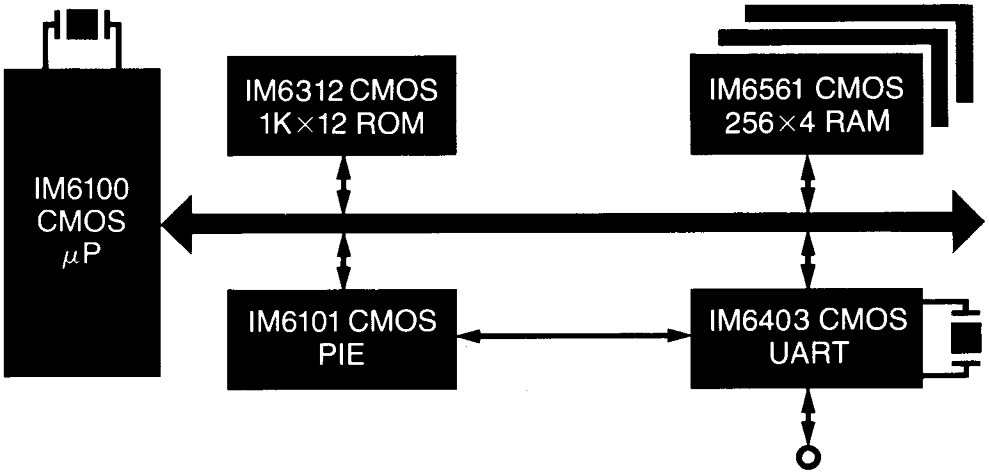
$7.50

INTERSIL

IM6100 CMOS FAMILY SAMPLER



CONTENTS

INTRODUCTION

An all-CMOS IM6100 based microprocessor system with 256 X 12 random access memory, 1K X 12 read only memory and a serial interface port can be fabricated utilizing seven (7) CMOS LSI devices.

The IM6100 is a single chip, 12-bit microprocessor implemented in silicon gate complementary metal oxide semiconductor (SiG CMOS) technology. It recognizes the instruction set of the Digital Equipment Corporation PDP-8 minicomputer.

The IM6561 is a high speed, low power silicon gate CMOS 1024 bit static RAM organized 256 words by 4 bits. Data-In and Data-Out are multiplexed on the same pins.

The IM6312 is a 1K X 12 mask programmable CMOS ROM with address and data multiplexed on the same pins. The IM6312-001 is 'masked' at the factory with the ODT program (Appendix A). This program allows the user to enter user programs into the RAM via the Teletype and then to edit and execute them.

A Parallel Interface Element, IM6101, provides the universal means of interfacing peripheral devices to the IM6100. It is programmable and configured by the IM6100 for a specific interface during system initialization.

The Universal Asynchronous Receiver/Transmitter, IM6402/03, is a CMOS LSI programmable subsystem for interfacing processors to an asynchronous data channel.

Additional requirements to enable the system to be fully operational are a single 5 volt power supply, two crystals, a 3.3 MHz crystal for the microprocessor and a 3.60 MHz crystal for a 110 baud serial port, control logic to reset and run the system and serial I/O level shifters to communicate with a Teletype or other ASCII terminals (Figure 1).

The dynamic power dissipation of the system (excluding that of the serial I/O level shifters) will be less than 50 mW at 4 MHz and 5 volts.

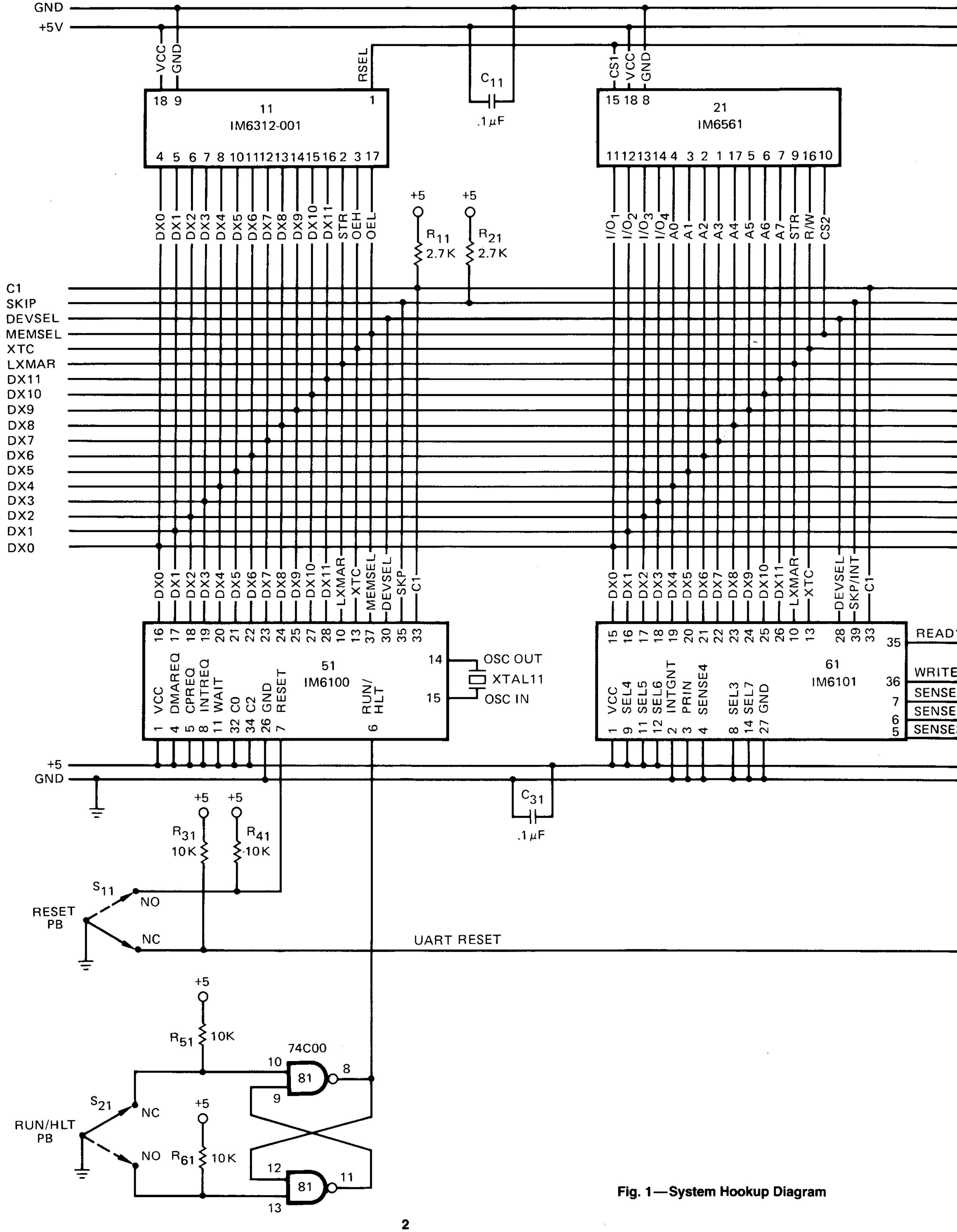
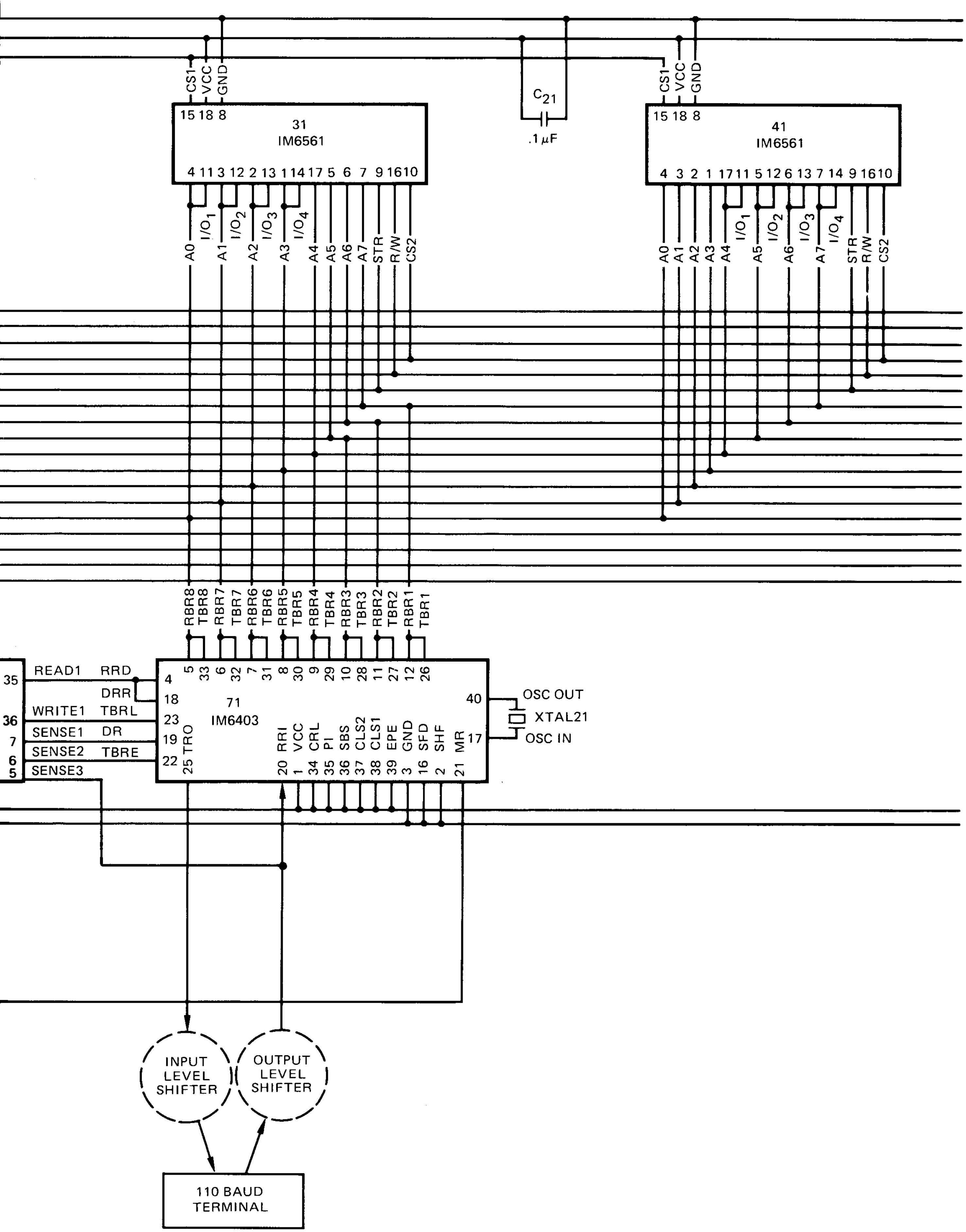


