OBJECTIVES: **11401/402 CLASS**

By the end of this class you will be able to do:

INSTALLATION:	ON CUSTOMER SITE.
CALIBRATION:	OF THE 11401/02 USING A PC TO RESTORE THE CAL CONSTANTS AN CALIBRATE THE INSTRUMENT ACCORDING TO PROCEDURES.
OPERATION:	WELL ENOUGH TO GIVE A SMALL INTRODUCTION TO THE CUSTOMER.
TROUBLESHOOTING:	THE INSTRUMENT TO BOARDLEVEL USING THE BUILTIN DIAGNOSTICS IN CONJUNCTION WITH A PC AND THE NECESSARY SOFTWARE (PROCOM).

PREREQUISITES: The student will be required to complete the pre study package for the 11401/402 training.

4 NEW PROBES

P6134	10	x	10MΩ	passive
P6135	10	x	matched pair of 1MΩ	
P6231	10	x	450Ω	active
P6204	10	x	10MΩ	active

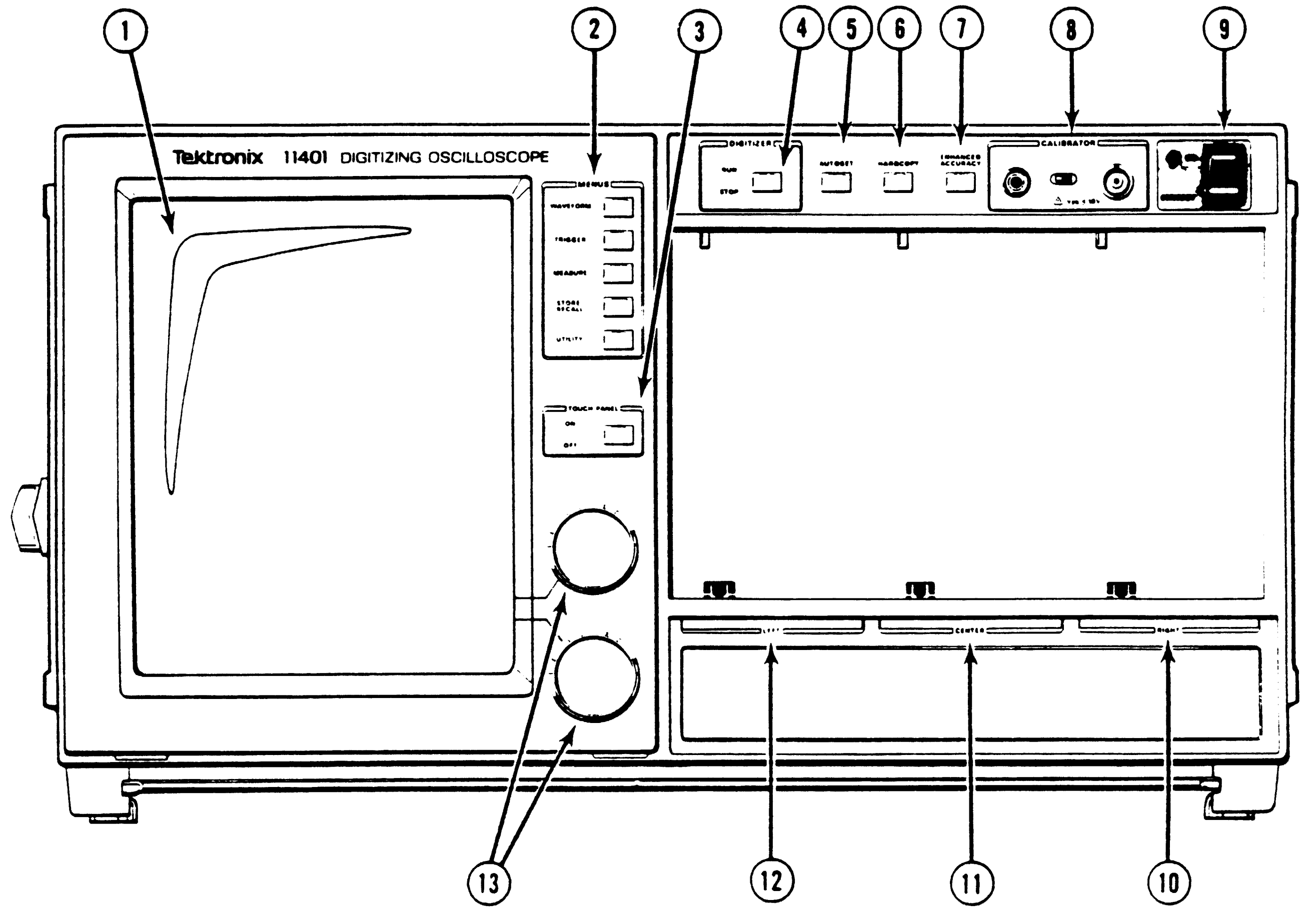
OVERVIEW

DIGITIZING OSCILLOSCOPES:

2 MAINFRAMES 11401 and 11402

5 PLUGINS

11A32 dual trace	400 MHz 50 Ω / 1 MΩ
11A33 differential	150 MHz 50 Ω / 1 MΩ/ 1 GΩ
11A34 four trace	250 MHz 50 Ω / 1 MΩ
11A52 dual trace	600 MHz 50 Ω
11A71 single trace	1000 MHz 50 Ω



5791-111A

Figure 2-1. Front-panel controls, connectors, and indicators.

Front-Panel Controls, Connectors and Indicators

(1) CRT screen touch-panel display. Refer to Figure 2-3 for detailed description.

(2) MENUS buttons

- WAVEFORM–Provides access to selectable characteristics of displayed waveforms from menus. Displays the Waveform major menu in the menu/status area. Refer to "Waveform Menu Function Selections" under "Waveform Control."
- TRIGGER–Selectable triggering characteristics are available on the screen. Displays the Trigger major menu in the menu/status area. Refer to "Triggering" in this section.
- MEASURE–Provides a list of measurements that can be performed. Displays the Measure major menu in the menu/status area. Refer to "Measuring Waveforms" in this section.
- STORE/RECALL–Waveforms and settings can be stored, recalled, deleted, or cleared. Displays the Store/Recall major menu in the menu/status area. Refer to "Storing and Recalling Waveforms" in this section.
- UTILITY–Functions that are not used to acquire, display, or measure waveforms are available through this menu. Displays the Utility major menu in the menu/status area. Refer to "Utilities" in this section.

(3) TOUCH PANEL ON/OFF–Activates or deactivates the touch panel. Refer to "Touch Panel Operation" in this section.

(4) DIGITIZER RUN/STOP–Starts or stops the process of acquiring waveforms from the plug-in units. Refer to "Waveform Control" in this section.

(5) AUTOSET–Sizes and positions the input signal for a stable display on the screen. Refer to "Displaying a Waveform Using Autoset" in this section.

(6) HARDCOPY–Sends data to a hardcopy unit (through the rear-panel PRINTER connector) to produce a permanent record of the display.

(7) ENHANCED ACCURACY–Increases the measurement accuracy. Refer to "Waveform Display Area" in this section for specific information.

(8) CALIBRATOR–Provides an accurate square wave for use as a reference for voltage calibration, frequency compensation, and time deskewing of oscilloscope probes. The Calibrator signal is active when the Probes pop-up menu is used to calibrate probes.

(9) ON/STANDBY–Sets instrument from a ready-to-operate state to an operational state.

(10) RIGHT
(11) CENTER } Plug-in compartments for amplifier units and special purpose units.
(12) LEFT

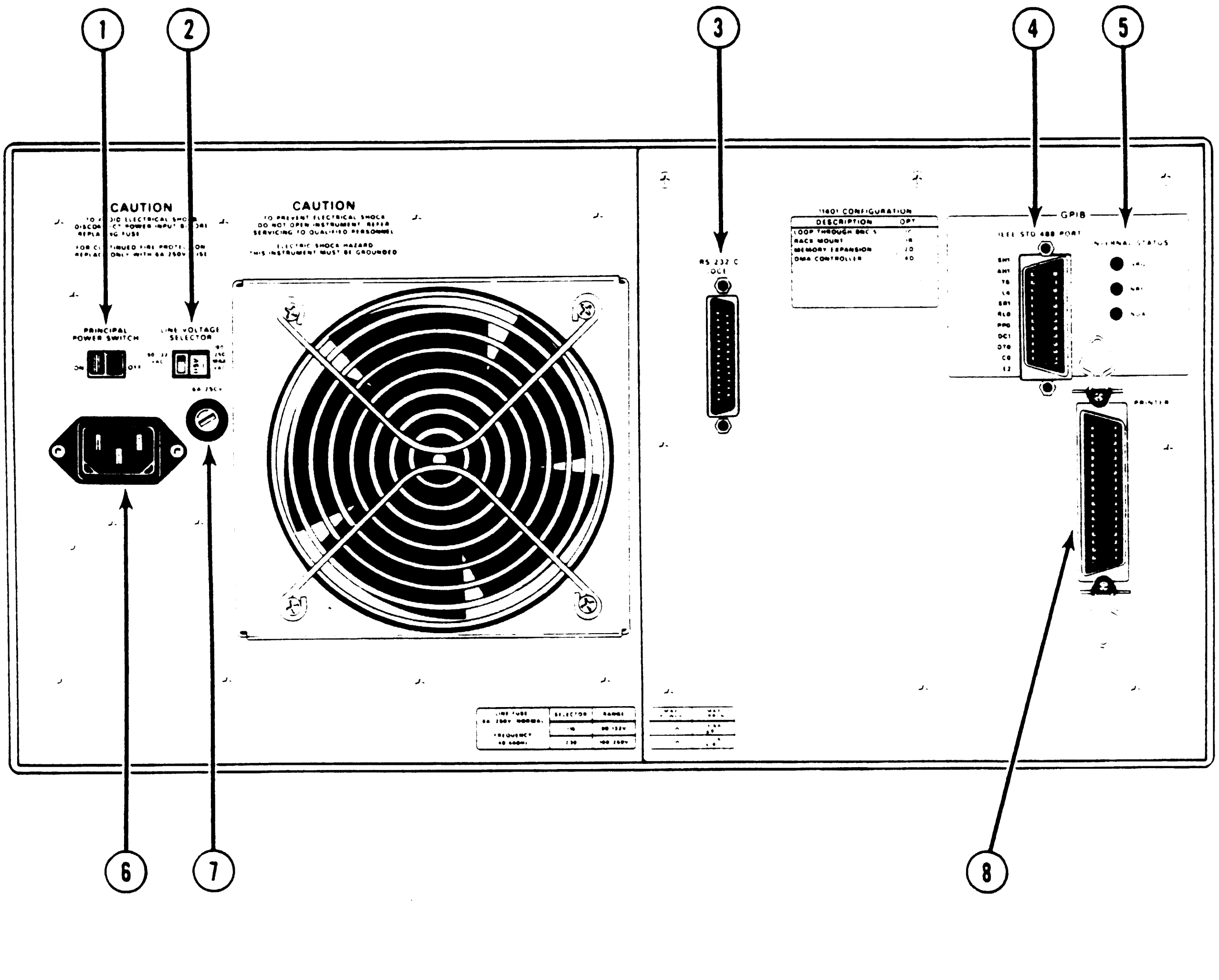
(13) Control Knobs–Functions are assigned to Control knobs by selecting an icon or by selections made from various menus. Refer to "Control Knobs and Numeric Keypad Functions" for detailed information.

5791-1118

Figure 2-1 (cont). Front-panel controls, connectors, and indicators.

Rear Panel

The rear-panel switches and connectors are briefly discussed here. Also, refer to Figure 2-8, Rear-panel controls, connectors, and indicators.



5791-113A

Figure 2-8. Rear-panel controls, connectors, and indicators.

Rear-Panel Controls, Connectors, Indicators, and Fuse

(1) PRINCIPAL POWER SWITCH–Controls line power to the 11401/11402.

(2) LINE VOLTAGE SELECTOR–Sets the 11401/11402 to accept either 115- or 230-volt nominal line voltage.

(3) RS-232-C Connector–Provides a serial data port to communicate with peripheral devices.

(4) GPIB Connector–Provides a parallel data port to communicate with peripheral devices for instrument control.

(5) INTERNAL STATUS Indicators–LEDs light to indicate present status of GPIB interface.

SRQ–(Service ReQuest)
NRFD–(Not Ready For Data)
NDAC–(Not Data ACcepted)

(6) Power Input Connector–Provides a means to connect the 11401/11402 to a line power source.

(7) Fuse–6 ampere, 250 volt normal blow.

(8) PRINTER Connector–Provides a means to transfer data from the 11401/11402 to a line printer unit for hardcopy.

5791-113B

Figure 2-8 (cont). Rear-panel controls, connectors, and indicators.

Principal Power Switch

The rear-panel PRINCIPAL POWER SWITCH controls ac power to the instrument. For detailed information refer to "Power-Up Information" earlier in this section.

Line Voltage Selector Switch

The LINE VOLTAGE SELECTOR switch allows you to select 115-volt or 230 -volt nominal line-voltage operation. For further information, refer to "Operating Voltage" in Section 1, Installation.

RS-232-C Connector

The RS-232-C connector provides a means of connecting an external serial interface to the 11401 Oscilloscope.

For specifics about controlling the 11401 via an external serial interface, see the following:

- Utilities in this section and;
- Section 3, External Interface (GPIB and RS-232-C), in this manual.

GPIB Connector & Status Indicators

The GPIB Interface connector lets you connect the 11401/11402 into a GPIB system.

For specifics about using the 11401/1402 in GPIB mode, see:

- Utilities in this section and;
- Section 3, GPIB and RS-232-C Interfaces, in this manual.

PRINTER Connector

The rear-panel PRINTER connector provides a means of transmitting hard-copy data to a hard-copy unit.

For specific information about obtaining hard copies of the screen display, see "How to Obtain a Hardcopy of the Screen Display" in the "Waveform Control" portion of this section. Printer connection information is located in Section 1, Installation.

Menu and Status Area

The menu/status area occupies the last eight lines of the screen below the waveform display area. After any MENUS button on the front panel is pushed, or the **Cursor** label is touched, the selectable items will be displayed in the menu/status area. Successive pushes on any MENUS button will alternately remove and restore the major menu display. This provides a less cluttered display, when desired. The system's operating status and any measurement results will also be displayed in the menu/status area. The types of information displayed in the menu/status area are:

- Waveform major menu and status.
- Trigger major menu and status.
- Measure major menu and results.
- Store/Recall major menu and status.
- Utility major menu and status.
- Cursor major menu and status.
- Top and Bottom Control knob labels.
- Remove Waveform.
- Pan/Zoom.
- Channel Select.

Menu/Status Area Display Conventions

The menu/status area follows certain conventions in displaying menu items, status and command functions. The information is displayed with four intensity levels, which are: off, low, medium, and high. Figure 2-7 shows the display conventions for major menus and status, with the Trigger major menu as an example.

The display conventions shown in Figure 2-7 operate as described below:

- **Nonselectable**—Nonselectable menu items are shown with off background and dim-intensity symbols.
- **Selectable**—Selectable menu functions are shown with low-intensity backgrounds and medium-intensity symbols.
- **Selected**—The currently selected menu function is displayed with medium-intensity background and high-intensity symbols.

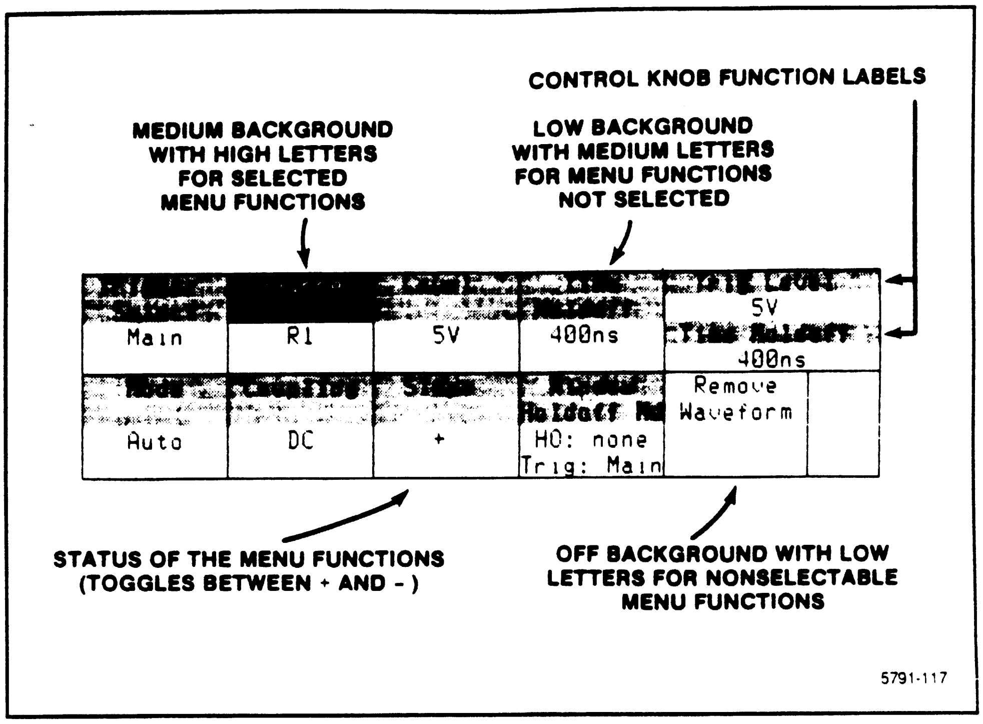
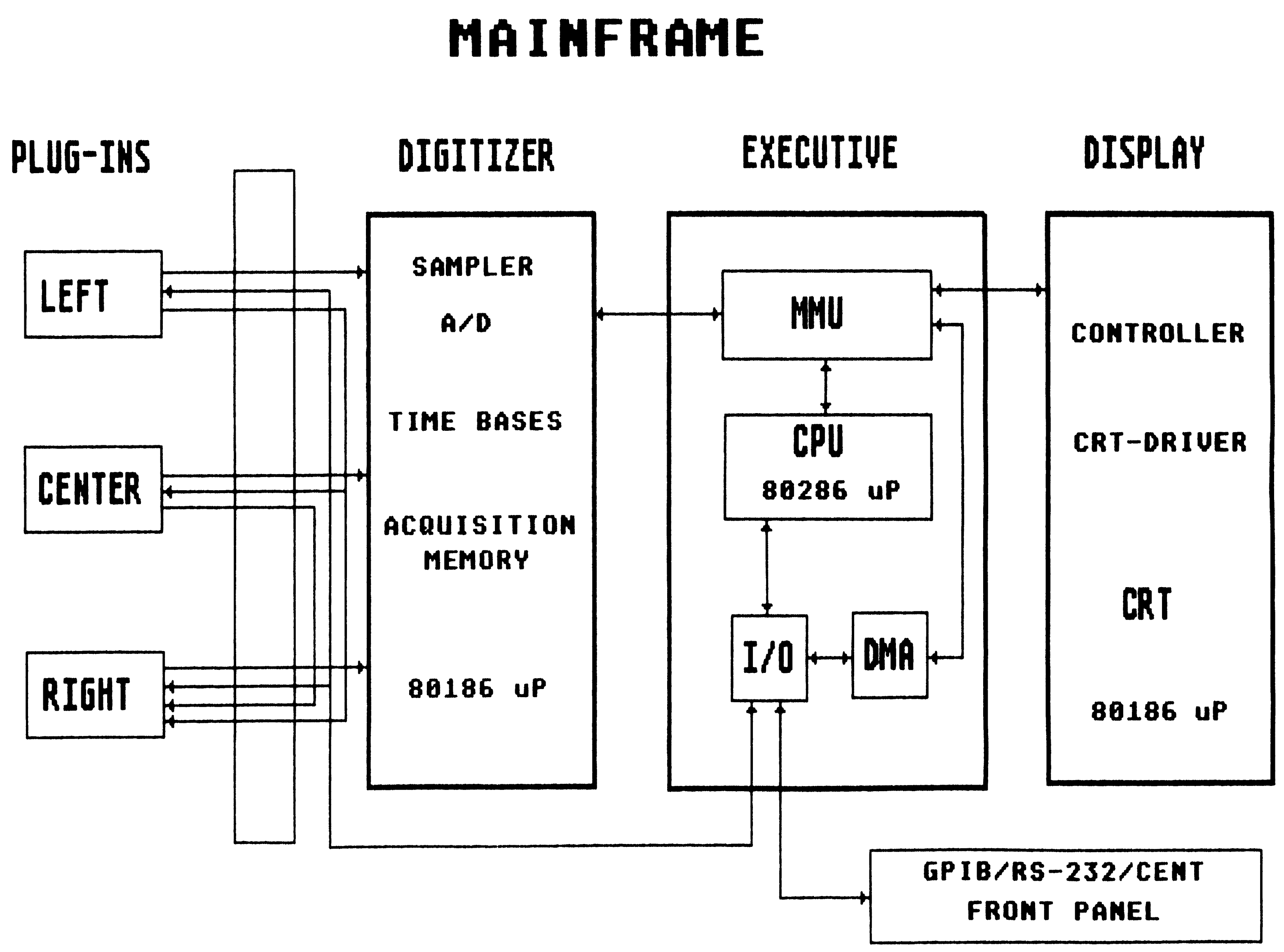

CONTROL KNOB FUNCTION LABELS
MEDIUM BACKGROUND WITH HIGH LETTERS FOR SELECTED MENU FUNCTIONS
LOW BACKGROUND WITH MEDIUM LETTERS FOR MENU FUNCTIONS NOT SELECTED
Main
R1
5V
400ns
5V
400ns
Remove Waveform
Auto
DC
+
HO: none
Trig: Main
STATUS OF THE MENU FUNCTIONS (TOGGLES BETWEEN + AND -)
OFF BACKGROUND WITH LOW LETTERS FOR NONSELECTABLE MENU FUNCTIONS
5791-117

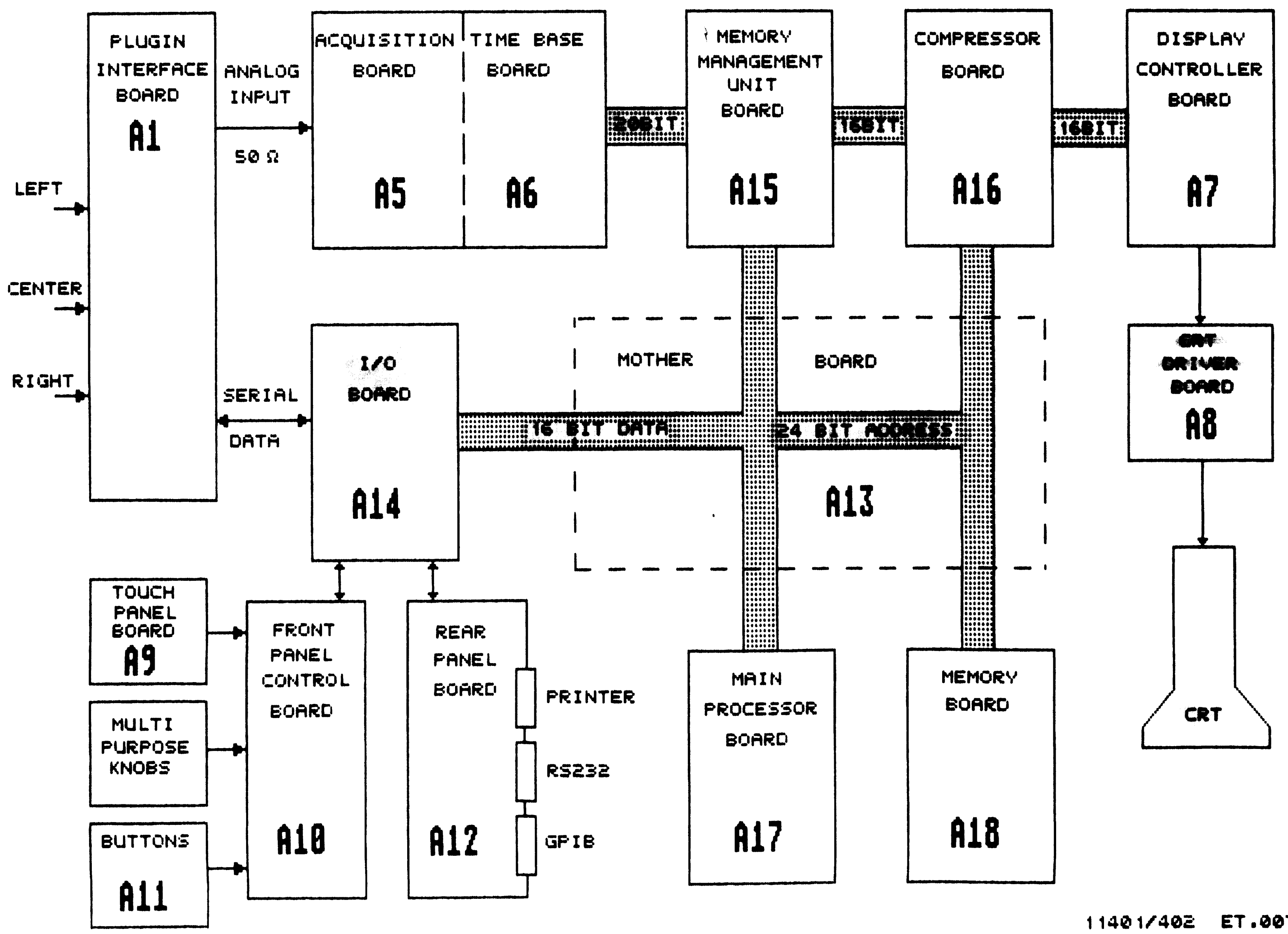
Figure 2-7. Menu/Status area display conventions.

- **Status**—The text immediately below a menu function label shows the present status of that function.
- **Toggle**—Functions that have only two settings change to the other value when the function is selected (e.g., the **Slope** label toggles between + and –).
- **Control Knob Function Labels**—Functions that require entry of numerical values will have the Control knobs assigned to them, so that values can be changed. Their functions are indicated by labels adjacent to the knobs.
- **Paging**—If there is insufficient space to display all the pertinent menu items and their status, the menu and status area must be paged to bring the additional menu items into view. The paging label is not shown in Figure 2-7.
- **Status Overflow**—When the status information for any function has more characters than the allotted space allows, an ellipsis (...) is used to indicate that not all the status information is displayed. The status overflow is not shown in Figure 2-7.



11401/402 ET.001

11401/11402 BLOCK DIAGRAM



11401/402 ET.007

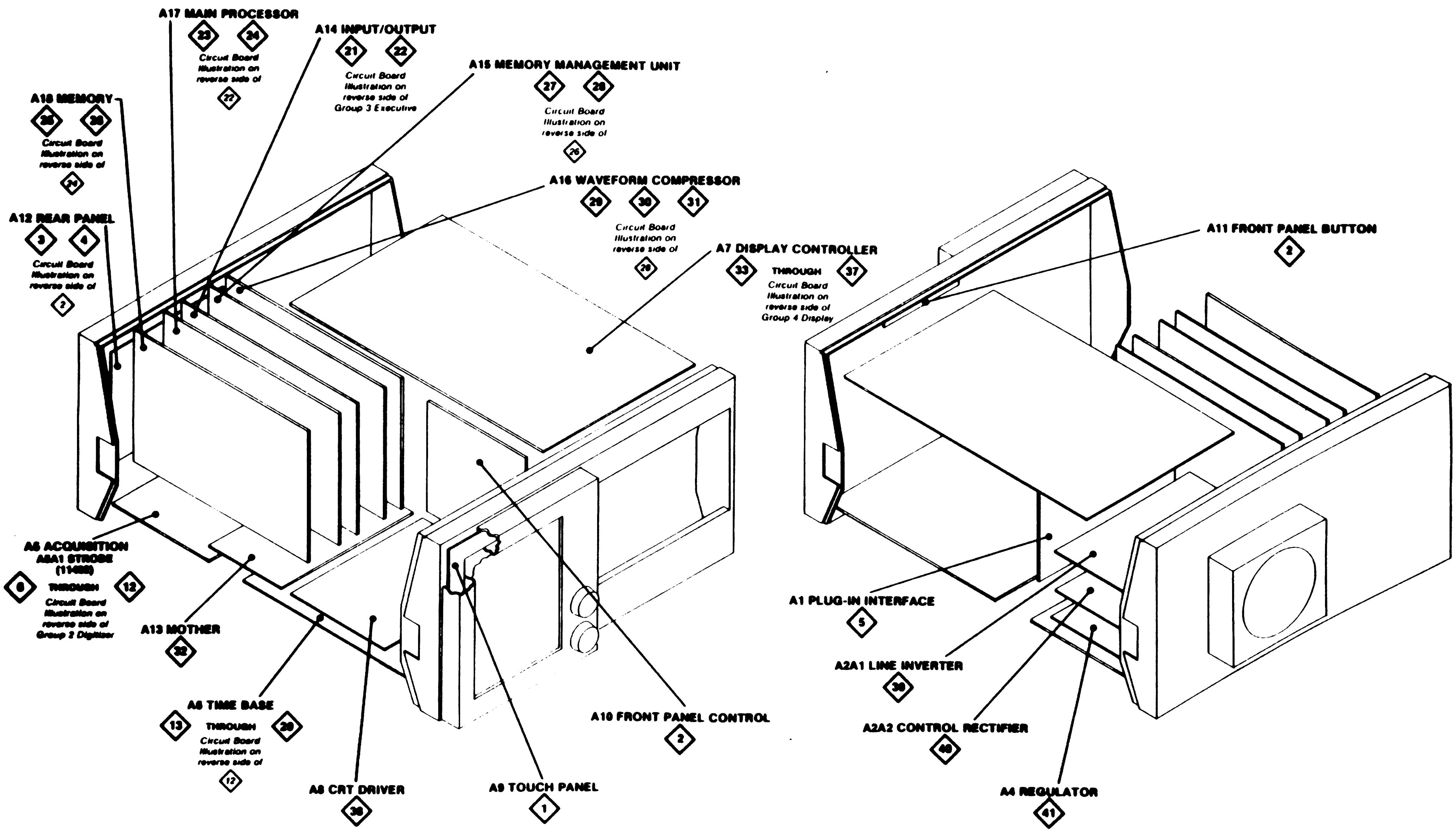
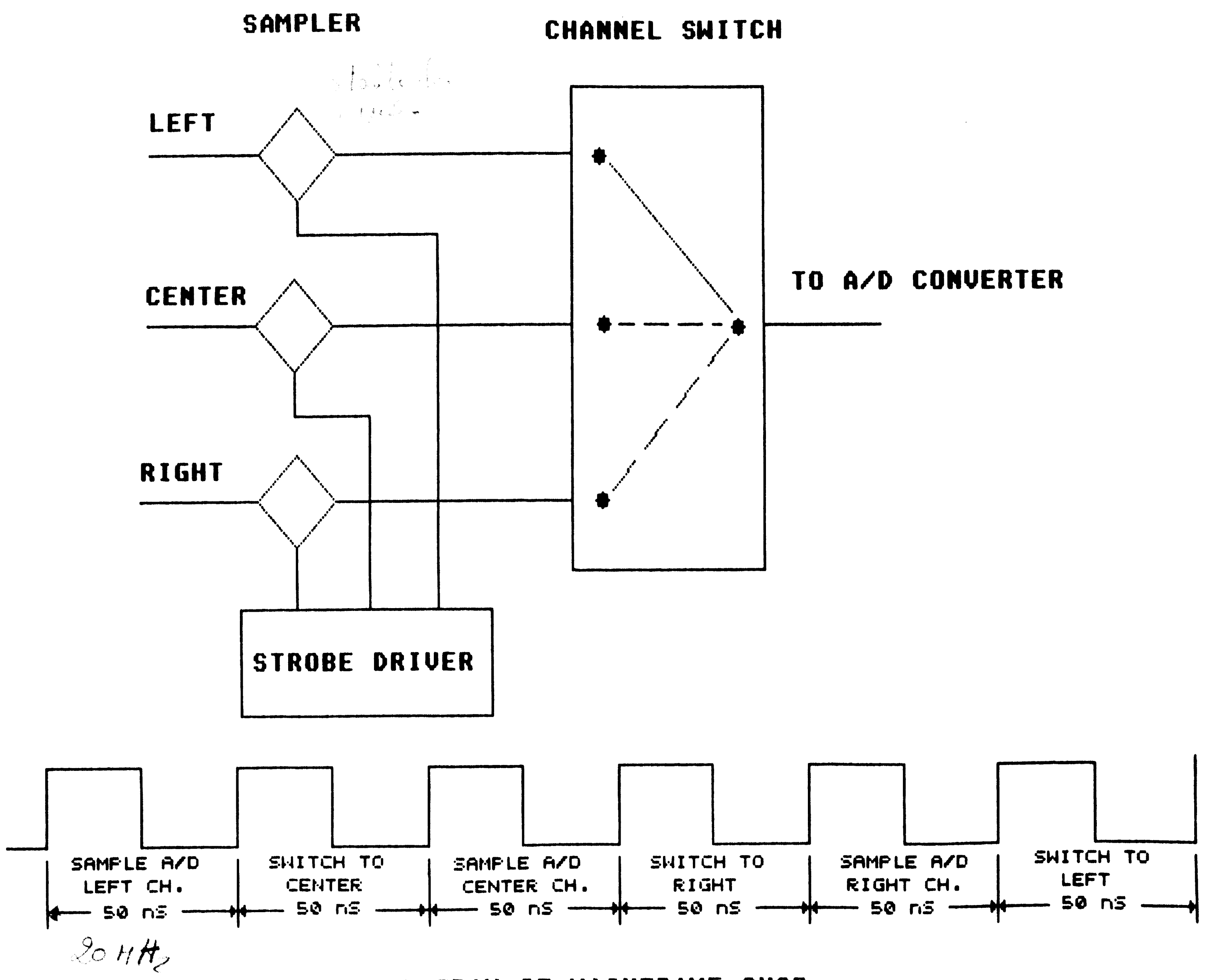
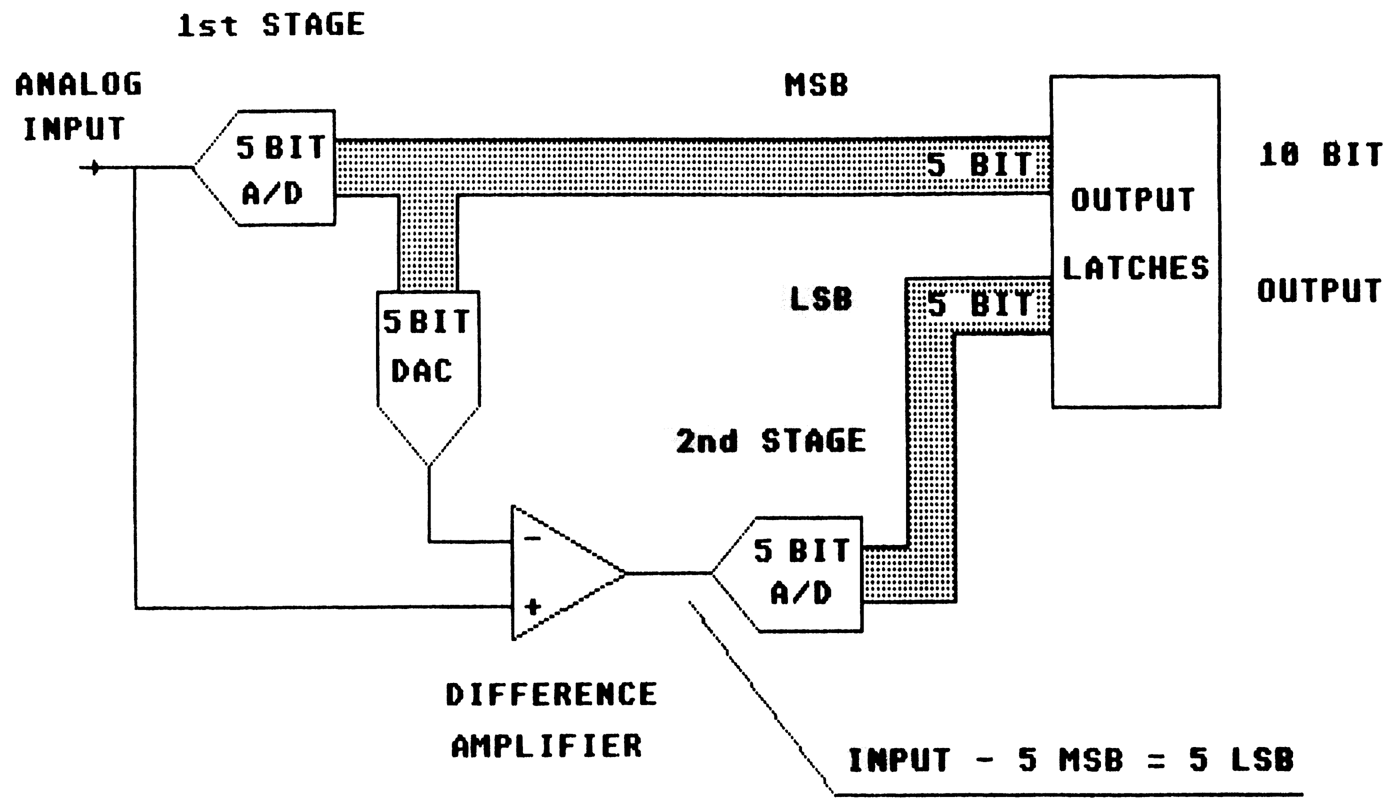

A17 MAIN PROCESSOR
23 24
Circuit Board Illustration on reverse side of 22
A14 INPUT/OUTPUT
21 22
Circuit Board Illustration on reverse side of Group 3 Executive
A15 MEMORY MANAGEMENT UNIT
27 28
Circuit Board Illustration on reverse side of 26
A18 MEMORY
25 26
Circuit Board Illustration on reverse side of 24
A16 WAVEFORM COMPRESSOR
29 30 31
Circuit Board Illustration on reverse side of 28
A12 REAR PANEL
3 4
Circuit Board Illustration on reverse side of 2
A7 DISPLAY CONTROLLER
33 THROUGH 37
Circuit Board Illustration on reverse side of Group 4 Display
A11 FRONT PANEL BUTTON
2
A5 ACQUISITION
A5A1 STROBE
(11402)
6 THROUGH 12
Circuit Board Illustration on reverse side of Group 2 Digitizer
A13 MOTHER
32
A6 TIME BASE
13 THROUGH 20
Circuit Board Illustration on reverse side of 12
A8 CRT DRIVER
38
A9 TOUCH PANEL
1
A10 FRONT PANEL CONTROL
2
A1 PLUG-IN INTERFACE
5
A2A1 LINE INVERTER
39
A2A2 CONTROL RECTIFIER
40
A4 REGULATOR
41

Figure 10-1. 11401/11402 Circuit Board Locator.

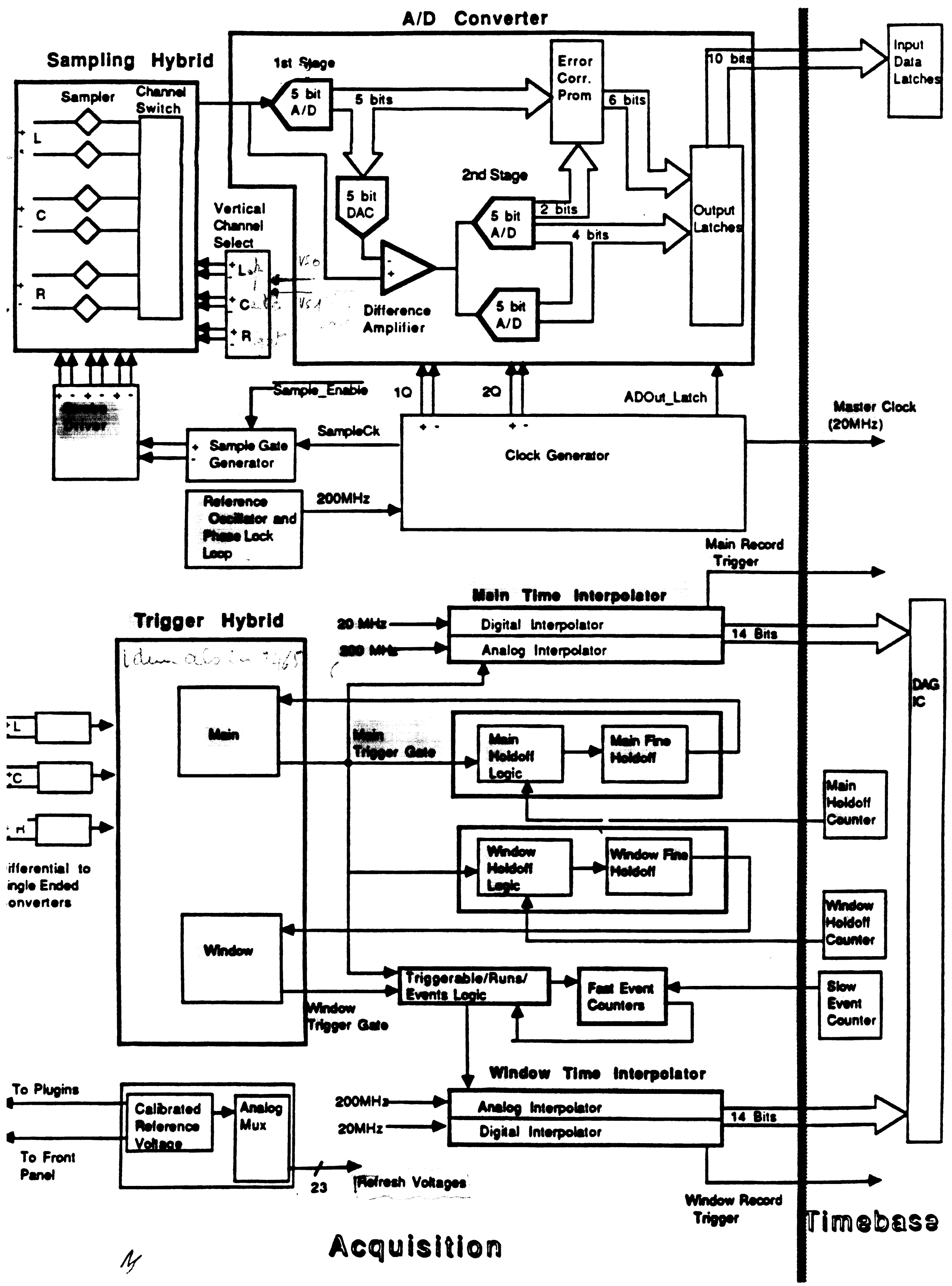
6102-1000



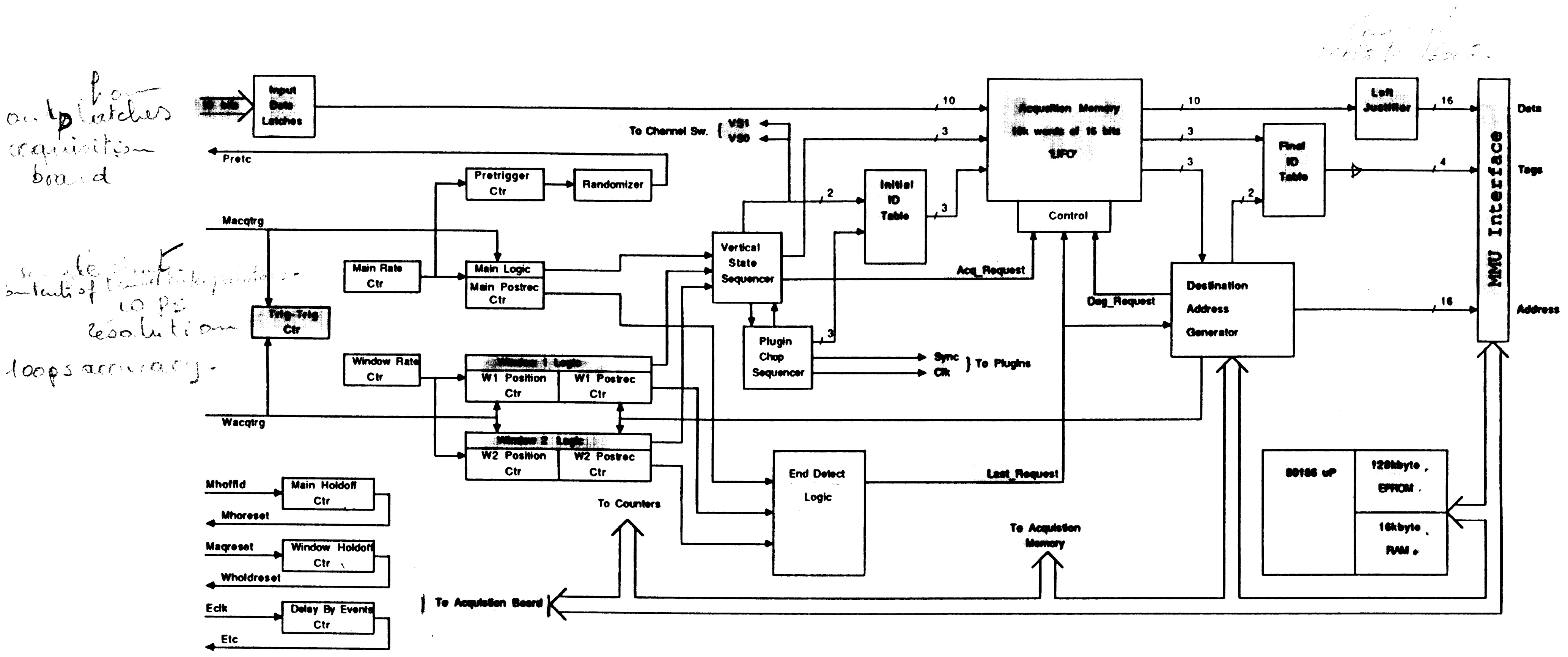
TIMING DIAGRAM OF MAINFRAME CHOP


1st STAGE
ANALOG INPUT
5 BIT A/D
MSB
5 BIT
OUTPUT LATCHES
10 BIT OUTPUT
5 BIT DAC
LSB
5 BIT
2nd STAGE
5 BIT A/D
DIFFERENCE AMPLIFIER
INPUT - 5 MSB = 5 LSB

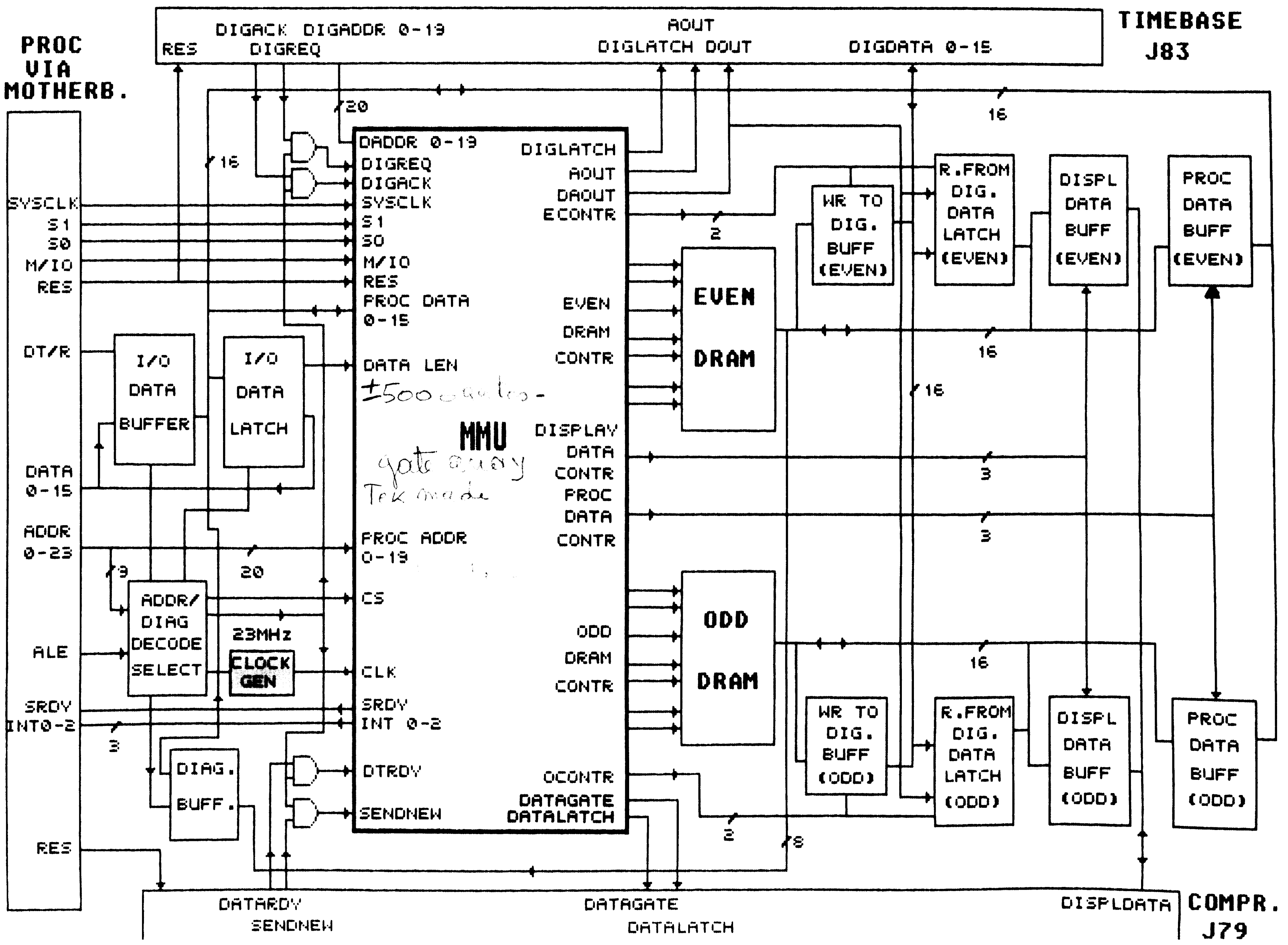
ACQUISITION BRD. A5



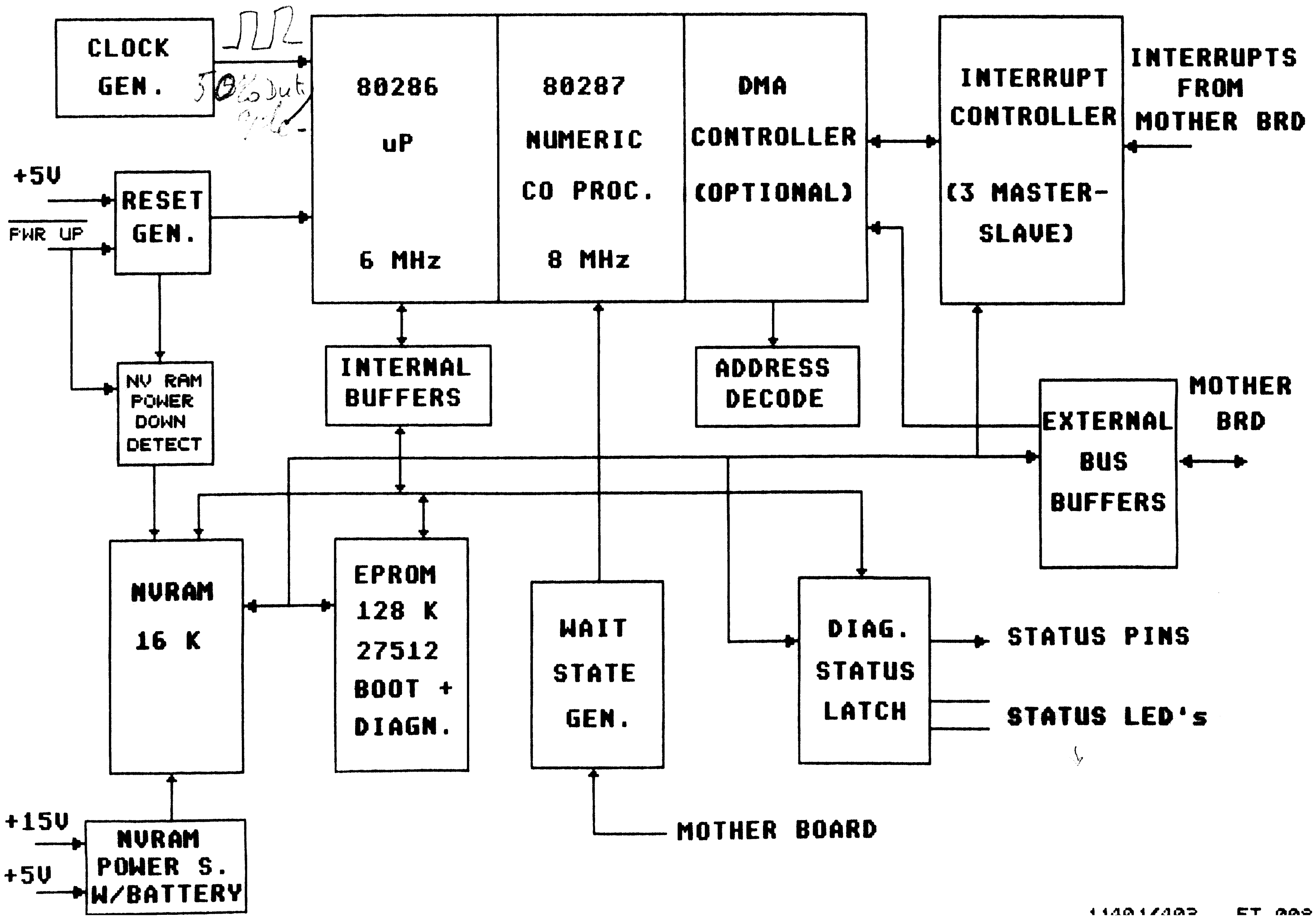
TIME BASE BRD. A6



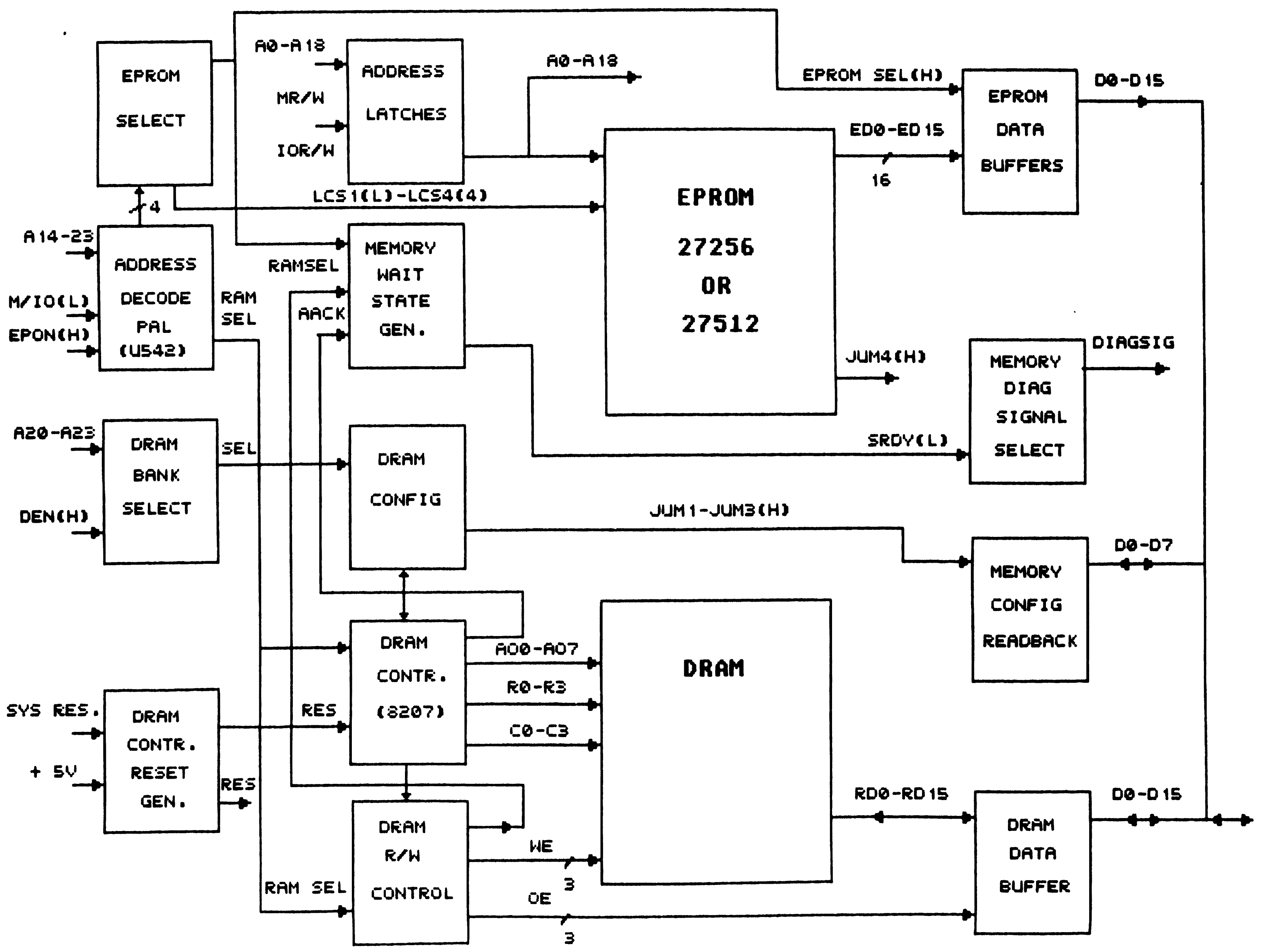
MEMORY MANAGEMENT BRD. A15



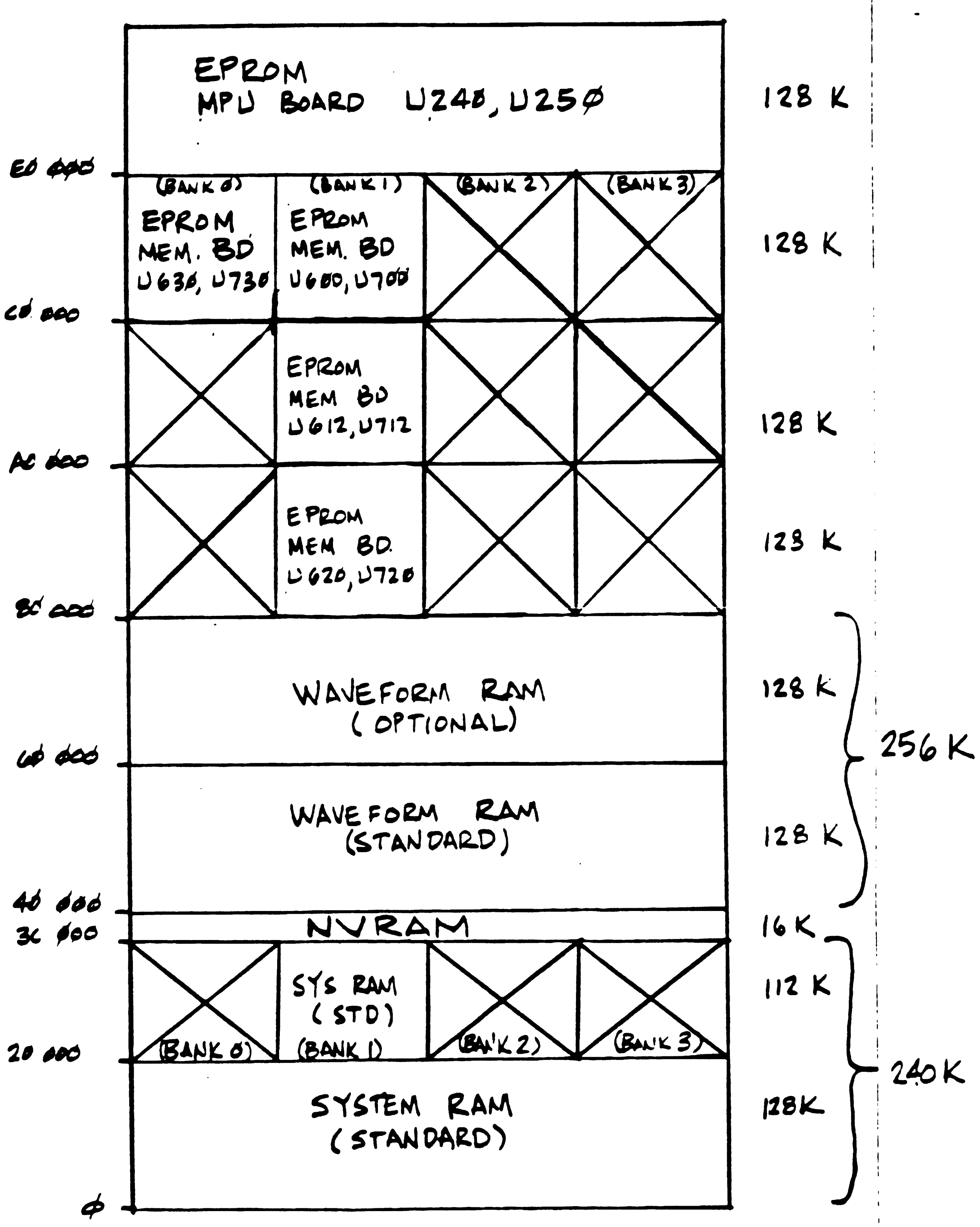
EXECUTIVE PROCESSOR BRD. A17



MAIN MEMORY BRD. A18



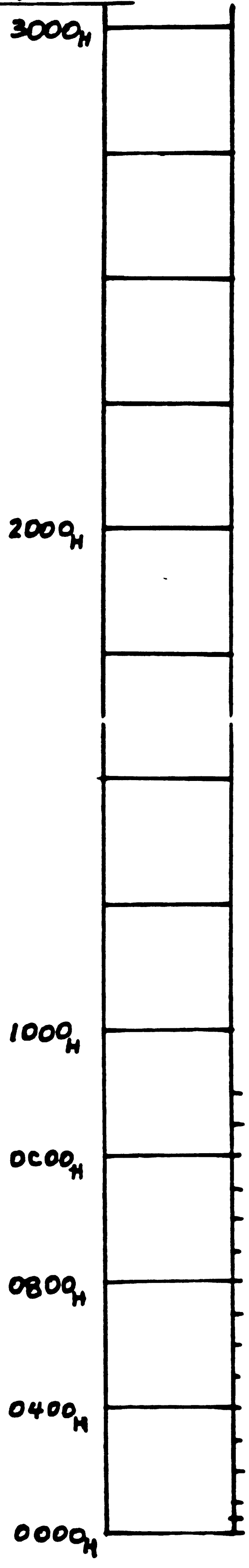
ET EXP MEMORY MAP



3-11-86

EΦ1 I/O MAP

BØ 10/11/85 1/3 ARK



NOTES:
1.) I/O IS ADDRESSED IN BLOCKS OF 0100_H MINIMUM. THEREFORE, I/O DEVICES MAY BE SELECTED AT MORE THAN ONE ADDRESS. THE RECOMENDED ADDRESS IS THE ONE GIVEN IN THE "DEFINED ADDRESS" COLLUMN.
2.) ALL I/O ADDRESSES ABOVE 8000_H ARE COMPLETELY USED BY THE 14 POSSIBLE DIAGNOSTICS ADDRESSES.

