THE Am29203 EVALUATION BOARD

EXERCISES & LABORATORIES

ED2900A

EVALUATION BOARD EXERCISES/LABORATORIES

ED2900A/B

OVERVIEW OF THE Am29203 EVALUATION BOARD FOR USE IN ED2900A/B LABORATORIES

I. The Am29203 Evaluation Board Characteristics

- Stand-alone evaluation board for AMD bit-slice components
- Requires only power supply and CRT terminal
- Architecture represents typical 16-bit computer
- Built in Monitor allows:
  - Loading and displaying of writeable control store
  - Loading and displaying of macro memory
  - Loading and displaying of ALU registers
  - Loading and displaying of pipeline registers
  - Loading and displaying of the macro IR
  - Loading and displaying the Am2904 status registers
  - Setting breakpoints to control execution
  - Starting execution at any microaddress
  - Running built-in test routines
- Demonstrates microprogramming of the Am2900 family:
  - Am2910 sequencer
  - Am29203 arithmetic/logic unit (ALU)
  - Am2904 status and shift control unit

II. Architecture

- See Figure EB-1
- Actually two systems
  - Two Am2910 sequencers
  - Two control stores and pipeline registers
  - One shared 16-bit Am29203-based ALU
- The Monitor controls the board operation
  - Transparent to the user
  - Allows interface to the controlling CRT
  - Executes the Monitor program (in microcode)
  - Controls execution of the Primary System
  - Allows examining and changing the Primary System state
- The Primary System is the system we will examine
  - Standard 16-bit architecture
  - Features Am29203, Am2910 and Am2904

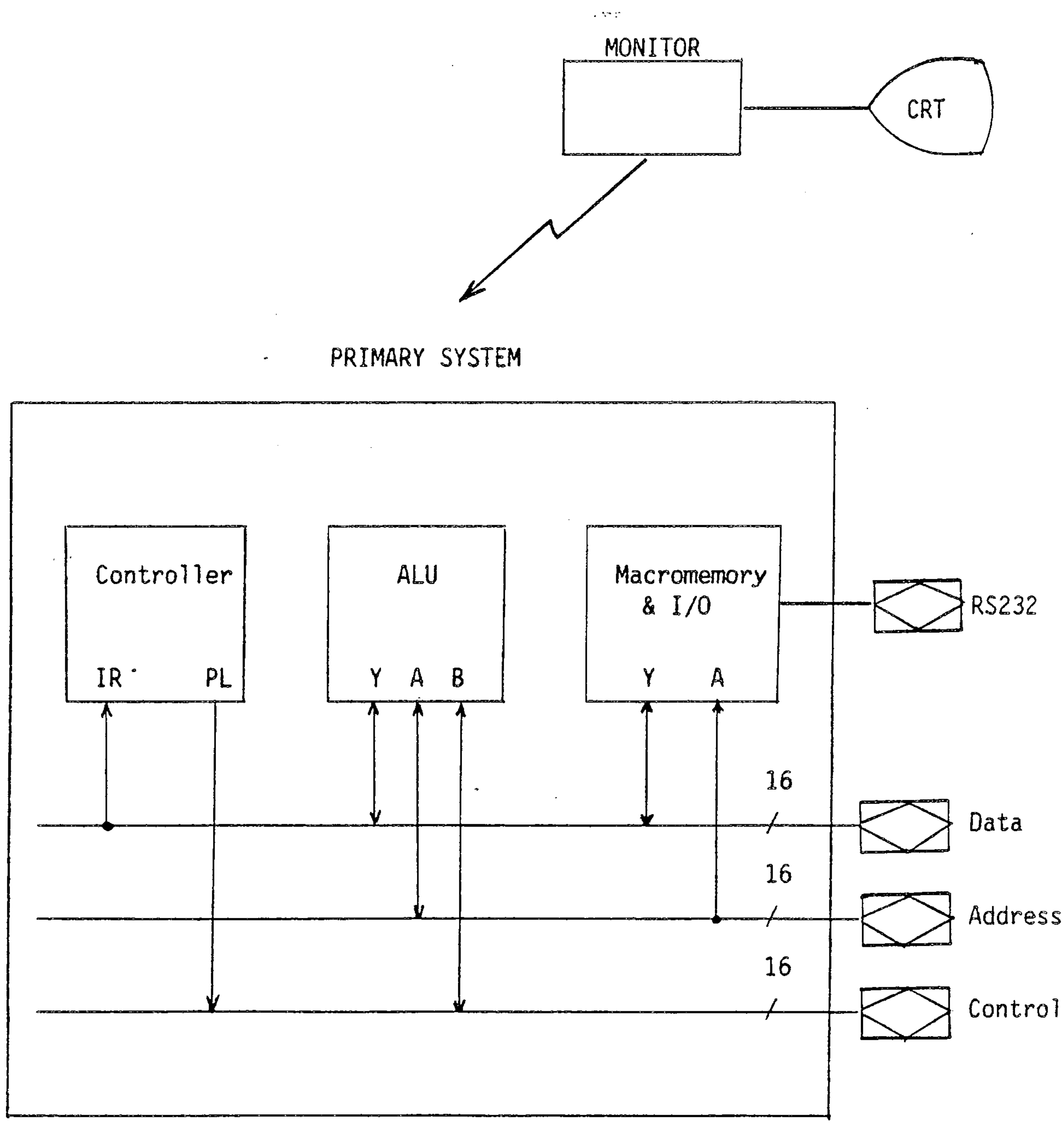


Figure EB-1. Evaluation Board Overview

III. The Primary System Architecture

- See Figure EB-2

- CCU
  - Uses 8-bit opcode through mapping PROM
  - Uses Am2910 sequencer
  - Has 1K x 48 bits of writeable control store
  - Has separate pipeline register
  - Uses decoding on some pipeline fields
  - Condition codes come from the Am2904 test mux
  - Am2910 $\overline{\text{CCEN}}$ is controllable from pipeline for forced pass

- ALU
  - Am29203
  - Uses Am2904 for shift linkage and carry-in MUX
  - A & B addresses come from IR or pipeline
  - Addresses 1K on board RAM via A-bus
  - Data transfers to/from RAM via Y-bus
  - Receives constants from pipeline via B-bus

- I/O
  - Second I/O UART is connected to the Y-bus
  - Uses memory-addressed I/O
  - Can drive a CRT, printer or similar devices

- Miscellaneous Features
  - Many signals available at connectors for expansion
  - Macro instruction set can be downloaded into WCS

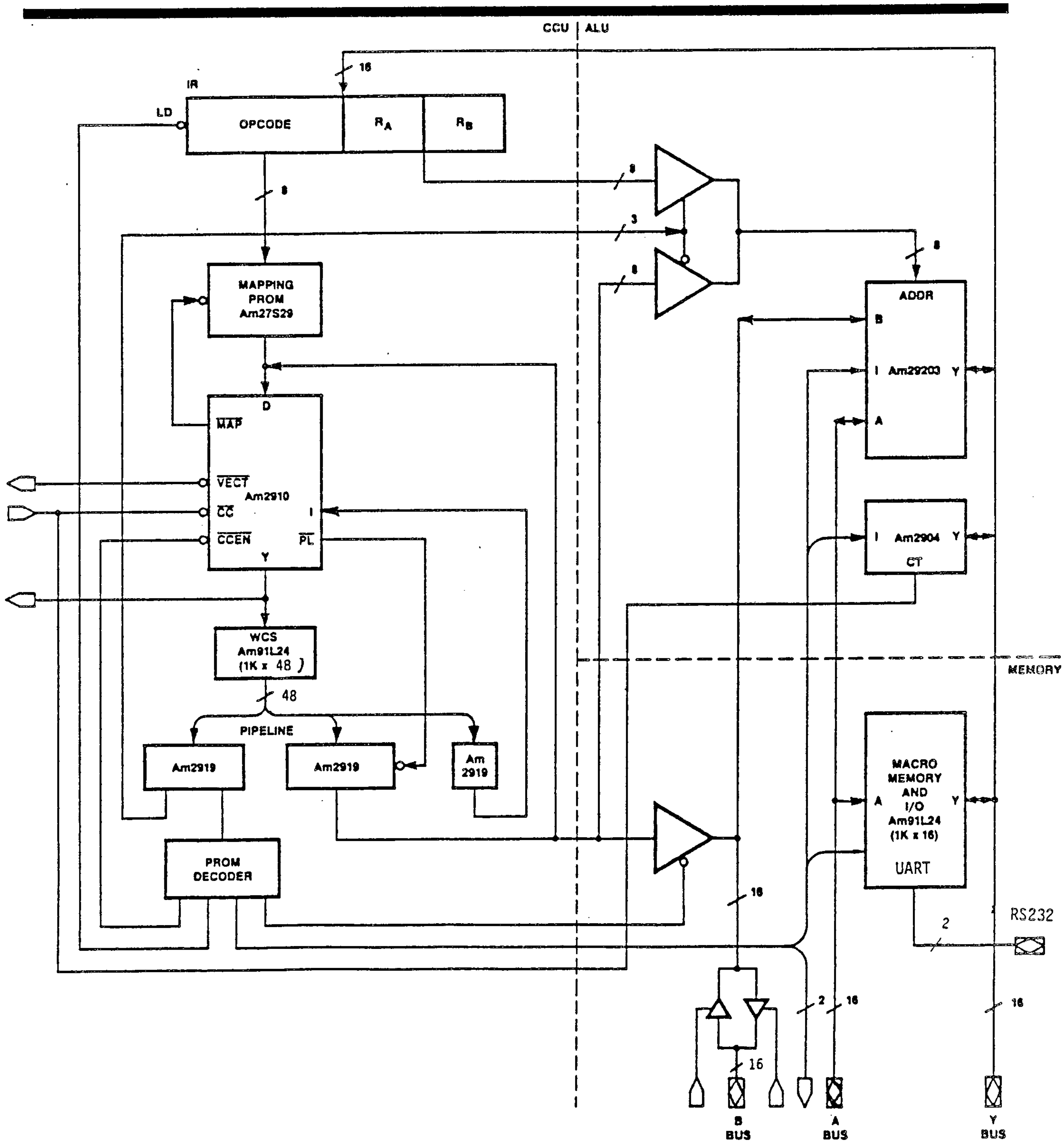


Figure EB-2. Primary-System Architecture

IV. The Primary System Microword

- 48-bits wide
- Bits 31-16 control the Am2904
  - Bits 19-16 are three overlaid fields
  - Bit 21 selects command field
  - Command field decoded by PROM
- Bit 15 is breakpoint bit (set and cleared by Monitor)
- Bits 13-4 make up the branch address field
  - Can be turned off by $\overline{PL}$ on Am2910
  - Bits 11-4 are three overlaid fields
    - -- Lowest 8 bits of branch address
    - -- ALU register addresses
    - -- Constant value for ALU

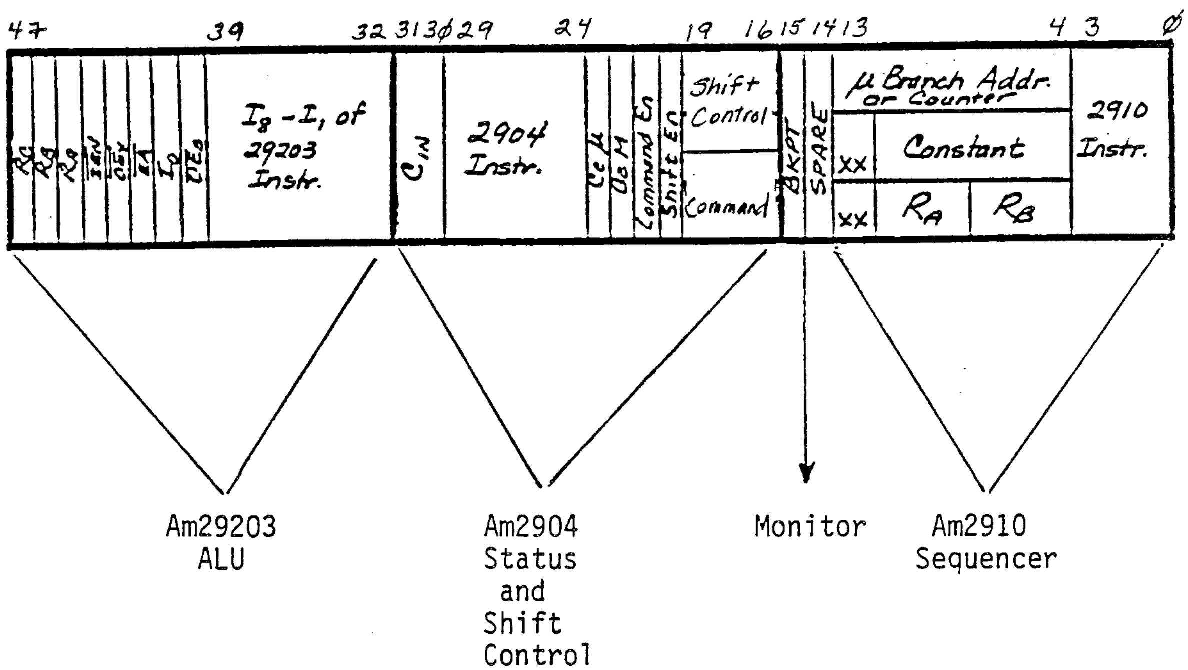


Figure EB-3.

