



AMD CUSTOMER EDUCATION

ED29116

DESIGNING

WITH THE Am29116

16-BIT

BIPOLAR MICROPROCESSOR

LECTURE
VOLUME I

ED29116

"Designing with the Am29116 16-Bit Bipolar Microprocessor"

by

Barbara Albert
Arndt Bode, Dr. rer. nat.
J.W. Locke, Ph.D.

March 1983

Customer Education Center
Advanced Micro Devices, Inc.

Ø-1Ø

ED29116

Ø-1Ø

ED29116

DAY 1

CHAPTER Ø

INTRODUCTION

Designing with the AM29116

The AM29116 combines the advantages of the AM29ØØ bitslice microprocessor family:

- bipolar (fast)
- microprogrammable (flexible)
- LSI (low cost and reliable)
- industry standard (many circuits
and second sources)
- development aids
(hardware/firmware/software)

with:

- a very high degree of integration (16 Bit,...)
- special purpose features (CRC,...)
- very high speed (1ØØ ns cycle)

Remember the advantages of a bipolar, microprogrammable, industry-standard LSI Family with development aids (as presented in ED29ØØA, EDSYS29).

From such a Family of parts you may construct:

- fast machines (125 ns cycle time)
- with almost any desired architecture
- with any desired instruction set
- allowing "soft" upgrades
(field changes, adaptations, system expansions)
simply by changing the contents
of the microprogram memory
- with relatively small parts count (5Ø DIPS)
allowing for low cost and high reliability
- with design time shortened by parallel
hardware/firmware development on System 29
- easily debugged with hardware & software aids
- with documentation automatically enforced
by programming techniques used on System 29

As you design with an the industry-standard Am29ØØ Family:

- your task is facilitated by the largest number of special purpose circuits fully compatible with your application
(we will see the part numbers on the next pages)
- the Am29ØØ Family is constantly expanding
(and is an increasingly-active product line)
- as AMD's technology advances you get pin-compatible faster devices
(e.g. Am29Ø1 / Am29Ø1A / Am29Ø1B / Am29Ø1C / ...!)
- you have the largest number of second sources
(but AMD products are faster and more reliable!!)

Am2900 Product Family

<u>Part Number</u>	<u>Description</u>
Am2901	4-bit Bipolar Microprocessor Slice
Am2902	High-Speed Look-Ahead Carry Generator
Am2903	4-bit Bipolar Microprocessor Slice
Am2904	Status and Shift Control Unit
Am2905	Quad 2-Input Bus Transceiver, O.C.
Am2906	Quad 2-Input Bus Transceiver, O.C.
Am2907/2908	Quad Bus Transceiver with Interface Logic
Am2909	4-Bit Microprogram Sequencer Slice
Am2910	12-Bit Microprogram Sequencer
Am2911	4-Bit Microprogram Sequencer Slice
Am2912	Quad Bus Transceiver
Am2913	8-Input Priority Interrupt Expander
Am2914	8-Input Vectored Priority Interrupt Controller
Am2915	Quad Bus Transceiver, Three-State
Am2916	Quad Bus Transceiver, Three-State
Am2917	Quad Bus Transceiver, Three-State
Am2918/Am29LS18	Quad D-Register
Am2919	Quad D-Register with Dual Three-State Outputs
Am2920	Octal D-Type Flip-Flop
Am2921	One-of-Eight Decoder with Polarity Control
Am2922	8-Input Multiplexer with Control Register

Am2900 Product Family (continued)

<u>Part Number</u>	<u>Description</u>
Am2923	8-Input, Three-State Multiplexer
Am2924	Three-Line to Eight-Line Decoder/Demultiplexer
Am2925	System Clock Generator and Driver
Am2926	Quad Bus Driver/Receiver, Inverting
Am2927/2928	Quad 3-State Bus Transceiver with Clock Enable
Am2929	Quad Bus Driver/Receiver, Non-Inverting
Am2930	4-Bit Program Control Unit Slice
Am2932	4-Bit Program Control Unit / Push-Pop Stack Slice
Am2940	DMA Address Generator
Am2942	Programmable Timer/Counter/DMA Address Generator
Am2946/2947	Octal 3-State Bidirectional Bus Transceiver
Am2948/2949	Octal 3-State Bidirectional Bus Transceiver
Am2950/2951	8-Bit Bidirectional I/O Port with Handshake
Am2952/2953	8-Bit Bidirectional I/O Port
Am2954/2955	Octal Register, Three-State
Am2956/2957	Octal Latch, Three-State
Am2958/2959	Octal Buffer/Line Driver/Line Receiver, 3-State
Am2960	16-Bit Memory Error Detection & Detection Unit
Am2961/2962	4-Bit Error Correction Multiple Bus Buffer
Am2964	Dynamic Memory Controller
Am2965	Octal RAM Driver, Inverting
Am2966	Octal RAM Driver, Non-Inverting

Am2900 Product Family (continued)

<u>Part Number</u>	<u>Description</u>
Am29112	Interruptable 8-bit Microprogram Sequencer Slice
Am29116	16-Bit Bipolar Microprocessor
Am29203	4-Bit Bipolar Microprocessor Slice
Am29501	Microprogrammable Signal Processor
Am29516	16 x 16-bit Parallel Multiplier
Am29520	Quad Octal Multilevel Pipeline Register
Am29521	Quad Octal Multilevel Pipeline Register
Am29540	Programmable FFT Address Sequencer
Am29700/29701	64-Bit Non-Inverting RAM
Am29702/29703	64-Bit Schottky RAM
Am29705	16-Word x 4-bit Two Port RAM
Am29707	16-Word x 4-bit Two Port RAM
Am29720/29721	256-Bit Low-Power Schottky RAM
Am29750/29751A	32-Word by 8-Bit PROM
Am29760A/29761A	256-Word by 4-Bit PROM
Am29770/29771	2048-Bit Generic Series Bipolar PROM
Am29774/29775	4096-Bit Registered PROM
Am29803A	16-Way Branch Control Unit for Am2909
Am29811A	Next Address Control Unit for Am2911

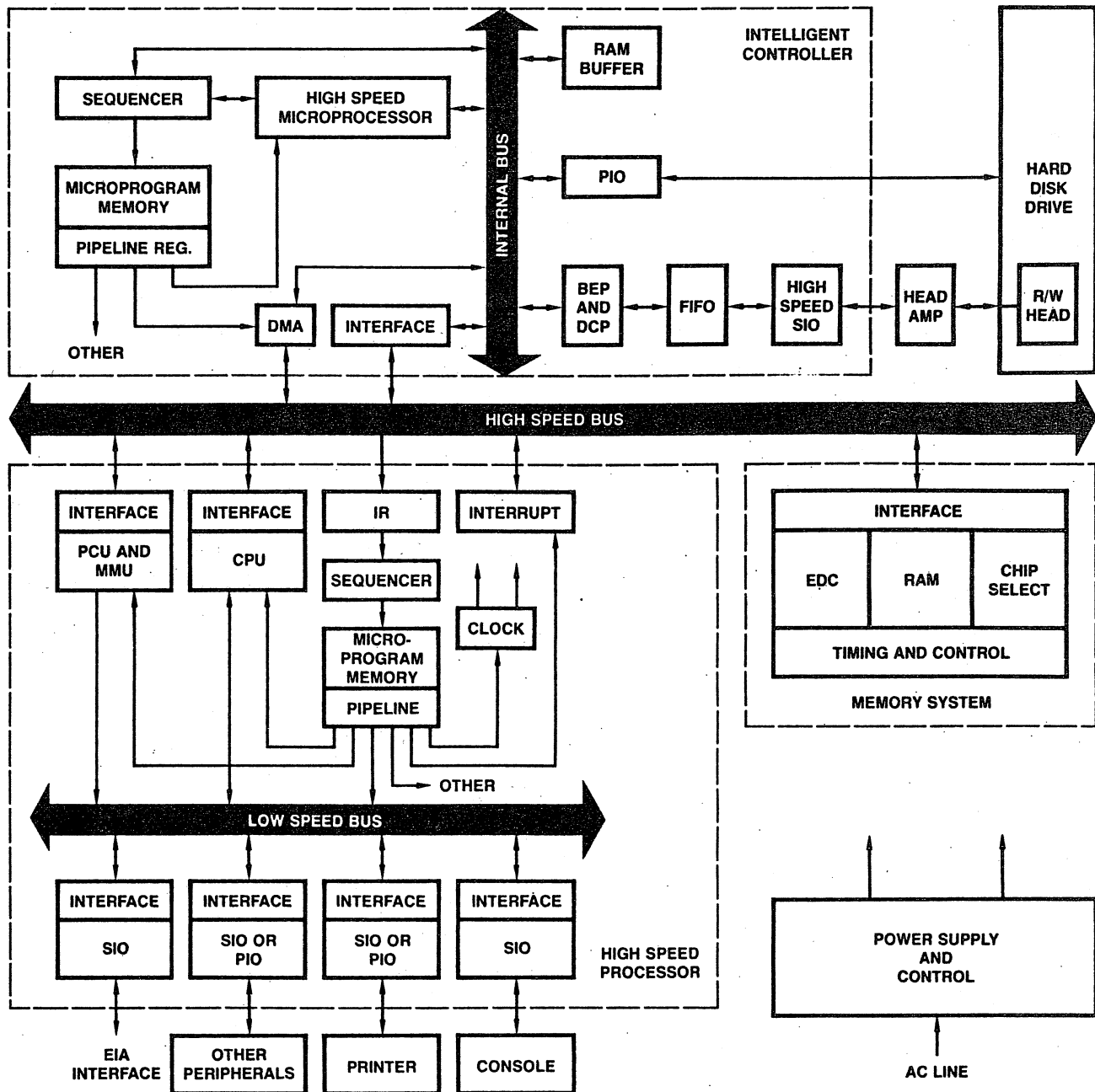
Am29116 Applications - a Preview

Let's have a look at a complete high performance microprogrammed computer system.

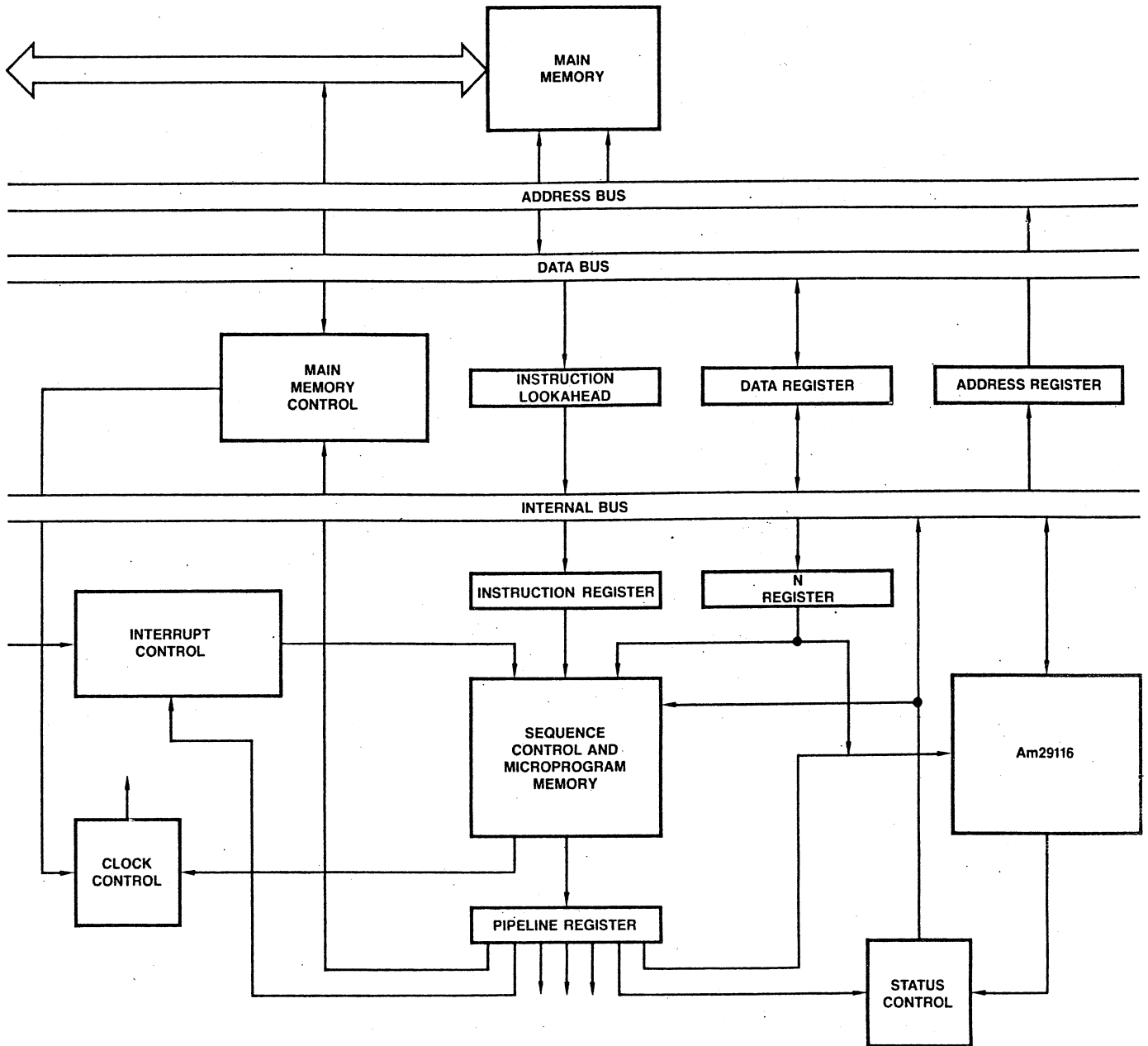
The Am29116 is useful in many different application areas. Typical application examples might include using Am29116:

- as a high speed microprocessor within an intelligent controller
- as the CPU of a high speed processor
- as a high-speed graphics controller

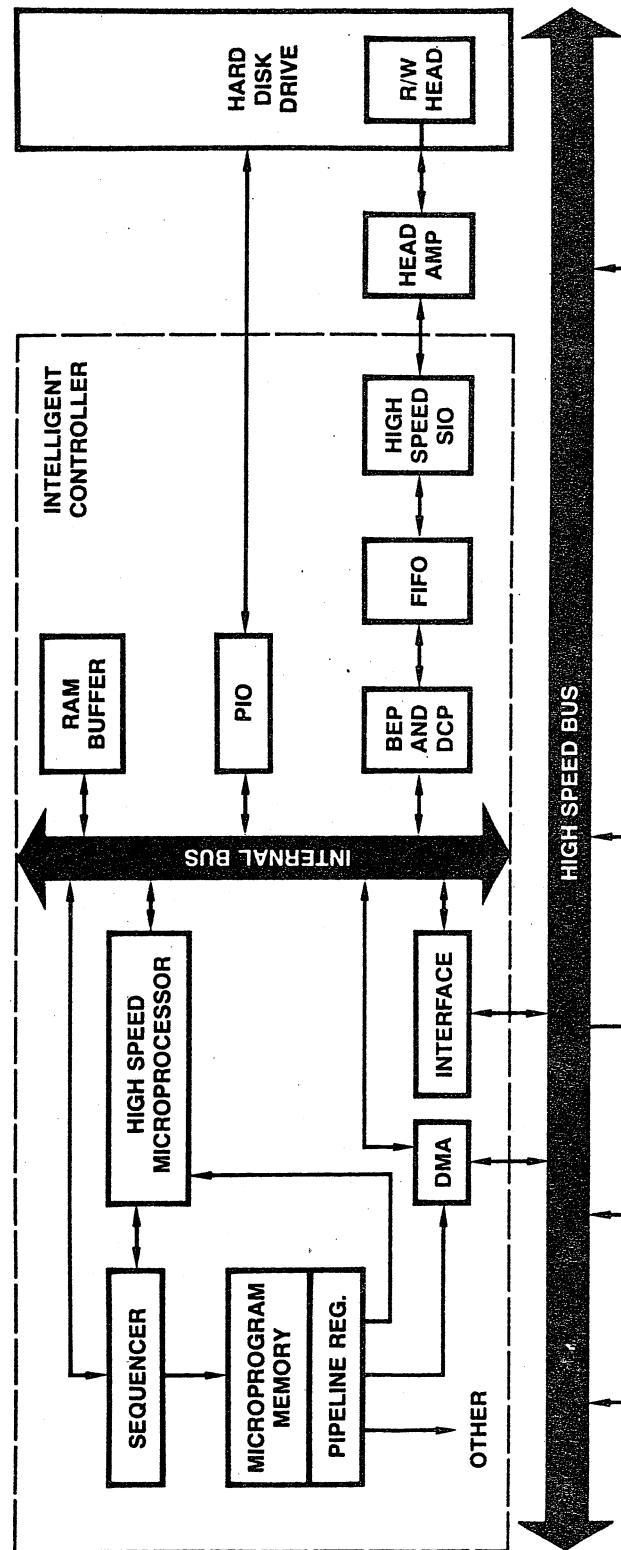
Typical Processor and Controller



The Am29116 as a 16-Bit CPU



MPR-819



Structure of an Intelligent Controller

Am29116 Applications - a Preview (continued)

- The Am29116 architecture and instruction set is optimized for high performance peripheral controllers:
 - graphics controllers
 - disk controllers
 - communication controllers
 - front-end concentrators
 - modems
 - .
 - .
 - .
- Am29116 bit-oriented, rotate-and-merge, rotate-and-compare, and CRC-calculation instructions are particularly useful in such controller applications.
- The Am29116 also performs well for general purpose CPU applications (especially when combined with the Am29516 16 x 16 multiplier) as it has a complete arithmetic and logic instruction set.

Am29116 Controller Applications

The original Am2900 Family products are used in many controller applications due to their features such as microprogrammability, short cycle times, arithmetic and logic instructions etc.

But many controllers require additional features:

- 1) bit manipulation
- 2) character or byte-wide data handling
- 3) CRC calculations for error control
- 4) a generally richer instruction set
- 5) faster execution

With the Am2901 and enough software you can accomplish 1) through 4) but some functions will execute too slowly.

AMD's Am29116 extends the Am2900 Family to meet all of these controller requirements!