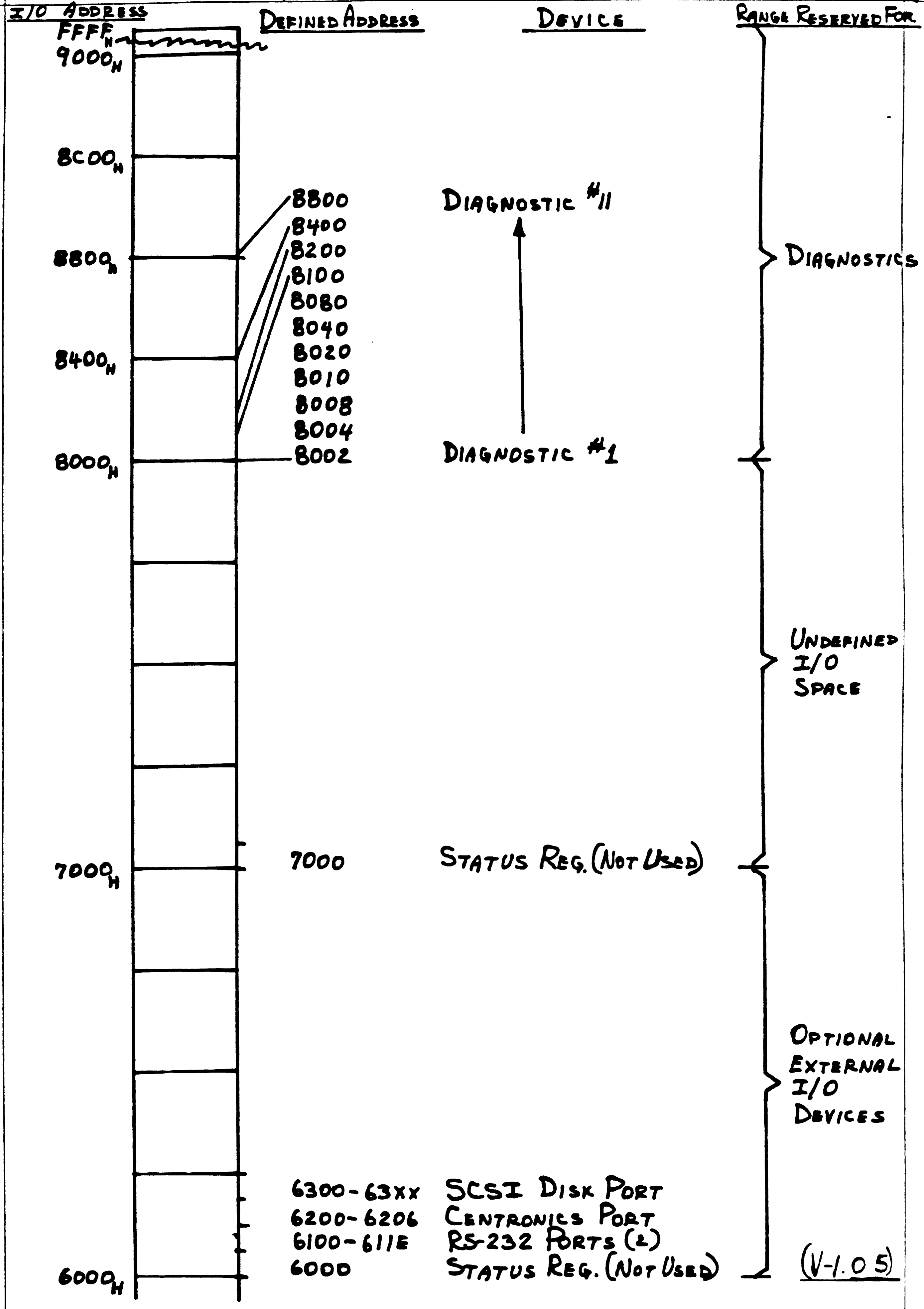
DEFINED ADDRESS	DEVICE	RANGE RESERVED FOR
		PLUG-IN I/O
2000-203E	SDI PLUG-IN INTERFACE	
		MMU
		COMPRESSOR
1000-1019	COMPRESSOR	
0A00-0AFF	DMA CONTROLLER	μP SUPPORT
0900-0902	MASTER INTR. CONTROLLER	
0800-0802	#7 INTR. CONTROLLER (NOT USED)	
0700-0702	#6 INTR. CONTROLLER (")	
0600-0602	#5 INTR. CONTROLLER (")	
0500-0502	#4 INTR. CONTROLLER (")	
0400-0402	#3 INTR. CONTROLLER	
0300-0302	#2 INTR. CONTROLLER (")	
0200-0202	#1 INTR. CONTROLLER	
0100-0102	#Φ INTR. CONTROLLER (")	
00F8-00FE	NUMERIC PROCESSOR	
000Ø	BANK SWITCH ADDRESS	

(V-1.05)

EΦ1 I/O MAP   BΦ   10/11 85   2/3 ARK

I/O Address	Defined Address	Device	Range Reserved For
6000H			Optional Internal I/O Devices
5000H	5000	Status Reg. (Not Used)	
			Standard External I/O Devices
	4200-421E	RS-232 Ports (2)	
	4100-410E	GPIB Port	
4000H	4000	Status Reg. (Not Used)	
			Standard Internal I/O Devices
	3900	Temp. Sensor	
	3800	Tone Generator	
	3700-373E	Real Time Clock	
	3600	Lt. Digital Pot	
	3500	Rt. Digital Pot	
	3400	T.P. LED ON/OFF	
	3300-3302	Touch Panel	
	3200	Timer Configuration	
	3100-3906	Timer	
3000H	3000	Status Req. (Not Used)	(V-1.05)

E∅1 I/O MAP | B∅ 10/11/85 3/3 ARK



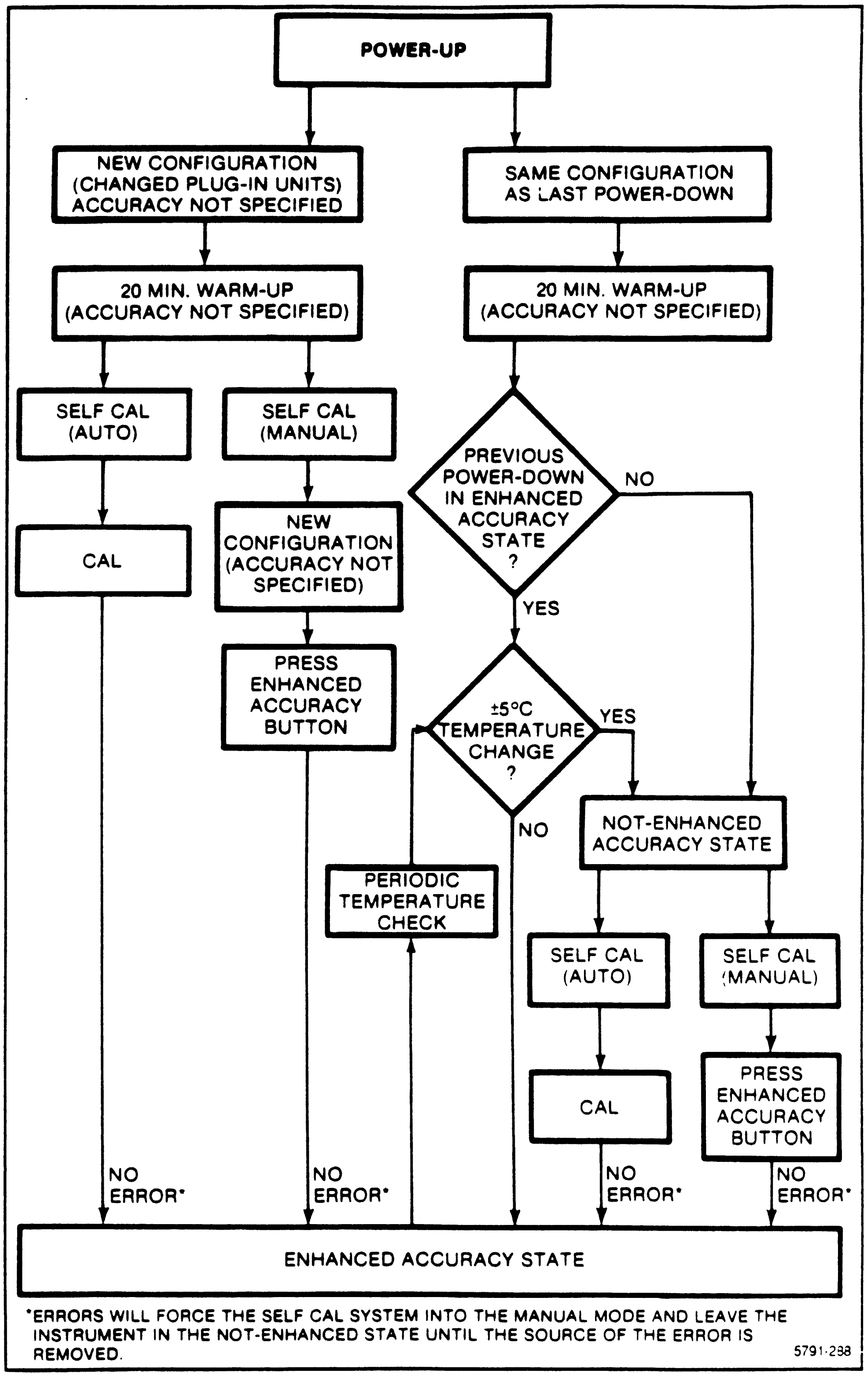
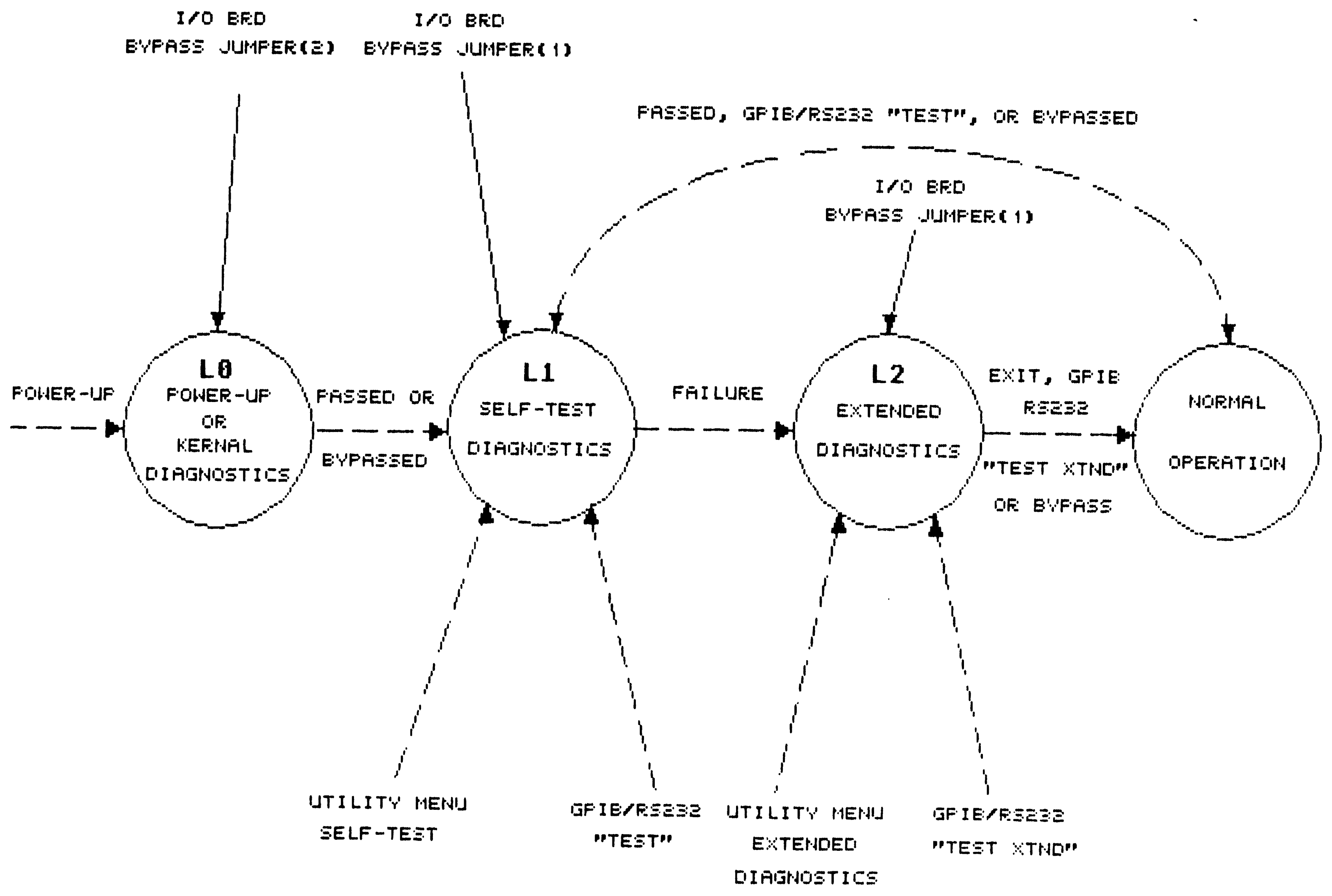

POWER-UP
NEW CONFIGURATION (CHANGED PLUG-IN UNITS) ACCURACY NOT SPECIFIED
SAME CONFIGURATION AS LAST POWER-DOWN
20 MIN. WARM-UP (ACCURACY NOT SPECIFIED)
20 MIN. WARM-UP (ACCURACY NOT SPECIFIED)
SELF CAL (AUTO)
SELF CAL (MANUAL)
PREVIOUS POWER-DOWN IN ENHANCED ACCURACY STATE ?
NO
CAL
NEW CONFIGURATION (ACCURACY NOT SPECIFIED)
YES
PRESS ENHANCED ACCURACY BUTTON
±5°C TEMPERATURE CHANGE ?
YES
NO
NOT-ENHANCED ACCURACY STATE
PERIODIC TEMPERATURE CHECK
SELF CAL (AUTO)
SELF CAL (MANUAL)
CAL
PRESS ENHANCED ACCURACY BUTTON
NO ERROR*
NO ERROR*
NO ERROR*
NO ERROR*
ENHANCED ACCURACY STATE
*ERRORS WILL FORCE THE SELF CAL SYSTEM INTO THE MANUAL MODE AND LEAVE THE INSTRUMENT IN THE NOT-ENHANCED STATE UNTIL THE SOURCE OF THE ERROR IS REMOVED.
5791-288

Figure 2-6. Calibration System Functional Block Diagram.



DIAGNOSTICS PHASES & CONTROL FLOW

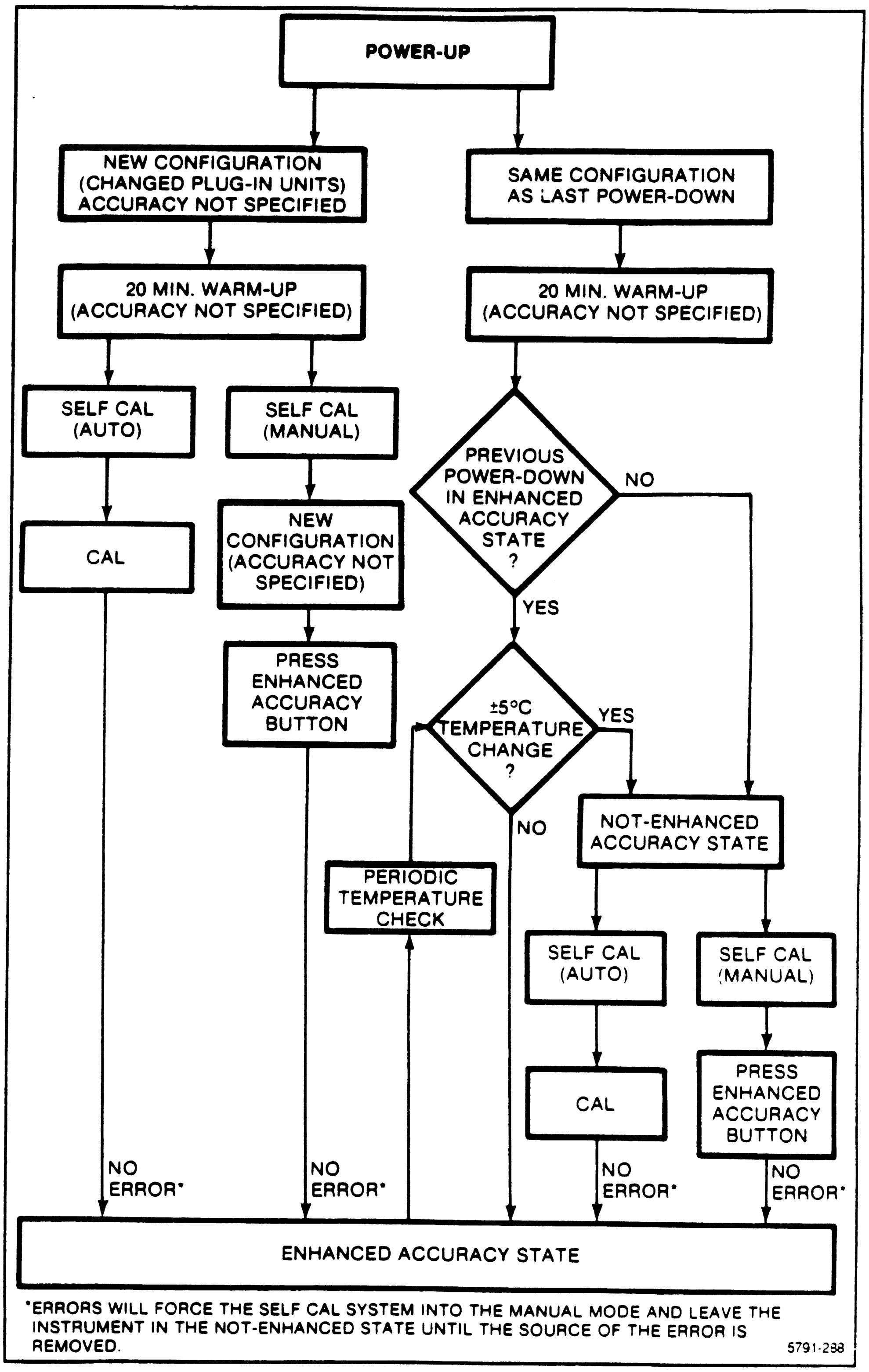
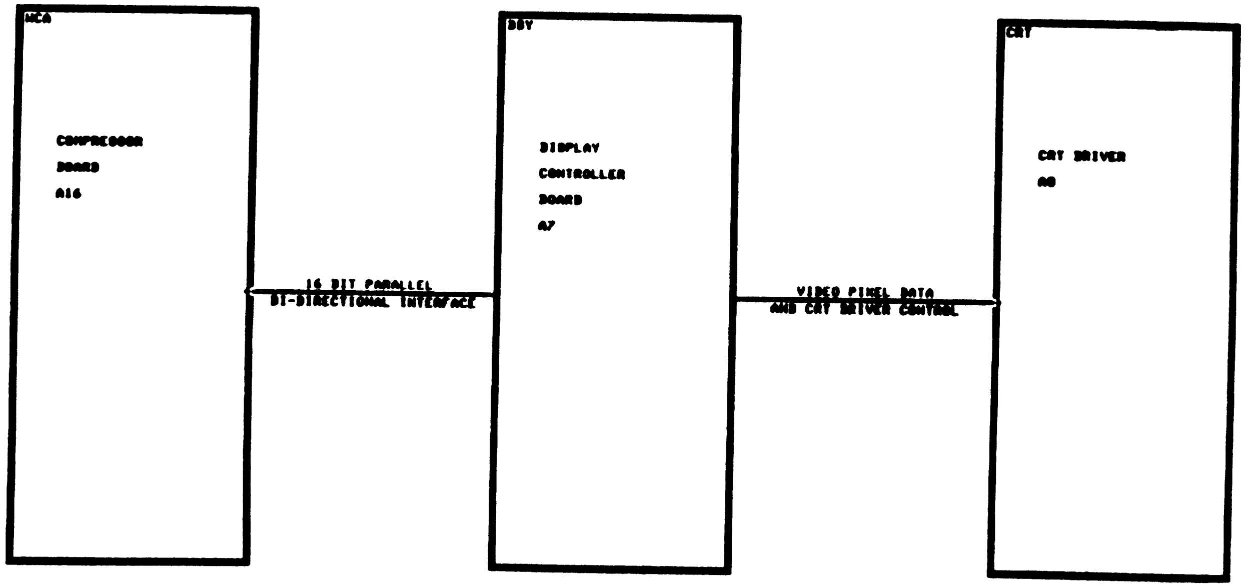

POWER-UP
NEW CONFIGURATION (CHANGED PLUG-IN UNITS) ACCURACY NOT SPECIFIED
SAME CONFIGURATION AS LAST POWER-DOWN
20 MIN. WARM-UP (ACCURACY NOT SPECIFIED)
20 MIN. WARM-UP (ACCURACY NOT SPECIFIED)
SELF CAL (AUTO)
SELF CAL (MANUAL)
PREVIOUS POWER-DOWN IN ENHANCED ACCURACY STATE ?
NO
CAL
NEW CONFIGURATION (ACCURACY NOT SPECIFIED)
YES
PRESS ENHANCED ACCURACY BUTTON
±5°C TEMPERATURE CHANGE ?
YES
NO
NOT-ENHANCED ACCURACY STATE
PERIODIC TEMPERATURE CHECK
SELF CAL (AUTO)
SELF CAL (MANUAL)
CAL
PRESS ENHANCED ACCURACY BUTTON
NO ERROR*
NO ERROR*
NO ERROR*
NO ERROR*
ENHANCED ACCURACY STATE
*ERRORS WILL FORCE THE SELF CAL SYSTEM INTO THE MANUAL MODE AND LEAVE THE INSTRUMENT IN THE NOT-ENHANCED STATE UNTIL THE SOURCE OF THE ERROR IS REMOVED.
5791-288

Figure 2-6. Calibration System Functional Block Diagram.

11401/11402 DISPLAY CONTROLLER BLOCK DIAGRAM

LEVEL 0

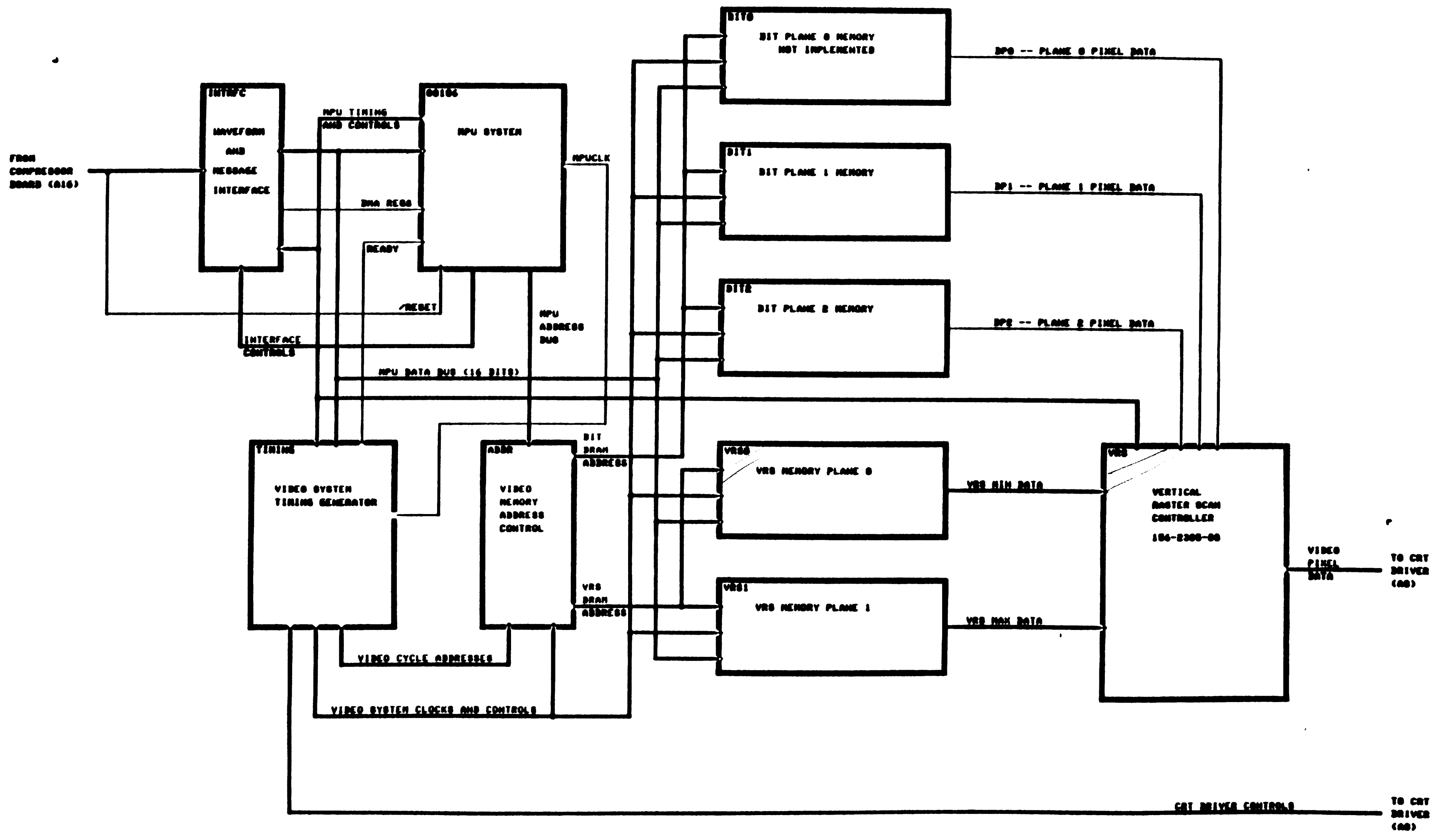


24 SEP 86 23:25 /USER/JOHNS/DBY 1.DRAW

11401/402

11401/11402 DISPLAY CONTROLLER BLOCK DIAGRAM

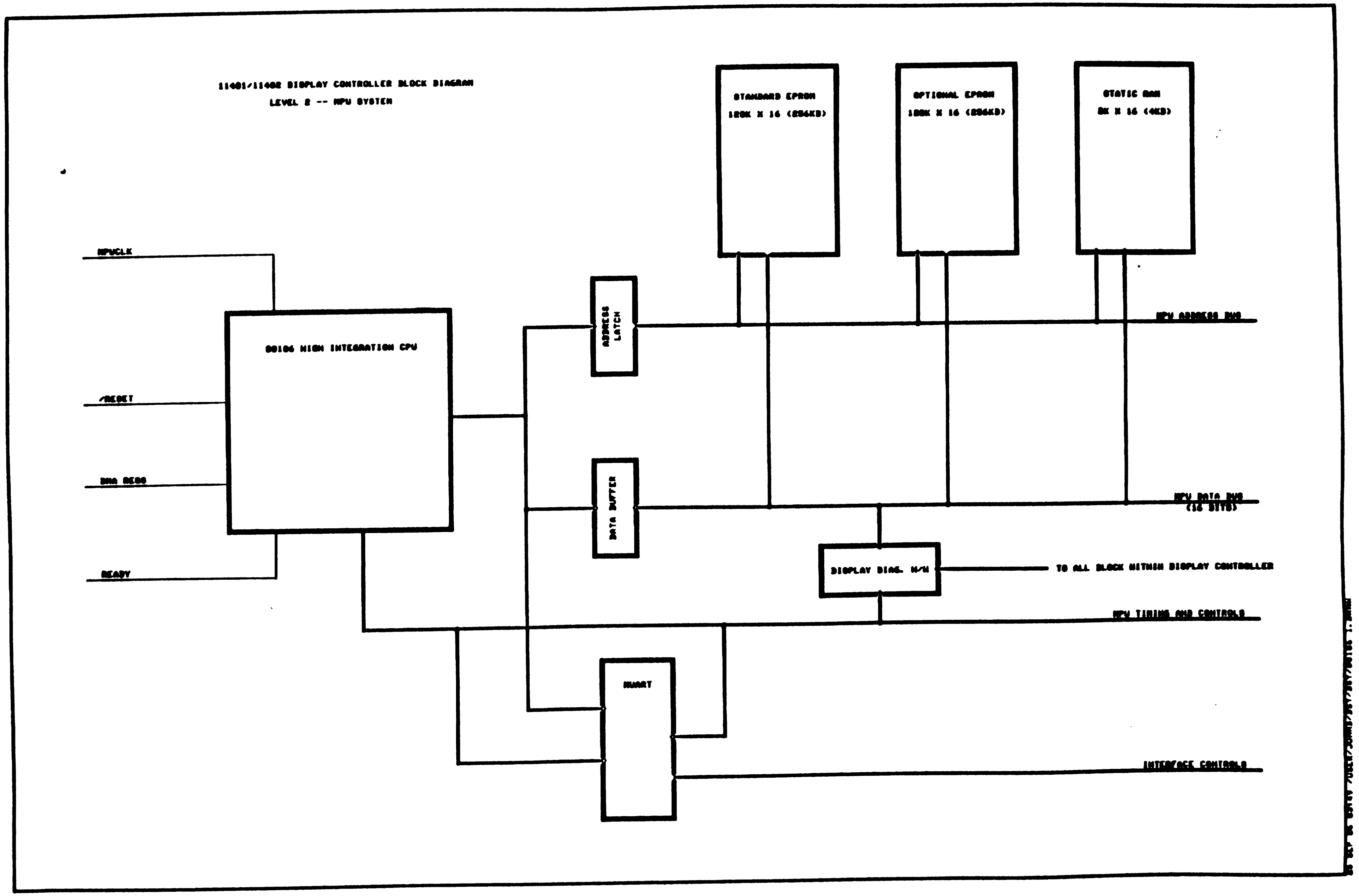
LEVEL 1 -- ENTIRE DISPLAY CONTROLLER



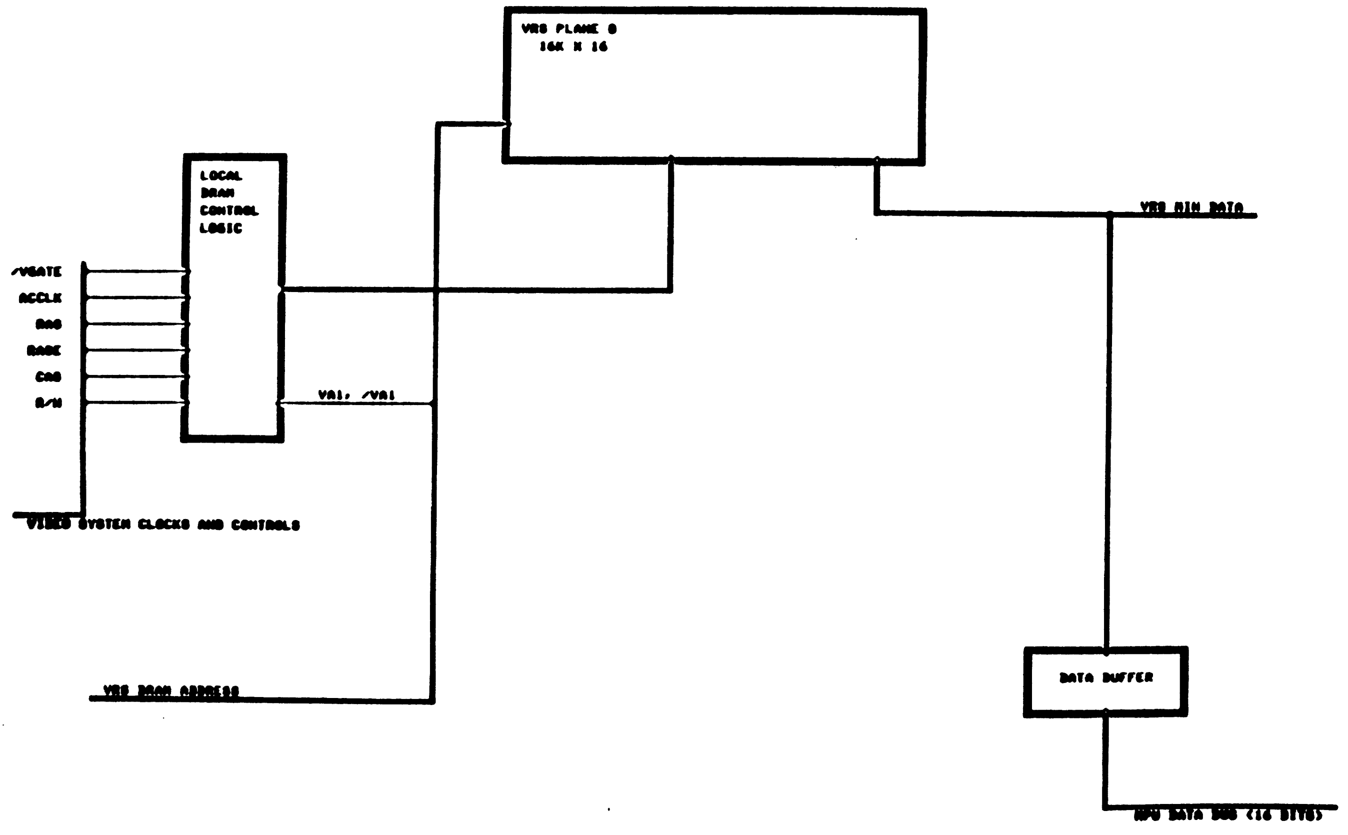
[illegible]

11401/11402 DISPLAY CONTROLLER BLOCK DIAGRAM

LEVEL 2 -- MPU SYSTEM



11401/11402 DISPLAY CONTROLLER BLOCK DIAGRAM
LEVEL 8 -- VRS PLANE 0 MEMORY

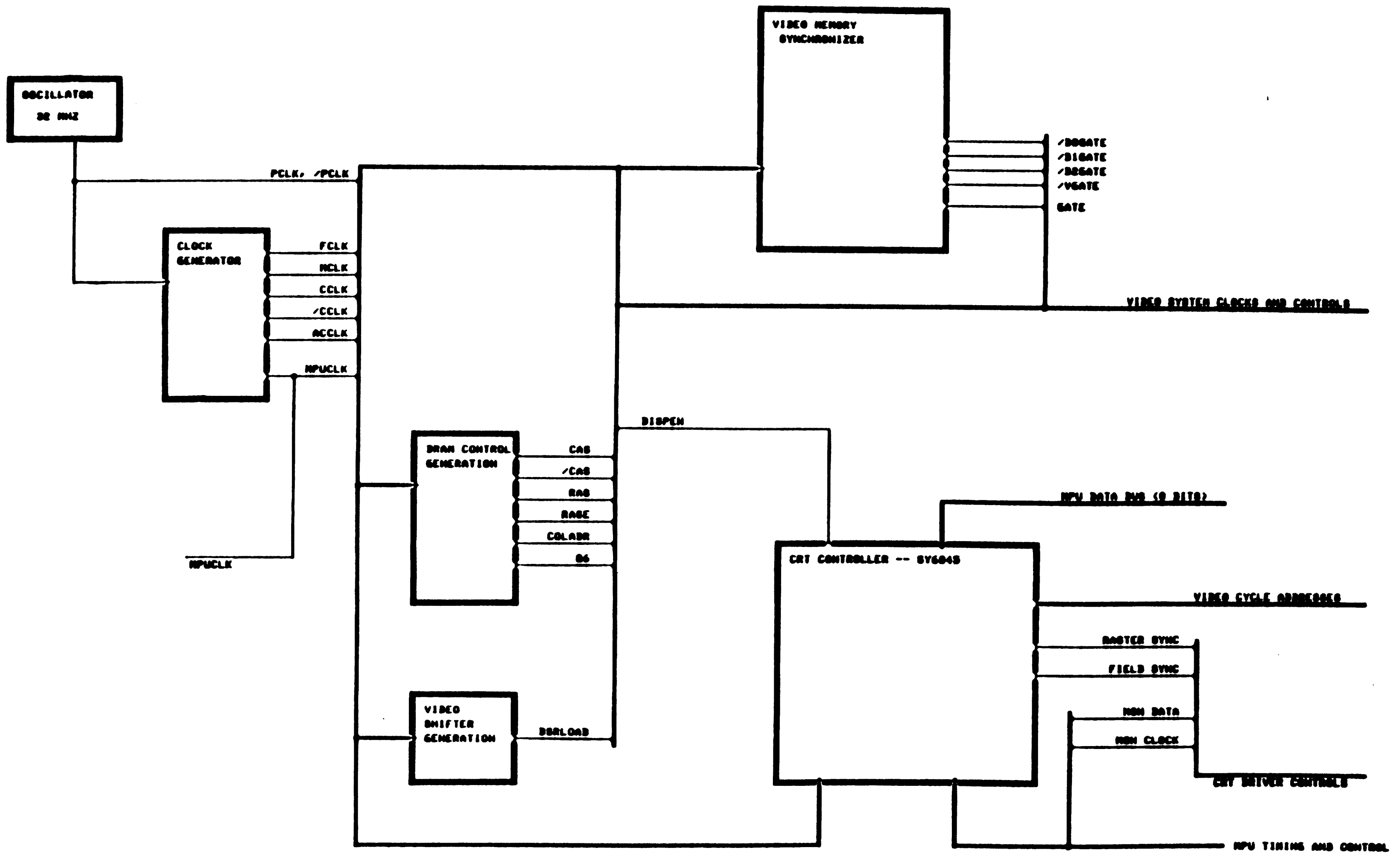


[illegible]

[illegible]

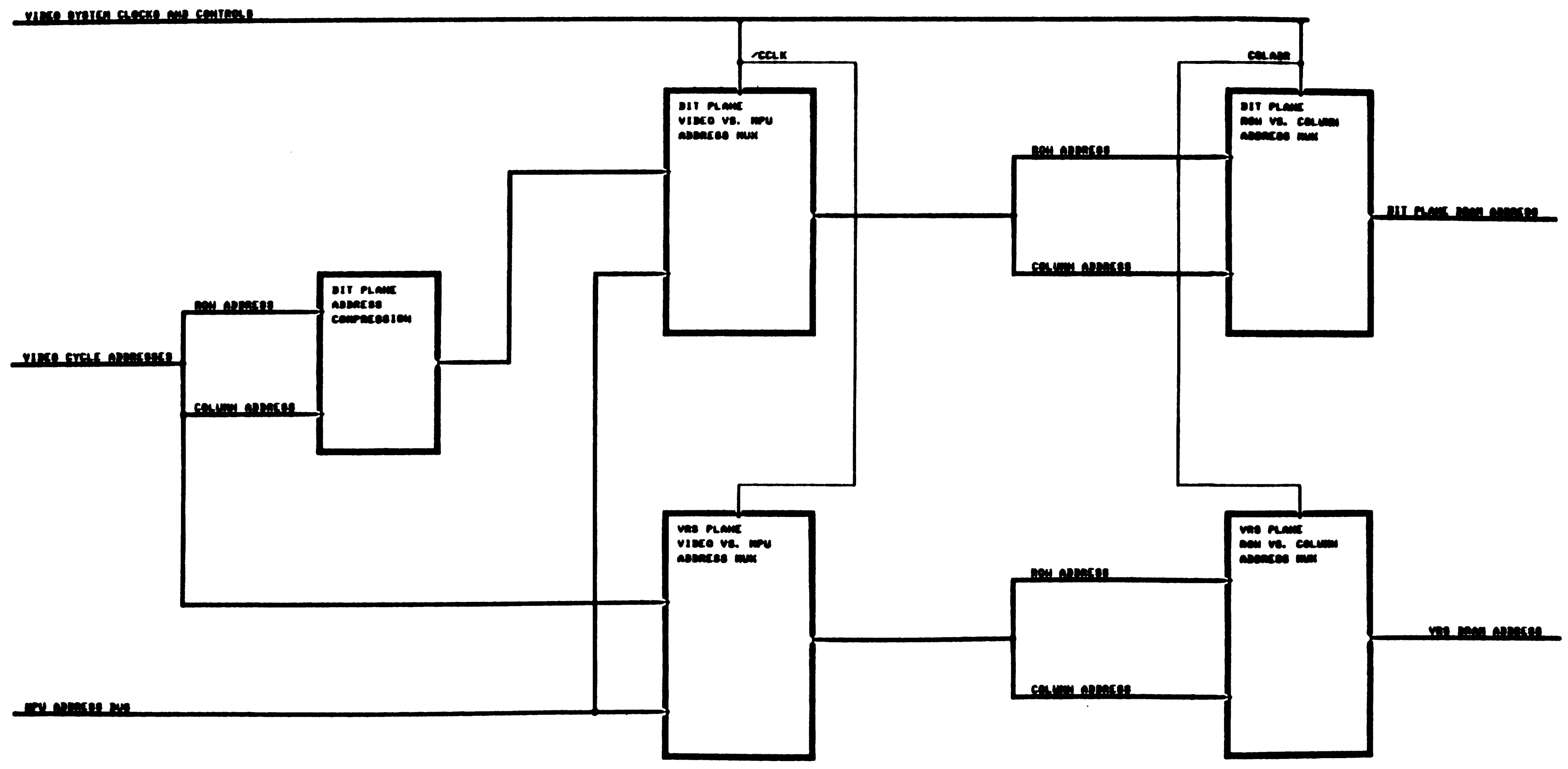
11401/11402 DISPLAY CONTROLLER BLOCK DIAGRAM

LEVEL 2 -- VIDEO SYSTEM TIMING GENERATOR



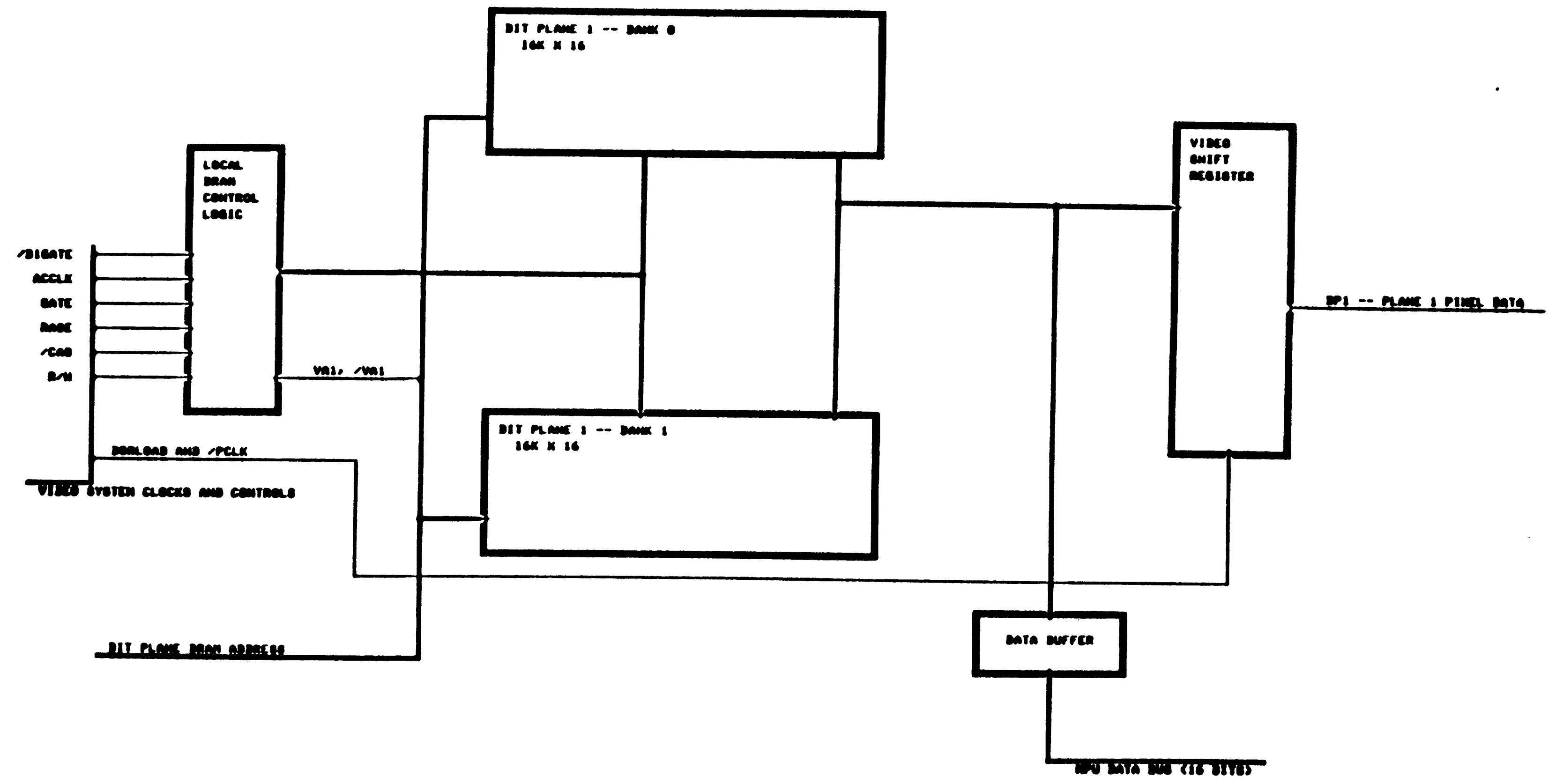
25 SEP 86 05:02 /USER/JOHNS/BSY/BSY/TIMING 1.DRAW

11401/11402 DISPLAY CONTROLLER BLOCK DIAGRAM
LEVEL2 -- VIDEO MEMORY ADDRESS CONTROL

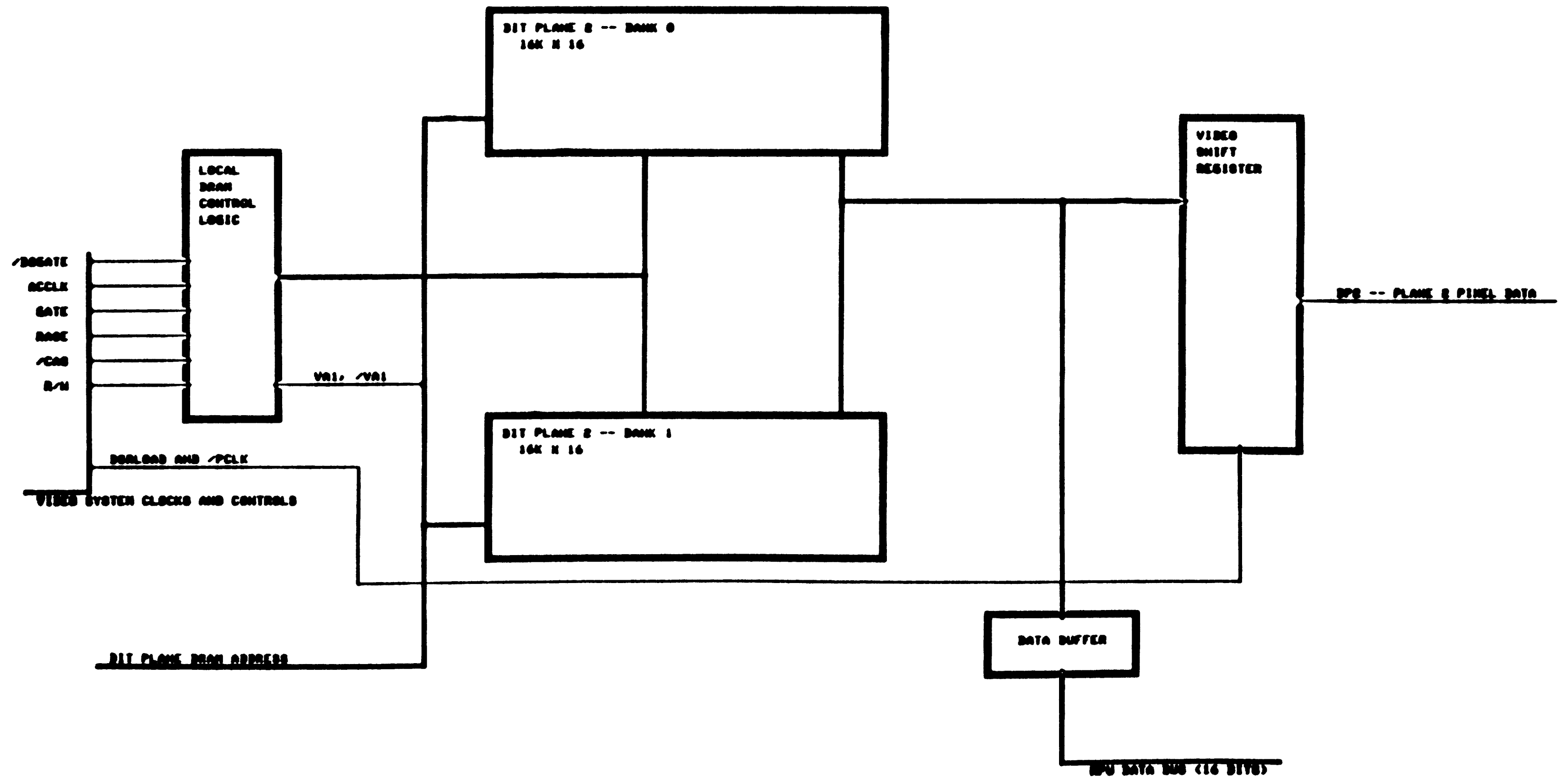

VIDEO SYSTEM CLOCKS AND CONTROLS
/CCLK
COLADR
BIT PLANE VIDEO VS. MPU ADDRESS MUX
BIT PLANE ROW VS. COLUMN ADDRESS MUX
ROW ADDRESS
COLUMN ADDRESS
BIT PLANE DRAM ADDRESS
VIDEO CYCLE ADDRESSES
ROW ADDRESS
COLUMN ADDRESS
BIT PLANE ADDRESS COMPRESSION
VRS PLANE VIDEO VS. MPU ADDRESS MUX
VRS PLANE ROW VS. COLUMN ADDRESS MUX
ROW ADDRESS
COLUMN ADDRESS
VRS DRAM ADDRESS
MPU ADDRESS BUS

