Advanced
Micro
Devices, Inc.

AmSYS29/10A Microprogram Development System

User's Manual

059920027-002 $15.00

	REVISION RECORD
REVISION	DESCRIPTION
01	Preliminary Issue
08/20/82	
A	Initial Release
1/10/83	

Publication No.
059920027-002

REVISION LETTERS I, O, Q AND X ARE NOT USED

Address comments concerning
this manual to:

ADVANCED MICRO DEVICES
Microcomputer Systems
Publications Department
P.O. Box 453
Sunnyvale, CA 94086

PREFACE

This manual provides complete installation, operation, and interface information for the Advanced Micro Devices AmSYS29/10A* Microprogram Development System. In addition, the manual includes thorough coverage on the theory of operation of the AmSYS29/10A system.

The information in this publication is believed to be accurate in all respects. However, no responsibility is assumed for errors that might appear in this publication. Advanced Micro Devices disclaims responsibility for any consequences resulting from the use thereof. No part of this manual may be copied or reproduced in any form without prior written permission from AMD.

These products are intended for use only as described in this document.

Advanced Micro Devices cannot be responsible for the proper functioning of undescribed features or parameters.

The information presented herein to configure the RAM boards is applicable only to the Am96/1064 board. The Am97/0064 board does not require configuration; if your system uses this board, configuration information is redundant.

Interconnect information and schematic drawings are contained in a separate drawing package.

Detailed information on the AMD Writable Control Store is presented in AMD document 059920003-001, High Speed Writable Control Store User's Manual.

This manual is intended to be used in conjunction with the following AMD documents:

AmSYS29/10 Microprogram Support Software User's Manual 059910514-001

AMDOS 29 Operating System User's Manual 059910500-001

* AmSYS is a trademark of Advanced Micro Devices, Inc.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPER 1
GENERAL INFORMATION

INTRODUCTION

AmSYS29/10A is an engineering tool designed to aid development of microprogrammed target systems. Designers of high-speed computers use bit-slice architecture, like the 2900 family, and microprogramming to optimize processor performance. The microprogrammed processor is more versatile and offers higher performance than the fixed instruction set microprocessor. However, this versatility demands development tools specific to microprogrammed machines.

AmSYS29/10A provides the bit-slice designer with the software development and hardware debug tools which have been available to MOS microprocessor developers. AmSYS29/10A is a Microprogram Development System offering symbolic assembly language development of microcode, high-speed control store emulation, target-system clock control, and logic state monitoring.

During the development cycle of a microprogrammed system, the greatest challenges are writing a microprogram for a user-defined microinstruction format, and debugging the prototype hardware and microprogram in the target environment. AmSYS29/10A supplies the designer with the AMDASM meta-assembler to support microprogram development. The meta-assembler allows definition of a mnemonic instruction set for any microinstruction format. AMDASM, with the language definition table, then assembles the microprogram source file of symbolic code into a binary object file.

During hardware and software integration, AmSYS29/10A greatly reduces debug time by emulating critical parts of the target system. AmSYS29/10A provides a writable control store replacing control store PROM with high-speed RAM. Target system clock control by the clock control logic allows single-step, multiple-step, and full-speed execution with multiple breakpoints. Software is provided to set the RUN address to any location, monitor and display the microinstructions (up to 128 bits) and 32 logic points, and edit the writable control store contents.

A microprogram sequencer is provided on a card as an evaluation vehicle for microprogram familiarization and software module testing. The microprogram sequencer, when combined with the clock control logic and writable control store, forms a complete microprogrammed controller for testing and debugging prototype hardware.

Software provided with AmSYS29/10A includes a disk operating system with a full set of file management commands and input-output routines required for peripheral communications. Additional software is provided for generating, loading, saving, and debugging microprograms. Microcode checkout functions allow the user to display, modify, locate, store, verify, and control target system execution via the CRT console or system mainframe front panel.

An optional high-speed trace unit is available that provides measurements of logic events and allows target system monitoring during full speed debugging. High-speed trace selectively records, under user-specified parameters, up to 48 bits of 256 words at a 10-MHz rate.

Eight 48-bit triggers specify data to be stored, turn data storage on or off, and provide additional hardware breakpoints to the Clock Control Logic (CCL). High-speed trace can be expanded to 96 bits of data storage and 96-bit triggers. Time tags can be stored with each event data storage to record the number of master clock pulses that have occurred prior to the event. Time tags also can be expanded. A standard time tag can be configured with a minimum of 16 bits up to a maximum of 48 bits. The time tag expansion has a 32-bit count.

PHYSICAL DESCRIPTION

The basic AmSYS29/10A consists of the mainframe cabinet and dual floppy disk drive cabinet; optional items include a CRT terminal and printer. Table 1-1 lists the major units, assemblies, and options of the AmSYS29A. Table 1-2 lists physical, environmental, and power specifications for AmSYS29/10A mainframe and associated floppy disk drives.

The AmSYS29/10A mainframe contains the central processor, single/double density floppy disk controller, 64K RAM, writable control store, microprogram sequencer, and clock control logic boards mounted in a 15-slot motherboard. The mainframe cabinet also contains +12Vdc, -12Vdc, and +5Vdc power supplies. Cooling for the cabinet is provided by two fans that move the air across the cards and power supplies.

The system mainframe front panel, figure 1-1, contains control switches, indicators, and connectors for the system processor and microprogrammed controller sections of the system mainframe.

The system mainframe rear panel contains five 25-pin connectors and one 50-pin connector for attaching peripherals to the system, and a master power switch. Table 1-3 lists the rear connector ports and addresses. Figure 1-1 shows the rear panel and connectors.

Table 1-1. AmSYS29/10A Units, Assemblies, and Options

BASIC SYSTEM	Support Processor:	CPU card with 4 serial ports (RS232), 3 parallel 8-bit ports
		System memory card, 64K bytes
		Single/double density floppy disk controller
		5V, 75A switching power supply
		12V, 1.7A linear power supplies
		Dual floppy disk drives (housed in a separate cabinet)
	Microprogrammed Controller:	Writable control store card, 1K x 64 bits. The system is pre-wired for 4 cards.
NOTE Equipment can be factory-equipped 115V/60Hz, 220V, 50Hz, 110V/50Hz, or 240V/50Hz.		Microprogram sequencer with on-board 256 X 12 mapping RAM
		Instrumentation (clock control logic) card, 10 MHz
		Blank diskettes (2)
Software:	AMDOS 29	Disk Operating System
	Microprogram Generation Software:	AMDASM microprogram assembler
		AMSCRM, AMPROM post-processing programs
		PROM programmer software (for Data I/O Model 7, 9, 17 or 19 and Pro-Log M900 or 920).
		AMMAP PROM mapping assembler
	Microprogram Support Software:	Loads, verifies, saves, and restores programs from diskette to either WCS or map RAM
		DDT29 debugger for the microcode in WCS: displays address, data and 20 monitor bits (single level trace); strings commands; single/multiple steps; break point on address, data, or control bits; jams address; displays and stores data in mapping RAM.
NOTE The software is licensed.		Am9080A software (relocatable assembler and dynamic debugger including disassembler and trace capability)
PERIPHERALS Option Numbers:	8/8210	Console, 12 inch, 24 lines x 80 characters, CRT display, keyboard
	*	Line printer,220 lines/minute, 132 columns. Serial interface
	*	Dot matrix character printer: 7 x 7 dot matrix print head, 120 CPS print speed. Parallel interface
CARDS	29/2064	Writable control store (WCS) card, 1K x 64 bits, 35 ns access
	290106 **	Writable control store (WCS) card, 1K x 64 bits, 60 ns access
	29/9040	Instrumentation (Clock Control Logic CCL) card, 10 MHz sys clk
	990015 **	Instrumentation (Clock Control Logic CCL) card, 5 MHz sys clk
SERVICES	29/6310	High speed trace
	29/6312	Trace buffer card, expands high speed trace to 96 bits; includes 6 8-bit data pods and 1, 2-clock input clock pod. Trace time tag expansion card adds 256 x 32 bits of time tag storage to 96-bit high speed trace; includes clock pod.
	8/8250	Diskettes, package of 10 - blank
	8/8653	PROM programmer cable, for connecting AmSYS29 to a Data I/O Model 7, 9, 17, or 19
	8/8652	PROM programmer cable, for connecting AmSYS29 to PRO-LOG Model 900 or 920

* Peripheral not included. System includes drivers that will support this peripheral.
** Obsolete, listed for reference only. Supported by AmSYS29/10A drivers.

Table 1-2. Power Requirements

All +/- 10%	110V/60Hz	110V/50Hz	220V/50Hz	240V/50Hz
Mainframe	4.00A	4.00A	2.00A	2.00A
CRT Console	0.50A	0.50A	0.25A	0.25A
Floppy Disk Drive	1.50A	1.50A	0.75A	0.75A
Frequency	±5Hz	±5Hz	±5Hz	±5Hz
Circuit Breakers	10A	10A	10A	10A
Fuses				
F102 (spare)	3A SB	3A SB	1.5A SB	1.5A SB
F103 (PS C)	3A SB	3A SB	1.5A SB	1.5A SB
F104 (PS B)	3A SB	3A SB	1.5A SB	1.5A SB
F105 (PS A)	3A SB	3A SB	1.5A SB	1.5A SB
F106 (Fans & 12V PS)	2A FB	2A FB	1.0A FB	1.0A FB
Floppy Disk Drive	3A SB	3A SB	1.5A SB	1.5A SB
CRT Console	1A SB	1A SB	0.5A SB	0.5A SB

SB = Slow-blow FB = Fast blow

Table 1-3. AmSYS29/10A Ports and Addressing

Originating Card/Plug	Port Addr. (hex)	Type and Baud Rate	Rear Panel Connector	Device
CPU P5	60,61	Serial 9600	J11	CRT Console
CPU P5	64,65	Serial 9600	J12	Serial Printer
CPU P5	68,69	Serial 600	J13	PROM Programmer
CPU P5	6C,6D	Serial 110/9600*	J14	Spare
CPU P4	70,71, 72,73	Parallel	J10	Parallel Printer
FDC P4**		Parallel	P17	FDD Subsystem

*Depending upon jumper configuration as shown in table 2-1.
**CPU I/O ports 7C, 7D, 7E, 7F are used to communicate with the floppy disk controller board.

FUNCTIONAL DESCRIPTION

The AmSYS29/10A development system supports microprogram development and verifies operation of target system hardware in the microprogrammed environment. Application of the AmSYS29/10A development system begins with definition of the microinstruction format for the target system. Each section of a proposed system is controlled by parallel control lines that form a formatted microinstruction or control word. A microprogram is a series of microinstructions whose changing bit patterns control operation of the proposed target system. Each field of microinstruction represents different control functions, constants, or variable data.

AMDASM is a meta-assembler program that is used to define the microword length and mnemonics for functions, addresses, and the operation control fields in the microinstruction. These definitions are stored on disk by AMDASM in the form of a lookup table. This table defines the assembly language for that particular microinstruction set. A microcode program can then be written in this newly-defined assembly language and assembled by AMDASM to produce an object-code file on disk.

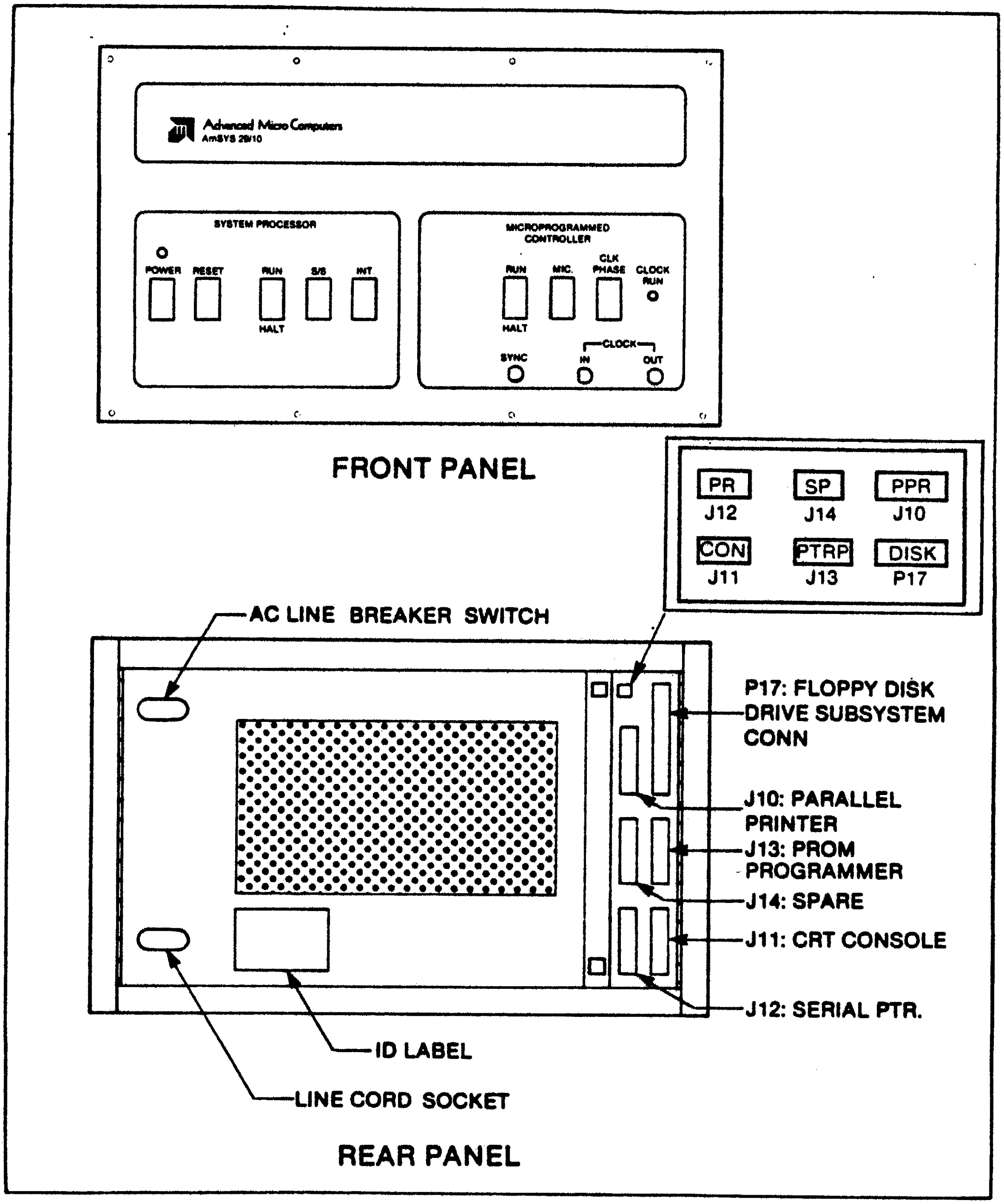


Figure 1-1. AmSYS29/10A Controls and Indicators

The microprogrammed controller integrates the microcode object file into the target system hardware. The writable control store replaces the target system microcode store, which is usually high-speed ROM,with RAM. LBPM (Load Bipolar Memory) is used to load the microcode object file created by AMDASM into the writable control store. DDT29 (Dynamic Debug Tool) interfaces the clock control logic, which controls the target system clock, to the user console, thus providig clock, addressing, and breakpoint control within the target system. The system can single-step through the microprogram and monitor the microinstructions, microaddress, and target system operations on the system console. Errors in microcode can be corrected immediately from the console. As the confidence level in the microprogram increases, full-speed testing can proceed in the target system under breakpoint control. The optional high-speed trace provides logic state monitoring of the target system during full-speed operation.

When the microprogram has been fully debugged, it is reassembled using AMDASM to create a new object file. The new object code is reloaded into writable control store and verified. The debugged object file is then broken into PROM-size files, via the program AMPROM, and burned into high-speed PROMs, via the program PPROG or PLPROG. These PROMs are placed into the target system. Testing of this final configuration and subsequent production line testing of the microprogrammed system can also be accomplished by the AmSYS29/10A with high-speed trace option installed.

HARDWARE FEATURES

AmSYS29/10A is divided into three main areas of support hardware: the system processor, microprogrammed controller, and optional high-speed trace. Figure 1-2 is a system block diagram of the AmSYS29/10A.

The system processor comprises a CPU card, 64K memory card,single/double density dual floppy disk drive cabinet, and peripheral support options. These options include the CRT console, line printer,and PROM programmer. The microprogrammed controller consists of the writable control store cards, microprogram sequencer card, and clock control logic. The optional high-speed trace unit forms a part of the microprogrammed controller and consists of a trace control and trace buffer board.

The mainframe cabinet also contains power supplies for both the system processor and microprogrammed controller. Three +5 Vdc power supplies at 25A each and a +/-12 Vdc power supply at 1.7A provide the system power.

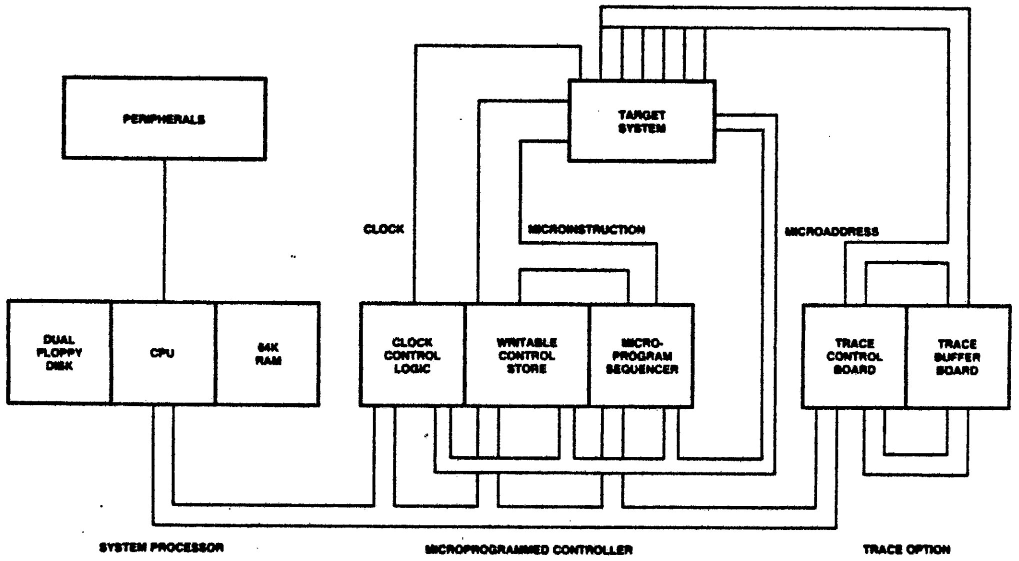


Figure 1-2. AmSYS29/10A Block Diagram

CPU CARD

The CPU card is located in the system mainframe. The principal circuits on the CPU card are the Am9080A microprocessor together with its support circuits, bus drivers, receivers, and system clock. In addition to the microprocessor, the CPU card contains circuits to accomplish seven other functions:

- Seven input/output ports
- Floppy disk interface circuits (DMA)
- Microprogrammed controller interface
- Initialization
- Bus control
- Clock control
- Priority interrupt

The input/output ports are the means by which the CPU communicates with the peripheral devices. There are four serial ports and three general-purpose 8-bit parallel ports.

Serial ports communicate with the CRT terminal at 9600 baud, a line printer at 9600 baud, and either a paper-tape reader/punch or a PROM programmer at 600 baud. The fourth port is a spare, factory-configured to 9600 baud but user configurable by a jumper to 110 baud. Each serial port has an Am9551 universal synchronous-asynchronous receiver-transmitter (USART), which serializes outgoing data and deserializes incoming data, simultaneously if desired. The USART also keeps track of its own operational status, so that the microprocessor can dispatch a new character to the USART immediately when the latter has finished serially transmitting the preceding character, and can accept an incoming character as soon as the USART has deserialized it.

General-purpose parallel ports consist of a single Am9555 programmable peripheral interface, which is user accessible through the connectors at the top of the CPU card. The user must provide the interface circuitry at the terminals of the Am9555. This is shipped configured to interface to a Centronics-compatible printer. The CPU card has sockets for installing interface circuits.

The floppy-disk interface sets-up direct memory access operations in the appropriate direction and keeps the microprocessor in the wait state while a DMA operation is in progress. Data transferred to or from the disk does not involve the microprocessor. There are two disk drives; the disk on drive A typically contains the system software, and that on drive B is a mass storage area.

During a DMA operation or initialization, the system bus must be kept available at all times so that a data byte can be transmitted to or from memory without notice. Therefore, other elements of the CPU card must be prevented from using the bus at such times. The bus control enforces this prohibition.

Using the microprogrammed controller interface, the CPU issues commands to receive status information from the prototype system and its microprogram.

The clock control implements the operation designated by pressing the front panel RUN or SINGLE STEP switch.

The priority interrupt ranks interrupt requests from four different sources and feeds them in order of rank to the microprocessor for its response. Interrupts 5 and 7 are used currently. Interrupt 5 is the instrumentation clock stopped interrupt and 7 is the system interrupt.

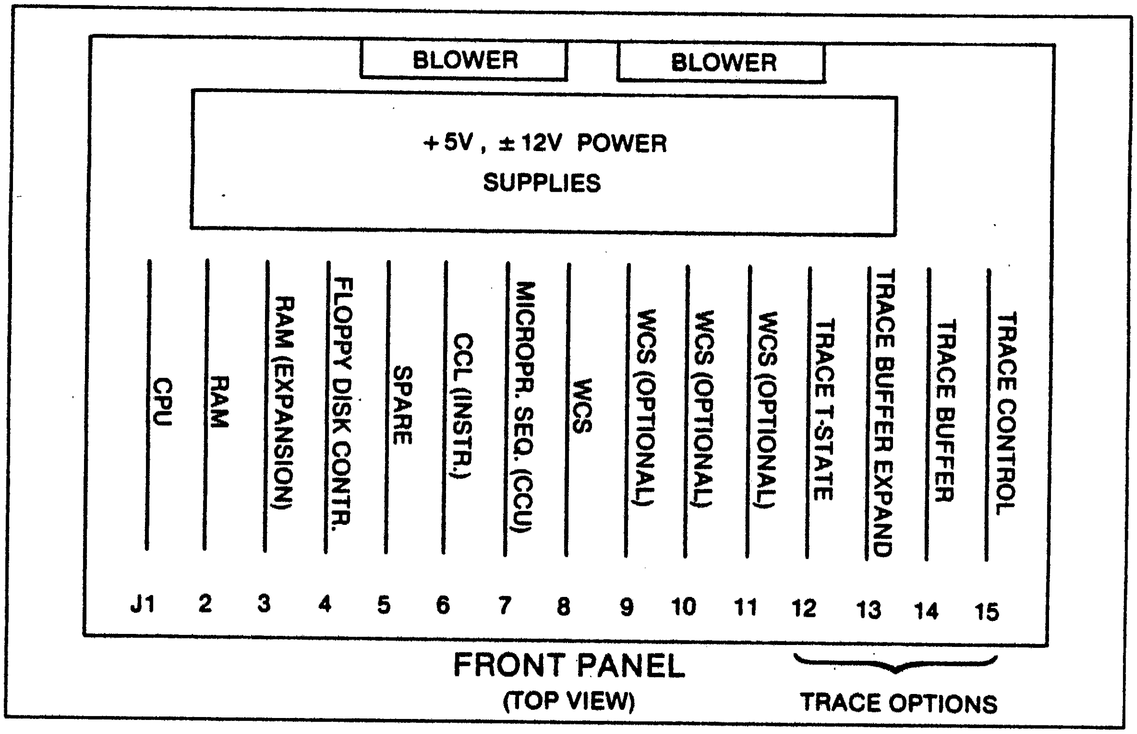


Figure 1-3. System Mainframe Configuration

MEMORY CARD

The memory card is located in the system mainframe and contains 64K bytes of dynamic memory with on-board refresh. The card contains 32 dynamic RAM integrated circuits, each storing 16,384 bits in a 16,384 x 1 format. Eight of these circuits hold the corresponding bits of 16,384 bytes and are addressed simultaneously.

SINGLE/DOUBLE-DENSITY FLOPPY DISKS

The dual disk drive cabinet contains two independent eight-inch floppy disk drives (drive A and drive B) and associated power supply. The dual-density floppy disk controller, located in the mainframe, provides the interface between the CPU card in the system mainframe and two floppy disk drives. The controller contains write protect logic, CRC generation and verification logic, and automatic track seek-verify logic in addition to the read/write circuits. Each disk drive uses IBM-compatible* soft-sectored double-density eight-inch diskettes. Each diskette has a maximum memory capacity of 512K bytes. The controller also allows single density diskettes to be used. Each drive has a front panel activity indicator that is lighted when the associated disk drive is active.

* IBM media specification GA21-9257-1

The CRT console is a full-duplex ASCII data terminal. Data is exchanged between the CRT console and the system mainframe via a 9600-baud RS232 port. The CRT console includes keyboard, CRT monitor, associated logic circuits, and power supply. Cursor addressing is provided to allow updating screen pages without scrolling; this capability is required when using high-speed trace.

LINE PRINTER (OPTION)

The line printer option is no longer offered.

PROM PROGRAMMER (OPTION)

The AmSYS29/10A operating system includes routines to drive a PROM programmer. The PROM programmer is connected to the 600-baud serial I/O port of the AmSYS29/10A mainframe. Support is available for various models of Data I/O and Pro-Log PROM programmers.

WRITABLE CONTROL STORE CARD

The writable control store card is configured as 1K x 64 bits of RAM which provides the target system microcode control store. The RAM has a maximum access time of 35 nanoseconds over the commercial temperature range. The mother board contains slots for four writable control store cards, allowing the basic writable control store supplied with the system to be expanded to 4K words in depth and 64 bits in width or 2K words by 128 bits width. The card provides an eight-bit interface to the system bus that allows the user to examine and update the contents.

CLOCK CONTROL LOGIC CARD

The clock control logic (CCL) card provides console control of the target system during emulation. The card takes one slot in the motherboard and communicates with the system processor over the system P2 bus. Control of the target system is accomplished via the microprogram address bus and clock control lines. The clock control logic card allows the user to single step one clock cycle, microstep one instruction cycle, multiple step, or run full speed emulation. The clock control logic also allows the user to set a breakpoint address, force a jump to a new address, monitor 32 logic points (including 12 address lines), and set a halt from the front panel, hardware, or software.

HIGH-SPEED TRACE CARD (OPTION)

The high-speed trace option allows monitoring of target system logic states during full-speed emulation. High-speed trace allows the user to selectively record, at a 10-MHz rate, up to 48 bits for a depth of 255 words. Eight 48-bit-wide general-purpose comparators are provided. These comparators allow the user to set a trigger sequence, to specify the data to be stored, and to provide two additional hardware breakpoints to the clock control logic. Time tags can be stored with each occurrence of the specified data storage to record the time (in increments of master clock pulses) of the event. Time tag storage uses 16 or 32 bits of data in the basic configuration. The high-speed trace unit can be expanded to 96 bits of data storage by the addition of an optional memory buffer board. This option also expands the eight trigger comparators from 48 bits wide to 96 bits wide.

MICROPROGRAM SEQUENCER CARD

The microprogram sequencer card provides a complete pipelined computer control unit that includes an address sequencer and an op code mapping memory for addressing the microcode contained on the writable control store (WCS) cards. The op code mapping memory is a 256 by 12-bit RAM that is used to store the starting addresses of microcode subroutines. The op code mapping memory translates an 8-bit instruction fetch from main memory into a starting address for the WCS microcode.

The microprogram sequencer card can be automatically paged into the upper 32K address range of the system processor by invoking the appropriate microprogramming support software command. Once paged into the system processor, the op code mapping memory can be loaded, verified, modified, and saved as required.

The card also contains a branch control unit, a pipelined register, and a next address controller. The next address controller determines the source of the next microinstruction to be executed, routing to the microprogram address sequencer, the output of the mapping memory, the pipeline register, the sequencer program counter, or the sequencer stack. The pipeline register stores 26 bits from the writable control store, and permits the address of the next microinstruction to be determined while the previous instruction is being executed by the target. In addition, this board provides 16 bits for conditional branching, controlled by external signals provided by the user, and four bits for multiway branching to one of 16 addresses.

The AmSYS29/10A system provides a complete software package that consists of a comprehensive disk operating system, microprogram generation software, microprogram support software, and example files.

OPERATING SYSTEM

AMDOS29 is a CP/M-compatible disk operating system that performs file management of floppy disk files and contains all input/output routines required for peripheral communications. AMDOS29 provides the basic capability to load programs for execution and service system requests made by executing programs. When the operating system has been loaded by the bootstrap operation, the command processor, disk I/O drivers, and device I/O drivers are resident in high memory. The operating system is then ready to load and service programs. The operating system is fully described in the AMDOS29 Operating System Manual. Functionally, each command issued at the console takes temporary control of the system and performs the appropriate operation. Except for intrinsic commands, each command exists as a separate file that is loaded into the transient program area (TPA)for execution.

File management utilities included in AMDOS29 are:

DIR	Display file names.
PIP	Copy specified file or files.
CPYDSK	Copy all files on a diskette.
TYPE	Display file contents.
DISPL	Display selected file contents.
DUMP	Dump file in hexadecimal and ASCII.
ERA	Erase file.
SAVE	Save memory contents as a file.
REN	Rename file.
FORMAT	Initialize a diskette by formatting.
SYSGEN	Copy the operating system onto tracks 0 and 1.

The utility for submitting a job is:

SUBT/XSUB	Submit command stream for processing.