Some Parts of Interest for Controllers

29116	16-Bit Bipolar Microprocessor
29112	Interruptable Sequencer
2914	Interrupt Controller
2940/42	DMA Address Generator
2950/51	Parallel I/O Port
9520	Burst Error Processor
29XXX	Asparagus

Controller Requirements

Commonly Needed Operations:

- Moving Data from Port to Port
- Testing Incoming Signals (Bits)
such as:
 - Status
 - Commands
- Generating Outgoing Signals (Bits)
such as:
 - Status
 - Commands
 - Timing Signals
- Rotating and Shifting
- Executing integer arithmetic
(usually 16-bit signed values)
- Assessing priority

Some of AMD's Design Goals for the Am29116

Microprogrammable

16 Bits

100ns Microcycle

52-Pin Dip

+ 5V only

TTL Compatible I/O

Extensive instruction set

Many registers

Hardware for faster shifting, priority-encoding
and Cyclic Redundancy Check calculations

CHAPTER 1

Presenting the Am29116

- Architecture
- Instruction Set
- Timing
- Definition File

[REDACTED]

Am29116 Outstanding Features

- Emitter-coupled logic (ECL) internally for speed
- TTL I/O for easy interfacing
- 16-bit data paths:
 - 16-bit ALU
 - . full carry lookahead
 - . 16-bit Word Mode or 8-bit Byte Mode
 - 32-word x 16-bit register file (here called RAM)
 - . single port architecture
 - . different source and destination addresses are selectable by using an additional, external multiplexer
 - 16-bit data latch
 - 16-bit barrel shifter
 - . works in byte or word mode
 - . rotates up 1 to 15 bits in one cycle
(remember: up shift by n bits is equivalent to down shift by (16-n) bits)

Am29116 Outstanding Features (continued)

- 8-bit status register
- Condition code generator/multiplexer
 - 12 different test conditions
- Immediate instruction capability:
 - . first microcycle - instruction is latched
 - . second microcycle - immediate data is read via the instruction lines
- CRC generation
 - . any polynomial of 16 bits or less
(80% of CRC applications require a 16-bit polynomial)
- powerful instruction set
- fixed width
- fast (100 ns cycle time)
- 52-pin DIP
- single 5 Volt power supply

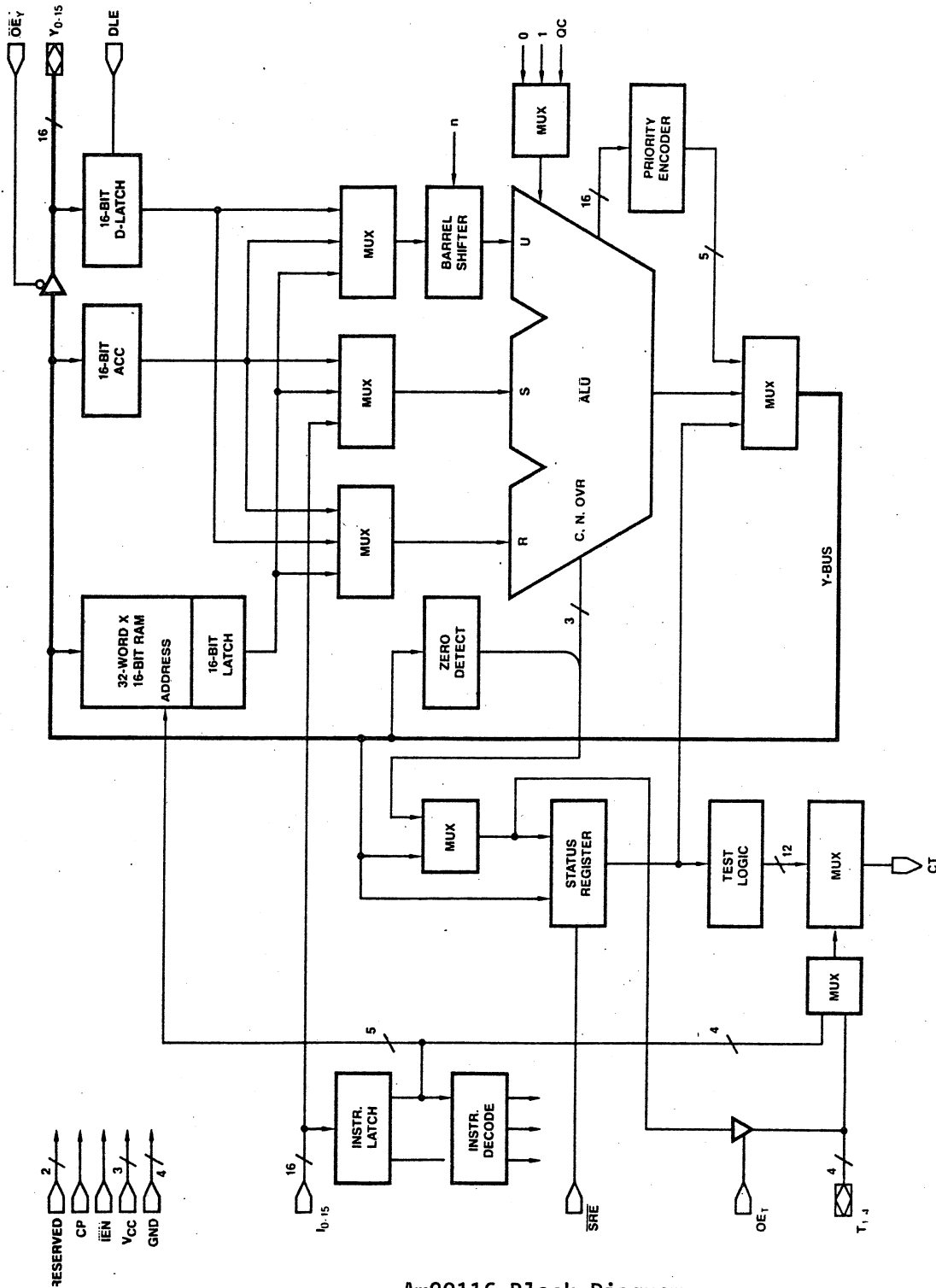
1-40

ED29116

1-40

Am29116 ARCHITECTURE

MPR-740



Am29116 Block Diagram

32-Word x 16-bit Register File - RAM

- Single Port
i.e.: only one of the two source operands
can come from the RAM
- 16-bit latch at RAM output:
 - . transparent when clock (CP) is high
 - . latched when clock goes low
- Data is written to RAM:
 - . when CP is low
 - AND IEN is low (instruction enable)
 - AND RAM is the destination of the instruction
- In Byte Mode:
Instructions alter only the lower 8 bits of a register
- In Word Mode:
Instructions alter all 16 bits of a register
- With extended timing and an external multiplexer:
The RAM address can be changed during an instruction.
(permitting different source and destination registers)

Accumulator - ACC

- 16-bit edge-triggered register
- Accepts data on rising edge of clock:
provided IEN is low
and ACC is the destination of the destination
- Responds to Byte Mode instructions
and to Word Mode instructions

Data Latch - D-Latch

- 16 bit level-controlled register
- Accepts data from the external Y-Bus
- Transparent when DLE input is high
(data latch enable)
- Latched when the DLE input is low:
 - latches all 16 bits at once
 - cannot latch a byte only

Barrel Shifter

- Rotates the "U"-input to the ALU
- Can rotate data from
 - RAM
 - ACC
 - D-latch
- Word Mode: Rotates up 1 to 15 bits
in a single microcycle
- Byte Mode: Rotates up 1 to 7 bits
in a single microcycle
(only the lower byte is altered)

Arithmetic Logic Unit - ALU

- High speed ALU
(full carry lookahead across all 16 bits)
- 16 bits wide
- One, two or three operands
- Executes all of the usual
one- and two-operand functions:

Pass

AND, NAND, OR, NOR, EXOR, EX-NOR

Addition, Subtraction

Complement, Negate

- Executes three-operand instructions:
 - rotate and merge
 - masked rotate and compare
- All of these operations function in Word or Byte Mode
- Includes hardware for Cyclic Redundancy Check calculations

Arithmetic Logic Unit - ALU (continued)

- Produces three status outputs:

C	N	OVR
Carry	Negative	Overflow

- Z status is generated by separate zero-detect logic
- Carry-in multiplexer allows the selection of:
zero, one, or the stored carry (QC)

Priority Encoder

- Produces a binary-weighted code indicating the location of the highest order one at its input
- Operates on the output of the ALU:
operand AND $\overline{\text{mask}}$
- Produces a 5-bit result
- Word and Byte Mode are available

Status Register

- Holds an 8-bit status word:

with content Q_i for $i = 0, \dots, 7$.

Flag3	Flag2	Flag1	LINK	OVR	N	C	Z
-------	-------	-------	------	-----	---	---	---

- Flag1-Flag3: user-definable flags
- LINK: shift-linkage bit
- OVR: overflow
- N: negative
- C: carry
- Z: zero
- Most instructions update the lower 4 bits of the status register when \overline{SRE} and \overline{IEN} are both low
- Certain instructions do not alter status:
 - . NOP
 - . Save-Status
 - . Test-Status
- The LINK bit is updated after each shift
- The user-definable flags are altered only by a Set-Status, Reset-Status or a Word-Mode Load-Status Instruction
- Link status is updated after each shift

Status Register (continued)

- Loaded from internal Y-bus
- Saved via internal Y-bus
- The status register may be a source:

- in Word Mode ...

Any 16-bit register:

0 0 0 0 0 0 0 0	S S S S S S S S
-----------------	-----------------

Zero fills
high byte

- in Byte Mode ...

Any 16-bit register:

unchanged	S S S S S S S S
-----------	-----------------

Alters only
lower byte

- The lower 4 status bits are available on the T-bus when OE_T is high

Condition-Code Generator/Multiplexer

- Generates 12 condition-code test signals
- The MUX selects one of them & puts it on the CT output
- The MUX can be controlled in 2 different ways:
 - by a TEST instruction
(which takes a microcycle to execute)
 - or by the use of the T-bus as an input
(requires wider microword but allows
the simultaneous execution of any instruction)

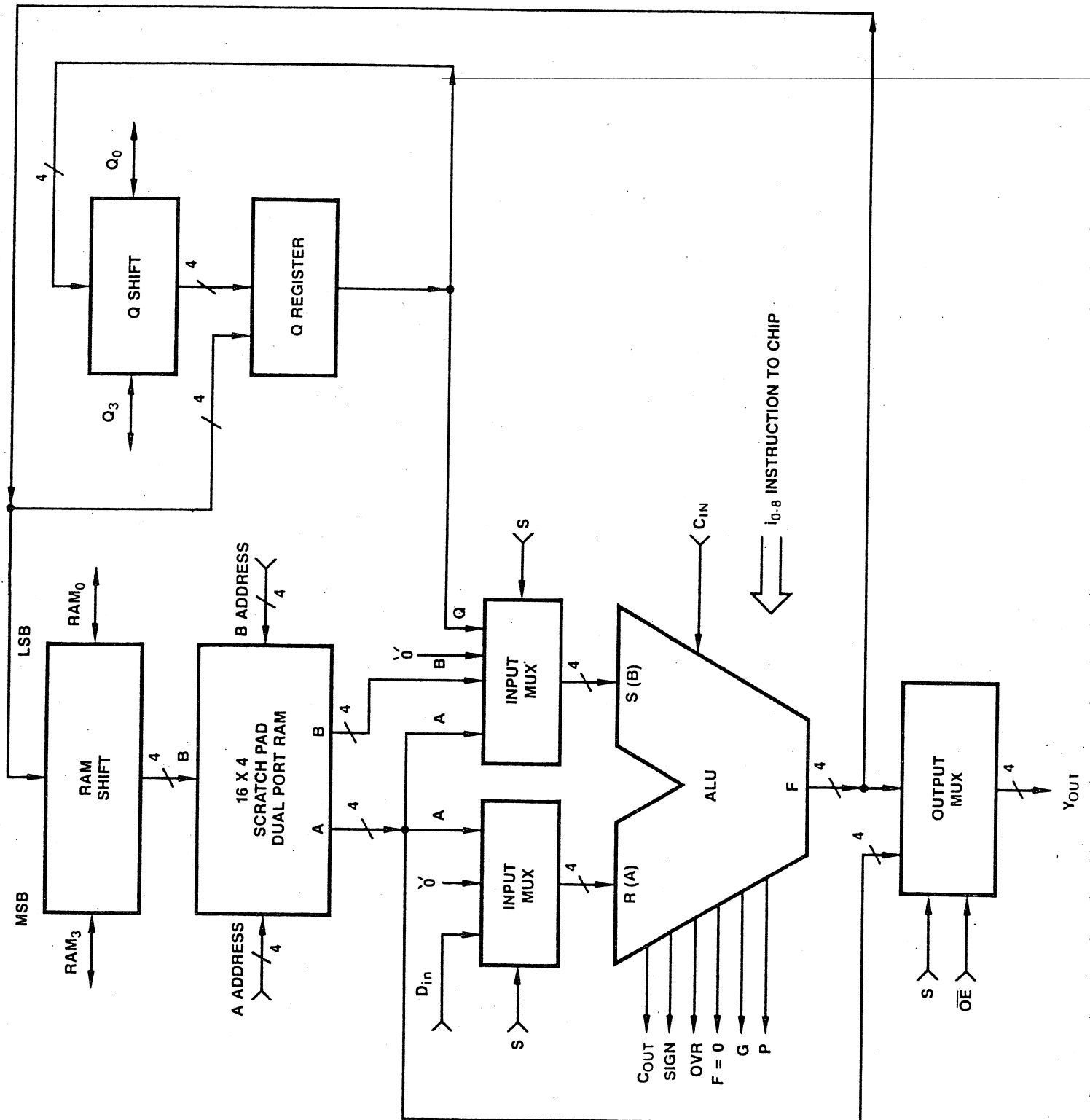
Tri-State Buffers

- Bidirectional 16-bit Y-bus
 - enabled when \overline{OE}_Y is low
- Bidirectional 4-bit T-bus
 - enabled when OE_T is high
 - makes OVR,N,C,Z available
 - with the T-bus output disabled (OE_T low)
you can drive the T-bus from an external source
to select the test condition

Instruction Latch and Decoder

- Normally transparent for all instructions except those specifying immediate data
- On receiving an instruction requiring immediate data
 - the instruction is latched on the first cycle
 - on the next cycle, the instruction lines are used as data that is conveyed to the S-input of the ALU

Am2901 4-Bit Microprocessor Slice



It is instructive to compare the Am2901 and the Am29116 processors.

Am2901 and Am29116 Compared

Am29116	Am2901
<ul style="list-style-type: none">● 16-bit fixed width <p>Fewer external lines</p> <ul style="list-style-type: none">- lower connection complexity- no delay between slices <p>16 bits is ideal for controllers and many other applications</p> <ul style="list-style-type: none">● 32 registers <p>Reduces the number of main memory accesses</p> <p>The availability of the ACC helps to offset the limitation of a single port. By having a single port, 5 pins are saved (that would have carried the second RAM address).</p> <p>Expandable to two register address structure with an external MUX, an additional 5-bit microinstruction field and extended timing</p>	<p>4-bit slice</p> <p>Flexibility in word length</p> <p>16 registers</p> <p>Two-Port Architecture</p>

Am2901 and Am29116 Compared (continued)

Am29116

- 16-bit barrel shifter
- Very fast
(rotates 1 to 15 bits
per microcycle)
- 8-bit status register
- 3 user-definable flag bits

- Condition-Code Generator/MUX

Allows testing of 12 conditions
on chip for minimum delay

You can extend the number of tests
by using the T-bus to output
OVR,N,C,Z to an external device
such as an Am2904.

Am2901

Q-shifter, ALU-shifter

Slower: 1 bit per cycle

Does support arithmetic
(multiplication and
double precision)

4-bit status register
via additional, external
hardware such as Am2904

1-160

ED29116

1-160

Am29116 INSTRUCTION SET

Am29116 Instruction Set

11 Types of Instructions:

- Single Operand
- Two Operand
- Single-Bit Shift
- Bit-Oriented
- Rotate by n Bits
- Rotate & Merge
- Rotate & Compare
- Prioritize
- CRC (cyclic redundancy check)
- Status
- No-Op

3 Types of Data:

- Bit
- Byte
- Word

ALU Sources

RAM

ACC

D-Latch

Immediate

ALU Destinations

RAM

ACC

None (i.e. no destination on the
Am29116 itself. The ALU output is
always put on the Y-bus, however)

Operand Source/Destination Combinations

Instruction Type	Operand Combinations (note 1)	
Single Operand	Source (R/S)	Destination
	RAM (note 2) ACC D D (ØE) (note 3) D (SE) (note 3) I (note 4) Ø	RAM ACC Y-bus Status ACC and Status
Two Operand	Source (R) Source (S)	Destination
	RAM ACC RAM I D RAM D ACC ACC I D I	RAM ACC Y-bus

- Notes: 1. When there is no dividing line between the R and S operands or between source and destination, the two must be used as a given pair. But where there exists such a separation, any combination of them is possible.
2. In the single-operand instruction, RAM cannot be used when both ACC and Status are designated as destinations.
3. ØE = zero extended, SE = sign extended.
4. "I" indicates immediate data

Operand Source/Destination Combinations (continued)

Instruction	Operand Combinations	
Single-bit Shift	Source (U)	Destination
	RAM ACC ACC D D D	RAM ACC Y-bus RAM ACC Y-bus
Rotate by n Bits	Source (U)	Destination
	RAM ACC D	RAM ACC Y-bus
Bit-Oriented	Source (R/S)	Destination
	RAM ACC D	RAM ACC Y-bus

Operand Source/Destination Combinations (continued)

Instruction Type	Operand Combinations		
	Rotated Source (U)	Mask (S)	Non-Rotated Source/Destination (R)
Rotate and Merge	D	I	ACC
	D	RAM	ACC
	D	I	RAM
	D	ACC	RAM
	ACC	I	RAM
	RAM	I	ACC
Rotate and Compare	Rotated Source (U)	Mask (S)	Non-Rotated Source/Destination (R)
	D	I	ACC
Rotate and Compare	D	I	RAM
	D	ACC	RAM
	RAM	I	ACC
Prioritize (note 1)	Source (R)	Mask (S)	Destination
	RAM ACC D	RAM ACC I Ø	RAM ACC Y-bus

Note: 1. In the prioritize instructions,
operand and mask must be from different sources.

Operand Source/Destination Combinations (continued)

Instruction Type	Operand Combinations		
Cyclic Redundancy Check	Data in	Destination	Polynomial
	QLINK	RAM	ACC
Set or Reset Status	Bits Affected		
	OVR, N, C, Z LINK Flag 1 Flag 2 Flag 3		
Store Status	Source		Destination
	Status		RAM ACC Y bus
Load Status	Source (R)	Source (S)	Destination
	D ACC D	ACC I I	Status Status and ACC

Operand Source/Destination Combinations (continued)

Instruction Type	Operand Combinations
Test Status	Test Condition
	(N \oplus OVR)+Z N \oplus OVR Z OVR Low C Z+ \overline{C} N LINK Flag 1 Flag 2 Flag 3

Single Operand Instructions

Field Definitions

	15	14	13	12	9	8	5	4	0
<u>Single Operand RAM:</u> SOR	B/W		Quad		Opcode		SRC-Dest		RAM Address

	15	14	13	12	9	8	5	4	0
<u>Single Operand Non-RAM:</u> SONR	B/W		Quad		Opcode		Source		Dest

Opcode Specifies:

- Move
- Complement
- Increment
- Negate (i.e. two's complement)

Status Generated:

- Flag 3, Flag 2, Flag 1 and LINK status remain unchanged
- OVR,N,C,Z are updated

Single Operand Instructions (continued)

15 14 13 12 9 8 5 4 0

Instruction ¹	B/W ²	Quad ³	Opcode	R/S ⁴ Dest ⁴		RAM Address	
SOR	$\emptyset = B$ $1 = W$	10	1100 MOVE SRC --> Dest	0000 SORA	RAM	000000	RAM Reg 00
			1101 COMP SRC --> Dest	0010 SORY	RAM
			1110 INC SRC + 1 --> Dest	0011 SORS	RAM
			1111 NEG SRC + 1 --> Dest	0100 SOAR	ACC	111111	RAM Reg 31
				0110 SODR	D		
SONR	$\emptyset = B$ $1 = W$	11		0111 SOIR	I		
				1000 SOZR	\emptyset		
				1001 SOZER	D($\emptyset E$)		
				1010 SOSER	D(SE)		
				1011 SORR	RAM		
Instruction	B/W	Quad	Opcode	R/S		Destination	
SONR	$\emptyset = B$ $1 = W$	11	1100 MOVE SRC --> Dest	0100 SOA	ACC	000000	NRY Y Bus
			1101 COMP SRC --> Dest	0110 SOD	D	000001	NRA ACC
			1110 INC SRC + 1 --> Dest	0111 SOI	I	001000	NRS Status
			1111 NEG SRC + 1 --> Dest	1000 SOZ	\emptyset	001001	NRAS ACC, Status
				1001 SOZE	D($\emptyset E$)		
				1010 SOSE	D(SE)		

Notes: 1. The instruction mnemonic designates different instruction formats used in the Am29116. They are useful in assembly microcode with the System 29 AMDASMTM meta assembler.

2. B = Byte Mode, W = Word Mode.

3. Quad: Each instruction format is divided into quadrants. These quadrants were defined mainly for convenience in classification of the instruction set and addressing modes.

1. R = Source, S = Source, Dest = Destination.

Two Operand Instructions

Field Definitions:

	15	14	13	12	9	8	5	4	0
<u>Two Operand RAM 1</u> (TOR1)	B/W		Quad		SRC-SRC -Dest		Opcode		RAM address
<u>Two Operand RAM 2</u> (TOR2)	B/W		Quad		SRC-SRC -Dest		Opcode		RAM address
<u>Two Operand Non-RAM</u> (TONR)	B/W		Quad		SRC-SRC		Opcode		Dest

Opcodes:

SUBR (S-R)
 SUBRC (S-R with carry)
 SUBS (R-S)
 SUBSC (R-S with carry)
 ADD (R+S)
 ADDC (R+S with carry)
 AND (R·S)
 NAND ($\overline{R \cdot S}$)
 EXOR ($R \oplus S$)
 NOR ($\overline{R + S}$)
 OR (R+S)
 EXNOR ($\overline{R \oplus S}$)

Two Operand Instructions (continued)

Subtraction on Am29116

Subtraction is executed by the ALU by means of an addition of the two's-complement of the subtrahend to the minuend.

That is, in general:

M	-	S	=	D
minuend		subtrahend		difference

On the Am29116: $M-S$ is replaced by $M+\bar{S}+1$

The effect of this mechanism is to relate the sense of the resultant carry to the state of the borrow condition in this way:

Carry SET --> no borrow
 Carry RESET --> a borrow has occurred

Similarly, a subtraction-with-carry on Am29116 is executed as an addition of the two's-complement of the subtrahend to the minuend with an adjustment based on the stored carry to properly implement a borrow:

SUBC --> $M-S\text{-borrow} = \begin{matrix} (M+\bar{S}+1) & \text{if no borrow} \\ \text{or } (M+\bar{S}+1)-1 & \text{if borrow} \end{matrix}$

i.e. SUBC --> $M-S-1+QC = M+\bar{S}+QC$ on Am29116

Two Operand Instructions (continued)

Status Generated:

- User-definable flags remain unchanged
- LINK status remains unchanged
- For arithmetic instructions: OVR, N, C and Z status are updated
- For logic instructions: N and Z status are updated.
OVR and C are cleared to 0.