11401/11402 DISPLAY CONTROLLER BLOCK DIAGRAM

LEVEL 2 -- BIT PLANE 1 MEMORY



11401/11402 DISPLAY CONTROLLER BLOCK DIAGRAM
LEVEL 2 -- BIT PLANE 2 MEMORY

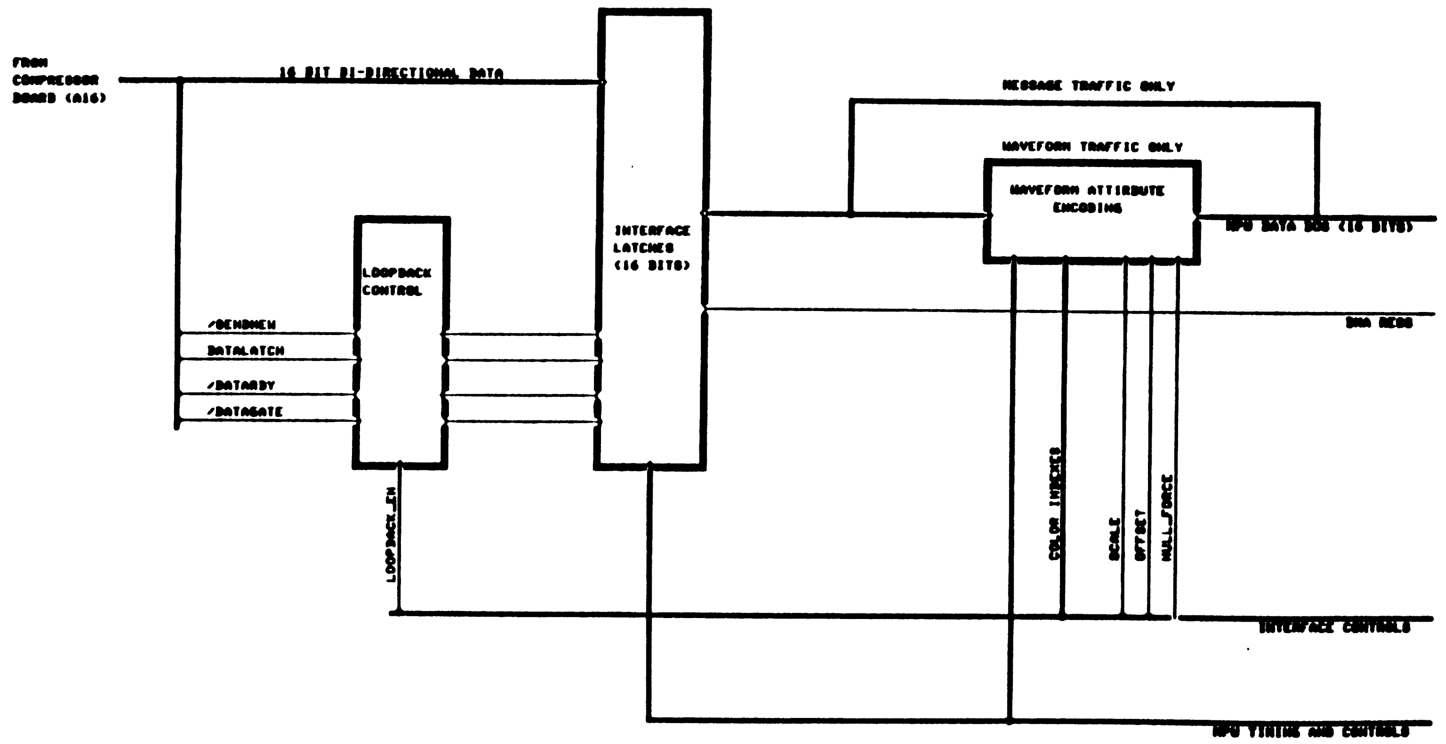


[illegible]

ET 011/6

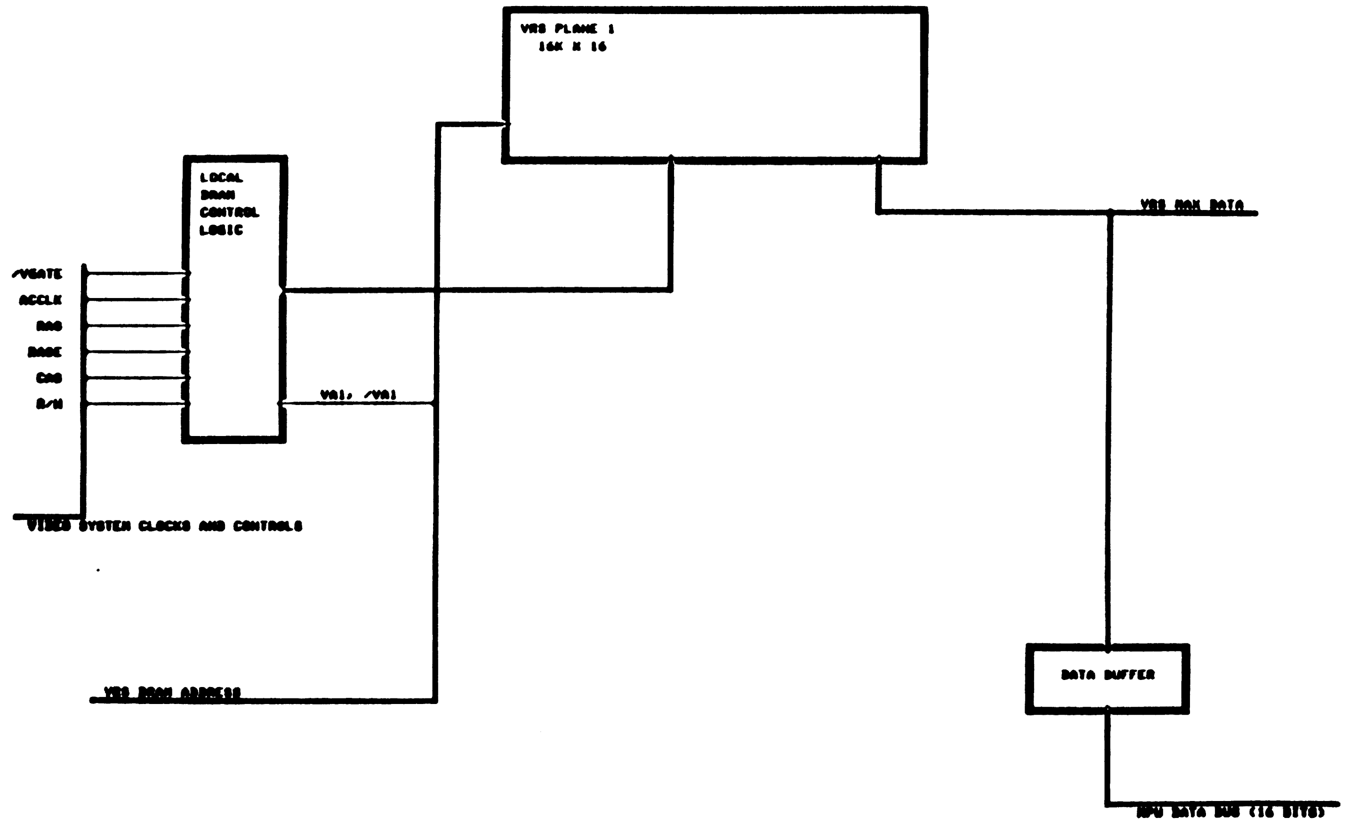
11401/11402 DISPLAY CONTROLLER BLOCK DIAGRAM

LEVEL 2 -- WAVEFORM AND MESSAGE INTERFACE



11481/11482 DISPLAY CONTROLLER BLOCK DIAGRAM

LEVEL 2 -- VRS PLANE 1 MEMORY

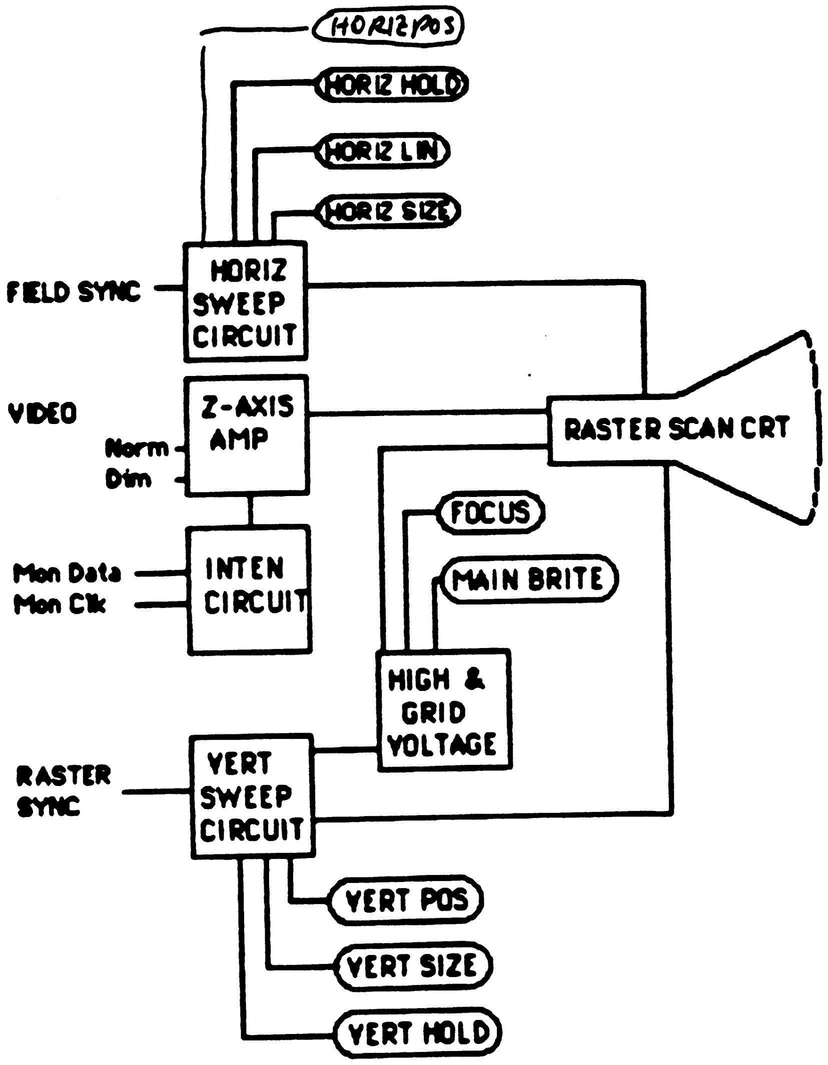


[illegible]

CT-011/9

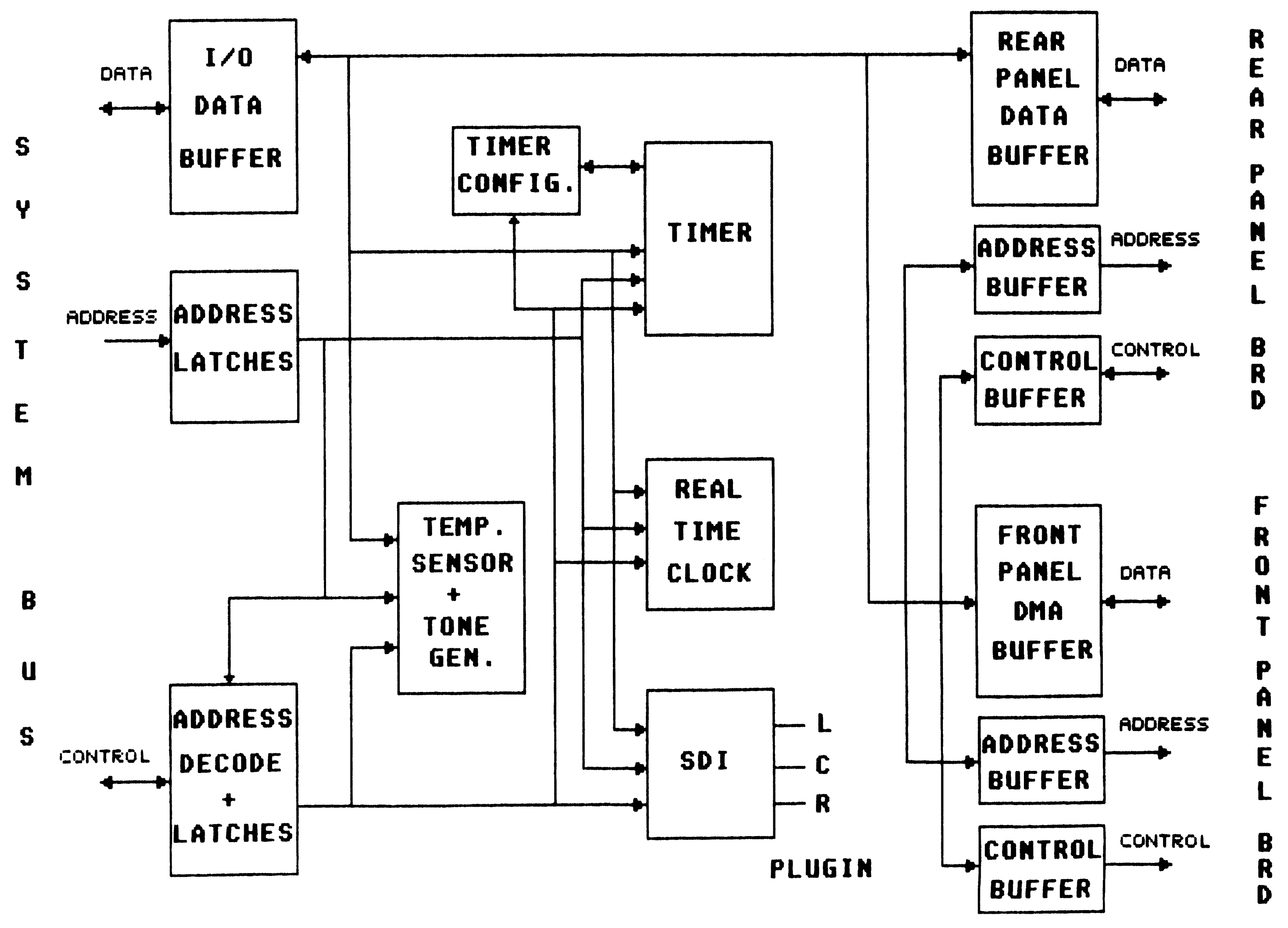
ET-011/10

CRT DRIVER BLOCK DIAGRAM



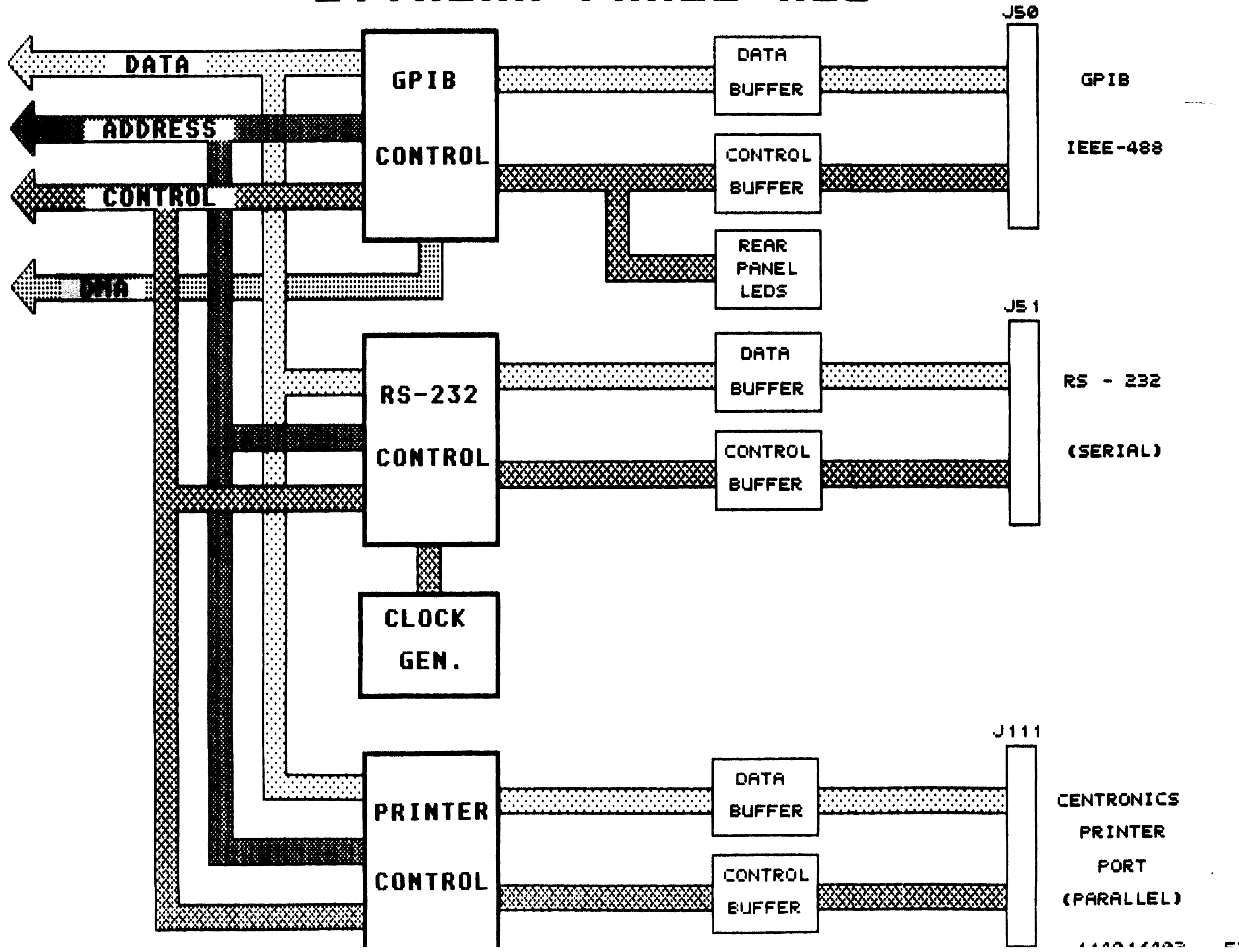
9-25-86

ET. I/O BOARD A14



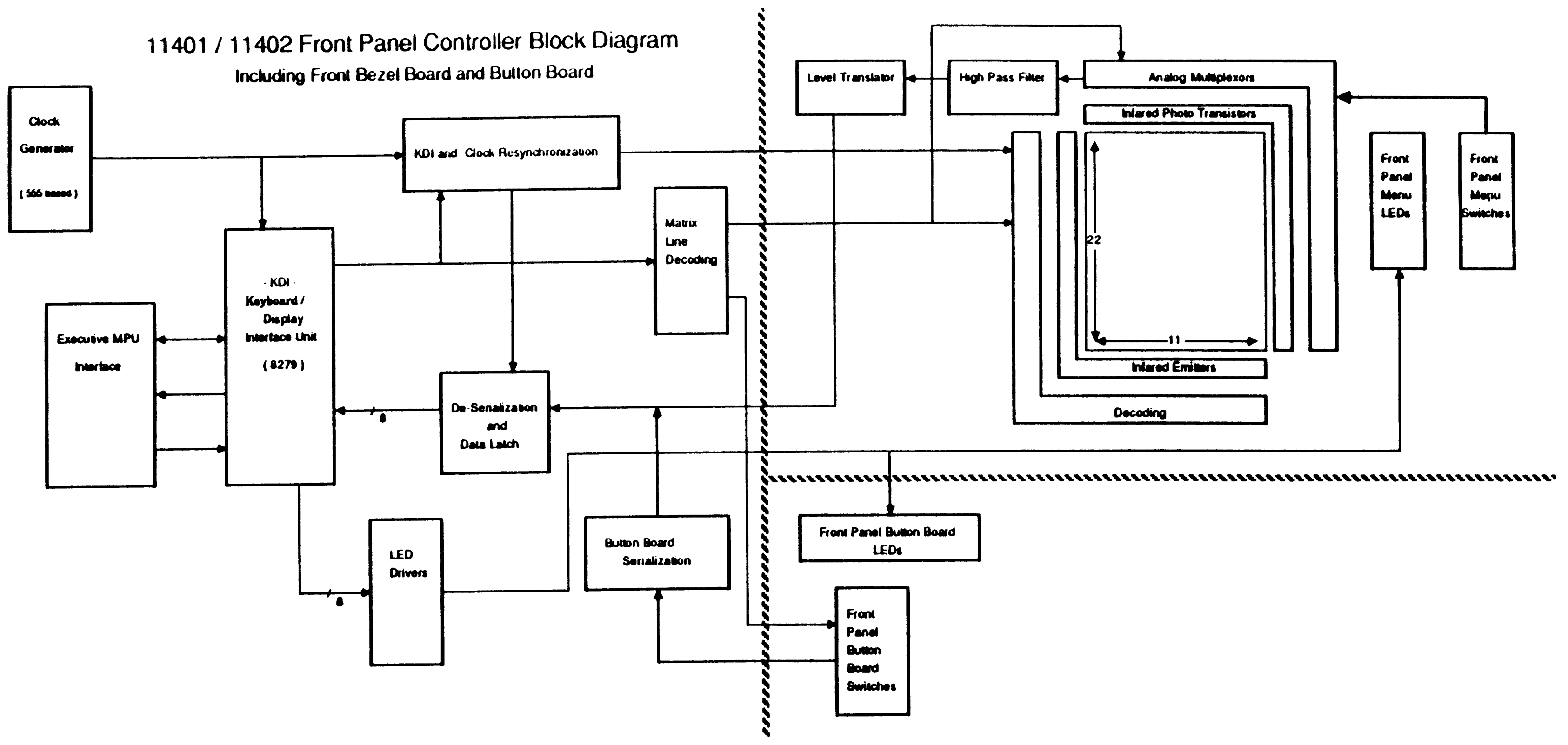
11401/402 ET.005

ET.REAR PANEL A12



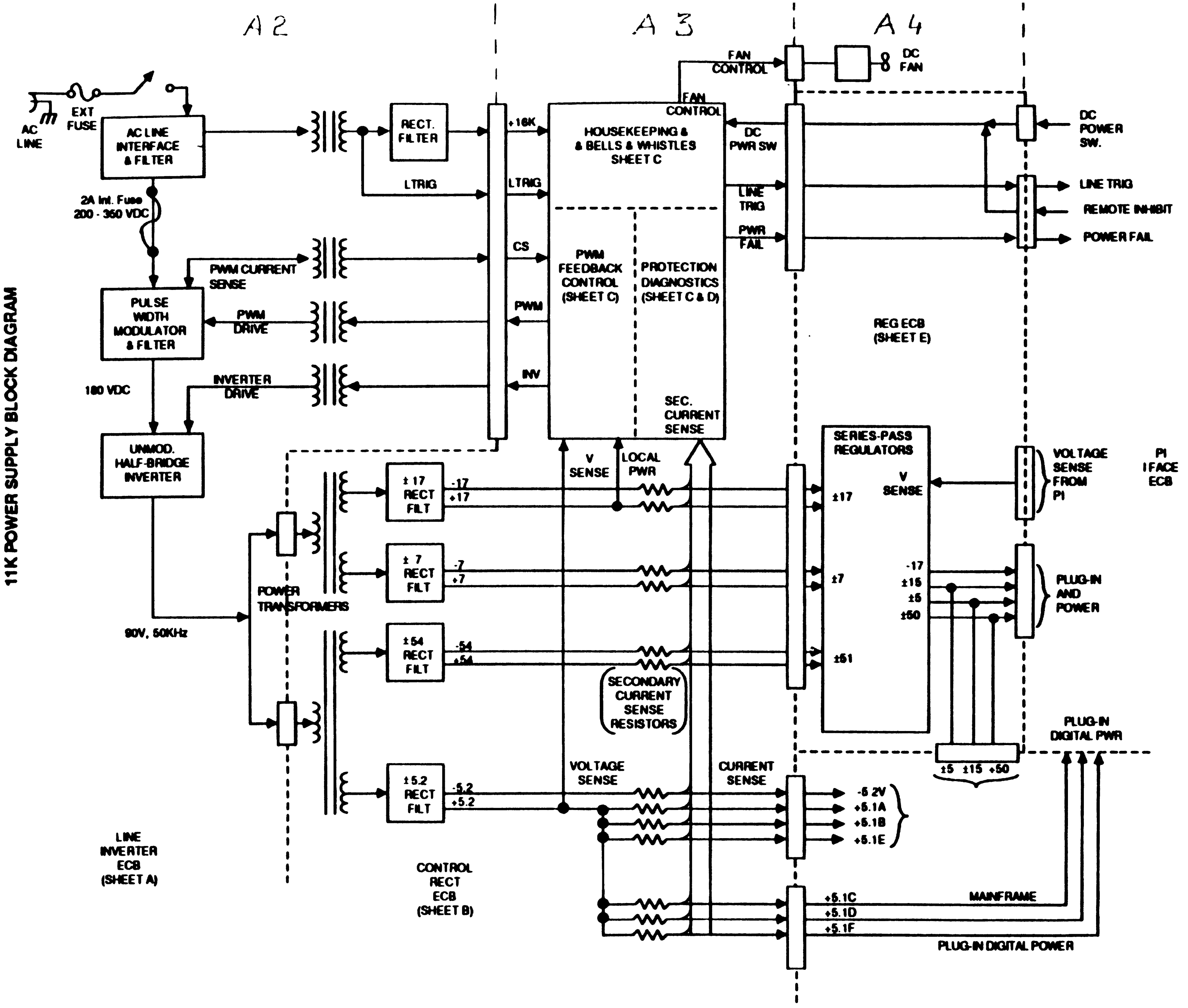
11401 / 11402 Front Panel Controller Block Diagram

Including Front Bezel Board and Button Board

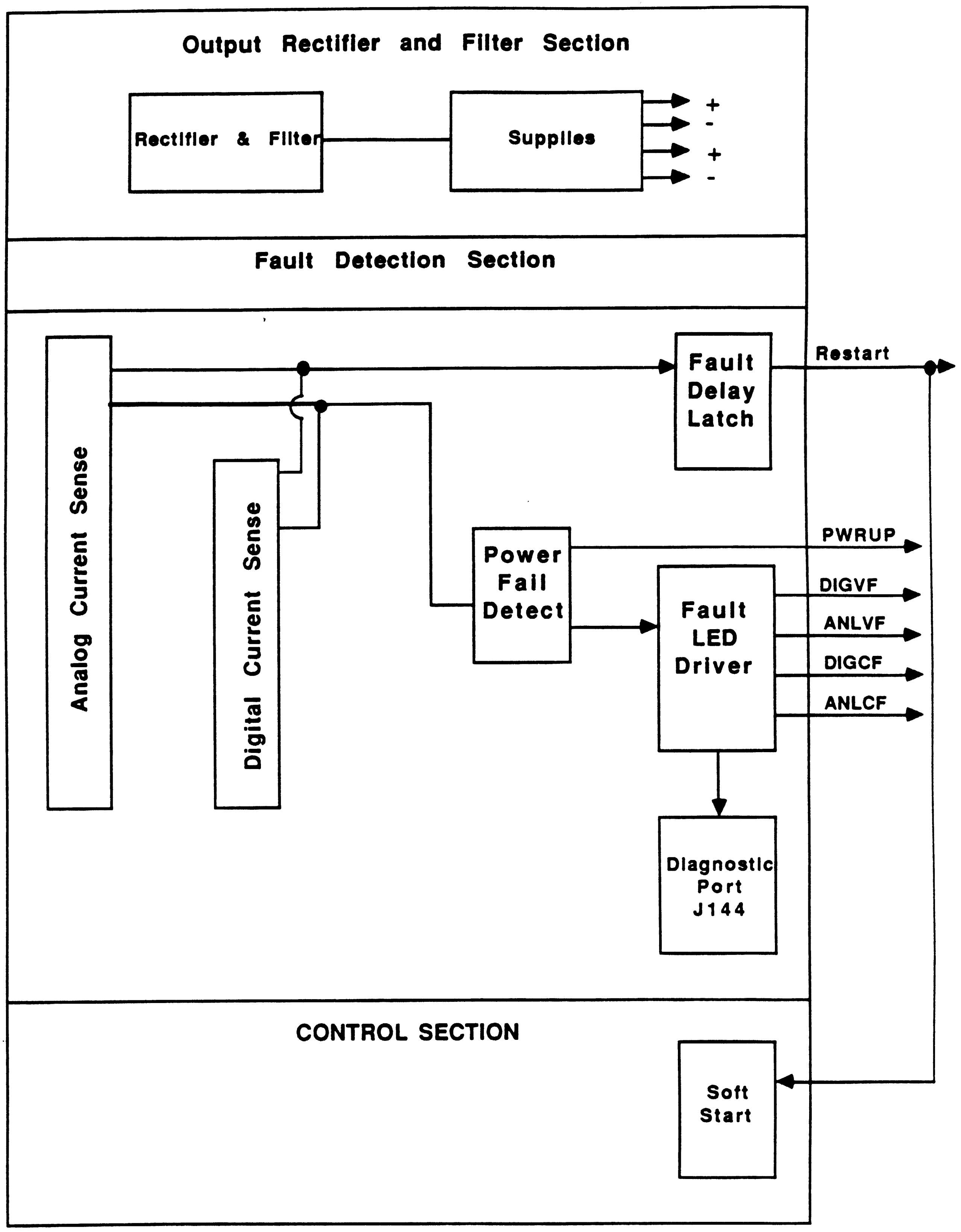


Clock Generator (555 based)
KDI and Clock Resynchronization
Executive MPU Interface
· KDI · Keyboard / Display Interface Unit (8279)
Matrix Line Decoding
De-Serialization and Data Latch
8
8
LED Drivers
Button Board Serialization
Level Translator
High Pass Filter
Analog Multiplexors
Infared Photo Transistors
22
11
Infared Emitters
Decoding
Front Panel Menu LEDs
Front Panel Menu Switches
Front Panel Button Board LEDs
Front Panel Button Board Switches

11K POWER SUPPLY BLOCK DIAGRAM



A3 BOARD


Output Rectifier and Filter Section
Rectifier & Filter
Supplies
+
-
+
-
Fault Detection Section
Analog Current Sense
Digital Current Sense
Fault Delay Latch
Restart
Power Fail Detect
PWRUP
Fault LED Driver
DIGVF
ANLVF
DIGCF
ANLCF
Diagnostic Port J144
CONTROL SECTION
Soft Start

CB

....

11000 INSTALLATION GUIDELINE

....

Schedule Time and Place

- Advise the customer that a user orientation will be part of the installation.

- The orientation will be about 30 to 45 minutes after your arrival on-site, and users attending the orientation should assemble at this time.

- Total installation should take approximately one (1) hour, maximum of one and a half (1 1/2) hours.

Unpacking and Setup

- Unpack the shipment

- Verify that the customer has received the complete order;
  All products ordered
  All options ordered
  All standard accessories
  Complete set of operators manuals

- Install rackmount option, if applicable (Instrument will not be installed in the rack).

- Verify that power selection and fuses are correct for local power line.

- Plug in the power cord and turn on the main power switch located on the back of the main frame.

NOTE: The main power switch is normally left "on" to minimize warm-up time and to provide maximum accuracy.

- Install 11000 series plug-ins at this time. One vertical amplifier with at least two vertical input channels must be installed in the left slot.

Start Up and Check Out of the Measurement System

- Set the front panel stand by/on switch to "ON" position.

- Verify that the power on self-test and auto-cal was executed without a fault within three (3) minutes. When self test is finished the CRT will display a message, SELF TEST PASSED to show the instrument is working correctly.

- Run the extended diagnostics in the repeated mode for five or ten minutes. Should an error occur, take corrective action. and repair the instrument.

- Apply the calibrator output signal, or TG501/PG506 signals to the inputs. Do an analog system verification as appropriate for the system configuration.

Install and Verify Operation of the Peripherals that are a Part of this Order.

- Connect peripherals included in this order

  . Terminals
  . Computers
  . Printers
  . Cameras

NOTE: Only Tek products sold at the same time with NO options are eligible for hook-up and checkout with the 11000 series oscilloscope system. Connection and verification of compatibility for customer supplied peripherals is not a part of the NO option but may be performed at local managements discretion and policy on an added cost basis.

- Turn on and check out the operation of the printer. this is only valid for the 11401 and 11402.

- Printer settings are:

- Turn on and check out the operation of the terminal (RS232 buss)

- Terminal settings are:

- Computer Verification (IEEE-488; GPIB)

....

E11000 ORIENTATION GUIDELINES

....

User Orientation

- When you are finished with the installation notify the customer that you are ready to begin the user orientation.

- Introduce yourself and explain the purpose of the orientation is to teach the users basic operation of the 11000 series oscilloscope.

  . How to display a waveform

  . How to measure a variety of pulse parameters, automatically

  . How to make a hard copy of a waveform and measurement results

  . Menu Overview

Getting Started/Basic Operation

- Pass out the manuals the customer received with his order and direct the users to the introduction manual

- Point out both power switches

- Set the front panel on/standby switch to ON

NOTE: At powerup the oscilloscope performs a self-test. When self-test is finished the crt will display a message to show self-test passed.

- Next, press the UTILITY button located in the MENUS selection area next to the display screen.

- Touch the INITIALIZE label displayed in the menu/status area at the bottom of the screen to reset settings to the factory state.

Displaying a Waveform

- Apply a probe to the calibrator output signal, or TG501/PG506 and press the channel selection button or the probe ID button to set a trace on the display screen

- Now press the AUTOSET pushbutton

NOTE: The AUTOSET function automatically chooses a setup that provides a stable display of the signal.

Making a Hardcopy

- IF applicable, push the HARD COPY button to get a hard copy of the displayed waveform and measurement results

NOTE: A paper copy of the information on the display screen is then printed

Adjusting Vertical Size and Position

NOTE: The touch panel ON/OFF button located to the right of the display must be on to use the touch screen functions

- To demonstrate the touch screen operation press the VERTICAL ICON, found in the upper left corner of the display screen

NOTE: This operation assigns the volts/division and vertical position functions for the selected channel to the control knobs.

- Turn the control knobs to demonstrate how vertical size and position of the waveform are adjusted

Adjusting the Horizontal Size and Position

- Touch the HORIZONTAL ICON and turn the selected channel control knobs to demonstrate how to change the time-base and horizontal position

Performing Measurements

- Press the measure button located in the menus area to the right of the display screen

- Touch the following labels in the measurements pop-up menu PK-PK, WIDTH, PERIOD, RISE, FREQUENCY, and FALL

NOTE: As each measurement is selected a label for it will appear in the Measurement menu at the bottom of the screen

- Touch the EXIT MENU label to remove the pop-up menu

NOTE: After the pop-up menu disappears the waveform being measured is again displayed.

- Press the TRIGGER Menu button located in the menus section to the right of the display screen

NOTE: Pressing the trigger menu button displays the trigger menu status. Each label allows you to set a particular waveform function by

.Toggling it

.Assigning the control knobs

.Or displaying the list of options in the pop up menu

- Touch the SLOPE label to demonstrate how the slope toggles between + and -

- Touch the LEVEL control to assign the level and TIME HOLDOFF functions to the control knobs

- Touch the SOURCE DESC label to display the pop up menu for trigger source alternatives and select one of the source labels

NOTE: The BACK SPACE label allows you to correct mistakes in a selection.

- After selecting one of the source labels touch the ENTER DESC label. The trigger source status will then be displayed below the SOURCE DESC label

Windows

- To demonstrate the window features touch the WINDOW 1 label at the top of the display screen. This function selects a dual time-base display causing two graticules to be displayed

NOTE: A waveform must be present for the window 1 ICON to become active.

Two windows can be displayed in the lower axis at one time. To view to different portions of the main waveform touch the WINDOW 2 ICON

NOTE: The WINDOW 2 ICON only becomes available after a waveform has been displayed with the WINDOW 1 ICON.

Store/Recall

- Press the STORE/RECALL button to located in the menus section to the right of the display screen to display the store /recall menu

- Touch the STORE WAVEFORM label on the display screen to display the store waveform pop-up menu

- Touch the L1 WINDOW 2 label or STORE ALL label to demonstrate how to store the current waveform

- Next, touch the STORE SETTING label to display the store setting pop-up menu

- TO DISPLAY A STORED WAVEFORM PRESS THE STORE/RECALL button in the menus section located to the right of the display area

- Then touch the RECALL/WAVEFORM label to display the stored waveform

You have now completed the user orientation

- Press the UTILITY button located in the menus section to the right of the display area

- Touch the INITIALIZE label to restore the instrument to factory setting

- Direct the users to the USER REFERENCE MANUAL as a reference to indepth information on the subjects you have covered

- Ask if there are any further questions

- Thank those attending

-Orientation Completed

Instrument Options

Your instrument may be equipped with one or more instrument options. A brief description of each available option is given in the following discussion. Option information is incorporated into the appropriate sections of the manual. Refer to Table 5-1 and the Table of Contents for location of option information. For further information and prices of instrument options, see your Tektronix Products catalog or contact your Tektronix Field Office

WARNING

To avoid electric shock hazard, operating personnel must not remove the protective instrument covers. Component replacement and internal adjustments must be made by qualified service personnel only.

List of Options

Option 1C	Option 1C adds eight bnc connectors to the front- and rear-panels so that signals may be internally routed directly between the two panels. This is especially useful for rackmounted applications. This option can be added at any time.
Option 1R	Option 1R adds slide rails and rackmounting hardware to convert the benchtop instrument to a standard 19-inch rackmount version. This option can be added at any time.
Option 2D	Option 2D expands total waveform memory from 64K to 128K points for storage of waveform records. This option can be added at any time.
Option 4D	Option 4D increases GPIB transfer speed as much as ten times. Improves the overall throughput of the oscilloscope system, especially the transmission of waveform and measurement data. This option can be added any time.
Option A1	Replaces the standard power cord with the Universal European 220 V type power cord.
Option A2	Replaces the standard power cord with the United Kingdom 240 V type power cord.
Option A3	Replaces the standard power cord with the Australian 240 V type power cord.
Option A4	Replaces the standard power cord with the North American 250 V type power cord.
Option A5	Replaces the standard power cord with the Switzerland 240 V type power cord.

TABLE 5-1
Option Information Locator

Option	Location in Manual		Information
	Section	Heading	
Option 1C (Provides back to front to connectors)	5 Instrument Options	Option 1C	Gives brief description of Option 1C.
Option 1R (Provides rack-mount hardware)	1 Installation	Rackmounting	Gives brief description of Option 1R.
	5 Instrument Options	Option 1R	Gives brief description of Option 1R.
Option 2D (Provides memory expansion)	5 Instrument Options	Option 2D	Gives brief description of Option 2D.
Option 4D (Provides increased GPIB data transfer speed)	5 Instrument Options	Option 4D	Gives brief description of Option 4D.
Option A1 (Provides Universal European power cord)	1 Installation	Power-Cord Information Table 1-2	Lists details of Option A1.
	5 Instrument Options	Option A1	Gives brief description of Option A1.
	Appendix A	Power-Cord Options	Lists Option A1.
Option A2 (Provides United Kingdom power cord)	1 Installation	Power-Cord Information Table 1-2	Lists details of Option A2.
	5 Instrument Options	Option A2	Gives brief description of Option A2.
	Appendix A	Power-Cord Options	Lists Option A2.

TABLE 5-1 (cont)
Option Information Locator

Option	Location in Manual		Information
	Section	Heading	
Option A3 (Provides Australian power cord)	1 Installation	Power-Cord Information Table 1-2	Lists details of Option A3.
	5 Instrument Options	Option A3	Gives brief description of Option A3.
	Appendix A	Power-Cord Options	Lists Option A3.
Option A4 (Provides North American power cord)	1 Installation	Power-Cord Information Table 1-2	Lists details of Option A4.
	5 Instrument Options	Option A4	Gives brief description of Option A4.
	Appendix A	Power-Cord Options	Lists Option A4.
Option A5 (Provides Switzerland power cord)	1 Installation	Power-Cord Information Table 1-2	Lists details of Option A5.
	5 Instrument Options	Option A5	Gives brief description of Option A5.
	Appendix A	Power-Cord Options	Lists Option A5.

iAPX 186 HIGH INTEGRATION 16-BIT MICROPROCESSOR

- **Integrated Feature Set**
  - **—Enhanced 8086-2 CPU**
  - **—Clock Generator**
  - **—2 Independent, High-Speed DMA Channels**
  - **—Programmable Interrupt Controller**
  - **—3 Programmable 16-bit Timers**
  - **—Programmable Memory and Peripheral Chip-Select Logic**
  - **—Programmable Wait State Generator**
  - **—Local Bus Controller**
- **High-Performance 8 MHz Processor**
  - **—2 Times the Performance of the Standard iAPX 86**
  - **—4 MByte/Sec Bus Bandwidth Interface**
- **Direct Addressing Capability to 1 MByte of Memory**
- **Completely Object Code Compatible with All Existing iAPX 86, 88 Software**
  - **—10 New Instruction Types**
- **Compatible with 8282/83/86/87, 8288, 8289 Bus Support Components**
- **Complete System Development Support**
  - **—Development Software: Assembler, PL/M, Pascal, Fortran, and System Utilities**
  - **—In-Circuit-Emulator (I²ICE™-186)**
  - **—iRMX™ 86, 88 Compatible (80130 OSF)**
- **Optional Numeric Processor Extension**
  - **—iAPX 186/20 High-Performance 80-bit Numeric Data Processor**

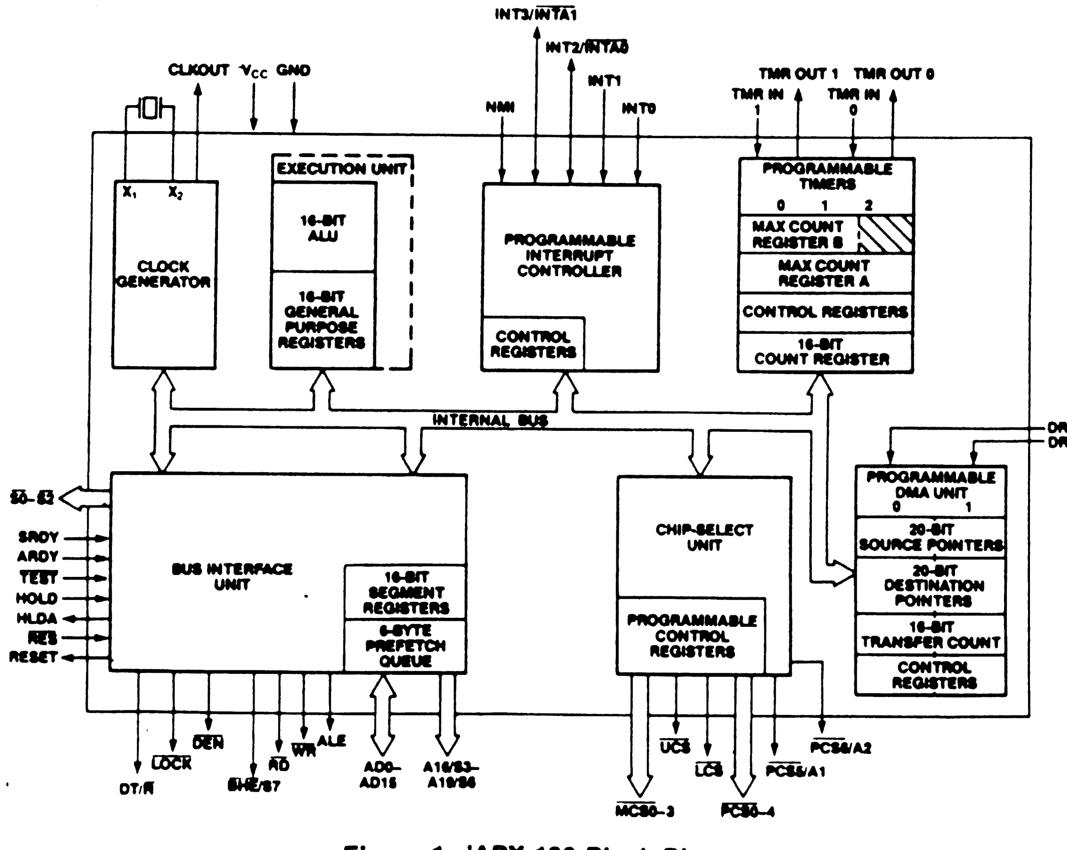


Figure 1. iAPX 186 Block Diagram

DECEMBER 1982
ORDER NUMBER: 210451-002

The Intel iAPX 186 (80186 part number) is a highly integrated 16-bit microprocessor. The iAPX 186 effectively combines 15–20 of the most common iAPX 86 system components onto one. The 80186 provides two times greater throughput than the standard 5 MHz iAPX 86. The iAPX 186 is upward compatible with iAPX 86 and 88 software and adds 10 new instruction types to the existing set.

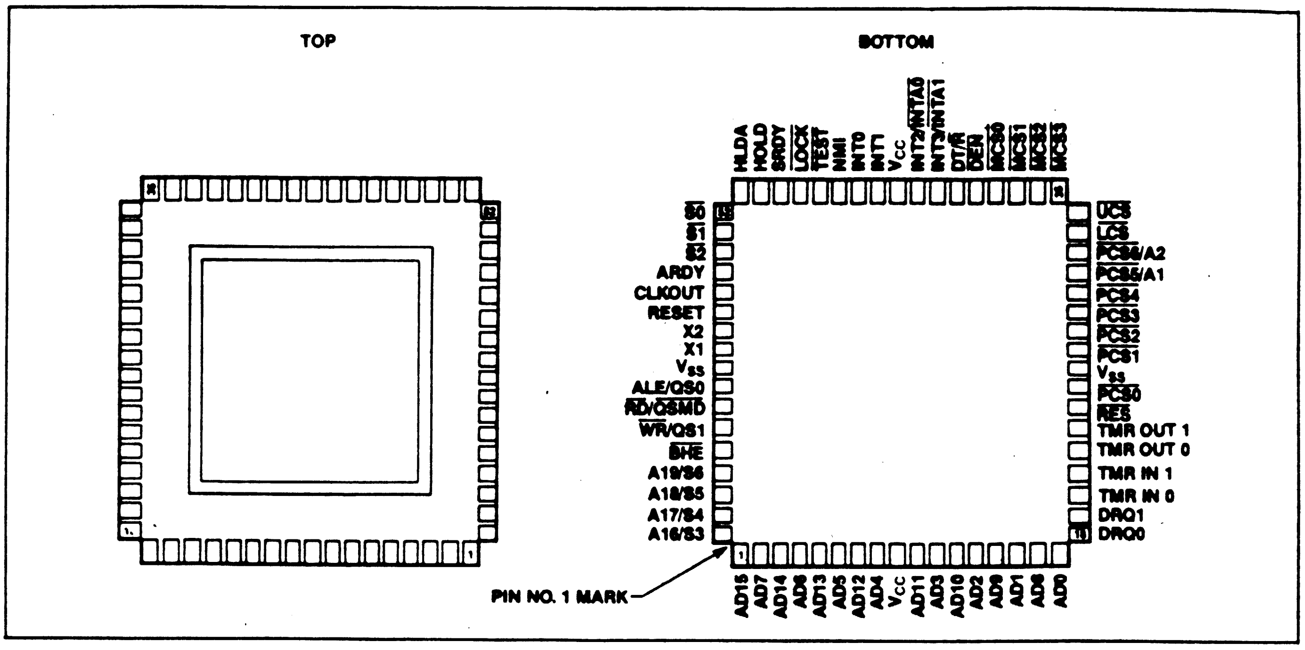


Figure 2. 80186 Pinout Diagram

Table 1. 80186 Pin Description

Symbol	Pin No.	Type	Name and Function
V_{CC}, V_{CC}	9,43	I	System Power: +5 volt power supply.
V_{SS}, V_{SS}	26,60	I	System Ground.
RESET	57	O	Reset Output indicates that the 80186 CPU is being reset, and can be used as a system reset. It is active HIGH, synchronized with the processor clock, and lasts an integer number of clock periods corresponding to the length of the $\overline{RES}$ signal.
X1, X2	59,58	I	Crystal Inputs, X1 and X2, provide an external connection for a fundamental mode parallel resonant crystal for the internal crystal oscillator. X1 can interface to an external clock instead of a crystal. The input or oscillator frequency is internally divided by two to generate the clock signal (CLKOUT).
CLKOUT	56	O	Clock Output provides the system with a 50% duty cycle waveform. All device pin timings are specified relative to CLKOUT. CLKOUT has sufficient MOS drive capabilities for the 8087 Numeric Processor Extension.
$\overline{RES}$	24	I	System Reset causes the 80186 to immediately terminate its present activity, clear the internal logic, and enter a dormant state. This signal may be asynchronous to the 80186 clock. The 80186 begins fetching instructions approximately 7 clock cycles after $\overline{RES}$ is returned HIGH. $\overline{RES}$ is required to be LOW for greater than 4 clock cycles and is internally synchronized. For proper initialization, the LOW-to-HIGH transition of $\overline{RES}$ must occur no sooner than 50 microseconds after power up. This input is provided with a Schmitt-trigger to facilitate power-on $\overline{RES}$ generation via an RC network. When $\overline{RES}$ occurs, the 80188 will drive the status lines to an inactive level for one clock, and then tri-state them.

Table 1. 80186 Pin Description (Continued)

Symbol	Pin No.	Type	Name and Function
$\overline{\text{TEST}}$	47	I	$\overline{\text{TEST}}$ is examined by the WAIT instruction. If the $\overline{\text{TEST}}$ input is HIGH when "WAIT" execution begins, instruction execution will suspend. $\overline{\text{TEST}}$ will be resampled until it goes LOW, at which time execution will resume. If interrupts are enabled while the 80186 is waiting for $\overline{\text{TEST}}$, interrupts will be serviced. This input is synchronized internally.
TMR IN 0, TMR IN1	20 21	I I	Timer Inputs are used either as clock or control signals, depending upon the programmed timer mode. These inputs are active HIGH (or LOW-to-HIGH transitions are counted) and internally synchronized.
TMR OUT 0, TMR OUT 1	22 23	O O	Timer outputs are used to provide single pulse or continuous waveform generation, depending upon the timer mode selected.
DRQ0 DRQ1	18 19	I I	DMA Request is driven HIGH by an external device when it desires that a DMA channel (Channel 0 or 1) perform a transfer. These signals are active HIGH, level-triggered, and internally synchronized.
NMI	46	I	Non-Maskable Interrupt is an edge-triggered input which causes a type 2 interrupt. NMI is not maskable internally. A transition from a LOW to HIGH initiates the interrupt at the next instruction boundary. NMI is latched internally. An NMI duration of one clock or more will guarantee service. This input is internally synchronized.
INT0, INT1, INT2/$\overline{\text{INTA0}}$ INT3/$\overline{\text{INTA1}}$	45,44 42 41	I I/O I/O	Maskable Interrupt Requests can be requested by strobing one of these pins. When configured as inputs, these pins are active HIGH. Interrupt Requests are synchronized internally. INT2 and INT3 may be configured via software to provide active-LOW interrupt-acknowledge output signals. All interrupt inputs may be configured via software to be either edge- or level-triggered. To ensure recognition, all interrupt requests must remain active until the interrupt is acknowleged. When iRMX mode is selected, the function of these pins changes (see Interrupt Controller section of this data sheet).
A19/S6, A18/S5, A17/S4, A16/S3	65–68	O O O O	Address Bus Outputs (16–19) and Bus Cycle Status (3–6) reflect the four most significant address bits during T_1. These signals are active HIGH. During T_2, T_3, T_W, and T_4, status information is available on these lines as encoded below:

	Low	High
S6	Processor Cycle	DMA Cycle

S3,S4, and S5 are defined as LOW during T_2–T_4.

Symbol	Pin No.	Type	Name and Function
AD15–AD0	10–17, 1–8	I/O	Address/Data Bus (0–15) signals constitute the time mutiplexed memory or I/O address (T_1) and data (T_2, T_3, T_W, and T_4) bus. The bus is active HIGH. A_0 is analogous to $\overline{\text{BHE}}$ for the lower byte of the data bus, pins D_7 through D_0. It is LOW during T_1 when a byte is to be transferred onto the lower portion of the bus in memory or I/O operations.
$\overline{\text{BHE}}$/S7	64	O	During T_1 the Bus High Enable signal should be used to determine if data is to be enabled onto the most significant half of the data bus, pins D_{15}–D_8. $\overline{\text{BHE}}$ is LOW during T_1 for read, write, and interrupt acknowledge cycles when a byte is to be transferred on the higher half of the bus. The S_7 status information is available during T_2, T_3, and T_4. S_7 is logically equivalent to $\overline{\text{BHE}}$. The signal is active LOW, and is tristated OFF during bus HOLD.

$\overline{\text{BHE}}$ and A0 Encodings

$\overline{\text{BHE}}$ Value	A0 Value	Function
0	0	Word Transfer
0	1	Byte Transfer on upper half of data bus (D15–D8)
1	0	Byte Transfer on lower half of data bus (D_7–D_0)
1	1	Reserved

Table 1. 80186 Pin Description (Continued)

Symbol	Pin No.	Type	Name and Function
ALE/QS0	61	O	Address Latch Enable/Queue Status 0 is provided by the 80186 to latch the address into the 8282/8283 address latches. ALE is active HIGH. Addresses are guaranteed to be valid on the trailing edge of ALE. The ALE rising edge is generated off the rising edge of the CLKOUT immediately preceding T_1 of the associated bus cycle, effectively one-half clock cycle earlier than in the standard 8086. The trailing edge is generated off the CLKOUT rising edge in T_1 as in the 8086. Note that ALE is never floated.
$\overline{\text{WR}}$/QS1	63	O	Write Strobe/Queue Status 1 indicates that the data on the bus is to be written into a memory or an I/O device. $\overline{\text{WR}}$ is active for T_2, T_3, and T_W of any write cycle. It is active LOW, and floats during "HOLD." It is driven HIGH for one clock during Reset, and then floated. When the 80186 is in queue status mode, the ALE/QS0 and $\overline{\text{WR}}$/QS1 pins provide information about processor/instruction queue interaction.

QS1	QS0	Queue Operation
0	0	No queue operation
0	1	First opcode byte fetched from the queue
1	1	Subsequent byte fetched from the queue
1	0	Empty the queue

Symbol	Pin No.	Type	Name and Function
$\overline{\text{RD}}$/$\overline{\text{QSMD}}$	62	O	Read Strobe indicates that the 80186 is performing a memory or I/O read cycle. $\overline{\text{RD}}$ is active LOW for T_2, T_3, and T_W of any read cycle. It is guaranteed not to go LOW in T_2 until after the Address Bus is floated. $\overline{\text{RD}}$ is active LOW, and floats during "HOLD." $\overline{\text{RD}}$ is driven HIGH for one clock during Reset, and then the output driver is floated. A weak internal pull-up mechanism on the $\overline{\text{RD}}$ line hols it HIGH when the line is not driven. During RESET the pin is sampled to determine whether the 80186 should provide ALE, $\overline{\text{WR}}$, and $\overline{\text{RD}}$, or if the Queue-Status should be provided. $\overline{\text{RD}}$ should be connected to GND to provide Queue-Status data.
ARDY	55	I	Asynchronous Ready informs the 80186 that the addressed memory space or I/O device will complete a data transfer. The ARDY input pin will accept an asynchronous input, and is active HIGH. Only the rising edge is internally synchronized by the 80186. This means that the falling edge of ARDY must be synchronized to the 80186 clock. If connected to V_{CC}, no WAIT states are inserted. Asynchronous ready (ARDY) or synchronous ready (SRDY) must be active to terminate a bus cycle.
SRDY	49	I	Synchronous Ready must be synchronized externally to the 80186. The use of SRDY provides a relaxed system-timing specification on the Ready input. This is accomplished by eliminating the one-half clock cycle which is required for internally resolving the signal level when using the ARDY input. This line is active HIGH. If this line is connected to V_{CC}, no WAIT states are inserted. Asynchronous ready (ARDY) or synchronous ready (SRDY) must be active before a bus cycle is terminated. If unused, this line should be tied LOW.
$\overline{\text{LOCK}}$	48	O	$\overline{\text{LOCK}}$ output indicates that other system bus masters are not to gain control of the system bus while $\overline{\text{LOCK}}$ is active LOW. The $\overline{\text{LOCK}}$ signal is requested by the LOCK prefix instruction and is activated at the beginning of the first data cycle associated with the instruction following the LOCK prefix. It remains active until the completion of the instruction following the LOCK prefix. No prefetches will occur while $\overline{\text{LOCK}}$ is asserted. $\overline{\text{LOCK}}$ is active LOW, is driven HIGH for one clock during RESET, and then floated. If unused, this line should be tied LOW.