Fig. 1—System Hookup Diagram



CS1
VCC
GND
15 18 8
31
IM6561
4 11 3 12 2 13 1 14 17 5 6 7 9 16 10
I/O1
I/O2
I/O3
I/O4
A0
A1
A2
A3
A4
A5
A6
A7
STR
R/W
CS2
C21
.1 μF
CS1
VCC
GND
15 18 8
41
IM6561
4 3 2 1 17 11 5 12 6 13 7 14 9 16 10
I/O1
I/O2
I/O3
I/O4
A0
A1
A2
A3
A4
A5
A6
A7
STR
R/W
CS2
RBR8
TBR8
RBR7
TBR7
RBR6
TBR6
RBR5
TBR5
RBR4
TBR4
RBR3
TBR3
RBR2
TBR2
RBR1
TBR1
5 33 6 32 7 31 8 30 9 29 10 28 11 27 12 26
35
READ1
RRD
4
DRR
18
36
WRITE1
TBRL
23
7
SENSE1
DR
19
6
SENSE2
TBRE
22
5
SENSE3
71
IM6403
25 TRO
RRI
VCC
CRL
PI
SBS
CLS2
CLS1
EPE
GND
SFD
SHF
MR
20 1 34 35 36 37 38 39 3 16 2 21
40
OSC OUT
XTAL21
17
OSC IN
INPUT LEVEL SHIFTER
OUTPUT LEVEL SHIFTER
110 BAUD TERMINAL

POWER ON PROCEDURE

1. Connect up the CMOS/LSI devices as shown in Figure 1.

2. Choose the level shifting network for the serial I/O device (for example a 20 mA current loop for Teletype) and make the appropriate connections.

3. Turn the power on. Press and release RESET pushbutton. Put the Teletype "ON LINE".

4. Press and release RUN/HLT pushbutton. The ODT program (Appendix A) responds with a CARRIAGE RETURN/LINE FEED and then waits for user commands.

FUNCTIONAL DESCRIPTION

The IM6312-001 ROM is mask programmed to be enabled when DX(0) = DX(1) = 1 at the falling edge of LXMAR to respond to addresses 6000-7777$_8$.

The RSEL output defines an area in the 4096 word addressing space dedicated to RAM. The IM6312-001 RSEL is programmed to be enabled when DX(0) = DX(1) = 0 at the falling edge of LXMAR to respond to addresses 0000-1777$_8$. If only 256 words of RAM are used, DX(2) and DX(3) bits are ignored during addressing. For 1K word RAM systems, DX(2) and DX(3) become the high order address bits.

The IM6402/03 UART is hardwired for eight data bits, two stop bits and no parity. If the IM6403 is used, the eleven stage divider chain is selected. This on-chip divider and oscillator allows an inexpensive crystal to be used as a timing source for the IM6403 UART. UARTs require the clocks to be 16X the bit rate (baud). For Teletype operation, it is ten characters per second or 110 baud. Each character consists of eleven bits (one start bit + eight data bits + two stop bits). The UART clock frequency must be 1.76 KHz (110 X 16). If a standard color TV crystal of 3.579545 MHz is used in conjunction with the eleven stage divider chain, one obtains a baud rate of 109.2 ($3.579545 \times 10^6 \div 2^{11} \div 16$) which is well within the tolerance limits for Teletype operation. (The crystal frequency should be 3.6044 MHz for 110 baud.) The same type of crystal can also be used to clock the IM6100. Refer to the section on UART timing for the circuits to be used for the IM6402.

The PIE is wired to respond to the select code 01110 on SEL3-SEL7 and to inhibit priority vectoring. The PIE control registers CRA and CRB are programmed by the ODT to be 0000_8 and 0060_8, respectively. The PIE control signals are then defined as shown below:

WRITE1 and WRITE2	Active Low
SENSE1 and SENSE2	Active on 0 to 1 transition
SENSE3 and SENSE4	Active on 1 to 0 transition
FLAG1, FLAG2, FLAG3 and FLAG4	Initialized Low

Refer to the IM6100, IM6101, IM6312, IM6402/03 and IM6551/61 data sheets for detailed descriptions of the CMOS/LSI devices used in the IM6100 CMOS FAMILY SAMPLER.

The RESET pushbutton is for system initialization. When the IM6100 is reset, it halts with the Program Counter set to 7777_8. The RUN/HLT pushbutton is used to alternatively run and halt the CPU. C_1 and SKP lines are pulled up to V_{CC} with 2.7K resistors, since these lines are driven by open drain outputs from the PIE.

The section on level shifting networks details the circuits for 20 mA current loop and RS232 serial I/O interface. User modifications that may be required to select 20 mA current loop and full duplex operation for Teletype are described in Application Bulletin M006 : "Teletype Interface for the IM6100".

20 mA CURRENT LOOP INTERFACE

Figures 2-A, 2-B and 2-C describe the discrete circuits for 20 mA current loops. For reliable Teletype operation, the receive current loop must be established by a voltage source of 15 volts or more.

FLAG1 output of the PIE can be used to control the Reader Relay, if one is installed. Note that the ODT program initializes FLAG1 to be 0 and, therefore, the reader relay to be OFF. The user program must explicitly control the relay by programming FLAG1 output.

The pin connections to a 20 mil 10 pin header are shown.

20mA Current Loop Interfaces For Teletype

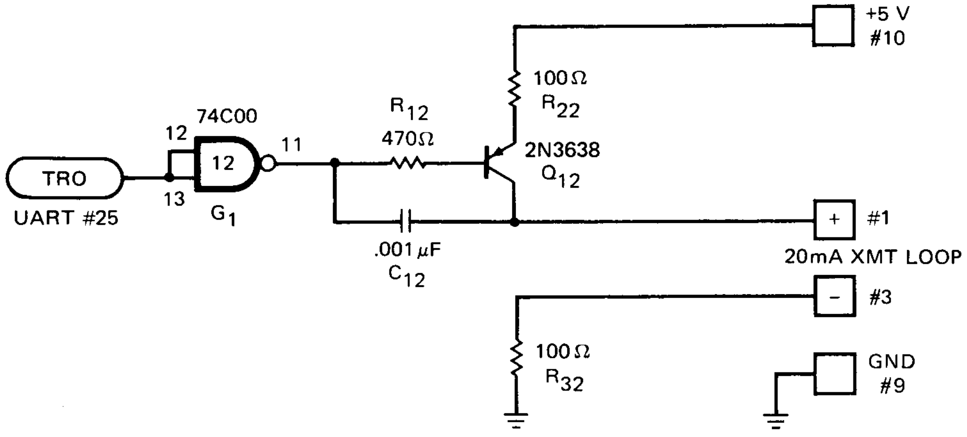


Fig. 2A—XMT Loop

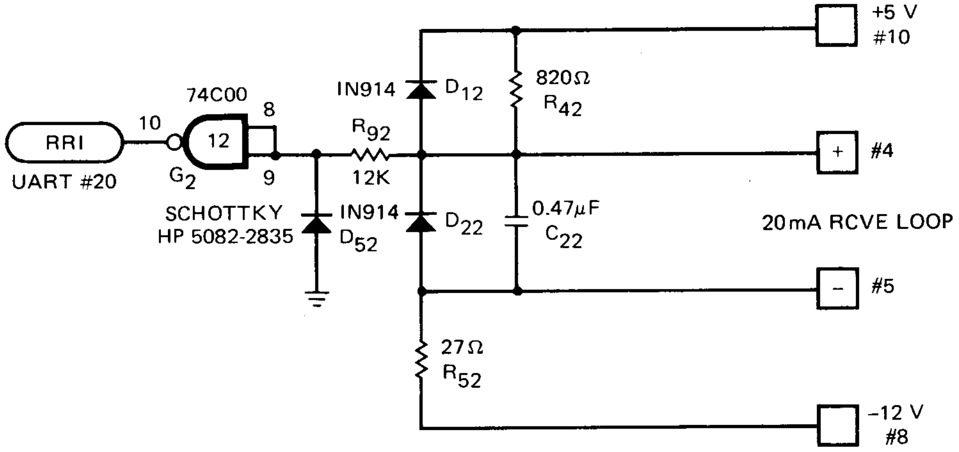


Fig. 2B—RCVE Loop

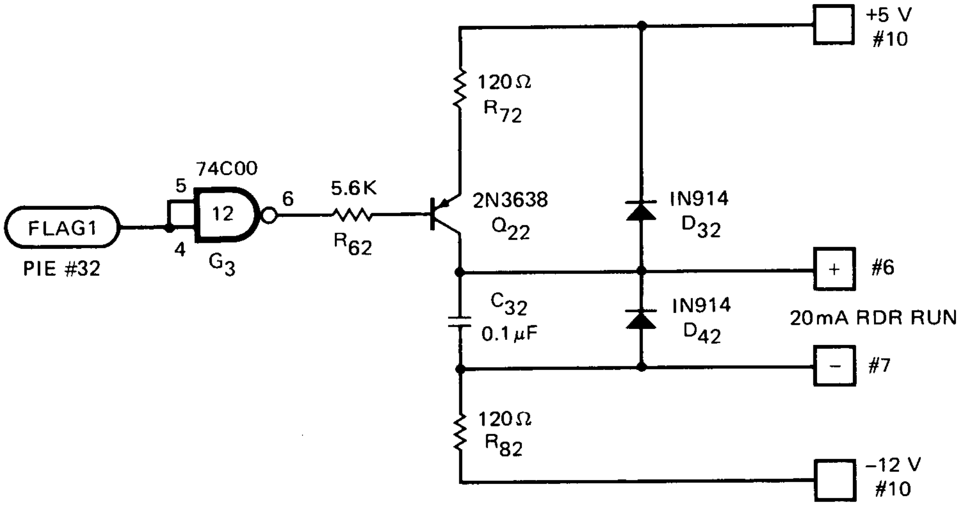


Fig. 2C—RDR RUN Loop

RS232C INTERFACE

The level shifting networks to conform to the RS232C formats are shown in Figures 3-A and 3-B. The pin connections shown are for an AMP 205858-1 connector as used on the 6960-SAMPLR Board.