Two Operand Instructions (continued)

15 14 13 12		9 8		5 4		0
Instruction	B/W	Quad	R ¹ S ¹	Dest ¹	Opcode	RAM Address
TOR1	0 = B 1 = W	00	0000 TORAA RAM ACC	ACC	0000 SUBR S-R	000000 R00 RAM Reg 00
			0010 TORIA RAM I	ACC	0001 SUBRC S-R-Cy	
			0011 TODRA D RAM	ACC	0010 SUBS R-S	.
			1000 TORAY RAM ACC	Y Bus	0011 SUBSC R-S-Cy	.
			1010 TORIY RAM I	Y Bus	0100 ADD R+S	1111 R31 RAM Reg 31
			1011 TODRY D RAM	Y Bus	0101 ADDC R+S+Cy	
			1100 TORAR RAM ACC	RAM	0110 AND R•S	
			1110 TORIR RAM I	RAM	0111 NAND $\overline{R \cdot S}$	
			1111 TODRR D RAM	RAM	1000 EXOR R0S	
					1001 NOR $\overline{R+S}$	
					1010 OR R+S	
					1011 EXNOR $\overline{R0S}$	

Note: 1. R = Source, S = Source, Dest = Destination

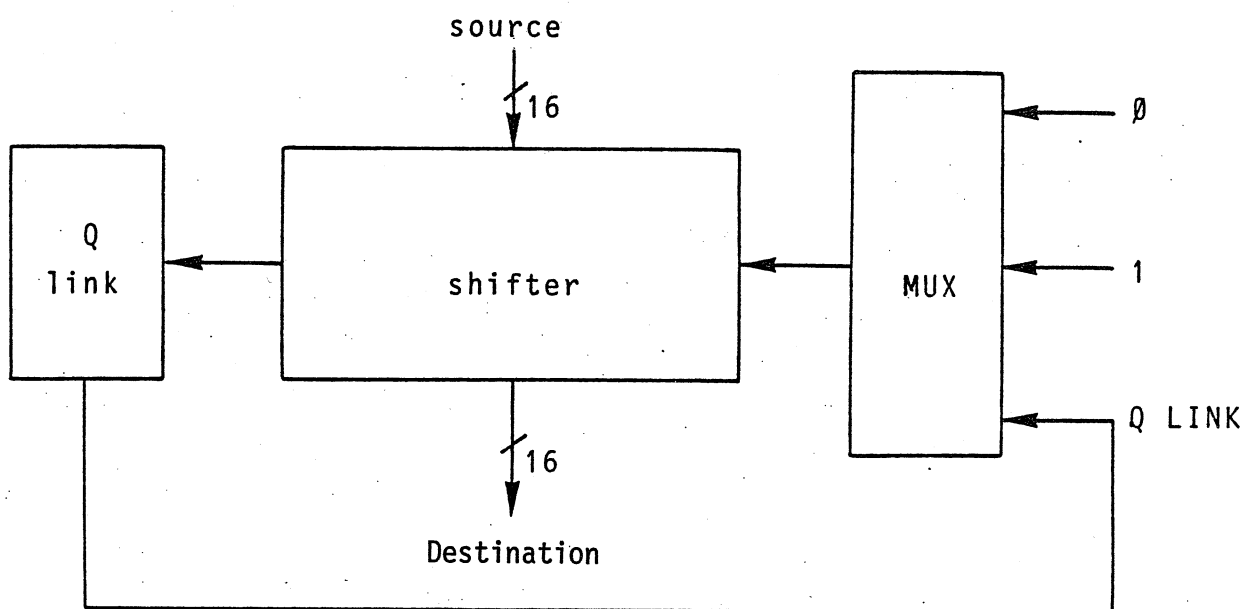
Two Operand Instructions (continued)

Instruction	B/W	Quad	R	S	Dest	Opcode	RAM Address
TOR2	$\emptyset = B$ $1 = W$	1 \emptyset	D	ACC	RAM	Note 1	$\emptyset\emptyset\emptyset\emptyset$ R $\emptyset\emptyset$ RAM Reg $\emptyset\emptyset$ $\cdot \cdot \cdot \cdot$ 11111 R31 RAM Reg 31
							Destination
TONR	$\emptyset = B$ $1 = W$	11	D	ACC	I	Note 1	$\emptyset\emptyset\emptyset\emptyset$ NRY Y Bus $\emptyset\emptyset\emptyset\emptyset 1$ NRA ACC $\emptyset\emptyset 1\emptyset\emptyset$ NRS Status $\emptyset\emptyset 1\emptyset 1$ NRAS ACC, Status

Note 1: Opcodes are the same as for TOR1 (see previous page).

Single Bit Shift Instructions (continued)

Shift Up Function



In Word Mode:

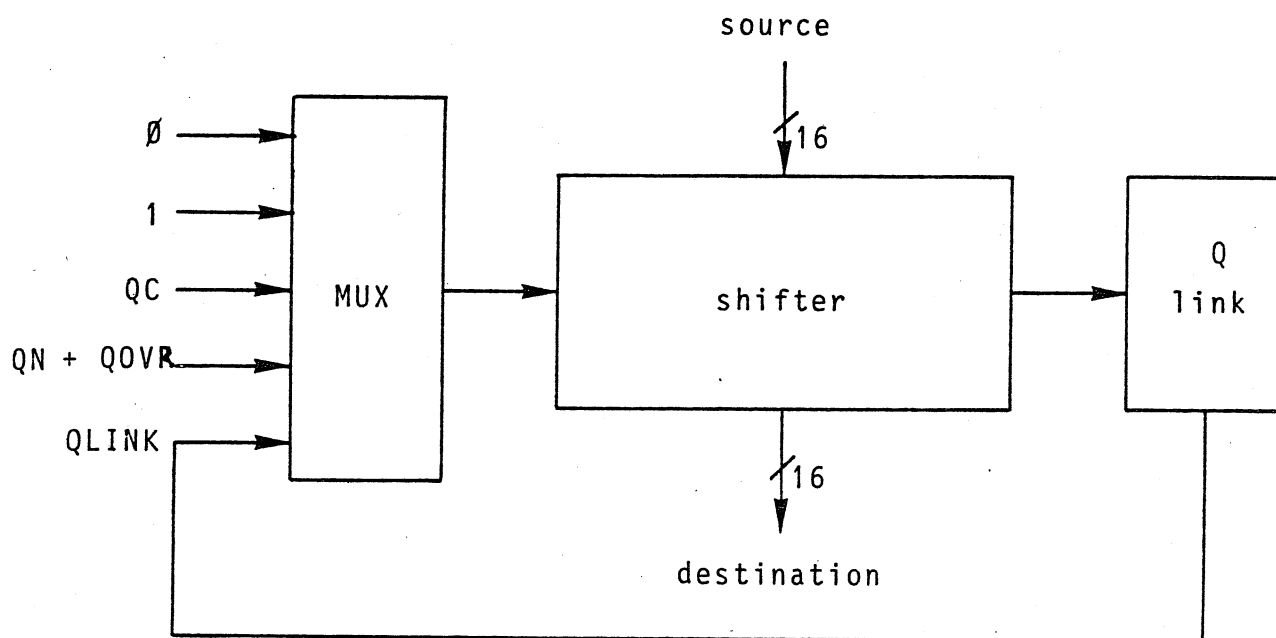
$Dest_i \leftarrow SRC_{i-1}$ for $i=1$ to 15
 $Dest_0 \leftarrow$ Shift Input

In Byte Mode:

$Dest_i \leftarrow SRC_{i-1}$ for $i=1$ to 7
 $Dest_0 \leftarrow$ Shift Input
 $Dest_i$ are not altered for $i=8$ to 15
 However, $Y_8 \leftarrow SRC_7$
 and $Y_i \leftarrow SRC_{i-8}$ for $i=9$ to 15

Single Bit Shift Instructions (continued)

Shift Down Function



In Word Mode:

$Dest_i \leftarrow SRC_{i+1}$ for $i=0$ to 14
 $Dest_{15} \leftarrow$ Shift Input

In Byte Mode:

$Dest_i \leftarrow SRC_{i+1}$ for $i=1$ to 6
 $Dest_7 \leftarrow$ Shift Input
 $Dest_i$ are not altered for $i=8$ to 15
 However, $Y_i \leftarrow SRC_{i-7}$ for $i=8$ to 14
 and $Y_{7,15} \leftarrow$ Shift Input

Single Bit Shift Instructions (continued)

Opcode specifies:

- Shift direction
 - up (multiply by 2)
 - down (divide by 2)
- Shift linkage
 - 0,1,QLINK are selectable in both directions
 - QC,QN0QOVR are selectable in down shifts only

Status generated:

- User-definable flags remain unchanged
- Z is updated
- OVR and C are forced to zero
- LINK and N depend on direction and Byte/Word mode
(shifted output is always loaded into the QLINK):

Direction	Byte/Word	LINK	N
Up	W	SCR ₁₅	SRC ₁₄
	B	SRC ₇	SRC ₆
Down	W	SRC ₀	Shift Input
	B	SRC ₀	Shift Input

Single Bit Shift Instructions (continued)

Appropriate Usage of the Shift Input:

- QLINK for multiple-word shifts.
- QN@QOVR for two's-complement multiplication.

That is, in the accumulation of each stage of the partial product an overflow may occur. In the case of an overflow the sign bit is incorrect. Hence, as the next down shift is executed, the sign extension is to be taken as the complement of the incorrect sign bit.

In the case in which overflow does not occur, the sign bit is correct as it stands. Hence, as the next down shift is executed, the sign bit is extended directly.

The Exclusive OR of the N bit with the OVR bit produces the correct sign extension in all cases.

Single Bit Shift Instructions (continued)

Instruction	B/W	Quad	U ¹ Dest			Opcode	RAM Address		
SHFTR			0110	SHRR	RAM	RAM	0000	R00	RAM Reg 00
			0111	SHDR	D	RAM	0001	SHUP1	Up 1
							0010	SHUPL	Up QLINK
	0 = B 1 = W	10					0100	SHDNZ	Down 0
							0101	SHDN1	Down 1
							0110	SHDNL	Down QLINK
							0111	SHDNC	Down QC
							1000	SHDNOV	Down QNQQOVR
Instruction	B/W	Quad	U ¹			Opcode	Destination		
SHFTNR	0 = B 1 = W	11	0110	SHA	ACC	Same as for SHFTR	000000	NRY	Y Bus
			0111	SHD	D		000001	NRA	ACC

Note: 1. U = Source

Bit Oriented Instructions

Field Definitions

	15	14	13	12	9	8		5	4		0
<u>Bit</u> <u>O</u> riented <u>R</u> AM <u>1</u> : BOR1											
<u>Bit</u> <u>O</u> riented <u>R</u> AM <u>2</u> : BOR2	B/W	Quad		n	Opcode			RAM address			

	15	14	13	12	9	8		5	4		0
<u>Bit</u> <u>O</u> riented <u>N</u> on- <u>R</u> AM: BONR											
	B/W	Quad		n	Opcode			RAM address			

- "n" is the address of bit position within word:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

- Opcodes:

Set Bit n	1 --> n th bit
Reset Bit n	0 --> n th bit
Test Bit n	set Z status from bit n
Load 2 ⁿ	1 --> bit n, 0 --> all other bits
Load 2 ⁿ	0 --> bit n, 1 --> all other bits
Incr by 2 ⁿ	SRC + 2 ⁿ --> SRC
Decr by 2 ⁿ	SRC - 2 ⁿ --> SRC

Bit Oriented Instructions (continued)

Status generated:

- User-definable flags remain unchanged
- N: - is always updated
- Z: - is updated after RESET, TEST, INCREMENT, DECREMENT
 - is cleared after SET or LOAD
- OVR and C: - are updated after INCREMENT or DECREMENT
 - are forced to zero after SET, RESET, TEST, LOAD

Bit Oriented Instructions (continued)

Instruction	B/W	Quad	n	Opcode	RAM Address
BOR1	$\emptyset = B$ $1 = W$	11	\emptyset to 15	1101 SETNR Set RAM, bit n	000000 R00 RAM Reg 00
				1110 RSTNR Reset RAM, bit n
				1111 TSTNR Test RAM, bit n	11111 R31 RAM Reg 31
				Opcode	RAM Address
BOR2	$\emptyset = B$ $1 = W$	10	\emptyset to 15	1100 LD2NR $2^n \rightarrow$ RAM	000000 R00 RAM Reg 00
				1101 LDC2NR $2^n \rightarrow$ RAM
				1110 A2NR RAM+ $2^n \rightarrow$ RAM	11111 R31 RAM Reg 31
				1111 S2NR RAM- $2^n \rightarrow$ RAM	

Note: • Normally: source register = destination register

• With additional external MUX you can specify another destination register

Bit Oriented Instructions (continued)

Instruction	B/W	Quad	n	Opcode ¹
BONR	$\emptyset = B$ $1 = W$	11	\emptyset to 15	$11\emptyset\emptyset$
				000000 TSTNA Test ACC, bit n
				000001 RSTNA Reset ACC, bit n
				000010 SETNA Set ACC, bit n
				000100 A2NA $ACC+2^n \rightarrow ACC$
				000101 S2NA $ACC-2^n \rightarrow ACC$
				000110 LD2NA $2^n \rightarrow ACC$
				000111 LDC2NA $\overline{2}^n \rightarrow ACC$
				100000 TSTND Test D, bit n
				100001 RSTND Reset D, bit n
				100010 SETND Set D, bit n
				100100 A2NDY $D+2^n \rightarrow Y$ Bus
				100101 S2NDY $D-2^n \rightarrow Y$ Bus
				100110 LS2NY $2^n \rightarrow Y$ Bus
				100111 LDC2NY $2^n \rightarrow Y$ Bus

Note 1: In this format the opcode field contains both source and destination.

Rotate By n Bits Instructions

Field Definitions:

			15	14	13	12	9	8		5	4		0
<u>Rotate</u> <u>RAM</u> <u>1</u> :	ROTR1												
<u>Rotate</u> <u>RAM</u> <u>2</u> :	ROTR2												
		B/W		Quad		n			SRC-Dest				RAM address

			15	14	13	12	9	8		5	4		0
<u>Rotate</u> <u>Non-RAM</u> :	ROTRN												
		B/W		Quad		n			1100				SRC-Dest

- "n" is the number of positions to shift:
 - n is in the range 0 to 15 in word mode
 - n is in the range 0 to 7 in byte mode
- There is no explicit opcode for n-bit rotates.
- These instructions rotate up.
To rotate down by "i" bits: $n=16-i$ or $8-i$
- Source is always presented to the U-MUX of the ALU

Rotate by n Bits Instructions (continued)

Example: $n = 3$... rotate up by 3 bit positions

● Word Mode:

Source 0 0 0 1 0 0 1 1 0 1 1 1 1 1 1 1

Destination 1 0 0 1 1 0 1 1 1 1 1 1 1 0 0 0

● Byte Mode:

Source 0 0 0 1 0 0 1 1 0 1 1 1 1 1 1 1

Destination 0 0 0 1 0 0 1 1 1 1 1 1 1 0 1 1

Rotate by n Bits Instructions (continued)

Status generated:

- User-definable flags remain unchanged.
- OVR and C are forced to zero.
- N and Z are updated
to correspond to resulting byte or word.

That is, the N bit of the status byte becomes:

SRC_{15-n} in Word Mode

SRC_{8-n} in Byte Mode

Rotate By n Bits Instructions (continued)

Instruction	B/W	Quad	n	U ¹ Dest ¹	RAM Address
ROTR1	$\emptyset = B$ $1 = W$	$\emptyset\emptyset$	\emptyset to 15	1100 RTRA RAM ACC	000000 R00 RAM Reg 00
				1110 RTRY RAM Y Bus
				1111 RTRR RAM RAM	11111 R31 RAM Reg 31
Instruction	B/W	Quad	n	U ¹ Dest ¹	RAM Address
ROTR2	$\emptyset = B$ $1 = W$	$\emptyset 1$	\emptyset to 15	0000 RTAR ACC RAM	000000 R00 RAM Reg 00
				0001 RTDR D RAM 11111 R31 RAM Reg 31
Instruction	B/W	Quad	n	U ¹ Dest ¹	U ¹ Dest ¹
ROTNR	$\emptyset = B$ $1 = W$	11	\emptyset to 15	1100	11000 RTDY D Y Bus
					11001 RTDA D ACC
					11100 RTAY ACC Y Bus
					11101 RTAA ACC ACC

Note 1: U = Rotated Source Dest = Destination

Rotate and Merge Instructions

Field Definition:

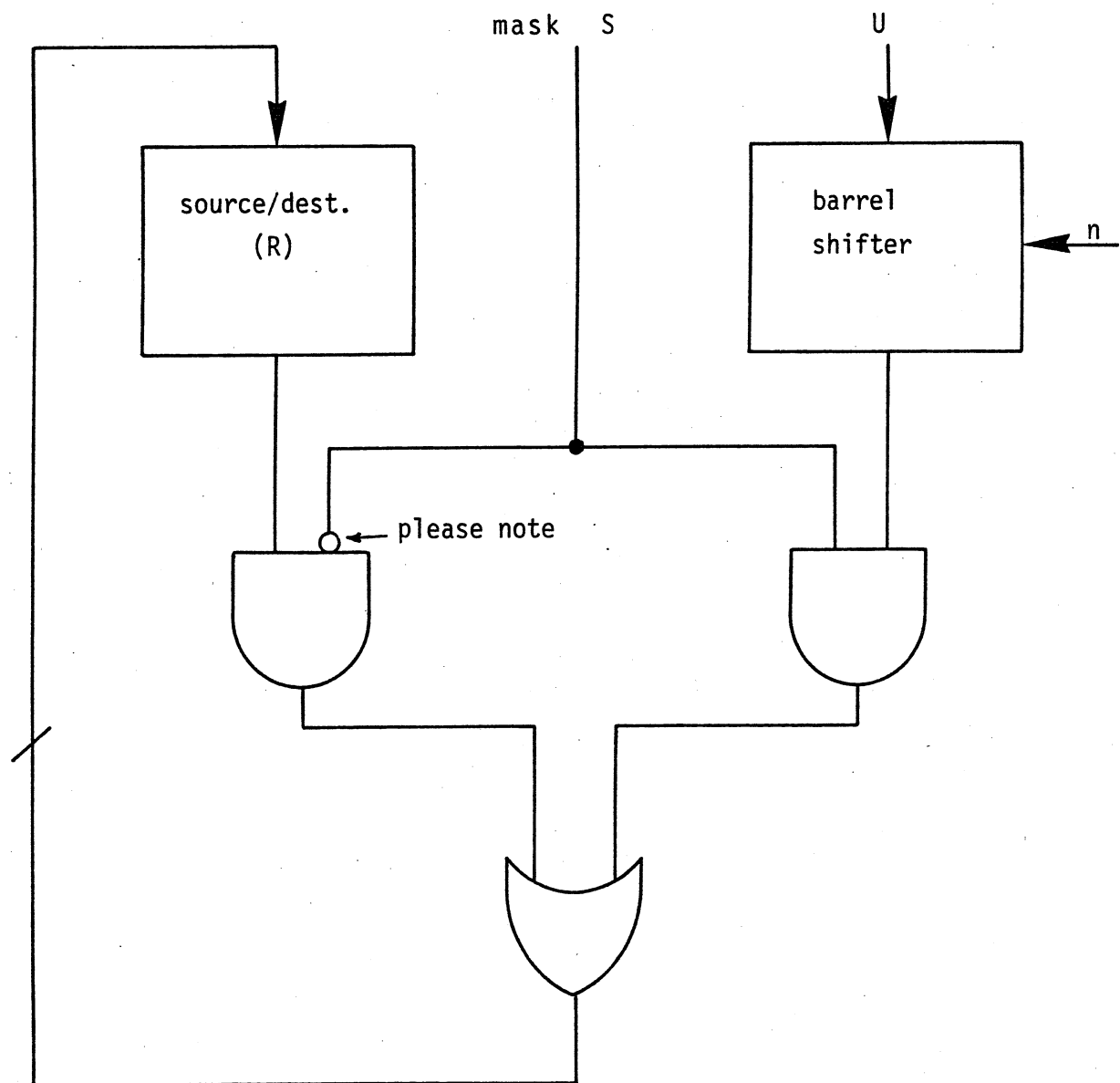
	15	14	13	12	9	8		5	4	0
<u>Rotate</u> and <u>Merge</u> : ROTM	B/W	Quad	n	Rot SRC- Non Rot SRC- Mask				RAM address		

- n is the number of positions to upshift operand U
- Operand R remains unrotated
- Use mask M to select bits:

$$\text{Destination}_i = U_i \quad \text{if } M_i = 1$$

$$\text{Destination}_i = R_i \quad \text{if } M_i = 0$$

- There is no explicit opcode for ROTM
- Choose one of six possible triplets for the two operands and the mask

ROTATE AND MERGE INSTRUCTIONS (continued)

Rotate and Merge Instructions (continued)

Examples:

n = 4 Word Mode

U: 0 0 1 1 0 0 0 1 0 1 0 1 0 1 1 0

Rotated U: 0 0 0 1 0 1 0 1 0 1 1 0 0 0 1 1

R: 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0

Mask S: 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1

 (= R R R R U U U U R R R R U U U U)

Dest (R): 1 0 1 0 0 1 0 1 1 0 1 0 0 0 1 1

n = 4 Byte Mode

U: 0 0 0 1 0 0 1 0 1 1 1 1 1 1 1 0

Rotated U: 0 0 0 1 0 0 1 0 1 1 1 0 1 1 1 1

R: 1 1 1 1 1 1 1 0 0 0 0 1 0 0 0 0

Mask S: 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

 (= R U R U R U R U R U R U R U R U)

Y-bus: x x x x x x x x 0 1 0 0 0 1 0 1

Dest (R): 1 1 1 1 1 1 1 0 0 1 0 0 0 1 0 1

Rotate and Merge Instructions (continued)