V. The Primary System CCU Architecture

- See Figure EB-4
- 8-bits of machine level opcode
- 10-bits of control-store addressing
- Two Am2910 D-input sources controlled by Am2910 $\overline{\text{OE}}$ lines
  - Routine starting address from mapping PROM
  - Branch address from pipeline register
- All Am2910 instructions available
- Vector-map-enable not used on this board
- Condition-code input comes from Am2904
- $\overline{\text{CCEN}}$ controlled from pipeline for forced pass
- 1K x 48-bit Writeable Control Store (WCS)
- Separate pipeline register has tri-state section
- A PROM is used to decode a command field of the microinstruction

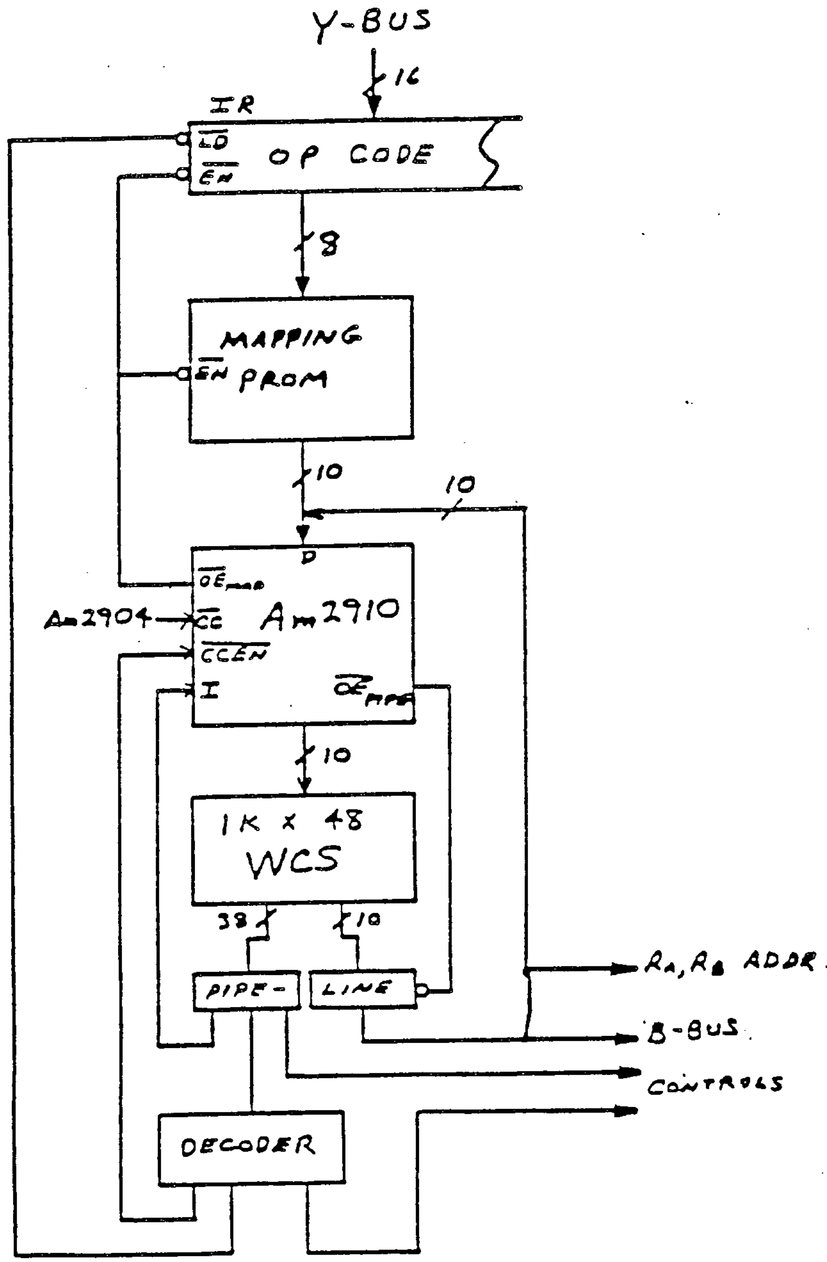


Figure EB-4. Primary-System CCU Architecture

Primary System Sequencer

- Am2910 sequencer is used. See Figure EB-5.
- 10 of the 12 address lines are used
- $\overline{RLD}$ tied high (unused)
- $\overline{CC}$ driven from Am2904 (condition code MUX and test)
- $\overline{CCEN}$ driven from pipeline through decoding PROM
- Next-address instructions come from pipeline
- $\overline{OE}$ controlled by Monitor
- $\overline{PL}$ controls tri-state section of pipeline
- $\overline{MAP}$ controls output of mapping PROM
- $\overline{VECT}$ not used
- CI (carry-in) tied high (normal operation)
- $\overline{FULL}$ not used

Am2910 BLOCK DIAGRAM

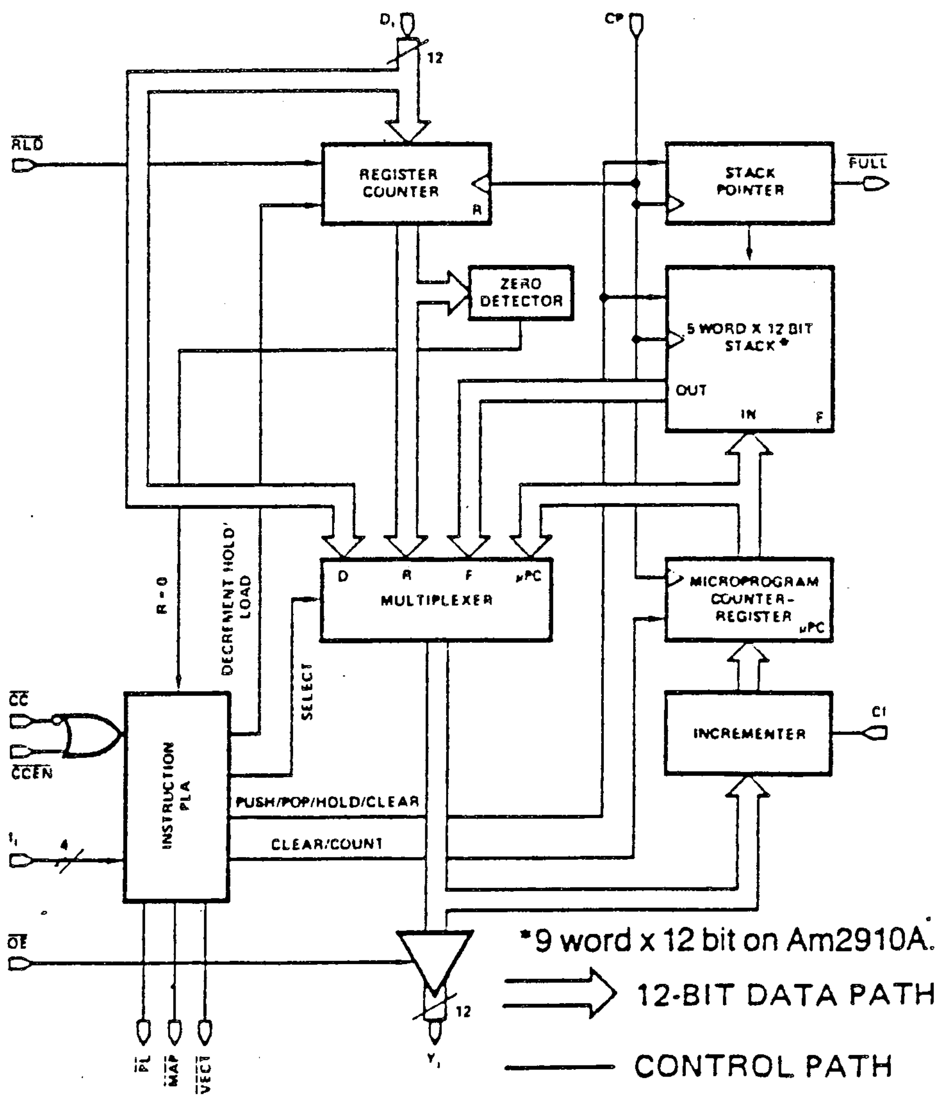


Figure EB-5. Am2910 Structure

Primary System Sequencer - Cont'd

- All 16 Am2910 instructions are usable with some restrictions. See Figure EB-6.
- CJV will conditionally jump to address 3FF
- $\overline{\text{CCEN}}$ must be used for forced pass
- No forced-fail condition is available (i.e. you cannot do a PUSH without loading the counter)
- Counter field is only 10-bits wide (max count = 1023)

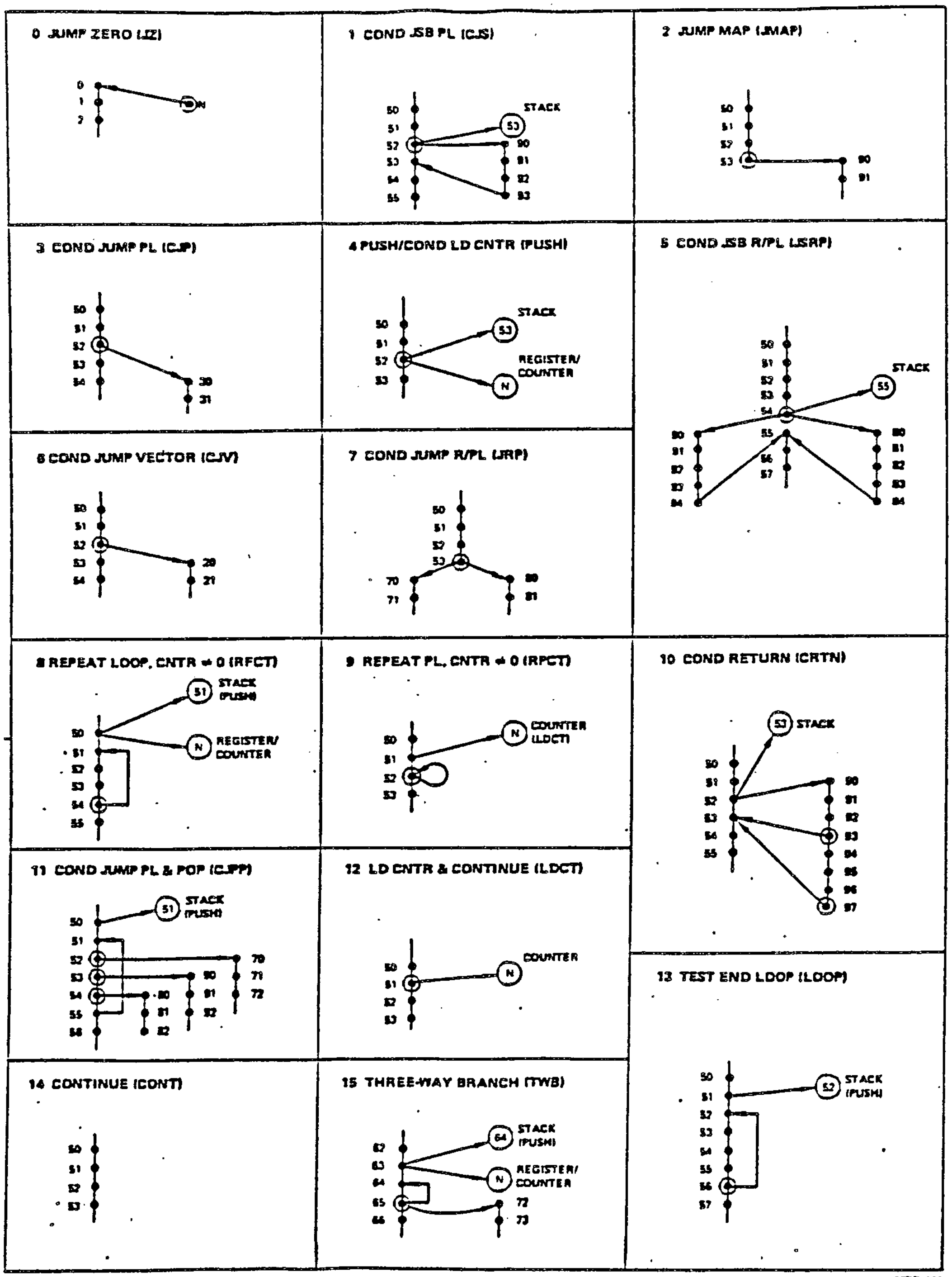


Figure EB-6. Am2910 Instruction Set

VI. Primary System Pipeline Registers

- Only the Branch Address/Count field is tri-stated:
  - Tri-state enable comes from Am2910 $\overline{\text{PL}}$
- Am2910 instruction field always enabled
- $\overline{\text{CCEN}}$ comes from decoding PROM
- $\overline{\text{LDIR}}$ (load the instruction register) comes from decoding PROM
- We will discuss the microword details shortly

VII. Primary System Mapping PROM

(see Figure EB-7)

A. Design Approach

- Maps opcodes to WCS addresses
- Based on a compromise 8-bit PROM approach
  - Uses only one chip
  - Maps to every words (even addresses) in WCS
- 8-bit opcode mapped to 10-bit microcode address -- achieved by two constraints:
  - All output addresses are even. The LSB of the address is tied low.
  - MSB of opcode is tied to MSB of output address so that 128 opcodes with MSB=Ø map to any of 256 even addresses <512 & 128 opcodes with MSB=1 map to any of 256 even addresses >=512

B. Mapping PROM Layout

- Upper 512 words of WCS are loaded with example microcode
  - All op codes start with 1
  - Automatically loaded on reset
  - Manually loaded using LI
  - Supports example macro instruction set
  - Op codes mapped to fit microroutines

- Lower 512 words intended for user routines
  - All op codes start with Ø
  - All 512 locations available
  - Op codes mapped to evenly spaced addresses
  - Four microwords are available for each op code
  - Larger routines invalidate the next op code(s)

- The user can replace the PROM to provide other mappings

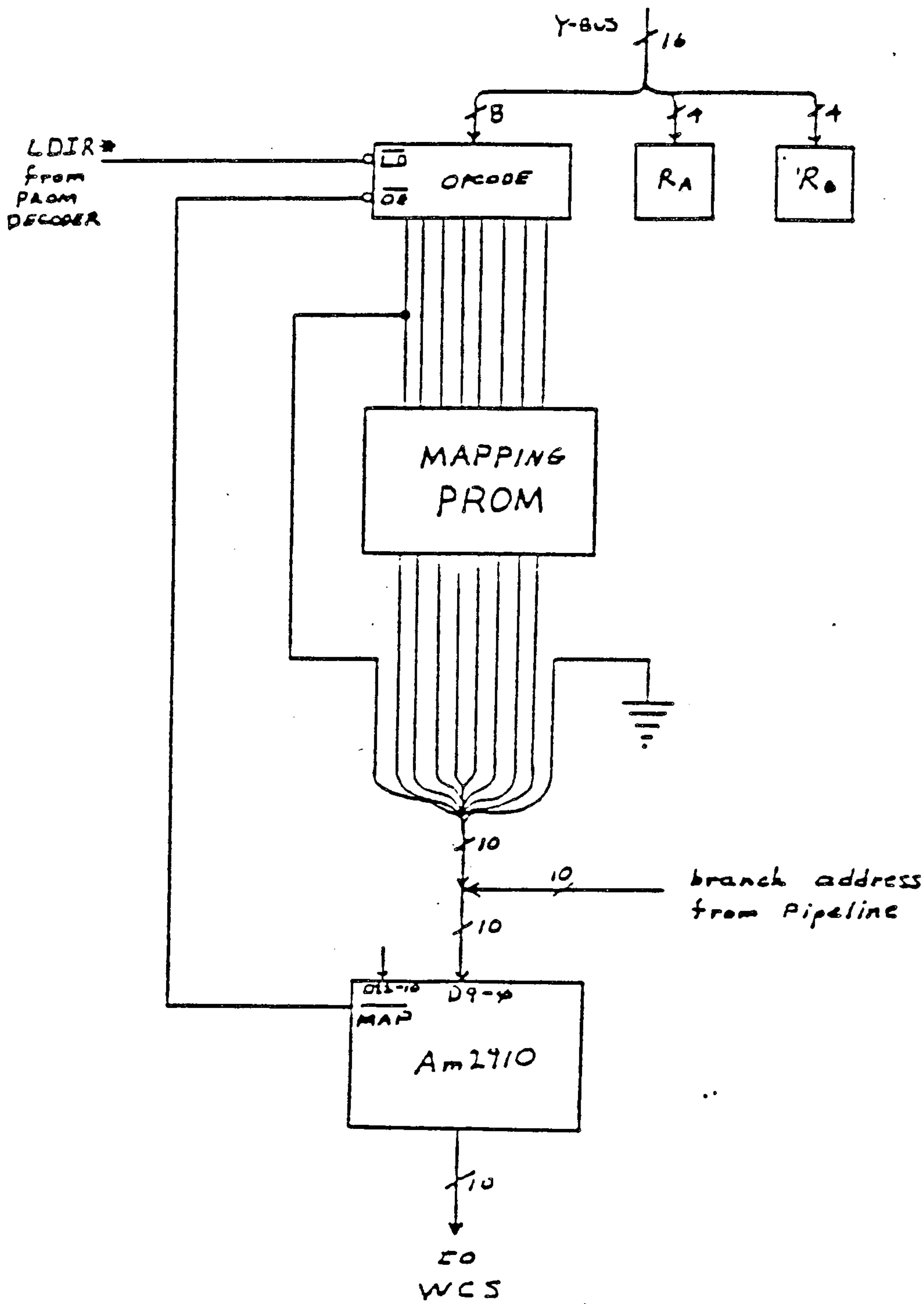


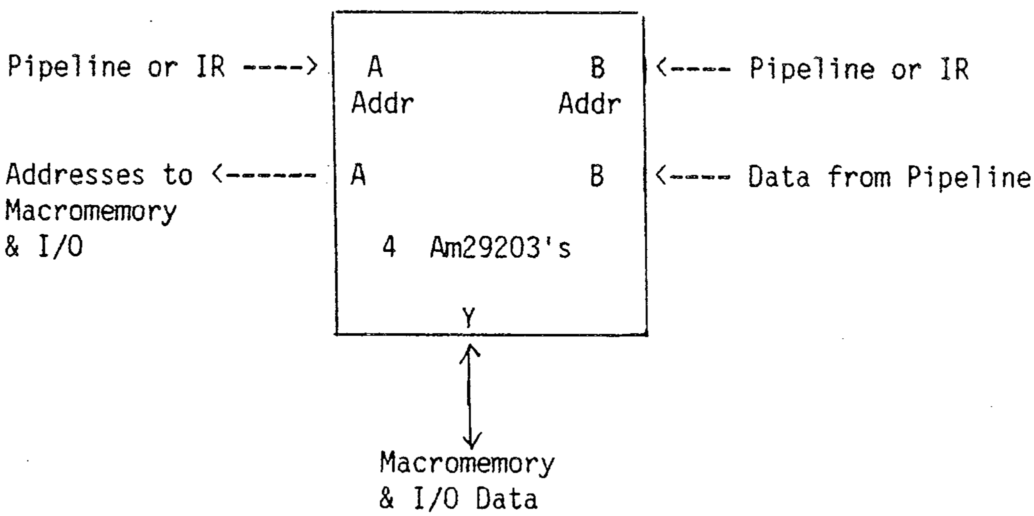
Figure EB-7. Mapping PROM Configuration

VIII. Primary System Writeable Control Store (WCS)

- 1K x 48-bit RAM used for storing microcode
- Loaded under Monitor control
- Upper 512 words loaded with example microcode
  - Loaded at board initialization
  - Can be overwritten by user routines
- In production environment, usually ROMs, PROMs, or Registered PROMs are used

