This page intentionally left blank.

## Models and Optional Features

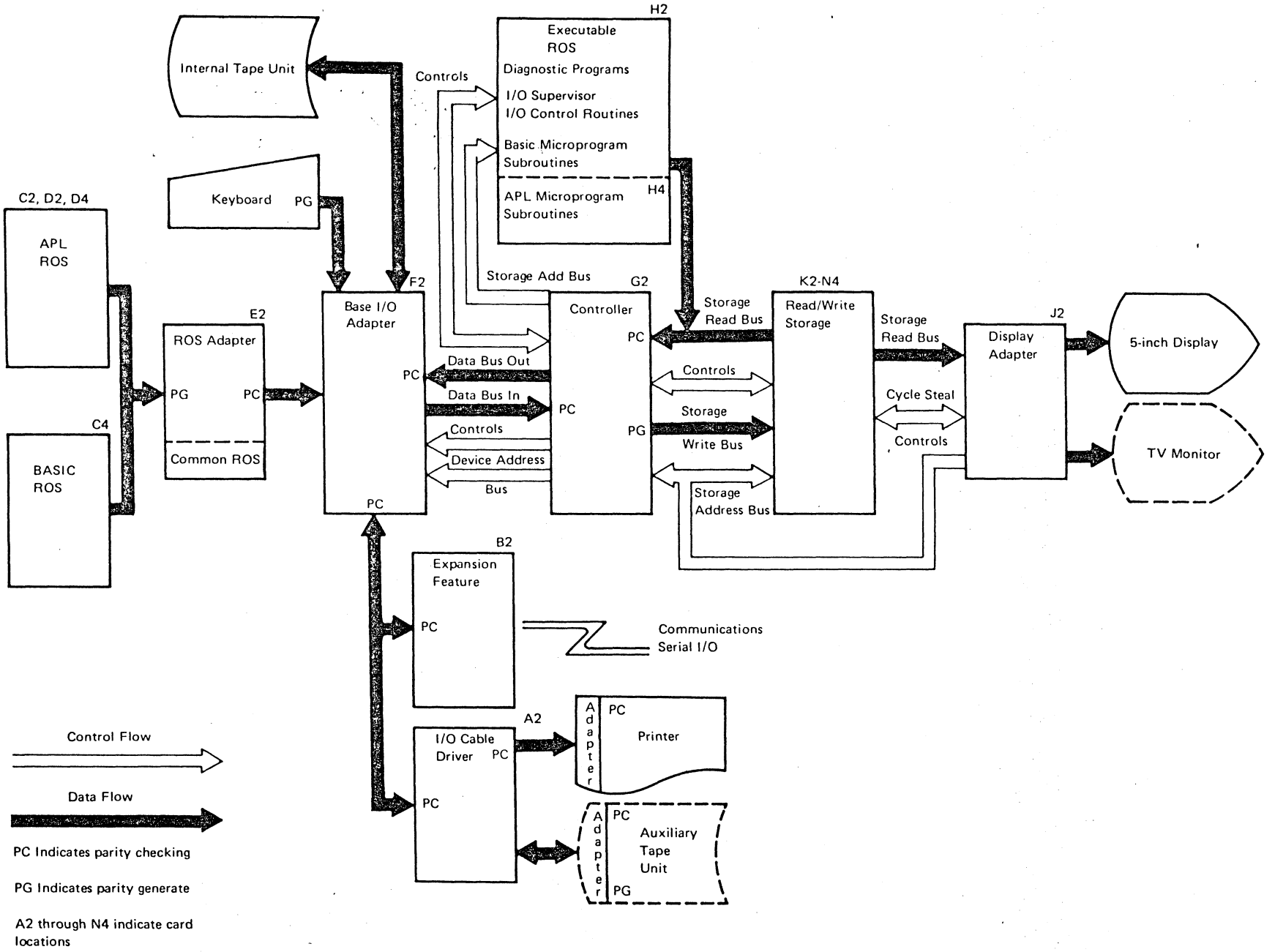
The 5100 Portable Computer is available in three basic models: A, B, and C. Model classification is derived from the programming language used by the 5100. The models are further divided according to the amount of storage available. The following chart shows the models with the various storage combinations.

| Model | Language | Storage |
|---|---|---|
| A1 | APL | 16K |
| A2 | APL | 32K |
| A3 | APL | 48K |
| A4 | APL | 64K |
| B1 | BASIC | 16K |
| B2 | BASIC | 32K |
| B3 | BASIC | 48K |
| B4 | BASIC | 64K |
| C1 | APL and BASIC | 16K |
| C2 | APL and BASIC | 32K |
| C3 | APL and BASIC | 48K |
| C4 | APL and BASIC | 64K |

APL – A programming language
BASIC – Beginners all-purpose symbolic instruction code

The optional features available on the 5100 Portable Computer are listed below. For a description of these features, refer to the last topic in this section, titled *Features.*

5103 Printer
5106 Auxiliary Tape Unit
Expansion Feature
  Communications Adapter
  Serial I/O Adapter


Internal Tape Unit
Controls
H2
Executable ROS
Diagnostic Programs
I/O Supervisor
I/O Control Routines
Basic Microprogram Subroutines
APL Microprogram Subroutines
H4
Keyboard
PG
C2, D2, D4
APL ROS
C4
BASIC ROS
E2
ROS Adapter
PG
PC
Common ROS
F2
Base I/O Adapter
PC
Storage Add Bus
G2
Controller
PC
PC
PG
Data Bus Out
Data Bus In
Controls
Device Address
Bus
Storage Read Bus
K2-N4
Read/Write Storage
Storage Read Bus
Cycle Steal
Controls
Storage Write Bus
Storage Address Bus
J2
Display Adapter
5-inch Display
TV Monitor
B2
Expansion Feature
PC
Communications Serial I/O
A2
I/O Cable Driver
PC
PC
Adapter
PC
Printer
Adapter
PC
Auxiliary Tape Unit
PG
Control Flow
Data Flow
PC Indicates parity checking
PG Indicates parity generate
A2 through N4 indicate card locations

## M 5100 Portable Computer Overview

### CONTROLLER AND CHANNELS

The controller contains all the control and processing capabilities of the 5100 Portable Computer. Microprograms control data flow into and out of the controller via the program controlled I/O channel and the storage and cycle steal channel. Refer to the *IBM 5100 Portable Computer Data Flow* diagram for an overview of the 5100 operation.

The storage and cycle steal channel connects the controller to the executable ROS, read/write storage, and the display adapter. This channel is composed of the following:

- Storage read bus (16 data bits plus 2 parity bits)
- Storage write bus (16 data bits plus 2 parity bits)
- Storage address bus (16 bits)
- Storage control lines to read/write storage and executable ROS
- Cycle steal control lines to the display adapter from read/write storage

The program controlled I/O channel connects the ROS adapter, expansion feature card, I/O cable driver, and I/O devices to the controller through the base I/O adapter. This channel is composed of the following:

- Data bus out (8 data bits plus 1 parity bit)
- Data bus in (8 data bits plus 1 parity bit)
- Storage address bus (16 bits)
- An 8-bit I/O device address bus (2-out-of-8 bit coding for 16 devices)
- Control lines (strobes, tags, and various controller coding for 16 devices)
- Control lines (strobes, tags, and various controller clocks, not shown in the data flow diagram)

### ADAPTERS

The ROS adapter (E2) connects the APL and/or BASIC ROS to the I/O channel. The ROS adapter also contains (in common ROS) CE diagnostics, microprogram routines, and common tables such as the keyboard table used with the I/O microprograms in executable ROS.

The base I/O adapter (F2) connects the controller to the internal tape unit, the keyboard, and the operator control panel (not shown). In addition, the base I/O adapter repowers the I/O channel for use by the ROS adapter, the expansion feature card (if present), and the I/O cable driver (if present). Some of the controller clock signals are supplied to the display as well as to the devices on the I/O channel.

The expansion feature card (B2) allows the 5100 to communicate with a host computer or a serial I/O device. The microprograms that control the communications or serial I/O feature are stored on tape and must be loaded into read/write storage before using the feature.

The I/O cable driver (A2) contains drivers and receivers that connect the 5103 printer and/or 5106 auxiliary tape features (if installed) with the I/O channel.

The display adapter (J2) fetches two characters at a time from the display buffer in read/write storage, stealing storage cycles from the controller for each fetch. The display adapter also generates all the synchronization signals required by the display.

### EXECUTABLE ROS

The microprograms in executable ROS (H2 and H4) are directly executed by the controller. These microprograms control the operation of the 5100 Portable Computer.

The microprogram subroutines in executable ROS are executed by the controller to analyze the instructions in the APL or BASIC interpreter so the controller can perform the APL or BASIC operations requested by the user.

The I/O supervisor and I/O control routines in executable ROS control all I/O functions. When the user statement specifies an I/O function such as a printer or tape operation, the interpreter sets up an IOCB (input/output control block) to request an I/O function and passes control to the I/O supervisor. The I/O supervisor checks the IOCB to determine what I/O device is requested and passes control to the appropriate device I/O control routine to perform the I/O function requested.

Theory

## NONEXECUTABLE ROS

The microprograms in nonexecutable ROS (APL and BASIC ROS) are not directly executed by the controller. They are accessed through the I/O channel and the ROS adapter one byte at a time and placed in a controller work register. The controller then executes the BASIC or APL microprogram subroutines that are located in executable ROS to perform the APL or BASIC language function.

The microprograms in APL ROS (APL interpreter) interpret and control the execution of all APL statements entered by the user.

The microprograms in BASIC ROS (BASIC interpreter) interpret and control the execution of all BASIC statements entered by the user.

## READ/WRITE STORAGE

Read/write storage (K2-N4) contains the work areas, buffers, IOCBs, user's APL or BASIC program, and microcode used by the 5100 internal microprograms. These areas in read/write storage can be displayed or altered for diagnostic purposes (see *DCP1 Functions*).

## MICROPROGRAMS AND DATA FLOW EXAMPLE

Using the following sample program, you can understand how the controller (by executing the microprograms) interprets and controls the execution of a BASIC program:

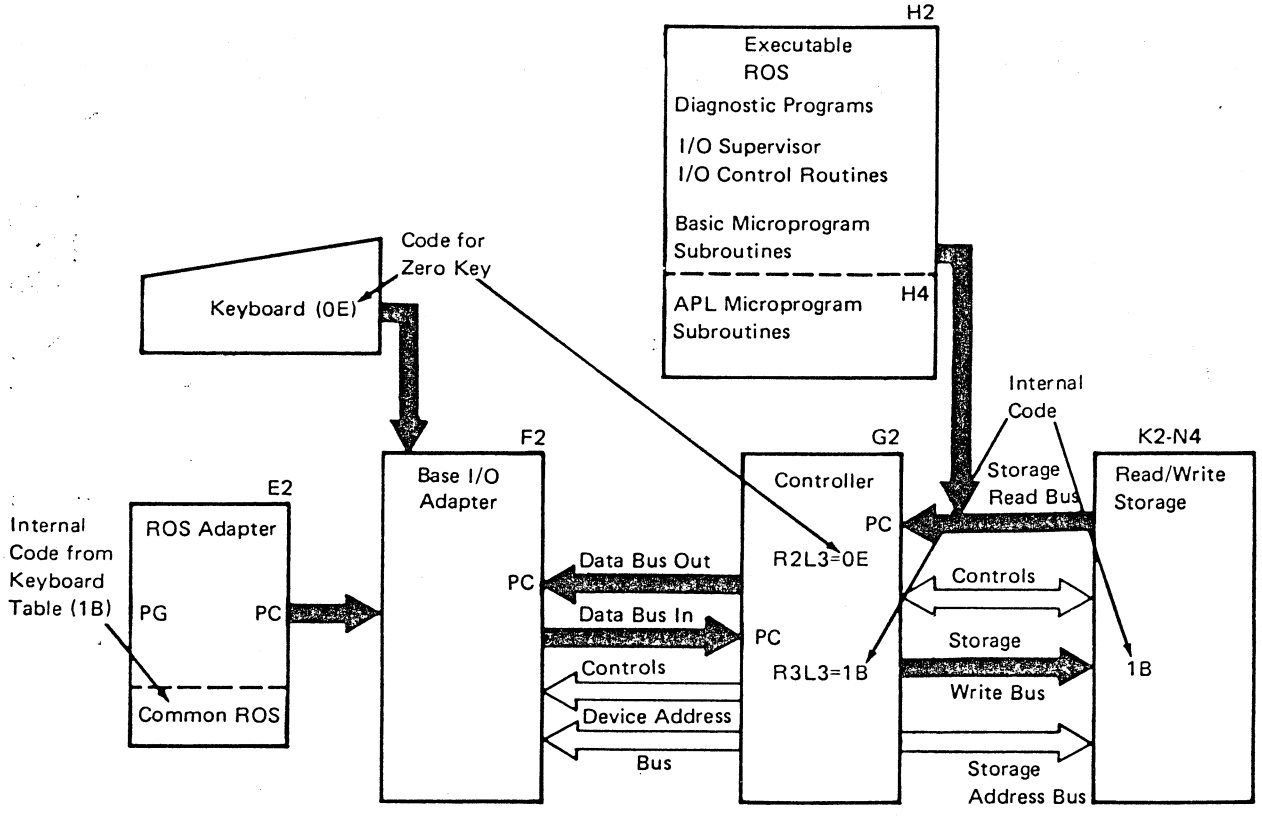
```
0100 A = 2
0020 B = 4
0030 C = A + B
0040 PRINT FLP, C
RUN
```

1. When the 5100 Portable Computer is not processing any user statements or programs, and data is not being entered from the keyboard, the controller executes an I/O supervisor routine that flashes the cursor and waits for a keyboard interrupt.

2. When the first key (0) in the first statement is pressed:

   a. An interrupt occurs indicating a key has been pressed. The code for the zero key (hex 0E) is placed in register 2 level 3 (R2L3) in the controller via the base I/O adapter and bus-in:

   b. The interrupt causes the I/O supervisor routine to suspend flashing the cursor and pass control to the keyboard I/O control routine.

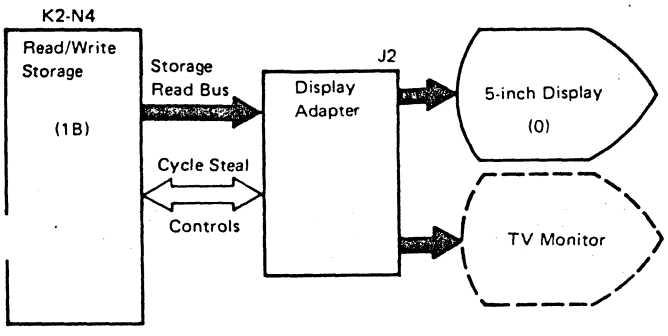


The keyboard I/O control routine converts the code from the 0 key to 5100 internal code that is stored in the keyboard table in common ROS, places it in R3L3, moves it to read/write storage, and resets the interrupt. When the interrupt is reset, control returns to the I/O supervisor.

4. The I/O supervisor then:

   a. Checks the internal code to determine if the key pressed was a data key or a function key.

   b. Moves the internal code to the display screen buffer since the 0 key is a data key, (see *BASIC Data Areas* for the location of the display screen buffer). Whenever data is moved to the display screen buffer, hardware displays the data on the screen via cycle steals:




   c. Resumes flashing the cursor and waits for the next keyboard interrupt.

The rest of the data keys are processed in the same way.

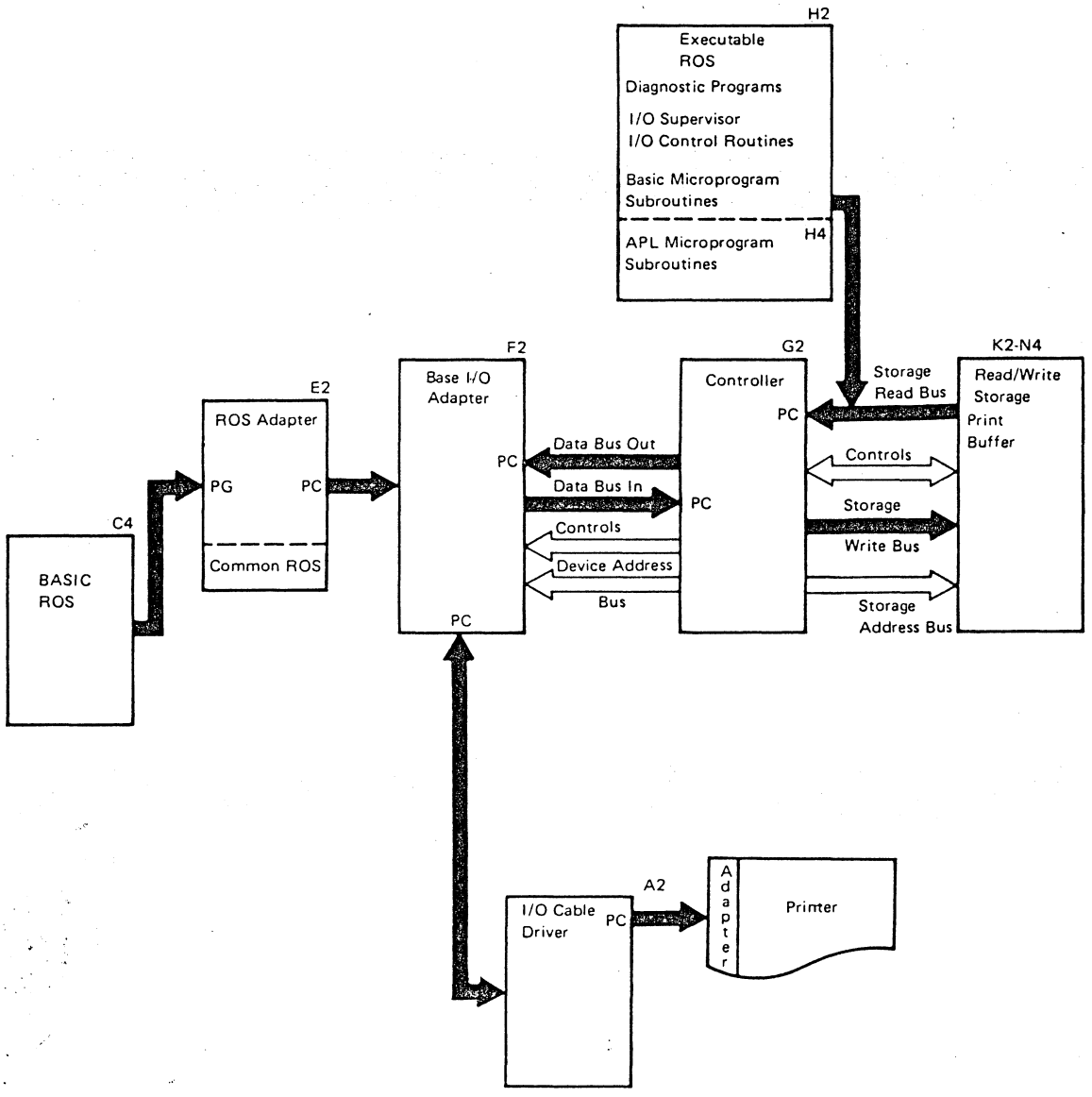
5. When the EXECUTE key is pressed at the end of a statement, the code of the EXECUTE key is moved to read/write storage by the keyboard I/O control routine the same as a data key. However, since the EXECUTE key is a function key, the I/O supervisor passes control to the BASIC microprogram subroutines, also in executable ROS.

6. The controller executes the BASIC microprogram subroutines, which enables it to use the microprograms in the BASIC ROS. The microprograms in the BASIC ROS check the statement in the display screen buffer and store it in the user area in read/write storage. (If this had been a calculator statement, the statement would now be interpreted and executed.) Each statement is stored in the same way until the RUN statement is encountered by the BASIC interpreter; the statements are then interpreted and executed.

7. When the PRINT FLP, C statement is interpreted, the interpreter places information in the IOCB in read/write storage for the print operation and passes control to the I/O supervisor.

8. The I/O supervisor checks the device address and passes control to the printer I/O control routine.

9. The printer I/O control routine prints the data, places a return code in the IOCB, and returns control to the I/O supervisor. The data flow is from the print buffer in read/write storage, to the printer, via the controller, base I/O adapter, I/O cable driver and printer adapter.




10. The I/O supervisor checks the return code for any errors that occurred during the print operation and, if no errors occurred, clears the print buffer to blanks and returns control to the BASIC microprogram subroutine.

11. The BASIC microprogram subroutines determine that there are no more statements to be interpreted, and return control to the I/O supervisor.

Essentially, the control and data flow is the same when using APL.

## ( ,trol Unit

The control unit consists of the controller, portions of base I/O, and all storage. The control unit contains the microprograms and the logic necessary to execute them. The microprograms are located in ROS (read only storage), which retains its contents when power is turned off. The contents of ROS are not destroyed or altered by any machine function.

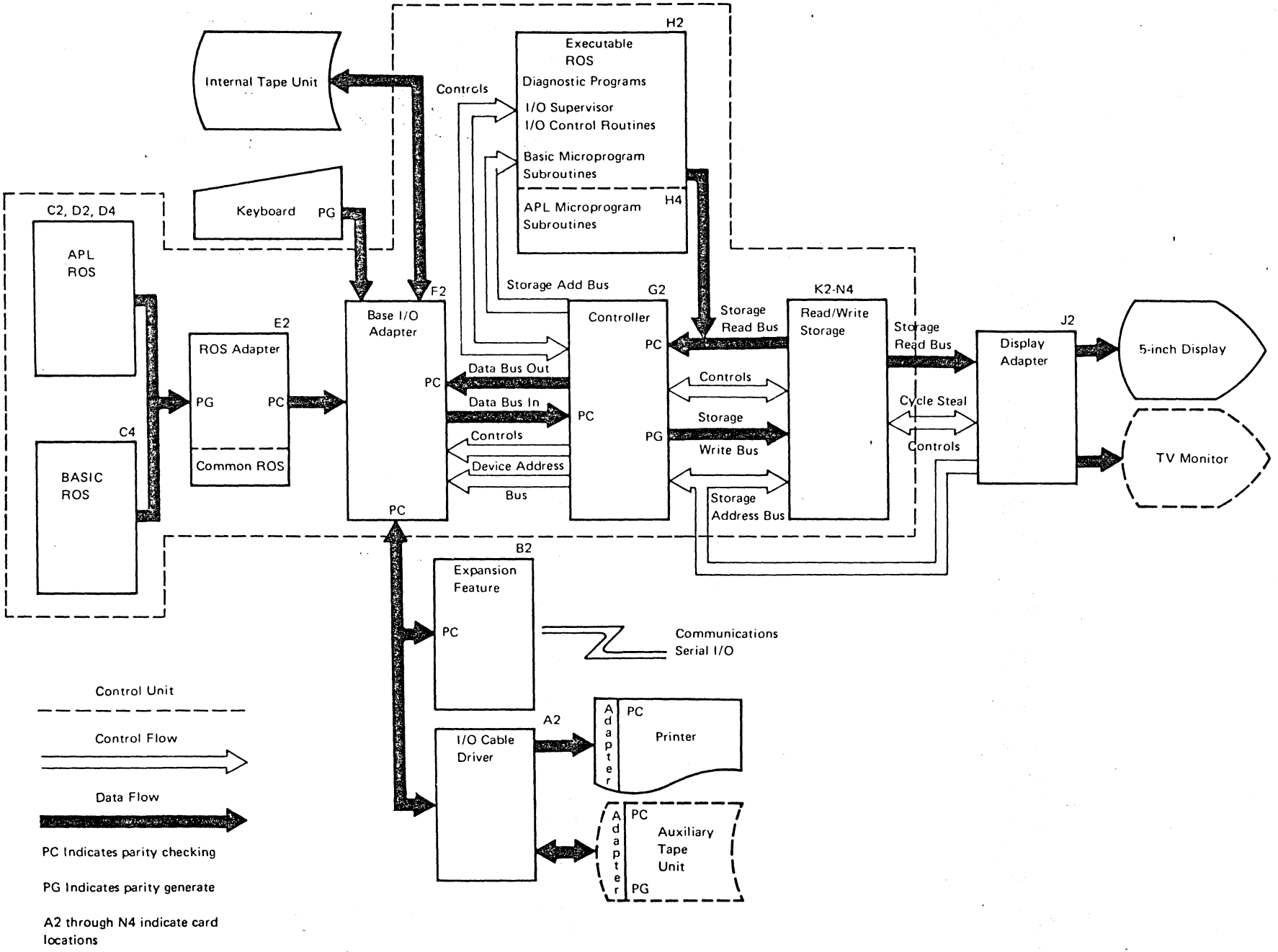
The 5100 Portable Computer contains two kinds of ROS, executable ROS and nonexecutable ROS. The controller (processing unit) executes microinstructions from executable ROS or from read/write storage. Microinstructions residing in nonexecutable ROS are first loaded into read/write storage and executed from there.

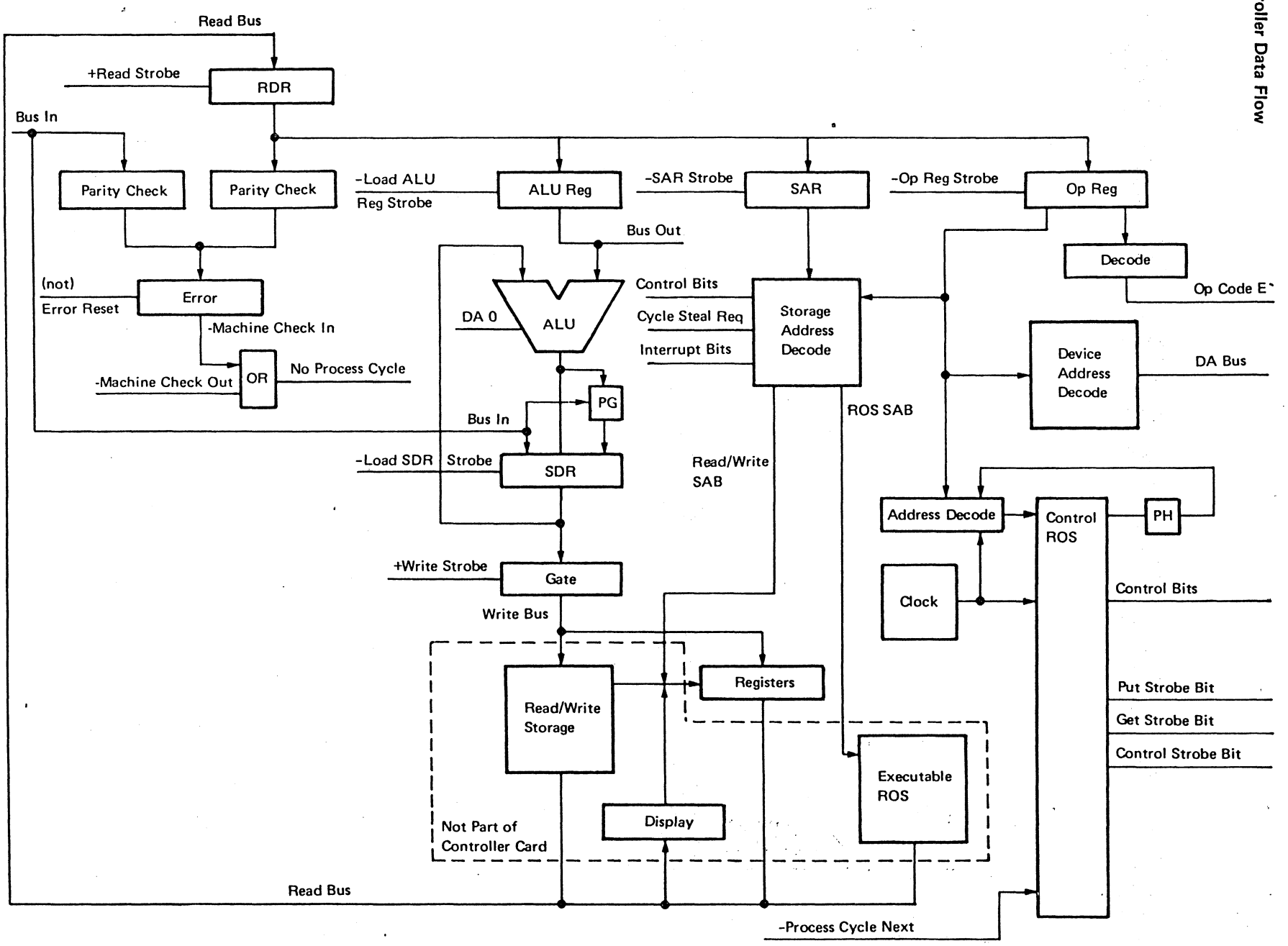
The IBM 5100 Data Flow diagram shows that the controller accesses executable ROS on the same channel as it accesses read/write storage. The controller accesses nonexecutable ROS through the I/O channel via the base I/O adapter card and the ROS adapter card.

Besides the microinstructions in the APL ROS and BASIC ROS, the nonexecutable ROS contains data in the common
S located on the ROS adapter card.

BASIC or APL statements are entered by the 5100 Portable Computer user into read/write storage (via the keyboard and tape). The user statements are interpreted and executed by the APL or BASIC interpreters under control of the APL or BASIC microprogram subroutines that reside in executable ROS. Portions of the APL interpreter or the BASIC interpreter are loaded from nonexecutable ROS into read/write storage by the bring up program.

All I/O operations (for example, when a key is pressed or the user program calls for printed output), except for serial I/O operations loaded from the serial I/O cartridge, are controlled by I/O subroutines residing in executable ROS. Executable ROS contains diagnostic programs and the bring up program (which also contains some diagnostic routines). For more detail, refer to the descriptions of the diagnostic and bring up programs.


Internal Tape Unit
Controls
H2
Executable ROS
Diagnostic Programs
I/O Supervisor
I/O Control Routines
Basic Microprogram Subroutines
APL Microprogram Subroutines
H4
Keyboard
PG
C2, D2, D4
APL ROS
C4
BASIC ROS
E2
ROS Adapter
PG
PC
Common ROS
F2
Base I/O Adapter
PC
PC
Storage Add Bus
G2
Controller
PC
PC
PG
Data Bus Out
Data Bus In
Controls
Device Address Bus
Storage Read Bus
Storage Write Bus
Storage Address Bus
K2-N4
Read/Write Storage
Storage Read Bus
Cycle Steal
Controls
J2
Display Adapter
5-inch Display
TV Monitor
B2
Expansion Feature
PC
Communications Serial I/O
A2
I/O Cable Driver
Adapter
PC
Printer
Adapter
PC
Auxiliary Tape Unit
PG
Control Unit
Control Flow
Data Flow
PC Indicates parity checking
PG Indicates parity generate
A2 through N4 indicate card locations


Read Bus
+Read Strobe
RDR
Bus In
Parity Check
Parity Check
-Load ALU
Reg Strobe
ALU Reg
-SAR Strobe
SAR
-Op Reg Strobe
Op Reg
Bus Out
Decode
Op Code E
(not)
Error Reset
Error
-Machine Check In
DA 0
ALU
Control Bits
Cycle Steal Req
Interrupt Bits
Storage
Address
Decode
Device
Address
Decode
DA Bus
-Machine Check Out
OR
No Process Cycle
PG
Bus In
ROS SAB
-Load SDR
Strobe
SDR
Read/Write
SAB
Address Decode
Control
ROS
PH
+Write Strobe
Gate
Clock
Control Bits
Write Bus
Registers
Read/Write
Storage
Put Strobe Bit
Get Strobe Bit
Control Strobe Bit
Executable
ROS
Not Part of
Controller Card
Display
Read Bus
-Process Cycle Next

## CONTROLLER

The controller is the 5100 Portable Computer processing unit. The controller uses an internal microprogram to control the internal functions. The controller communicates with read/write storage and executable ROS via the storage address bus and with I/O devices such as the keyboard, printer, and nonexecutable ROS via bus in, bus out, and various control lines.

The Controller Data Flow diagram shows the internal organization of the controller. The clock and control ROS provide the signals to operate the controller. In addition, control ROS provides pulses used by the I/O devices (these control pulses are described under *I/O Data Flow Control* in this section).

The registers shown in parallel with read/write storage are 128 bytes of high speed read/write storage located on the controller card. They displace the first 128 addresses of read/write storage. Four groups of registers, each containing 16 halfword (2 byte) registers, are associated with four levels of program processing. The program levels are determined by interrupts (refer to *Interrupts* in this section).

The RDR (read data register) and bus in are parity checked on the controller card. A parity error causes a process check, which stops the machine and turns the PROCESS CHECK light on. Parity errors detected on other cards can also cause a process check. The 5100 Portable Computer can operate with the RDR and bus in parity checking disabled as explained under *Operation* in this section.

### Operation

When the 5100 Portable Computer is turned on, or if it is already on and RESTART is pressed, the controller begins executing the bring up program in executable ROS using the registers associated with program level 0. Interrupts are disabled during the bring up program (keeping the controller in level 0), except for testing; and are enabled just before completing the bring up program. RDR and bus in parity checking is disabled at the beginning of the test and enabled after a few microinstructions have been executed.

The bring up program first determines the amount of read/write storage installed by writing a byte of all 0's (zeros) to the lowest byte of each possible storage increment (every 16K bytes) and then reading it back. If the read/write storage increment is installed, 0's are read back. If it is not, all 1's (ones) except the parity bit are read back. As soon as the read/write storage size is determined, all parity checking is enabled.

The various diagnostic tests are made as described under *Diagnostic Aids* in Section 3. During the bring up program, the setting of the BASIC-APL switch is sampled and the common ROS and either the BASIC interpreter or the APL interpreter is loaded into read/write storage. More diagnostic tests follow. When the bring up program is completed, all interrupts are enabled and control is passed to the APL or BASIC supervisor program.

### Interrupts

The 5100 Portable Computer has an interrupt system that allows interruption of microprogram processing by the I/O devices. There are four program levels in which processing occurs. The program levels and their associated I/O devices are:

Level 0 – Normal operation
Level 1 – Communications adapter/serial I/O adapter
Level 2 – Tape and printer
Level 3 – Keyboard

Each level has 16 halfword registers located on the controller. These registers are addressable as the lowest 128 bytes of read/write storage. Register 0 of each level is the IAR (instruction address register). Refer to *Microinstruction Processing* in this section. The remaining 14 registers can be used as general purpose registers in each program level.

The bring up program initializes the controller and begins program execution in level 0. Switching between program levels is controlled by the I/O devices through the interrupt request lines. The I/O devices are assumed by the controller to be connected in priority with the highest priority connected to the highest numbered line. Therefore, when two or more interrupt requests are active simultaneously, the controller responds only to the one with the highest number.

The controller inspects the interrupt request lines for a higher numbered interrupt after completing each microinstruction. For example, if the controller is processing in level 0, and upon completion of the current microinstruction finds interrupt request lines 1 and 3 active, three levels (0, 1 and 3) are in contention for the controller. The controller selects level 3 because it is the highest number in contention, and executes the next microinstruction using the registers associated with level 3.

### :roinstruction Processing

The operation of each microinstruction is divided into I cycles (fetch) and E cycles (execute). During I cycles the microinstruction address is routed from register 0 of the current program level through the RDR (refer to the Controller Data Flow diagram in this section) to the ALU (arithmetic logic unit), and to the SAR (storage address register). The ALU adds two to the microinstruction address (to get the next sequential microinstruction address), which is then routed through the SDR (storage data register) back to register 0. The microinstruction is fetched (using the address in SAR) from read/write storage or executable ROS to the ALU register and the operation register. This completes the I cycles, which are exactly the same for every microinstruction.

The E cycles vary depending on the microinstruction executed. The data in the operation register (the operation codes and modifiers described in the individual microinstruction) forms part of the address for the control ROS that operates the controller.

The data in the ALU register is either ignored, sent out on bus out, or is processed by the ALU depending on the ·icroinstruction. At the completion of the E cycles for one microinstruction, the I cycles of the next microinstruction begin.

### Microinstruction Fetch Switch

Each microinstruction consists of a halfword in either read/write storage or executable ROS. The microinstruction address in register 0 does not distinguish between the two. However, a line ('-select ROS', G2-B04) is down when microinstructions are fetched from executable ROS and is up when they are fetched from read/write storage.

Although microinstructions are fetched from either read/write storage or executable ROS, the data (addressed during E cycles) for processing the microinstructions is always in read/write storage.

Switching of the line that determines whether the microinstructions are fetched from read/write storage or executable ROS ('-select ROS') is accomplished with a control microinstruction.

The switch is always made at location hex 0600 for an ROS to read/write storage switch and at location hex 0604 for a read/write storage to ROS switch. The switch microinstructions in read/write storage are loaded from common ROS by the bring up program.

When the control microinstruction at hex 0600 in ROS is executed, the next microinstruction executed is at hex 0602 in read/write storage. Hex 0602 in ROS is a halt microinstruction that stops processing if the switch fails. After an ROS to read/write storage switch at hex 0600, subsequent microinstructions are fetched from read/write storage until a program in executable ROS needs to be called. After return addresses and other parameters are set up, the program running in read/write storage branches to hex 0604 (the location of the read/write storage to ROS switch). The switch is executed and the next and all subsequent microinstructions are fetched from ROS until a switch is made by branching to hex 0600 in ROS. Like hex 0602 in ROS, hex 0606 in read/write storage is a halt microinstruction.

### I/O Data Flow Control

The controller provides a microinstruction controlled interface for transferring data between the I/O devices and the control unit.

The controller receives data from the I/O devices on the 9 bit (8 data, 1 parity) bus in from the base I/O card. Data is sent to the I/O devices on the 9 bit bus out to the base I/O card. Control, put, and get strobes and the op code E tag signal the I/O devices when I/O microinstructions are executed.

The control strobe has two purposes. It identifies the data on bus out as control data. It also serves as a timing pulse to indicate to the I/O device exactly when the bus out data is valid.

The put strobe also has two purposes. Besides serving as a timing pulse, it identifies the bus out as containing data for a put instruction.

The get strobe signals the I/O device that the data placed on bus in by the device was sampled by the controller.

The op code E tag is a decode of the get byte microinstruction (op code E). The decode occurs well before the get strobe and allows the I/O device to distinguish the get byte microinstructions (op code E) intended primarily for data from the logical get microinstructions (op code 0) intended for status. When this check is active, the DA lines are also active to allow probing of the device causing the error.

*Error Checking*

All errors that cause process checks (stop the machine with the PROCESS CHECK light on) are funneled through the controller card. A process check can be caused by any of the following errors:

*-Rd data error (G2-S08)* — This is a parity error on data in the read data register of the controller. Cards that can cause this error are the read/write storage (K2, K4, L2, L4, M2, M4, N2, and N4) cards, the controller (G2) card, the display (J2) card, the BASIC, I/O, and diagnostic (H2) card, and the APL supervisor (H4) card.

A parity error, resulting in an Rd data error, occurs if there is an attempt to read from a read/write storage address for which the read/write storage cards are not installed. This means that if less than 64K of read/write storage is installed (see *Read/Write Storage*), an error in the microprogram can cause a process check.

*-Bus in error (G2-U09)* — This error is a parity check on bus in. Cards that can cause this error are the controller (G2) card, the base I/O (F2) card, the ROS adapter (E2) card, the expansion feature (B2) card, and the I/O cable driver (A2) card. In addition, this error can be caused by the following I/O devices: keyboard, tape unit, printer, or auxiliary tape unit.

*+Address check (F2-B13)* — This is a device address check on the base I/O (F2) card. The base I/O (F2) card and the controller (G2) card can cause this error.

*+Address check, ROS adapter (E2-D10)* — The device addresses are checked at the ROS adapter (E2) card and the error can be probed there.

*+Address check, expansion feature (B2-J13)* — The device addresses are checked at the expansion feature (B2) card and the error can be probed there.

*+Address check, printer (P02)* — The device addresses are checked at the printer adapter card and the error can be probed there.

*+Address check, auxiliary tape (P05)* — The device addresses are checked at the auxiliary tape adapter card and the error can be probed there.

Only one of the device address lines X0, X1, X2, X3 and only one of the device address lines Y0, Y1, Y2, Y3 are supposed to be up when addressing a device. The device address check, however, occurs when any odd number of all eight lines is up when addressing a device.

*+Bus out parity check (F2-D13)* — This is a parity check on bus out on the base I/O (F2) card. The controller (G2) card or the base I/O (F2) card can cause this error.

*+Bus out parity check, ROS adapter (E2-B11)* — The parity of bus out is tested on the ROS adapter (E2) card and the error can be probed there.

*+Bus out parity check, expansion feature (B2-M02)* — The parity of bus out is tested on the expansion feature (B2) card and the error can be probed there.

*+Bus out parity check, printer (P11)* — The parity of bus out is tested on the printer adapter card and the error can be probed there.

*+Bus out parity check, auxiliary tape (P04)* — The parity of bus out is tested on the auxiliary tape adapter card and the error can be probed there.

**Microinstructions**

I/O control and the high level languages (APL and BASIC) are implemented with microinstructions in read/write storage and executable ROS. All 5100 Portable Computer microinstructions are a halfword (2 bytes). The first 4 bits of the halfword is the op code. The meaning of the remaining 12 bits depends on the op code. Some op codes have a modifier (bits 12-15) that expands the number of microinstructions beyond 16.

## STORAGE

Storage in the 5100 Portable Computer consists of executable ROS (read only storage), read/write storage, and nonexecutable ROS.

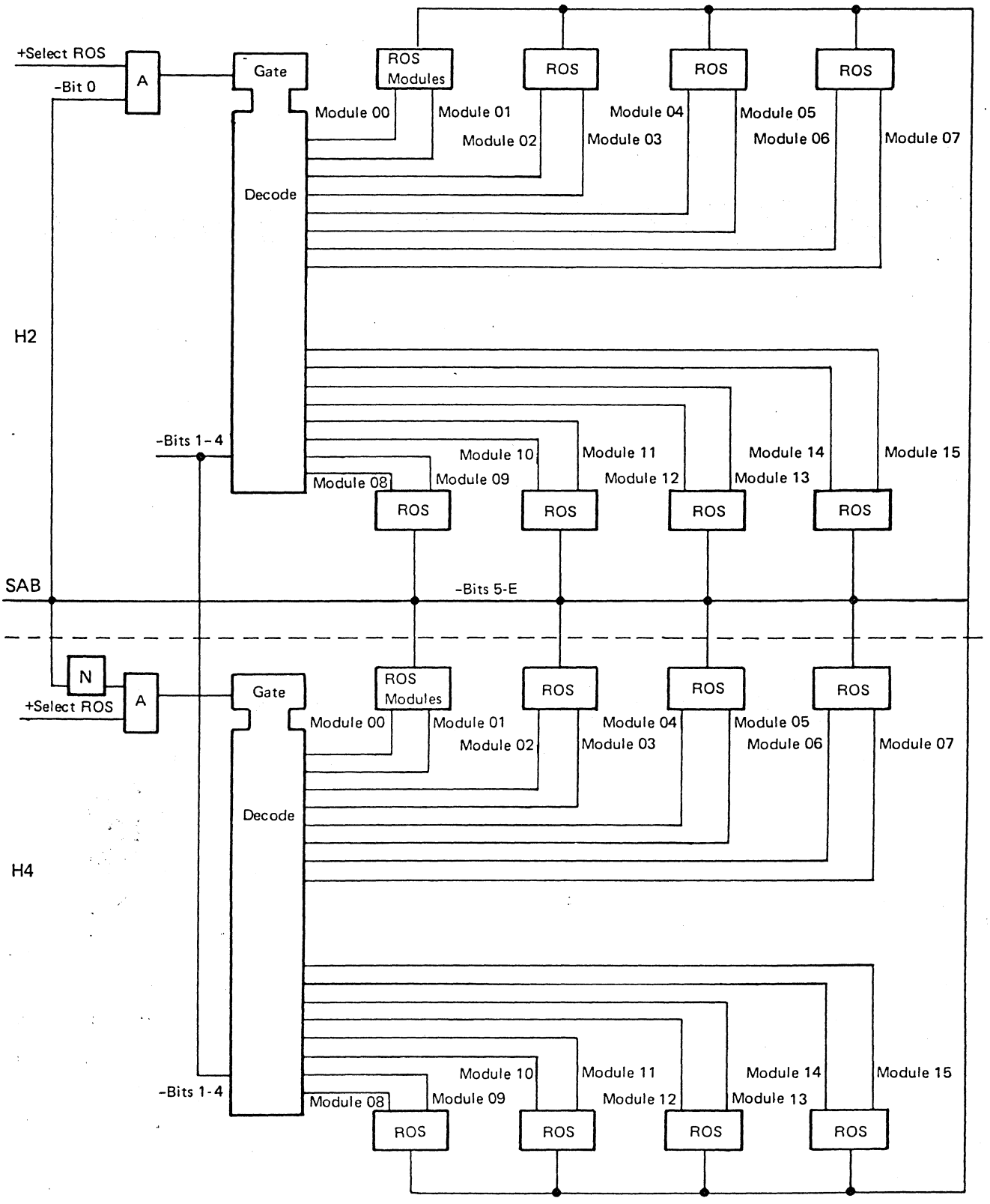
### Executable ROS

Executable ROS provides rapidly accessible resident storage for frequently used programs. Executable ROS contains the bring up program (a diagnostic program), the ROS-read/write storage switches, the I/O supervisor and I/O control routines, and the BASIC and APL micro-program subroutines. The BASIC, I/O, and diagnostic (H2) card and the APL supervisor (H4) card house the executable ROS in the 5100 Portable Computer.

The controller (G2) card addresses executable ROS via the storage address bus (SAB). The 16 SAB bits are represented by 0123 4567 89AB CDEF. Bit 0 is the card select bit, a down level selects the APL supervisor (H4) card and an up level selects the BASIC, I/O, and diagnostic (H2) card. Bits 123 4 select one of the 16 chips on each card. Bits 567 89AB CDE select the address location on the chip. Bit F is not used because each microinstruction is fetched as a halfword (2 bytes) and an even-odd pair of bytes is fetched.

Executable ROS contains no data other than micorinstructions.

Each byte of data in executable ROS contains a parity bit, which is checked in the read data register in the controller as the microinstructions are fetched. An error causes a process check (Rd Data Error).


+Select ROS
-Bit 0
A
Gate
ROS Modules
ROS
ROS
ROS
Module 00
Module 01
Module 02
Module 03
Module 04
Module 05
Module 06
Module 07
Decode
H2
-Bits 1-4
Module 08
Module 09
Module 10
Module 11
Module 12
Module 13
Module 14
Module 15
ROS
ROS
ROS
ROS
SAB
-Bits 5-E
N
+Select ROS
A
Gate
ROS Modules
ROS
ROS
ROS
Module 00
Module 01
Module 02
Module 03
Module 04
Module 05
Module 06
Module 07
Decode
H4
-Bits 1-4
Module 08
Module 09
Module 10
Module 11
Module 12
Module 13
Module 14
Module 15
ROS
ROS
ROS
ROS

*:ecutable ROS Routines*

The following routines and subroutines in executable ROS have hexadecimal addresses as listed below. The hex numbers in column 1 are the routine address and the hex numbers in column 2 are subroutine addresses within a routine.

| Routine Address | Subroutine Address | Routine Description |
|---|---|---|
| 0000 | | Bring up routine: Refer to Section 3 for a list of the bring up routine tests.<br><br>The following four tests, which are automatically run once during the bring up program, can be entered for looping by using the branch or call function from the DCP1 diagnostic mode program. (Refer to *DCP1 Diagnostic Mode Functions* in Section 3.) |
| | 0096 | Loop on op code test. |
| | 02AE | Loop on ROS read back test: This test checks the capability to read back the last address accessed. APL or BASIC addresses are tested depending on the position of the APL-BASIC switch when the bring up program was last run. Addresses in each module are sent to the ROS adapter, various bytes are read back to step the address, and the resulting addresses are read back and compared with the expected result. The test is repeated until terminated by the operator. |
| | 02B2 | Loop on ROS CRC and sequence test: This test checks the CRC and the sequence number of the common ROS module (sequence number 18), the BASIC ROS modules (sequence numbers 10 through 17), or the APL ROS modules (sequence numbers 20 through 2F) depending on the position of the APL-BASIC switch when the bring up program was last run. The test is repeated until terminated by the operator. |
| | 0430 | Loop on read/write storage content test: This test checks read/write storage from address hex 0100 through the last address installed. First hex 55, hex AA, and hex D6 are stored and read back from each address once. Then numbers hex 00 to hex FF, hex 01 to hex FF, etc, are put in each address. When read/write storage is filled, they are read back and compared. The data is written again, but shifted by one position. This part of the test is continually looped on so that eventually (time depends on the amount of read/write storage) every possible byte is stored in every address. This part of the test continues until terminated by the operator. |
| 0600 | | ROS to read/write storage and read/write storage to ROS switches. |
| 06D0 | | Diagnostic tape load/dump control routine. |
| 0AA0 | | DCP1 (diagnostic control program 1). Refer to *Diagnostic Aids*, Section 3. |
| 1120 | | Program level 1 code for communications. The rest of the communications code is in read/write storage. |
| 1200 | | BASIC microprogram routines. |
| 5000 | | I/O supervisor routine. |

| Routine Address | Subroutine Address | Routine Description |
|---|---|---|
| 5B00 | | Program level 3 code for the keyboard. |
| 5EE0 | | I/O routine that gets data from nonexecutable ROS. The user sets up a parameter block containing addresses, byte count, and other pertinent data needed by this routine. |
| 6000 | | Printer I/O routine. |
| 6800 | | I/O tape read/write routine. |
| 8000 | | APL microprogram routines. |

### Read/Write Storage

Each read/write storage card contains 8K bytes of byte addressable read/write storage. Since halfwords are also addressed by some microinstructions, the storage is organized so that the cards K2, L2, M2, and N2 contain the even bytes and K4, L4, M4, and N4 contain the odd bytes. The following read/write storage configurations are allowed:

| Storage | Cards |
|---|---|
| 16K | K2, K4 Base machine size |
| 32K | K2, K4, L2, L4 |
| 48K | K2, K4, L2, L4, M2, M4 |
| 64K | K2, K4, L2, L4, M2, M4, N2, N4 |

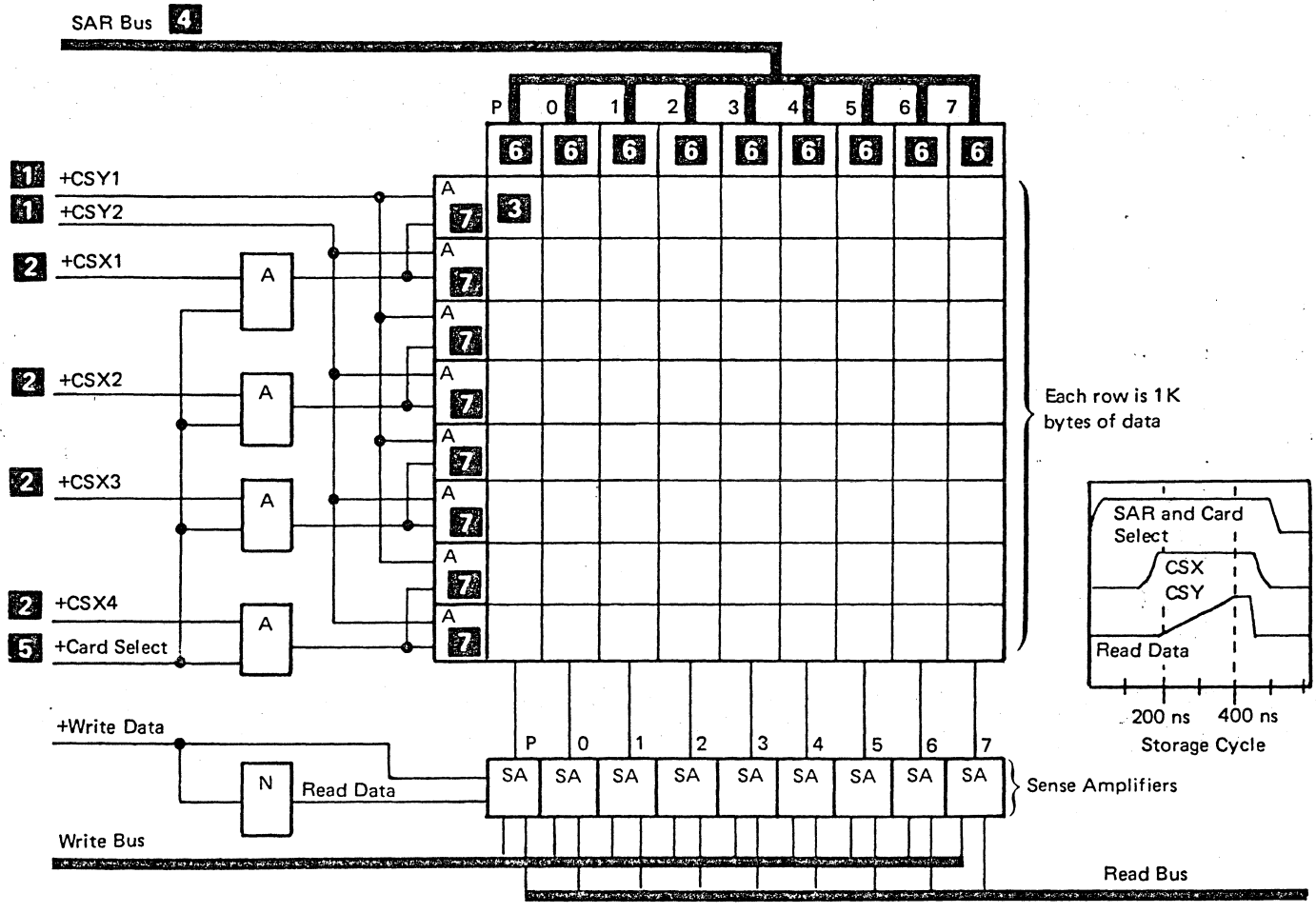
**CAUTION**
The machine must be powered down before you remove or add read/write storage cards; otherwise, they can be damaged.

Defective read/write storage cards can sometimes be found by removing pairs of cards and observing if the smaller configuration is free of the failure (see the preceding caution).

Because the lowest 128 bytes of read/write storage are the registers and are contained on the controller (G2) card, the first 64 bytes of K2 and the first 64 bytes of K4 are not accessible and the machine might run error free even if these bytes are defective.

Read/write storage is nondestructive; that is, data is not changed in read/write storage when read. When power is turned on, data in read/write storage can be anything and, in general, will not have correct parity. However, the bring up program routine writes data into every available byte of read/write storage (including the registers) so that if the bring up program runs to completion, every byte of read/write storage has correct parity if it is not defective.

*Note:* See *Error Checking* for information about how the internal programs can cause an apparent read/write storage error.




1 Only one of these lines should be up on a storage cycle. These two lines are decoded from bit 4 of the address.

2 These four lines are decoded from bits 3 and 5 of the storage address. Only one line should be up.

3 Each square represents 1,024 bits.

4 See the controller logic for the address decode of these lines.

5 This line is decoded from bits 0 and 1 of the storage address.

6 Decodes the 10 SAR lines to select one of 1,024 bits in each block.

7 Only one AND is active to select one byte of data during each cycle.

The controller (G2) card or the display (J2) card address read/write storage via the storage address bus. The data on the storage address bus is decoded into the storage address. Let the halfword address on the storage address bus be represented by 0123 4567 89AB CDEF. If all of bits 0-8 are a logical 0, then the read/write storage (registers) on the controller card is addressed by the remaining bits, 9-E.

If any of bits 0-8 is a logical 1, then the data is addressed as follows. Bits 0 and 1 are decoded into 'card select' lines according to the following table:

| Address Bits 0 | 1 | Card Select Lines | Read/Write Storage Cards |
|---|---|---|---|
| 0 | 0 | Card Select 0 | K2, K4 |
| 0 | 1 | Card Select 1 | L2, L4 |
| 1 | 0 | Card Select 2 | M2, M4 |
| 1 | 1 | Card Select 3 | N2, N4 |

Each read/write storage card contains 72 linear arrays of 1,024 bits each. These are further grouped into groups of 9 by 1,024 bits. The SAR, CSX, and CSY inputs are used to address a byte of storage on the read/write storage card. One of the four CSX and one of the two CSY together select one of the 8 groups. The 10 SAR inputs select one 9 bit byte from the 1,024 bytes in the group.

| Address Bits 3 | 4 | 5 | Input Lines |
|---|---|---|---|
| 0 | – | 0 | CSX1 |
| 0 | – | 1 | CSX2 |
| 1 | – | 0 | CSX3 |
| 1 | – | 1 | CSX4 |
| – | 0 | – | CSY1 |
| – | 1 | – | CSY2 |

The SAR lines are developed from the remainder of the data bits.

| Data Bits | F | E | D | C | B | A | 9 | 8 | 7 | 6 | 2 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| SAR Line | – | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

Bit F is used only during the write microinstruction. If bit F is 0, the 'write even' line becomes active; if bit F is 1, the 'write odd' line becomes active. The 'write even' line goes to all of the even cards (K2, L2, M2, N2) and the 'write odd' line goes to all of the odd cards (K4, L4, M4, N4). When writing halfwords of data, both lines become active.

*Note:* During I cycles the 'select ROS' line acts as another bit of address, which selects either read/write storage or executable ROS.

*Cycle Stealing*

Read/write storage address cycles can be stolen by the display (J2) card to obtain data from read/write storage to display on the 5-inch display and/or the TV monitor. Two consecutive cycles cannot be stolen.

When the DISPLAY REGISTERS switch is pressed, the data from read/write storage addresses 0 through hex 1FF (decimal 511) are displayed in hexadecimal.

When NORMAL is pressed, character data from read/write storage addresses hex 200 (decimal 512) through hex 5FF (decimal 1,535) are displayed.