Utilities for resource management are:

VFU	Printer control.
STAT	Display file and option characteristics.
AMUSRTTY	User-created TTY driver loaded with BIOS at boot.

Utilities for program generation are:

ED	File editor to create and update ASCII files.
MAC	8080/8085/Z80 relocatable macroassembler.
PREHEAT	Prepares assembler object files (AMDASM/MAC) for PROM programmer drivers.
LINK	Links relocatable object files produced by MAC.
LIB	Forms library of relocatable files produced by MAC.
PPROG	Data I/O Models 17 or 19 PROM programmer driver

PLPROG	Pro-Log M900 & M920 PROM Programmer driver.
DDT	Dynamic debug tool to debug MAC programs.

For complete details see AMDOS 29 Operating System User's Manual.

MICROPROGRAM GENERATION SOFTWARE

AMDASM is a meta-assembler that assembles user-generated prototype microcode into machine language. AMDASM is a two-phase assembler. The first phase is a definition phase that allows the user to define the microword length, constants, and formats used to write source programs for the prototype system. The second phase is a two-pass assembler. In the first pass, the source program statements are read and values are assigned to labels and constants. In the second pass, the source program executable statements are translated by the definition phase output into binary format for prototype execution. The following is a list of AMDASM programs.

AMDASM Phase 1 (definition) of meta-assembler that permits user to define microword length, constants, and formats used to write source programs for prototype systems.

Phase 2 (assembly) of meta-assembler that reads source programs and assigns labels and constants. Source program executable statements are translated into binary format for prototype execution of the user-developed microcode.

AMPROM Post processor that permits conversion of binary object code to a form corresponding to the PROM organization of the developed system. Word width must be consistent with the width of the developed system.

AMSCRM Post processor that permits the user to reassign bit positions of microword contents by specifying source and destination bit positions and the length of each field to be moved, thus producing a reorganized microcode object file.

AMMAP Generates microprogram entry point addresses that are loaded into the MPS mapping RAM to decode instructions and generate microprogram starting addresses.

MICROPROGRAM SUPPORT SOFTWARE

The microprogramming debug software contains a number of programs that support the prototype microcode development. The debug programs are described as follows:

LBPM Loads microcode from disk file into WCS card memory or loads microcode entry points from disk file into MPS card op code mapping RAM.

VBPM Verifies loaded contents of WCS card memory or of MPS card mapping RAM.

SBPM Saves WCS card memory contents or MPS card op code mapping RAM contents on a disk file.

RBPM Restores file previously saved on diskette to WCS card memory.

DDT29 Dynamic debugging tool that permits display and modification of portions of WCS card memory or MPS card opcode mapping RAM. Controls halt/run states of microprogrammed controller clock. Loads address breakpoints and addresses for jamming into Clock Control Logic card registers. Permits display of Clock Control Logic card status register. Stores data in one-byte increments into particular locations of WCS card or MPS card memories. Permits 20 monitor bits (single level trace) to be displayed. Allows automatic stepping through the microcode and the convenient display of relevant data (microcode address and data,and monitor bits).

For complete details see AmSYS29/10 Microprogram Support Software User's Manual.

EXAMPLE FILES

The following example files are included to allow the user to become familiar with definition files, source files, execution, and debugging. Most files have comments on their headers which describe the purpose of the file.

ASMDEMO.SUB — This submit file, when executed with the command SUBMIT ASMDEMO JUMP, will display the JUMP.DEF and JUMP.SRC files, execute AMDASM, then display the print files. SUBMIT and DISPL utilities should be studied first.

DDTDEMO.SUB — This submit file loads the small program generated from ASMDEMO.SUB, and executes and displays some DDT29 commands.

JUMP.DEF — A definition file for the above demonstration programs.

JUMP.SRC — A source file that is assembled and executed by the above submit files.

AM29CPU.DEF — A definition file for the AM2900 family.

AM29203.SRC — A sample source file for the AM29203.

CONTROLR.DEF — A definition file for the AM29116.

CONTROLR.SRC — A sample source file to exercise the CONTROLR.DEF statements and demonstrate their usage.

DISKCTLR.DEF — An AM29116 definition file. This file uses the same AM29116 mnemonics and instruction layout as the CONTROLR.DEF file; however, the invocation of the instruction is different and worthy of study.

DISKCTLR.SRC — A source file for a high-performance disk controller that uses the DISKCTLR.DEF file.

CHAPTER 2

INSTALLATION

INTRODUCTION

This chapter provides instructions for initial inspection, preparation for use, peripheral device interconnection, and initialization of the AmSYS29/10A support processor. For detailed information on the individual peripheral devices, refer to the appropriate manual for that device.

UNPACKING AND INSPECTION

Inspect the shipping cartons immediately for evidence of damage or mishandling during transit. When there is evidence of severe damage, request the carrier's agent to be present when the carton is opened. If the carrier's agent is not present, retain the carton and all shipping materials for the agent's inspection. Report all shipping damage to the carrier immediately.

NOTE

> If there is evidence of damage to the equipment, DO NOT attempt any servicing of the equipment. Any attempt to service the equipment, without the express consent of the Advanced Micro Devices Field Service Manager, voids the warranty.

When all equipment is removed from the shipping cartons and it is ascertained that there is no obvious shipping damage, inventory the equipment received against the enclosed packing slip. If any discrepancies are noted, notify Advanced Micro Devices immediately. It is suggested that all salvageable shipping materials be retained in case the product must be shipped in the future.

PREPARATION FOR USE

Prior to interconnecting the AmSYS29/10A units and applying power to the mainframe, proceed as follows:

1. Check line input power and ratings on floppy disk drive and mainframe to ensure supplied power is correct.

2. Open top and bottom access covers on mainframe cabinet.

3. Remove packing material.

4. It is normally not necessary to check DC voltages during installation. However, if voltage checking is desired, use a DVM (or equivalent device) to check the connector pins shown in figure 2-1.

5. Verify that board jumpers and switches are set per table 2-1.

6. Ensure that all boards are well seated and in their assigned slots on the motherboard, as shown in figure 2-2.

7. Ensure that ribbon cables are connected to correct board edge connectors shown in figure 2-2.

8. Unpack and set up peripheral devices.

9. Connect CRT and floppy disk drives to system main frame cabinet back-panel with cables provided. See figure 2-3 for proper location. Pin assignments for back-panel connectors are shown in tables 2-2 through 2-7.

10. Connect AC power to the AmSYS29/10A mainframe, CRT, printer, and floppy disk drives.

11. Ensure that processor front-panel RUN/HALT switch is in the RUN position.

NOTE

Appendix A contains a demo program available on the supplied system disks. This program can be used to verify system operation.

Table 2-1. Board Jumper and Strapping Information

Jumpers and switches on floppy disk controller board (6120)

Jumpers :	3 and 8
Switch 1:	all open (off)
Switch 2:	1 open; 2-6 closed (on)

Strapping of 64K RAM (1064) board:

2 to 3	38 to 39	55 to 56	63 to 64	88 to 95
17 to 18	43 to 44	57 to 58	65 to 66	92 to 93
31 to 32	46 to 48	59 to 60	67 to 68	*135 to 133
34 to 35	53 to 54	61 to 62	88 to 95	*135 to 139

*Note: 135 is connected to both 133 and 139.
97/0064 board requires no user configuration.

Strapping of CPU board:

2-3	110 baud on spare I/O port
1-2	9600 baud on spare I/O port

Table 2-2. J10 (Parallel Printer)

PIN	SIGNAL	PIN	SIGNAL
1	DATA1	10	GROUND
2	DATA2	11	GROUND
3	DATA3	12	GROUND
4	DATA4	13	NC
5	DATA5	14	DATA STROBE
6	DATA6	15	GROUND
7	DATA7	16	GROUND
8	DATA8	17	BUSY
9	GROUND	18-25	NC

Table 2-3. J11 (CRT Terminal)

PIN	SIGNAL
1	NC
2	RX DATA
3	TX DATA
4	NC
5	REQUEST TO SEND
6	DATA TERMINAL READY
7	GROUND
8-25	NC

Table 2-4. J12 (Serial Printer)

PIN	SIGNAL
1	NC
2	RX DATA
3	TX DATA
4	CLEAR TO SEND
5	REQUEST TO SEND
6	DATA TERMINAL READY
7	GROUND
8-19	NC
20	DATA SET READY
21-25	NC

Table 2-5. J13 (PROM Programmer Or Reader/Punch)

PIN	SIGNAL
1	NC
2	RX DATA
3	TX DATA
4	NC
5	REQUEST TO SEND
6	DATA TERMINAL READY
8-19	GROUND
20	DATA SET READY
21-25	NC

Table 2-6. J14 (Spare)

PIN	SIGNAL
1	NC
2	RX DATA
3	TX DATA
4	CLEAR TO SEND
5	REQUEST TO SEND
6	DATA TERMINAL READY
7	GROUND
8-19	NC
20	DATA SET READY
21-25	NC

Table 2-7. P17 (Floppy Disk Drive)

PIN	SIGNAL NAME	PIN	SIGNAL NAME
2	TC 43*	1	GROUND
4	---------------	3	GROUND
6	---------------	5	GROUND
8	---------------	7	GROUND
10	TWO SIDED*	9	GROUND
12	---------------	11	GROUND
14	SIDE SELECT*	13	GROUND
16	IN USE*	15	GROUND
18	HEAD LOAD*	17	GROUND
20	INDEX*	19	GROUND
22	READY*	21	GROUND
24	---------------	23	GROUND
26	DS01*	25	GROUND
28	DS02*	27	GROUND
30	DS03*	29	GROUND
32	DS04*	31	GROUND
34	DIRECTION*	33	GROUND
36	STEP*	35	GROUND
38	WRITE DATA*	37	GROUND
40	WRITE GATE*	39	GROUND
42	TRACK DATA*	41	GROUND
44	WRITE PROTECT*	43	GROUND
46	READ DATA*	45	GROUND
48	---------------	47	GROUND
50	---------------	49	GROUND

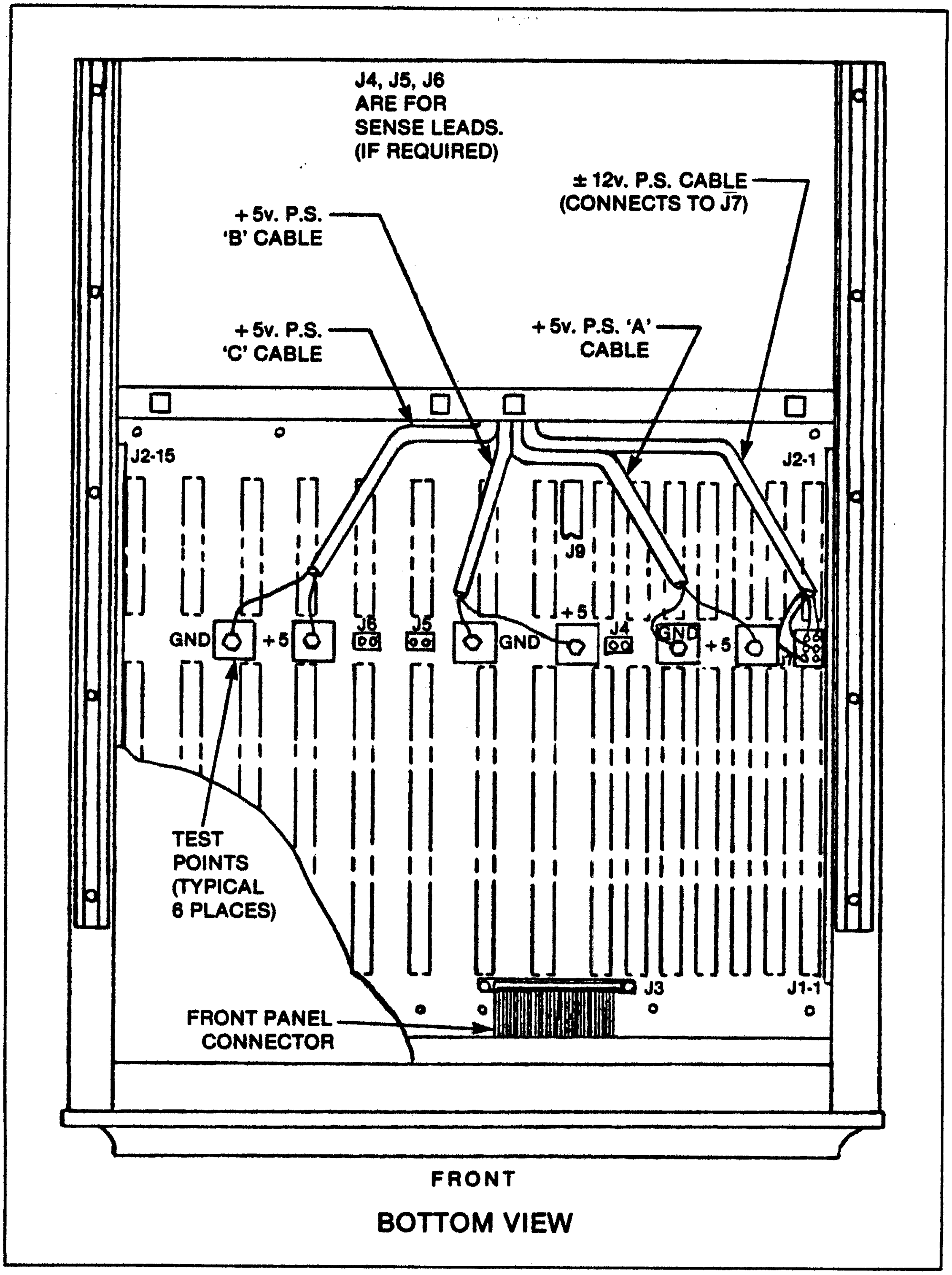
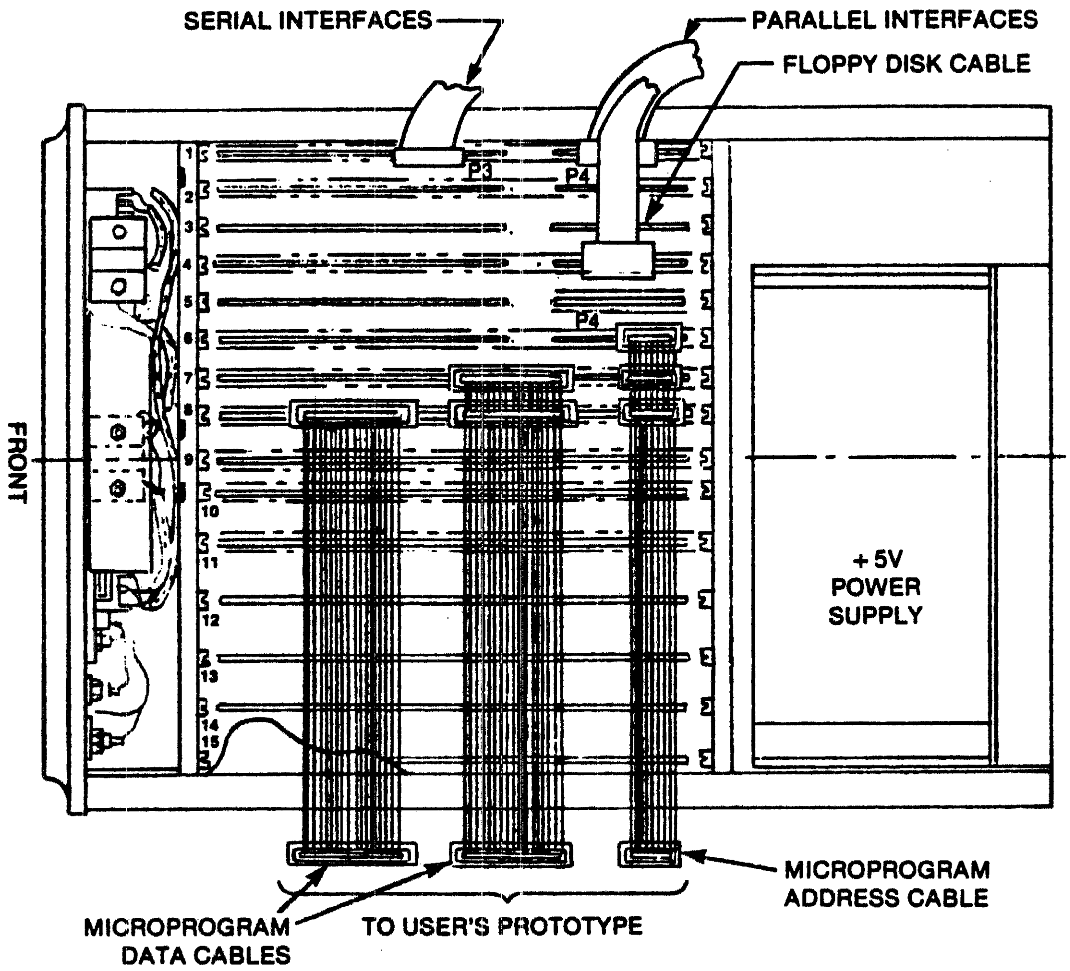


Figure 2-1. Voltage Test Points



INSTALLATION CHART	
SLOT	BOARD DESCRIPTION
1	CPU
2	Memory
3	Memory Expansion
4	Floppy Disk Controller
5	Spare
6	Clock Control Logic CCL (Instrumentation)
7	Microprogram Sequencer (Computer Control Unit)
8	Writable Control Store (WCS)
9	WCS Expansion
10	WCS Expansion
11	WCS Expansion
12	Trace Option
13	
14	
15	

Figure 2-2. Board Edge Connectors

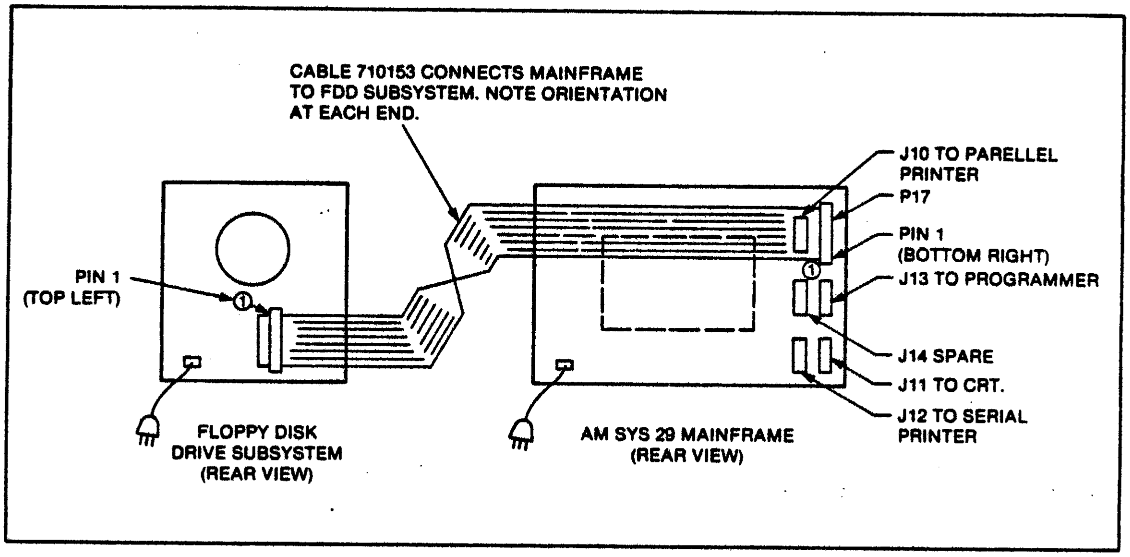

CABLE 710153 CONNECTS MAINFRAME TO FDD SUBSYSTEM. NOTE ORIENTATION AT EACH END.
J10 TO PARELLEL PRINTER
P17
PIN 1 (BOTTOM RIGHT)
J13 TO PROGRAMMER
PIN 1 (TOP LEFT)
1
1
J14 SPARE
J11 TO CRT.
J12 TO SERIAL PRINTER
FLOPPY DISK DRIVE SUBSYSTEM (REAR VIEW)
AM SYS 29 MAINFRAME (REAR VIEW)

Figure 2-3. AmSYS29/10A Interconnection Diagram

CHAPTER 3
OPERATION

INTRODUCTION

Read the descriptions of diskette insertion and system bootstrap operation before proceeding with system initialization.

DISKETTE INSERTION

Diskettes must be inserted into the dual-drive floppy disk unit openings with label up and the write protect notch toward the floppy disk drive. See figure 3-1.

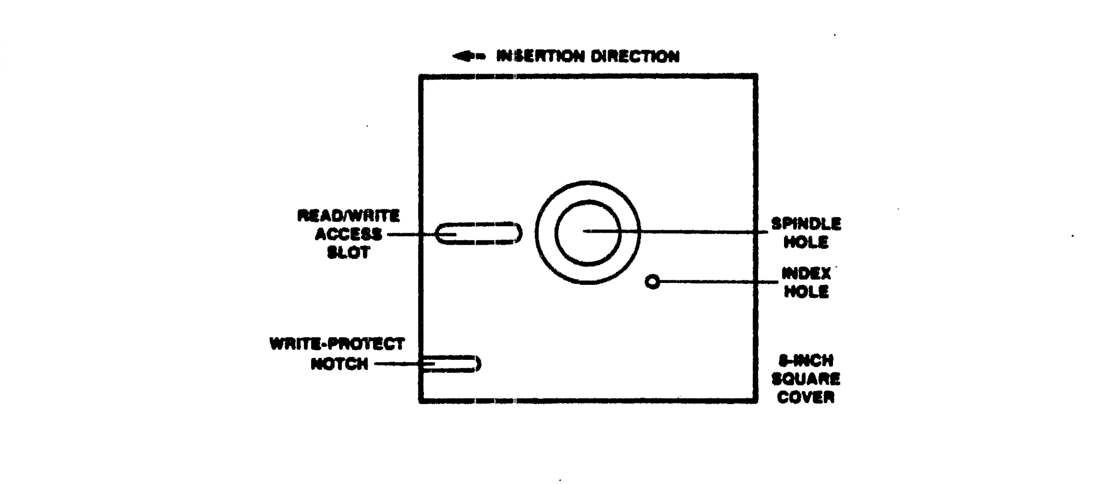


Figure 3-1. Floppy Diskette

Exercise care when handling diskettes so as not to touch the recording surface exposed through the openings on the cover. Use good quality diskettes, Dysan part number 800838 or equivalent; poor quality diskettes cause excessive wear of the read/write head.

CAUTION

> Information stored on the diskette can be destroyed by static electricity discharge from a person touching the equipment. Synthetic carpets, in particular, can cause a build-up of static electricity. Periodic application of Neutro Stat (or equivalent) antistatic spray can retard the static electricity build-up; however a safer solution is to install the system on an antistatic mat (see Velostat type 1854 4'x 8' x 1/8" as manufactured by 3M-Static Control Systems, St. Paul, Minn. 55101, or equivalent).

The floppy disk unit engages with the diskette when the door is closed. The door is closed by pushing the top of the door down until it latches.

CAUTION

Do not replace diskettes while programs or commands are being executed. DO NOT TURN THE POWER ON OR OFF WHILE A DISKETTE IS IN THE UNIT. Otherwise, a track of data may be destroyed. If drive units must be powered down when a red activity indicator is lighted, open drive unit door and remove diskette before powering down.

FLOPPY DISKETTE

The floppy diskette is sealed in an 8-inch square cover that is lined to minimize static electricity. When not in use, the diskette should be kept in the storage envelope provided with each diskette. When the write-protect notch is covered with a tape strip, the diskette is write-enabled. Some commands, such as the SUBMIT command, require that the accessed diskette be write-enabled. Files on the diskette can be write protected by removing the tape strip from the write-protect notch. Each diskette is shipped with several tape strips. Store diskettes in normal work areas where temperatures are within the range of 50-125 degrees Fahrenheit (27-50C), and do not allow diskettes to be exposed to magnetic flux lines such as can be present around large power supply transformers. Do not allow the diskettes to be exposed to direct sunlight for any extended period of time.

BLANK DISKETTE FORMATTING PROCEDURE

Diskettes provided with AmSYS29/10A are preformatted in soft sector format. The index address mark,record identification fields, and gaps are prerecorded. During normal use, the soft sector format is untouched while data field records are changed and updated as required by the user. Occasionally, an index address mark or record identification field can be inadvertently erased from the diskette. An erasure can result, for example, if the system is powered down while a diskette is engaged in the floppy disk drive unit.

Usually, when part of the diskette formatting is erased or when a diskette has a physical fault that prevents correct formatting, an error message is displayed at the CRT console. The error message form is n PERM ERR DISK x, where n is an error number that indicates the type of error and x indicates the floppy disk drive unit (A or B) that contains the permanent error. No error message is displayed at the CRT console when an index address mark is erased. Instead, the floppy disk drive unit activity indicator remains lighted, and communication with the diskette via the CRT console is impossible. To restore the index address marks on the erased diskette: 1) remove the erased diskette; 2) get the format procedure from another (backup) diskette; 3) reformat the erased diskette by performing the diskette formatting procedure described in the AMDOS29 Operating System manual.

NOTE

Data on the diskette is erased during formatting.

Diskette formatting is accomplished by invoking the AMDOS29 utility FORMAT, which records the required information into the index address mark area, record identification fields, and gaps. Blank diskettes purchased from AMD are preformatted in single density. Diskettes purchased elsewhere must be formatted before being used with AmSYS29/10A.

SYSTEM INITIALIZATION OPERATION

Diskettes containing the AmSYS29/10A software are provided with the system. The operating system and certain selected utilities are on the system diskette. To initialize the system and load the operating system, perform the following steps:

1. Make sure that all the peripheral devices have their power switches turned on.

2. Set the AmSYS29/10A front panel power ON/OFF switch to ON and the RUN/HALT switch to RUN.

3. Insert the system diskette into drive A (the lower drive).

4. Press the RESET switch on the system cabinet. The bootstrap operation begins. The red LED on drive A lights to indicate that drive A is being selected to load the operating system.

5. When the system boot is complete, the following message is displayed:

```
AMDOS29 64K, VERSION 3.0
```

6. The system executes the PROFILE.SUB file, if one exists on the disk. If a PROFILE. SUB file exists, the commands in the file are echoed one by one to the console as they are executed.

7. The prompt

```
A>
```

then appears. When the system prompt appears, the system is ready to accept the user's first command.

During initial system installation, it is wise to make a backup copy of the system diskettes. To copy the system diskettes, enter:

```
CPYDSK
```

The system responds with:

```
SOURCE ON A, DESTINATION ON B
```

Insert a blank diskette into drive B and press the RETURN (NEW LINE) key. The system begins to copy the diskette in drive A onto the diskette in drive B. A message is displayed to indicate the track being copied. When track 76 has been copied, insert another blank disk into drive B and press the RETURN (NEW LINE) key to make another copy. Enter CONTROL-C (CONTROL and C keys simultaneously) to terminate the copy operation.

Next, it is useful to obtain a hard copy of the directory that is on each system diskette. To select the printer, enter:

```
STAT LST: = UL1:
```

(for a parallel printer - otherwise a serial printer is assumed)

followed by:

CONTROL-P (CONTROL and P keys simultaneously)

To list the directory, enter:

```
DIR
```

The directory is displayed on the CRT and printed at the same time. Attach the directory listing to the diskette envelope for reference. Insert the next diskette and list that directory. After all directories have been printed, discontinue display printing by entering:

CONTROL P

At this point, another diskette can be selected for the work to be done next.

The editor is resident on the system disk when it is shipped from the factory. If the editor is required, see the ED User's Manual for usage information.

CHAPTER 4
SYSTEM SOFTWARE

INTRODUCTION

AmSYS29/10A software can be divided into two general categories: operating system software and microprogram support software. As shown in figure 4-1, the operating system software can be further divided into the memory resident operating system components and disk resident utilities. Microprogram support software includes microprogram generation software and microprogram debug software. This chapter briefly describes each software product; detailed descriptions are provided in the AMDOS29 Operating System User's Manual and the Microprogram Support Software User's Manual.

AMDOS29 OPERATING SYSTEM

AMDOS29, a CP/M* 2.2-compatible operating system, consists of the Basic Input/Output System (BIOS), the Basic Disk Operating System (BDOS), and the Console Command Processor (CCP). The operating system resides in high memory (locations DC00-FFFFH); user programs are loaded into the transient program area (TPA), locations 0100H to DBFFH. These memory locations change from time to time; however, memory location 0005H is always a JMP to start of the operating system. BIOS provides the drivers for accessing the physical devices, including the floppy disk drive, and provides the interface to system peripheral devices such as a line printer and CRT console. This component thus defines the hardware environment. Specific user-system configurations (logical-to-physical device assignment) can be examined and altered by means of the statistics (STAT) utility.

BDOS is the file-management component,controlling logical access to the files on the diskettes. BDOS incorporates primitive commands to enable disk-drive and file-manipulation operations such as: search for a file by name; file create, open/close, read/write, rename; disk-drive select, reset, and interrogation; and set diskette memory buffer address. A detailed description of program usage of the BIOS and BDOS primitives is contained in the AMDOS29 Interface Guide.

Interface between the user's CRT console and the remainder of the system is provided by CCP, which executes commands for functions such as: list file directives, display file contents, and execute user programs and AmSYS29 utilities.

* CP/M is a trademark of Digital Research Corporation.

As shown in figure 4-1, some utilities are intrinsic to AMDOS29 and reside permanently in CCP; others are disk resident and are executed in memory under CCP. The following description defines the utility programs supported by AMDOS29.

