Phone: (415) 961-3611

Hamilton/Avnet Electronics
Phone: (415) 961-7000

San Diego
Hamilton/Avnet Electronics
Phone: (714) 279-2421

Liberty Electronics
Phone: (714) 565-9171

Sunnyvale
Intermark Electronics
Phone: (408) 738-1111

CANADA

Downsview. Ontario
Cesco Electronics
Phone: (416) 661-0220

Mississauga. Ontario
Hamilton/Avnet Electronics
Phone: (416) 677-7432

Montreal. Quebec
Cesco Electronics
Phone: (514) 735-5511

Zentronics Ltd.
Phone: (514) 735-5361

Ottawa. Ontario
Hamilton/Avnet Electronics
Phone: (613) 226-1700

Zentronics Ltd.
Phone: (613) 238-6411

Toronto. Ontario
Zentronics Ltd.
Phone: (416) 789-5111

Vancouver. B.C.
Bowtek Electronics Co., Ltd.
Phone: (604) 736-1141

Ville St. Laurent. Quebec
Hamilton/Avnet Electronics
Phone: (514) 331-6443

COLORADO

Commerce City
Elmar Electronics
Phone: (303) 287-9611

Denver
Hamilton/Avnet Electronics
Phone: (303) 534-1212

Lakewood
Acacia Sales, Inc.
Phone: (303) 232-2882

CONNECTICUT

Danbury
Schweber Electronics
Phone: (203) 792-3500

Georgetown
Hamilton/Avnet Electronics
Phone: (203) 762-0361

Hamden
Arrow Electronics
Phone: (203) 248-3801

FLORIDA

Ft. Lauderdale
Arrow Electronics
Phone: (305) 776-7790

Hamilton/Avnet Electronics
Phone: (305) 971-2900

Hollywood
Schweber Electronics
Phone: (305) 922-4506

Orlando
Hammond Electronics
Phone: (305) 241-6601

GEORGIA

Atlanta
Schweber Electronics
Phone: (404) 449-9170

Norcross
Hamilton/Avnet Electronics
Phone: (404) 448-0800

ILLINOIS

Elk Grove
Schweber Electronics
Phone: (312) 593-2740

Elmhurst
Semiconductor Specialists
Phone: (312) 279-1000

Schieler Park
Hamilton/Avnet Electronics
Phone: (312) 671-6082

INDIANA

Indianapolis
Semiconductor Specialists
Phone: (317) 243-8271

KANSAS

Lenexa
Hamilton/Avnet Electronics
Phone: (913) 888-8900

MARYLAND

Baltimore
Arrow Electronics
Phone: (301) 247-5200

Gaithersburg
Pioneer Washington
Electronics
Phone: (301) 948-0710

Hanover
Hamilton/Avnet Electronics
Phone: (301) 796-5000

Rockville
Schweber Electronics
Phone: (301) 881-2970

MASSACHUSETTS

Waltham
Schweber Electronics
Phone: (617) 890-8484

Woburn
Arrow Electronics
Phone: (617) 933-8130

Hamilton/Avnet Electronics
Phone: (617) 933-8000

MICHIGAN

Farmington
Semiconductor Specialists
Phone: (313) 478-2700

Livonia
Hamilton/Avnet Electronics
Phone: (313) 522-4700

Troy
Schweber Electronics
Phone: (313) 583-9242

MINNESOTA

Eden Prairie
Schweber Electronics
Phone: (612) 941-5280

Edina
Hamilton/Avnet Electronics
Phone: (612) 941-3801

Minneapolis
Semiconductor Specialists
Phone: (612) 854-8841

MISSOURI

Hazelwood
Hamilton/Avnet Electronics
Phone: (314) 731-1144

NEW MEXICO

Albuquerque
Hamilton/Avnet Electronics
Phone: (505) 765-1500

NEW YORK

Buffalo
Summit Distributors
Phone: (716) 884-3450

East Syracuse
Hamilton/Avnet Electronics
Phone: (315) 437-2642

Farmingdale. L.I.
Arrow Electronics
Phone: (516) 694-6800

Johnson City
Wilshire Electronics
Phone: (607) 797-1236

Rochester
Hamilton/Avnet Electronics
Phone: (716) 442-7820

Schweber Electronics
Phone: (716) 461-4000

Westbury. L.I.
Hamilton/Avnet Electronics
Phone: (516) 333-5800

Schweber Electronics
Pmne: (516) 334-7474

NORTHERN NEW JERSEY

Cedar Grove
Hamilton/Avnet Electronics
Phone: (201) 239-0800

Saddlebrook
Arrow Electronics
Phone: (201) 797-5800

**SOUTHERN NEW JERSEY
AND PENNSYLVANIA**

Cherry Hill. N.J.
Milgray-Delaware Valley