Cycle steals can be disabled or enabled by a control micro-instruction. 'I/O display off' is the signal line affected. When cycle steals are disabled, the display is blank and the IN PROCESS light is on. When cycle steals are enabled, the display has characters from read/write storage and the IN PROCESS light is off. Cycle steals are automatically enabled if a PROCESS CHECK occurs.

**Nonexecutable ROS**

Nonexecutable ROS contains the language interpreters, the tape read diagnostic microprogram, and the keyboard translate tables. Nonexecutable ROS is accessed through the base I/O adapter and the ROS adapter and is technically an I/O device. Nonexecutable ROS is controlled with the I/O microinstructions. Nonexecutable ROS has device address 1 (DA = 1). The control microinstruction is used for subdevice addressing (APL ROS, BASIC and common ROS).

Nonexecutable ROS contains the microprograms for implementing the APL interpreters, the BASIC interpreter and the tape read diagnostic microprogram. These are all addressed (and all data passes from them) through the ROS adapter (E2) card. All data to the ROS adapter (E2) card passes through the base I/O (F2) card.

*Nonexecutable ROS Addressing*

The ROS adapter (E2) card provides an interface between the base I/O (F2) card and the ROS storage cards. The ROS adapter contains an ROS address register that is set to an address by two consecutive microinstructions. Bit 15 of the address is set to a logical 0 so that the ROS address register always addresses an even address to start. The ROS adapter card automatically updates the ROS address register so that the microinstruction need be issued only at the beginning of each block of data to be read. Two microinstructions are issued to retrieve a halfword of data.

The contents of the ROS address register are read by issuing two consecutive microinstructions. The address returned is always even (bit 15 = 0) and bit 15 of the ROS address register is reset to logical 0 if it is not already 0. The ROS address register read function is exercised by the bring up microprogram.

*Error Checking*

The nonexecutable ROS storage is packaged in 6K ROS modules. Several modules are mounted on each ROS storage card. Three are also mounted on the ROS adapter (E2) card. Refer to the following chart.

| Sequence Number[3] | ROS Module Card |
|---|---|
| 10, 11, 12, 13, 14, and 15 | C4 (BASIC ROS) |
| 16, 17, and 18 | E2 (ROS adapter) |
| 20, 21, 22, 23, and 24 | D2 (APL ROS 1) |
| 25, 26, 27, 28, and 29 | D4 (APL ROS 2) |
| 2A, 2B, 2C, 2D, 2E, and 2F | C2 (APL ROS 3) |

The last two halfwords of each ROS module contain error checking data. The last halfword of each ROS module contains a two-byte CRC (cyclic redundancy check) generated over the entire module. The second to last halfword contains hex 00 in the high order byte and a module sequence number in the low order byte. The CRC and the sequence number are verified by the bring up program. Only APL or BASIC ROS is tested depending on the position of the APL-BASIC switch.

BASE I/O

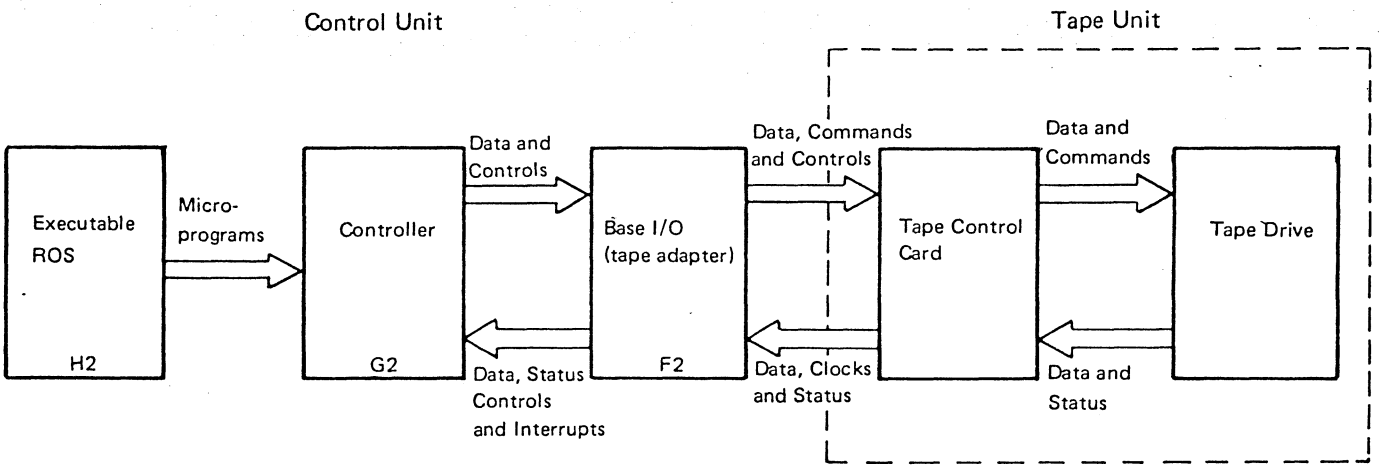
The base I/O (F2) card provides the interface between the controller and the various I/O devices. Data is sent to devices on bus out with put microinstructions. Data is received on bus in with get microinstructions. In either case, the specified device address (DA) selects the I/O device. The base I/O card gates the data to the I/O device from the controller and vice versa. Error checking for base I/O is described in *Error Checking* under *Controller* in this section.

| Device Address | Address Lines | Device Name |
|---|---|---|
| 0 | X0Y0 | Controller |
| 1 | X0Y1 | Nonexecutable ROS[1] |
| 2 | X0Y2 | Not assigned |
| 3 | X0Y3 | Not assigned |
| 4 | X1Y0 | Keyboard and the APL-BASIC switch |
| 5 | X1Y1 | Printer |
| 6 | X1Y2 | Not assigned |
| 7 | X1Y3 | Not assigned |
| 8 | X2Y0 | Expansion feature[2] |
| 9 | X2Y1 | Not assigned |
| A | X2Y2 | Not assigned |
| B | X2Y3 | Not assigned[1] |
| C | X3Y0 | Not assigned[1] |
| D | X3Y1 | Not assigned[1] |
| E | X3Y2 | Tape units[1] |
| F | X3Y3 | All I/O |

[1] Device has subdevice addressing capability.
[2] Hardware device address is 8. Microprogram device address is A for serial I/O.
[3] Sequence numbers appear on the display screen during bring up.

## TAPE UNIT OVERVIEW

The tape unit stores and retrieves information by writing information on the tape for later use and by reading information from the tape for immediate use. The following diagram shows the data flow between the control unit and the tape unit.




### Data Flow

Microprograms located in executable ROS and executed by the controller control the writing and reading of tape. The microprograms generate data and control signals in the controller that are sent to the tape adapter on the base I/O card.

The tape adapter uses the data and control signals from the controller to send commands, data, and control signals to the tape control card. The tape adapter also returns data and status to the controller.

Commands from the tape adapter determine the tape operation within the tape unit.

Status from the tape adapter determines the next step in the microprogram.

Interrupts are generated by the tape adapter when the tape unit is reading or writing tape.

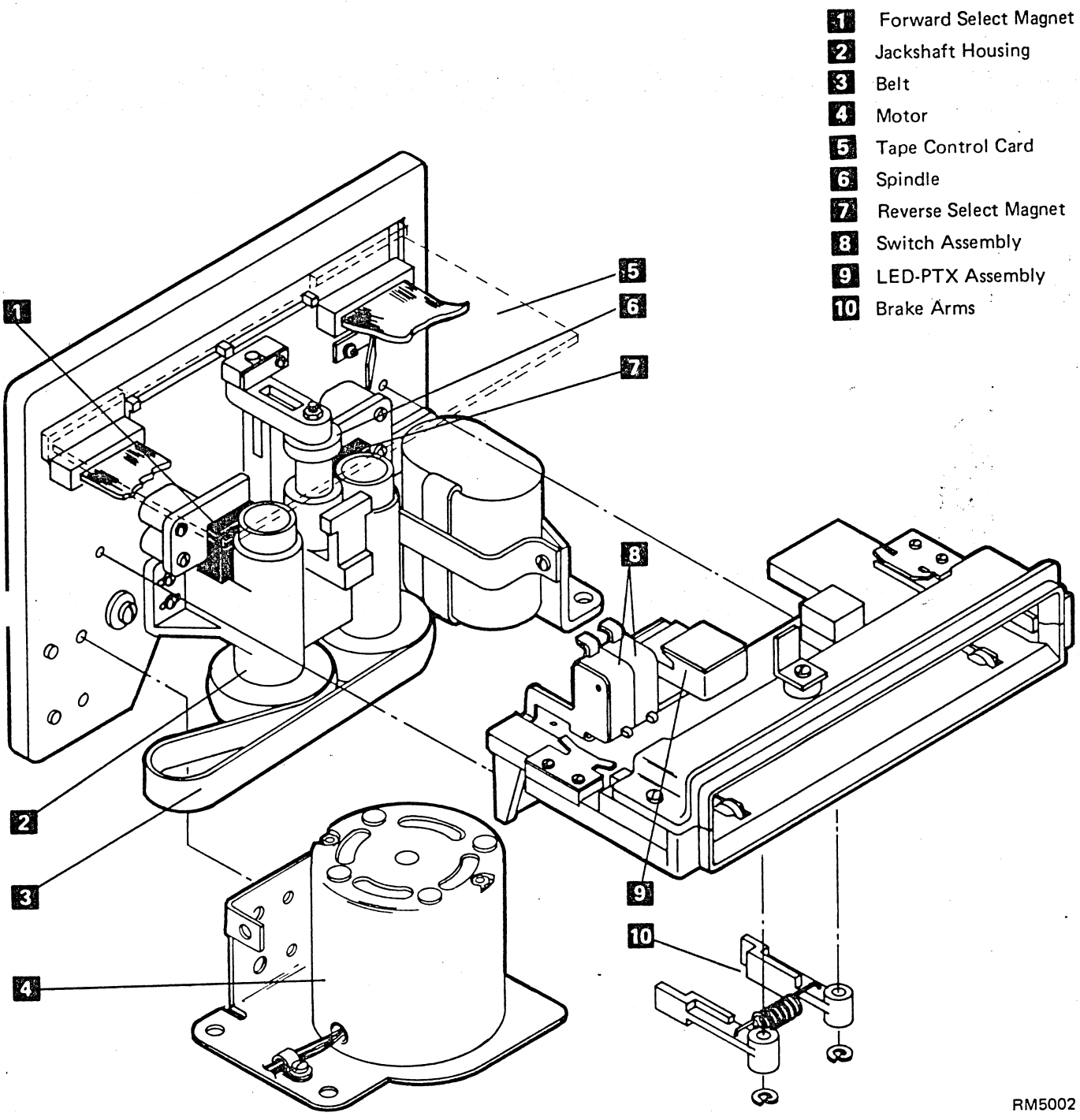
### Tape Drive Components

Tape motion is produced by a synchronous ac motor that runs when power is on. The motor, via a drive belt, rotates two rollers on the jackshaft housing in opposite directions. The spindle rotates when attracted to either one of the jackshaft rollers by a select magnet.

There are two select magnets, one for forward tape motion and another for reverse tape motion. The direction of spindle rotation is determined by the select magnet that attracts the spindle. When the select magnet releases the spindle, the brake arms stop the spindle.

Sense information is provided by a switch assembly and an LED-PTX (light emitting diode-phototransistor) assembly. The switch assembly senses the presence of a tape cartridge in the tape unit and the position of the file protect window in the tape cartridge.

As the tape moves, it travels past a mirror located inside the tape cartridge. The mirror reflects the infrared light generated by two separate LEDs back to two separate PTXs. One LED and PTX pair detects the EOT (end of tape) holes. The other pair detects the BOT (beginning of tape) holes.


1 Forward Select Magnet
2 Jackshaft Housing
3 Belt
4 Motor
5 Tape Control Card
6 Spindle
7 Reverse Select Magnet
8 Switch Assembly
9 LED-PTX Assembly
10 Brake Arms
RM5002

### Tape Operations

Tape operations of reading and writing occur when the controller is issued a mark, load, or save command via the keyboard or when using a BASIC or APL program function. These keyboard initiated operations result in tape motion associated with searching, reading, and writing a tape.

The tape unit moves tape at 40 in/sec during both the search and the read operations. Tape movement might appear faster during the search operation because tape movement is continuous. During the read operation, the tape unit might pause after each file while waiting for instructions from the controller. These pauses might give you the illusion that the tape movement is slower, but it is not.

Tape movement while writing tape is uniquely different from searching or reading. The tape unit appears to be starting and stopping frequently and makes a clicking noise. The start-stop action and clicking noise result from the tape unit writing, backspacing, and reading each record before it proceeds to the next.

When a mark command is issued, the tape unit rewinds to the beginning of an unmarked tape or searches backward to the previous header record of a marked tape. After finding the beginning of tape or the header record, the tape unit begins marking the specified number of files indicated by the mark command.

On previously marked tapes, the tape unit searches for a header record containing a FF (end of marked tape) before it begins to mark files. If the files to be marked have been previously marked, an error code (150) appears on the display.

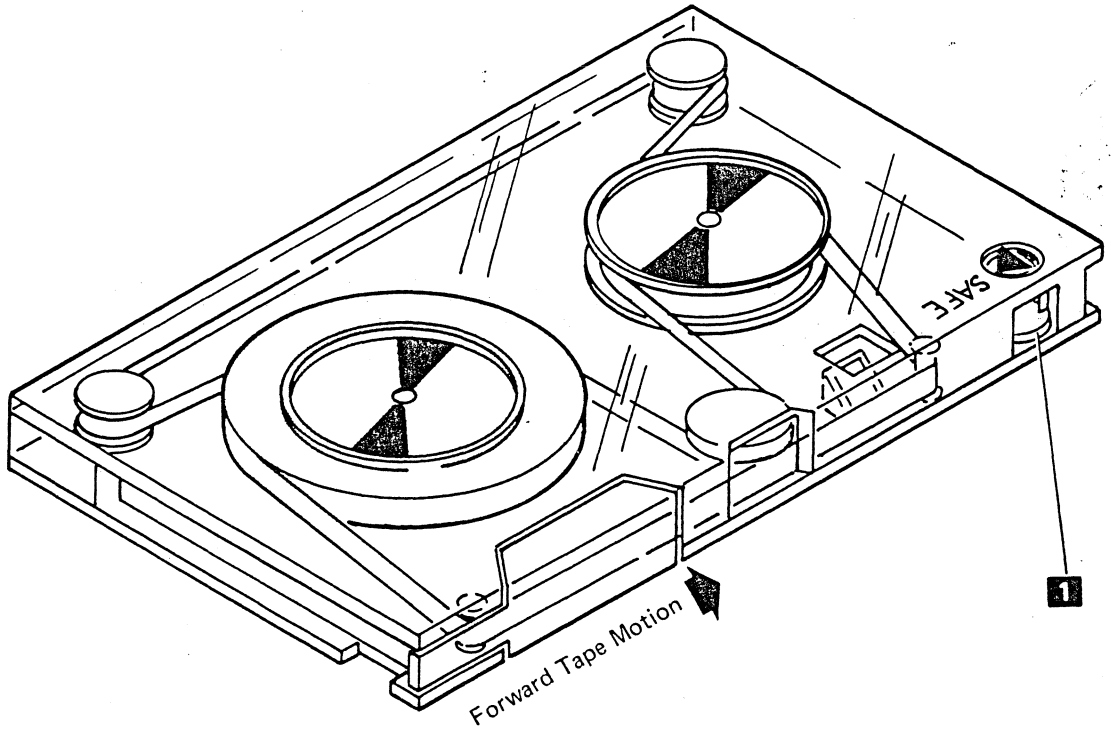
The save command instructs the controller to get data from read/write storage which then directs the tape unit to write that data on tape. The tape unit first searches for the specified file indicated by the save command, then writes the data on the tape.

The load command instructs the tape unit (via the controller) to read data from tape and puts it into read/write storage. The tape unit searches the tape for the file specified by the load command, then reads the file and puts it into read/write storage.

## ıPE CARTRIDGE

The tape cartridge used by the tape unit for storing data and programs is the IBM Data Cartridge. The data cartridge contains 300 feet of 1/4 inch (6.35 mm) tape and stores a minimum of 200K bytes of formatted data. The tape unit and tape cartridge are keyed to prevent inserting the tape cartridge incorrectly.

File protection is insured by a file protect window **1** on the tape cartridge. When this window is turned to the SAFE position, an error code is displayed if the write operation is attemped.



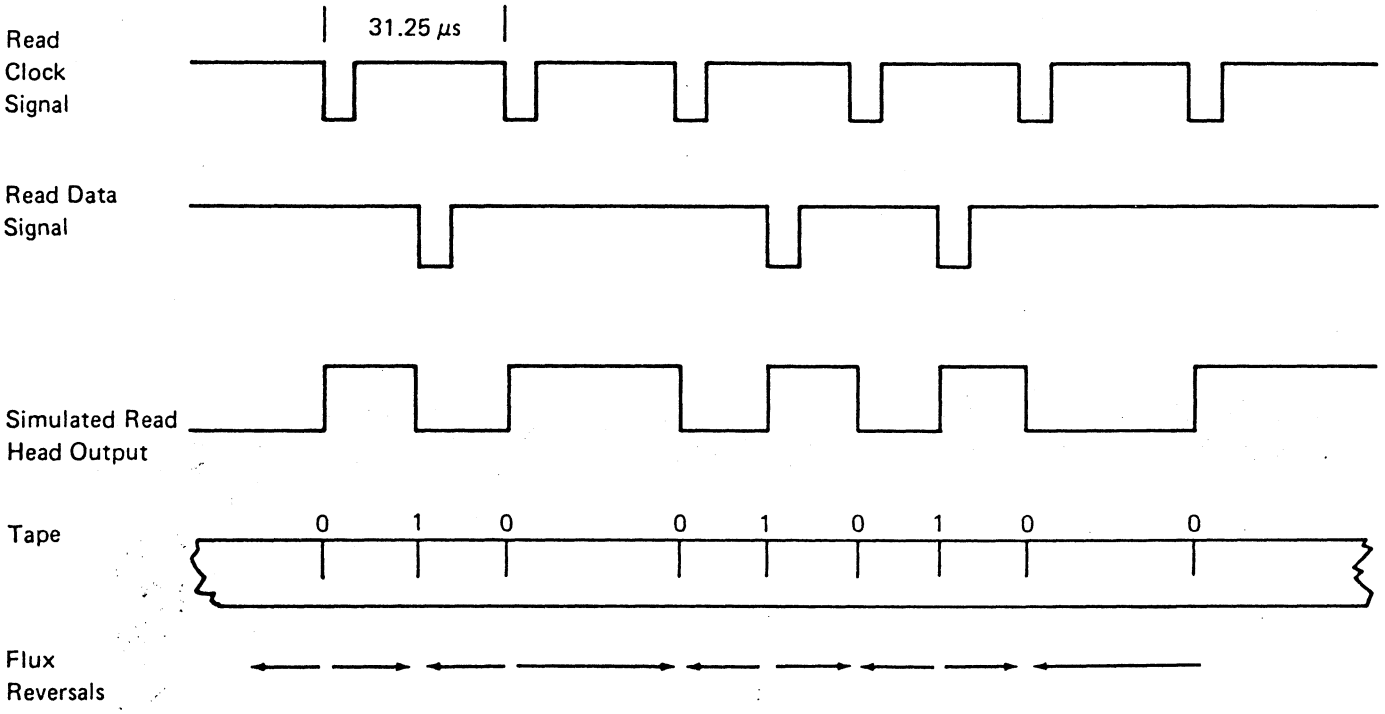
**Tape Writing and Formatting**

A bit is written on the tape when the magnetic field reverses at the write head causing a flux reversal on the tape.

Bits are written as clock pulses or data pulses. When writing, clock pulses are written every 31.25 $\mu$s. If data bits are written, they are written between clock pulses.

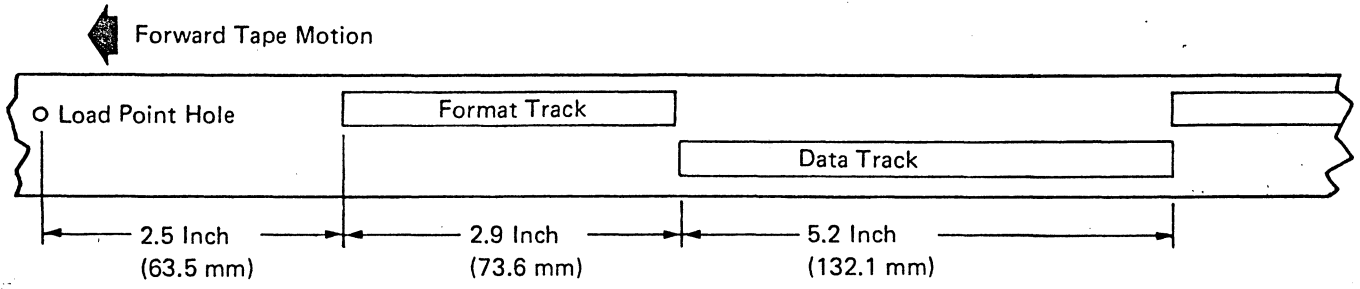
When reading data, a read clock pulse is generated every 31.25 $\mu$s from the clock bits. A read data signal is generated if there is a bit between the clock bit.

The following illustration shows the relationship of the flux reversals (bits), clock signals, and data signals:



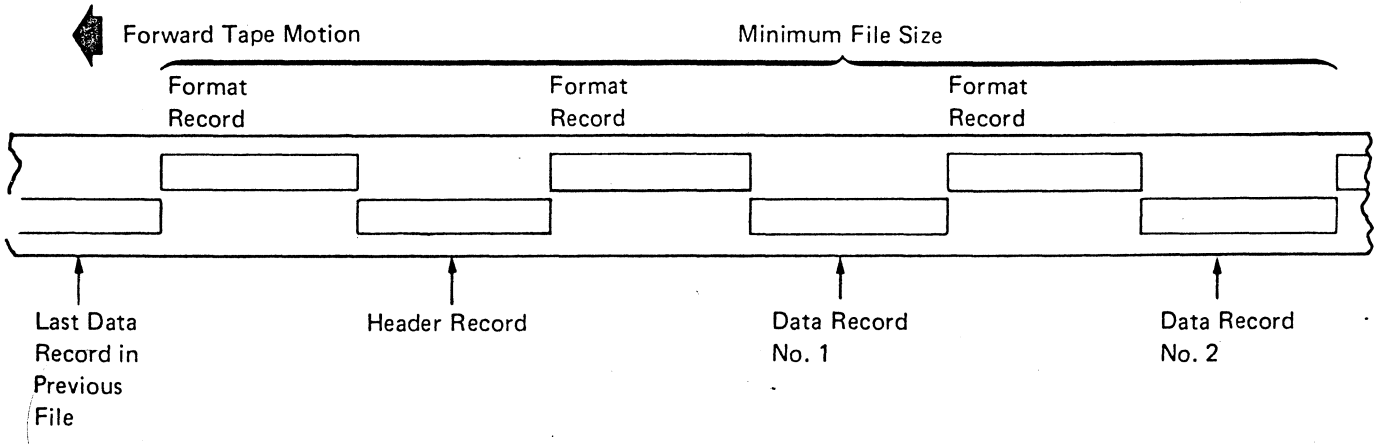
**acks**

The tape unit records data on two tracks, each occupying approximately half of the tape width. The top track (channel 0) contains format information and the bottom track (channel 1) contains the data. See the following illustration:




**Files**

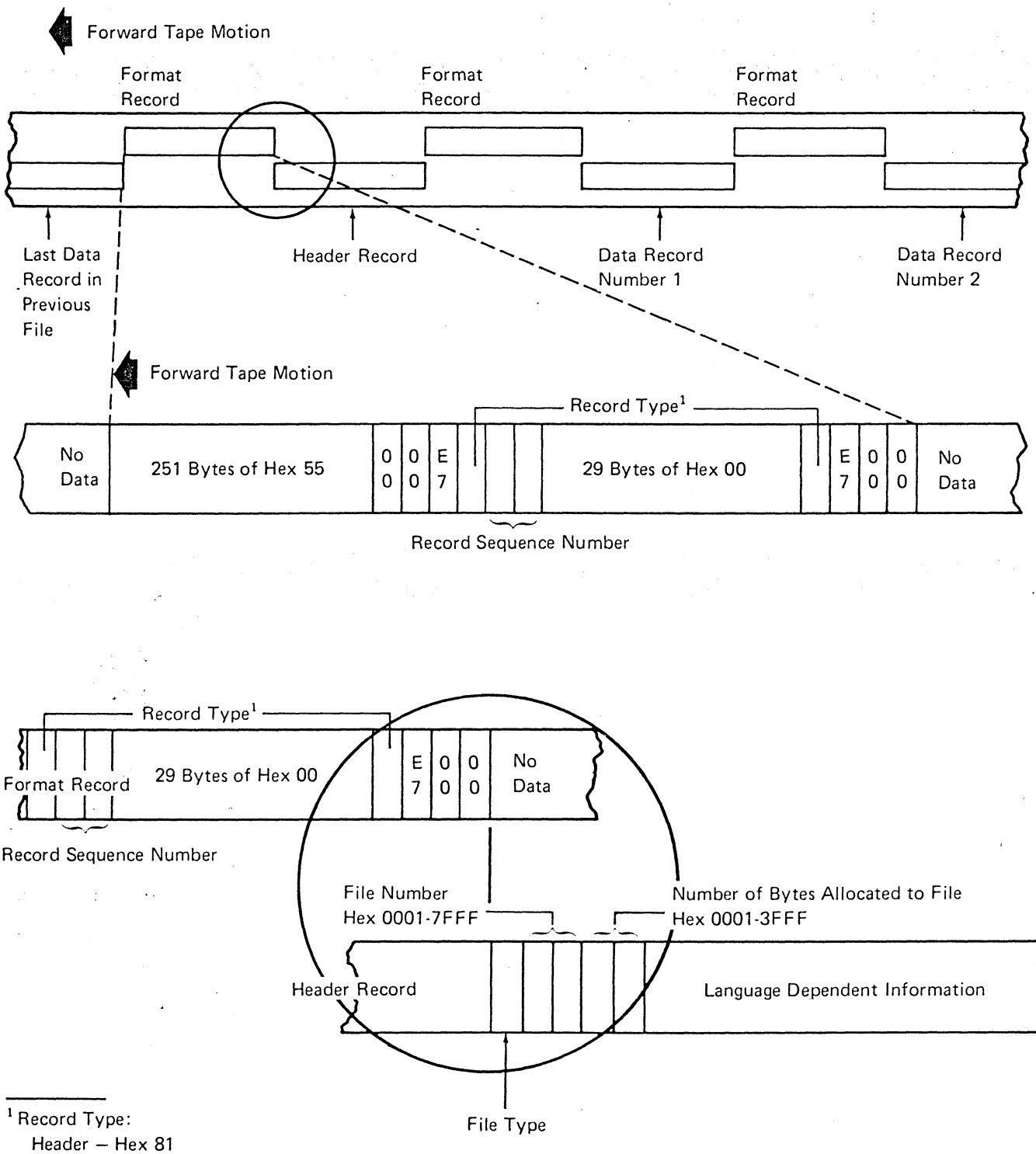
Records are grouped into files. A file consists of one header record and an even number of data records. Each header record and data record is preceded by a format record. The following illustration shows the tape file format:




Theory

*Format Record*

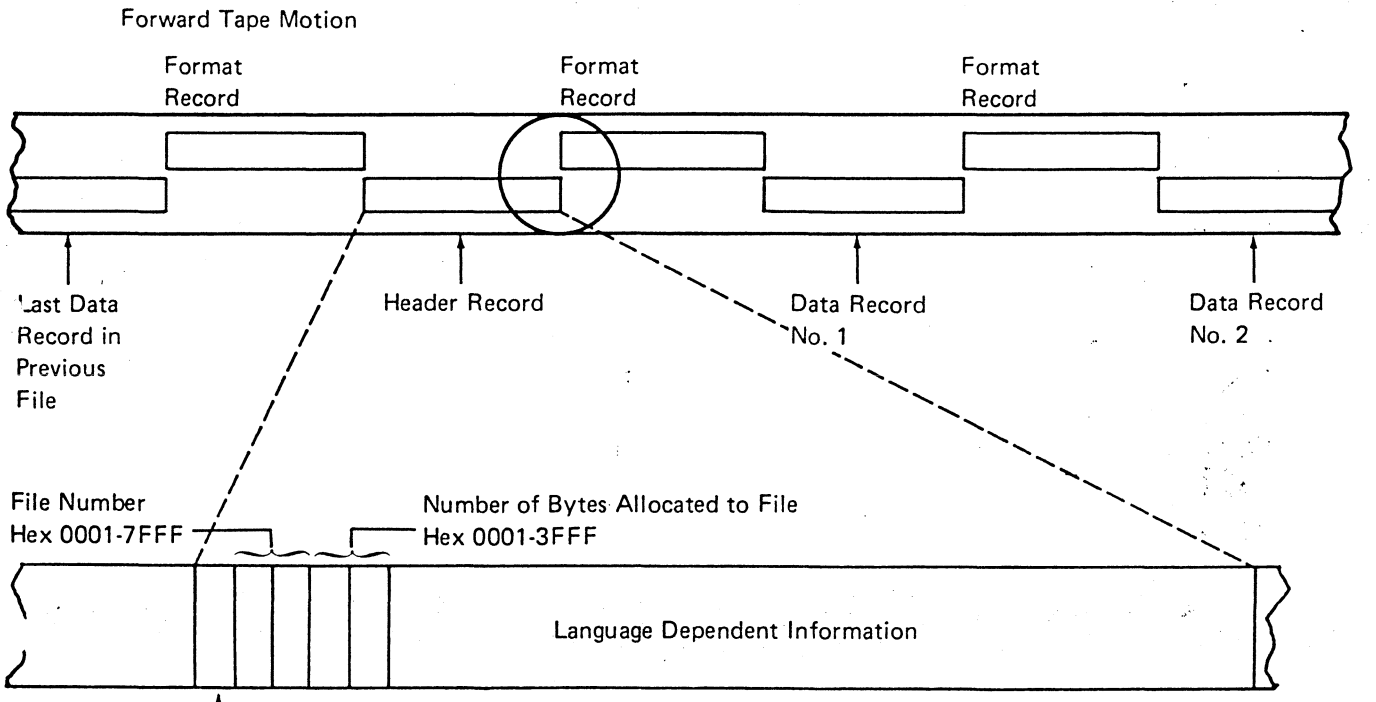
Format records are located on the format track. They provide position and timing information to the controller. Refer to the following illustration for the format record content and format:

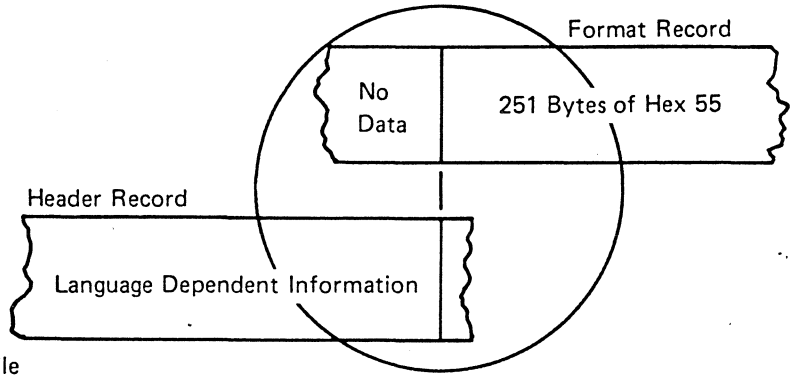



[1] Record Type:
Header – Hex 81
Data – Hex 18

*Header Record*

Header records are located on the data track. They are the first record in a file preceding the data records. The following illustration shows the contents and format of a header record.



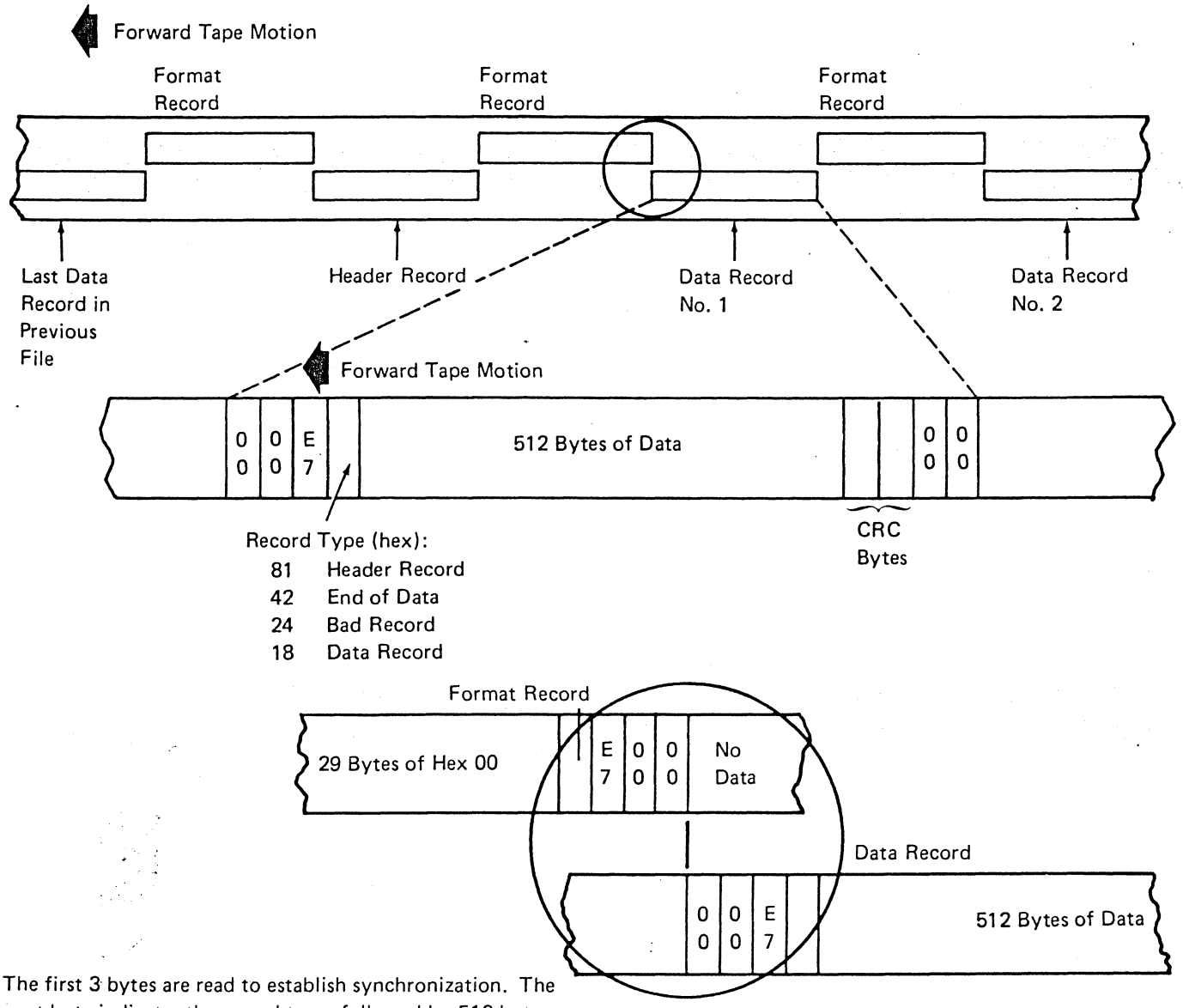




00 Marked/Unused File
01 Data File
02 General File
03 BASIC Source File
04 BASIC Workspace File
05 BASIC Keys File
06 APL Continued File
07 APL Save File
08 APL Internal Format Data File
09-15 Reserved
16 Patch
17 Diagnostic
18 Communications Adapter/Serial I/O Adapter Load File
72 Read/Write Storage to Tape Dump
FF (hex) End of Marked Tape

*ote:* The preceding decimal numbers for file type are used by UTIL or LIB commands. DCP1 uses the hex equivalents.

*Data Record*

The data record provides data and error checking information. Data is recorded in records of 512 bytes, plus sequencing and checking bytes.



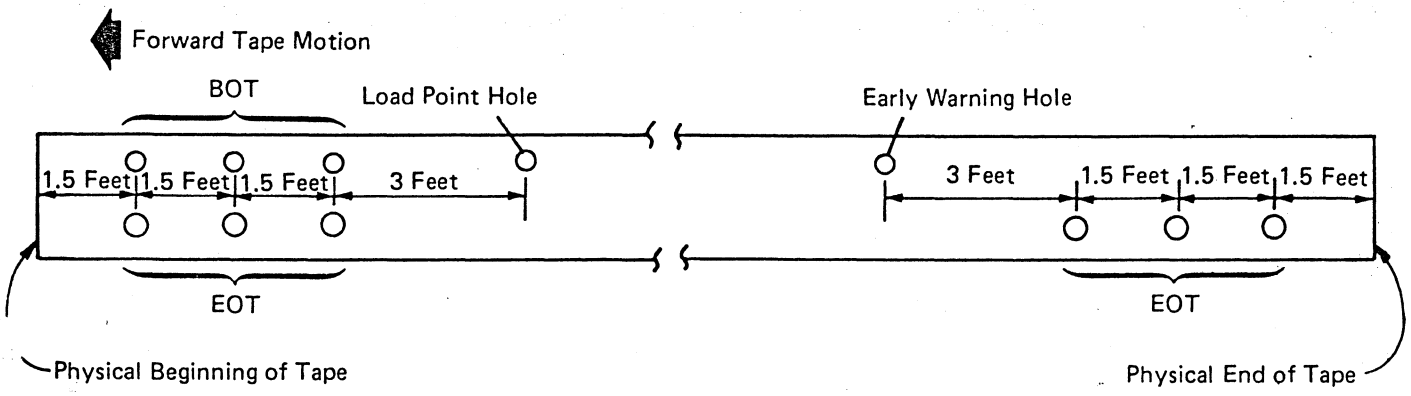

The first 3 bytes are read to establish synchronization. The next byte indicates the record type, followed by 512 bytes of data.

The CRC (cyclic redundancy check) is contained in the next two bytes. This CRC tests the 512 bytes of data and the record type byte for an error. The controller counts 513 bytes after it detects E7 and then reads the next two bytes as CRC bytes. The last two bytes of 0's in the data record generate extra byte counts in case any of the 513 bytes are missed.

A format record and a header record, which indicates the end of marked tape, follow the last data record in the last file.

## Position Markers

The following illustration shows the position of the holes (markers) on the tape. These holes generate a BOT (beginning of tape) status and an EOT (end of tape) status.




*BOT Markers*

The top five holes in the preceding figure all generate a BOT status when they are sensed. BOT status is sensed when infrared light from the LED passes through any of the top holes and is detected by a PTX.

The load point and early warning BOT holes are slightly smaller than the other three BOT holes. Therefore, in a failing condition, the PTX might detect the light coming through a large BOT hole but not detect the light coming through the smaller load point or early warning BOT holes.

The load point hole indicates the beginning of the recording area on the tape. The early warning hole indicates the end the recording area on the tape.

*EOT Markers*

The bottom six holes in the preceding figure all generate EOT status when light from the LED is sensed by the PTX. All of these holes are the same size.

*Physical Beginning and End of Tape*

The physical beginning and end of tape are determined by using both the BOT and EOT status. The beginning of tape has three groups of both BOT and EOT holes. Therefore, at the beginning of tape, both a BOT and an EOT status are sensed. The end of tape has three EOT holes only. Therefore, at the end of tape, only EOT status is sensed.

The normal customer operation of the tape unit uses only the load point hole and the early warning hole. All information is recorded between these two holes.

Sensing an EOT hole is an abnormal condition indicating that the load point or early warning hole was not found. EOT status stops the tape drive. The three EOT holes on each end of the tape ensure that EOT status is sensed and prevents the tape from running off either reel.

## TAPE MOTION

### Forward Tape Motion

Tape moves in a forward direction (counterclockwise) during mark, search, read, and write operations.

When forward tape motion stops, the momentum of the tape unit carries the tape into the 251 bytes of hex 55s in the format record. These 251 bytes provide a time delay when the tape stops; they are not read as useful information. Refer to *Format Record*.

The next three bytes (two bytes 00 and E7) provide position information. The control unit reads these bytes one bit at a time until the hex E7 byte is detected. After hex E7 is detected, the controller is synchronized (it reads 8 bits at a time rather than 1 bit at a time) at the start of the format record.

The record type byte indicates whether the data track associated with this format record has a header record or a data record written on it.

Following the record type byte are the record sequence bytes. These bytes contain the record number of the next record on the data track. A header record always has a record sequence number of hex 0000.

Another hex E7 byte is expected 33 bytes after the first hex E7 byte is read. This hex E7 byte indicates that the tape is now positioned correctly for reading or writing the data track.

When the tape is moving forward (counterclockwise), it passes the erase coils before the read/write coils. If the data track is to be written, its erase coil is activated after the first hex E7 is read from the format track and the write coil is activated after the second hex E7 is read from the format track.

The 29 hex 00 bytes (read data) in the format track provide a time delay between erasing the data track and writing on the data track. This time delay causes the data track to be erased well in advance of any data and ensures that all previously written information is erased.

The last 4 bytes in the format track establish synchronization and determine the record type when reading in the reverse direction.

### Reverse Tape Motion

Tape moves in a reverse direction (clockwise) during the search, rewind, and backspace operations.

When searching the format track in a reverse direction, the tape unit reads two bytes of 00, the E7 (sync) byte, then the record type byte.

If the record type byte contains a hex 18 (data record), the tape unit continues searching in the reverse direction until it finds a header record to determine if this is the file the microprogram is looking for.

If the record type byte contains a hex 81 (header record), the tape unit stops searching the format record and reads the header record in the forward direction. If the header record is the one the microprogram is looking for, the entire file is read in the forward direction. (The microprogram compares the file number from the tape to the file number the microprogram is searching for).

If the header record does not contain the correct file number, the tape unit resumes moving tape backward until it finds the correct file number.

## OPERATIONS

The tape unit can be used for APL, BASIC, and communications because it is language independent. Error detection and correction, and several operations such as formatting new files and finding old ones is performed by the controller.

Operations provided by the controller and tape unit are mark, search, read, write, rewind and backspace.

### Mark

The mark operation formats new files on a tape, or reformats existing ones. In a mark operation, the file header record is written and checked for CRC errors, and the format track information is written for the requested number of data records. During this operation, the format track is never checked after writing and the data track is always checked after writing.

### arch

The search operation locates specific files on the tape. This must be done before any records can be read from or written to a file.

To perform the search operation, the tape adapter sends commands to the tape unit to move the tape and select the format track (channel 0), The format track indicates when there is a header record on the data track (channel 1). (A header record contains information about the file; that is, type of file, file number, and file size.) As the tape moves, the data and clock signals are detected from the format track by the tape control card. The tape adapter generates an interrupt to the controller for each data bit (every 31.25 µs) until a sync byte (hex E7) is found.

The next byte read from the format track is the record type, which indicates whether the record is a header record or a data record.

If the record type byte indicates a data record, the controller continues to read format records until the record type byte indicates a header record.

When a header record is indicated, the controller sends instructions to the tape adapter to read the data track. The data track is read 1 bit at a time until a sync byte (hex E7) is found, then it is read 1 byte at a time.

The file number of the header record is read to determine if this is the file specified. If the file is not the correct file, instructions are issued by the controller to continue the search for the correct file.

### Read

During a read operation the controller sends instructions to the tape adapter to search for the user specified file. After the file is found, the remaining format records and data records in the file are read.

Format records contain the record sequence number. The controller verifies that the records are read in the proper sequence.

Each data record contains data and 2 CRC bytes. The CRC in the data record is compared to a CRC generated by the microprogram. If the CRCs are not equal, an error existed when the data record was read.

### Write

During a write operation, the controller sends instructions to the tape adapter to search for the user specified file (refer to *Search* under *Operations* in this section). The user must ensure that the specified file was previously marked and that the file is large enough to save the user specified information. After the file is found, the data track is written (the format track is not written except during a mark operation) with the user specified information.

The controller formats each data record correctly (refer to *Tape Encoding and Formats* in this section) and instructs the tape unit to read each data record after it is written to check for a CRC error.

The controller writes a hex 42 in the record type byte of the last data record to indicate that this record is the end of the data.

### Rewind

The tape unit rewinds when it is given a run and a (not) forward command. Reverse (clockwise) tape direction continues until the load point hole is detected, which indicates that the beginning of tape was found. Reverse tape motion then stops and forward tape motion begins to move the load point hole (that coasted past the BOT LED-PTX when going in reverse) to the forward side of the BOT LED-PTX. The forward motion moves the tape only a few inches. The tape is now positioned correctly to read the first file.

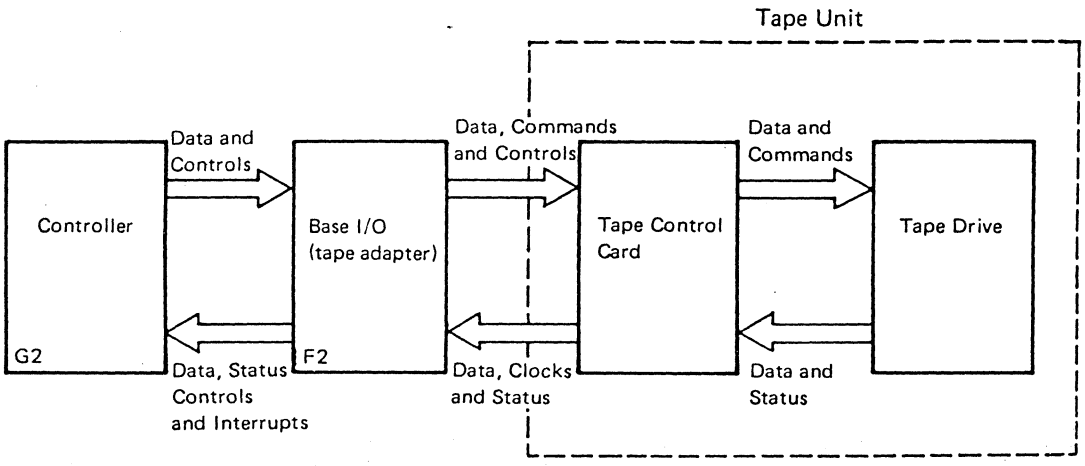
### Backspace

During a backspace operation, tape movement is in the reverse (clockwise) direction. The tape unit reads the format track backwards until a record byte indicating a header is read. Then the tape unit stops tape movement in position to read the entire file.

## TAPE ADAPTER

The tape adapter is located on the base I/O (F2) card. It receives data and controls from the controller and sends commands, control signals, and write data to the tape unit. The tape adapter also receives status, read data, and clock signals from the tape unit.

Read data and status are sent to the controller on bus in. An interrupt line (interrupt request 2) signals the controller when the tape unit is ready to accept data from the controller or ready to give data to the controller. The following illustration shows the control, data, command, status, and interrupt lines connecting the tape unit to the controller via the tape adapter.



## Tape Adapter I/O Lines

*Refer to the Tape Control Card logic diagram in Section 5.*

*Controls*

The control lines control all tape operations. The control lines and their uses are:

*'-POR Switch'* – Resets the registers in the tape adapter when the RESTART switch is pressed.

*'+Oscillator'* – Provides a 15.091 MHz signal to generate interrupts when writing data on tape. This signal is sent to the tape control card as a 3.7728 MHz tape clock signal.

*'+Tape Clock'*. Clocks the read clock counter that distinguishes between clock pulses and data pulses (flux reversals) coming from the read head. 'Tape clock' also provide timing pulses for the bit interval timer counter. The bit interval timer counter measures the time between clock pulses coming from the read head. The 'tape clock' signal is a free running square wave signal at 3.7728 MHz; symmetry must be better than 33%.

*'+Device Address X0-X3, Y0-Y3'* – Indicates the device to be used by the controller. The tape unit uses device address F for clearing all latches and registers, and device address E for all other operations. 'Device address X3 and Y3' lines must be up for a device address F. 'Device address lines X3 and Y2' must be up for device address E.

*'-Control Strobe'* – Signals the tape adapter that the information on bus out 0-7 is not a command data.

*'-Put Strobe'* – Signals the tape adapter that the information on bus out 0-7 is data, not a command.

*'+Op Code E'* – Signals the tape adapter to gate data or status onto bus in. It is also used to simulate data from the tape control card during diagnostic testing and for subdevice addressing.

*'-Get Strobe'* – Signals the tape adapter that information from its registers on bus in was sampled by the controller. The tape adapter then clears this information from the registers.

*'+Start Execute'* – Signals the tape adapter that it can sample the device address lines and bus out lines for information. During start execute time, parity is checked on bus out and the device address lines are tested for a device address check.

*'-Machine Check'* – Signals the controller that either a bus out check or a device address check occurred.

A bus out parity check occurs when bus out parity is not odd. The bus out condition can be cleared only by pressing RESTART or by turning power off.

A device address check occurs when the eight 'device address' lines do not have even parity. A correct device address for the tape unit is 'device address X3 and Y3' up (device address F) or 'device address X3 and Y2' up (device address E) (refer to the Base I/O Card logic diagrams). A device address check can be cleared only by pressing RESTART or by turning power off.



*Commands*

Tape unit commands are generated by the microprogram microinstructions located in executable ROS. The microprogram sends commands from the controller to the tape adapter. The following table indicates the bus out bits and the associated commands when used with a device address E.

| Bus Out Bit | Bit = 0 (Off) | Bit = 1 (On) |
|---|---|---|
| 0 | Run | Stop |
| 1 | Forward | Reverse |
| 2 | Channel 1 select | Channel 0 select |
| 3 | Write | Read |
| 4 | Channel 0 erase | Not channel 0 erase |
| 5 | Channel 1 erase | Not channel 1 erase |
| 6 | Diagnostic mode | Not diagnostic mode |
| 7 | Interrupt enabled | Interrupt disabled |

The command signals are:

*'-Run'* – With the 'forward' line, activates the select magnet drivers.

*'-Forward'* – Selects the direction of tape motion. When this line is down (bus out bit 1 = 0), the forward select magnet is energized. The reverse select magnet is energized when the line is up.

*'-Channel Select'* – With the 'write enable' line, controls the read/write channel multiplexer as follows:

| Bus Out Bit 2 (Channel Select) | Bus Out Bit 3 (Write Enable) | Function |
|---|---|---|
| 1 | 1 | Read channel 0 |
| 0 | 1 | Read channel 1 |
| 1 | 0 | Write channel 1 |
| 0 | 0 | Write channel 0 |

Channel 0 = Format Track
Channel 1 = Data Track

*'-Write Enable'* – Gates the write driver on the tape control card.

*'-Channel 0'* and *'-Channel 1 Erase'* – Provide independent control of the erase coils for each channel.

*'-Diagnostic Mode'* – Activates an interval timer on the tape control card for analyzing the tape unit. The interval timer measures the time between clock pulses. This time is sent to the tape adaper via the 'read data' line as an 8 bit serial byte (high order bit first). The 'read clock' line provides eight strobe times, one for each bit, to clock the byte to the tape adapter.

*'-Interrupt Enabled'* – Is used only on the tape adapter card and does not generate a command to the tape control card.

*Status*

Tape unit status is generated by the tape control card. This status is held in the status register in the tape adapter. The controller gets the status on 'bus in' with a microinstruction to the tape unit. Tape status is stored in storage location 008F. This status is retrieved after every command; therefore, 008F contains the latest tape status. Tape status is shown in the following table:

| Bus in Bit | Bit Position | Bit = 0 (Off) | Bit = 1 (On) |
|---|---|---|---|
| 0 | 8 | No end of tape | End of tape (EOT) |
| 1 | 4 | Device address E response | No device address E response |
| 2 | 2 | Tape stopped | Tape running |
| 3 | 1 | No cartridge in place | Cartridge in place |
| 4 | 8 | Erase off | Erase on |
| 5 | 4 | LED or erase defective | LED and erase OK |
| 6 | 2 | Allow write | File protected |
| 7 | 1 | Beginning of tape (BOT) | No beginning of tape |

To see the tape status byte, display read/write storage location 008F.

Signal lines from the tape control card that generate status are:

*'-End of Tape' (EOT)* — Indicates that the end of tape hole was sensed. EOT status is held in the 'EOT error stop' latch on the base I/O card. The tape cannot move until this latch is reset because this latch blocks the run command. (EOT also blocks the run command on the tape control card.)

*'-Beginning of Tape' (BOT)* — Indicates that the beginning of tape hole was sensed.

*'-Select Mag Active'* — Indicates that either the forward or reverse select magnet coil is conducting current.

*'-Cartridge in Place'* — Indicates the presence of cartridge in the tape unit.

*'-Erase Inactive'* — When down, indicates that neither channel 0 nor channel 1 erase coil is erasing. If either erase coil is erasing, this line is up.

*'-LED and Erase OK'* — Indicates that the EOT and BOT LEDs are conducting and that the erase coils do not have an open circuit. If either erase coil has an open circuit during an erase operation or either LED has an open circuit anytime, this line is up.

*'+File Protect'* — Indicates the position of the file protect window in the cartridge. This line must be down before write instructions are initiated.

*Interrupts*

The tape adapter generates interrupts to the controller on the 'interrupt request 2' line when reading information from or writing information to the tape unit. Interrupts tell the controller that the tape adapter needs it to process the tape data. The controller executes microinstructions in ROS to operate the tape unit.

When reading, the interrupt frequency depends upon whether the tape adapter is programmed for bit mode or byte mode. The mode is controlled by the microprogram with a put microinstruction and bus out bit 2 = 1 (for bit mode), or bus out bit 2 = 0 (for byte mode). In bit mode, the interrupt occurs after each bit is read (every 31.25 μs). In byte mode, the interrupt occurs after 8 bits are read (every 250 μs).

Byte mode is used after synchronization is established because this mode causes fewer interrupts to the controller. The controller operates more effectively if it has fewer interrupts to the controller. The controller operates more effectively if it has fewer interrupts to process.

When writing, the interrupts occur every 31.25 μs to signal the controller that the tape adapter sent one data bit to the tape control card and the tape adapter is now ready for another bit.

*Data and Clocks*

Data is read from the tape one bit at a time. The tape control card detects the flux reversal on the tape and generates 'read data' and 'read clock' signals. The 'read clock' signal indicates when the 'read data' signal is valid. 'Read clock' is a 265 ± 50 ns pulse.

The 'read clock' signal gates the 'read data' signal into the RDDR (read data deserializer register) in the tape adapter (refer to the Base I/O Card logic diagrams). In bit mode, an interrupt is generated after 1 bit is stored in the RDDR. In byte mode, the interrupt is generated after 8 bits are collected in the RDDR.

After the interrupt is detected, the controller gets data from the tape adapter on the bus in lines 1 bit at a time in bit mode or 1 byte at a time in byte mode.

The controller sends data to the tape adapter 1 bit at a time. Information from the tape adapter plus clock bits are written on the tape each time the 'write data' line changes.

A clock on the tape adapter synchronizes data and clock pulses.

**Error Checking**

The tape adapter checks parity on both bus out and the device address bus. If an error is sensed, the 'machine check' line halts processing and turns on the PROCESS CHECK light. To clear this error condition, press RESTART.

Parity generated by the tape adapter is put on bus in to the controller for parity checking. If the controller senses an error on bus in parity, processing halts and the PROCESS CHECK light is turned on. Press RESTART to clear the error condition.

*CRC (Cyclic Redundancy Check)*

Cyclic redundancy checking is a mathematical method of checking transmitted bits to see if all bits were received.

The tape is written with a two-byte CRC added to each header and data record. This CRC is generated by the microprogram using the record type byte and the 512 data bytes.

Read CRC Errors

When a record is read, the CRC is generated by the microprogram and compared to the CRC bytes read from the tape. If the generated CRC bytes are equal to the CRC bytes read, the next operation begins.

If the CRC bytes are not equal, the tape unit backspaces, rereads the data record, and compares the CRC bytes again. If the retry is successful (CRC bytes equal), the next operation is allowed. If a CRC error still exists after 10 retries, the tape unit stops and the controller puts an error message on the display.

Write CRC Errors

When a record is written, the tape unit erases the track to provide a clean surface to write a new record, backspaces, and reads the new record to check the CRC. This is done on the same pass since the erase coil is located ahead of the read/write coil.

If the CRC generated by the microprogram is equal to the CRC read from the tape, the next operation begins.

If the CRCs do not compare, the tape unit backspaces to the beginning of the record and attempts to write the record a maximum of ten times. After the tenth write, a hex 24 (bad record) is written in the record type byte. The tape unit then skips to the next record where it attempts the write operation again. There is no limit to the number of records that can be labeled bad and skipped.

If the tape unit is unable to write a hex 24 in the record type byte after ten attempts, the controller puts an error message (07) on the display.

*Error Reporting*

When any error condition is detected by the controller, it is reported by an error code. Many error codes (such as end-of-file) indicate a programming problem rather than a tape unit problem. Several codes, such as a status error (C and a CRC error (07), can indicate a tape unit problem (refer to *Error Codes* in Section 3).

In addition to the error codes, other information is available in read/write storage that can be examined by using the DISPLAY REGISTER switch on the control panel. The most recent status byte is located at address hex 008F.

Many programming errors are never reported because the microprogram automatically retries the operation for any CRC error. These retries are seen as additional tape motion and cause extra time for customer jobs.

## INTRODUCTION

Data entered at the keyboard is displayed on the 5100 Portable Computer display unit (5-inch screen) and is shown simultaneously on a TV monitor (if one is installed).