DIR	List file directory.
TYPE	Display contents of specified file.
ERA	Erase specified file or files.
REN	Rename specified file.
SAVE	Save memory on a diskette file.
USER	Change user area on diskette.
PIP	Perform inter-media data transfer.
SUBMIT/XSUB	Submit specified file of commands for batch processing.
STAT	Identify or alter current peripheral-device as signments; display specified diskette and file characteristics.
SYSGEN	Duplicate AMDOS region of one diskette (tracks 0 and 1) on another diskette.
DISPL	Display ASCII coded file on console (screen-oriented).
DUMP	Dump file, in hex format, to console.
VFU	Download VFU information to 120 CPS parallel printer.
CPYDSK	Copy contents of one diskette to another.
FORMAT	Initialize a diskette for single or double density use.
PREHEAT	PROM programming utilities.
ED	Text editor.
DDT	Program debugging tool.
FILCOM	Perform ASCII or binary compare of two files.

The BDOS internal-function utilities are illustrated in figure 4-1 and briefly defined below for reference only; refer to the System User's Interface Guide for a detailed description of the internals.

SEARCH	Search directory for specific filename.
OPEN	Open file for operations.
CLOSE	Close file after operation.
RENAME	Change filename.
READ	Read file contents.
WRITE	Write new file.
SELECT	Select diskette drive for operations.
ALLOC	Interrogate allocation map for current drive.
SETDMA	Set memory buffer address.
LOGDRV	Interrogate for drive number of current logged-in diskette.
ONLDRV	Interrogate online (logged in) drive.
LOGIN	Log in and select diskette.
CREATE	Create new file.

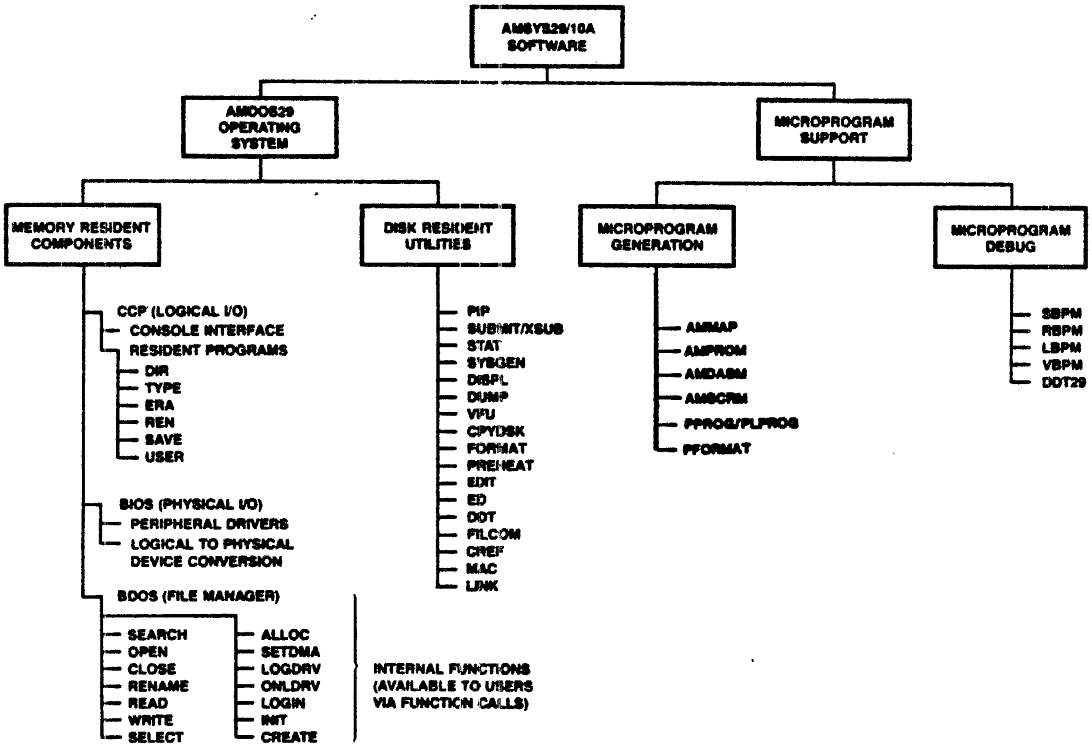
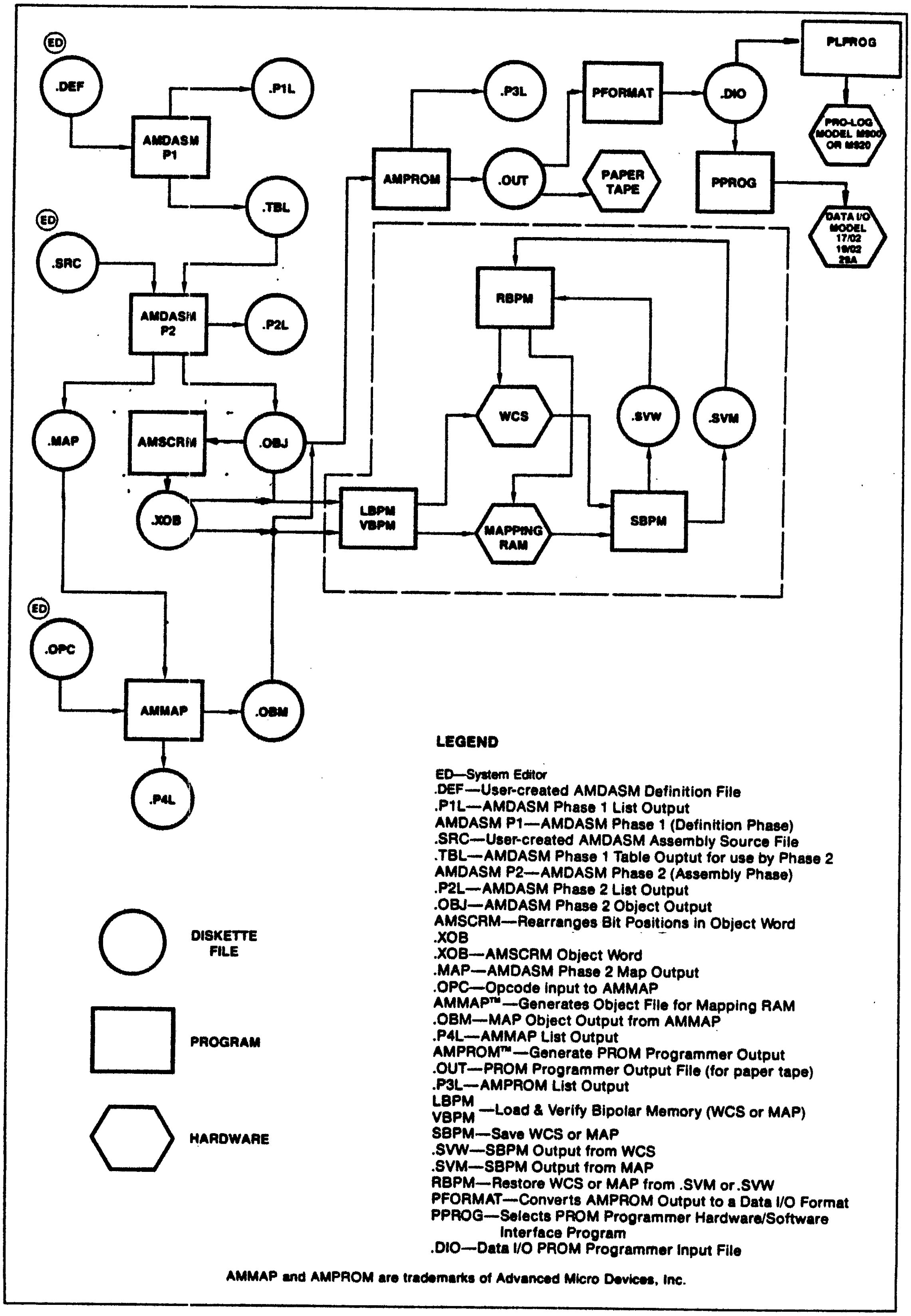


Figure 4-1. AMDOS29 System Structure

MICROPROGRAM SUPPORT SOFTWARE

The microprogram support software includes microprogram generation software and microprogram debug software. Microprogram generation software consists of the AMDASM meta-assembler, AMSCRM object file handler, AMPROM PROM formatter, and the AMMAP mapping RAM/PROM data assembler. The microprogram debug software consists of programs to load, save, and debug the object code generated by AMDASM. A visual description of the interrelationships of the microprogram support software is presented in figure 4-2.



LEGEND

ED—System Editor
.DEF—User-created AMDASM Definition File
.P1L—AMDASM Phase 1 List Output
AMDASM P1—AMDASM Phase 1 (Definition Phase)
.SRC—User-created AMDASM Assembly Source File
.TBL—AMDASM Phase 1 Table Ouptut for use by Phase 2
AMDASM P2—AMDASM Phase 2 (Assembly Phase)
.P2L—AMDASM Phase 2 List Output
.OBJ—AMDASM Phase 2 Object Output
AMSCRM—Rearranges Bit Positions in Object Word
.XOB
.XOB—AMSCRM Object Word
.MAP—AMDASM Phase 2 Map Output
.OPC—Opcode Input to AMMAP
AMMAP™—Generates Object File for Mapping RAM
.OBM—MAP Object Output from AMMAP
.P4L—AMMAP List Output
AMPROM™—Generate PROM Programmer Output
.OUT—PROM Programmer Output File (for paper tape)
.P3L—AMPROM List Output
LBPM
VBPM —Load & Verify Bipolar Memory (WCS or MAP)
SBPM—Save WCS or MAP
.SVW—SBPM Output from WCS
.SVM—SBPM Output from MAP
RBPM—Restore WCS or MAP from .SVM or .SVW
PFORMAT—Converts AMPROM Output to a Data I/O Format
PPROG—Selects PROM Programmer Hardware/Software
Interface Program
.DIO—Data I/O PROM Programmer Input File

AMMAP and AMPROM are trademarks of Advanced Micro Devices, Inc.

Figure 4-2. Microprogram Support Software Relationship

AMDASM is a meta-assembler used to assemble the user-generated microcode into the object program. AMDASM is a two-pass assembler. The first phase is the definition of the microinstruction length, mnemonics, constants, and formats used to write source programs for the target system. The second phase reads the source program and assigns labels and constants. The source program consisting of executable statements is translated into the binary format object program which is stored on a disk file.

AMSCRM is an object code file handler. This program allows the user to reorganize the microinstruction word and produce a new microcode object file. The user can reassign the bit positions of the microinstruction by specifying source and destination bit positions and the length of each field to be moved. The program generates a new object code file.

AMPROM is used to generate object code arranged in a form corresponding to the organization of the PROM used in the target system.

AMMAP is used to generate non-microinstruction PROM data. This program enables the user to create a map PROM from the assembler entry point table or a constant lookup table PROM. The output from AMMAP is an AMDASM/AMPROM-compatible object file.

LBPM, Load BiPolar Memory, moves the AMDASM generated object files to the Microprogrammed Controller, Writable Control Store, or Microprogram Sequencer mapping RAM.

VBPM, Verify BiPolar Memory, verifies the contents of the writable control store by comparing the contents with the AMDASM-generated file on disk. VBPM is used to load a program and to compare an updated assembler output with a debugged save file.

SBPM, Save BiPolar Memory, saves the current contents of the Writable Control Store as a new file on disk. In addition to saving the file before removing power from the system, this also provides storage of a working version of the program that can be used to verify an updated assembler output.

RBPM, Restore BiPolar Memory, moves the contents of a save file on disk to writable control store.

DDT29, Dynamic Debug Tool, provides the software interface between the CRT console and the microprogrammed controller. The contents of writable control store can be displayed and modified under DDT29. DDT29 controls the target system clock which allows the target system to be halted, single stepped, or run at full speed. Monitor points and microprogram address can be displayed when requested by the user at the CRT console.

CHAPTER 5

THEORY OF OPERATION

INTRODUCTION

This chapter describes the details of operation of the AmSYS29/10A mainframe. Description is limited to the system support processor and the microprogrammed controller and their component parts. For details of peripheral equipment operation, see the appropriate manual for the particular device.

The AmSYS29/10A configuration, as shown in the block diagram of figure 5-1, is logically divided into two sections: the microprogrammed controller (MPC) section and the system processor section. The system processor consists of a RAM card, a floppy disk controller card, the CPU card housed in the system mainframe, and the dual drive floppy disk unit. The CRT console is supplied, and the system processor includes drivers to support the CRT console plus three optional peripherals (printer, teletype, and PROM programmer). The microprogrammed controller includes the writable control store cards, the clock control logic card, and microprogram sequencer housed in the system mainframe.

The system processor incorporates all necessary hardware, and accommodates all required software, to provide the user with an interactive interface to the microcode and hardware of the system under development (microprogrammed controller). In addition, the system processor provides storage and output capabilities for the developed microcode. How the system processor hardware accomplishes the stated purpose is described in the following paragraphs. During the description, refer to figure 5-1.

SYSTEM PROCESSOR

The system processor hardware is configured around a standard Am9080A microprocessor arrangement that includes the Am8224 clock generator and driver and the Am8228 system controller and bus driver all of which are contained on the CPU card. The CPU card also contains four RS232 serial I/O ports, three 8-bit parallel I/O ports, a page register, and buffers for the system processor address, data, and control busses that are routed to the microprogrammed controller.

The microprogrammed controller as well as the system processor, under control of the Am9080A microprocessor, can gain access to the bus. The bus, besides being connected to the CPU card buffers, is also connected on the CPU card to the four RS232 serial I/O ports, the parallel I/O ports, and the floppy disk interface so the Am9080A microprocessor can service the system mainframe peripherals.

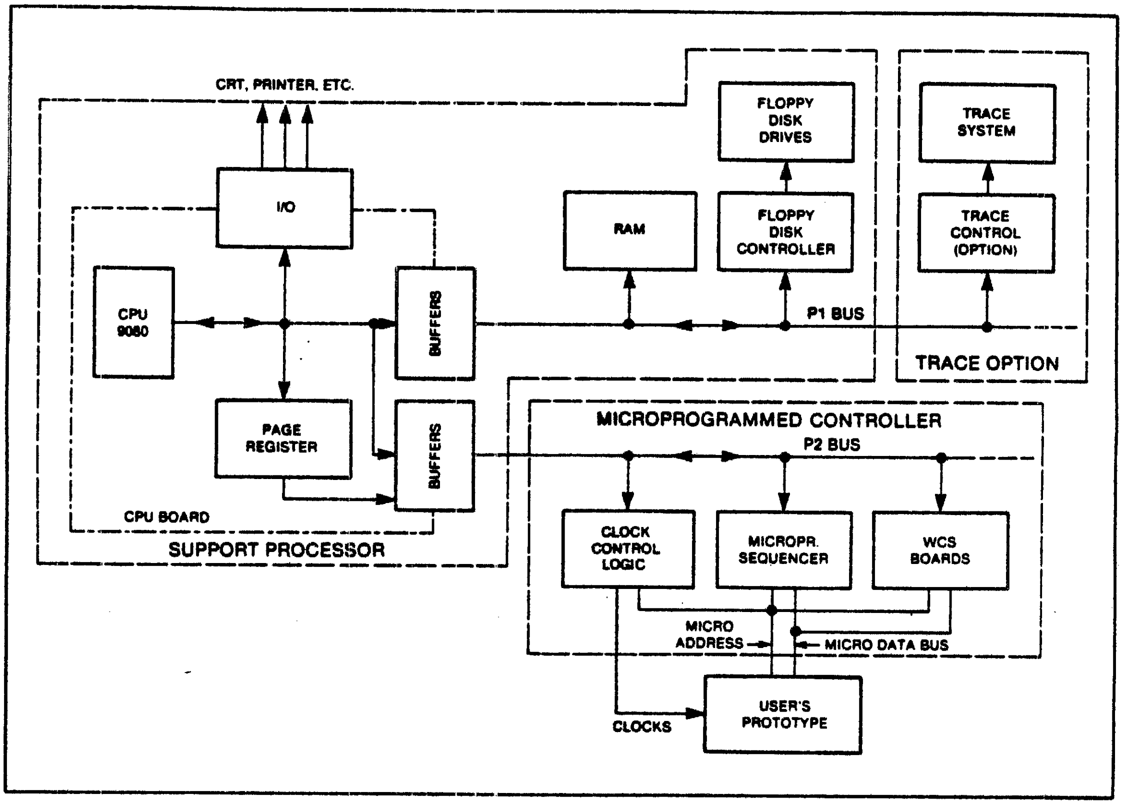


Figure 5-1. Detailed Block Diagram of AmSYS29/10A

In addition, the bus is connected to the page register on the CPU card to enable the system processor to map various microprogrammed functions into the upper 32K address range of the Am9080A microprocessor. Consequently, microcode for the microprogrammed controller can be operated on by the Am9080A microprocessor under user control or, other microprogrammed controller functions (as described later) can be monitored or controlled by the user. Thus, the CPU card, under user control from the CRT console, provides an interface to the rest of AmSYS29/10A.

Finally, to complete connections of the system processor buses, the bus is routed to the two system mainframe card connectors allocated for RAM cards. One RAM card is supplied with the basic AmSYS29/10A configuration to the provide 65,536 bytes of MOS read/write memory. Part of the RAM card is used as a transient program area (TPA) that constitutes a working memory for the Am9080A microprocessor. The rest of the RAM card memory area (other than TPA) provides storage for the AMDOS29 software disk operating system, command processor, and I/O routines. Table 5-1 lists the AmSYS29/10A bus signals and compares them to the Multibus* signals.

* Multibus is a registered trademark of Intel Corporation.

Table 5-1. System 29 Bus vs. Intel Multibus

SYS 29		INTEL MB		SYS 29		INTEL MB	
Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	Ground	1	Ground	44	MA B15	44	MA BF
2	Ground	2	Ground	45	MA B12	45	MA BC
3	+5V	3	+5V	46	MA B13	46	MA BD
4	+5V	4	+5V	47	MA B10	47	MA BA
5	+5V	5	+5V	48	MA B11	48	MA BB
6	+5V	6	+5V	49	MA B8	49	MA B8
7	+12V	7	+12V	50	MA B9	50	MA B9
8	+12V	8	+12V	51	MA B6	51	MA B6
9	No conn	9	-5V	52	MA B7	52	MA B7
10	No conn	10	-5V	53	MA B4	53	MA B4
11	Ground	11	Ground	54	MA B5	54	MA B5
12	Ground	12	Ground	55	MA B2	55	MA B2
13	B CLOCK	13	B CLOCK	56	MA B3	56	MA B3
14	INIT*	14	INIT	57	MA B0	57	MA B0
15	BPRN*	15	BUS PR IN	58	MA B1	58	MA B1
16	No conn	16	BUS PR OUT	59	PG REG B0	59	DB BE
17	BUSY*	17	BUS BUSY	60	PG REG B1	60	DB BF
18	BUS REQ	18	BUS REQ	61	PG REG B2	61	DB BC
19	MEM READ*	19	MEM READ	62	PG REG B3	62	DB B0
20	MEM WRITE*	20	MEM WRITE	63	PG REG B4	63	DB BA
21	I/O READ*	21	I/O READ	64	PG REG B5	64	DB BB
22	I/O WRITE*	22	I/O WRITE	65	PG REG B6	65	DB B8
23	XACK*	23	XACK	66	PG REG B7	66	DB B9
24	INH*	24	DSABL MN MEM	67	DB B6	67	DB B6
25	No conn	25	SPEC ACKN	68	DB B7	68	DB B7
26	Run*/Halt	26	DSABL ROM	69	DB B4	69	DB B4
27	Reset*	27	BYTE HIGH EN	70	DB B5	70	DB B5
28	SGL STEP	28	MA 10	71	DB B2	71	DB B2
29	SGL STEP	29	COM BUS REQ	72	DB B3	72	DB B3
30	EPROM	30	MA 11	73	DB B0	73	DB B0
31	PH 2 TTL	31	CONST CLK	74	DB B1	74	DB B1
32	MONSEL	32	ADDR 12	75	Ground	75	Ground
33	No conn	33	INTR ACKN	76	Ground	76	Ground
34	CPIM	34	MA 13	77	-	77	-
35	INTR 6*	35	INTR 6	78	-	78	-
36	INTR 7*	36	INTR 7	79	-12V	79	-12V
37	INTR 4*	37	INTR 4	80	-12V	80	-12V
38	No conn	38	INTR 5	81	+5V	81	+5V
39	No conn	39	INTR 2	82	+5V	82	+5V
40	INTR 3*	40	INTR 3	83	+5V	83	+5V
41	No conn	41	INTR 0	84	+5V	84	+5V
42	INTR 5*	42	INTR 1	85	Ground	85	Ground
43	MA B14	43	MAB E	86	Ground	86	Ground

INTR = Interrupt MA = Memory address PGREG =Page register
DB = Data bus B = Bit * = Active low

The RAM card TPA is used to contain application program modules which are loaded from the floppy disk unit under direct memory access (DMA) control. That is, any data moved in either direction between the microprogrammed controller and the floppy disk unit or the floppy disk unit and any other part of the system processor is routed through the RAM card.

In order to achieve this DMA capability, the Am9080 gives up control of the three system buses to a floppy disk controller (FDC). The FDC, when moving data to the system mainframe, places data, address, and control signals on the corresponding RAM card and interface busses. Once the data is stored in the RAM card, the Am9080A microprocessor operates on the data in accordance with selected AMDOS29 or AMDASM programs. The data moved by the FDC to the system mainframe comes from diskettes inserted in either floppy disk drive unit A or B, which are also parts of the floppy disk unit. The diskette on drive A normally contains the AmSYS29/10A software (AMDOS29, AMDASM, and microprogramming support software). The diskette on drive B is normally used as a mass memory storage area for data such as the microcode under development for the microprogrammed controller. Any data moved from the disk unit to the RAM card, or from the microprogrammed controller or the Am9080A accumulator, can be placed on-line to any of the four RS232 serial I/O ports or to the parallel I/O ports which are enabled under Am9080A control. The Am9080A microprocessor enables I/O port accessing by executing AMDOS29 I/O service routines that regularly monitor the status of the I/O ports and then cause the Am9080A to service an active I/O port.

The parallel I/O port incorporates a versatile Programmable Peripheral Interface (PPI) device capable of supporting a variety of peripherals that could be connected to the parallel I/O port. Connection is through an edge connector on the CPU card. Depending on the mode of operation desired, driver or terminator chips must be added to the CPU card. The port is shipped configured for a Centronics printer.

Each of the four RS232 serial I/O ports on the CPU card employs a universal synchronous/asynchronous receiver/transmitter (USART). The USART accepts parallel data from the Am9080A, formats and serializes the data for RS232 serial transmission, and then transmits the data as a serial bit stream. The USART can simultaneously transmit and receive data. That is, while parallel data from the Am9080A is being formatted and transmitted serially to a system mainframe peripheral, the USART can accept serial data from the same peripheral, convert the data into parallel form and present the parallel data to the Am9080A. The USARTs contain a status register that maintains information about the current operational status of the USART. The status registers of the USARTs can be monitored by the Am9080A under AMDOS29 software control to determine if any serial I/O port contains data for the Am9080A. If serial I/O port status indicates data is available, the data register in that I/O port is enabled by the Am9080A (under AMDOS29 control) so the data can be read out and acted upon by Am9080A. The Am9080A can operate on the data, or store the data in the RAM card, or transfer the data to the microprogrammed controller by setting the CPU card page register contents to the appropriate hexadecimal value.

Serial ports communicate with the CRT terminal at 9600 baud, a line printer at 9600 baud, and either a paper-tape reader/punch or a PROM programmer at 600 baud. The fourth port is a spare, user presettable by a jumper to either 110 or 9600 baud.

The page register enables up to 128 AmSYS29/10A circuit elements, functions,or data storage areas to be mapped into the upper 32K address range of the Am9080A. The Am9080A can then operate on the data or function mapped into the upper 32K range. The RAM residing in the Am9080A upper 32K address range also is accessed through the page register. Thus, no conflict exists between the upper 32K RAM and some other AmSYS29/10A function which is mapped into the upper 32K address range. Microprogrammed controller functions or data that can be mapped into the Am9080A upper 32K address range reside on the microprogram sequencer card, the clock control logic card and the writable control store (WCS) card comprising the microprogrammed controller.

The 9080A instruction execution control implements the operation designated by pressing the front panel RUN or SINGLE STEP switches.

The priority interrupt ranks interrupt requests from four different sources and feeds them in order of rank to the microprocessor for its response.

INITIALIZATION

When the RESET switch on the front panel is pressed, the CPU initialization control generates a reset/bootstrap signal, which has two principal functions. Internally, it clears the page register to 0, resets the four serial I/O ports to idle, and resets the microprocessor to address 0 with all internal flags and registers cleared. Externally, it requests the floppy disk controller to transmit the bootstrap program stored on the disk into the AmSYS29/10A mainframe.

MEMORY ACCESS

When the CPU executes a read or write in the memory, it first issues an address of a memory location. Shortly after the cycle begins, the microprocessor status appears on the data bus, identifying the nature of the machine cycle (that is, a read or a write - or, in other cases, an instruction fetch, an input, or an output). At the end of the first clock period, the status bits are stored in the Am8228 system controller on the CPU card.

If the status indicates a memory read, the read command is immediately forwarded to the memory, while the microprocessor switches the data bus to the input mode (the data bus is always in output mode at the start of every machine cycle, to permit the status bits to progagate to external control circuits). If the status indicates a write, the command is delayed until the third clock period. During the second clock period, the microprocessor leaves the data bus in output mode and places the data to be written on the bus. The write command is then issued and the memory cycle begins, finishing just after the microprocessor has ended its last cycle.

INPUT/OUTPUT CONTROL

During any input or output operation except those involving the floppy disk, this portion of the CPU circuitry generates control signals that govern data transfer to or from the various peripheral devices. It includes a baud rate generator that clocks the data at the rate appropriate for each particular device. The microprocessor is in control at all times. Addresses of the various devices are shown in table 5-2. Page register and floppy disk operations, listed in the table, are described in the next two sections.

Table 5-2. Input/Output Addresses

I/OAddr	Device Name	Remark	Location and Function
60 (62)	Serial I/O #1 data	R/W	J11-System console:9600 baud
61 (63)	Serial I/O #1 cmd	R/W	
64 (66)	Serial I/O #2 data	R/W	J12-TTY Mod.40 printer:9600 baud
65 (67)	Serial I/O #2 cmd		
68 (6A)	Serial I/O #3 data	R/W	J13-Paper tape/reader punch PROM programmer 600 baud
69 (6B)	Serial I/O #3 cmd	R/W	
6C (6E)	Serial I/O #4 data	R/W	J14-Spare 110/9600 baud
6D (6F)	Serial I/O #4 cmd	R/W	
70	Parallel I/O port A	R/W	J10-Pin out, terminators and drivers configured for Centronics printer
71	Parallel I/O port B	R/W	
72	Parallel I/O port C	R/W	
73	Parallel I/O control register	R/W	CPU-P4 General purpose interface
78(79-7B)	Page register	W	CPU Internal use only
7C-7F	Floppy disk interface	R/W	J17-Floppy disk drive subsystem
80	High spd trace data	R/W	Internal use only
81	High spd trace cmd	R/W	

Note: Parentheses indicate alternate address.

PAGE REGISTER ADDRESSING

The central processing unit communicates with the microprogrammed controller along the same bus that addresses the system memory. (Electrically the microprogrammed controller bus is separate, because the two buses have separate driver circuits.) In order that the MPS and the memory each react only to their intended addresses, the CPU includes a page register that identifies the proper destination for a subsequent address.

The page register, shown in figure 5-2, contains eight bits; of these, the high-order bit is set to 1 when an address is intended for the memory, and the other seven bits are ignored. The high-order bit is 0 when a function of the microprogrammed controller is called; the seven low-order bits of the page register are decoded to identify the particular operation for the addressed element. These seven bits can address 128 elements or functions in the microprogrammed controller, each of which has its own memory space addressed in the upper half of the microprocessor's range (between 32,768 and 65,535). However, in AmSYS29/10A, only six of the possible 128 are decoded with the four low-order bits, as shown in table 5-3. Of these, the WCS uses the entire available address space, the clock control logic card uses only eight addresses (eight not 8K), and the microprogram sequencer uses 256. For all these units, the high-order byte of the address is always between 80 and FF inclusive, since it is the upper half of the microprocessor memory which is page addressed.