Phone: (609) 424-1300

Moorestown. N.J.
Arrow/Angus Electronics
Phone: (609) 235-1900

Mt. Laurel. N.J.
Hamilton/Avnet Electronics
Phone: (609) 234-2133

**CENTRAL NEW JERSEY
AND PENNSYLVANIA**

Somerest. N.J.
Schweber Electronics
Phone: (201) 469-6008

Horsham. PA
Schweber Electronics
Phone: (215) 441-0600

NORTH CAROLINA

Greensboro
Hammond Electronics
Phone: (919) 275-6391

OHIO

Beechwood
Schweber Electronics
Phone: (216) 464-2970

Cleveland
Arrow Electronics
Phone: (216) 464-2000

Hamilton/Avnet Electronics
Phone: (216) 461-1400

Pioneer Standard Electronics
Phone: (216) 587-3600

Dayton
Arrow Electronics
Phone: (513) 253-9176

Hamilton/Avnet Electronics
Phone: (513) 433-0610

Pioneer Standard Electronics
Phone: (513) 236-9900

TEXAS

Dallas
Component Specialists
Phone: (214) 357-6511

Hamilton/Avnet Electronics
Phone: (214) 661-8204

Schweber Electronics
Phone: (214) 661-5010

Houston
Component Specialists
Phone: (713) 771-7237

Hamilton/Avnet Electronics
Phone: (713) 780-1771

Schweber Electronics
Pone: (713) 784-3600

UTAH

Salt Lake City
Alta Electronics
Phone: (801) 486-7227

Hamilton/Avnet Electronics
Phone: (801) 262-8451

WASHINGTON

Bellevue
Hamilton/Avnet Electronics
Phone: (206) 746-8750

Seattle
Intermark Electronics
Phone: (206) 767-3160

Liberty Electronics
Phone: (206) 763-8200

FOR SIGNETICS PRODUCTS WORLDWIDE

ARGENTINA

Fapesa I.y.C.
Buenos-Aires
Phone: 652-7438/7478

AUSTRIA

Osterreichische Philips
Wien
Phone: 93 26 11

AUSTRALIA

Philips Industries-ELCOMA
Lane-Cove N.S.W.
Phone: 421261

BELGIUM

M.B.L.E.
Bruselles
Phone: 523 00 00

BRAZIL

Ibrape. S.A.
Sao Paulo
Phone: 287-7144

CANADA

Philips Electron Devices
Toronto
Phone: 425-5161

CHILE

Philips Chilena S.A.
Phone: 39-4001

DENMARK

Miniwatt A/S
Kobenhavn
Phone: (01) 69 16 22

FINLAND

Oy Philips Ab
Helsinki
Phone: 1 72 71

FRANCE

R.T.C.
Paris
Phone: 355-44-99

GERMANY

Valvo
Hamburg
Phone: (040) 3296-1

HONG KONG

Philips Hong Kong. Ltd.
Kwuntong
Phone: 3-427232

INDIA

Semiconductors. Ltd.
(REPRESENTATIVE ONLY)
Bombay
Phone: 293-667

INDONESIA

P.T. Philips-Ralin Electronics
Jakarta
Phone: 581058

IRAN

Berkeh Company. Ltd.
Tehran
Phone: 831564

ISRAEL

Rapac Electronics. Ltd.
Tel Aviv
Phone: 477115-6-7

ITALY

Philips S.p.A.
Milano
Phone: 2-6994

JAPAN

Signetics Japan. Ltd.
Tokyo
Phone: (03) 230-1521

KOREA

Philips Electronics Korea. Ltd.
Seoul
Phone: 44-4202

MEXICO

Electronica S.A. de C.V.
Mexico D.F.
Phone: 533-1180

NETHERLANDS

Philips Nederland B.V.
Eindhoven
Phone: (040) 79 33 33

NEW ZEALAND

E.D.A.C. Ltd.
Wellington
Phone: 873-159

NORWAY

Electronica A.S.
Oslo
Phone: (02) 15 05 90

PHILIPPINES

Philips Industrial Dev.. Inc.
Makata-Rizal
Phone: 868951-9

SINGAPORE/MALAYSIA

Philips Singapore Pte.. Ltd.
Toa Payoh
Phone: 538811

SOUTH AFRICA

E.D.A.C. (PTY). Ltd.
Johannesburg
Phone: 24-6701-3

SPAIN

Copresa S.A.
Barcelona
Phone: 329 63 12

SWEDEN

Elcoma A.B.
Stockholm
Phone: 08/67 97 80

SWITZERLAND

Philips A.G.
Zurich
Phone: 01/44 22 11

TAIWAN

Philips Taiwan. Ltd.
Taipei
Phone: (02) 551-3101-5

THAILAND/LAOS

Saeng Thong Radio. Ltd.
Bangkok
Phone: 527195. 519763

UNITED KINGDOM

Mullard. Ltd.
London
Phone: 01-580 6633

UNITED STATES

Signetics International Corp.
Sunnyvale. California
Phone: (408) 739-7700

**VENEZUELA. PANAMA.
ARUBA. TRINIDAD**

Instrulab C.A.
Caracas
Phone: 614138

Feb. 1977