The display has a capacity of 1,024 characters with 64 characters per line and 16 lines per display. Each line is made up of 12 rows; 8 rows for the characters and 4 rows for the space between the characters.

The display unit displays alphabetic, numeric, and special characters, or hexadecimal characters depending on the position of the DISPLAY REGISTERS switch on the control panel. The display unit can also display characters in an expanded mode (space between characters) by displaying either the right 32 characters or left 32 characters.

The TV monitor displays the *same* information as the 5100 Portable Computer 5-inch screen. However, the TV monitor always has white characters on a black background, whereas the 5100 display has either white characters on a black background or black characters on a white background depending the position of the REVERSE DISPLAY switch.

## DISPLAY CONTROLS AND INDICATORS

The display controls allow the operator or CE to manipulate information on the display. Display controls are located on the keyboard and control panel. The cursor (flashing horizontal line) indicates the position of the next character to be entered from the keyboard. If the cursor is moved to a position already containing a character, the character in that position flashes to indicate the position of the cursor.

### Keys

*Backspace* (left arrow key) moves the cursor one character to the left.

*Forward Space* (right arrow key) moves the cursor one position to the right.

*Scroll Up* (up arrow key) moves each line on the display up one line. See the note in the scroll down description.

*Scroll Down* (down arrow key) moves each line on the display down one line.

*Note:* If the machine is processing in BASIC, only the top 15 lines move; the bottom line is the status line and does not move.


*Insert* (combination of two keys, the CMD key and the forward space key) inserts a blank into the cursor position you select and shifts all the characters to the right of the cursor one position to the right. This creates a space for inserting a character as shown:

```
                    ┌──── flashing
1 2 3 4 6 7

1 2 3 4 _ 6 7
          └──────── flashing

1 2 3 4 5 6 7
            └────── flashing
```


*Delete* (combination of two keys; the CMD key and the backspace key) deletes the character above the cursor and moves all characters on the right of the cursor one position to the left.

```
                ┌──── flashing
1 2 3 4 4 5 6

1 2 3 4 5 6
          └──────── flashing
```

### Switches

*L32-64-R32 (three-position switch):* When in the L32 position, the left half of the screen is expanded with blanks between characters and displays 32 characters per line. The 64 position is the normal display position and displays 64 characters per line. The R32 position displays the right half of the screen with blanks between characters and displays 32 characters per line.

*POWER ON-OFF:* This switch applies power to the 5100 Portable Computer when the POWER switch is in the ON position. The 5100 becomes ready in about 10 seconds and the display in about 25 seconds.

If the switch is in the OFF position, power is removed from the 5100. Wait five seconds before turning power on if it were just turned off.

*REVERSE DISPLAY:* This switch changes the display screen to show white characters on a dark background when the top of the switch is pressed and dark characters on a white background when the bottom of the switch is pressed. The position of the switch determines the polarity of the video signal that modulates the intensity of the electron beam. The BRIGHTNESS control usually needs adjusting after changing the REVERSE DISPLAY switch setting.

*BASIC-APL:* This switch exists only on dual-language machines. The switch selects which language will be in operation when power is turned on. To change languages with machine power on, press RESTART after changing the switch setting.

*DISPLAY REGISTERS:* This switch has two positions; DISPLAY REGISTERS and NORMAL. With this switch in DISPLAY REGISTERS position, the first 512 bytes of storage are displayed in hexadecimal code on the display screen (refer to 248). Otherwise, the normal display of APL, BASIC, or bring up program data in read/write storage locations hex 0200 through hex 05FF is displayed. Refer to Section 3 for bring up program information; refer to Section 6 for language information.

*RESTART:* This switch sends the 'POR reset' signal to the controller, which resets all logic circuits and starts the bring up program. Refer to Section 3 for details. This switch is also used as a lamp test switch for the PROCESS CHECK and IN PROCESS lights.

*BRIGHTNESS:* This control selects the intensity of the CRT electron beam and the brightness of the characters or the background of the display depending on the setting of the REVERSE DISPLAY switch.

### Lights

*PROCESS CHECK:* This light indicates a machine malfunction. Further operation cannot be attempted. Error conditions present at this time can be used as a guide to repair the 5100 Portable Computer. Refer to Section 3 for more information about process checks.

*IN PROCESS:* This light indicates that the controller is processing data and that the cycle steal controls are disabled. Therefore, when this light is on, no information is displayed on the display.

## DISPLAY ADAPTER

The display adapter (J2) card receives data from the 'read data bus' via the cycle steal controls. Signals generated by the display adapter are sent to the display unit and TV monitor to form characters. The adapter supplies three signals to the display assembly ('-machine video', '+external horizontal drive', and '-external vertical sync') and one signal to the TV monitor jack ('+monitor video').

### Adapter I/O Lines

*Display Unit I/O Lines*

The '-machine video' line consists of data video pulses and horizontal and vertical blanking pulses. '-Machine video' goes through the BRIGHTNESS potentiometer to the display unit connector, through the display PC board, and then to the cathode of the CRT. It is possible for '-machine video' to be pulsing and not have characters on the display. This occurs when the blanking pulses are present but the data video pulses are missing.

Missing blanking pulses result in visible white retrace lines, along with any video (characters), being displayed. The blanking pulses on the '-machine video' line blank the beam so that the white retrace lines are not visible.

When displaying black characters on a white background (refer to *Reverse Display* in this section), the video signal is sent to the cathode of the CRT and blanks the beam where the character appears on the display. Therefore, if no video signal is sent to the display unit, the display appears white.

When displaying white characters on a black background, the beam is blanked everywhere except where a character appears, and if no video signal is present, the display is completely dark.

The '-external vertical sync' line goes directly to the display PC board to keep the video signal synchronized with the vertical and horizontal signals. If the '-external vertical sync' signal is missing, the video information rolls vertically.

The '+external horizontal drive' line controls the beam as it sweeps horizontally across the display. If this signal is missing, the display is black (no raster).

*Read/Write Storage I/O Lines*

The display card I/O lines to read/write storage consists of the storage read bus (input) and the storage address bus (output). Refer to the Control Unit Data Flow diagram in this section and to the Display Card logic diagram in Section 5.

The storage address bus on the display card is sent to a base address (CRT buffer address) determined by the position of the DISPLAY REGISTERS switch (DISPLAY REGISTERS or NORMAL).

The base address in the DISPLAY REGISTERS position is 0000 and displays the contents of addresses 0000-01FE. (Refer to 248 in Section 2 and *Control Unit* in this section for the contents of these addresses.)

The base address for NORMAL is 0200 and displays the contents of addresses 0200-05FF in read/write storage.

If the L32-64-R32 switch is set to R32, the base addresses are 0020 and 0220 respectively because the leftmost characters are not displayed.

After the base address is set, the character counter updates the addresses by two (the low order bit is always logical 0) every other character count (CC1 time). A halfword of data is transferred from the storage read bus to the display card for each address from the display card.

If the 5100 Portable Computer display is set to L32 or R32, the character counter updates the addresses every fourth character count. These address lines address read/write storage to gate the information from the storage positions onto the storage read bus and into the display data register.

The data in the display data register is decoded to select the correct character dot pattern residing in the 1,024 x 16 ROS on the display card. These patterns are serialized by the 10-bit shift register and superimposed on the '-machine video' line that goes to the display unit via the brightness potentiometer. The output of the 10-bit shift register also is superimposed on the '+monitor video' line that goes to the TV monitor.

*Cycle Steal Control Lines*

The controller and the display adapter both access read/write storage via the storage read bus. They time share the storage read bus and the storage access cycles. Time sharing is controlled by the cycle steal controls.

The cycle steal control lines on the display card are '-display request', '-stolen cycle next', '-stolen cycle', and '-I/O display off'.

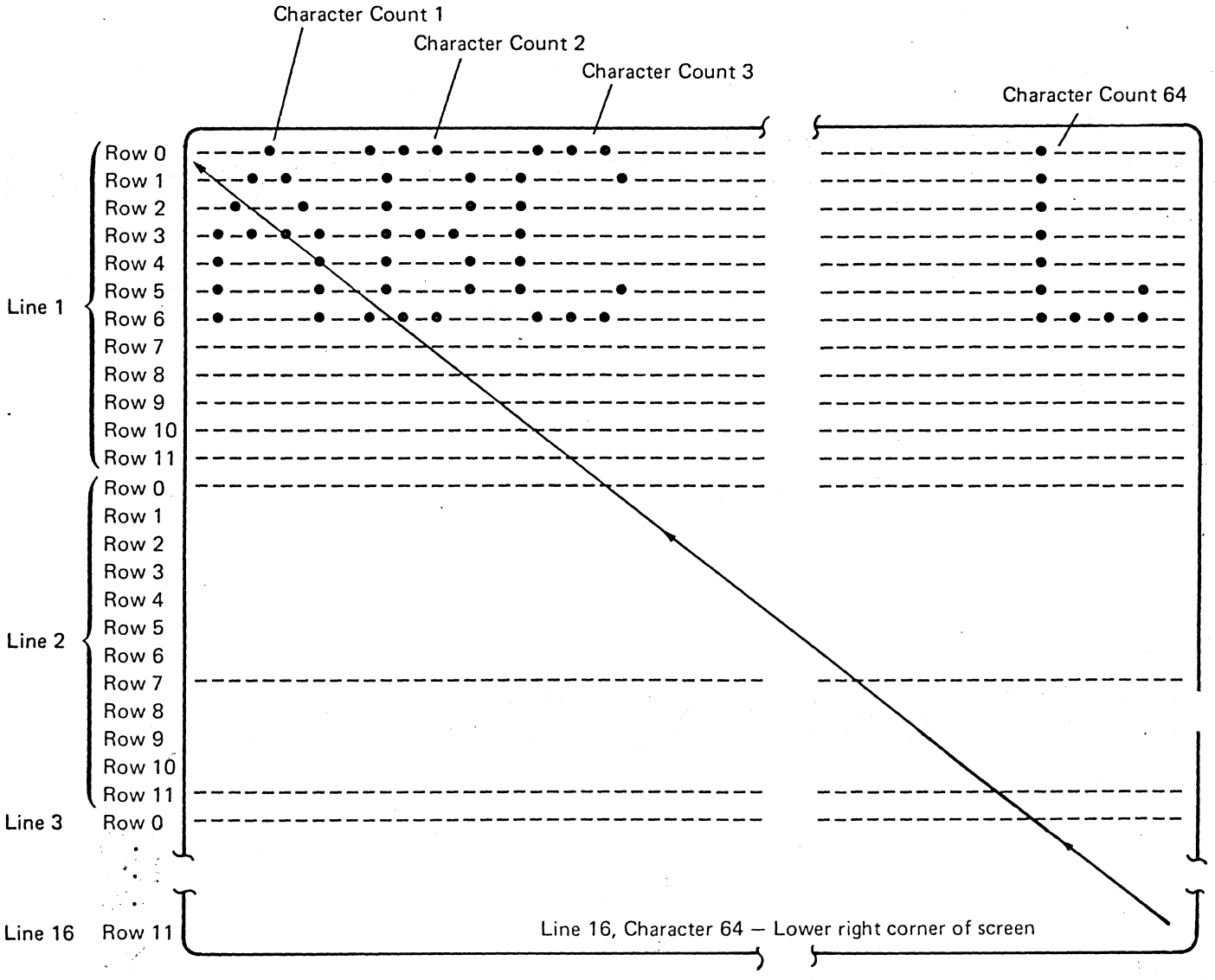
'-Display request' is activated by the display adapter when it is ready to receive the next halfword of data into the display data register. '-Display request' is a request for a stolen storage cycle.

The controller activates '-stolen cycle next' on the storage cycle preceding the stolen storage cycle. On the display card this signal deactivates '-display request' so that consecutive storage cycles are not stolen.

During the stolen storage cycle, the controller activates '-stolen cycle' and a halfword of data addressed by the storage address bus is put onto the storage read bus. On the display card, '-stolen cycle' gates the data from the storage read bus into the display data register.

Microinstructions are processed faster when no storage cycles are stolen. The '-I/O display off' line prevents storage cycle steals by the display. It is set or reset by a microinstruction. The display is blank and the IN PROCESS light is on when cycle steals are disabled.

Display Screen Example: Character, Row, and Line Counters




*Note:* Six blank rows (not shown) are above row 0 of line 1 to center the display. The line counter is 0 on line 1, 1 on line 2, . . ., and 15 on line 16.

, display counters control the timing in the display adapter, increment the storage address lines, and shift out the serialized video pulses. Three counters (the character counter, the row counter, and the line counter) control the video as the beam sweeps across and down the display. Refer to the preceding display screen example and to the display card logic in Section 5.

The video pattern is stored in the 1,024 x 16 display ROS as 8-bit horizontal rows of the character to be displayed. The rows are read into the 10-bit shift register (along with two spacer bits) each time the character counter is advanced. The address of each 8-bit horizontal row is derived from the data in the display data register that selects the character pattern, and from the row counter that selects the horizontal row of the character pattern.

When the beam begins at the upper left corner of the display screen, the character, line, and row counters are all set to zero. The base address is put on the storage address bus. (The base address if 0000 for DISPLAY REGISTERS and 0200 for NORMAL if the L32-64-R32 switch is set to L32 or to 64. The base address is 0020 for DISPLAY REGISTERS and 0220 for NORMAL if the L32-64-R32 switch is to R32.) The storage address bus contains information from the character counter, the line counter, the DISPLAY REGISTERS switch, and the L32-64-R32 switch.

The first halfword of data is read from read/write storage into the display data register. The even byte of the halfword is used to read the first 8-bit horizontal row from the 1,024 x 16 display ROS into the 10-bit shift register. This is the top row (row 0) of the first character because the row counter is 0. These 10 bits are shifted out serially as video pulses and the character counter is advanced.

The odd byte of the halfword in the display data register is used to address the next character pattern. The top row of this pattern is put into the 10-bit shift register and shifted out. The character counter is advanced again and because it now contains an even number, the next halfword in read/write storage is read into the display data register. This process continues until the top row of all 64 characters is accessed. Then the character counter returns to 0, the address bus returns to the base address, and the row counter becomes 1. The entire process is repeated for the second row of the first line of characters and for each row of the line.

After the row counter reaches 12, it is reset to 0 and the line counter is advanced. This adds 64 to the base address so that the next 64 characters in read/write storage are accessed.

The preceding process continues for each line. One is added to the row counter each time the character counter reaches 64, and one is added to the line counter each time the row counter reaches 12. When the line counter reaches 16, the row counter 12, and the character counter 64, the frame is done. The counters continue advancing to maintain synchronization while the beam retraces from the lower right to the upper left of the display screen. Then the counters are reset to zero and the next frame begins.

The previous discussion applies when the DISPLAY REGISTERS switch is set to NORMAL and the L32-64-R32 switch is set to 64 although the differences for other settings is slight. When the DISPLAY REGISTERS switch is set to DISPLAY REGISTERS, hexadecimal characters are displayed with the first hexadecimal digit of the byte on line 1 and the second digit on line 2. Therefore, the base address is advanced only on even lines.

The first 4 bits of each byte in the display data register are used to address the character pattern on odd numbered lines (line counter even) and the second 4 bits of each byte are used to address the character pattern on the even numbered lines.

When the L32-64-R32 switch is set to L32 or R32, the only difference is that a blank character is inserted between the characters so that only the left 32 or the right 32 characters are displayed. In this case, the sequential halfwords are read from read/write storage on every fourth count of the character counter rather than on every second count.

## TV MONITOR

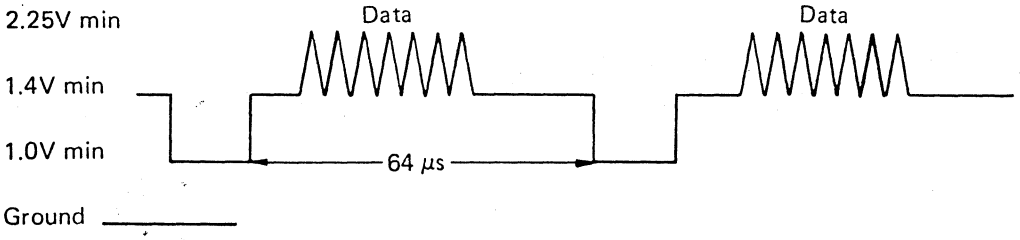
A TV monitor (separate display unit) can attach to the 5100 Portable Computer via a connector on the back of the 5100. Information presented on the 5100 5-inch display screen is shown simultaneously on the TV monitor.

The TV monitor, when connected, should provide a 75 ohm termination load to the 5100 source ground. When connecting several monitors, they should be connected in parallel (daisy-chained) and the last monitor in the string should be terminated with 75 ohms. In some cases, the 5100 might overdrive the TV monitor. Therefore, the customer might have to add a 75 ohm attenuator to the rear of the 5100.

A composite video/sync signal ('+monitor video') is sent to the TV monitor via a coaxial cable connected to the 5100.

The TV monitor has its own ac power source and develops its own dc voltages.

The following illustration shows the '+monitor video' signal as it appears on an oscilloscope with the 5100 power on and the DISPLAY REGISTERS switch at DISPLAY, REGISTERS.




*Scope Set-up:*

| | |
|---|---|
| Sync trigger | int (-) dc |
| Sweep | Auto |
| Sweep speed | 10 μs |
| Vertical amp | 0.1 V/div |
| Probe point (Use a grounded 10X probe). | A1-K6B04 (+monitor video) |

## yboard

### INTRODUCTION

The keyboard provides a means for manually entering data into the 5100 Portable Computer. The keyboard is an I/O device with device address 4 (DA = 4), and interrupts the controller with the highest priority program level (level 3). Refer to *Control Unit* in Section 4 for more information about how I/O devices interface with the controller.

The base I/O (F2) card contains the adapter between the controller and the keyboard assembly. The keyboard assembly consists of the all keys assembly, the keyboard, the PC board, and the keyboard cable.

### OPERATION

The controller continually samples each of the 74 key positions (the space bar has two positions but only one is used). The scan counter on the keyboard PC board directs the sample pulse to each key, one at a time. Sampling one key position takes about 140 ns and sampling all 74 positions takes about 10 ms.

When a key is pressed, a flyplate in the key module moves away from printed circuit pads on the keyboard PC board and decreases the capacitance at that key position. (A pressed key is sensed when a sample pulse is not transmitted. That is, the absence of the sample pulse indicates that a key is pressed.) Releasing the key restores the flyplate to its normal position.

When a pressed key is detected, its code (decoded from the number in the scan counter and other data such as a shift or command key being pressed) is placed into a character register on the keyboard PC board. The data, which includes an odd parity check bit, remains in the character register until the key is released. A single '-keyboard strobe' signals the keyboard adapter on the base I/O card that the keyboard has data ready.

If the '+typamatic' line from the keyboard adapter to the keyboard PC board is up, '-keyboard strobe' is repeated every 100 ms after an original 700 ms delay for as long as the key remains pressed. The data is then sent to the adapter every 100 ms as long as the key is pressed.

Three keys, the command key and both shift keys, do not generate a '-keyboard strobe' but set latches instead. These latches modify the key code of other pressed keys. The key code for each key is given in reference 250 of Section 2.

The keyboard lock function is controlled by disabling keyboard interrupts with microinstructions. The '-keyboard lockout' signal to the keyboard PC board is not used. (Therefore, the '-keyboard lockout' signal should always be up.)

The '-POR' line initializes all of the keyboard logic circuits. It must be present for at least 100 ms after power is turned on or after RESTART is pressed.

### KEYBOARD ADAPTER

The keyboard adapter generates the interrupts for the keyboard. Control microinstructions from the controller enable or disable interrupts, reset interrupts, and set the repeat action function.

Keyboard interrupts are not generated if they are disabled by a control microinstruction. Keyboard interrupts are disabled during part of the bring up program.

When keyboard interrupts are enabled, each '-keyboard strobe' causes the keyboard adapter to generate a program level 3 interrupt. The controller immediately begins operating at program level 3 because 3 is the highest priority interrupt.

The level 3 microprogram gets the data from the keyboard adapter with a get instruction. The 'op code E' strobe signals the keyboard adapter to put the key code on bus in. The 'get strobe' signals the adapter that the data was taken by the controller.

The controller resets the interrupt when it finishes processing the data.

#### Keyboard Lock Function

The level 3 microprogram tests the received keyboard key code for HOLD or ATTN. If either HOLD or ATTN was pressed, the microprogram sets a flag so that the corresponding function is performed when the controller returns to level 0. For all the other key codes, the microprogram determines if the keyboard is locked (indicated by a flag in read/write storage) and if it is locked, the interrupt is reset and the received data is ignored.

### Keyboard Code Translation

The controller converts key codes into the 5100 internal code using a translation table. (The translation table is located in nonexecutable ROS (E2). When a key is pressed, the controller does a table look-up to obtain the translated character.) The translated character is stored in a register in read/write storage. The interrupt generated when the key was pressed is now reset and the character is processed by the level 0 microprogram.

### Repeat Action Function

The microprogram examines the key code (received from the keyboard and translated from ROS) to determine whether it matches one of the codes of the repeat action keys. If the key code is that of a repeat action key, the microprogram sets 'typamatic' to the keyboard assembly by issuing a CTL microinstruction to the keyboard adapter. The interrupt generated when the key was pressed is now reset and the internal code is processed by the microprogram in level 0.

The space,  ,  ,  , and  keys are the repeat action keys on the 5100 Portable Computer keyboard.

### Error Checking

The keyboard adapter checks for odd parity on the bus out lines and for even parity on the device address lines. Either error causes a PROCESS CHECK, which stops processing. For more information, including probe points, see *Error Checking* under *Control Unit* in Section 4.

The bring up program provides a convenient test for stuck keys or stuck flyplates of keys.

When any key, except CMD, ATTN, HOLD, and either shift keys, is stuck, the bring up program stops with ABCDEFG displayed on line 1. Line 3 contains a 3 in position 1, and the key code (refer to 250 in Section 2) in positions 3 and 4 of line 3. Line 5 contains keyboard status information.

A stuck key can be simulated by holding down a key and pressing RESTART. The key code can then be compared with reference 250 of Section 2.

## AC POWER BOX

The 5100 Portable Computer operates with any of five single-phase, ac power sources (refer to Appendix B). Ac power enters the 5100 via a line cord that attaches to the line filter in the ac power box. The ac power box also contains ac capacitors and a fuse holder. A power ON/OFF switch controls distribution of ac power to the cooling fan, the tape drive motor, and the dc power supply (refer to 207 in Section 2). The ac power box components and their functions are:

- Line Cord 6 : Supplies ac line input voltage to the ac power box.
- Line Filter (L1) 1 : Filters line noise to frame ground.
- AC Capacitors 2 , 3 (old style): Filters line noise from line to line.
- Fuse Holder (F1) 5 : Holds the fuse that protects the internal ac wiring and components.

## POWER SUPPLY PC BOARD

The power supply is a high-frequency TSR (transistor switching regulator) supply that develops five dc voltages:

- +12 Vdc: Display, tape select magnets, tape LEDs, and expansion feature card

  +8.5 Vdc: Tape amplifier card, storage card, keyboard, display adapter, printer adapter, and all ROS
- +5 Vdc: Basic logic voltage (TTL)
- -5 Vdc: Tape amplifier card, storage cards, and non-executable ROS
- -12 Vdc: Tape unit, bias voltage, and expansion feature card

These output voltages can vary from +10% to -9% before they affect the function of the machine.

The power supply, which includes the ac input and dc output cables, is an FRU and requires no adjustments. If the supply fails, replace the entire unit.

## POWER SUPPLY PROTECTION

The power supply has built in overvoltage, undervoltage, and overcurrent protection. Overvoltage protection shuts down the power supply whenever the +12 Vdc output voltage exceeds its nominal value by 33%. Undervoltage shutdown occurs whenever the -5 Vdc level raises above -3 Vdc. Overcurrent protection monitors the current at the primary winding of the transformer. When the current is excessive, caused by overloaded or short circuited output voltages, the power supply shuts down. Any time the power supply shuts down, the machine must be powered down for four seconds before it can be powered up again.

## REFERENCE VOLTAGE

A closely controlled reference voltage of +6 Vdc is provided to calibrate the CE meter. This reference voltage is developed on the display (J2) card by a power supply output to a zener diode. The voltage across this diode is used as a reference for all critical voltage checks. The reference voltage is on pins J2-S02 (+6V) and J2-D08 (gnd).

## POWER FREQUENCY

The peak-to-peak ripple voltage at 20 kHz to 25 kHz, as measured with an oscilloscope should not exceed 3% of the nominal value of the dc voltage for each of the dc voltages.

## Features

### IBM 5103 PRINTER

The IBM 5103 Printer is a bidirectional wire matrix printer. For complete details on the 5103 Printer, refer to the *IBM 5103 Printer Maintenance Information Manual,* SY31-0414.

A list of the 5103 Printer capabilities follows:

- 128 different characters
- 132 characters per print line
- 10 characters per inch horizontal spacing
- 6 lines per inch vertical spacing
- 15 inch pressure feed platen
- Forms tractor for printing continuous forms

### IBM 5106 AUXILIARY TAPE UNIT

The IBM 5106 Auxiliary Tape Unit is a stand alone tape unit that attaches to the IBM 5100 Portable Computer to provide additional flexibility and increased data storage. An I/O cable assembly provides all dc power and input/output lines from the 5100.

The tape unit, tape control card, and ac motor used in the 5106 Auxiliary Tape Unit are identical to those in the tape unit in the 5100 Portable Computer. The 5106 Auxiliary Tape Unit contains its own ac power, power switch, POWER ON indicator, cooling fan, adapter board and card, and an external I/O interface port.

The adapter card contains the write clock generation logic, the read data deserialization register and counter, the latches and gates for connecting to the I/O cable driver (A2) card in the 5100, the interrupt logic, and subdevice address decode. These circuits are similar to those on the base I/O (F2) card. The 5106 Auxiliary Tape Unit also uses the same IBM Data Cartridge and data format as the tape unit in the 5100 Portable Computer.

The tape operations of the 5106 Auxiliary Tape Unit are identical to those of the tape unit in the 5100 Portable Computer. The controlling microprograms, command definition, and status definition are also identical. Refer to *Tape Unit* in this section for a complete description of tape operations. The subdevice address is the only pointer that determines if a command is directed to the 5106 Auxiliary Tape Unit or to the tape unit in the 5100.

#### Error Codes

The 5106 Auxiliary Tape Unit error codes are identical to those of the tape unit in the 5100 Portable Computer. Refer to *Error Codes* in Section 3 for information on these codes.

#### Error Detection and FRU Isolation

Diagnostic programs and MAPs for error detection and FRU isolation are the same as those used by the tape unit in the 5100 Portable Computer. The MAPs reside in the diagnostic tape cartridge under DCP2 MDI control. The 5106 Auxiliary Tape MDI is 840 and the tape write MDI (same as tape unit in 5100) is 860 on the DCP2 menu. Before beginning to diagnose 5106 errors, be sure that the tape unit in the 5100 is functioning correctly. This is important because the MDI for the 5106 is loaded into read/write storage from the diagnostic tape cartridge using the tape unit in the 5100.

### EXPANSION FEATURE

The Expansion feature is a prerequisite for the Communications Adapter feature and/or the Serial I/O Adapter feature. This feature consists of one logic card in location B2.

### COMMUNICATIONS ADAPTER

The Communications Adapter feature allows the 5100 Portable Computer to communicate with a remote system. The 5100 appears to the remote system to be a 2741 Communications Terminal. During communications operations, the 5100 is a terminal and cannot use its APL or BASIC languages. The 5100 transmits data from the keyboard or tape unit, writes data on tape, or prints data on the printer.

The Communications Adapter feature consists of the following:

- Data set cabling
- Communications microprogram

Refer to the *IBM 5100 Communications/Serial I/O Maintenance Information Manual,* SY31-0429, for additional information.

## SERIAL I/O ADAPTER

The Serial I/O Adapter feature allows the attachment of I/O devices to the 5100 Portable Computer. Some devices that can be attached are:

- Plotters and displays
- Card readers/punches
- Printers
- Instrumentation devices

These devices attach to the 5100 via an EIA RS232C interface that uses many common bit rates.

Because the Serial I/O Adapter feature is operated directly from the APL or BASIC languages, the Communications Adapter feature, if installed, cannot be used while the Serial I/O Adapter feature is being used.

The Serial I/O Adapter feature consists of the following:

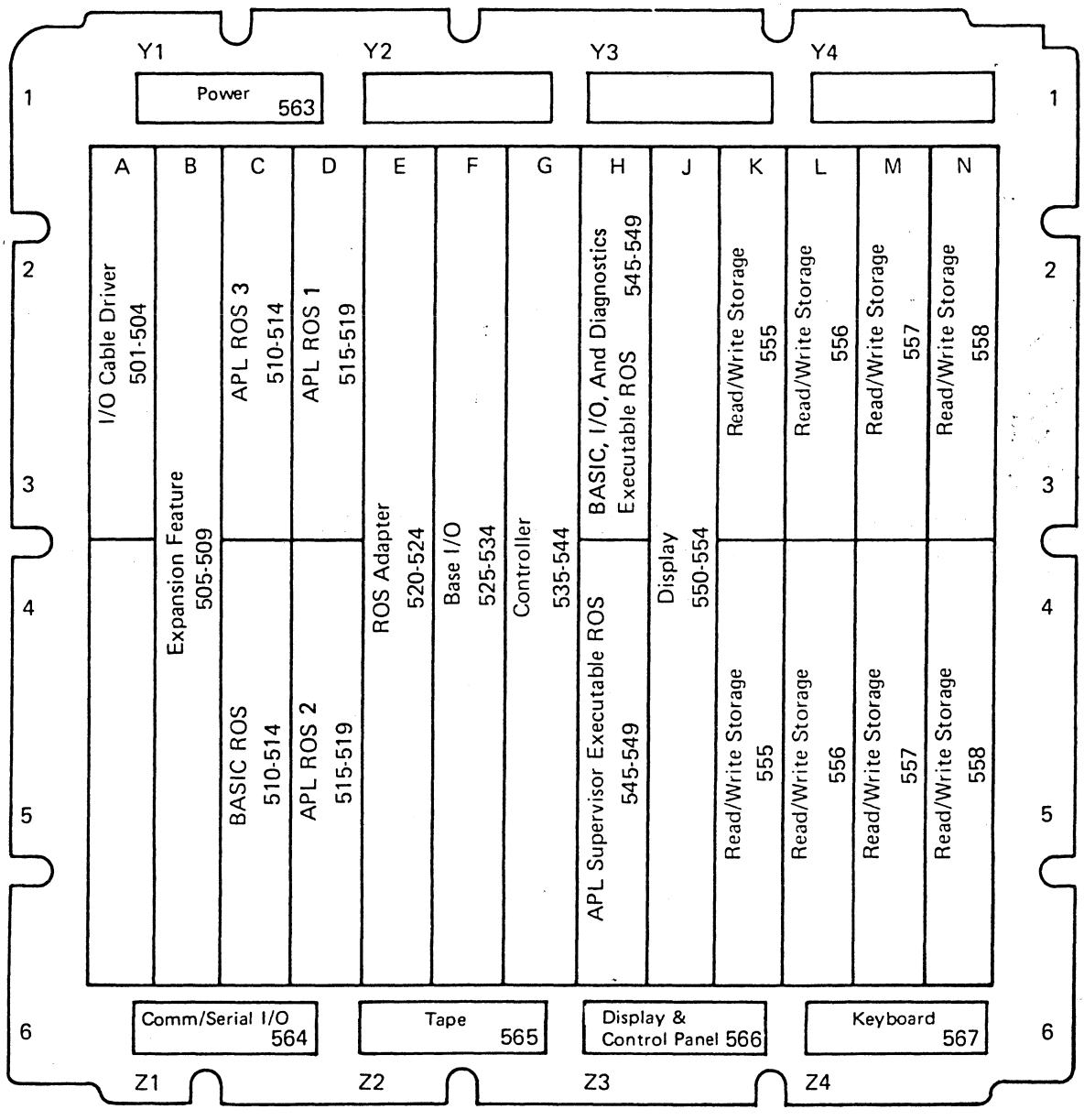
- Serial I/O adapter microprogram
- Serial I/O adapter cabling

Refer to the *IBM 5100 Communications/Serial I/O Maintenance Information Manual,* SY31-0429, for additional information.

This page intentionally left blank.

This page intentionally left blank.

## A1 Board Locations




RM5009

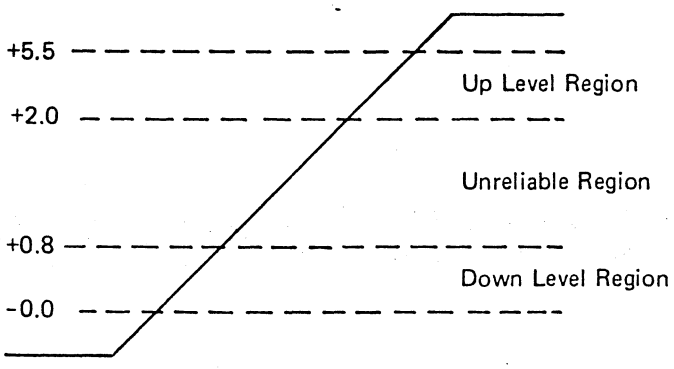
| | |
|---|---|
| Tape Unit | 570-574 |
| Keyboard Adapter | 575-579 |
| Auxiliary Tape Unit | 580-589 |
| DC Power | 590-594 |
| AC Power | 595-599 |

*Note:* Refer to the MAP binder for card part numbers.

## Logic Symbol Legend

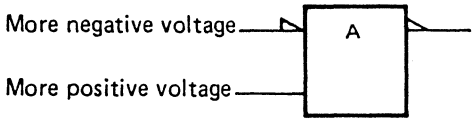
### GENERAL LOGIC INFORMATION

**Voltage Switching Levels**



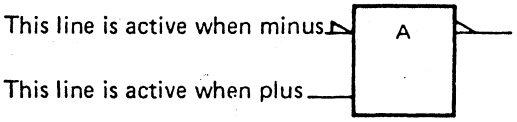

**Polarity**

Polarity is indicated by a wedge ( ◺ ) or no wedge.




**Active Level**

Active level is the line level that conforms to the edge-of-block character for that line.




**Logic Symbols**

| Line Function Symbol | Description |
|---|---|
| S | Sets a storage bit. |
| R | Resets a storage bit or register. It can also be used to hold a logic block off.<br><br>*Note:* Multiple R lines can be present in a logic block. Any one line resets the logic block and does not depend on all R lines being active. |
| J | Sets a storage bit. |
| K | Resets a storage bit.<br><br>*Note:* Simultaneous application of J and K complements the stored bit. |
| T | Complements a storage bit. |
| D | Data into a storage bit. Sets indicated polarity; resets opposite polarity. |
| C | Control input to a polarity hold, storage device, or register. |
| CD | The control data line under control of the C line. |
| SD | Shifts bits in register down one bit position. |
| SU | Shifts bits in register up one bit position. |
| SDD | Simultaneously sets a data bit into a register and shifts all the other bits in the register down one bit position. |
| SUD | Simultaneously sets a data bit into a register and shifts all other bits in the register up one bit position. |
| +n | Increases binary count in register by n (any decimal number). |
| -n | Decreases binary count in register by n (any decimal number). |
| G | Gate control. |
| n | Gated dependent line (for example, G1, G2, G3). |
| A | Common read/write address to MREG. |
| R/W | Read/write control line to MREG. |
| R/B | Reset output buffer to MREGB. |
| Decimal Numbers | Used to weight DCD (decoder) input and output lines. |

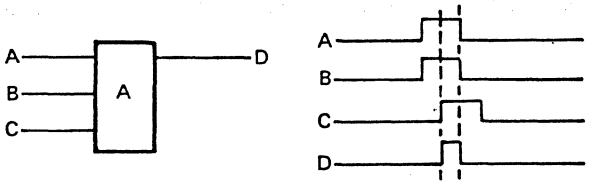
| Line Function Symbol | Description |
|---|---|
| Z | Common line used to indicate more than one input in a stacked block representation. |
| (N) | Indicates that the number of lines below the (N) symbol must be active in an MREG or MREGB. |

| Logic Block Symbol | Logic Function |
|---|---|
| A | AND |
| ALU | Arithmetic logic unit |
| ANO | Analog OR |
| AR | Amplifier |
| AR-CD | Core driver |
| AR-DF | Differential amplifier |
| AR-HD | Magnetic head driver |
| AR-ID | Indicator driver |
| AR-LD | Transmission line driver |
| AR-LT | Transmission line terminator |
| AR-MD | Magnet driver |
| AR-V | Voltage amplifier |
| COM | Common |
| CR | Diode |
| DCD | Decode |
| DET | Detector |
| EVEN | Even count |
| FF | Flip-flop |
| FL | Flip latch |
| MREG | Multiple register |
| MREGB | Multiple register with buffer output |
| MTX | Matrix |
| N | Inverter |
| ODD | Odd count |
| OE | Exclusive OR |
| OR | OR |
| DOT OR | Dot OR |
| OSC | Oscillator |
| PG | Parity generator |
| PH | Polarity hold |
| PWR | Power block |
| R | Resistor |
| REG | Register |
| REG – Bit Counter | Register with bit counter |
| REG – Clock Ring | Clock ring |
| REG – Bit Shift | Register with bit shift |
| SEL | Selector |
| SS | Singleshot |
| TD | Time delay |

## FUNCTIONAL BLOCKS

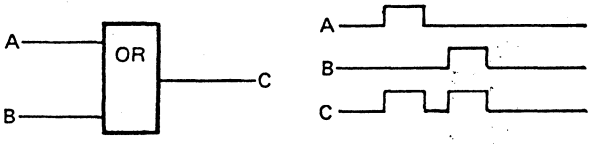
### AND (A)

The output of the AND block is active when all of its inputs are active.



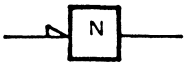

### OR (OR)

The output of the OR block is active when one or more of its inputs are active.



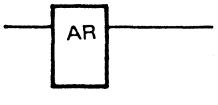

### Inverter (N)

The output of the inverter is of opposite potential to the input.



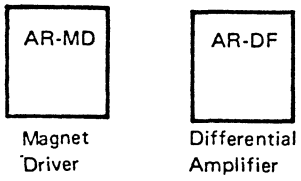

### Amplifier (AR)

The amplifier provides driving energy and an impedance match to other blocks. The amplifier output is active only when the input is active.



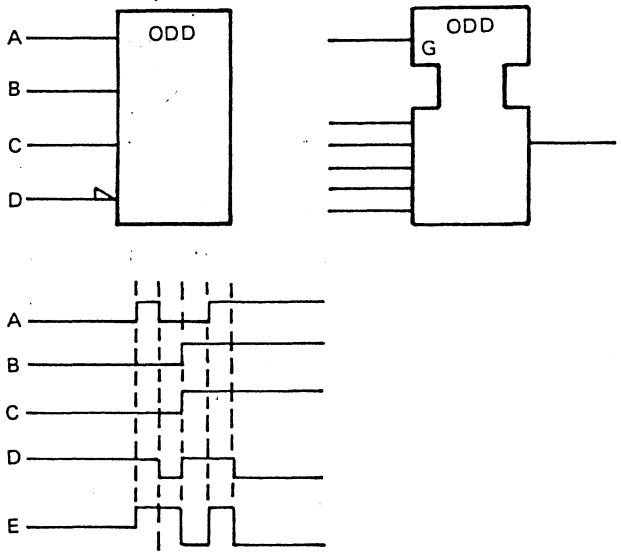
A suffix is added to a logic block function to clarify logic usage. The suffix symbol is always placed to the right of the block function symbol and is separated by a blank or a dash. The following suffixes are used with amplifier blocks:

| | |
|---|---|
| AR-LT | Transmission line terminator |
| AR-LD | Transmission line driver |
| AR-ID | Indicator driver |
| AR-CD | Core driver |
| AR-HD | Magnetic head driver |
| AR-MD | Magnet driver (relay, clutch, solenoid, etc) |
| AR-V | Voltage amplifier/analog voltage signal |
| AR-DF | Differential amplifier. |



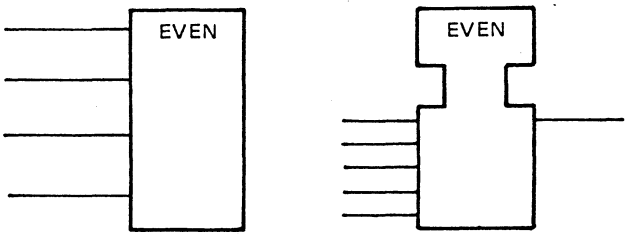

### Odd Count (ODD)

The output of odd count is active only when an odd number (such as 1, 3, 5, and 7) of inputs are active. A G (gate control) input might be present in the common section of an ODD block (refer to *Functional Logic Blocks with Common Inputs*). When the G line is at the indicated polarity, it gates the input lines into the ODD logic block and the output line is determined by the odd function of the block. When the G line is opposite the indicated polarity, the output line is also opposite its indicated polarity.



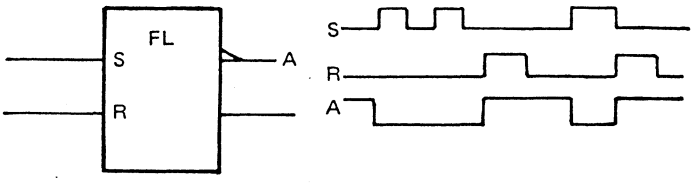

### Even Count (EVEN)

The output of even count is active only when an even number (such as 0, 2, 4, and 6) of inputs are active. A G (gate control) input might be present in the common section of an EVEN block (refer to *Functional Logic Blocks with Common Inputs*). When the G line is at the indicated polarity, it gates the input lines into the EVEN logic block and the output line is determined by the even function of the block. When the G line is opposite the indicated polarity, the output line is also opposite its indicated polarity.

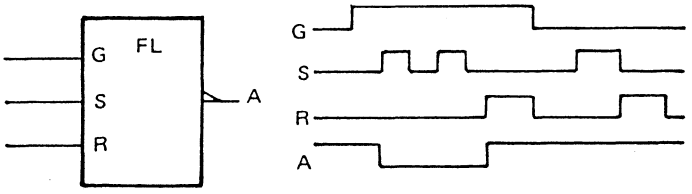



### Flip Latch (FL)

The FL is a storage element that has S (set) and R (reset) inputs. When the set input assumes its indicated polarity, the outputs assume their indicated polarity. The FL remains set until the R input assumes its indicated polarity and the outputs become inactive.



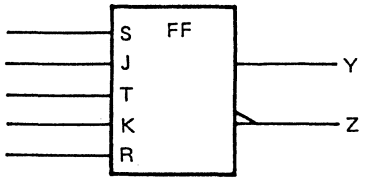

A G (gate control) input can also enter the FL block. When the G is at its indicated polarity, it allows the shift to the active state of the S line to set the FL. When the G line is opposite its indicated polarity, it blocks the active state of the S line from setting the FL.



### Flip-Flop (FF)

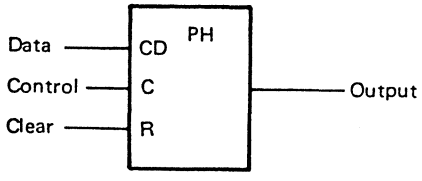
The flip-flop has two stable states, the 1, or set state and the 0, or reset state. The flip-flop block normally has two outputs, a 1 output and a 0 output. A line from the upper part of the block represents the 1 output and a line from the lower part of the block represents the 0 output.

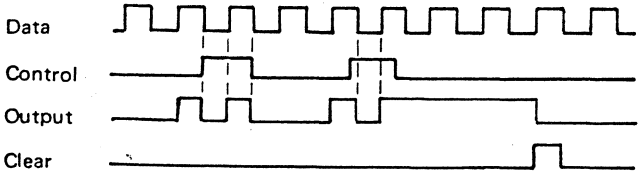
A flip-flop can have five types (S, R, J, K, and T) of inputs, in different combinations. Inputs J and K, respectively, act like inputs S and R in the flip latch except that simultaneous application of a J set and K reset complements the output. The T input complements each output. In the following illustration, a simultaneous S-R (set-reset) input causes output Y to follow the set (+) and output Z to follow the reset (-). If any other inputs are active during simultaneous S-R input, the outputs are undefined.




### Polarity Hold (PH)

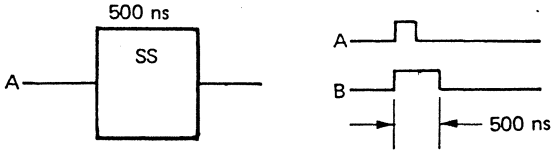
The output of this block follows the CD (data) line as long as the C (control) line is active. When the control input goes inactive, the output remains at whatever polarity it has at that moment. The PH block can have an R (reset) input; if so, when the reset input is active, the output is inactive.



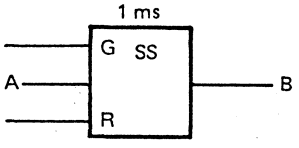

### Singleshot (SS)

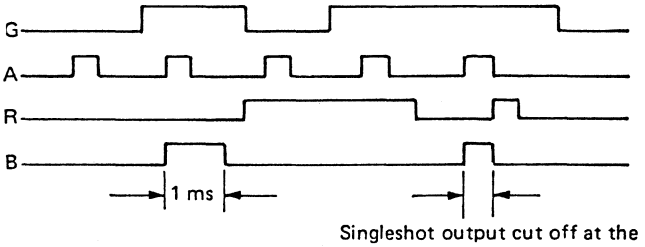
The output of the singleshot becomes active when the input is active. The output remains active for a time characteristic of the particular block. Regardless of the length of the input signal, the singleshot always has the time duration shown above the block.




*With Gate Control*

A G (gate control) input, when it stands at its indicated polarity, allows the line that fires the singleshot to fire the singleshot when that line is at its indicated polarity. When the G line is opposite its indicated polarity, it blocks the line that fires the singleshot.



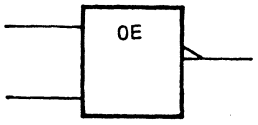

Singleshot output cut off at the time the R input went active.

*With Reset*

The R (reset) input to a singleshot, when it is at the indicated polarity, resets the output regardless of the singleshot time duration.

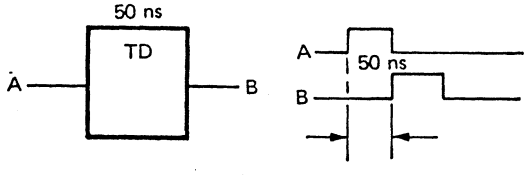
### Exclusive OR (OE)

The output of an exclusive OR block is active when only one of its inputs is active.



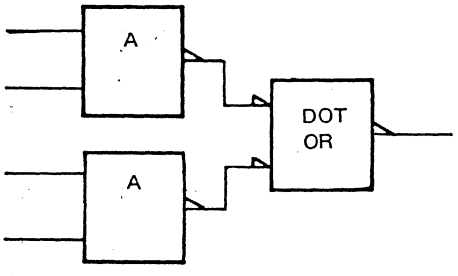

### Time Delay (TD)

The time delay block delays the signal without distorting the signal. The time delay of the signal is above the block.



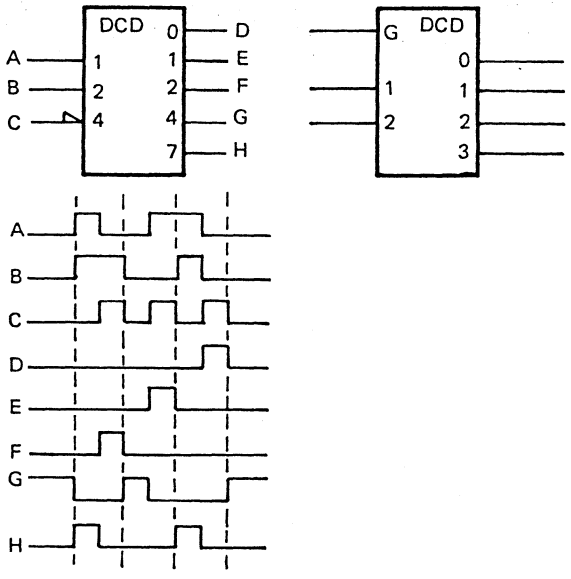

### Dot OR (DOT OR)

This block represents the physical connection of the outputs from two or more logic blocks. The polarity of all inputs and outputs is the same. The DOT OR block represents outputs wired together to form the OR function. In the following illustration, either AND being active causes the output of the DOT OR to be minus.

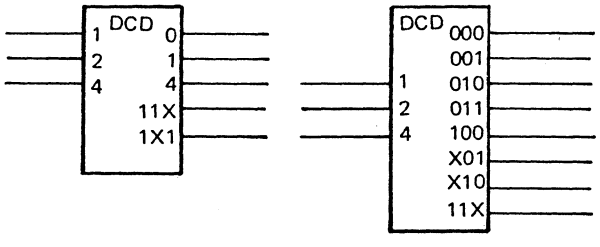



### Decoder (DCD)

The decoder translates a group of related inputs into a specific output. Inputs are numbered from the top in binary progression: 1, 2, 4, 8, and so on. The output equals the sum of the active inputs. When the G (gate control) line, if present, is at the indicated polarity, it allows the decode function to be performed. When the G line is opposite its indicated polarity, no decode function is performed and all the output lines are inactive.

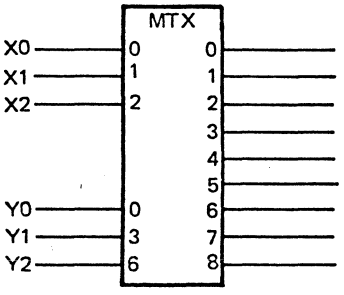



A *don't care* condition can exist and is defined as follows: When a particular input line does not affect an output, the bit position corresponding to that input line is an X. For example, binary output 11X is active when input lines 2 and 4 are active. The status of line 1 has no effect. When using the don't care condition, it is possible to have more than one output line active at the same time. For example, if input 1 is active and inputs 2 and 4 are inactive, output 001 and X01 are active.



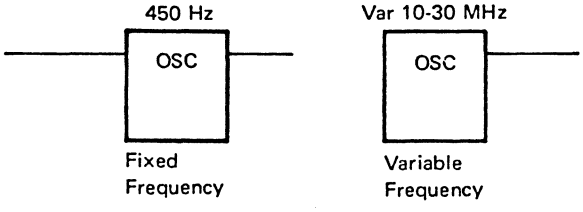
### Matrix (MTX)

The matrix expands the capability of the decoder for storage addressing when partial decoding is used. The matrix logic block has two or more groups of inputs. The decimal numbered output remains at the indicated polarity when it equals the decimal sum of one line from each input group at its indicated polarity. In the following illustration an X and Y line are required to address one position. To activate output line 7, X1 and Y2 input lines must be active.



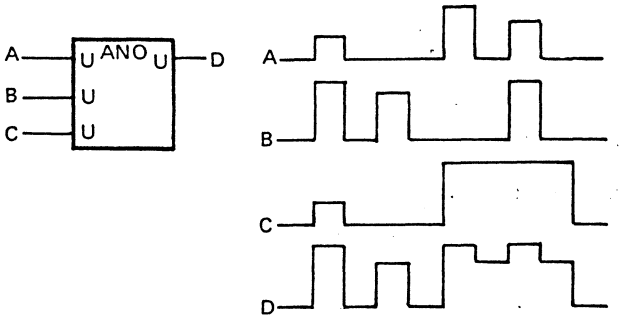

### Oscillator (OSC)

The oscillator produces a uniform, repetitive output either continuously or during the application of a single input at the polarity indicated. The operating frequency is shown above the block.



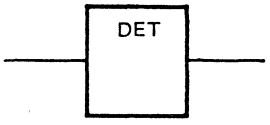

### Analog OR (ANO)

The amplitude of the output signal remains at a value corresponding to that of the input signal having the greatest amplitude in the direction shown by the line input edge-of-block character, U (up) or D (down). The direction must be the same for all inputs.



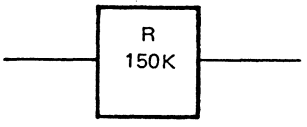

### Detector (DET)

The detector acts upon modulated signals to recover a carried signal of lower frequency. There is only one input to this block.



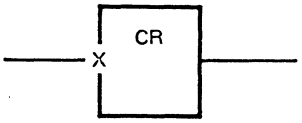

### Resistor or Network of Resistors (R)

A resistor provides the resistance specified within the block. Power dissipation and tolerance are also specified within the block, when applicable.



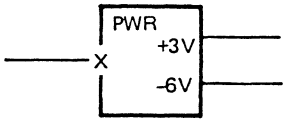
### Diode (CR)

The diode block specifies a diode. For a zener diode, the breakdown voltage is also specified. The edge-of-block character X indicates the anode connection.



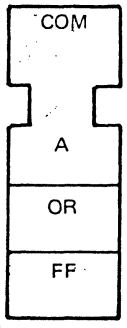

### Power Block (PWR)

The power block generates signal levels. No polarity assignment is made to the signal outputs. Voltage levels are placed inside the block adjacent to their respective output. Nonlogic lines, such as a bias or shield, are indicated by an edge-of-block character X.



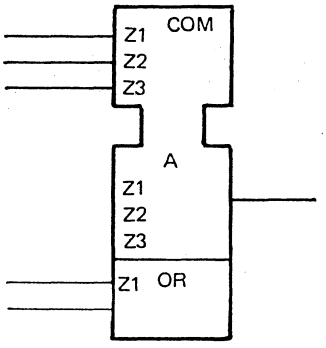

## FUNCTIONAL LOGIC BLOCKS WITH COMMON INPUTS

Functional logical blocks with common inputs have a data section and a common section as shown in the illustration. The data section is a group of stacked functional blocks. The common section contains input lines common to all functions.



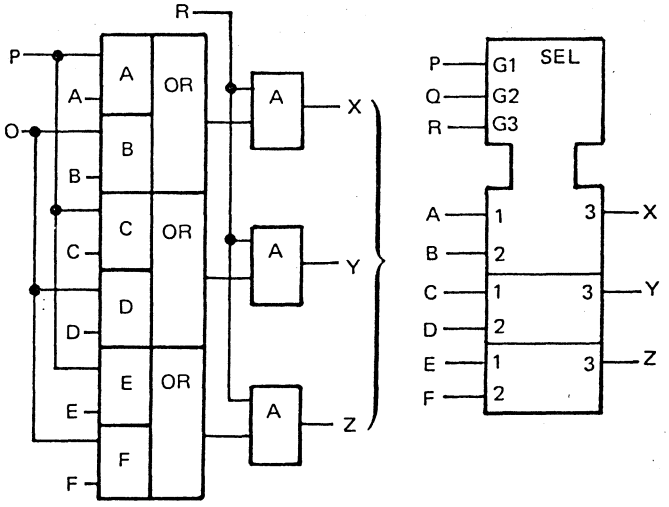

### Common (COM)

Common is a way of condensing logic. Common inputs are indicated by the line function symbol Z. The Z input line enters the common section of the logic block and the specified Z (1, 2, or 3) designation is repeated in all the functional blocks affected. However, if a common input is common to all the data blocks, it might not be labeled at all, or it might be labeled Z only.



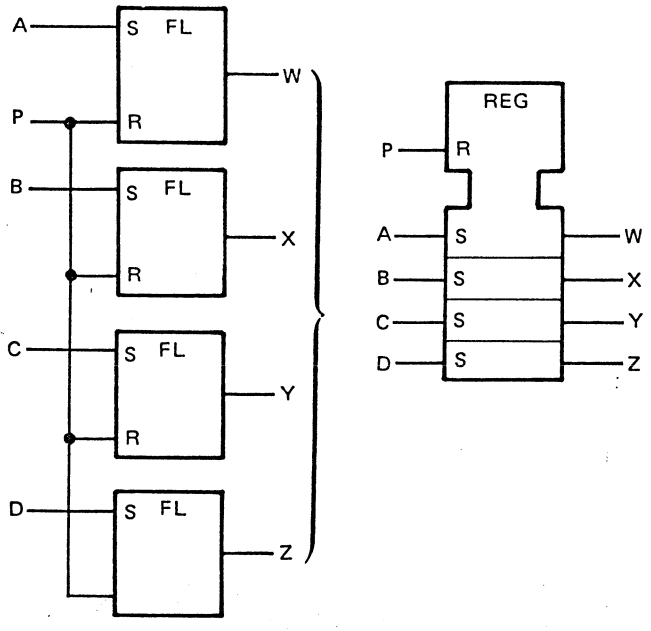

### Selector (SEL)

The selector is a gating device. The common section of the block contains the gates. These lines are designated G1, G2, . . . Gn. The lower section contains the gated data lines. In the following illustration, G1 and G2 are input gates and G3 is an output gate as indicated by their position in the data block.



## Register (REG)

The register is a storage device composed of storage blocks (FF, FL, PH) that have a common control such as set, reset, gating, etc.

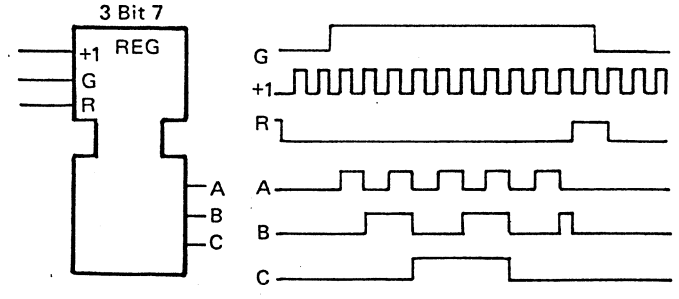



*Bit Counter*

The REG symbol can represent a bit counter. The common section is labeled with the REG symbol and the following inputs enter the common section:

+n (any decimal number) — When this input assumes its indicated polarity, the decimal quantity n is added to the bit count contained in the register; that is, the register is increased by n.