IX. The Primary System ALU Architecture

- Four Am29203s connected in ripple-carry architecture. See Figure EB-8.
- Y-bus is the primary data transfer bus
- A-bus is used for addressing memory and I/O
  - No explicit MAR is used
- B-bus is a data input from the pipeline
- A, B addresses can come from the IR or pipeline
- Am2904 provides all ALU support
  - Carry-in MUX
  - RAM and Q shifter MUXs
  - Micro and Macro status registers
  - Condition-code MUX and test selection
- Y-bus allows Am2904 contents to be saved in memory



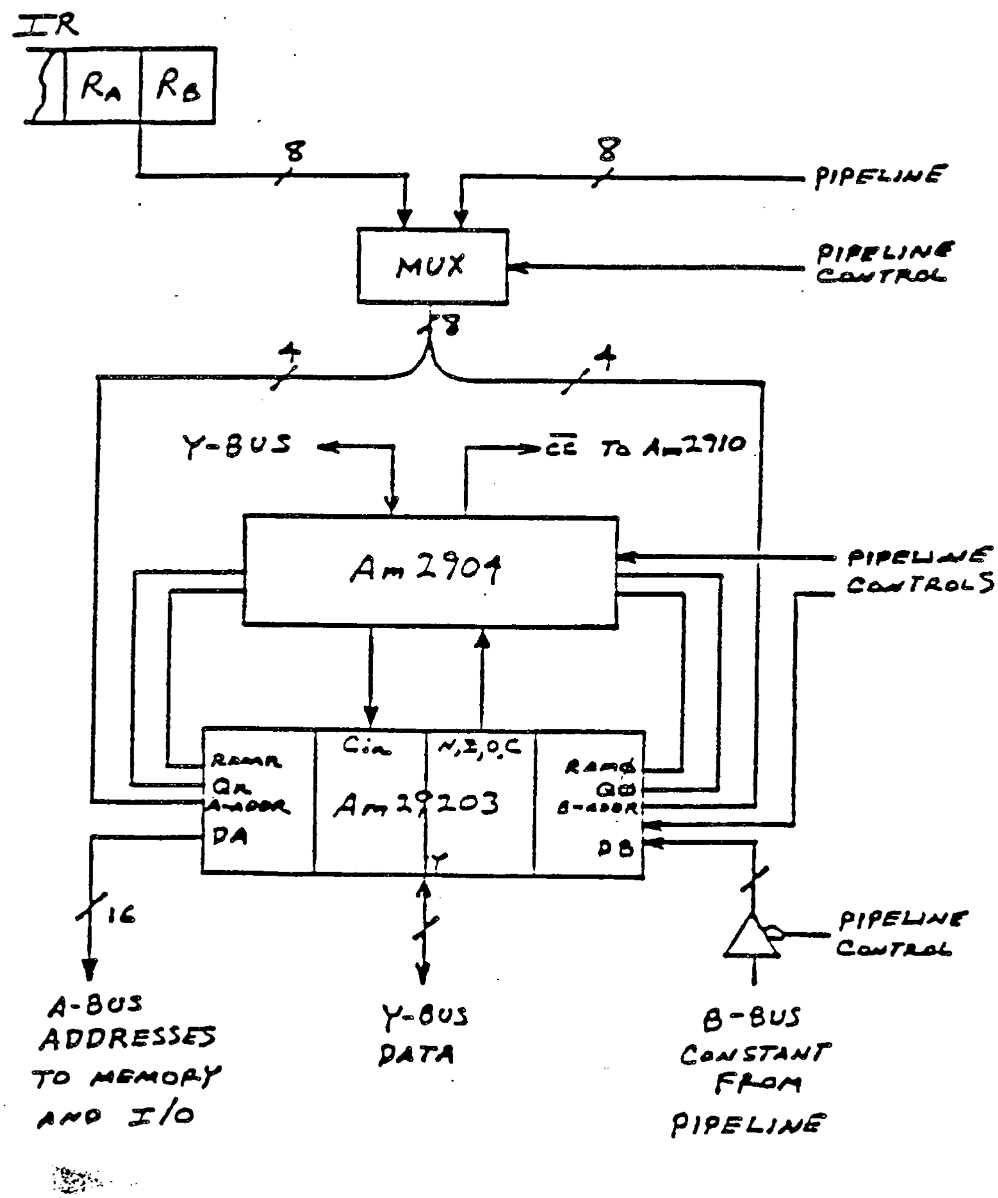


Figure EB-8. Primary-System ALU Architecture

X. The Primary System Memory and I/O

- All addressing via the A-bus
- I/O addressed just like memory
- Memory enable and read/write control from decoding PROM
- Address space divided
  - 1K x 16-bit RAM
  - 4K x 16-bit PROM for microcode examples (downloaded to WCS on Monitor command)
  - Two I/O addresses for second UART
  - 32K for offboard memory
- All data transfers via the Y-bus

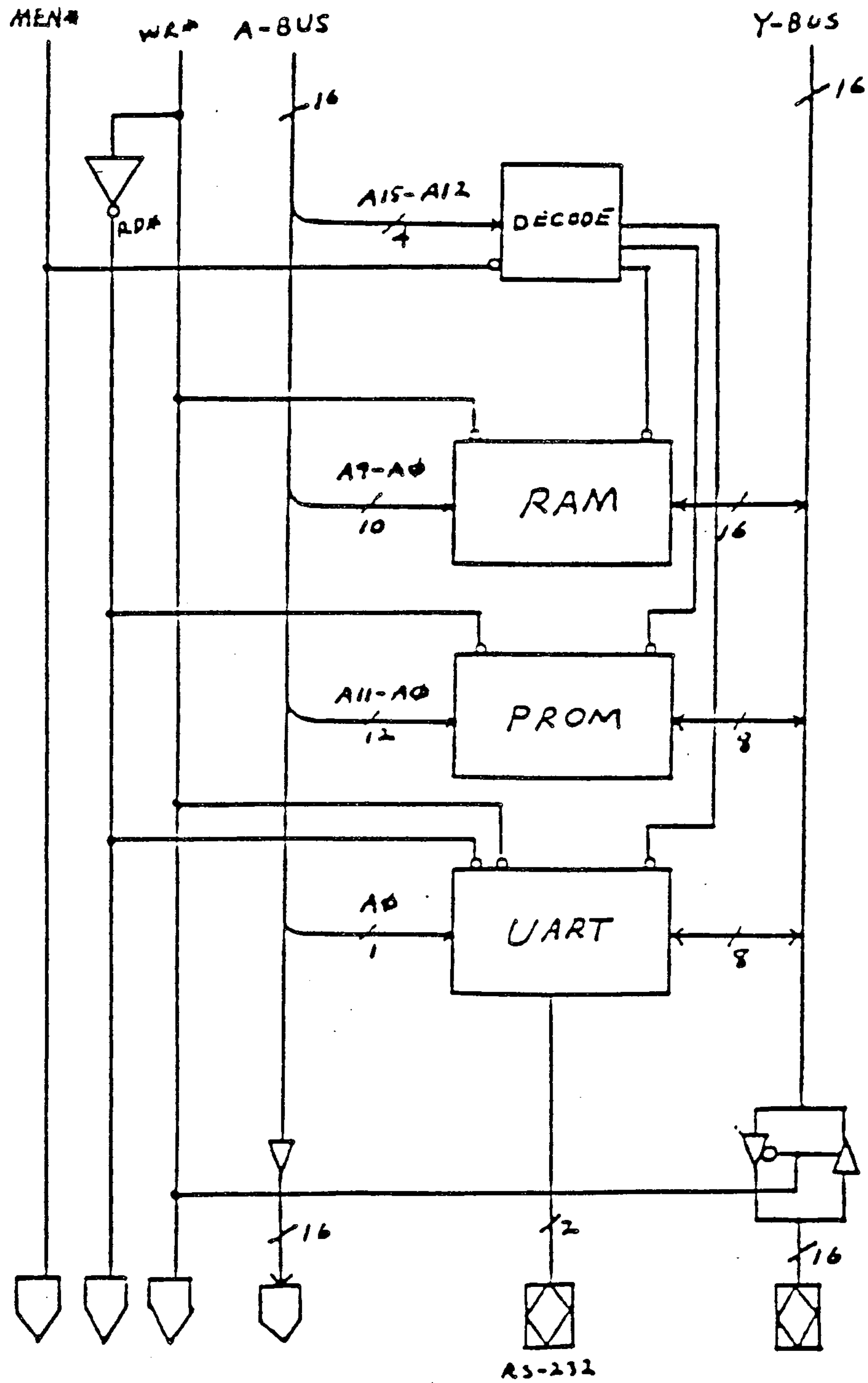


Figure EB-9. Memory and I/O Architecture

XI. Microinstruction Field Overlays

- Two areas of overlay occur on the Evaluation Board
- Bits 13-4 actually have four overlaid fields!
  - Micro Branch Address for Am2910
  - Counter value (10-bits only) for Am2910
  - Ra and Rb register addresses for Am29203
  - Constant value for the Am29203 via the B-bus
  - For example, CJP cannot be done if Ra, Rb are specified
- These fields require bit steering:
  - Branch Address selected by Am2910 instruction
  - Counter selected by Am2910 instruction
  - Ra, Rb selected by bits 47-45
  - Constant is selected by $\overline{\text{CON}}$ from the decoder
- Bits 19-16 have two overlaid fields
  - Am2904 Shift Control
  - Encoded-command field
- Status-Enable field selected by bit 22
- Shift-Control field selected by bit 20
- Encoded-Command field selected by bit 21
- This overlaying imposes some limitations on parallel operations

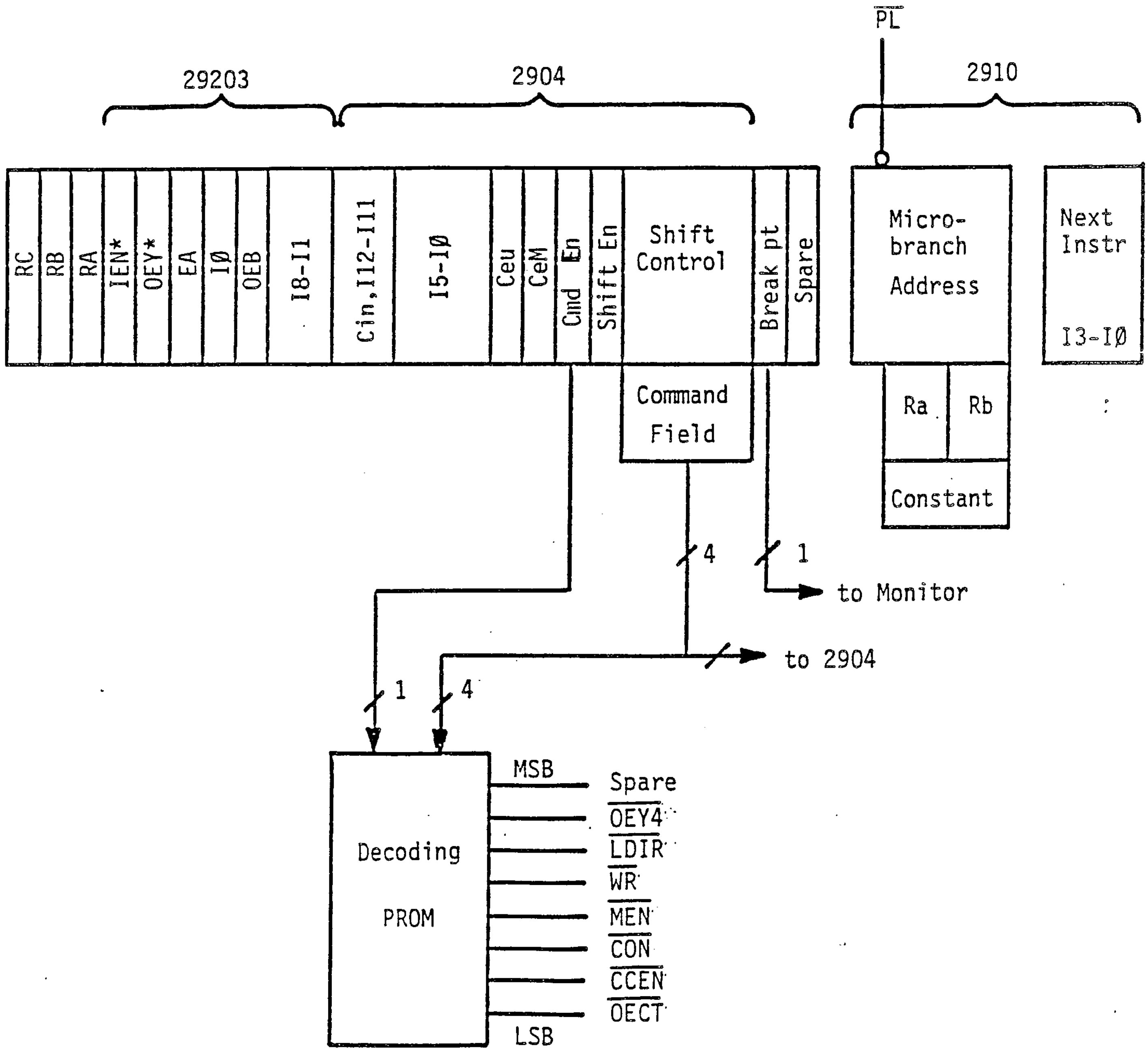


Figure EB-10. Microword and Decoding PROM

XII. Microinstruction Field Encoding

- Encoding is heavily used on the Command field, bits 19-16

- Microword is effectively reduced by 3 bits:
  - Only seven control bits from the decoder are used
  - Four bits are needed to generate these seven
  - Steering bit (21) would be needed anyway
  - Adding 3 bits to the pipeline would have added two more ICs: memory plus pipeline register