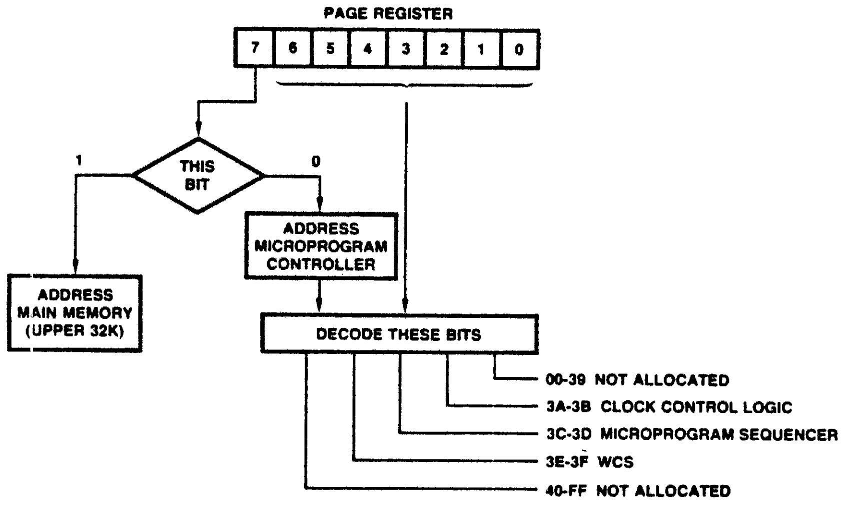

PAGE REGISTER
7
6
5
4
3
2
1
0
1
THIS BIT
0
ADDRESS MICROPROGRAM CONTROLLER
ADDRESS MAIN MEMORY (UPPER 32K)
DECODE THESE BITS
00-39 NOT ALLOCATED
3A-3B CLOCK CONTROL LOGIC
3C-3D MICROPROGRAM SEQUENCER
3E-3F WCS
40-FF NOT ALLOCATED

Figure 5-2. Page Register Bit Allocation

Table 5-3. Page Address Allocation

HEX ADDRESS	ALLOCATION
FF	System memory (inclusive)
80	
3F	Writable control store, 2K-4K range
3E	Writable control store, 0-2K range
3D	
3C	Microprogram sequencer
3B	
3A	Clock control logic

FLOPPY DISK CONTROL

The floppy disk controller card controls operations of the floppy disk unit. The start command, together with an address of the location on the disk to or from which data is to be transferred, is transmitted to the floppy disk controller through the Pl bus (figure 5-3 is a combination interface diagram and timing chart, showing the flow of signals left and right, and their sequence top and bottom). When the disk signals that data transfer is about to begin, usually several milliseconds later, the floppy disk controller issues a Hold signal to the CPU. At the completion of its current instruction (up to 9 microseconds later), the CPU issues a hold acknowledge, indicating that it is in the wait state and has released the buses. Then the floppy disk controller takes control of the buses, allowing transfers to/from memory. The microprocessor remains in the wait state until the disk operation is complete.

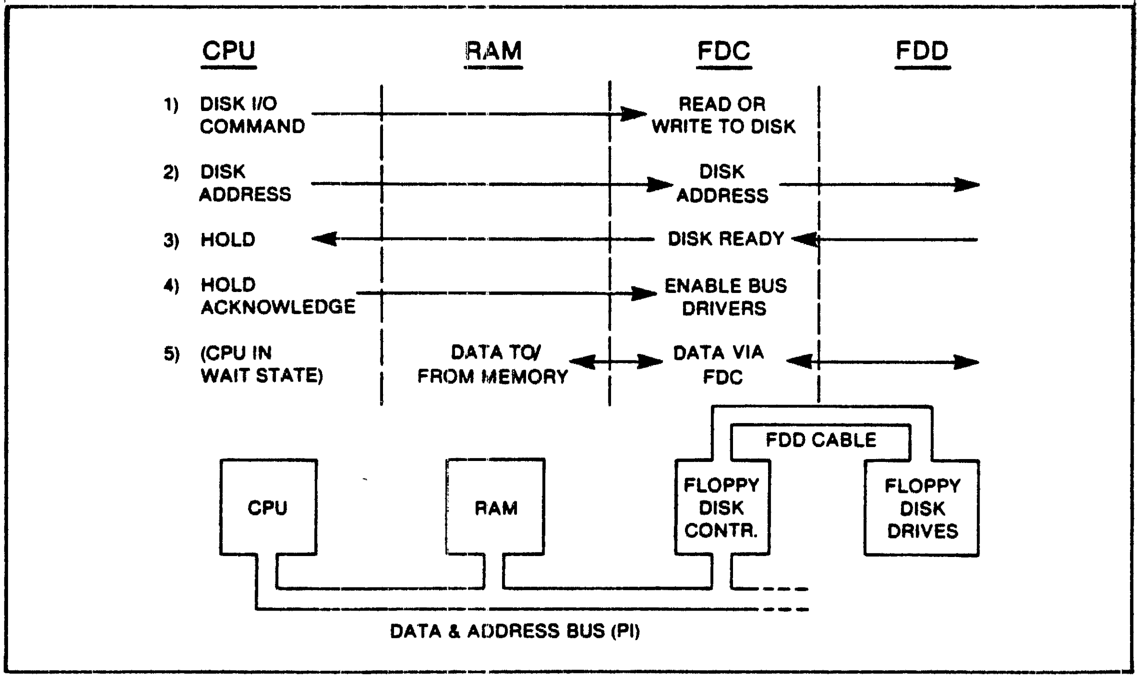


Figure 5-3. Floppy Disk Interface Operation

SINGLE-STEP CONTROL

A debounce latch, a single step flip-flop, and a decoder are the circuits that control single step and run modes in the system processor, as shown in figure 5-4. The single step flip-flop is held off when the front panel STOP/RUN switch is in the RUN position. In the off state, the flip-flop output enables the decoder, which indirectly generates the READY line to the Am9080 microprocessor. With READY up, the micro processor runs normally. While running, the Am8224 clock generator produces a status strobe during every machine cycle.

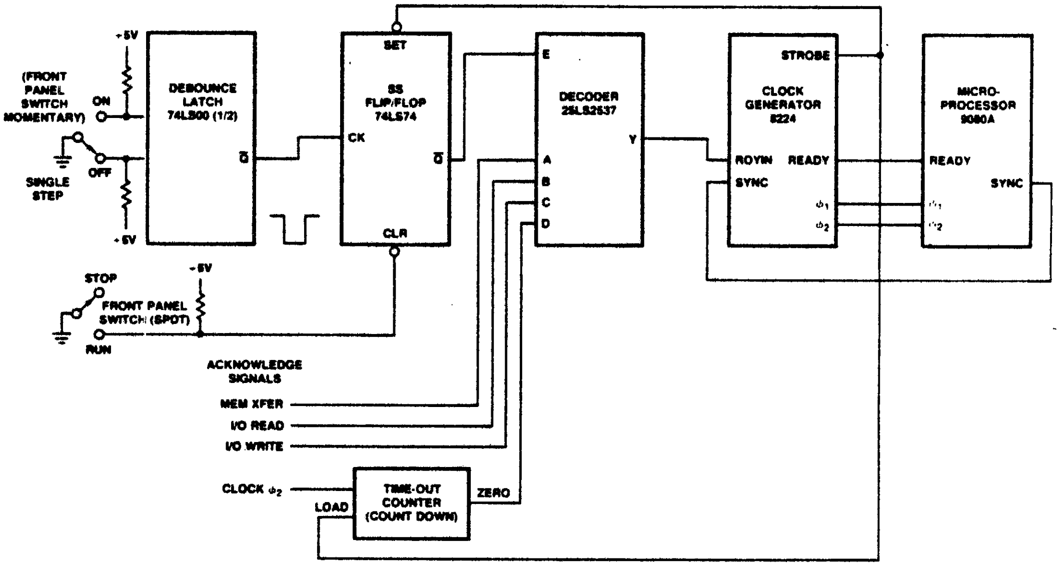


Figure 5-4. Single Step Control

When the STOP/RUN switch is moved to the STOP position, it releases the single-step flip-flop, which turns on with the next status strobe. This disables the decoder and drops READY so that the microprocessor enters the wait state and inhibits the Am8224 from generating status strobes.

The debounce latch is directly connected to the contacts of the SINGLE STEP switch on the front panel. When the switch is pressed and then released, the debounce latch output turns off the single step flip-flop, enables the decoder, brings up READY, and the microprocessor starts running. However, it executes only one machine cycle, during which another status strobe is generated, turning on the single-step flip-flop and stopping the microprocessor again.

INTERRUPT

The Am9080 microprocessor has one interrupt input; therefore, any system using the Am9080 that has more than one interrupt signal also must have means to identify the interrupt. In AmSYS29/10A, the Am2913 priority interrupt expander on the CPU card performs this function. The expander accepts interrupt requests from up to eight sources; four are used in AmSYS29/10A. These are, in descending priority, the front panel interrupt switch (interrupt 7), the operating system (interrupt 6), the clock stop circuit on the clock control logic card (interrupt 5), and a general-purpose line (interrupt 4) that can be used by the prototype system, and the floppy disk controller (interrupt 3). When an interrupt request comes in from any of these, the expander sends an interrupt signal to the microprocessor. When the instruction being executed at the moment is completed, and if the microprocessor's interrupt enable is on, the microprocessor responds with interrupt acknowledge status and immediately switches the data bus to input mode. The expander encodes the identity of the interrupt and places it on the bus as part of the restart instruction. If the interrupt were not present, an instruction fetch is the current status signal instead of interrupt acknowledge, and input mode would bring in the new instruction from the system memory.

Restart places the previous contents of the program counter on the stack and places the new address into the program counter from one of eight addresses near the bottom of the memory. With five interrupt sources, only five of these eight addresses are used. The new address in the program counter is the address for the first instruction in the interrupt service routine. Since there are five such addresses and five possible interrupt sources, each one can have its own processing routine. If two or more interrupts are received simultaneously, the expander encodes and forwards only the one with the highest priority, which is 7; the others must wait their turn or be lost.

Interrupt enable is turned on by the enable interrupt instruction in the microprocessor program and turned off by the disable interrupt instruction or automatically when the microprocessor enters interrupt acknowledge status. Consequently, the enable interrupt instruction should be placed at the end of every interrupt-processing subroutine.

If interrupt enable is not on, attempts by the expander to interrupt the microprocessor are ignored. However, if the source of the interrupt keeps the request up, the interrupt is acknowledged if and when enable comes on.

MICROPROGRAMMED CONTROLLER

The microprogrammed controller consists of three basic circuit cards: the clock control logic, the writable control store, and the microprogram sequencer. The microprogrammed controller section of AmSYS29/10A uses a separate memory mapped (P2) bus (containing address, data, and control) to communicate with the system processor section. In addition, the MPC has a bipolar microprogram data bus and a microprogram address bus that provide a microcode path from the WCS cards to the system under development. The rest of the microprogrammed controller provides control functions via the clock control logic card, a computer control function via the microprogram sequencer card, and PROM simulation via the WCS cards.

Even though only one WCS card is supplied with the basic AmSYS29/10A, the system mainframe is prewired to accept three additional optional WCS cards. Each WCS card provides 1K by 64 bits of read/write memory for development system microcode storage. Three switches on each card allow the card to be configured as the upper or lower 64 bits of a 128-bit microcode word starting at location 0, 1K, 2K, or 3K. Therefore, a four-card high-speed WCS memory can be configured into a 4K by 64-bit array or a 2K by 128-bit array. All or any part of the microcode in WCS memory can, by invoking the appropriate microprogramming support software command, be examined or operated on from the CRT console and be verified and modified as required.

When examination or modification is completed, the microcode can be returned to the WCS by again using the appropriate microprogramming support software command. The microcode is moved back and forth between the system processor and the WCS by using the facilities of the P2 bus.

The clock control logic card contains microprogrammed controller clock circuitry as well as a variety of other control functions that can be invoked by the user as an aid to developing the prototype system microcode and hardware. An internal crystal-controlled oscillator on the clock control logic card is supplied to the CLOCK OUT BNC connector on the system mainframe front panel. A coaxial jumper, supplied with AmSYS29/10A, can be used to connect the signal at the CLOCK OUT connector to the CLOCK IN front panel connector. The signal present at the CLOCK IN conector is supplied to the clock circuitry on the clock control logic card. The clock circuitry develops a controlled oscillator clock signal output that clocks the system under development.

In addition to the internal oscillator and clock circuitry, the clock control logic card contains the logic necessary to respond to system mainframe front panel controls allocated for microprogrammed controller functions. Also, the clock control logic card provides registers that contain microcode breakpoint and jamming addresses as well as the address of the last microinstruction executed. The breakpoint and jamming addresses can be set and the address of the last microinstruction executed can be displayed at the CRT console through microprogramming support software commands that automatically implement the page register memory mapping technique. In addition to the breakpoint and jamming registers, the clock control logic card provides a monitor register that can hold up to 20 bits of user selected and wired test points or other data from the system under development. The contents of the monitor register are displayed at the CRT console for interpretation by invoking the appropriate microprogramming support software command.

During normal microprogrammed controller operation, the optional microprogram sequencer card sequences and addresses the microcode in the WCS cards via the microprogram address bus. The resultant microcode output is routed over the instruction bus to the user developed circuitry and to the microprogram sequencer card.

The microprogram sequencer card provides a complete pipelined computer control unit that includes an address sequencer and an op code mapping memory for addressing the microcode contained on the WCS cards. The op code mapping memory is a 256 by 12-bit RAM that is used to store the starting addresses of microcode subroutines. The op code mapping memory translates an 8-bit instruction fetch from main memory into a starting address for the WCS card microcode. The microprogram sequencer card op code mapping memory can be automatically paged into the system processor upper 32K address range by invoking the appropriate microprogramming support software command. Once paged into the system processor, the op code mapping memory can be loaded, verified, and modified, as required.

CLOCK CONTROL LOGIC (INSTRUMENTATION) CARD

Six functions are performed by the circuitry on the clock control logic card. They are the control and routing of data along an internal bus, MPC timing, clock stop interrupt generation, clock stop status storage, manipulation of or with microprogram addresses, and the monitoring of system status, as shown in figure 5-5. These functions are addressed from the CPU, as shown in table 5-4.

Internal Bus

The clock control logic card is served by an internal bus, which is the channel for data transmitted to or from the CPU. Data is placed on the bus by the clock stop status register or by the 32-bit monitor register and data passes through the bus to be loaded into the control register or the microprogram address register.

Timing

The control store logic clock consists of an oscillator and logic to synchronize the oscillator and generate a system clock pulse. The oscillator output is a 18.0 MHz square wave. Synchronizing logic is controlled by three internal signals: run, clock phase step, and micro cycle. They originate in the debugging program or at the front panel switches. When they are all inactive, a synchronizing gate blocks the oscillator pulses from the prototype system and the rest of AmSYS29/10A. The line that controls this gate is called stop acknowledge and is available to the system. It must be active (clock stopped) when the system processor addresses the writable control store.

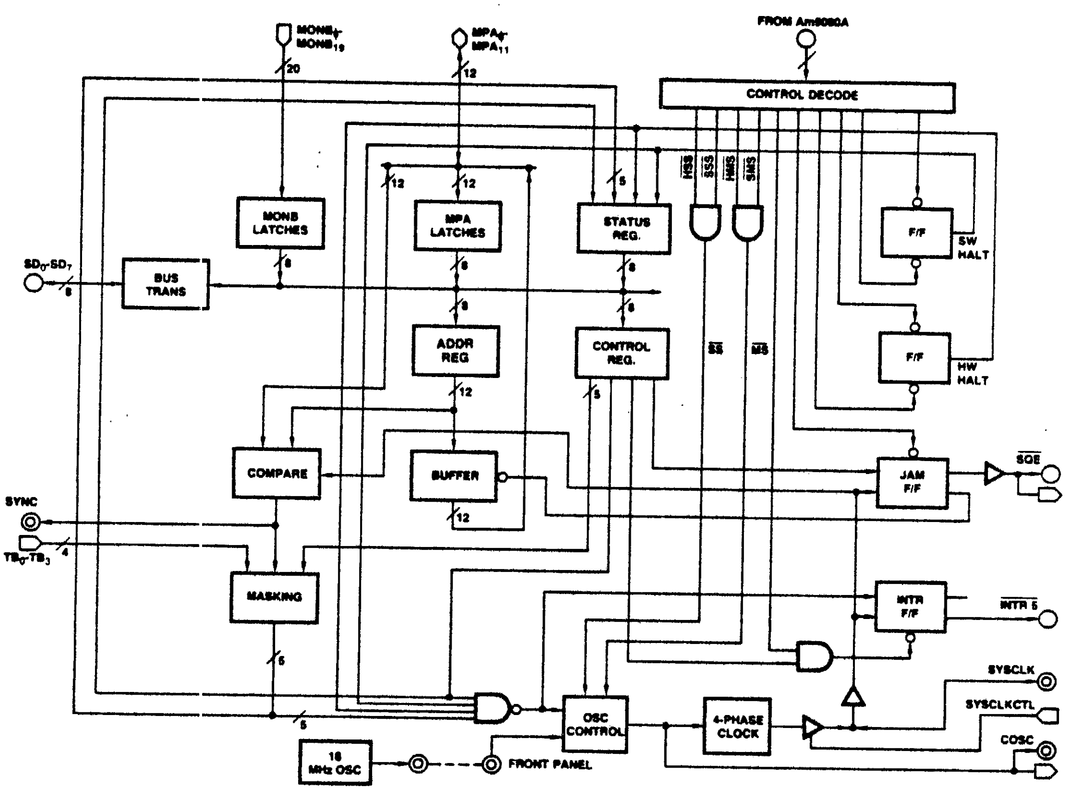


Figure 5-5. Clock Control Logic Card

Oscillator Control

When run is active, the synchronizing gate is open, making oscillator pulses available directly to the system, as shown in figure 5-6. These pulses feed internal logic that reshapes them into system clock pulses. The micro cycle step signal, through the synchronizing logic, creates one system clock pulse and permits oscillator pulses to pass through the gate to the system. Similarly, the clock phase step signal passes one oscillator pulse to the system; if repeated, every nth clock phase signal generates a system clock pulse. For synchronizing logic on the clock control system, n=4; for synchronizing logic in the prototype system, n can have any value greater than or equal to 3, subject to restrictions imposed by the writable control store.

Table 5-4. Clock Control Logic Card Address Allocations

ADDRESS	OPERATION	DATA	DESCRIPTION
00	WRITE	00	Jam address
00	WRITE	01	Stop MPC clock
00	WRITE	02	MPC micro cycle
00	WRITE	03	Enable MPC clock
00	WRITE	04	MPC clock step
00	WRITE	05	Reset interrupt latch
00	WRITE	06	(Not allocated)
00	WRITE	FF	(Not allocated)
01	WRITE	YY	Microprogram address register, bits 0-7
02	WRITE	XY	Microprogram address register, bits 8-11
03	WRITE	YY	Control register
03	READ	YY	Clock status register
04	READ	YY	Monitor register, external bits 0-7
05	READ	YY	Monitor register, external bits 8-15
06	READ	Y1Y0	Monitor register, external bits 16-19 (Y1) last address 8-11 (Y0)
07	READ	Y1Y0	Monitor register, last address 0-7

When run becomes active, the change alone is not enough to open the synchronizing gate. One system clock is required to turn on the synchronizing flip-flop; it is generated by one micro cycle signal or one to N clock phase signals, depending on the state of the clock logic when the system stopped. System clock opens the gate and the clock begins to run freely.

After run becomes inactive, or after execution of micro cycle step, both the gated oscillator line and the system clock line stop at their high levels. System clock stops in the low state following every fourth phase step - beginning with the third clock phase step after the step, and thereafter the seventh, eleventh, and so on.

When the page register is set to 3A or 3B, the address part of subsequent instructions always has a high-order byte whose high-order bit is 1 and whose other bits can be either 1 or 0; the low-order byte of the address is as specified in table 5-4. The operation is always a read or a write (as with memory); if the address is other than 00, the data goes to or from the specified register. For address 00, the data specifies a sub-operation involving the microprogram address, the MPS clock, or the MPS interrupt latch (X = don't care).

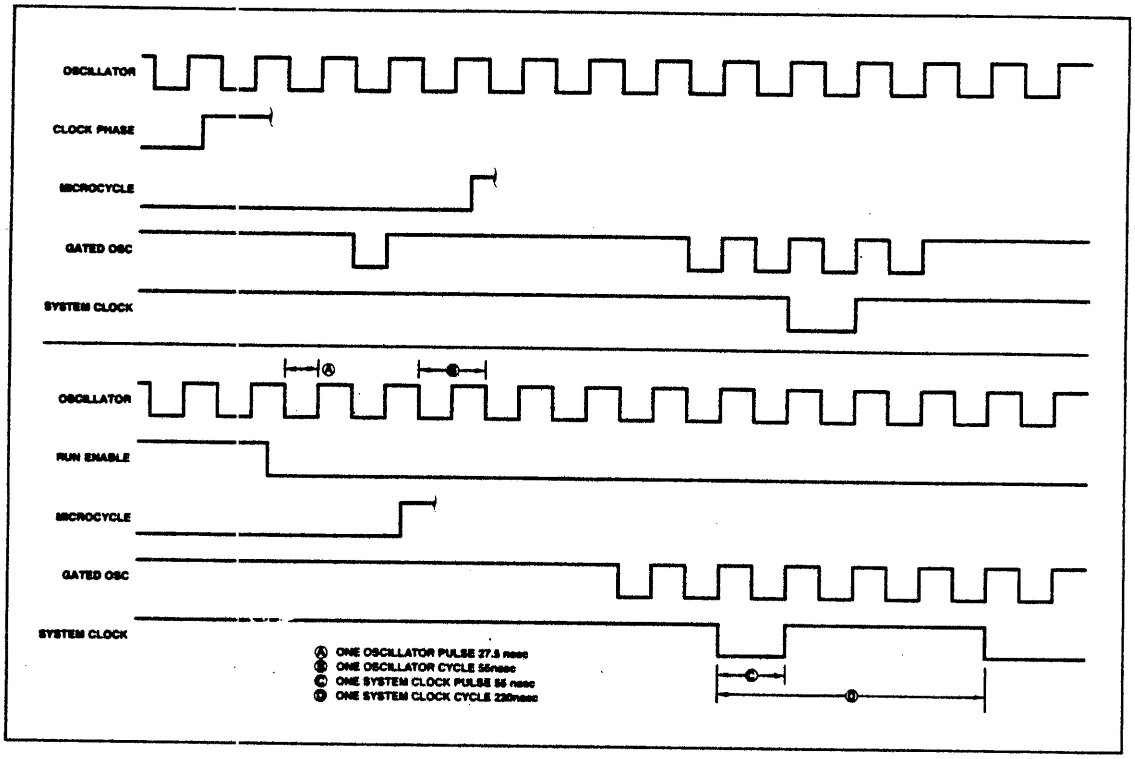

OSCILLATOR
CLOCK PHASE
MICROCYCLE
GATED OSC
SYSTEM CLOCK
Ⓐ
Ⓑ
OSCILLATOR
RUN ENABLE
MICROCYCLE
GATED OSC
SYSTEM CLOCK
Ⓐ ONE OSCILLATOR PULSE 27.5 nsec
Ⓑ ONE OSCILLATOR CYCLE 55nsec
Ⓒ ONE SYSTEM CLOCK PULSE 55 nsec
Ⓓ ONE SYSTEM CLOCK CYCLE 220nsec
Ⓒ
Ⓓ

Figure 5-6. Typical Oscillator Control Timing

System Clock

System clock pulses are generated in any of three ways: on the clock control logic card, in the system prototype, or on the microprogram sequencer card. Interconnections for the clock control logic and microprogram sequencer cards are shown in figure 5-7.

If the microprogram sequencer card is installed, it generates the

system clock from the controlled oscillator line. If the microprogram sequencer card is not installed, internal logic on the clock control logic card can generate the system clock. This logic is essentially the same as that on the microprogram sequencer card, but is disabled if the microprogram sequencer card is installed; its output includes a three-state buffer whose control line is automatically grounded, disabling the buffer when the microprogram sequencer card is installed.

Finally, if the system clock pulses are generated in the prototype, the clock logic on the clock control logic card must be disabled. A pin is available on a connector for grounding the buffer control line on the clock control logic card for this purpose.

When the prototype generates the system clock, it must observe the restrictions of the clock control logic card. This requires the positive level of the pulse to be at least twice as long as a full cycle of the oscillator.

As generated on the clock control logic and microprogram sequencer cards, the system clock phases are 55ns wide and the inactive level is three times as long as the active level.

Run Enable Control

The run input to the clock synchronizing logic is controlled by eight conditions, absence of any one of which causes run to become inactive, stops the clock, and could generate a clock-stop interrupt. Two of these eight conditions are controlled by the front panel switches and by software. In both cases, the conditions are asynchronous and are independent of each other and all other conditions. Therefore, if the clock is stopped from the front panel, it must be enabled from the front panel; if stopped by software, it must be enabled by software. Once enabled, however, the clock can be restarted from either source.

Five of the eight conditions are trap bits. Four trap bits originate in the prototype system; the fifth is a breakpoint signal, which indicates that the current microprogram address matches an address previously specified by the software and stored in a register on the clock control logic card. Any one of the five can stop the clock immediately upon arrival.

The eighth run condition is a bit in the control register (described later). When this bit is 0, the clock is totally inhibited except for micro cycle step and clock phase step.

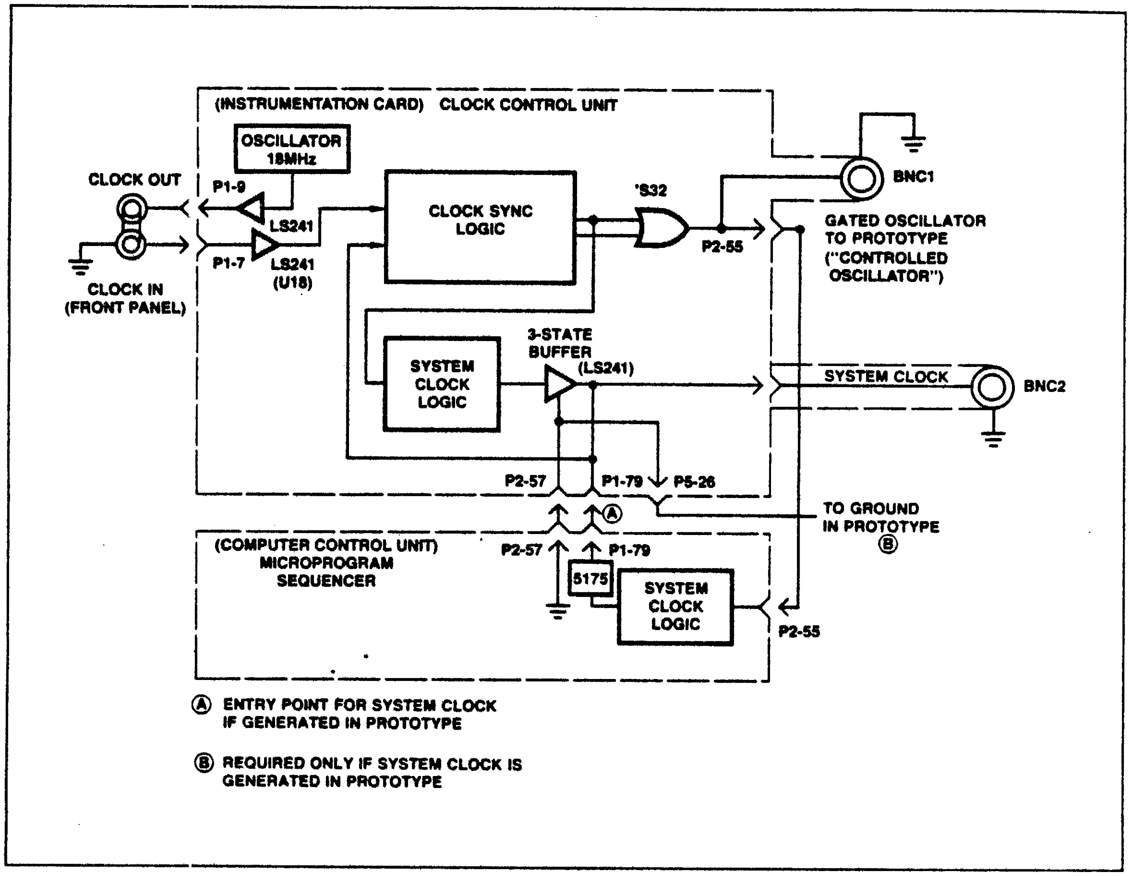
Control Register and Trap Mask

The control register is a set of eight flip-flops in one medium-scale IC that can be loaded by the CPU through the clock control logic card bus. Figure 5-8 and table 5-5 describe the functions of each position in the register. Five of the eight bits stored in this register are trap masks, which prevent the trap bits from stopping the clock. If the appropriate bit in the register is a logical 1, the corresponding trap bit is masked and cannot affect the run line.

Of the three remaining bits in the control register, one can stop the clock directly, unmasked and independent of any trap signals; one resets and masks the clock-stop interrupt latch; and one starts an address jamming operation into the microprogram sequencer.

Interrupt Generation

Only one of the five interrupts in AmSYS29/10A is generated on the clock control logic card. It comes from a flip-flop, shown in figure 5-9, that turns on when the run enable flip-flop turns off. Thus, when any trap bit, software or hardware command, or control register bit causes run to become inactive, run enable turns off, and the clock stops; at the same moment, the Interrupt flip-flop turns on. Its on state sends the interrupt 5 signal to the CPU.