Instruction	B/W	Quad	n	U ¹ R/Dest ¹ S ¹		RAM Address	
ROTM	$\emptyset = B$ $1 = W$	$\emptyset 1$	\emptyset to 15	$\emptyset 111$	MDAI D ACC	$\emptyset \emptyset \emptyset \emptyset$	R $\emptyset \emptyset$
				1000	MDAR D ACC	.	.
				1001	MDRI D RAM	11111	R31
				1010	MDRA D RAM	11111	R31
				1100	MARI ACC RAM		
				1110	MRAI RAM ACC		

Note 1:

U = Rotated Source

R/Dest = Non-Rotated Source and Destination

S = Mask

Rotate and Merge Instructions (continued)

Status generated:

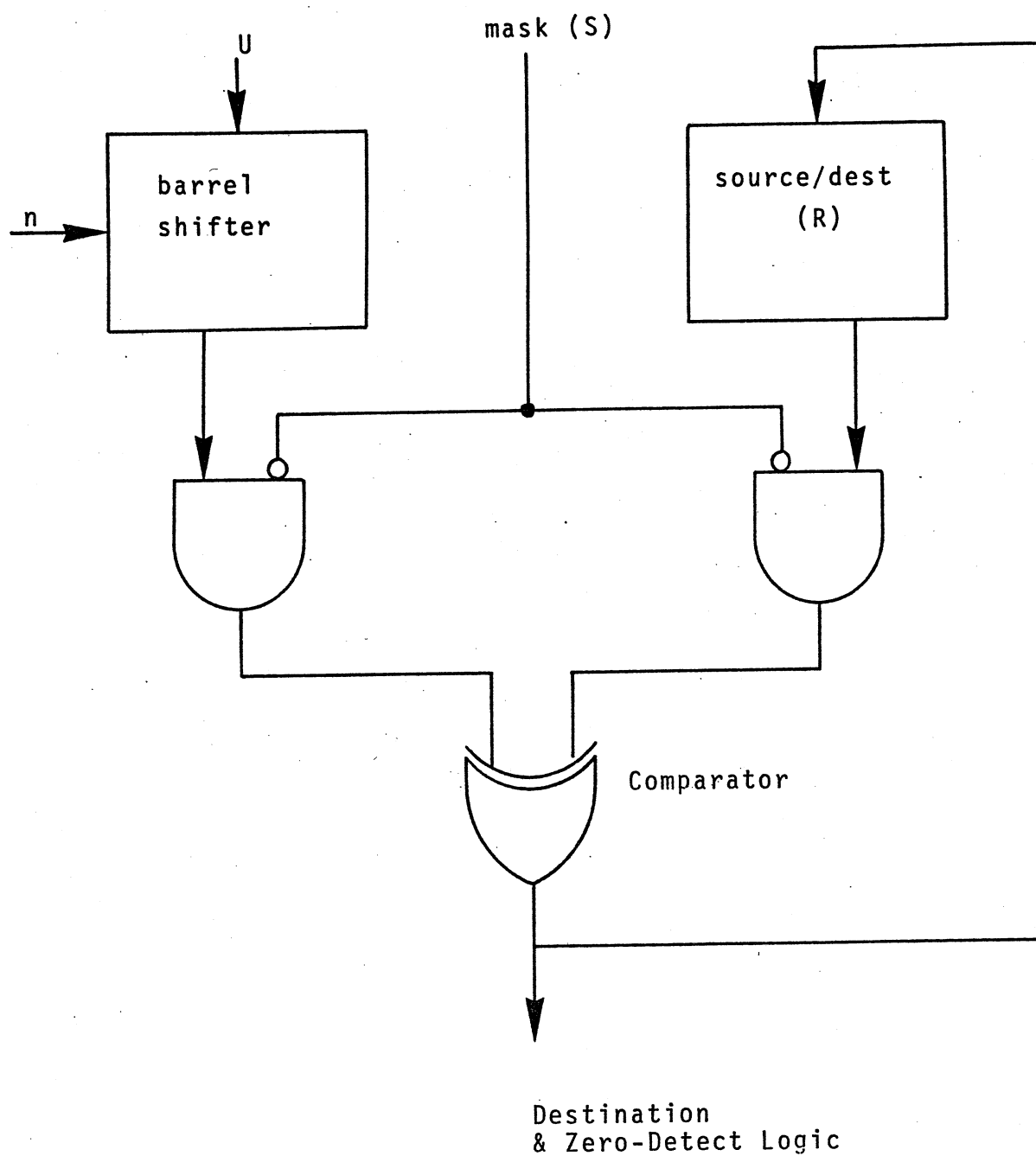
- User-definable flags remain unchanged
- LINK status remains unchanged
- OVR and C are forced to zero
- Z and N are updated to reflect the resulting word or byte

Rotate and Compare Instructions

Field Definitions

	15	14	13	12	9	8		5	4	0
<u>Rotate</u> and <u>Compare</u> : ROTC	B/W	Quad	n	Rot SRC= Non Rot SRC= Mask				RAM address		

- n is the number of positions to upshift operand U
- Operand R is unrotated
- $(\text{Rotated } U \text{ AND } \overline{\text{Mask}}) \text{ EXOR } (R \text{ AND } \overline{\text{Mask}})$
- There is no explicit opcode for ROTC
- Choose one of four possible triplets for the two operands and the mask

ROTATE AND COMPARE INSTRUCTION (continued)

Rotate and Compare Instructions (continued)

Examples:

n = 4 Word Mode

U: 0 0 1 1 0 0 0 1 0 1 0 1 0 1 1 0

Rotated U: 0 0 0 1 0 1 0 1 0 1 0 1 1 0 0 0 1 1

R: 0 0 0 1 0 1 0 1 1 1 1 1 0 0 0 0

Mask S: 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 ...select HI byte

Dest: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
& Z Status=1 ...i.e. does match

Mask S: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 ...select LO byte

Dest: 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1
& Z status=0 ...i.e. does not match

Mask S: 1 1 1 1 1 1 1 1 0 0 1 0 0 0 1 0 ...select
various bits

Dest: 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 1
& Z Status=0 ...i.e. does not match

Rotate and Compare Instructions (continued)

Instruction	B/W	Quad	n	U ¹	R/Dest ¹ S ¹	RAM Address
ROTC	0 = B 1 = W	01	0 to 15	0010 CDAI D	ACC I	000000 R00 RAM Reg 00
				0011 CDRI D	RAM I
				0100 CDRA D	RAM ACC	11111 R31 RAM Reg 31
				0101 CRAI RAM	ACC I	

Note 1: U = Rotated Source R/Dest = Non-Rotated Source and Destination S = Mask

Status generated:

- User definable flags remain unchanged
- Link status remains unchanged
- OVR and C are forced to zero
- N and Z are updated

Prioritize Instructions

Field Definitions:

15 14 13 12 9 8 5 4 0

Prioritize RAM 1: PRT1

B/W	Quad	Destination	Source (R)	RAM Address/ Mask (S)
-----	------	-------------	------------	--------------------------

Prioritize RAM 2: PRT2

B/W	Quad	Mask (S)	Destination	RAM Address/ Source (R)
-----	------	----------	-------------	----------------------------

Prioritize RAM 3: PRT3

B/W	Quad	Mask (S)	Source (R)	RAM Address/ Destination
-----	------	----------	------------	-----------------------------

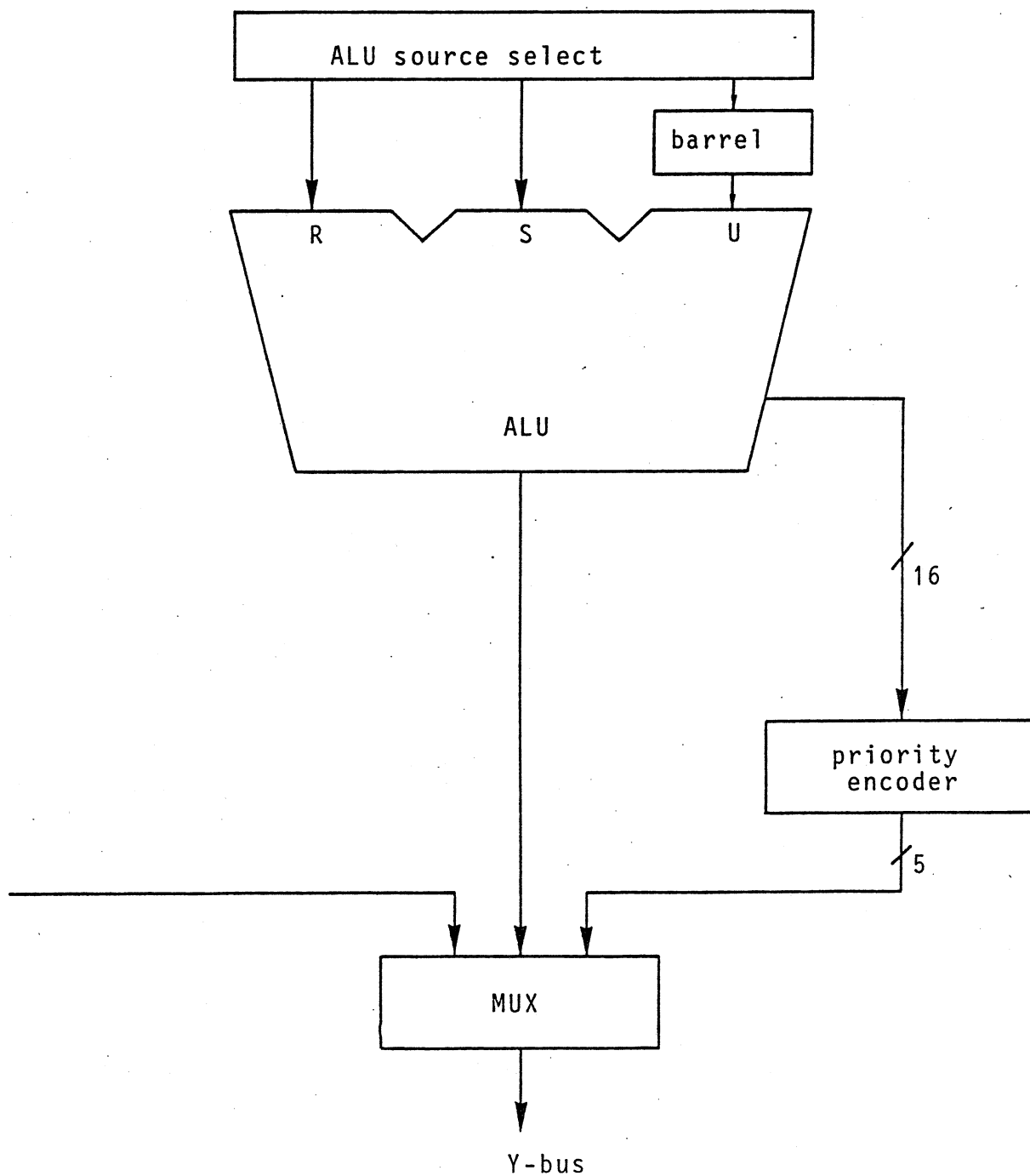
Prioritize Non-RAM: PRTNR

B/W	Quad	Mask (S)	Source (R)	Destination
-----	------	----------	------------	-------------

Note: there is no explicit opcode for prioritize instructions.

Prioritize Instructions (continued)

Remember the logic diagram of the AM 29116:



Prioritize Instructions (continued)

- The input to the priority encoder comes from the output of the ALU.
- ALU operation:
 - . the operand is applied to the R-port
 - . the mask is applied to the S-port
 - . the ALU calculates: Operand AND $\overline{\text{Mask}}$
 - mask bit = 1 Forces the operand bit to zero, thus eliminating it from participation in the priority encoding function
 - mask bit = 0 Passes the operand bit
- The output of the priority encoder is a 5-bit binary code indicating the bit position of the highest priority active bit. That is, the output designates the most significant unmasked bit.

Prioritize Instructions (continued)Word Mode

Highest Priority Active Bit	Encoder Output
--------------------------------	-------------------

NONE

0

15

1

14

2

.

.

.

.

.

.

1

15

0

16

.i.e. output is 16-n
where n is the position of the
highest priority active bit

Byte Mode

Highest Priority Active Bit	Encoder Output
--------------------------------	-------------------

NONE

0

7

1

6

2

.

.

.

.

.

.

1

7

0

8

.i.e. output is 8-n
where n is the position of the
highest priority active bit

Note that the upper byte
is ignored.

Prioritize Instructions (continued)

Examples:

Word Mode

	15	12	11	8	7	4	3	0
Operand:	0	0	0	1	0	0	1	0
Mask:	1	1	1	1	0	0	0	0
ALU result:	0	0	0	0	0	0	1	0

Highest priority active bit is in position 9.

Priority result = 16 - 9 = 7

Byte Mode

	15	12	11	8	7	4	3	0
Operand:	0	0	0	1	0	0	1	0
Mask:	1	1	1	1	0	0	0	0
ALU result:	Not involved				0	0	1	0

Highest priority active bit is in position 3.

Priority result = 8 - 3 = 5

Prioritize Instructions (continued)

Instruction	B/W	Quad	Destination	Source (R)	RAM Address/Mask(S)
PRT1	0 = B 1 = W	10	1000 PRTA ACC 1010 PRTY Y Bus 1011 PRTX RAM	0111 PRTA ACC 1001 PRTD D	000000 R00 RAM Reg 00 111111 R31 RAM Reg 31
Instruction	B/W	Quad	Mask (S)	Destination	RAM Address/Source(R)
PRT2	0 = B 1 = W	10	1000 PRA ACC 1010 PRZ 0 1011 PRI I	0000 PR2A ACC 0010 PR2Y Y Bus	000000 R00 RAM Reg 00 111111 R31 RAM Reg 31

Prioritize Instructions (continued)

Instruction	B/W	Quad	Mask (S)	Source (R)	RAM Address/Destination
PRT3	0 = B 1 = W	10	1000 PRA ACC 1010 PRZ 0 1011 PRI I	0011 PR3R RAM 0100 PR3A ACC 0110 PR3D D	000000 R00 RAM Reg 00 000000 R01 RAM Reg 01 000000 R02 RAM Reg 02 000000 R03 RAM Reg 03 000000 R04 RAM Reg 04 000000 R05 RAM Reg 05 000000 R06 RAM Reg 06 000000 R07 RAM Reg 07 000000 R08 RAM Reg 08 000000 R09 RAM Reg 09 000000 R10 RAM Reg 10 000000 R11 RAM Reg 11 000000 R12 RAM Reg 12 000000 R13 RAM Reg 13 000000 R14 RAM Reg 14 000000 R15 RAM Reg 15 000000 R16 RAM Reg 16 000000 R17 RAM Reg 17 000000 R18 RAM Reg 18 000000 R19 RAM Reg 19 000000 R20 RAM Reg 20 000000 R21 RAM Reg 21 000000 R22 RAM Reg 22 000000 R23 RAM Reg 23 000000 R24 RAM Reg 24 000000 R25 RAM Reg 25 000000 R26 RAM Reg 26 000000 R27 RAM Reg 27 000000 R28 RAM Reg 28 000000 R29 RAM Reg 29 000000 R30 RAM Reg 30 000000 R31 RAM Reg 31
PRTNR	0 = B 1 = W	11	1000 PRA ACC 1010 PRZ 0 1011 PRI I	0100 PRTA ACC 0110 PRTD D	000000 NRY Y Bus 000000 NRA ACC

Prioritize Instructions (continued)

Status generated:

- User-definable flags remain unchanged
- Link status remains unchanged
- OVR and C are forced to zero
- N and Z are updated to reflect the result

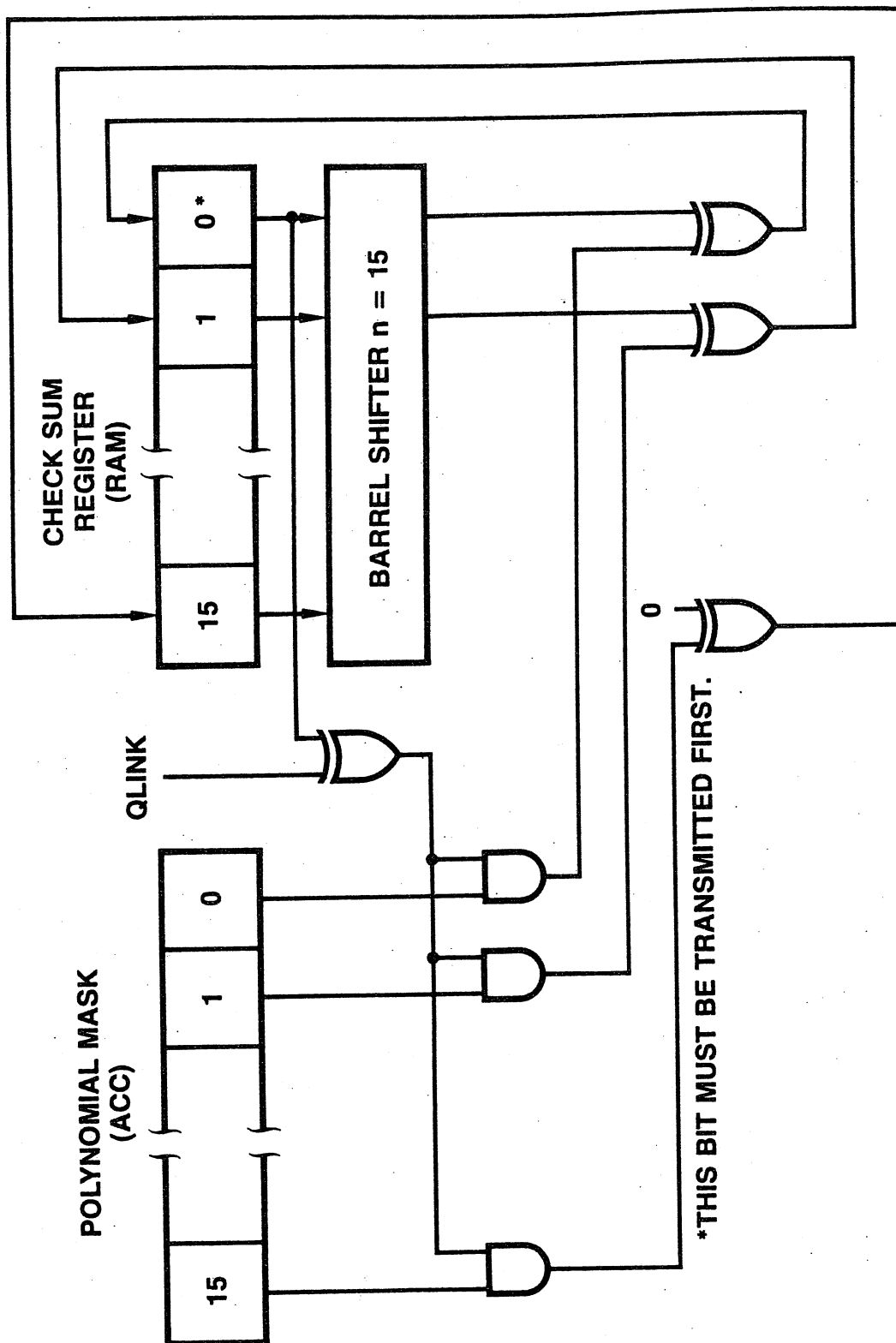
Use this instruction in:

- Selecting the next request to be serviced out of several active requests for service.
- Arithmetic operations (multiplication, normalization) to shorten the number of microcycles.
(not available on Am2901/2903/29203's!)
- N-way branching

CRC Instructions

- Am29116 generates CRC check bits for any polynomial with a remainder of 16 bits or less
(80 to 95% of CRC calculations use 16-bit remainders)
- Two CRC calculations are available:
 - CRC forward - checksum bit 15 is to be transmitted first
 - CRC reverse - checksum bit 0 is to be transmitted first
- CRC calculations are done in word mode.

However, the active number of bits in the remainder may be less than 16.

CRC Reverse Function

CRC Instructions (continued)

Using the CRC instructions to calculate the CRC remainder for a message:

- Place the polynomial mask in the ACC.
- Initialize check sum register in RAM to zero (in most cases).
- Generating a CRC remainder for a k-bit message takes $>2k$ cycles.

For each bit of the message, m_i you execute 2 instructions:

- Shift the bit into the LINK status bit by means of a single-bit shift instruction.
- Execute a CRCF or CRCR instruction. This will update the remainder in RAM in accordance with the current bit.

Mathematically, for the i^{th} message bit m_i , the 'j' CRC remainder bits $C_{i,j}$ are generated as follows:

$$(m_i \oplus (C_{i-1,15} \cdot \text{Mask}_j)) \oplus C_{i-1,j-1} \rightarrow C_{i,j} \quad \dots \text{ for } j = 1 \text{ thru } 15$$

$$m_i \oplus (C_{i-1,15} \cdot \text{Mask}_0) \rightarrow C_{i,0} \quad \dots \text{ for bit } 0$$

- Of course, further instructions will have to be used, generally every 8 or 16 bits, to fetch another byte or word of the message from a source outside the Am29116.
- With the microcycle of 100 ns the Am29116 is able to generate CRC bits at a rate of 5 MHz.