-n (any decimal number) — When this input assumes its indicated polarity, the decimal quantity n is subtracted from the bit count contained in the register; that is, the register is decreased by n.

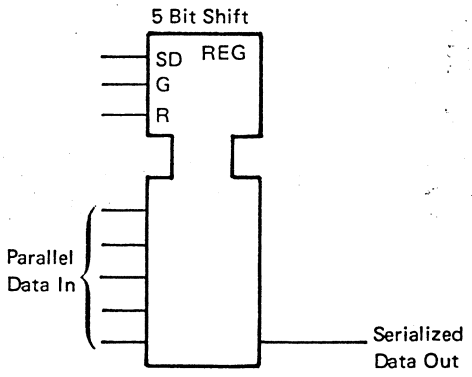



*Bit Shift*

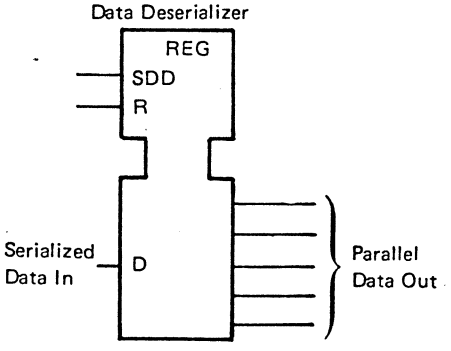
The REG symbol can represent a shift register. The common section is labeled with the REG symbol and the following inputs enter the common section:

SD (shift down) — When this input assumes its indicated polarity, the data content (1 or 0) shifts from the top bit position in the data section to the bit position below. Similarly, the content of every other bit position in the data section shifts to the bit position below.

SU (shift up) — When this input assumes its indicated polarity, the data content (1 or 0) shifts from the bottom bit position in the data section to the bit position above. Similarly, the content of every other bit position in the data section shifts to the bit position above.



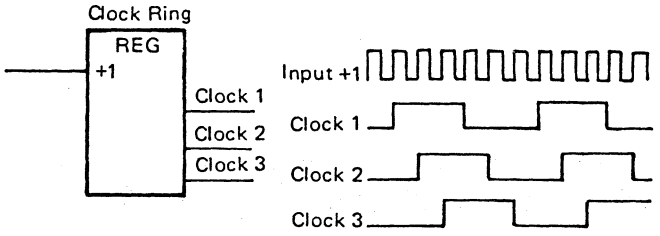

The shift input gates in a D (serial data bit) and the notation becomes SDD (shift down data) or SUD (shift up data).




Circuits

*Clock Ring*

A clock ring is a free-running binary trigger ring consisting of a number of polarity hold latches connected in series. The output of one latch is the input to the next latch.




**Parity Generator (PG)**

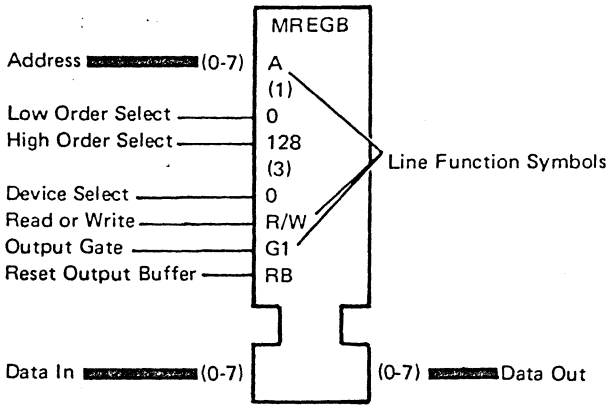
This block generates a parity (P) bit to keep the bit count odd.

Read only storage and read/write storage – The symbol for the functional block is MREG. An MREG block with buffered output is labeled MREGB.

Address lines – Address lines are identified by line function symbol A. The active bits in these lines select the positions to be accessed. The number in parentheses above the input group indicates the number of lines that must be active to get an active output. Activating the 0 line selects addresses 0 through 127. Activating the 128 line selects addresses 128 through 255.

R/W (read/write control) line – The R/W (read/write control line causes a read operation to be performed when the line is at the indicated polarity, and a write operation when the line is at the opposite polarity.

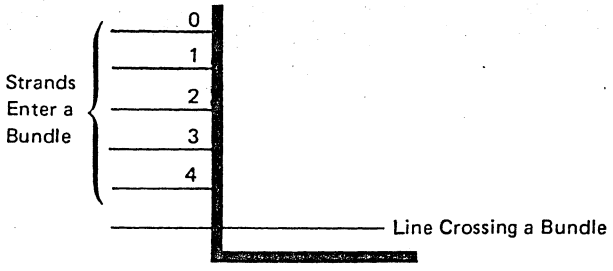
Output buffer reset line (RB) – The output buffer reset line, when it is at the indicated polarity, resets the output buffer.




## BUNDLING OF SIGNAL LINES

Bundling groups signal lines together that represent buses, and in some cases, similar functions, or similar sources (block output) and sinks (block input).

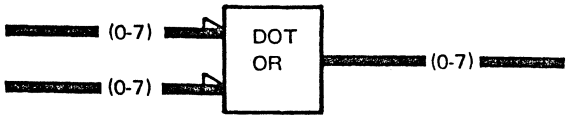
Each line (strand) is identified both at the point it enters the bundle and at the point it leaves the bundle. A single line that crosses the bundle is not identified, as shown in the following illustration:




*Note:* Logic lines always enter a bus from the left and exit a bus on the right. The active level of each line is shown at its source logic block or by its line name. All lines of a particular bundle might not have the same active level.

### Line Bundling with Dot OR Function

In some cases one bundle is shown dot ORed with another bundle such as:

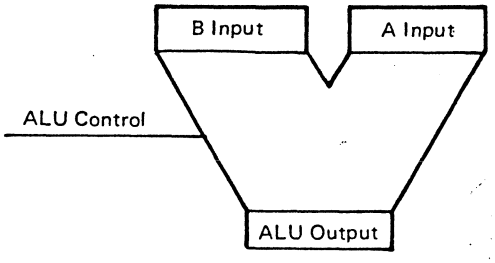



This representation conveys that a separate dot OR exists for each corresponding line of the two bundles. Also note the method used to identify the lines (0-7) within the bundle. This indicates 8 bits, beginning with bit 0 and ending with bit 7.

## SPECIAL SYMBOLS

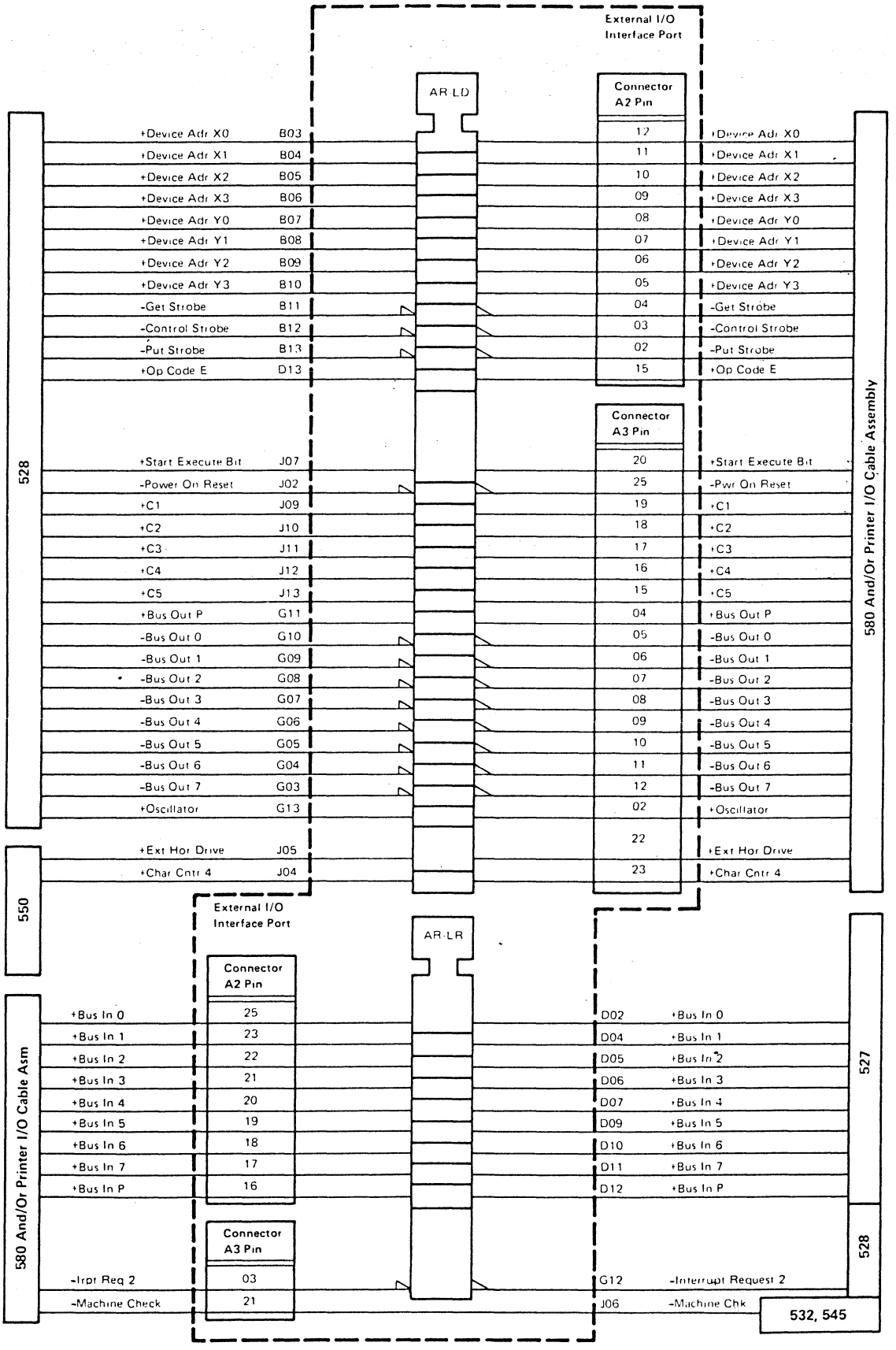
### ALU (Arithmetic Logic Unit)

Two inputs are fed into an ALU, one via the A input and one via the B input and some function is performed on the two inputs. The function such as add, subtract, complement, shift, and other arithmetic or logical functions is determined by the ALU control input. The result is passed to the ALU output.



This page intentionally left blank.

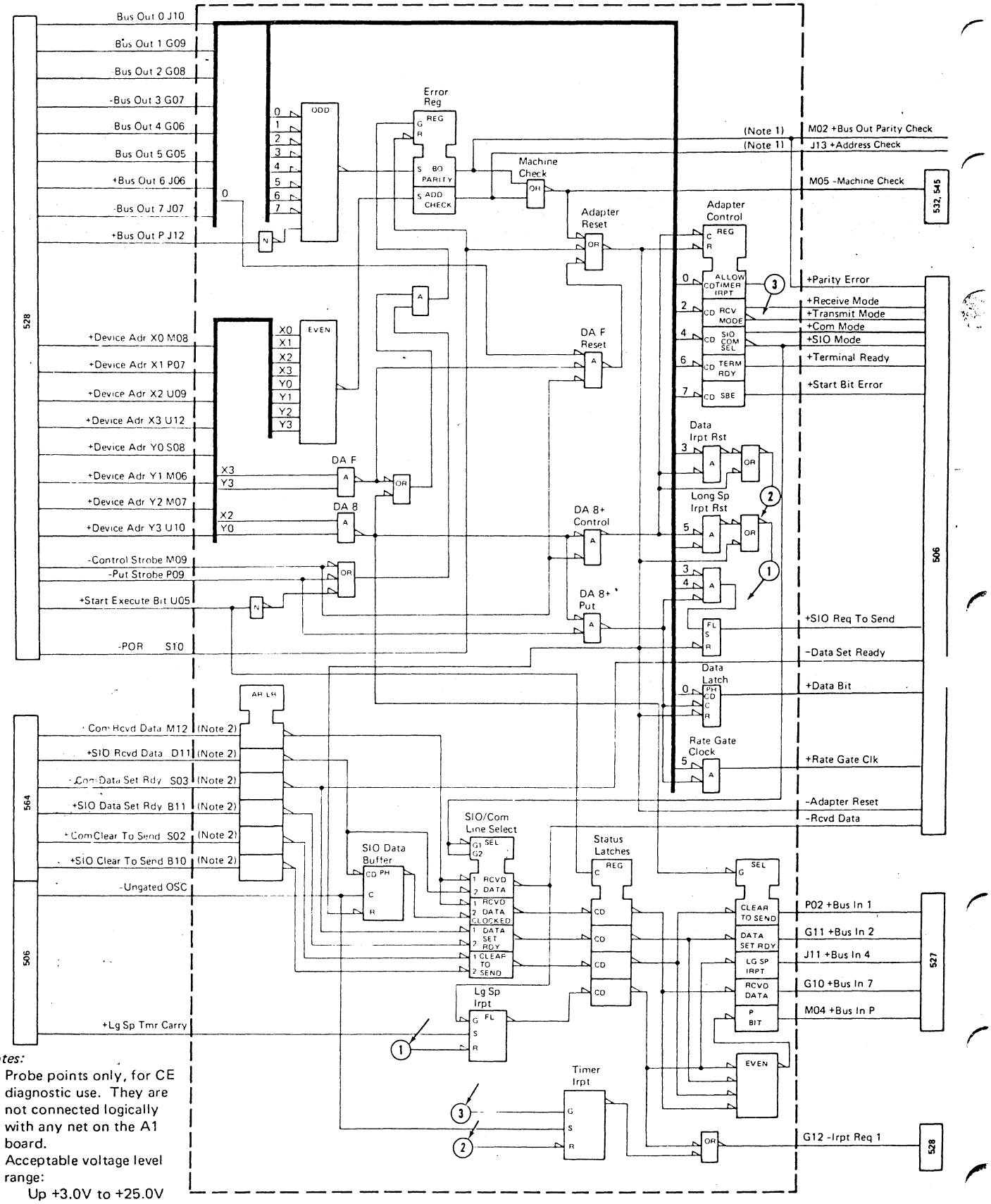
# 501 I/O CABLE DRIVER CARD A2




RM5022

# 505 EXPANSION FEATURE CARD B2

(Page 1 of 2)

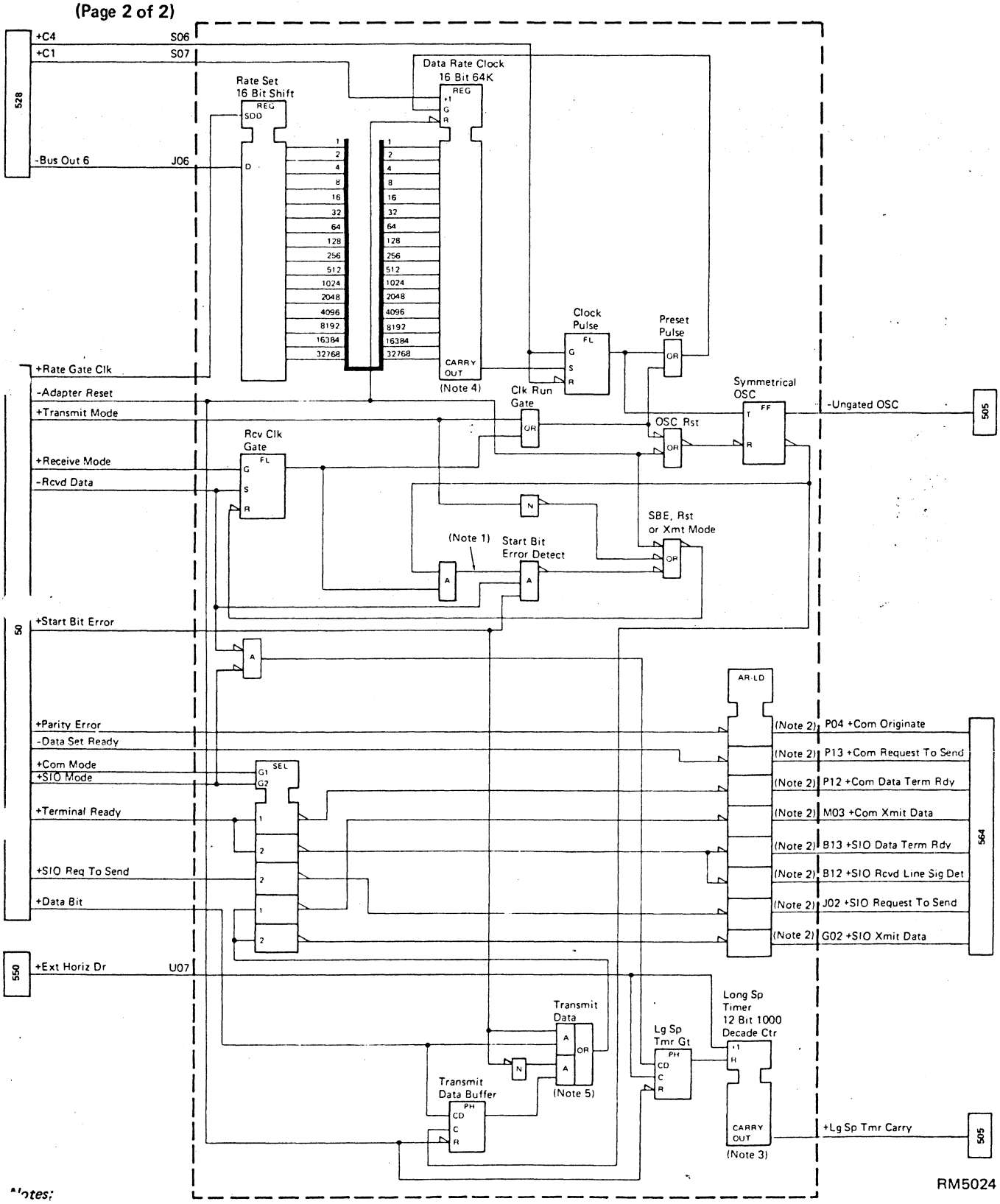



*Notes:*

1. Probe points only, for CE diagnostic use. They are not connected logically with any net on the A1 board.
2. Acceptable voltage level range:
   Up +3.0V to +25.0V
   Down -3.0V to -25.0V

# 506 EXPANSION FEATURE CARD B2

(Page 2 of 2)



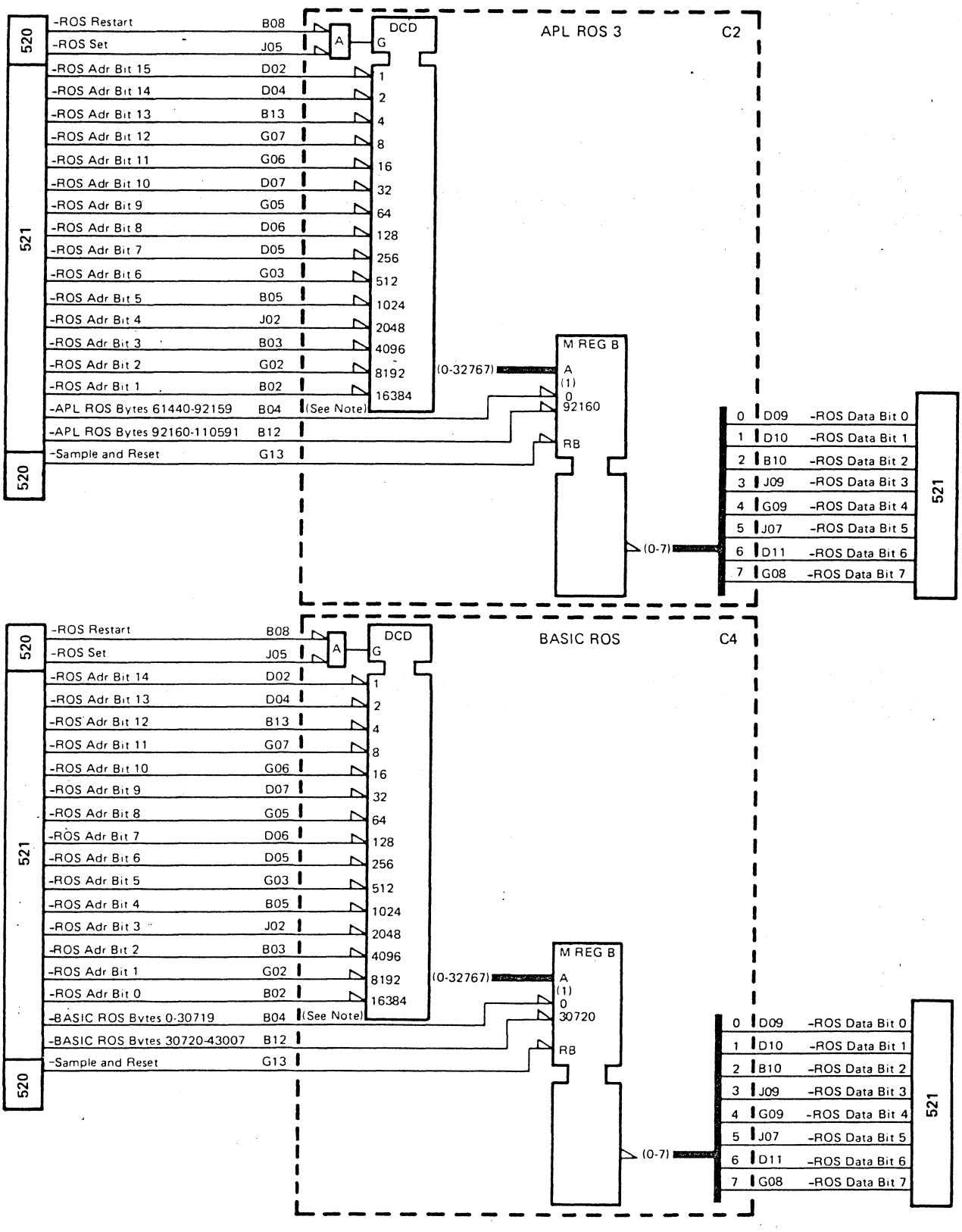

RM5024

Notes:

1. Logic line leaves card at P06, is looped on the board, and reenters the card at P10. Either point may be tested with an oscilloscope as a CE aid to determine clock frequency.
2. Acceptable voltage level range: Up +3.0V to +25.0V, down -3.0V to -25.0V.
3. Four internal lines of the long space timer leave the card, are looped on the board, and return to the card at the following pairs of pins: G03-G04, J04-J05, M10-S05 and S13-U13.
4. Four internal lines of the data rate clock leave the card, are looped on the board, and return to the card at the following pairs of pins: M11-M13, S04-U06, S09-S12, and S11-U11.
5. The 'start bit error' latch is on during data transmission only while running CE diagnostic programs; otherwise, normal data transfer is through the transmit data buffer.

Circuits

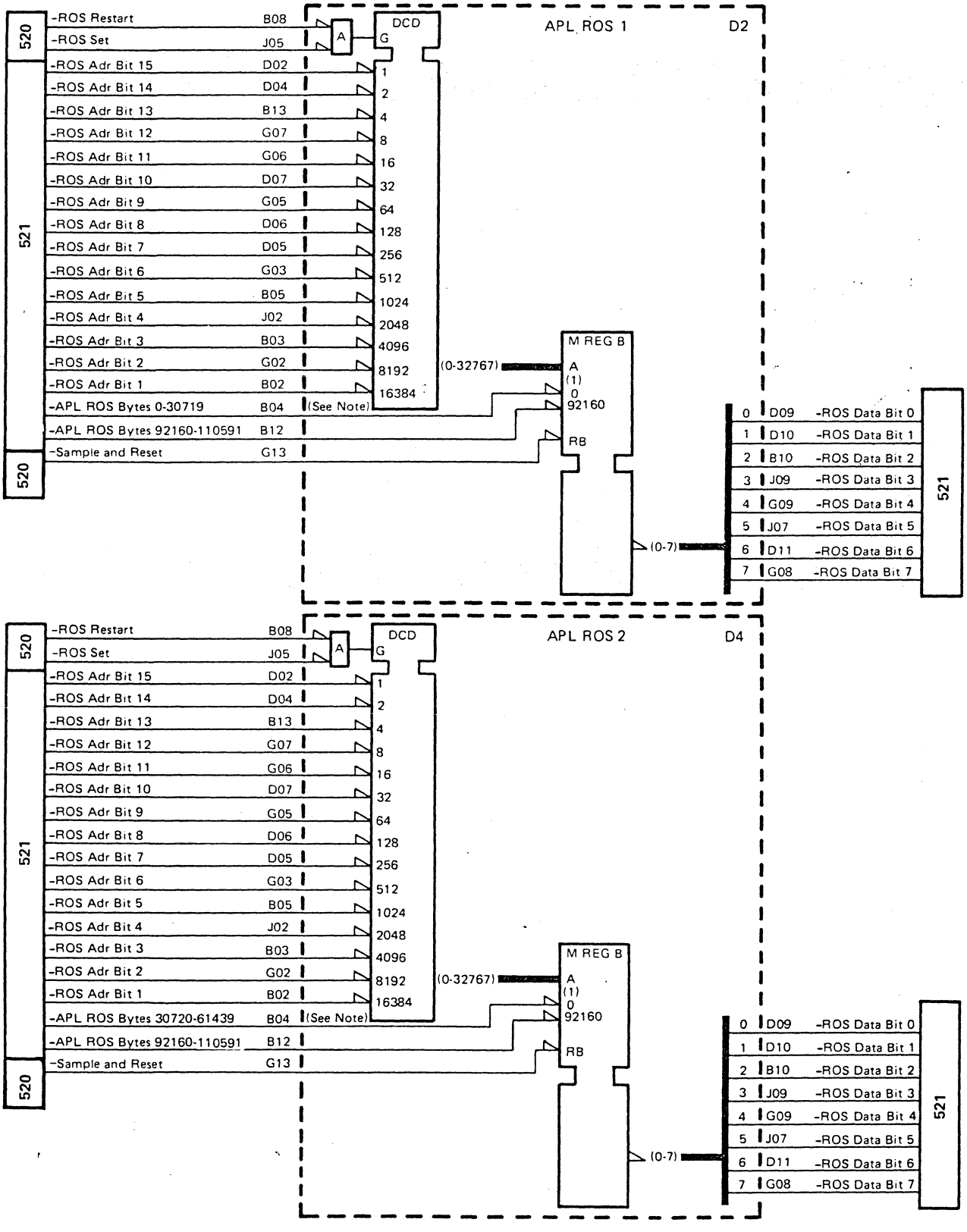
# 510 NONEXECUTABLE ROS CARDS C2 AND C4




*Note:* Pins B04, B06, B07, G04 and J04 are in this net — wired together on the board.

RM5025

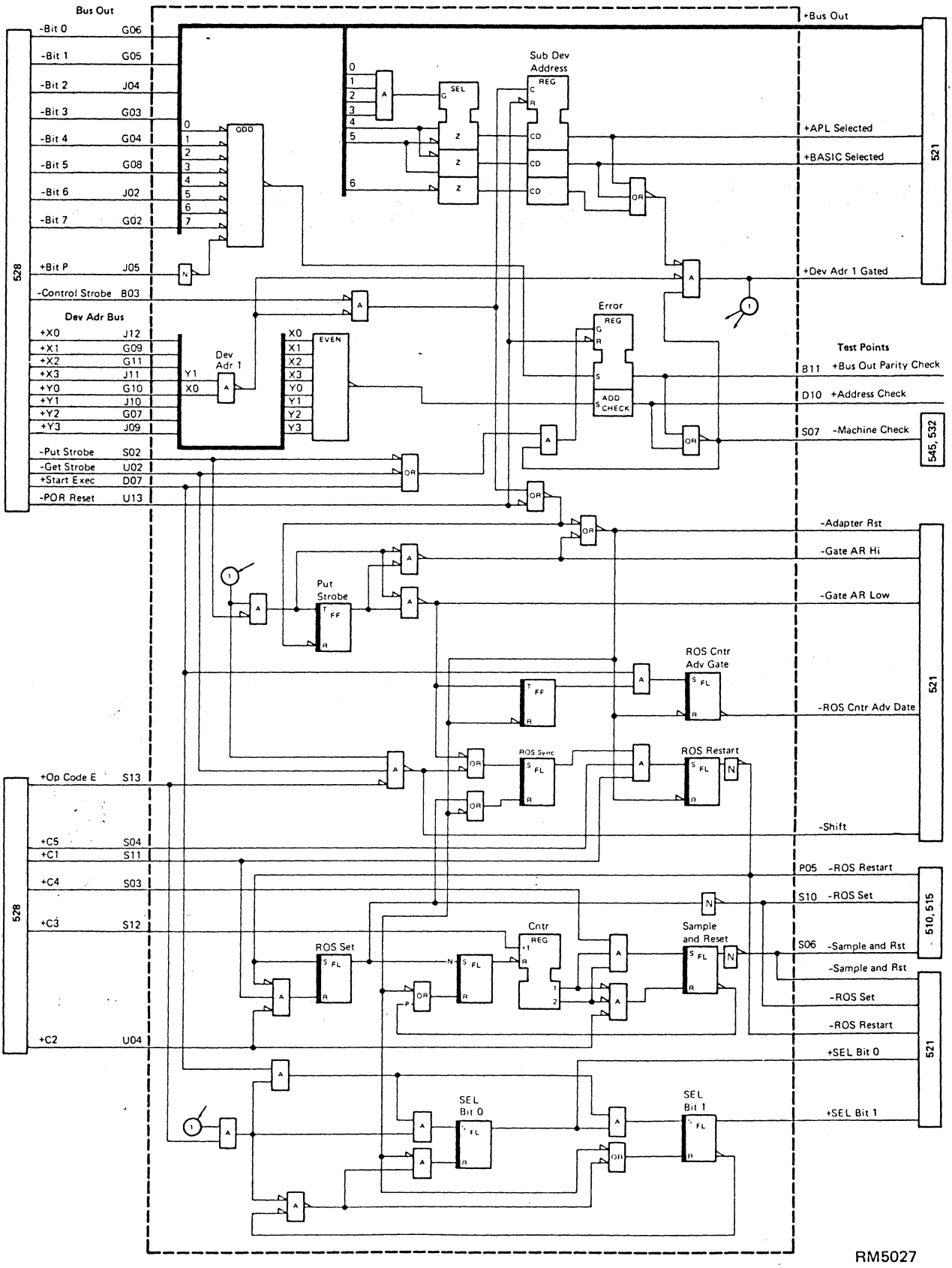
## 5.5 NONEXECUTABLE ROS CARDS D2 AND D4




*Note:* Pins B04, B06, B07, G04, J04 and B11 are in this net — wired together on the board.

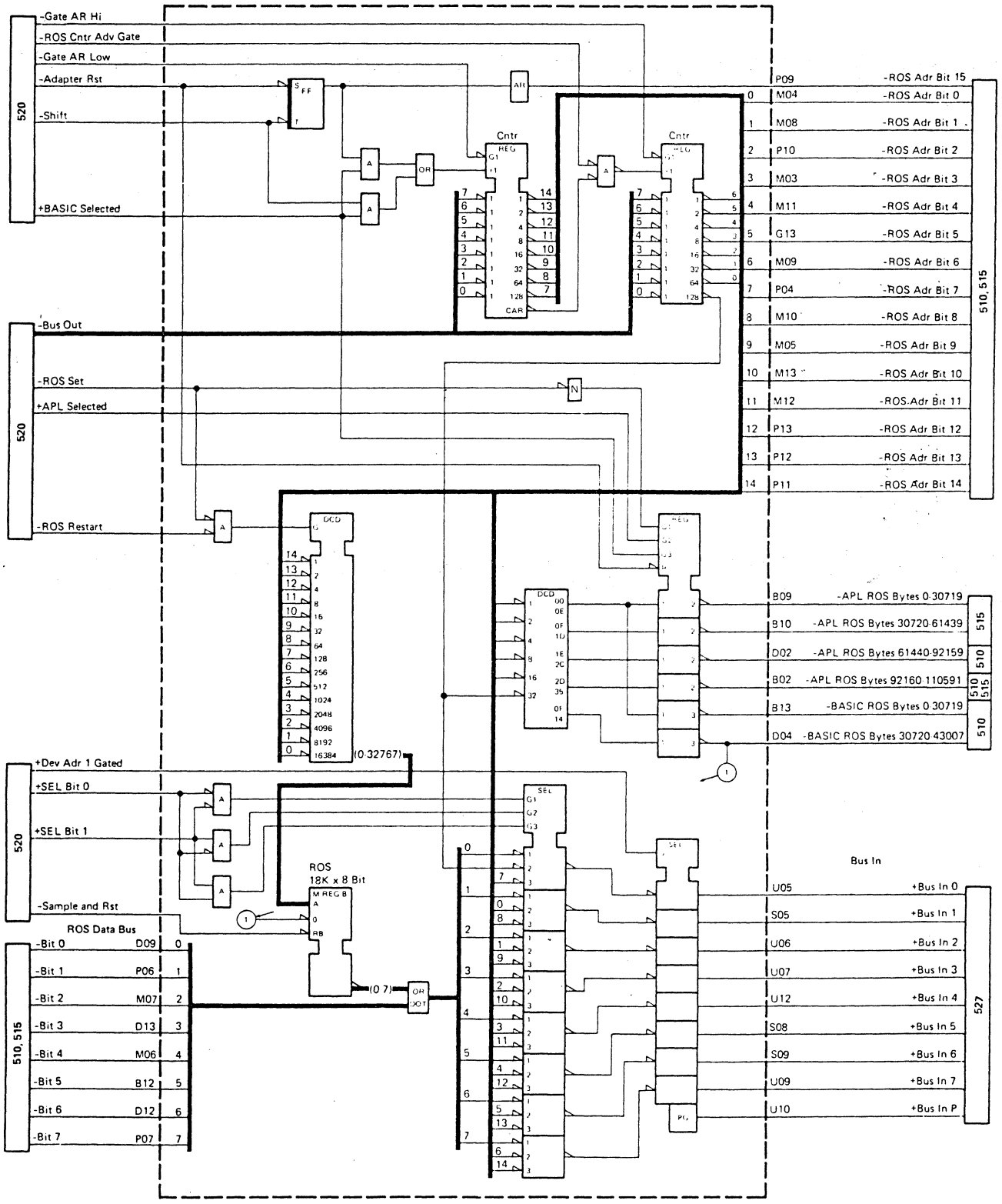
RM5026

# 520 ROS ADAPTER CARD E2
(Page 1 of 2)



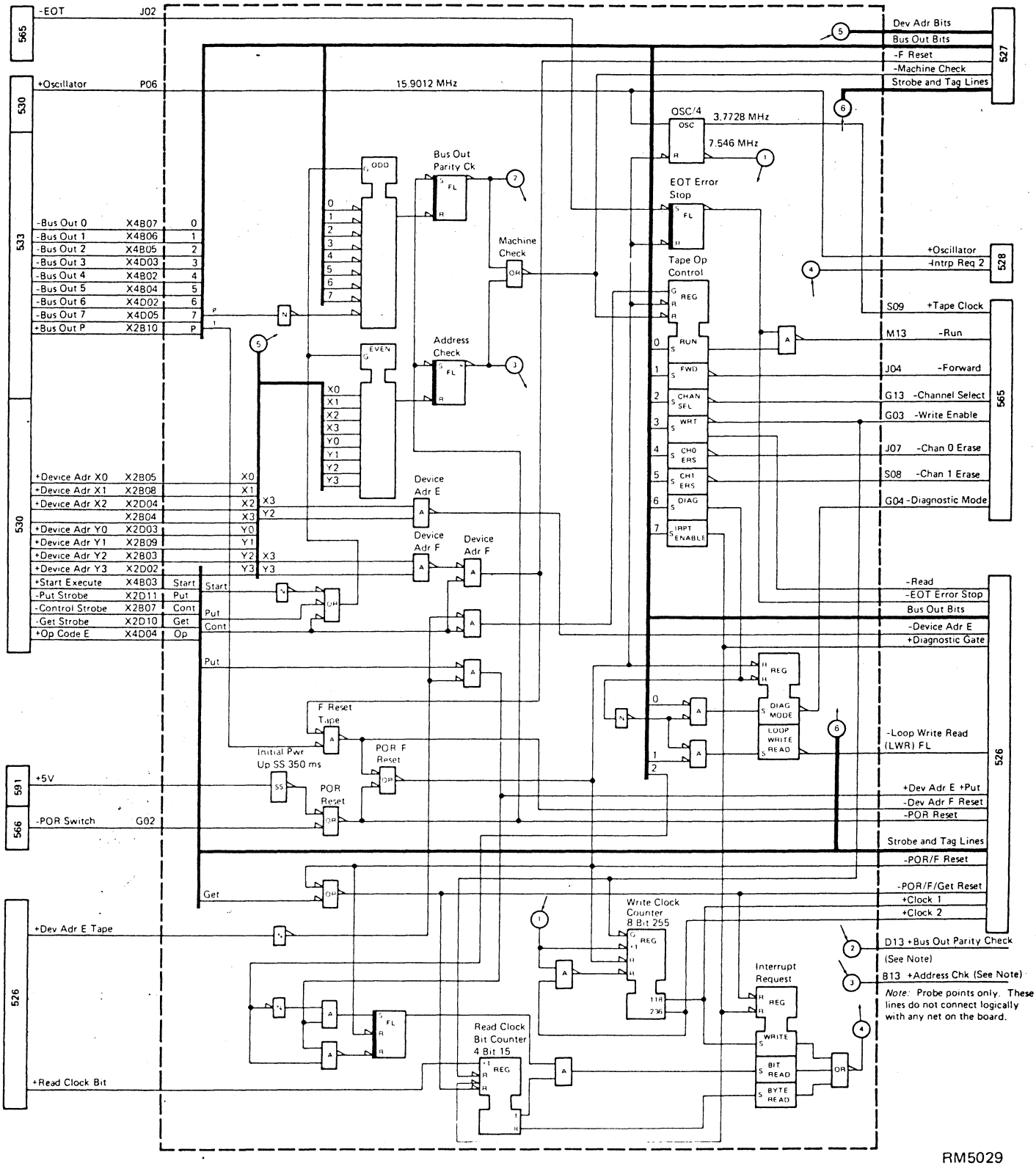
# ROS ADAPTER CARD E2
(Page 2 of 2)



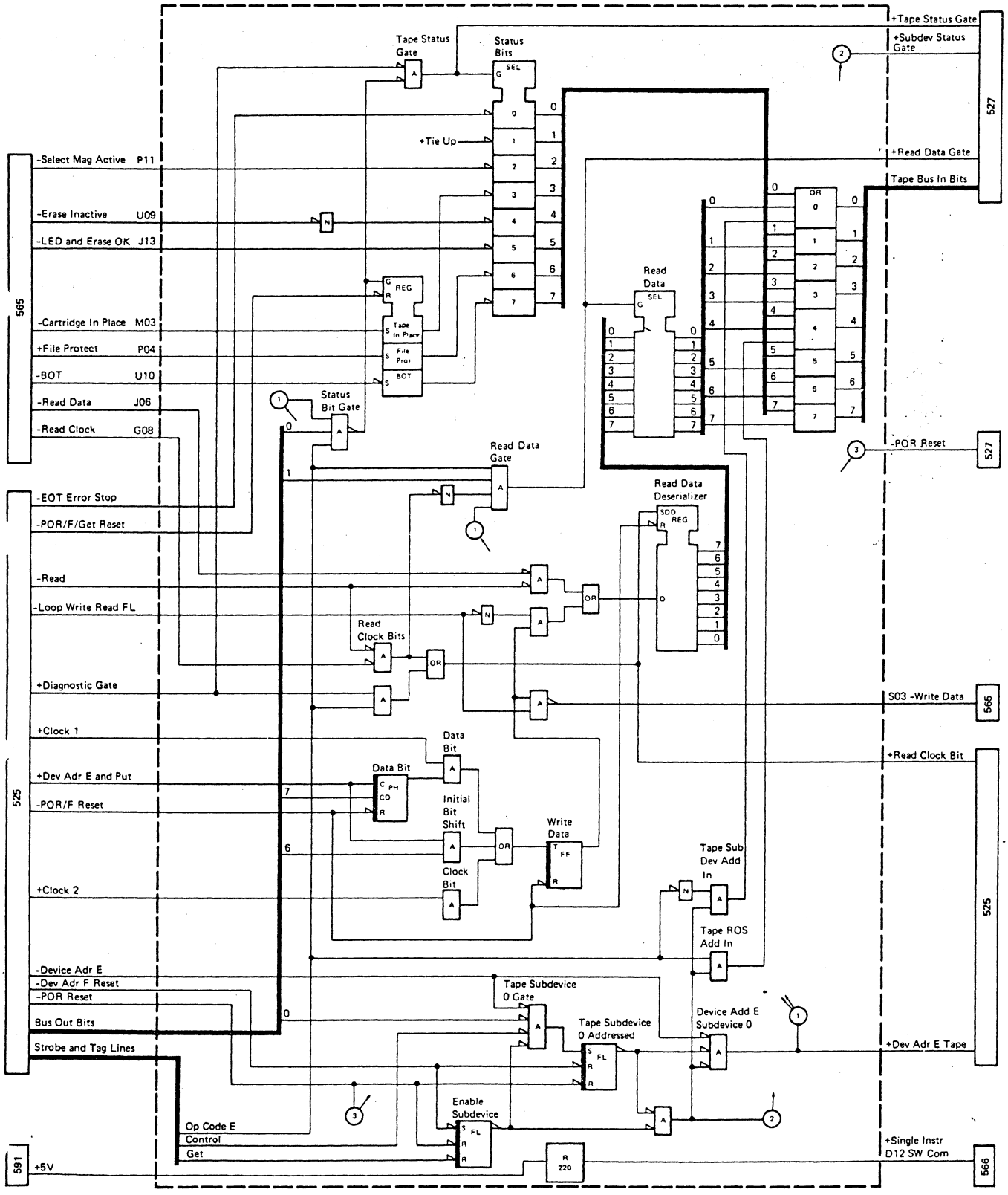



RM5028

# 525 BASE I/O CARD F2
(Page 1 of 4)

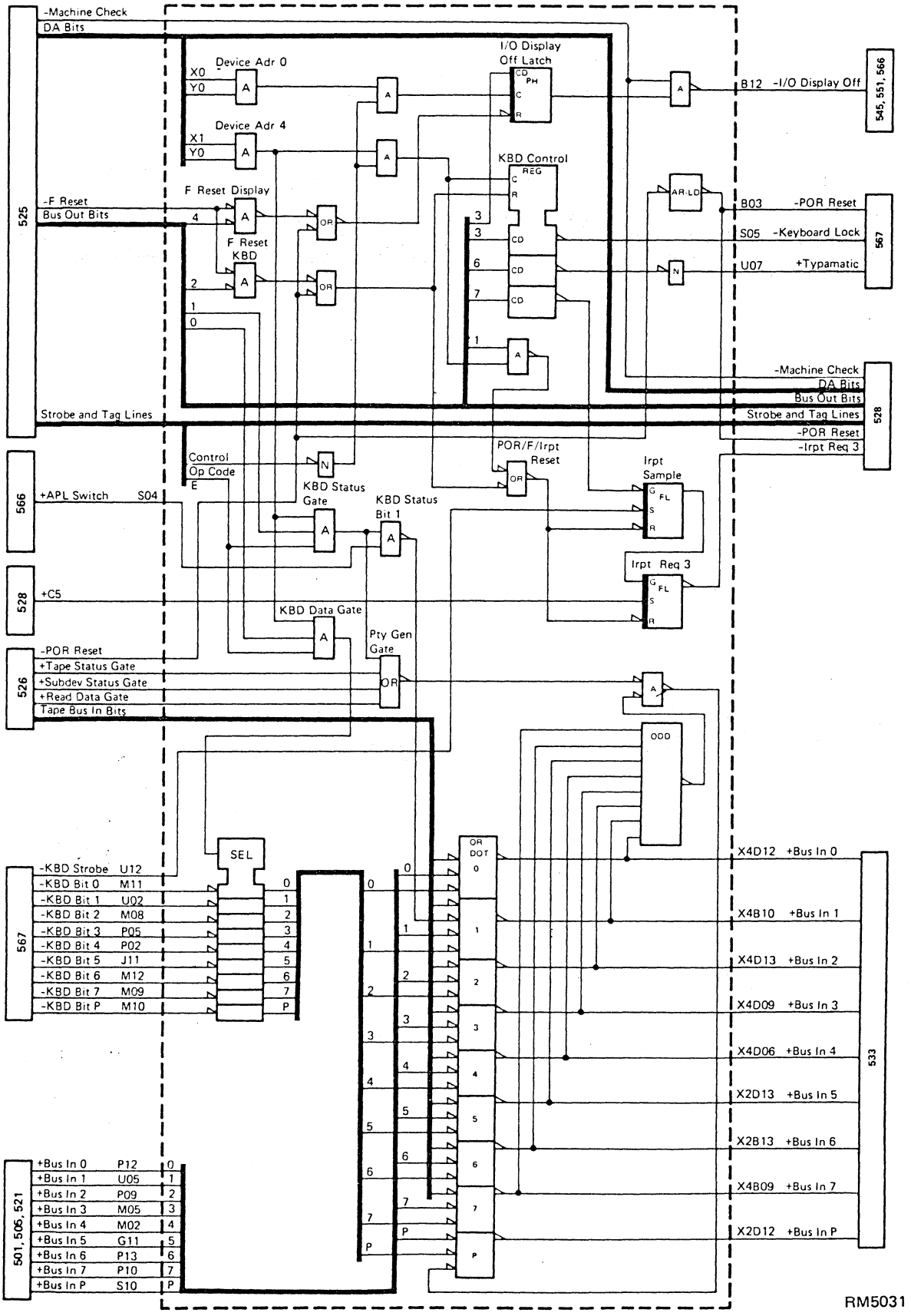


# 6 BASE I/O CARD F2
(Page 2 of 4)



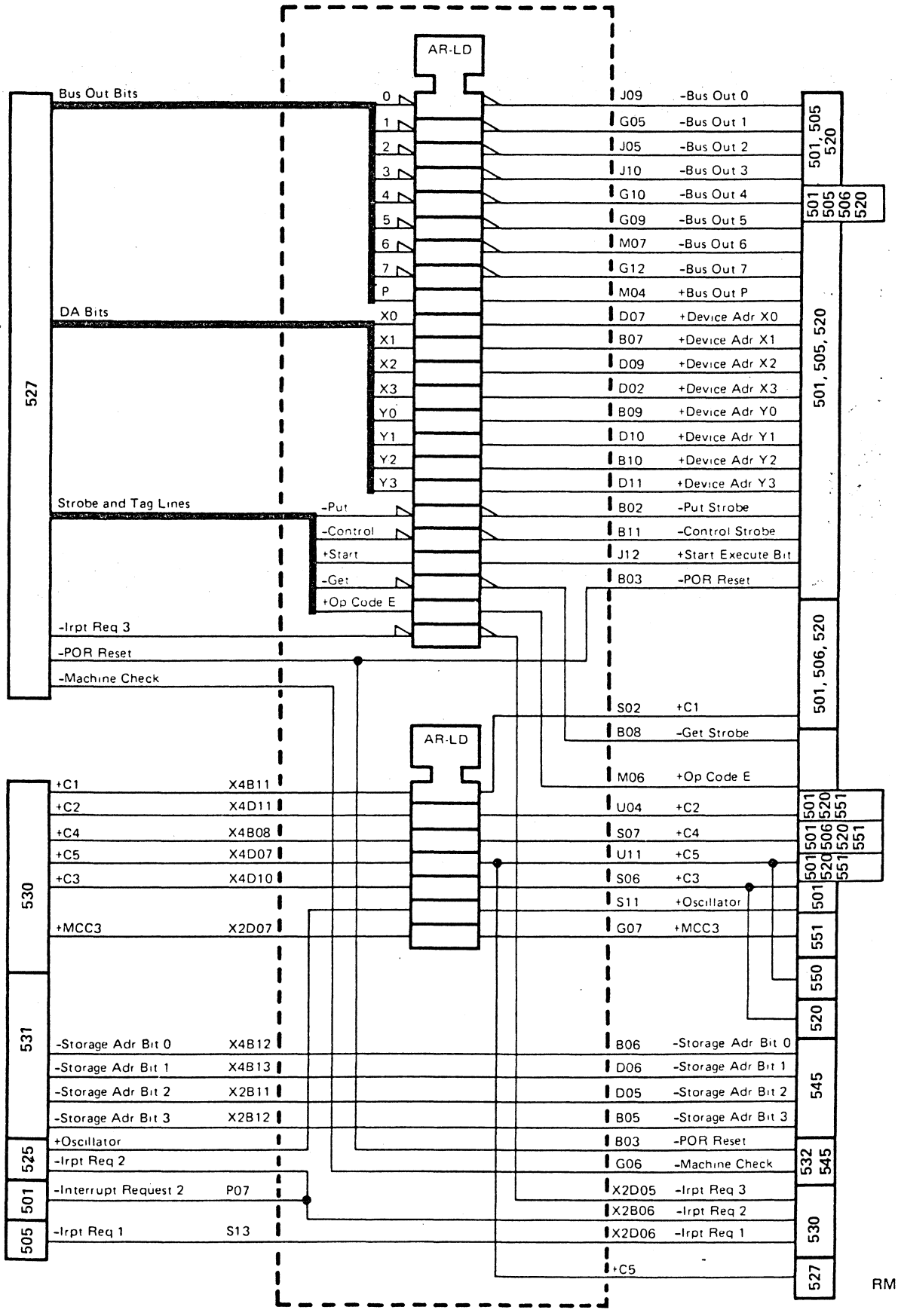

RM5030

# 527 BASE I/O CARD F2
(Page 3 of 4)




RM5031

# 528 BASE I/O CARD F2
(Page 4 of 4)

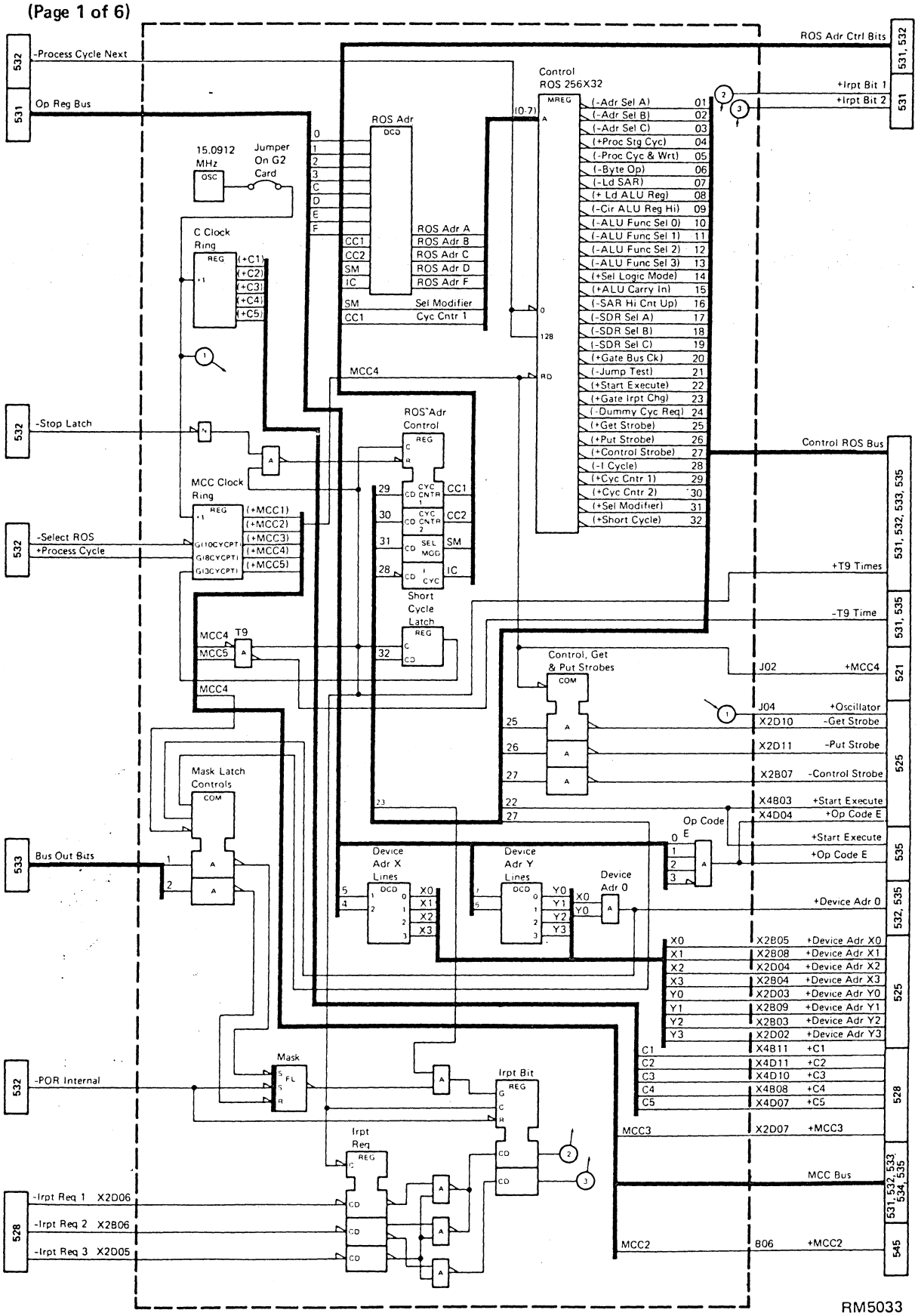


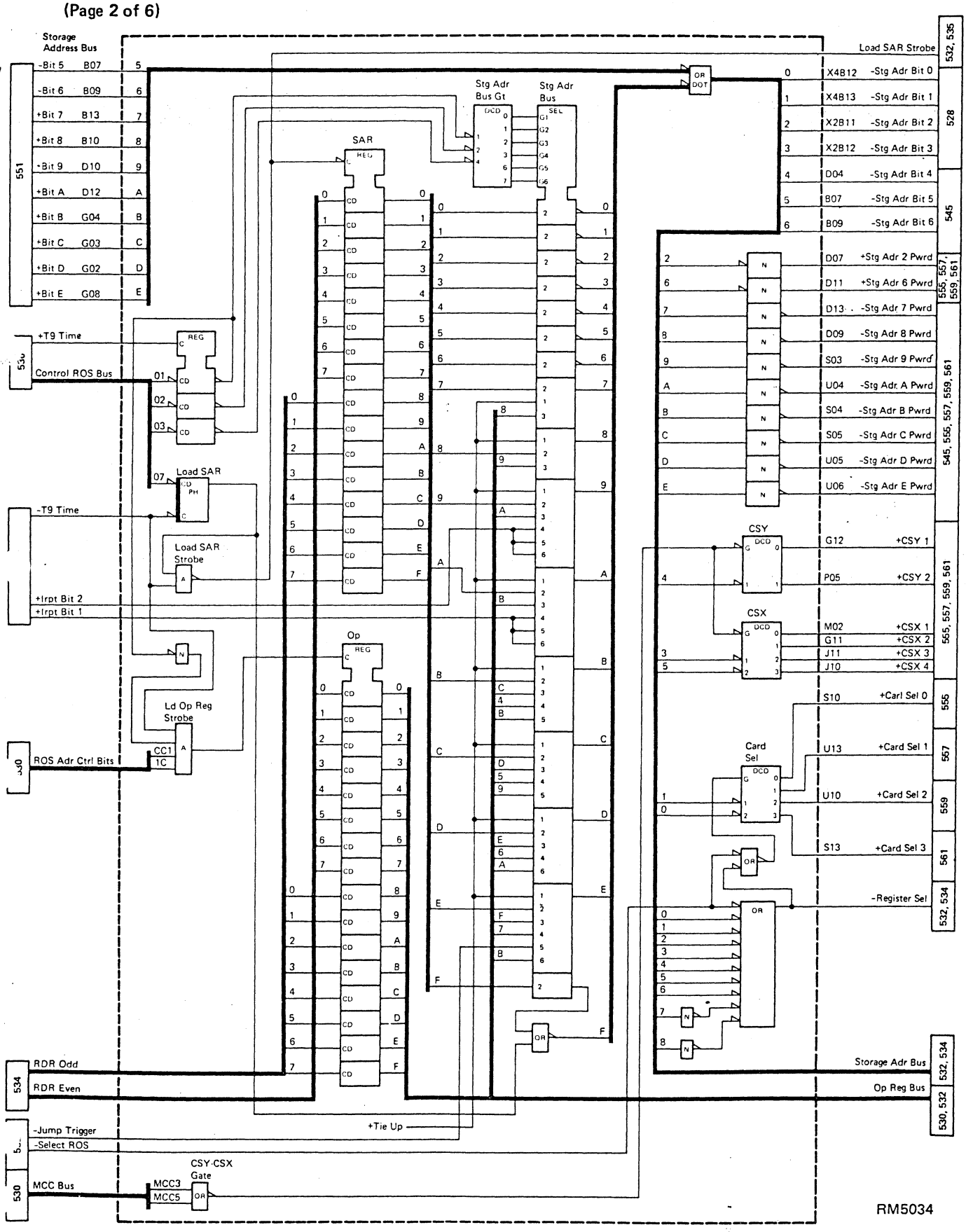

RM5032

Circuits

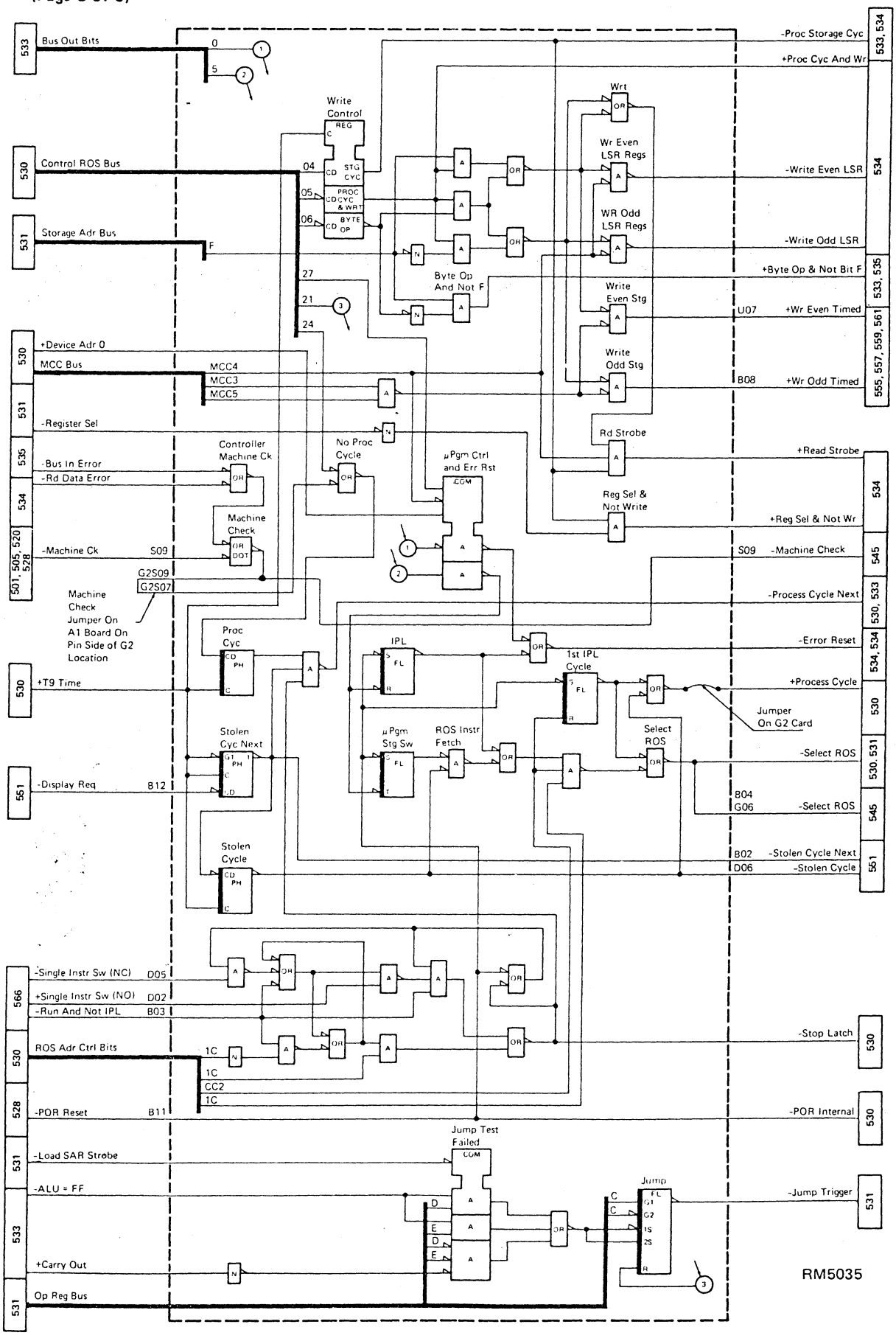
# 530 CONTROLLER CARD G2

(Page 1 of 6)