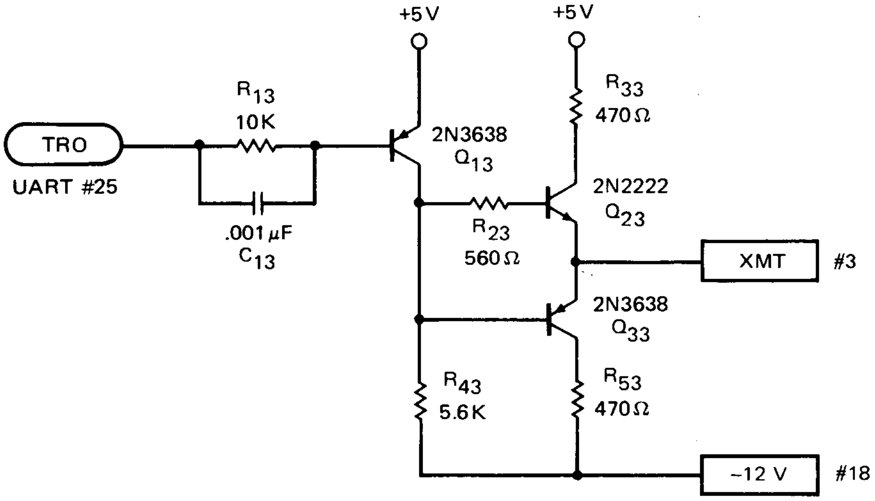


Fig. 3A

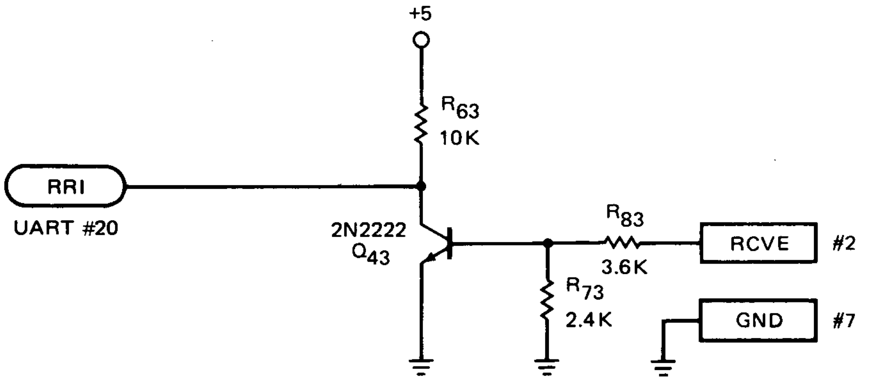
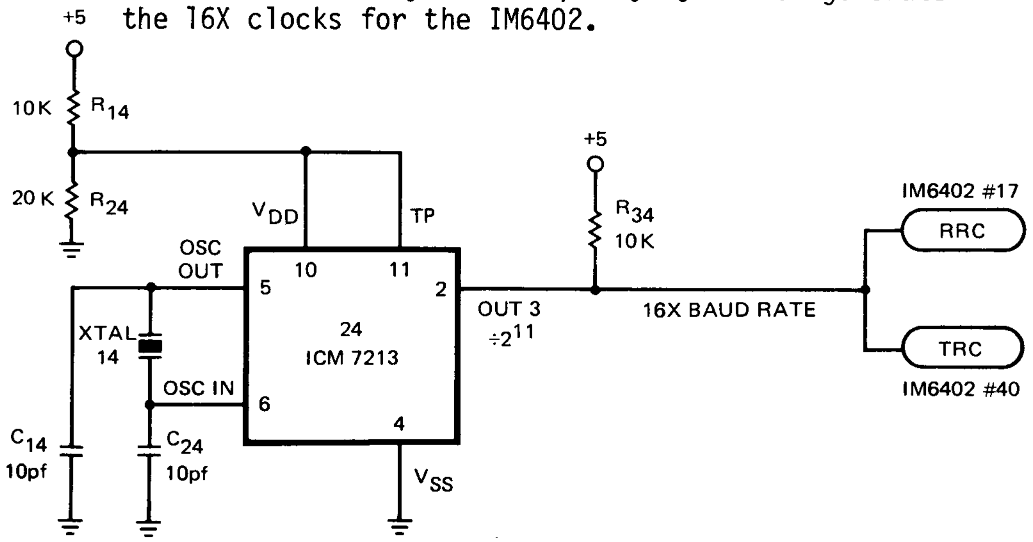


Fig. 3B

IM6402 UART TIMING

ICM7213

ICM7213 is a fully integrated oscillator and frequency divider with buffered outputs. The circuit in Figure 4-A divides the crystal frequency by 2^{11} to generate the 16X clocks for the IM6402.



XTAL	OUT3	BAUD RATE
3.6044	1.76 kHz	110
3.5795	1.74 kHz	109.2

Fig. 4A—IM6402 UART Timing with ICM 7213

555 TIMER

The analog timer circuit shown in Figure 4-B can be used for the UART clocks. The resistors and capacitors in the timing circuit must be chosen to be temperature stable.

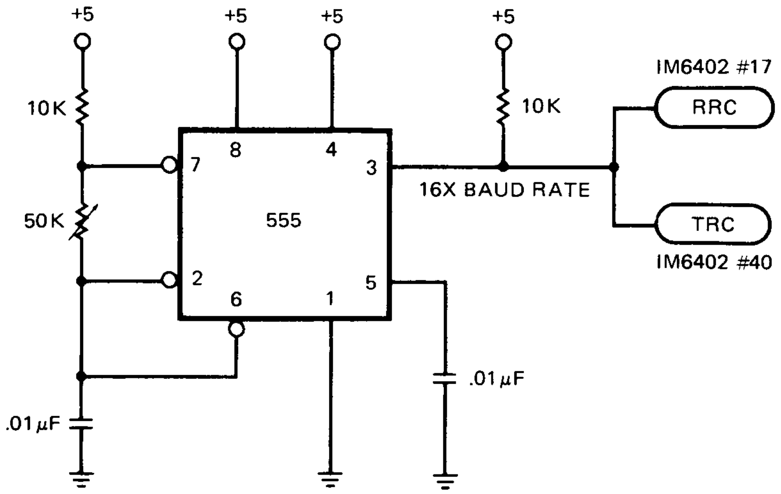


Fig. 4B—IM6402 UART Timing with 555

BAUD RATE GENERATOR

The 34702 CMOS Baud Rate Generator circuit can be programmed via S_0, S_1, S_2 and S_3 select lines to generate 13 of the most commonly used baud rates.

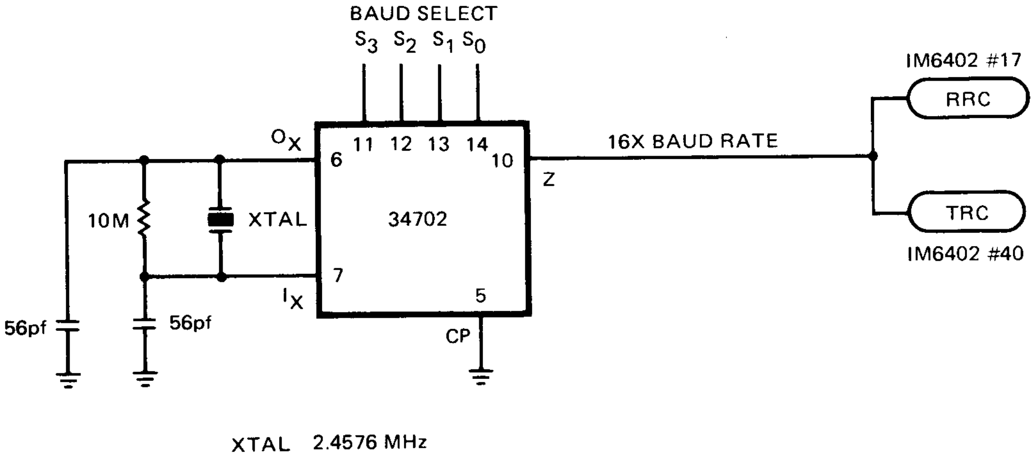
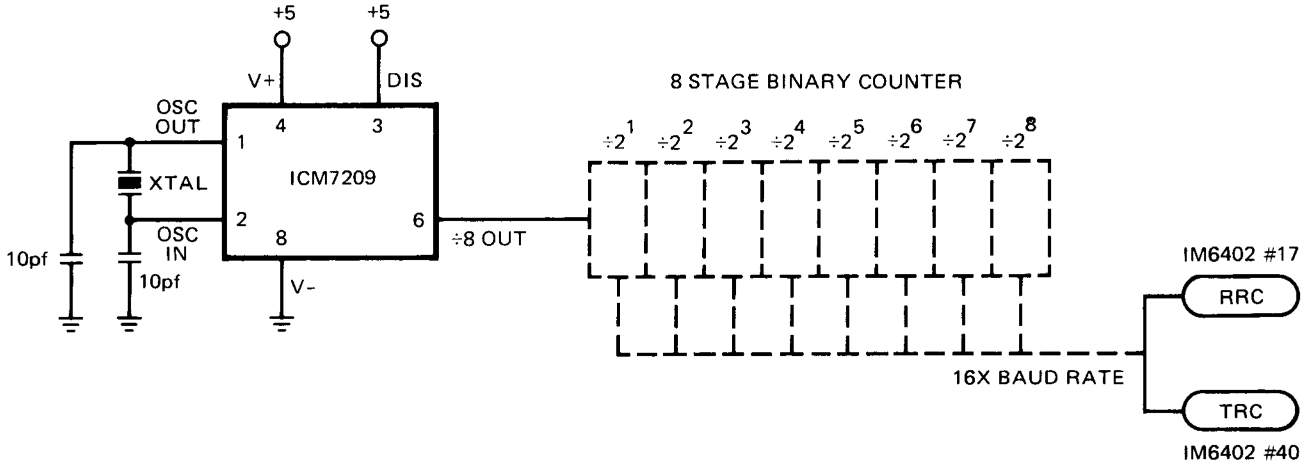


Fig. 4C—IM6402 UART Timing with Baud Rate Generator

XTAL 2.4576 MHz	S_3	S_2	S_1	S_0	Baud Rate
	0	0	0	0	
	0	0	0	1	
	0	0	1	0	50
	0	0	1	1	75
	0	1	0	0	134.5
	0	1	0	1	200
	0	1	1	0	600
	0	1	1	1	2400
	1	0	0	0	9600
	1	0	0	1	4800
	1	0	1	0	1800
	1	0	1	1	1200
	1	1	0	0	2400
	1	1	0	1	300
	1	1	1	0	150
	1	1	1	1	110

ICM7209

An 8-stage divider chain along with the oscillator and 3-stage divider circuit in the ICM7209 with the appropriate XTAL frequency (Figure 4-D) can generate some of the more commonly used baud rates.