Table 1. 80186 Pin Description (Continued)

Symbol	Pin No.	Type	Name and Function
$\overline{S0}$,$\overline{S1}$,$\overline{S2}$	52–54	O	Bus cycle status $\overline{S0}$–$\overline{S2}$ are encoded to provide bus-transaction information:

80186 Bus Cycle Status Information

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	Bus Cycle Initiated
0	0	0	Interrupt Acknowledge
0	0	1	Read I/O
0	1	0	Write I/O
0	1	1	Halt
1	0	0	Instruction Fetch
1	0	1	Read Data from Memory
1	1	0	Write Data to Memory
1	1	1	Passive (no bus cycle)

Symbol	Pin No.	Type	Name and Function
			The status pins float during "HOLD." $\overline{S2}$ may be used as a logical M/$\overline{IO}$ indicator, and $\overline{S1}$ as a DT/$\overline{R}$ indicator. The status lines are driven HIGH for one clock during Reset, and then floated until a bus cycle begins.
HOLD (input) HLDA (output)	50 51	I O	HOLD indicates that another bus master is requesting the local bus. The HOLD input is active HIGH. HOLD may be asynchronous with respect to the 80186 clock. The 80186 will issue a HLDA in response to a HOLD request at the end of T_4 or T_i. Simultaneous with the issuance of HLDA, the 80186 will float the local bus and control lines. After HOLD is detected as being LOW, the 80186 will lower HLDA. When the 80186 needs to run another bus cycle, it will again drive the local bus and control lines.
$\overline{UCS}$	34	O	Upper Memory Chip Select is an active LOW output whenever a memory reference is made to the defined upper portion (1K–256K block) of memory. This line is not floated during bus HOLD. The address range activating $\overline{UCS}$ is software programmable.
$\overline{LCS}$	33	O	Lower Memory Chip Select is active LOW whenever a memory reference is made to the defined lower portion (1K–256K) of memory. This line is not floated during bus HOLD. The address range activating $\overline{LCS}$ is software programmable.
$\overline{MCS0}$–3	38,37,36,35	O	Mid-Range Memory Chip Select signals are active LOW when a memory reference is made to the defined mid-range portion of memory (8K–512K). These lines are not floated during bus HOLD. The address ranges activating $\overline{MCS0}$–3 are software programmable.
$\overline{PCS0}$–4	25,27–30	O	Peripheral Chip Select signals 0–4 are active LOW when a reference is made to the defined peripheral area (64K byte I/O space). These lines are not floated during bus HOLD. The address ranges activating $\overline{PCS0}$–4 are software programmable.
$\overline{PCS5}$/A1	31	O	Peripheral Chip Select 5 or Latched A1 may be programmed to provide a sixth peripheral chip select, or to provide an internally latched A1 signal. The address range activating $\overline{PCS5}$ is software programmable. When programmed to provide latched A1, rather than $\overline{PCS5}$, this pin will retain the previously latched value of A1 during a bus HOLD. A1 is active HIGH.
$\overline{PCS6}$/A2	32	O	Peripheral Chip Select 6 or Latched A2 may be programmed to provide a seventh peripheral chip select, or to provide an internally latched A2 signal. The address range activating $\overline{PCS6}$ is software programmable. When programmed to provide latched A2, rather than $\overline{PCS6}$, this pin will retain the previously latched value of A2 during a bus HOLD. A2 is active HIGH.
DT/$\overline{R}$	40	O	Data Transmit/Receive controls the direction of data flow through the external 8286/8287 data bus transceiver. When LOW, data is transferred to the 80186. When HIGH the 80186 places write data on the data bus.
$\overline{DEN}$	39	O	Data Enable is provided as an 8286/8287 data bus transceiver output enable. $\overline{DEN}$ is active LOW during each memory and I/O access. $\overline{DEN}$ is HIGH whenever DT/$\overline{R}$ changes state.

iAPX 286/10 HIGH PERFORMANCE MICROPROCESSOR WITH MEMORY MANAGEMENT AND PROTECTION

- **High Performance 8 and 10 MHz Processor (Up to six times iAPX 86)**
- **Large Address Space:**
  —16 Megabytes Physical
  —1 Gigabyte Virtual per Task
- **Integrated Memory Management, Four-Level Memory Protection and Support for Virtual Memory and Operating Systems**
- **Two iAPX 86 Upward Compatible Operating Modes:**
  —iAPX 86 Real Address Mode
  —Protected Virtual Address Mode
- **Optional Processor Extension:**
  —iAPX 286/20 High Performance 80-bit Numeric Data Processor
- **Complete System Development Support:**
  —Development Software: Assembler, PL/M, Pascal, FORTRAN, and System Utilities
  —In-Circuit-Emulator (ICE™-286)
- **High Bandwidth Bus Interface (8 or 10 Megabyte/Sec)**

The iAPX 286/10 (80286 part number) is an advanced, high-performance microprocessor with specially optimized capabilities for multiple user and multi-tasking systems. The 80286 has built-in memory protection that supports operating system and task isolation as well as program and data privacy within tasks. A 10 MHz iAPX 286/10 provides up to six times greater throughput than the standard 5 MHz iAPX 86/10. The 80286 includes memory management capabilities that map up to 2^{30} bytes (one gigabyte) of virtual address space per task into 2^{24} bytes (16 megabytes) of physical memory.

The iAPX 286 is upward compatible with iAPX 86 and 88 software. Using iAPX 86 real address mode, the 80286 is object code compatible with existing iAPX 86, 88 software. In protected virtual address mode, the 80286 is source code compatible with iAPX 86, 88 software and may require upgrading to use virtual addresses supported by the 80286's integrated memory management and protection mechanism. Both modes operate at full 80286 performance and execute a superset of the iAPX 86 and 88's instructions.

The 80286 provides special operations to support the efficient implementation and execution of operating systems. For example, one instruction can end execution of one task, save its state, switch to a new task, load its state, and start execution of the new task. The 80286 also supports virtual memory systems by providing a segment-not-present exception and restartable instructions.

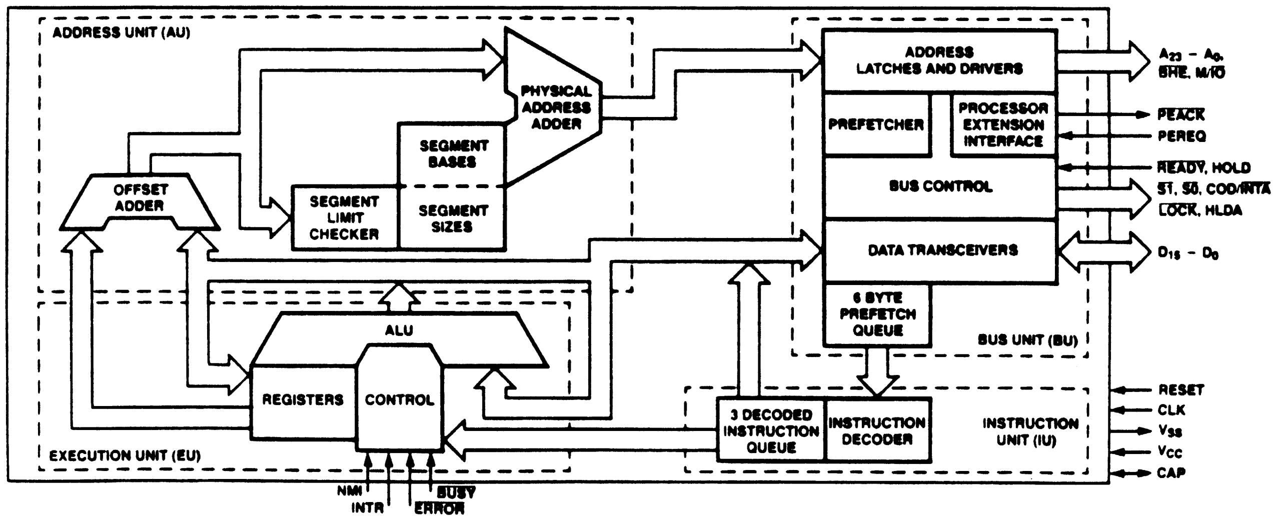
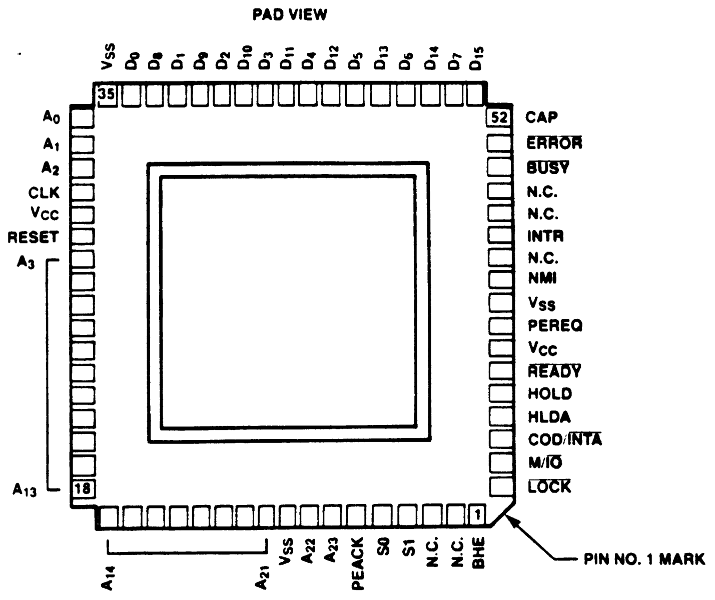


Figure 1. 80286 Internal Block Diagram

JANUARY 1982
ORDER NUMBER: 210253-001



NOTE: N.C. pads must not be connected.

Figure 2. 80286 Pin Configuration

Table 1. Pin Description

The following pin function descriptions are for the 80286 microprocessor:

Symbol	Type	Name and Function
CLK	I	**System Clock** provides the fundamental timing for iAPX 286 systems. It is a 16 MHz signal divided by two inside the 80286 to generate the 8 MHz processor clock. The internal divide-by-two circuitry can be synchronized to an external clock generator by a LOW to HIGH transition on the RESET input.
D_{15}-D_0	I/O	**Data Bus** inputs data during memory, I/O, and interrupt acknowledge read cycles; outputs data during memory and I/O write cycles. The data bus is active HIGH and floats to 3-state OFF during bus hold acknowledge.
A_{23}-A_0	O	**Address Bus** outputs physical memory and I/O port addresses. A0 is LOW when data is to be transferred on pins D_{7-0}. A_{23}-A_{16} are LOW during I/O transfers. The address bus is active HIGH and floats to 3-state OFF during bus hold acknowledge.
$\overline{\text{BHE}}$	O	**Bus High Enable** indicates transfer of data on the upper byte of the data bus, D_{15-8}. Eight-bit oriented devices assigned to the upper byte of the data bus would normally use $\overline{\text{BHE}}$ to condition chip select functions. $\overline{\text{BHE}}$ is active LOW and floats to 3-state OFF during bus hold acknowledge.

$\overline{\text{BHE}}$ and A0 Encodings

$\overline{\text{BHE}}$ Value	A0 Value	Function
0	0	Word transfer
0	1	Byte transfer on upper half of data bus (D_{15-8})
1	0	Byte transfer on lower half of data bus (D_{7-0})
1	1	Reserved

Table 1. Pin Description (Cont.)

Symbol	Type	Name and Function
$\overline{\text{S1}}$, $\overline{\text{S0}}$	O	**Bus Cycle Status** indicates initiation of a bus cycle and, along with M/$\overline{\text{IO}}$ and COD/$\overline{\text{INTA}}$, defines the type of bus cycle. The bus is in a T_S state whenever one or both are LOW. $\overline{\text{S1}}$ and $\overline{\text{S0}}$ are active LOW and float to 3-state OFF during bus hold acknowledge.

80286 Bus Cycle Status Definition

COD/$\overline{\text{INTA}}$	M/$\overline{\text{IO}}$	$\overline{\text{S1}}$	$\overline{\text{S0}}$	**Bus cycle initiated**
0 (LOW)	0	0	0	Interrupt acknowledge
0	0	0	1	Reserved
0	0	1	0	Reserved
0	0	1	1	None; not a status cycle
0	1	0	0	IF A1 = 1 then halt; else shutdown
0	1	0	1	Memory data read
0	1	1	0	Memory data write
0	1	1	1	None; not a status cycle
1 (HIGH)	0	0	0	Reserved
1	0	0	1	I/O read
1	0	1	0	I/O write
1	0	1	1	None; not a status cycle
1	1	0	0	Reserved
1	1	0	1	Memory instruction read
1	1	1	0	Reserved
1	1	1	1	None; not a status cycle

Symbol	Type	Name and Function
M/$\overline{\text{IO}}$	O	**Memory/IO Select** distinguishes memory access from I/O access. If HIGH during T_S, a memory cycle or a halt/shutdown cycle is in progress. If LOW, an I/O cycle or an interrupt acknowledge cycle is in progress. M/$\overline{\text{IO}}$ floats to 3-state OFF during bus hold acknowledge.
COD/$\overline{\text{INTA}}$	O	**Code/Interrupt Acknowledge** distinguishes instruction fetch cycles from memory data read cycles. Also distinguishes interrupt acknowledge cycles from I/O cycles. COD/$\overline{\text{INTA}}$ floats to 3-state OFF during bus hold acknowledge.
$\overline{\text{LOCK}}$	O	**Bus Lock** indicates that other system bus masters are not to gain control of the system bus following the current bus cycle. The $\overline{\text{LOCK}}$ signal may be activated explicitly by the "LOCK" instruction prefix or automatically by 80286 hardware during memory XCHG instructions, interrupt acknowledge, or descriptor table access. $\overline{\text{LOCK}}$ is active LOW and floats to 3-state OFF during bus hold acknowledge.
$\overline{\text{READY}}$	I	**Bus Ready** terminates a bus cycle. Bus cycles are extended without limit until terminated by $\overline{\text{READY}}$ LOW. $\overline{\text{READY}}$ is an active LOW synchronous input requiring setup and hold times relative to the system clock be met for correct operation. $\overline{\text{READY}}$ is ignored during bus hold acknowledge.
HOLD HLDA	I O	**Bus Hold Request and Hold Acknowledge** control ownership of the 80286 local bus. The HOLD input allows another local bus master to request control of the local bus. When control is granted, the 80286 will float its bus drivers to 3-state OFF and then activate HLDA, thus entering the bus hold acknowledge condition. The local bus will remain granted to the requesting master until HOLD becomes inactive which results in the 80286 deactivating HLDA and regaining control of the local bus. This terminates the bus hold acknowledge condition. HOLD may be asynchronous to the system clock. These signals are active HIGH.
INTR	I	**Interrupt Request** requests the 80286 to suspend its current program execution and service a pending external request. Interrupt requests are masked whenever the interrupt enable bit in the flag word is cleared. When the 80286 responds to an interrupt request, it performs two interrupt acknowledge bus cycles to read an 8-bit interrupt vector that identifies the source of the interrupt. To assure program interruption, INTR must remain active until the first interrupt acknowledge cycle is completed. INTR is sampled at the beginning of each processor cycle and must be active HIGH at least two processor cycles before the current instruction ends in order to interrupt before the next instruction. INTR is level sensitive, active HIGH, and may be asynchronous to the system clock.
NMI	I	**Non-maskable Interrupt Request** interrupts the 80286 with an internally supplied vector value of 2. No interrupt acknowledge cycles are performed. The interrupt enable bit in the 80286 flag word does not affect this input. The NMI input is active HIGH, may be asynchronous to the system clock, and is edge triggered after internal synchronization. For proper recognition, the input must have been previously LOW for at least four system clock cycles and remain HIGH for at least four system clock cycles.

Table 1. Pin Description (Cont.)

Symbol	Type	Name and Function
PEREQ $\overline{\text{PEACK}}$	I O	**Processor Extension Operand Request and Acknowledge** extend the memory management and protection capabilities of the 80286 to processor extensions. The PEREQ input requests the 80286 to perform a data operand transfer for a processor extension. The PEACK output signals the processor extension when the requested operand is being transferred. PEREQ is active HIGH and may be asynchronous to the system clock. $\overline{\text{PEACK}}$ is active LOW.
$\overline{\text{BUSY}}$ $\overline{\text{ERROR}}$	I I	**Processor Extension Busy and Error** indicate the operating condition of a processor extension to the 80286. An active $\overline{\text{BUSY}}$ input stops 80286 program execution on WAIT and some ESC instructions until BUSY becomes inactive (HIGH). The 80286 may be interrupted while waiting for BUSY to become inactive. An active ERROR input causes the 80286 to perform a processor extension interrupt when executing WAIT or some ESC instructions. These inputs are active LOW and may be asynchronous to the system clock.
RESET	I	**System Reset** clears the internal logic of the 80286 and is active HIGH. The 80286 may be re-initialized at any time with a LOW to HIGH transition on RESET which remains active for more than 16 system clock cycles. During RESET active, the output pins of the 80286 enter the state shown below:

80286 Pin State During Reset

Pin Value	Pin Names
1 (HIGH) 0 (LOW) 3-state OFF	$\overline{\text{S0}}$, $\overline{\text{S1}}$, $\overline{\text{PEACK}}$, A23-A0, $\overline{\text{BHE}}$, $\overline{\text{LOCK}}$ M/$\overline{\text{IO}}$, COD/$\overline{\text{INTA}}$, HLDA D_{15}–D_0

Symbol	Type	Name and Function
		Operation of the 80286 begins after a HIGH to LOW transition on RESET. The HIGH to LOW transition of RESET must be synchronous to the system clock. Approximately 50 system clock cycles are required by the 80286 for internal initializations before the first bus cycle to fetch code from the power-on execution address is performed. A LOW to HIGH transition of RESET synchronous to the system clock, will begin a new processor cycle at the next HIGH to LOW transition of the system clock. The LOW to HIGH transition of RESET may be asynchronous to the system clock; however, in this case it can not be predetermined which phase of the processor clock will occur during the next system clock period. Synchronous LOW to HIGH transitions of RESET are only required for systems where the processor clock must be phase synchronous to another clock.
V_{SS}	I	**System Ground:** 0 VOLTS.
V_{CC}	I	**System Power:** + 5 Volt Power Supply.
CAP	I	**Substrate Filter Capacitor:** a 0.047µf ± 20% 12V capacitor must be connected between this pin and ground. This capacitor filters the output of the internal substrate bias generator. A maximum DC leakage current of 1 µa is allowed through the capacitor. For correct operation of the 80286, the substrate bias generator must charge this capacitor to its operating voltage. The capacitor chargeup time is 5 milliseconds (max.) after V_{CC} and CLK reach their specified AC and DC parameters. RESET may be applied to prevent spurious activity by the CPU during this time. After this time, the 80286 processor clock can be phase synchronized to another clock by pulsing RESET LOW synchronous to the system clock.

FUNCTIONAL DESCRIPTION

Introduction

The 80286 is an advanced, high-performance microprocessor with specially optimized capabilities for multiple user and multi-tasking systems. Depending on the application, the 80286's performance is up to six times faster than the standard 5 MHz 8086's, while providing complete upward software compatibility with Intel's iAPX 86, 88, and 186 family of CPU's.

The 80286 operates in two modes: iAPX 86 real address mode and protected virtual address mode. Both modes execute a superset of the iAPX 86 and 88 instruction set.

In iAPX 86 real address mode programs use real addresses with up to one megabyte of address space. Programs use virtual addresses in protected virtual address mode, also called protected mode. In protected mode, the 80286 CPU automatically maps 1 gigabyte of virtual addresses per task into a 16 megabyte real address space. This mode also provides memory protection to isolate the operating system and ensure privacy of each tasks' programs and data. Both modes provide the same base instruction set, registers, and addressing modes.

The following Functional Description describes first, the base 80286 architecture common to both modes, second, iAPX 86 real address mode, and third, protected mode.

iAPX 286/10 BASE ARCHITECTURE

The iAPX 86, 88, 186, and 286 CPU family all contain the same basic set of registers, instructions, and addressing modes. The 80286 processor is upward compatible with the 8086, 8088, and 80186 CPU's.

Register Set

The 80286 base architecture has fifteen registers as shown in Figure 3. These registers are grouped into the following four categories:

General Registers: Eight 16-bit general purpose registers used to contain arithmetic and logical operands. Four of these (AX, BX, CX, and DX) can be used either in their entirety as 16-bit words or split into pairs of separate 8-bit registers.

Segment Registers: Four 16-bit special purpose registers select, at any given time, the segments of memory that are immediately addressable for code, stack, and data. (For usage, refer to Memory Organization.)

Base and Index Registers: Four of the general purpose registers may also be used to determine offset addresses of operands in memory. These registers may contain base addresses or indexes to particular locations within a segment. The addressing mode determines the specific registers used for operand address calculations.

Status and Control Registers: Three 16-bit special purpose registers record or control certain aspects of the 80286 processor state. These include the Instruction Pointer, which contains the offset address of the next sequential instruction to be executed.

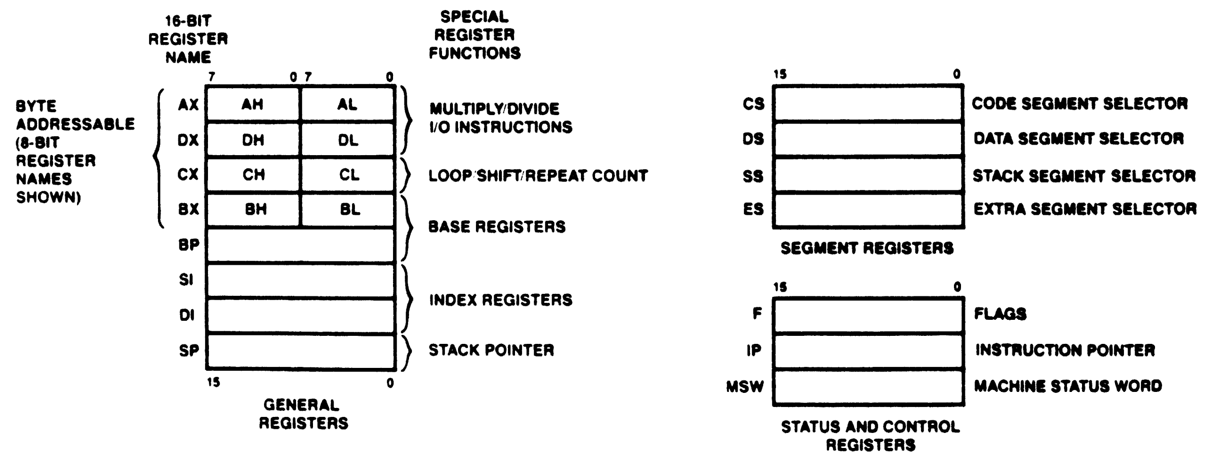


Figure 3. Register Set

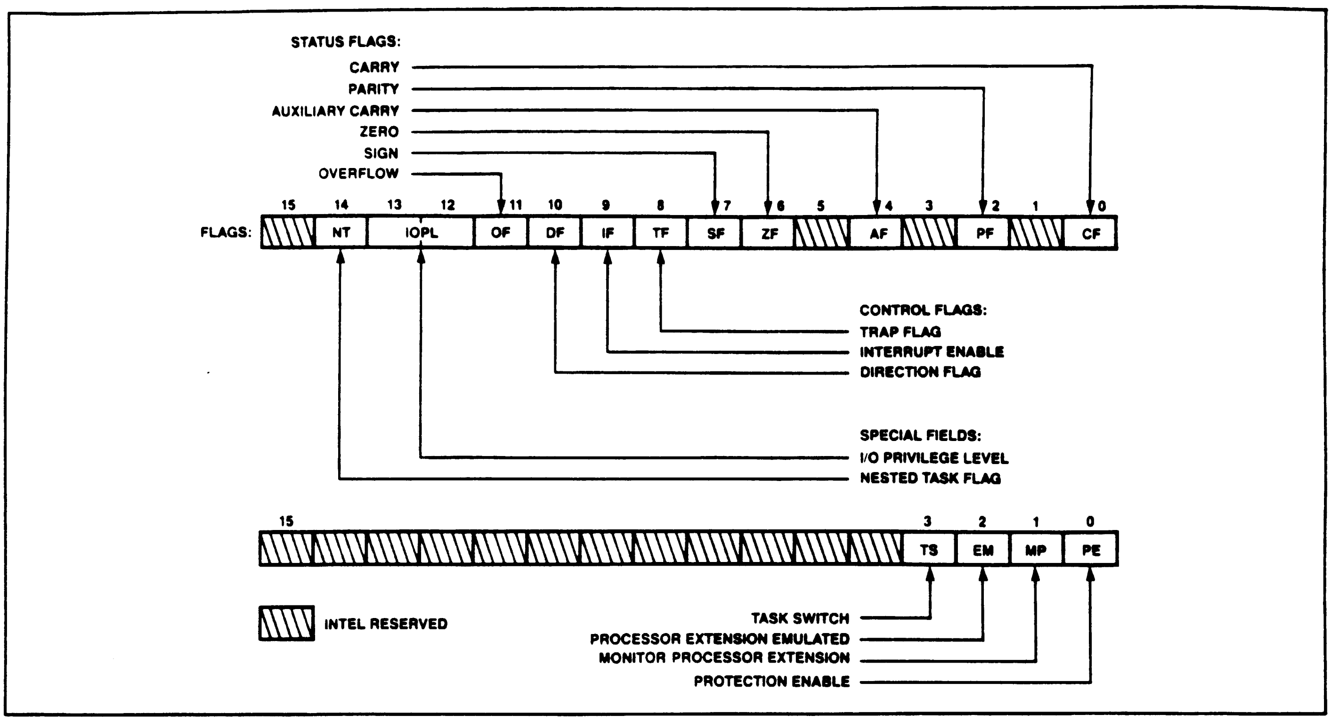


Figure 3a. Status and Control Register Bit Functions

Flags Word Description

The Flags word (Flags) records specific characteristics of the result of logical and arithmetic instructions (bits 0, 2, 4, 6, 7, and 11) and controls the operation of the 80286 within a given operating mode (bits 8 and 9). Flags is a 16-bit register. The function of the flag bits is given in Table 2.

Instruction Set

The instruction set is divided into seven categories: data transfer, arithmetic, shift/rotate/logical, string manipulation, control transfer, high level instructions, and processor control. These categories are summarized in Figure 4.

An 80286 instruction can reference zero, one, or two operands; where an operand resides in a register, in the instruction itself, or in memory. Zero-operand instructions (e.g. NOP and HLT) are usually one byte long. One-operand instructions (e.g. INC and DEC) are usually two bytes long but some are encoded in only one byte. One-operand instructions may reference a register or memory location. Two-operand instructions permit the following six types of instruction operations:

—Register to Register
—Memory to Register
—Immediate to Register
—Memory to Memory
—Register to Memory
—Immediate to Memory

Table 2. Flags Word Bit Functions

Bit Position	Name	Function
0	CF	Carry Flag—Set on high-order bit carry or borrow; cleared otherwise
2	PF	Parity Flag—Set if low-order 8 bits of result contain an even number of 1-bits; cleared otherwise
4	AF	Set on carry from or borrow to the low order four bits of AL; cleared otherwise
6	ZF	Zero Flag–Set if result is zero; cleared otherwise
7	SF	Sign Flag—Set equal to high-order bit of result (0 if positive, 1 if negative)
11	OF	Overflow Flag—Set if result is a too-large positive number or a too-small negative number (excluding sign-bit) to fit in destination operand; cleared otherwise
8	TF	Single Step Flag—Once set, a single step interrupt occurs after the next instruction executes. TF is cleared by the single step interrupt.
9	IF	Interrupt-enable Flag—When set, maskable interrupts will cause the CPU to transfer control to an interrupt vector specified location.
10	DF	Direction Flag—Causes string instructions to auto increment the appropriate index registers when set. Clearing DF causes auto decrement.

Two-operand instructions (e.g. MOV and ADD) are usually three to six bytes long. Memory to memory operations are provided by a special class of string instructions requiring one to three bytes. For detailed instruction formats and encodings refer to the instruction set summary at the end of this document.

GENERAL PURPOSE	
MOV	Move byte or word
PUSH	Push word onto stack
POP	Pop word off stack
PUSHA	Push all registers on stack
POPA	Pop all registers from stack
XCHG	Exchange byte or word
XLAT	Translate byte
INPUT/OUTPUT	
IN	Input byte or word
OUT	Output byte or word
ADDRESS OBJECT	
LEA	Load effective address
LDS	Load pointer using DS
LES	Load pointer using ES
FLAG TRANSFER	
LAHF	Load AH register from flags
SAHF	Store AH register in flags
PUSHF	Push flags onto stack
POPF	Pop flags off stack

Figure 4a. Data Transfer Instructions

ADDITION	
ADD	Add byte or word
ADC	Add byte or word with carry
INC	Increment byte or word by 1
AAA	ASCII adjust for addition
DAA	Decimal adjust for addition
SUBTRACTION	
SUB	Subtract byte or word
SBB	Subtract byte or word with borrow
DEC	Decrement byte or word by 1
NEG	Negate byte or word
CMP	Compare byte or word
AAS	ASCII adjust for subtraction
DAS	Decimal adjust for subtraction
MULTIPLICATION	
MUL	Multiply byte or word unsigned
IMUL	Integer multiply byte or word
AAM	ASCII adjust for multiply
DIVISION	
DIV	Divide byte or word unsigned
IDIV	Integer divide byte or word
AAD	ASCII adjust for division
CBW	Convert byte to word
CWD	Convert word to doubleword

Figure 4b. Arithmetic Instructions

MOVS	Move byte or word string
INS	Input bytes or word string
OUTS	Output bytes or word string
CMPS	Compare byte or word string
SCAS	Scan byte or word string
LODS	Load byte or word string
STOS	Store byte or word string
REP	Repeat
REPE/REPZ	Repeat while equal/zero
REPNE/REPNZ	Repeat while not equal/not zero

Figure 4c. String Instructions

LOGICALS	
NOT	"Not" byte or word
AND	"And" byte or word
OR	"Inclusive or" byte or word
XOR	"Exclusive or" byte or word
TEST	"Test" byte or word
SHIFTS	
SHL/SAL	Shift logical/arithmetic left byte or word
SHR	Shift logical right byte or word
SAR	Shift arithmetic right byte or word
ROTATES	
ROL	Rotate left byte or word
ROR	Rotate right byte or word
RCL	Rotate through carry left byte or word
RCR	Rotate through carry right byte or word

Figure 4d. Shift/Rotate/Logical Instructions

CONDITIONAL TRANSFERS	
JA/JNBE	Jump if above/not below nor equal
JAE/JNB	Jump if above or equal/not below
JB/JNAE	Jump if below/not above nor equal
JBE/JNA	Jump if below or equal/not above
JC	Jump if carry
JE/JZ	Jump if equal/zero
JG/JNLE	Jump if greater/not less nor equal
JGE/JNL	Jump if greater or equal/not less
JL/JNGE	Jump if less/not greater nor equal
JLE/JNG	Jump if less or equal/not greater
JNC	Jump if not carry
JNE/JNZ	Jump if not equal/not zero
JNO	Jump if not overflow
JNP/JPO	Jump if not parity/parity odd
JNS	Jump if not sign
JO	Jump if overflow
JP/JPE	Jump if parity/parity even
JS	Jump if sign

UNCONDITIONAL TRANSFERS	
CALL	Call procedure
RET	Return from procedure
JMP	Jump
ITERATION CONTROLS	
LOOP	Loop
LOOPE/LOOPZ	Loop if equal/zero
LOOPNE/LOOPNZ	Loop if not equal/not zero
JCXZ	Jump if register CX = 0
INTERRUPTS	
INT	Interrupt
INTO	Interrupt if overflow
IRET	Interrupt return

Figure 4e. Program Transfer Instructions

FLAG OPERATIONS	
STC	Set carry flag
CLC	Clear carry flag
CMC	Complement carry flag
STD	Set direction flag
CLD	Clear direction flag
STI	Set interrupt enable flag
CLI	Clear interrupt enable flag
EXTERNAL SYNCHRONIZATION	
HLT	Halt until interrupt or reset
WAIT	Wait for $\overline{\text{TEST}}$ pin active
ESC	Escape to extension processor
LOCK	Lock bus during next instruction
NO OPERATION	
NOP	No operation
EXECUTION ENVIRONMENT CONTROL	
LMSW	Load machine status word
SMSW	Store machine status word

Figure 4f. Processor Control Instructions

ENTER	Format stack for procedure entry
LEAVE	Restore stack for procedure exit
BOUND	Detects values outside prescribed range

Figure 4g. High Level Instructions

Memory Organization

Memory is organized as sets of variable length segments. Each segment is a linear contiguous sequence of up to 64K (2^{16}) 8-bit bytes. Memory is addressed using a two-component address (a pointer) that consists of a 16-bit segment selector, and a 16-bit offset. The segment selector indicates the desired segment in memory. The offset component indicates the desired byte address within the segment.

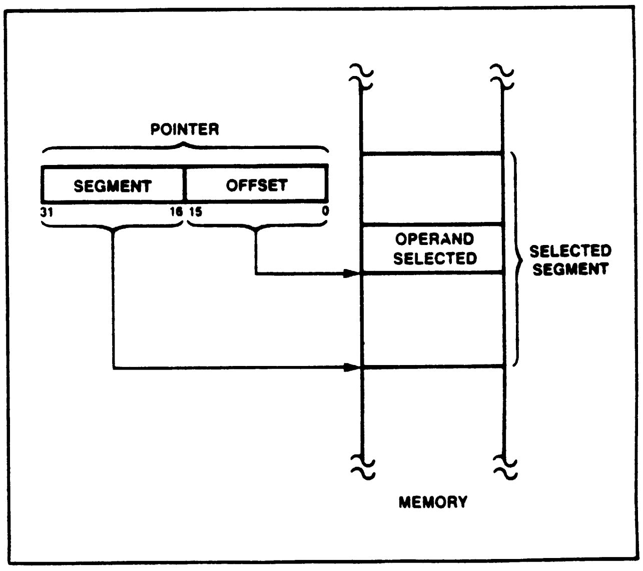


Figure 5. Two Component Address

Table 3. Segment Register Selection Rules

Memory Reference Needed	Segment Register Used	Implicit Segment Selection Rule
Instructions	Code (CS)	Automatic with instruction prefetch
Stack	Stack (SS)	All stack pushes and pops. Any memory reference which uses BP as a base register.
Local Data	Data (DS)	All data references except when relative to stack or string destination
External (Global) Data	Extra (ES)	Alternate data segment and destination of string operation

All instructions that address operands in memory must specify the segment and the offset. For speed and compact instruction encoding, segment selectors are usually stored in the high speed segment registers. An instruction need specify only the desired segment register and an offset in order to address a memory operand.

Most instructions need not explicitly specify which segment register is used. The correct segment register is automatically chosen according to the rules of Table 3. These rules follow the way programs are written (see Figure 6) as independent modules that require areas for code and data, a stack, and access to external data areas.

Special segment override instruction prefixes allow the implicit segment register selection rules to be overridden for special cases. The stack, data, and extra segments may coincide for simple programs. To access operands that do not reside in one of the four immediately available segments, either a full 32-bit pointer can be used or a new segment selector must be loaded.

Addressing Modes

The 80286 provides a total of eight addressing modes for instructions to specify operands. Two addressing modes are provided for instructions that operate on register or immediate operands:

Register Operand Mode: The operand is located in one of the 8 or 16-bit general registers.

Immediate Operand Mode. The operand is included in the instruction.

Six modes are provided to specify the location of an operand in a memory segment. A memory operand address consists of two 16-bit components: segment selector and offset. The segment selector is supplied by a segment register either implicitly chosen by the addressing mode or explicitly chosen by a segment override prefix. The offset is calculated by summing any combination of the following three address elements:

the **displacement** (an 8 or 16-bit immediate value contained in the instruction)

the **base** (contents of either the BX or BP base registers)

the **index** (contents of either the SI or DI index registers)

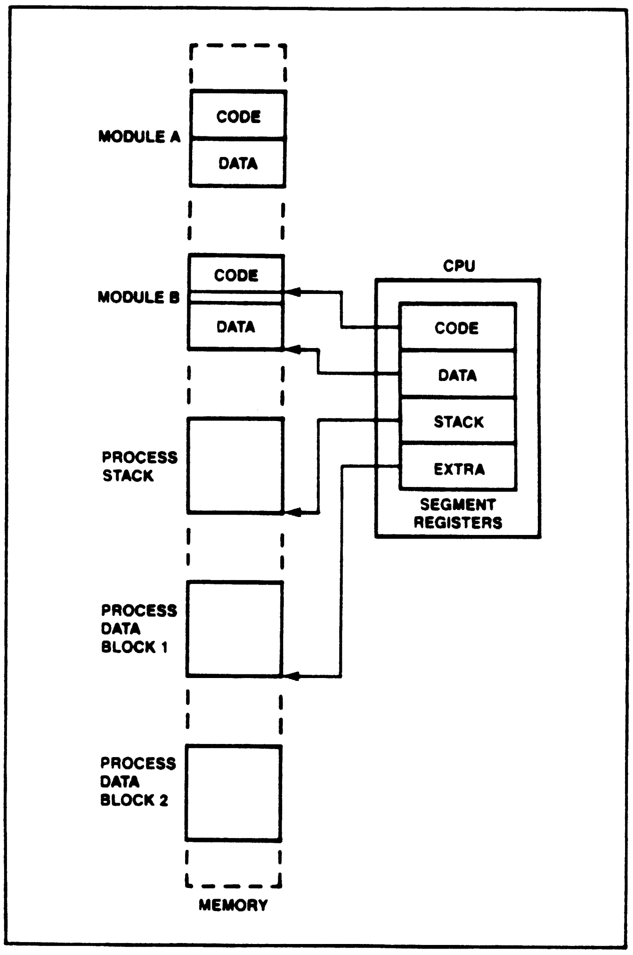


Figure 6. Segmented Memory Helps Structure Software

Any carry out from the 16-bit addition is ignored. Eight-bit displacements are sign extended to 16-bit values.

Combinations of these three address elements define the six memory addressing modes, described below.

Direct Mode: The operand's offset is contained in the instruction as an 8 or 16-bit displacement element.

Register Indirect Mode: The operand's offset is in one of the registers SI, DI, BX, or BP.

Based Mode: The operand's offset is the sum of an 8 or 16-bit displacement and the contents of a base register (BX or BP).

Indexed Mode: The operand's offset is the sum of an 8 or 16-bit displacement and the contents of an index register (SI or DI).

Based Indexed Mode: The operand's offset is the sum of the contents of a base register and an index register.

Based Indexed Mode with Displacement: The operand's offset is the sum of a base register's contents, an index register's contents, and an 8 or 16-bit displacement.

Data Types

The 80286 directly supports the following data types:

Integer:	A signed binary numeric value contained in an 8-bit byte or a 16-bit word. All operations assume a 2's complement representation. Signed 32 and 64-bit integers are supported using the iAPX 286/20 Numeric Data Processor.
Ordinal:	An unsigned binary numeric value contained in an 8-bit byte or 16-bit word.
Pointer:	A 32-bit quantity, composed of a segment selector component and an offset component. Each component is a 16-bit word.
String:	A contiguous sequence of bytes or words. A string may contain from 1 byte to 64K bytes.
ASCII:	A byte representation of alphanumeric and control characters using the ASCII standard of character representation.
BCD:	A byte (unpacked) representation of the decimal digits 0–9.
Packed BCD:	A byte (packed) representation of two decimal digits 0–9 storing one digit in each nibble of the byte.
Floating Point:	A signed 32, 64, or 80-bit real number representation. (Floating point operands are supported using the iAPX 286/20 Numeric Processor configuration.)

Figure 7 graphically represents the data types supported by the iAPX 286.

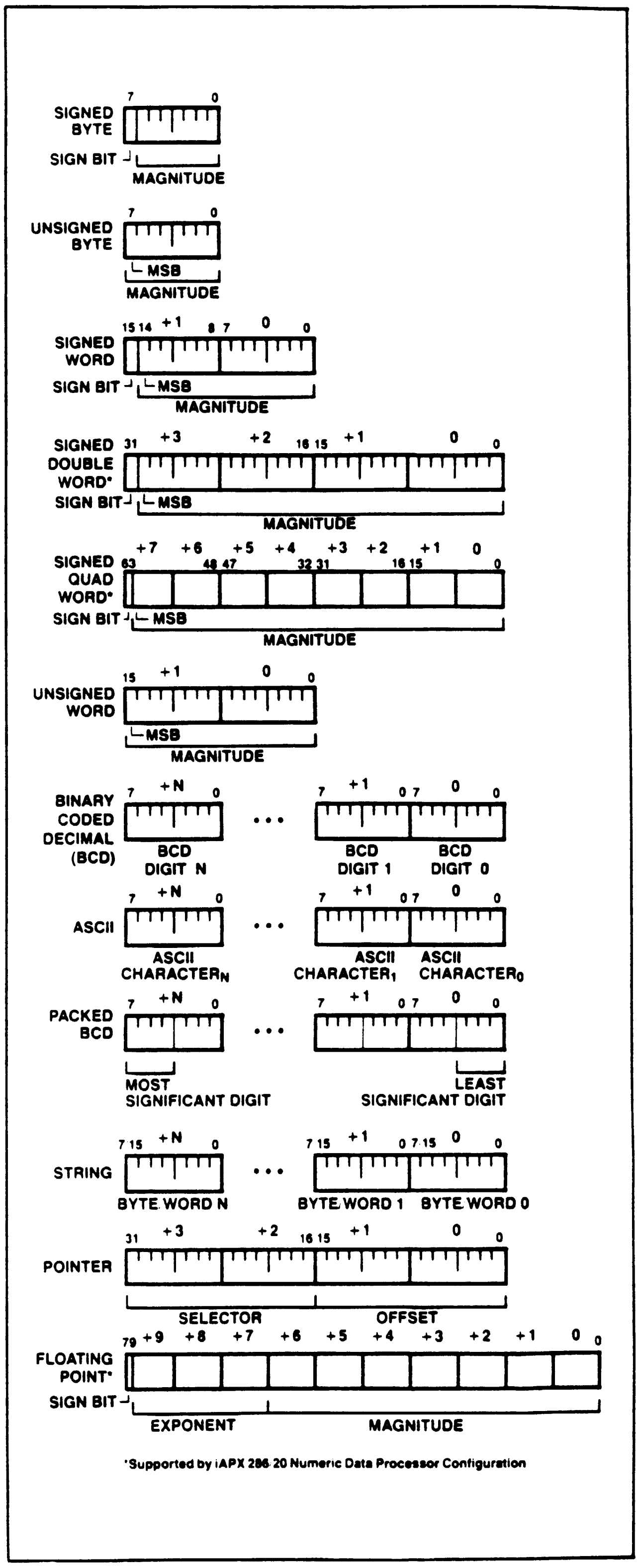


Figure 7. iAPX 286 Supported Data Types

I/O Space

The I/O space consists of 64K 8-bit or 32K 16-bit ports. I/O instructions address the I/O space with either an 8-bit port address, specified in the instruction, or a 16-bit port address in the DX register. 8-bit port addresses are zero extended such that A_{15}–A_8 are LOW. I/O port addresses 00F8(H) through 00FF(H) are reserved.

Table 4. Interrupt Vector Assignments

Function	Interrupt Number	Related Instructions	Return Address Before Instruction Causing Exception?
Divide error exception	0	DIV, IDIV	Yes
Single step interrupt	1	All	
NMI interrupt	2	All	
Breakpoint interrupt	3	INT	
INTO detected overflow exception	4	INTO	No
BOUND range exceeded exception	5	BOUND	Yes
Invalid opcode exception	6	Any undefined opcode	Yes
Processor extension not available exception	7	ESC or WAIT with	Yes
Reserved	8–15		
Processor extension error interrupt	16	ESC or WAIT	
Reserved	17–31		
User defined	32–255		

Interrupts

An interrupt transfers execution to a new program location. The old program address (CS:IP) and machine state (Flags) are saved on the stack to allow resumption of the interrupted program. Interrupts fall into three classes: hardware initiated, INT instructions, and instruction exceptions. Hardware initiated interrupts occur in response to an external input and are classified as non-maskable or maskable. Programs may cause an interrupt with an INT instruction. Instruction exceptions occur when an unusual condition, which prevents further instruction processing, is detected while attempting to execute an instruction. The return address from an exception will always point at the instruction causing the exception and include any leading instruction prefixes.

A table containing up to 256 pointers defines the proper interrupt service routine for each interrupt. Interrupts 0–31, some of which are used for instruction exceptions, are reserved. For each interrupt, an 8-bit vector must be supplied to the 80286 which identifies the appropriate table entry. Exceptions supply the interrupt vector internally. INT instructions contain or imply the vector and allow access to all 256 interrupts. Maskable hardware initiated interrupts supply the 8-bit vector to the CPU during an interrupt acknowledge bus sequence. Non-maskable hardware interrupts use a predefined internally supplied vector.

MASKABLE INTERRUPT (INTR)

The 80286 provides a maskable hardware interrupt request pin, INTR. Software enables this input by setting the interrupt flag bit (IF) in the flag word. All 224 user-defined interrupt sources can share this input, yet they can retain separate interrupt handlers. An 8-bit vector read by the CPU during the interrupt acknowledge sequence (discussed in System Interface section) identifies the source of the interrupt.

Further maskable interrupts are disabled while servicing an interrupt by resetting the IF but as part of the response to an interrupt or exception. The saved flag word will reflect the enable status of the processor prior to the interrupt. Until the flag word is restored to the flag register, the interrupt flag will be zero unless specifically set. The interrupt return instruction includes restoring the flag word, thereby restoring the original status of IF.

NON-MASKABLE INTERRUPT REQUEST (NMI)

A non-maskable interrupt input (NMI) is also provided. NMI has higher priority than INTR. A typical use of NMI would be to activate a power failure routine. The activation of this input causes an interrupt with an internally supplied vector value of 2. No external interrupt acknowledge sequence is performed.

While executing the NMI servicing procedure, the 80286 will service neither further NMI requests, INTR requests, nor the processor extension segment overrun interrupt until an interrupt return (IRET) instruction is executed or the CPU is reset. If NMI occurs while currently servicing an NMI, its presence will be saved for servicing after executing the first IRET instruction. IF is cleared at the beginning of an NMI interrupt to inhibit INTR interrupts.

SINGLE STEP INTERRUPT

The 80286 has an internal interrupt that allows programs to execute one instruction at a time. It is called the single step interrupt and is controlled by the single step flag bit (TF) in the flag word. Once this bit is set, an internal single step interrupt will occur after the next instruction has been executed. The interrupt clears the TF bit and uses an internally supplied vector of 1. The IRET instruction is used to set the TF bit and transfer control to the next instruction to be single stepped.

Interrupt Priorities

When simultaneous interrupt requests occur, they are processed in a fixed order as shown in Table 5. Interrupt processing involves saving the flags, return address, and setting CS:IP to point at the first instruction of the interrupt handler. If other interrupts remain enabled they are processed before the first instruction of the current interrupt handler is executed. The last interrupt processed is therefore the first one serviced.

Table 5. Interrupt Processing Order

Order	Interrupt
1	INT instruction or exception
2	Single step
3	NMI
4	Processor extension segment overrun
5	INTR

Initialization and Processor Reset

Processor initialization or start up is accomplished by driving the RESET input pin HIGH. RESET forces the 80286 to terminate all execution and local bus activity. No instruction or bus activity will occur as long as RESET is active. After RESET becomes inactive and an internal processing interval elapses, the 80286 begins execution in real address mode with the instruction at physical location FFFFF0(H). RESET also sets some registers to predefined values as shown as shown in Table 6.

Table 6. 80286 Initial Register State after RESET

Flag word	0002(H)
Machine Status Word	FFF0(H)
Instruction pointer	FFF0(H)
Code segment	F000(H)
Data segment	0000(H)
Extra segment	0000(H)
Stack segment	0000(H)

Machine Status Word Description

The machine status word (MSW) records when a task switch takes place and controls the operating mode of the 80286. It is a 16-bit register of which the lower four bits are used. One bit places the CPU into protected mode, while the other three bits, as shown in Table 7, control the processor extension interface. After RESET, this register contains FFF0(H) which places the 80286 in iAPX 86 real address mode.

Table 7. MSW Bit Functions

Bit Position	Name	Function
0	PE	Protected mode enable places the 80286 into protected mode and can not be cleared except by RESET.
1	MP	Monitor processor extension allows WAIT instructions to cause a processor extension not present exception (number 7).
2	EM	Emulate processor extension causes a processor extension not present exception (number 7) on ESC instructions to allow emulating a processor extension.
3	TS	Task switched indicates the next instruction using a processor extension will cause exception 7, allowing software to test whether the current processor extension context belongs to the current task.

The LMSW and SMSW instructions can load and store the MSW in real address mode. The recommended use of TS, EM, and MP is shown in Table 8.

Table 8. Recommended MSW Encodings For Processor Extension Control

TS	MP	EM	Recommended Use	Instructions Causing Exception
0	0	0	iAPX 86 real address mode only. Initial encoding after RESET. iAPX 286 operation is identical to iAPX 86, 88.	None
0	0	1	No processor extension is available. Software will emulate its function.	ESC
1	0	1	No processor extension is available. Software will emulate its function. The current processor extension context may belong to another task.	ESC
0	1	0	A processor extension exists.	None
1	1	0	A processor extension exists. The current processor extension context may belong to another task. The exception on WAIT allows software to test for an error pending from a previous processor extension operation.	ESC or WAIT

Halt

The HLT instruction stops program execution and prevents the CPU from using the local bus until restarted. Either NMI, INTR with IF = 1, or RESET will force the 80286 out of halt. If interrupted, the saved CS:IP will point to the next instruction after the HLT.

iAPX 86 REAL ADDRESS MODE

The 80286 executes a fully upward-compatible superset of the 8086 instruction set in real address mode. In real address mode the 80286 is object code compatible with 8086 and 8088 software. The real address mode architecture (registers and addressing modes) is exactly as described in the iAPX 286/10 Base Architecture section of this Functional Description.

Memory Size

Physical memory is a contiguous array of up to 1,048,576 bytes (one megabyte) addressed by pins A_0 through A_{19} and $\overline{BHE}$. A_{20} through A_{23} are ignored.

Memory Addressing

In real address mode the processor generates 20-bit physical addresses directly from a 20-bit segment base address and a 16-bit offset.

The selector portion of a pointer is interpreted as the upper 16 bits of a 20-bit segment address. The lower four bits of the 20-bit segment address are always zero. Segment addresses, therefore, begin on multiples of 16 bytes. See Figure 8 for a graphic representation of address formation.

All segments in real address mode are 64K bytes in size and may be read, written, or executed. An exception or interrupt can occur if data operands or instructions attempt to wrap around the end of a segment (e.g. a word with its low order byte at offset FFFF(H) and its high order byte at offset 0000(H)). If, in real address mode, the information contained in a segment does not use the full 64K bytes, the unused end of the segment may be overlayed by another segment to reduce physical memory requirements.

Reserved Memory Locations

The 80286 reserves two fixed areas of memory in real address mode (see Figure 9); system initialization area and interrupt table area. Locations from addresses FFFF0(H) thorugh FFFFF(H) are reserved for system initialization. Initial execution begins at location FFFF0(H). Locations 00000(H) through 003FF(H) are reserved for interrupt vectors.

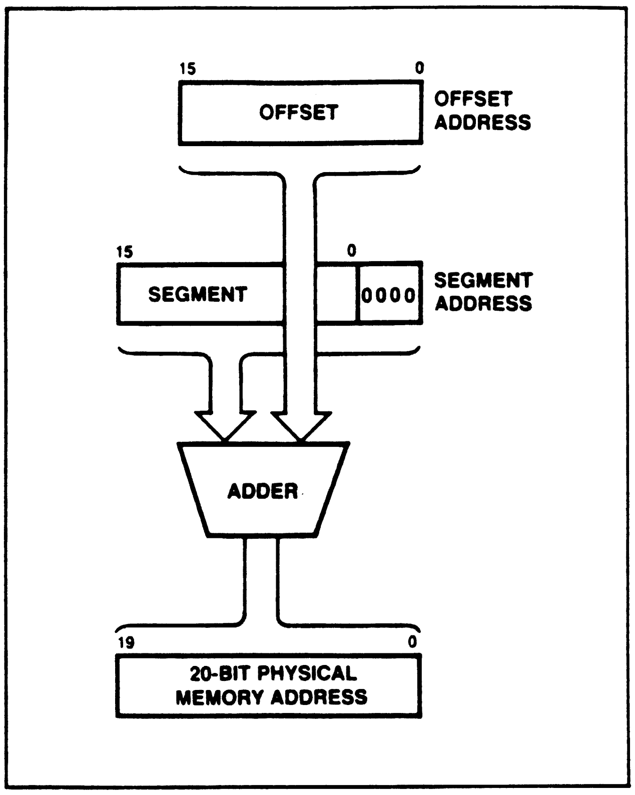


Figure 8. iAPX 86 Real Address Mode Address Calculation

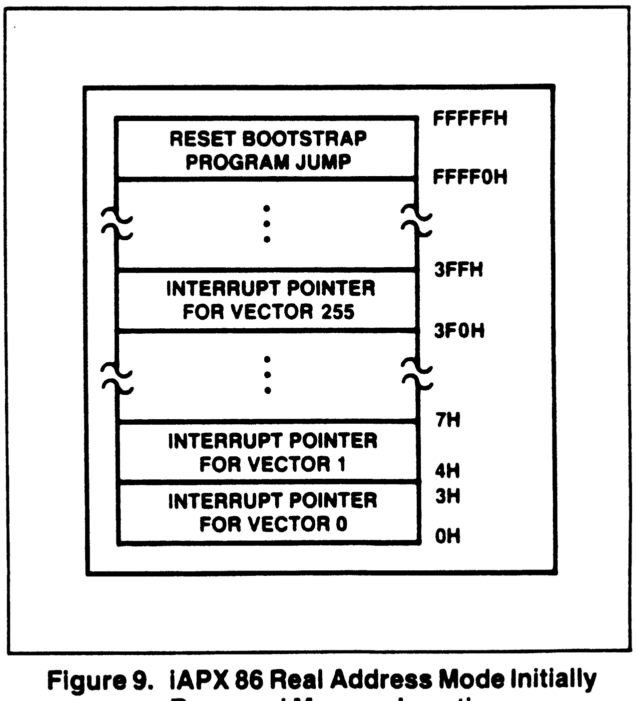

RESET BOOTSTRAP PROGRAM JUMP
FFFFFH
FFFF0H
3FFH
INTERRUPT POINTER FOR VECTOR 255
3F0H
7H
INTERRUPT POINTER FOR VECTOR 1
4H
3H
INTERRUPT POINTER FOR VECTOR 0
0H

Figure 9. iAPX 86 Real Address Mode Initially Reserved Memory Locations

Table 9. Real Address Mode Addressing Interrupts

Function	Interrupt Number	Related Instructions	Return Address Before Instruction?
Interrupt table limit too small exception	8	INT vector is not within table limit	Yes
Processor extension segment overrun interrupt	9	ESC with memory operand extending beyond offset FFFF(H)	No
Segment overrun exception	13	Word memory reference with offset = FFFF(H) or an attempt to execute past the end of a segment	Yes

Interrupts

Table 9 shows the interrupt vectors reserved for exceptions and interrupts which indicate an addressing error. The exceptions leave the CPU in the state existing before attempting to execute the failing instruction (except for PUSH, POP, PUSHA, or POPA). Refer to the next section on protected mode initialization for a discussion on exception 8.

Protected Mode Initialization

To prepare the 80286 for protected mode, the LIDT instruction is used to load the 24-bit interrupt table base and 16-bit limit for the protected mode interrupt table. This instruction can also set a base and limit for the interrupt vector table in real address mode. After reset, the interrupt table base is initialized to 000000(H) and its size set to 03FF(H). These values are compatible with iAPX 86, 88 software. LIDT should only be executed in preparation for protected mode.

Shutdown

Shutdown occurs when a severe error is detected that prevents further instruction processing by the CPU. Shutdown and halt are externally signalled via a halt bus operation. They can be distinguished by A_1 HIGH for halt and A_1 LOW for shutdown. In real address mode, shutdown can occur under two conditions:

- Exceptions 8 or 13 happen and the IDT limit does not include the interrupt vector.
- A CALL, INT, or POP instruction attempts to wrap around the stack segment when SP is not even.

An NMI input can bring the CPU out of shutdown if the IDT limit is at least 000F(H) and SP is greater than 0005(H), otherwise shutdown can only be exited via the RESET input.

PROTECTED VIRTUAL ADDRESS MODE

The 80286 executes a fully upward-compatible superset of the 8086 instruction set in protected virtual address mode (protected mode). Protected mode also provides memory management and protection mechanisms and associated instructions.

The 80286 enters protected virtual address mode from real address mode by setting the PE (Protection Enable) bit of the machine status word with the Load Machine Status Word (LMSW) instruction. Protected mode offers extended physical and virtual memory address space, memory protection mechanisms, and new operations to support operating systems and virtual memory.

All registers, instructions, and addressing modes described in the iAPX 286/10 Base Architecture section of this Functional Description remain the same. Programs for the iAPX 86, 88, 186, and real address mode 80286 can be run in protected mode; however, embedded constants for segment selectors are different.

Memory Size

The protected mode 80286 provides a 1 gigabyte virtual address space per task mapped into a 16 megabyte physical address space defined by the address pins A_{23}–A_0 and $\overline{BHE}$. The virtual address space may be larger than the physical address space since any use of an address that does not map to a physical memory location will cause a restartable exception.

Memory Addressing

As in real address mode, protected mode uses 32-bit pointers, consisting of 16-bit selector and offset components. The selector, however, specifies an index into a memory resident table rather than the upper 16-bits of a real memory address. The 24-bit base address of the desired segment is obtained from the tables in memory. The 16-bit offset is added to the segment base address to form the physical address as shown in Figure 10. The tables are automatically referenced by the CPU whenever a segment register is loaded with a selector. All iAPX 286 instructions which load a segment register will reference the memory based tables without additional software. The memory based tables contain 8 byte values called descriptors.

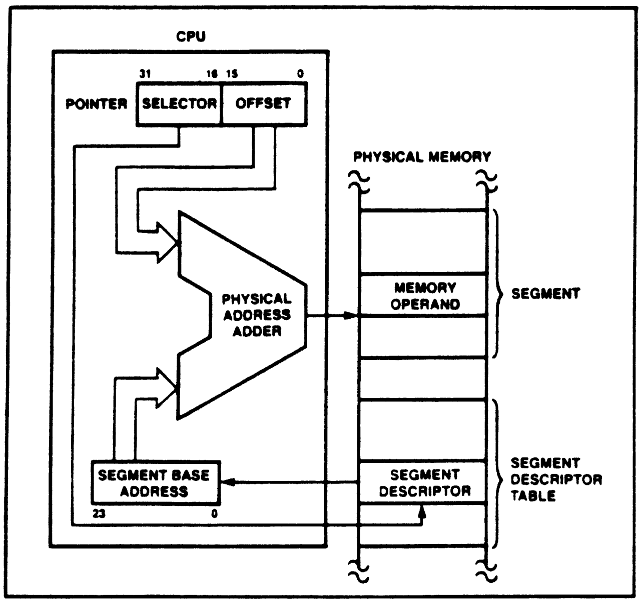


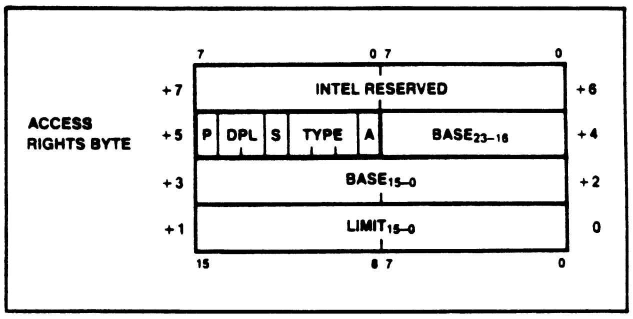
Figure 10. Protected Mode Memory Addressing

DESCRIPTORS

Descriptors define the use of memory. Special types of descriptors also define new functions for transfer of control and task switching. The 80286 has segment descriptors for code, stack and data segments, and system control descriptors for special system data segments and control transfer operations. Descriptor accesses are performed as locked bus operations to assure descriptor integrity in multi-processor systems.