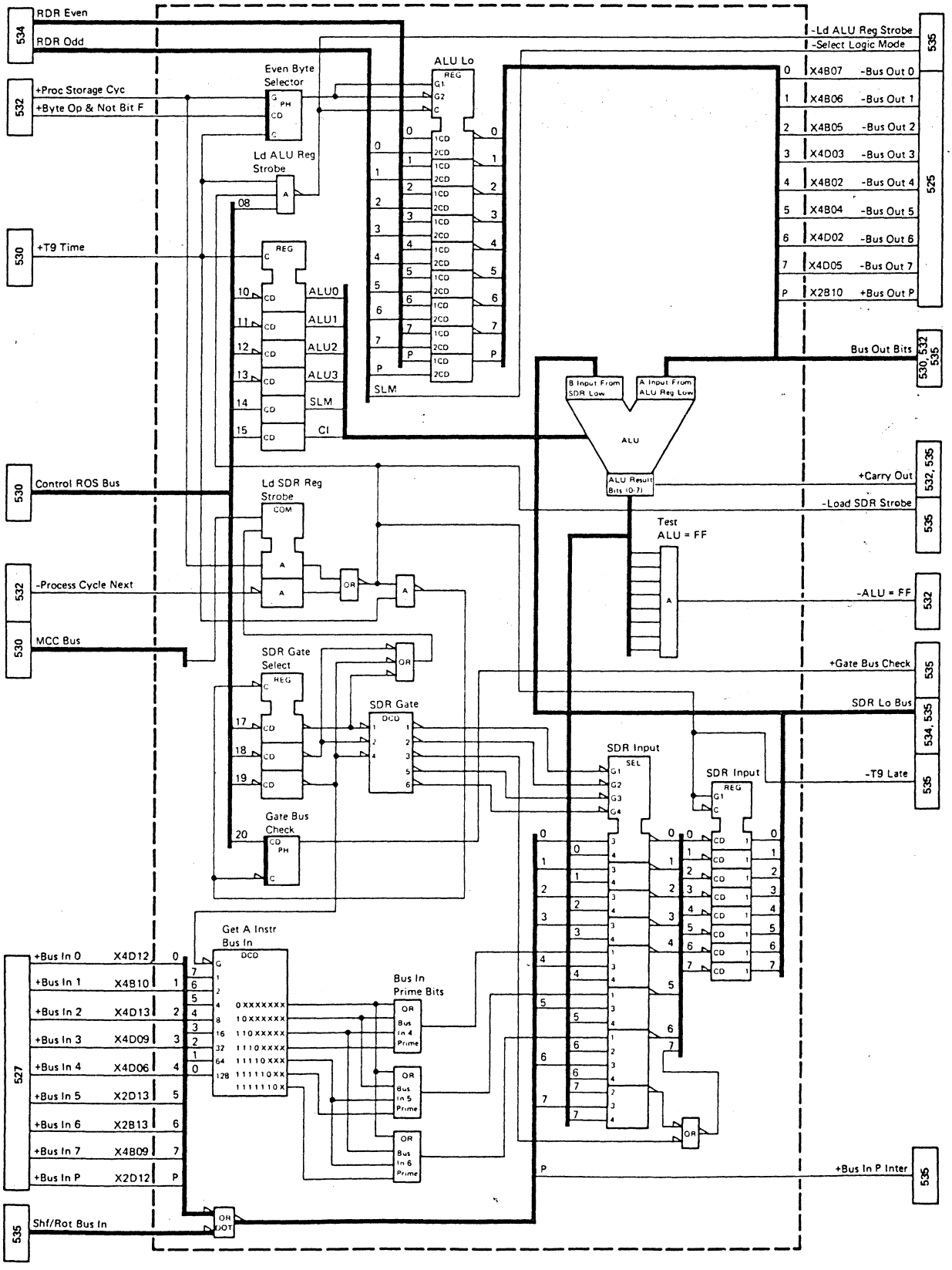



Storage Address Bus
-Bit 5 B07
-Bit 6 B09
+Bit 7 B13
+Bit 8 B10
-Bit 9 D10
+Bit A D12
+Bit B G04
+Bit C G03
+Bit D G02
+Bit E G08
551
+T9 Time
Control ROS Bus
530
-T9 Time
+Irpt Bit 2
+Irpt Bit 1
Load SAR
Load SAR Strobe
Ld Op Reg Strobe
ROS Adr Ctrl Bits
550
SAR
Op
Stg Adr Bus Gt
Stg Adr Bus
OR DOT
Load SAR Strobe
X4B12 -Stg Adr Bit 0
X4B13 -Stg Adr Bit 1
X2B11 -Stg Adr Bit 2
X2B12 -Stg Adr Bit 3
D04 -Stg Adr Bit 4
B07 -Stg Adr Bit 5
B09 -Stg Adr Bit 6
528
545
D07 +Stg Adr 2 Pwrd
D11 +Stg Adr 6 Pwrd
D13 -Stg Adr 7 Pwrd
D09 -Stg Adr 8 Pwrd
S03 -Stg Adr 9 Pwrd
U04 -Stg Adr A Pwrd
S04 -Stg Adr B Pwrd
S05 -Stg Adr C Pwrd
U05 -Stg Adr D Pwrd
U06 -Stg Adr E Pwrd
555, 557, 559, 561
545, 555, 557, 559, 561
CSY
G12 +CSY 1
P05 +CSY 2
CSX
M02 +CSX 1
G11 +CSX 2
J11 +CSX 3
J10 +CSX 4
565, 557, 559, 561
S10 +Carl Sel 0
Card Sel
U13 +Card Sel 1
U10 +Card Sel 2
S13 +Card Sel 3
-Register Sel
555
567
559
561
532, 534
RDR Odd
RDR Even
534
Storage Adr Bus
Op Reg Bus
532, 534
530, 532
+Tie Up
-Jump Trigger
-Select ROS
MCC Bus
530
CSY-CSX Gate
MCC3
MCC5
RM5034

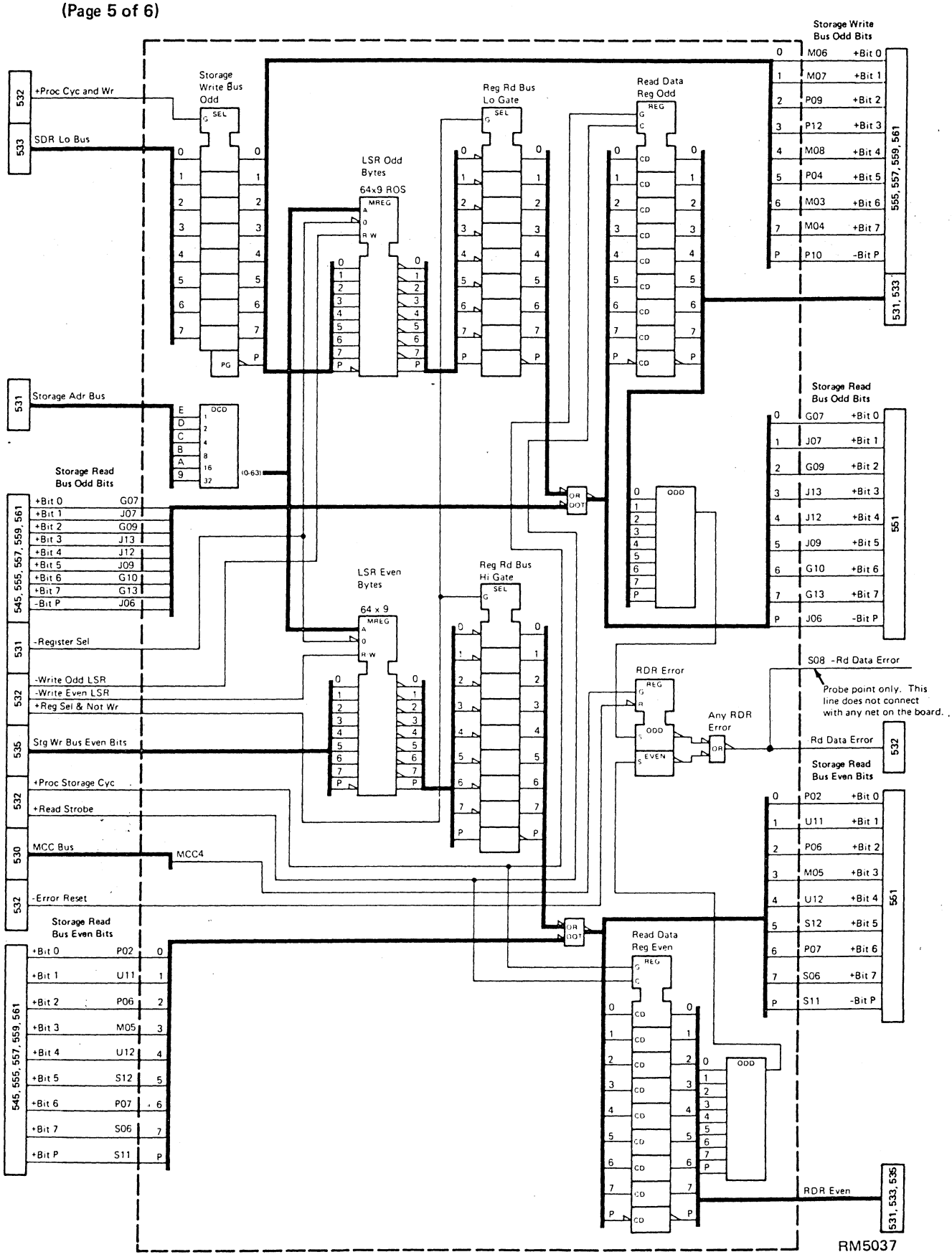
# 532 CONTROLLER CARD G2
(Page 3 of 6)

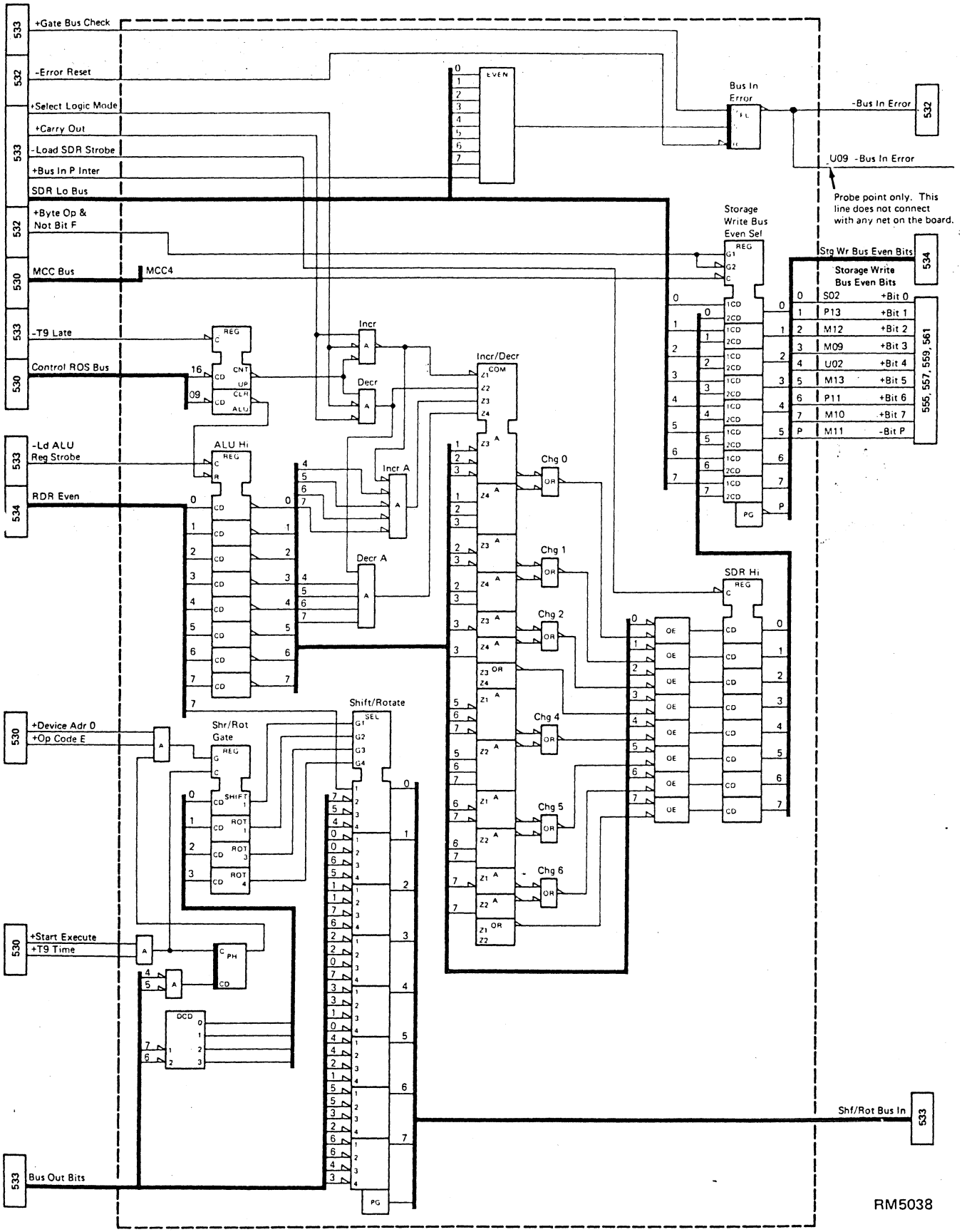


# 533 CONTROLLER CARD G2
(Page 4 of 6)



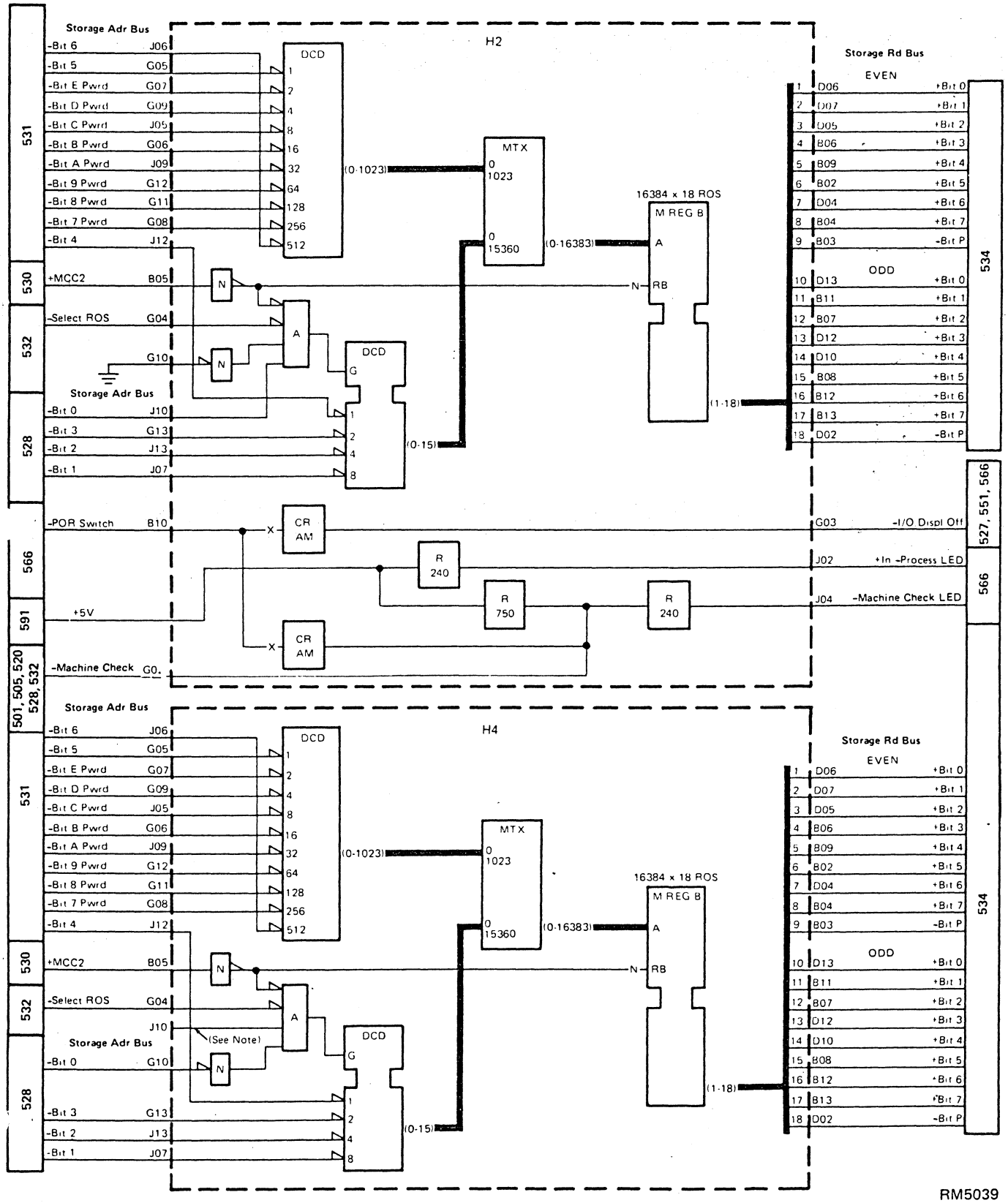
# 534 CONTROLLER CARD G2
(Page 5 of 6)




533 +Gate Bus Check
532 -Error Reset
+Select Logic Mode
+Carry Out
533 -Load SDR Strobe
+Bus In P Inter
SDR Lo Bus
532 +Byte Op & Not Bit F
530 MCC Bus
MCC4
533 -T9 Late
530 Control ROS Bus
533 -Ld ALU Reg Strobe
534 RDR Even
530 +Device Adr 0
+Op Code E
530 +Start Execute
+T9 Time
533 Bus Out Bits
EVEN
Bus In Error
-Bus In Error
532
U09 -Bus In Error
Probe point only. This line does not connect with any net on the board.
Storage Write Bus Even Sel
Stg Wr Bus Even Bits
534
Storage Write Bus Even Bits
0 S02 +Bit 0
1 P13 +Bit 1
2 M12 +Bit 2
3 M09 +Bit 3
4 U02 +Bit 4
5 M13 +Bit 5
6 P11 +Bit 6
7 M10 +Bit 7
P M11 -Bit P
555, 557, 559, 561
Incr
Decr
Incr/Decr
Incr A
Decr A
ALU Hi
Chg 0
Chg 1
Chg 2
Chg 4
Chg 5
Chg 6
SDR Hi
Shift/Rotate
Shr/Rot Gate
Shf/Rot Bus In
533
RM5038

This page intentionally left blank.

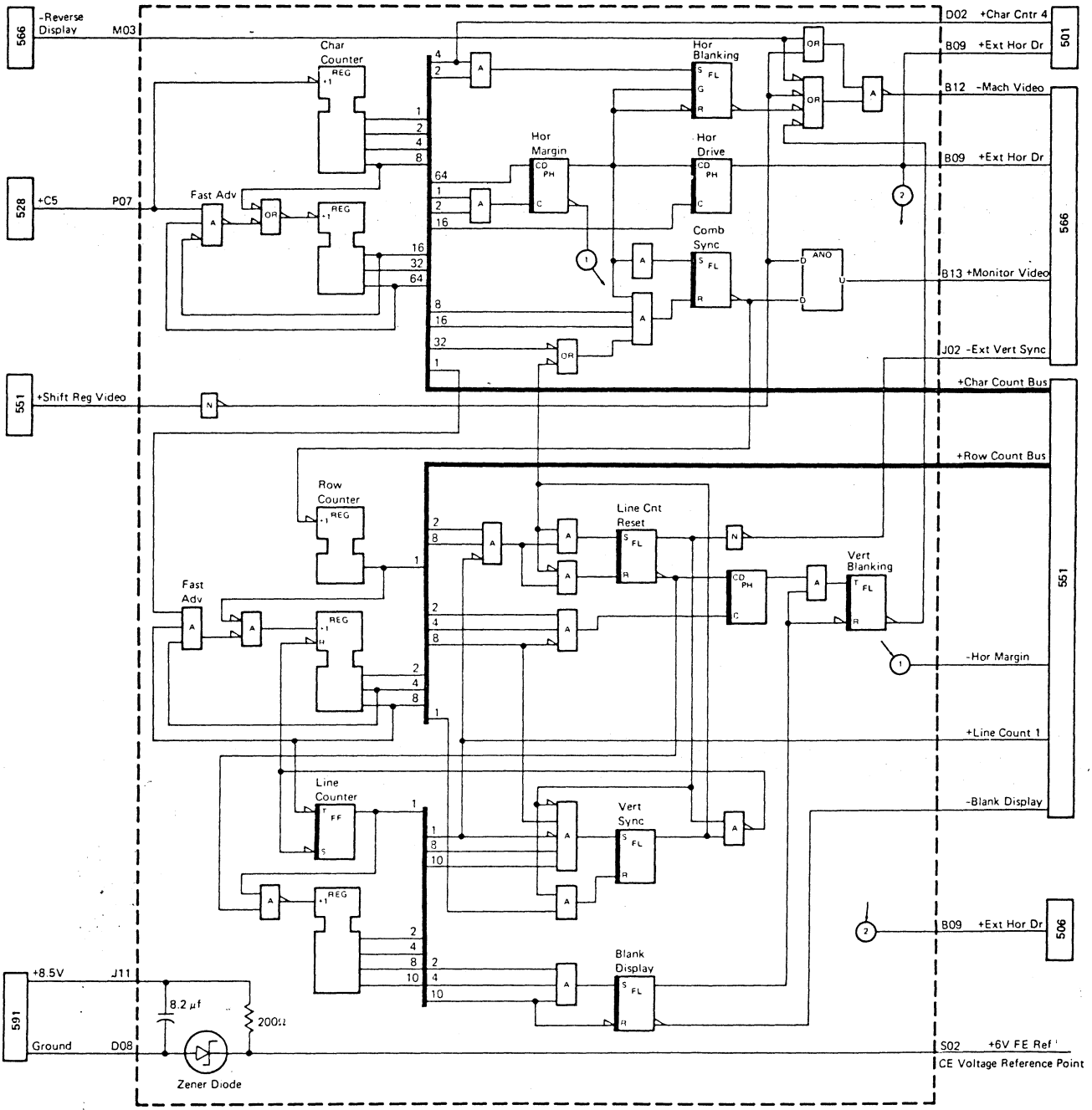
# J45 EXECUTABLE ROS CARDS H2 AND H4




RM5039

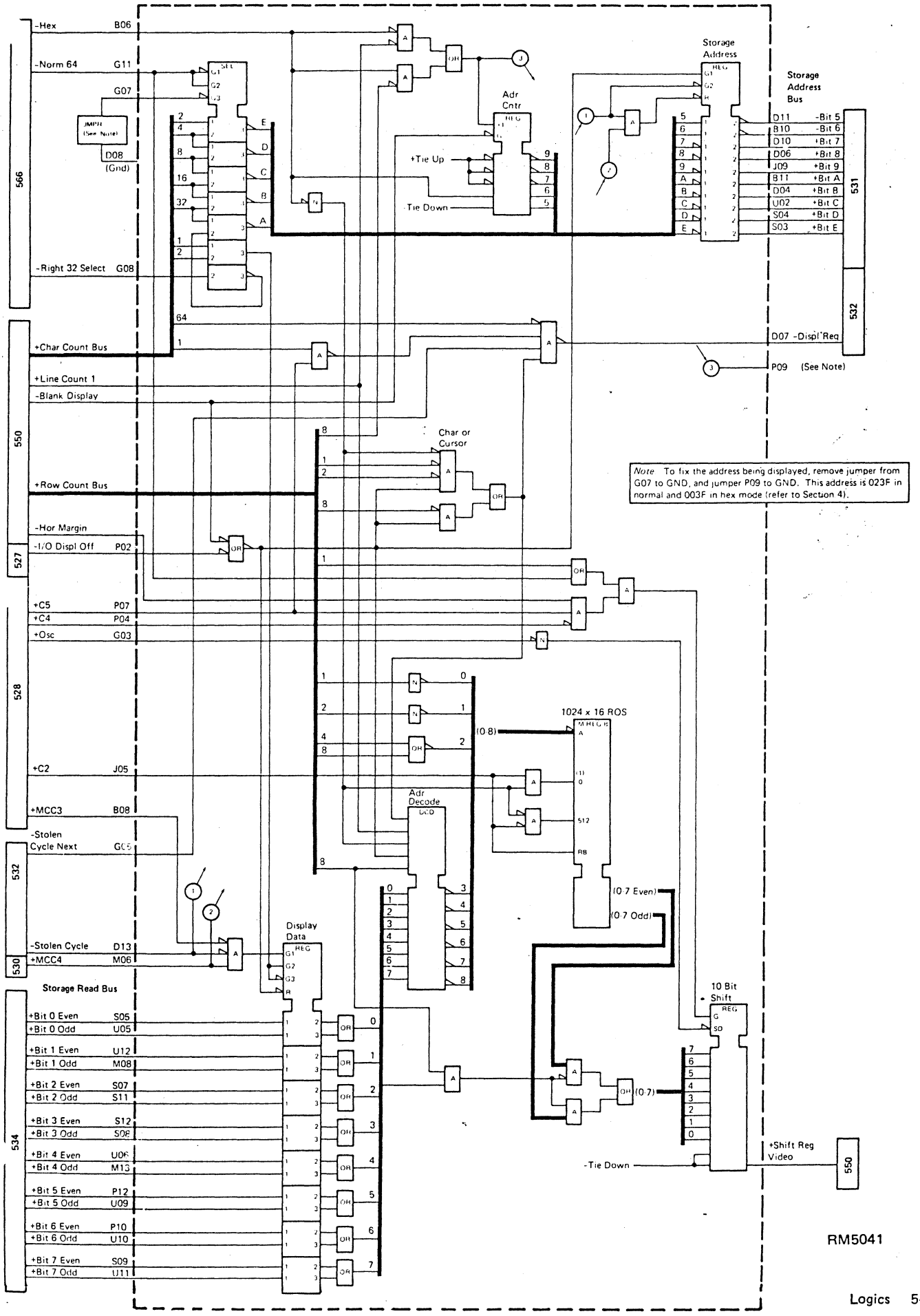
*Note:* Line is "floating" — not wired to anything.

# 550 DISPLAY CARD J2
(Page 1 of 2)




RM5040

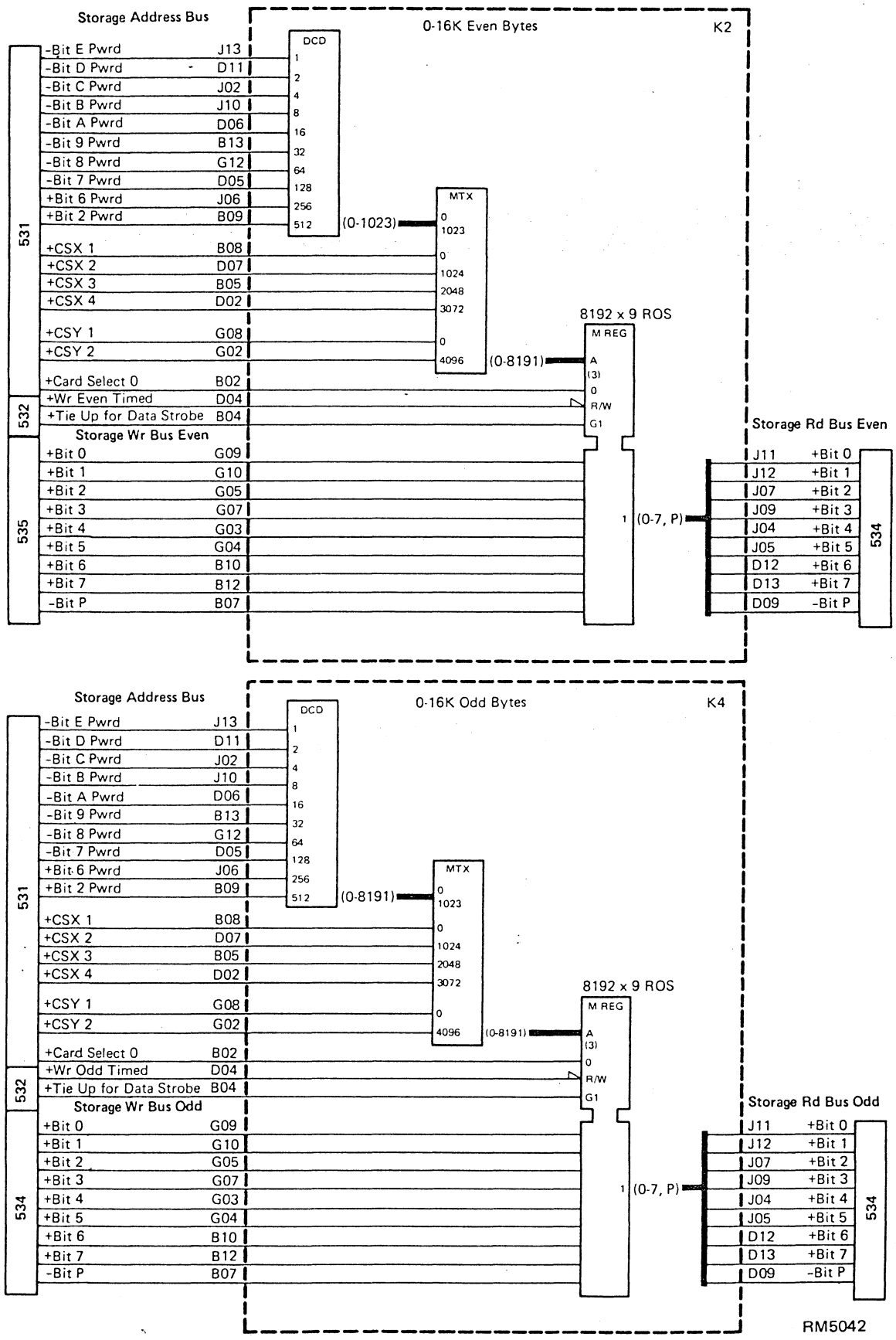
# 551 DISPLAY CARD J2
(Page 2 of 2)



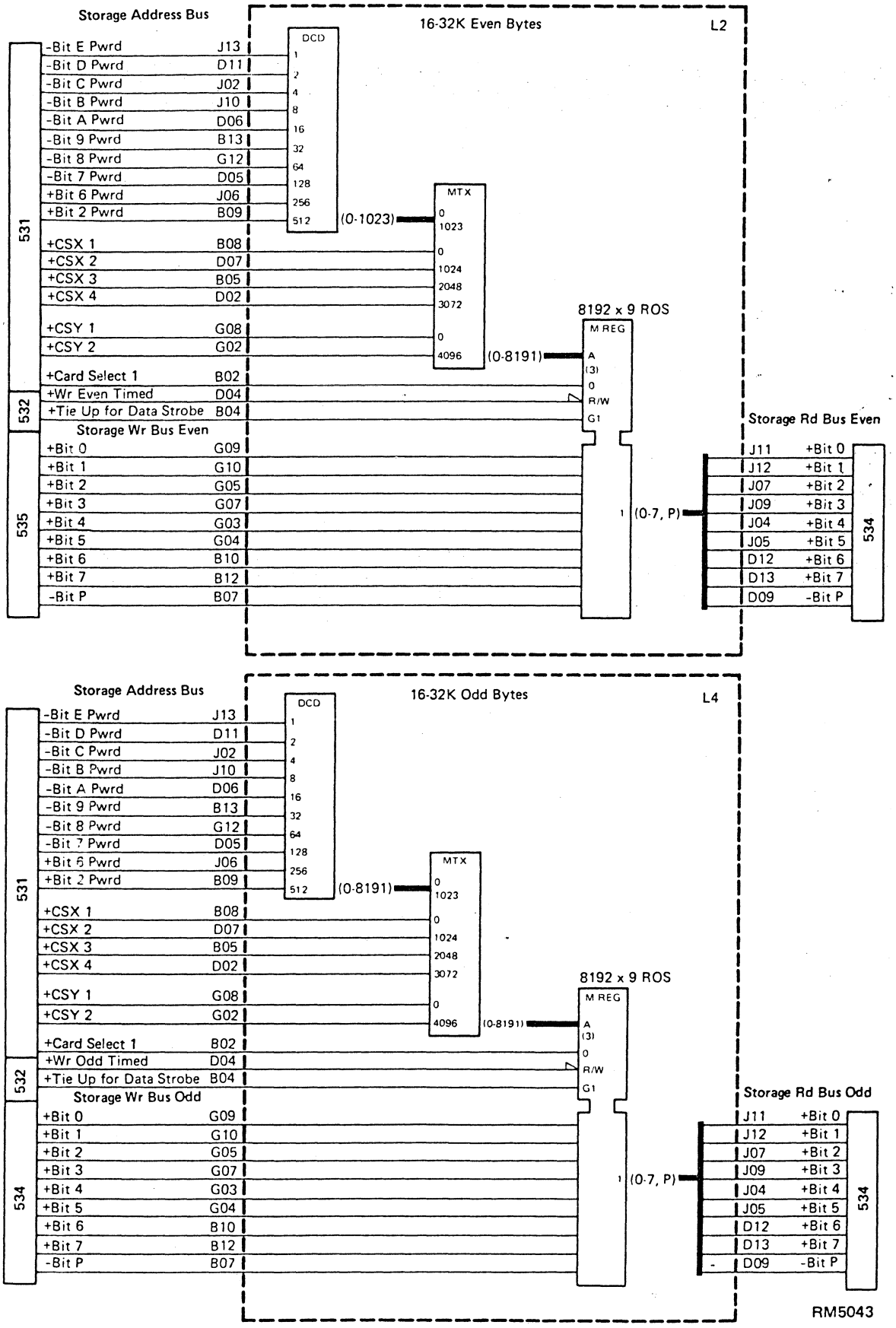

*Note* To fix the address being displayed, remove jumper from G07 to GND, and jumper P09 to GND. This address is 023F in normal and 003F in hex mode (refer to Section 4).

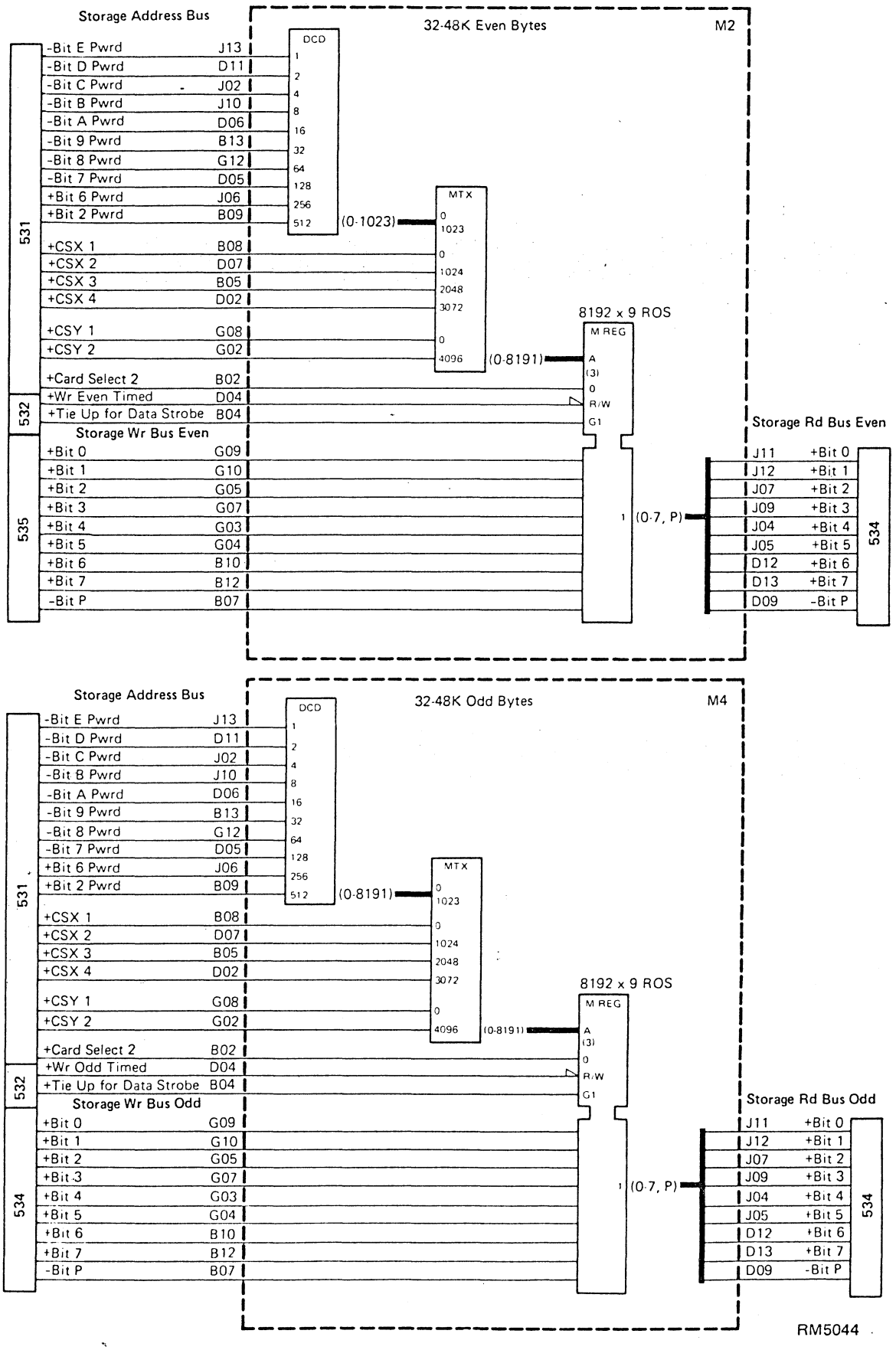
RM5041

# 555 READ/WRITE STORAGE CARDS K2 AND K4

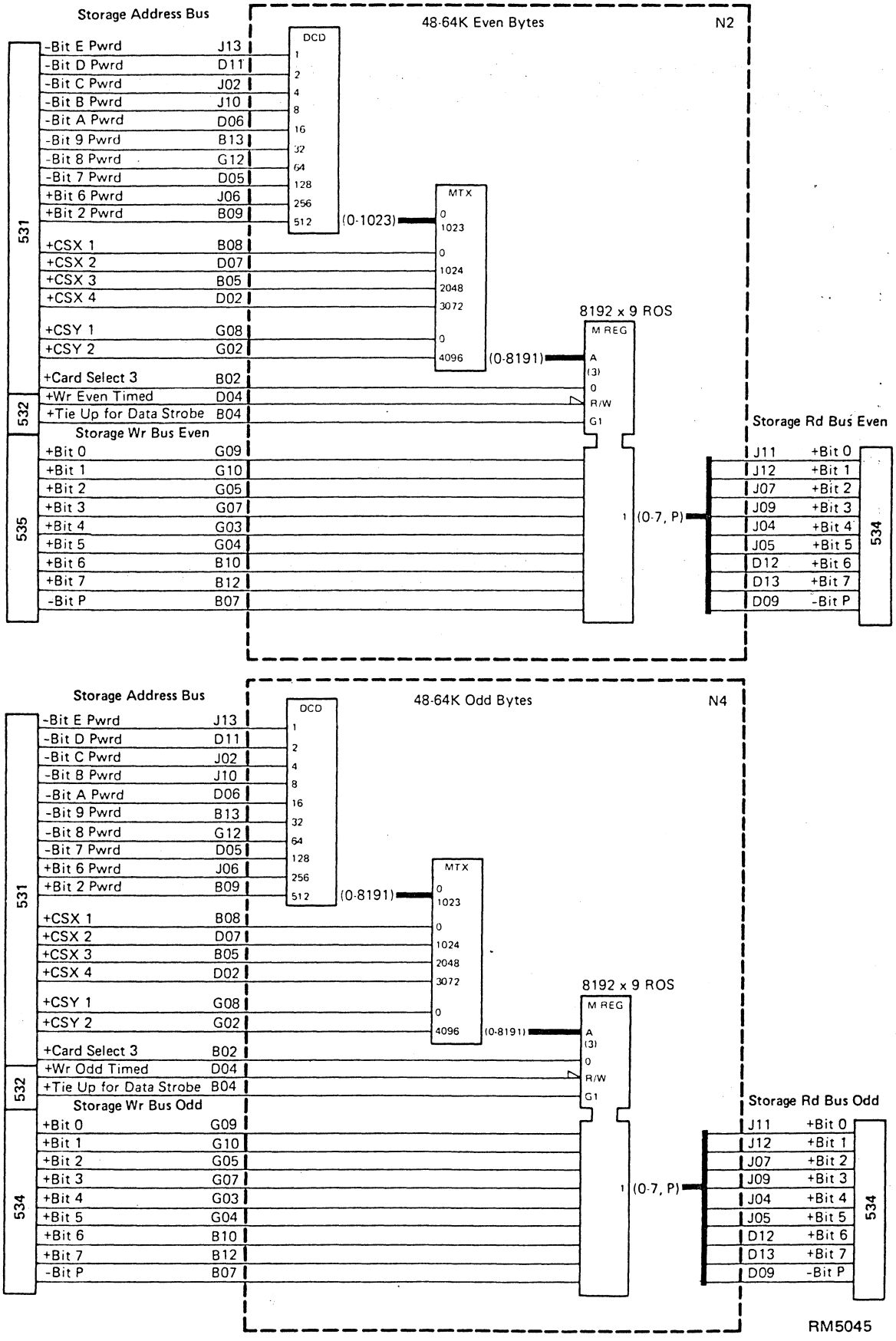


# 557 READ/WRITE STORAGE CARDS L2 AND L4




Storage Address Bus
32-48K Even Bytes
M2
-Bit E Pwrd J13
-Bit D Pwrd D11
-Bit C Pwrd J02
-Bit B Pwrd J10
-Bit A Pwrd D06
-Bit 9 Pwrd B13
-Bit 8 Pwrd G12
-Bit 7 Pwrd D05
+Bit 6 Pwrd J06
+Bit 2 Pwrd B09
531
DCD
1
2
4
8
16
32
64
128
256
512
(0-1023)
MTX
0
1023
0
1024
2048
3072
0
4096
(0-8191)
8192 x 9 ROS
M REG
A
(3)
0
R/W
G1
+CSX 1 B08
+CSX 2 D07
+CSX 3 B05
+CSX 4 D02
+CSY 1 G08
+CSY 2 G02
+Card Select 2 B02
+Wr Even Timed D04
+Tie Up for Data Strobe B04
532
Storage Wr Bus Even
+Bit 0 G09
+Bit 1 G10
+Bit 2 G05
+Bit 3 G07
+Bit 4 G03
+Bit 5 G04
+Bit 6 B10
+Bit 7 B12
-Bit P B07
535
1
(0-7, P)
Storage Rd Bus Even
J11 +Bit 0
J12 +Bit 1
J07 +Bit 2
J09 +Bit 3
J04 +Bit 4
J05 +Bit 5
D12 +Bit 6
D13 +Bit 7
D09 -Bit P
534
Storage Address Bus
32-48K Odd Bytes
M4
-Bit E Pwrd J13
-Bit D Pwrd D11
-Bit C Pwrd J02
-Bit B Pwrd J10
-Bit A Pwrd D06
-Bit 9 Pwrd B13
-Bit 8 Pwrd G12
-Bit 7 Pwrd D05
+Bit 6 Pwrd J06
+Bit 2 Pwrd B09
531
DCD
1
2
4
8
16
32
64
128
256
512
(0-8191)
MTX
0
1023
0
1024
2048
3072
0
4096
(0-8191)
8192 x 9 ROS
M REG
A
(3)
0
R/W
G1
+CSX 1 B08
+CSX 2 D07
+CSX 3 B05
+CSX 4 D02
+CSY 1 G08
+CSY 2 G02
+Card Select 2 B02
+Wr Odd Timed D04
+Tie Up for Data Strobe B04
532
Storage Wr Bus Odd
+Bit 0 G09
+Bit 1 G10
+Bit 2 G05
+Bit 3 G07
+Bit 4 G03
+Bit 5 G04
+Bit 6 B10
+Bit 7 B12
-Bit P B07
534
1
(0-7, P)
Storage Rd Bus Odd
J11 +Bit 0
J12 +Bit 1
J07 +Bit 2
J09 +Bit 3
J04 +Bit 4
J05 +Bit 5
D12 +Bit 6
D13 +Bit 7
D09 -Bit P
534
RM5044

# 561 READ/WRITE STORAGE CARDS N2 AND N4



# 563 Y1 INPUT POWER CABLE

## Y1 Connector To TSR DC Power Supply

590 DC Voltage Distribution

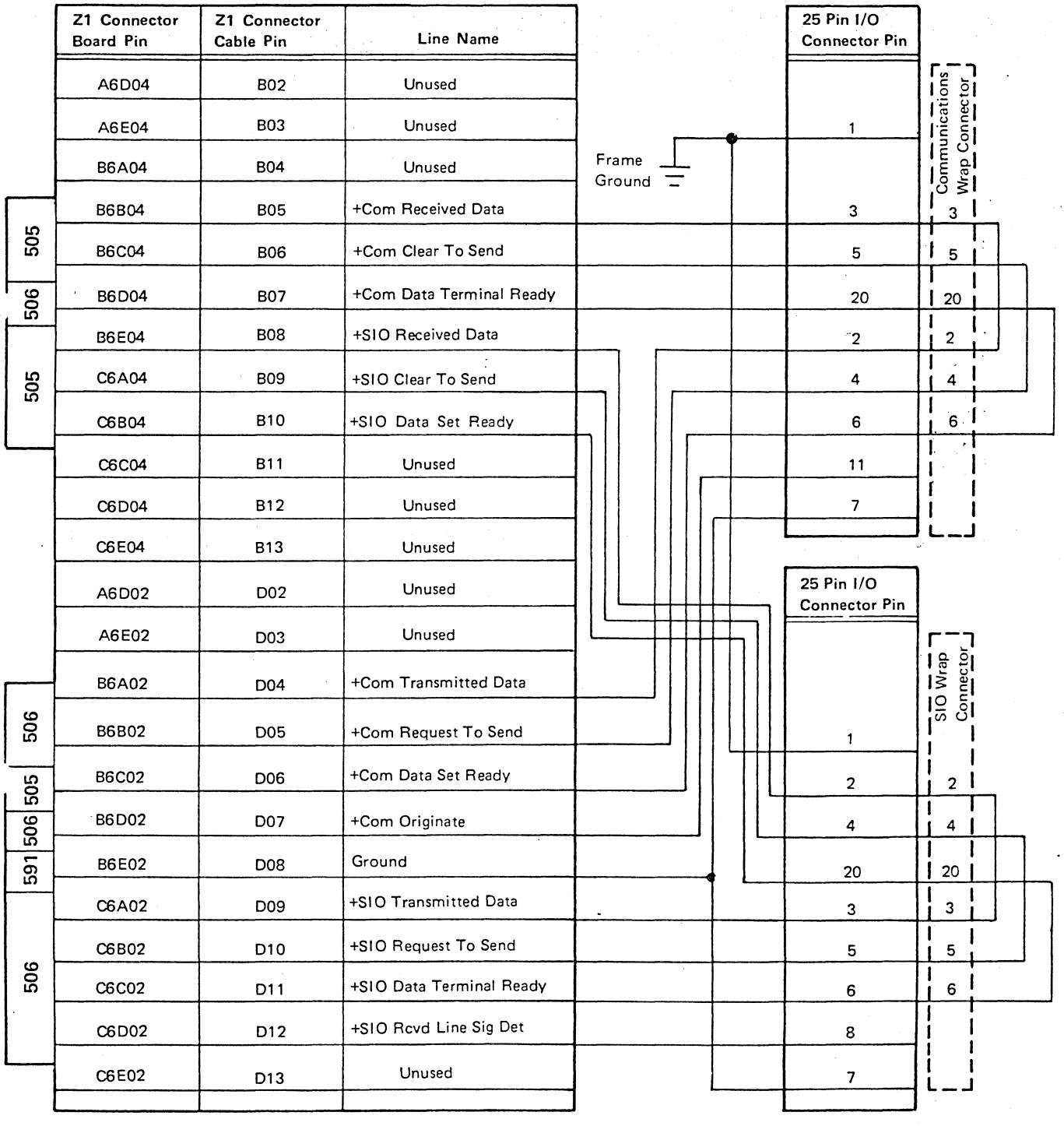
| Line Name | Y1 Connector Cable Pin | Y1 Connector Board Pin |
|---|---|---|
| +5V | B02 | A1D13 |
| +5V | B03 | A1E13 |
| +5V | B04 | B1A13 |
| +5V | B05 | B1B13 |
| Ground | B06 | B1C13 |
| Ground | B07 | B1D13 |
| Ground | B08 | B1E13 |
| Ground | B09 | C1A13 |
| Unused | B10 | C1B13 |
| +8.5V | B11 | C1C13 |
| +12V | B12 | C1D13 |
| -12V | B13 | C1E13 |
| +5V | D02 | A1D11 |
| +5V | D03 | A1E11 |
| +5V | D04 | B1A11 |
| +5V | D05 | B1B11 |
| Ground | D06 | B1C11 |
| Ground | D07 | B1D11 |
| Ground | D08 | B1E11 |
| Ground | D09 | C1A11 |
| Ground | D10 | C1B11 |
| +8.5V | D11 | C1C11 |
| +12V | D12 | C1D11 |
| -5V | D13 | C1E11 |

591 A1 Board DC Power Distribution

## 64 DATA SET CABLE

### Z1 Connector To Communications and Serial I/O Ports

| | Z1 Connector Board Pin | Z1 Connector Cable Pin | Line Name |
|---|---|---|---|
| | A6D04 | B02 | Unused |
| | A6E04 | B03 | Unused |
| | B6A04 | B04 | Unused |
| 505 | B6B04 | B05 | +Com Received Data |
| | B6C04 | B06 | +Com Clear To Send |
| 506 | B6D04 | B07 | +Com Data Terminal Ready |
| | B6E04 | B08 | +SIO Received Data |
| 505 | C6A04 | B09 | +SIO Clear To Send |
| | C6B04 | B10 | +SIO Data Set Ready |
| | C6C04 | B11 | Unused |
| | C6D04 | B12 | Unused |
| | C6E04 | B13 | Unused |
| | A6D02 | D02 | Unused |
| | A6E02 | D03 | Unused |
| | B6A02 | D04 | +Com Transmitted Data |
| 506 | B6B02 | D05 | +Com Request To Send |
| | B6C02 | D06 | +Com Data Set Ready |
| 505 | B6D02 | D07 | +Com Originate |
| 506 | B6E02 | D08 | Ground |
| 591 | C6A02 | D09 | +SIO Transmitted Data |
| | C6B02 | D10 | +SIO Request To Send |
| 506 | C6C02 | D11 | +SIO Data Terminal Ready |
| | C6D02 | D12 | +SIO Rcvd Line Sig Det |
| | C6E02 | D13 | Unused |




*Note:* Wrap connectors used for testing are indicated by the dashed lines and jumpered pins.