XTAL MHz	DIVIDER STAGE	BAUD RATE
3.6044	2^8	110
3.5795	2^8	109.2
2.4576	2^6	300
	2^5	600
	2^4	1200
	2^3	2400
	2^2	4800
	2^1	9600

Fig. 4D—IM6402 UART Timing with ICM7209

SINGLE CLOCK

Since the IM6100 family of devices are static, the system may be single clocked. Figure 5 shows the circuit that stops the free running oscillator and introduces manually generated single clocks. ICM7209 is used as a buffered oscillator. The SC EN SW, Single Clock ENable, selects the ICM7209 output or the single clock as the input to the IM6100. The 74C74 is used to ensure integral clocking and to avoid clock slivers while switching from one mode to another. The single clocks are generated by the debounced SC, Single Clock, pushbutton.

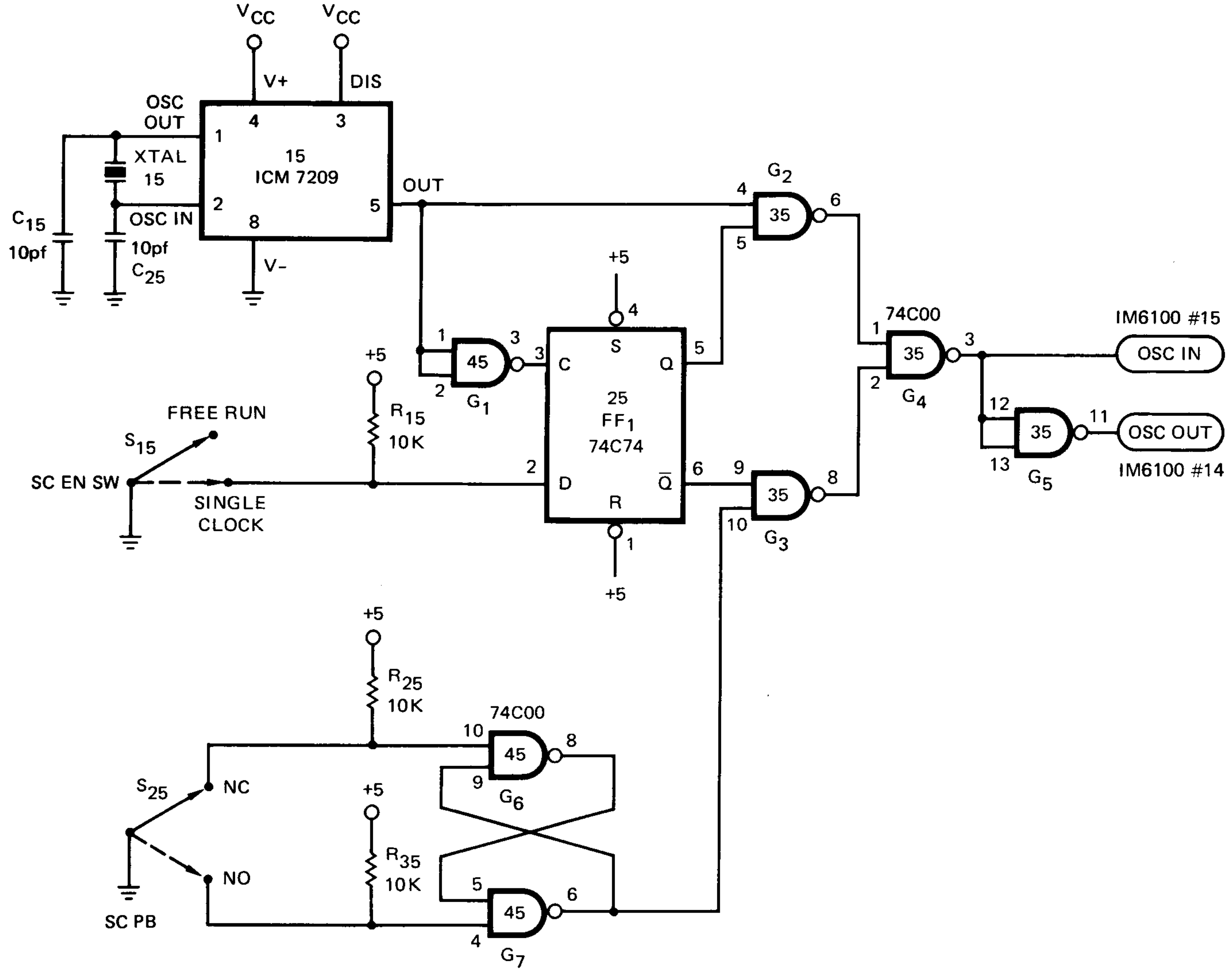


Fig. 5—Single Clock

SINGLE INSTRUCTION

The IM6100 can be controlled to execute one instruction at a time using the circuit in Figure 6-A. When the SI EN SW, Single Instruction ENable, is in the free run mode, the WAIT signal is high, and the normal operation of the processor is not affected. When the SI EN SW is in the single instruction mode, the WAIT line is pulled down during IFETCH cycle, pausing the IM6100 in the instruction read state. Activating the SI pushbutton will clear the WAIT line, and the processor will go on to execute the instruction and then pause while reading the next instruction and so on.

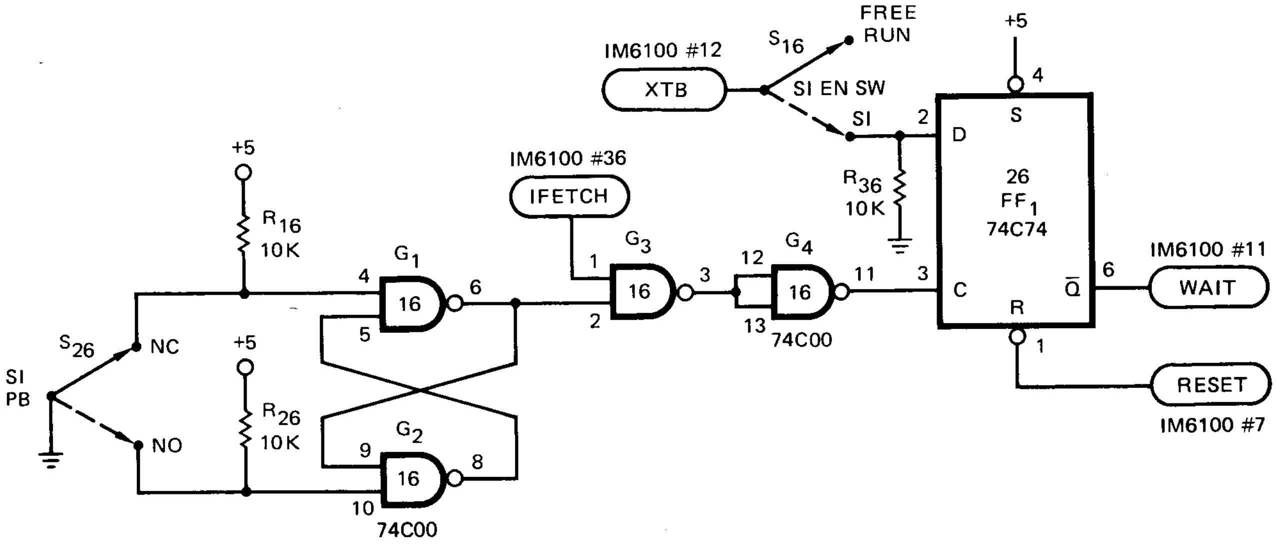


Fig. 6A—Single Instruction

The simple display circuit in Figure 6-B can be used to display instruction addresses, or the state of the DX bus. If the DX EN SW is in the INSTR ADDRESS mode, the quad latches are clocked by IFETCH•LXMAR to display the instruction address. If the DX EN SW is in the DX mode, the DX data flows through the latch.

In the single clock mode, it is useful to have the DX EN SW in the DX state to read the state of the DX bus at every clock.

In the single instruction mode, first set the DX EN SW in the INSTR ADDR mode. The LEDs will then display the address of the instruction about to be executed. Then switch the DX EN SW to the

DX mode, and the display will then show the instruction that is about to be executed since the DX lines contain the instruction as the IM6100 is 'paused' in the read state. Flip the DX EN switch to the INSTR ADDR state before activating the SI pushbutton for the next instruction and so on to step through instructions, one at a time.