CODE AND DATA SEGMENT DESCRIPTORS

Besides segment base addresses, code and data descriptors contain other segment attributes including segment size (1 to 64K bytes), access rights (read only, read/write, execute only, and execute/read), and presence in memory (for virtual memory systems) (See Figure 11). Any segment usage violating a segment attribute indicated by the segment descriptor will prevent the memory cycle and cause an exception or interrupt.



Access Rights Byte Definition

	Bit Position	Name	Function		
	7	Present (P)	P = 1	Segment is mapped into physical memory.	
			P = 0	No mapping to physical memory exists, base and limit are not used.	
	6–5	Descriptor Privilege Level (DPL)		Segment privilege attribute used in privilege tests.	
	4	Segment Descriptor (S)	S = 1	Code or Data segment descriptor	
			S = 0	Non-segment descriptor	
Type Field Definition	3	Executable (E)	E = 0	Data segment descriptor type is:	Data Segment
	2	Expansion Direction (ED)	ED = 0	Grow up segment, offsets must be ≤ limit.	
			ED = 1	Grow down segment, offsets must be > limit.	
	1	Writeable (W)	W = 0	Data segment may not be written into.	
			W = 1	Data segment may be written into.	
	3	Executable (E)	E = 1	Code Segment Descriptor type is:	Code Segment
	2	Conforming (C)	C = 0	Code segment may only be executed when CPL ≥ DPL.	
	1	Readable (R)	R = 0	Code segment may not be read.	
			R = 1	Code segment may be read.	
	0	Accessed (A)	A = 0	Segment has not been accessed.	
			A = 1	Segment selector has been loaded into segment register or used by selector test instructions.	

Figure 11. Code and Data Segment Descriptors

Code and data are stored in two types of segments: code segments and data segments. Both types are identified and defined by segment descriptors. Code segments are identified by the executable (E) bit set to 1 in the descriptor access rights byte. The access rights byte of both code and data segment descriptor types have three fields in common: present (P) bit, Descriptor Privilege Level (DPL), and accessed (A) bit. If P = 0, any attempted use of this segment will cause a not-present exception. DPL specifies the privilege level of the segment descriptor. DPL effects when the descriptor may be used by a task (refer to privilege discussion below). The A bit shows whether the segment has been previously accessed for usage profiling, a necessity for virtual memory systems. The CPU will always set this bit when accessing the descriptor.

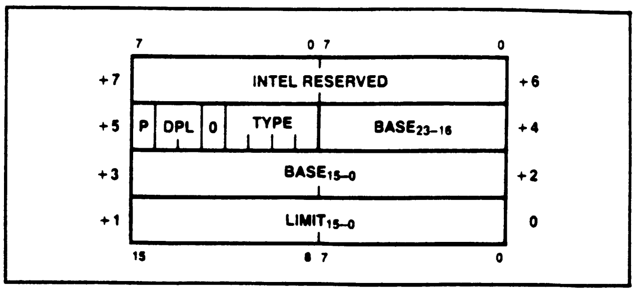
Data segments (S = 1, E = 0) may be either read-only or read-write as controlled by the W bit of the access rights byte. Read-only (W = 0) data segments may not be written into. Data segments may grow in two directions, as determined by the Expansion Direction (ED) bit: upwards (ED = 0) for data segments, and downwards (ED = 1) for a segment containing a stack. The limit field for a data segment descriptor is interpreted differently depending on the ED bit (see Figure 11).

A code segment (S = 1, E = 1) may be execute-only or execute/read as determined by the Readable (R) bit. Code segments may never be written into and execute-only code segments (R = 0) may not be read. A code segment may also have an attribute called conforming (C). A conforming code segment may be shared by programs that execute at different privilege levels. The DPL of a conforming code segment defines the range of privilege levels at which the segment may be executed (refer to privilege discussion below).

SYSTEM CONTROL DESCRIPTORS

In addition to code and data segment descriptors, the protected mode 80286 defines system control descriptors. These descriptors define special system data segments and control transfer mechanisms in the protected environment. The special system data segment descriptors define segments which contain tables of descriptors (Local Descriptor Table Descriptor) and segments which contain the execution state of a task (Task State Segment Descriptor).

The control transfer descriptors are call gates, task gates, interrupt gates and trap gates. Gates provide a level of indirection between the source and destination of the control transfer. This indirection allows the CPU to automatically perform protection checks and control the entry point of the destination. Call gates are used to change privilege levels (see Privilege), task gates are used to perform a task switch, and interrupt and trap gates are used to specify interrupt service routines. The interrupt gate disables interrupts (resets IF) while the trap gate does not.



System Segment Descriptor Fields

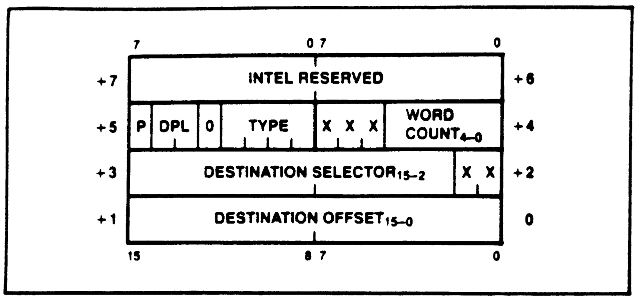
Name	Value	Description
TYPE	1 2 3	Available Task State Segment Local Descriptor Table Descriptor Busy Task State Segment
P	0 1	Descriptor contents are not valid Descriptor contents are valid
DPL	0–3	Descriptor Privilege Level
BASE	24-bit number	Base Address of special system data segment in real memory
LIMIT	16-bit number	Offset of last byte in segment

Figure 12. Special System Data Segment

Figure 12 gives the formats for the special system data segment descriptors. The descriptors contain a 24-bit base address of the segment and a 16-bit limit. The access byte defines the type of descriptor, its state and privilege level. The descriptor contents are valid and the segment is in physical memory if P = 1. If P = 0, the segment is not valid. The DPL field is only used in Task State Segment descriptors and indicates the privilege level at which the descriptor may be used (see Privilege). Since the Local Descriptor Table descriptor may only be used by a special privileged instruction, the DPL field is not used. Bit 4 of the access byte is 0 to indicate that it is a system control descriptor. The type field specifies the descriptor type as indicated in Figure 12.

Figure 13 shows the format of the gate descriptors. The descriptor contains a destination pointer that points to the descriptor of the target segment and the entry point offset. The destination selector in an interrupt gate, trap gate, and call gate must refer to a code segment descriptor. These gate descriptors contain the entry point to prevent a program from constructing and using an illegal entry point. Task gates may only refer to a task state segment. Since task gates invoke a task switch, the destination offset is not used in the task gate.

Exception 13 is generated when the gate is used if a destination selector does not refer to the correct de-



7 0 7 0
+7 INTEL RESERVED +6
+5 P DPL 0 TYPE X X X WORD COUNT4-0 +4
+3 DESTINATION SELECTOR15-2 X X +2
+1 DESTINATION OFFSET15-0 0
15 8 7 0

Gate Descriptor Fields

Name	Value	Description
TYPE	4 5 6 7	–Call Gate –Task Gate –Interrupt Gate –Trap Gate
P	0 1	–Descriptor Contents are not valid –Descriptor Contents are valid
DPL	0–3	Descriptor Privilege Level
WORD COUNT	0–31	Number of words to copy from callers stack to called procedures stack. Only used with call gate.
DESTINATION SELECTOR	16-bit selector	Selector to the target code segment (Call, Interrupt or Trap Gate) Selector to the target task state segment (Task Gate)
DESTINATION OFFSET	16-bit offset	Entry point within the target code segment

Figure 13. Gate Descriptor Format

scriptor type. The word count field is used in the call gate descriptor to indicate the number of parameters (0–31 words) to be automatically copied from the caller's stack to the stack of the called routine when a control transfer changes privilege levels. The word count field is not used by any other gate descriptor.

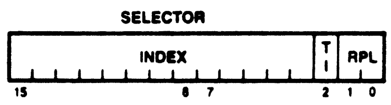
The access byte format is the same for all gate descriptors. P = 1 indicates that the gate contents are valid. P = 0 indicates the contents are not valid and causes exception 11 if referenced. DPL is the descriptor privilege level and specifies when this descriptor may be used by a task (refer to privilege discussion below). Bit 4 must equal 0 to indicate a system control descriptor. The type field specifies the descriptor type as indicated in Figure 13.

SEGMENT DESCRIPTOR CACHE REGISTERS

A segment descriptor cache register is assigned to each of the four segment registers (CS, SS, DS, ES). Segment descriptors are automatically loaded (cached) into a segment descriptor cache register (Figure 14) whenever the associated segment register is loaded with a selector. Only segment descriptors may be loaded into segment descriptor cache registers. Once loaded, all references to that segment of memory use the cached descriptor information instead of reaccessing memory. The descriptor cache registers are not visible to programs. No instructions exist to store their contents. They only change when a segment register is loaded.

SELECTOR FIELDS

A protected mode selector has three fields: descriptor entry index, local or global descriptor table indicator (TI), and selector privilege (RPL) as shown in Figure 15. These fields select one of two memory based tables of descriptors, select the appropriate table entry and allow high-speed testing of the selector's privilege attribute (refer to privilege discussion below).



BITS	NAME	FUNCTION
1–0	REQUESTED PRIVILEGE LEVEL (RPL)	INDICATES SELECTOR PRIVILEGE LEVEL DESIRED

2	TABLE INDICATOR (TI)	TI = 0 USE GLOBAL DESCRIPTOR TABLE (GDT)
		TI = 1 USE LOCAL DESCRIPTOR TABLE (LDT)

15–3	INDEX	SELECT DESCRIPTOR ENTRY IN TABLE

Figure 15. Selector Fields

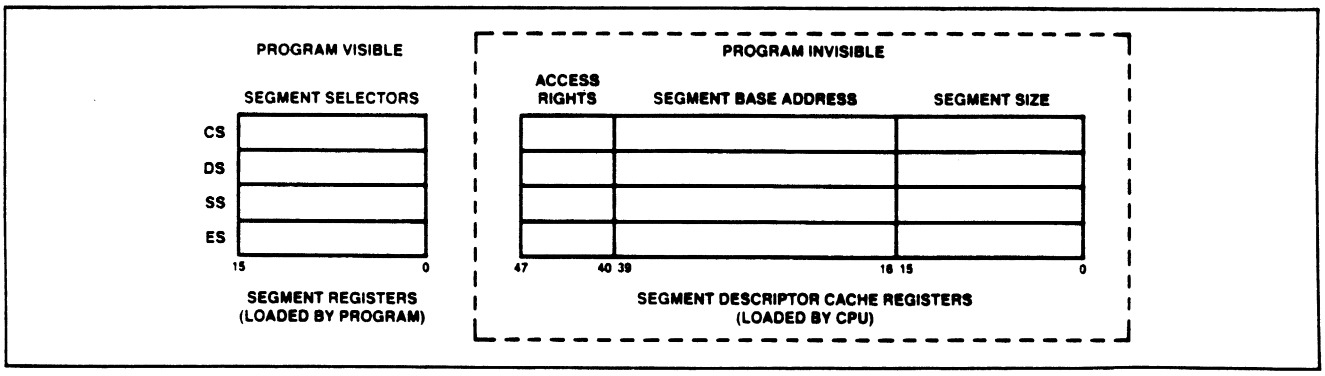


Figure 14. Descriptor Cache Registers

LOCAL AND GLOBAL DESCRIPTOR TABLES

Two tables of descriptors, called descriptor tables, contain all descriptors accessible by a task at any given time. A descriptor table is a linear array of up to 8192 descriptors. The upper 13 bits of the selector value are an index into a descriptor table. Each table has a 24-bit base register to locate the descriptor table in physical memory and a 16-bit limit register that confine descriptor access to the defined limits of the table as shown in Figure 16. A restartable exception (13) will occur if an attempt is made to reference a descriptor outside the table limits.

One table, called the Global Descriptor Table (GDT), contains descriptors available to all tasks. The other table, called the Local Descriptor Table (LDT), contains descriptors that can be private to a task. Each task may have its own private LDT. The GDT may contain all descriptor types except interrupt and trap descriptors. The LDT may contain only segment, task gate, and call gate descriptors. A segment cannot be accessed by a task if its segment descriptor does not exist in either descriptor table at the time of access.

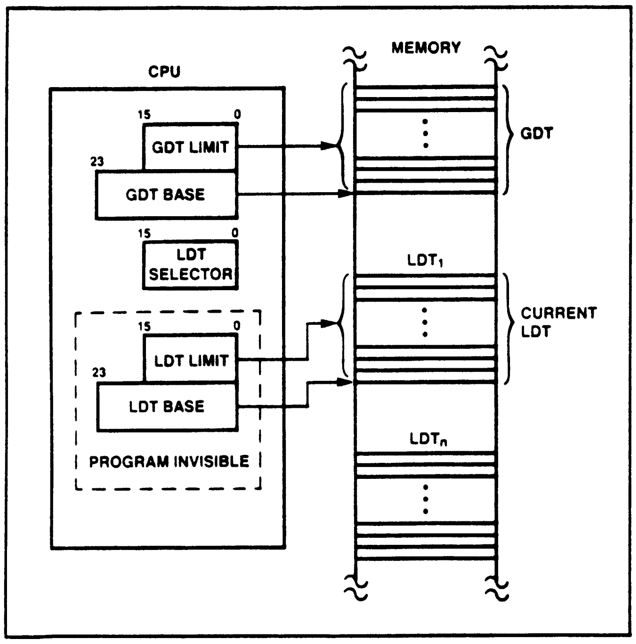


Figure 16. Local and Global Descriptor Table Definition

The LGDT and LLDT instructions load the base and limit of the global and local descriptor tables. LGDT and LLDT are protected. They may only be executed by trusted programs operating at level 0. The LGDT instruction loads a six byte field containing the 16-bit table limit and 24-bit base address of the Global Descriptor Table as shown in Figure 17. The LLDT instruction loads a selector which refers to a Local Descriptor Table descriptor containing the base address and limit for an LDT, as shown in Figure 12.

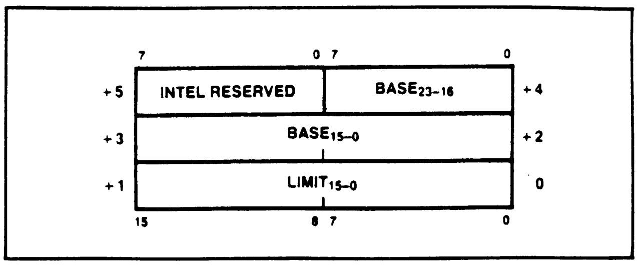


Figure 17. Global Descriptor Table and Interrupt Descriptor Data Type

INTERRUPT DESCRIPTOR TABLE

The protected mode 80286 has a third descriptor table, called the Interrupt Descriptor Table (IDT) (see Figure 18), used to define up to 256 interrupts. It may contain only task gates, interrupt gates and trap gates. The IDT (Interrupt Descriptor Table) has a 24-bit base and 16-bit limit register in the CPU. The protected LIDT instruction loads these registers with a six byte value of identical form to that of the LGDT instruction (see Figure 17 and Protected Mode Initialization).

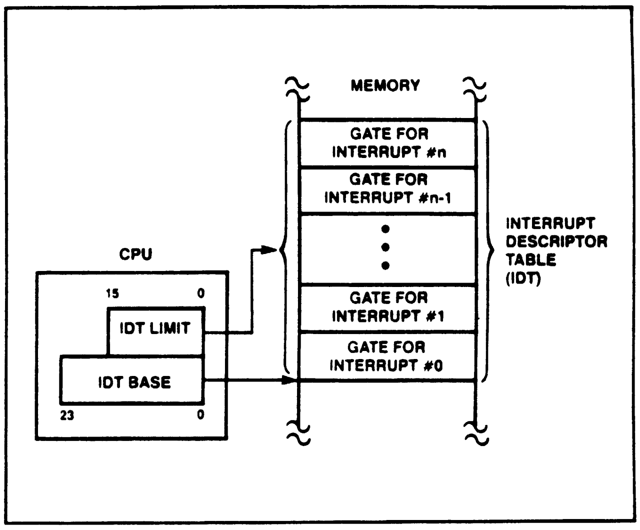


Figure 18. Interrupt Descriptor Table Definition

References to IDT entries are made via INT instructions, external interrupt vectors, or exceptions. The IDT must be at least 256 bytes in size to allocate space for all reserved interrupts.

Privilege

The 80286 has a four-level hierarchical privilege system which controls the use of privileged instructions and access to descriptors (and their associated segments) within a task. Four-level privilege, as shown in Figure 19, is an extension of the user/supervisor mode commonly found in minicomputers. The privilege levels are numbered 0 through 3. Level 0 is the most privileged level. Privilege

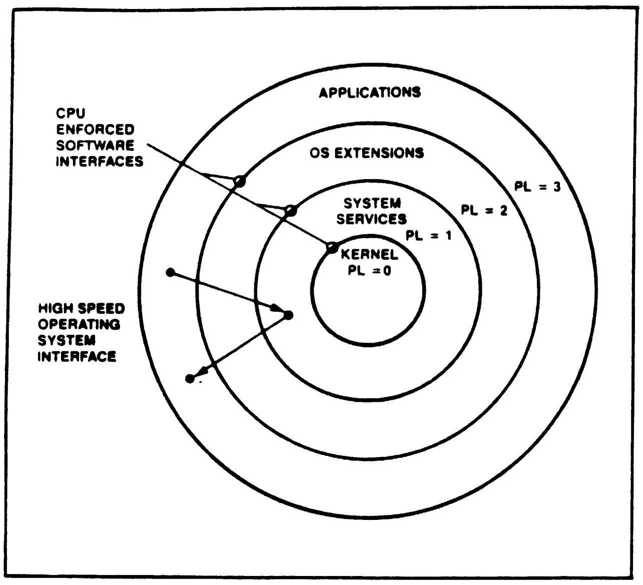


Figure 19. Hierarchical Privilege Levels

levels provide protection within a task. (Tasks are isolated by providing private LDT's for each task.) Operating system routines, interrupt handlers, and other system software can be included and protected within the virtual address space of each task using the four levels of privilege. Tasks may also have a separate stack for each privilege level.

Tasks, descriptors, and selectors have a privilege level attribute that determines whether the descriptor may be used. Task privilege effects the use of instructions and descriptors. Descriptor and selector privilege only effect access to the descriptor.

TASK PRIVILEGE

A task always executes at one of the four privilege levels. The task privilege level at any specific instant is called the Current Privilege Level (CPL) and is defined by the lower two bits of the CS register. CPL cannot change during execution in a single code segment. A task's CPL may only be changed by control transfers through gate descriptors to a new code segment (See Control Transfer). Tasks begin executing at the CPL value specified by the code segment when the task is initiated via a task switch operation. A task executing at Level 0 can access all data segments defined in the GDT and the task's LDT and is considered the most trusted level. A task executing at Level 3 has the most restricted access to data and is considered the least trusted level.

DESCRIPTOR PRIVILEGE

Descriptor privilege is specified by the Descriptor Privilege Level (DPL) field of the descriptor access byte. DPL specifies the least trusted task privilege level (CPL) at which a task may access the descriptor. Descriptors with DPL = 0 are the most protected. Only tasks executing at privilege level 0 (CPL = 0) may access them. Descriptors with DPL = 3 are the least protected (i.e. have the least restricted access) since tasks can access them when CPL = 0, 1, 2, or 3. This rule applies to all descriptors, except LDT descriptors.

SELECTOR PRIVILEGE

Selector privilege is specified by the Requested Privilege Level (RPL) field in the least significant two bits of a selector. Selector RPL may establish a less trusted privilege level than the current privilege level for the use of a selector. This level is called the task's effective privilege level (EPL). RPL can only reduce the scope of a task's access to data with this selector. A task's effective privilege is the numeric maximum of RPL and CPL. A selector with RPL = 0 imposes no additional restriction on its use while a selector with RPL = 3 can only refer to segments at privilege Level 3 regardless of the task's CPL. RPL is generally used to verify that pointer parameters passed to a more trusted procedure are not allowed to use data at a more privileged level than the caller (refer to pointer testing instructions).

Descriptor Access and Privilege Validation

Determining the ability of a task to access a segment involves the type of segment to be accessed, the instruction used, the type of descriptor used and CPL, RPL, and DPL. The two basic types of segment accesses are control transfer (selectors loaded into CS) and data (selectors loaded into DS, ES or SS).

DATA SEGMENT ACCESS

Instructions that load selectors into DS and ES must refer to a data segment descriptor or readable code segment descriptor. The CPL of the task and the RPL of the selector must be the same as or more privileged (numerically equal to or lower than) than the descriptor DPL. In general, a task can only access data segments at the same or less privileged levels than the CPL or RPL (whichever is numerically higher) to prevent a program from accessing data it cannot be trusted to use.

An exception to the rule is a readable conforming code segment. This type of code segment can be read from any privilege level.

If the privilege checks fail (e.g. DPL is numerically less than the maximum of CPL and RPL) or an incorrect type of descriptor is referenced (e.g. gate descriptor or execute only code segment) exception 13 occurs. If the segment is not present, exception 11 is generated.

Instructions that load selectors into SS must refer to data segment descriptors for writable data segments. The descriptor privilege (DPL) and RPL must equal CPL. All other descriptor types or a privilege level violation will cause exception 13. A not present fault causes exception 12.

CONTROL TRANSFER

Four types of control transfer can occur when a selector is loaded into CS by a control transfer operation (see Table 10). Each transfer type can only occur if the operation which loaded the selector references the correct descriptor type. Any violation of these descriptor usage rules (e.g. JMP through a call gate or RET to a Task State Segment) will cause exception 13.

The ability to reference a descriptor for control transfer is also subject to rules of privilege. A CALL or JUMP instruction may only reference a code segment descriptor with DPL equal to the task CPL or a conforming segment with DPL of equal or greater privilege than CPL. The RPL of the selector used to reference the code descriptor must have as much privilege as CPL.

RET and IRET instructions may only reference code segment descriptors with descriptor privilege equal to or less privileged than the task CPL. The selector loaded into CS is the return address from the stack. After the return, the selector RPL is the task's new CPL. If CPL changes, the old stack pointer is popped after the return address.

When a JMP or CALL references a Task State Segment descriptor, the descriptor DPL must be the same or less privileged than the task's CPL. Reference to a valid Task State Segment descriptor causes a task switch (see Task Switch Operation). Reference to a Task State Segment descriptor at a more privileged level than the task's CPL generates exception 13.

When an instruction or interrupt references a gate descriptor, the gate DPL must have the same or less privilege than the task CPL. If DPL is at a more privileged level than CPL, exception 13 occurs. If the destination selector contained in the gate references a code segment descriptor, the code segment descriptor DPL must be the same or more privileged than the task CPL. If not, Exception 13 is issued. After the control transfer, the code segment descriptors DPL is the task's new CPL. If the destination selector in the gate references a task state segment, a task switch is automatically performed (see Task Switch Operation).

The privilege rules on control transfer require:

—JMP or CALL direct to a code segment (code segment descriptor) can only be to a conforming segment with DPL of equal or greater privilege than CPL or a non-conforming segment at the same privilege level.

—interrupts within the task or calls that may change privilege levels, can only transfer control through a gate at the same or a less privileged level than CPL to a code segment at the same or more privileged level than CPL.

—return instructions that don't switch tasks can only return control to a code segment at the same or less privileged level.

—task switch can be performed by a call, jump or interrupt which references either a task gate or task state segment at the same or less privileged level.

Table 10. Descriptor Access Rules for Control Transfer

Control Transfer Types	Operation Types	Descriptor Referenced	Descriptor Table
Intersegment within the same privilege level	JMP, CALL, RET, IRET*	Code Segment	GDT/LDT
Intersegment to the same or higher privilege level Interrupt within task may change CPL.	CALL	Call Gate	GDT/LDT
	Interrupt Instruction, Exception, External Interrupt	Trap or Interrupt Gate	IDT
Intersegment to a lower privilege level (changes task CPL)	RET, IRET*	Code Segment	GDT/LDT
Task Switch	CALL, JMP	Task State Segment	GDT
	CALL, JMP	Task Gate	GDT/LDT
	IRET** Interrupt Instruction, Exception, External Interrupt	Task Gate	IDT

*NT (Nested Task bit of flag word) = 0
**NT (Nested Task bit of flag word) = 1

PRIVILEGE LEVEL CHANGES

Any control transfer that changes CPL within the task, causes a change of stacks as part of the operation. Initial values of SS:SP for privilege levels 0, 1, and 2 are kept in the task state segment (refer to Task Switch Operation). During a JMP or CALL control transfer, the new stack pointer is loaded into the SS and SP registers and the previous stack pointer is pushed onto the new stack.

When returning to the original privilege level, its stack is restored as part of the RET or IRET instruction operation. For subroutine calls that pass parameters on the stack and cross privilege levels, a fixed number of words, as specified in the gate, are copied from the previous stack to the current stack. The inter-segment RET instruction with a stack adjustment value will correctly restore the previous stack pointer upon return.

Protection

The 80286 includes mechanisms to protect critical instructions that affect the CPU execution state (e.g. HLT) and code or data segments from improper usage. These mechanisms are grouped under the term "protection" and have three forms:

Restricted usage of segments (e.g. no write allowed to read-only data segments). The only segments available for use are defined by descriptors in the Local Descriptor Table (LDT) and Global Descriptor Table (GDT).

Restricted access to segments via the rules of privilege and descriptor usage.

Privileged instructions or operations that may only be executed at certain privilege levels as determined by the CPL and I/O Privilege Level (IOPL). The IOPL is defined by bits 14 and 13 of the flag word.

These checks are performed for all instructions and can be split into three categories: segment load checks (Table 11), operand reference checks (Table 12), and privileged instruction checks (Table 13). Any violation of the rules shown will result in an exception. A not-present exception related to the stack segment causes exception 12.

The IRET and POPF instructions do not perform some of their defined functions if CPL is not of sufficient privilege (numerically small enough). No exceptions or other indication are given when these conditions occur.

The IF bit is not changed if CPL > IOPL.

The IOPL field of the flag word is not changed if CPL > 0.

Table 11
Segment Register Load Checks

Error Description	Exception Number
Descriptor table limit exceeded	13
Segment descriptor not-present	11 or 12
Privilege rules violated	13
Invalid descriptor/segment type segment register load: —Read only data segment load to SS —Special control descriptor load to DS, ES, SS —Execute only segment load to DS, ES, SS —Data segment load to CS —Read/Execute code segment load to SS	13

Table 12. Operand Reference Checks

Error Description	Exception Number
Write into code segment	13
Read from execute-only code segment	13
Write to read-only data segment	13
Segment limit exceeded¹	12 or 13

Note 1: Carry out in offset calculations is ignored.

Table 13. Privileged Instruction Checks

Error Description	Exception Number
CPL ≠ 0 when executing the following instructions: LIDT, LLDT, LGDT, LTR, LMSW, CTS, HLT	13
CPL > IOPL when executing the following instructions: INS, IN, OUTS, OUT, STI, CLI, LOCK	13

EXCEPTIONS

The 80286 detects several types of exceptions and interrupts, in protected mode (see Table 14). Most are restartable after the exceptional condition is removed. Interrupt handlers for most exceptions receive an error code, pushed on the stack after the return address, that identifies the selector involved (0 if none). The return address normally points to the failing instruction, including all leading prefixes. For a processor extension segment overrun exception, the return address will not point at the ESC instruction that caused the exception; however, the processor extension registers may contain the address of the failing instruction.

Table 14. Protected Mode Exceptions

Interrupt Vector	Function	Return Address At Failing Instruction?	Always Restart-able?	Error Code on Stack?
8	Double exception detected	Yes	No	Yes
9	Processor extension segment overrun	No	No	No
10	Invalid task state segment	Yes	Yes	Yes
11	Segment not present	Yes	Yes	Yes
12	Stack segment overrun or segment not present	Yes	Yes[1]	Yes
13	General protection	Yes	No	Yes

Note 1: When a PUSHA or POPA instruction attempts to wrap around the stack segment, the machine state after the exception will not be restartable. This condition is identified by the value of the saved SP being either 0000(H), 0001(H), FFFE(H), or FFFF(H).

All these checks are performed for all instructions and can be split into three categories: segment load checks (Table 11), operand reference checks (Table 12), and privileged instruction checks (Table 13). Any violation of the rules shown will result in an exception. A not-present exception related to the stack segment causes exception 12.

Special Operations

TASK SWITCH OPERATION

The 80286 provides a built-in task switch operation which saves the entire 80286 execution state (registers, address space, and a link to the previous task), loads a new execution state, and commences execution in the new task. Like gates, the task switch operation is invoked by executing an inter-segment JMP or CALL instruction which refers to a Task State Segment (TSS) or task gate descriptor in the GDT or LDT. An INT n instruction, exception, or external interrupt may also invoke the task switch operation by selecting a task gate descriptor in the associated IDT descriptor entry.

The TSS descriptor points at a segment (see Figure 20) containing the entire 80286 execution state while a task gate descriptor contains a TSS selector. The limit field must be > 002B(H).

Each task must have a TSS associated with it. The current TSS is identified by a special register in the 80286 called the Task Register (TR). This register contains a selector referring to the task state segment descriptor that defines the current TSS. A hidden base and limit register associated with TR are loaded whenever TR is loaded with a new selector.

The IRET instruction is used to return control to the task that called the current task or was interrupted. Bit 14 in the flag register is called the Nested Task (NT) bit. It controls the function of the IRET instruction. If NT = 0, the IRET instruction performs the regular current task return; when NT = 1, IRET performs a task switch operation back to the previous task.

When a CALL or INT instruction initiates a task switch, the old and new TSS will be marked busy and the back link field of the new TSS set to the old TSS selector. The NT bit of the new task is set by CALL or INT initiated task switches. An interrupt that does not cause a task switch will clear NT. NT may also be set or cleared by POPF or IRET instructions.

The task state segment is marked busy by changing the descriptor type field from Type 1 to Type 3. Use of a selector that references a busy task state segment causes Exception 13.

PROCESSOR EXTENSION CONTEXT SWITCHING

The context of a processor extension (such as the 80287 numerics processor) is not changed by the task switch operation. A processor extension context need only be changed when a different task attempts to use the processor extension (which still contains the context of a previous task). The 80286 detects the first use of a processor extension after a task switch by causing the processor extension not present exception (7). The interrupt handler may then decide whether a context change is necessary.

Whenever the 80286 switches tasks, it sets the Task Switched (TS) bit of the MSW. TS indicates that a processor extension context may belong to a different task than the current one. The processor extension not present exception (7) will occur when attempting to execute an ESC or WAIT instruction if TS = 1 and a processor extension is present (MP = 1 in MSW).

POINTER TESTING INSTRUCTIONS

The iAPX 80286 provides several instructions to speed pointer testing and consistency checks for maintaining system integrity (see Table 15). These instructions use the memory management hardware to verify that a selector value refers to an appropriate segment without risking an exception. A condition flag indicates whether use of the selector or segment will cause an exception.

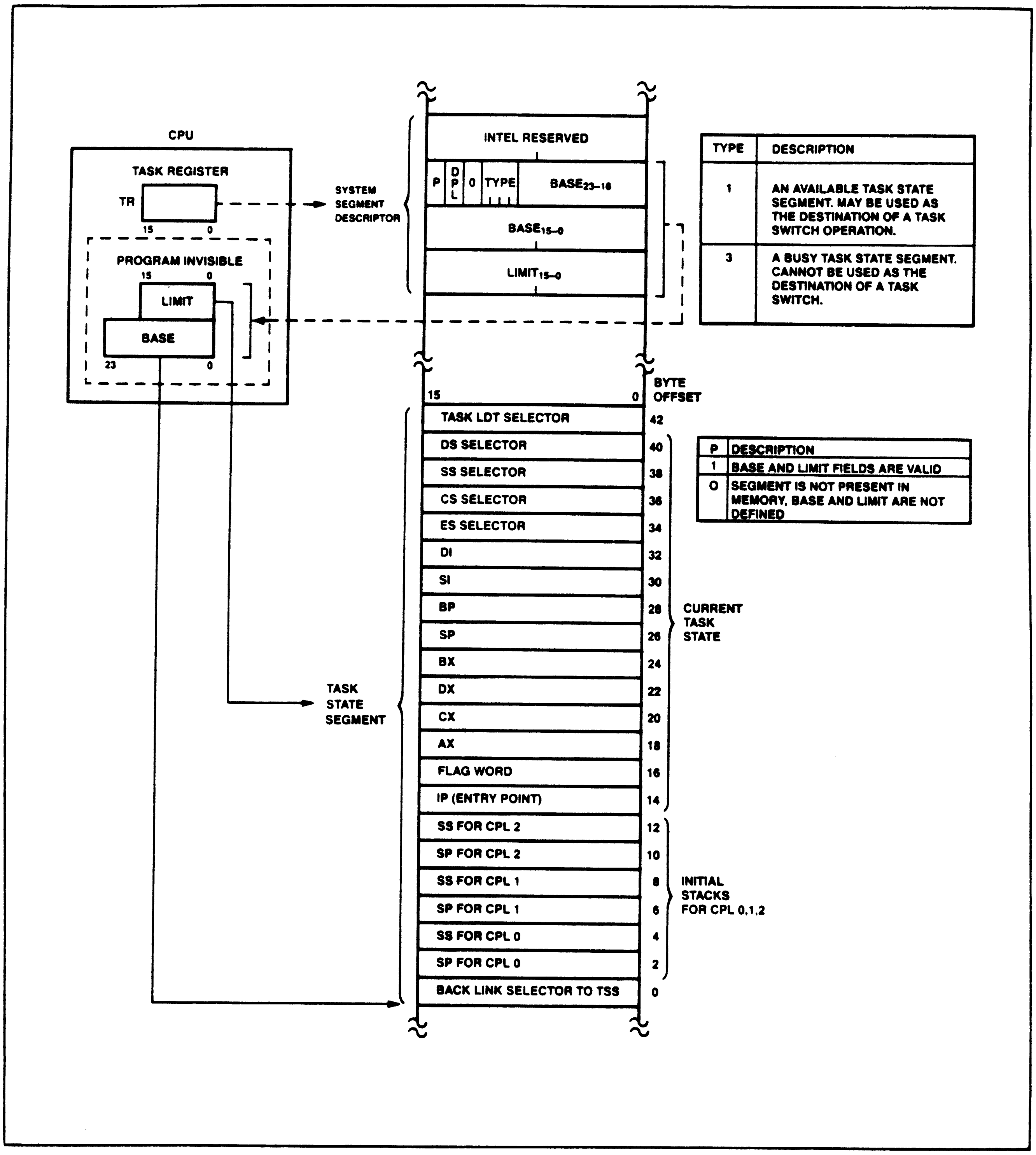


Figure 20. Task State Segment and TSS Registers

Table 15. Pointer Test Instructions

Instruction	Operands	Function
ARPL	Selector, Register	Adjust Requested Privilege Level: adjusts the RPL of the selector to the numeric maximum of current selector RPL value and the RPL value in the register. Set zero flag if selector RPL was changed.
VERR	Selector	VERify for Read: sets the zero flag if the segment referred to by the selector can be read.
VERW	Selector	VERify for Write: sets the zero flag if the segment referred to by the selector can be written.
LSL	Register, Selector	Load Segment Limit: reads the segment limit into the register if privilege rules and descriptor type allow. Set zero flag if successful.
LAR	Register, Selector	Load Access Rights: reads the descriptor access rights byte into the register if privilege rules allow. Set zero flag if successful.

DOUBLE FAULT AND SHUTDOWN

If two separate exceptions are detected during a single instruction execution, the 80286 performs the double fault exception (8). If an exception occurs during processing of the double fault exception, the 82086 will enter shutdown. During shutdown no further instructions or exceptions are processed. Either NMI (CPU remains in protected mode) or RESET (CPU exits protected mode) can force the 80286 out of shutdown. Shutdown is externally signalled via a HALT bus operation with A_1 HIGH.

PROTECTED MODE INITIALIZATION

The 80286 initially executes in real address mode after RESET. To allow initialization code to be placed at the top of physical memory, A_{23-20} will be HIGH when the 80286 performs memory references relative to the CS register, until CS is changed. A_{23-20} will be zero for references to the DS, ES, or SS segments. Changing CS in real address mode will force A_{23}–A_{20} LOW whenever using CS thereafter. The initial CS:IP value of FF00:FFF0 provides 64K bytes of code space for initialization code without changing CS.

Before placing the 80286 into protected mode, several registers must be initialized. The GDT and IDT base registers must refer to a valid GDT and IDT. After executing the LMSW instruction to set PE, the 80286 must immediately execute an intra-segment JMP instruction to clear the instruction queue of instructions decoded in real address mode.

To force the 80286 CPU registers to match the initial protected mode state assumed by software, execute a JMP instruction with a selector referring to the initial TSS used in the system. This will load the task register, local descriptor table register, segment registers and initial general register state. The TR should point at a valid TSS since a task switch operation involves saving the current task state.

SYSTEM INTERFACE

The 80286 system interface appears in two forms: a local bus and a system bus. The local bus consists of address, data, status, and control signals at the pins of the CPU. A system bus is any buffered version of the local bus. A system bus may also differ from the local bus in terms of coding of status and control lines and/or timing and loading of signals. The iAPX 286 family includes several devices to generate standard system buses such as the IEEE 796 standard Multibus™.

Bus Interface Signals and Timing

The iAPX 286 microsystem local bus interfaces the 80286 to local memory and I/O components. The interface has 24 address lines, 16 data lines, and 8 status and control signals.

The 80286 CPU, 82284 clock generator, 82288 bus controller, 82289 bus arbiter, 8286/7 transceivers, and 8282/3 latches provide a buffered and decoded system bus interface. The 82284 generates the system clock and synchronizes $\overline{\text{READY}}$ and RESET. The 82288 converts bus operation status encoded by the 80286 into command and bus control signals. The 82289 bus arbiter generates multibus bus arbitration signals. These components can provide the timing and electrical power drive levels required for most system bus interfaces including the multibus.

Physical Memory and I/O Interface

A maximum of 16 megabytes of physical memory can be addressed in protected mode. One megabyte can be addressed in real address mode. Memory is accessible as bytes or words. Words consist of any two consecutive bytes addressed with the least significant byte stored in the lowest address.

Byte transfers occur on either half of the 16-bit local data bus. Even bytes are accessed over D_{7-0} while odd bytes are transferred over D_{15-8}. Even-addressed words are transferred over D_{15-0} in one bus cycle, while odd-addressed words require two bus operations. The first transfers data on D_{15-8}, and the second transfers data on D_{7-0}. Both byte data transfers occur automatically, transparent to software.

Two bus signals, A_0 and $\overline{\text{BHE}}$, control transfers over the lower and upper halves of the data bus. Even address byte transfers are indicated by A_0 LOW and $\overline{BHE}$ HIGH. Odd address byte transfers are indicated by A_0 HIGH and $\overline{BHE}$ LOW. Both A_0 and $\overline{BHE}$ are LOW for even address word transfers.

The I/O address space contains 64K addresses in both modes. The I/O space is accessible as either bytes or words, as is memory. Byte wide peripheral devices may be attached to either the upper or lower byte of the data bus. Byte-wide I/O devices attached to the upper data byte (D_{15-8}) are accessed with odd I/O addresses. Devices on the lower data byte are accessed with even I/O addresses. An interrupt controller such as Intel's 8259A must be connected to the lower data byte (D_{7-0}) for proper return of the interrupt vector.

Bus Operation

The 80286 uses a double frequency system clock (CLK input) to control bus timing. All signals on the local bus are measured relative to the system CLK input. The CPU divides the system clock by 2 to produce the internal processor clock, which determines bus state. Each processor clock is composed of two system clock cycles named phase 1 and phase 2. The 82284 clock generator output (PCLK) identifies the next phase of the processor clock. (See Figure 21.)

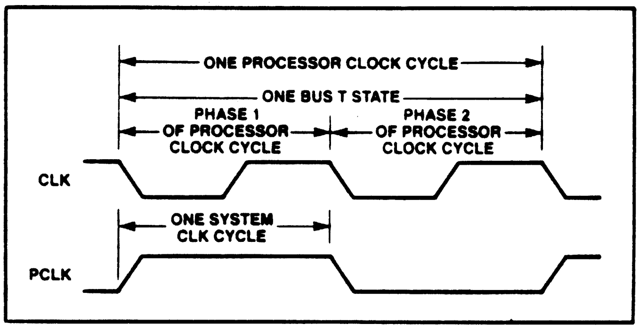

ONE PROCESSOR CLOCK CYCLE
ONE BUS T STATE
PHASE 1 OF PROCESSOR CLOCK CYCLE
PHASE 2 OF PROCESSOR CLOCK CYCLE
CLK
ONE SYSTEM CLK CYCLE
PCLK

Figure 21. System and Processor Clock Relationships

Six types of bus operations are supported; memory read, memory write, I/O read, I/O write, interrupt acknowledge, and halt/shutdown. Data can be transferred at a maximum rate of one word per two processor clock cycles.

The iAPX 286 bus has three basic states: idle (T_i), send status (T_s), and perform command (T_c). The 80286 CPU also has a fourth local bus state called hold (T_h). T_h indicates that the 80286 has surrendered control of the local bus to another bus master in response to a HOLD request.

Each bus state is one processor clock long. Figure 22 shows the four 80286 local bus states and allowed transitions.

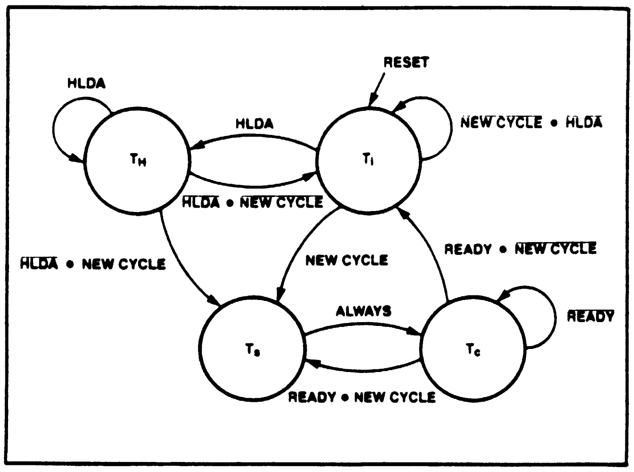

RESET
HLDA
HLDA
T_H
T_I
$\overline{NEW CYCLE}$ • $\overline{HLDA}$
$\overline{HLDA}$ • $\overline{NEW CYCLE}$
$\overline{HLDA}$ • NEW CYCLE
NEW CYCLE
READY • $\overline{NEW CYCLE}$
ALWAYS
$\overline{READY}$
T_S
T_C
READY • NEW CYCLE

Figure 22. 80286 Bus States

Bus States

The idle (T_i) state indicates that no data transfers are in progress or requested. The first active state, T_s is signalled by either status line $\overline{S1}$ or $\overline{S0}$ going LOW also identifying phase 1 of the processor clock. During T_s, the command encoding, the address, and data (for a write operation) are available on the 80286 output pins. The 82288 bus controller decodes the status signals and generates Multibus compatible read/write command and local transceiver control signals.

After T_s, the perform command (T_c) state is entered. Memory or I/O devices respond to the bus operation during T_c, either transferring read data to the CPU or accepting write data. T_c states may be repeated as often as necessary to assure sufficient time for the memory or I/O device to respond. The $\overline{READY}$ signal determines whether T_c is repeated.

During hold (T_h), the 80286 will float all address, data, and status output pins enabling another bus master to use the local bus. The 80286 HOLD input signal is used to place the 80286 into the T_h state. The 80286 HLDA output signal indicates that the CPU has entered T_h.

Pipelined Addressing

The 80286 uses a local bus interface with pipelined timing to allow as much time as possible for data access. Pipelined timing allows bus operations to be performed in two processor cycles, while allowing each individual bus operation to last for three processor cycles.

The timing of the address outputs is pipelined such that the address of the next bus operation becomes available during the current bus operation. Or in other words, the first clock of the next bus operation is overlapped with the last clock of the current bus operation. Therefore, address decode and routing logic can operate in ad-

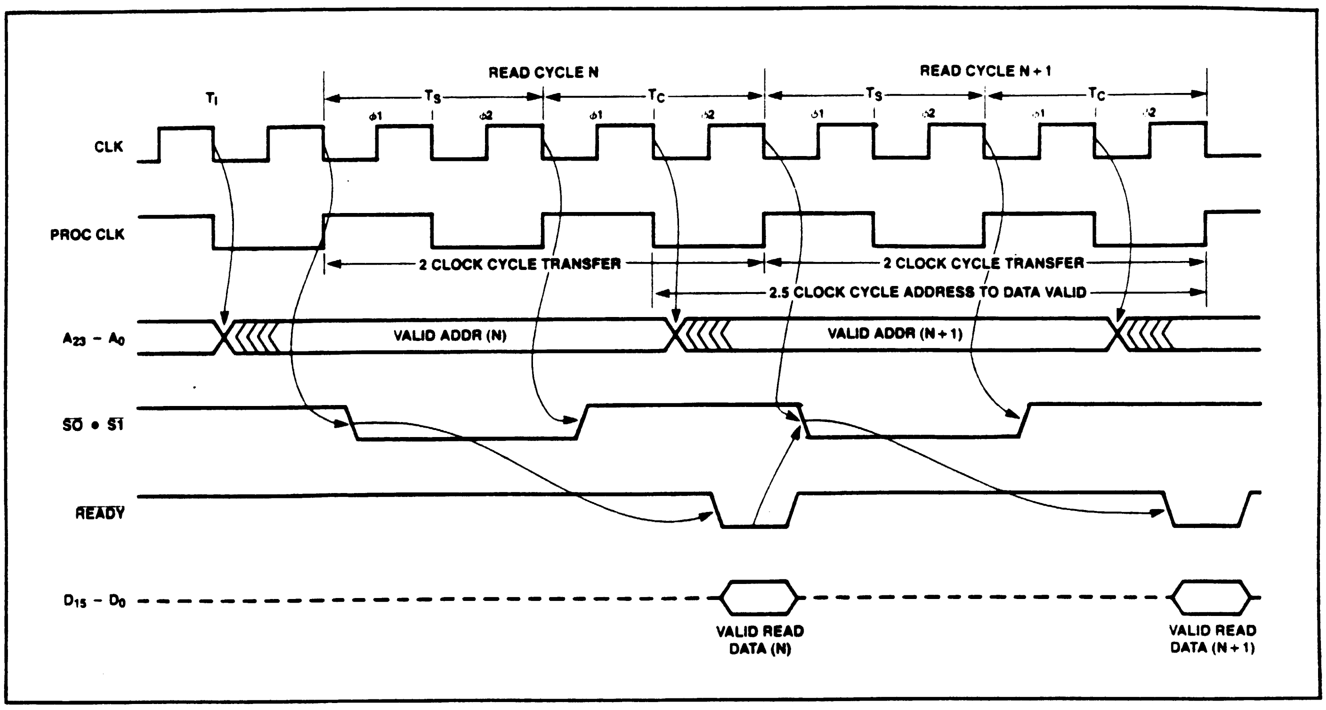


Figure 23. Basic Bus Cycle

vance of the next bus operation. External address latches may hold the address stable for the entire bus operation, and provide additional AC and DC buffering.

The 80286 does not maintain the address of the current bus operation during all T_C states. Instead, the address for the next bus operation may be emitted during phase 2 of any T_C. The address remains valid during phase 1 of the first T_C to guarantee hold time, relative to ALE, for the address latch inputs.

Bus Control Signals

The 82288 bus controller provides control signals; address latch enable (ALE), Read/Write commands, data transmit/receive (DT/$\overline{R}$), and data enable (DEN) that control the address latches, data transceivers, write enable, and output enable for memory and I/O systems.

The Address Latch Enable (ALE) output determines when the address may be latched. ALE provides at least one system CLK period of address hold time from the end of the previous bus operation until the address for the next bus operation appears at the latch outputs. This address hold time is required to support Multibus® and common memory systems.

The data bus transceivers are controlled by 82288 outputs Data Enable (DEN) and Data Transmit/Receive (DT/$\overline{R}$). DEN enables the data transceivers; while DT/$\overline{R}$ controls transceiver direction. DEN and DT/$\overline{R}$ are timed to prevent bus contention between the bus master, data bus transceivers, and system data bus tranceivers.

Command Timing Controls

Two system timing customization options, command extension and command delay, are provided on the iAPX 286 local bus.

Command extension allows additional time for external devices to respond to a command and is analogous to inserting wait states on the 8086. External logic can control the duration of any bus operation such that the operation is only as long as necessary. The $\overline{READY}$ input signal can extend any bus operation for as long as necessary.

Command delay allows an increase of address or write data setup time to system bus command active for any bus operation by delaying when the system bus command becomes active. Command delay is controlled by the 82288 CMDLY input. After T_S, the bus controller samples CMDLY at each failing edge of CLK. If CMDLY is HIGH, the 82288 will not activate the command signal. When CMDLY is LOW, the 82288 will activate the command signal. After the command becomes active, the CMDLY input is not sampled.

When a command is delayed, the available response time from command active to return read data or accept write data is less. To customize system bus timing, an address decoder can determine which bus operations require delaying the command. The CMDLY input does not affect the timing of ALE, DEN, or DT/$\overline{R}$.

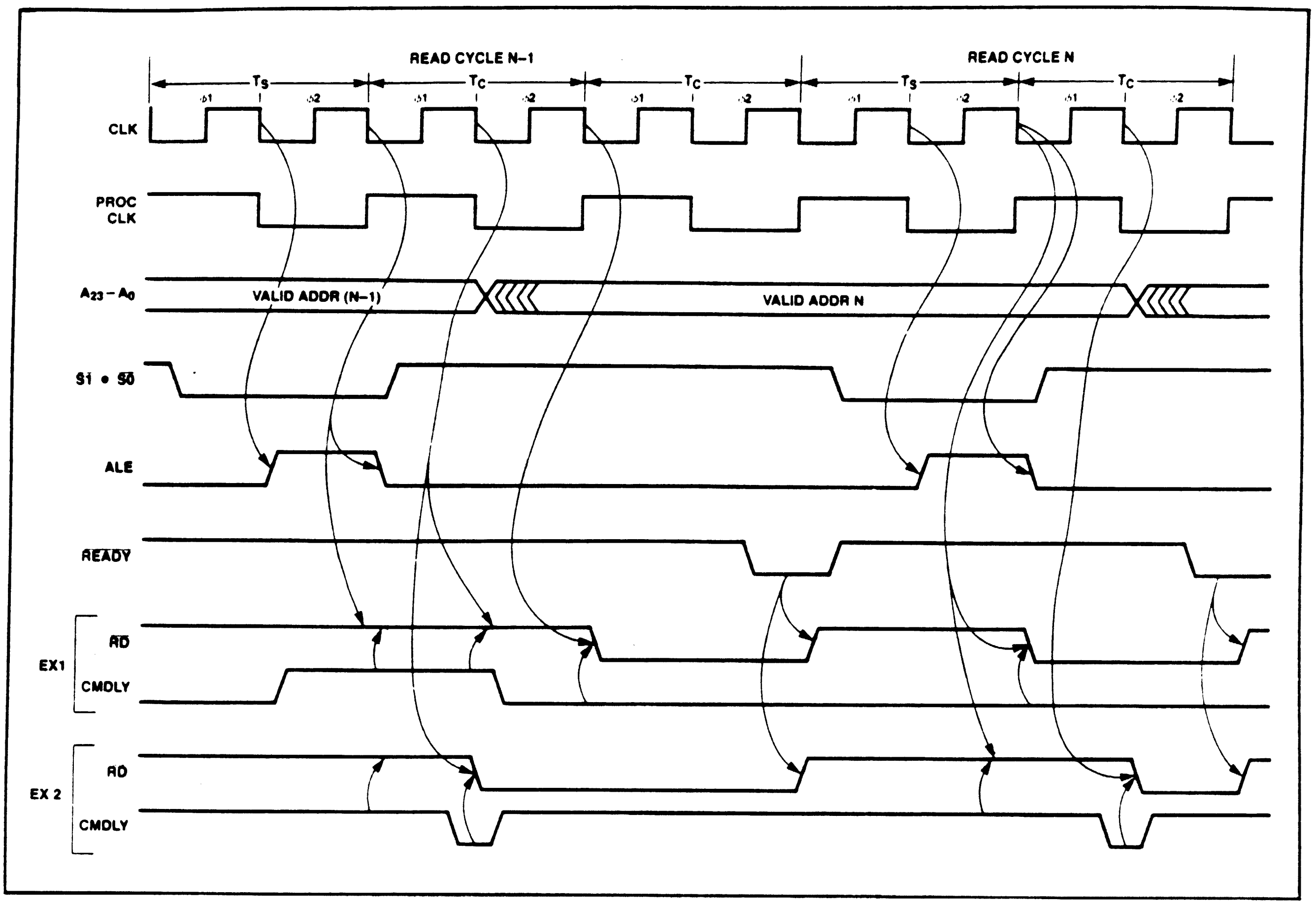


Figure 24. CMDLY Controls and Leading Edge of the Command

Figure 24 illustrates four uses of CMDLY. Example 1 shows delaying the read command two system CLKs for cycle N-1 and no delay for cycle N, and example 2 shows delaying the read command one system CLK for cycle N-1 and one system CLK delay for cycle N.

Bus Cycle Termination

At maximum transfer rates, the iAPX 286 bus alternates between the status and command states. The bus status signals become inactive after T_S so that they may correctly signal the start of the next bus operation after the completion of the current cycle. No external indication of T_C exists on the iAPX 286 local bus. The bus master and bus controller enter T_C directly after T_S and continue executing T_C cycles until terminated by $\overline{\text{READY}}$.

$\overline{\text{READY}}$ Operation

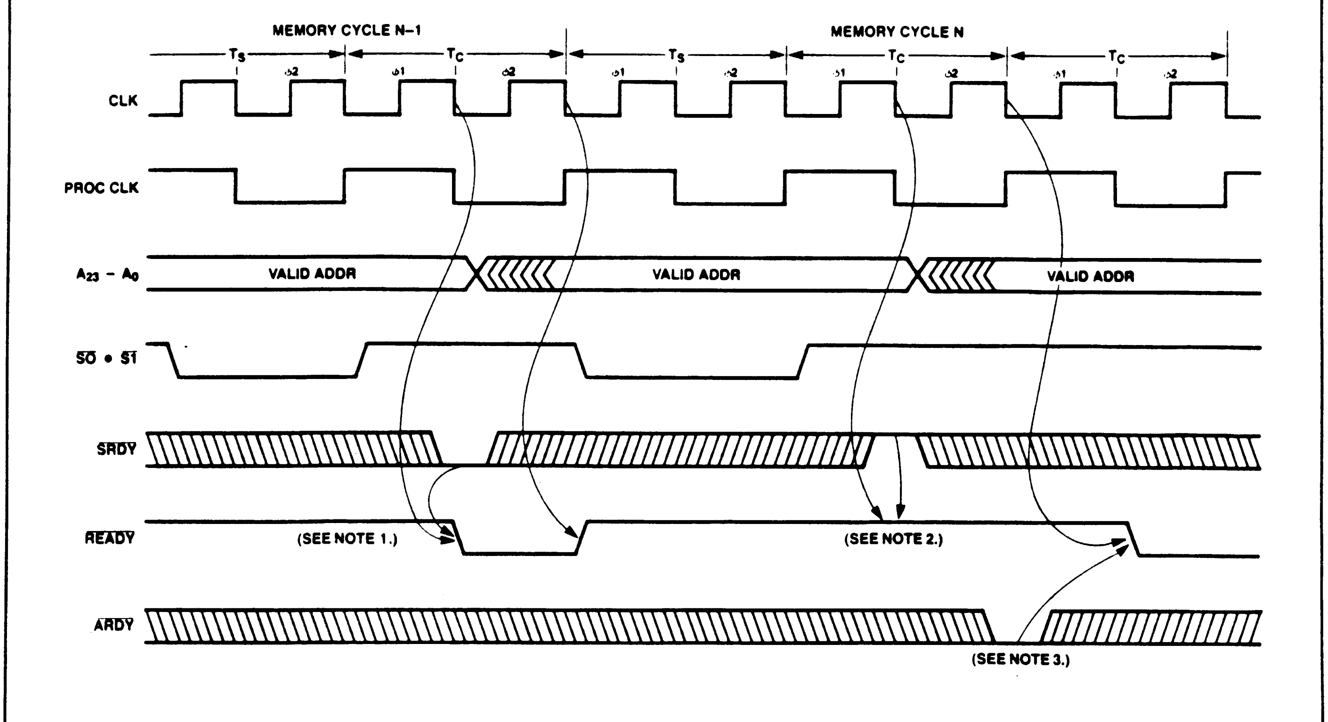
The current bus master and 82288 bus controller terminate each bus operation simultaneously to achieve maximum bus bandwidth. Both are informed in advance by $\overline{\text{READY}}$ active which identifies the last T_C cycle of the current bus operation. The bus master and bus controller must see the same sense of the $\overline{\text{READY}}$ signal, thereby requiring $\overline{\text{READY}}$ be synchronous to the system clock.