- The seven resources controlled are:

  - $\overline{OEY4}$ = Am2904 Status Output Enable
  - $\overline{LDIR}$ = Load Macroinstruction Register
  - $\overline{WR}$ = Memory write/read
  - $\overline{MEN}$ = Memory enable
  - $\overline{CON}$ = Enable constant field to Am29203
  - $\overline{CCEN}$ = Forced pass to the Am2910
  - $\overline{OECT}$ = Am2904 condition-code-test mux enable

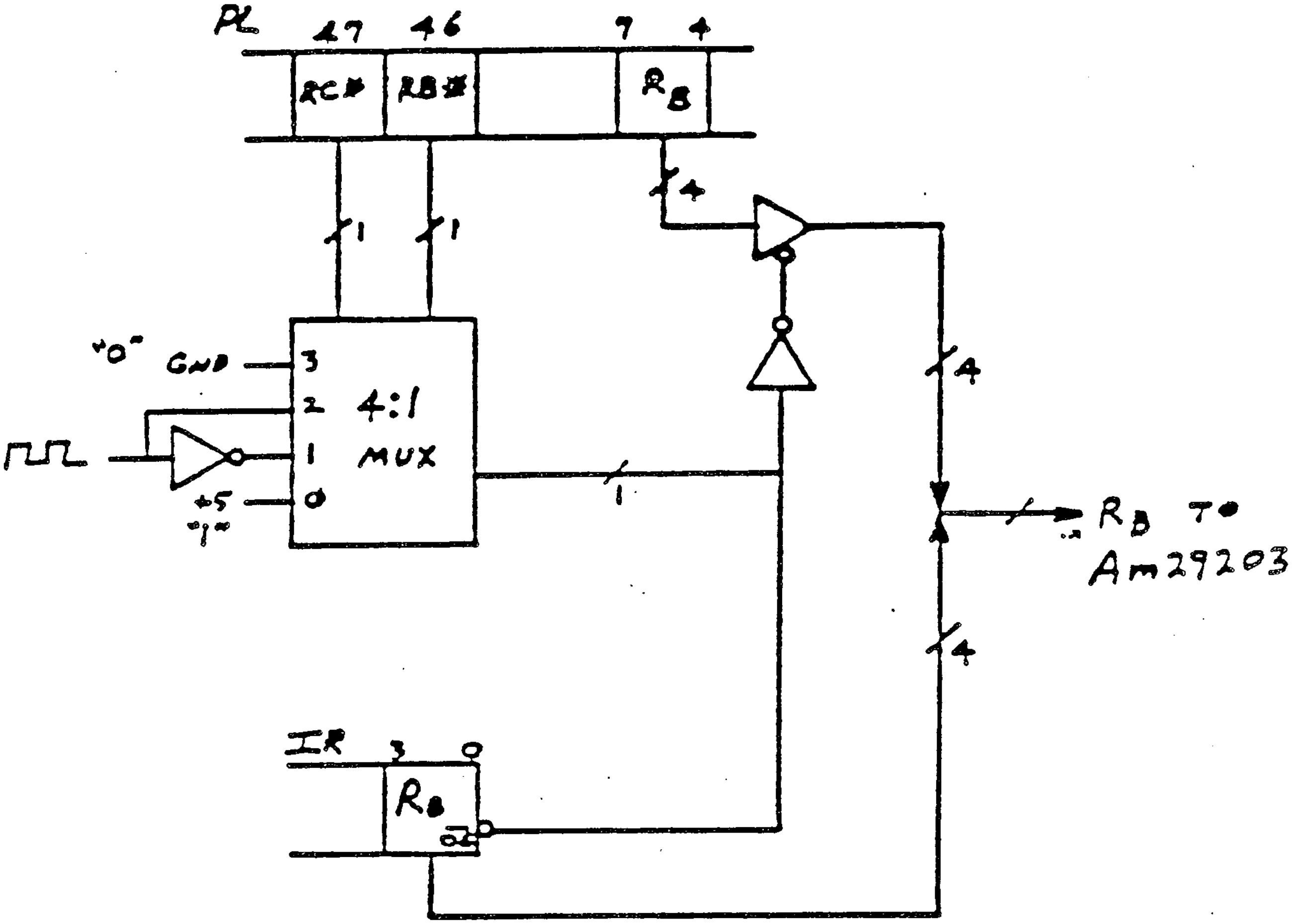
- Combinations of these seven are decoded to create fourteen meaningful commands

More Microinstruction Field Encoding

- Bits 47-46 control Rb and Rc for three-address instructions
- These bits select the Am29203 Rb-address from the IR or from the pipeline
- These bits are encoded to provide four possible conditions
  - ØØ => Rb comes from the pipeline (2-address)
  - Ø1 => Rb comes from IR first half cycle,
    Rb comes from pipeline second half cycle
  - 1Ø => Rb comes from pipeline first half cycle,
    Rb comes from IR second half cycle
  - 11 => Rb comes from the IR (2-address)

Table EB-1. Decoding-PROM Map

Addr	Spare	OEY4*	LDIR*	WR*	MEN*	CON*	CCEN*	OECT*	.DEF	Hex Value	Explanation
ØØ	1	Ø	1	1	1	1	1	1	OEYØ4	BF	Enable 2904 Y-output
Ø1	1	1	Ø	1	1	1	1	1	LDIR	DF	Load Instruction Register (IR)
Ø2	1	1	Ø	1	1	Ø	1	1	CONAB	DB	Register Address thru ALU to IR
Ø3	1	1	1	1	Ø	1	1	1	RDMEM	F7	Read Memory
Ø4	1	1	1	Ø	Ø	1	1	1	WRTMEM	E7	Write to memory
Ø5	1	1	1	1	1	Ø	1	1	CONBUS	FB	Enable constant to B-bus
Ø6	1	1	Ø	1	Ø	1	1	1	IFTCH	D7	Instruction fetch
Ø7	Ø	1	1	1	1	1	1	1	SPARE	7F	Enable spare command line
Ø8	1	1	1	1	1	1	Ø	1	SCCEN	FD	CCEN input to Am2910
Ø9	1	1	1	1	1	1	Ø	Ø	ALUTST	FC	Enable 2904 CT to 2910 CC input
ØA	1	1	1	1	1	1	1	1	READ	FF	Read enable
ØB	1	1	1	Ø	1	1	1	1	WRITE	EF	Write enable
ØC	1	Ø	1	Ø	Ø	1	1	1	SAVESTAT	A7	Write 2904 status to memory
ØD	1	1	1	Ø	Ø	Ø	1	1	SAVECON	E3	Write constant to memory
ØE	1	1	1	1	1	1	1	1		FF	Not used
ØF	1	1	1	1	1	1	1	1		FF	Not used
1Ø	1	1	1	1	1	1	1	1		FF	Not enabled
11	1	1	1	1	1	1	1	1		FF	Not enabled
.											
.											
.											
1E	1	1	1	1	1	1	1	1		FF	Not enabled
1F	1	1	1	1	1	1	1	1		FF	Not enabled



RC#	RB#	R_B
0	0	PL
0	1	IR | PL
1	0	PL | IR
1	1	IR

3-ADDRESS

Figure EB-11. Register Address Source Encoding

ED2900A

B. - Laboratory 1 - Introduction to Evaluation Board Monitor

- Make certain power is connected to the board.
- Make certain that the CRT is connected.
- Turn on the power. Allow the CRT to warm up.
- Press the reset button on the board.
- You should see the prompt ">" followed by a short summary of the available commands
- If this message doesn't appear, get help.

Monitor Command Summary

- The Evaluation Board Monitor prompt is a ">".
- Commands are terminated by a carriage return <CR>.
- All displays are in hexdecimal (Base 16).
- Four types of error control/recovery are available:
  - The ESC key aborts any command.
  - The backspace key can be used to correct input.
  - Keep typing. Only the last 4 hex digits are used.
  - Illegal commands are ignored and "beeped".
- The major commands (entered after the >) are:

  >L - load
  >D - display
  >G - go - execute microcode
  >T - test - run test routines
  >Z - zeros all registers

- Except for "T", these commands need further input.
- Evaluation Board resources are identified by:

  R - Registers
  M - Main memory (not control store)
  C - Control store
  P - Pipeline
  B - Breakpoint
  I - Instruction set (macro)
  A - Address of current pipeline galue
  N - Address of Next pipeline value

Using the Display Register Command

- Type DR on the terminal. The current register values are displayed in the following format:

```
>DR REG:
 Ø    1    2    3    4    5    6   ....  D    E    F
ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ      ØØØØ ØØØØ ØØØØ
  Q     IR     MS     US(VCNZ)
ØØØØ  ØØØØ     Ø      Ø
```

The first row displays the ALU register numbers.

The second row displays the ALU register contents.

The third and fourth rows show the contents of

Q - The Am29203 Q-register

IR - The Macro Instruction Register

MS - The Am2904 Macro Status Register (4-bits)

US - The Am2904 Micro Status Register (4-bits)

The sequence, "VCNZ", reminds you of the bit-sequence of the status bits in the MS and US registers:

overflow, carry, sign, zero

Using the Display Main Memory Command

- Type DM on the terminal.
- You are prompted for a starting address

  >DM ADDR:

- Enter up to 4 hex digits followed by a carriage return.
- The starting address and the contents of the next eight sequential locations are displayed.
- Typing any key displays the next 8 locations.
- Typing the ESC key terminates this mode.
- Display the 24 locations starting with address 200.

Using the Display Control Store Command

- Operation is the same as for main memory.
- The display format is slightly different.
- Type DC on the terminal.
- You are prompted for a starting address

  >DC ADDR:

- Enter up to 4 hex digits followed by a carriage return.
- The address and one 48-bit WCS word are displayed.
- Typing the ESC key terminates this mode.
- Display five locations starting with address 100.

Using the Display Pipeline Command

- Type DP on the terminal.
- The 48-bit pipeline is displayed.
- This is the actual contents of the pipeline register during the current microcycle (that is about to be executed).

Using the Display Address Command

- Type DA on the terminal.
- The address of the instruction in the pipeline is displayed.

Using the Display Next Address Command

- Type DN on the terminal.
- The address of the next instruction to be put into the pipeline is displayed.

Using the Display Breakpoints Command

- Type DB on the terminal.
- The addresses of all microinstructions with the breakpoint set are displayed.

Using the Load Register Command

- Type LR on the terminal.
- You are prompted for a register identifier:

    >LR REG:

- Enter the register number and you are prompted for data, i.e. to alter the contents of register R3:

    >LR REG: 3 DATA:

- Enter up to 4 hex digits terminated by a <CR>.

    >LR REG: 3 DATA: 3A6<CR>.

- Another register is prompted for. Use ESC to quit.
- Now load "ØØØØ" into the ALU registers.
- Load "FFFFF" into the Q register.
- Load "ABCD" into the IR (register I).
- Load "E" into the macro status register (register S).
- Load "F" into the macro status register (register U).
- Next verify your actions by using "DR" to display.

Using the Load Main Memory Command

- Type LM on the terminal.
- You are prompted for an address:

```
>LM ADDR:
```

- Enter the address, terminated by a <CR> and you are prompted for data:

```
>LM ADDR:  Ø37<CR>
DATA:
```

- Enter up to 4 hex digits terminated by a <CR>.

```
>LM ADDR:  Ø37<CR>
DATA:  FFFF<CR>
DATA:
```

- Data for the next sequential location is prompted for. Use ESC to quit.
- Now load the numbers Ø through F into the sixteen memory locations beginning at 2ØØ.
- Use DM to verify your actions.

Using the Load Control Store Command

- Type LC on the terminal.

- You are prompted for an address:

```
>LC ADDR:
```

- Enter the address, terminated by a <CR> and you are prompted for data:

```
>LC ADDR:  Ø37<CR>
DATA:
```

- Enter up to 12 hex digits in groups of 4, separated by <SPACE> or by <CR> (which is not echoed):

```
>LC ADDR:  Ø37<CR>
DATA:  FFFF<CR> 1234<CR> ABCD<CR>
DATA:
```

or

```
>LC ADDR:  Ø37<CR>
DATA:  FFFF 1234 ABCD<CR>
DATA:
```

- Data for the next sequential location requested. Use ESC to quit.

- Now load the following locations in WCS:

Address	Word (HEX)
1ØØ	AAAA BBBB CCCC
1Ø1	DDDD EEEE FFFF
1Ø2	1234 5678 9ABC

- Use DC to verify your actions.

Using the Load Pipeline Command

- Type LP on the terminal.
- You are prompted for data:

```
>LP DATA:
```

- Enter up to 12 hex digits in groups of 4, separated by <SPACE> or by <CR> (which is not echoed):

```
>LP DATA:  FFFF<CR> 1234<CR> ABCD<CR>
>
```

or

```
>LP DATA:  FFFF 1234 ABCD<CR>
>
```

- Now load the following word into the pipeline register:

```
AAAA BBBB CCCC
```

- Use DP to verify your actions.

Using the Load Instructions Command

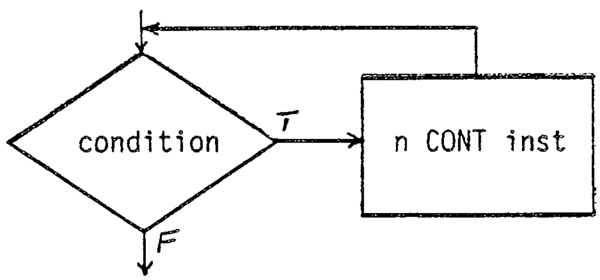
- Type LI on the terminal.
- The example microcode for a macro instruction set is loaded from PROM into WCS from ØØØØ to Ø2ØØ.
- Now verify the load using DC.

C. - Exercise 1 - Am2910 Sequencer

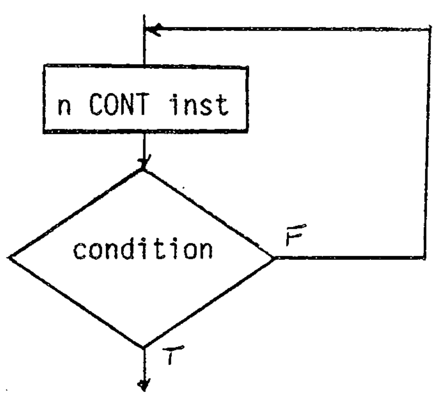
1. Using the Evaluation Board Am2910 mnemonic commands determine the HEX instructions using the standard evaluation board microinstruction format sheet. Generate the code for the Am2910 fields only.