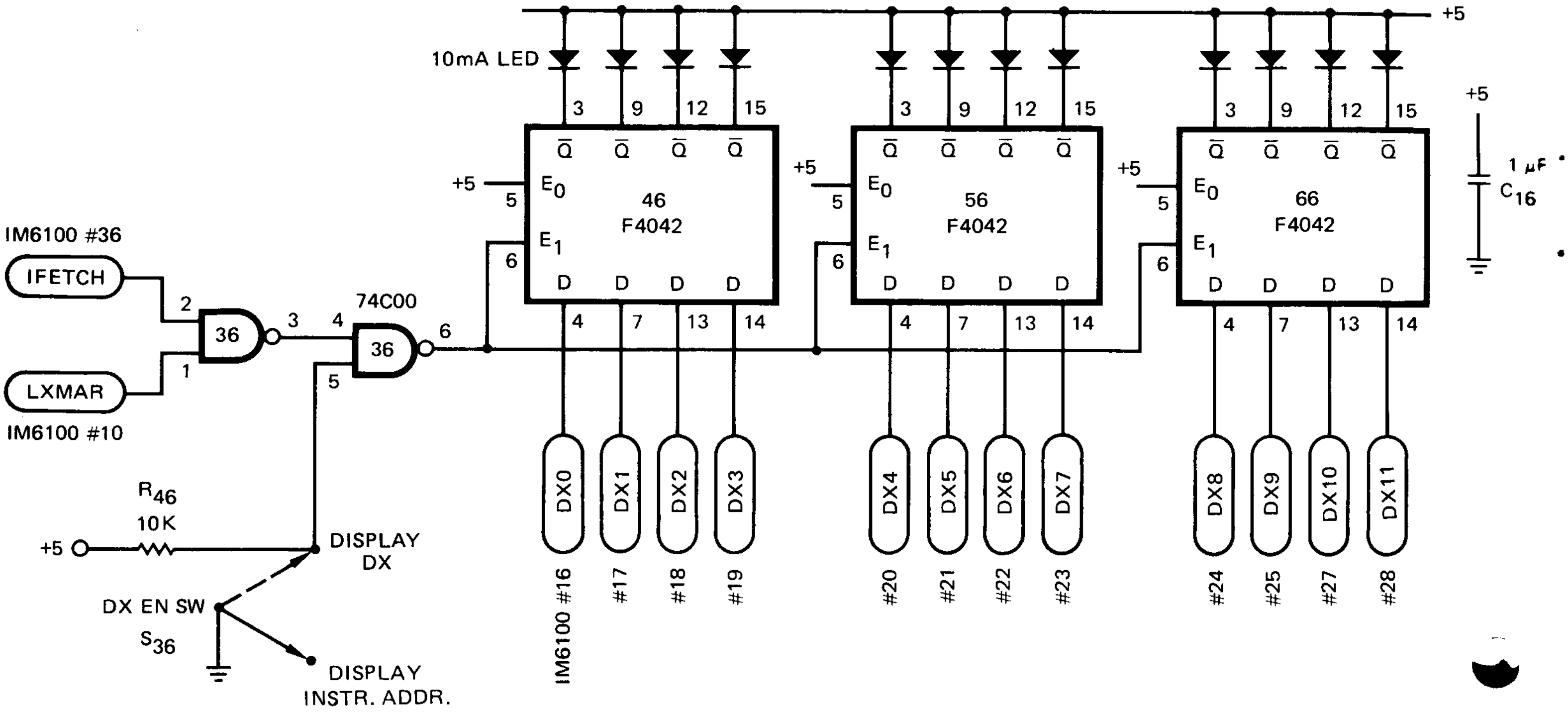


Fig. 6B—Program Counter and DX bus display

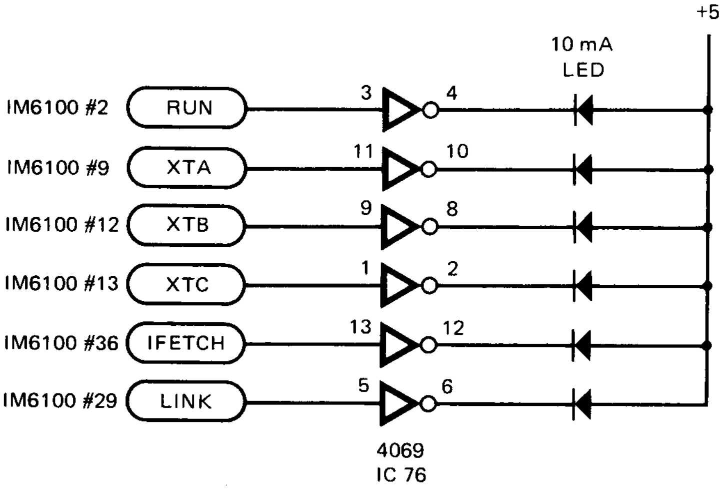


Fig. 6C Link and Timing Signal Display

PARALLEL DATA I/O

The unused WRITE2 and READ2 control signals of the PIE may be used to implement the parallel data I/O circuit shown in Figure 7.

When the user program executes a READ2 instruction, the 80C95 tri-state buffers enable the data on the DX bus. The DX information can be latched in the 74C174 hex DFFs by executing a WRITE2 instruction. Note that the WRITE2 is programmed to be active low by ODT, and the data is latched by the second edge of the write pulse.

SENSE3-4 and FLAG 1-4 of the PIE are also available to the user.

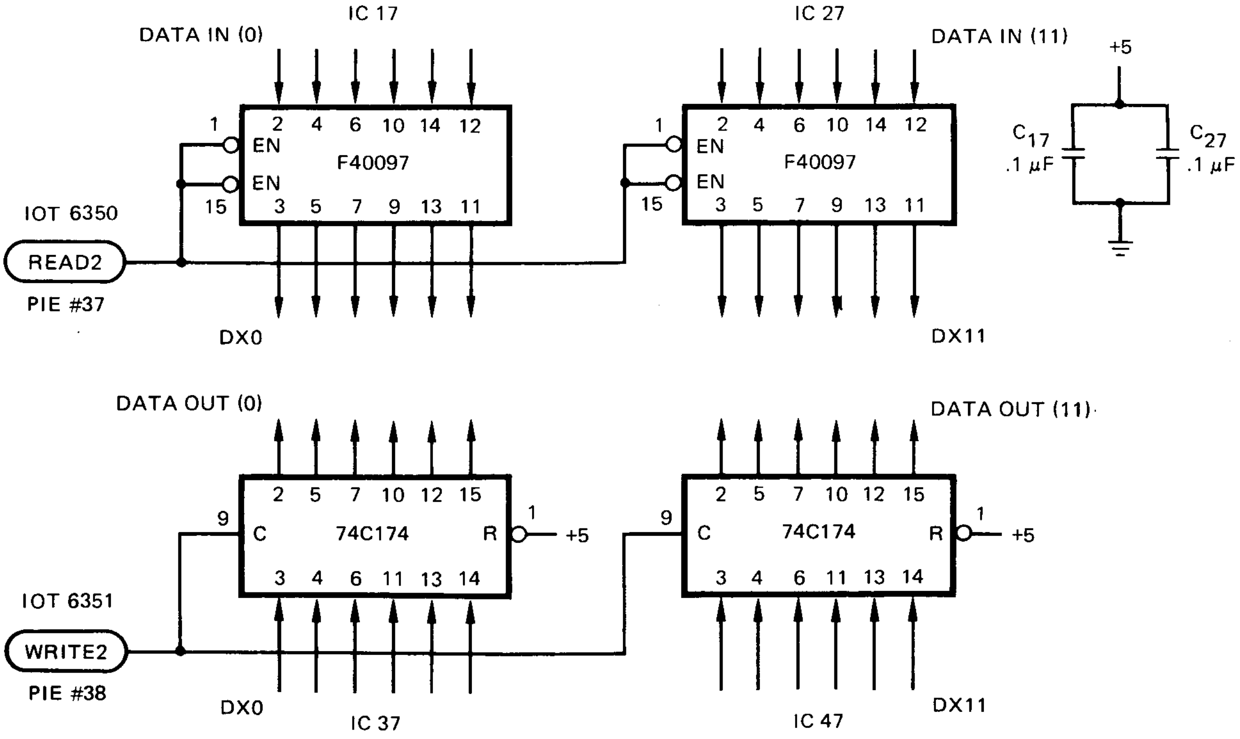


Fig. 7—Parallel Data I/O

APPENDIX A

1. IM6312-001 CMOS OCTAL DEBUGGING TECHNIQUE (ODT) ROM

INTRODUCTION

This describes the use of the Intersil Octal Debugging Technique (ODT) program.

The commands for Intersil ODT are very similar to those for Digital Equipment Corporation's ODT for use on the PDP-8 in particular, and the DDT family of programs in general.

RESERVED LOCATIONS

The RAM locations in page zero with octal addresses 5, 6 and 20 through 77 are reserved for use by the ODT program. User programs should not modify these locations.

COMMANDS

ODT commands consist of a control character or an octal number followed by a control character. All commands are echoed, that is, each character is printed as it is typed in. Octal numbers may consist of octal digits only (no 8's or 9's) and may be from one to four digits long.

Note that ODT does not print a special prompt character (such as an asterisk) before commands may be typed, and does not require that commands be terminated by a special character, such as carriage return (in fact, carriage return is a special command). Commands may be typed in any time the Teletype is idle and are executed as soon as the control character is typed.

BINARY LOAD COMMAND

L - Load from the tape reader

Typing an L will load BIN tape (ignoring change field characters) from a reader. To use command, type L after a prompt, place tape in reader on the leader-trailer part, then start the tape reader. The BIN tape will be read into the current field and the checksum will be printed out on the Teletype following the end of the load. This should be 0000 for a proper load

EXAMINE/MODIFY COMMANDS

/ (slash) - opens a location

Typing an octal number nnnn followed by a slash causes the location whose address is nnnn to be opened. When a location is opened, its content is printed out as an octal number. Typing a slash not preceded by a number causes the most recently opened location to be reopened.

(carriage return) - closes a location

When a location is open, typing an octal number, nnnn, followed by a carriage return causes the contents of the location to be changed to the number nnnn and closes the location. Once a location is closed, its contents cannot be changed without re-opening the location. Typing a carriage return not preceded by a number causes the location to be closed without modifying its contents. If a carriage return (possibly preceded by a number) is typed when no location is open, it has no effect.

(line feed) - closes and opens next

When a location is open, typing a line feed causes the location to be closed and the next memory location (that with an address one higher than the current location) to be opened. The address of the new location will be typed out, followed by a slash, followed by the contents of the new location. The effect is the same as if the user had typed a carriage return, followed by typing the address of the next location and the slash. Typing an octal number, nnnn, before typing the line feed causes the contents of the old location to be changed to nnnn. Then the old location is closed and the next location is opened, as described above.

← (back arrow) - closes location and opens indirect reference

When a location is open, typing a back arrow causes the location to be closed. The contents of the location are then treated as an indirect reference. That is, the content of the old location is taken as an address, and the new location is opened. As with the line feed command, the address of the newly opened location is typed, followed by a slash, and the effect is the same as if the user had typed the address and slash. If while a location is open, an octal number, nnnn, is typed followed by a back arrow, the content of the open location is changed to nnnn and proceeds as above. Note in this case that the address of the new location opened will be the same as the number just typed in. On most Teletypes, the back arrow character is the same as shift O.

↑ (up arrow) - closes location and opens memory reference

This command behaves identically to the back arrow command except that the contents of the location are treated as a memory reference instruction, and it is the location referenced by that instruction that is opened. The location opened is that immediately referenced by the instruction. If the instruction is indirect (bit 3 is set to 1), then typing the up arrow only opens the location containing the pointer to the operand of the instruction. To open the effective location referred to by an indirect instruction, type an up arrow (memory reference) followed by a back arrow (indirection). On most Teletypes, the up arrow character is the same as shift N.

Example

Say that the simple program

```
200   7200    CLA
201   1604    TAD I X
202   2204    ISZ X
203   7402    HLT
204   0205 X, Y
205   0000 Y, 0
```

is stored in memory. Then the following might be the result of a session with ODT. (Note: The underlined portion is typed by the user, and the remainder is typed by the computer. The symbol CR stands for carriage return, and LF stands for line feed.)

```
200/7200 LF              list the program in octal
0201 /1604 LF
0202 /2204 LF
0203 /7402 LF
0204 /0205 LF
0205 /0000 CR
200/7200 7201CR          change CLA to CLA IAC
/7201LF                  verify the change
0201 /1604 ↑             find the location referenced by TAD I X
0204 /0205 ←
0205 /0000 CR
202/2204 2205 LF         change ISZ X to ISZ Y
0203 /7402
```

PROGRAM CONTROL AND BREAKPOINT COMMANDS

G - go to

Typing an octal number, nnnn, followed by a G causes ODT to begin executing the program stored in memory, starting at location nnnn.