Synchronous Ready

The 82284 clock generator provides $\overline{\text{READY}}$ synchronization from both synchronous and asynchronous sources (see Figure 25). The synchronous ready input ($\overline{\text{SRDY}}$) of the clock generator is sampled with the falling edge of CLK at the end of phase 1 of each T_C. The state of $\overline{\text{SRDY}}$ is then broadcast to the bus master and bus controller via the $\overline{\text{READY}}$ output line.

Asynchronous Ready

Many systems have devices or subsystems that are asynchronous to the system clock. As a result, their ready outputs cannot be guaranteed to meet the 82284 $\overline{\text{SRDY}}$ setup and hold time requirements. The 82284 asynchronous ready input ($\overline{\text{ARDY}}$) is designed to accept such signals. The $\overline{\text{ARDY}}$ input is sampled at the beginning of each T_C cycle by 82284 synchronization logic. This provides a system CLK cycle time to resolve its value before broadcasting it to the bus master and bus controller.



NOTES:
1. $\overline{\text{SRDYEN}}$ is active low
2. If $\overline{\text{SRDYEN}}$ is high, the state of $\overline{\text{SRDY}}$ will not effect $\overline{\text{READY}}$
3. $\overline{\text{ARDYEN}}$ is active low

Figure 25. Synchronous and Asynchronous Ready

Each ready input of the 82284 has an enable pin ($\overline{\text{SRDYEN}}$ and $\overline{\text{ARDYEN}}$) to select whether the current bus operation will be terminated by the synchronous or asynchronous ready. Either of the ready inputs may terminate a bus operation. These enable inputs are active low and have the same timing as their respective ready inputs. Address decode logic usually selects whether the current bus operation should be terminated by $\overline{\text{ARDY}}$ or $\overline{\text{SRDY}}$.

Data Bus Control

Figures 26, 27, and 28 show how the DT/$\overline{\text{R}}$, DEN, data bus, and address signals operate for different combinations of read, write, and idle bus operations. DT/$\overline{\text{R}}$ goes active (LOW) for a read operaton. DT/$\overline{\text{R}}$ remains HIGH before, during, and between write operations.

The data bus is driven with write data during the second phase of T_S. The delay in write data timing allows the read data drivers, from a previous read cycle, sufficient time to enter 3-state OFF before the 80286 CPU begins driving the local data bus for write operations. Write data will always remain valid for one system clock past the last T_C to provide sufficient hold time for Multibus or other similar memory or I/O systems. During write-read or write-idle sequences the data bus enters 3-state OFF during the second phase of the processor cycle after the last T_C. In a write-write sequence the data bus does not enter 3-state OFF between T_C and T_S.

Bus Usage

The 80286 local bus may be used for several functions: instruction data transfers, data transfers by other bus masters, instruction fetching, processor extension data transfers, interrupt acknowledge, and halt/shutdown. This section describes local bus activities which have special signals or requirements.

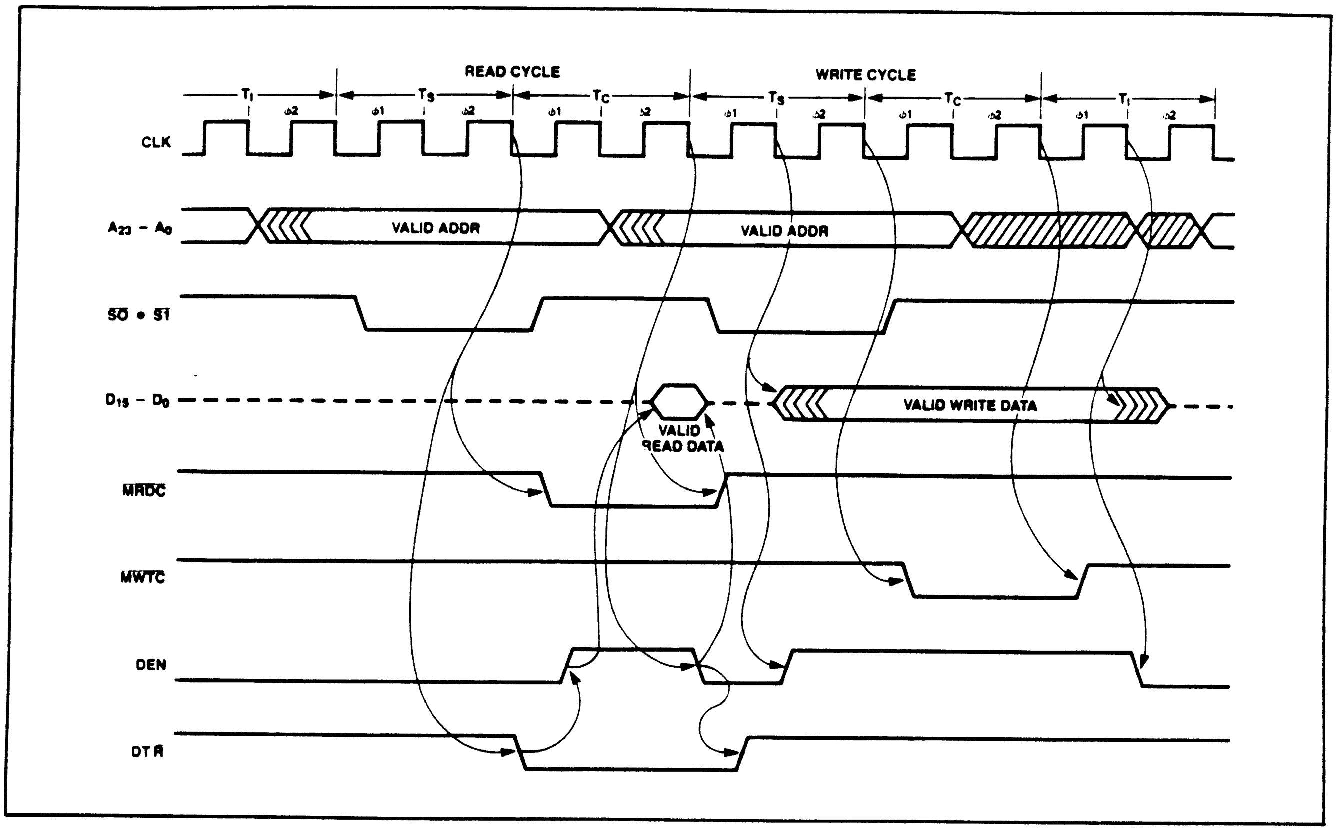


Figure 26. Back to Back Read-Write Cycles

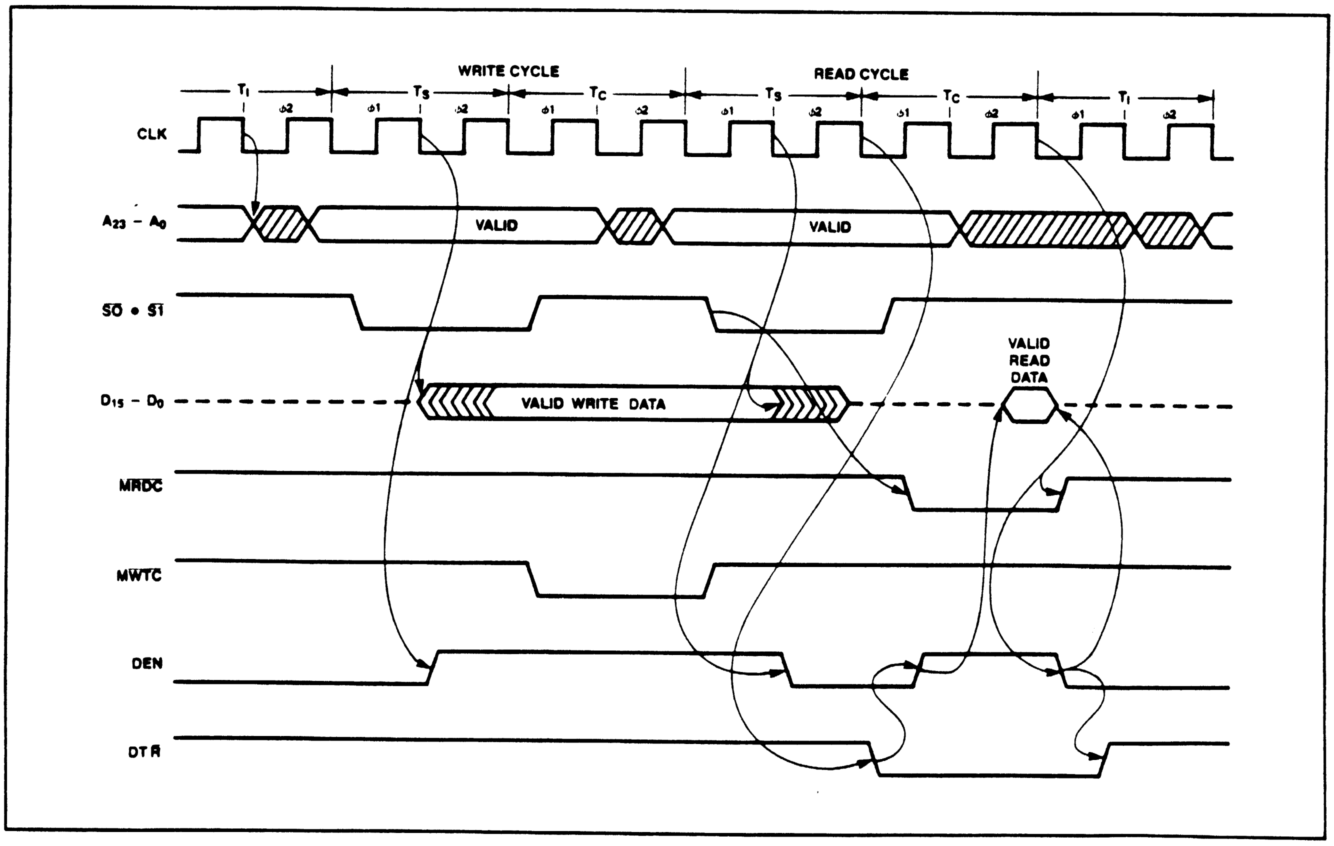


Figure 27. Back to Back Write-Read Cycles

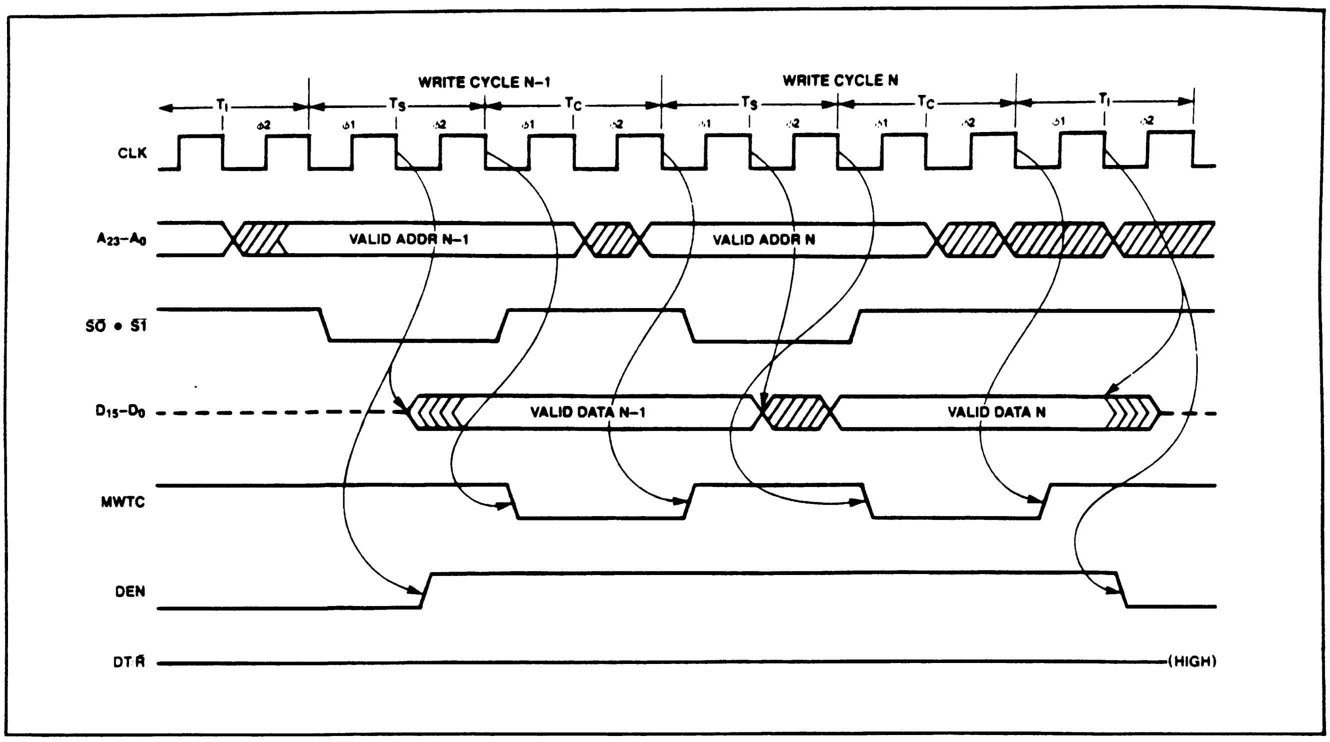


Figure 28. Back to Back Write-Write Cycles

HOLD and HOLDA

HOLD and HLDA allow another bus master to gain control of the local bus by placing the 80286 bus into the T_h state. The sequence of events required to pass control between the 80286 and another local bus master are shown in Figure 29.

In this example, the 80286 is initially in the T_h state as signaled by HLDA being active. Upon leaving T_h, as signaled by HLDA going inactive, a write operation is started. During the write operation another local bus master requests the local bus from the 80286 as shown by the HOLD signal. After completing the write operation, the 80286 performs one T_i bus cycle, to guarantee write data hold time, then enters T_h as signaled by HLDA going active.

The CMDLY signal and $\overline{\text{ARDY}}$ ready are used to start and stop the write bus command, respectively. Note that $\overline{\text{SRDY}}$ must be inactive or disabled by $\overline{\text{SRDYEN}}$ to guarantee $\overline{\text{ARDY}}$ will terminate the cycle.

Instruction Fetching

The 80286 Bus Unit (BU) will fetch instructions ahead of the current instruction being executed. This activity is called prefetching. It occurs when the local bus would otherwise be idle and obeys the following rules:

A prefetch bus operation starts when at least two bytes of the 6-byte prefetch queue are empty.

The prefetcher normally performs word prefetches independent of the byte alignment of the code segment base in physical memory.

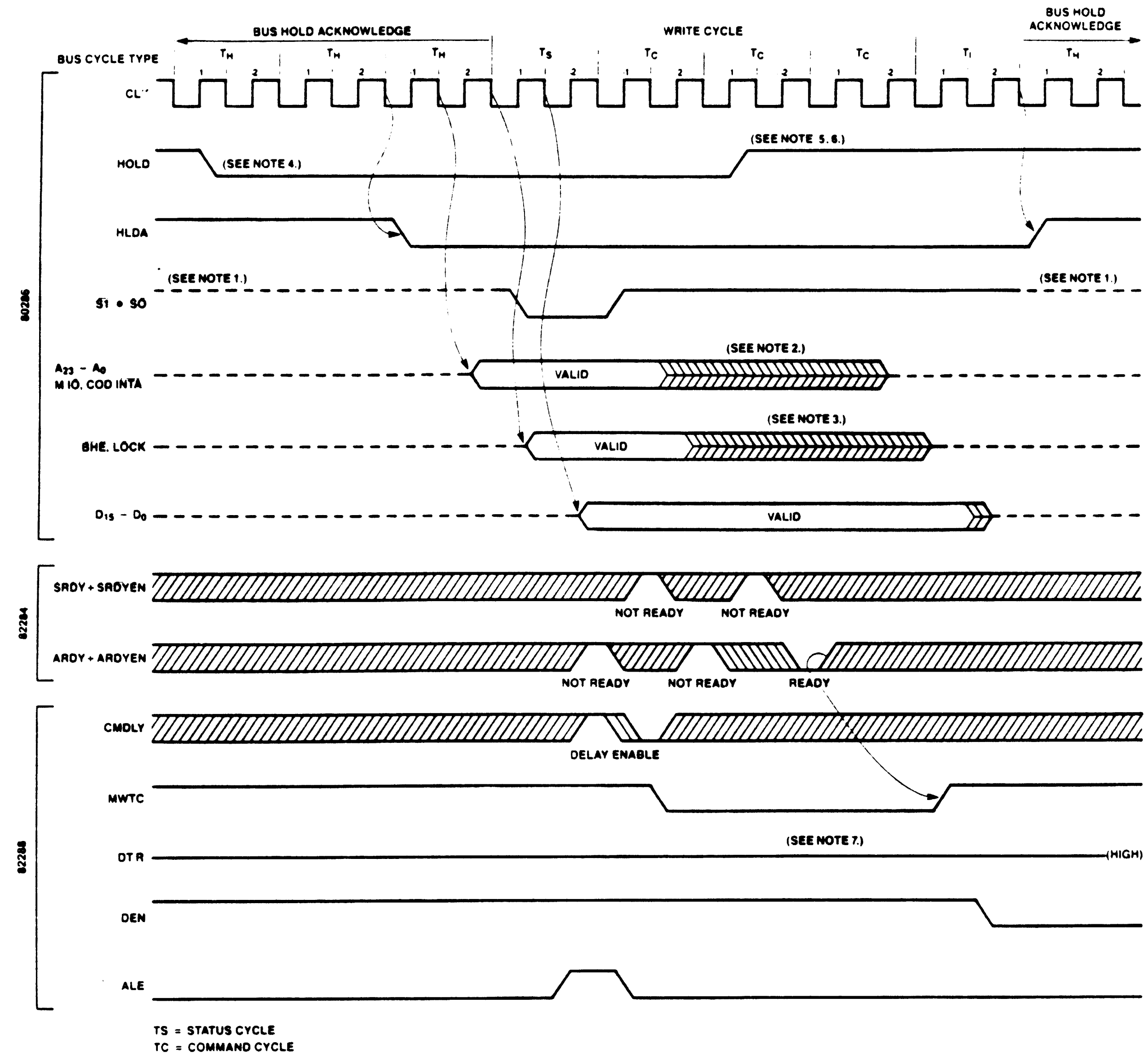
The prefetcher will perform only a byte code fetch operation for control transfers to an instruction beginning on a numerically odd physical address.

Prefetching stops whenever a control transfer or HLT instruction is decoded by the IU and placed into the instruction queue.

In real address mode, the prefetcher may fetch up to 5 bytes beyond the last control transfer or HLT instruction in a code segment.

In protected mode, the prefetcher will never cause a segment overrun exception. The prefetcher stops at the last physical memory word of the code segment. Exception 13 will occur if the program attempts to execute beyond the last full instruction in the code segment.

If the last byte of a code segment appears on an even physical memory address, the prefetcher will read the next physical byte of memory (perform a word code fetch). The value of this byte is ignored and any attempt to execute it causes exception 13.



NOTES:
1. Status lines are not driven by 80286, yet remain high due to pullup resistors in 82288 and 82289 during HOLD state.
2. Address, M/$\overline{\text{IO}}$ and COD/$\overline{\text{INTA}}$ may start floating during any TC depending on when internal 80286 bus arbiter decides to release bus to external HOLD. The float starts in $\phi 2$ of TC.
3. $\overline{\text{BHE}}$ and $\overline{\text{LOCK}}$ may start floating after the end of any TC depending on when internal 80286 bus arbiter decides to release bus to external HOLD.
4. The minimum HOLD ↓ to HLDA ↓ time is shown. Maximus is one T_H longer.
5. The earliest HOLD ↑ time is shown which will always allow a subsequent memory cycle if pending.
6. The minimum HOLD ↑ to HLDA ↑ time is shown. Maximum is a function of the instruction, type of bus cycle and other machine status (i.e., Interrupts, Waits, Lock, etc.)
7. Asynchronous ready allows termination of the cycle. Synchronous ready does not signal ready in this example. Synchronous ready state is ignored after ready is signaled via the asynchronous input.

Figure 29. Multibus Write Terminated by Asynchronous Ready

Processor Extension Transfers

The processor extension interface uses I/O port addresses 00F8(H), 00FA(H), and 00FC(H) which are part of the I/O port address range reserved by Intel. An ESC instruction with EM = 0 and TS = 0 will perform I/O bus operations to one or more of these I/O port addresses independent of the value of IOPL and CPL.

ESC instructions with memory references enable the CPU to accept PEREQ inputs for processor extension operand transfers. The CPU will determine the operand starting address and read/write status of the instruction. For each operand transfer, two or three bus operations are performed, one word transfer with I/O port address 00FA(H) and one or two bus operations with memory. Three bus operations are required for each word operand aligned on an odd byte address.

Interrupt Acknowledge Sequence

Figure 30 illustrates an interrupt acknowledge sequence performed by the 80286 in response to an INTR input. An interrupt acknowledge sequence consists of two INTA bus operations. The first allows a master 8259A Programmable Interrupt Controller (PIC) to determine which if any of its slaves should return the interrupt vector. An eight bit vector is read by the 80286 during the second INTA bus operation to select an interrupt handler routine from the interrupt table.

The Master Cascade Enable (MCE) signal of the 82288 is used to enable the cascade address drivers, during INTA bus operations (See Figure 30), onto the local address bus for distribution to slave interrupt controllers via the system address bus. The 80286 emits the $\overline{\text{LOCK}}$ signal (active LOW) during T_S of the first INTA bus operation. A local bus "hold" request will not be honored until the end of the second INTA bus operation.

Three idle processor clocks are provided by the 80286 between INTA bus operations to allow for the minimum INTA to INTA time and CAS (cascade address) out delay of the 8259A. The second INTA bus operation must always have at least one extra T_C state added via logic controlling $\overline{\text{READY}}$. A_{23}–A_0 are in 3-state OFF until after the first T_C state of the second INTA bus operation. This prevents bus contention between the cascade address drivers and CPU address drivers. The extra T_C state allows time for the 80286 to resume driving the address lines for subsequent bus operations.

Local Bus Usage Priorities

The 80286 local bus is shared among several internal units and external HOLD requests. In case of simultaneous requests, their relative priorities are:

(Highest) Any transfers which assert $\overline{\text{LOCK}}$ either explicitly (via the LOCK instruction prefix) or implicitly (i.e. segment descriptor access, interrupt acknowledge sequence, or an XCHG with memory).

The second of the two byte bus operations required for an odd aligned word operand.

Local bus request via HOLD input.

Processor extension data operand transfer via PEREQ input.

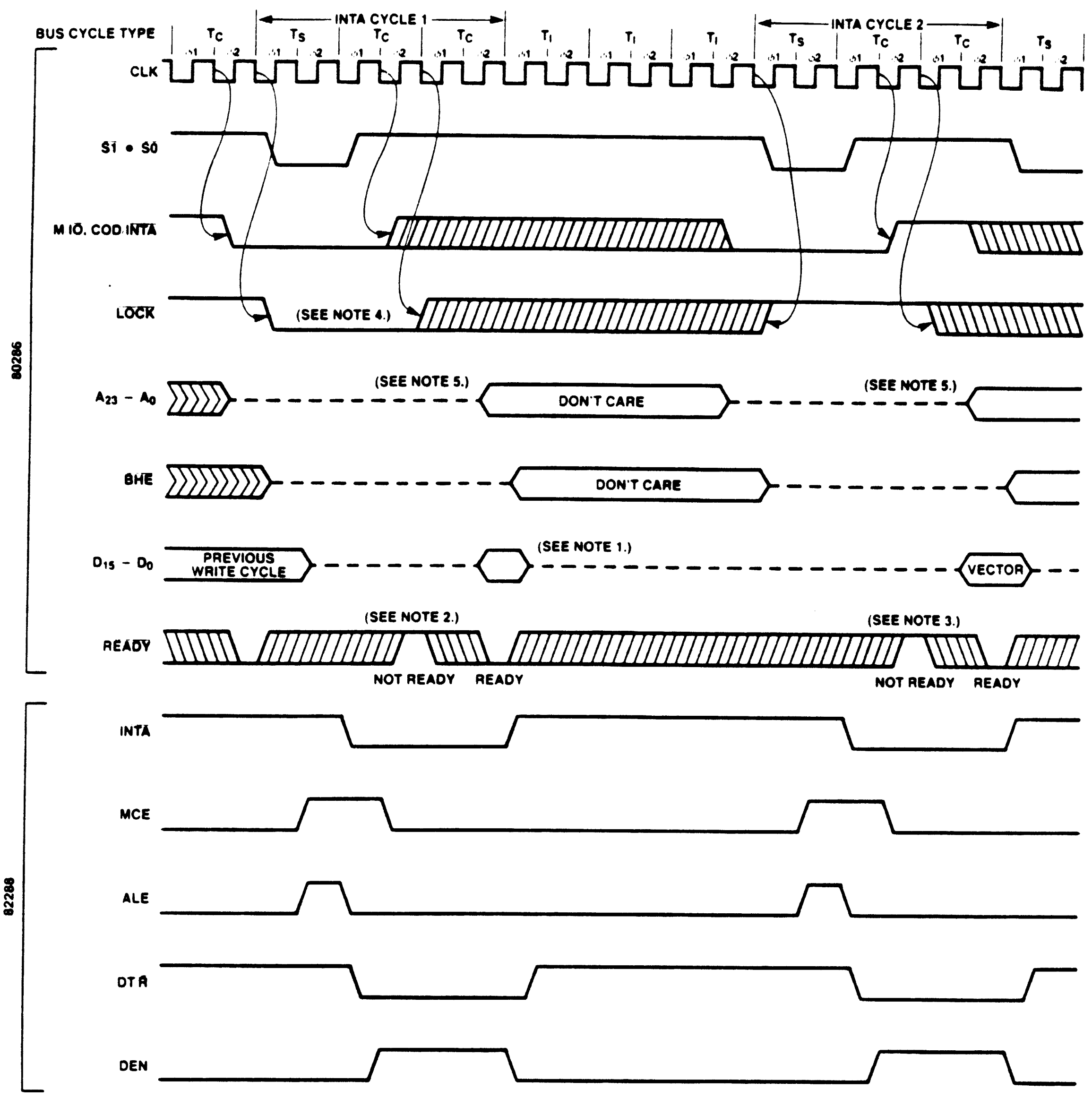
Data transfer performed by EU as part of an instruction.

(Lowest) An instruction prefetch request from BU. The EU will inhibit prefetching two processor clocks in advance of any data transfers to minimize waiting by EU for a prefetch to finish.

Halt or Shutdown Cycles

The 80286 externally indicates halt or shutdown conditions as a bus operation. These conditions occur due to a HLT instruction or multiple protection exceptions while attempting to execute one instruction. A halt or shutdown bus operation is signalled when $\overline{\text{S1}}$, $\overline{\text{S0}}$ and COD/$\overline{\text{INTA}}$ are LOW and M/$\overline{\text{IO}}$ is HIGH. A_1 HIGH indicates halt, and A_1 LOW indicates shutdown. The 82288 bus controller does not issue ALE, nor is $\overline{\text{READY}}$ required to terminate a halt or shutdown bus operation.

During halt or shutdown, the 80286 may service PEREQ or HOLD requests. A processor extension segment overrun exception during shutdown will inhibit further service of PEREQ. Either NMI or RESET will force the 80286 out of either halt or shutdown. An INTR, if interrupts are enabled, or a processor extension segment overrun exception will also force the 80286 out of halt.



NOTES:

1. Data is ignored.
2. First INTA cycle should have at least one wait state inserted to meet 8259A minimum INTA pulse width.
3. Second INTA cycle must have at least one wait state inserted since the CPU will not drive $A_{23} - A_0$, $\overline{BHE}$, and $\overline{LOCK}$ until after the first TC state.

   The CPU imposed one/clock delay prevents bus contention between cascade address buffer being disabled by MCE ↓ and address outputs.

   Without the wait state, the 80286 address will not be valid for a memory cycle started immediately after the second INTA cycle. The 8259A also requires one wait state for minimum INTA pulse width.
4. $\overline{LOCK}$ is active for the first INTA cycle to prevent the 82289 from releasing the bus between INTA cycles in a multi-master system.
5. $A_{23} - A_0$ exits 3-state OFF during $\phi 2$ of the second T_C in the INTA cycle.

Figure 30. Interrupt Acknowledge Sequence

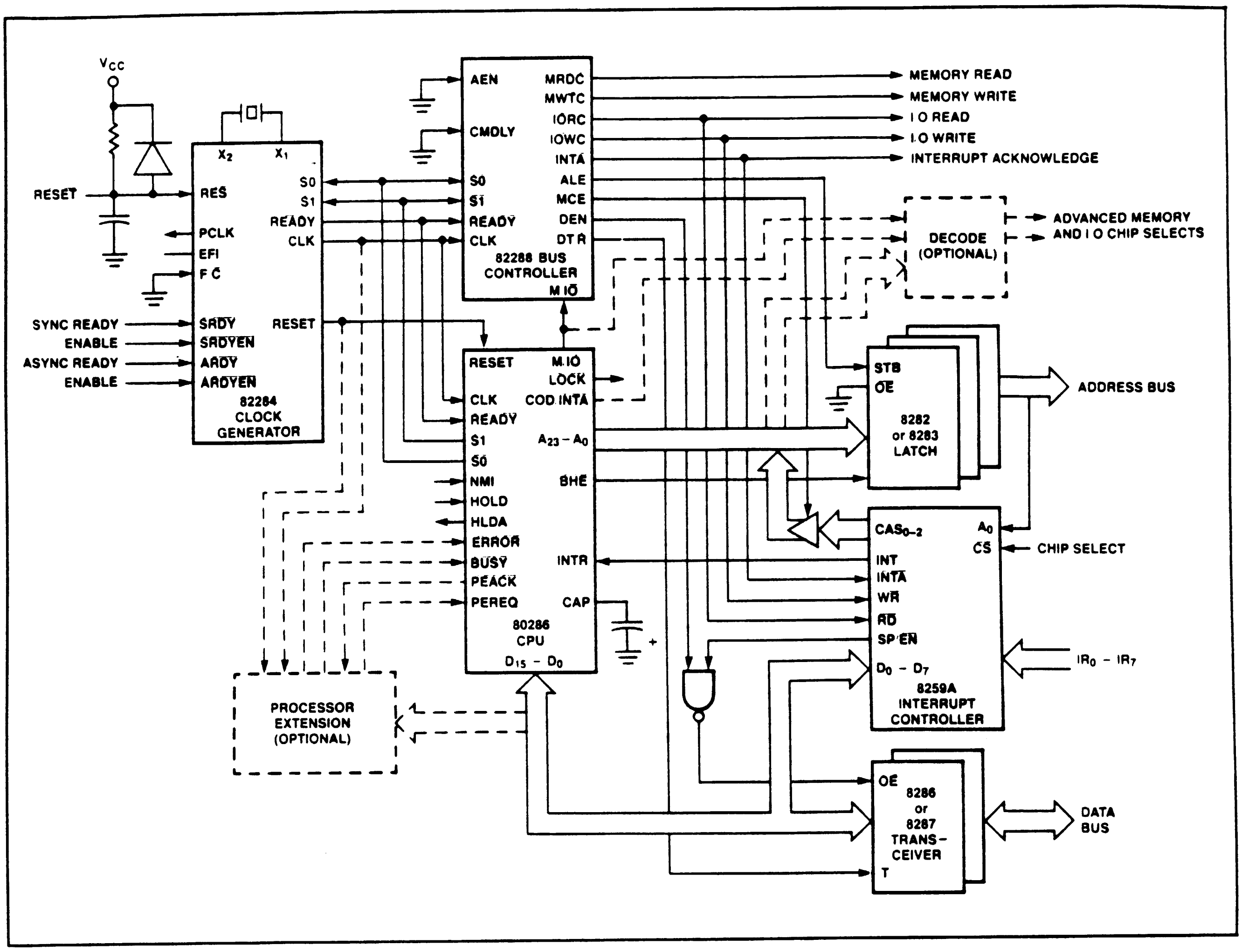


Figure 31. Basic iAPX 286 System Configuration

SYSTEM CONFIGURATIONS

The versatile bus structure of the iAPX 286 microsystem, with a full complement of support chips, allows flexible configuration of a wide range of systems. The basic configuration, shown in Figure 31, is similar to an iAPX 86 maximum mode system. It includes the CPU plus an 8259A interrupt controller, 82284 clock generator, and the 82288 Bus Controller. The iAPX 86 latches (8282 and 8283) and transceivers (8286 and 8287) may be used in an iAPX 286 microsystem.

As indicated by the dashed lines in Figure 31, the ability to add processor extensions is an integral feature of iAPX 286 microsystems. The processor extension interface allows external hardware to perform special functions and transfer data concurrent with CPU execution of other instructions. Full system integrity is maintained because the 80286 supervises all data transfers and instruction execution for the processor extension.

The iAPX 286/20 numeric data processor which includes the 80287 numeric processor extension (NPX) uses this interface. The iAPX 286/20 has all the instructions and data types of an iAPX 86/20 or iAPX 88/20. The 80287 NPX can perform numeric calculations and data transfers concurrently with CPU program execution. Numerics code and data have the same integrity as all other information protected by the iAPX 286 protection mechanism.

The 80286 can overlap chip select decoding and address propagation during the data transfer for the previous bus operation. This information is latched into the 8282/3's by ALE during the middle of a T_S cycle. The latched chip select and address information remains stable during the bus operation while the next cycles address is being decoded and propagated into the system. Decode logic can be implemented with a high speed bipolar PROM.

The optional decode logic shown in Figure 31 takes advantage of the overlap between address and data of the 80286 bus cycle to generate advanced memory and IO-select signals. This minimizes system performance

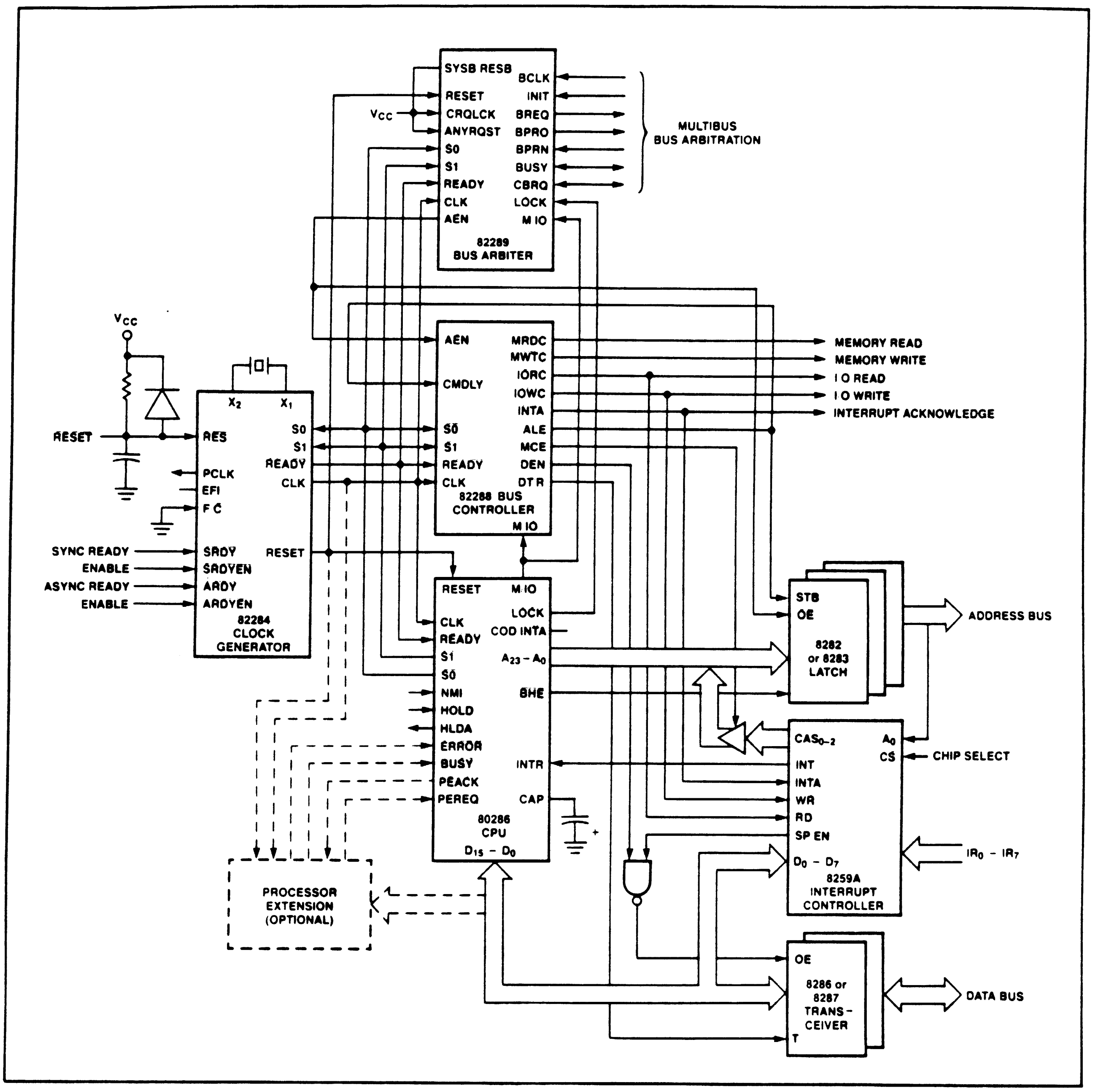


Figure 32. Multibus System Bus Interface

degradation caused by address propogation and decode delays. In addition to selecting memory and I/O, the advanced selects may be used with configurations supporting local and system buses to enable the appropriate bus interface for each bus cycle. The COD/$\overline{\text{INTA}}$ and M/$\overline{\text{IO}}$ signals are applied to the decode logic to distinguish between interrupt, I/O, code and data bus cycles.

By adding the 82289 bus arbiter chip the 80286 provides a Multibus system bus interface as shown in Figure 32. The ALE output of the 82288 for the Multibus bus is connected to its CMDLY input to delay the start of commands one system CLK as required to meet Multibus address and write data setup times. This arrangement will add at least one extra T_C state to each bus operation which uses the Multibus.

A second 82288 bus controller and additional latches and transceivers could be added to the local bus of Figure 32. This configuration allows the 80286 to support an on-board bus for local memory and peripherals, and the Multibus for system bus interfacing.

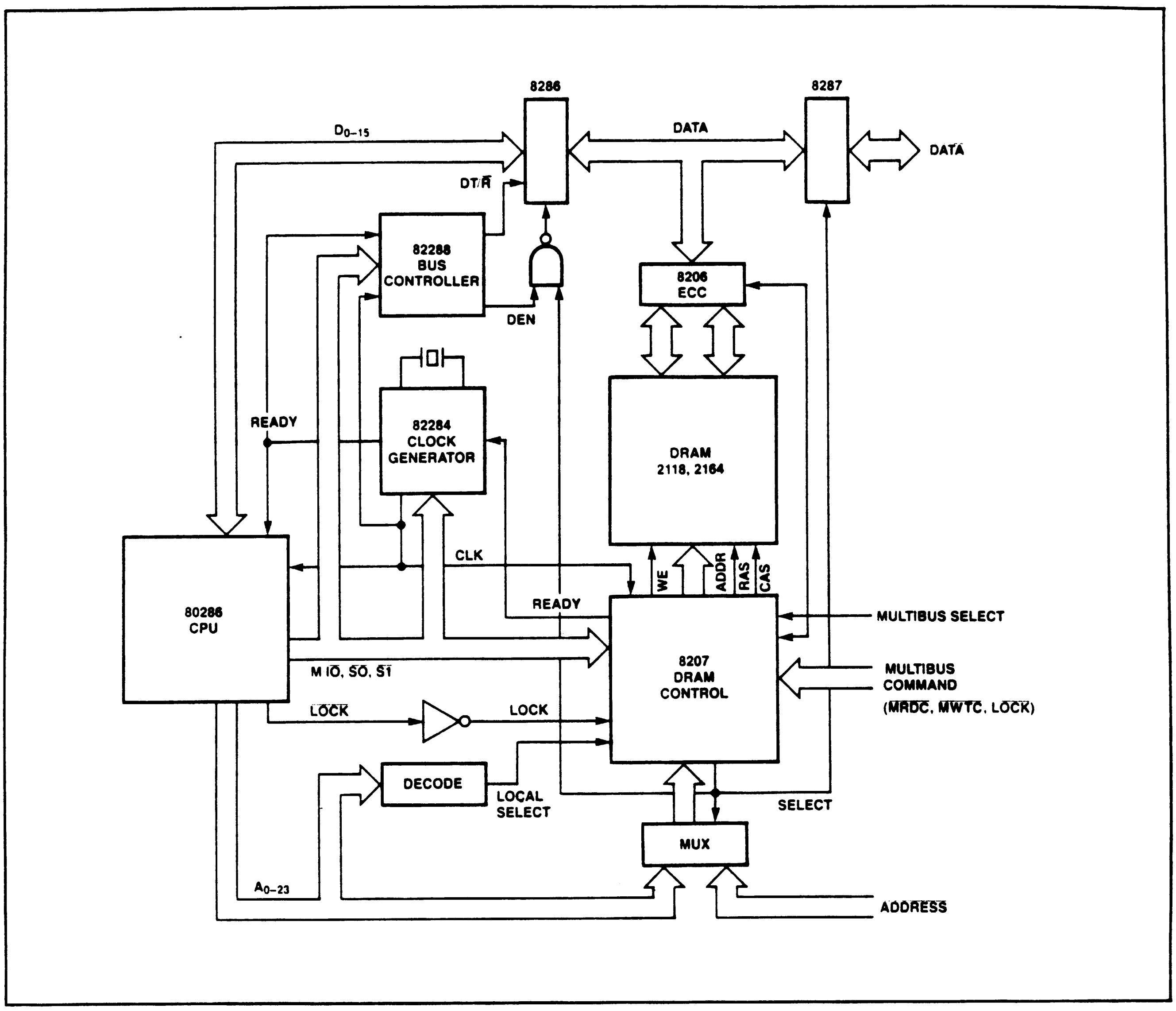


Figure 33. iAPX 286 System Configuration with Dual-Ported Memory

Figure 33 shows the addition of dual ported dynamic memory between the Multibus system bus and the iAPX 286 local bus. The dual port interface is provided by the 8207 Dual Port DRAM Controller. The 8207 runs synchronously with the CPU to maximize throughput for local memory references. It also arbitrates between requests from the local and system buses and performs functions such as refresh, initialization of RAM, and read/modify/write cycles. The 8207 combined with the 8206 Error Checking and Correction memory controller provide for single bit error correction. The dual-ported memory can be combined with a standard Multibus system bus interface to maximize performance and protection in multiprocessor system configurations.

PACKAGE

The 80286 is packaged in a 68-pin, leadless JEDEC type A hermetic chip carrier. Figure 34 illustrates the package, and Figure 2 shows the pinout.

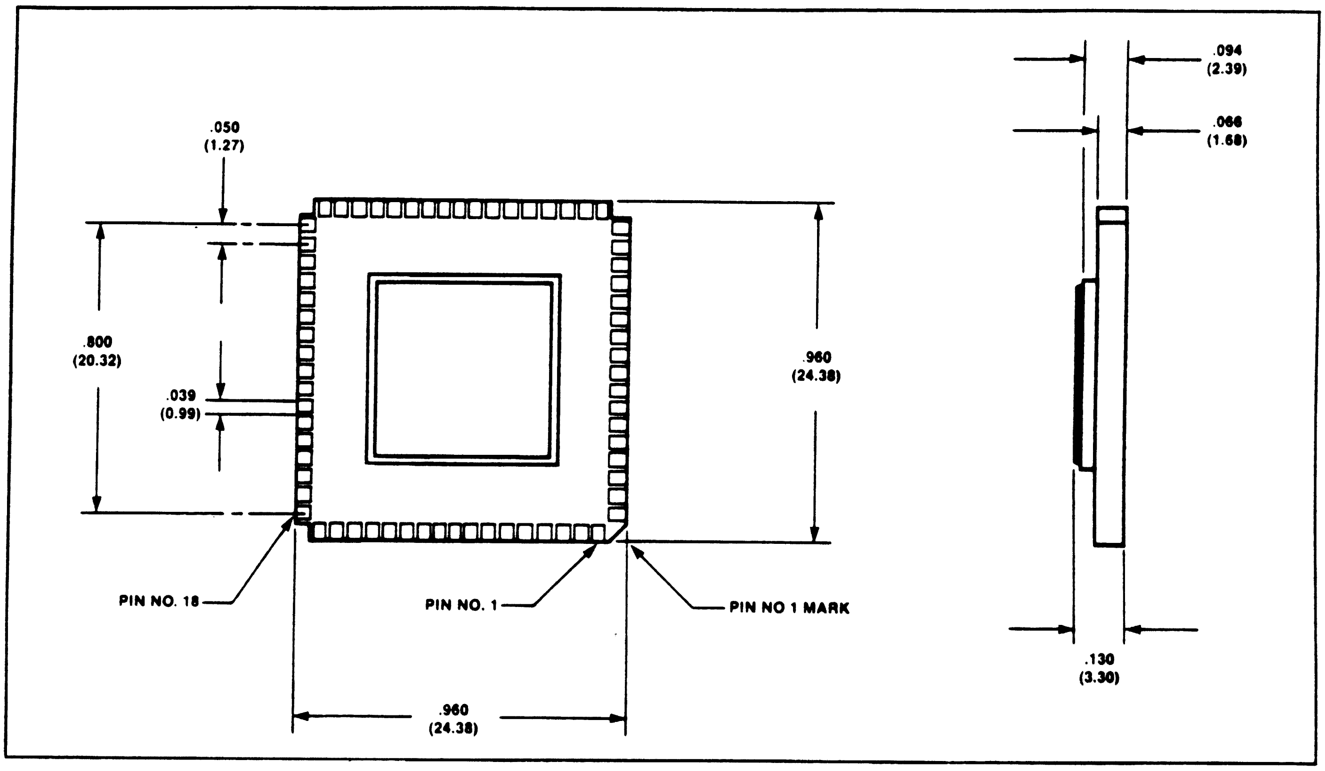


Figure 34. 80286 JEDEC Type A Package

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias	0°C to 70°C
Storage Temperature	−65°C to +150°C
Voltage on Any Pin with Respect to Ground	−0.3 to +7V
Power Dissipation	3.6 Watt

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

D.C. CHARACTERISTICS (80286: T_A = 0°C to 70°C, V_{CC} = 5V ± 10%)

Symbol	Parameter	Min.	Max.	Units	Test Conditions
V_{IL}	Input Low Voltage	−0.5	+0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC} + 0.5	V	
V_{OL}	Output Low Voltage		0.45	V	I_{OL} = 3.0 mA
V_{OH}	Output High Voltage	2.4		V	I_{OH} = −400 μA
I_{CC}	Power Supply Current		600	mA	T_A = 25°C
I_{LI}	Input Leakage Current		±10	μA	0V ≤ V_{IN} ≤ V_{CC}
I_{LO}	Output Leakage Current		±10	μA	0.45V ≤ V_{OUT} ≤ V_{CC}
V_{CL}	Clock Input Low voltage	−0.5	+0.6	V	
V_{CH}	Clock Input High Voltage	3.8	V_{CC} + 1.0	V	
C_{IN}	Capacitance of Inputs (All input except CLK)		10	pF	fc = 1 MHz
C_O	Capacitance of I/O or outputs		20	pF	fc = 1 MHz
C_{CLK}	Capacitance of CLK Input		12	pF	f_c = 1 MHz

A.C. CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = 5V ± 10%)

80286 Timing Requirements

Symbol	Parameter	Min.	Max.	Units	Test Conditions
1	System clock period	62.5	250	ns	
2	System clock low time	15	230	ns	at .6 Volts
3	System clock high time	20	235	ns	at 3.2 Volts
4	Asynchronous input setup time	20		ns	See note 1
5	Asynchronous input hold time	20		ns	See note 1
6	RESET setup time	20		ns	
7	RESET hold time	0		ns	
8	Read data in setup time	10		ns	
9	Read data in hold time	5		ns	
10	READY setup time	38.5		ns	
11	READY hold time	25		nx	
12	STATUS/$\overline{\text{PEACK}}$ valid delay	0	40	ns	
13	Address valid delay	0	60	ns	
14	Write data valid delay	0	50	ns	C_L = 100 Pfd max
15	Address/Status/Data float delay	0	60	ns	
16	HLDA valid delay	0	60	ns	

82284 Timing Requirements

Symbol	Parameter	Min.	Max.	Units	Test Conditions
17	$\overline{\text{SRDY}}$/$\overline{\text{SRDYEN}}$ setup time	15		ns	
18	$\overline{\text{SRDY}}$/$\overline{\text{SRDYEN}}$ hold time	0		ns	
19	$\overline{\text{ARDY}}$/$\overline{\text{ARDYEN}}$ setup time	−5		ns	See note 1
20	$\overline{\text{ARDY}}$/$\overline{\text{ARDYEN}}$ hold time	16		ns	See note 1.
21	PCLK delay	0	40	ns	C_L = 75 pfd I_{OL} = 5.25 ma I_{OH} = −1.05 ma

NOTE 1: These times are given for testing purposes to assure a predetermined action.

82288 Timing Requirements

Symbol	Parameter	Min.	Max.	Units	Test Conditions
22	CMDLY setup time	20		ns	
23	CMDLY hold time	0		ns	
24	Command delay	5	25	ns	C_L = 300 pfd max I_{OL} = 32 ma max I_{OH} = −5 ma max
25	ALE active delay	2.5	12	ns	
26	ALE inactive delay	0	15	ns	
27	DT/$\overline{\text{R}}$ read active delay	0	25	ns	
28	DT/$\overline{\text{R}}$ read inactive delay	15	40	ns	C_L = 80 pfd max I_{OL} = 16 ma max I_{OH} = −1 ma max
29	DEN read active delay	10	50	ns	
30	DEN read inactive delay	2.5	20	ns	
31	DEN write active delay	17.5	40	ns	
32	DEN write inactive delay	12.5	47.5	ns	

intel ADVANCE INFORMATION

8207 ADVANCED DYNAMIC RAM CONTROLLER

- Provides All Signals Necessary to Control 16K (2118), 64K (2164A) and 256K Dynamic RAMs
- Directly Addresses and Drives up to 2 Megabytes without External Drivers
- Supports Single and Dual-Port Configurations
- Automatic RAM Initialization in All Modes
- Five Programmable Refresh Modes
- Transparent Memory Scrubbing in ECC Mode
- Supports Intel iAPX 86, 88, 186, and 286 Microprocessors
- Data Transfer Acknowledge Signals for Each Port
- Provides Signals to Directly Control the 8206 Error Detection and Correction Unit
- Supports Synchronous or Asynchronous Operation on Either Port
- +5 Volt Only HMOSII Technology for High Performance and Low Power

The Intel 8207 Advanced Dynamic RAM Controller (ADRC) is a high-performance, systems-oriented, Dynamic RAM controller that is designed to easily interface 16K, 64K and 256K Dynamic RAMs to Intel and other microprocessor Systems. A dual-port interface allows two different busses to independently access memory. When configured with an 8206 Error Detection and Correction Unit the 8207 supplies the necessary logic for designing large error-corrected memory arrays. This combination provides automatic memory initialization and transparent memory error scrubbing.

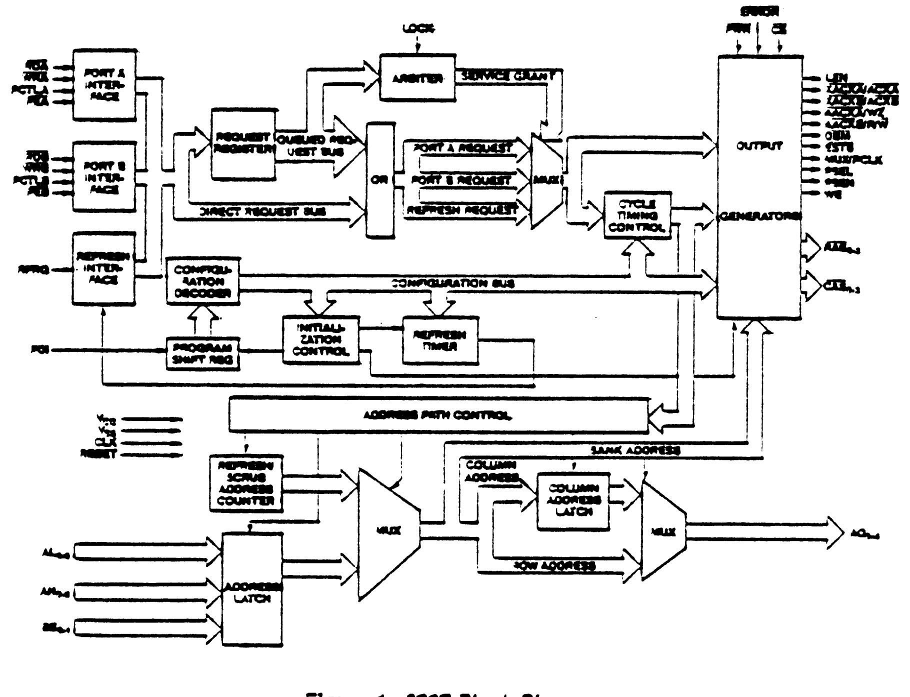


Figure 1. 8207 Block Diagram

APRIL, 1982
ORDER NUMBER: 210463-001

NOTICE: The pin descriptions are not final specifications and are subject to change.

Table 1. Pin Description

Symbol	Pin	Type	Name and Function
LEN	1	O	**ADDRESS LATCH ENABLE:** In two-port configurations, when port A is running with iAPX 286 Status interface mode, this output replaces the ALE signal from the system bus controller and generates an address latch enable signal which provides optimum setup and hold timing for the 8207.
$\overline{\text{XACKA}}$/ $\overline{\text{ACKA}}$	2	O	**TRANSFER ACKNOWLEDGE PORT A/ACKNOWLEDGE PORT A:** In non-ECC mode, this pin is $\overline{\text{XACKA}}$ and indicates that data on the bus is valid during a read cycle or that data may be removed from the bus during a write cycle for Port A. $\overline{\text{XACKA}}$ is a Multibus-compatible signal. In ECC mode, this pin is $\overline{\text{ACKA}}$ which can be configured, depending on the programming of the X program bit, as an XACK or AACK strobe. The SA programming bit determines whether AACK will be early or late.
$\overline{\text{XACKB}}$/ $\overline{\text{ACKB}}$	3	O	**TRANSFER ACKNOWLEDGE PORT B/ACKNOWLEDGE PORT B:** In non-ECC mode, this pin is $\overline{\text{XACKB}}$ and indicates that data on the bus is valid during a read cycle or that data may be removed from the bus during a write cycle for Port B. $\overline{\text{XACKB}}$ is a Multibus-compatible signal. In ECC mode, this pin is $\overline{\text{ACKB}}$ which can be configured, depending on the programming of the X program bit, as an XACK or AACK strobe. The SB programming bit determines whether AACK will be early or late.
$\overline{\text{AACKA}}$/ $\overline{\text{WZ}}$	4	O	**ADVANCED ACKNOWLEDGE PORT A/WRITE ZERO:** In non-ECC mode, this pin is $\overline{\text{AACKA}}$ and indicates that the processor may continue processing and that data will be available when required. This signal is optimized for the system by programming the SA program bit for syncnronous or asynchronous operation. After a RESET, this signal will cause the 8206 to force the data to all zeros and generate the appropriate check bits.
$\overline{\text{AACKB}}$/ R/$\overline{\text{W}}$	5	O	**ADVANCED ACKNOWLEDGE PORT B/READ/WRITE:** In non-ECC mode, this pin is $\overline{\text{AACKB}}$ and indicates that the processor may continue processing and that data will be available when required. This signal is optimized for the system by programming the SB program bit for synchronous or asynchronous operation. This signal causes the 8206 EDCU to latch the syndrome and error flags and generate check bits.
$\overline{\text{DBM}}$	6	O	**DISABLE BYTE MARKS:** This is an ECC control output signal indicating that a read or refresh cycle is occurring. This output forces the byte address decoding logic to enable all 8206 data output buffers. In ECC mode, this output is also asserted during memory initialization and the 8-cycle dynamic RAM wake-up exercise.
$\overline{\text{ESTB}}$	7	O	**ERROR STROBE:** In ECC mode, this strobe is activated when an error is detected and allows a negative-edge triggered flip-flop to latch the status of the 8206 EDCU CE for systems with error logging capabilities.
LOCK	8	I	**LOCK:** This input instructs the 8207 to lock out the port not being serviced at the time LOCK was issued.
V_{CC}	9 43	I I	**LOGIC POWER:** +5 Volts ± 10%. Supplies V_{CC} for the internal logic circuits. **DRIVER POWER:** −5 Volts ± 10%. Supplies V_{CC} for the output drivers.
CE	10	I	**CORRECTABLE ERROR:** This is an ECC input from the 8206 EDCU which instructs the 8207 whether a detected error is correctable or not. A high input indicates a correctable error. A low input inhibits the 8207 from activating WE to write the data back into RAM. This should be connected to the CE output of the 8206.
$\overline{\text{ERROR}}$	11	I	**ERROR:** This is an ECC input from the 8206 EDCU and instructs the 8207 that an error was detected. This pin should be connected to the ERROR output of the 8206.
MUX/ PCLK	12	O	**MULTIPLEXER CONTROL/PROGRAMMING CLOCK:** Immediately after a RESET this pin is used to clock serial programming data into the PDI pin. In normal two-port operation, this pin is used to select memory addresses from the appropriate port. When this signal is high, port A is selected and when it is low, port B is selected. This signal may change state before the completion of a RAM cycle, but the RAM address hold time is satisfied.
PSEL	13	O	**PORT SELECT:** This signal is used to select the appropriate port for data transfer.
PSEN	14	O	**PORT SELECT ENABLE:** This signal used in conjunction with PSEL provides contention-free port exchange. When PSEN is low, PSEL is allowed to change state.
WE	15	O	**WRITE ENABLE:** This signal provides the dynamic RAM array the write enable input for a write operation.