CRC Instructions (continued)Deriving the CRC Mask from the Polynomial: CRC Forward Instruction

Polynomial	Mask Bit Position																29116 Mask (hex)	
	15 14 13 12				11 10 9 8				7 6 5 4				3 2 1 0					
	Coefficients																	
	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
CRC-16 $x^{16}+x^{15}+x^2+1$	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	8005
CRC-16 Reverse $x^{16}+x^{14}+x+1$	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	4003
CRC-CCITT $x^{16}+x^{12}+x^5+1$	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	1021
CRC-CCITT Reverse $x^{16}+x^{11}+x^4+1$	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0811
CRC-12 $x^{12}+x^{11}+x^3+x^2+x+1$	(12	11	10	9	8	7	6	5	4	3	2	1	0)*					80F0
	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	
LRC-8 x^8+1	(8	7	6	5	4	3	2	1	0)*									0100
	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	

* Note: The coefficients for the short polynomials, CRC-12 and LRC-8, are left-justified.

CRC Instructions (continued)Deriving the CRC Mask from the Polynomial: CRC Reverse Instruction

Polynomial	Mask Bit Position																29116 Mask (hex)	
	15 14 13 12				11 10 9 8				7 6 5 4				3 2 1 0					
	Coefficients																	
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
CRC-16 $x^{16}+x^{15}+x^2+1$	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	A001
CRC-16 Reverse $x^{16}+x^{14}+x+1$	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	C002
CRC-CCITT $x^{16}+x^{12}+x^5+1$	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	8408
CRC-CCITT Reverse $x^{16}+x^{11}+x^4+1$	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	8810
CRC-12 $x^{12}+x^{11}+x^3+x^2+x+1$	0	0	0	0	(0	1	2	3	4	5	6	7	8	9	10	11	12)*	0F01
LRC-8 x^8+1	0	0	0	0	0	0	0	0	(0	1	2	3	4	5	6	7	8)*	0080

* Note: The coefficients for the short polynomials, CRC-12 and LRC-8, are right-justified.

CRC Instructions (continued)

Common Polynomials for CRC Applications

Name	Polynomial	Factorization	29116 Mask (hex)
CRC-16	$x^{16}+x^{15}+x^2+1$	$(x+1)(x^{15}+x+1)$	8005
CRC-16 Reverse	$x^{16}+x^{14}+x+1$	$(x+1)(x^{15}+x^{14}+1)$	4003
CRC-CCITT	$x^{16}+x^{12}+x^5+1$	$(x+1)(x^{15}+x^{14}+x^{13}+x^{12}+x^4+x^3+x^2+x+1)$	1021
CRC-CCITT Reverse	$x^{16}+x^{11}+x^4+1$	$(x+1)(x^{15}+x^{14}+x^{13}+x^{12}+x^{11}+x^3+x^2+x+1)$	0811

Useful test examples:

CRC-16 (A759) = 53DF

CRC-16 (A759 EC4D) = 8166

...for case in which the most significant bit
of each byte or word is transmitted first

CRC Instructions (continued)

Field Definitions

	15	14	13	12		5	4		0
CRCF (forward)	1	Quad	0110 0011			RAM address			
CRCR (reverse)	1	Quad	0110 1001			RAM address			

Status generated:

- User-definable flags remain unchanged
- OVR and C are forced to zero
- N and Z are updated
to reflect the resulting CRC remainder.
- LINK status:
 - Forward CRC- loaded from bit 15 of remainder
in RAM prior to execution
 - Reverse CRC- loaded from bit 0 of remainder
in RAM prior to execution

CRC Instructions (continued)

Instruction	B/W	Quad	Opcode	RAM Address		
CRCF	1	10	0110 0011	00000	R00	RAM Reg 00
			
				11111	R31	RAM Reg 31
CRCR	1	10	0110 1001	00000	R00	RAM Reg 00
			
				11111	R31	RAM Reg 31

Status Instructions

Remember the status register:

7	6	5	4	3	2	1	0
Flag3	Flag2	Flag1	LINK	OVR	N	C	Z

Field Definitions:

	15	14	13	12	9	8	5	4	0
Set Status: SETST	0	Quad	1011	1010	Opcode				

Reset Status: RSTST	0	Quad	1010	1010	Opcode				
---------------------	---	------	------	------	--------	--	--	--	--

Save Status to RAM: SVSTR	B/W	Quad	0111	1010	RAM Address/Dest				
---------------------------	-----	------	------	------	------------------	--	--	--	--

Save Status to Non RAM: SVSTNR	B/W	Quad	0111	1010	Destination				
--------------------------------	-----	------	------	------	-------------	--	--	--	--

Status Instructions (continued)

- Set Status

- set all ALU bits (Z,C,N,OVR)
- set LINK
- set Flag1
- set Flag2
- set Flag3

- Reset Status

- reset all ALU bits (Z,C,N,OVR)
- reset LINK
- reset Flag1
- reset Flag2
- reset Flag3

- Save Status

- save status to RAM, ACC or just Y-Bus

- Load Status (Included in the single operand group.
Also two-operand non-RAM instructions
can have status register as destination.)

- in byte mode only the 4 ALU bits are loaded
- in word mode all 8 bits are loaded

Status Instructions (continued)

- ALU status is loaded after all instructions except NOP or status instructions.
- If the status register enable signal ($\overline{\text{SRE}}$) is high the status register is not updated.
- QLINK is updated after each shift but not after a rotate.
- Flag1, Flag2 and Flag3 can be altered by SETST or RESET. They can also be loaded by load status in the word mode.

Status Instructions (continued)

Instruction	B/W	Quad		Opcode	
SETST	0	11	1011 1010	00011	SONCZ Set OVR,N,C,Z
				00101	SL Set LINK
				00110	SF1 Set Flag 1
				01001	SF2 Set Flag 2
				01010	SF3 Set Flag 3
RSTST	0	11	1010 1010	00011	RONCZ Reset OVR,N,C,Z
				00101	RL Reset LINK
				00110	RF1 Reset Flag 1
				01001	RF2 Reset Flag 2
				01010	RF3 Reset Flag 3

Status Instructions (continued)

Instruction	B/W	Quad	Opcode	RAM Address/Destination		
SVSTR	\emptyset = B 1 = W	10	\emptyset 111 1010	000000	R00	RAM Reg 00
			
				111111	R31	RAM Reg 31
Instruction	B/W	Quad	Opcode	Destination		
SVSTNR	\emptyset = B 1 = W	11	\emptyset 111 1010	000000	NR Y	Y Bus
				000001	NRA	ACC

Note: ● Save-Status instructions store 8 status bits.

- Byte Mode: upper byte of destination remains unchanged
- Word Mode: upper byte of destination is forced to zero
- Load-Status instructions are included in the single-operand group.

Test Status

- All testing is performed on the values stored in the status register.
- There are 12 conditional tests:

7 ALU tests

5 other tests

Z	LINK
C	Flag1
N	Flag2
OVR	Flag3
N \oplus OVR	LOW (usually to force
(N \oplus OVR)+Z	an unconditional jump)
Z+C	

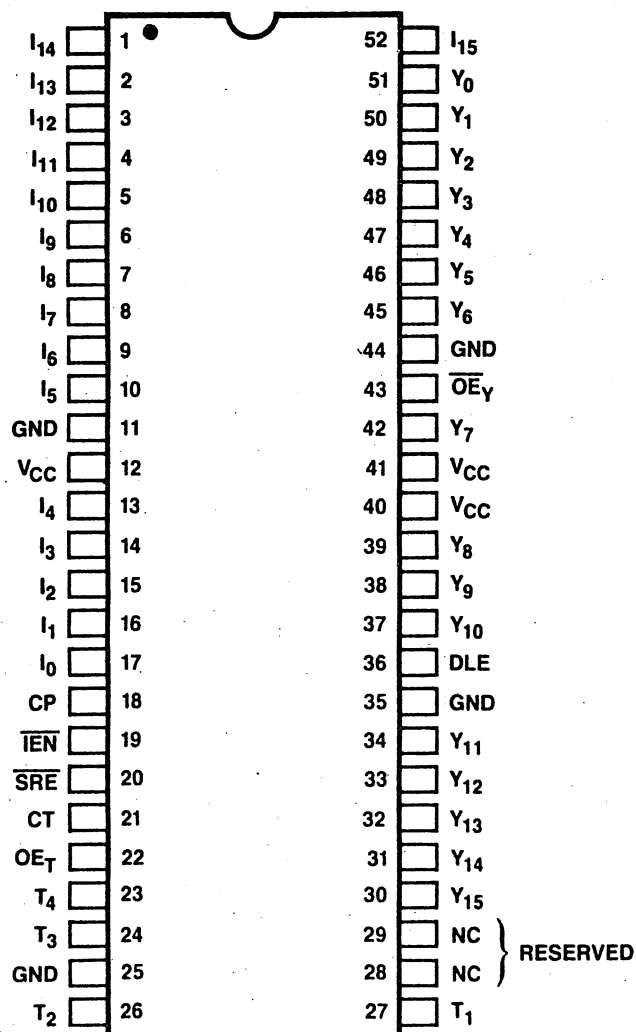
- Testing can be performed during the execution of another instruction using the 4 T-bus lines as inputs to select the test.
 - Obviously if status is tested with a status-test instruction no ALU operation is possible in this same cycle
 - Testing using the T_{bus} lines for test selection permits an ALU operation in the same cycle but requires a wider microword.

Test Status (continued)

Instruction	B/W	Quad		Opcode (CT)		
TEST	0	11	1001 1010	000000	TNOZ	Test (N0OVR)+Z
				000100	TNO	Test N0OVR
				001000	TZ	Test Z
				001100	TOVR	Test OVR
				010000	TLOW	Test LOW
				010100	TC	Test C
				011000	TZC	Test Z + \bar{C}
				011100	TN	Test N
				100000	TL	Test LINK
				100100	TF1	Test Flag 1
				101000	TF2	Test Flag 2
				101100	TF3	Test Flag 3

Am29116 Timing and Electrical Characteristics

CONNECTION DIAGRAM Top View



Note: Pin 1 is marked for orientation.

MPR-812

Pin Definitions:

- Y_0-Y_{15} Y-Bus (input/output)
- & • \overline{OE}_Y Y-Bus Output Enable
 - \overline{OE}_Y HIGH: Y-Bus output drivers are disabled (high-impedance).
Data may be input to the D-latch on Y_0-Y_{15} .
 - \overline{OE}_Y LOW: Y-Bus output drivers are enabled (HIGH or LOW).
 Y_0-Y_{15} outputs ALU data.
- DLE Data Latch Enable
 - DLE HIGH: 16-bit data latch is transparent
 - DLE LOW: 16-bit data latch is latched

Pin Definitions (continued)

- I_0-I_{15} Instruction Inputs (16 pins)
 - used to select one of the operations of the Am29116
 - instructions that require immediate data use these lines as the data path
- \overline{IEN} Instruction Enable
 - \overline{IEN} LOW: . data can be written into the RAM (when the clock is LOW)
 - . ACC can accept data during the LOW-HIGH transition of the clock
 - . status register can be updated when \overline{SRE} is LOW
 - \overline{IEN} HIGH: . CT is disabled as a function of the instruction inputs
 - should be LOW for the first half of the first cycle of an immediate instruction
- \overline{SRE} Status Register Enable
 - \overline{SRE} and \overline{IEN} LOW: status register is updated at the end of all instructions except: NOOP, Save Status & Test Status instructions
 - \overline{SRE} or \overline{IEN} HIGH: inhibits the status register from changing

Pin Definitions (continued)

- CP Clock Pulse (input)
 - CP HIGH: . RAM latch is transparent
 - CP goes LOW: . RAM output is latched.
 - . The instruction is latched in the first cycle of the execution of an immediate instruction
 - CP LOW: . data is written into the RAM if \overline{IEN} is LOW and if RAM is the destination of the operation
 - CP LOW to HIGH transition:
 - . ACC and status register will accept data if \overline{IEN} is low
 - . at end of the second cycle of an immediate instruction causes the exit from immediate mode and the instruction register becomes transparent again

Pin Definitions (continued)

- T_1-T_4 Test Bus (input or output)
 - OE_T HIGH: the 4 lower status bits (Z, C, N, OVR) become outputs on T_1-T_4
 - OE_T LOW: T_1-T_4 are used as inputs to the condition-code MUX which selects CT

- OE_T Output Enable for the T-bus
 - OE_T LOW: T-bus outputs are disabled (high impedance)
 - OE_T HIGH: T-bus outputs are enabled (HIGH or LOW)

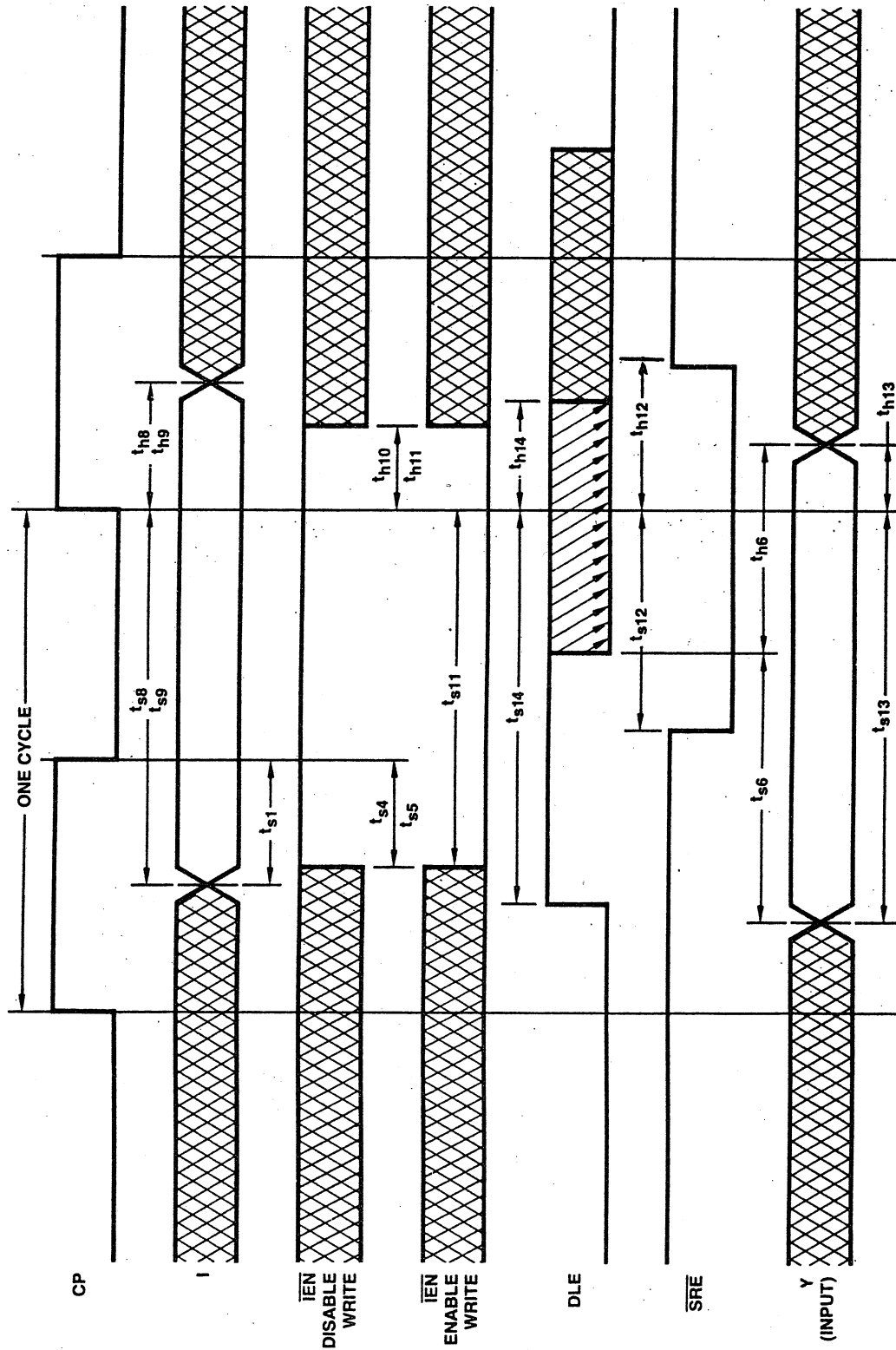
- CT Conditional Test
 - output from one of twelve condition code signals as selected by the condition-code MUX
 - CT HIGH: indicates a passed condition
 - CT LOW: indicates a failed condition

Operating Ranges

The Am29116 is manufactured for commercial and military operating conditions:

Part Number	Range	Temperature	V _{CC}
Am29116DC, DCB	Commercial	T _A = 0 to +70°C	5.0V plus or minus 5%
Am29116DM, DMB	Military	T _A = -55 to +125°C	5.0V plus or minus 10%

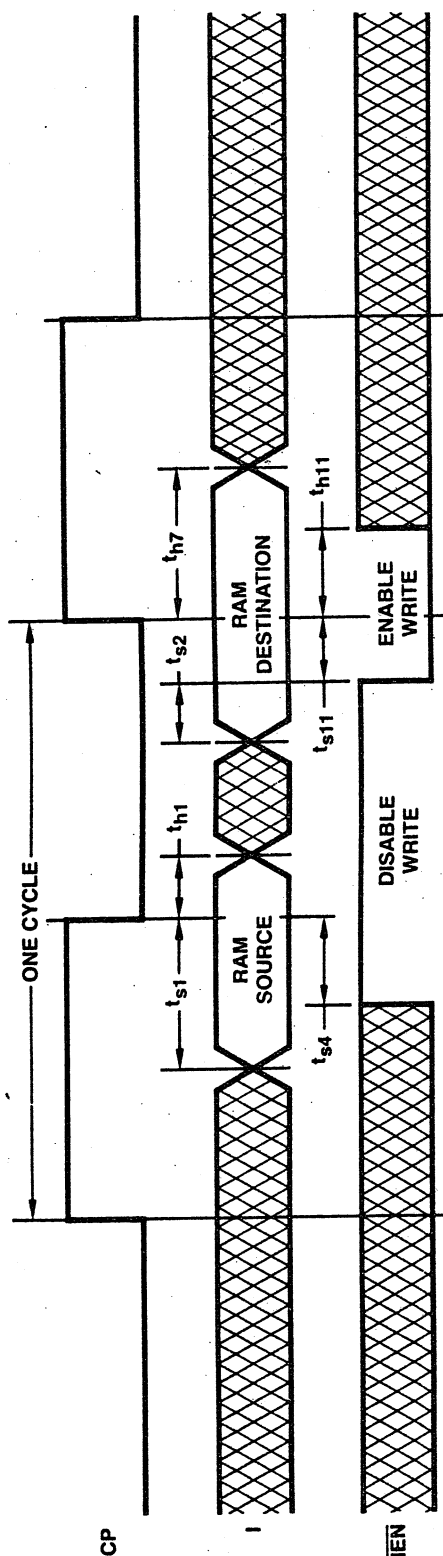
Am29116 SINGLE ADDRESS ACCESS TIMING



If t_{h6} is satisfied, t_{h13} need not be satisfied

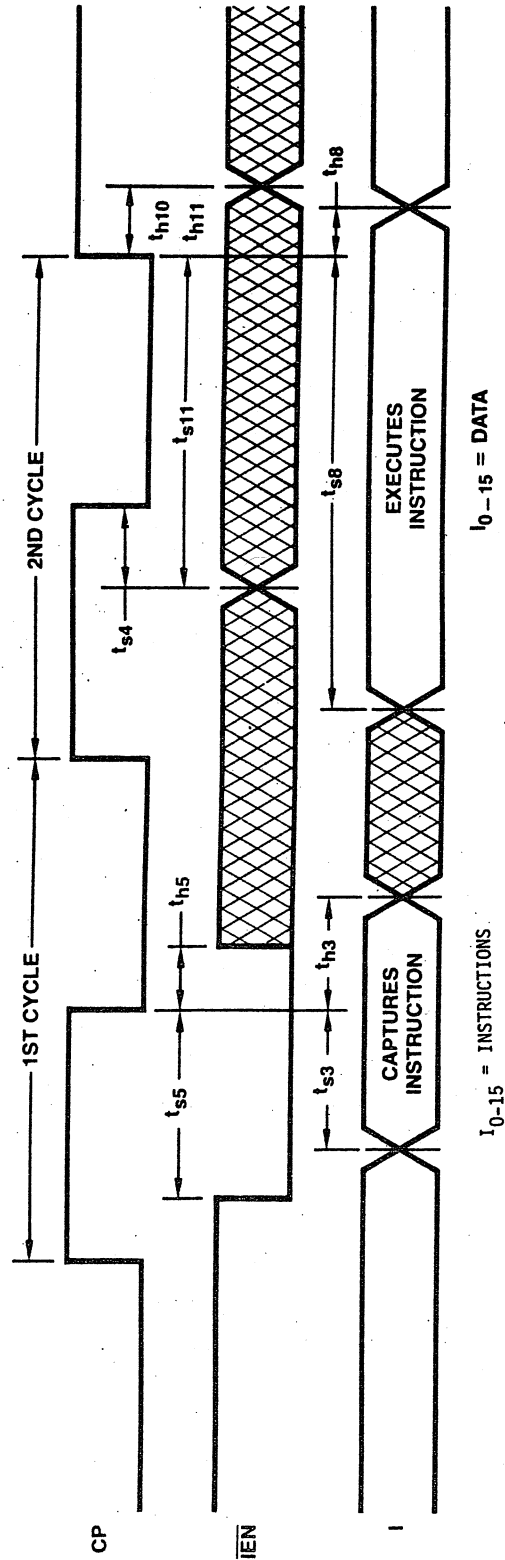
MPR-836

DOUBLE ADDRESS ACCESS TIMING



MPR-837

IMMEDIATE INSTRUCTION CYCLE TIMING



Switching Characteristics

commercial operating range:

$$T_A = 0^{\circ} \text{ to } 70^{\circ} \text{ C}, V_{CC} = 4.75 - 5.25\text{V}, C_L = 50 \text{ pF}$$

Combinational Delays (nsec)

		Outputs		
		Y ₀₋₁₅	T ₁₋₄	CT
	I ₀₋₄ (ADDR)	79	84	—
	I ₀₋₁₅ (DATA)	79	84	—
	I ₀₋₁₅ (INSTR)	79	84	48
Input	DLE	58**	60	—
	T ₁₋₄	—	—	39
	CP	56	62	36
	Y ₀₋₁₅	62**	64*	—
	IEN	—	—	43

*Y₀₋₁₅ must be stored in the Data Latch and is source disabled before the delay to Y₀₋₁₅ as an output can be measured.