2. Write a partial microroutine for

   a. For a DO - WHILE type loop; i.e., check condition and if true repeat n CONT instructions:

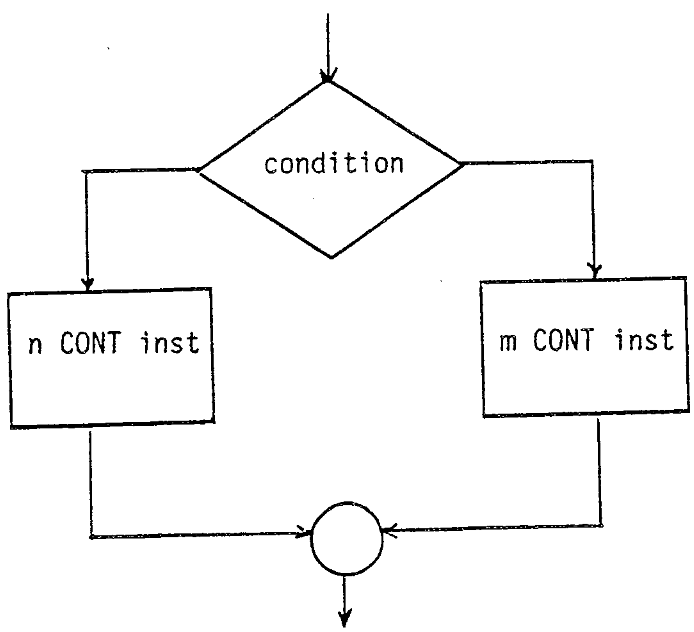


   b. For a DO - UNTIL type loop; i.e., perform n CONT instructions and repeat if condition is false:



Exercise 1 (cont'd)

3. Write a partial microroutine for an IF - THEN - ELSE Structure; i.e., if a condition is true, then execute n CONT instructions. If the condition is false then execute m CONT instructions:



D. - Laboratory 2 - Microprogramming the Sequencer

- The purpose of this laboratory is to acquaint the student with the Am2910 microaddress sequencing capabilities. The laboratory consists of exercises that emphasize the use of the standard structured microprogram control flow operations.

- This lab uses only the sequencer portion of the Am29203 Evaluation Board.

- You are given values to enter in those fields that cannot be treated as don't cares.

- The objective here is to provide hands on experience programming the Am2910 sequencer.

- Not all sixteen Am2910 instructions are exercised.

- If time permits, try additional programs that exercise Am2910 commands of interest.

- Use Appendix A for selecting proper mnemonics .

Sequencer Microword Fields

- Only the sequencer fields are programmed

  - Bits 3-0 = Am2910 instruction field

  - Bits 13-4 = Branch address & counter field

  - Bits 23-20 = Command enable field and status latch controls

    Use "C" to allow conditional tests

    Use "E" for forced pass via $\overline{CCEN}$

- The Am2903 macro status register is used to provide true and false test conditions.

- Bits 23-22 must be "1" to prevent changing the Am2904 status registers inadvertently.

- Bit 15 (breakpoint) must be loaded with "1".

- All other fields are don't cares, set to "1".

- It is suggested that the Standard Evaluation Board Microinstruction Format be used for clarity

Standard Evaluation Board Microinstruction Format

Bits	Device	Field	Value		Explanation
47-45		REGSRC	XXXXX	Q#X	;
44		IEN	XXXXX	B#X	;
43		OEY	XXXXX	B#X	;
42-40	Am29203	SOURCE	XXXXX	Q#X	;
39-36		DEST	XXXXX	H#X	;
35-32		FUNCT	XXXXX	H#X	;
31-30		CARRY	XXXXX	B#XX	;
29-24		TEST	MZ	Q#2H#4	;Test Macro Zero
23	Am2904	CEu	NOMICRO	B#1	;Don't latch micro status
22		CEM	NOMACRO	B#1	;Don't latch macro status
21-20		CMDSHFT	______	B#__	;Command or No Command
19-16		CMD/SHIFT	ALUTST	H#9	;Test Am2904 CT
15		BKPT	NOBREAK	B#1	;don't set breakpoint
14		SPARE		X	;not used
13-12		ADDRESS		B#__	;Branch Address MSBs
11-4		ADDRESS		H#__	;Branch Address LSBs
3-0	Am2910 --	INSTR	____	H#_	;

Resulting Microword: FFFF E4_9 ____ (X=1)

use Hex C for test

use HEX E for pass

BR ADDR/COUNTER

Am2910 instruction

Specific Laboratory 2 Exercises

1. Starting at micromemory address Ø, execute 4 continue instructions.

2. Add a fifth instruction to the above microprogram to branch back to address Ø unconditionally.

3. Starting at micromemory address 1Ø, execute a microprogram which loops on one microinstruction 5 times

   a) using RPCT

   b) using RFCT

   Don't forget to set the counter before you begin.

4. Generate a 3-instruction loop using the LOOP instruction.

For each exercise follow the following procedure:

- Load the control store with the microcode
- Load the "S" register with "Ø" for Pass, "1" for Fail
- In response to the ">" prompt, type G (for Go)
- You are prompted for an address:

```
>G ADDR:
```

- Enter the starting address of your routine and <CR>

```
>G ADDR: ØØØØ<CR>
```

- You are prompted for single stepping:

```
>G ADDR: ØØØØ<CR>
 STEP?
```

- Enter Y for yes

```
>G ADDR: ØØØØ<CR>
 STEP?Y
```

- The monitor enters Trace mode. The first microinstruction is fetched into the pipeline to be executed. The address, pipeline, and registers are displayed. Any key causes the next microstep to occur. The ESC key terminates the process.

- Single step through the routine, watching the address and pipeline register contents.

- For conditional statements, exercise both options by changing the contents of the "S" register using LR.

E. - Exercise 2 - Am29203 ALU

1. Write and excute a microroutine incrementing the register F by 1.

2. Write and execute a microroutine for clearing all ALU RAM registers.

3. Write a microroutine for unsigned multiply using the the special functions.

4. Write a microroutine for signed two's complement multiply using the special functions.

F. - Laboratory 3 - Microprogramming the ALU Basic Functions

- The purpose of this laboratory is to provide an understanding of the Am29203 ALU and associated operations through the use of microprogramming. The laboratory consists of 6 exercises starting with a set of simple operations and ending with special Am29203 operations.

1. Using individually selected initial values entered via the monitor: write, run, and debug each register transfer language statement for the specified microcode. Define each operation with comments.

   Initially, load the general purpose, IR, S, U and Q registers with values and check these values by means of a DR command.

2. Microcode

Microprogram Address	ALU Operation	Operand Address Source	Comments
n	R2<--R3-R2	PL	
n+1	R4<--R4-R5-1	PL	
n+2	R5<--R5+1	PL	
n+3	R5<--R5+Q	PL	
n+4	IR_{7-0}<--67_{16}		
n+5	R8<--R6+R7	R_C from PL R_A , R_B from IR	
n+6	R7<--R6+R8	R_C from IR R_A , R_B from PL	
n+7	R7<--Ø	PL	
n+8	$R8_{7-Ø}$<--AA_{16}		
n+9	R8<--$D13E_{16}$		
n+A	RB<--RA+RB with IEN high		
n+B	RB<--RA+RB with IEN low and OEy high	PL	
n+C	RC,Q<--R_A+R_B	PL	
n+D	RD<--Q	PL	
n+E	Q<--RE	PL	

Worksheet for Exercise 2

Microprogram Address					Microcode							
n	___	___	___	___	7	F	3	F	___	___	___	___
n+1	___	___	___	___	3	F	F	F	___	___	___	___
n+2	___	___	___	___	___	___	___	___	___	___	___	___
n+3	___	___	___	___	___	___	___	___	___	___	___	___
n+4	___	___	___	___	1	F	1	2	___	___	___	___
n+5	___	___	___	___	___	___	___	___	___	___	___	___
n+6	___	___	___	___	___	___	___	___	___	___	___	___
n+7	___	___	___	___	___	___	___	___	___	___	___	___
n+8	___	___	___	___	___	___	___	___	___	___	___	___
n+9	___	___	___	___	___	___	___	___	___	___	___	___
n+A	___	___	___	___	___	___	___	___	___	___	___	___
n+B	___	___	___	___	___	___	___	___	___	___	___	___
n+C	___	___	___	___	___	___	___	___	___	___	___	___
n+D	___	___	___	___	___	___	___	___	___	___	___	___
n+E	___	___	___	___	___	___	___	___	___	___	___	___

Exercise 3

Clear R9. Loop 5 times, incrementing R9 each time through the loop. Write one version using RPCT, and a second version using RFCT.

Exercise 4

Using the monitor LR command, load a general purpose register with the value 5. Then write the microcode, using the LOOP instruction to decrement the general purpose register by 1 until the register is zero. You will be required to use the special function for decrement.

Exercise 5

Enter the sample microcode for single-length normalize, unsigned multiply and binary-to-BCD and BCD-to-binary conversions. Load the appropriate registers with the values using the monitor LR command.

Worksheet for Exercise 3

n ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

n+1 ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

n+2 ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

n+3 ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

n ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

n+1 ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

n+2 ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

n+3 ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

Worksheet for Exercise 4

n ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

n+1 ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

n+2 ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

Exercise 5

Eval board microcode for a simple single length normalize. This code differs from the code in the ED2900A lecture in that the normalization takes two microwords per necessary shift.

Addr	Flowstep	Code	Comment
1ØØ	1	E266 3FFF FFFE	<Ra> --> Q
1Ø1	2	FØ8Ø 6Ø22 FFFE	SLN, Ien high
1Ø2	3	FFFF E5D9 DØ83	Jump to 108 on zero
1Ø3	4	E248 2BD9 DØ63	Jump to 106 on done
1Ø4	5(a)	EØ8Ø 6Ø22 FFFE	SLN
1Ø5	5(b)	FFFF E6D9 DØ43	Repeat if not done
1Ø6	-	FFFF FFFF FFF?	Return?
1Ø7	-		
1Ø8	-		Handle the zero case

This code does a two's complement multiply of the contents of Ra by Q, with the result in Rb. Both of the registers are selected by the contents of the IR.

1ØØ	FFFF FFFF CØEC	-	LDCT with 14
1Ø1	EØ2Ø 3FE3 DØ19	-	Multiply step
1Ø2	EØ6Ø BFE3 FFFE	-	Multiply last step

Binary to BCD conversion, the binary number in the Q register is converted to a BCD number in Rb, where Rb is selected by the IR.

1ØØ	E248 3FFF CØF4	-	Push, Ld cnt, clear Rb
1Ø1	EØ9Ø 3FE4 FFF8	-	Bin/BCD and loop on file

BCD to Binary conversion, BCD in Rb to binary in Q.

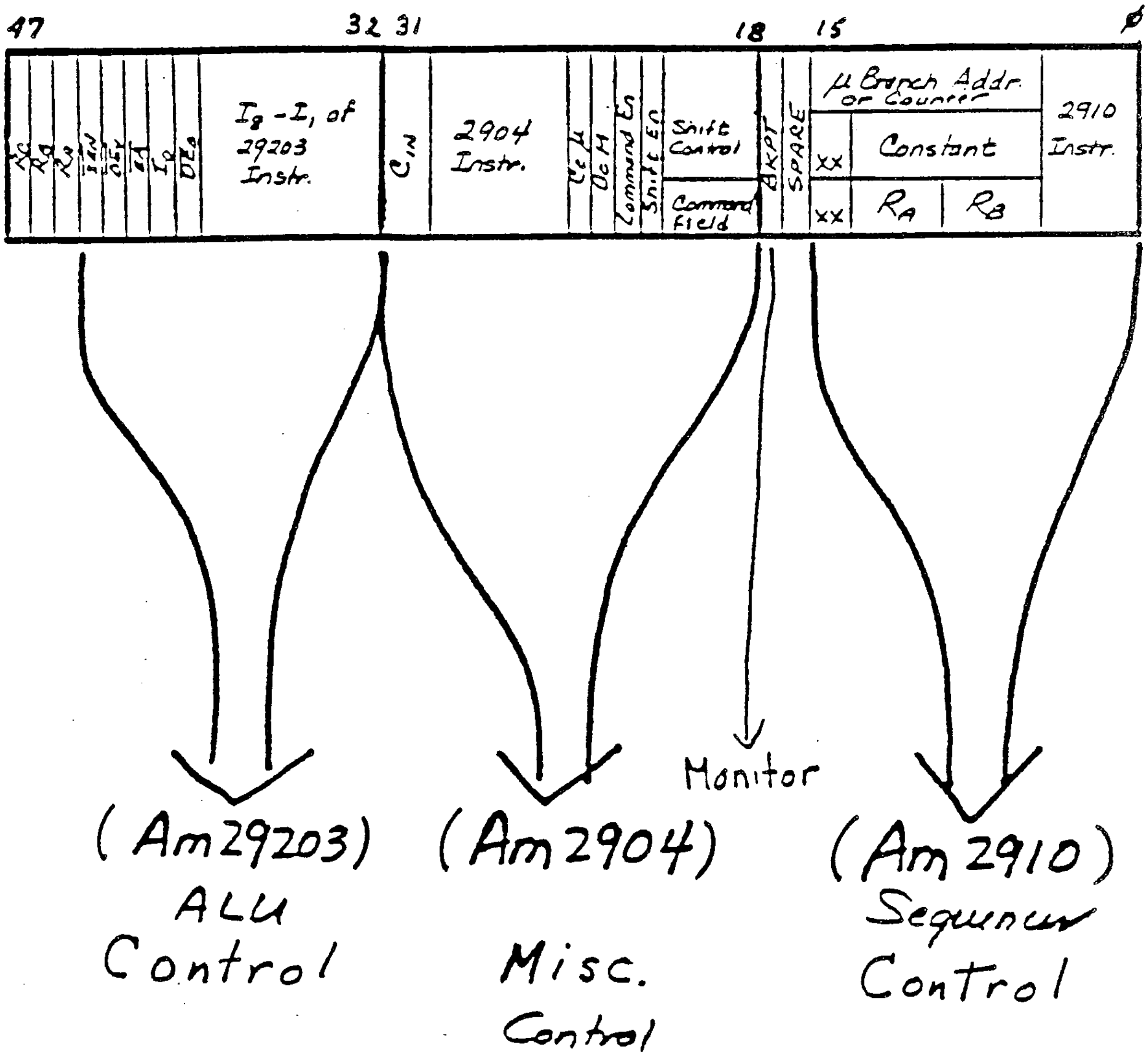
1ØØ	E034 3FE6 CØEC	-	Ld cnt, first downshift to Q
1Ø1	EØ1Ø 3FE6 DØ19	-	Convert, loop on pipeline

6. Implement and exercise the unsigned multiply operation with special functions. Use registers RØ and R1 for the augend and addend repectively, and registers R3 and R4 for storing the result.