Circuits

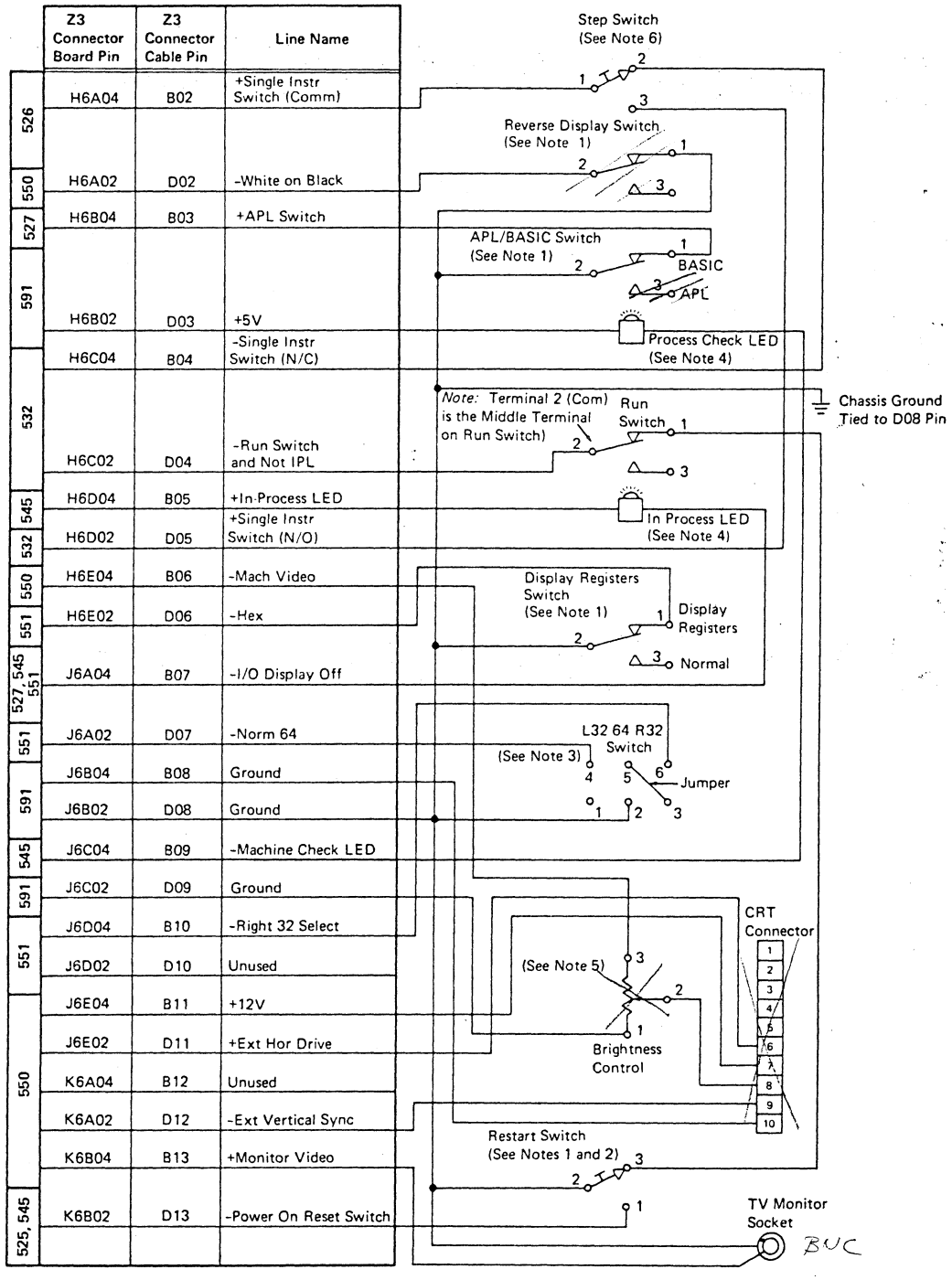
# 565 TAPE INTERFACE CABLE

## Z2 Connector To Tape Unit

| | Z2 Connector Board Pin | Z2 Connector Cable Pin | Line Name | Tape Drive Card Pin | |
|---|---|---|---|---|---|
| 525 | D6E04 | B02 | +Tape Clock | U02 | 570 |
| 591 | E6A04 | B03 | +5V | U03 | |
| 525 | E6B04 | B04 | -Forward | U04 | |
| | E6C04 | B05 | -Run | U05 | |
| | E6D04 | B06 | -Write Enable | U06 | |
| 526 | E6E04 | B07 | -Write Data | U07 | |
| 591 | F6A04 | B08 | Ground | U08 | |
| 525 | F6B04 | B09 | -Channel Select | U09 | |
| | F6C04 | B10 | -Channel 0 Erase | U10 | |
| | F6D04 | B11 | -Channel 1 Erase | U11 | |
| 526 | F6E04 | B12 | -Select Magnet Active | U12 | |
| | G6A04 | B13 | +File Protect | U13 | |
| 525 | D6E02 | D02 | -EOT | S02 | |
| 526 | E6A02 | D03 | -Erase Inactive | S03 | |
| | E6B02 | D04 | -BOT | S04 | |
| 525 | E6C02 | D05 | -Diagnostic Mode | S05 | |
| 591 | E6D02 | D06 | -5V | S06 | |
| 526 | E6E02 | D07 | -LED and Erase OK | S07 | |
| | F6A02 | D08 | Ground | S08 | |
| | F6B02 | D09 | -Read Data | S09 | |
| | F6C02 | D10 | -Read Clock | S10 | |
| | F6D02 | D11 | +12V | S11 | |
| | F6E02 | D12 | -Cartridge in Place | S12 | |
| | G6A02 | D13 | -12V | S13 | |

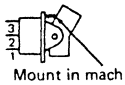
## Z3 Connector to CRT Connector and Control Panel Switches

| | Z3 Connector Board Pin | Z3 Connector Cable Pin | Line Name |
|---|---|---|---|
| 526 | H6A04 | B02 | +Single Instr Switch (Comm) |
| 550 | H6A02 | D02 | -White on Black |
| 527 | H6B04 | B03 | +APL Switch |
| 591 | H6B02 | D03 | +5V |
| | H6C04 | B04 | -Single Instr Switch (N/C) |
| 532 | H6C02 | D04 | -Run Switch and Not IPL |
| 545 | H6D04 | B05 | +In-Process LED |
| 532 | H6D02 | D05 | +Single Instr Switch (N/O) |
| 550 | H6E04 | B06 | -Mach Video |
| 551 | H6E02 | D06 | -Hex |
| 527, 545, 551 | J6A04 | B07 | -I/O Display Off |
| 551 | J6A02 | D07 | -Norm 64 |
| | J6B04 | B08 | Ground |
| 591 | J6B02 | D08 | Ground |
| 545 | J6C04 | B09 | -Machine Check LED |
| 591 | J6C02 | D09 | Ground |
| | J6D04 | B10 | -Right 32 Select |
| 551 | J6D02 | D10 | Unused |
| | J6E04 | B11 | +12V |
| | J6E02 | D11 | +Ext Hor Drive |
| 550 | K6A04 | B12 | Unused |
| | K6A02 | D12 | -Ext Vertical Sync |
| | K6B04 | B13 | +Monitor Video |
| 525, 545 | K6B02 | D13 | -Power On Reset Switch |

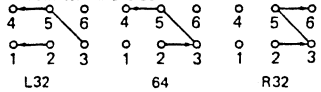



*Notes:*

1. Top of switch when in machine.



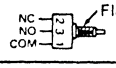
2. Mount in machine with spring up.
3. Switch contact positions. View from terminal side:




4. White dot or short lead to minus pin.


5. Brightness control connections.


6. S/I switch connections.




RM5010

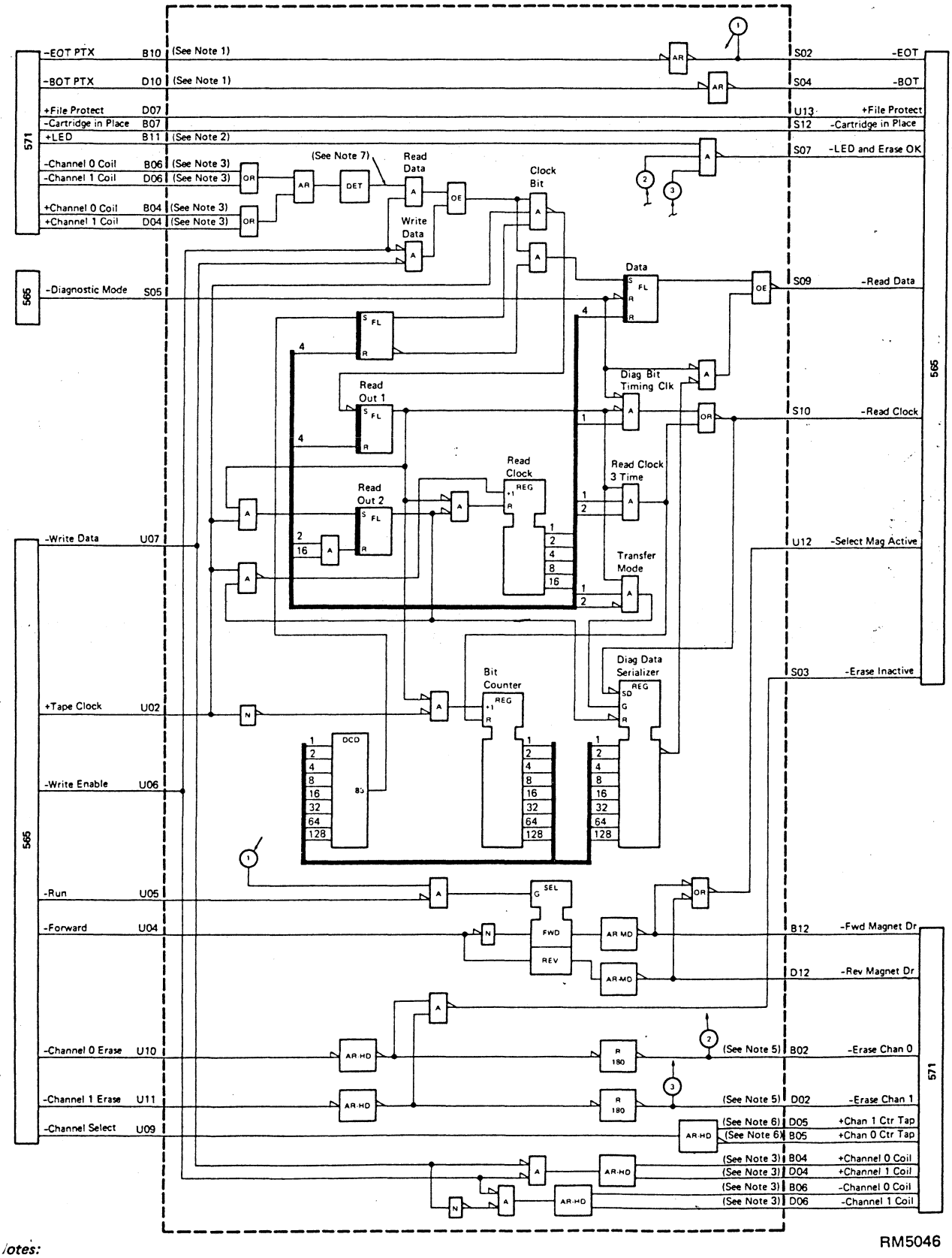
# 567 KEYBOARD CABLE

D08 = GND

## Z4 Connector To Keyboard PC Board Connector

| | Z4 Connector Board Pin | Z4 Connector Cable Pin | Line Name | Keyboard PC Board Connector Pin | |
|---|---|---|---|---|---|
| 527 | L6B04 | B02 | Unused | B02 | 575 |
| | L6C04 | B03 | Unused | B03 | |
| | L6D04 | B04 | -Keyboard Bit P | B04 | |
| | L6E04 | B05 | -Keyboard Bit 7 | B05 | |
| | M6A04 | B06 | Unused | B06 | |
| | M6B04 | B07 | -Keyboard Strobe | B07 | |
| | M6C04 | B08 | -Keyboard Bit 4 | B08 | |
| | M6D04 | B09 | -Keyboard Bit 3 | B09 | |
| | M6E04 | B10 | -Keyboard Bit 2 | B10 | |
| 591 | N6A04 | B11 | +8.5V | B11 | |
| 527 | N6B04 | B12 | -Keyboard Bit 0 | B12 | |
| | N6C04 | B13 | -Keyboard Bit 1 | B13 | |
| | L6B02 | D02 | Unused | D02 | |
| 591 | L6C02 | D03 | +5V | D03 | |
| 527 | L6D02 | D04 | Unused | D04 | |
| | L6E02 | D05 | Unused | D05 | |
| | M6A02 | D06 | -Keyboard Bit 6 | D06 | |
| | M6B02 | D07 | -Power On Reset | D07 | |
| 591 | M6C02 | D08 | Ground | D08 | |
| 527 | M6D02 | D09 | +Typamatic | D09 | |
| | M6E02 | D10 | Unused | D10 | |
| | N6A02 | D11 | -Keyboard Lock | D11 | |
| | N6B02 | D12 | Unused | D12 | |
| | N6C02 | D13 | -Keyboard Bit 5 | D13 | |

# 570 TAPE CONTROL CARD

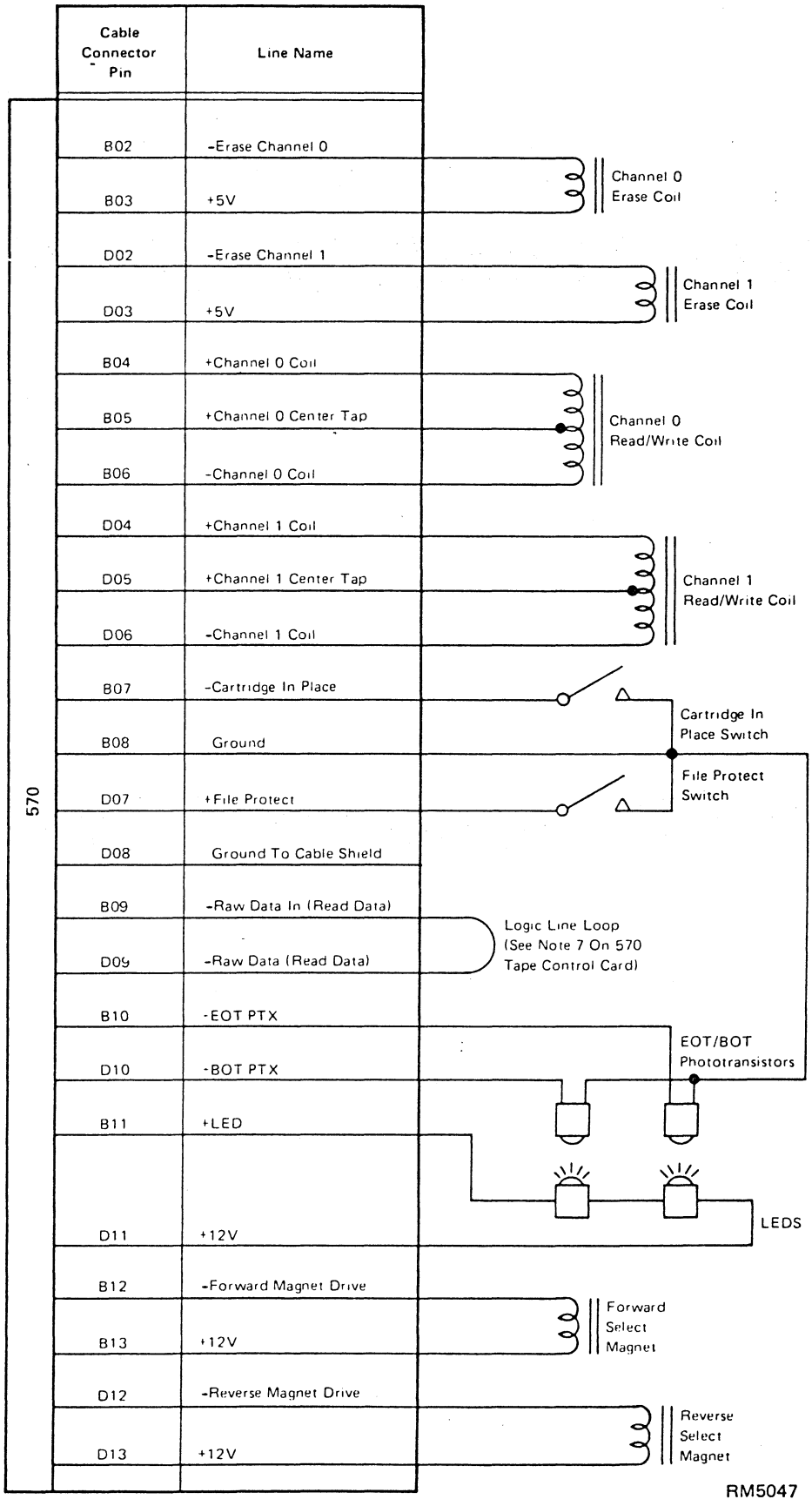



RM5046

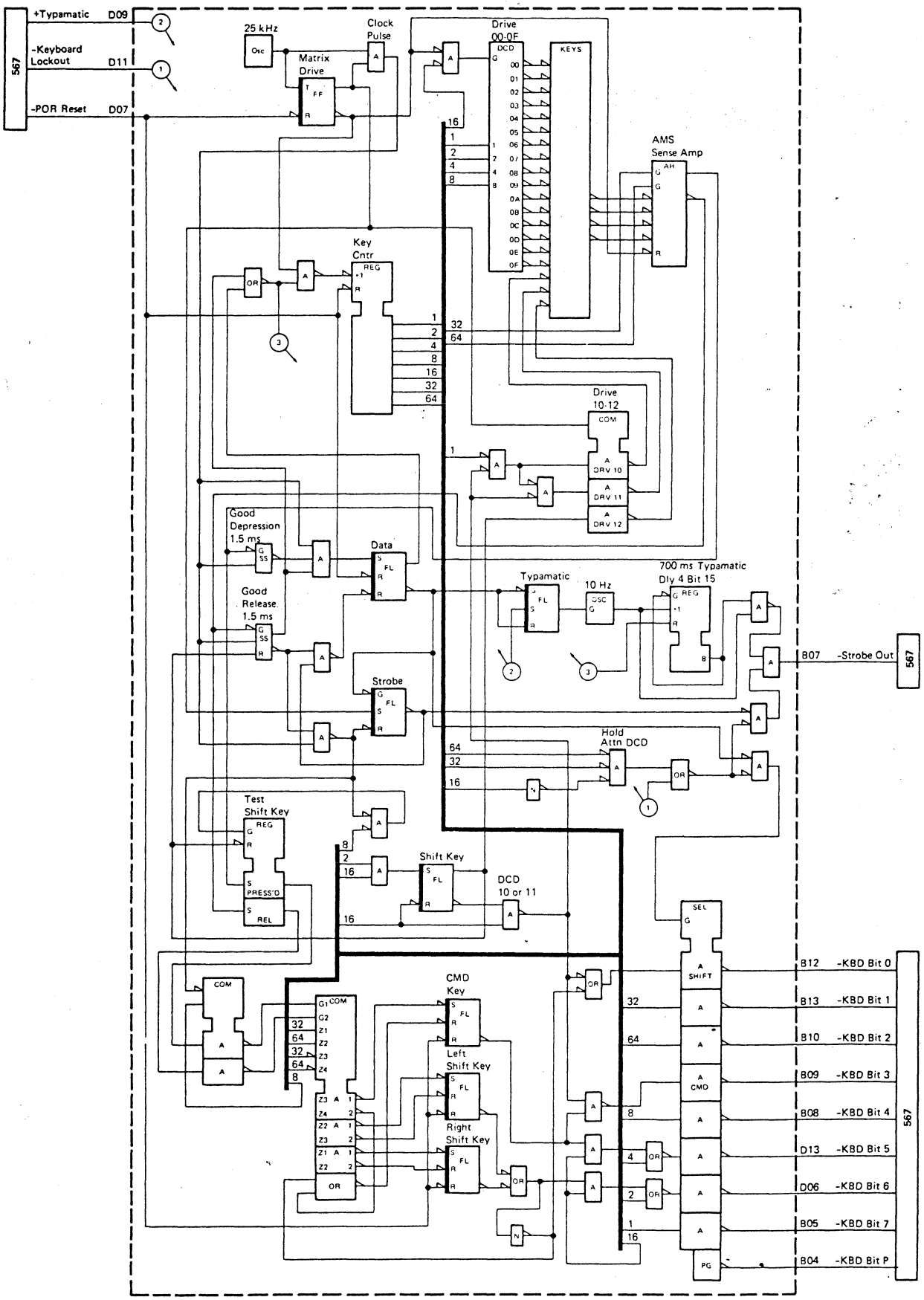
*Notes:*

1. Up +1.2V, down +0.6V to -0.7V.
2. Normally approximately +9.5V.
3. 20V P-P voltage centered about a +1V reference level during write operations. A 20 mV P-P voltage around a 0 volt reference level exists during read operations.
4. Up +12V, down 0V.
5. Up +5V, down +4.3V.
6. Up +12V (write operation), down 0V (read operation).
7. Logic line at this point exits card at D09 and reenters at B09 via a jumper on the tape internal cable.

# 571 TAPE INTERNAL CABLE

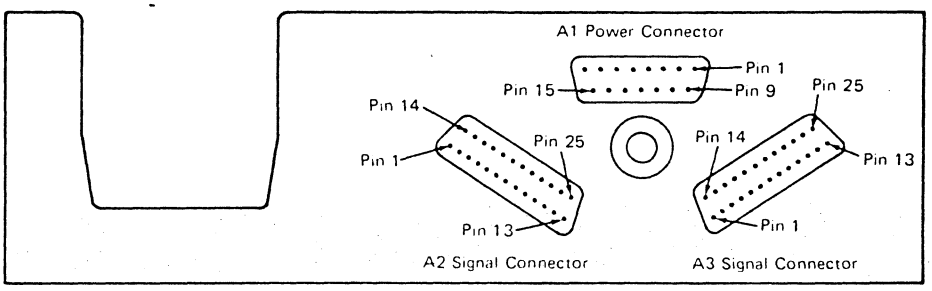


# 5 KEYBOARD PC BOARD

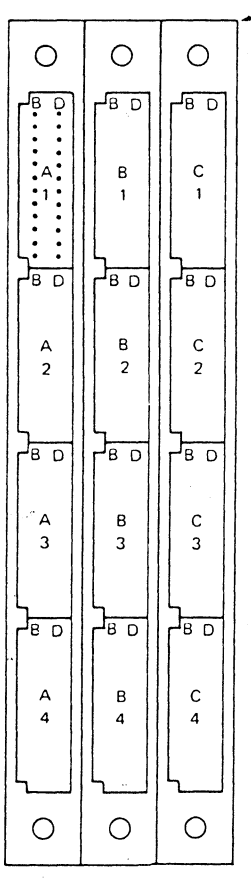



RM5048

External I/O Interface Port (On Rear Of Base Machine)



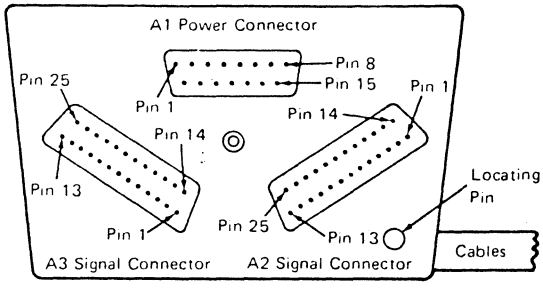

Auxiliary Tape Unit A1 Board
(Plug Side)



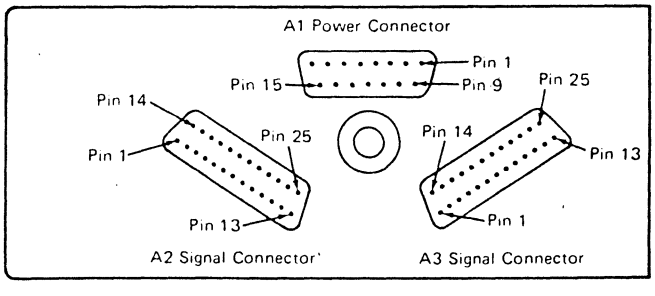

Auxiliary Tape A1 Board
Socket List

| Socket Location | Cable Or Card |
|---|---|
| A1 | A2 Signal Cable Entry Point |
| A2 | A3 Signal Cable Entry Point |
| A3 | A1 Power Cable Exit Point |
| A4 | A1 Power Cable Entry Point |
| B1 | A2 Signal Cable Exit Point |
| B2 | A3 Signal Cable Exit Point |
| B3 | Unused |
| B4 | Tape Drive Cable |
| C1-C4 | Auxiliary Tape Adapter Card |

I/O Cable Assembly Connector




Interface Connector (On Rear of Tape Unit)




RM5020

## A2 Signal Connector and A1 Board Pin Locations

| | External I/O Interface Connector Pin | Line Name | A2 Cable Entry Pin | Auxiliary Tape Adapter Card Pin | | A2 Cable Exit Pin (To Cable Connector On Rear Of Unit) | Interface Connector Pin On Rear Of Tape Unit | |
|---|---|---|---|---|---|---|---|---|
| 501 | 01 | -Ground | A1D08 | C1D08 | 581, 582 | B1D08 | 01 | Printer I/O Cable Assembly |
| | 02 | -Put Strobe | A1B13 | C1B13 | | B1B13 | 02 | |
| | 03 | -Control Strobe | A1B12 | C1B12 | | B1B12 | 03 | |
| | 04 | -Get Strobe | A1B11 | C1B11 | | B1B11 | 04 | |
| | 05 | +Device Adr Y3 | A1B10 | C1B10 | | B1B10 | 05 | |
| | 06 | +Device Adr Y2 | A1B09 | C1B09 | | B1B09 | 06 | |
| | 07 | +Device Adr Y1 | A1B08 | C1B08 | | B1B08 | 07 | |
| | 08 | +Device Adr Y0 | A1B07 | C1B07 | | B1B07 | 08 | |
| | 09 | +Device Adr X3 | A1B06 | C1B06 | | B1B06 | 09 | |
| | 10 | +Device Adr X2 | A1B05 | C1B05 | | B1B05 | 10 | |
| | 11 | +Device Adr X1 | A1B04 | C1B04 | | B1B04 | 11 | |
| | 12 | +Device Adr X0 | A1B03 | C1B03 | | B1B03 | 12 | |
| | 13 | -Ground | A1D08 | C1D08 | | B1D08 | 13 | |
| | 14 | -Ground | A1D08 | C1D08 | | B1D08 | 14 | |
| | 15 | +Op Code E | A1D13 | C1D13 | | B1D13 | 15 | |
| | 16 | +Bus In P | A1D12 | C1D12 | 582 | B1D12 | 16 | |
| | 17 | +Bus In 7 | A1D11 | C1D11 | | B1D11 | 17 | |
| | 18 | +Bus In 6 | A1D10 | C1D10 | | B1D10 | 18 | |
| | 19 | +Bus In 5 | A1D09 | C1D09 | | B1D09 | 19 | |
| | 20 | +Bus In 4 | A1D07 | C1D07 | | B1D07 | 20 | |
| | 21 | +Bus In 3 | A1D06 | C1D06 | | B1D06 | 21 | |
| | 22 | +Bus In 2 | A1D05 | C1D05 | | B1D05 | 22 | |
| | 23 | +Bus In 1 | A1D04 | C1D04 | | B1D04 | 23 | |
| | 24 | Unused | | | | | | |
| | 25 | +Bus In 0 | A1D02 | C1D02 | | B1D02 | 25 | |

## A3 Signal Connector and A1 Board Pin Locations

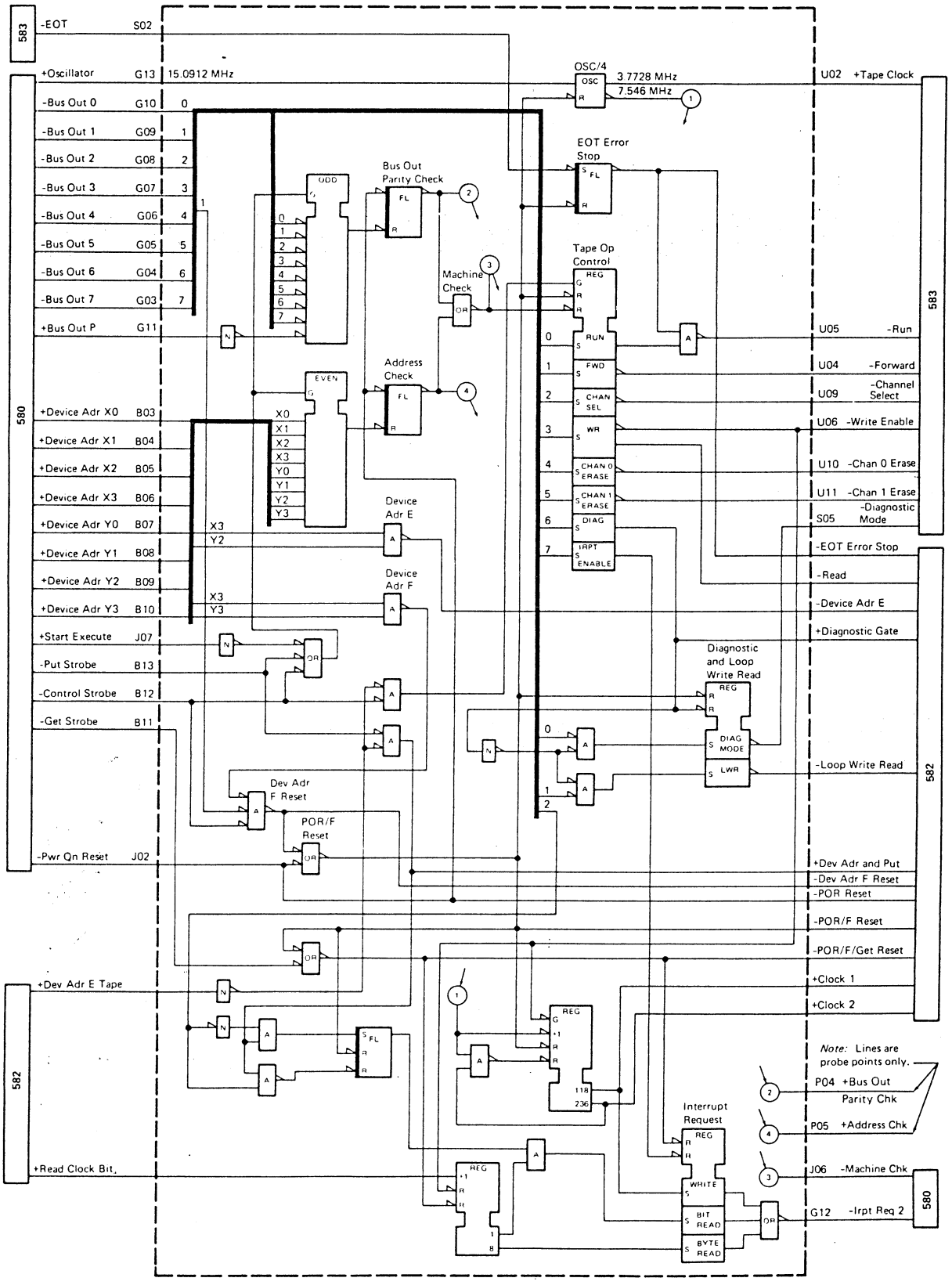
| | External I/O Interface Connector Pin | Line Name | A3 Cable Entry Pin | Auxiliary Tape Adapter Card Pin | | A3 Cable Out Pin (To Cable Connector On Rear of Unit) | Interface Connector Pin On Rear Of Tape Unit | |
|---|---|---|---|---|---|---|---|---|
| 501 | 01 | -Ground | A2D08 | C1D08 | 581, 582 | B2D08 | 01 | Printer I/O Cable Assembly |
| | 02 | +Oscillator | A2B13 | C1G13 | | B2B13 | 02 | |
| | 03 | -Interrupt Req 2 | A2B12 | C1G12 | | B2B12 | 03 | |
| | 04 | +Bus Out Bit P | A2B11 | C1G11 | | B2B11 | 04 | |
| | 05 | -Bus Out Bit 0 | A2B10 | C1G10 | | B2B10 | 05 | |
| | 06 | -Bus Out Bit 1 | A2B09 | C1G09 | | B2B09 | 06 | |
| | 07 | -Bus Out Bit 2 | A2B08 | C1G08 | | B2B08 | 07 | |
| | 08 | -Bus Out Bit 3 | A2B07 | C1G07 | | B2B07 | 08 | |
| | 09 | -Bus Out Bit 4 | A2B06 | C1G06 | | B2B06 | 09 | |
| | 10 | -Bus Out Bit 5 | A2B05 | C1G05 | | B2B05 | 10 | |
| | 11 | -Bus Out Bit 6 | A2B04 | C1G04 | | B2B04 | 11 | |
| | 12 | -Bus Out Bit 7 | A2B03 | C1G03 | | B2B03 | 12 | |
| | 13 | -Ground | A2D08 | C1D08 | | B2D08 | 13 | |
| | 14 | -Ground | A2D08 | | | B2D08 | 14 | |
| | 15 | +C5 | A2D13 | | | B2D13 | 15 | |
| | 16 | +C4 | A2D12 | | | B2D12 | 16 | |
| | 17 | +C3 | A2D11 | | | B2D11 | 17 | |
| | 18 | +C2 | A2D10 | | | B2D10 | 18 | |
| | 19 | +C1 | A2D09 | | | B2D09 | 19 | |
| | 20 | +Start Execute Bit | A2D07 | C1J07 | 581, 582 | B2D07 | 20 | |
| | 21 | -Machine Check | A2D06 | C1J06 | | B2D06 | 21 | |
| | 22 | +Ext Horiz Drive | A2D05 | | | B2D05 | 22 | |
| | 23 | +Char Cntr 4 | A2D04 | | | B2D04 | 23 | |
| | 24 | Unused | | | | | | |
| | 25 | -Power On Reset | A2D02 | C1J02 | | B2D02 | 25 | |

## A1 Power Connector and A1 Board Pin Locations

| | External I/O Interface Connector Pin | Line Name | Power Cable Entry Pin | Auxiliary Tape Adapter Card Pin(s) | Tape Unit Cable Pins | Voltage Pins Commoned Together On The A1 Board | Power Cable Exit Pin (To Tape Connector On Rear Of Unit) | Interface Connector Pin (On Rear Of Tape Unit) | |
|---|---|---|---|---|---|---|---|---|---|
| 590 | 01<br>02<br>03<br>04<br>05 | +5V<br>+5V<br>+5V<br>+5V<br>+5V | A4D03<br>A4B03<br>A4D04<br>A4D02<br>A4B02 | C1D03<br>C1J03<br>C1P03<br>C1U03 | B4B03 | A3B02,A3B03,A3D02<br>A3D03,A3D04,A4B02<br>A4B03,A4D02,A4D03<br>A4D04,B4B03,C1D03<br>C1J03,C1P03,C1U03 | A3D03<br>A3B03<br>A3D04<br>A3D02<br>A3B02 | 01<br>02<br>03<br>04<br>05 | |
| | 06[1]<br>07[1]<br>08[1]<br>09[2]<br>10[2]<br>11[2] | Ground<br>Ground<br>Ground<br>Ground<br>Ground<br>Ground | No<br>Ground<br>via<br>Power<br>Cable | C1D08<br>C1J08<br>C1P08<br>C1U08 | B4B08<br>B4D08 | A1D08,A2D08,A3B07<br>A3B08,A3B09,A3D07<br>A3D08,A3D09,A4B07<br>A4B08,A4B09,A4D07<br>A4D08,A4D09,B1D08<br>B2D08,B4B08,B4D08<br>C1D08,C1J08,C1P08<br>C1U08 | No<br>Ground<br>via<br>Power<br>Cable | | |
| | 12 | +8.5V | A4B11 | | | A3B11,A4B11 | A3B11 | 12 | |
| | 13 | +12V | A4B12 | | B4D11 | A3B12,A4B12,B4D11 | A3B12 | 13 | |
| | 14 | -12V | A4D12 | | B4D13 | A3D12,A4D12,B4D13 | A3D12 | 14 | |
| | 15 | -5V | A4D11 | | | A3D11,A4D11 | A3D11 | 15 | |

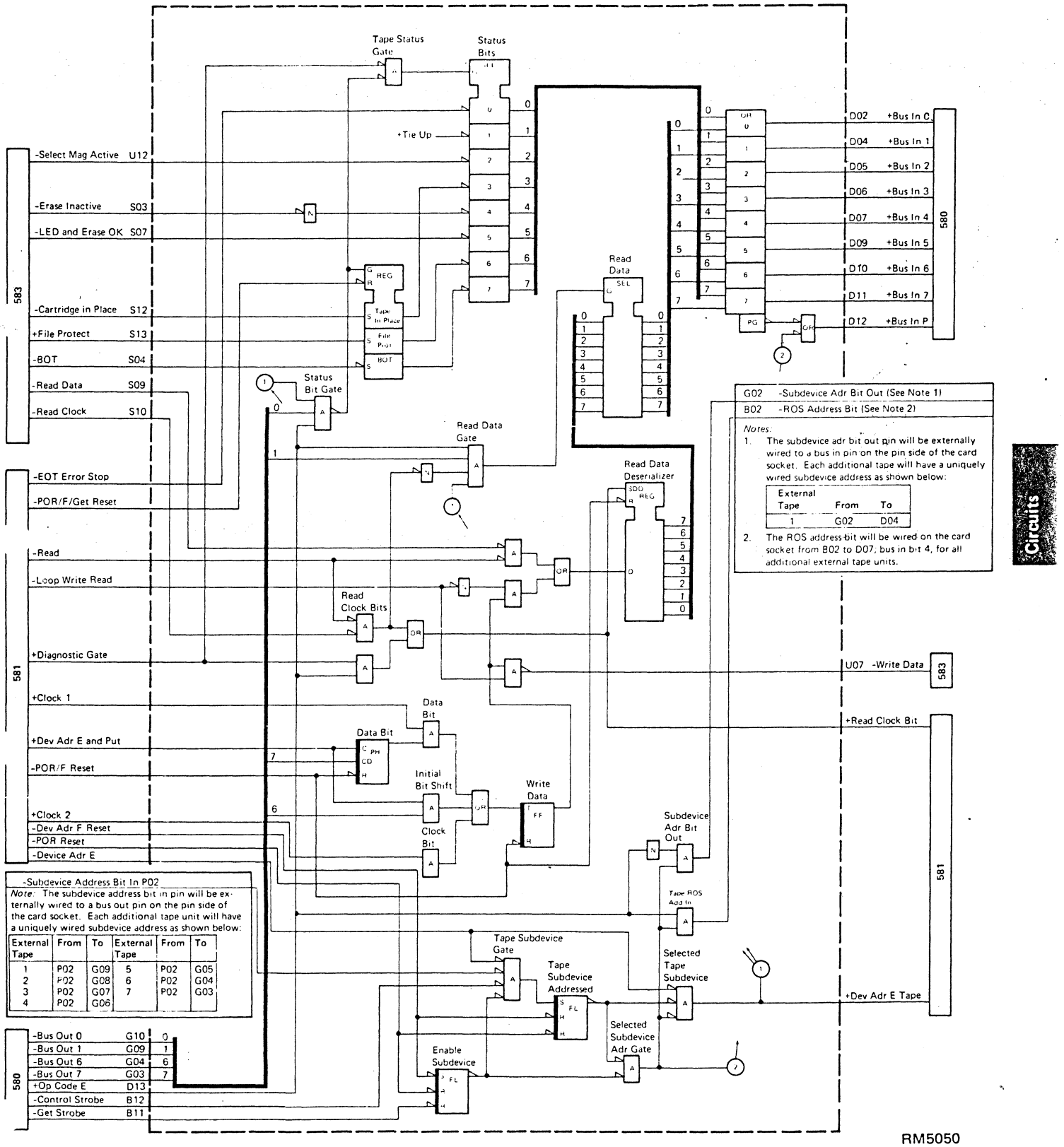
[1] Pins 6, 7, and 8 are wired to pins 1, 13, and 14 of the A2 signal connectors via external jumpers.
[2] Pins 9, 10, and 11 are wired to pins 1, 13, and 14 of the A3 signal connectors via external jumpers.

# 581 AUXILIARY TAPE ADAPTER CARD
(Page 1 of 2)




RM5049

# 2 AUXILIARY TAPE ADAPTER CARD
(Page 2 of 2)

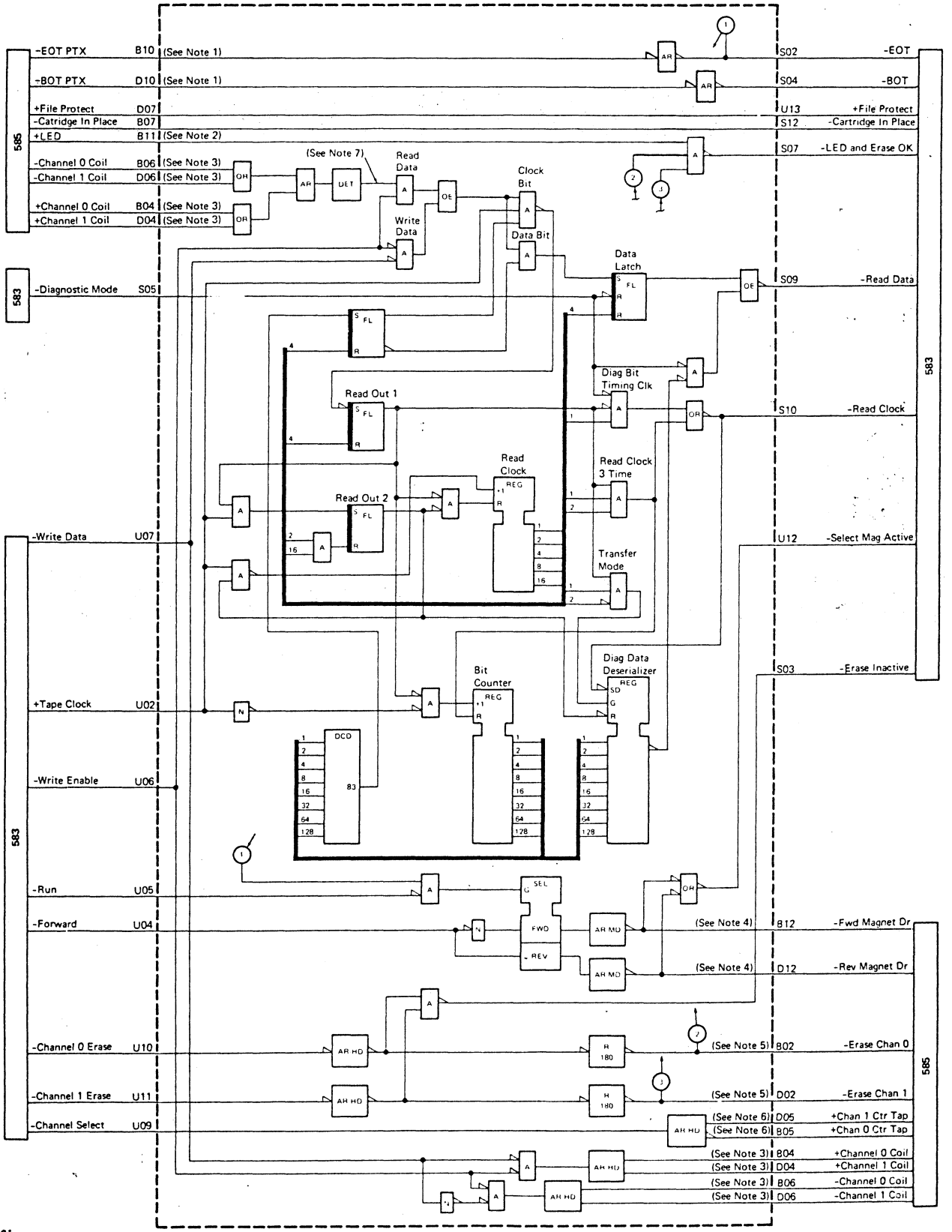



Circuits

# 583 AUXILIARY TAPE CABLE

## Auxiliary Tape A1 Board To Auxiliary Tape Control Card

| | Auxiliary Tape Adapter Card Pin | Tape Unit Cable Pin | Line Name | Auxiliary Tape Control Card Pin | |
|---|---|---|---|---|---|
| 581 | C1U02 | B4B02 | +Tape Clock | U02 | 584 |
| 580 | C1U03 | B4B03 | +5V | U03 | |
| 581 | C1U04 | B4B04 | -Forward | U04 | |
| | C1U05 | B4B05 | -Run | U05 | |
| | C1U06 | B4B06 | -Write Enable | U06 | |
| 582 | C1U07 | B4B07 | -Write Data | U07 | |
| 580 | C1U08 | B4B08 | Ground | U08 | |
| 581 | C1U09 | B4B09 | -Channel Select | U09 | |
| | C1U10 | B4B10 | -Channel 0 Erase | U10 | |
| | C1U11 | B4B11 | -Channel 1 Erase | U11 | |
| 584 | C1U12 | B4B12 | -Select Magnet Active | U12 | 582 |
| | C1U13 | B4B13 | +File Protect | U13 | |
| | C1S02 | B4D02 | -EOT | S02 | 581 |
| | C1S03 | B4D03 | -Erase Inactive | S03 | 582 |
| | C1S04 | B4D04 | -BOT | S04 | |
| 581 | C1S05 | B4D05 | -Diagnostic Mode | S05 | 584 |
| 580 | C1S06 | B4D06 | -5V | S06 | |
| 584 | C1S07 | B4D07 | -LED And Erase OK | S07 | 582 |
| 580 | C1S08 | B4D08 | Ground | S08 | 584 |
| 584 | C1S09 | B4D09 | -Read Data | S09 | 582 |
| | C1S10 | B4D10 | -Read Clock | S10 | |
| 580 | C1S11 | B4D11 | +12V | S11 | 584 |
| 584 | C1S12 | B4D12 | -Cartridge In Place | S12 | 582 |
| 580 | C1S13 | B4D13 | -12V | S13 | 584 |

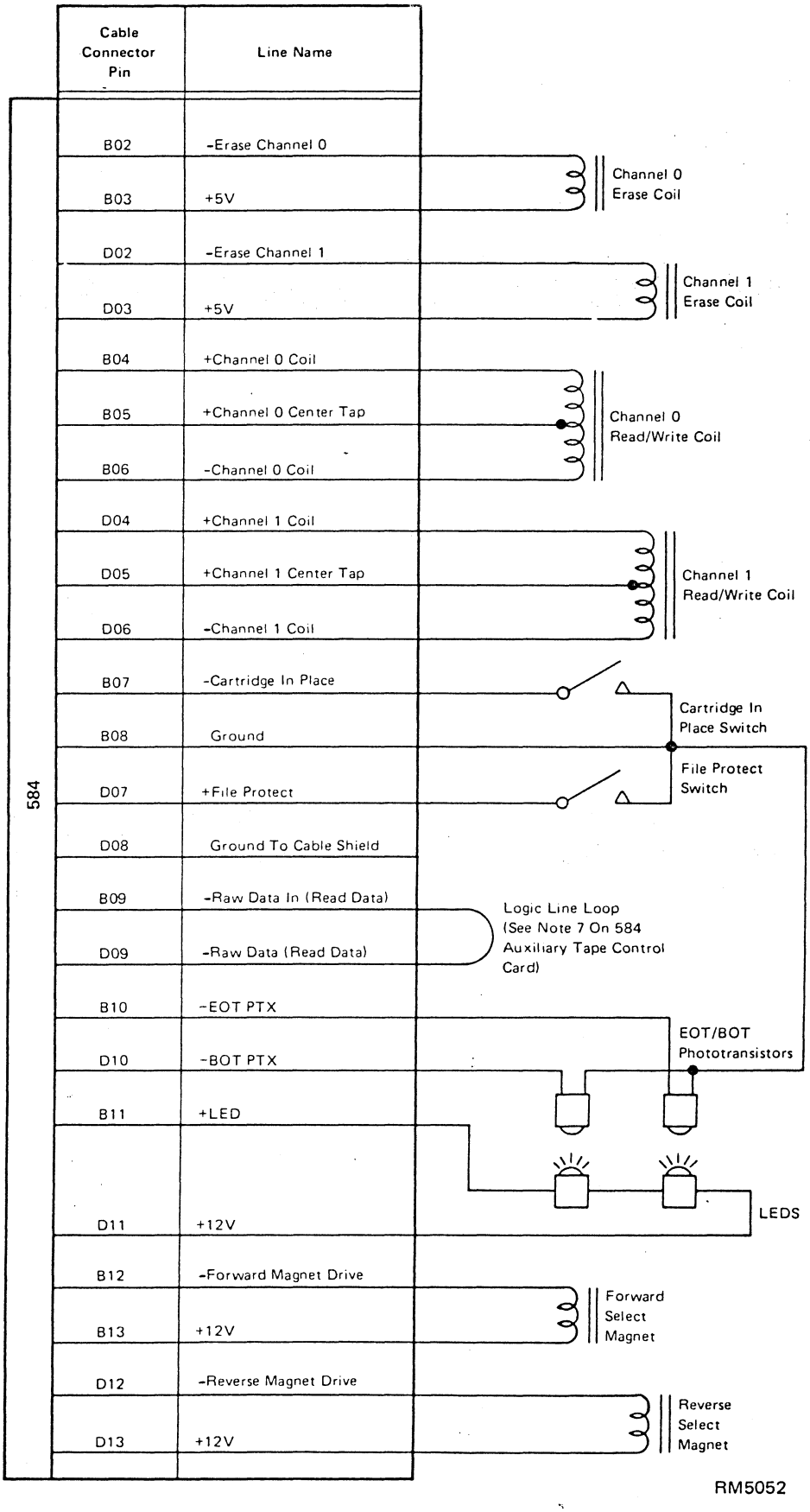



RM5051

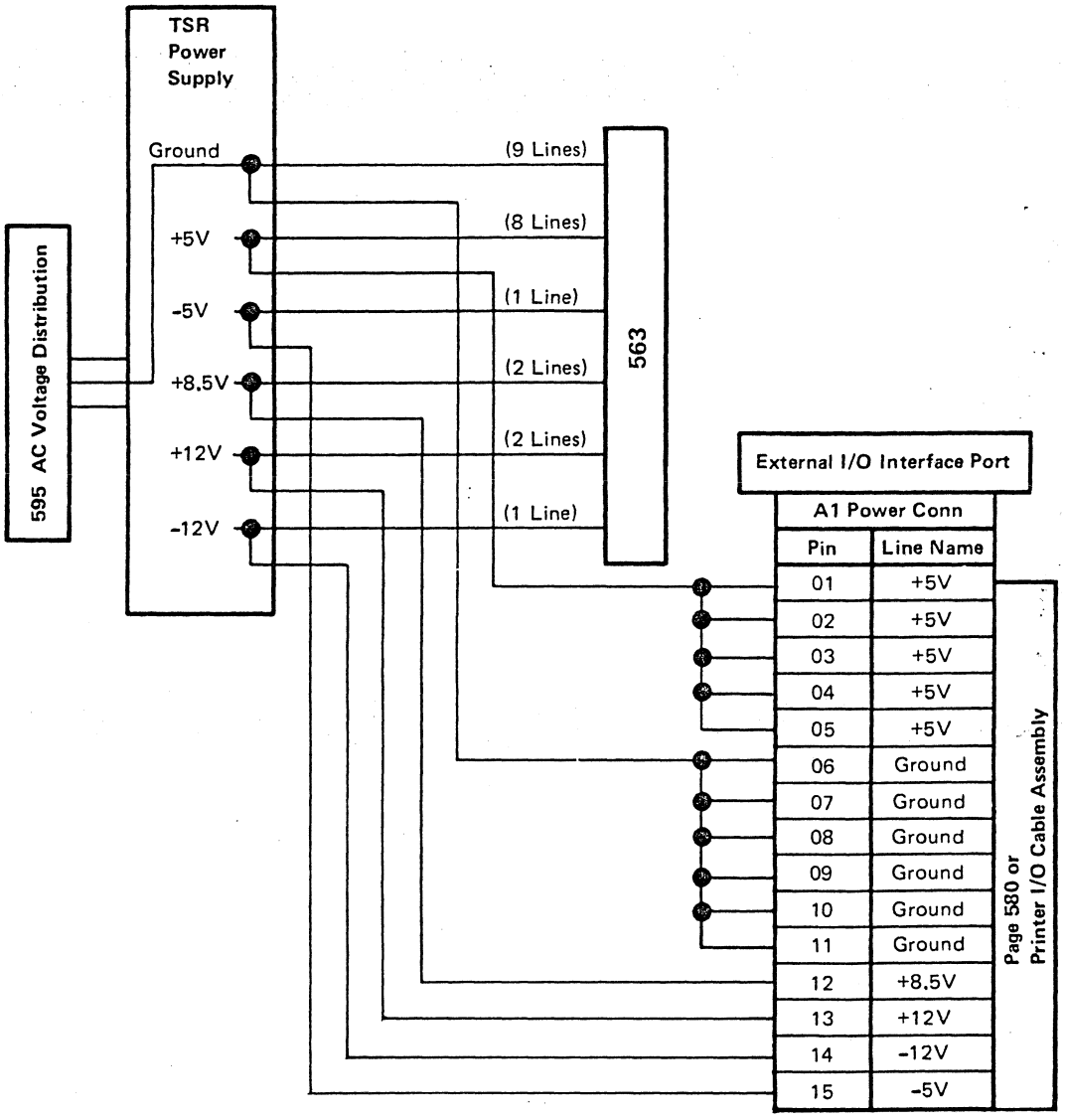
*Notes:*

1. Up +1.2V, down +0.6V to -0.7V.
2. Normally approximately +9.5V.
3. 20 mV P-P voltage centered about a +1V reference level during write operations. A 20 mV P-P voltage around a 0 volt reference level exists during read operations.
4. Up +12V, down 0V.
5. Up +5V, down +4.3V.
6. Up +12V (write operation), down 0V (read operation).
7. Logic line at this point exits card at D09 and reenters at B09 via a jumper on the tape internal cable.

# 585 AUXILIARY TAPE INTERNAL CABLE



# 590 DC VOLTAGE DISTRIBUTION

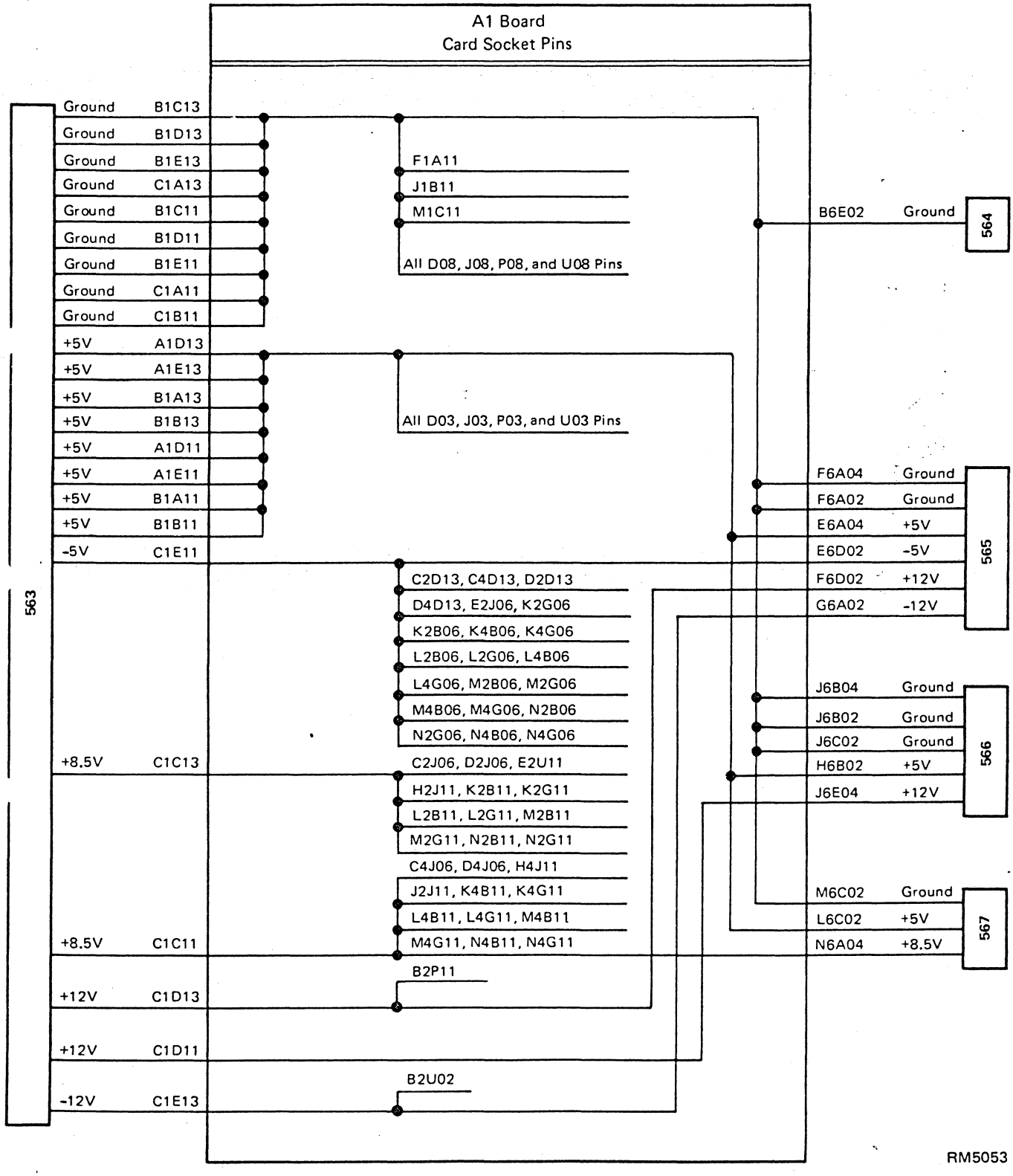



**Voltage Specifications**

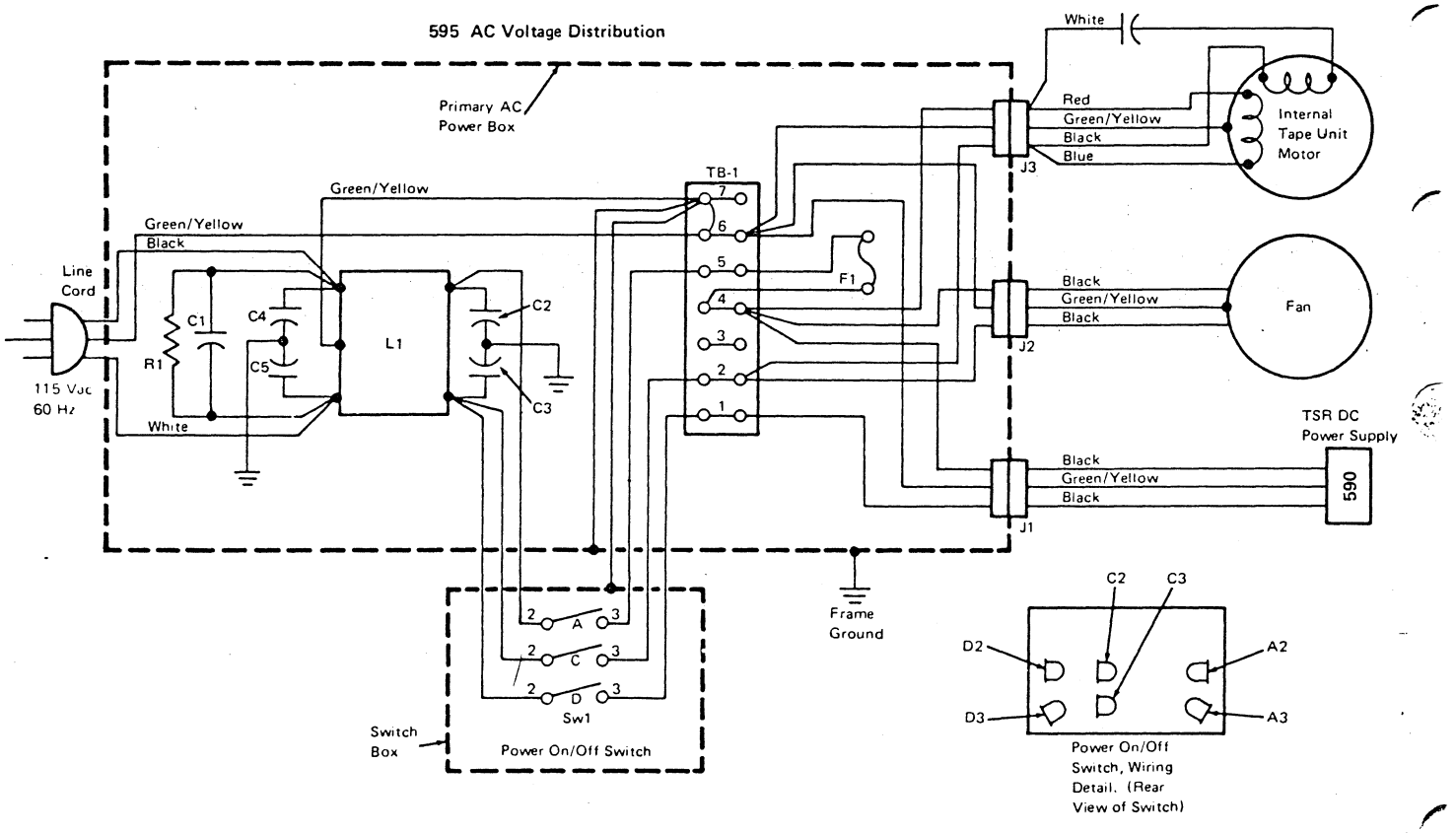
| Voltage | With Load | No Load (Y1 Removed) | Ripple P-P |
|---|---|---|---|
| +5V | 4.6 to 5.5 | 5.5 to 6.5 | .1 |
| -5V | 4.6 to 5.5 | -3.7 to -4.7 | .1 |
| +8.5V | 7.9 to 9.35 | 7.4 to 9.0 | .17 |
| +12V | 11.0 to 13.2 | 9.8 to 12.2 | .24 |
| -12V | 11.0 to 13.2 | -9.0 to -11.5 | .24 |

This page intentionally left blank.