Ⓐ ENTRY POINT FOR SYSTEM CLOCK IF GENERATED IN PROTOTYPE

Ⓑ REQUIRED ONLY IF SYSTEM CLOCK IS GENERATED IN PROTOTYPE

Figure 5-7. System Clock Generator Circuits

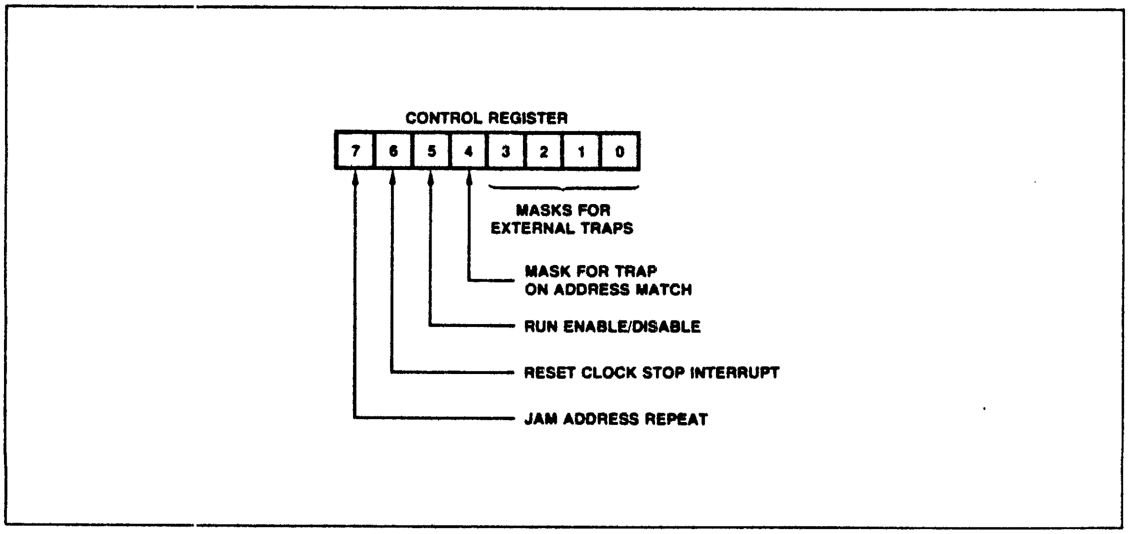


Figure 5-8. Control Register Bit Assignments

Table 5-5. Control Register Bits.

BIT	VALUE	EFFECT
0	0	Trap Bit 0 has access to clock
	1	Trap Bit 0 is masked
1	0	Trap Bit 1 has access to clock
	1	Trap Bit 1 is masked
2	0	Trap Bit 2 has access to clock
	1	Trap Bit 2 is masked
3	0	Trap Bit 3 has access to clock
	1	Trap Bit 3 is masked
4	0	Microprogram address comparator has access to clock
	1	Microprogram address comparator is masked
5	0	System clock is disabled except for clock step and micro cycle
	1	System clock is enabled for free-running operation
6	0	Reset and disable clock stop interrupt
	1	Enable clock stop interrupt
7	0	Address jam, if called, resets after one cycle
	1	Address jam is in effect for all cycles (CPU must supply microinstruction sequence)

Neither this interrupt nor any other interrupt is recognized by the CPU (specifically by the Am9080 microprocessor) unless the CPU's interrupt enable status is on. In the process of recognizing the interrupt and thereby branching to an interrupt service routine, the CPU automatically resets this status to off; no further interrupts can be recognized unless the enable interrupt instruction is executed to turn the status on again, ordinarily at the end of the service routine. If the clock stop interrupt flip-flop is still on at that time, the CPU immediately starts the service routine over again; therefore the flip-flop must be reset by writing 05 into address 00. (This is the low-order byte of the address; the high-order byte, as described under page register addressing, is anything from 80 to FF inclusive.) This write instruction must be executed before the enable interrupt instruction.

This write instruction is one of two that activates the asynchronous clear of the interrupt flip-flop. The other writes any byte that has a logical 0 in bit position 6 in address 03 (the control register), which turns off the interrupt flip-flop and holds it off. Again, the high-

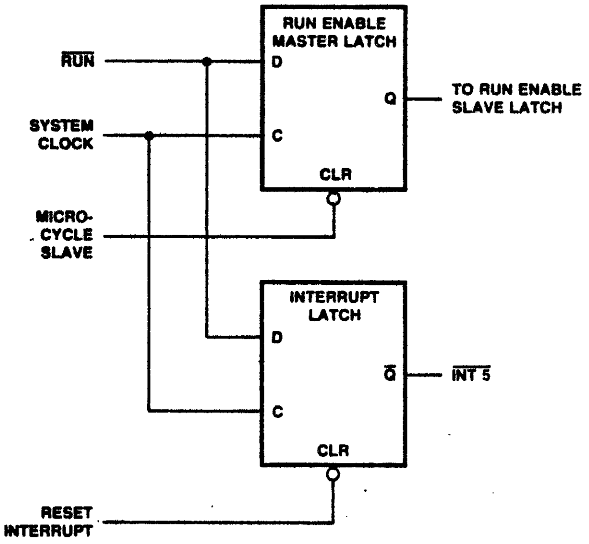


Figure 5-9. Interrupt and Run Enable Latch Relationships

order byte of the address is 80 to FF. The operating system provides an interrupt 5 service routine that outputs the message INT 5 at the console. If interrupt 5 is utilized by the user's software, the operating system service routine should be replaced by the user's routine.

Status Register

The status register stores the identity of any condition that stops the clock, as shown in table 5-6. This register is set by the trailing edge of the system clock pulse. The eight stop conditions arrive asynchronously, and the design of the clock synchronizing logic permits oscillator pulses to continue after a stop condition occurs, just long enough to generate one more system clock pulse. This pulse resets the Run Enable flip-flop in the synchronizing logic, and simultaneously stores the identity of the stop condition in the status register as well as setting the interrupt 5 latch (described under Interrupt Generation) and the monitor register.

Contents of the status register are available on the clock control logic card internal bus when the CPU issues a read command to address 03 (high-order address byte 80-FF). This command might be given, for example, in response to a clock-stop interrupt to bring out the stop condition for use elsewhere in the microprogrammed controller, by the prototype, or by the system support processor.

Table 5-6. Status Register

BIT	CAUSE OF CLOCK STOP
0	Trap Bit 0
1	Trap Bit 1
2	Trap Bit 2
3	Trap Bit 3
4	Microprogram address compare
5	Disabled by control register
6	Disabled by software command
7	Disabled by front-panel switch

When the microprogrammed controller clock stops, the bits with value 0 indicate the cause.

Microprogram Address Register

The MOS microprocessor may load the 12-bit address of any word in the microprogram into a pair of registers on the clock control logic card (addresses 01 and 02), shown in figure 5-10. From these registers, the address can be compared with an address from the microprogram sequencer or jammed into the sequencer.

Breakpoint and Sync Pulse

Contents of the microprogram address register are continuously compared with the address on the bus from the microprogram sequencer card or other microprogram sequencer. Whenever they are equal, the comparator generates a breakpoint signal that can stop the MPC clock if not masked. If the system stops on a breakpoint, a single micro cycle step returns it to free running. For repeated stepping through the program following the breakpoint, another stop condition must be imposed. Suitable stop conditions include setting control register bit 5 or one of the four trap bits to 0, issuing a halt command through the debugging program, or pressing the HALT switch on the front panel.

Whether masked or not, the breakpoint signal conditions a flip-flop that is set by the next system clock pulse. The microprogram sequencer address changes with every machine cycle, so if the system keeps running, the equal condition vanishes and the flip-flop output is available as a sync pulse at the front panel for monitoring with external instruments.

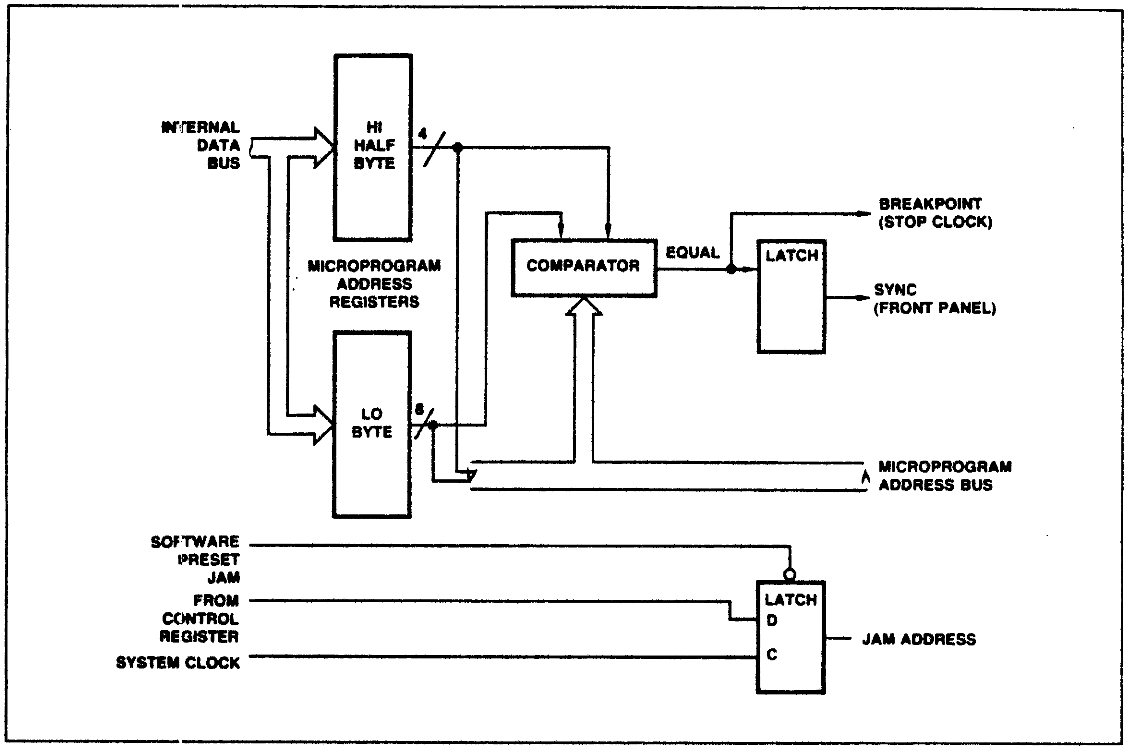


Figure 5-10. Microprogram Address Register

Since the same register is also the source of the jammed address when that operation is taking place, an equal condition also arises when the jammed address is placed on the bus.

Jam Address

While debugging, the microprogram must sometimes be made to start at a point where the sequencer would not require it to start during normal operation. To make this possible, a new starting address is jammed into the sequencer through the clock control logic card in either of two ways.

The first way begins by placing the desired address in the Microprogram address register. Then the microprocessor writes 00 in address 00 (high-order address byte 80-FF). This is the jam address command, which sets the jam enable flip-flop asynchronously.

The other address-jamming method places a 1 in the high-order bit of the control register (register set to 80 hexadecimal or higher). Then the jam enable flip-flop is set synchronously with the next system clock, regardless of the contents of address 00. In this case, the flip-flop stays on until the control register is reset, after which the next system clock pulse returns control to the sequencer for normal operation.

Turning on the jam enable disables the output of the sequencer, which is the normal source of microinstruction addresses, and transfers the new address from the clock control logic card register onto the microprogram bus. The microinstruction at the new address is fetched from the control store, and the MPC and prototype take the steps called for by that microinstruction.

In the simplest case, the steps do not affect the sequencer's microprogram counter, and the microprogram sequencer is ready to continue with the microprogram as if nothing had happened. If, however, the microprogram counter is changed as a result of the address jamming, the subsequent operation of the microprogram and the entire system is changed. Similarly, if the high-order bit of the control register is 1, the jam enable latch stays on and addresses are jammed into the sequencer repeatedly from the clock control logic card register, forcing the system operation into a tight loop.

If the system stops in a breakpoint and then a new address is jammed in, the breakpoint is lost unless it is explicitly restored, because the breakpoint and the jammed address both use the same clock control logic card register.

Monitor Register

Registers with a total capacity of 32 bits, shown in figure 5-11, store status conditions of various kinds on the clock control logic card. Twenty of these bits are supplied via cable to the card from a source in the prototype system and represent anything the designer wishes. The other 12 bits are the microprogram address.

All 32 bits are automatically loaded into the registers by the next system clock pulse. The registers, in turn, can be gated in groups of eight bits onto the internal bus of the clock control logic card, under control of the CPU.

Presence of any particular set of eight bits on the internal bus is fleeting, because the monitor registers are reloaded every clock cycle with the then-current contents of the microprogram address bus and the prototype cable. Therefore, their primary utility is to indicate the conditions that existed just prior to a stop, supplementing the bits of the status register.

WRITABLE CONTROL STORE CARD

The microprogram that controls the operation of the prototype system is kept in a writable control store (WCS), a special memory in which the microprogram can be modified easily and updated during the development process. In most cases, it will be replaced by a read-only memory when the prototype goes into production.

The WCS has a basic word length of 64 bits, which can be doubled to 128 bits; in either case, the maximum number of words is 4,096. However, these long words cannot be directly processed by the system processor, which works exclusively with eight-bit bytes when working with the microprogrammed controller. As a result, the WCS design includes a means to communicate with the prototype system in 64-bit or 128-bit words and with the system processor in 8-bit bytes. For the system processor, therefore, the maximum capacity of the WCS is 65,536 bytes.

Bits of the microprogram word are numbered from 0 to 127, right to left, in accordance with the customary representation of least and most significance. Bytes are numbered 0 to 15 (or 0 to F in hexadecimal) left to right to represent sequential addresses, as shown in figure 5-12. The 64-bit word is considered to be the right half of the 128-bit length, consisting of bits 0 to 63 or bytes 8 to F.

Whether the system processor or the prototype initiates a cycle, each memory chip in the WCS requires its own 10-bit address. Every chip on the board decodes all of these 10 bits to select one of 1,024 bits. The 10 address bits are part of a 12-bit microprogram address generated by the prototype system, and also part of a 16-bit address from the system processor, as shown in figure 5-13.

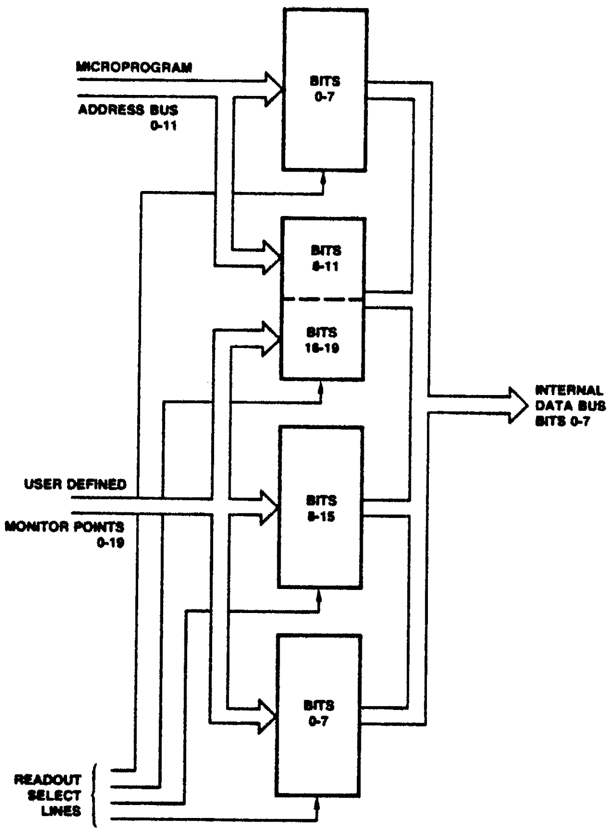


Figure 5-11. Monitor Register

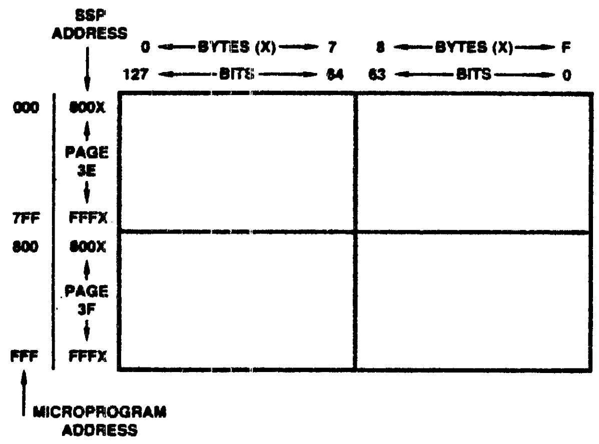


Figure 5-12. Organization of WCS

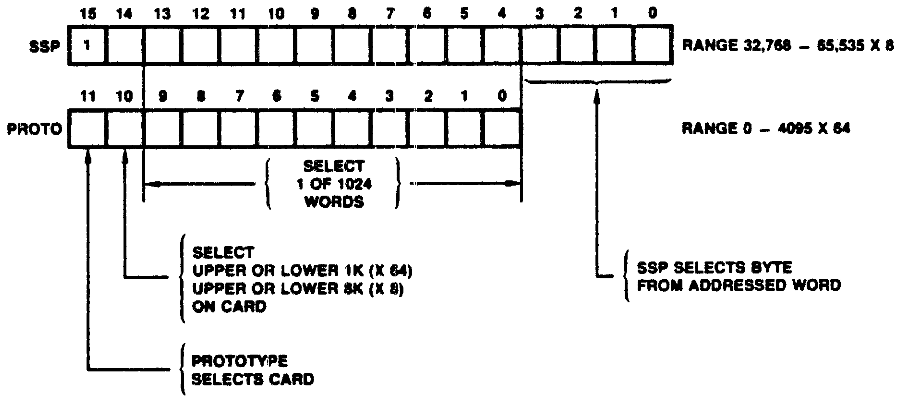


Figure 5-13. Microprogram Addresses From System Processor and Prototype

The two extra microprogram address bits are high-order bits. They are decoded to designate one of four banks of 1,024 words, without reference to word length.

When the system processor initiates a WCS cycle, its 16-bit address consists of the 10 that are decoded on the chip, two higher-order bits and four lower-order bits. The most significant bit is always a logical 1 because the system processor addresses the WCS as if it were the upper half of the system main memory. The upper and lower half of the WCS, which has the same maximum capacity as the main memory, are distinguished by the least significant bit of the page register, which is a logical 0 for the 0-2K range and a logical 1 for the 2K-4K range. Thus, the most significant bit of the address is replaced, functionally, by the least significant bit of the page register. The next most significant bit of the address specifies the upper or lower quarter within the half specified by the page register.

The four low-order bits select one byte from the addressed word to be read or written by the system processor. When the WCS has a 64-bit word length, each word consists of eight bytes, which are selected by bits 0, 1, and 2; address bit 3 in that case is always a logical 1, in accordance with the view that the 64 bits are in the right half of the 128-bit-wide space.

Control of the WCS resides in the address bits, except for the read and write commands and the system clock. Figure 5-14 shows the address assignments for the WCS.

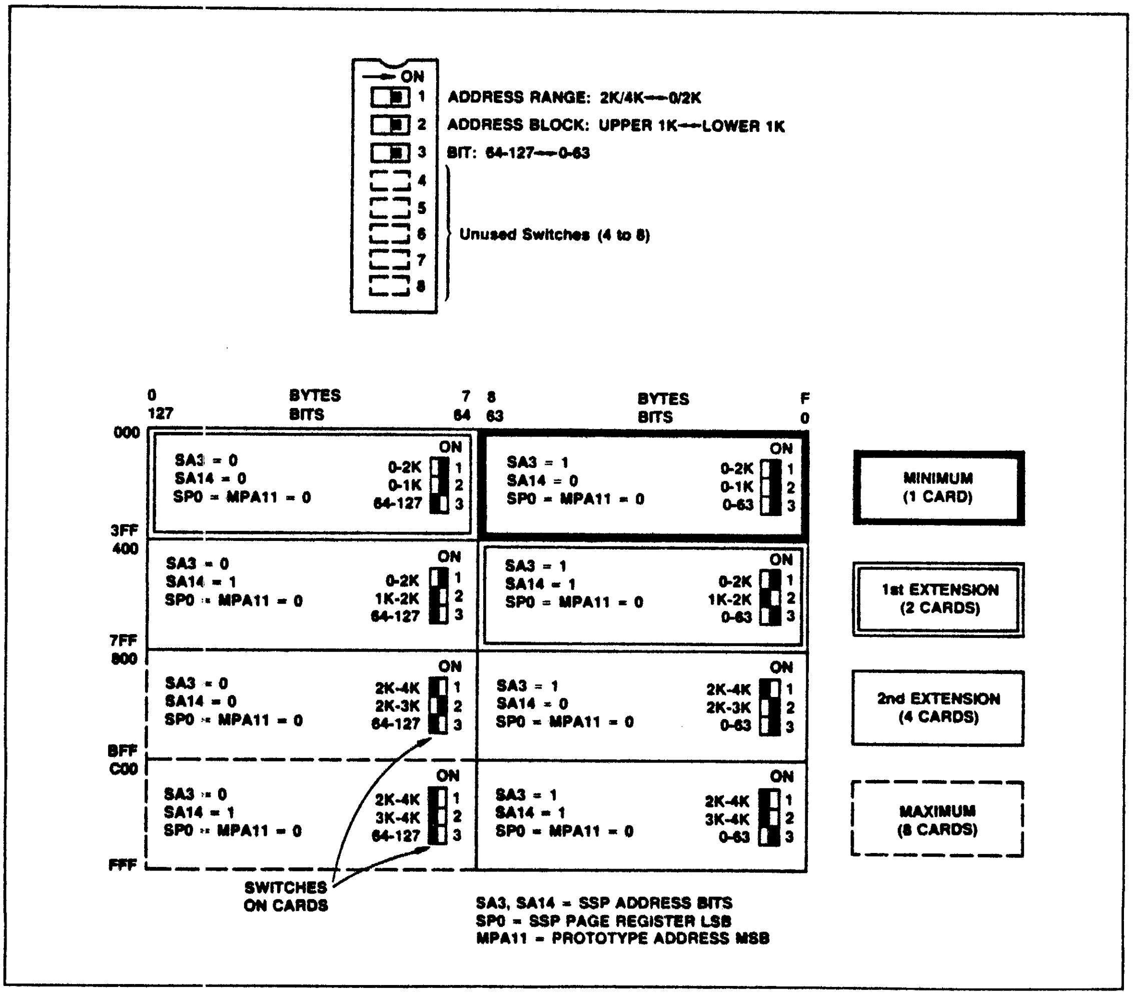


Figure 5-14. Address Assignment, WCS

Address bit 15 from the system processor is always a 1. Bit 11 of the microprogram address and bit 0 from the page register select the lower or upper half of the memory, in accordance with discrete switches that identify individual cards in the WCS. Address bits 4 to 13 inclusive from the system processor are common, bit for bit, with the microprogram address bits 0 to 9 at the outputs of the drivers that take them off the busses from their sources. System processor bits 3 and 14 and page register bit 0 enable the passage of data between the memory circuits and the system processor if the particular card is addressed; microprogram bits 10 and 11 similarly enable passage to the prototype. Whenever the WCS is addressed by the system processor, the system clock must be stopped, by any of the means described under run enable control, to prevent interference between the system processor and the prototype if they happen to address the WCS simultaneously.

MICROPROGRAM SEQUENCER (COMPUTER CONTROL UNIT) CARD

Operation of both AmSYS29/10A and the prototype system revolves around a microprogram sequencer, which generates the microprogram addresses for the control store. The sequence of these addresses, and the information they extract from the control store, is the secret of success of the prototype design; AmSYS29/10A exists specifically to perfect the sequencer, as well as the microprogram itself.

When the prototype design goes into production, it contains a microprogram sequencer tailored for the particular application. With that in mind, a preliminary sequencer suitable for use with AmSYS29/10A, called the microprogram sequencer is supplied with AmSYS29/10A. It is designed to work specifically with the development system, while retaining the flexibility that is necessary during the early stages of system design. The next several paragraphs describe this microprogram sequencer card specifically, but the same considerations apply to a custom sequencer design. Specifically, the mapping memory is writable, for the same reason that the control store is writable: to permit easy changes during development.

Microprogram Address Generation

The principal integrated circuits of the microprogram sequencer are the Am2909 and Am29811 Next-Address Control Unit, as shown in the block diagram of figure 5-15. The Am2909 and Am2911 are almost identical; their differences are discussed in the branch input paragraph. Other important components are the instruction register, the mapping memory, the pipeline register, the condition code multiplexer, and the event counter.

The instruction register is loaded with a machine instruction sent by the prototype system over its data bus. That instruction is decoded into an address for the mapping memory, which contains the starting address for the particular sequence of microinstructions required to execute each such machine instruction. This address, and other addresses derived during instruction execution, are sent to the writable control store along the microprogram address bus; the corresponding microinstruction return from the WCS along the microprogram data bus to the prototype system. Certain fields in the microinstruction, which affect how the condition code multiplexer and other elements of the sequencer are used, also are placed in the pipeline register in the MPS.

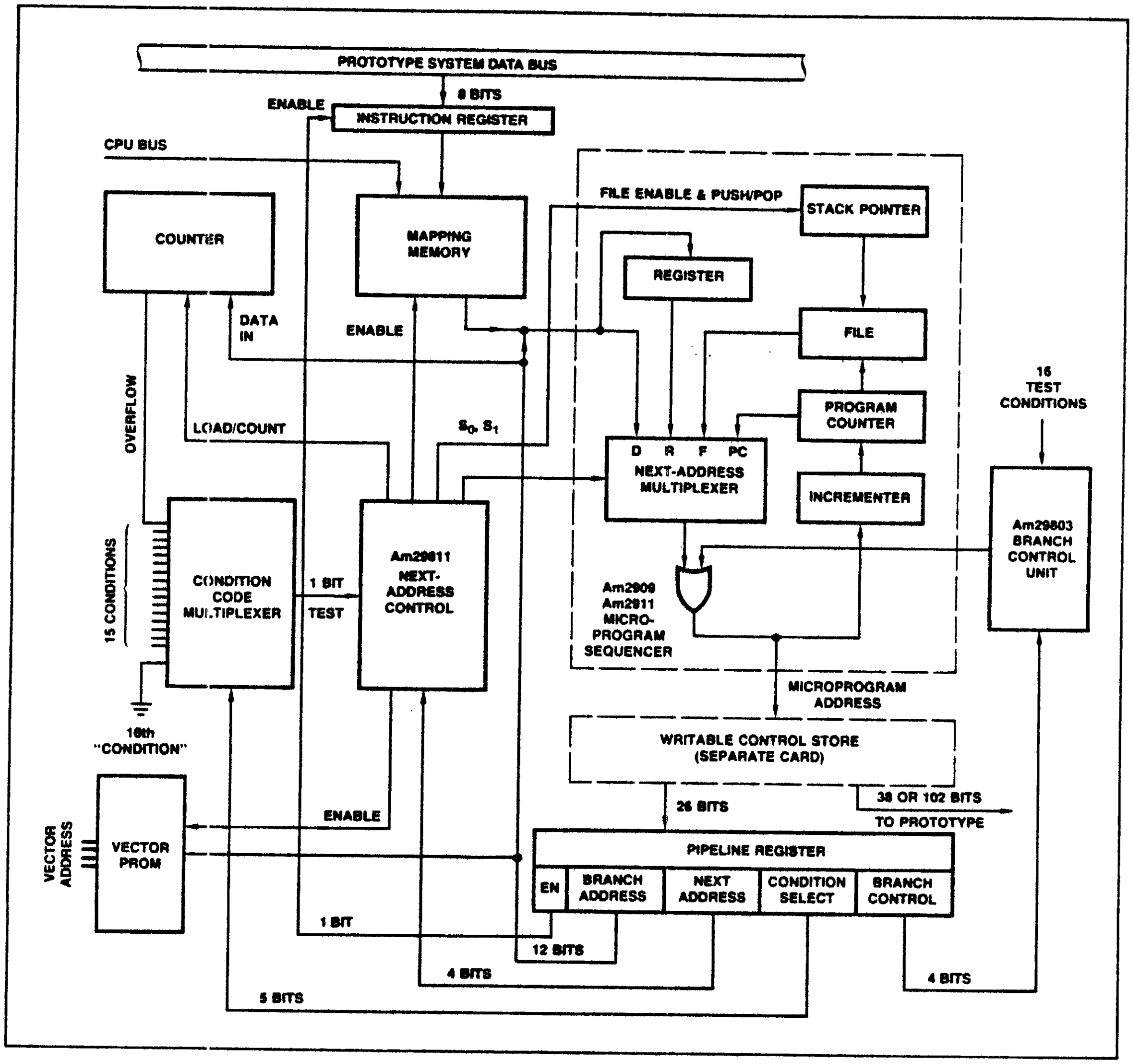


Figure 5-15. Microprogram Sequencer

The microprogram sequencer, diagrammed in figure 5-16, contains a next-address multiplexer that provides four different alternatives. These are direct input (D), the register (R), the program counter (PC), and the file (F). The last three are part of the sequencer. Outputs of both the mapping memory (including the vector map) and the pipeline register are operated in the three-state mode so that they can both feed the D input without interference.