**Guaranteed indirectly by other tests.

Enable/Disable Times (nsec) (C_L = 5pF for disable only)

From Input	To Output	Enable		Disable	
		t _{PZH}	t _{PZL}	t _{PHZ}	t _{PLZ}
\overline{OE}_Y	Y ₀₋₁₅	20	20	20	20
OE _T	T ₁₋₄	25	25	25	25

Clock and Pulse Requirements (nsec)

Input	Min Low Time	Min High Time
CP	20	30
DLE	—	15
IEN	22	—

Switching Characteristics (continued)

Setup and Hold Times (nsec)

Input	With Respect to	High-to-Low Transition		Low-to-High Transition		Comment
		Setup	Hold	Setup	Hold	
I ₀₋₄ (RAM ADDR)	CP	(t _{s1}) 24	(t _{h1}) 0	—	—	Single ADDR (Source)
I ₀₋₄ (RAM ADDR)	CP and $\overline{\text{IEN}}$ both LOW	(t _{s2}) 10	Do Not Change		(t _{h7}) 0	Two ADDR (Destination)
I ₀₋₁₅ (DATA)	CP	—	—	(t _{s8}) 65	(t _{h8}) 0	
I ₀₋₁₅ (INSTR)	CP	(t _{s3}) 38 [†]	(t _{h3}) [†] 17	(t _{s9}) 65	(t _{h9}) 0	
$\overline{\text{IEN}}$ HIGH	CP	(t _{s4}) 10	—	—	(t _{h10}) 0	Disable
$\overline{\text{IEN}}$ LOW	CP	— (t _{s5}) 20	— (t _{h5}) [†] 0	(t _{s11}) 22 —	(t _{h11}) ^{††} 0 —	Enable Immediate first cycle
$\overline{\text{SRE}}$	CP	—	—	(t _{s12}) 17	(t _{h12}) 0	
Y	CP	—	—	(t _{s13}) 44	(t _{h13}) 0	
Y	DLE	(t _{s6}) 10	(t _{h6}) 6	—	—	
DLE	CP	—	—	(t _{s14}) 42	(t _{h14}) 0	

[†]Timing for immediate instruction for the first cycle.

^{††}Status register and accumulator destination only.

Examples of Source Code Lines

```
TOR1    W, SUBS, TORAA, R03          & CONT ; R03-ACC --> ACC

SOR      W, MOVE, SORY, R04 & OEY & MADR & CONT ; R04 --> MAR (via Y-bus)
                                                ; (MADR is the MAR write-enable).

SOR      B, MOVE, SOZR, R00          & NOSRE & RTN ; 0 --> R00 low byte only.
                                                ; Do not alter status.
                                                ; Return to caller.

SHFTNR   W, SHA, SHUPZ, NRA          & CONT ; 2*ACC --> ACC
```


CHAPTER 2

Microprogrammed Control of the Am29116, Interfaces for the Am29116

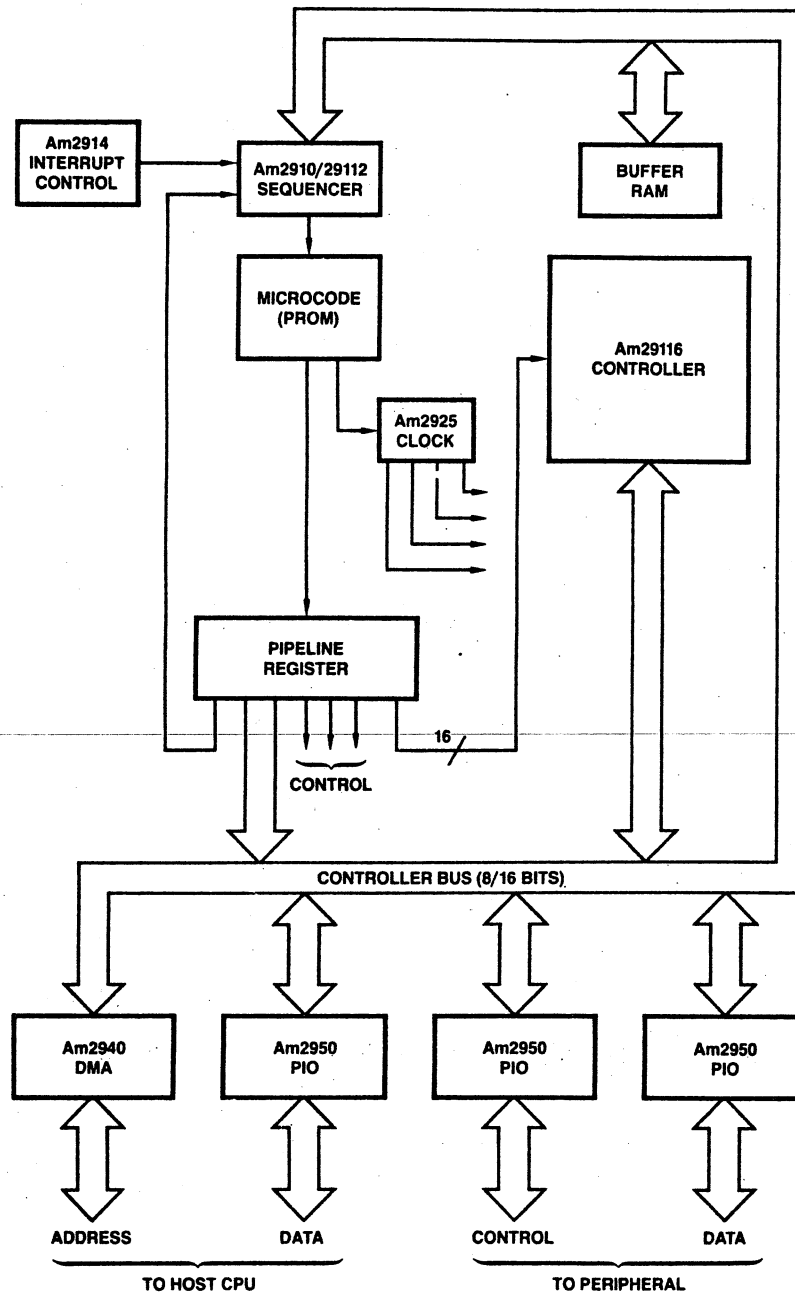
[REDACTED]

The following two pages show two possible applications of the Am29116:

- a controller
- a general purpose CPU

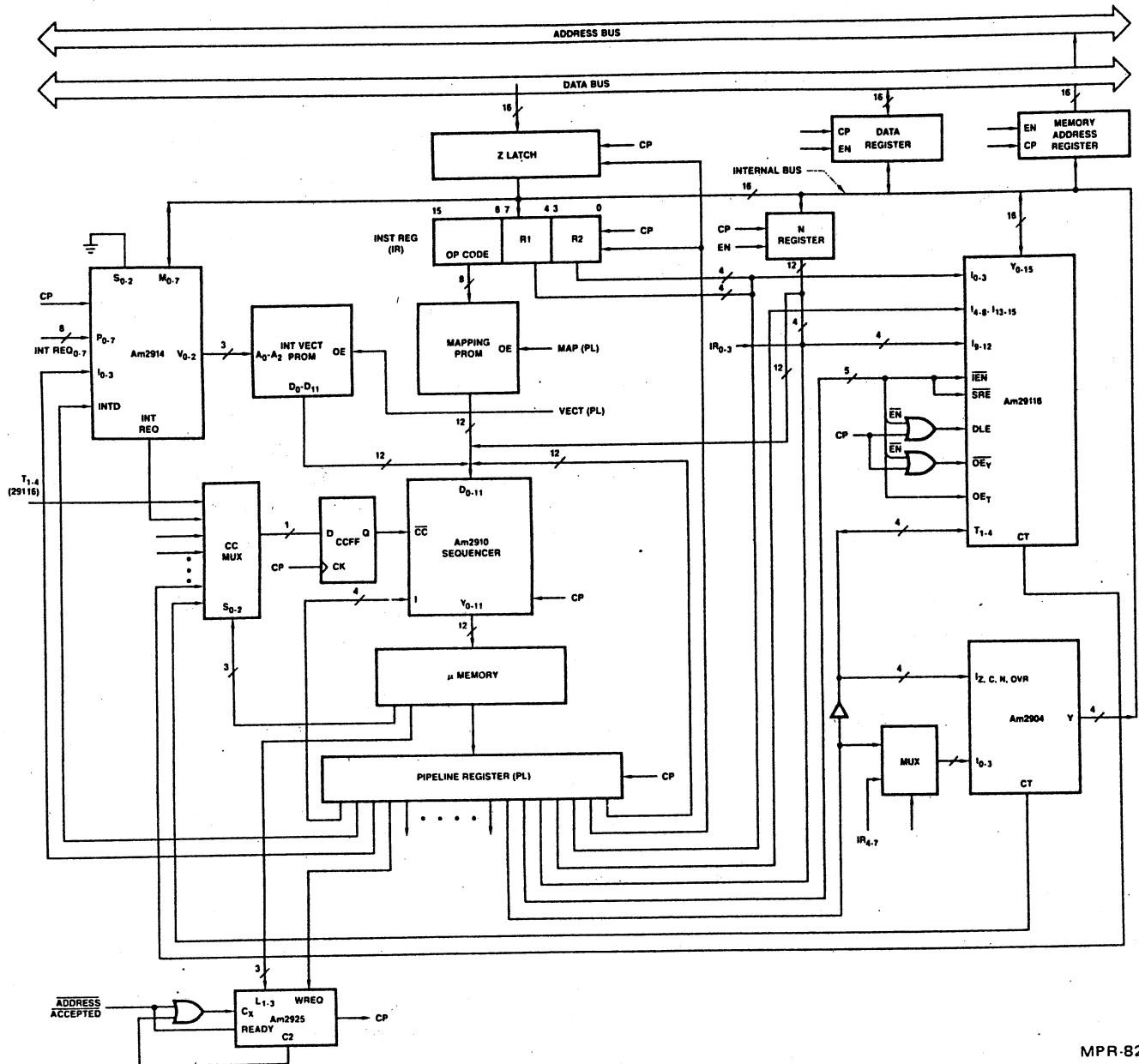
Concentrate on the control sections of these diagrams.

We will discuss other aspects in more detail later.



MPR-835

Controller Based on Am29116



MPR-824

CPU Based on Am29116

Whether you use the Am29116 for

- controller applications
- or as a general purpose CPU

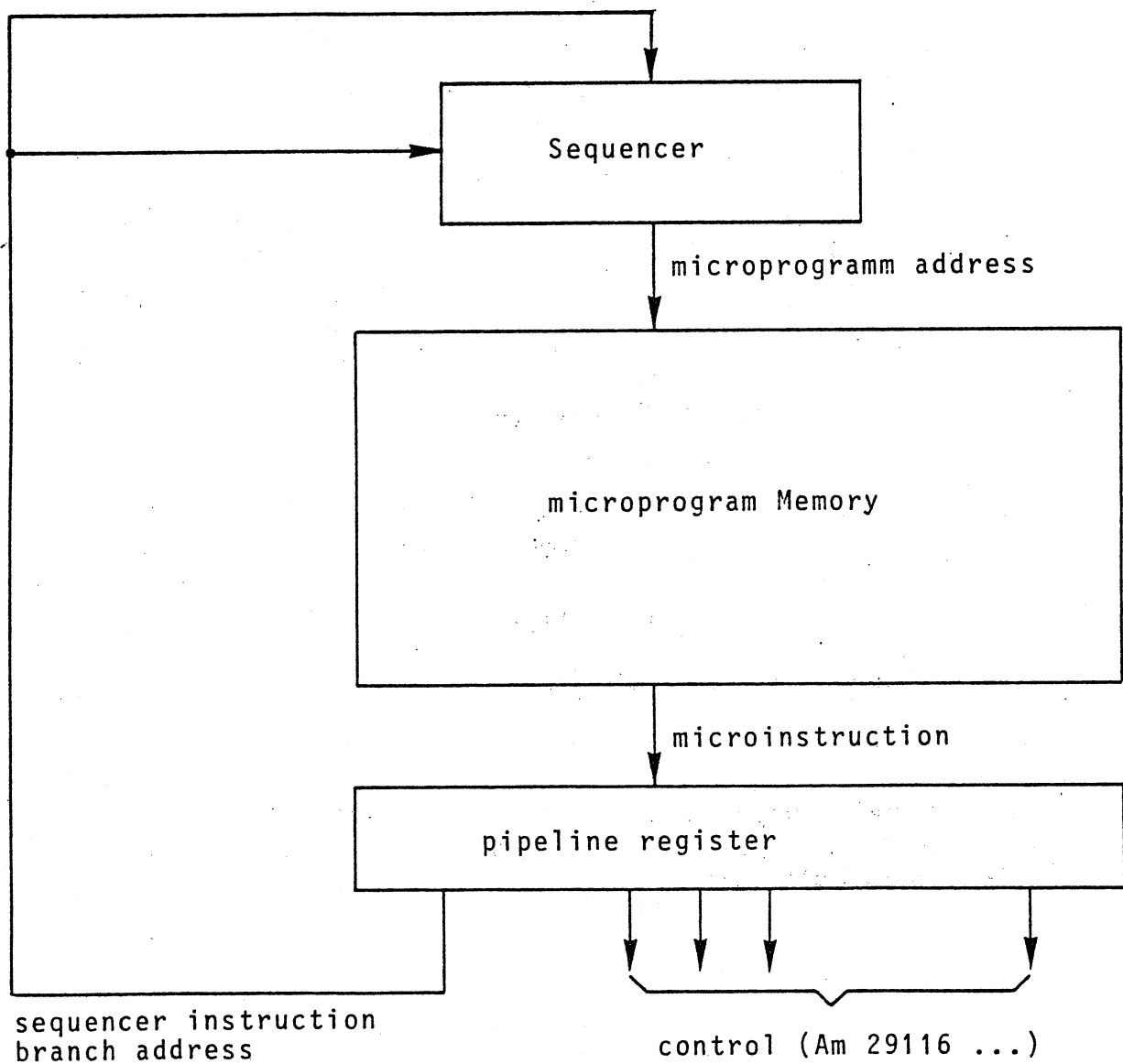
you have to design a microprogrammed computer control unit. (CCU)

Microroutines for controller applications may be different (longer, as algorithms are more complex) from microroutines for general purpose machine instructions. Also, starting addresses may be generated differently.

However, the overall structure of the control unit is the same in both cases:

- the "heart" of the control unit is always the microprogram store and a sequencer for the address generation
- depending on the special application, several subunits may be connected to it: status control, interrupt control, timing control, DMA

The Am2900 family of bipolar microprogrammed LSI devices gives you a number of choices for the design of powerful CCU structures.

Microprogram Memory and Sequencer

Sequencer Implementations

- Bitslice solution: use Am2909A/2911A 4-bit-slice sequencer, Am29811 next address control and Am29803 n-way branch devices for maximum flexibility
- Monolithic solution: use Am2910 for a highly integrated, single-chip sequencer
- In the near future, use Am29112 8-bit-slice interruptable sequencer for very-high-speed, highly-integrated control unit implementations

The following pages give a short introduction to the different devices. A more detailed presentation is given in ED2900A and ED2900B course.

Interfacing to the Am29116:

Let us take a short look at interrupt controller structures.

Am2914 Vectored Priority Interrupt Controller

Distinctive Characteristics:

- Accepts 8 interrupt inputs:
Interrupts may be pulses or levels
and are stored internally.
- Built-in mask register:
Six different operations
can be performed on the mask register.
- Built-in status register:
Status register holds code for lowest allowed interrupt
- Vectored output:
Output is binary code for highest priority
unmasked interrupt.
- Expandable:
Any number of Am2914's may be stacked
for large interrupt systems.
- Microprogrammable:
Executes 16 different microinstructions.
Instruction-enable pin aids in vertical microprogramming.
- High-speed operation:
Delay from an interrupt clocked into the interrupt
register to interrupt-request output is typically 60 ns.

Am2914 Vectored Priority Interrupt ControllerMicroinstruction Set

Microinstruction Description	Microinstruction Code 3 2 1 0
Master clear	0000
Clear all interrupts	0001
Clear interrupts from M-bus	0010
Clear interrupts from mask register	0011
Clear interrupt, last vector read	0100
Read vector	0101
Read status register	0110
Read mask register	0111
Set mask register	1000
Load status register	1001
Bit clear mask register	1010
Bit set mask register	1011
Clear mask register	1100
Disable interrupt request	1101
Load mask register	1110
Enable interrupt request	1111

Am2914 Vectored Priority Interrupt Controller (continued)

Application Suggestions:

For more than 8 interrupt inputs, use several Am2914's connected together. Am2902A carry-look-ahead and Am2913 priority interrupt expander devices are needed.

Implementation of software-level interrupts or firmware-level interrupts is possible. Use the vector mapping PROM output as macro- or micro-address. For more details, refer to ED2900A.

Address and Data Interfacing

- Am 2950, Am 2951 eight-bit bidirectional I/O ports with handshake.
- Am2940, Am2942 DMA address generators

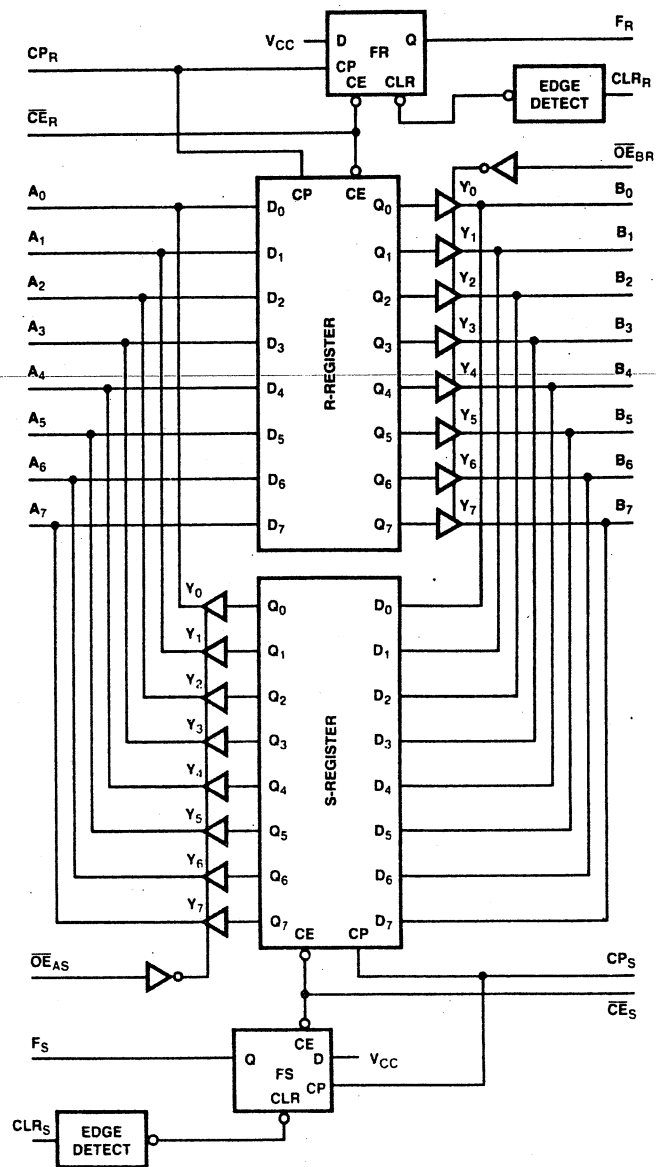
Am2950 / Am2951 Eight-bit Bidirectional I/O Ports with Handshake

Distinctive Characteristics:

- Eight-Bit, Bidirectional I/O Port with Handshake -
Two eight-bit, back-to-back registers store data moving in both directions between two bidirectional busses.
- Register Full/Empty Flags -
On-chip flag flip-flops provide data-transfer handshaking signals.
- Separate Clock, Clock Enable and Three-State Output Enable for Each Register.
- Separate, Edge-Sensitive Clear Control for Each Flag Flip-Flop.
- Inverting and Non-Inverting Versions-
The Am2950 provides non-inverting data outputs.
The Am2951 provides inverting data outputs.
- 24mA Output Current Sink Capability.