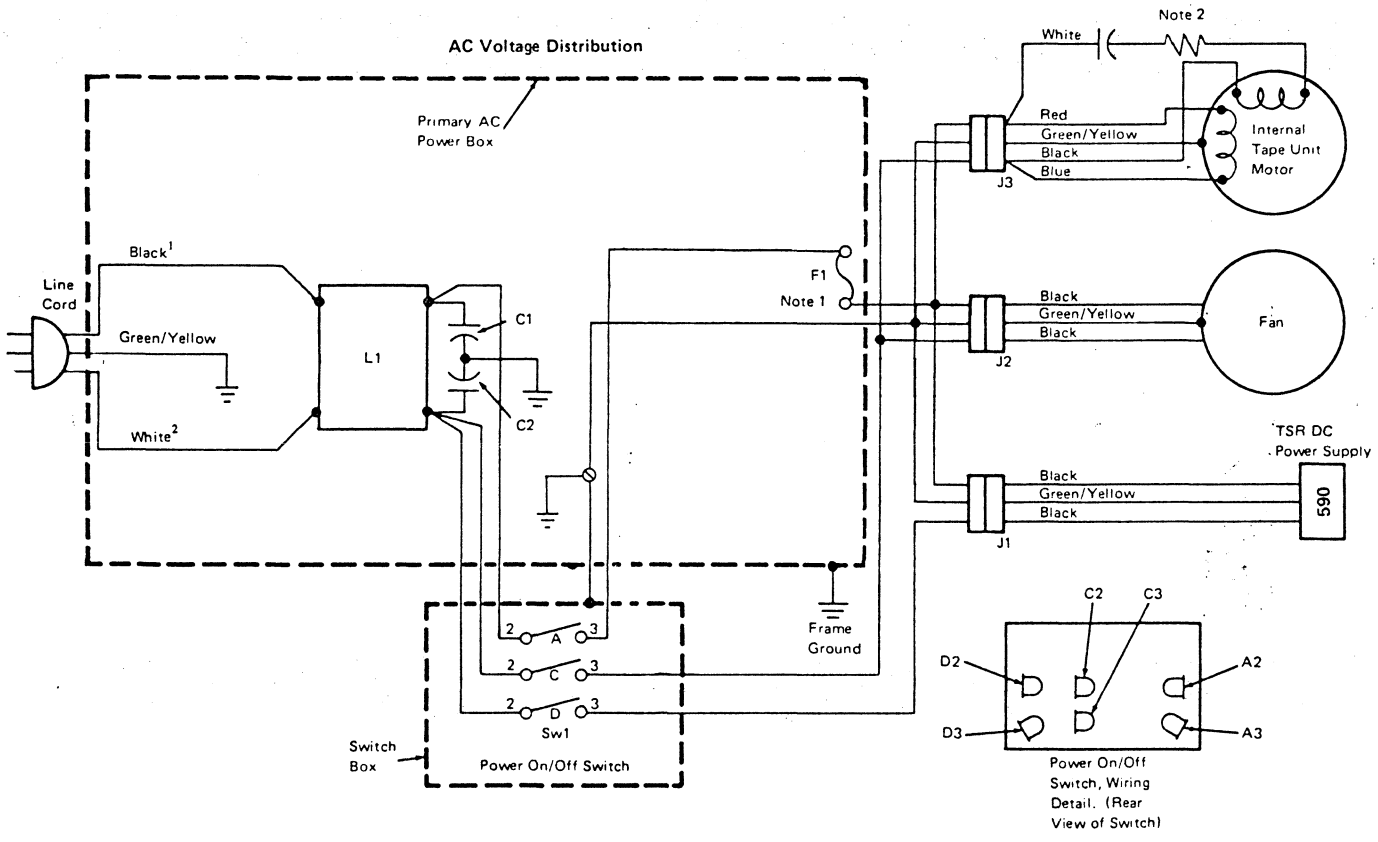
# 591 A1 BOARD-DC VOLTAGE DISTRIBUTION



# 595 AC VOLTAGE DISTRIBUTION (old style)
(Page 1 of 2)



## AC VOLTAGE DISTRIBUTION (new style)
(Page 2 of 2)




RM5073

---

[1] Brown on 220 volt and 235 volt machines
[2] Blue on 220 volt and 235 volt machines

*Notes:*

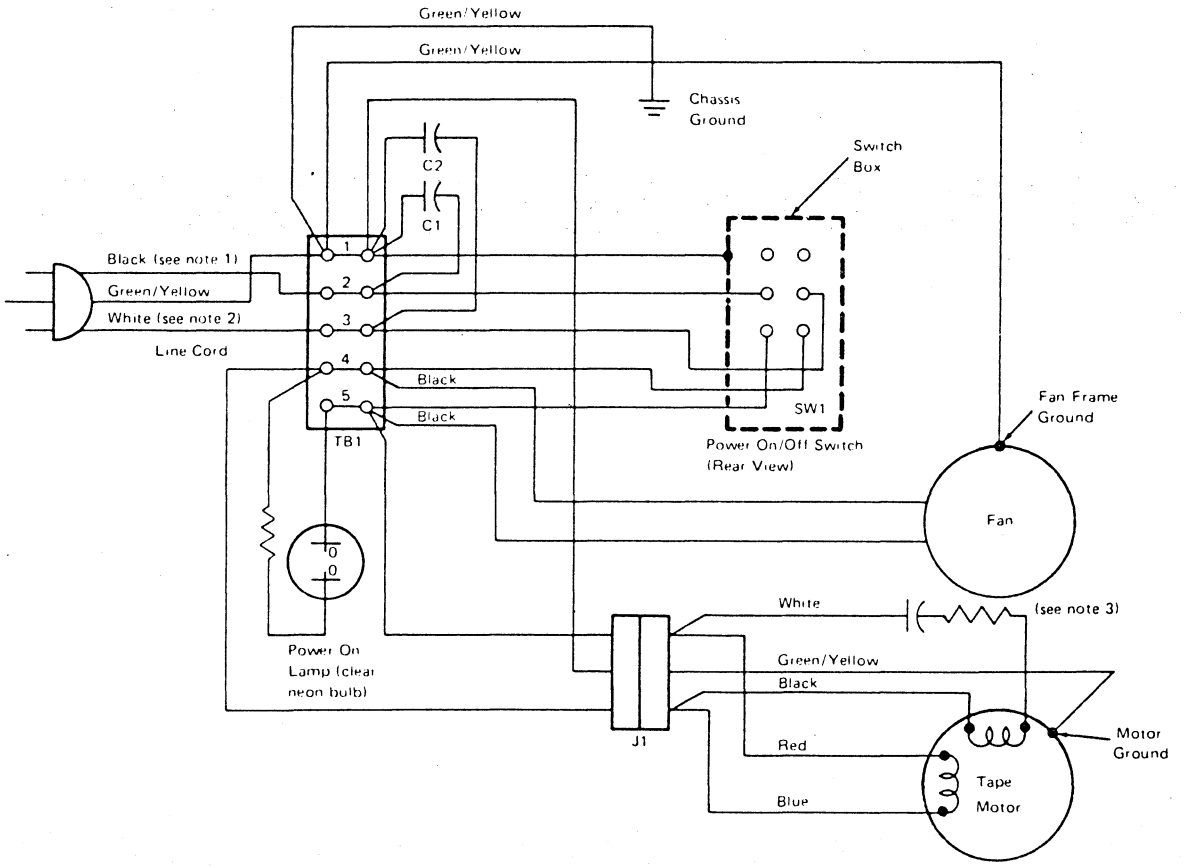
1. F1 is 5A, 125 volts on both the 100 volt and 115 volt machines.

   F1 is 3A, 250 volt on both the 220 volt and 235 volt machines.

2. Resistor installed on 220 volt and 235 volt machines only.

| | |
|---|---|
| L1 | Line Filter |
| J1, 2, 3 | Connectors |
| C1, 2 | Capacitors |
| F1 | Fuse (see note 1) |
| SW1 | Power ON/OFF switch |

# 596 AC VOLTAGE DISTRIBUTION FOR AUXILIARY TAPE UNIT



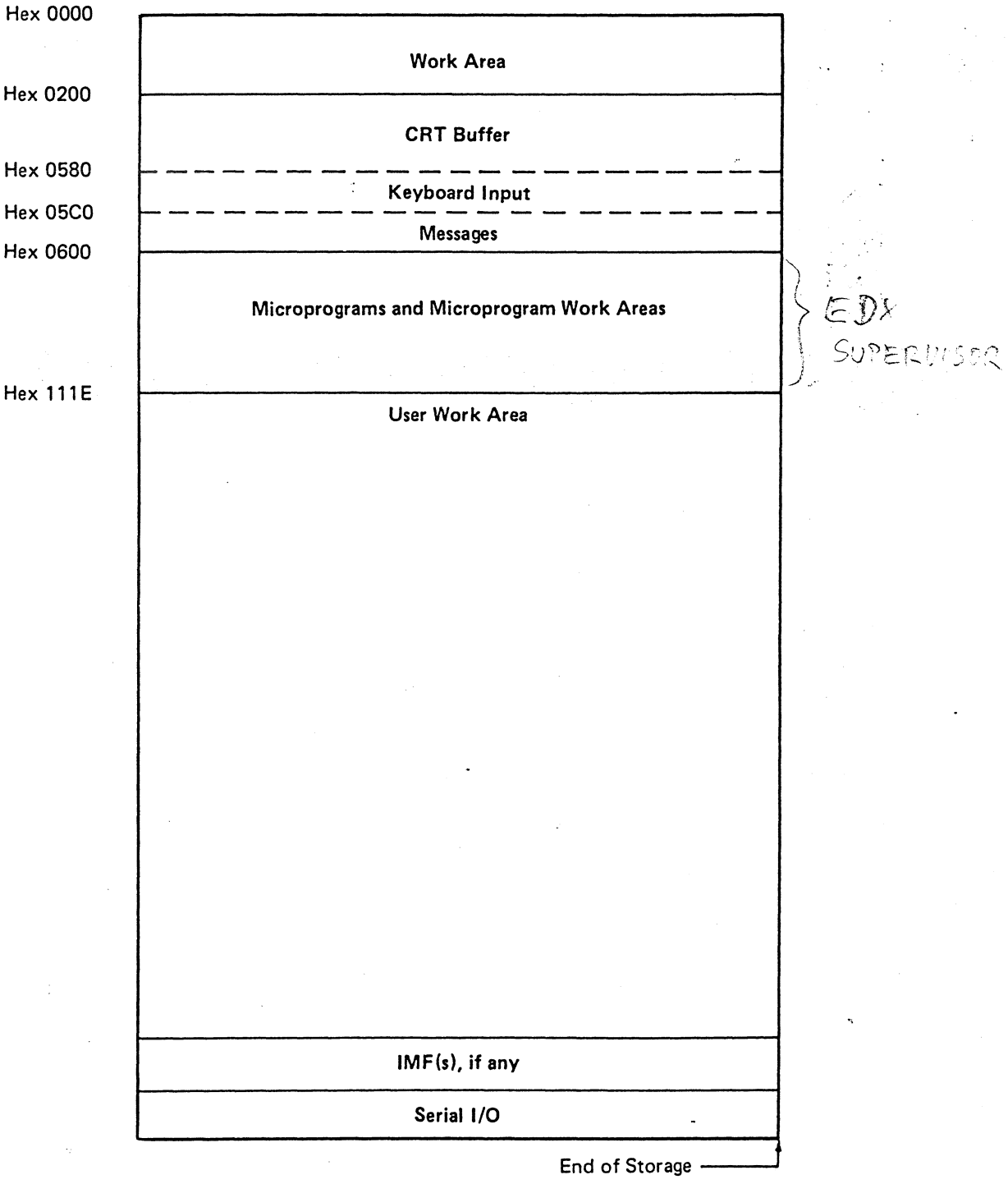

*Notes:*

1. Brown on 220V and 235V machines.
2. Blue on 220V and 235V machines.
3. Resistor installed on 220V and 235V machines only.

This page is intentionally left blank.

## BASIC Data Areas

The following diagram shows the relative locations of read/write storage data areas when performing BASIC language operations:



## HOW TO FIND THE IOCBs FOR THE BASIC LANGUAGE

By using the following chart, you can find all of the IOCBs in read/write storage that are used during BASIC language operations:

| | |
|---|---|
| 0B00 | CRT IOCB |
| 0B14 | Keyboard IOCB |
| 0B29 | Address of Printer IOCB |
| 0B2B | Address of FL0 IOCB |
| 0B2D | Address of FL1 IOCB |
| 0B2F | Address of FL2 IOCB |
| 0B31 | Address of FL3 IOCB |
| 0B33 | Address of FL4 IOCB |
| 0B35 | Address of FL5 IOCB |
| 0B37 | Address of FL6 IOCB |
| 0B39 | Address of FL7 IOCB |
| 0B3B | Address of FL8 IOCB |
| 0B3D | Address of FL9 IOCB |
| 0B9E | Address of IOCB with device error |
| 0D03 | Address of last IOCB used |
| 0DE4 | Address of Command IOCB |

*Note:* File designation for FL0-FL9 IOCBs is specified by the user in the OPEN statement.

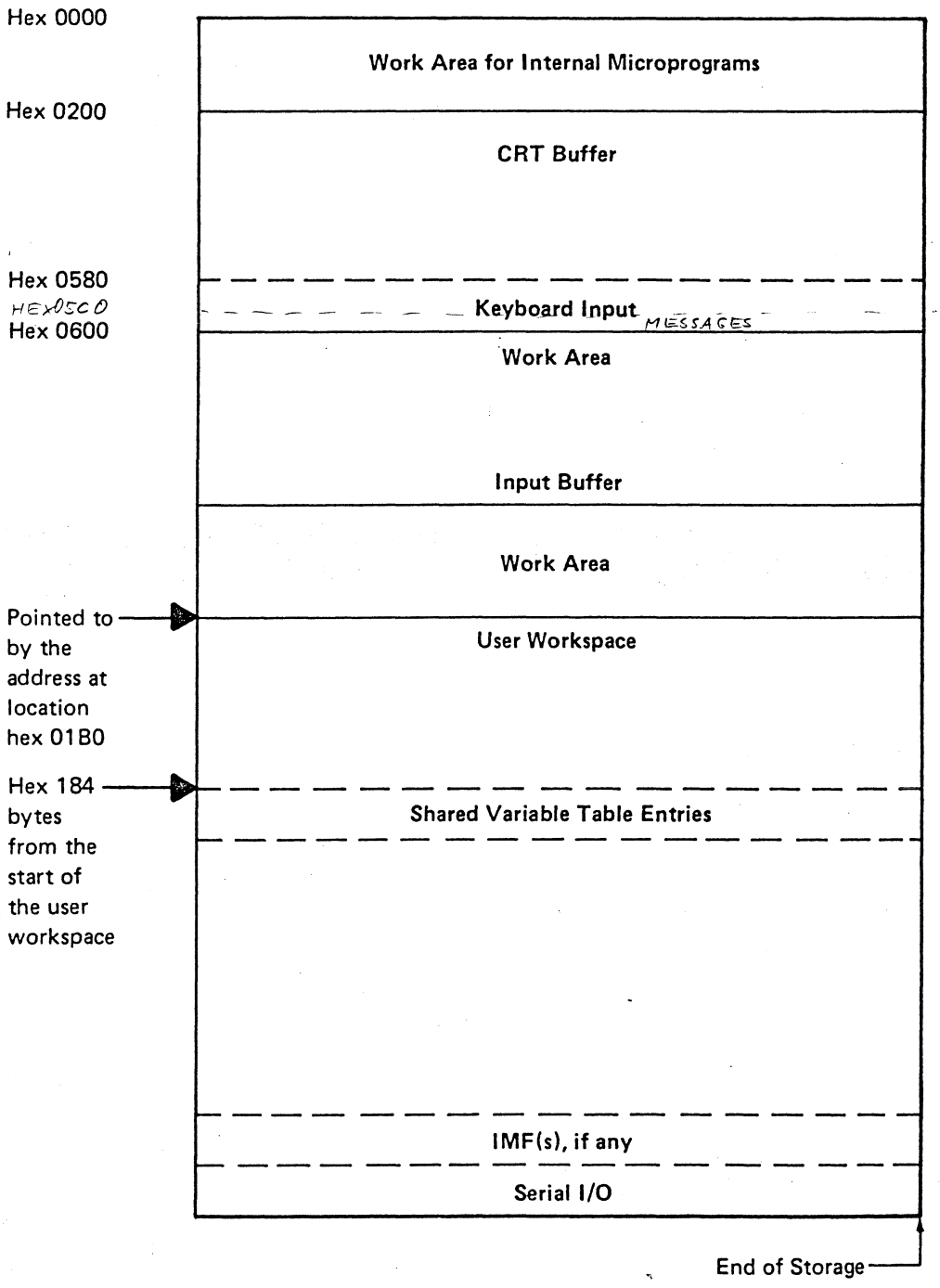
For example, the address contained at location hex 0B29 is the address of the printer IOCB. By displaying the printer IOCB (see *DCP1 Functions*) and using the format of the IOCB (see *Input/Output Control Blocks*), you can find important information about the operation of the printer. For example, if you want to know where the printer buffer is located in read/write storage:

- Display the printer IOCB.
- Count 4 hex bytes from the beginning of the IOCB, starting with byte 0.
- Hex bytes 4 and 5 contain the buffer start address, and bytes 6 and 7 contain the buffer size.

By using this procedure, any information in any IOCB can be found.

## PL Data Areas

The following diagram shows the relative locations of read/write storage data areas during APL operations:



## SHARED VARIABLE TABLE ENTRIES

Use the shared variable table entries to find the IOCBs, I/O buffers, and control vectors when using APL shared variables for I/O operations. See *How To Find the I/O Buffer* for an example of how to use this information.

The shared variable table entries are found by adding hex 0184 to the address of the start of the user workspace. The address of the start of the user workspace is a 4-byte address and is stored at location hex 01B0.

There are eight table entries, each 8 bytes long. The first 4-byte address is an I/O buffer area address and the second 4-byte address is a control vector address. These addresses are displacements from the start of the user workspace. Any of the eight entries can be active at any time. If any entry contains all hex zeros, it is inactive.

## I/O BUFFER AREA

The I/O Buffer area is found by adding the I/O buffer area displacement address, found in the table entry, to the address at location hex 01B0 (the start of the user workspace). The I/O buffer area contains the following information:

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 0-3 | 4 | Address displacement | Displacement from the beginning of the user workspace pointing to the table entry containing the displacement address for this I/O buffer area. |
| 4-7 | 4 | Binary | Number of bytes in this I/O buffer area. |
| 8-9 | 2 | Address displacement | Displacement from the beginning of this I/O buffer area pointing to the first byte of the I/O buffer. |
| A-B | 2 | Address displacement | Displacement from the beginning of this area pointing to the current position of the I/O buffer. |
| C-D | 2 | Address displacement | Displacement from the beginning of this area pointing to the start of the I/O workspace (1 byte past the end of the I/O buffer). |
| E-F | 2 | Address displacement | Displacement from the beginning of this I/O buffer area pointing to the logical record buffer (interchange input only). |
| 10-21 | 18 (decimal) | Variable | IOCB (see *Input/Output Control Blocks*). |
| 22-variable | Variable | Variable | I/O buffer (see *I/O Buffer*). |
| Variable (pointed to by bytes C and D) | Variable | Variable | I/O work area. |
| Variable (pointed to by bytes E and F) | Variable | Variable | Logical record buffer. |

## JNTROL VECTOR

The control vector is found by adding the control vector displacement address, located in the table entry, to the address located at hex 01B0 (the start of the user workspace).

The control vector contains information pertaining to its associated I/O buffer in the following format:

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 0-3 | 4 | Address displacement | Displacement from the beginning of the user workspace pointing to the table entry containing the address of this control vector. |
| 4-7 | 4 | Binary | Number of bytes in the control vector. |
| 8-9 | 4 | Binary | The maximum record length for interchange format only. For all other formats, these bytes are meaningless. |
| A-B | 2 | Binary | The current record length for interchange format only. For all other formats, these bytes are meaningless. |
| C-D | 2 | Binary | Return code:<br>X'0000' – Successful completion<br>X'0001' – Error (see error code in bytes E-11)<br>X'0002' – Invalid file<br>X'0003' – Invalid device number (OPEN only)<br>X'0004' – Invalid file number (OPEN only)<br>X'0005' – Device already assigned (OPEN only)<br>X'0006' – Invalid parameter (OPEN only)<br>X'0007' – Workspace full (OPEN only)<br>X'0008' – Device not open<br>X'0009' – 0 length record (not end of file)<br>X'000A' – Exceeded maximum record length<br>X'000B' – Invalid data type |
| E-11 | 4 | Binary | Error code. The error code will be all zeros unless the return code is X'0001'. The error code bytes are the same as the return code in the associated IOCB (see *Input/Output Control Blocks*). |
| 12-13 | 2 | Binary | File number. |
| 14 | 1 | Binary | Output interchange file type. |

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 15 | 1 | Binary | Status flags: |

| Bit | Meaning |
|---|---|
| 0 | File is open |
| 1 | Interchange format |
| 2 | Input file |
| 3 | Add file |
| 4 | Internal use |
| 5 | Return code has been set |
| 6 | ' ID=' specified |
| 7 | Printer format |

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 16 | 1 | Binary | Status flags: |

| Bit | Meaning |
|---|---|
| 0 | Printer previously on by )OUTSEL |
| 1 | MSG – OFF specified by the user |
| 2 | Invalid processor specified by the user |
| 3 | Do not write last (previous error) |
| 4-7 | Not used |

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 17 | 1 | Binary | Reserved |

## I/O BUFFER

The I/O buffer is contained in the I/O buffer area (see *I/O Buffer Area*) and can be in one of three formats. By checking the status flag byte in the associated control vector, the data type can be determined. If bit 1 is on, the data is in interchange format. If bit 7 is on, the data is in printer format. If both of these bits are off, the data is in internal format.

### Printer Format

If the data in the I/O buffer is in printer format, the data is in 5100 internal code. If no error occurred, the buffer is all hex zeros.

### Interchange Format

If the data in the I/O buffer is in interchange format, it is in 5100 internal code and contains from part of one logical record to multiple logical records. Hex E3 indicates the end of meaningful data in the buffer.

### Internal Format

If the data in the I/O buffer is in internal format, it contains from part of one logical record to multiple logical records. An end of record word, hex FF050E04, occurs after each logical record. This word should appear just before the 4-byte length word of the next logical record. Using this length word, you should find another end of record word. This will verify that you have found a valid logical record. The internal format is:

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 0-3 | 4 | Binary | Length of the logical record minus these 4 bytes. |
| 4 | 1 | Binary | Format of the data:<br><br>Hex 01 – Boolean (logical)<br>Hex 02 – Integer<br>Hex 03 – Floating point<br>Hex 04 – Character |
| 5 | 1 | Binary | Reserved. |
| 6-7 | 2 | Binary | Number of dimensions (rank) times 4. |
| 8-variable | Number specified in bytes 6 and 7 | Binary | Dimension (shape) of the vector. There is a 4-byte entry for each dimension, which contains the number of elements in that dimension. |
| Variable | Variable | Variable | Data in the format specified in byte 4:<br><br>Boolean (logical) – binary<br>Integer – 4 bytes per integer<br>Character – 1 byte per character and is stored in Z code (see *Z Code*)<br>Floating point – 8 bytes per number (see *APL Floating-Point Format*) |

### APL Floating-Point

The APL floating-point data format is a method of storing arithmetic data and is always 8 bytes long. The format of the 8 bytes is:

| Byte | Bit | Description |
|---|---|---|
| 0 | 0 | Indicates the sign of the arithmetic data. If bit 0 is off, the number is positive; if bit 0 is on, the number is negative. |
| 0 | 1-7 | Hex value indicating the direction to move the hexadecimal point and how far. |
| 1-7 | | Hex value of the arithmetic data. |

If the value specified in bits 1-7 of byte 0 is greater than hex 40, subtract hex 40 from that value. The difference is the number of half bytes (hex digits) to move the hexadecimal point to the right. If the value specified in bits 1-7 of byte 0 is less than hex 40, subtract that number from hex 40. The difference is the number of half bytes to move the hexadecimal point to the left.

The following examples show how to convert the floating-point data stored in read/write storage from hexadecimal to decimal:

*Example 1:* In the floating-point number 43 30 88 00 00 00 00 00, bit 0 of byte 0 is off so the number is positive. Bits 1-7 of byte 0 equal hex 43 indicating that the hexadecimal point be moved 3 half bytes to the right. After dropping the trailing zeros and moving the hexadecimal point, the hex number .10 88 00 00 00 00 00 becomes 108.8. After the hexadecimal point has been moved, the number must be converted to its decimal value. To convert the number to decimal, multiply the values shown in the following chart by the value of each half byte in the hexadecimal number and add the results:

Hexadecimal point after conversion

| ..... 1048576 | 65536 | 4096 | 256 | 16 | 1 | . | 1/16 | 1/256 | 1/4096 | 1/65536 | 1/1048576 ..... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| The converted hexadecimal number | | | 1 | 0 | 8 | . | 8 | | | | |

1 x 256 = 256
0 x 16 = 0
8 x 1 = 8
8 x 1/16 = 0.5

Result = 264.5

The decimal value of the floating-point number 43 10 88 00 00 00 00 00 is 264.5.

Note that each half byte to the left of the hexadecimal point increases by a multiple of 16, and each half byte to the right of the hexadecimal point decreases by a multiple of 1/16.

*Example 2:* In the floating-point number 40 C0 00 00 00 00 00 00, bit 0 of byte 0 is off, so the number is positive. Bits 1-7 equal hex 40 indicating no movement of the hexadecimal point. After dropping trailing zeros, the hexadecimal number is .C (hex C = 12). Using the chart in example 1, multiply 12 by 1/16, which equals 0.75. The decimal value of the floating-point number 40 C0 00 00 00 00 00 00 is 0.75.

*Example 3:* In the floating-point number BF 50 00 00 00 00 00 00, bit 0 of byte 0 is on so the number is negative. Bits 1-7 equal 3F indicating that the hexadecimal point be moved 1 half byte to the left. After dropping the trailing zeros and moving the hexadecimal point, the hex number .50000000000000 becomes -0.05. Using the chart in example 1, multiply:

0 x 1/16 = 0
5 x 1/256 = 0.01953125

Result = -0.01953125

So the decimal value of the floating-point number BF 50 00 00 00 00 00 00 is -0.01953125.

By using this procedure, any floating-point number can be converted to its decimal value.

## HOW TO FIND THE I/O BUFFER IN APL

In the following example, data was received from an I/O device using the serial I/O Adapter feature. The I/O operation was not completed because an end of buffer character was not received from the I/O device. Therefore, the data received from the I/O device is still in the I/O data buffer.

This data can be displayed and interpreted as follows:

1. Using the DCP1 display function, display the address of the start of the user workspace, which is located at hex 01B0 in read/write storage:

```
DIAG                DCP1                                    DCP1

D 01B0          Address of the start of the user workspace
  LOC@
  01B0      0000  0E80  8274  8342  AC00  0000  0000  0000

  01C0      4003  5682  0000  0000  0000  0000  818E  8142
```

2. Add hex 0184 to the address of the start of the user workspace:

   Hex 0E80 (start of user workspace)
   Hex 0184
   Hex 1004 (address of shared variable table entries)

3. Display the shared variable table entries at hex address 1004.

```
DIAG                DCP1                                    DCP1

D 1004          Displacement address of I/O buffer area
  LOC@
  1004      0000  21C4  0000  2184  0000  0000  0000  0000

  1014      0000  0000  0000  0000  0000  0000  0000  0000
```

4. Now, to find the I/O buffer area, add the I/O buffer area displacement address to 0E80 (address of the start of the user workspace):

   Hex 21C4 (I/O buffer area displacement address)
   Hex 0E80 (address of start of user workspace)
   Hex 3044 (address of the I/O buffer area)

5. Display the I/O buffer area:
   Refer to *I/O Buffer Area*

```
DIAG                DCP1                                    DCP1

D 3044
  LOC@                                          Displacement to first position of I/O buffer
  3044    0000  0184  0000  01BC  0022  00EE  00EE  00F0

  3054    0A20  0100  3066  00CC  0001  3132  0000  0000
                      I/O buffer start address
```

6. The I/O buffer is located 22 hex bytes from the beginning of the I/O buffer area (refer to *I/O Buffer Area*). Add hex 22 to the address of the I/O buffer area:

   Hex 22 (displacement to first position of I/O buffer)
   Hex 3044 (beginning of I/O buffer area)
   Hex 3066 (buffer start address)

   You can also use the buffer start address in the IOCB as shown in step 5.

7. Display the I/O buffer:

   The I/O buffer is set to all hex zeros when the I/O operation is complete. If you stop the I/O operation by pressing the CMD and HOLD keys, you might cause data in the workspace and/or tape files to be lost.

Use the 5100 internal code chart to decode the data characters in the I/O buffer.

```
DIAG                DCP1                                    DCP1

D 3066                                  End of record character
  LOC@          Data
  3066    190F  1500  0801  1605  0006  0F15  0E04  0014

  3076    0805  0009  250F  0002  1506  0605  1200  0000
```

## CODE CHART

When doing APL language operations, some data is stored in Z code (see *Internal Format*). Use the following chart to decode this data.

| Hex | Graphic | Hex | Graphic | Hex | Graphic | Hex | Graphic | Hex | Graphic |
|---|---|---|---|---|---|---|---|---|---|
| 01 | | 26 | = | 4B | ⍀ | 70 | Δ | 95 | 9 |
| 02 | | 27 | ≥ | 4C | ⍁ | 71 | A | 96 | • |
| 03 | | 28 | > | 4D | ⍒ | 72 | B | 97 | ‾ |
| 04 | | 29 | ≠ | 4E | ⍋ | 73 | C | 98 | space |
| 05 | | 2A | ⍺ | 4F | & | 74 | D | 99 | ! |
| 06 | | 2B | ∈ | 50 | @ | 75 | E | 9A | : |
| 07 | | 2C | ⍳ | 51 | # | 76 | F | 9B | ▽ |
| 08 | | 2D | ⍴ | 52 | $ | 77 | G | 9C | Cursor ret |
| 09 | | 2E | ⍵ | 53 | | 78 | H | 9D | Idle |
| 0A | | 2F | , | 54 | | 79 | I | 9E | Backspace |
| 0B | | 30 | ! | 55 | | 7A | J | 9F | Line feed |
| 0C | | 31 | ⍶ | 56 | A | 7B | K | A0 | ⍒ |
| 0D | | 32 | ⊥ | 57 | B | 7C | L | A1 | |
| 0E | ] | 33 | ⊤ | 58 | C | 7D | M | A2 | |
| 0F | [ | 34 | o | 59 | D | 7E | N | A3 | |
| 10 | ( | 35 | ? | 5A | E | 7F | O | A4 | |
| 11 | ) | 36 | ~ | 5B | F | 80 | P | A5 | |
| 12 | ; | 37 | ↑ | 5C | G | 81 | Q | A6 | |
| 13 | / | 38 | ↓ | 5D | : | 82 | R | A7 | |
| 14 | \ | 39 | ⊂ | 5E | I | 83 | S | A8 | |
| 15 | ← | 3A | ⊃ | 5F | J | 84 | T | A9 | ▯ (see note) |
| 16 | → | 3B | ∩ | 60 | K | 85 | U | AA | ¬ |
| 17 | | 3C | ∪ | 61 | L | 86 | V | AB | ¨ |
| 18 | | 3D | _ | 62 | M | 87 | W | AC | % |
| 19 | ¨ | 3E | ⍱ | 63 | N | 88 | X | AD | Ã |
| 1A | + | 3F | I | 64 | O | 89 | Y | AE | ⊛ |
| 1B | - | 40 | ° | 65 | P | 8A | Z | AF | Ä |
| 1C | × | 41 | ⍞ | 66 | Q | 8B | Δ | B0 | Ö |
| 1D | ÷ | 42 | ⍠ | 67 | R | 8C | 0 | B1 | Ü |
| 1E | * | 43 | ⊛ | 68 | S | 8D | 1 | B2 | Ä |
| 1F | ⌈ | 44 | ⍲ | 69 | T | 8E | 2 | B3 | Æ |
| 20 | ⌊ | 45 | ⍱ | 6A | U | 8F | 3 | B4 | Ñ |
| 21 | K | 46 | ⍝ | 6B | V | 90 | 4 | B5 | Ñ |
| 22 | ∧ | 47 | ⍫ | 6C | W | 91 | 5 | B6 | £ |
| 23 | ∨ | 48 | ⍙ | 6D | X | 92 | 6 | B7 | Ç |
| 24 | < | 49 | ⊖ | 6E | Y | 93 | 7 | B8 | Ö |
| 25 | ≤ | 4A | ⍟ | 6F | Z | 94 | 8 | B9 | Ã |

*Note:* OUT Character

## Input/Output Control Blocks

The IOCB (input/output control block) is the interface between the APL or BASIC interpreter and the I/O supervisor. When the APL or BASIC interpreter executes an APL or BASIC statement that requires an I/O function, the APL or BASIC interpreter places the appropriate I/O code in the appropriate IOCB and branches to the I/O supervisor. The I/O supervisor executes the I/O function specified by the I/O code and places a return code in the IOCB to inform the APL or BASIC interpreter how the I/O operation is completed. This return code is checked by the APL or BASIC interpreter.

The following IOCB formats are the same for both languages unless specified otherwise:

### COMMAND IOCB (BASIC ONLY)

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 0 | 1 | Binary | Defines the device assigned to the IOCB:<br>Hex 00 – Display screen<br>Hex 04 – Keyboard<br>Hex 01 – ROS<br>Hex 05 – Printer<br>Hex 0E – Tape<br>Hex 08 – Expansion feature[1] (COMMUNICATION)<br>Hex 0A – SERIAL I/O |
| 1 | 1 | Binary | Specifies subdevice:<br>Hex 00 – Not used<br>Hex 80 – Built-in tape unit<br>Hex 40 – Auxiliary tape unit<br>Hex 08 – Serial I/O command device<br>Hex 04 – Serial I/O output device<br>Hex 02 – Serial I/O input device<br>Hex 20 – Serial I/O BASIC load<br>Hex 40 – Serial I/O BASIC save |

[1] Hardware and communications device address is hex 08. Serial I/O device address is hex 0A.

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 2 | 1 | Binary | I/O codes:<br>X'00' – Sense<br>X'01' – Read for tape and ramp head for printer<br>X'02' – Write<br>X'03' – Write last<br>X'04' – Find<br>X'05' – Mark<br>X'06' – Initialize and Mark<br>X'07' – Rewind<br>X'08' – Forward space record<br>X'09' – Backspace record<br>X'0A' – Find next header<br>X'0B' – Write header |
| 3 | 1 | | Hex 80 – Ignore ATTN key<br>Hex 00 – Honor ATTN key |
| ~~4-5~~ | ~~2~~ | ~~Address~~ | ~~Buffer start address.~~ |
| 4-5 | 2 | Address | Buffer start address. |
| 6-7 | 2 | Binary | Buffer size. |
| 8-9 | 2 | Binary | Control information:<br>If Find – File number to find<br>If Mark – Number of records to allocate<br>If Print – Space information:<br>Hex 0001-Hex 000F indicates 1-15 lines of spacing |
| A-B | 2 | Address | I/O work area address. |

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| C-D | 2 | Binary or character | Return code:<br>X'0000' – Successful completion<br>X'1B1D' – Command error<br>X'1B1E' – Machine error<br>X'1B1F' – Time-out<br>X'1B20' – Tape not mounted<br>X'1B21' – File protect<br>X'1B22' – CRC error<br>X'1B23' – Position error<br>X'1B24' – End of data<br>X'1C1B' – End of file<br>X'1C1C' – End of marked tape<br>X'1C1D' – End of tape<br>X'1C1E' – Device not attached<br>X'1C1F' – Device not selected<br>X'201B' – End of forms<br>X'201C' – Printer not ready<br>X'201D' – Forms step time-out error<br>X'201E' – Line length too large<br>X'201F' – Wire check<br>X'2020' – Undefined interrupt occurred<br>X'2021' – Incorrect print emitter sequence<br>X'2022' – Lack of print emitter pulses when stepping print head<br>X'2023' – Timer interrupt time-out<br>X'2024' – Overspeed error (minimum time between emitter pulses was exceeded) |
| E-F | 2 | Binary | Number of files to be marked. |
| 10-11 | 2 | | Not used. |

KEYBOARD IOCB

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 0 | 1 | Binary | Defines the device assigned to the IOCB:<br><br>Hex 04 — Keyboard |
| 1 | 1 | Binary | Specifies subdevice:<br><br>Hex 00 — Not used for this device<br>Hex 80 — Communications request |
| 2 | 1 | Binary | I/O codes:<br><br>X'01' — Read |
| 3 | 1 | Binary | Displacement in buffer of initial cursor position:<br><br>Hex 00-Hex 3F for BASIC<br>Hex 00-Hex 7F for APL<br><br>For communications and diagnostics, this byte contains the 5100 internal code for the key pressed. For APL, this byte might contain the current cursor position. |
| 4-5 | 2 | Address | Buffer start address |
| 6-7 | 2 | Binary | Not used. |
| 8 | 1 | Binary | Used to pass information back to the interpreter when one of the function keys (CMD with numeric), scroll up key, or scroll down key is pressed. (Function keys are not supported by APL.)<br><br>Hex 80 — Scroll up key<br>Hex 40 — Scroll down key<br>Hex 20 — EXECUTE key<br>Hex 0X — Function key (X = 0-9) |
| 9 | 1 | Binary | Hex 80 — Scroll inhibit. |
| A-B | 2 | Address | Address of keyword table. |
| C-D | 2 | Binary | Return code:<br><br>X'0000' — Successful completion |
| 1 | 4 | | Not used. |
| 12-13 | 2 | Address | Current buffer pointer (BASIC only). |

## 5103 PRINTER IOCB

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 0 | 1 | Binary | Defines the device assigned to the IOCB:<br><br>Hex 05 — Printer |
| 1 | 1 | Binary | Specifies subdevice:<br><br>Hex 00 — Not used for this device |
| 2 | 1 | Binary | I/O codes:<br><br>X'00' — Sense<br>X'01' — Ramp head (no spacing occurs)<br>X'02' — Write<br>X'FF' — Diagnostic write |
| 3 | 1 | | Not used. |
| 4-5 | 2 | Address | Buffer start address. |
| 6-7 | 2 | Binary | Buffer size. If value is hex 0000, no printing occurs. Values hex 0001 to hex 0084 are valid and cause 1 line to be printed. |
| 8-9 | 2 | Binary | Forms control (spacing):<br><br>Hex 0000 — No spacing of forms<br>Hex 0001 to hex 000F — Spacing of forms from 1 to 15 lines |
| A-B | 2 | Address | I/O work area address. |
| C-D | 2 | Binary | Return code:<br><br>X'0000' — Successful completion<br>X'201B' — End of forms<br>X'201C' — Printer not ready<br>X'201D' — Form step time-out error<br>X'201E' — Line length too large<br>X'201F' — Wire check<br>X'2020' — Undefined interrupt occurred<br>X'2021' — Incorrect print emitter sequence<br>X'2023' — Timer interrupt time-out<br>X'2024' — Overspeed (minimum time between emitter impulses was exceeded) |
| E-11 | 4 | | Not used. |

The following bytes are for BASIC only:

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 12-13 | 2 | Address | Current buffer pointer. |
| 14-15 | 2 | | Not used. |
| 16 | 1 | Binary | Flags:<br>Hex 80 – Device supports input<br>Hex 40 – Device supports output<br>Hex 20 – Device is tape<br>Hex 10 – Reserved<br>Hex 08 – File is open<br>Hex 04 – File is open for output<br>Hex 02 – File is in use for PUT or MAT PUT<br>Hex 01 – File is a PRINT or MAT PRINT file |

**TAPE IOCB (FL0-FL9 – specified by user in the OPEN statement)**

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 0 | 1 | Binary | Defines the device assigned to the IOCB:<br>Hex 0E – Tape |
| 1 | 1 | Binary | Specifies subdevice:<br>Hex 80 – Built-in tape unit<br>Hex 40 – Auxiliary tape unit |
| 2 | 1 | Binary | I/O codes:<br>X'00' – Sense<br>X'01' – Read<br>X'02' – Write<br>X'03' – Write last<br>X'04' – Find<br>X'05' – Mark<br>X'06' – Initialize and mark<br>X'07' – Rewind<br>X'08' – Forward space record<br>X'09' – Backspace record<br>X'0A' – Find next header<br>X'0B' – Write header |

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 3 | 1 | | Hex 80 – Ignore ATTN key<br>Hex 00 – Honor ATTN key |
| 4-5 | 2 | Address | Buffer start address. |
| 6-7 | 2 | Binary | Buffer size. |
| 8-9 | 2 | Binary | Control information:<br>If Find – File number to find.<br>If Mark – Number of K-bytes to allocate. This field can be modified by the I/O supervisor when the entire file cannot be marked. |
| A-B | 2 | Address | I/O work area address. |
| C-D | 2 | Binary | Return code:<br>X'0000' – Successful completion<br>X'1B1D' – Command error<br>X'1B1E' – Machine error<br>X'1B1F' – Time-out<br>X'1B20' – Tape not mounted<br>X'1B21' – File protect<br>X'1B22' – CRC error<br>X'1B23' – Position error<br>X'1B24' – End of data<br>X'1C1B' – End of file<br>X'1C1C' – End of marked tape<br>X'1C1D' – End of tape<br>X'1C1E' – Device not attached<br>X'1C1F' – Device not selected |
| E-F | 2 | Binary | Number of files to be marked. |
| 10-11 | 2 | | Not used. |

The following bytes are for BASIC only:

| Hex Displacement | Length | Form | Description |
|---|---|---|---|
| 12-13 | 2 | Address | Current buffer pointer. |
| 14-15 | 2 | | Not used. |
| 16 | 1 | Binary | Flags:<br>Hex 80 – Device supports input<br>Hex 40 – Device supports output<br>Hex 20 – Device is tape<br>Hex 10 – Reserved<br>Hex 08 – File is open<br>Hex 04 – File is open for output<br>Hex 02 – File is in use for PUT or MAT PUT<br>Hex 01 – File is a PRINT file or MAT PRINT file |

## 5100 Internal Code Chart

Use the following chart to convert the hex values of 5100 internal code to their corresponding graphic:

| HEX | GRAPHIC | HEX | GRAPHIC | HEX | GRAPHIC | HEX | GRAPHIC |
|---|---|---|---|---|---|---|---|
| 00 | BLANK | 20 | 5 | 40 | ~ | 60 | ! |
| 01 | A | 21 | 6 | 41 | ↓ | 61 | ◊ |
| 02 | B | 22 | 7 | 42 | ∪ | 62 | ▲ |
| 03 | C | 23 | 8 | 43 | ω | 63 | ⊙ |
| 04 | D | 24 | 9 | 44 | ⊃ | 64 | ⊟ |
| 05 | E | 25 | / | 45 | ↑ | 65 | ⊠ |
| 06 | F | 26 | + | 46 | ⊂ | 66 | ⊓ |
| 07 | G | 27 | x | 47 | ∧ | 67 | ▽ |
| 08 | H | 28 | ← | 48 | ¨ | 68 | ◇ |
| 09 | I | 29 | [ | 49 | - | 69 | Φ |
| 0A | J | 2A | ] | 4A | < | 6A | ⊕ |
| 0B | K | 2B | , | 4B | ≤ | 6B | ∓ |
| 0C | L | 2C | . | 4C | = | 6C | ± |
| 0D | M | 2D | α | 4D | ≥ | 6D | ¬ |
| 0E | N | 2E | ⊥ | 4E | > | 6E | ¨ |
| 0F | O | 2F | ∩ | 4F | ≠ | 6F | & |
| 10 | P | 30 | L | 50 | ∨ | 70 | @ |
| 11 | Q | 31 | ∈ | 51 | \ | 71 | # |
| 12 | R | 32 | _ | 52 | — | 72 | $ |
| 13 | S | 33 | ∇ | 53 | ÷ | 73 | % |
| 14 | T | 34 | Δ | 54 | → | 74 | Ä |
| 15 | U | 35 | ι | 55 | ( | 75 | Ⓞ (Note) |
| 16 | V | 36 | ° | 56 | ) | 76 | Ö |
| 17 | W | 37 | ' | 57 | ; | 77 | Ü |
| 18 | X | 38 | □ | 58 | : | 78 | Å |
| 19 | Y | 39 | ∣ | 59 | Φ | 79 | Æ |
| 1A | Z | 3A | τ | 5A | ⊖ | 7A | Ŗ |
| 1B | 0 | 3B | ○ | 5B | ⊠ | 7B | Ñ |
| 1C | 1 | 3C | * | 5C | ⊕ | 7C | £ |
| 1D | 2 | 3D | ? | 5D | ≠ | 7D | ç |
| 1E | 3 | 3E | ρ | 5E | ↖ | 7E | ö |
| 1F | 4 | 3F | ⌈ | 5F | ⊡ | 7F | ä |

*Note:* OUT character

| HEX | GRAPHIC | HEX | GRAPHIC | HEX | GRAPHIC | HEX | GRAPHIC |
|---|---|---|---|---|---|---|---|
| 80 | Δ | A0 | 5 | C0 | ~ | E0 | - |
| 81 | A | A1 | 6 | C1 | ↓ | E1 | |
| 82 | B | A2 | 7 | C2 | ∪ | E2 | |
| 83 | C | A3 | 8 | C3 | ⊆ | E3 | |
| 84 | D | A4 | 9 | C4 | ⊃ | E4 | |
| 85 | E | A5 | / | C5 | ↑ | E5 | |
| 86 | F | A6 | + | C6 | ⊂ | E6 | |
| 87 | G | A7 | × | C7 | > | E7 | |
| 88 | H | A8 | ← | C8 | : | E8 | |
| 89 | I | A9 | | C9 | | E9 | |
| 8A | J | AA | | CA | | EA | |
| 8B | K | AB | , | CB | | EB | |
| 8C | L | AC | - | CC | | EC | |
| 8D | M | AD | ∅ | CD | | ED | |
| 8E | N | AE | ⊥ | CE | | EE | |
| 8F | O | AF | ∩ | CF | | EF | |
| 90 | P | B0 | | D0 | < | F0 | @ |
| 91 | Q | B1 | ≤ | D1 | / | F1 | # |
| 92 | R | B2 | ≡ | D2 | | F2 | $ |
| 93 | S | B3 | ∇ | D3 | | F3 | % |
| 94 | T | B4 | Δ | D4 | → | F4 | ^ |
| 95 | U | B5 | ∠ | D5 | ( | F5 | |
| 96 | V | B6 | ° | D6 | ) | F6 | |
| 97 | W | B7 | ' | D7 | ; | F7 | |
| 98 | X | B8 | | D8 | ∴ | F8 | & |
| 99 | Y | B9 | | D9 | ∈ | F9 | |
| 9A | Z | BA | | DA | ∋ | FA | |
| 9B | 0 | BB | 0 | DB | ⊗ | FB | |
| 9C | 1 | BC | * | DC | ⊕ | FC | £ |
| 9D | 2 | BD | ? | DD | ∨ | FD | |
| 9E | 3 | BE | ρ | DE | ∧ | FE | |
| 9F | 4 | BF | | DF | | FF | ¤ |

## Patch Command

The PATCH command applies internal machine fixes to the internal program or loads the tape recovery program. When the PATCH command is executed, the IMF (internal machine fix) program is read from tape. The user then has the option of selecting the following:

1. Copy the IMF tape
2. Load the IMFs
3. Displays the EC level of each ROS module

| Module | EC |
|---|---|
| 20 | 00 |
| 28 | 00 |
| 10 | 00 |
| 18 | 00 |
| 21 | 01 |
| 29 | 00 |
| 11 | 00 |
| 22 | 00 |
| 2A | 00 |
| 12 | 00 |
| 23 | 00 |
| 2B | 01 |
| 13 | 00 |
| 24 | 00 |
| 2C | 01 |
| 14 | 00 |
| 25 | 00 |
| 2D | 01 |
| 15 | 00 |
| 26 | 00 |
| 2E | 01 |
| 16 | 00 |
| 27 | 00 |
| 2F | 00 |
| 17 | 00 |

| Sequence Number[1] | ROS Module Card |
|---|---|
| 10, 11, 12, 13, 14, and 15 | C4 (BASIC ROS) |
| 16, 17, and 18 | E2 (ROS adapter) |
| 20, 21, 22, 23, and 24 | D2 (APL ROS 1) |
| 25, 26, 27, 28, and 29 | D4 (APL ROS 2) |
| 2A, 2B, 2C, 2D, 2E, and 2F | C2 (APL ROS 3) |

[1] Sequence numbers appear on the display screen during bring up.

4. Key-enter an IMF
5. Request end of job
6. Request the tape recovery program
7. Tape copy

Option 4 allows the user to enter an IMF from the keyboard. After the IMF is entered, it can be written to the IMF tape cartridge.

Refer to the BASIC or APL reference manuals for information about the use of the PATCH command.

## crocode Trouble Report

The microcode trouble report (MTR), form number Z150-0038, is designed to provide a uniform notification procedure for reporting problems encountered in IBM microcode functions. The following is a copy of an MTR and an explanation of the information required in the report.

**IBM**

MICROCODE TROUBLE REPORT

# M T R

INTERNAL USE ONLY

PREASSIGNED MTR SERIAL #

Ⓐ CUSTOMER NAME

Ⓑ CUSTOMER NO.

Ⓒ CUSTOMER MAILING ADDRESS

Ⓙ MTR SUBMITTED MO. DAY YR.

MTR IDENTITY

ASSIGNED BY MTR CONTROL

Ⓚ SEVERITY CODE 1 2 3 4

Ⓛ TYPE OF APPLICATION

| Ⓜ CPU | Controller Storage Size | Host Support | Host Support REL Level | Host Sys | Trans Control Unit Type | Prog Lev |
|---|---|---|---|---|---|---|
| | | | | | | |

Ⓝ MICROCODE IN ERROR/SUSPECTED IDENTITY AND L/C LEVEL

| ID NUMBER | EC LEVEL | REA/PATCH LEVEL |
|---|---|---|
| | | |

IBM REPRESENTATIVE — NAME AND ADDRESS

Ⓓ NAME

MAILING ADDRESS

Ⓟ MATERIAL SUBMITTED WITH MTR

| | | | |
|---|---|---|---|
| MICROCODE STORAGE DUMP | | VTOC LIST | |
| INTERPRETIVE EXEC DUMP | | MODULE E/C LEVEL LIST | |
| DISK DUMP | | OPER PANEL INDICATIONS | |
| CORE MAP | | ZAP LIST | |
| CUST SOURCE OBJECT | | NETWORK CONFIGURATION | |
| TEST DATA | | — INCLUDE MODEM TYPE, LINE SPEED, AND TYPE OF LINE (SWITCHED OR LEASED) AT HOST LOCATION. | |
| SYSTEM LOG | | | |
| DIAG OUTPUT | | | |

Ⓔ FE REGION | BRANCH OFF NO

Ⓕ WORLD TRADE COUNTRY NO | NAME

Ⓖ ITPS CODE

Ⓗ IBM — BRANCH OFFICE PHONE

OTHER:

Ⓠ SPECIAL ACTIVITIES

Ⓡ SYMPTOM / Ⓢ FAILURE KEYWORD

Ⓣ ABSTRACT

Ⓥ Error description text — Note variations between expected and actual output — differences from previous successful runs — suspected problem area — verify EC level of hardware as adequate for microcode — special configuration, teleprocessing, I/O switching, multi-systems, etc. — identify any bypass, circumvention, or relief given.

**Mail MTRs to:**
**IBM Corporation**
**5100 MTR Dept**
**Hwy 52 and 37th Street**
**Rochester, MN 55901**

DISTRIBUTION: 1,2 — MTR PROCESSING
4 — ORIGINATOR
3 — MTR PROCESSING

Ⓦ SUBMITTERS NAME (Print) AND SIGNATURE

ORIGINATOR IS:

| FE | GSD | OTHER |
|---|---|---|
| | | |

PAGE _____ OF _____

*Z150-0038-0 (U/M 025)*

| Item | Description | Explanation |
|---|---|---|
| Ⓐ | Customer name | Enter the full customer name. |
| Ⓑ | Customer number | This number can be obtained from the territory maintenance analysis (TMA) report or from the sales office orders and movements group. |
| Ⓒ | Customer mailing address | Use the complete customer mailing address. |
| Ⓓ | Name and mailing address | Enter the name (not the title) of the person responsible for handling MTR correspondence. Print the address of the branch office where MTR correspondence can be directed. This address is used for any follow-up required to resolve this MTR. |
| Ⓔ | Location numbers | Enter the branch office and region number. |
| Ⓕ | World Trade countries | World Trade country name and country number |
| Ⓖ | ITPS | Enter the ITPS (internal teleprocessing system) code for the responsible branch office. This code is listed in the branch office field directory telephone listing. |
| Ⓗ | Phone number | Branch office phone number or the MTR originator phone number. |
| Ⓘ | MTR submitted | Enter month/day/year |
| Ⓚ | Severity | Four levels (1 through 4) of severity codes are used to reflect the CE's appraisal of the customer's problem. The severity code is used to determine the priority in processing. |
| | Code 1 | Code 1 indicates that the customer is unable to use the microcode or that the problem results in a critical impact on his operations. In either case, an immediate solution is required. Contact Rochester field support immediately. |
| | Code 2 | The user is able to use the microcode, but the operation is severely restricted. |
| | Code 3 | The user is able to use the microcode with a limited function that is not critical to the overall operation. |
| | Code 4 | A circumvention has been found. However, the MTR is evaluated and action is taken as dictated by the problem. |
| | | *Note:* If this block is left blank, severity code 3 is assigned. |

| Item | Description | Explanation |
|---|---|---|
| (L) | Type of application | Enter the application type:<br>APL<br>BASIC<br>APL—Communications<br>BASIC—Communications |
| (M) | System type and configuration | CPU: Enter 5100.<br>Controller Storage Size: Enter 5100 read/write storage size.<br>Leave remaining boxes blank. |
| (N) | Microcode in error | Not applicable. |
| (P) | Materials submitted with the MTR | Materials listed on the MTR that are applicable to the 5100, and any other materials and information that the CE believes will be helpful in the resolution of the problem, should be submitted. |
| (Q) | Special activities | This block is left blank unless special instructions are available for its use. |
| (R) | Symptom code | AI |
| (S) | Failure keyword | APL<br>BASIC<br>COM (Communications) |
| (T) | Abstract | Up to 66 characters and spaces to describe the problem. |
| (V) | Error description text | The problem description should contain three major items:<br>• Conditions required to produce the problem<br>• External logic leading to the failure<br>• Identify any bypass, circumvention, or relief given<br><br>*Note:* If additional space is required, use additional MTR forms. Indicate the original customer number and page number on the additional forms. Attach the additional forms to the original form. |
| (W) | Submitter's name and signature | The person submitting the MTR should print his name above or below his signature. |

This page intentionally left blank.

## CE General Logic Probe (Part 453212)

The universal logic probe provides a visual indication of a line level. The probe can also be used to detect pulses and as a babysitter. (Refer to handbook that comes with probe.)

Probe UP and DOWN lights will momentarily flash on during power up if the probe is connected to its machine power source. Please ignore.

**Indicator Lights**

UP indicates an up level (+).
DOWN indicates a down level (-).

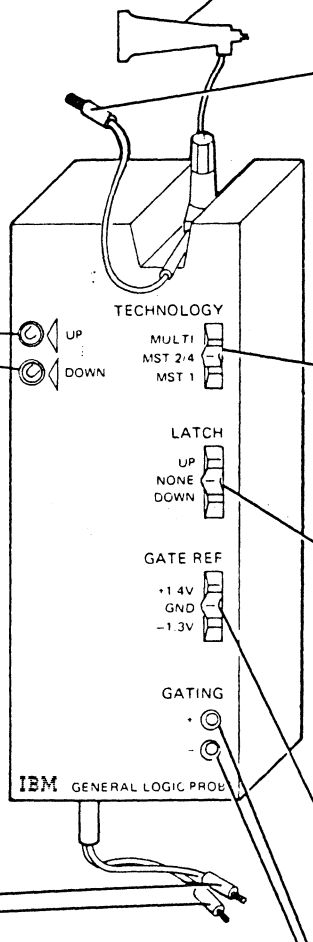
A pulsing line is indicated by both lights being on.