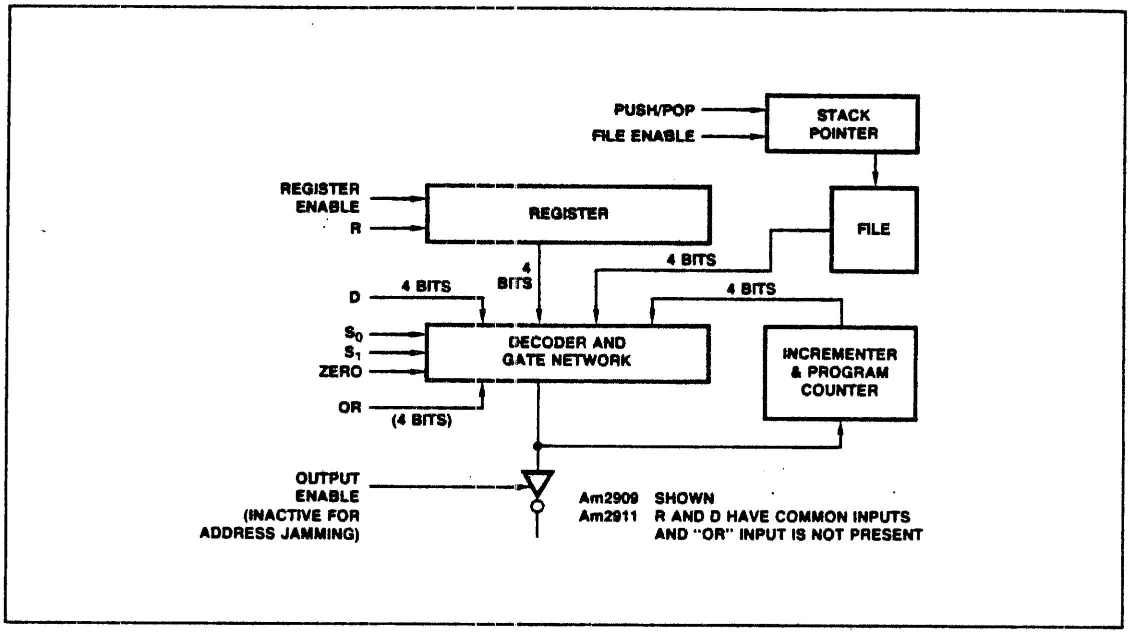


Figure 5-16. Microprogram Address Sequencer

The Am29811 next-address control unit makes use of these four inputs to establish the address of the next microinstruction to be executed.

This next microinstruction can be the first in a sequence of micro instructions, the next in a consecutive series, or a nonconsecutive microinstruction to which the current microinstruction jumps. For either the first in a sequence or the target of a jump, the Am29811 selects the D input of the multiplexer. The D input enables the output of either the mapping memory or the pipeline register, in which the branch address goes into the sequencer register as well as into the D input so that it becomes available at the R input of the multiplexerduring the next microcycle.This permits a conditional jump to one of two subroutines or branch addresses by using the branch address field of two sequential microinstructions and appropriately selecting either D or R input of the multiplexer.

For the third alternative, the next microinstruction in a consecutive series, the Am29811 selects the PC input of the multiplexer, so that the address in the program counter is the one sent to the writable control store. In addition, most of these consecutive microinstructions enable the outputs of the branch address field in the pipeline register, so that this branch address goes into the register in the sequencer.

These three alternatives use three of the inputs to the multiplexer. The fourth input, F, from the file, permits the microprogram to contain loops and use subroutines of up to four levels, with loops and subroutines intermixed, if desired. The file is a simple push-down stack within the sequencer, with a pointer that always indicates the most recently stored address.

The Am29811 can execute 16 different next-address control functions. These are listed in table 5-7; most of them are conditional. These functions are specified by the 16 combinations of four inputs that come from bits 16-19 of the pipeline register, together with a single condition code input from the condition code multiplexer - itself funneling 14 user-defined test conditions and two internal conditions onto this one line. These conditions can be such things as carry, zero, overflow, sign parity, or anything that can arise during the exectuion of the machine program in the prototype system. Outputs of the Am29811 enable the outputs of the pipeline register and of the mapping memory, and control the next-address multiplexer, the stack pointer (both in the sequencer), and the event counter, which can count executions of a loop or other events.

Table 5-7. Functions of Next-Address Control Unit

Mnemonic	Function	Inst Code	Addr Form	Cntr	Enbl
JZ	Jump zero	0000	D	**	P
CJS	Conditional jump to subroutine via pipeline	0001	PC or D	--	P
JMAP	Jump map	0010	D	--	M
CJP	Conditional jump via pipeline	0011	PC or D	--	P
PUSH	Push and conditionally load counter from pipeline	0100	PC	Load*	P
JSRP	Conditional jump to subroutine via registor or pipeline	0101	R or D	--	P
CJV	Conditional jump via vector	0110	PC or D	--	--
JRP	Conditional jump via register or	0111	R or D	--	P
RFCT	Repeat loop if counter not zero	1000	F or PC	DEC*	P
RPCT	Repeat pipeline address if counter not zero	1001	D or PC	DEC*	P
CRTN	Conditional return	1010	F or PC	--	P
CJPP	Conditional jump via pipeline	1011	D or PC	--	P
LDCT	Load counter	1100	PC	Load	P
LOOP	Test for end of loop	1101	PC or F	--	P
CONT	Continue	1110	PC	--	P
JP	Jump via pipeline	1111	D	--	P

* Conditional P = Pipeline

** Load and count both enabled M = Mapping memory

Mapping Memory

The instruction register length is 8 bits, so its contents can be decoded into as many as 256 starting addresses in the microprogram. This requires the mapping memory to have a maximum capacity of 256 words. (It need not have that many; for example, in some applications a much smaller programmable logic array is adequate.) Since the writable control store has a maximum capacity of 4,096 words, callng for a 12-bit microprogram address, this establishes the word length of the mapping memory as 12-bit.

In AmSYS29/10A, the mapping memory is divided into two parts. The major section is a 256 x 12 read/write memory containing starting addresses loaded into it by the system processor. In the complete design, this portion of the mapping memory would be implemented as a masked or programmable read-only memory (ROM or PROM).

The other section of the mapping memory is the vector map for executing interrupts. It contains 32 starting addresses and is implemented with two 32 x 8 PROMs (two bits in each word of each PROM are unused). It is addressed by the prototype system, but only when the next address does not come from the mapping memory or pipeline register. This secondary mapping memory is not accessible from the system processor.

Pipeline Register

When a microinstruction is read from the control store, most of its 64 or 128 bits go directly to the prototype system to manage operations there. However 26 of them return to the Microprogram Sequencer and are stored in the pipeline register. They are listed in table 5-8. The use of this pipeline register permits the address of a microinstruction to be determined and the memory cycle to fetch that microinstruction begun while the previous microinstruction is being executed. Thus the pipeline significantly improves the performance of the microprogram sequencer and the system as a whole. It leaves the results of the microprogram fetch, the microinstruction currently being executed, and the previous microinstruction available with respect to each other simultaneously; or in other words at any moment some registers in the prototype system contain the results of the previous microinstruction executed, others contain the microinstruction being currently executed, while data for the next microinstruction to be executed is on the microprogram bus.

Bits 0 through 11 of the pipeline register represent the branch address corresponding to the current microinstruction. This part of the register has two sets of outputs; one with three-state control which, when activated, is connected to the D inputs of the sequencers; and one with continuous outputs that are connected to the parallel-load inputs of the event counter.

Table 5-8. Pipeline Register Bit Assignments

BIT	DESTINATION	BIT	DESTINATION
0	Branch Address LSB	16	Next-Address Control
1	↓	17	↓
2		18	
3		19	Next Address Control
4		20	Condition Test Multiplexer
5		21	↓
6		22	
7		23	
8		24	Condition Test Multiplexer
9		25	Enable Instruction Register
10		26	Am29803 Branch Control
11	Branch Address MSB	27	↓
12	Not Used	28	
13	↓	29	Am29803 Branch Control
14			
15	Not Used		
Remainder to Prototype System			

Bits 12 through 15 are not used in the microprogram sequencer. Bits 16 through 19 are the instruction field for the next-address control unit. They specify which one of the 16 possibilities for the next address is actually to be used, as described in the next paragraph.

Some of these possibilities are conditional, depending on a single input to the control unit. This input in turn is multiplexed from 16 different conditions, 14 of which are brought in from the prototype system, one from the loop counter, and one of which is merely a ground connection. The multiplexing takes place in a pair of Am25LS2535 multiplexers, each of which accepts eight inputs. Bits 20-22 of the microinstruction select one of the eight conditions at each device; bit 23 selects one of the two multiplexers, effectively creating a 1-out-of-16 function from them; and bit 24 specifies whether the multiplexed condition line is presented to the next-address control unit in true or complement polarity.

Bit 25 enables the instruction register so that a new machine instruction can be loaded. It is a logical 1 only in the last instruction of a sequence.

Finally, bits 26 through 29 or the microinstruction are a command for the Am29803 Branch Control Unit, which is discussed in the branch input paragraph.

The ground connection as one of the conditions is a means of converting any conditional instruction in the next-address control to an unconditional one, simply by specifying the grounded input and the appropriate polarity as the condition.

Next-Address Control

The Am29811 next-address control unit, shown in figure 5-17 generates four inputs for the microprogram sequencers; S0, S1 Push/Pop (Pup), and File Enable; plus two inputs for the counter, and signals to enable the outputs of the pipeline register and the mapping memory.

The four combinations of S0 and S1 respectively specify the four sources of the next address: the D input, the internal register, the program counter, or the stack. If the D input is specified, either the pipeline or the mapping memory must be enabled so that the address will come from the proper source. However, for the Conditional Jump to Vector instruction, neither pipeline-enable nor map-enable output is active (both are high); the combination enables the vector map and feeds its output, like that of the pipeline or the mapping memory when enabled separately, to the D input of the multiplexer.

The particular address read from the vector map is determined by which one of up to 32 interrupts has occurred. The prototype system must include circuitry to encode the particular interrupt request into the five lines that the special PROM sees as an address, and to force the microprogram to jump to the routine that services the interrupt.

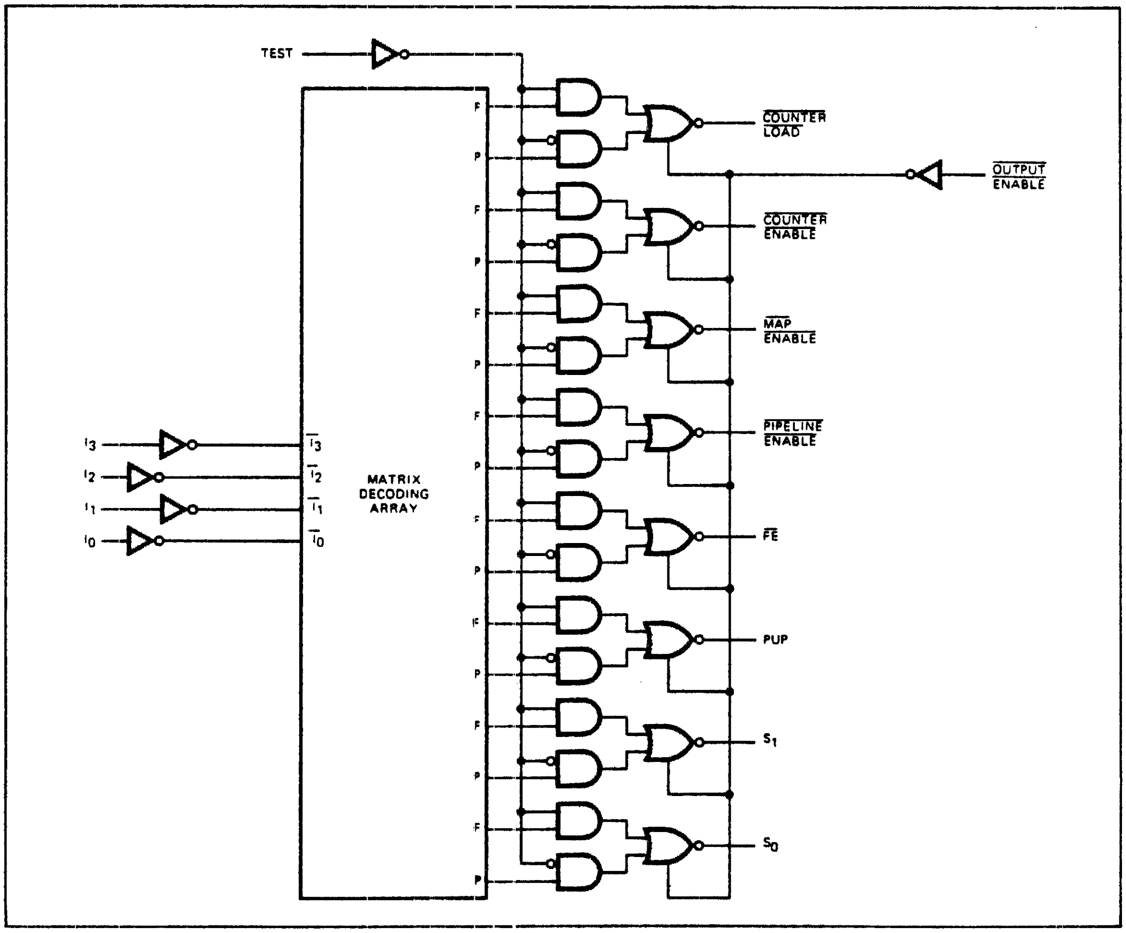


Figure 5-17. Next-Address Control Unit

In the absence of microprogram interrupt capability, the pipeline enable and the map enable outputs of the Am29811 could each drive their respective loads directly. However, AmSYS29/10A includes an Am25S139 decoder, shown in figure 5-18, by which the active state of each of these lines is given a meaning that depends on whether the other line is high or low.

Event Counter

The decoder chip that generates pipeline enable and map enable also interprets the counter load and count enable signals from the next-address controller. These signals when used alone, enable the event counter to be loaded from the pipeline register (bits 0 through 11, the continous outputs), or to count machine cycles, as delineated by the system clock. When the counter contains 0000, it is a condition that can control the execution of two commands to the next-address controller, namely Repeat Loop if Counter = 0 and Repeat Pipeline if Counter = 0.

When Count Enable and Counter Load are both active, the sequencer is forced to address 0; the result of the previous microinstruction's containing the command JZ (bits 16-19 = 0000) for the next-address control unit.

Both versions of the microprogram sequencer, Am2909 and Am2911, work with four-bit addresses, so the 12-bit addresses of the AmSYS29/10A microprogram require three sequencers in parallel. These consist of two Am2911s and one Am2909; the latter generating the low-order four bits of the microprogram address.

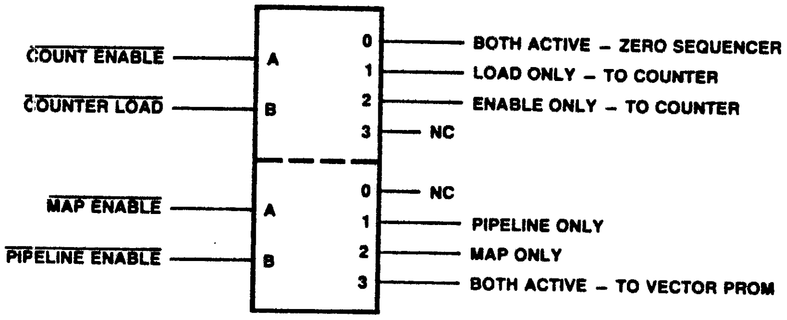


Figure 5-18. Next-Address Control Decoding

In the Am2911, one set of input pins is internally connected to both the R input and the D input so that both the next-address multiplexer and the internal register receive an externally generated microprogram address. In the Am2909, these functions are provided by separate pins, but in AmSYS29/10A, the pins are externally connected in pairs anyway. In addition, the Am2909 has a set of OR inputs through which any bit of the output is forced to 1, much as the Zero input forces all bits of the output to 0. In the AmSYS29/10A, these OR inputs come from corresponding outputs of the Am29803 16-way branch control unit.

Inputs to this branch control unit are four test bits and a four-bit command, much like that to the next-address control unit but originating in bits 26-29 of the microinstruction. Each command bit controls a branch on one of the test inputs; therefore, the 16 possible combinations of command bits control branching to the 16 addresses controlled by the Am2909 sequencer in 81 different ways, shown in table 5-9. The test inputs originate in the prototype system. When all the command bits are low, no test inputs are effective and no OR-input to the Am2909 is high; therefore, no branch occurs, and the next address is determined solely by the next-address multiplexer in the sequencer. If only one command bit is high, the corresponding test input, if active, forces a logical 1 into OR0, and thus a two-way branch on any of four different conditions. (Only OR0 is affected, regardless of which command bit or test is used, but the commands and tests correspond one-for-one.) Similarly, any combination of two command bits in conjunction with two corresponding test inputs forces a logical 1 into OR0 and OR1; the latter follow the test inputs, but no matter which pair of test inputs is selected, their condition is mapped to the OR0 and OR1 inputs. Likewise, selecting any three test inputs maps their condition to OR0, OR1, and OR2. Selecting all four test inputs sets up a one-to-one correspondence between all the test inputs and the OR outputs.

Since forcing an address through the sequencer in this way is tantamount to forcing a jump, clearly the point jumped to in the microprogram is critical. Since all such points are adjacent, use of the 16-way branch control should be in conjunction with a command to the next-address control to get a branch address from the pipeline register; this is an address ending in one, two, three, or four 0s corresponding to the number of test inputs, and designating the first line of a branch table elsewhere in the microprogram. Since the 16-way branch forces some of those ending 0s to be 1s, the actual branch is to a line somewhere in the table which in turn immediately branches to a microroutine whose action is a response to the particular test inputs used and their current value.

System Clock Generation

The microprogram sequencer also contains circuitry for generating the system clock pulse as an alternative source to the similar circuitry in the clock control logic card. For details see the system clock description earlier in this chapter.

Table 5-9. Branch Test Conditions

	Command Bits				Test Inputs				Branch Control Bits			
FUNCTION	I_3	I_2	I_1	I_0	T_3	T_2	T_1	T_0	OR_3	OR_2	OR_1	OR_0
No Test	L	L	L	L	X	X	X	X	L	L	L	L
Test T_0	L	L	L	H	X	X	X	L	L	L	L	L
					X	X	X	H	L	L	L	H
Test T_1	L	L	H	L	X	X	L	X	L	L	L	L
					X	X	H	X	L	L	L	H
Test T_0 & T_1	L	L	H	H	X	X	L	L	L	L	L	L
					X	X	L	H	L	L	L	H
					X	X	H	L	L	L	H	L
					X	X	H	H	L	L	H	H
Test T_2	L	H	L	L	X	L	X	X	L	L	L	L
					X	H	X	X	L	L	L	H
Test T_0 & T_2	L	H	L	H	X	L	X	L	L	L	L	L
					X	L	X	H	L	L	L	H
					X	H	X	L	L	L	H	L
					X	H	X	H	L	L	H	H
Test T_1 & T_2	L	H	H	L	X	L	L	X	L	L	L	L
					X	L	H	X	L	L	L	H
					X	H	L	X	L	L	H	L
					X	H	H	X	L	L	H	H
Test T_0, T_1 & T_2	L	H	H	H	X	L	L	L	L	L	L	L
					X	L	L	H	L	L	L	H
					X	L	H	L	L	L	H	L
					X	L	H	H	L	L	H	H
					X	H	L	L	L	H	L	L
					X	H	L	H	L	H	L	H
					X	H	H	L	L	H	H	L
					X	H	H	H	L	H	H	H
Test T_3	H	L	L	L	L	X	X	X	L	L	L	L
					H	X	X	X	L	L	L	H
Test T_0 & T_3	H	L	L	H	L	X	X	L	L	L	L	L
					L	X	X	H	L	L	L	H
					H	X	X	L	L	L	H	L
					H	X	X	H	L	L	H	H
Test T_1 & T_3	H	L	H	L	L	X	L	X	L	L	L	L
					L	X	H	X	L	L	L	H
					H	X	L	X	L	L	H	L
					H	X	H	X	L	L	H	H
Test T_0, T_1 & T_3	H	L	H	H	L	X	L	L	L	L	L	L
					L	X	L	H	L	L	L	H
					L	X	H	L	L	L	H	L
					L	X	H	H	L	L	H	H
					H	X	L	L	L	H	L	L
					H	X	L	H	L	H	L	H
					H	X	H	L	L	H	H	L
					H	X	H	H	L	H	H	H
Test T_2 & T_3	H	H	L	L	L	L	X	X	L	L	L	L
					L	H	X	X	L	L	L	H
					H	L	X	X	L	L	H	L
					H	H	X	X	L	L	H	H
Test T_0, T_2 & T_3	H	H	L	H	L	L	X	L	L	L	L	L
					L	L	X	H	L	L	L	H
					L	H	X	L	L	L	H	L
					L	H	X	H	L	L	H	H
					H	L	X	L	L	H	L	L
					H	L	X	H	L	H	L	H
					H	H	X	L	L	H	H	L
					H	H	X	H	L	H	H	H
Test T_1, T_2 & T_3	H	H	H	L	L	L	L	X	L	L	L	L
					L	L	H	X	L	L	L	H
					L	H	L	X	L	L	H	L
					L	H	H	X	L	L	H	H
					H	L	L	X	L	H	L	L
					H	L	H	X	L	H	L	H
					H	H	L	X	L	H	H	L
					H	H	H	X	L	H	H	H
Test T_0, T_1, T_2 & T_3	H	H	H	H	L	L	L	L	L	L	L	L
					L	L	L	H	L	L	L	H
					L	L	H	L	L	L	H	L
					L	L	H	H	L	L	H	H
					L	H	L	L	L	H	L	L
					L	H	L	H	L	H	L	H
					L	H	H	L	L	H	H	L
					L	H	H	H	L	H	H	H
					H	L	L	L	H	L	L	L
					H	L	L	H	H	L	L	H
					H	L	H	L	H	L	H	L
					H	L	H	H	H	L	H	H
					H	H	L	L	H	H	L	L
					H	H	L	H	H	H	L	H
					H	H	H	L	H	H	H	L
					H	H	H	H	H	H	H	H

L = Low, H = High, X = Don't Care

CHAPTER 6

AMSYS29/10A INTERFACE

MICROPROGRAMMED CONTROLLER SYSTEM CLOCK

The microprogrammed controller system clock can be used as supplied to run the prototype system or can be modified to meet user requirements.

The following paragraphs describe the microprogrammed controller system clock circuit supplied with AmSYS29/10A and how the user can modify the clock circuit to obtain the desired output for the prototype system. The clock circuit provides a multiphase clock. In single-step operation, a single clock pulse is generated. In microstep, a single microinstruction is executed while n clock pulses are generated (where n is a minimum of three clock pulses at a maximum clock frequency of 40 MHz).

The microprogrammed controller system clock circuit is illustrated in the block diagram of figure 6-1. Figure 6-2 provides a clock circuit timing diagram for the various modes of operation.

When both the microprogram sequencer card and the clock control logic card are installed in the system mainframe, a system clock control signal from the microprogram sequencer card supplies a ground to the clock control logic card three-state buffer. The low system clock control signal inhibits the clock control logic card four-phase clock generator output by placing the three-state buffer output in the high impedance state. Consequently, the microprogram sequencer card clock logic provides the system clock signal to the clock input of the Q4 D-latch shown in figure 6-1. When the microprogram sequencer card is not installed, the four-phase clock generator output is used as the system clock signal.

The following description assumes that the microprogram sequencer card is installed.

FREE-RUNNING CLOCK OPERATION

During free-running clock operation, the crystal-controlled oscillator output is routed through the coaxial jumper cable connecting the CLOCK-OUT connector to the CLOCK-IN connector. The signal at the CLOCK-IN connector is applied to the clock control logic card as the oscillator signal. The oscillator signal clocks the Q2, Q3/Q5, Q7, Q8 D-latches and is applied to the controller oscillator (COSC) OR gate on the clock control logic card.

Since the software microstep (SMS), hardware microstep (HMS), software single-step (SSS), and hardware single-step (HSS) are inactive (high), the Q3 and Q8 outputs are high. As long as the run signal is active (low), the Q5 output is low. The Q3/Q8/Q5 outputs are ANDed and the resultant output inhibits or enables the COSC OR gate to pass the oscillator signal to the user prototype and the microprogram sequencer card decoder. Since the run signal is active (low) and the Q5 output is low, the COSC signal is passed to the user prototype.

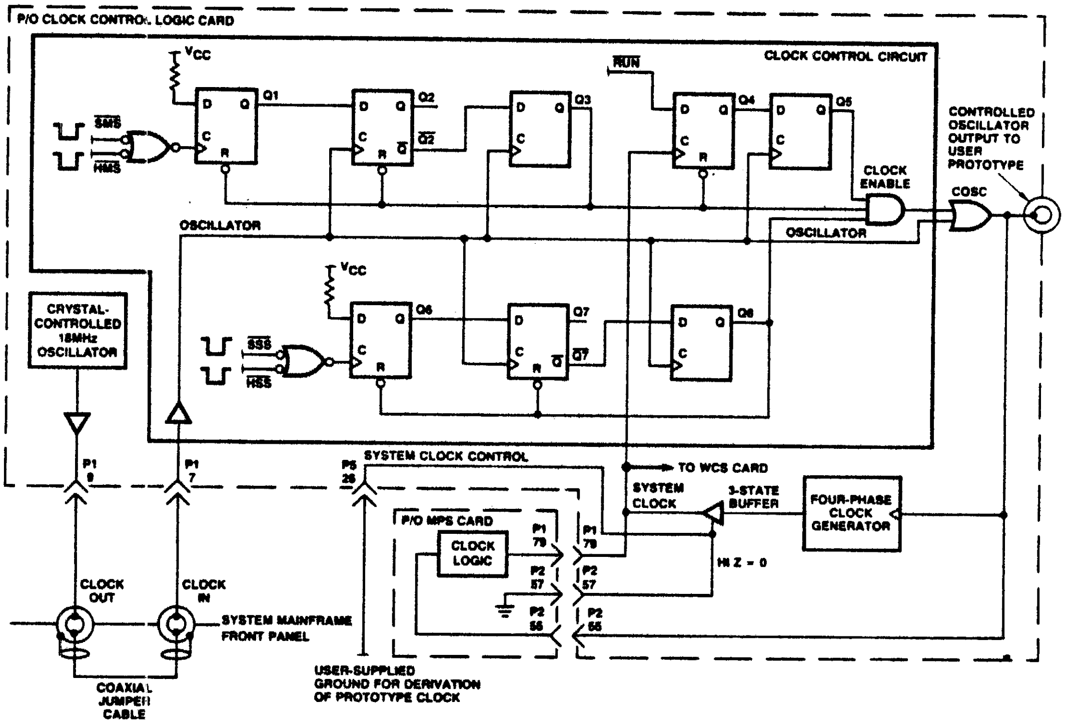
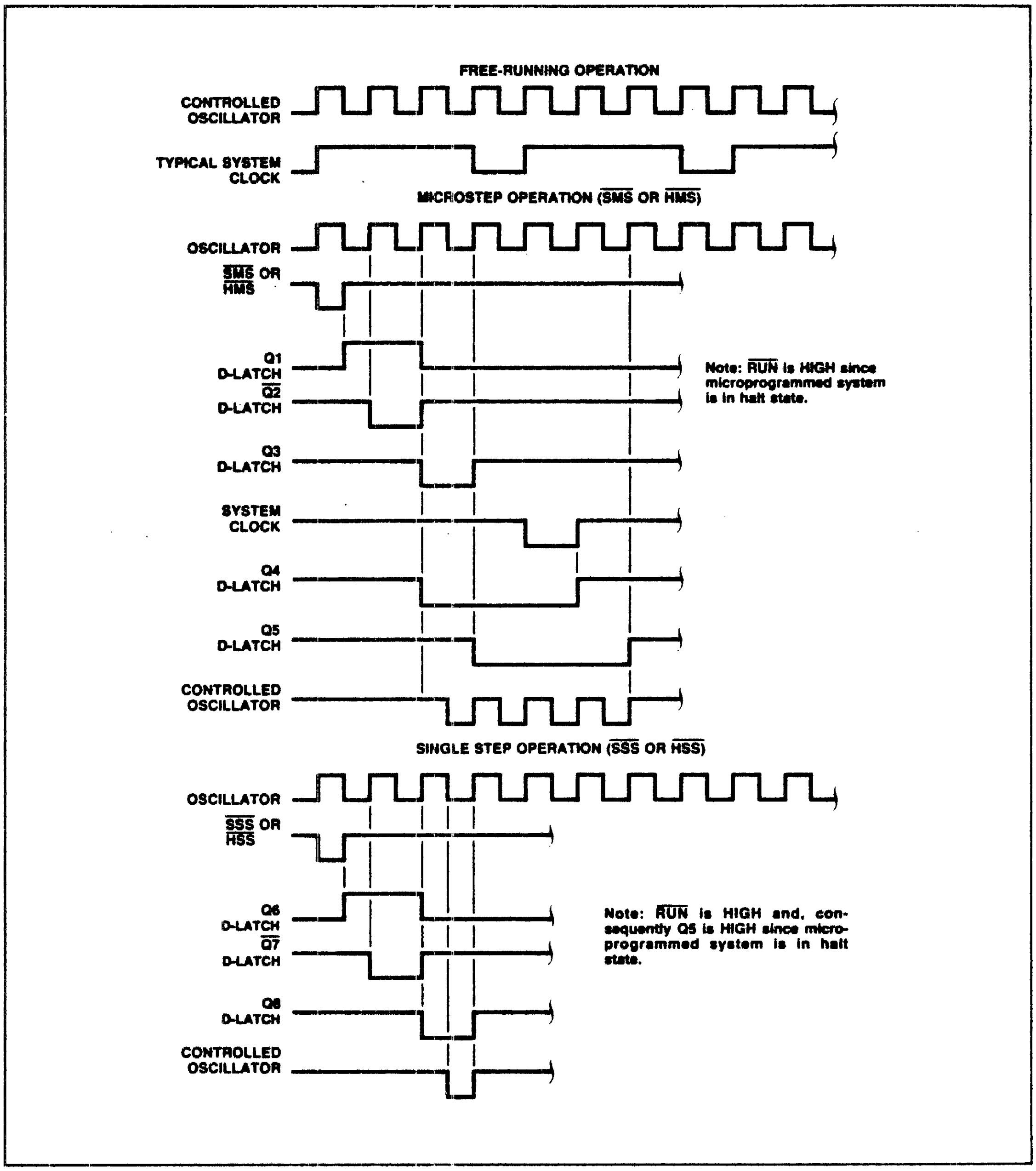


Figure 6-1. Microprogrammed Controller System Clock Circuit Block Diagram

MICROSTEP OPERATION

When the microprogrammed system is in the HALT state, microstep operation can be initiated by pressing the front panel MICRO CYCLE STEP switch (HMS signal) or by using the DDT29 MS n command (SMS signal). Whenever the HMS or SMS signal makes a low-to-high transition, the Q1 output goes high (see figures 6-1 and 6-2). The high Q1 output is applied to the D input of the Q2 latch. The Q2 output then goes low during the next rising edge of the oscillator signal. The low Q2 output is applied to the D input of the Q3 latch. The next rising edge of the oscillator signal clocks the Q3 output low.