APPENDIX A

EVALUATION BOARD FIELD DEFINITIONS

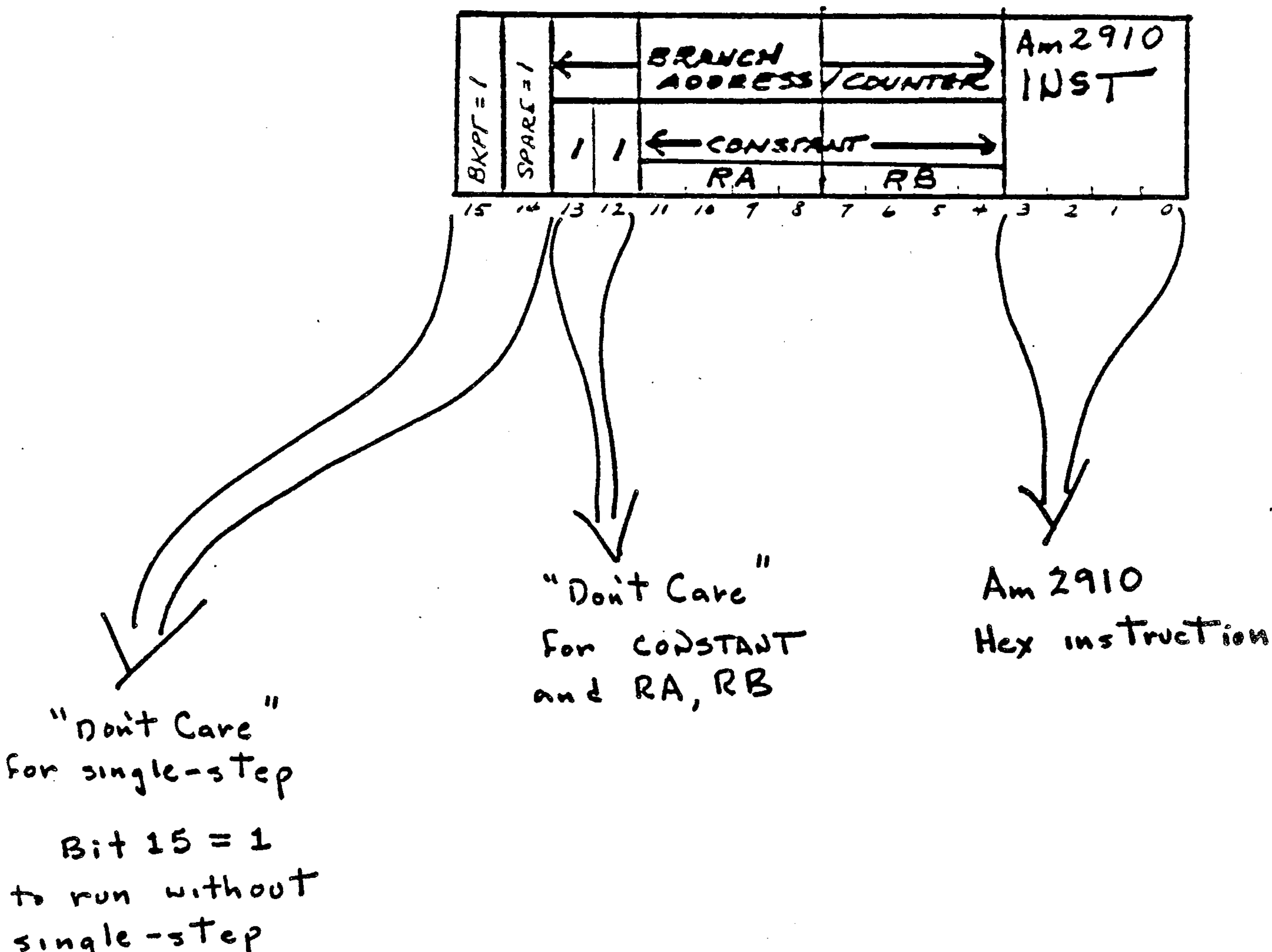
Evalution Board - Microword Format



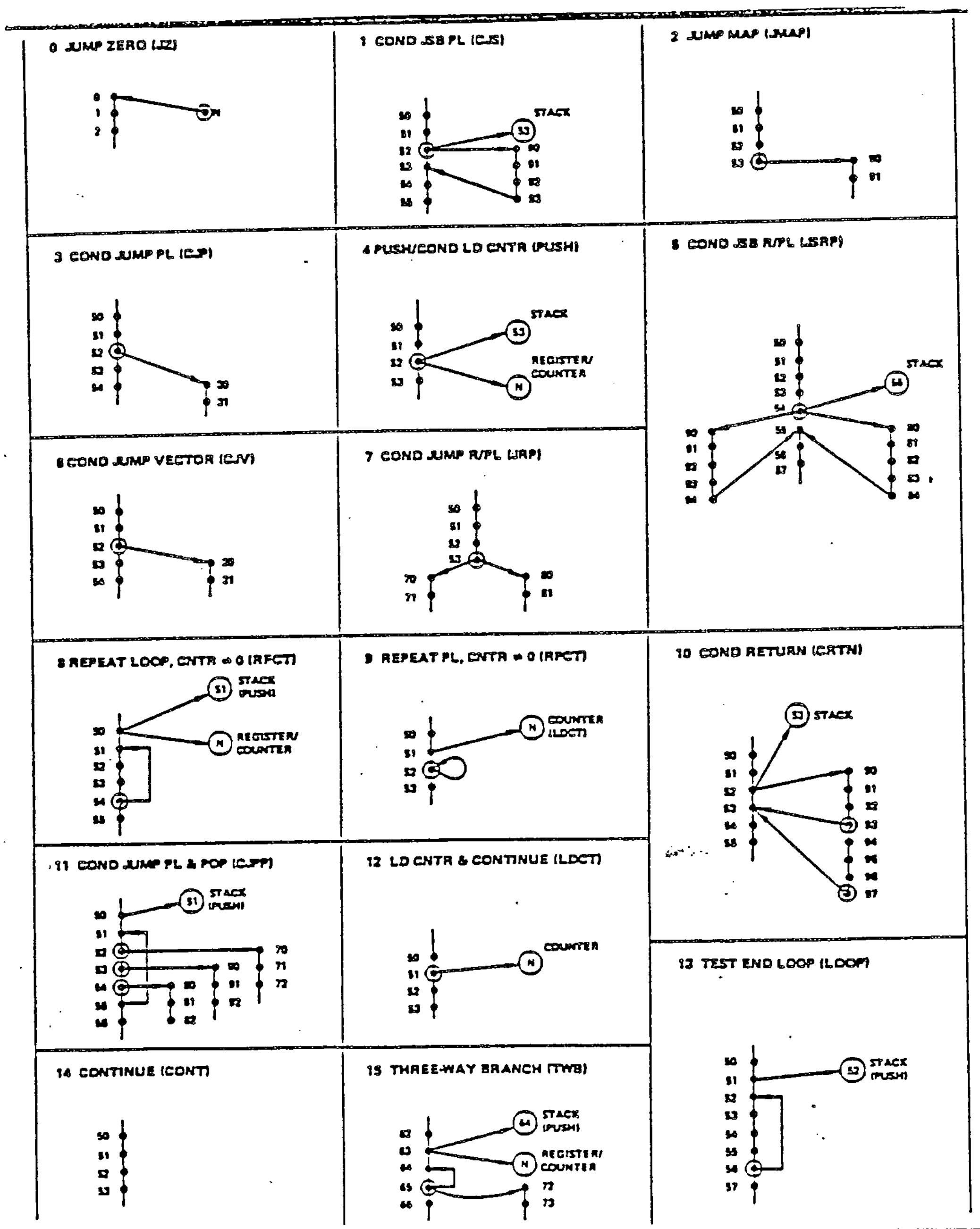
Bits	Device	Field	Value	Field HEX	Explanation
47-45		REGSRC	_____	Q#_	;
44	AM29203	IEN	_____	B#_	;
43		OEY	_____	B#_	;
42-40		SOURCE	_____	Q#_	;
39-36		DEST	_____	H#_	;
35-32		FUNCT	_____	H#_	;
31-30	AM2904	CARRY	_____	B#__	;
29-24		STAT/TST	_____	Q#__	;
23		CEu	_____	B#_	;
22		CEM	_____	B#_	;
21-20		CMDSHFT	_____	B#__	;
19-16		CMD	_____	H#_	;
15		BKPT	NOBREAK	B#1	;don't set breakpoint
14		SPARE		X	;don't care
13-12		CONSTANT	_____	B#__	;MSB for br address
11-8	REG.SEL	Ra	__	H#_	;Ra=
7-4		Rb	__	H#_	;Rb=
3-0	AM2910	INSTR	_____	H#_	;

Resulting Microword: ____ ____ ____ (X=_)

Bits 15–ø Sequencer Control (Am2910)

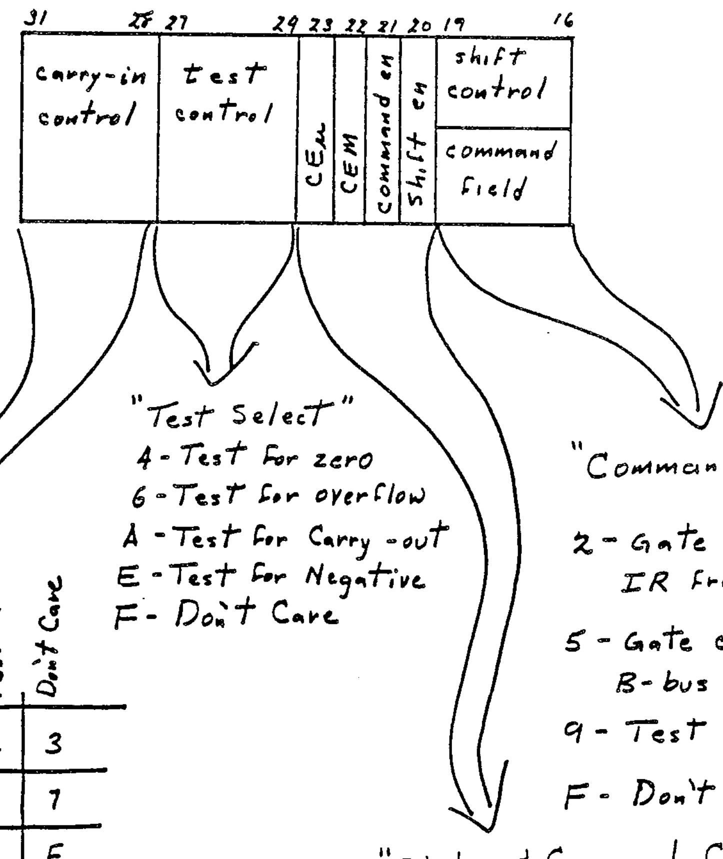


3 ϕ
2910


0 JUMP ZERO (JZ)
1 COND JSB PL (CJS)
2 JUMP MAP (JMAP)
3 COND JUMP PL (CJP)
4 PUSH/COND LD CNTR (PUSH)
5 COND JSB R/PL (JSRP)
6 COND JUMP VECTOR (CJV)
7 COND JUMP R/PL (JRP)
8 REPEAT LOOP, CNTR ≠ 0 (RFCT)
9 REPEAT PL, CNTR ≠ 0 (RPCT)
10 COND RETURN (CRTN)
11 COND JUMP PL & POP (CJPP)
12 LD CNTR & CONTINUE (LDCT)
13 TEST END LOOP (LOOP)
14 CONTINUE (CONT)
15 THREE-WAY BRANCH (TWB)

Bits 31-16 Misc. Control (Am2904)

"A Beginning"



"Test Select"
4 - Test for zero
6 - Test for overflow
A - Test for Carry-out
E - Test for Negative
F - Don't Care

"Command PROM" (HEX)

2 - Gate constant to IR from PL via ALU

5 - Gate constant to B-bus from PL

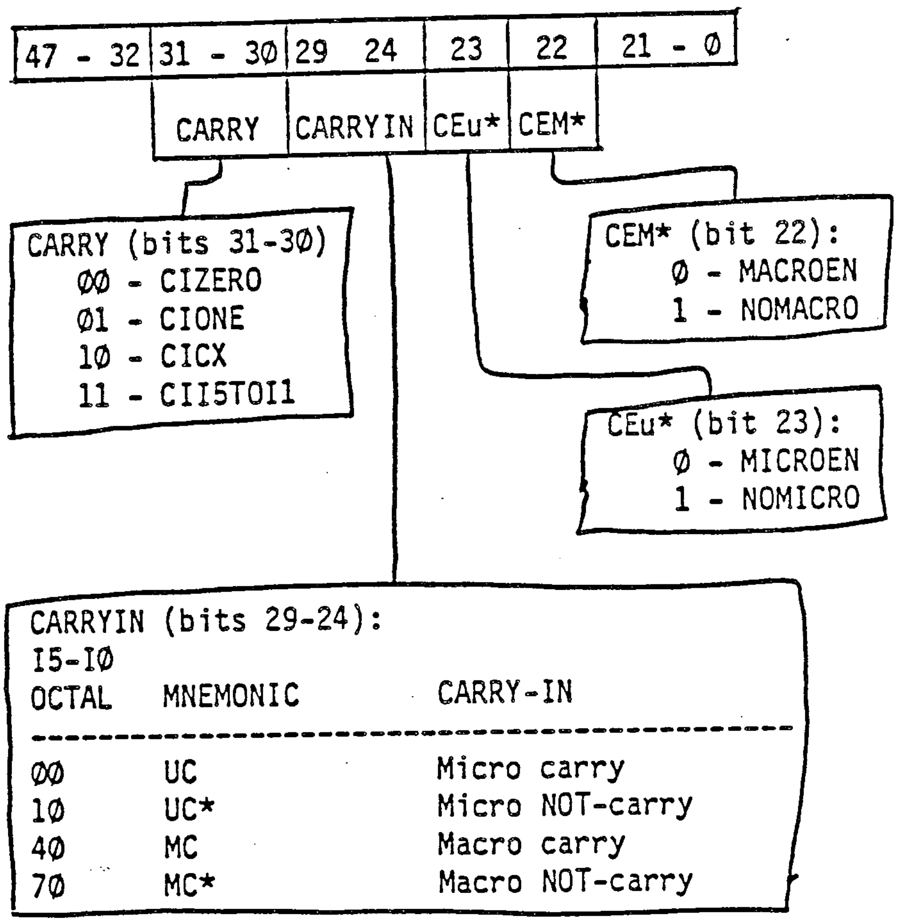
9 - Test ALU status

F - Don't Care

	Test μ	Test M	Don't Care
CI=∅	0,1	2	3
CI=1	5	6	7
Don't Care	0	E	F

"Status & Command Control" (HEX)

1 - Enable command and load status
3 - Disable command and load status
D - Enable command and don't load status
F - Don't Care



Carry-in Mux Control Fields & Mnemonics

Carry-In Control Multiplexer Codes

I12	I11	I5	I3	I2	I1	CØ
Ø	Ø	X	X	X	X	Ø
Ø	1	X	X	X	X	1
1	Ø	X	X	X	X	Cx
1	1	Ø	Ø	X	X	uC
1	1	Ø	X	1	X	uC
1	1	Ø	X	X	1	uC
1	1	Ø	1	Ø	Ø	uC*
1	1	1	Ø	X	X	MC
1	1	1	X	1	X	MC
1	1	1	X	X	1	MC
1	1	1	1	Ø	Ø	MC*

TABLE 1. MICRO STATUS REGISTER INSTRUCTION CODES.