NOTICE: The pin descriptions are not final specifications and are subject to change.

Table 1. Pin Description (Continued)

Symbol	Pin	Type	Name and Function
$\overline{\text{FWR}}$	16	I	**FULL WRITE:** This is an ECC input signal that instructs the 8207, in an ECC configuration, whether the present write cycle is normal RAM write (full write) or a RAM partial write (read-modify-write) cycle.
RESET	17	I	**RESET:** This signal causes all internal counters and state flip-flops to be reset and upon release of RESET, data appearing at the PDI pin is clocked in by the PCLK output. The states of the PDI, PCTLA, PCTLB and RFRQ pins are sampled by RESET going inactive and are used to program the 8207.
$\overline{\text{CAS0}}$ $\overline{\text{CAS1}}$ $\overline{\text{CAS2}}$ $\overline{\text{CAS3}}$	18 20 22 24	O O O O	**COLUMN ADDRESS STROBE:** These outputs are used by the dynamic RAM array to latch the column address, present on the AO0–8 pins. These outputs are selected by the BS0 and BS1 as programmed by program bits RB0 and RB1. These outputs drive the dynamic RAM array directly and need no external drivers.
$\overline{\text{RAS0}}$ $\overline{\text{RAS1}}$ $\overline{\text{RAS2}}$ $\overline{\text{RAS3}}$	19 21 23 25	O O O O	**ROW ADDRESS STROBE:** These outputs are used by the dynamic RAM array to latch the row address, present on the AO0–8 pins. These ouptuts are selected by the BS0 and BS1 as programmed by program bits RB0 and RB1. These outputs drive the dynamic RAM array directly and need no external drivers.
V_{SS}	26 60	I I	**DRIVER GROUND:** Provides a ground for the output drivers. **LOGIC GROUND:** Provides a ground for the remainder of the device.
AO0 AO1 AO2 AO3 AO4 AO5 AO6 AO7 AO8	35 34 33 32 31 30 29 28 27	O O O O O O O O O	**ADDRESS OUTPUTS:** These outputs are designed to provide the row and column addresses of the selected port to the dynamic RAM array. These outputs drive the dynamic RAM array directly and need no external drivers.
BS0 BS1	36 37	I I	**BANK SELECT:** These inputs are used to select one of four banks of the dynamic RAM array as defined by the program bits RB0 and RB1.
AL0 AL1 AL2 AL3 AL4 AL5 AL6 AL7 AL8	38 39 40 41 42 44 45 46 47	I I I I I I I I I	**ADDRESS LOW:** These lower-order address inputs are used to generate the row address for the internal address multiplexer.
AH0 AH1 AH2 AH3 AH4 AH5 AH6 AH7 AH8	48 49 50 51 52 53 54 55 56	I I I I I I I I I	**ADDRESS HIGH:** These higher-order address inputs are used to generate the column address for the internal address multiplexer.
PDI	57	I	**PROGRAM DATA INPUT:** This input programs the various user-selectable options in the 8207. The PCLK pin shifts programming data into the PDI input from optional external shift registers. This pin may be strapped high or low to a default ECC (PDI = V_{CC}) or non-ECC (PDI = Ground) mode configuration.
RFRQ	58	I	**REFRESH REQUEST:** This input is sampled on the falling edge of RESET. If it is high at RESET, then the 8207 is programmed for internal refresh request or external refresh request with failsafe protection. If it is low at RESET, then the 8207 is programmed for external refresh without failsafe protection or burst refresh. Once programmed the RFRQ pin accepts signals to start an external refresh with failsafe protection or external refresh without failsafe protection or a burst refresh.

NOTICE: The pin descriptions are not final specifications and are subject to change.

Table 1. Pin Description (Continued)

Symbol	Pin	Type	Name and Function
CLK	59	I	**CLOCK:** This input provides the basic timing for sequencing the internal logic.
$\overline{RDB}$	61	I	**READ FOR PORT B:** This pin is the read memory request command input for port B. This input also directly accepts the $\overline{S1}$ status line from Intel processors.
$\overline{WRB}$	62	I	**WRITE FOR PORT B:** This pin is the write memory request command input for port B. This input also directly accepts the $\overline{S0}$ status line from Intel processors.
$\overline{PEB}$	63	I	**PORT ENABLE FOR PORT B:** This pin serves to enable a RAM cycle request for port B. It is generally decoded from the port address.
PCTLB	64	I	**PORT CONTROL FOR PORT B:** This pin is sampled on the falling edge of RESET. It configures port B to accept command inputs or processor status inputs. If low after RESET, the 8207 is programmed to accept command or iAPX 286 status inputs or Multibus commands. If high after RESET, the 8207 is programmed to accept status inputs from iAPX 86 or iAPX 186 processors. The $\overline{S2}$ status line should be connected to this input if programmed to accept iAPX 86 or iAPX 186 status inputs. When programmed to accept commands or iAPX 286 status, it should be tied low or it may be used as a Multibus-compatible inhibit signal.
$\overline{RDA}$	65	I	**READ FOR PORT A:** This pin is the read memory request command input for port A. This input also directly accepts the $\overline{S1}$ status line from Intel processors.
$\overline{WRA}$	66	I	**WRITE FOR PORT A:** This pin is the write memory request command input for port A. This input also directly accepts the $\overline{S0}$ status line from Intel processors.
$\overline{PEA}$	67	I	**PORT ENABLE FOR PORT A:** This pin serves to enable a RAM cycle request for port A. It is generally decoded from the port address.
PCTLA	68	I	**PORT CONTROL FOR PORT A:** This pin is sampled on the falling edge of RESET. It configures port A to accept command inputs or processor status inputs. If low after RESET, the 8207 is programmed to accept command or iAPX 286 status inputs or Multibus commands. If high after RESET, the 8207 is programmed to accept status inputs from iAPX 86 or iAPX 186 processors. The $\overline{S2}$ status line should be connected to this input if programmed to accept iAPX 86 or iAPX 186 status inputs. When programmed to accept commands or iAPX 286 status, it should be tied low or it may be connected to INHIBIT when operating with Multibus.

GENERAL DESCRIPTION

The Intel 8207 Advanced Dynamic RAM Controller (ADRC) is a microcomputer peripheral device which provides the necessary signals to address, refresh and directly drive 16K, 64K and 256K dynamic RAMs. This controller also provides the necessary arbitration circuitry to support dual-port access of the dynamic RAM array.

The ADRC supports several microprocessor interface options including synchronous and asynchronous connection to iAPX 86, iAPX 88, iAPX 186, iAPX 286 and Multibus.

This device may be used with the 8206 Error Detection and Correction Unit (EDCU). When used with the 8206, the 8207 is programmed in the Error Checking and Correction (ECC) mode. In this mode, the 8207 provides all the necessary control signals for the 8206 to perform memory initialization and transparent error scrubbing during refresh.

FUNCTIONAL DESCRIPTION

Processor Interface

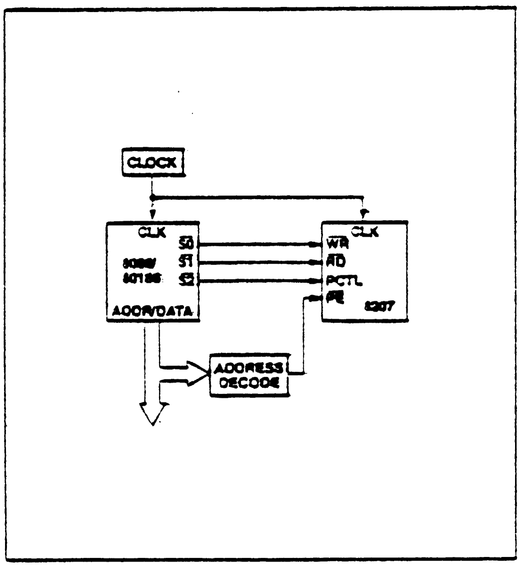
The 8207 has control circuitry for two ports each capable of supporting one of several possible bus structures. The ports are independently configurable allowing the dynamic RAM to serve as an interface between two different bus structures.

Each port of the 8207 may be programmed to run synchronous or asynchronous to the processor clock. (See Synchronous/Asynchronous Mode) The 8207 has been optimized to run synchronously with Intel's iAPX 86, iAPX 88, iAPX 186 and iAPX 286. When the 8207 is programmed to run in asynchronous mode, the 8207 inserts the necessary synchronization circuitry for the $\overline{RD}$, $\overline{WR}$, $\overline{PE}$, and PCTL inputs.

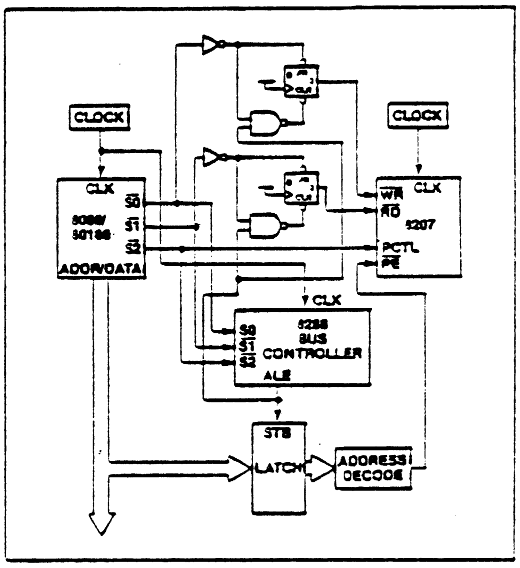
The 8207 can also decode the status lines directly from the iAPX 86, iAPX 88, iAPX 186 and the iAPX 286 or can be programmed to receive read or write Multibus commands or commands from a bus controller. (See Status/Command Mode)

The 8207 may be programmed to accept the clock of the iAPX 86, 88, 186, or 286. The 8207 adjusts its internal timing to allow for the different clock frequencies of these microprocessors. (See Microprocessor Clock Frequency Option)

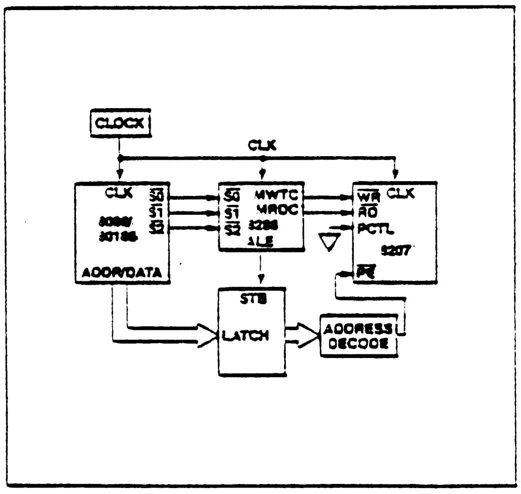
Figure 2 shows the different processor interfaces to the 8207 using the synchronous or asynchronous mode and status or command interface.



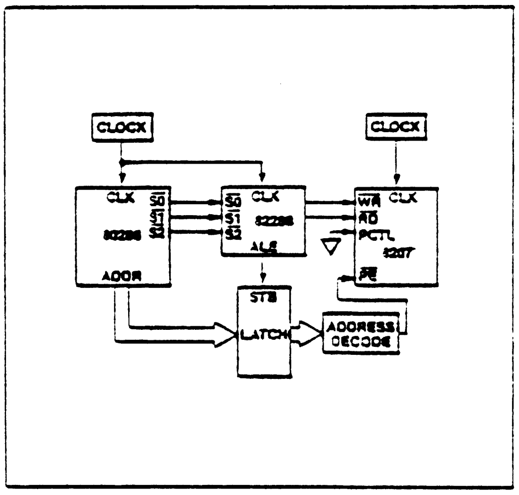
Slow-Cycle Synchronous-Status Interface



Slow-Cycle Asynchronous-Status Interface



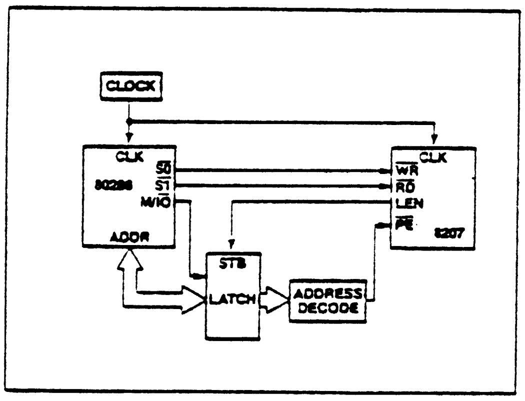
Slow-Cycle Synchronous-Command Interface



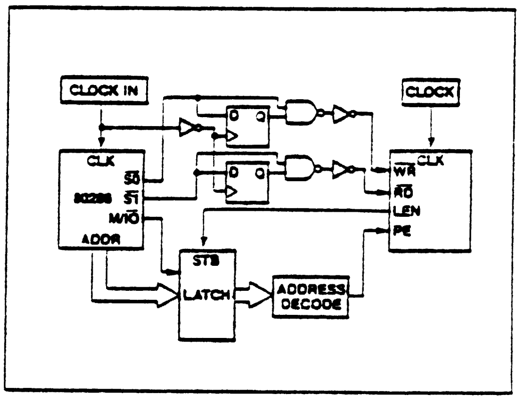
Slow-Cycle Asynchronous-Command Interface

Figure 2A. Slow-cycle Port Interfaces Supported by the 8207

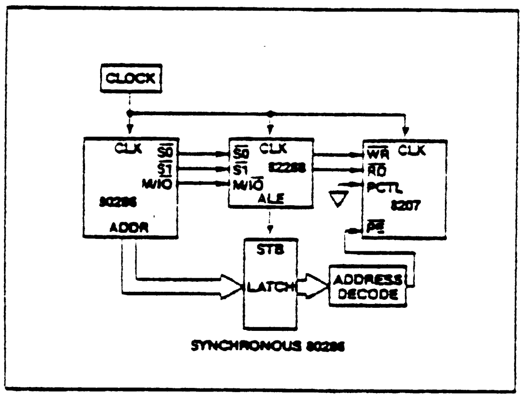
AFN-02218A



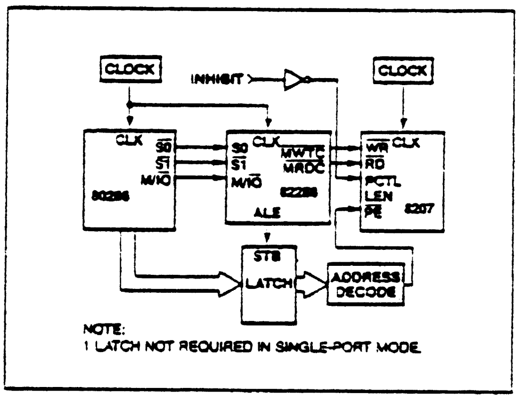
Fast-Cycle Synchronous-Status Interface



Fast-Cycle Asynchronous-Status Interface



Fast-Cycle Synchronous-Command Interface



Fast-Cycle Asynchronous-Command Interface

Figure 2B. Fast-cycle Port Interfaces Supported by the 8207

Dual-Port Operation

The 8207 provides for two-port operation. Two independent processors may access memory controlled by the 8207. The 8207 arbitrates between each of the processor requests and directs data to or from the appropriate port. Selection is done on a priority concept that reassigns priorities based upon past history. Processor requests are internally queued.

Figure 3 shows a dual-port configuration with two iAPX 86 systems interfacing to dynamic RAM. One of the processor systems is interfaced synchronously using the status interface and the other is interfaced asynchronously also using the status interface.

Dynamic RAM Interface

The 8207 is capable of addressing 16K, 64K and 256K dynamic RAMs. Figure 4 shows the connection of the processor address bus to the 8207 using the different RAMs. The 8207 directly supports the 2118 RAM family or any RAM with similar timing requirements and responses including the Intel 2164A RAM.

The 8207 divides memory into four banks, each bank having its own Row ($\overline{RAS}$) and Column ($\overline{CAS}$) Address Strobe pair. This organization permits RAM cycle interleaving and permits error scrubbing during ECC refresh cycles. RAM cycle interleaving overlaps the start of the next RAM cycle with the RAM Precharge period of the previous cycle. Hiding the

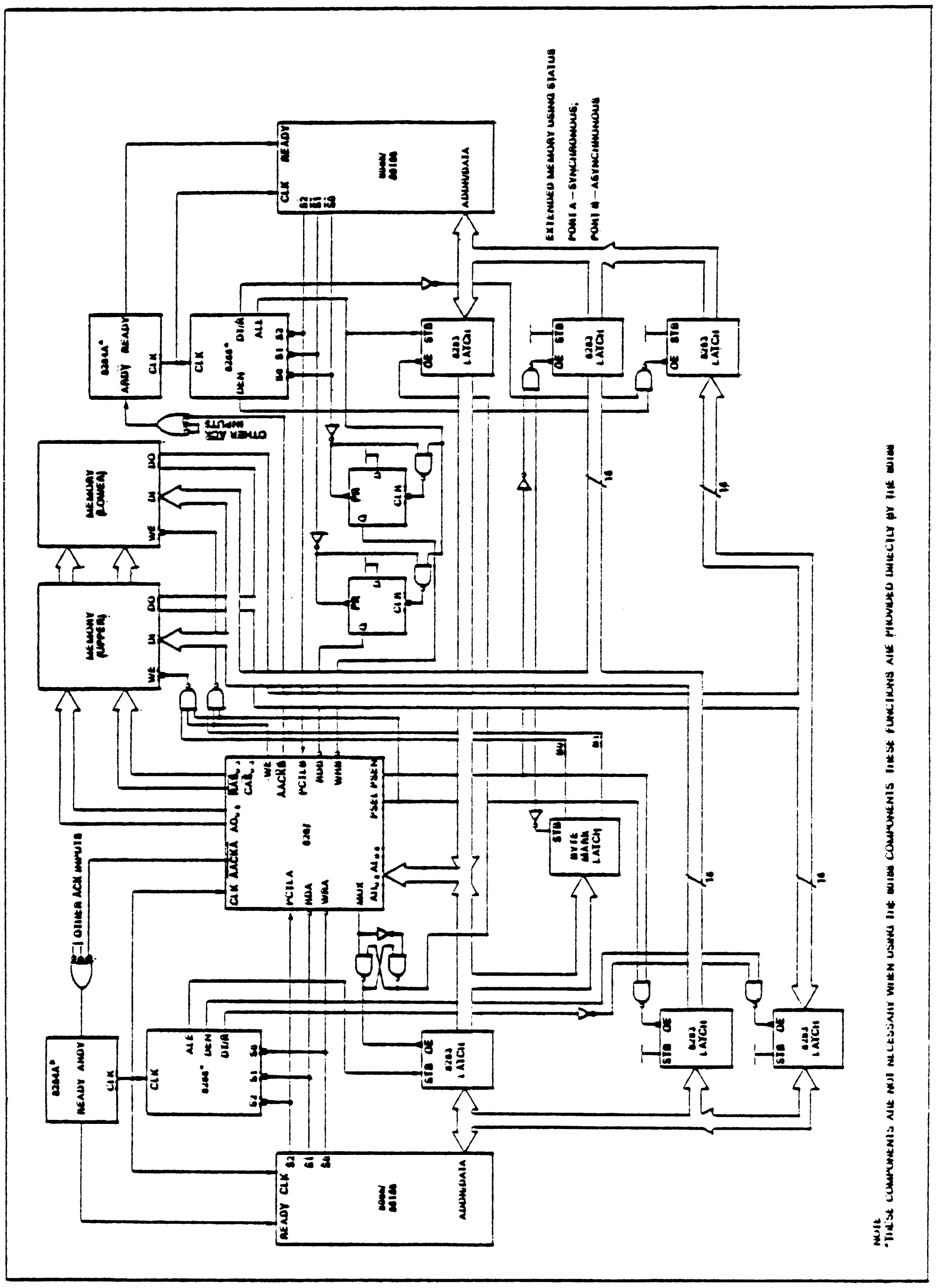
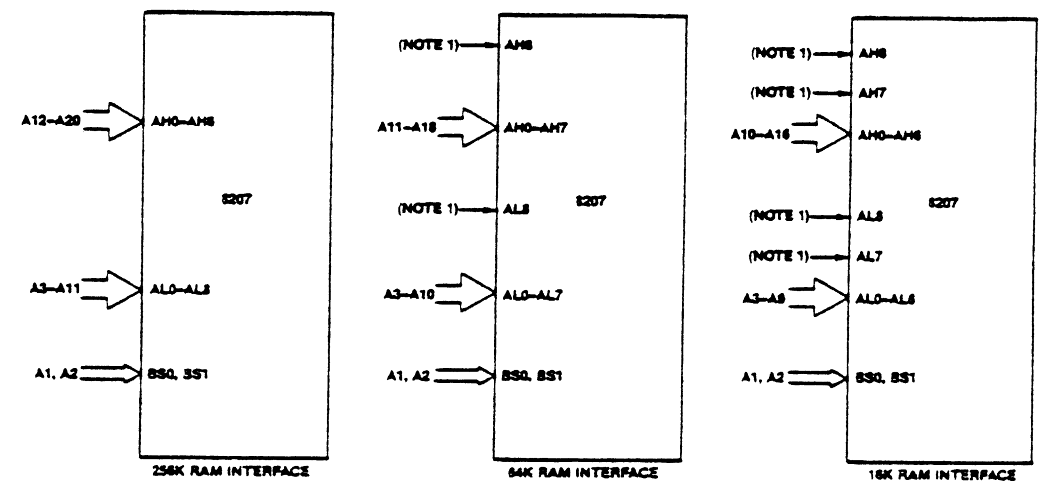


Figure 3. 8086/80186 Dual Port System



NOTES:
[1] UNASSIGNED ADDRESS INPUT PINS SHOULD BE STRAPPED HIGH OR LOW.
[2] A0 ALONG WITH BHE ARE USED TO SELECT A BYTE WITHIN A PROCESSOR WORD.
[3] LOW ORDER ADDRESS BITS ARE USED AS BANK SELECT INPUTS SO THAT CONSECUTIVE MEMORY ACCESS REQUESTS ARE TO ALTERNATE BANKS ALLOWING BANK INTERLEAVING OF MEMORY CYCLES.

Figure 4. Processor Address Interface to the 8207 Using 16K, 64K, and 256K RAMS

precharge period of one RAM cycle behind the data access period of the next RAM cycle optimizes memory bandwidth and is effective as long as successive RAM cycles occur in alternate banks.

Successive data access to the same bank will cause the 8207 to wait for the precharge time of the previous RAM cycle.

If not all RAM banks are occupied, the 8207 reassigns the $\overline{RAS}$ and $\overline{CAS}$ strobes to allow using wider data words without increasing the loading on the $\overline{RAS}$ and $\overline{CAS}$ drivers. Table 2 shows the bank selection decoding and the word expansion, including $\overline{RAS}$ and $\overline{CAS}$ assignments. For example, if only two RAM banks are occupied, then two $\overline{RAS}$ and two $\overline{CAS}$ strobes are activated per bank.

The 8207 can interface to fast (e.g., 2118-10) or slow (e.g., 2118-15) RAMs. The 8207 adjusts and optimizes internal timings for either the fast or slow RAMs as programmed. (See RAM Speed Option)

Memory Initialization

After programming, the 8207 performs eight RAM "warm-up" cycles to prepare the dynamic RAM for proper device operation and, if configured for operation with error correction, the 8207 and 8206 EDCU will proceed to initialize all of memory (memory is written with zeros with corresponding check bits).

Table 2. Bank Selection Decoding and Word Expansion

Program Bits		Bank Input		
RB1	RB0	B1	B0	RAS/CAS Pair Allocation
0	0	0	0	RAS_{0-3}, CAS_{0-3} to Bank 0
0	0	0	1	Bank 1 unoccupied
0	0	1	0	Bank 2 unoccupied
0	0	1	1	Bank 3 unoccupied
0	1	0	0	$RAS_{0,1}$, $CAS_{0,1}$ to Bank 0
0	1	0	1	$RAS_{2,3}$, $CAS_{2,3}$ to Bank 1
0	1	1	0	Bank 2 unoccupied
0	1	1	1	Bank 3 unoccupied
1	0	0	0	RAS_0, CAS_0 to Bank 0
1	0	0	1	RAS_1, CAS_1 to Bank 1
1	0	1	0	RAS_2, CAS_2 to Bank 2
1	0	1	1	Bank 3 unoccupied
1	1	0	0	RAS_0, CAS_0 to Bank 0
1	1	0	1	RAS_1, CAS_1 to Bank 1
1	1	1	0	RAS_2, CAS_2 to Bank 2
1	1	1	1	RAS_3, CAS_3 to Bank 3

Because the time to initialize memory is fairly long, the 8207 may be programmed to skip initialization in ECC mode. The time required to initialize all of memory is dependent on the clock cycle time to the 8207 and can be calculated by the following equation:

eq.1 $T_{INIT} = (2^{23})\ T_{CY}$

if $T_{CY} = 125$ ns then $T_{INIT} \approx 1$ sec.

8206 ECC Interface

For operation with Error Checking and Correction (ECC), the 8207 adjusts its internal timing and changes some pin functions to optimize performance and provide a clean dual-port memory interface between the 8206 EDCU and memory. The 8207 directly supports a master-only (16-bit word plus 6 check bits) system. Under extended operation and reduced clock frequency, the 8207 will support any ECC master-slave configuration up to 80 data bits, which is the maximum set by the 8206 EDCU. (See Extend Option)

Correctable errors detected during memory read cycles are corrected immediately and then written back into memory.

In a synchronous bus environment, ECC system performance has been optimized to enhance processor throughput, while in an asynchronous bus environment (the Multibus), ECC performance has been optimized to get valid data onto the bus as quickly as possible. Performance optimization, processor throughput or quick data access may be selected via the Transfer Acknowledge Option.

The main difference between the two ECC implementations is that, when optimized for processor throughput, RAM data is always corrected and an advanced transfer acknowledge is issued at a point when, by knowing the processor characteristics, data is guaranteed to be valid by the time the processor needs it.

When optimized for quick data access, (valid for Multibus) the 8206 is configured in the uncorrecting mode where the delay associated with error correction circuitry is transparent, and a transfer acknowledge is issued as soon as valid data is known to exist. If the $\overline{\text{ERROR}}$ flag is activated, then the transfer acknowledge is delayed until after the 8207 has instructed the 8206 to correct the data and the corrected data becomes available on the bus. Figure 5 illustrates a dual-port ECC system.

Figure 6 illustrates the interface required to drive the $\overline{\text{CRCT}}$ pin of the 8206, in the case that one port (PORT A) receives an advanced acknowledge (not Multibus-compatible), while the other port (PORT B) receives $\overline{\text{XACK}}$ (which is Multibus-compatible).

Error Scrubbing

The 8207/8206 performs error correction during refresh cycles (error scrubbing). Since the 8207 must refresh RAM, performing error scrubbing during refresh allows it to be accomplished without additional performance penalties.

Upon detection of a correctable error during refresh, the RAM refresh cycle is lengthened slightly to permit the 8206 to correct the error and for the corrected word to be rewritten into memory. Uncorrectable errors detected during scrubbing are ignored.

Refresh

The 8207 provides an internal refresh interval counter and a refresh address counter to allow the 8207 to refresh memory. The 8207 will refresh 128 rows every 2 milliseconds or 256 rows every 4 milliseconds, which allows all RAM refresh options to be supported. In addition, there exists the ability to refresh 256 row address locations every 2 milliseconds via the Refresh Period programming option.

The 8207 may be programmed for any of five different refresh options: Internal refresh only, External refresh with failsafe protection, External refresh without failsafe protection, Burst Refresh mode, or no refresh. (See Refresh Options)

It is possible to decrease the refresh time interval by 10%, 20% or 30%. This option allows the 8207 to compensate for reduced clock frequencies. Note that an additional 5% interval shortening is built-in in all refresh interval options to compensate for clock variations and non-immediate response to the internally generated refresh request. (See Refresh Period Options)

External Refresh Requests after RESET

External refresh requests are not recognized by the 8207 until after it is finished programming and preparing memory for access. Memory preparation includes 8 RAM cycles to prepare and ensure proper

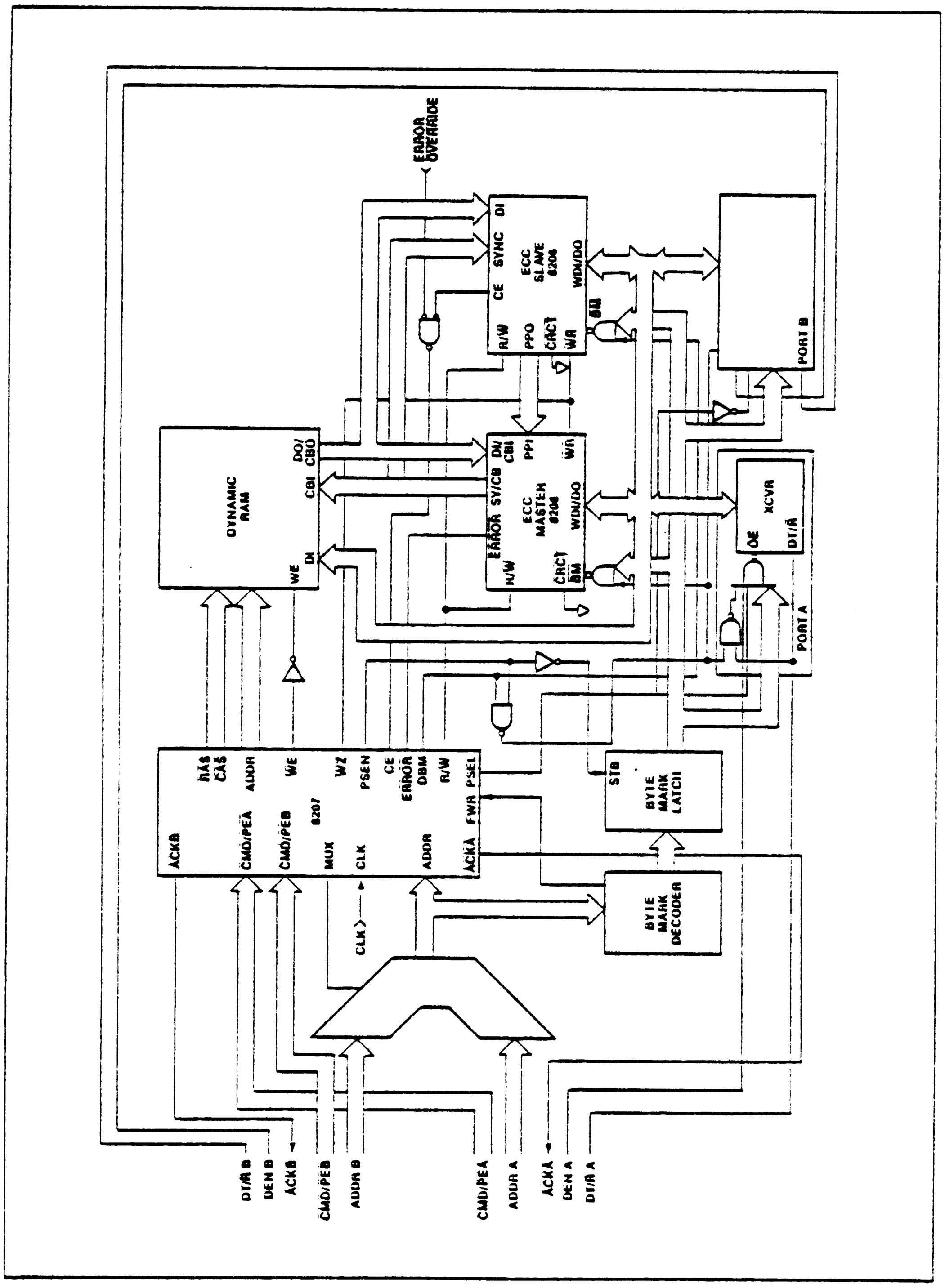


Figure 5. Two-Port ECC Implementation Using the 8207 and the 8206

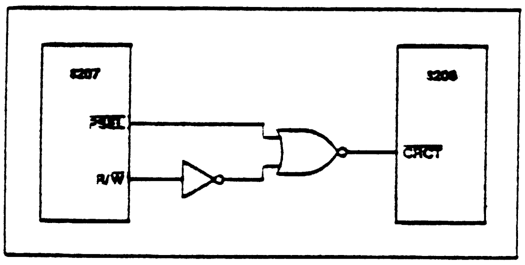


Figure 6. Interface to 8206 $\overline{\text{CRCT}}$ Input When Port A Receives $\overline{\text{AACK}}$ and Port B Receives $\overline{\text{XACK}}$

dynamic RAM operation, and memory initialization if error correction is used. Many dynamic RAMs require this warm-up period for proper operation. The time it takes for the 8207 to recognize a request is shown below.

eq. 2 Non-ECC Systems: $T_{RESP} = T_{PROG} + T_{PREP}$

eq. 3 where: $T_{PROG} = (66)\ (T_{CY})$ which is programming time

eq. 4 $T_{PREP} = (8)\ (32)\ (T_{CY})$ which is the RAM warm-up time

if $T_{CY} = 125$ ns then $T_{RESP} = 41$ us

eq. 5 ECC Systems: $T_{RESP} = T_{PROG} + T_{PREP} + T_{INIT}$

if $T_{CY} = 125$ ns then $T_{RESP} = 1$ sec

RESET

RESET is an asynchronous input, the falling edge of which is used by the 20 to directly sample the logic levels of the PCTLA, PCTLB, RFRQ, and PDI inputs. The internally synchronized falling edge of RESET is used to begin programming operations (shifting in the contents of the external shift register into the PDI input).

Until programming is complete the 8207 registers but does not respond to command or status inputs. A simple means of preventing commands or status from occurring during this period is to differentiate the system reset pulse to obtain a smaller reset pulse for the 8207. The total time of the reset pulse and the 8207 programming time must be less than the time before the first command in systems that alter the default port synchronization programming bits (default is Port A synchronous, Port B asynchronous). Differentiated reset is unnecessary when the default port synchronization programming is used.

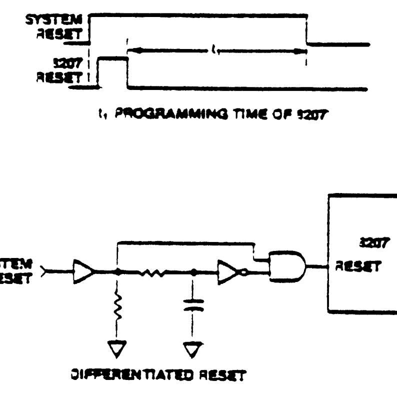
The differentiated reset pulse would be shorter than the system reset pulse by at least the programming period required by the 8207. The differentiated reset pulse first resets the 8207, and system reset would reset the rest of the system. While the rest of the system is still in reset, the 8207 completes its programming. Figure 7 illustrates a circuit to accomplish this task.

Within four clocks after RESET goes active, all the 8207 outputs will go high, except for PSEN, WE, and AO0-8, which will go low.

OPERATIONAL DESCRIPTION

Programming the 8207

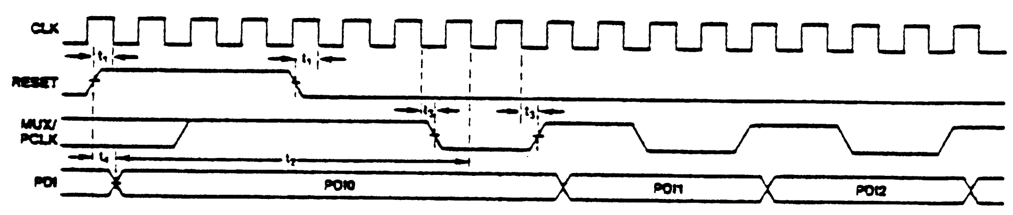
The 8207 is programmed after reset. On the falling edge of RESET, the logic states of several input pins are latched internally. The falling edge of RESET actually performs the latching, which means that the logic levels on these inputs must be stable prior to that time. The inputs whose logic levels are latched at the end of reset are the PCTLA, PCTLB, REFRQ, and PDI pins. Figure 3 shows the necessary timing for programming the 8207.



NOTES:
(1) REQUIRED ONLY WHEN THE PORT SYNCHRONIZATION OPTIONS (SA & SB) ARE ALTERED FROM THEIR INITIAL DEFAULT VALUES.
(2) V_{CC} MUST BE STABLE BEFORE SYSTEM RESET IS ACTIVATED WHEN USING THIS CIRCUIT.

Figure 7. 8207 Differentiated Reset Circuit

AFN-02218A



NOTES:
t_1—RESET IS AN ASYNCHRONOUS INPUT. IF RESET OCCURS BEFORE t_1, THEN IT IS GUARANTEED TO BE RECOGNIZED.
t_2—MINIMUM PDI VALID TIME PRIOR TO CLOCK EDGE THAT RECOGNIZES END OF RESET.
t_3—MUX/PCLK DELAY.
t_4—ASYNCHRONOUS LOAD DATA PROPAGATION DELAY.

Figure 8. Timing Illustrating External Shift Register Requirements for Programming the 8207

Status/Command Mode

The two processor ports of the 8207 are configured by the states of the PCTLA and PCTLB pins. Which interface is selected depends on the state of the individual port's PCTL pin at the end of reset. If PCTL is high at the end of the reset, the 8086 Status interface is selected; if it is low, then the Command interface is selected.

The status lines of the 80286 are similar in code and timing to the Multibus command lines, while the status code and timing of the 8086 and 8088 are identical to those of the 80186 (ignoring the differences in clock duty cycle). Thus there exists two interface configurations, one for the 80286 status or Multibus memory commands, which is called the Command interface, and one for 8086, 8088 or 80186 status, called the 8086 Status interface. The Command interface can also directly interface to the command lines of the bus controllers for the 8086, 8088, 80186 and the 80286.

The 8086 Status interface allows direct decoding of the status of the iAPX 86, iAPX 88, and the iAPX 186. Table 3 shows how the status lines are decoded. While in the Command mode the iAPX 286 status can be directly decoded. Microprocessor bus controller read or write commands or Multibus commands can also be directed to the 8207 when in Command mode.

Refresh Options

Immediately after system reset, the state of the REFRQ input pin is examined. If REFRQ is high, the 8207 provides the user with the choice between self-refresh or user-generated refresh with failsafe protection. Failsafe protection guarantees that if the

Table 3A. Status Coding of 8086, 80186 and 80286

Status Code			Function	
$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	8086/80186	80286
0	0	0	INTERRUPT	INTERRUPT
0	0	1	I/O READ	I/O READ
0	1	0	I/O WRITE	I/O WRITE
0	1	1	HALT	IDLE
1	0	0	INSTRUCTION FETCH	HALT
1	0	1	MEMORY READ	MEMORY READ
1	1	0	MEMORY WRITE	MEMORY WRITE
1	1	1	IDLE	IDLE

Table 3B. 8207 Response

8207 Command			Function	
PCTL	$\overline{RD}$	$\overline{WR}$	8086 Status Interface	Command Interface
0	0	0	IGNORE	IGNORE
0	0	1	IGNORE	READ
0	1	0	IGNORE	WRITE
0	1	1	IGNORE	IGNORE
1	0	0	READ	IGNORE
1	0	1	READ	INHIBIT
1	1	0	WRITE	INHIBIT
1	1	1	IGNORE	IGNORE

APN-02218A

user does not come back with another refresh request before the internal refresh interval counter times out, a refresh request will be automatically generated. If the REFRQ pin is low immediately after a reset, then the user has the choice of a single external refresh cycle without failsafe, burst refresh or no refresh.

Internal Refresh Only

For the 8207 to generate internal refresh requests, it is necessary only to strap the REFRQ input pin high.

External Refresh with Failsafe

To allow user-generated refresh requests with failsafe protection, it is necessary to hold the REFRQ input high until after reset. Thereafter, a low-to-high transition on this input causes a refresh request to be generated and the internal refresh interval counter to be reset. A high-to-low transition has no effect on the 8207. A refresh request is not recognized until a previous request has been serviced.

External Refresh without Failsafe

To generate single external refresh requests without failsafe protection, it is necessary to hold REFRQ low until after reset. Thereafter, bringing REFRQ high for one clock period causes a refresh request to be generated. A refresh request is not recognized until a previous request has been serviced.

Burst Refresh

Burst refresh is implemented through the same procedure as a single external refresh without failsafe (i.e., REFRQ is kept low until after reset). Thereafter, bringing REFRQ high for a least two clock periods causes a burst of 128 row address locations to be refreshed.

In ECC-configured systems, 128 locations are scrubbed. A burst refresh request is not recognized until a previous request has been serviced.

No Refresh

It is necessary to hold REFRQ low until after reset. This is the same as programming External Refresh without Failsafe. No refresh is accomplished by keeping REFRQ low.

Option Program Data Word

The program data word consists of 16 program data bits, PD0–PD15. If the first program data bit PD0 is set to logic 1, the 8207 is configured to support ECC. If it is logic 0, the 8207 is configured to support a non-ECC system. The remaining bits, PD1–PD15, may then be programmed to optimize a selected configuration. Figures 9 and 10 show the Program word for non-ECC and ECC operation.

Using an External Shift Register

The 8207 may be configured to use an external shift register with asynchronous load capability such as a 74LS165. The reset pulse serves to parallel load the shift register and the 8207 supplies the clocking signal to shift the data in. Figure 11 shows a sample circuit diagram of an external shift register circuit. Serial data is shifted into the 8207 via the PDI pin (57), and clock is provided by the MUX/PCLK pin (12), which generates a total of 16 clock pulses. After programming is complete, data appearing at the input of the PDI pin is ignored. MUX/PCLK is a dual-function pin. During programming, it serves to clock the external shift register, and after programming is completed, it reverts to a MUX control pin. As the pin changes state to select different port addresses, it continues to clock the shift register. This does not present a problem because data at the PDI pin is ignored after programming. Figure 8 illustrates the timing requirements of the shift register circuitry.

ECC Mode (ECC Program Bit)

The state of PDI (Program Data In) pin at reset determines whether the system is an ECC or non-ECC configuration. It is used internally by the 8207 to begin configuring timing circuits, even before programming is completely finished. The 8207 then begins programming the rest of the options.

Default Programming Options

After reset, the 8207 serially shifts in a program data word via the PDI pin. This pin may be strapped either high or low, or connected to an external shift register. Strapping PDI high causes the 8207 to default to a particular system configuration with error correction, and strapping it low causes the 8207 to default to a particular system configuration without error correction. Table 4 shows the default configurations.

PO15							PO8	PO7							PO0
0	0	TM 1	$\overline{PPR}$	$\overline{FFS}$	EXT	$\overline{PLS}$	CI0	CI1	$\overline{RB1}$	$\overline{RB0}$	$\overline{RFS}$	$\overline{CFS}$	SB	$\overline{SA}$	0

PROGRAM DATA BIT	NAME	POLARITY/FUNCTION
PO0	ECC	ECC = 0 FOR NON-ECC MODE
PO1	$\overline{SA}$	$\overline{SA}$ = 0 PORT A IS SYNCHRONOUS $\overline{SA}$ = 1 PORT A IS ASYNCHRONOUS

PO2	SB	SB = 0 PORT B IS ASYNCHRONOUS
		SB = 1 PORT B IS SYNCHRONOUS

PO3	$\overline{CFS}$	$\overline{CFS}$ = 0 FAST-CYCLE iAPX 286 MODE $\overline{CFS}$ = 1 SLOW-CYCLE iAPX 86 MODE
PO4	$\overline{RFS}$	$\overline{RFS}$ = 0 FAST RAM $\overline{RFS}$ = 1 SLOW RAM
PO5 PO6	$\overline{RB0}$ $\overline{RB1}$	RAM BANK OCCUPANCY SEE TABLE 2
PO7	CI1	COUNT INTERVAL BIT 1; SEE TABLE 6
PO8	CI0	COUNT INTERVAL BIT 0; SEE TABLE 6
PO9	$\overline{PLS}$	$\overline{PLS}$ = 0 LONG REFRESH PERIOD $\overline{PLS}$ = 1 SHORT REFRESH PERIOD

PO10	EXT	EXT = 0 NOT EXTENDED
		EXT = 1 EXTENDED

PO11	$\overline{FFS}$	$\overline{FFS}$ = 0 FAST CPU FREQUENCY $\overline{FFS}$ = 1 SLOW CPU FREQUENCY
PO12	$\overline{PPR}$	$\overline{PPR}$ = 0 MOST RECENTLY USED PORT PRIORITY $\overline{PPR}$ = 1 PORT A PREFERRED PRIORITY

PO13	TM1	TM1 = 0 TEST MODE 1 OFF
		TM1 = 1 TEST MODE 1 ENABLED

PO14	0	RESERVED MUST BE ZERO
PO15	0	RESERVED MUST BE ZERO

Figure 9. Non-ECC Mode Program Data Word

PO15							PO8	PO7							PO0
$\overline{TM2}$	RB1	RB0	PPR	FFS	$\overline{EXT}$	PLS	$\overline{CI0}$	$\overline{CI1}$	XB	$\overline{XA}$	RFS	CFS	$\overline{SB}$	SA	1

PROGRAM DATA BIT	NAME	POLARITY/FUNCTION
PO0	ECC	ECC = 1 ECC MODE

PO1	SA	SA = 0 PORT A ASYNCHRONOUS
		SA = 1 PORT A SYNCHRONOUS

PO2	$\overline{SB}$	$\overline{SB}$ = 0 PORT B SYNCHRONOUS $\overline{SB}$ = 1 PORT B ASYNCHRONOUS

PO3	CFS	CFS = 0 SLOW-CYCLE iAPX 86 MODE
		CFS = 1 FAST-CYCLE iAPX 286 MODE

PO4	RFS	RFS = 0 SLOW RAM
		RFS = 1 FAST RAM

PO5	$\overline{XA}$	$\overline{XA}$ = 0 MULTIBUS-COMPATIBLE ACKA $\overline{XA}$ = 1 ADVANCED ACKA NOT MULTIBUS-COMPATIBLE

PO6	XB	XB = 0 ADVANCED ACKB NOT MULTIBUS COMPATIBLE
		XB = 1 MULTIBUS-COMPATIBLE ACKB

PO7	$\overline{CI1}$	COUNT INTERVAL BIT 1; SEE TABLE 6
PO8	$\overline{CI0}$	COUNT INTERVAL BIT 0; SEE TABLE 6

PO9	PLS	PLS = 0 SHORT REFRESH PERIOD
		PLS = 1 LONG REFRESH PERIOD

PO10	$\overline{EXT}$	$\overline{EXT}$ = 0 MASTER AND SLAVE EDCU $\overline{EXT}$ = 1 MASTER EDCU ONLY

PO11	FFS	FFS = 0 SLOW CPU FREQUENCY
		FFS = 1 FAST CPU FREQUENCY

PO12	PPR	PPR = 0 PORT A PREFERRED PRIORITY
		PPR = 1 MOST RECENTLY USED PORT PRIORITY

PO13 PO14	RB0 RB1	RAM BANK OCCUPANCY SEE TABLE 2
PO15	$\overline{TM2}$	$\overline{TM2}$ = 0 TEST MODE 2 ENABLED $\overline{TM2}$ = 1 TEST MODE 2 OFF

Figure 10. ECC Mode Program Data Word

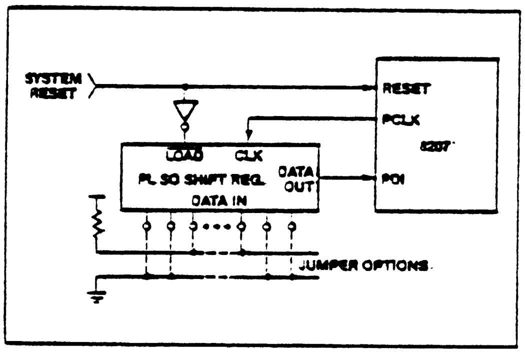


Figure 11. External Shift Register Interface

Table 4A.
Default Non-ECC Programming, POI Pin (57) Tied to Ground.

Port A is Synchronous
Port B is Asynchronous
Fast-cycle Processor Interface
Fast RAM
Refresh Interval uses 236 clocks
128 Row refresh in 2 ms; 256 Row refresh in 4 ms
Fast Processor Clock Frequency (16 MHz)
"Most Recently Used" Priority Scheme
4 RAM banks occupied

Table 4B.
Default ECC Programming, POI Pin (57) Tied to V_{CC}.

Port A is Synchronous
Port B is Asynchronous
Fast-cycle Processor Interface
Fast RAM
Port A has Advanced $\overline{ACKA}$ strobe (non-multibus)
Port B has Non-advance ACKB strobe (multibus)
Refresh interval uses 236 clocks
128 Row refresh in 2 ms; 256 Row refresh in 4 ms
Master EDCU only (16-bit system)
Fast Processor Clock Frequency (16 MHz)
"Most Recently Used" Priority Scheme
4 RAM banks occupied

If further system flexibility is needed, one or two external shift registers can be used to tailor the 8207 to its operating environment.

Synchronous/Asynchronous Mode (SA and SB Program Bits)

Each port of the 8207 may be independently configured to accept synchronous or asynchronous port commands ($\overline{RD}$, $\overline{WR}$, PCTL) and Port Enable ($\overline{PE}$) via the program bits SA and SB. The state of the SA and SB programming bits determine whether their associated ports are synchronous or asynchronous.

While a port may be configured with either the Status or Command interface in the synchronous mode, certain restrictions exist in the asynchronous mode. An asynchronous Command interface using the control lines of the Multibus is supported, and an asynchronous 8086 interface using the control lines of the 8086 is supported, with the use of TTL gates as illustrated in Figure 2. In the 8086 case, the TTL gates are needed to guarantee that status does not appear at the 8207 inputs too much before address, so that a cycle would start before address was valid.

Microprocessor Clock Frequency Option (CFS and FFS Program Bits)

The 8207 can be programmed to interface with slow-cycle microprocessors like the 8086, 8088, and 80186 or fast-cycle microprocessors like the 80286. The CFS bit configures the microprocessor interface to accept slow or fast cycle signals from either microprocessor group.

This option is used to select the speed of the microprocessor clock. Table 5 shows the various microprocessor clock frequency options that can be programmed.

Table 5.
Microprocessor Clock Frequency Options

Program Bits		Processor	Clock Frequency
CFS	FFS		
0	0	iAPX 86, 88, 186	5 MHz
0	1	iAPX 86, 88, 186	8 MHz
1	0	iAPX 286	10 MHz
1	1	iAPX 286	16 MHz

The external clock frequency must be programmed so that the failsafe refresh repetition circuitry can adjust its internal timing accordingly to produce a refresh request as programmed.

RAM Speed Option (RFS Program Bit)

The RAM Speed programming option determines whether RAM timing will be optimized for a fast or slow RAM. Whether a RAM is fast or slow is measured relative to the 2118-10 (Fast) or the 2118-15 (Slow) RAM specifications.

Refresh Period Options (CI0, CI1, and PLS Program Bits)

The 8207 refreshes with either 128 rows every 2 milliseconds or 256 rows every 4 milliseconds. This translates to one refresh cycle being executed approximately once every 15.6 microseconds. This rate can be changed to 256 rows every 2 milliseconds or a refresh approximately once every 7.8 microseconds via the Period Long/Short, program bit PLS, programming option. The 7.8 microsecond refresh request rate is intended for those RAMs, 64K and above, which may require a faster refresh rate.

In addition to PLS program option, two other programming bits for refresh exist: Count Interval 0 (CI0) and Count Interval 1 (CI1). These two programming bits allow the rate at which refresh requests are generated to be increased in order to permit refresh requests to be generated close to the same 15.6 or 7.8 microsecond period when the 8207 is operating at reduced frequencies. The interval between refreshes is decreased by 0%, 10%, 20%, or 30% as a function of how the count interval bits are programmed. A 5% guardband is built-in to allow for any clock frequency variations. Table 6 shows the refresh period options available.

The numbers tabulated under Count Interval represent the number of clock periods between internal refresh requests. The percentages in parentheses represent the decrease in interval between refresh requests. Note that all intervals have a built-in 5% (approximately) safety factor to compensate for minor clock frequency deviations and non-immediate response to internal refresh requests.

Extend Option (EXT Program Bit)

The Extend option lengthens the memory cycle to allow longer access time which may be required by the system. Extend alters the RAM timing to compensate for increased loading on the Row and Column Address Strobes, and in the multiplexed Address Out lines.

Port Priority Option and Arbitration (PPR Program Bit)

The 8207 has to internally arbitrate among three ports: Port A, Port B and Port C—the refresh port. Port C is an internal port dedicated to servicing refresh requests, whether they are generated internally by the refresh inverval counter, or externally by the user. Two arbitration approaches are available via

Table 6. Refresh Count Interval Table

Freq. (MHz)	Ref. Period (μS)	CFS	PLS	FFS	Count Interval CI1, CI0 (8207 Clock Periods) 00 (0%)	01 (10%)	10 (20%)	11 (30%)
16	15.6	1	1	1	236	212	188	164
	7.8	1	0	1	118	106	94	82
10	15.6	1	1	0	148	132	116	100
	7.8	1	0	0	74	66	58	50
8	15.6	0	1	1	118	106	94	82
	7.8	0	0	1	59	53	47	41
5	15.6	0	1	0	74	66	58	50
	7.8	0	0	0	37	33	29	25

the Port Priority programming option, program bit PPR. PPR determines whether the most recently used port will remain selected (PPR = 1) or whether Port A will be favored or preferred over Port B (PPR = 0).

A port is selected if the arbiter has given the selected port direct access to the timing generators. The front-end logic, which includes the arbiter, is designed to operate in parallel with the selected port. Thus a request on the selected port is serviced immediately. In contrast, an unselected port only has access to the timing generators through the front-end logic. Before a RAM cycle can start for an unselected port, that port must first become selected (i.e., the MUX output now gates that port's address into the 8207 in the case of Port A or B). Also, in order to allow its address to stabilize, a newly selected port's first RAM cycle is started by the front-end logic. Therefore, the selected port has direct access to the timing generators. What all this means is that a request on a selected port is started immediately, while a request on an unselected port is started two to three clock periods after the request, assuming that the other two ports are idle. Under normal operating conditions, this arbitration time is hidden behind the RAM cycle of the selected port so that as soon as the present cycle is over a new cycle is started. Table 7 lists the arbitration rules for both options.

Port LOCK Function

The LOCK function provides each port with the ability to obtain uninterrupted access to a critical region of memory and, thereby, to guarantee that the opposite port cannot "sneak in" and read from or write to the critical region prematurely.

Only one LOCK pin is present and is multiplexed between the two ports as follows: when MUX is high, the 8207 treats the LOCK input as originating at PORT A, while when MUX is low, the 8207 treats LOCK as originating at PORT B. When the 8207 recognizes a LOCK, the MUX output will remain pointed to the locking port until LOCK is deactivated. Refresh is not affected by LOCK and can occur during a locked memory cycle.