Note: $\overline{RUN}$ is HIGH since microprogrammed system is in halt state.

Note: $\overline{RUN}$ is HIGH and, consequently Q5 is HIGH since microprogrammed system is in halt state.

Figure 6-2. Microprogrammed Controller System Clock Circuit Timing

The low Q3 output accomplishes two tasks. First, the Q1, Q2, and Q4 latches are reset so the Q1, Q2, and Q4 outputs are low, high, and low, respectively. When Q1 and Q2 are reset, the clock control circuit is ready to accept another HMS or SMS signal. Second, the low Q3 output causes the clock enable AND gate output to go low and enables the COSC OR gate to pass the applied oscillator signal as a controlled oscillator output to the user prototype and the microprogram sequencer card decoder. The next oscillator signal rising edge sets the Q3 output high and Q5 output low. The low Q5 output continues to cause the clock enable AND gate to supply an enabling low output to the COSC OR gate.

When the microprogram sequencer card decoder output causes the system clock signal to go through a low-to-high transition, the Q4 latch is clocked. Since the microprogrammed system is in the HALT state, the run signal is high at the D input of the Q4 latch. Consequently, the system clock rising edge clocks the Q4 output high. Then, the next oscillator signal rising edge clocks the Q5 output high, thereby driving the clock enable AND gate output high. Thus, COSC OR gate output goes high and controlled oscillator output is inhibited.

SINGLE-STEP OPERATION

When the microprogrammed system is in the HALT state, single-step operation can be initiated by pressing the front panel CLOCK PHASE STEP switch (HSS signal) or using the DDT29 SS n command (SSS signal). Whenever the HSS or SSS signal makes a low-to-high transition, the Q6 latch output goes high. The high Q6 output is applied to the D input of the Q7 latch. The next oscillator signal rising edge clocks the Q7 output low. The low Q7 output is applied to the D input of the Q8 latch. The next oscillator signal clocks the Q8 output low.

The low Q8 output accomplishes two tasks. First, the low Q8 output resets the Q6 and Q7 outputs low and high, respectively. Second, the low Q8 output causes the clock enable AND gate output to go low and enables the COSC OR gate to pass the controlled oscillator signal to the user prototype. Since the Q7 output is reset high, the next oscillator signal rising edge clocks Q8 and forces the AND gated output to go high and inhibit the COSC OR gate controlled oscillator signal output. Thus, when an SSS or HSS signal occurs, only one oscillator pulse is supplied as the controlled oscillator output to the user prototype.

USER-MODIFIED OSCILLATOR FREQUENCY

If a different oscillator frequency is desired, the oscillator output can be easily modified. The coaxial jumper cable can be removed from the front panel OSC OUT and OSC-IN connectors and an external oscillator can be used to supply an oscillator signal.

The on-board divide-by-four counter can be disabled by grounding SYSCLK CTL. The user can divide the COSC output (BNC connector on CCL card) by anynumber n the target system needs,where n is greater than or equal to 3.

Grounding SYSCLK CTL configures the CCL so that the SYSCLK BNC connector is an input, and the user must now supply the target system clock to this connector.

CAUTION

When the microsequencer (MS) and CCL are used together, the MS grounds SYSCLK CTL to disable the CCL SYSCLK output; the MS has its own divide-by-four counter, which has no output disable controls, to supply SYSCLK to the CCL and the target system. If the user wishes to supply SYSCLK, it is necessary to:

1. Not have the MS board in the backplane, or
2. Modify the MS to unground SYSCLK CTL, disable the divide-by-four output, and route the target system SYSCLK signal to the MS circuitry.

Backplane Connections:	
System Clock	J1-6 pin 79
System Clock Control	J2-6 pin 57
Top Card Edge Connections:	
System Clock	BNC2 Connector (located between P4 and P5 on card edge)
SystemClockControl	P5-26
Controlled Oscillator	BNC1 Connector (located between P3 and P4 on card edge) or J2-6 pin 55

DESIGNING SYSTEM CLOCK

In order for AmSYS29/10A to step the user system one microstep at a time, AmSYS29/10A requires a signal one microcycle in length (called system clock in AmSYS29/10A). When the user designs the clock circuit for the prototype system, provision should also be made for providing the system clock signal to AmSYS29/10A. The user-designed clock circuit in the prototype system can be connected to AmSYS29/10A as illustrated in figure 6-3.

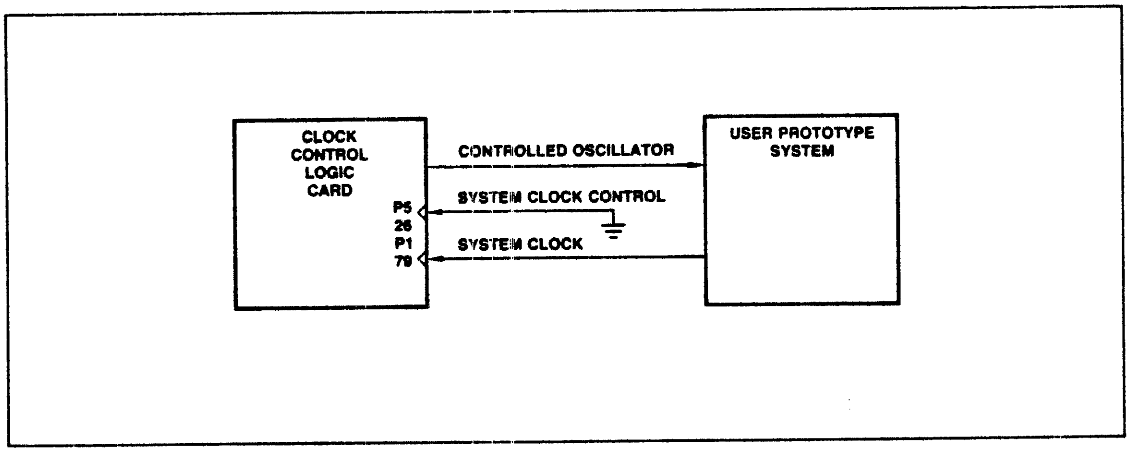


Figure 6-3. Connection Required For User-Generated System Clock

EXTERNAL PROTOTYPE CONNECTION

Connecting an external prototype package to the AmSYS29/10A microprogrammed system section requires three basic connections: (1) microprogram address lines, (2) microprogram data lines, and (3) clock connection. The AmSYS29/10A controlled oscillator signal can be used as supplied or modified as described. Figure 6-4 illustrates the interconnection of microprogram address and data busses between AmSYS29/10A and the external prototype package for various prototype configurations.

The microprogram address bus should be connected to the clock control logic card for all prototype configurations to permit address trapping and display of the last microprogram address. Display of the last microprogram address is affected by storing the last microprogram address in the appropriate registers included on the clock control logic card. The other clock control logic card registers are user-implemented for monitoring 20 bits of information for display purposes (data, test points, etc.), providing the appropriate connections are made to P4 of the clock control logic card.

Three ribbon cables are supplied with AmSYS29/10A. Two of the cables have 50-pin connectors on one end for connection to the WCS microprogram data bus connectors P3 and P4. One of the cables has a 26-pin connector on one end for connection to the micrprogram address connector P5. The free ends of the three cables must be connected to user-compatible connectors. If more than one connector is required for daisy-chaining buses from card to card in AmSYS29/10A, the user must fabricate the required cables.

Once AmSYS29/10A is connected to the external prototype package, all of the hardware and software features of AmSYS29/10A can be employed to develop and debug the microcode and prototype hardware. Tables 6-1 and 6-2 list the microprogram address and data pin assignments for the microprogram sequencer card, clock control logic card, and the WCS card. Clock and control signal, monitor bit, and test bit pin assignments are shown in tables 6-3, 6-4, and 6-5.

INTERFACE SIGNAL FUNCTIONS

The following paragraphs describe functions of the interface signals. Table 6-6 lists these signals: source, destination, signal type and where applicable, software interface information. Throughout the description, the following abbreviations are used for circuit card names.

MPS	Microprogram Sequencer
CCL	Clock Control Logic Card
WCS	Writable Control Store

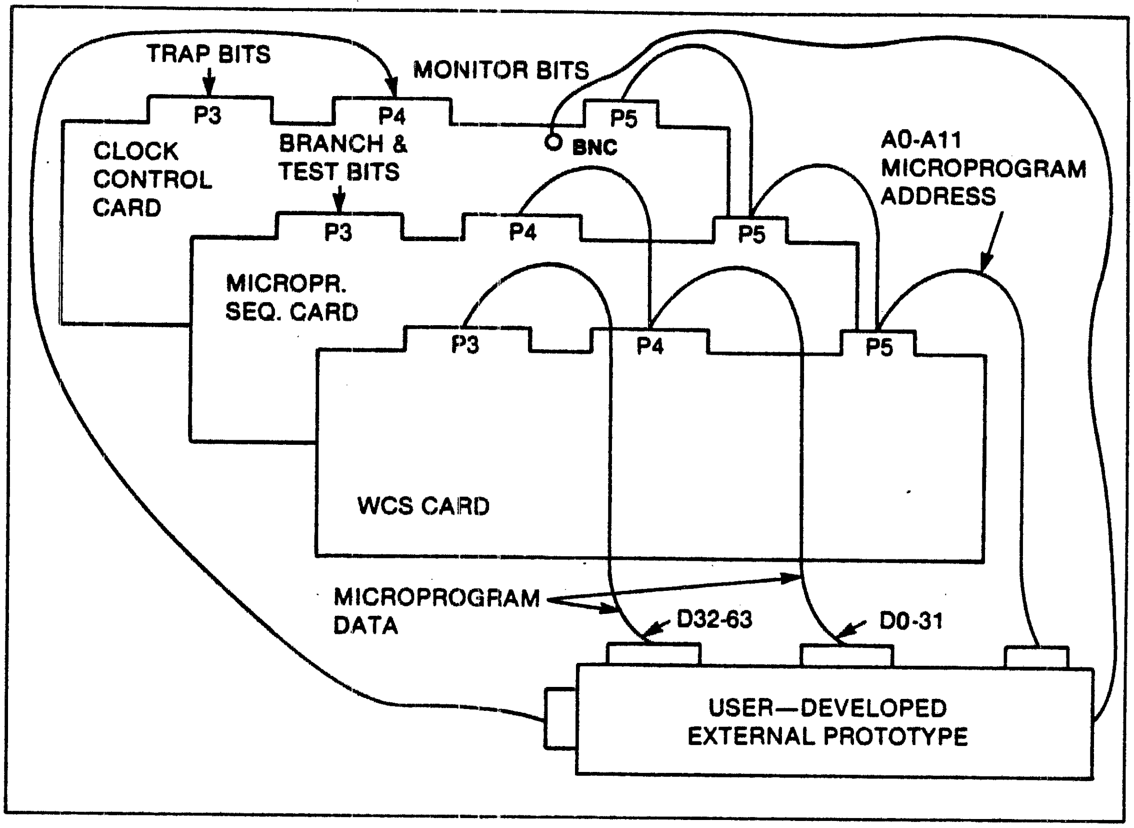


Figure 6-4. Connection of External Prototype Package to AmSYS29/10A

Table 6-1. Microprogram Address Pin Assignments for WCS, Microprogram Sequencer (MPS), and Clock Control Logic (CCL) Cards

ADDRESS BIT	MPS CARD	WCS CARD	CCL CARD
MPA 00	P5-2	P5-2	P5-2
MPA 01	-4	-4	-4
MPA 02	-6	-6	-6
MPA 03	-8	-8	-8
MPA 04	-10	-10	-10
MPA 05	-12	-12	-12
MPA 06	-14	-14	-14
MPA 07	-16	-16	-16
MPA 08	-18	-18	-18
MPA 09	-20	-20	-20
MPA 10	-22	-22	-22
MPA 11	-24	-24	-24

Table 6-2. Microprogram Data Pin Assignments for WCS Card

DATA BIT	CONNECTOR/ PIN	DATA BIT	CONNECTOR/ PIN	DATA BIT	CONNECTOR/ PIN
M00	P4 -4	M22	P4-34	M44	P3-20
M01	-5	M23	-35	M45	-21
M02	-6	M24	-40	M46	-22
M03	-7	M25	-41	M47	-23
M04	-8	M26	-42	M48	-28
M05	-9	M27	-43	M49	-29
M06	-10	M28	-44	M50	-30
M07	-11	M29	-45	M51	-31
M08	-16	M30	-46	M52	-32
M09	-17	M31	-47	M53	-33
M10	-18	M32	P3 -4	M54	-34
M11	-19	M33	-5	M55	-35
M12	-20	M34	-6	M56	-40
M13	-21	M35	-7	M57	-41
M14	-22	M36	-8	M58	-42
M15	-23	M37	-9	M59	-43
M16	-28	M38	-10	M60	-44
M17	-29	M39	-11	M61	-45
M18	-30	M40	-16	M62	-46
M19	-31	M41	-17	M63	-47
M20	-32	M42	-18		
M21	-33	M43	-19		

Table 6-3. Clock and Control Pin Assignments

CLOCKS/CONTROL	CLOCK CONTROL LOGIC	MICROPROGRAM SEQUENCER
COSC	BNC-1,P2-55	P2-55
System Clock	BNC-2, P1-79	P1-79
System Clock Control	P5-26, P2-57	P5-26, P2-57
SQE	P5-25	P5-25
TB0	P3-9	------------
TB1	P3-7	------------
TB2	P3-5	Trace "BREAK"
TB3	P3-3	Trace "SYNC"

Table 6-4. Monitor Bit Pin Assignments

MONITOR BITS	CLOCK CONTROL LOGIC	MONITOR BITS	CLOCK CONTROL LOGIC
MONB0	P4-2	MONB10	P4-22
MONB1	P4-4	MONB11	P4-24
MONB2	P4-6	MONB12	P4-26
MONB3	P4-8	MONB13	P4-28
MONB4	P4-10	MONB14	P4-30
MONB5	P4-12	MONB15	P4-32
MONB6	P4-14	MONB16	P4-34
MONB7	P4-16	MONB17	P4-36
MONB8	P4-18	MONB18	P4-38
MONB9	P4-20	MONB19	P4-30

Table 6-5. Test Bit Pin Assignments

TEST BITS	MPS CARD	TEST BITS	MPS CARD
TEST1	P3-7	TEST8	P3-23
TEST2	P3-8	TEST9	P3-13
TEST3	P3-5	TEST10	P3-14
TEST4	P3-10	TEST11	P3-16
TEST5	P3-9	TEST12	P3-15
TEST6	P3-12	TEST13	P3-18
TEST7	P3-11	TEST14	P3-17

MICROPROGRAM ADDRESS BUS (MPA0-MPA11)

The microprogram address bus provides a 12-bit address that is supplied to the WCS to select one of 4096 microinstructions to be executed. Normally, this bus is sourced by a microprogram address sequencer (Am2909, Am2911, or Am2910).

During a jam operation (see DDT29 subcommand JAM), the sequencer is inhibited from sourcing the bus by the sequencer disable (SQE) signal at which time the address is supplied by the address register on the CCL. The jam address therefore should be loaded into the address register (see DDT29 subcommand IR) previous to a JAM. Note that since the jam address is forced only onto this address bus, which is connected to the output of the sequencer, the sequencer address is not affected internally unless the instruction at the jam address is a branch type. The value on the address bus is also clocked into a set of registers on the CCL every SYSCLK cycle so that it may be examined when the clock is stopped (see DDT29 subcommand DLA and DTR).

SEQUENCER DISABLE (SQE)

The sequencer disable is a control line that normally stays low so that the sequencer output is enabled onto the MPA bus. During a JAM, SQE goes high to float the sequencer output to a high impedance state and at the same time enables a 3-state buffer on the CCL in order that the onboard address register can source the MPA bus at this time.

Table 6-6. Interface Signals

SIGNAL	FUNCTION	SOURCE	DESTINATION	SIGNAL INTERFACE
BDO-BD7	Main memory data bus	User system machine-level main memory	Microprogram sequencer	Connected to output of user system main memory for machine-level instruction set emulation.
BRTIO-BRT13	16-way branch test inputs	User system	Microprogram sequencer	User defined inputs to microprogram sequencer.
CLOCK-IN	Oscillator input	Clock control logic, user system	Clock control logic	Pulse train.
COSC	Controlled oscillator	Clock control logic	Clock control logic, microprogram sequencer, user system	Pulse train when system is running; high level when system is halted. DDT29 subcommand HALT forces COSC to a logical high state; SS subcommand activates COSC for one cycle; RUN subcommand sets COSC to free-run; CTL subcommand clears bit 5 of clock control logic control register, which stops the controlled oscillator.
INT5	Interrupt 5	Clock control logic	CPU card	Negative level. DDT29 subcommand CTL controls mask bit 6 in the control register for interrupt 5.
MO-M63	Microprogram data	Writable control store	Microprogram sequencer, user system	Changes when MPA changes and is stable after access time. DDT29 subcommand DTR displays MO-M63 value for previous SYSCLK.
MONBO-MONB19	Monitor bits	User system	Clock control logic	Must be stable prior to rising edge of SYSCLK. DDT29 subcommand DMB or DTR displays these lines logic levels at the previous SYSCLK.
MPAO-MPA11	Microprogram address bus	Microprogram sequencer, user-designed sequencer, clock control logic	Writable control store, clock control logic	Pulse at rising edge of SYSCLK. DDT29 subcommand DLA or DTR displays the MPA value for the previous SYSCLK.
OAO-OA4	Interrupt vector inputs as	User system	Microprogram sequencer	Interrupt vector via mapping memory.
SQE	STADROO-STADRO4 Sequencer disable	Clock control logic	Clock control logic, Microprogram sequencer, user-designated sequencer	Set high at JAM. DDT29 subcommand JAM forces CCL address register content onto MPA bus. CTL subcommand can be used to clear bit 7 of control register on CCL to force repeated asynchronous JAM, thus affecting a tight loop on a single microinstruction.
SYNC	Address breakpoint sync pulse	Clock control logic	Clock control logic, front panel	Single negative pulse at breakpoint address. DDT29 subcommand ECL loads breakpoint address into address register.
SYSCLK	System clock	Clock control logic, Microprogram sequencer, user system	Clock control logic, Microprogram sequencer, user system	Pulse train when system is running; high level when system is halted. DDT29 subcommand HALT forces SYSCLK to a logical high state at the leading edge of the second phase of the next SYSCLK; MS subcommand activates SYSCLK for up to one multiphase cycle until the next phase two; RUN subcommand sets SYSCLK to free-run; CTL subcommand clears bit 5 of clock control logic control register stopping COSC, which in turn stops SYSCLK.
SYSCLKCTL	System clock control	Microprogram sequencer, user system	Clock control logic	Static level, high enable.
TBO-TB3	Trap bits	User system	Clock control logic	User supplied signal. DDT29 subcommand CTL may change mask bits for trap inputs.
TESTI-TEST14	Test condition multiplexer inputs	User design	Microprogram sequencer	User defined condition under test.

MICROPROGRAM DATA BUS (M0-M63)

The microprogram data bus is the microinstruction as stored in the WCS at the location specified by the microprogram address. It is normally connected and clocked into the pipeline register on each SYSCLK cycle. If the microprogram sequencer is used, the low order 30 bits are clocked into the partial pipeline register on the MPS card to perform predefined functions. If the microprogram sequencer card is not used, all bits are open and can be freely assigned. If an additional WCS is used, it can be configured in parallel with the first one to obtain a 128-bit wide microprogram data bus.

OSCILLATOR INPUT (CLOCK-IN)

The CCL has a crystal and clock circuitry to generate a square wave train. This oscillator is brought out to a BNC connector on the front panel labeled CLOCK-OUT. A connection is then made from the CLOCK-OUT BNC connector to another BNC connector labeled CLOCK-IN immediately to the left. Instead of using the supplied oscillator, the user can remove the connection to CLOCK-OUT and connect the CLOCK-IN BNC to an oscillator of his own design. The signal at the CLOCK-IN BNC, whether it be the CCL or user-supplied, is then brought back into the CCL to generate the COSC signal. Note that this signal, at CLOCK-IN BNC, is a continuous pulse train that runs even when SYSCLK is stopped.

CONTROLLED OSCILLATOR (COSC)

When the CLOCK-IN basic oscillator is brought back into the CCL, it is gated by an oscillator control circuit (see figure 6-5). The gated oscillator then becomes the controlled oscillator, COSC. The purpose of the gating is to force COSC into a high state regardless of the state of the basic oscillator when the system is halted. Therefore, the difference between the two oscillators is that while the non-gated oscillator never stops, the gated one stops when the system is halted. The gated COSC is then used to generate the system clock, SYSCLK. Thus when the system halts and COSC stops, which in turn leaves SYSCLK idle.

The CCL takes this onboard COSC to generate a four phase SYSCLK. The COSC is also input to the MPS where another SYSCLK is generated. However, when the MPS is in the system, the CCL SYSCLK is disabled (see the following SYSCLKCTL description).

The user can also choose to design his own SYSCLK from the COSC, in which case he also needs to disable the CCL SYSCLK (see SYSCLKCTL).

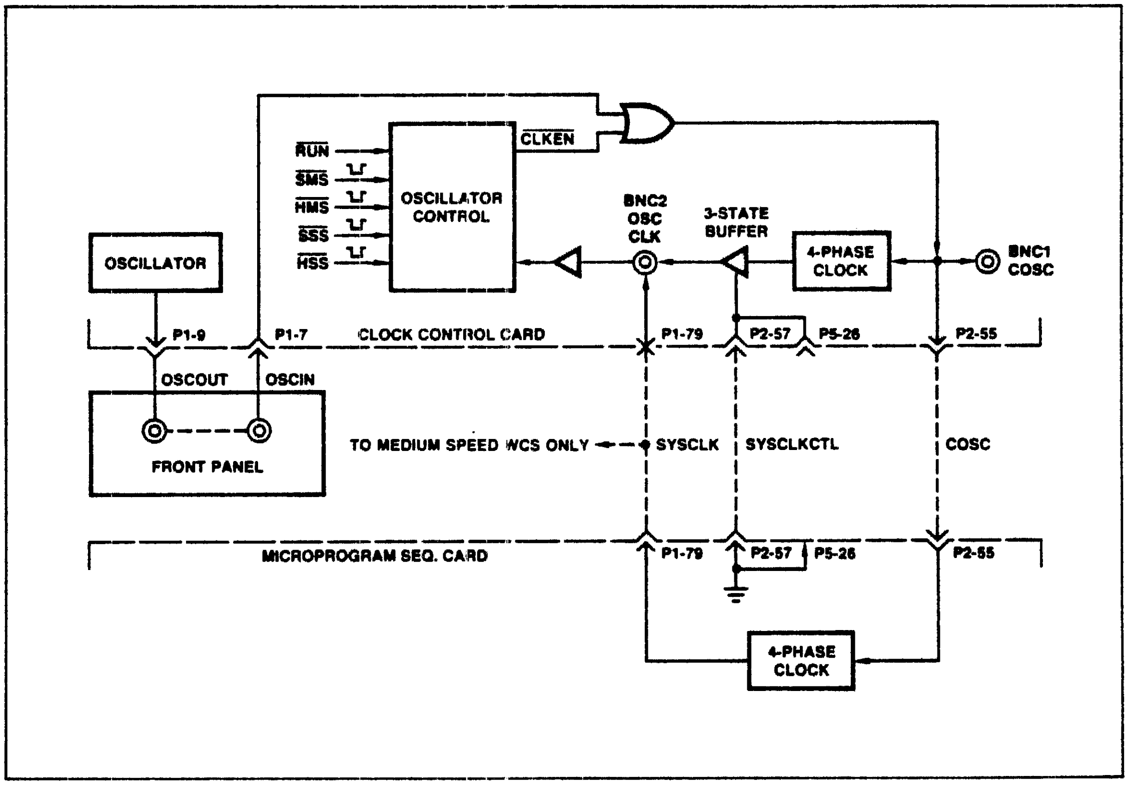


Figure 6-5. System Clock Interfacing

SYSTEM CLOCK (SYSCLK)

The SYSCLK signal is the clock that constitutes a microcycle. Its rising edge is normally used to clock the pipeline register and most of the Am2900 components. It can be generated from one of three places.

Both the CCL and MPS generate a four-phase SYSCLK from the COSC. The user can also generate a SYSCLK signal if desired. There is one restriction, the minimum SYSCLK allowed is a two-up one-down combination from the COSC. In other words, the high level of the SYSCLK must consist of at least two COSC pulse, and the low level at least one, as shown in figure 6-6. The minimum COSC pulse width is 25 nsec.

SYSTEM CLOCK CONTROL (SYSCLKCTL)

As previously mentioned, both the CCL and the MPS generate a four-phase SYSCLK from the COSC. The clock, as generated on the CCL, is buffered through a 3-state gate before it is connected to the SYSCLK line. The control line for this gate is the SYSCLKCTL signal. When both the CCL and MPS are in the system, the MPS supplies a low level on the SYSCLKCTL line into the CCL, thus floating the gate and effectively disconnecting the CCL clock from the SYSCLK line. In this case, the SYSCLK line becomes sourced by the MPS, rather than the CCL as is normally the case when the MPS is not present. Should the user elect to design an MPS and SYSCLK, a logical low to the SYSCLKCTL line into the CCL also must be supplied so that the CCL generated clock does not cause any interference on the SYSCLK line.

MONITOR BITS (MONB0-MONB19)

The monitor bits are twenty general purpose lines supplied by the user. The logic states of these lines are clocked into a set of registers on the CCL every SYSCLK cycle so that they can be displayed when the system halts (see DDT29 subcommand DMB and DTR). The function therefore is similar to a logic state analyzer in that the monitor bits provide a general purpose single level trace capability, allowing users to peek into the problem areas of their system.

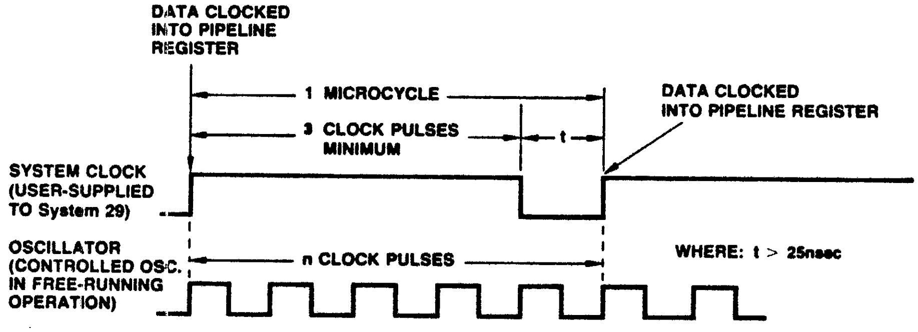


Figure 6-6. User-Supplied System Clock Signal Timing Considerations

ADDRESS BREAKPOINT SYNC PULSE (SYNC)

Whenever the MPA value compares with that in the CCL address register, a pulse is generated and routed to a BNC connector on the front panel. This output can be used as a trigger for an oscilloscope or a logic-state analyzer. This same pulse also may be used to stop the COSC and SYSCLK, provided that bit 4 of the CCL control register is cleared (see DDT29 subcommand CTL).

INTERRUPT 5 (INT 5)

When SYSCLK is stopped, an interrupt 5 can be generated to activate a service routine in Am9080A memory. This requires two conditions: SYSCLK stopped and the interrupt 5 mask bit in the control register (bit 6) enabled. When both conditions are met, a restart 5 (RST 5) instruction is executed in Am9080A.

Naturally, the user is responsible for depositing, at location 28 hex, either the service routine itself or a branch instruction to the service routine. The interrupt 5 can be cleared in one of two ways: directly from the control register by inhibiting bit 6, or under software by writing a data value of 05 into location 8000 with the page register set to the instrumentation page (3A or 3B). Note that the Am9080A interrupt acknowledge does not automatically clear the interrupt. Furthermore, the user is also responsible for enabling the CPU interrupt again by executing an enable interrupt (EI) instruction at the end of the service routine.

TRAP BITS (TB0-TB3)

The trap bits are four control signals supplied by the user to asynchronously stop the COSC, and thus the SYSCLK. The conditions under which this would happen are defined by the user. Four of the CCL control register bits (bits 0-3) act as masks for these trap signals so that, when not in use, the user may ignore these traps by turning the masks off. The signals should be synchronized with SYSCLK or be clean and free of glitches.

NOTE

The following groups of interface signals are related to the MPS only. Microcode bit assignment for the partial pipeline register on the microprogram sequencer card is shown in figure 6-7. If a user-designed sequencer is employed, the following discussions do not apply.