Bit Operations

I_{543210} Octal	μSR Operation	Comments
10	$0 \rightarrow \mu_Z$	RESET ZERO BIT
11	$1 \rightarrow \mu_Z$	SET ZERO BIT
12	$0 \rightarrow \mu_C$	RESET CARRY BIT
13	$1 \rightarrow \mu_C$	SET CARRY BIT
14	$0 \rightarrow \mu_N$	RESET SIGN BIT
15	$1 \rightarrow \mu_N$	SET SIGN BIT
16	$0 \rightarrow \mu_{OVR}$	RESET OVERFLOW BIT
17	$1 \rightarrow \mu_{OVR}$	SET OVERFLOW BIT

Register Operations

I_{543210} Octal	μSR Operation	Comments
00	$M_X \rightarrow \mu_X$	LOAD MSR TO μSR
01	$1 \rightarrow \mu_X$	SET μSR
02	$M_X \rightarrow \mu_X$	REGISTER SWAP
03	$0 \rightarrow \mu_X$	RESET μSR

Load Operations

I_{543210} Octal	μSR Operation	Comments
06, 07	$I_Z \rightarrow \mu_Z$ $I_C \rightarrow \mu_C$ $I_N \rightarrow \mu_N$ $I_{OVR} + \mu_{OVR} \rightarrow \mu_{OVR}$	LOAD WITH OVERFLOW RETAIN
30, 31 50, 51 70, 71	$I_Z \rightarrow \mu_Z$ $\overline{I_C} \rightarrow \mu_C$ $I_N \rightarrow \mu_N$ $I_{OVR} \rightarrow \mu_{OVR}$	LOAD WITH CARRY INVERT
04, 05 20-27 32-47 52-67 72-77	$I_Z \rightarrow \mu_Z$ $I_C \rightarrow \mu_C$ $I_N \rightarrow \mu_N$ $I_{OVR} \rightarrow \mu_{OVR}$	LOAD DIRECTLY FROM I_Z, I_C, I_N, I_{OVR}

Note: The above tables assume $\overline{CE}_\mu$ is LOW.

TABLE 2. MACHINE STATUS REGISTER INSTRUCTION CODES.

Register Operations

I_{543210} Octal	MSR Operation	Comments
00	$Y_X \rightarrow M_X$	LOAD Y_Z, Y_C, Y_N, Y_{OVR} TO MSR
01	$1 \rightarrow M_X$	SET MSR
02	$\mu_X \rightarrow M_X$	REGISTER SWAP
03	$0 \rightarrow M_X$	RESET MSR
05	$\overline{M_X} \rightarrow M_X$	INVERT MSR

Load Operations

I_{543210} Octal	MSR Operation	Comments
04	$I_Z \rightarrow M_Z$ $M_{OVR} \rightarrow M_C$ $I_N \rightarrow M_N$ $M_C \rightarrow M_{OVR}$	LOAD FOR SHIFT THROUGH OVERFLOW OPERATION
10, 11 30, 31 50, 51 70, 71	$I_Z \rightarrow M_Z$ $\overline{I_C} \rightarrow M_C$ $I_N \rightarrow M_N$ $I_{OVR} \rightarrow M_{OVR}$	LOAD WITH CARRY INVERT
06, 07 12-17 20-27 32-37 40-47 52-67 72-77	$I_Z \rightarrow M_Z$ $I_C \rightarrow M_C$ $I_N \rightarrow M_N$ $I_{OVR} \rightarrow M_{OVR}$	LOAD DIRECTLY FROM I_Z, I_C I_N, I_{OVR}

Notes: 1. The above tables assume $\overline{CE}_M$, $\overline{E}_Z$, $\overline{E}_C$, $\overline{E}_N$, $\overline{E}_{OVR}$ are LOW.
2. A shift-through-carry instruction loads M_C irrespective of I_5-I_0.

TABLE 3. Y OUTPUT INSTRUCTION CODES.

$\overline{OE}_Y$	I_5	I_4	Y Output	Comment
1	X	X	Z	Output Off High Impedance
0	0	X	$\mu_i \rightarrow Y_i$	See Note 1
0	1	0	$M_i \rightarrow Y_i$	
0	1	1	$I_i \rightarrow Y_i$	

Notes: 1. For the conditions:
$I_5, I_4, I_3, I_2, I_1, I_0$ are LOW, Y is an input.
$\overline{OE}_Y$ is "Don't Care" for this condition.
2. X is "Don't Care" condition.

STATUS (bits 29-24):

I5-IØ OCTAL	CEu*=Ø MICRO	CEM*=Ø MACRO	BOTH	Action if OEY*=Ø
ØØ	MSRTOUSR	YTOMSR	YMSRUSR	UTOY
Ø1	SETUSR	SETMSR	SETREGS	(ØØ)
Ø2	MSRTOUSR	USRTOMSR	SWAPREGS	(ØØ)
Ø3	RESETUSR	RESETMSR	RESETREGS	(ØØ)
Ø4	(2Ø)	SWAPMCMO	?	(ØØ)
Ø5	(2Ø)	INVERTMSR	?	(ØØ)
Ø6	IRETOVR1	(2Ø)	?	(ØØ)
(Ø7)	(Ø6)	(Ø6)	?	(ØØ)
1Ø	RESETUZ	(3Ø)	?	(ØØ)
11	SETUZ	(3Ø)	?	(ØØ)
12	RESETUC	(2Ø)	?	(ØØ)
13	SETUC	(2Ø)	?	(ØØ)
14	RESETUN	(2Ø)	?	(ØØ)
15	SETUN	(2Ø)	?	(ØØ)
16	RESETUO	(2Ø)	?	(ØØ)
17	SETUO	(2Ø)	?	(ØØ)
2Ø	ITOUSR	ITOMSR	ITOREGS	(ØØ)
21-27	(2Ø)	(2Ø)	(2Ø)	(ØØ)
3Ø	IWITHUC*	IWITHMC*	IWITHC*	(ØØ)
31	(3Ø)	(3Ø)	(3Ø)	(ØØ)
32-37	(2Ø)	(2Ø)	(2Ø)	(ØØ)
4Ø-47	(2Ø)	(2Ø)	(2Ø)	MTOY
5Ø-51	(3Ø)	(3Ø)	(3Ø)	(4Ø)
52-57	(2Ø)	(2Ø)	(2Ø)	(4Ø)
6Ø-67	(2Ø)	(2Ø)	(2Ø)	ITOY
7Ø-71	(3Ø)	(3Ø)	(3Ø)	(6Ø)
72-77	(2Ø)	(2Ø)	(2Ø)	(6Ø)

TABLE 4. CONDITION CODE OUTPUT (CT) INSTRUCTION CODES.

I_{3-0} HEX	I_3	I_2	I_1	I_0	$I_5 = I_4 = 0$	$I_5 = 0, I_4 = 1$	$I_5 = 1, I_4 = 0$	$I_5 = I_4 = 1$
0	0	0	0	0	$(\mu_N \oplus \mu_{OVR}) + \mu_Z$	$(\mu_N \oplus \mu_{OVR}) + \mu_Z$	$(M_N \oplus M_{OVR}) + M_Z$	$(I_N \oplus I_{OVR}) + I_Z$
1	0	0	0	1	$(\mu_N \odot \mu_{OVR}) \cdot \overline{\mu}_Z$	$(\mu_N \odot \mu_{OVR}) \cdot \overline{\mu}_Z$	$(M_N \odot M_{OVR}) \cdot \overline{M}_Z$	$(I_N \odot I_{OVR}) \cdot \overline{I}_Z$
2	0	0	1	0	$\mu_N \oplus \mu_{OVR}$	$\mu_N \oplus \mu_{OVR}$	$M_N \oplus M_{OVR}$	$I_N \oplus I_{OVR}$
3	0	0	1	1	$\mu_N \odot \mu_{OVR}$	$\mu_N \odot \mu_{OVR}$	$M_N \odot M_{OVR}$	$I_N \odot I_{OVR}$
4	0	1	0	0	μ_Z	μ_Z	M_Z	I_Z
5	0	1	0	1	$\overline{\mu}_Z$	$\overline{\mu}_Z$	$\overline{M}_Z$	$\overline{I}_Z$
6	0	1	1	0	μ_{OVR}	μ_{OVR}	M_{OVR}	I_{OVR}
7	0	1	1	1	$\overline{\mu}_{OVR}$	$\overline{\mu}_{OVR}$	$\overline{M}_{OVR}$	$\overline{I}_{OVR}$
8	1	0	0	0	$\mu_C + \mu_Z$	$\mu_C + \mu_Z$	$M_C + M_Z$	$\overline{I}_C + I_Z$
9	1	0	0	1	$\overline{\mu}_C \cdot \overline{\mu}_Z$	$\overline{\mu}_C \cdot \overline{\mu}_Z$	$\overline{M}_C \cdot \overline{M}_Z$	$I_C \cdot \overline{I}_Z$
A	1	0	1	0	μ_C	μ_C	M_C	I_C
B	1	0	1	1	$\overline{\mu}_C$	$\overline{\mu}_C$	$\overline{M}_C$	$\overline{I}_C$
C	1	1	0	0	$\overline{\mu}_C + \mu_Z$	$\overline{\mu}_C + \mu_Z$	$\overline{M}_C + M_Z$	$\overline{I}_C + I_Z$
D	1	1	0	1	$\mu_C \cdot \overline{\mu}_Z$	$\mu_C \cdot \overline{\mu}_Z$	$M_C \cdot \overline{M}_Z$	$I_C \cdot \overline{I}_Z$
E	1	1	1	0	$I_N \oplus M_N$	μ_N	M_N	I_N
F	1	1	1	1	$I_N \odot M_N$	$\overline{\mu}_N$	$\overline{M}_N$	$\overline{I}_N$

Notes: 1. $\oplus$ Represents EXCLUSIVE-OR $\odot$ Represents EXCLUSIVE-NOR or coincidence.

TEST (bits 29-24):

I5-I4 OCTAL	I3-IØ HEX	MNEMONIC	TEST
1	4	UZ	Micro Zero
2	4	MZ	Macro Zero
3	4	IZ	I-bus Zero
1	5	UZ*	Micro Not-Zero
2	5	MZ*	Macro Not-Zero
3	5	IZ*	I-bus Not-Zero
1	6	UOVR	Micro Overflow
2	6	MOVR	Macro Overflow
3	6	IOVR	I-bus Overflow
1	7	UOVR*	Micro Not-Overflow
2	7	MOVR*	Macro Not-Overflow
3	7	IOVR*	I-bus Not-Overflow
1	A	UC	Micro Carry
2	A	MC	Macro Carry
3	A	IC	I-bus Carry
1	B	UC*	Micro Not-Carry
2	B	MC*	Macro Not-Carry
3	B	IC*	I-bus Not-Carry
1	E	UN	Micro Negative
2	E	MN	Macro Negative
3	E	IN	I-bus Negative
1	F	UN*	Micro Not-Negative
2	F	MN*	Macro Not-Negative
3	F	IN*	I-bus Not-Negative

8-13. Test Control Fields & Mnemonics

19 16
COMMAND

Decoding PROM Map.

Addr	Spare	OEY4*	LDIR*	WR*	MEN*	CON*	CCEN*	OECT*	.DEF.	Hex Value	Explanation
ØØ	1	Ø	1	1	1	1	1	1	OEYØ4	BF	Enable 29Ø4 Y-output.
Ø1	1	1	Ø	1	1	1	1	1	LDIR	DF	Load Instruction Register (IR).
Ø2	1	1	Ø	1	1	Ø	1	1	CONAB	DB	Register Address thru ALU to IR.
Ø3	1	1	1	1	Ø	1	1	1	RDMEM	F7	Read Memory.
Ø4	1	1	1	Ø	Ø	1	1	1	WRTMEM	E7	Write to memory.
Ø5	1	1	1	1	1	Ø	1	1	CONBUS	FB	Enable constant to B-bus.
Ø6	1	1	Ø	1	Ø	1	1	1	IFTCH	D7	Instruction fetch.
Ø7	Ø	1	1	1	1	1	1	1	SPARE	7F	Enable spare command line.
Ø8	1	1	1	1	1	1	Ø	1	SCCEN	FD	CCEN input to Am291Ø.
Ø9	1	1	1	1	1	1	Ø	Ø	ALUTST	FC	Enable 29Ø4 CT to 291Ø CC input.
ØA	1	1	1	1	1	1	1	1	READ	FF	Read enable.
ØB	1	1	1	Ø	1	1	1	1	WRITE	EF	Write enable.
ØC	1	Ø	1	Ø	Ø	1	1	1	SAVESTAT	A7	Write 29Ø4 status to memory.
ØD	1	1	1	Ø	Ø	Ø	1	1	SAVECON	E3	Write constant to memory.
ØE	1	1	1	1	1	1	1	1		FF	Not used.
ØF	1	1	1	1	1	1	1	1		FF	Not used.
1Ø	1	1	1	1	1	1	1	1		FF	Not enabled.
11	1	1	1	1	1	1	1	1		FF	Not enabled.
.											
.											
.											
1E	1	1	1	1	1	1	1	1		FF	Not enabled.
1F	1	1	1	1	1	1	1	1		FF	Not enabled.