Both lights are off if the line level is from +1.0 Vdc to +2.0 Vdc for MULTI logic setting.

Safe Operating Ranges:

| | | |
|---|---|---|
| | MULTI | +60.0V |
| Logic | MST 2/4 | +14.0V |
| Selector | MST 1 | +14.0V |

Voltages greater than the above ranges will damage the probe.




**Power Leads**

RM5055

**CAUTION**
Improper connection of the power lead might cause the probe to malfunction.

Connect the black (-) lead to M2D08 (gnd). Connect the other (+) lead to M2D03. A voltage difference of 4V to 12V is needed to power the probe, with the black lead always the most negative. For the 5103 Printer, connect the black (-) lead to test point G6 (gnd) and connect the other (+) lead to test point V4 (+5V). For the 5106 Auxiliary Tape Unit, connect the black (-) lead to A1-A4D08 (gnd) and connect the other lead to A1-A4D03 (+5V).

*Note:* Power for the probe can be obtained from any of the above devices when probing any other device.

**Test Terminal**

The line being probed is connected to this terminal. (Various probes may be attached, other than the one shown, to aid in probing.) Do not use tip longer than 3 inches (76.2 mm).

**Ground Lead**

Connect this lead to any signal ground near the probe point. Do not use frame ground.

**CAUTION**
Improper indications result if this lead is not connected to signal ground. A maximum length of 4 inches (101.6 mm) can be used.

**Logic Selector (TECHNOLOGY)**

- 5100: MULTI

Selects the type of logic to be probed. Circuits probed in the 5100 Portable Computer require the MULTI setting.

**LATCH Switch**

- 5100: NONE

Allows the probe to be used as a babysitter. The up position allows latching the UP light on a positive pulse. The down position allows latching the DOWN light on a negative pulse. NONE position resets the lights and prevents any latching action. This position is used for most probing in the 5100 Portable Computer.

**GATE REF Volts Switch**

- 5100: GND

This switch affects only the gating terminals and is not required for probing the 5100 Portable Computer.

**GATING Terminals**

These terminals are not required for probing the 5100 Portable Computer.

## Numbering Systems

### HEX NUMBERING SYSTEM

Binary numbers require about three times as many positions as decimal numbers to express the equivalent number. This is not much of a problem for the computer. However, binary numbers are bulky for humans when talking or writing, or when communicating with a computer. A long string of 1's and 0's cannot be effectively transmitted from one individual to another. Some shorthand method is necessary. The hex numbering system fills this need.

Because of the simple relationship of hex to binary, numbers can be converted from one system to another by inspection. The base of the hex system is 16. This means there are 16 symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. The letters A, B, C, D, E, and F represent the decimal (base 10) values of 10, 11, 12, 13, 14, and 15, respectively.

Four binary positions are equivalent to one hex position. The following table shows the comparable values of the three numbering systems:

| Decimal | Binary | Hex |
|---|---|---|
| 0 | 0000 | 0 |
| 1 | 0001 | 1 |
| 2 | 0010 | 2 |
| 3 | 0011 | 3 |
| 4 | 0100 | 4 |
| 5 | 0101 | 5 |
| 6 | 0110 | 6 |
| 7 | 0111 | 7 |
| 8 | 1000 | 8 |
| 9 | 1001 | 9 |
| 10 | 1010 | A |
| 11 | 1011 | B |
| 12 | 1100 | C |
| 13 | 1101 | D |
| 14 | 1110 | E |
| 15 | 1111 | F |

At this point, all 16 hexadecimal symbols were used, and a carry to the next higher position of the number is necessary. For example:

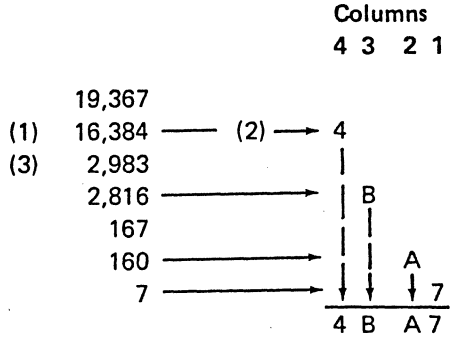
| Decimal | Binary | Hex |
|---|---|---|
| 16 | 0001 0000 | 10 |
| 17 | 0001 0001 | 11 |
| 18 | 0001 0010 | 12 |
| 19 | 0001 0011 | 13 |
| 20 | 0001 0100 | 14 |
| 21 | 0001 0101 | 15 |

—and so on—

| Hex Columns | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | |
| Hex = Dec | | Hex = Dec | | Hex = Dec | | Hex = Dec | | Hex = Dec | | Hex = Dec | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 1 | 1,048,576 | 1 | 65,536 | 1 | 4,096 | 1 | 256 | 1 | 16 | 1 | 1 | |
| 2 | 2,097,152 | 2 | 131,072 | 2 | 8,192 | 2 | 512 | 2 | 32 | 2 | 2 | |
| 3 | 3,145,728 | 3 | 196,608 | 3 | 12,288 | 3 | 768 | 3 | 48 | 3 | 3 | |
| 4 | 4,194,304 | 4 | 262,144 | 4 | 16,384 | 4 | 1,024 | 4 | 64 | 4 | 4 | |
| 5 | 5,242,880 | 5 | 327,680 | 5 | 20,480 | 5 | 1,280 | 5 | 80 | 5 | 5 | |
| 6 | 6,291,456 | 6 | 393,216 | 6 | 24,576 | 6 | 1,536 | 6 | 96 | 6 | 6 | |
| 7 | 7,340,032 | 7 | 458,752 | 7 | 28,672 | 7 | 1,792 | 7 | 112 | 7 | 7 | |
| 8 | 8,388,608 | 8 | 524,288 | 8 | 32,768 | 8 | 2,048 | 8 | 128 | 8 | 8 | |
| 9 | 9,437,184 | 9 | 589,824 | 9 | 36,864 | 9 | 2,304 | 9 | 144 | 9 | 9 | |
| A | 10,485,760 | A | 655,360 | A | 40,960 | A | 2,560 | A | 160 | A | 10 | |
| B | 11,534,336 | B | 720,896 | B | 45,056 | B | 2,816 | B | 176 | B | 11 | |
| C | 12,582,912 | C | 786,432 | C | 49,152 | C | 3,072 | C | 192 | C | 12 | |
| D | 13,631,488 | D | 851,968 | D | 53,248 | D | 3,328 | D | 208 | D | 13 | |
| E | 14,680,064 | E | 917,504 | E | 57,344 | E | 3,584 | E | 224 | E | 14 | |
| F | 15,728,640 | F | 983,040 | F | 61,440 | F | 3,840 | F | 240 | F | 15 | |
| 0123 | | 4567 | | 0123 | | 4567 | | 0123 | | 4567 | | |
| Byte | | | | Byte | | | | Byte | | | | |

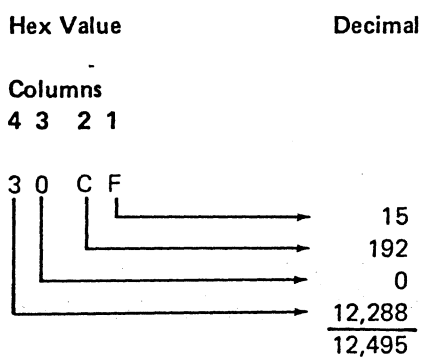
*From decimal to hex:* (1) Locate the largest decimal value in the table that will fit into the decimal number to be converted, (2) note its hex equivalent and hex column position, and (3) find the decimal remainder. Repeat the process on this and subsequent remainders.

Example: **Decimal Value** **Hex Equivalent**



*From hex to decimal:* Locate each hex digit in its corresponding column position and note the decimal equivalents. Add these to obtain the decimal value.

Example:



## Installation Procedures

### IBM 5100 PORTABLE COMPUTER

#### Prepower Check

Check with the customer to verify that the ac voltage outlet is grounded properly.

#### Power On Check

1. Make sure that the POWER switch is off.
2. Connect the mainline cord to the ac power outlet.
3. Turn the POWER switch on.
4. Observe the 5100 Portable Computer for signs of overheating or smoke. Turn off the POWER switch immediately if any abnormal conditions are noted.
5. Check that the fan is turning.
6. Use the MACHINE CHECKOUT, MAP 0900, to check the 5100 Portable Computer operation and performance.

#### Completion of Installation

The 5100 Portable Computer serial number is engraved on the rear of the base.

Fill out the IR form and report the installation according to local procedures.

#### Cover Cleaning

Use a mild soap or isopropyl alcohol (part 2200200). **DO NOT** use IBM cleaning fluid.

#### IBM 5100 Portable Computer Specifications

*Dimensions:*

| | F | S | H |
|---|---|---|---|
| Inches | 17.5 | 24.0 | 8.0 |
| Millimeters | 445 | 609.6 | 203 |

*Weight:* 50 pounds (24 kg)

*Heat Output/hr:* 780 Btu

*Power Requirements:*

| AC Voltage (single phase) | Hertz | kVA |
|---|---|---|
| 100 V | 50 | 0.4 |
| 100 V | 60 | 0.4 |
| 115 V | 60 | 0.4 |
| 220 V | 50 | 0.4 |
| 235 V | 50 | 0.4 |

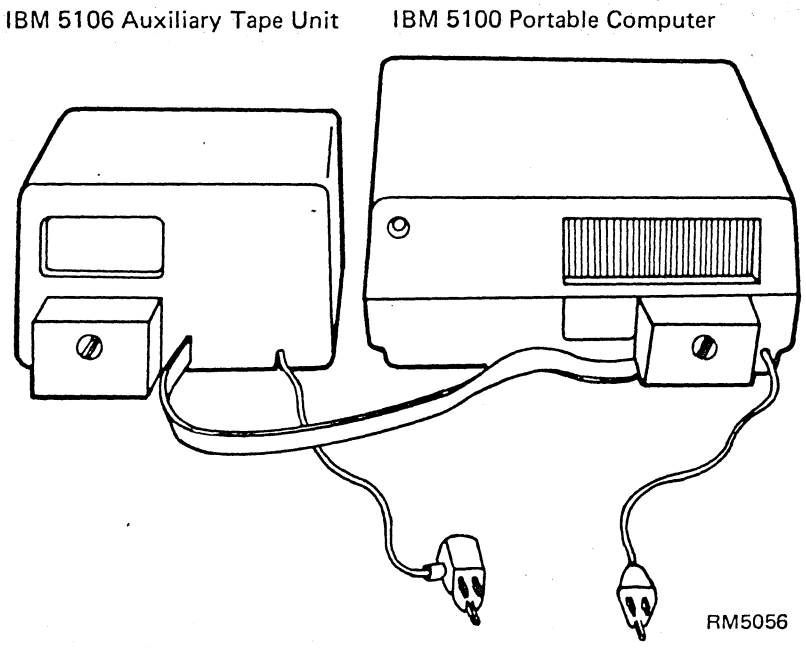
*Power Cord Specifications (220, 235):*

Cable PD – 0.4 ± 0.015 inches (10.16 ± 0.38 mm)
Shields – none
Conductors – 3
Conductor size – 16 Awg (1.3 $mm^2$)

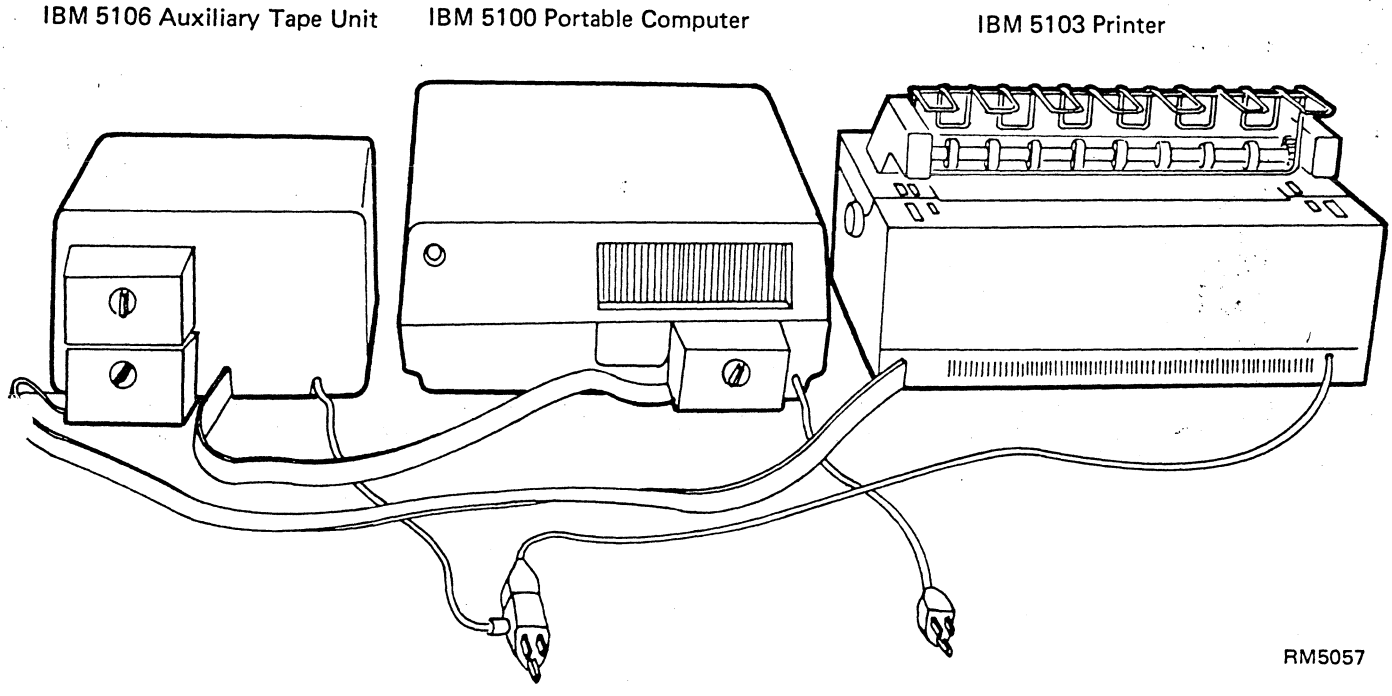
### IBM 5103 PRINTER

Refer to the IBM 5103 Printer Maintenance Information Manual, SY31-0414 for the 5103 installation procedures.

## IBM 5106 Auxiliary Tape Unit to the IBM 5100 Portable Computer



**IBM 5106 Auxiliary Tape Unit and IBM 5103 Printer to the IBM 5100 Portable Computer**



### Prepower Check

Check with the customer to verify that the ac voltage outlet is grounded properly.

### Cable Installation

1. Turn the 5100 Portable Computer POWER switch off.
2. Attach the line terminator to the cable connectors on the back of the 5106 Auxiliary Tape Unit.
3. Attach the external I/O interface cable assembly to the back of the 5100.

### Power On Check

1. Make sure that the POWER switch is off.
2. Connect the mainline cord to the ac power outlet.
3. Turn the POWER switch on.
4. Observe the 5100 Portable Computer for signs of overheating or smoke. Turn off the POWER switch immediately if any abnormal conditions are noted.
5. Check that the fan is turning.
6. Use the Auxiliary Tape MAP 0850 to check the 5100 operation and performance.
7. Replace the cover.

### Completion of Installation

The 5106 Auxiliary Tape serial number is on a tag on the bottom of the base. Refer to 208.

1. Place the MAPs in the separate binder provided inside the 5100 Portable Computer MLM binder. Place the Maintenance Information Manual in the 5100 MLM binder. Insert the MAPs binder into the 5100 MLM binder by inserting its back cover into the slot in the front inside cover of the 5100 MLM binder. Place the 5100 Parts Catalog into the 5100 MLM binder.
2. Fill out the IR form and report the installation according to local procedures. Place the IR form carbon copy into a pocket in the 5100 MLM binder.

### Cover Cleaning

Use a mild soap or isopropyl alcohol (part 2200200). **DO NOT** use IBM cleaning fluid.

### Safety

Remove all electrical power from the 5106 Auxiliary Tape Unit by unplugging the mainline cord.

### IBM 5106 Auxiliary Tape Unit Specifications

*Dimensions*

| | F | S | H |
|---|---|---|---|
| Inches | 13.25 | 9.85 | 7.1 |
| Millimeters | 336.5 | 250 | 180 |

*Weight:* 18 pounds (8 Kg)

*Heat Output/hr:* 130 Btu

*Power Requirements:*

| AC Voltage (single phase) | Hertz | kVA |
|---|---|---|
| 100 V | 50 | 0.1 |
| 100 V | 60 | 0.1 |
| 115 V | 60 | 0.1 |
| 220 V | 50 | 0.1 |
| 235 V | 50 | 0.1 |

*Power Cord Specifications (220, 235):*

Cable OD – 0.4 ± 0.015 inches (10.16 ± 0.38 mm)
Shields – none
Conductors – 3
Conductor size – 16 Awg (1.3 $mm^2$)

*Operating Environment:*

| | |
|---|---|
| Temperature | 60° F to 90° F (15.6° C to 32.2° C) |
| Relative humidity | 8% to 80% |
| Maximum wet bulb | 73 ° F (22.8° C) |

## Microinstructions

I/O control and the high level languages (APL and BASIC) are implemented with microinstructions in read/write storage and executable ROS. All 5100 Portable Computer microinstructions are a halfword (2 bytes). The first 4 bits of the halfword is the op code. The meaning of the remaining 12 bits depends on the op code (refer to *Formats*). Some op codes have a modifier (bits 12-15) that expands the number of microinstructions beyond 16.

Op codes, along with their modifier if there is one, are classified according to function as follows:

I/O microinstructions
Fetch and store microinstructions
Register microinstructions
Arithmetic register microinstructions
Logical register microinstructions
Jump microinstructions

*Formats*

This chart shows the formats of the microinstructions arranged according to operation codes. The microinstructions mnemonics are given at the right for each op code.

| Op Code | Second Hex Digit | Third Hex Digit | Fourth Hex Digit | Instruction Mnemonic |
|---|---|---|---|---|
| 0 | Rx | Ry | AM | ADD, ADDS1, ADDS2, AND, GETA, GETR, HTL, LTH, MOVE, MVM1, MVM2, MVP1, MVP2, ORB, SUB, XOR |
| 1 | DA | Command | | CTL |
| 2 | Rx | Address | | LDHD |
| 3 | Rx | Address | | STHD |
| 4 | DA | Ry | M | PUTB |
| 5 | Rx | Ry | M | STHI |
| 6 | Rx | Ry | M | LDBI |
| 7 | Rx | Ry | M | STBI |
| 8 | Rx | Data | | EMIT |
| 9 | Rx | Mask | | CLRI |
| A | Rx | Data | | ADDI |
| B | Rx | Mask | | SETI |
| C | Rx | Ry | JM | ALL JUMPS |
| D | Rx | Ry | M | LDHI |
| E | DA | Ry | SM | ROTR, SHFTR, GETB, GETRB |
| F | Rx | Data | | SUBI |

*Notes:*

Rx = Register specified by the second hex digit in the microinstruction
Ry = Register specified by the third hex digit in the microinstruction
DA = Device address
AM = ALU modifier
Command = Byte of control data
Address = Read/write storage data address
M = Normal modifier
Data = Byte of immediate data
Mask = Byte of mask data
JM = Jump modifier
SM = Special modifier

*Mnemonics*

This chart gives the microinstruction mnemonics in alphabetical order, the meaning of the mnemonic, the corresponding op code, the modifier (if any), and the type of microinstruction.

| Mnemonic | Microinstruction Name | Op Code | Modifier | Microinstruction Type |
|---|---|---|---|---|
| ADD | Add | 0 | 8 | Arithmetic reg |
| ADDI | Add immediate | A | – | Arithmetic reg |
| ADDS1 | Add special 1 | 0 | A | Arithmetic reg |
| ADDS2 | Add special 2 | 0 | B | Arithmetic reg |
| AND | And | 0 | 5 | Logical reg |
| CLRI | Clear immediate | 9 | – | Logical reg |
| CTL | Control | 1 | – | I/O |
| EMIT | Emit byte | 8 | – | Logical reg |
| GETA | Get | 0 | F | I/O |
| GETB | Get byte | E | SM[1] | I/O |
| GETR | Get to register | 0 | E | I/O |
| GETRB | Get byte to register | E | SM[1] | I/O |
| HTL | High to low | 0 | C | Register |
| JALL | Jump all ones | C | 4 | Jump |
| JALLM | Jump all masked | C | 5 | Jump |
| JEQ | Jump equal | C | 2 | Jump |
| JHAM | Jump high all masked | C | 7 | Jump |
| JHE | Jump high or equal | C | 9 | Jump |
| JHI | Jump high | C | 8 | Jump |
| JHL | Jump high or low (not equal) | C | A | Jump |
| JHSNM | Jump high some bit not masked | C | F | Jump |
| JLE | Jump low or equal | C | 0 | Jump |
| JLO | Jump low | C | 1 | Jump |
| JNO | Jump no ones | C | 3 | Jump |
| JNOM | Jump no ones masked | C | 6 | Jump |
| JSB | Jump some bits | C | B | Jump |
| JSM | Jump some masked | C | E | Jump |
| JSN | Jump some not ones | C | C | Jump |
| JSNM | Jump some not masked | C | D | Jump |
| LDBI | Load byte indirect[2] | 6 | M[1] | Fetch and store |
| LDHD | Load halfword direct[2] | 2 | – | Fetch and store |
| LDHI | Load halfword indirect[2] | D | – | Fetch and store |
| LTH | Low to high | 0 | D | Register |
| MOVE | Move register | 0 | 4 | Register |
| MVM1 | Move minus 1 | 0 | 1 | Register |
| MVM2 | Move minus 2 | 0 | 0 | Register |

[1] See the description of *Microinstructions* in this section.

[2] Direct means that the read/write storage address is in the microinstruction itself and indirect means that the read/write storage address is in a register.

| Mnemonic | Microinstruction Name | Op Code | Modifier | Microinstruction Type |
|---|---|---|---|---|
| MVP1 | Move plus 1 | 0 | 2 | Register |
| MVP2 | Move plus 2 | 0 | 3 | Register |
| ORB | Or byte | 0 | 6 | Logical reg |
| PUTB | Put byte | 4 | M[1] | I/O |
| ROTR | Rotate register | E | SM[1] | Logical reg |
| SETI | Set immediate | B | — | Logical reg |
| SHFTR | Shift right | E | C | Logical reg |
| STBI | Store byte indirect[2] | 7 | M[1] | Fetch and store |
| STHD | Store halfword direct[2] | 3 | — | Fetch and store |
| STHI | Store halfword indirect[2] | 5 | M[1] | Fetch and store |
| SUB | Subtract | 0 | 9 | Arithmetic reg |
| SUBI | Subtract immediate | F | — | Arithmetic reg |
| XOR | Exclusive or | 0 | 7 | Logical reg |

[1] See the description of *Microinstructions* in this section.
[2] Direct means that the read/write storage address is in the microinstruction itself and indirect means that the read/write storage address is in a register.

*Descriptions*

*I/O Microinstructions and Data Tables:* The I/O microinstructions are used to communicate with the I/O devices. They send and receive data, and perform various control functions.

Up to 16 device addresses are selected directly with a device address (DA) in the microinstruction. In addition, the capacity of some device addresses is increased from one I/O device per device address to eight I/O devices per device address by using subdevice addressing.

When an I/O operation to a device having a subdevice address is performed, the subdevice must first be selected. This is accomplished by resetting all subdevices for the pertinent device and then selecting the subdevice (for example, refer to the Control Command Data Table, device address 1). All subsequent I/O microinstructions for the pertinent device access the selected subdevice. Devices with subdevice addressing capabilities have device addresses 1, B, C, D, and E.

CTL DA, Command (op code 1): The command data (bits 8-15 of the microinstruction) is put on bus out (bits 0-7) for the device defined by the device address (DA). A control strobe occurs when command data is on bus out.

**Control Command Data Table**

| Device Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition |
|---|---|---|---|---|
| 0 | Controller | 8 | 0 | No action |
| | | | 1 | Reset controller errors |
| | | 9 | 00 | Not valid |
| | | 10 | 01 | Allow interrupt level changes |
| | | | 10 | Block interrupt level changes |
| | | | 11 | No action |
| | | 11 | 0 | Disable cycle steal (display off and IN PROCESS light on) |
| | | | 1 | Enable cycle steal (display on and IN PROCESS light off) |
| | | 12 | 0 | No action |
| | | | 1 | Not used |
| | | 13 | 0 | State transition, from BUP (BRING UP PROGRAM) / ROS / Read/write storage; to ROS / Read/write storage / ROS |
| | | | 1 | No action |
| | | 14 | 0 | No action |
| | | | 1 | Not used |
| | | 15 | 0 | No action |
| | | | 1 | Not used |
| 1[1] | Nonexecutable ROS | 8 | 0 | No action |
| | | | 1 | Reserved |
| | | 9 | 0 | No action |
| | | | 1 | Not used |
| | | 10 | 0 | No action |
| | | | 1 | Not used |
| | | 11 | 0 | No action |
| | | | 1 | Not used |
| | | 12 | 0 | Reset select APL |
| | | | 1 | Select APL |
| | | 13 | 0 | Reset select BASIC or common |
| | | | 1 | Select BASIC or common |
| | | 14 | 0 | No action |
| | | | 1 | No action |
| | | 15 | 0 | No action |
| | | | 1 | Reserved |

[1] Device has subdevice addressing capability. See the I/O microinstructions description under *Microinstructions* in this section.

**Control Command Data Table (Continued)**

| Device Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition |
|---|---|---|---|---|
| 2 | Not assigned | | | |
| 3 | Not assigned | | | |
| 4 | Keyboard and control panel switches | 8 | 0 | No action |
| | | | 1 | Not used |
| | | 9 | 0 | Reset interrupt |
| | | | 1 | No action |
| | | 10 | 0 | No action |
| | | | 1 | Not used |
| | | 11 | 0 | Keyboard lock (not used). See *Keyboard* in this section |
| | | | 1 | Enable keyboard interrupt |
| | | 12 | 0 | No action |
| | | | 1 | Not used |
| | | 13 | 0 | No action |
| | | | 1 | Not used |
| | | 14 | 0 | Set repeat action function |
| | | | 1 | No action |
| | | 15 | 0 | No action |
| | | | 1 | Disable keyboard interrupt |
| 5 | Printer | 8 | 0 | Disable (selection) unless bits 9 & 10 & 11 = 110, then ROS address bit 0 on |
| | | | 1 | Enable (selection) unless bits 9 & 10 & 11 = 110, then ROS address bit 1 off |
| | | 9 & 10 & 11 | 000 | Both motors on (bit 0 = 1 also) |
| | | | 001 | (Select) forms |
| | | | 010 | (Select) print |
| | | | 011 | Not used |
| | | | 100 | (Select) not ready interrupt |
| | | | 101 | (Select) timer unless 14 & 15 = 00, then allow ROS addressing enable-disable via bit 0 above |
| | | | 110 | ROS addressing indicator |
| | | | 111 | Both motors off (bit 0 = 1 also) |
| | | 12 | 0 | If 9 & 10 & 11 = 001 or 010 select not A<br>If 9 & 10 & 11 = 110 ROS address bit 1 on, otherwise no action |
| | | | 1 | If 9 & 10 & 11 = 001 or 010 select latch A<br>If 9 & 10 & 11 = 110 ROS address bit 1 off, otherwise ROS control |
| | | 13 | 0 | If 9 & 10 & 11 = 001 or 010 select not B<br>If 9 & 10 & 11 = 110 ROS address bit 2 on, otherwise no action |
| | | | 1 | If 9 & 10 & 11 = 001 or 010 select latch B<br>If 9 & 10 & 11 = 110 ROS address bit 2 off, otherwise not used |
| | | 14 & 15 | 00 | (Print or forms) go latch or ROS control (14 & 15 = 00 for all ROS control) |
| | | | 01 | (Select) interrupts |
| | | | 10 | Reset interrupts |
| | | | 11 | (Select) (print or forms) motor latches |

Example: Hex 51 = B 01010001 = Disable timer interrupts
Hex 93 = B 10010011 = Enable forms motor latches not A and not B

**Control Command Data Table (Continued)**

| Device Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition | |
|---|---|---|---|---|---|
| 6 | Not assigned | | | | |
| 7 | Not assigned | | | | |
| 8 | Expansion feature | | | Comm Adapter | Serial I/O Adapter |
| | | 8 | 0 | Disable timer interrupt | Disable timer interrupt |
| | | | 1 | Enable timer interrupt | Enable timer interrupt |
| | | 9 | 0 | No action | No action |
| | | | 1 | Not assigned | Not assigned |
| | | 10 | 0 | Select transmit mode | Select transmit mode |
| | | | 1 | Select receive mode | Select receive mode |
| | | 11 | 0 | No action | No action |
| | | | 1 | Reset timer interrupt | Reset timer interrupt |
| | | 12 | 0 | No action | Select serial I/O adapter |
| | | | 1 | Not assigned | No action |
| | | 13 | 0 | No action | No action |
| | | | 1 | Long space interrupt reset | Long space interrupt reset |
| | | 14 | 0 | Terminal not ready | Not SIO received line signal detector |
| | | | 1 | Terminal ready | SIO received line signal detector |
| | | 15 | 0 | Disable start bit test | Disable start bit test |
| | | | 1 | Enable start bit test | Enable start bit test |
| 9 | Not assigned | | | | |
| A | Not assigned | | | | |
| B[1] | Not assigned | | | | |
| C[1] | Not assigned | | | | |
| D[1] | Not assigned | | | | |
| E[1] | Tape unit | 8 | 0 | Run | |
| | | | 1 | Stop | |
| | | 9 | 0 | Forward | |
| | | | 1 | Reverse | |
| | | 10 | 0 | Data write track select | |
| | | | 1 | Format write track selec | |
| | | 11 | 0 | Write | |
| | | | 1 | Read | |
| | | 12 | 0 | Format track erase | |
| | | | 1 | Not format track erase | |
| | | 13 | 0 | Data track erase | |
| | | | 1 | Not data track erase | |
| | | 14 | 0 | Diagnostic mode | |
| | | | 1 | Not diagnostic mode | |
| | | 15 | 0 | Enable interrupt | |
| | | | 1 | Disable interrupt | |

[1] Device has subdevice addressing capability. See the I/O microinstructions description under *Microinstructions* in this section.

ontrol Command Data Table (Continued)

| Device Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition |
|---|---|---|---|---|
| F | All I/O | 8 | 0 | No action |
| | | | 1 | Reset expansion feature (DA = 8) |
| | | 9 | 0 | No action |
| | | | 1 | Reset tape (DA = E) |
| | | 10 | 0 | No action |
| | | | 1 | Reset keyboard (DA = 4) |
| | | 11 | 0 | No action |
| | | | 1 | Reset printer (DA = 5) |
| | | 12 | 0 | No action |
| | | | 1 | Enable cycle steal for display (unblank display) |
| | | 13 | 0 | No action |
| | | | 1 | Reset (DA = B) |
| | | 14 | 0 | No action |
| | | | 1 | Reset (DA = C) |
| | | 15 | 0 | No action |
| | | | 1 | Reset (DA = D) |

GETA DA, Ry (op code 0): A data byte is transferred from an I/O device designated by DA to the controller on bus in. The contents of the data register (Ry) have a value added to it depending on the data received from the device:

| Bus In Bits 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Quantity Added |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | X | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | X | 2 |
| 1 | 1 | 1 | 1 | 1 | 0 | X | X | 4 |
| 1 | 1 | 1 | 1 | 0 | X | X | X | 6 |
| 1 | 1 | 1 | 0 | X | X | X | X | 8 |
| 1 | 1 | 0 | X | X | X | X | X | A |
| 1 | 0 | X | X | X | X | X | X | C |
| 0 | X | X | X | X | X | X | X | E |

The Xs are don't cares.
Bus in is not parity checked on this microinstruction.
'Get strobe' is activated on this microinstruction.

GETB DA, Ry, SM (op code E): A data byte is transferred from an I/O device designated by DA to a storage location designated by the storage address register (Ry). The data byte is sent from the device on bus in. The indirect address in the storage address register is modified after the microinstruction is executed.

**Modifier**

| | |
|---|---|
| 0 | Plus 1 |
| 1 | Plus 2 |
| 2 | Plus 3 |
| 3 | Plus 4 |
| 4 | Minus 1 |
| 5 | Minus 2 |
| 6 | Minus 3 |
| 7 | Minus 4 |
| 8, 9, A, B | No change |
| C, D, E, F | This is the GETRB instruction |

'Get strobe' and 'op code E' are activated during this microinstruction to signal the I/O device that this is a GETB microinstruction.

ΓR DA, Ry (op code 0): A data byte is transferred from an I/O device designated by DA, via bus in, to the low order byte of data register (Ry).

The 'get strobe' is activated on this microinstruction.

**GETR Data Table**

| Device Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition |
|---|---|---|---|---|
| 0 | Controller | | | Not used |
| 1[1] | Nonexecutable ROS | | | Use two GETB microinstructions to get even-odd bytes |
| | | 8 | | ROS data byte bit 0 |
| | | 9 | | ROS data byte bit 1 |
| | | 10 | | ROS data byte bit 2 |
| | | 11 | | ROS data byte bit 3 |
| | | 12 | | ROS data byte bit 4 |
| | | 13 | | ROS data byte bit 5 |
| | | 14 | | ROS data byte bit 6 |
| | | 15 | | ROS data byte bit 7 |
| 2 | Not assigned | | | |
| 4 | Keyboard and control panel switches | 8 | 0 | Keyboard data bit 0 |
| | | | 1 | Not keyboard data bit 0 |
| | | 9 | 0 | Keyboard data bit 1 |
| | | | 1 | Not keyboard data bit 1 |
| | | 10 | 0 | Keyboard data bit 2 |
| | | | 1 | Not keyboard data bit 2 |
| | | 11 | 0 | Keyboard data bit 3 |
| | | | 1 | Not keyboard data bit 3 |
| | | 12 | 0 | Keyboard data bit 4 |
| | | | 1 | Not keyboard data bit 4 |
| | | 13 | 0 | Keyboard data bit 5 |
| | | | 1 | Not keyboard data bit 5 |
| | | 14 | 0 | Keyboard data bit 6 |
| | | | 1 | Not keyboard data bit 6 |
| | | 15 | 0 | Keyboard data bit 7 |
| | | | 1 | Not keyboard data bit 7 |

[1] Device has subdevice addressing capability. See the I/O microinstructions description under *Microinstructions* in this section.

GETR Data Table (Continued)

| Device Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition |
|---|---|---|---|---|
| 5 | Printer | | | If Ry is even, status byte A |
| | | 8 | | Print emitter latch 3 |
| | | 9 | | Print emitter latch 2 |
| | | 10 | | Print emitter latch 1 |
| | | 11 | | Wire check or not ready |
| | | 12 | | Forms emitter B |
| | | 13 | | Forms emitter A |
| | | 14 | | Not end of forms |
| | | 15 | | Left margin switch or not ready |
| | | | | If Ry is odd, status byte B |
| | | 8 | | Print motor latch B (0 = not B) |
| | | 9 | | Print motor latch A (0 = not A) |
| | | 10 | | Print emitter interrupt |
| | | 11 | | Not ready interrupt |
| | | 12 | | Forms motor latch B (0 = not B) |
| | | 13 | | Forms motor latch A (0 = not A) |
| | | 14 | | Not used |
| | | 15 | | 1 or 3 ms timer interrupt |
| 6 | Not assigned | | | |
| 7 | | | | |
| 8 | Expansion feature | | | Comm Adapter: Not used; Serial I/O Adapter: Not used |
| 9 | Not used | | | |
| A | | | | |
| B[1] | | | | |
| C[1] | | | | |
| D[1] | | | | |
| E[1] | Tape unit | 8 | | Tape read data in 0 |
| | | 9 | | Tape read data in 1 |
| | | 10 | | Tape read data in 2 |
| | | 11 | | Tape read data in 3 |
| | | 12 | | Tape read data in 4 |
| | | 13 | | Tape read data in 5 |
| | | 14 | | Tape read data in 6 |
| | | 15 | | Tape read data in 7 |
| F | All I/O | | | Not used |

[1] Device has subdevice addressing capability. See the I/O microinstructions description under *Microinstructions* in this section.

GETRB DA, Ry (op code E): A data byte is transferred from an I/O device designated by DA, via bus in, to the low order byte of the register Ry.

If DA = 0, the microinstruction is SHFTR or ROTR.

'Op code E' is activated on this microinstruction but 'get strobe' is not. (Modifier C, D, E, or F selects GETRB.)

**GETRB Data Table**

| Device Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition |
|---|---|---|---|---|
| 0 | Controller | | | See SHFTR and ROTR microinstructions |
| 1[1] | Nonexecutable ROS | | | Use two GETRB microinstructions to get even-odd bytes |
| | | 8 | | Address bits 0/8 |
| | | 9 | | Address bits 1/9 |
| | | 10 | | Address bits 2/A |
| | | 11 | | Address bits 3/B |
| | | 12 | | Address bits 4/C |
| | | 13 | | Address bits 5/D |
| | | 14 | | Address bits 6/E |
| | | 15 | | Address bits 7/F |
| 2 | Not assigned | | | |
| 3 | " " | | | |
| 4 | Keyboard and control panel switches | 8 | | Not used |
| | | 9 | 0 | APL switch on |
| | | | 1 | BASIC switch on |
| | | 10-15 | | Not used |
| 5 | Printer | | | Not used |
| 6 | Not assigned | | | |
| 7 | " " | | | |
| 8 | Expansion feature | | | Comm Adapter / Serial I/O Adapter |
| | | 8 | | Not used / Not used |
| | | 9 | | Clear to send / Clear to send |
| | | 10 | | Data set ready / Data set ready |
| | | 11 | | Timer interrupt / Timer interrupt |
| | | 12 | | Long space interrupt / Long space interrupt |
| | | 13 | | Not used / Not used |
| | | 14 | | Not used / Not used |
| | | 15 | | Received data bit / Received data bit |

[1] Device has subdevice addressing capability. See the I/O microinstructions description under *Microinstructions* in this section.

GETRB Data Table (Continued)

| Device Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition |
|---|---|---|---|---|
| 9 | Not used | | | |
| A | " " | | | |
| B[1] | " " | | | |
| C[1] | " " | | | |
| D[1] | " " | | | |
| E[1] | Tape unit | 8 | 1 | End of tape (EOT) |
| | | 9 | 1 | No DA E response |
| | | 10 | 1 | Select magnet active (drive running) |
| | | 11 | 1 | Cartridge in place |
| | | 12 | 1 | Erase active (either channel 0 or 1) |
| | | 13 | 1 | Source active (LED and erase coils OK) |
| | | 14 | 1 | File protect (do not write) |
| | | 15 | 0 | Beginning of tape (BOT) |
| F | All I/O | | | Not used |

[1] Device has subdevice addressing capability. See the I/O microinstructions description under *Microinstructions* in this section.

P 3 DA, Ry, M (op code 4): A data byte from storage (indirectly addressed by the storage address register Ry) is sent, via bus out, to the device designated by DA.

The indirect address in the storage address register (Ry) is modified as follows:

| Modifier | |
|---|---|
| 0 | Plus 1 |
| 1 | Plus 2 |
| 2 | Plus 3 |
| 3 | Plus 4 |
| 4 | Minus 1 |
| 5 | Minus 2 |
| 6 | Minus 3 |
| 7 | Minus 4 |
| Greater than 7 | No change |

'Put strobe' is active on this microinstruction to signal the device that this is a PUT microinstruction.

**PUTB Data Table**

| evice Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition |
|---|---|---|---|---|
| 0 | Controller | | | Not used |
| 1[1] | Nonexecutable ROS | | | Halfword address (two consecutive PUTB microinstructions) |
| | | 8 | | ROS halfword address bit 0/8 |
| | | 9 | | ROS halfword address bit 1/9 |
| | | 10 | | ROS halfword address bit 2/A |
| | | 11 | | ROS halfword address bit 3/B |
| | | 12 | | ROS halfword address bit 4/C |
| | | 13 | | ROS halfword address bit 5/D |
| | | 14 | | ROS halfword address bit 6/E |
| | | 15 | | ROS halfword address bit 7/F |
| 2 | Not assigned | | | |
| 3 | Not assigned | | | |
| 4 | Keyboard and panel switches | | | Not used |

[1] Device has subdevice addressing capability. See the I/O microinstructions description under *Microinstructions* in this section.

**PUTB Data Table (Continued)**

| Device Address Bits 4-7 | Device Name | Bits 8-15 | State of Bit | Definition | |
|---|---|---|---|---|---|
| 5 | Printer | 8 | | Print data bit 0 | |
| | | 9 | | Print data bit 1 | |
| | | 10 | | Print data bit 2 | |
| | | 11 | | Print data bit 3 | |
| | | 12 | | Print data bit 4 | |
| | | 13 | | Print data bit 5 | |
| | | 14 | | Print data bit 6 | |
| | | 15 | | Print data bit 7 | |
| 6 | Not assigned | | | | |
| 7 | Not assigned | | | | |
| 8 | Expansion | | | Comm Adapter | Serial I/O Adapter |
| | feature | 8 | 0 | No data on 'xmit data' line | No data on 'xmit data' line |
| | | | 1 | Data on 'xmit data' line | Data on 'xmit data' line |
| | | 9 | 0 | Not Used | Not used |
| | | | 1 | Not used | Not used |
| | | 10 | 0 | Not used | Not used |
| | | | 1 | Not used | Not used |
| | | 11 | 0 | Not used | No action |
| | | | 1 | Not used | Allows 'request to send' to be changed |
| | | 12 | 0 | Not used | Resets request to send |
| | | | 1 | Not used | Sets request to send |
| | | 13 | 0 | No action | No action |
| | | | 1 | Allows rate data to be loaded | Allows rate data to be loaded |
| | | 14 | 0 | No action | No action |
| | | | 1 | Rate data | Rate data |
| | | 15 | 0 | 134.5 bps[1] | Not used |
| | | | 1 | 300.0 bps[1] | Not used |
| 9 | Not assigned | | | | |
| A | | | | | |
| B[2] | | | | | |
| C[2] | | | | | |
| D[2] | | | | | |
| E[2] | Tape unit | 8-9 | | Not used | |
| | | 10 | 0 | Byte mode | |
| | | | 1 | Bit mode | |
| | | 11-13 | | Not used | |
| | | 14 | 0 | First transition | |
| | | | 1 | No action | |
| | | 15 | 0 | + serial write data out | |
| | | | 1 | No action | |
| F | All I/O | | | Not used | |

[1] Expansion feature card, P/N 1607004, uses bits 8 and 15 only. Bit 15 is not used by later expansion feature cards (see bits 13 and 14).
[2] Device has subdevice addressing capability. See the I/O microinstructions description under *Microinstructions* in this section.

*etch and Store Microinstructions:*

LDBI, Rx, Ry, M (op code 6): The byte of data at the storage location designated by the address register (Ry) is read into the low order byte of the data register (Rx). Then the address in the address register (Ry) is modified as shown by the following chart:

| Modifier | |
|---|---|
| 0 | Plus 1 is added to the contents of Ry. |
| 1 | Plus 2 is added to the contents of Ry. |
| 2 | Plus 3 is added to the contents of Ry. |
| 3 | Plus 4 is added to the contents of Ry. |
| 4 | Minus 1 is subtracted from the contents of Ry. |
| 5 | Minus 2 is subtracted from the contents of Ry. |
| 6 | Minus 3 is subtracted from the contents of Ry. |
| 7 | Minus 4 is subtracted from the contents of Ry. |
| Greater than 7 | Ry – No change. |

LDHD Rx, ADDRESS (op code 2): A halfword from the location defined by the halfword address is read into the data register (Rx).

DHI Rx, Ry, M (op code D): The halfword located at the address in the address register Ry is read into the data register (Rx). Then the address in the address register (Ry) is modified as shown by the chart following the LDBI microinstruction.

STBI Rx, Ry, M (op code 7): The low order byte in the data register (Rx) is stored in the location designated by the address register (Ry). Then the address in the address register is modified as shown by the chart following the LDBI microinstruction.

STHD Rx, ADDRESS (op code 3): The halfword in the data register (Rx) is stored in the location at the halfword address defined by bits 8-15 of the microinstruction.

STHI Rx, Ry, M (op code 5): The halfword in the data register (Rx) is stored at the location specified by the address register (Ry). Then the address register (Ry) is modified as indicated in the chart following the LDBI microinstruction.

*Register Operation Microinstructions:*

HTL Rx, Ry (op code 0): The high order byte of register Ry is moved to the low order byte of register Rx. Register Ry is not changed unless Ry and Rx are designated as the same register. (MOD = C)

LTH Rx, Ry (op code 0): The low order byte of register Ry is moved to the high order byte of register Rx. Register Ry is not changed unless Ry and Rx are designated as the same register. (MOD = D)

MOVE Rx, Ry (op code 0): The halfword in register Ry is moved to register Rx and Ry is not changed. (MOD = 4)

MVM1 Rx, Ry (op code 0): The halfword in register Ry is moved to register Rx and the Rx is decremented by 1. Register Ry is not changed. (MOD = 1)

MVM2 Rx, Ry (op code 0): The halfword in register Ry is moved to register Rx and then Rx is decremented by 2. Register Ry is not changed. (MOD = 0)

MVP1 Rx, Ry (op code 0): The halfword in register Ry is moved to register Rx and then Rx is incremented by one. Register Ry is not changed. (MOD = 2)

MVP2 Rx, Ry (op code 0): The halfword in register Ry is moved to register Rx and then Rx is incremented by 2. Register Ry is not changed. (MOD = 3)

*Logical Register Microinstructions:*

AND Rx, Ry (op code 0): The low order byte of register Ry is ANDed with the low order byte of register Rx and the results are placed into the low order byte of Rx. (MOD = 5)

CLRI Rx, MASK (op code 9): The 1 bits in the mask (8-15 of the microinstruction) set the corresponding bits in the low order byte of register Rx to 0. Zeros in the mask have no effect on register Rx.

EMIT Rx, DATA (op code 8): The data (bits 8-15 of the instruction) is put into the low order byte of register Rx.

ORB Rx, Ry (op code 0): The low order byte of register Ry is ORed with the low order byte of Rx and the results are placed into the low order byte of Rx. (MOD = 6)

ROTR Ry, M (op code E): (Device address [DA] = 0)

**Modifier**

D — The bits of the low order byte of register Ry are shifted one position to the right and bit 7 of the low order byte is placed into bit 0 of the low order byte.

E — The bits of the low order byte of register Ry are shifted three positions to the right and the spill bits are placed in the high order bits of the low order byte of Ry.

F — The bits in the low order byte of Ry are shifted four positions to the right and the spill bits are placed in the high order bits of the low order byte of Ry.

SETI Rx, MASK (op code B): The 1 bits in the mask (8-15 of the microinstruction) set the corresponding bits in the low order byte of register Rx to 1. Zeros in the mask have no effect on register Rx.

SHFTR Ry, 1 (op code E): (Device address [DA] = 0) The bits in the low order byte of register Ry are shifted to the right one position and bit 0 of the low order byte of Ry is set to the value of bit 7 of the high order byte of Ry. The low order bit of Ry is shifted out of Ry and hence lost.

XOR Rx, Ry (op code 0): The low order byte of Ry is exclusive ORed with the low order byte of Rx and the result is placed in the low order byte of Rx. (MOD = 7)

*Arithmetic Register Microinstructions:*

ADD Rx, Ry (op code 0): The low order byte of register Ry is added to the low order byte of register Rx and the results are placed into the low order byte of Rx. Any resulting carry is added to the high order byte of Rx. (MOD = 8)

ADDI Rx, DATA (op code A): The data (in bits 8-15 of the microinstruction) plus 1 is added to the data in register Rx and the result is stored in register Rx.

ADDS1 Rx, Ry (op code 0): The low order byte of register Ry is added to the high order byte of register Rx and the results are placed into the low order byte of Rx. The high order byte of register Rx is set to hex 01 if there is a carry. (MOD = 4)

ADDS2 Rx, Ry (op code 0): The low order byte of register Ry is added to the high order byte of register Rx and the results are placed into the low order byte of Rx. The high order byte of Rx is set to hex 00 if there is a carry. The high order byte of Rx is set to hex FF if there is no carry. (MOD = B)

SUB Rx, Ry (op code 0): The low order byte of register Ry is subtracted from the low order byte of register Rx and the results are placed into the low order byte of Rx. Any resulting borrow is subtracted from the high order byte of Rx. (MOD = 9)

SUBI Rx, DATA (op code F): The data (bits 8-15 of the microinstruction) ~~minus~~ PLUS 1 is subtracted from the data in register Rx and the result is stored in Rx.

e.g. F0 25 → 26 → R0 = 1876 − 26 = R0 = 1850

*Jump Microinstructions:* A jump microinstruction tests for a condition and jumps over (skips) the next sequential microinstruction if that condition is met. If the jump condition is not met, the microinstruction following the jump microinstruction is executed and the address of the jump microinstruction plus 4 is stored in register 1 for use as a return address in case the next sequential microinstruction (after the jump microinstruction) branches to a subroutine.

JALL Rx (op code C): If the low order byte of register Rx is hex FF, a jump occurs. Register Ry is not used. The jump modifier is 4.

JALLM Rx, Ry (op code C): If the low order byte of register Rx has a 1 bit at every position that the low order byte of register Ry has a 1 bit, a jump occurs. Bits equal to 0 in Ry are not tested in Rx. The jump modifier is 5.

JEQ Rx, Ry (op code C): If the low order byte of register Rx equals the low order byte of register Ry, a jump occurs. The jump modifier is 2.

JHAM Rx, Ry (op code C): If the high order byte of register Rx has a 1 bit in every position that the low order byte of register has a 1 bit, a jump occurs. Bits equal to 0 in register Ry are not tested in register Rx. The jump modifier is 7.

JHE Rx, Ry (op code C): If the low order byte of register Rx is greater than or equal to the low order byte of register Ry, a jump occurs. The jump modifier is 9.

JHI, Rx, Ry (op code C): If the low order byte of register Rx is greater than the low order byte of register Ry, a jump occurs. The jump modifier is 8.

⌐ Rx, Ry (op code C): If the low order byte of register Rx is not equal to the low order byte of register Ry, a jump occurs. The jump modifier is A.

JHSNM Rx, Ry (op code C): If the high order byte of register Rx has a 0 bit at every position that the low order byte of register Ry has a 1 bit, a jump occurs. Bits equal to 0 in Ry are not tested in Rx. The jump modifier is F.

JLE Rx, Ry (op code C): If the low order byte of register Rx is less than or equal to the low order byte of register Ry, a jump occurs. The jump modifier is 0.

JLO Rx, Ry (op code C): If the low order byte of register Rx is less than the low order byte of register Ry, jump occurs. The jump modifier is 1.

JNO Rx (op code C): If the low order byte of register Rx is 0, a jump occurs. Register Ry is not used. The jump modifier is 3.

JNOM Rx, Ry (op code C): If the low order byte of register Rx has a 0 bit at every position that the low order byte of register Ry has a 1 bit, a jump occurs. Bits equal to 0 in Ry are not tested in Rx. The jump modifier is 6.

JSB Rx (op code C): If the low order byte of register Rx is not hex 00, a jump occurs. Register Ry is not used. The jump modifier is B.

JSM Rx, Ry (op code C): If the low order byte of register Rx has a 1 bit at every position that the low order byte of register Ry has a 1 bit, a jump occurs. Bits equal to 0 in Ry are not tested in Rx. The jump modifier is E.

JSN Rx (op code C): If the low order byte of register Rx is not hex FF, a jump occurs. Register Ry is not used. The jump modifier is C.

JSNM Rx, Ry (op code C): If the low order byte of register Rx has a 0 bit at every position that the low order byte of register Ry has a 1 bit, a jump occurs. Bits equal to 0 in Ry are not tested in Rx. The jump modifier is D.

This page intentionally left blank.

## Glossary

**adapter:** A hardware device that connects two channels on the same computing system or on different systems

**address label:** One or more characters used to identify an address in a computer program

**alphameric keys:** That part of the keyboard that resembles a typewriter keyboard

**ALU:** Arithmetic logic unit

**APL:** A programming language

**ATTN:** Attention

**BASIC:** Beginners all-purpose symbolic instruction code

**BCD:** Binary coded decimal

**bits per second:** Communication line transmission rate

**BOT:** Beginning of tape

**bps:** Bits per second

**BUP:** Bring up program

**CC1:** Character count 1

**CMD:** Command key

**control unit:** That portion of the A1 board in the 5100 that contains the controller, portions of the base I/O card, and all storage. The control unit contains microinstructions and the logic necessary to execute them

**controller:** The microinstruction processor within the 5100 Portable Computer

**CRC:** Cyclic redundancy check

**CRT:** Cathode ray tube

**cyclic redundancy check:** An error check. Counting of the bits on a record

**DA:** Device address

**DCP1:** Diagnostic control program 1

**DCP2:** Diagnostic control program 2

**EC:** Error code

**EOT:** End of tape

**executable ROS:** Contains microinstructions that can be executed directly by the controller

**flags:** A character or bit that signals a condition to a program, such as an I/O error. A record on the format track that provides positioning and timing information for records on the data track

**flyplate:** The pad on the bottom of a keyboard key module. When a key is pressed, the flyplate raises and the capacitive change indicates to the keyboard printed circuit that the key is pressed

**FRU:** Field replaceable unit

**header record:** A record containing identifying information pertaining to a group of records that follow

**hex:** Hexadecimal

**I/O:** Input/output

**I/O interface port:** A removable panel located on the back of the 5100 that contains the signal and power connectors for attaching I/O devices

**IAR:** Instruction address register

**IMF:** Internal machine fix

**interpreter:** A computer program stored in ROS that controls execution of BASIC and APL instructions

**interval timer:** Measures the time between clock pulses coming from the read head

**IOCB:** Input/output control block

**IR:** Incident report

**ITPS code:** Internal teleprocessing system code

**jackshaft:** A mechanical device in the tape unit that transfers motion from the motor (via a belt) to the spindle

**KBD:** Keyboard

**kVA:** Kilovolt amperes

**LED:** Light emitting diode

**logical record:** A group of data independent of its physical location

**loop:** A group of instructions that are executed repeatedly

**LWR:** Loop write read

**MAP:** Maintenance analysis procedures

**MDI:** Maintenance and diagnostics integrated

**MHz:** Megahertz

**ms:** Millisecond

**nonexecutable ROS:** Contains microinstructions that are first loaded into read/write storage and executed from there

**ns:** Nanosecond

**NTF:** No trouble found

**numeric keys:** That portion of the keyboard that resembles a calculator keyboard

**PC:** Printed circuit

**PC board:** A printed circuit board consists of electrical circuits mounted on a board to distribute signals and voltages

**PG:** Parity generator

**PH:** Polarity hold

**PLFP:** Print line failure position

**POR:** Power on reset

**power on reset:** A signal occurring during power up, used to reset all circuits to an operational starting point

**PTX:** Phototransistor

**RDDR:** Read data deserializer register

**RDR:** Read data register

**read only storage:** A storage whose contents are not changed by computer instructions

**record:** A group of related data items

**ROS:** Read only storage

**RRA:** Remove, replace, adjust

**SA:** Status byte A

**SAR:** Storage address register

**SB:** Status byte B

**scroll:** Move data on the display screen up or down

**SDR:** Storage data register

**spindle:** A mechanical device in the tape unit for transferring motion from the jackshaft pulleys to the cartridge capstan

**steps:** An offset that occurs in a reel of tape when exposed to extreme temperature drops

**supervisor:** That part of the control program that coordinates the use of resources and maintains the flow of processor operations

**sync:** Synchronize or synchronous

**syntax:** Structural rules of a programming language

**truncate:** To stop an operation at a specified point

**TS:** Test status

**TSR:** Transistor switching regulator

**TTL:** Transistor-transistor logic

**TV monitor:** An external display assembly that displays the same information as the 5100 5-inch display screen

**typamatic:** A keyboard signal generated by the repeat action keys when held down for more than 700 ms

**video:** Information relating to or used in receiving an image on the display screen

**work area:** A storage location reserved for intermediate use in programming

**ap connector:** A communications adapter feature tool that allows testing of transmit and receive signals without attaching to a remote device

**Z-code:** The code used by the APL interpreter stored in APL ROS

**μs:** Microsecond

This page intentionally left blank.

# READER'S COMMENT FORM

IBM 5100
Portable Computer
Maintenance Information Manual

SY31-0405-2

## YOUR COMMENTS, PLEASE . . .

Your comments assist us in improving the usefulness of our publications; they are an important part of the input used in preparing updates to the publications. All comments and suggestions become the property of IBM.

Please do not use this form for technical questions about the system or for requests for additional publications; this only delays the response. Instead, direct your inquiries or requests to your IBM representative or to the IBM branch office serving your locality.

Corrections or clarifications needed:

*Page* *Comment*

I would like a reply. ☐

Name _____

Address _____

IBM 5100 Maintenance Information Manual Printed in U.S.A. SY31-0405-2

International Business Machines Corporation
General Systems Division
5775D Glenridge Drive N.E.
Atlanta, Georgia 30301
(USA Only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)