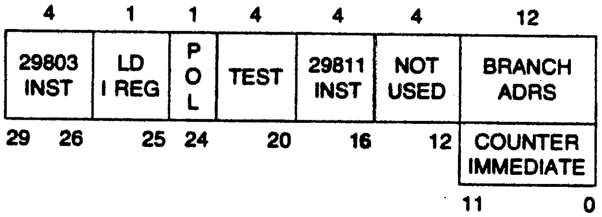


Figure 6-7. Microprogram Sequencer

TEST CONDITION MULTIPLEXER INPUTS (TEST1-TEST14)

During a conditional microinstruction, one of sixteen possibilities can be selected as the condition under test. Fourteen of these are defined and supplied by the user. Of the remaining two predefined conditions, test 0 is connected to ground so that either a logical 1 or a logical 0, depending on the polarity control, can be forced; test 15 is connected to the counter to detect an end-of-count (zero) condition. Test bits should meet a set-up time to the SYSCLK.

16-WAY BRANCH TEST INPUTS (BRT10-BRT13)

These four lines are defined and supplied by the user system to the inputs of the Am29803 chip on the MPS. Depending on the state of these four lines, the Am29803 generates a four-bit index OR'd into the Am2909 sequencer, thus changing the microprogram address to one of sixteen different values. It can be thought of as a hardware equivalent of a CASE statement in higher level languages. Note that while in this case all four inputs affect the operation simultaneously, only one of the sixteen test inputs has any effect in the case of a conditional instruction; the states of the other fifteen test inputs are ignored.

INTERRUPT VECTOR INPUTS (STADR0-STADR4 or OA0-OA4)

When a micro interrupt occurs, this 5-bit interrupt vector goes through a mapping memory and is transformed into a 12-bit microprogram address. This then becomes the interrupt service entry point for this particular interrupt.

With five bits, a total of 32 micro interrupts are possible. Care must be taken to ensure that the vector stabilizes prior to the recognition of the interrupt request.

MAIN MEMORY DATA BUS (BD0-BD7)

Where a machine-level instruction set is emulated by microprogram, the program is stored in a higher level (and typically slower) memory. The BD bus is connected to the output of that memory. During a machine-level instruction fetch, the instruction is clocked through the bus into register on the MPS card. The output of this instruction register is transformed by the address mapping memory into a 12-bit microprogram address. That address is the start of the microprogram that implements the particular machine-level instruction. In some controller applications, there is no machine-level instruction set; all programs are written at the micro level and the BD bus is not significantly employed.

SIGNAL CONNECTOR INFORMATION

Signals available to the user are listed in tables 6-7 through 6-12. Table 6-7 lists the available signals in alphabetical order. A brief description of the signals as well as the signal source also is shown on the table. Tables 6-8 through 6-12 provide signal-to-connector pin information. The tables are presented in connector number order (connector P1 is shown first and connector P5 is shown last). The signals and pin numbers listed are associated with the boards shown across the top of the table. In table 6-7, input signals and input pins are shown enclosed in brackets ([]).

Table 6-7. System 29 Signal Chart

NAME	SOURCE	DESCRIPTION
+5V	P1/2-5,81-83	System V_{CC} line; +5V @ 75A
+12V	P1/7,8	Available on J1 to J5 only; +12V @ 1.7A
-12V	P1/79,80	Available on J1 to J5 only, -12V @ 1.7A
[BD0-7]	[MPS: P2/43-50] [38]	Opcode portion of application data bus
[BRT0-3]	[MPS: P3/2, 3, 4, 1]	16-way branch test inputs on CCU
CNDTR	CPU: P5/11	System O/P to indicate ready to XMIT data to control console
[CNRCVD]	[CPU: P5/31]	Received data from control console
CNRTS	CPU: P5/14	System request to send to console
CNTXD	CPU: P5/5	Transmit data port to console
CP/M	CPU: P1/34	(For internal use only)
COSC	CCL: P2/55	BIPOLAR controlled oscillator O/P
[CTS]	[CPU: P5/6]	Clear to send on spare serial port (I/P)
[DSR]	[CPU: P5/7]	Data set ready on spare serial port (I/P)
DTR	CPU: P5/9	Data terminal ready on spare serial port (O/P)
EPROM	CPU: P1/30	(For internal use only)
[$\overline{FPRESET}$]	[CPU: P1/14]	Front panel reset switch
$\overline{INSP}$	CPU: P2/53	Instrumentation page strobe
[$\overline{INTRPT3}$]	[CPU: P1/40]	Interrupt line from FDC.
[$\overline{INTRPT4}$]	[CPU: P1/37]	User accessible interrupt line.
[$\overline{INTRPT5}$]	[CCL: P1/42]	BIPOLAR clock stop interrupt line
[$\overline{INTRPT6}$]	[CPU: P1/35]	Reserved for CP/M
[$\overline{INTRPT7}$]	[CPU: P1/36]	Front panel reset (Jmp 0000) highest priority
$\overline{IOR}$	CPU: P1/21	Read strobe to an I/O device
$\overline{IOW}$	CPU: P1/22	Write strobe to an I/O device
$\overline{M0-63}$	WCS: P3, P4	Microprogram data bits
$\overline{MAPP}$	MPS: P2/52	CCU map RAM page strobe
$\overline{MD0-7}$	CPU: P1/67-74	System data bus
$\overline{MA0-15}$	CPU: P1/43-58	System address bus
[MONB0-19]	[CCL: P4/]	User defined monitor bits
[$\overline{MONSEL}$]	[CPU: P1/32]	For internal use only
$\overline{MEMR}$	CPU: P1/19	System memory read strobe
$\overline{MEMW}$	CPU: P1/20	System memory write strobe
MPA0-11	CCL: PROTOTYPE	Microprogram address bus
[MSR/$\overline{MSR}$]	[CCL: P2/48, 49]	Single micro step switch
[OA0-4]	[MPS: P3/24, 21, 22, 19, 20]	Vector input on CCU
[OSCIN]	[CCL: P1/7]	Oscillator input
OSCOUT	CCL: P1/9	On-board oscillator O/P
O2TTL	CPU: P1/31	System clock (2-MHz)
PA0-7	CPU: P4	8-Bit Parallel I/O Port (Am9555, Port A)
PB0-7	CPU: P4	8-Bit Parallel I/O Port (Am9555, Port B)
PC0-7	CPU: P4	8-Bit Parallel I/O Port (Am9555, Port C)
[PRCTS]	[CPU: P5/2]	Printer clear to RCVD data
[PRDSR]	[CPU: P/18]	Printer ready to RCVD or XMIT data
PRRTS	CPU: P5/20	System request to SND to printer
PRTXD	CPU: P5/4	System XMIT data port to printer
[PTDSR]	[CPU: P5/19]	Paper tape reader/punch or PROM Programmer ready to XMIT or RCVD data
PTDTR	CPU: P5/17	Reader/punch terminal ready?
[PTRCVD]	[CPU: P5/8]	RCVD data port from reader/punch or PROM Programmer
PTRTS	CPU: P5/21	System request to send to reader/punch or PROM Programmer
PTTXD	CPU: P5/10	System XMIT data port to reader/punch or PROM Programmer
[RCVD]	[CPU: P5/22]	RCVD data port on spare serial I/O
$\overline{RD}$	CPU: P2/33	System read strobe to BIPOLAR Interface
$\overline{RESET}$	CPU: P1/27	Buffered system reset line (active LO)
[RR/$\overline{RR}$]	[CCL: P2/45,44]	Front panel BIPOLAR run enable function
RTS	CPU: P5/15	CPU request to send to spare serial I/O
$\overline{RW}$	CPU: P2/37	True whenever there is a read or write through the system interface to the BIPOLAR side
SA0-15	CPU: P2/3-18	System interface address lines to the BIPOLAR side
SD0-7	CPU: P2/21-28	System interface data lines to the BIPOLAR side
SGLSTP/SGLSTP	[CPU: P1/29,28]	Front panel BIPOLAR single cycle function
SP0-7	CPU: P1/59-66	Page register bits
$\overline{SQE}$	CCL: P1/80, P5/25	Program address source select. H1; address registers on instrumentation, LO: OUTPUT of sequnecers on CCU
[SR/$\overline{SR}$]	[CCL: P2/46,47]	Front panel, clock stop function
[SSR/$\overline{SSR}$]	[CCL: P2/43,51]	Front panel, single cycle function
[STOP/$\overline{RUN}$]	[CPU: P1/26]	System CPU STOP/$\overline{RUN}$ function
STPACK	CCL: P2/50	H1 indicates the BIPOLAR system clock is Stopped
$\overline{STPREQ}$	CPU: P1/24, P2/58	General purpose signal indicating a read or write to any one of the 128-32K pages
SYNC	CCL: P1/13	Active LO SYNC. when instrumentation address register compares with the Program address register
SYSCLK	INS: P1/79	BIPOLAR system clock
SYSCLKCTL	INS: P2/57, P5/26	LO on this line puts the BIPOLAR system clock output on the instrumentation into H1-impedance. This allows the user to bring onto the instrumentation his own clock.
[TEST 1-14]	[MPS: P1/59, 72]	Conditional test inputs on CCU
[$\overline{TB0-3}$]	[CCL: P3/3. 5, 7, 9]	Trap bits on instrumentation
TXD	CPU: P5/16	Transmit data port on spare serial I/O.
$\overline{WCSP}$	CPU: P2/51	$\overline{WCS}$ page strobe
$\overline{WT}$	CPU: P2/35	System write strobe on the interface to the BIPOLAR pages
$\overline{XACK}$	CPU: P1/23	Transfer acknowledge

Table 6-8. Connector Chart for Connector P1

FUNCTION PIN#	CPU	RAM	RAM	FDC	CCL	MPS	WCS
1	GND	GND	GND	GND	GND	GND	GND
2	GND	GND	GND	GND	GND	GND	GND
3	+5V	+5V	+5V	+5V	+5V	+5V	+5V
4	+5V	+5V	+5V	+5V	+5V	+5V	+5V
5	+5V	+5V	-5V	+5V	+5V	+5V	+5V
6	+5V	+5V	+5V	+5V	+5V	+5V	+5V
7	+12V	+12V	+12V	+12V	OSCIN		
8	+12V	+12V	+12V	+12V	OSCIN RT		
9					OSCOUT		
10					OSCOUT RT		
11	GND	GND	GND	GND	GND	GND	GND
12	GND	GND	GND	GND	GND	GND	GND
13	BCLOCK			BCLOCK	SYNC		
14	FRPRESET			FRPRESET	GND		
15	BPRN			BPRN			
16	-						
17	BUSY			BUSY			
18	BREQ			BREQ			
19	MEMR	MEMR	MEMR	MEMR			
20	MEMW	MEMW	MEMW	MEMW			
21	IOR			IOR			
22	IOW			IOW			
23	XACK	XACK	XACK	XACK			
24	STPREQ	STPREQ	STPREQ				
25	-						
26	STOP/RUN						
27	RESET				RESET		
28	SGLSTP						
29	SGLSTP						
30	EPROM						
31	02(TTL)						
32	MONSEL						
33	-						
34	CP/M						
35	INTRPT6						
36	INTRPT7						
37	INTRPT4						
38	-						
39	-						
40	INTRPT3			INTRPT3			
41	-						
42	INTRPT5				INTRPT5		
43	MA14	MA14	MA14	MA14			
44	MA15	MA15	MA15	MA15			
45	MA12	MA12	MA12	MA12			
46	MA13	MA13	MA13	MA13			
47	MA10	MA10	MA10	MA10			
48	MA11	MA11	MA11	MA11			
49	MA8	MA8	MA8	MA8			
50	MA9	MA9	MA9	MA9			
51	MA6	MA6	MA6	MA6			
52	MA7	MA7	MA7	MA7			
53	MA4	MA4	MA4	MA4			
54	MA5	MA5	MA5	MA5			
55	MA2	MA2	MA2	MA2			
56	MA3	MA3	MA3	MA3			
57	MA0	MA0	MA0	MA0			
58	MA1	MA1	MA1	MA1			
59	SP0						
60	SP1						
61	SP2						
62	SP3						
63	SP4						
64	SP5						
65	SP6				SP6		
66	SP7				SP7		
67	MD6	MD6	MD6	MD6			
68	MD7	MD7	MD7	MD7			
69	MD4	MD4	MD4	MD4			
70	MD5	MD5	MD5	MD5			
71	MD2	MD2	MD2	MD2			
72	MD3	MD3	MD3	MD3			
73	MD0	MD0	MD0	MD0			
74	MD1	MD1	MD1	MD1			
75	GND	GND	GND	GND	GND	GND	GND
76	GND	GND	GND	GND	GND	GND	GND
77					XACK	XACK	XACK
78							
79	-12V	-12V	-12V	-12V	SYSCLK	SYSCLK	SYSCLK
80	-12V	-12V	-12V	-12V	SQE		SP0
81	+5V	+5V	+5V	+5V	+5V	+5V	+5V
82	+5V	+5V	+5V	+5V	+5V	+5V	+5V
83	+5V	+5V	+5V	+5V	+5V	+5V	+5V
84	+5V	+5V	+5V	+5V	+5V	+5V	+5V
85	GND	GND	GND	GND	GND	GND	GND
86	GND	GND	GND	GND	GND	GND	GND

Table 6-9. Connector Chart for Connector P2

FUNCTION PIN#	RAM	CPU	CCL	MPS	WCS
1		GND	GND	GND	
2		GND	GND	GND	
3		SA0	SA0	SA0	SA0
4		SA1	SA1	SA1	SA1
5		SA2	SA2	SA2	SA2
6		SA3	SA3	SA3	SA3
7		SA4	SA4	SA4	SA4
8		SA5	SA5	SA5	SA5
9		SA6	SA6	SA6	SA6
10		SA7	SA7	SA7	SA7
11		SA8		SA8	SA8
12		SA9			SA9
13		SA10			SA10
14		SA11			SA11
15		SA12			SA12
16		SA13			SA13
17		SA14			SA14
18		SA15			
19					
20		GND			
21		SD0	SD0	SD0	SD0
22		SD1	SD1	SD1	SD1
23		SD2	SD2	SD2	SD2
24		SD3	SD3	SD3	SD3
25		SD4	SD4	SD4	SD4
26		SD5	SD5	SD5	SD5
27		SD6	SD6	SD6	SD6
28		SD7	SD7	SD7	SD7
29		GND			
30		GND			
31					
32		GND			
33		$\overline{RD}$	$\overline{RD}$	$\overline{RD}$	$\overline{RD}$
34		GND			
35		$\overline{WT}$	$\overline{WT}$	$\overline{WT}$	$\overline{WT}$
36		GND			
37		$\overline{RW}$	$\overline{RW}$	$\overline{RW}$	
38		GND			
39					
40					
41		GND			
42		GND			
43		$\overline{PGDEC7}$	SSR	BD0	
44			RR	BD1	
45		$\overline{PGDEC6}$	$\overline{RR}$	BD2	
46			SR	BD3	
47		$\overline{PDGEC5}$	$\overline{SR}$	BD4	
48		$\overline{PGDEC4}$	MSR	BD5	
49		$\overline{PGDEC3}$	$\overline{MSR}$	BD6	
50		STPACK	STPACK	BD7	STPACK
51		$\overline{WCSP}$	$\overline{SSR}$		$\overline{WCSP}$
52		$\overline{MAPP}$		$\overline{MAPP}$	
53		$\overline{INSP}$	$\overline{INSP}$		
54			GND	GND	
55			COSC	COSC	
56				GND	
57			SYSCLKCTL	SYSCLKCTL	
58		$\overline{STPREQ}$			
59		GND	GND	GND	
60		GND	GND		

Table 6-10. Connector Chart for Connector P3

FUNCTION PIN#	RAM	CPU	CCL	MPS	WCS
1		(Not	GND	BRTI3	GND
2		Used)	GND	BRTI0	GND
3			TB3	BRTI1	GND
4			GND	BRTI2	M32
5			TB2	T3	M33
6			GND	$\overline{\text{IF}}$	M34
7			TB1	T1	M35
8			GND	T2	M36
9			TB0	T5	M37
10			GND	T4	M38
11				T7	M39
12				T6	GND
13				T9	GND
14				T10	GND
15				T12	GND
16				T11	M40
17				T14	M41
18				T13	M42
19				0A3	M43
20				0A4	M44
21				0A1	M45
22				0A2	M46
23				T8	M47
24				0A0	GND
25					GND
26					GND
27					GND
28					M48
29					M49
30					M50
31					M51
32					M52
33					M53
34					M54
35					M55
36					GND
37					GND
38					GND
39					GND
40					M56
41					M57
42					M58
43					M59
44					M60
45					M61
46					M62
47					M63
48					GND
49					GND
50					GND

Table 6-11. Connector Chart for Connector P4

FUNCTION PIN#	RAM	CPU	CCL	MPS	WCS
1			GND	GND	GND
2		GND	MONB0	GND	GND
3		PB3	GND	GND	GND
4		GND	MONB1	M0	M0
5		PB0	GND	M1	M1
6		GND	MONB2	M2	M2
7		PB1	GND	M3	M3
8		GND	MONB3	M4	M4
9		PB2	GND	M5	M5
10		GND	MONB4	M6	M6
11		PB4	GND	M7	M7
12		GND	MONB5	GND	GND
13		PB5	GND	GND	GND
14		GND	MONB6	GND	GND
15		PB6	GND	GND	GND
16		GND	MONB7	M8	M8
17		PB7	GND	M9	M9
18		GND	MONB8	M10	M10
19		PC3	GND	M11	M11
20		GND	MONB9		M12
21		PC2	GND		M13
22		GND	MONB14		M14
23		PC1	GND		M15
24		GND	MONB11	GND	GND
25		PC0	GND	GND	GND
26		GND	MONB12	GND	GND
27		PC4	GND	GND	GND
28		GND	MONB13	M16	M16
29		PC5	GND	M17	M17
30		GND	MONB14	M18	M18
31		PC6	GND	M19	M19
32		GND	MONB15	M20	M20
33		PC6	GND	M21	M21
34		GND	MONB16	M22	M22
35		PA7	GND	M23	M23
36		GND	MONB17	GND	GND
37		PA6	GND	GND	GND
38		GND	MONB18	GND	GND
39		PA5	GND	GND	GND
40		GND	MONB19	M24	M24
41		PA4	GND	M25	M25
42		GND		M26	M26
43		PA0		M27	M27
44		GND		M28	M28
45		PA1		M29	M29
46		GND			M30
47		PA2			M31
48		GND		GND	GND
49		PA3		GND	GND
50		GND		GND	GND

Table 6-12. Connector Chart for Connector P5

FUNCTION PIN#	RAM	CPU	CCL	MPS	WCS
1		GND	GND	GND	GND
2		PRCTS	MPA0	MPA0	MPA0
3		CNRCVD	GND	GND	GND
4		PRTXD	MPA1	MPA1	MPA1
5		CNTXD	GND	GND	GND
6		CTS	MPA2	MPA2	MPA2
7		DSR	GND	GND	GND
8		PTRCVD	MPA3	MPA3	MPA3
9		DTR	GND	GND	GND
10		PTTXD	MPA4	MPA4	MPA4
11		CNDTR	GND	GND	GND
12		PRDTR	MPA5	MPA5	MPA5
13		GND	GND	GND	GND
14		CNRTS	MPA6	MPA6	MPA6
15		RTS	GND	GND	GND
16		TXD	MPA7	MPA7	MPA7
17		PTDTR	GND	GND	GND
18		PRDSR	MPA8	MPA8	MPA8
19		PTDSR	GND	GND	GND
20		PRRTS	MPA9	MPA9	MPA9
21		PTRTS	GND	GND	GND
22		RCVD	MPA10	MPA10	MPA10
23		GND			MPAX-1
24		GND	MPA11	MPA11	MPA11
25			$\overline{SQE}$	$\overline{SQE}$	
26			SYSCLKCTL	SYSCLKCTL	

APPENDIX A
ASMDEMO PROGRAM

(User input lines are underscored)

```
MICROPROGRAM GENERATION COMMANDS    A>SUBMIT ASMDEMO JUMP

                                    A>DISPL ASMDEMO.SUB
                                    DISPL ASMDEMO.SUB
                                    DISPL $1.DEF
                                    DISPL $1.SRC
                                    AMDASM P1 $1 P2 $1 NL=23
                                    DISPL $1,P2L
                                    AMPROM 0 $1
                                    DISPL $1,P3L
                                    SUBMIT DDTDEMO $1
                                    <RETURN>
TYPE 'E <RETURN> TO EXIT DISPL

THE DEFINITION FILE
                                    A>DISPL JUMP.DEF
DEFINES 1 30-BIT MNEMONIC -'GOTO'
                                    WORD 30
                                    GOTO;  DEF 4H#0, 6X, H#F,16V%:H#
                                    END

                                    *E<RETURN>

THE SOURCE FILE                     A>DISPL JUMP.SRC

SHORT LOOP USING'GOTO'AND LABELS    ORG H#0000
                                    POINT1: GOTO POINT2
                                    POINT2: GOTO POINT3
                                    POINT3: GOTO POINT4
                                    POINT4: GOTO POINT5
                                    POINT5: GOTO POINT6
                                    POINT6: GOTO POINT1
                                    END

                                    E<RETURN>

ASSEMBLE A MICROPROGRAM

LN=23 ADJUSTS FOR CRT SCREEN SIZE   A>AMDASM P1 JUMP P2 JUMP LN23

                                    AMDOS/29AMDASMMICRO ASSEMBLER,V1.4

                                    TOTAL PHASE 1 ERRORS = 0
                                    TOTAL PHASE 2 ERRORS = 0
```

THE SOURCE FILE

```
A>DISPL JUMP.P2L

AMDOS/29 AMDASM MICRO ASSEMBLER,V1.4 PAGE 1

0000 ORG H#0000
0000 POINT1: GOTO POINT2
0001 POINT2: GOTO POINT3
0002 POINT3: GOTO POINT4
0003 POINT4: GOTO POINT5
0004 POINT5: GOTO POINT6
0005 POINT6: GOTO POINT1
     END

*<RETURN>

*<RETURN> Displays The Next Page
```

MICROCODE LISTING

1'S, 0'S, and don't cares

```
AMDOS/29 AMDASM MICRO ASSEMBLER,V1.4 PAGE 2

0000 0000XXXXXX111100 00000000000001
0001 0000XXXXXX111100 00000000000010
0002 0000XXXXXX111100 00000000000011
0003 0000XXXXXX111100 00000000000100
0004 0000XXXXXX111100 00000000000101
0005 0000XXXXXX111100 00000000000000

 <RETURN>
```

ENTRY POINTS

```
AMDOS/29 AMDASM MICRO ASSEMBLER,V1.4 PAGE 3

ENTRY POINTS

<RETURN>
```

SYMBOL TABLE

```
AMDOS/29 AMDASM MICRO ASSEMBLER,V1.4 PAGE 4

SYMBOLS

POINT1  0000
POINT2  0001
POINT3  0002
POINT4  0003
POINT5  0004
POINT6  0005

TOTAL PHASE ERRORS = 0

E<RETURN>
```

FRAGMENT CODE FOR PROMS

```
A>AMPROM O JUMP

DON'T CARES?  O
ENTER PROM WIDTHS: 4
ENTER PROM DEPTHS: 6
WHICH PROMS DO YOU WISH TO PRINT? A

PROM 1 STARTING PC = 0000 6 WORDS
PROM 2 STARTING PC = 0000 6 WORDS
PROM 3 STARTING PC = 0000 6 WORDS
PROM 4 STARTING PC = 0000 6 WORDS
PROM 5 STARTING PC = 0000 6 WORDS
PROM 6 STARTING PC = 0000 6 WORDS
PROM 7 STARTING PC = 0000 6 WORDS
PROM 8 STARTING PC = 0000 6 WORDS
```

NUMBER PROMS,
LIST CONTENTS

```
A>DISPL JUMP.P3L

AMD AMPROM UTILITY

PROM MAP

  PC     C1   C2   C3   C4   C5   C6   C7   C8

R1 0000   1    2    3    4    5    6    7    8

PROM CONTENTS

PC ADD P1 P2 P3 P4 P5 P6 P7 P8

0000 000 0000 0000 0011 1100 0000 0000 0000 0100
0001  001 0000 0000 0011 1100 0000 0000 0000 1000
0002  002 0000 0000 0011 1100 0000 0000 0000 1100
0003  003 0000 0000 0011 1100 0000 0000 0001 0000
0004  004 0000 0000 0011 1100 0000 0000 0001 0100
0005  005 0000 0000 0011 1100 0000 0000 0000 0000
*
<RETURN>
```

EMULATOR COMMAND
SOFTWARE

```
A>SUMBIT DDTDEMO JUMP

A>DISPL DDTDEMO.SUB

DISPL DDTDEMO.SUB
DDT29 H PA WC E
LBPM $1 WC
DDT29 CTL 3F IR O JAM RUN Z SP E
DDT29 H IR O JAM M DTR MX Z SP ME E
DDT29 S 5 OF 03/ D 5.1 IR O JAM DTR Z SP E
DDT29 M DTR MS Z 2000 ME E
DDT29 IR O J DTR CTL 2F IR 4 Z SP R DTR CTL 3F Z SP 3
<RETURN>
```

```
RESET HARDWARE, LOAD
CODE,VERIFY LOAD      A>DDT29 H PA WC E

                      DDT29 VERSION 2.1

                      A>LBPM JUMP WC

                      LOAD BIPOLAR MEMORY, VER 2.1, AMDOS 2.0
                      LOADING:  JUMP.OBJ
                      TITLE:
                      LOAD COMPLETE

                      VERIFY BIPOLAR MEMORY, VER 2.1,AMDOS 2.0
                      VERIFYING:  JUMP.OBJ
                      TITLE:
                      VERIFY COMPLETE

CLEAR INTERRUPTS,
RUN                   A>DDT29 CTL 3F IR 0 JAM RUN Z SP E

                      DDT29 VERSION 2.1

                      SPACE>

Step Thru Program
With Space Bar        A>DDT29 IR 0 JAM M DTR MS Z SP ME E

                      DDT29 VERSION 2.1

INSTRUCTION 0 GOTO 1       MICROWORD AT 0000 - 0001 0203 000F 0001
                           MONITOR BITS      -    (FFFFF)
                           SPACE>
                           MICROWORD AT 0001 - 0001 0203 000F 0002
                           MONITOR BITS      -    (FFFFF)
                           SPACE>
                           MICROWORD AT 0002 - 0001 0203 000F 0003
                           MONITOR BITS      -    (FFFFF)
                           SPACE>
                           MICROWORD AT 0003 - 0001 0203 000F 0004
                           MONITOR BITS      -    (FFFFF)
                           SPACE>
                           MICROWORD AT 0004 - 0001 0203 000F 0005
                           MONITOR BITS      -    (FFFFF)
                           <PACE>
INSTRUCTION 5 GOTO 0       MICROWORD AT 0005 - 0001 0203 000F 0000
                           MONITOR BITS      -    (FFFFF)
                           SPACE>
                           MICROWORD AT 0000 - 0001 0203 000F 0000
                           MONITOR BITS      -    (FFFFF)
                           SPACE>
                           MICROWORD AT 0001 - 0001 0203 000F 0002
                           MONITOR BITS      -    (FFFFF)
                           SPACE>
```

```
MODIFY THE CODE          A>DDT29 S 5 OF 03/ D 5.1 IR 0 JAM DTR Z SP E
                         DDT29 VERSION 2.1

Instruction 5 goto 3     0005 - 0001 0203 000F 0003
                         MICROWORD AT 0000 - 0001 0203 000F 0001
                         MONITOR BITS      -    (FFFFF)
                         SPACE>

AUTOMATIC STEP           A>DDT29 M DTR MS Z 2000 ME E

                         DDT29 VERSION 2.1

                         MICROWORD AT 0000 - 0001 0203 000F 0001
                         MONITOR BITS           (FFFFF)

                         MICROWORD AT 0001 - 0001 0203 000F 0002
                         MONITOR BITS      -    (FFFFF)

                         MICROWORD AT 0002 - 0001 0203 000F 0003
                         MONITOR BITS      -    (FFFFF)

                         MICROWORD AT 0003 - 0001 0203 000F 0004
                         MONITOR BITS      -    (FFFFF)

                         MICROWORD AT 0004 - 0001 0203 000F 0005
                         MONITOR BITS      -    (FFFFF)

                         MICROWORD AT 0005 - 0001 0203 000F 0003
                         MONITOR BITS      -    (FFFFF)

                         MICROWORD AT 0003 - 0001 0203 000F 0003
                         MONITOR BITS      -    (FFFFF)

                         DELETE>

SET BREAKPOINT     A>DDT29 IR 0 J DTR CTL 2F IR 4 Z SP R DTR CTL 3F Z SP E

                   DDT29 VERSION 2.1

                   MICROWORD AT 0003 - 0001 0203 000F 0001
                   MONITOR BITS      -    (FFFFF)
                   SPACE>

RUN FROM 0
BREAK AT 4         MICROWORD AT 0003 - 0001 0203 000F 0005
                   MONITOR BITS      -    (FFFFF)
                   SPACE>
```

COMMENT SHEET

Address comments to:

Advanced Micro Devices, Inc.
Microcomputer Systems
Publications Department
P.O. Box 453
Sunnyvale, CA 94086

TITLE: AmSYS29/10A Microprogram Development System
Publication Number: 059920027-002 Revision A

COMMENTS: (Describe errors, suggested additions or deletions, and include page numbers, etc.)

From: Name: ______________________ Position: ______________________

Company: ____

Address: ____

ADVANCED
MICRO
DEVICES, INC.
901 Thompson Place
Sunnyvale
California 94086
(408) 732-2400
TWX: 910-339-9280
TELEX: 34-6306
TOLL FREE
(800) 538-8450