Am2950 / Am2951 Eight-bit Bidirectional I/O Ports with Handshake

Am2950 BLOCK DIAGRAM



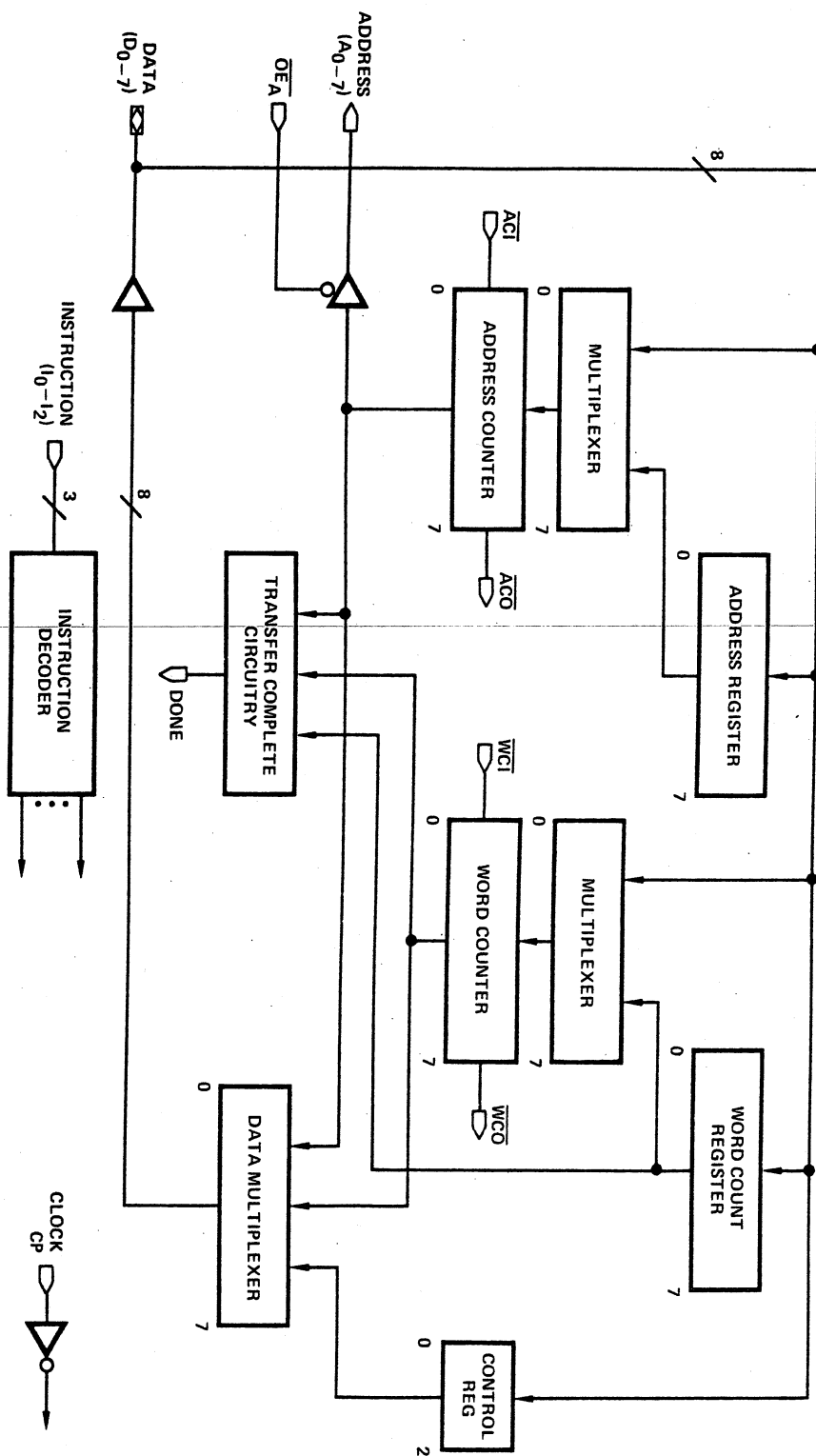
MPR-573

Am2940 DMA Address Generator

Distinctive Characteristics:

- **DMA Address Generation**
Generates memory address, word count and DONE signal for DMA transfer operation.
- **Expandable Eight-bit Slice**
Any number of Am2940's can be cascaded to form larger memory addresses - three devices address 16 megawords.
- **Repeat Data Transfer Capability**
Initial memory address and word count are saved so that the data transfer can be repeated.
- **Programmable Control Modes**
Provides four types of DMA transfer control plus memory-address increment/decrement.
- **High Speed, Bipolar LSI**
Advanced Low-Power Schottky TTL technology provides typical CLOCK to DONE propagation delay of 50ns and 24mA output current sink capability.
- **Microprogrammable**
Executes 8 different instructions.

Am2940 DMA Address Generator



Am2940 Control Modes

- Word count equals zero mode
- Word count compare mode
- Address compare mode
- Word counter carry out mode

Am2940 Instruction Definitions

I_2	I_1	I_0	Octal Code	Function	Mnemonic	Control Mode	Word Reg.	Word Counter	Address Reg.	Address Counter	Control Register	Data D_0 - D_7
L	L	L	0	WRITE CONTROL REGISTER	WRCR	0, 1, 2, 3	HOLD	HOLD	HOLD	HOLD	D_0 - D_2 →CR	INPUT
L	L	H	1	READ CONTROL REGISTER	RDCR	0, 1, 2, 3	HOLD	HOLD	HOLD	HOLD	HOLD	CR→ D_0 - D_2 (Note 1)
L	H	L	2	READ WORD COUNTER	RDWC	0, 1, 2, 3	HOLD	HOLD	HOLD	HOLD	HOLD	WC→D
L	H	H	3	READ ADDRESS COUNTER	RDAC	0, 1, 2, 3	HOLD	HOLD	HOLD	HOLD	HOLD	AC→D
H	L	L	4	REINITIALIZE COUNTERS	REIN	0, 2, 3	HOLD	WCR→WC	HOLD	AR→AC	HOLD	Z
						1	HOLD	ZERO→WC	HOLD	AR→AC	HOLD	Z
H	L	H	5	LOAD ADDRESS	LDAD	0, 1, 2, 3	HOLD	HOLD	D→AR	D→AC	HOLD	INPUT
H	H	L	6	LOAD WORD COUNT	LDWC	0, 2, 3	D→WR	D→WC	HOLD	HOLD	HOLD	INPUT
						1	D→WR	ZERO→WC	HOLD	HOLD	HOLD	INPUT
H	H	H	7	ENABLE COUNTERS	ENCT	0, 1, 3	HOLD	ENABLE COUNT	HOLD	ENABLE COUNT	HOLD	Z
						2	HOLD	HOLD	HOLD	ENABLE COUNT	HOLD	Z

CR = Control Reg.
AR = Address Reg.
AC = Address Counter

WCR = Word Count Reg.
WC = Word Counter
D = Data

L = LOW
H = HIGH
Z = High Impedance

Note 1:
Data Bits D_3 - D_7 are high during this instruction.

Am2942: A 22 pin version of the Am2940 useable as

- DMA address generator
- programmable timer/counter

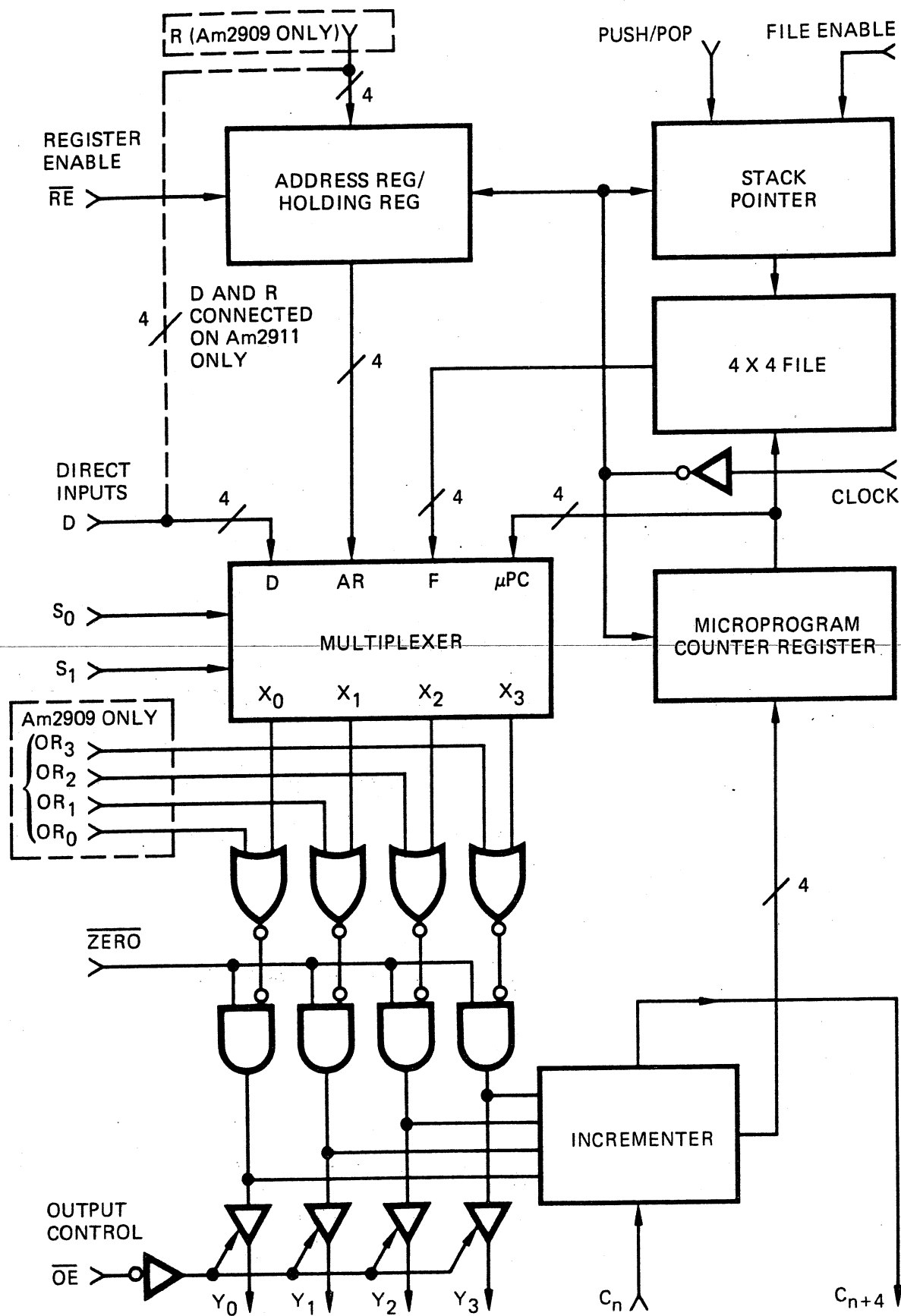
Bitslice Sequencer

Am2909A, Am2911A microprogram sequencers

Distinctive Characteristics

- 4-bit slice cascadable to any number of microwords
- Internal address register
- Branch input for N-way branches
- Cascadable 4-bit microprogram counter
- 4 x 4 file with stack pointer and push pop control for nesting microsubroutines
- Zero-input for returning to the microcode word at address zero
- Individual OR input for each bit for branching to higher microinstructions (Am2909 only)
- Three-state outputs
- All internal register change state on the LOW-to-HIGH transition of the clock
- Am2909 in 28-pin package
- Am2911 in 20-pin package
- New high-speed versions (Am2909A and Am2911A) are plug-in replacements for original Am2909 and Am2911. Critical path speeds will be improved by about 25%.

Am2909A and Am2911A Sequencers



Am29811A Next Address Control

Used to decode 4 instruction Bits (from the pipeline register) and to generate the control signals for Am2909/11 sequencers.

Distinctive Characteristics

- Next-address control unit for Am2909/11 sequencers
- 16 next address instructions
- Test-input for conditional instructions
- Separate outputs to control the Am2909/11, an independent event counter and a mapping-PROM/branch-address interface

Am29811 ANext-Address Instructions

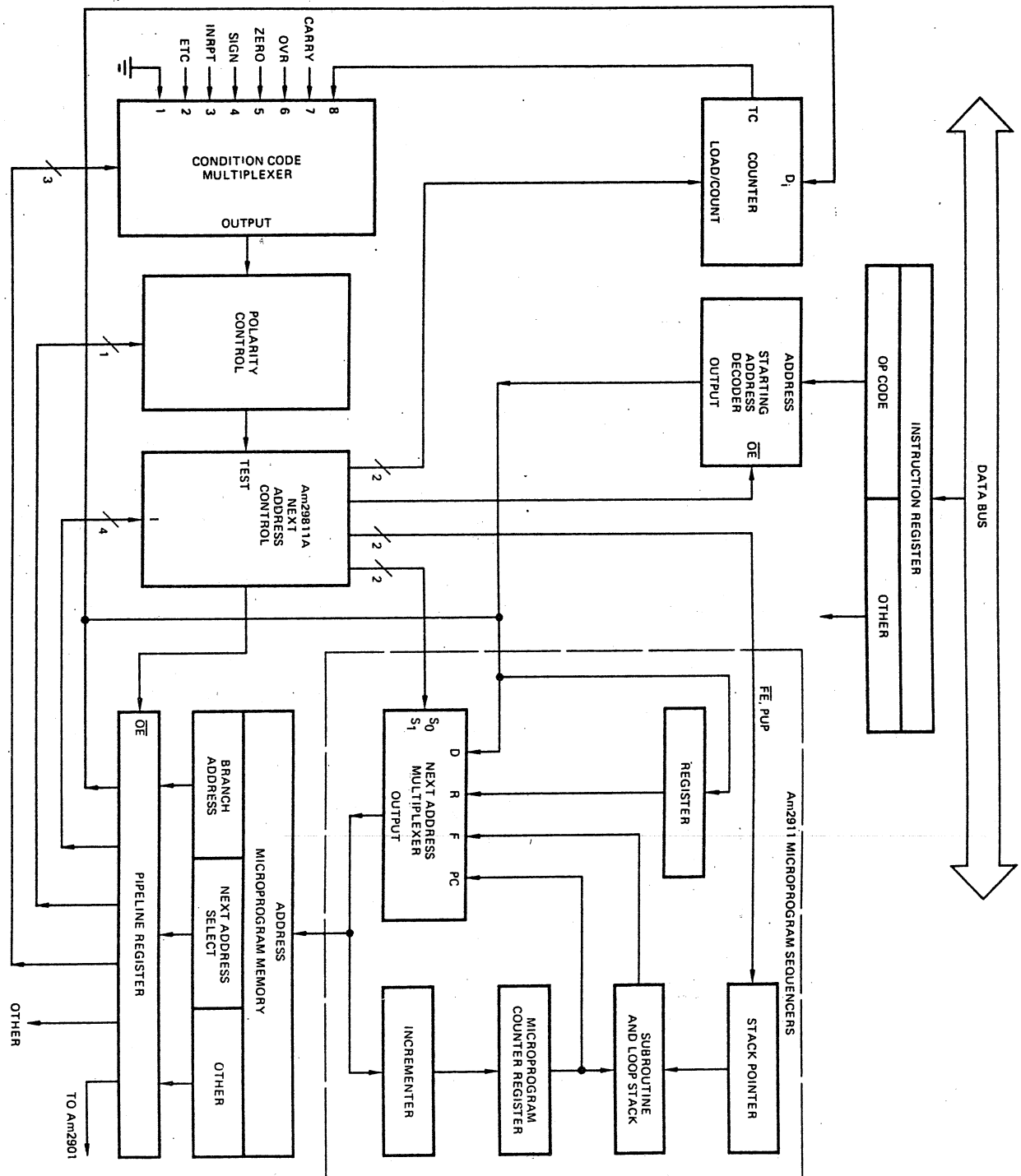
MNEMONIC	INPUTS			OUTPUTS				
	INSTRUCTION I ₃ I ₂ I ₁ I ₀	FUNCTION	TEST INPUT	NEXT ADDR SOURCE	FILE	COUNTER	MAP-E	PL-E
JZ	L L L L	JUMP ZERO	X	D	HOLD	LL*	H	L
CJS	L L L H	COND JSB PL	L	PC	HOLD	HOLD	H	L
			H	D	PUSH	HOLD	H	L
JMAP	L L H L	JUMP MAP	X	D	HOLD	HOLD	L	H
CJP	L L H H	COND JUMP PL	L	PC	HOLD	HOLD	H	L
			H	D	HOLD	HOLD	H	L
PUSH	L H L L	PUSH/COND LD CNTR	L	PC	PUSH	HOLD	H	L
			H	PC	PUSH	LOAD	H	L
JSRP	L H L H	COND JSB R/PL	L	R	PUSH	HOLD	H	L
			H	D	PUSH	HOLD	H	L
CJV	L H H L	COND JUMP VECTOR	L	PC	HOLD	HOLD	H	H
			H	D	HOLD	HOLD	H	H
JRP	L H H H	COND JUMP R/PL	L	R	HOLD	HOLD	H	L
			H	D	HOLD	HOLD	H	L
RFCT	H L L L	REPEAT LOOP, CNTR ≠ 0	L	F	HOLD	DEC	H	L*
			H	PC	POP	HOLD	H	L
RPCT	H L L H	REPEAT PL, CNTR ≠ 0	L	D	HOLD	DEC	H	L
			H	PC	HOLD	HOLD	H	L
CRTN	H L H L	COND RTN	L	PC	HOLD	HOLD	H	L
			H	F	POP	HOLD	H	L
CJPP	H L H H	COND JUMP PL & POP	L	PC	HOLD	HOLD	H	L
			H	D	POP	HOLD	H	L
LDCT	H H L L	LOAD CNTR & CONTINUE	X	PC	HOLD	LOAD	H	L
LOOP	H H L H	TEST END LOOP	L	F	HOLD	HOLD	H	L
			H	PC	POP	HOLD	H	L
CONT	H H H L	CONTINUE	X	PC	HOLD	HOLD	H	L
JP	H H H H	JUMP PL	X	D	HOLD	HOLD	H	L