B - breakpoint

A breakpoint should be set at some location that will be executed by the user's program, upon the execution of which he wishes control to return to ODT. Typing an octal number, nnnn, followed by a B causes ODT to set a breakpoint at location nnnn. Only one breakpoint can be set at a time, therefore, setting a breakpoint clears the previous breakpoint. Typing a B without preceding it by a number causes the current breakpoint to be cleared. When execution of a program with a breakpoint resumes (via the G or C commands) the content of the breakpoint location is replaced with a special trap instruction that returns control to ODT. When control returns to ODT, the trap instruction is replaced by the original contents of the location. Then ODT prints out the location at which the trap occurred followed by a left parenthesis followed by the contents of the accumulator when the trap occurred. ODT then waits for the user to type in a new command.

C - continue

After a breakpoint causes control to return to ODT from a user program, typing C causes the program to resume execution where it left off. Execution will resume from the location that the program left off, even if the breakpoint was reset to a new location. If an octal number, nnnn, is typed before the C, then after the resumed program has executed the breakpoint once it will continue execution (rather than returning control to ODT) until the breakpoint has been executed nnnn more times; then control returns to ODT just as it does after a normal breakpoint.

A - examine/modify accumulator, link, MQ

Three consecutive ODT RAM locations are reserved for storing the contents of the AC, link and MQ registers when a breakpoint occurs. When execution of the user's program resumes (via the G or C command), the contents of these registers are restored from these locations. Typing A causes the first of these locations, containing the contents of the AC, to be opened. Following this by a line feed opens the next location, containing the contents of the link (either a one or a zero). Following this by a line feed opens the last location, containing the contents of the MQ, to be opened. All of the commands of the previous section (/,

↑, ←, carriage return, line feed) can be used to examine and modify these registers. If one of the registers, AC, is changed during the breakpoint, then the AC will have the new value when program execution resumes. This can be used to change the contents of the AC. The same facility can also be used to set the initial contents of the registers before the first G command begins the program's execution.

Example

Say that the simple program

```
                  300    7001      START, IAC
                  301    7440             SZA
                  302    5300             JMP START
                  303    7402             HLT
```

is stored in memory. Then the following might be the result of a session with ODT.

```
A1764 0LF             accumulator contains garbage, clear it
0050 /0001 0CR        same for link
302B                  set breakpoint at JMP START
300G                  execute program
0302 (0001            breakpoint occurs; accumulator has been
                      incremented
7774C                 Go past breakpoint 1 + 7774 times
0302 (7776            breakpoint occurs
303B                  reset breakpoint to HLT instruction
C
0303 (0000            program stops when AC reaches 0 again
A0000LF
0050 /0001CR          link has been changed by overflow
B                     clear all breakpoints
```

WORD SEARCH COMMANDS

M - open search mask, lower bound, upper bound

The mask, lower bound and upper bound for word searches are kept in that order in three consecutive reserved ODT locations. The first of these locations, the mask, can be opened by typing M. This will cause the current value of the mask to be printed out. The user can enter a new value by typing it followed by any of the word modifying commands (carriage return, line feed, up arrow or back arrow). Typing a line feed causes the next location, containing the lower search bound, to be opened. Its contents will be printed out and it can be modified in the same way. Typing a line feed again causes the final location, containing the upper search bound, to be opened. Initially, the value of all three locations is unspecified. The user should set these locations to the desired value before giving the first W command.

W - word search command

Typing an octal number, nnnn, followed by a W causes a word search to occur. The search proceeds as follows: The number, nnnn, that was typed is masked and remembered as the quantity which is being searched for. (The operation of masking is to take the bitwise boolean AND of the given word with the contents of the mask word.) Then each location, beginning with the location whose address is stored in the lower bound word, is masked and compared with the quantity being searched for. If the two are equal, then the address of the word, followed by a slash and the (unmasked) contents of the word are printed out. Then the next location is examined and so on until (and including) the location whose address is stored in the upper bound word is reached. The word search command does not change the contents of any word in the user's programs.

Example

Say that a program is stored between locations 200 and 377 in memory. Then typing in

```
     M0000 7000LF
 0054/0000 200LF
 0055/0000 377CR
      1000W
```

causes all TAD instructions in the program (those words beginning with the octal digit 1) to be printed out, for example

```
      0210 /1305
      0265 /1305
      0354 /1711
```

if these were the only words in the program containing TAD instructions. Then typing in

```
     M7000 7777CR
     0W
```

would cause all words in the program whose contents were exactly 0000 to be printed out.

Typing in

```
          M7777 0LF
          0054 /0200 250LF
          0055 /0377 300CR
          0W
```

will cause the contents of all words whose addresses are between 250 and 300 (inclusive) to be printed out.

TAPE PUNCHING COMMANDS

The following commands can be used to punch out paper tapes that can be read in by the BIN loader.

T - punch leader/trailer

Typing a T will cause about four inches of leader/trailer tape (tape punched with 200 octal) to be punched. The T command also causes the accumulated checksum to be set to zero (cleared). The tape punch should be turned on immediately after the T command is given, and turned off when the Teletype stops punching. Note that as long as the punch is turned on, anything (including commands) typed into the Teletype will be punched onto the tape, so the punch should always be turned off before giving a command.

P - punch tape

Typing an octal number, nnnn, followed by a semi-colon (;) followed by a second octal number, mmmm, followed by a P, causes a tape corresponding to the contents of the block of memory beginning at location nnnn and ending at location mmmm to be punched. No checksum is punched at the end of the block so that several blocks can be punched together with one inclusive checksum. The punch should be off when the P command is given (otherwise the P command will be punched on the tape, causing errors when the tape is read). ODT halts before punching the tape to give the user time to turn on the punch. ODT is restarted by pressing the RUN/HLT button. Thus, the sequence of actions is: Make sure the punch is off; give P command; ODT halts; turn on punch; press RUN/HLT switch; ODT punches tape; turn off punch.

E - punch checksum and trailer

Typing an E will cause the accumulated checksum to be punched, followed by about four inches of leader/trailer tape. The checksum is also reset to zero (cleared). Like the P command, ODT halts after the command is typed to give the user time to turn on the punch. The sequence of actions to use is: Make sure punch is off; give E command; ODT halts; turn on punch; press continue switch; ODT punches checksum and trailer; turn off punch.

Example

Say that the user program occupies locations 100 through 170 and 200 through 377 of memory. To punch out the program, the user should give the following commands to ODT:

```
T
100;170P
200;377P
E
```

ERRORS

If an error occurs while typing a command, ODT types "?", ignores the erroneous command and waits for a new one. ODT detects that an error occured if the user types more than four digits for an octal number or if a decimal digit (8 or 9) is typed. It also detects an error if some character other than a command character (/, carriage return, line feed, ↑, ←, G, B, C, A, M, W, T, ;, P, E) is typed. ODT does not detect errors such as leaving the punch on while typing a command or giving a numerical argument for a command that should have none. For example, 100A is not detected as an error and is treated the same as typing A . If the user makes a mistake while typing a number, he can abort the command by purposely creating an error by typing too many digits or an illegal character.

Example

Here are some errors in ODT commands.

12345?	too many digits
248?	decimal digit
101X?	illegal character

FIELD COMMANDS

Intersil ODT also contains commands for working with multiple fields. These commands have no effect in the SAMPLER system, which does not contain hardware for handling fields.

The additional command symbols are period (.) and Q. Hence, these are not illegal symbols and typing them does, not cause ODT to recognize an error, but it may cause ODT to act strangely. If the user types one of these in by mistake, and ODT acts strangely, he should restart ODT.

Occasionally, ODT may print out 0. (a zero followed by a period) on the terminal. This just means that ODT thinks that you are operating in field zero (which you are) and may be ignored.

INTERRUPTS DURING ODT

The ODT program does not use any interrupt features. It does not disable the interrupt system if it is enabled by the user. The user must be cautious when enabling interrupt requests while the user program is 'controlled' by ODT via break points.

USEFUL SUBPROGRAMS

Some of the subprograms in the ODT ROM are general subroutines that the user may find it useful to call from his own program. These are described below.

CALLING AND RETURNING FROM A SUBROUTINE

For programs designed to exist in ROM, the JMS instruction cannot be used to call a subroutine because the first word of the routine cannot be used to store the return address. Instead, the ODT program makes use of a stack of return addresses kept in RAM by means of special call and return routines. (Refer to Applications Note M008: ROM Based Subroutine Calls With The IM6100.)

The user may make use of the ODT subroutine stack mechanism as follows. To call a routine requires two consecutive words; the octal number 4056 should be stored in the first word, and the address of the subroutine to be called should be in the second. Executing this piece of code causes the return address to be pushed onto the top of the stack and for control to be transferred to the given address. The AC, link and MQ are unaffected by executing the call instruction.

To return from the subroutine, execute a location containing the octal number 5461. This causes a return address to be taken from the top of the stack and returns control to that location. The AC, link and MQ are again unaffected.

The return address stack begins in location 75 and grows upwards (i.e. the first address pushed goes into location 75, the next into location 76, etc.) ODT may nest subroutines up to three deep, therefore, three stack locations in addition to what the user program needs should be reserved for use by ODT.

The stack pointer is stored in location 62, that is, location 62 contains the address of the location where the most recently pushed return address is stored.

IMPORTANT NOTE: The read and change commands in ODT involve calling a subroutine. Hence, giving the command "62/" will print out a value one greater than the true value of the stack pointer.

The user must be cautious when using the software stack for servicing interrupt requests since the software stack routine saves AC in a temporary location which could get overwritten by calling the stack routine again by an interrupt service routine before the first routine could exit.

TYPE SUBROUTINE

The TYPE subroutine prints on the Teletype the character whose ASCII code is stored in the AC. It also clears the AC before returning.

The subroutine may be called by using the call procedure described in the previous section. The starting address is 6102. Allocate one position on the stack for its use.

PNUM SUBROUTINE

The PNUM subroutine prints on the Teletype the contents of the AC as a four digit octal number followed by a space. It also clears the AC before returning. The starting address is 6107.