Table 7.
The Arbitration Rules for the Most Recently Used Port Priority and for Port A Priority Options Are As Follows:

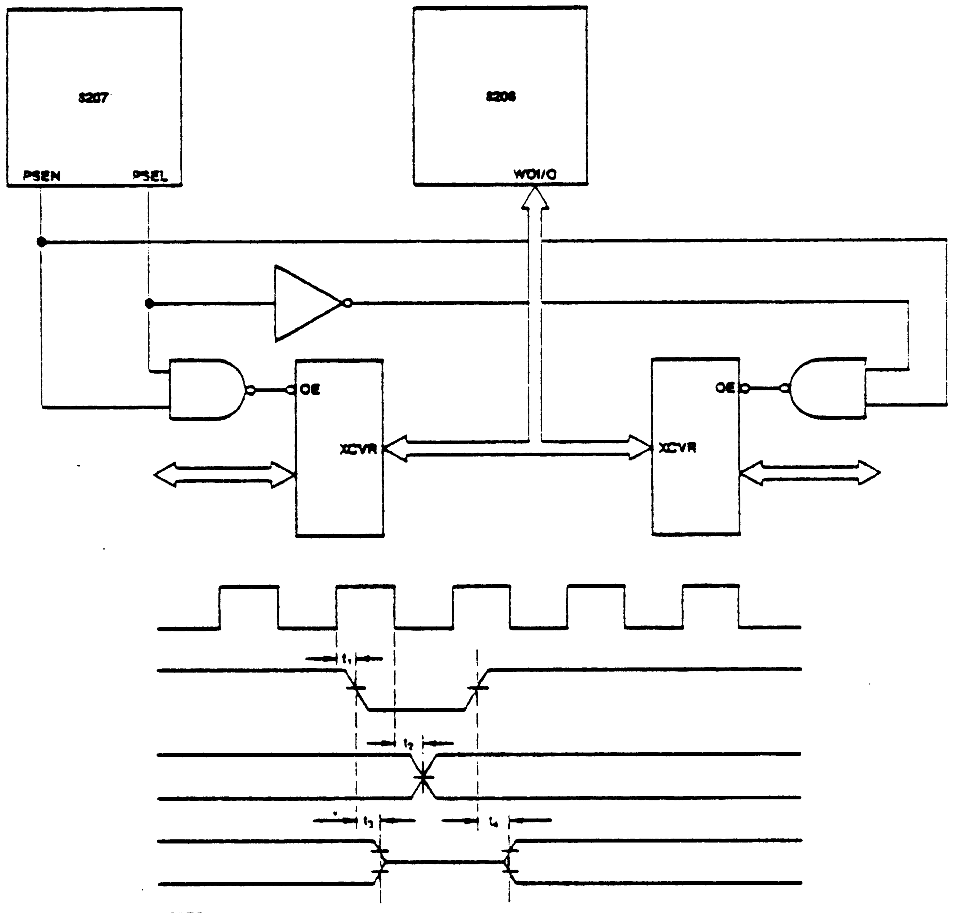
1.	If only one port requests service, then that port—if not already selected—becomes selected.
2a.	When no service requests are pending, the last selected processor port (Port A or B) will remain selected. (Most Recently Used Port Priority Option)
2b.	When no service requests are pending, Port A is selected whether it requests service or not. (Port A Priority Option)
3.	During reset initialization only Port C, the refresh port, is selected.
4.	If no processor requests are pending after reset initialization, Port A will be selected.
5a.	If Ports A and B simultaneously(*) request service while Port C is being serviced, then the next port to be selected is the one which was not selected prior to servicing Port C. (Most Recently Used Port Priority Option)
5b.	If Ports A and B simultaneously(*) request service while Port C is selected, then the next port to be selected is Port A. (Port A Priority Option)
6.	If a port simultaneously requests service with the currently selected port, service is granted to the selected port.
7.	The MUX output remains in its last state whenever Port C is selected.
8.	If Port C and either Port A or Port B (or both) simultaneously request service, then service is granted to the requester whose port is already selected. If the selected port is not requesting service, then service is granted to Port C.
9.	If during the servicing of one port, the other port requests service before or simultaneously with the refresh port, the refresh port is selected. A new port is not selected before the presently selected port is deactivated.
10.	Activating LOCK will mask off service requests from Port B if the MUX output is high, or from Port A if the MUX output is low.

* By "simultaneous" it is meant that two or more requests are valid at the clock edge at which the internal arbiter samples them.

Dual-Port Considerations

For both ports to be operated synchronously, several conditions must be met. The processors must be the same type (Fast or Slow Cycle) as defined by Table 8 and they must have synchronized clocks. Also when processor types are mixed, even though the clocks may be in phase, one frequency may be twice that of the other. So to run both ports synchronous using the status interface, the processors must have related timings (both phase and frequency). If these conditions cannot be met, then one port must run synchronous and the other asynchronous.

Figure 3 illustrates an example of dual-port operation using the processors in the slow cycle group. Note the use of cross-coupled NAND gates at the MUX output for minimizing contention between the two latches, and the use of flip flops on the status lines of the synchonous processor for delaying the status and thereby guaranteeing RAS will not be issued, even in the worst case, until address is valid. Figure 12 shows the timing associated with Port switching.

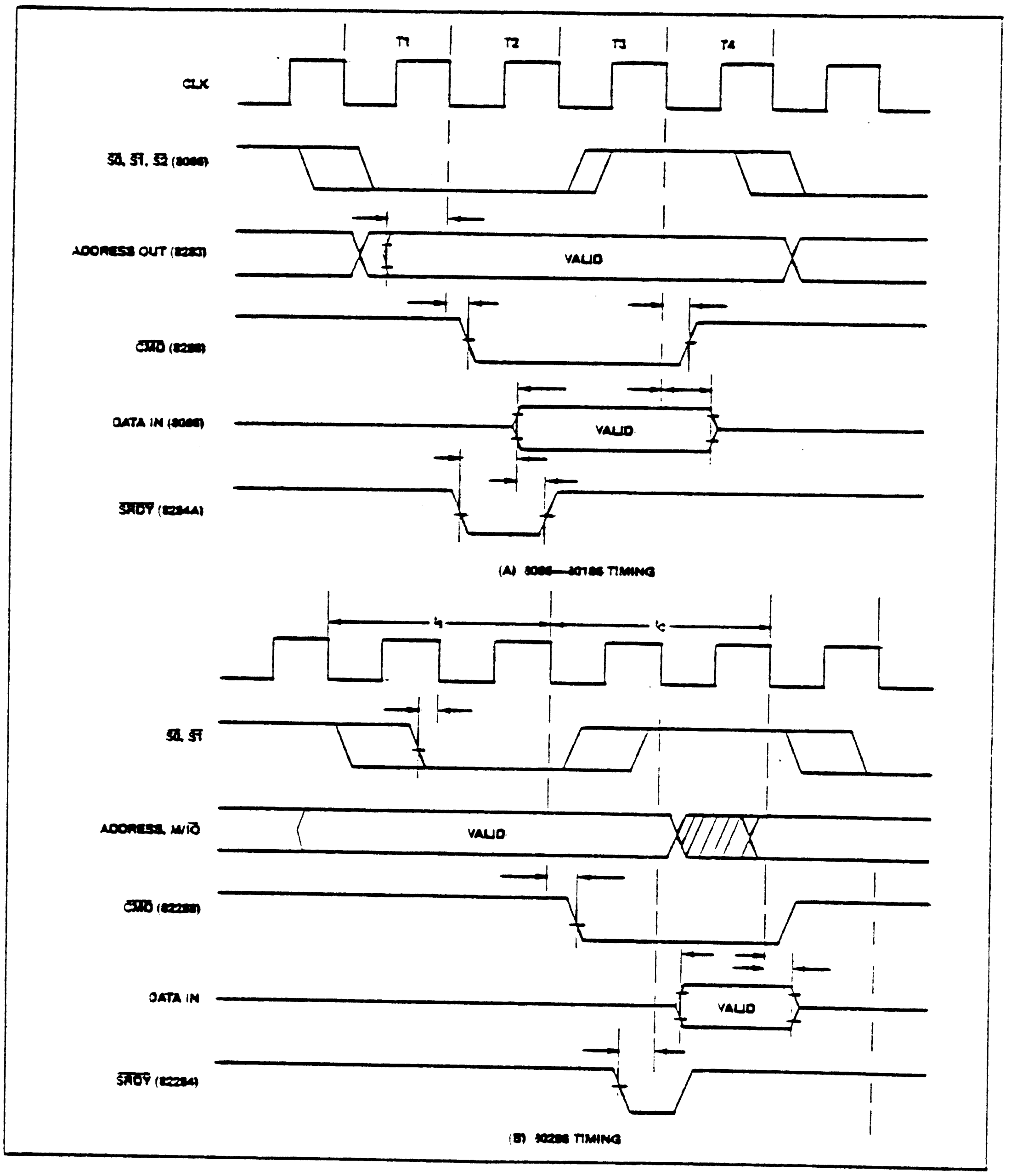


NOTES:
t_1—PSEN DELAY
t_2—PSEL DELAY
t_3—TRANSCEIVER OUTPUT DISABLE TIME
t_4—TRANSCEIVER OUTPUT ENABLE TIME

Figure 12.

Processor Timing

Timing for the 8086, 80186, and 80286 processors is given in Figure 13. In order to run without wait states, $\overline{AACK}$ must be used and connected to the $\overline{SRDY}$ input of the appropriate bus controller. $\overline{AACK}$ is issued relative to a point within the RAM cycle and has no fixed relationship to the processor's request. The

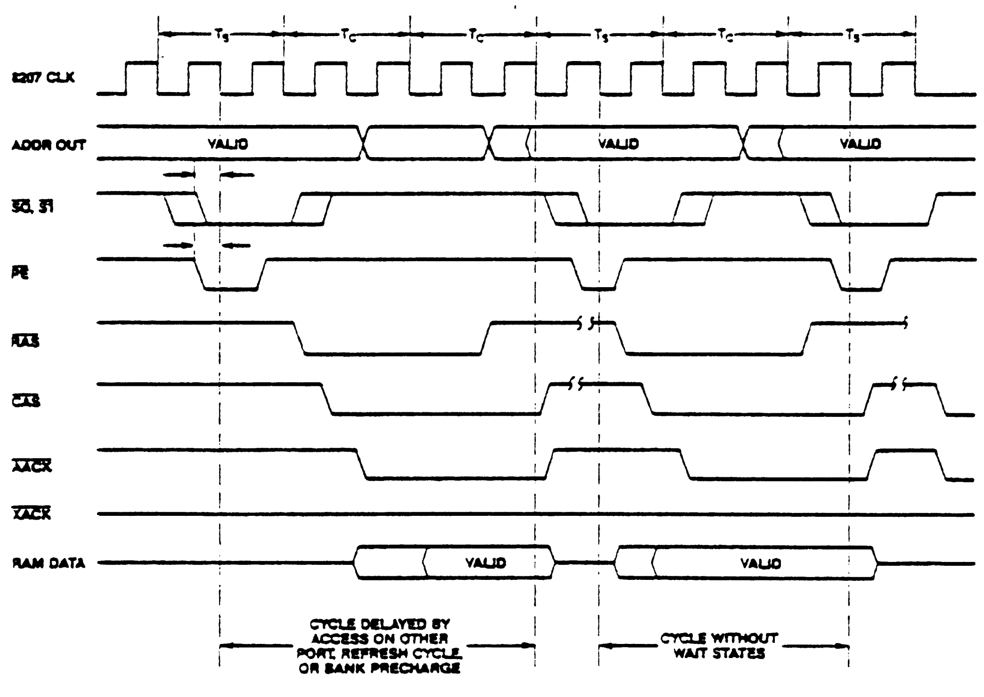


T1
T2
T3
T4
CLK
$\overline{S0}$, $\overline{S1}$, $\overline{S2}$ (8086)
ADDRESS OUT (8283)
VALID
$\overline{CMD}$ (8288)
DATA IN (8086)
VALID
$\overline{SRDY}$ (8284A)
(A) 8086—80186 TIMING
t_c
t_c
$\overline{S0}$, $\overline{S1}$
ADDRESS, M/$\overline{IO}$
VALID
$\overline{CMD}$ (82288)
DATA IN
VALID
$\overline{SRDY}$ (82284)
(B) 80286 TIMING

Figure 13. 8086, 80186 and 80286 Read Timing

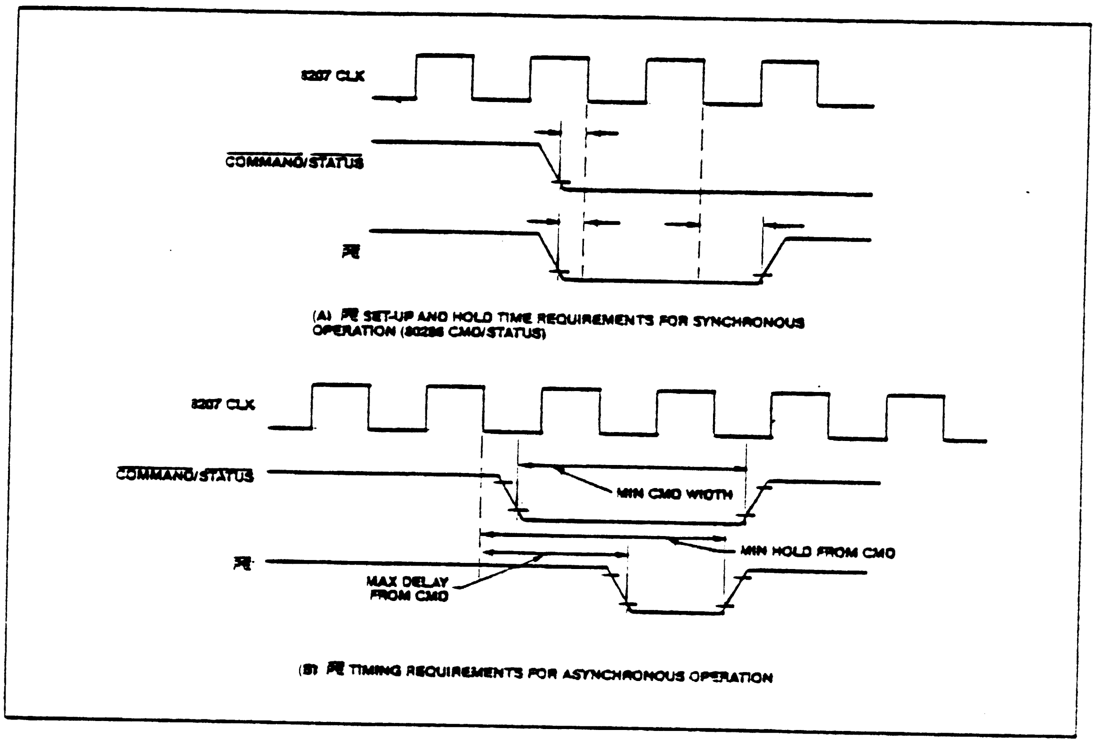
timing is such, however, that the processor will run without wait states, barring refresh cycles, bank precharge, and RAM accesses from the other port. In non-ECC fast cycle, fast RAM, non-extended configurations (80286), $\overline{\text{AACK}}$ is issued on the next falling edge of the clock after the edge that issues $\overline{\text{RAS}}$. In non-ECC, slow cycle, non-extended, or extended with fast RAM cycle configurations (8086, 80186), $\overline{\text{AACK}}$ is issued on the same clock cycle that issues $\overline{\text{RAS}}$. Figure 14 illustrates the timing relationship between $\overline{\text{AACK}}$, the RAM cycle, and the processor cycle for several different situations.

Port Enable ($\overline{\text{PE}}$) setup time requirements depend on whether the associated port is configured for synchronous or asynchronous operation. In synchronous operation, $\overline{\text{PE}}$ is required to be setup to the same clock edge as the status or commands. If $\overline{\text{PE}}$ is true (low), a RAM cycle is started; if not, the cycle is aborted. In asynchronous operation, $\overline{\text{PE}}$ is required to be setup to the same clock edge as the internally synchronized status or commands. Externally, this allows the internal synchronization delay to be added to the status (or command)-to-$\overline{\text{PE}}$ delay time, thus allowing for more external decode time than is available in synchronous operation. The minimum synchronization delay is the additional amount that $\overline{\text{PE}}$ must be held valid. If $\overline{\text{PE}}$ is not held valid for the maximum synchronization delay time, it is possible that $\overline{\text{PE}}$ will go invalid prior to the status or command being synchronized. In such a case the 8207 aborts the cycle. If a memory cycle intended for the 8207 is aborted, then no acknowledge ($\overline{\text{AACK}}$ or $\overline{\text{XACK}}$) is issued and the processor locks up in endless wait states. Figure 15 illustrates the status (command) timing requirements for synchronous and asynchronous systems. Figures 16 and 17 show a more detailed hook-up of the 8207 to the 8086 and 80286, respectively.



(A) 80286 TIMING

Figure 14.



(A) $\overline{\text{PE}}$ SET-UP AND HOLD TIME REQUIREMENTS FOR SYNCHRONOUS OPERATION (80286 CMD/STATUS)

(B) $\overline{\text{PE}}$ TIMING REQUIREMENTS FOR ASYNCHRONOUS OPERATION

Figure 15.

Memory Acknowledge (ACK, AACK, XACK)

In system configurations without error correction, two memory acknowledge signals per port are supplied by the 8207. They are the Advanced Acknowledge strobe ($\overline{\text{AACK}}$) and the Transfer Acknowledge strobe ($\overline{\text{XACK}}$). The CFS programming bit determines for which processor $\overline{\text{AACKA}}$ and $\overline{\text{AACKB}}$ are optimized, either 80286 (CFS = 1) or 8086/186 (CFS = 0), while the SA and SB programming bits optimize $\overline{\text{AACK}}$ for synchronous operation ("early" $\overline{\text{AACK}}$) or asynchronous operation ("late" $\overline{\text{AACK}}$).

Both the early and late $\overline{\text{AACK}}$ strobes are three clocks long for CFS = 1 and two clocks long for CFS = 0. The $\overline{\text{XACK}}$ strobe is asserted when data is valid (for reads) or when data may be removed (for writes) and meets the Multibus requirements. $\overline{\text{XACK}}$ is removed asynchronously by the command going inactive. Since in a synchronous operation the 8207 removes read data before late $\overline{\text{AACK}}$ or $\overline{\text{XACK}}$ is recognized by the CPU, the user must provide for data latching in the system until the CPU reads the data. In synchronous operation, data latching is unnecessary since the 8207 will not remove data until the CPU has read it.

In ECC-based systems there is one memory acknowledge ($\overline{\text{ACK}}$) per port and a programming bit associated with each acknowledge. If the X programming bit is high, the strobe is configured as $\overline{\text{XACK}}$, while if the bit is low, the strobe is configured as $\overline{\text{AACK}}$. As in non-ECC, the SA and SB programming bits determine whether the $\overline{\text{AACK}}$ strobe is early or late.

Data will always be valid a fixed time after the occurrence of the advanced acknowledge. Table 9 summarizes the various transfer acknowledge options.

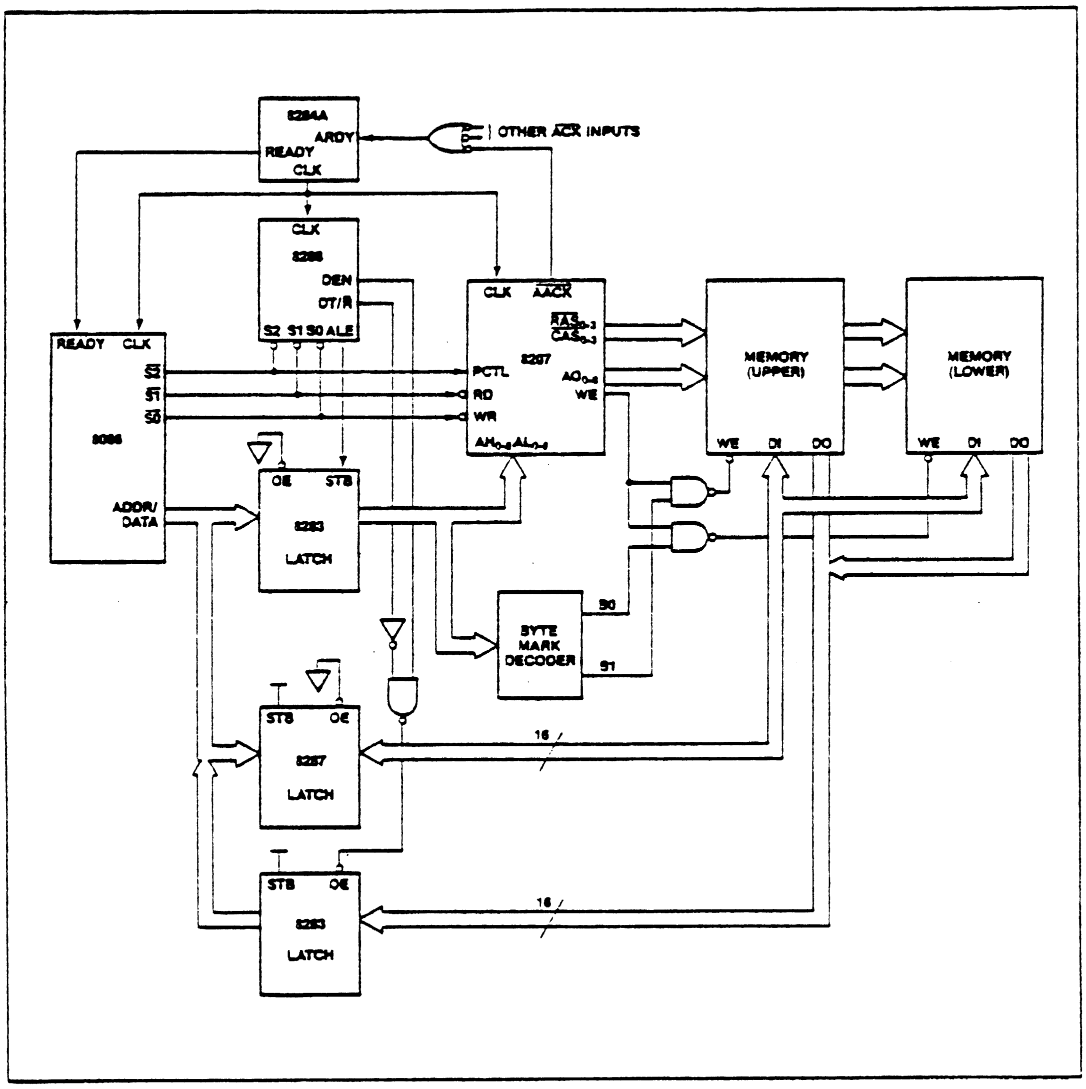


Figure 16. 8086 Hook-up to 8207 Non-ECC Synchronous System-Single Port.

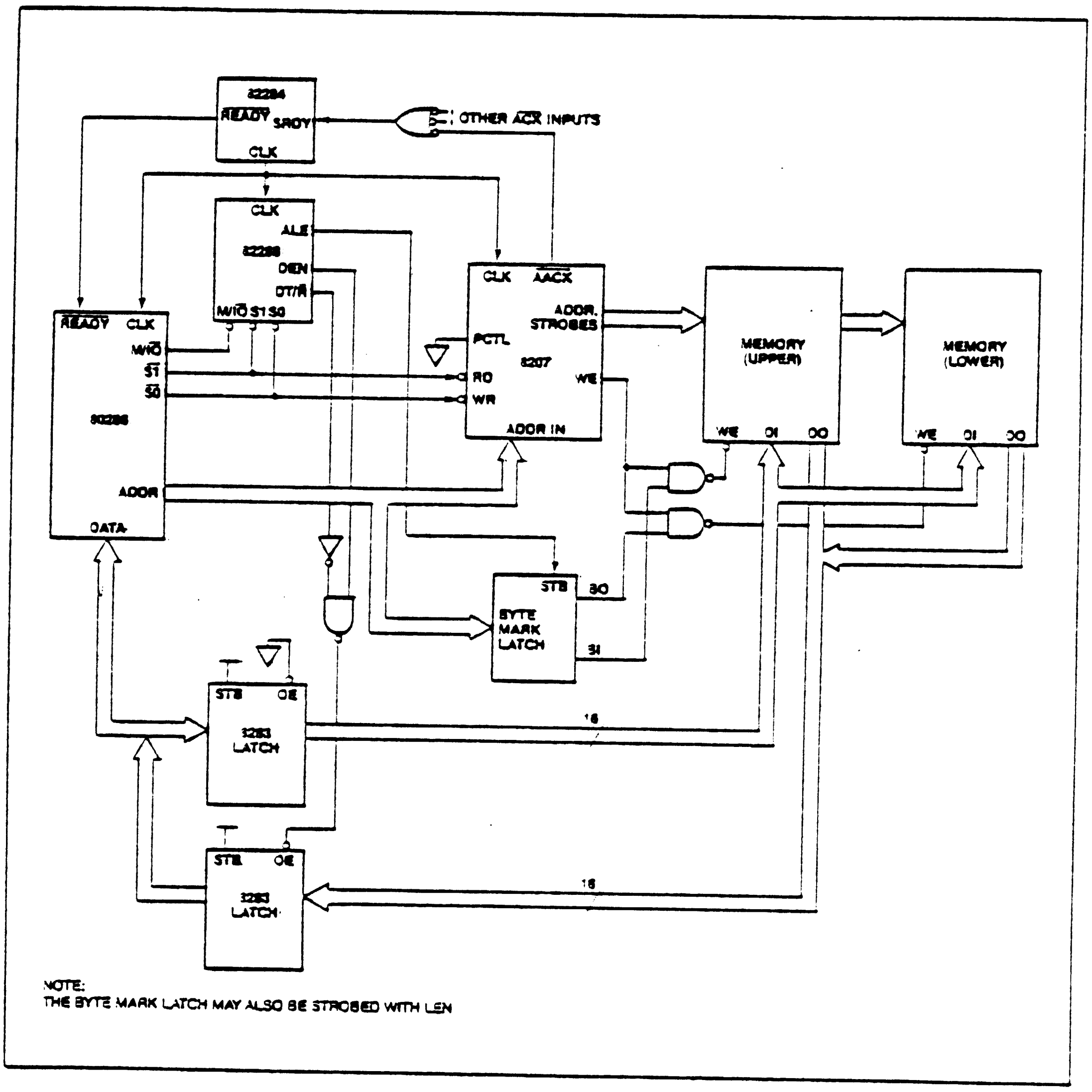


Figure 17. 80286 Hook-up to 8207 Non-ECC Synchronous System-Single Port.

Table 8. Processor Interface/Acknowledge Summary

CYCLE	PROCESSOR	REQUEST TYPE	SYNC/ASYNC INTERFACE	ACKNOWLEDGE TYPE
FAST CYCLE CFS=1	80286	STATUS	SYNC	E$\overline{\text{AACK}}$
	80286	STATUS	ASYNC	L$\overline{\text{AACK}}$
	80286	COMMAND	SYNC	E$\overline{\text{AACK}}$
	80286	COMMAND	ASYNC	L$\overline{\text{AACK}}$
	8086/80186	STATUS	ASYNC	L$\overline{\text{AACK}}$
	8086/80186	COMMAND	ASYNC	L$\overline{\text{AACK}}$
	MULTIBUS	COMMAND	ASYNC	$\overline{\text{XACK}}$
SLOW CYCLE CFS=0	8086/80186	STATUS	SYNC	E$\overline{\text{AACK}}$
	8086/80186	STATUS	ASYNC	L$\overline{\text{AACK}}$
	8086/80186	COMMAND	SYNC	E$\overline{\text{AACK}}$
	8086/80186	COMMAND	ASYNC	L$\overline{\text{AACK}}$
	MULTIBUS	COMMAND	ASYNC	$\overline{\text{XACK}}$

Table 9. Memory Acknowledge Option Summary

	Synchronous	Asynchronous	XACK
Fast Cycle	AACK Optimized for Local 80286	AACK Optimized for Remote 80286	Multibus Compatible
Slow Cycle	AACK Optimized for Local 8086/186	AACK Optimized for Remote 8086/186	Multibus Compatible

Test Modes

Two special test modes exist in the 8207 to facilitate testing. Test Mode 1 (non-ECC mode) splits the refresh address counter into two separate counters and Test Mode 2 (ECC mode) presets the refresh address counter to a value slightly less than rollover.

Test Mode 1 splits the address counter into two, and increments both counters simultaneously with each refresh address update. By generating external refresh requests, the tester is able to check for proper operation of both counters. Once proper individual counter operation has been established, the 8207 must be returned to normal mode and a second test performed to check that the carry from the first counter increments the second counter. The outputs of the counters are presented-on the address out bus with the same timing as the row and column addresses of a normal scrubbing operation. During Test Mode 1, memory initialization is inhibited, since the 8207, by definition, is in non-ECC mode.

Test Mode 2 sets the internal refresh counter to a value slightly less than rollover. During functional testing other than that covered in Test Mode 1, the 8207 will normally be set in Test Mode 2. Test Mode 2 eliminates memory initialization in ECC mode. This allows quick examination of the circuitry which brings the 8207 out of memory initialization and into normal operation. Test Mode 2 is also useful for quick reset response in ECC systems.

PACKAGE

The 8207 is packaged in a 68-pin, leadless JEDEC type A hermetic chip carrier. Figure 18 illustrates the package, and Figure 19 is the pinout.

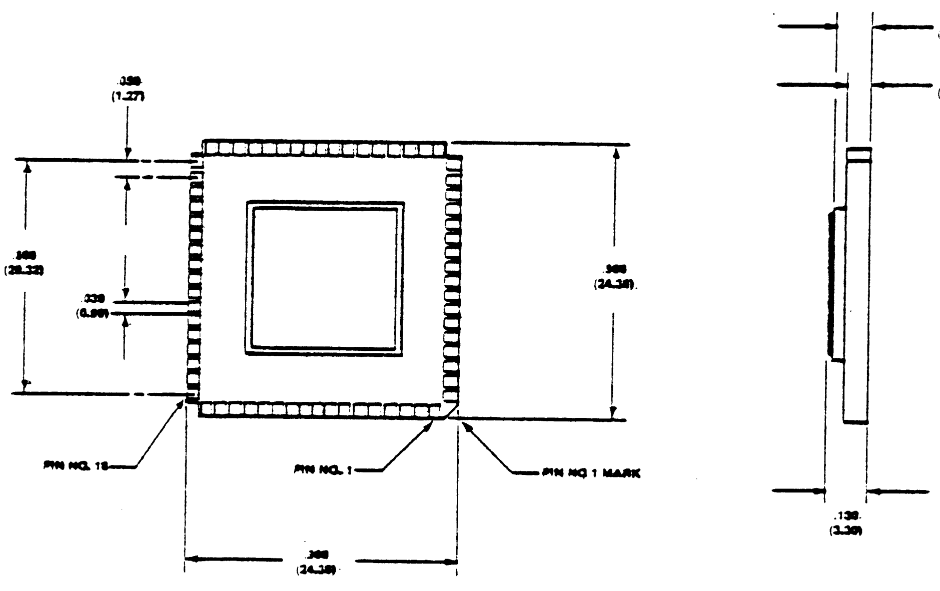


Figure 18. 8207 JEDEC Type A Package

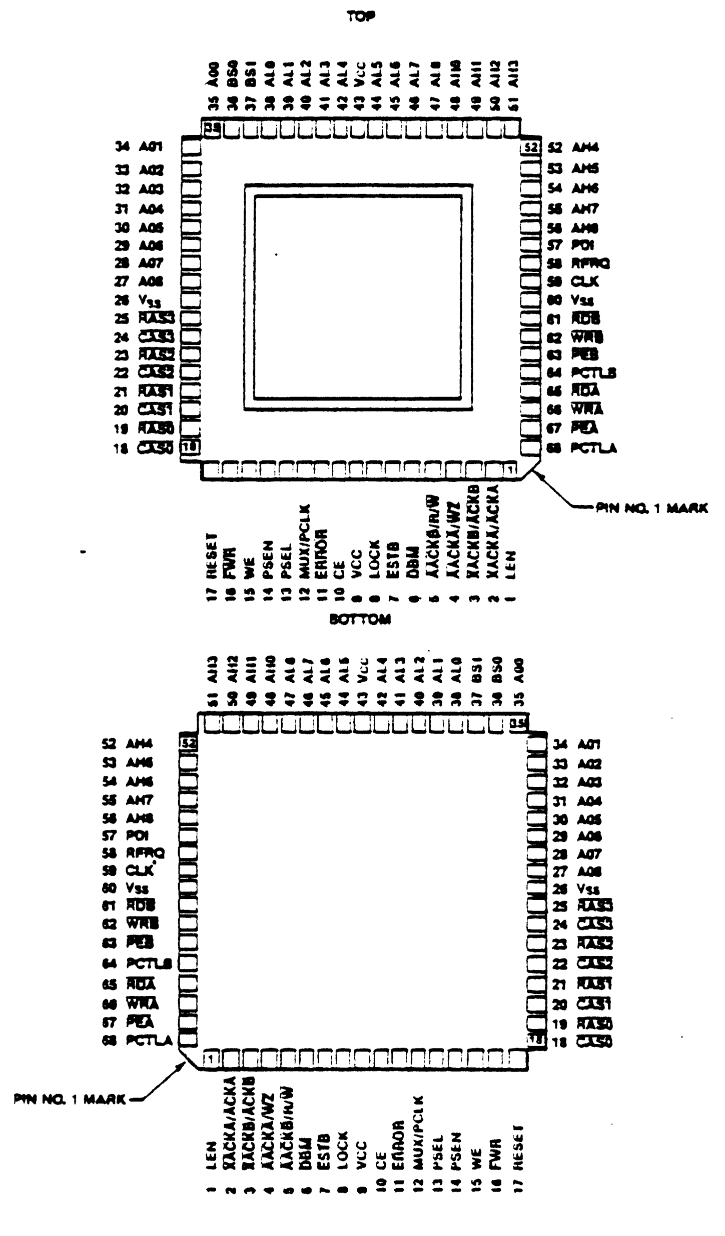
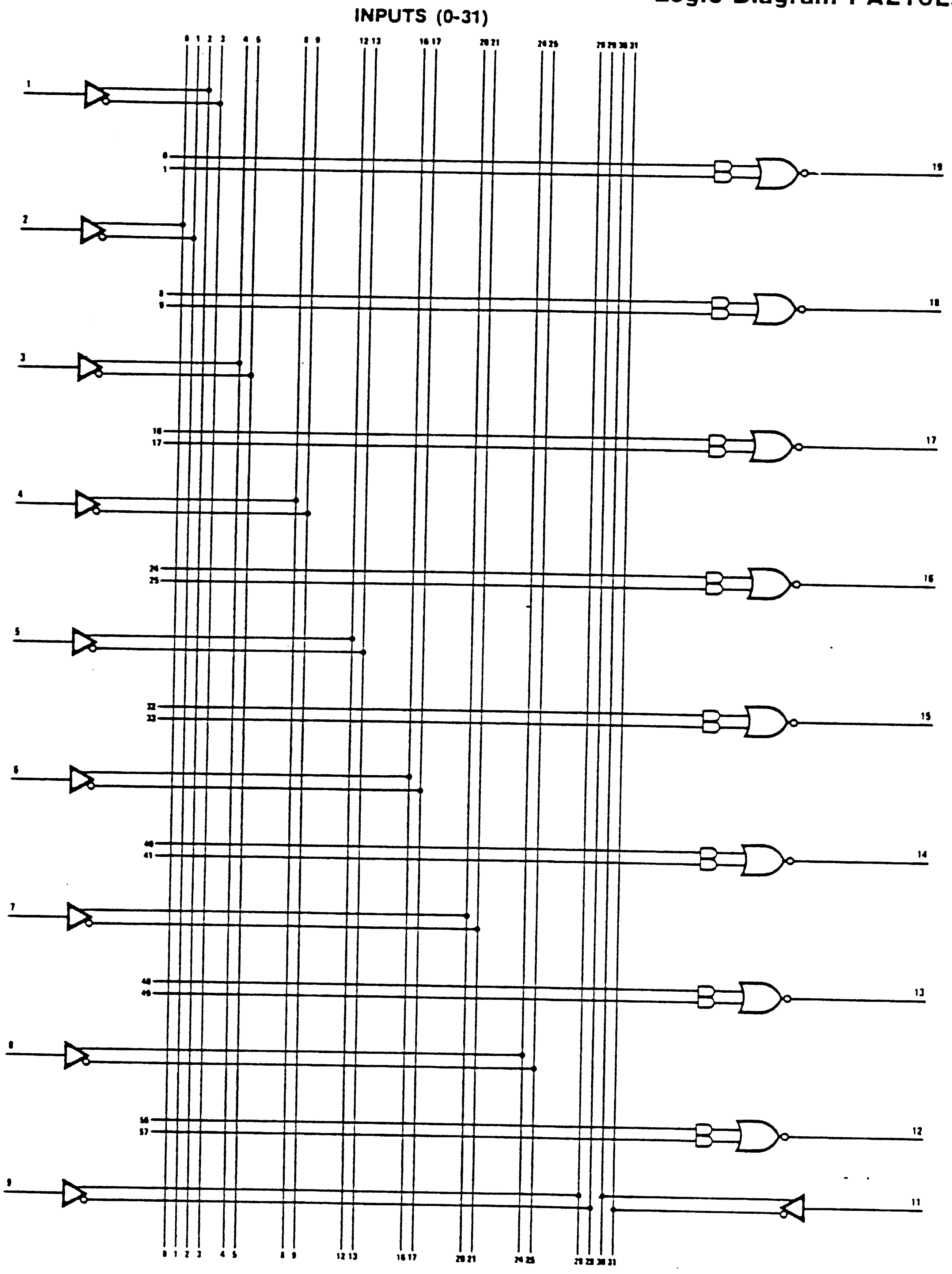


Figure 19. 8207 Pinout Diagram

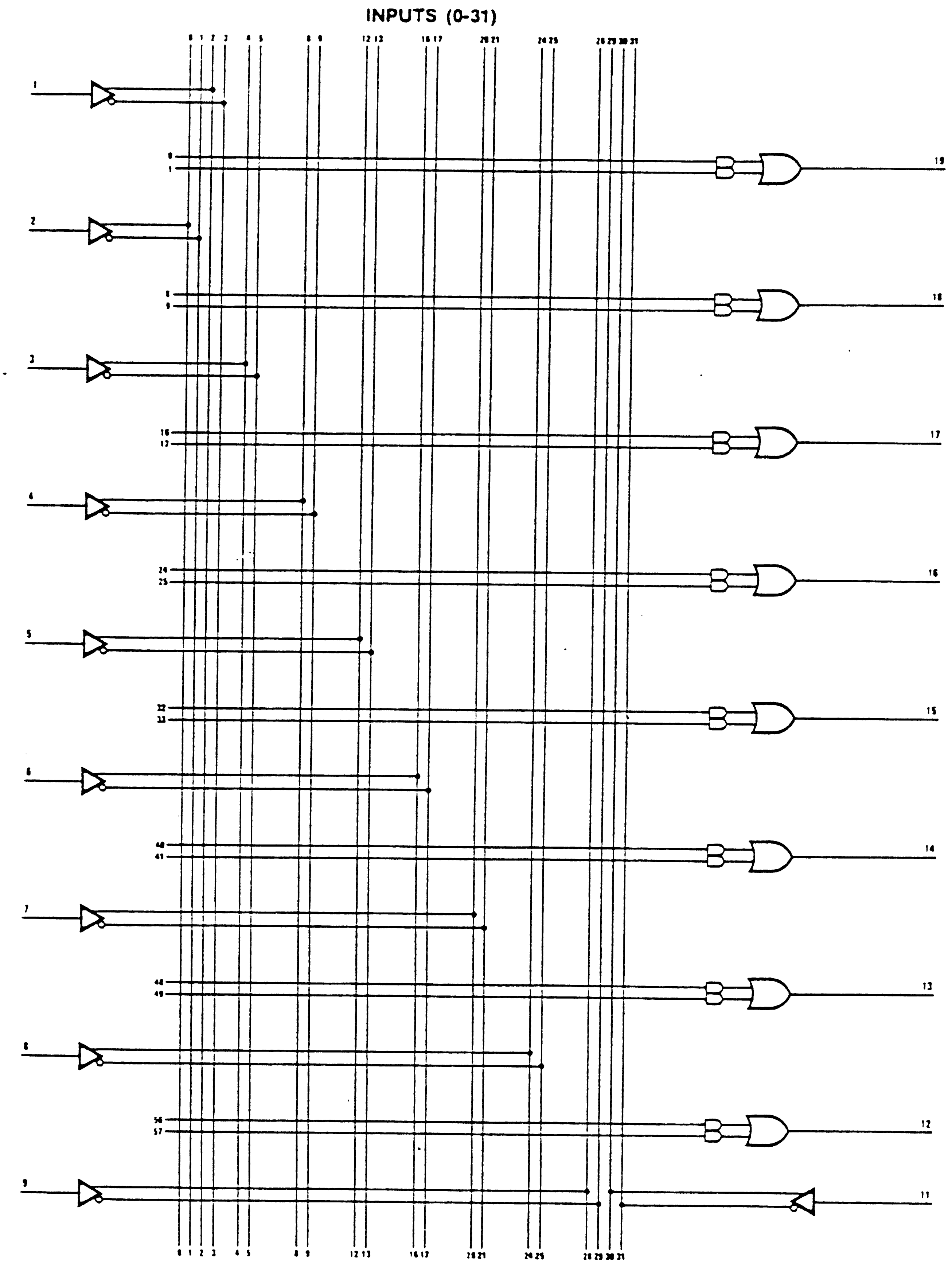
NOTICE: The pinouts are not final specifications and are subject to change.

AFN-02218A

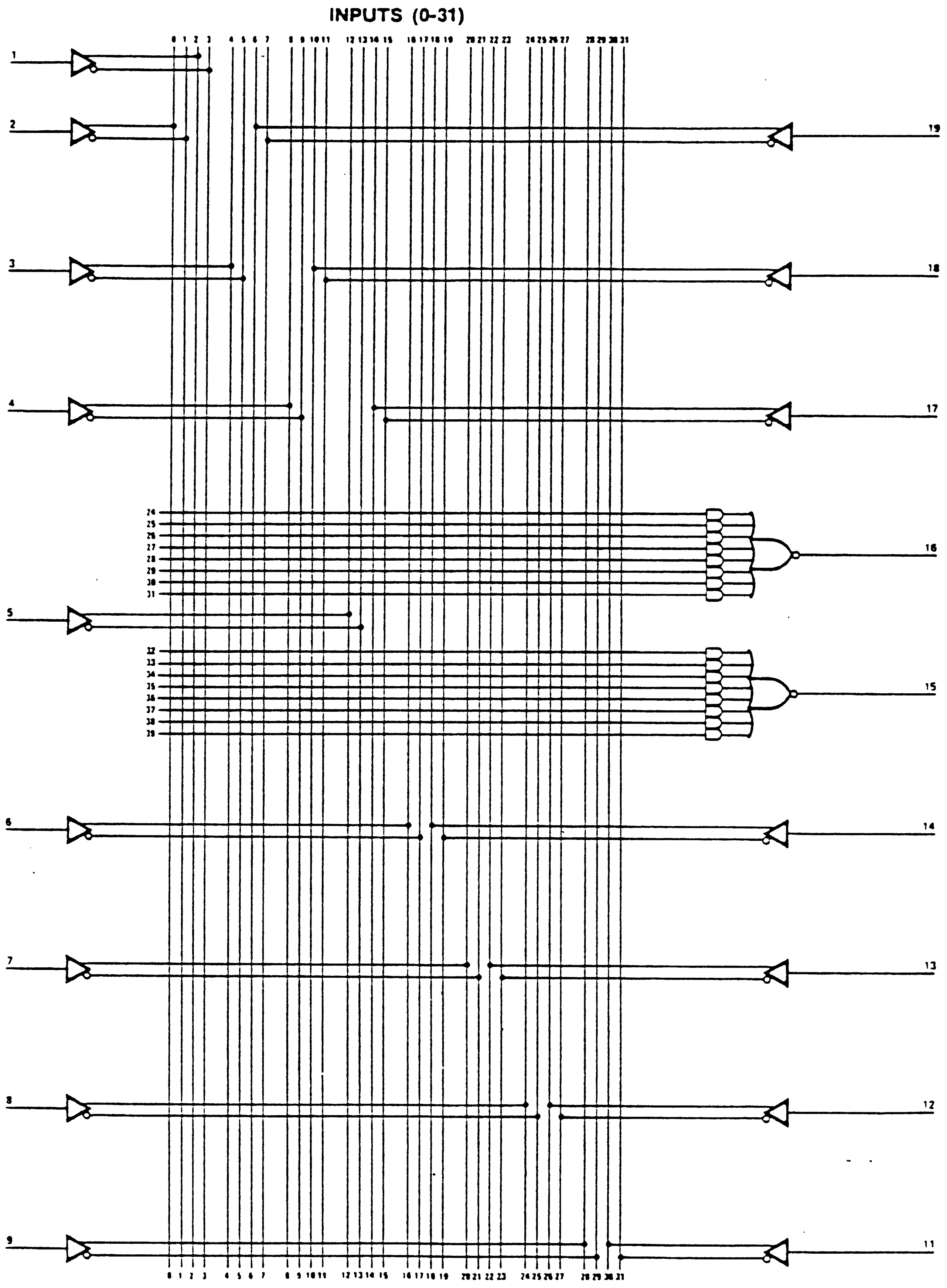
Logic Diagram PAL10L8-2



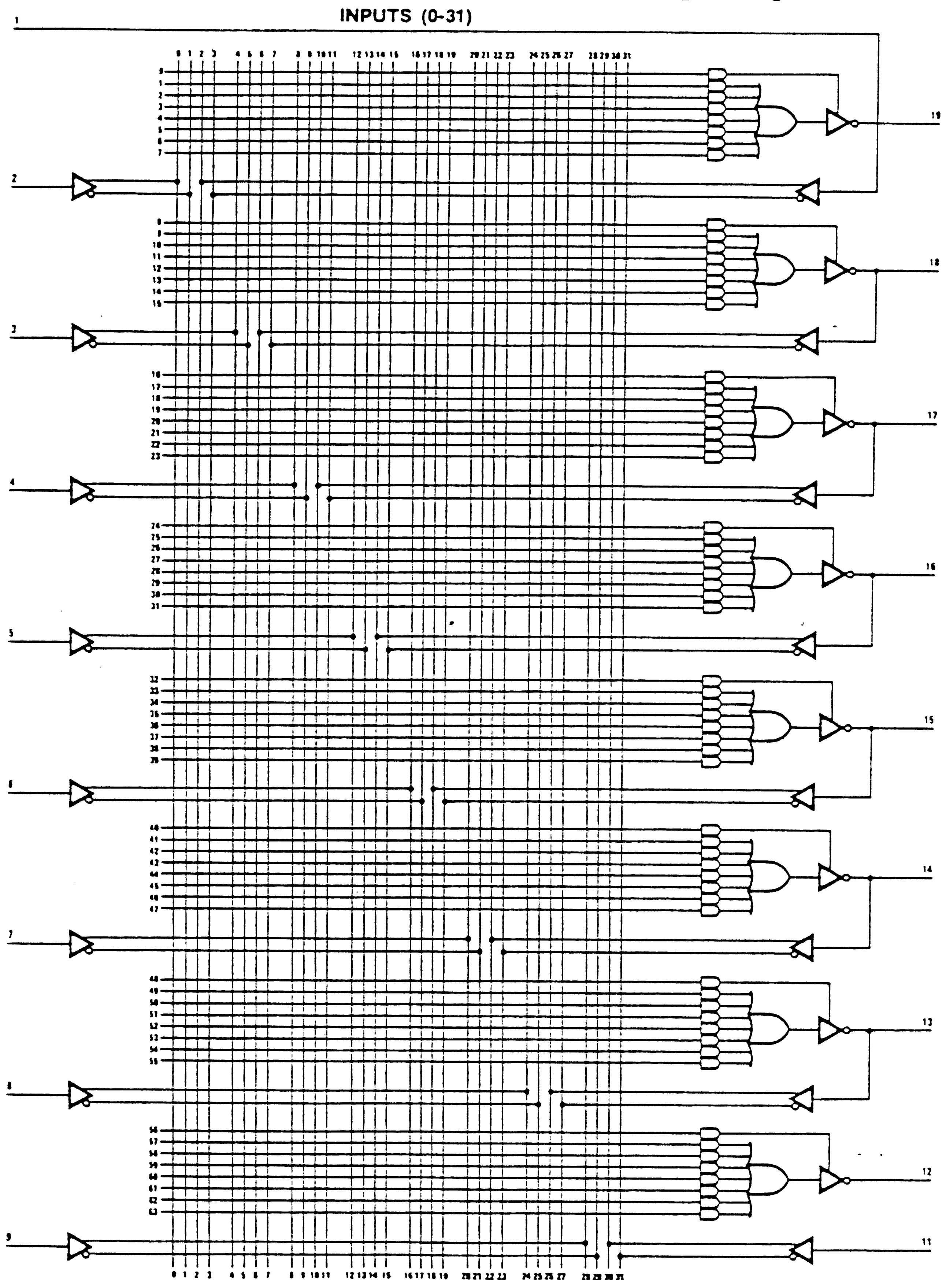
Logic Diagram PAL10H8-2



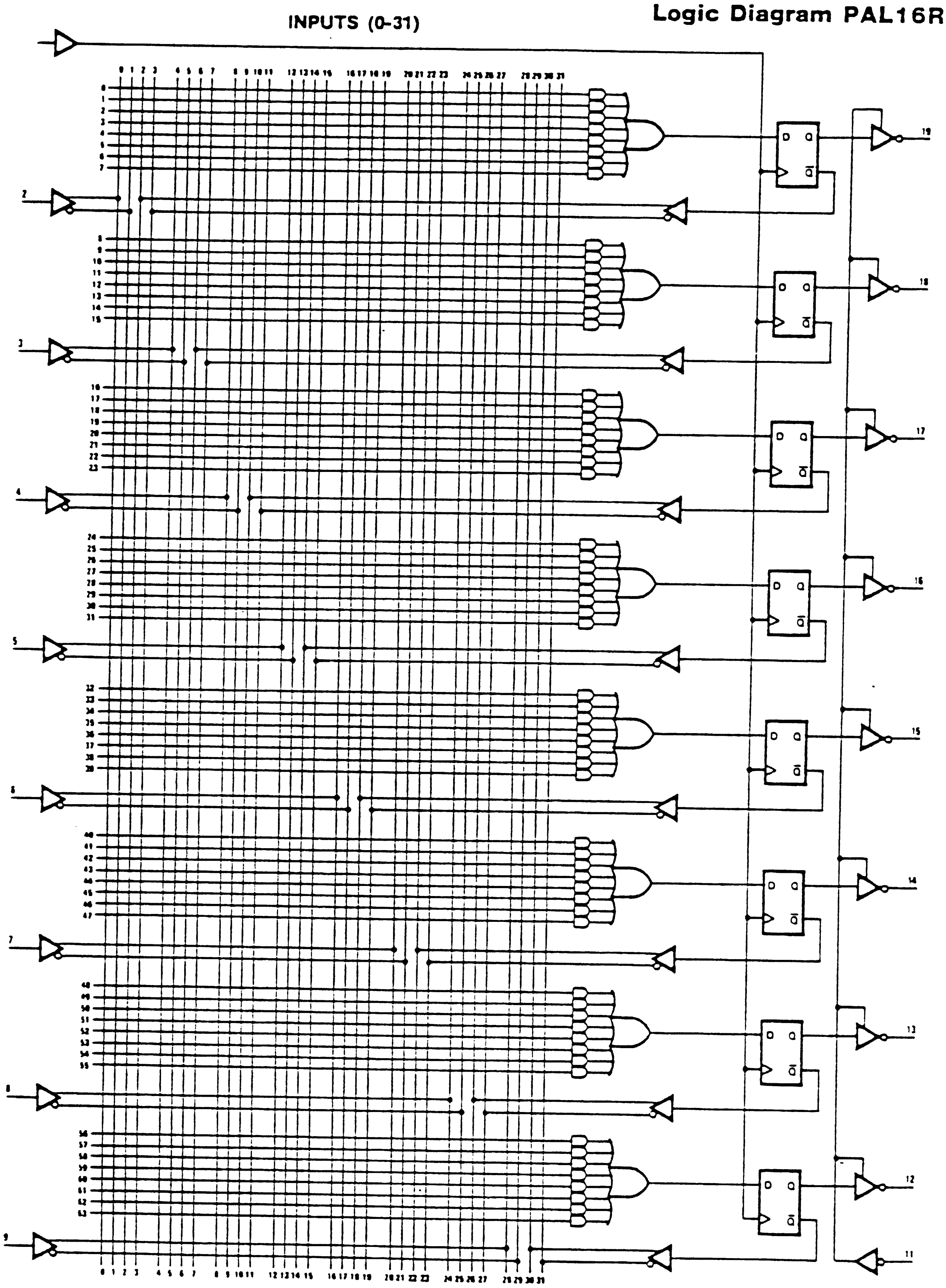
Logic Diagram PAL16L2-2



Logic Diagram PAL16L8-2



Logic Diagram PAL16R8-2



GLOSSARY

Accuracy
Absolute accuracy error of a DAC is the difference between the analog output that is expected when a given digital code is applied and the output that is actually measured with that code applied to the converter.

For an ADC, the accuracy is the difference between the analog input theoretically required to produce a given digital output code and the analog input actually required to produce that code.

The overall accuracy is affected by quantizing, offset error, bandwidth, linearity, monotonicity, settling time and long-term drift.

Acquisition Time
This is the time elapsed between the sample of track command and the point at which the output tracks the input (see text for more information).

A/D (or ADC)
Analog-to-Digital Converter

Aliasing
When a signal is sampled at a rate less than twice per period of the highest frequency component, an effect known as "aliasing" occurs. This is due to undersampling where the sampled amplitude information takes on the characteristic of an "alias" waveform of a lower frequency.

Amplitude Uncertainty (or Amplitude Error)
This error is a function of the analog signal and the aperture uncertainty. This is expressed as:

$$\text{Delta V} = \frac{T_a \, dV}{dt}$$

Where Delta V is the amplitude uncertainty, Ta is the aperture uncertainty, and dV/dt is the rate at which the signal is changing at the instant of sampling.

Aperture Time Delay
This is the time elapsed between the hold command and the point at which the sampling switch is completely open.

Aperture Uncertainty (or Aperture Time)
This is the variation in aperture delay. It is the difference between the maximum and minimum aperture.

Bandwidth
The input buffer amplifier and digitizer circuitry in the ADC have a finite bandwidth. This limited frequency response may cause lengthening of rise times which shows up as an apparent time shift of the time of sampling.

D/A (or DAC)
Digital to Analog Converter

Equivalent-Time Bandwidth
The highest attainable bandwidth for repetitive signals being sampled is referred to as Equivalent-Time Bandwidth.

Equivalent-Time Sampling
For repetitive signals which require multiple sweeps, samples are taken at varying times during the different sweeps until the waveform memory is filled.

Glitches
When a major transition occurs in the input code to a DAC, the worst case being the switching of all bits (a transition at ½ - scale), the analog output may momentarily slew away from the value represented by the new code. This produces a large transient spike referred to as a DAC glitch.

Linearity
Linearity error of a converter is the deviation of the analog values, in a plot of the measured conversion relationship, from a straight line. The straight line can be either a "best straight line," determined by manipulation of the gain and/or offset to equalize the positive and negative deviations, or it can be a straight line passing through the end points of the transfer characteristic after they have been calibrated. The latter is referred to as "end-point" non-linearity, a more conservative measure, and is much easier to verify in actual practice.

Long-Term Drift
This is due mainly to resistor and semiconductor aging and can affect all specifications except resolution and quantizing error.

Memory Size
This may be expressed in a number of ways. Digitizers which feature selectable memory management will typically let the user determine the desired points-per-waveform. Digitizers with fixed resolution typically will specify the number of points-per-waveform.

Monotonicity
The output of a monotonic ADC or DAC never decreases in response to an increasing input stimulus (and vice versa).

Nyquist Frequency
If a continuous bandwidth limited signal contains no frequency components higher than f_n, then the original signals can be completely recovered without distortion if sampled at a rate greater than 2 f_n samples per second. f_n is referred to as the Nyquist frequency.

Offset Error
The output voltage of a DAC with zero code input, or the required mean value of input voltage of an ADC to set zero code out.

Perceptual Aliasing
An optical illusion caused by the eye's inability to perceive the proper time order of the displayed data points. This occurs at frequencies much lower than a tenth of the sample frequency.

Quantizing Error
All analog values within a given quantum are represented by the same digital code, usually assigned to the midrange value. There is, therefore, an inherent quantization uncertainty of ± ½ LSB.

Real-Time Sampling
For single-sweep acquisition, the signal must be sampled so that all data points are taken at equal increments of time, one immediately following the previous, until the end of sweep.

Record Length
This is generally defined as the number of samples that are acquired per waveform.

Resolution
"Nominal resolution" is the relative value of the LSB, or 2-n for binary n-bit converters. This may be expressed as 1 part in 2n, as a percentage of full scale, in parts per million, or simply by "n bits."

Sampling: Equivalent-Time
For repetitive signals which require multiple sweeps, samples are taken at varying times during the different sweeps until the waveform memory is filled.

Sampling: Real-Time
For single-sweep acquisition, the signal must be sampled so that all data points are taken at equal increments of time, one immediately following the previous, until the end of sweep.

Sample and Hold (S/H)
This is an acquisition technique in which a very short sample pulse is applied to the input when a new sample needs to be obtained at the output. An S/H is normally left in the hold condition.

Sample Rate
This is the actual frequency at which a sample of the analog signal is taken. This may be expressed in samples/second, or hertz. This is usually limited by the maximum conversion and settling time.

Sampling Theorem
If a continuous bandwidth limited signal contains no frequency components higher than f_n, then the original signals can be completely recovered without distortion if sampled at a rate greater than 2 f_n samples per second. f_n is referred to as the Nyquist frequency.

Settling Time
This is the time it takes for a DAC's output to settle for a full-scale code change, usually to within the analog equivalent of + ½ LSB.

Track and Hold (T/H)
The T/H technique is allowed to "track" the input signal for a period of time prior to initiating a "hold command." During the track period, the output follows the input. The T/H is normally left in track.

Useful Bandwidth
The frequency limit at which, for most measurements, minimal error is encountered. The useful bandwidth is constrained by perceptual aliasing and envelope error.

REFERENCES

1. "Specifying A/D and D/A Converters," 1976, National Semiconductor, An-156.
2. "Understanding High-Speed A/D Converter Specification," 1974, Computer Labs.
3. "Analog-Digital Conversion Notes," 1979, Analog Devices.
4. "Principles of Data Acquisition and Conversion," 1978, DATEL, Modules for Data Conversion Catalog.
5. "Universal Process Interfaces, Camac—vs.—HPIB,®" No. 8, August 1976, Instrumentation Technology.

Milt Ringe, Portable Test Instruments Division, Tektronix, Inc.

Material from the following authors and Tek employees have made significant contributions to this package.

Bruce Miller, Tek Labs
Luis Navarro
Chet Heyberger
Bruce Blair
Ed Averill
John Moore
Jim Larsen
Reinhold Grob
Tom Dagostino
Dave Olsen
Joe Peter
Fred Azinger
Rod Bristol

Tektronix®
COMMITTED TO EXCELLENCE