L = LOW
H = HIGH
X = Don't Care

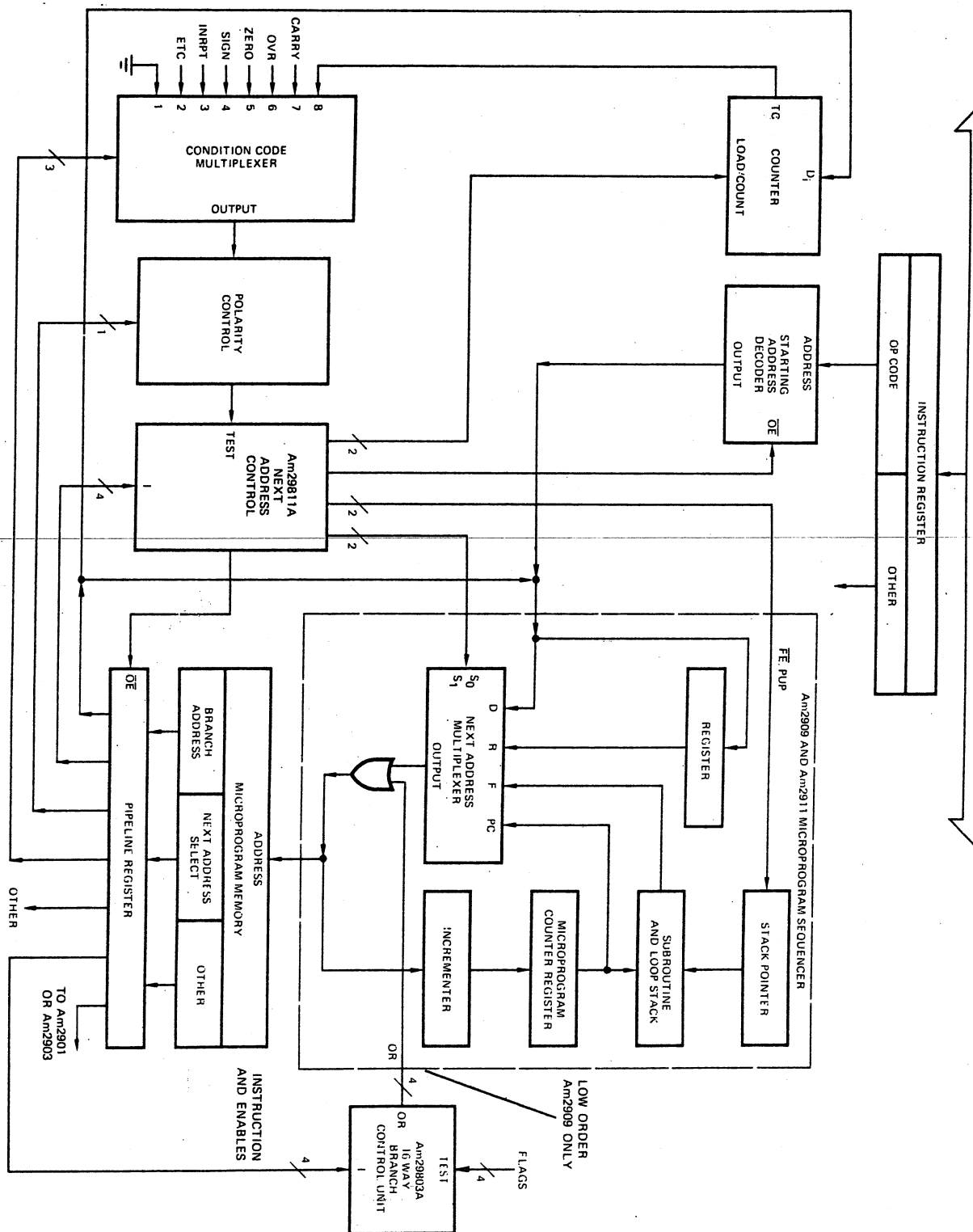
DEC = Decrement
*LL = Special Case

A Typical Computer Control Unit Using

Am2911 and Am29811A



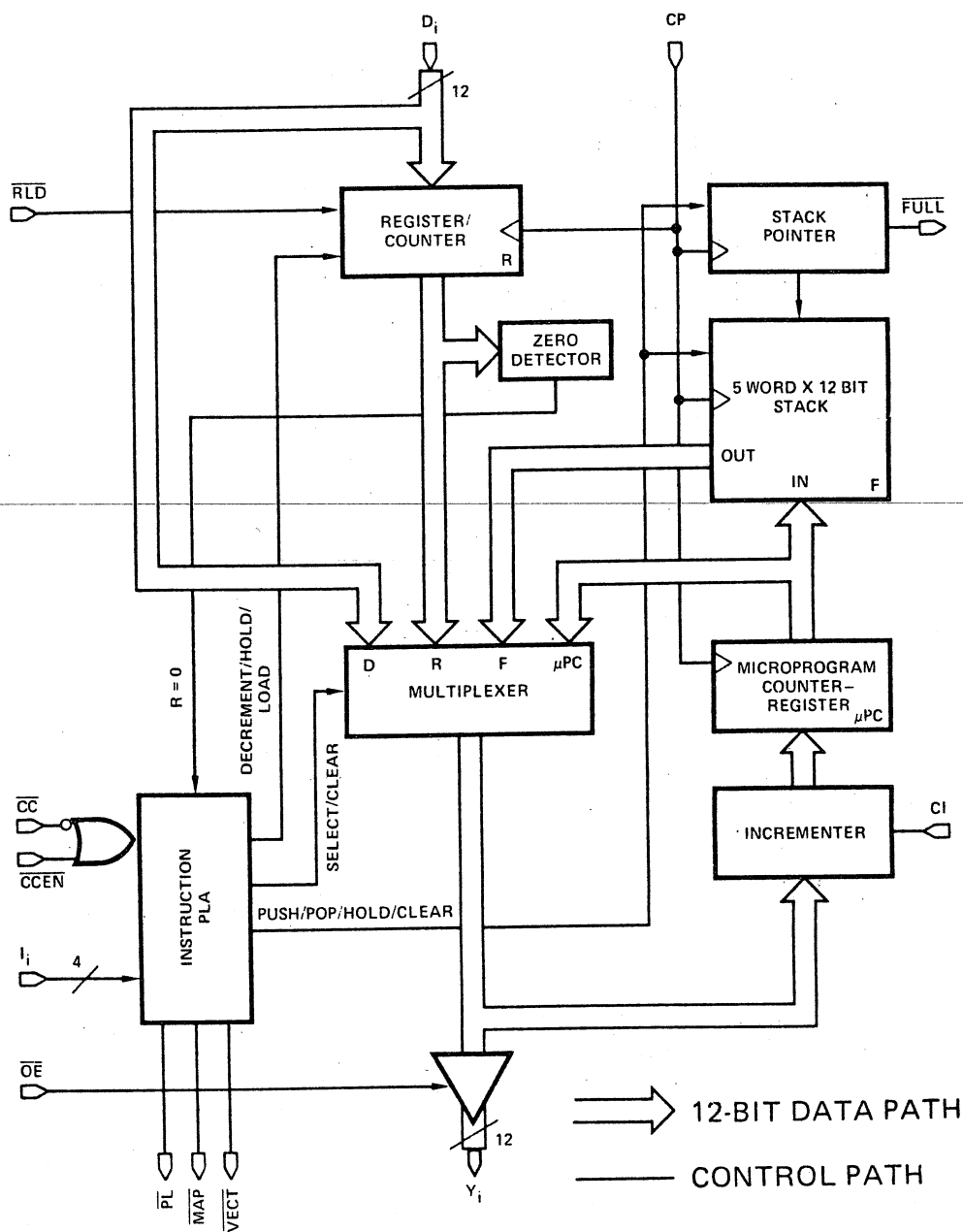
Am29803A 16-way Branch Control Unit



Am2910 Supersequencer

Distinctive Characteristics:

- Twelve Bits Wide. Fixed width -- not a slice.
Address up to 4096 words of microcode with one chip.
All internal elements are full 12 bits wide.
- Internal Loop Counter
Pre-settable 12-bit down-counter for repeating instructions and counting Loop iterations
- Four Address Sources
Microprogram Address may be selected from microprogram counter, branch address bus, 5-level push/pop stack, or internal holding register
- Sixteen Powerful Microinstructions
Executes 16 sequence control instructions, most of which are conditional on external condition input, state of internal loop counter, or both
- Output-Enable Controls for Three Branch Address Sources
Built-in decoder function to enable external devices onto the branch-address bus. Eliminates external decoder
- All Registers Positive Edge-triggered
Simplifies timing problems. Eliminates long set-up times
- Fast Control from Condition Input
Delay from condition code input to address output only
21ns typical

Am2910 Supersequencer

Am2910 Supersequencer

Next-Address Instructions

13-10	MNEMONIC	NAME	REG/ CNTR CON- TENTS	FAIL CCEN = LOW and CC = HIGH		PASS CCEN = HIGH or CC = LOW		REG/ CNTR	ENABLE
				Y	STACK	Y	STACK		
0	JZ	JUMP ZERO	X	0	CLEAR	0	CLEAR	HOLD	PL
1	CJS	COND JSB PL	X	PC	HOLD	D	PUSH	HOLD	PL
2	JMAP	JUMP MAP	X	D	HOLD	D	HOLD	HOLD	MAP
3	CJP	COND JUMP PL	X	PC	HOLD	D	HOLD	HOLD	PL
4	PUSH	PUSH/COND LD CNTR	X	PC	PUSH	PC	PUSH	Note 1	PL
5	JSRP	COND JSB R/PL	X	R	PUSH	D	PUSH	HOLD	PL
6	CJV	COND JUMP VECTOR	X	PC	HOLD	D	HOLD	HOLD	VECT
7	JRP	COND JUMP R/PL	X	R	HOLD	D	HOLD	HOLD	PL
8	RFCT	REPEAT LOOP, CNTR ≠ 0	≠ 0	F	HOLD	F	HOLD	DEC	PL
			= 0	PC	POP	PC	POP	HOLD	PL
9	RPCT	REPEAT PL, CNTR ≠ 0	≠ 0	D	HOLD	D	HOLD	DEC	PL
			= 0	PC	HOLD	PC	HOLD	HOLD	PL
10	CRTN	COND RTN	X	PC	HOLD	F	POP	HOLD	PL
11	CJPP	COND JUMP PL & POP	X	PC	HOLD	D	POP	HOLD	PL
12	LDCT	LD CNTR & CONTINUE	X	PC	HOLD	PC	HOLD	LOAD	PL
13	LOOP	TEST END LOOP	X	F	HOLD	PC	POP	HOLD	PL
14	CONT	CONTINUE	X	PC	HOLD	PC	HOLD	HOLD	PL
15	TWB	THREE-WAY BRANCH	≠ 0	F	HOLD	PC	POP	DEC	PL
			= 0	D	POP	PC	POP	HOLD	PL

Note 1: If \overline{CCEN} = LOW and \overline{CC} = HIGH, hold; else load. X = Don't Care

ADVANCE INFORMATION

Am29112 Interruptable 8-bit Microprogram Sequencer

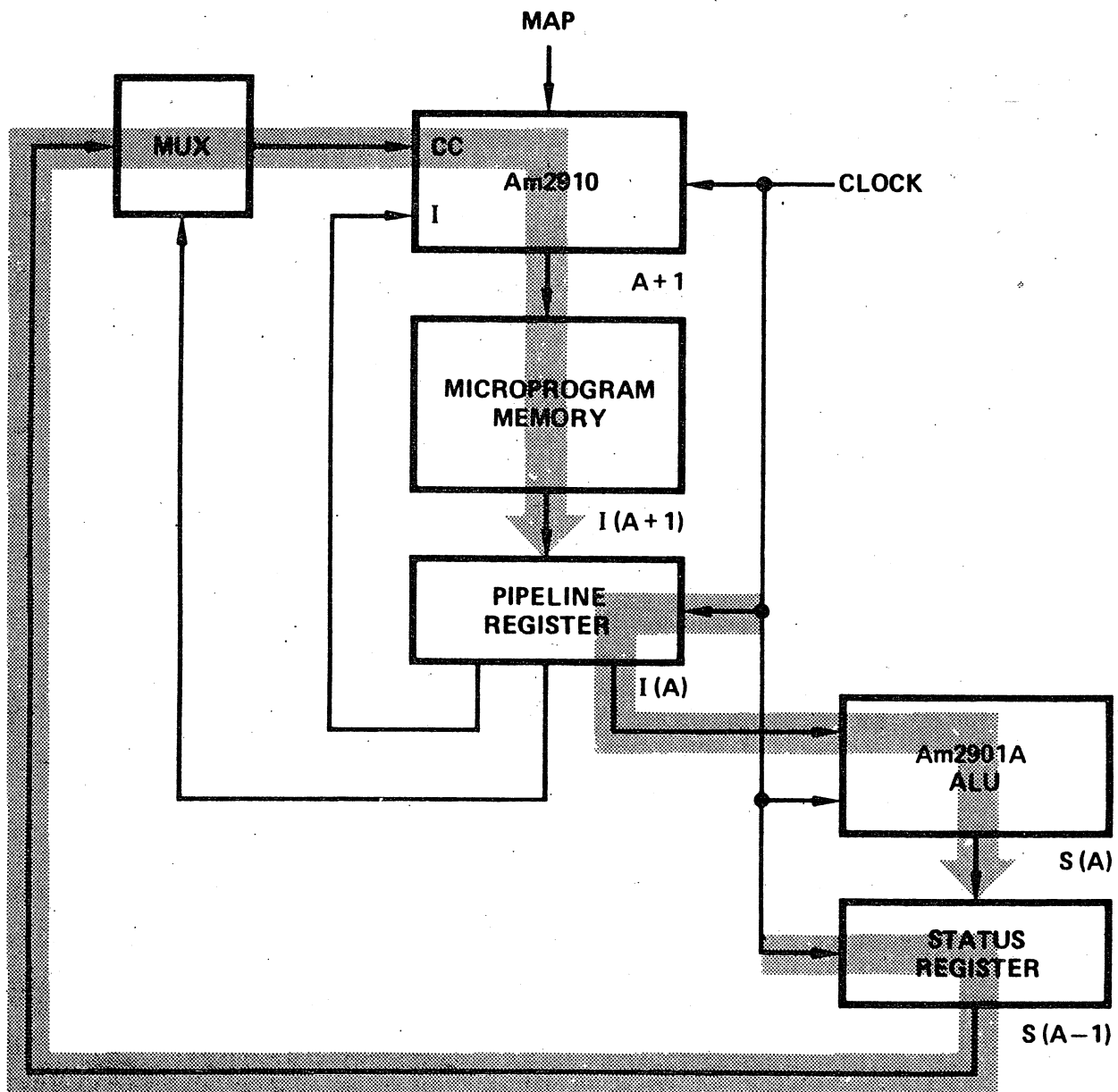
Distinctive Characteristics:

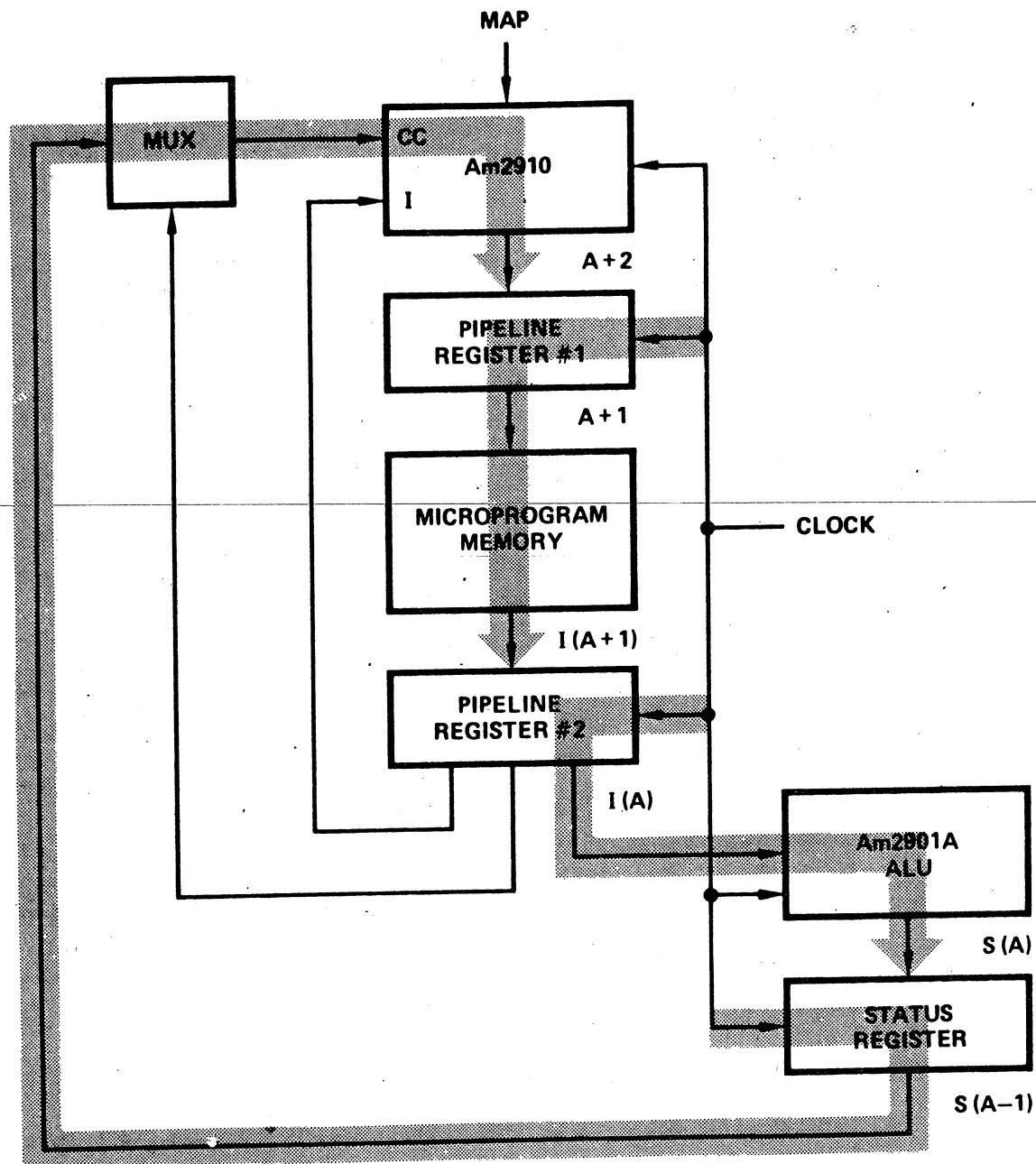
- Fast:
designed to operate in 10 MHz microprogrammed systems
- Expandable:
A single Am29112 is 8 bits wide and addresses 256 words of microprogram memory. Two Am29112's may be cascaded to directly address up to 64K of microprogram memory
- Deep stack:
A 33 deep on-chip stack is used for subroutine linkage, interrupt handling and loop control
- Interruptable at the microprogram level:
Two kinds of interrupts: maskable and unmaskable
- Powerful loop control:
Features an 8-bit counter for loop control. When two Am29112's are cascaded, the counters can act as a single 16-bit counter or as two independent 8-bit counters.
- Powerful addressing modes:
Features direct, multiway, multiway relative and program-counter-relative addressing
- Support for writable control store
- Hold feature:
A hold pin facilitates multiple-sequencer systems
- 48-pin Hermetic dual-in-line package

Some Speed Considerations...

- pipelining on the microinstruction level reduces microcycle length. A one level pipeline is recommended. A two level pipeline may be faster but complicates microprogramming. (conditional branching may require you to flush the pipeline -- refer to ED2900B)
- Parallelism on the microinstruction level is an important feature for high speed systems: the more hardware you control within one microinstruction by different microoperations, the more processing will be done within one cycle. Wide microinstructions and redundant microprogram memory may be the price to pay. Encoding and bit-steering tends to sequentialize your microprograms
- Variable cycle length may also contribute to the speed. Use the Am2925 clock generator for cycle length control.

(Pipelining and parallelism may also be used on the macroinstruction level.)

Single-level Pipelined System

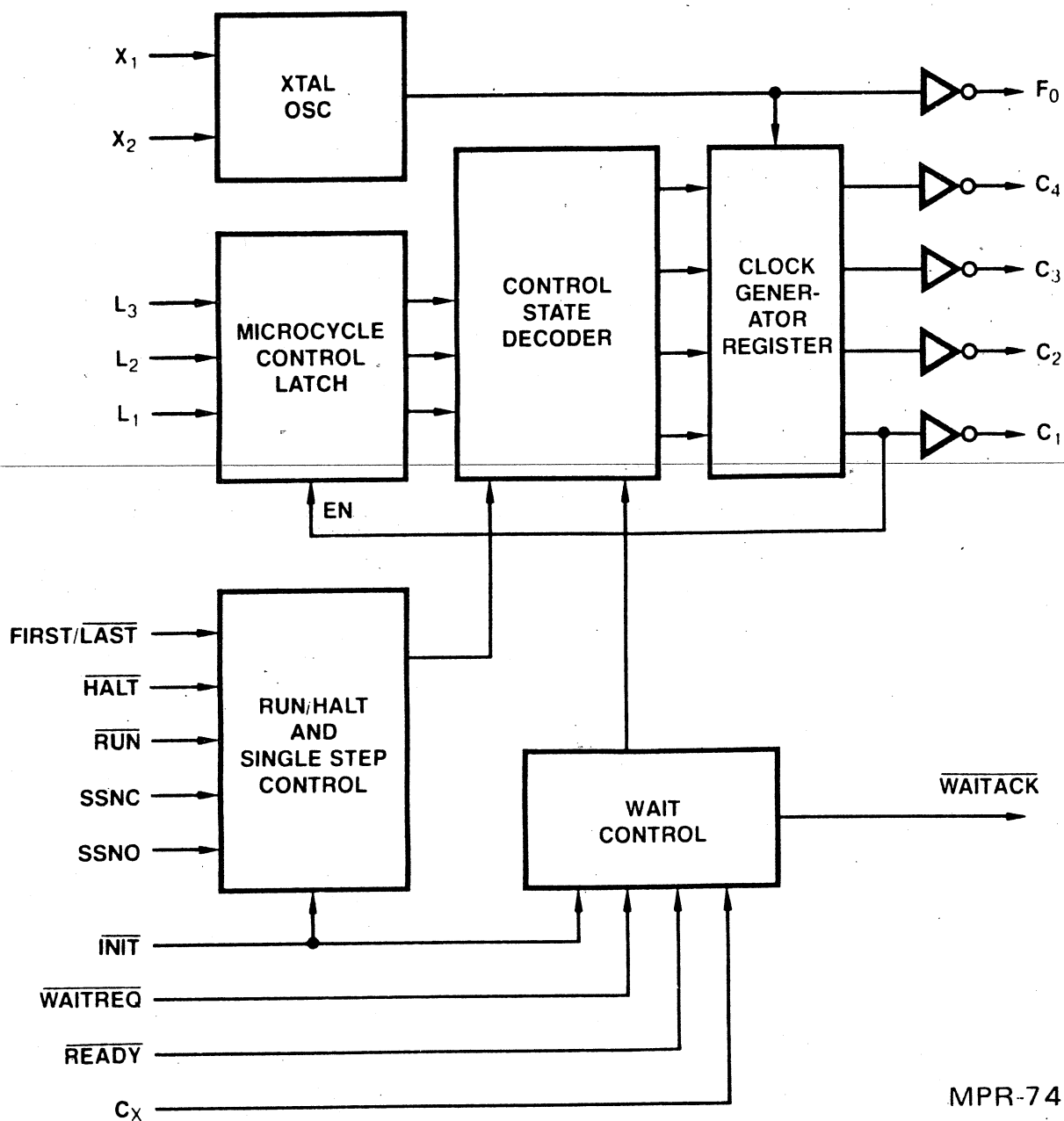
Two-level Pipelined System

MPR-485

Am2925 Clock Generator and Microcycle Length Controller

Distinctive Characteristics:

- Crystal controlled oscillator
Stable operation from 1 MHz to over 31 MHz
- Four microcode controlled clock outputs
Allows clock cycle length control for 15-30% increase in system throughput. Microcode selects one of eight clock patterns from 3 to 10 oscillator cycles in length
- System controls for Run/Halt and Single Step:
Switch debounced inputs provide flexible halt controls
- Slim 0.3" 24-pin package
LSI complexity in minimum board area

Am2925 Clock Generator and Microcycle Length Controller

MPR-743