19 16
SHIFT

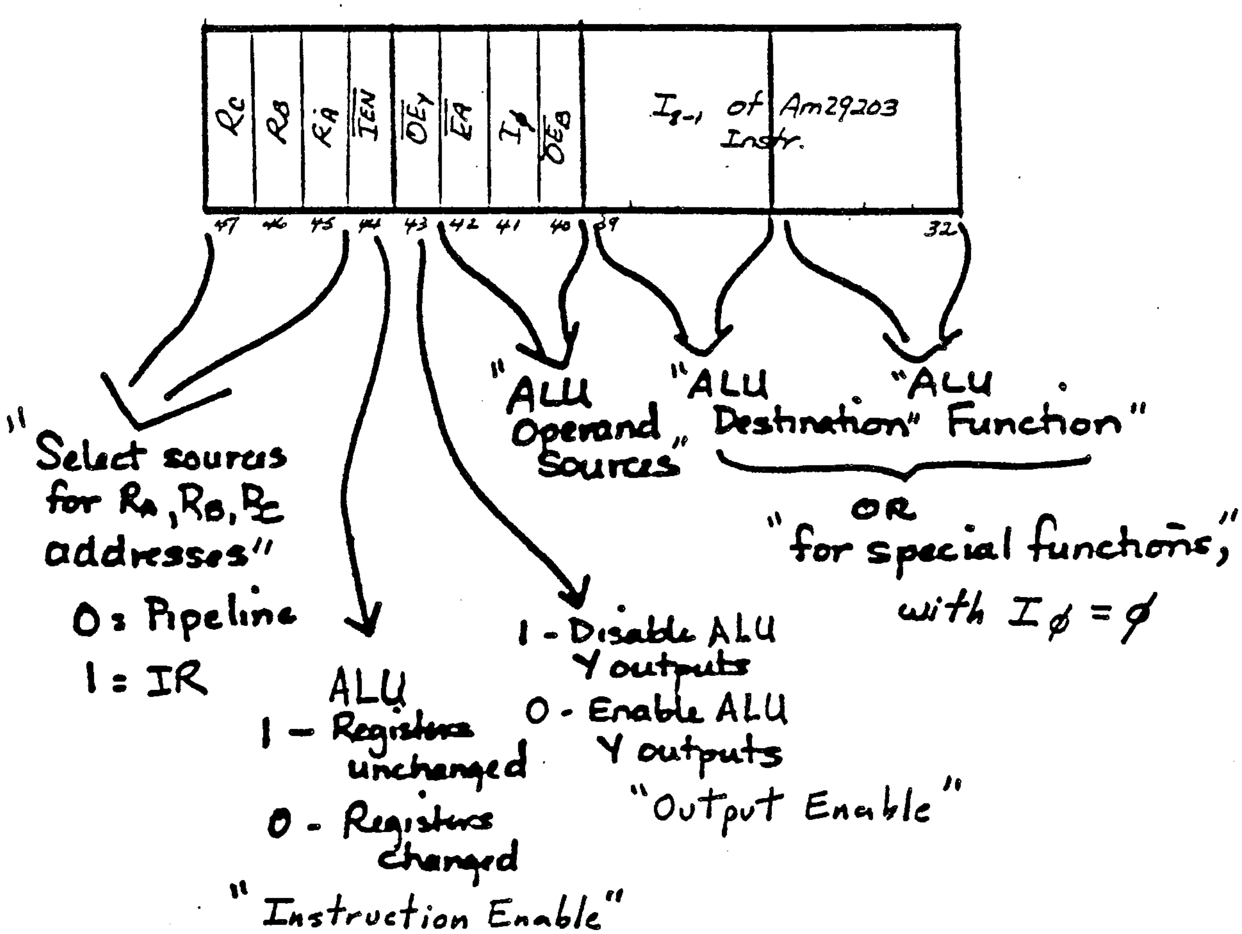
Am2904

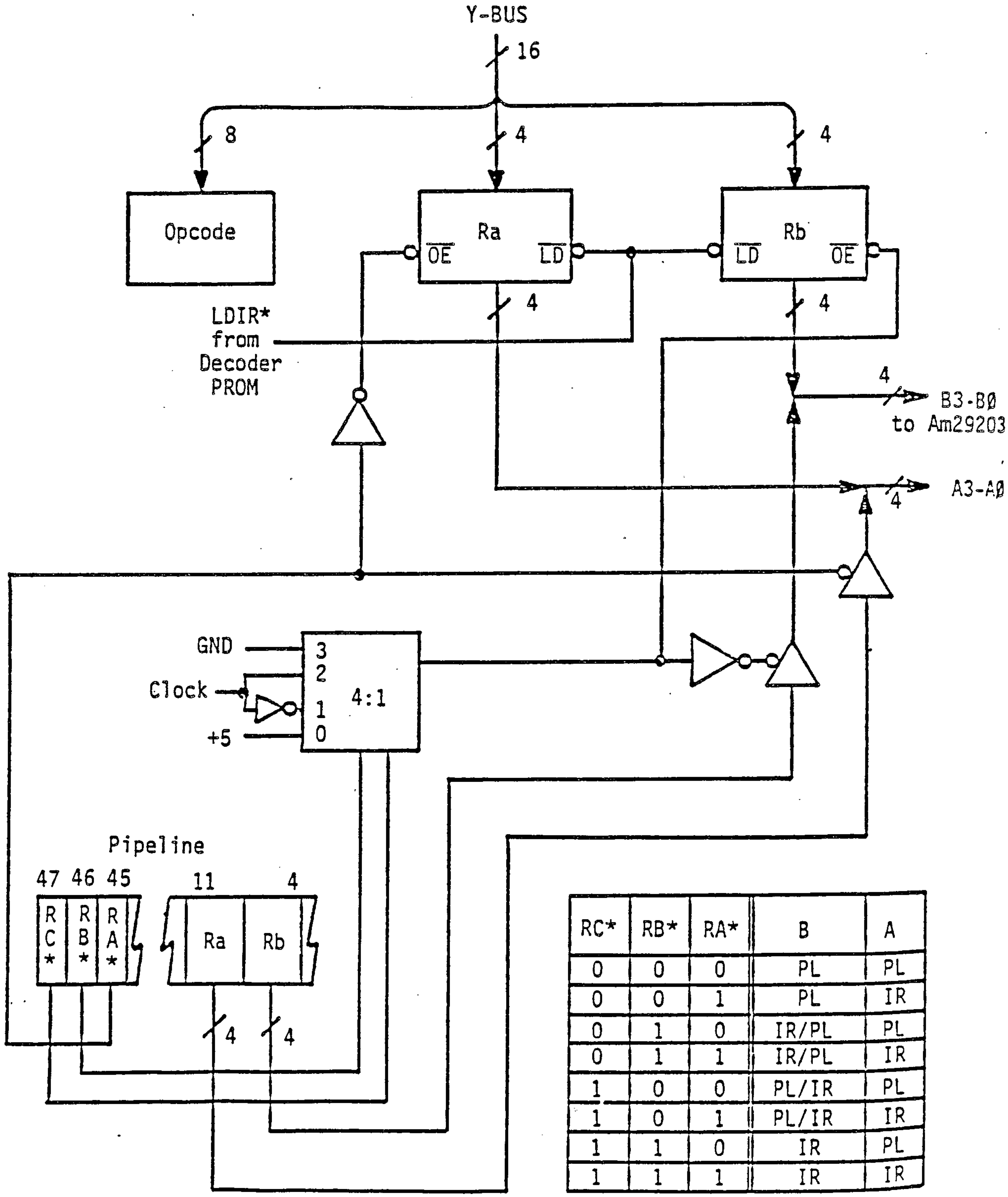
TABLE 7. SHIFT LINKAGE MULTIPLEXER INSTRUCTION CODES.

I_{10}	I_9	I_8	I_7	I_6	M_C RAM Q	SIO_0	SIO_n	QIO_0	QIO_n	Loaded into M_C
0	0	0	0	0		Z	0	Z	0	
0	0	0	0	1		Z	1	Z	1	
0	0	0	1	0		Z	0	Z	M_N	SIO_0
0	0	0	1	1		Z	1	Z	SIO_0	
0	0	1	0	0		Z	M_C	Z	SIO_0	
0	0	1	0	1		Z	M_N	Z	SIO_0	
0	0	1	1	0		Z	0	Z	SIO_0	
0	0	1	1	1		Z	0	Z	SIO_0	QIO_0
0	1	0	0	0		Z	SIO_0	Z	QIO_0	SIO_0
0	1	0	0	1		Z	M_C	Z	QIO_0	SIO_0
0	1	0	1	0		Z	SIO_0	Z	QIO_0	
0	1	0	1	1		Z	I_C	Z	SIO_0	
0	1	1	0	0		Z	M_C	Z	SIO_0	QIO_0
0	1	1	0	1		Z	QIO_0	Z	SIO_0	QIO_0
0	1	1	1	0		Z	$I_N \oplus I_{OVR}$	Z	SIO_0	
0	1	1	1	1		Z	QIO_0	Z	SIO_0	
1	0	0	0	0		0	Z	0	Z	SIO_n
1	0	0	0	1		1	Z	1	Z	SIO_n
1	0	0	1	0		0	Z	0	Z	
1	0	0	1	1		1	Z	1	Z	
1	0	1	0	0		QIO_n	Z	0	Z	SIO_n
1	0	1	0	1		QIO_n	Z	1	Z	SIO_n
1	0	1	1	0		QIO_n	Z	0	Z	
1	0	1	1	1		QIO_n	Z	1	Z	
1	1	0	0	0		SIO_n	Z	QIO_n	Z	SIO_n
1	1	0	0	1		M_C	Z	QIO_n	Z	SIO_n
1	1	0	1	0		SIO_n	Z	QIO_n	Z	
1	1	0	1	1		M_C	Z	0	Z	
1	1	1	0	0		QIO_n	Z	M_C	Z	SIO_n
1	1	1	0	1		QIO_n	Z	SIO_n	Z	SIO_n
1	1	1	1	0		QIO_n	Z	M_C	Z	
1	1	1	1	1		QIO_n	Z	SIO_n	Z	

Notes: 1. Z = High impedance (outputs off) state.
2. Outputs enabled and M_C loaded only if $\overline{SE}$ is LOW.
3. Loading of M_C from I_{10-6} overrides control from I_{5-0}, $\overline{CE}_M$, $\overline{E}_C$.

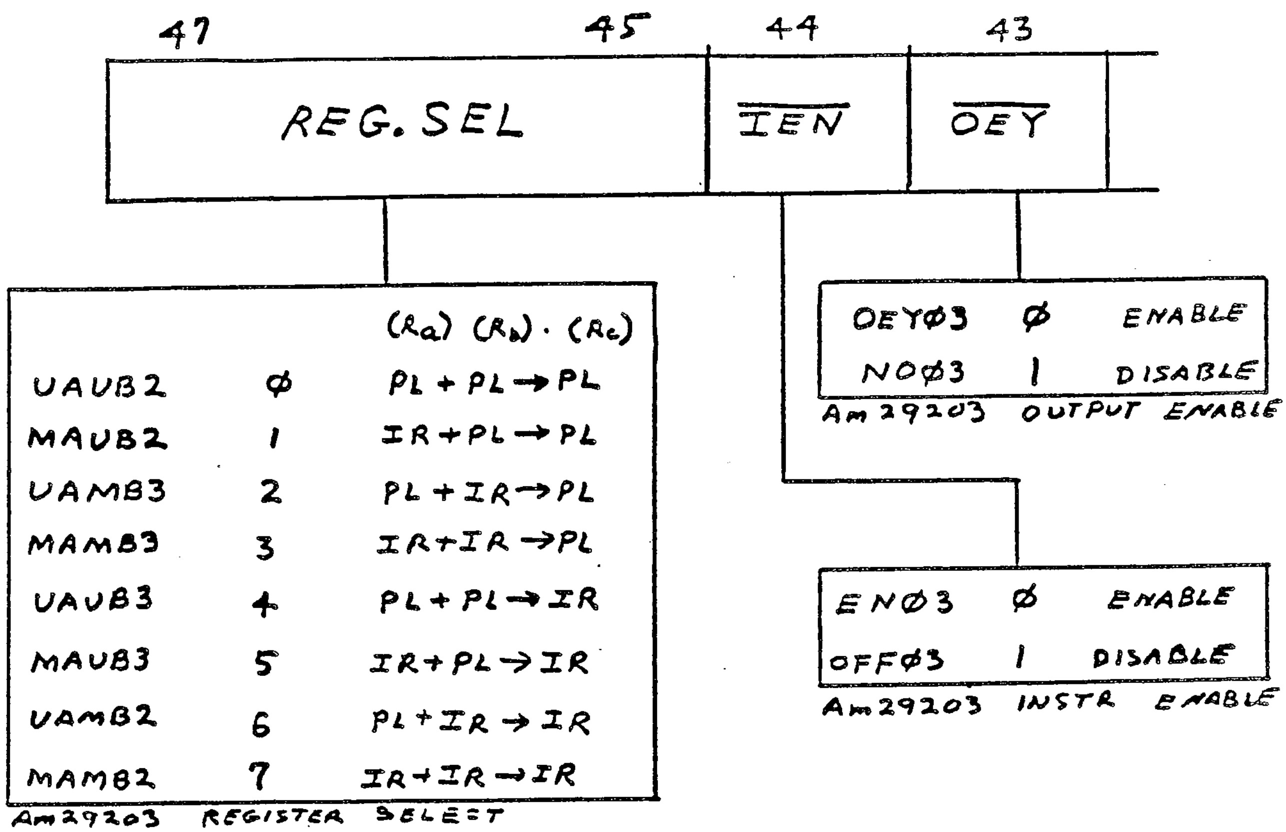
Bits 47-32 The Am29203-ALU control (Four Hex Digits)





RC*	RB*	RA*	B	A
0	0	0	PL	PL
0	0	1	PL	IR
0	1	0	IR/PL	PL
0	1	1	IR/PL	IR
1	0	0	PL/IR	PL
1	0	1	PL/IR	IR
1	1	0	IR	PL
1	1	1	IR	IR

Am29203 Register Address Selection



M - MACRO IR

U - MICROINSTRUCTION (PIPELINE)

A - REGISTER A ADDRESS

B - REGISTER B ADDRESS

2 - TWO ADDRESS OPERATION (Ra + Rb → Rb)

3 - THREE ADDRESS OPERATION (Ra + Rb → Rc)

Am29203 Source Codes & Mnemonics

Mnemonic	42 Ea*,IØ,OEb* 40
RAMAB	Ø
RAMADB	1
RAMAQ	2++
RAMAQ	3++
DARAMB	4
DADB	5
DAQ	6++
DAQ	7++

++ IØ HIGH

39 - 36

| PEST |

DEF File for Am29203 ALU Destinations

```
RAMDA:   EQU   H#0   ;F to RAM, Arith F/2->Y,
RAMDL:   EQU   H#1   ;F to RAM, Log   F/2->Y,
RAMQDA:  EQU   H#2   ;F to RAM, Arith F/2->Y, Q/2->Q
RAMQDL:  EQU   H#3   ;F to RAM, Log F/2->Y  , Q/2->Q
RAM:     EQU   H#4   ;F to RAM, F->Y        ,
QD:      EQU   H#5   ;        , F->Y        , Q/2->Q
LOADQ:   EQU   H#6   ;        , F->Y        , F->Q
RAMQ:    EQU   H#7   ;F to RAM, F->Y        , F->Q
RAMUPA:  EQU   H#8   ;F to RAM, Arith 2F->Y ,
RAMUPL:  EQU   H#9   ;F to RAM, Log 2F->Y   ,
RAMQUPA: EQU   H#A   ;F to RAM, Arith 2F->Y , 2Q->Q
RAMQUPL: EQU   H#B   ;F to RAM, Log 2F->Y   , 2Q->Q
YBUS:    EQU   H#C   ;        , F->Y        ,
QUP:     EQU   H#D   ;        , F->Y        , 2Q->Q
SIGNEXT: EQU   H#E   ;F to RAM, SIO0->Y     ,
RAMEXT:  EQU   H#F   ;F to RAM, F->Y        ,
```

35 — 32

FUNCT

DEF File for Am292Ø3 ALU Basic Functions

```
SUBR:        EQU   H#1   ;F =  S - R - 1 + Cin
SUBS:        EQU   H#2   ;F =  R - S - 1 + Cin
ADD:         EQU   H#3   ;F =  R + S + Cin
INCRS:       EQU   H#4   ;F =  S + Cin
INCRSNON:    EQU   H#5   ;F = .S + Cin

NOTRS:       EQU   H#9   ;Fi = .Ri AND Si
EXNOR:       EQU   H#A   ;Fi =  Ri EXNOR Si
EXOR:        EQU   H#B   ;Fi =  Ri EXOR Si
AND:         EQU   H#C   ;Fi =  Ri AND Si
NOR:         EQU   H#D   ;Fi =  Ri NOR Si
NAND:        EQU   H#E   ;Fi =  Ri NAND Si
OR:          EQU   H#F   ;Fi =  Ri OR Si

; *** The following require that RAMAQ or DAQ be the source:

HIGH:        EQU   H#Ø   ;Fi = HIGH
INCRR:       EQU   H#6   ;F  = R + Cin
INCRNON      EQU   H#7   ;F  = .R + Cin
LOW:         EQU   H#8   ;Fi = LOW
```

39 - 36
DEST

Mnemonics for Am29203 Special Functions

I8-I5 HEX	Mnemonic	Function
0	MULT	Unsigned multiply
1	BCD.BIN	**BCD to binary conversion
1*	MULTIBCD	**Multiprecision BCD to binary
2	TWOMULT	Two's complement multiply
3	DECRMNT	**Decrement by 1 or 2
4	INCRMNT	Increment by 1 or 2
5	SGN.TWO	Sign Magnitude - 2's complement
6	TWOLAST	Two's complement multiply last step
7	BCDDIV2	**BCD divide by two
8	SLN	Single length normalize
9	BIN.BCD	**Binary to BCD conversion
9*	MULTIBIN	**Multiprecision binary to BCD
A	DLN	Double length normalize
A	DIVFIRST	Two's complement divide - first step
B	BCDADD	**BCD add
C	DIVIDE	Two's complement divide-middle step
D	BCDSUBS	**BCD subtract R-S-1+Cin
E	DIVLAST	Two's complement divide - last step
F	BCDSUBR	**BCD subtract S-R-1+Cin

* Requires I4=1

** Not available on Am2903