The subroutine may be called by using the call procedure described above. It makes use of the TYPE routine described above, therefore, two stack positions should be allocated for its use.

Example

Here is a subroutine that counts from 0000 to 7777 octal, printing each number followed by a space and an asterisk on the Teletype.

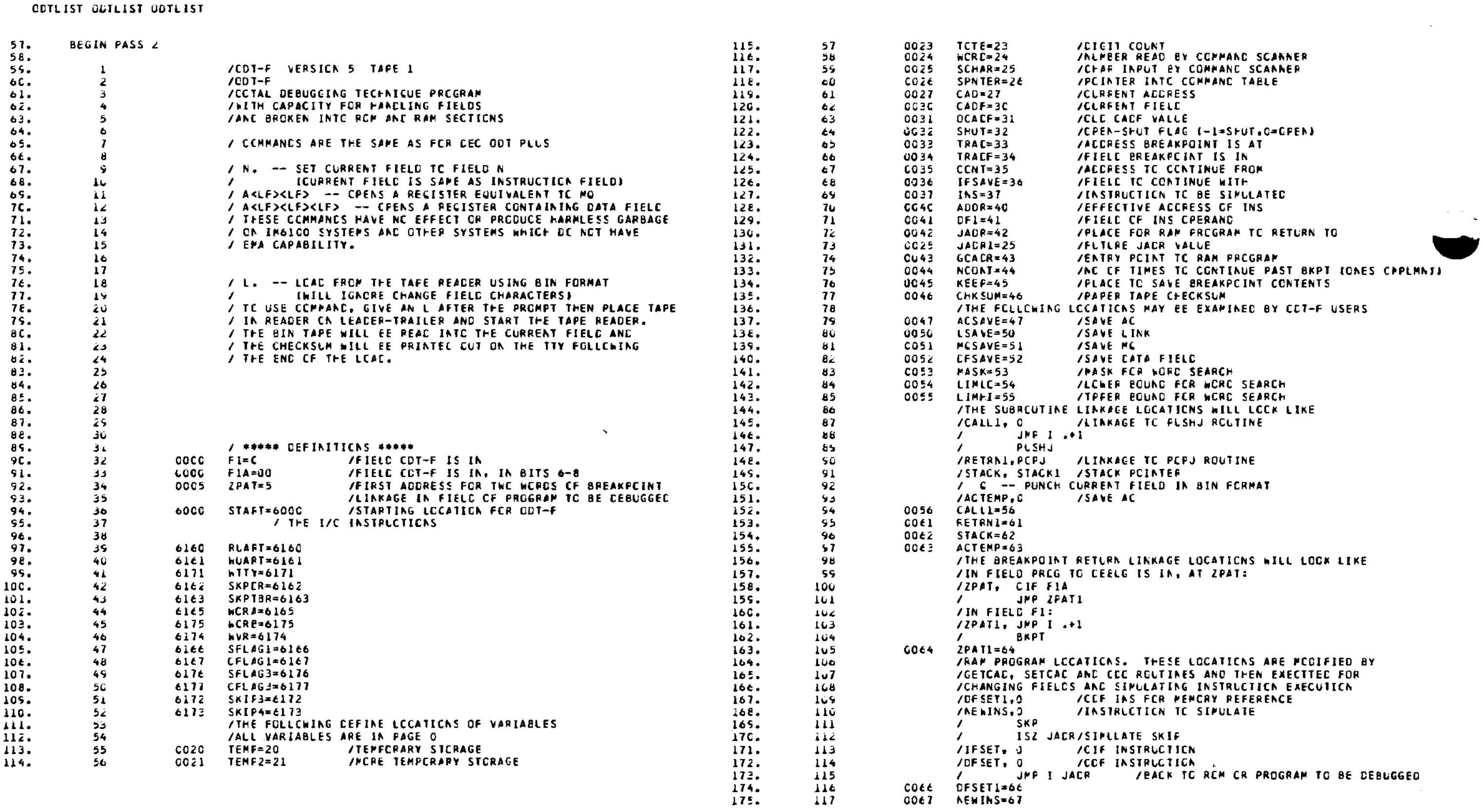
```
                              CALL    =   4056
                              RETURN  =   5461
                              TYPE    =   6102
                              PNUM    =   6107


   200      7200         COUNT, CLA                 /set count to zero
   201      3214                DCA X               /count stored in X
   202      1214          LOOP, TAD X               /get count
   203      4056                CALL
   204      6107                PNUM                /print it and space
   205      1213                TAD ASTRSK
   206      4056                CALL
   207      6102                TYPE                /print asterisk
   210      2214                ISZ X               /increment count
   211      5202                JMP LOOP            /repeat loop
   212      5461                RETURN              /return
   213      0252        ASTRSK, 252                 /ASCII for asterisk
   214      0000             X, 0                   /count stored here,
                                                     this must be a RAM
                                                     location (all others
                                                     can be ROM)
```

Three stack locations should be reserved for calling COUNT, one for the return address from COUNT itself, and two for using the PNUM subroutine. A stack location need not be allocated for calling TYPE because the two locations required for PNUM are unused when it is called. (In general, for each subroutine reserve one location plus the maximum number of locations required by all the subroutines called by it.)

Since three stack locations should also be reserved for ODT and the stack begins in location 75, locations 75 through 102 should be reserved for use by the stack.

PROGRAM LISTING:



```
ODTLIST ODTLIST ODTLIST

     BEGIN PASS 2

                   /ODT-F  VERSION 5  TAPE 1
                   /ODT-F
                   /OCTAL DEBUGGING TECHNIQUE PROGRAM
                   /WITH CAPACITY FOR HANDLING FIELDS
                   /AND BROKEN INTO ROM AND RAM SECTIONS

                   / COMMANDS ARE THE SAME AS FOR DEC ODT PLUS

                   / N.  -- SET CURRENT FIELD TO FIELD N
                   /        (CURRENT FIELD IS SAME AS INSTRUCTION FIELD)
                   / A<LF><LF>  -- OPENS A REGISTER EQUIVALENT TO MQ
                   / A<LF><LF><LF>  -- OPENS A REGISTER CONTAINING DATA FIELD
                   / THESE COMMANDS HAVE NO EFFECT OR PRODUCE HARMLESS GARBAGE
                   / ON IM6100 SYSTEMS AND OTHER SYSTEMS WHICH DO NOT HAVE
                   / EMA CAPABILITY.


                   / L.  -- LOAD FROM THE TAPE READER USING BIN FORMAT
                   /        (WILL IGNORE CHANGE FIELD CHARACTERS)
                   / TO USE COMMAND, GIVE AN L AFTER THE PROMPT THEN PLACE TAPE
                   / IN READER ON LEADER-TRAILER AND START THE TAPE READER.
                   / THE BIN TAPE WILL BE READ INTO THE CURRENT FIELD AND
                   / THE CHECKSUM WILL BE PRINTED OUT ON THE TTY FOLLOWING
                   / THE END OF THE LOAD.






                   / ***** DEFINITIONS *****
            0000   FI=0             /FIELD ODT-F IS IN
            0000   FIA=00           /FIELD ODT-F IS IN, IN BITS 6-8
            0005   ZPAT=5           /FIRST ADDRESS FOR TWO WORDS OF BREAKPOINT
                                    /LINKAGE IN FIELD OF PROGRAM TO BE DEBUGGED
            6000   START=6000       /STARTING LOCATION FOR ODT-F
                         / THE I/O INSTRUCTIONS

            6160   RUART=6160
            6161   WUART=6161
            6171   WTTY=6171
            6162   SKPCR=6162
            6163   SKPTBR=6163
            6165   WCRA=6165
            6175   WCRB=6175
            6174   WVR=6174
            6166   SFLAG1=6166
            6167   CFLAG1=6167
            6176   SFLAG3=6176
            6177   CFLAG3=6177
            6172   SKIP3=6172
            6173   SKIP4=6173
                   /THE FOLLOWING DEFINE LOCATIONS OF VARIABLES
                   /ALL VARIABLES ARE IN PAGE 0
            0020   TEMP=20          /TEMPORARY STORAGE
            0021   TEMP2=21         /MORE TEMPORARY STORAGE
            0023   TCTE=23          /DIGIT COUNT
            0024   WORD=24          /NUMBER READ BY COMMAND SCANNER
            0025   SCHAR=25         /CHAR INPUT BY COMMAND SCANNER
            0026   SPNTER=26        /POINTER INTO COMMAND TABLE
            0027   CAD=27           /CURRENT ADDRESS
            0030   CADF=30          /CURRENT FIELD
            0031   OCADF=31         /OLD CADF VALUE
            0032   SHUT=32          /OPEN-SHUT FLAG (-1=SHUT,0=OPEN)
            0033   TRAD=33          /ADDRESS BREAKPOINT IS AT
            0034   TRADF=34         /FIELD BREAKPOINT IS IN
            0035   CONT=35          /ADDRESS TO CONTINUE FROM
            0036   IFSAVE=36        /FIELD TO CONTINUE WITH
            0037   INS=37           /INSTRUCTION TO BE SIMULATED
            0040   ADDR=40          /EFFECTIVE ADDRESS OF INS
            0041   DF1=41           /FIELD OF INS OPERAND
            0042   JADR=42          /PLACE FOR RAM PROGRAM TO RETURN TO
            0025   JADR1=25         /FUTURE JADR VALUE
            0043   GOADR=43         /ENTRY POINT TO RAM PROGRAM
            0044   NCONT=44         /NO OF TIMES TO CONTINUE PAST BKPT (ONES CMPLMNT)
            0045   KEEP=45          /PLACE TO SAVE BREAKPOINT CONTENTS
            0046   CHKSUM=46        /PAPER TAPE CHECKSUM
                   /THE FOLLOWING LOCATIONS MAY BE EXAMINED BY ODT-F USERS
            0047   ACSAVE=47        /SAVE AC
            0050   LSAVE=50         /SAVE LINK
            0051   MQSAVE=51        /SAVE MQ
            0052   DFSAVE=52        /SAVE DATA FIELD
            0053   MASK=53          /MASK FOR WORD SEARCH
            0054   LIMLO=54         /LOWER BOUND FOR WORD SEARCH
            0055   LIMHI=55         /UPPER BOUND FOR WORD SEARCH
                   /THE SUBROUTINE LINKAGE LOCATIONS WILL LOOK LIKE
                   /CALL1, 0        /LINKAGE TO PUSHJ ROUTINE
                   /       JMP I .+1
                   /       PUSHJ
                   /RETRN1,POPJ     /LINKAGE TO POPJ ROUTINE
                   /STACK, STACK1   /STACK POINTER
                   /  G  -- PUNCH CURRENT FIELD IN BIN FORMAT
                   /ACTEMP,0        /SAVE AC
            0056   CALL1=56
            0061   RETRN1=61
            0062   STACK=62
            0063   ACTEMP=63
                   /THE BREAKPOINT RETURN LINKAGE LOCATIONS WILL LOOK LIKE
                   /IN FIELD PROG TO DEBUG IS IN, AT ZPAT:
                   /ZPAT,  CIF FIA
                   /       JMP ZPAT1
                   /IN FIELD FI:
                   /ZPAT1, JMP I .+1
                   /       BKPT
            0064   ZPAT1=64
                   /RAM PROGRAM LOCATIONS.  THESE LOCATIONS ARE MODIFIED BY
                   /GETCAD, SETCAD AND ODD ROUTINES AND THEN EXECUTED FOR
                   /CHANGING FIELDS AND SIMULATING INSTRUCTION EXECUTION
                   /DFSET1,0        /CDF INS FOR MEMORY REFERENCE
                   /NEWINS,0        /INSTRUCTION TO SIMULATE
                   /       SKP
                   /       ISZ JADR/SIMULATE SKIP
                   /IFSET, 0        /CIF INSTRUCTION
                   /DFSET, 0        /CDF INSTRUCTION
                   /       JMP I JADR      /BACK TO ROM OR PROGRAM TO BE DEBUGGED
            0066   DFSET1=66
            0067   NEWINS=67
            0072  IFSET=72
            0073  DFSET=73
                  /FIRST LOCATION OF RETURN ADDRESS STACK
            0074  STACK1=74
                  PAUSE
                  /ODT-F  VERSION 5  TAPE 2
                  / ***** FIRST ROM PAGE
            6000  *START
                  / ***** INITIALIZE ROUTINE *****
      6000  7200  INIT,   CLA
      6001  1245          TAD SUB1          /INIT SUBROUTINE LINKAGE
      6002  3057          DCA CALL1+1
      6003  1246          TAD SUB2
      6004  3060          DCA CALL1+2
      6005  1247          TAD SUB3
      6006  3061          DCA RETRN1
      6007  1250          TAD STACKI
      6010  3062          DCA STACK
      6011  1251          TAD RAM1          /INIT RAM PROGRAM LOCATIONS
      6012  3070          DCA NEWINS+1
      6013  1252          TAD RAM2
      6014  3071          DCA NEWINS+2
      6015  1253          TAD RAM3
      6016  3074          DCA DFSET+1
      6017  1254          TAD TRAP3         /INIT BKPT JUMP LINKAGE
      6020  3064          DCA ZPAT1
      6021  1255          TAD TRAP4
      6022  3065          DCA ZPAT1+1
      6023  7040          CMA               /SET DFSAVE TO "UNDEFINED" VALUE
      6024  3052          DCA DFSAVE
      6025  3030          DCA CADF          /START IN FIELD 0
      6026  1256          TAD KCRA
      6027  6165          WCRA
      6030  7300          CLA CLL
      6031  1257          TAD KCRB
      6032  6175          WCRB
      6033  7300          CLA CLL
      6034  1260          TAD KTTY
      6035  6171          WTTY
      6036  7300          CLA CLL
      6037  3031          DCA CCADF
      6040  3046  INIT1,  DCA CHKSUM        /CLEAR CHECKSUM
      6041  4056          CALL              /TYPE CR.LF
      6042  6400          CRLF
      6043  5644          JMP I PCREAD      /GO TO COMMAND SCANNER
                  / *** CONSTANTS
      6044  6200  PCREAD, READ
      6045  5460  SUB1,   JMP I CALL1+2
      6046  6061  SUB2,   PUSHJ
      6047  6072  SUB3,   POPJ
      6050  0074  STACKI, STACK1
      6051  7410  RAM1,   SKP
      6052  2042  RAM2,   ISZ JADR
      6053  5442  RAM3,   JMP I JADR
      6054  5465  TRAP3,  JMP I ZPAT1+1
      6055  7236  TRAP4,  BKPT
      6056  7200  KCRA,   7200
      6057  1560  KCRB,   1560
      6060  7600  KTTY,   7600
                  / ***** CALL AND RETURN ROUTINES *****
                  /JMS WILL NOT WORK IN A ROM SINCE THE FIRST WORD OF THE
                  /SUBROUTINE CANNOT BE WRITTEN WITH THE RETURN ADDRESS.SSS.
                  /HENCE, A SUBROUTINE IS CALLED BY
                  /       CALL
                  /       <SUBROUTINE NAME>
                  /AND RETURNED FROM BY
                  /       RETURN
                  /WHERE
            4056  CALL=JMS CALL1          /CALL SUBROUTINE OPCODE
            5461  RETURN=JMP I RETRN1     /RETURN FROM SUBROUTINE OPCODE
                  /CALL THE FOLLOWING ROUTINES THROUGH THE SUBROUTINE
                  /LINKAGE IN PAGE ZERO (SEE TAPE 1)
                  /ROUTINE TO CALL A SUBROUTINE AND PUSH
                  /RETURN ADDRESS ON STACK
      6061  3063  PUSHJ,  DCA     ACTEMP  /SAVE AC
      6062  2062          ISZ     STACK   /UPDATE STACK PTR
      6063  1056          TAD     CALL1   /GET USER RETURN ADDRESS
      6064  7001          IAC     /INCREMENT PAST ARGUMENT
      6065  3462          DCA     I STACK /PUT IT ON STACK
      6066  1456          TAD     I CALL1 /GET USER ENTRY ADDRESS
      6067  3056          DCA     CALL1   /PUT IT IN CALL
      6070  1063          TAD     ACTEMP  /RESTORE AC
      6071  5456          JMP     I CALL1 /GO TO USER ROUTINE
                  /ROUTINE TO RETURN FROM A SUBROUTINE, POPPING RETURN ADDRESS
                  /OFF STACK
      6072  3063  POPJ,   DCA     ACTEMP  /SAVE AC
      6073  1462          TAD     I STACK /GET RETURN ADDRESS
      6074  3056          DCA     CALL1   /SAVE IT IN CALL1
      6075  7060          CMA CML /-1 IN AC, COMPLEMENT L
      6076  1062          TAD     STACK   /DECREMENT STACK PTR
                                  /(CARRY RESTORES L)
      6077  3062          DCA     STACK   /RESTORE UPDATED STACK PTR.
      6100  1063          TAD     ACTEMP  /RESTORE AC
      6101  5456          JMP     I CALL1 /RETURN
                  / ***** PRINT SUBROUTINES *****
                  /SUBROUTINE TO TYPE A CHARACHTER
                  /CHAR ASSUMED IN AC
      6102  6161  TYPE,   WUART     /OUTPUT CHAR TO TTY
      6103  6163          SKPTBR      /READY FOR NEXT CHAR?
      6104  5303          JMP     .-1     /LOOP IF NOT
      6105  7200          CLA     /CLEAR AC
      6106  5461          RETURN  /RETURN
                  /SUBROUTHNE TO PRINT A NUMBER
                  /PRINT CONTENTS OF AC IN OCTAL FOLLOWED BY A SPACE
      6107  3020  PNUM,   DCA     TEMP    /SAVE NUMBER
      6110  1334          TAD     M4      /INITIALIZE COUNT OF DIGITS
      6111  3023          DCA     TOTE    / PRINTED
      6112  1020          TAD     TEMP    /GET BACK NUMBER
      6113  7004          RAL     /FIRST SHIFT INTO LINK
                  /LOOP 4 TIMES -, PRINT A DIGIT
      6114  7004  PNUM2,  RAL     /SHIFT AC,L THREE LEFT
      6115  7006          RTL
      6116  3020          DCA     TEMP    /SAVE NUMBER
      6117  1020          TAD     TEMP    /GET IT BACK
      6120  0335          AND     P7      /ISOLATE DIGIT
      6121  1340          TAD     P260    /CONVERT TO ASCII
      6122  4056          CALL; TYPE      /TYPE IT OUT
      6123  6102
      6124  1020          TAD     TEMP    /GET BACK AC (NOTE L STILL SAME)
      6125  2023          ISZ     TOTE    /FOURTH ITERATION?
      6126  5314          JMP     PNUM2   /LOOP IF NOT.
                  /PRINT A SPACE
      6127  7200          CLA
      6130  1337          TAD     P240    /GET ASCII CODE FOR SPACE
      6131  4056          CALL; TYPE      /TYPE IT OUT
      6132  6102
      6133  5461          RETURN  /RETURN
                  / *** CONSTANTS
      6134  7774  M4,     -4
      6135  0007  P7,     7
      6136  0007  7
      6137  0240  P240,   240
      6140  0260  P260,   260
                  PAUSE
                  /ODT-F  VERSION 5 TAPE 3
                  / ***** SECOND ROM PAGE
            6200  *START+200
                  / ***** COMMAND SCANNER *****
                  /THE COMMAND SCANNER INPUTS A COMMAND OF THE FORM
                  /[<NUMBER>]<I>]<CHAR>  (WHERE THE NUMBER IS OPTIONAL).
                  /IT STORES THE NUMBER IN "WORD" AND JUMPS TO THE
                  /ROUTINE ASSOCIATED WITH THE CHARACHTER.
                  /INITIALIZE NUMBER SCANNER
      6200  7200  READ,   CLA
      6201  3024          DCA  WORD       /SET WORD TO ZERO
      6202  1333          TAD  SM5
      6203  3023          DCA  TOTE       /SET TOTE TO -5
                  /INPUT A CHAR
      6204  6162  READ1,  SKPDR              /CHAR IN INPUT BUFFER?
      6205  5204          JMP  .-1        /LOOP IF NOT
      6206  7200    CLA;     RUART                /PUT CHAR INTO AC
      6207  6160
      6210  0336   AND K0377;       DCA  SCHAR      /SAVE CHAR
      6211  3025
                  /ECHO CHAR ON TTY
      6212  1025          TAD  SCHAR      /RETRIEVE CHAR
      6213  4056          CALL
      6214  6102          TYPE            /TYPE IT OUT
                  /NUMBER SCANNER
                  /SEE IF SCHAR IS OCTAL DIGIT
      6215  1025          TAD  SCHAR      /GET CHAR ("0"=260,"7"=267)
      6216  1332          TAD  SM270      /SUB 270 ("0"=-10,"7"=-1)
      6217  7500          SMA             /IF AC NOT NEG. THEN CHAR HAS
                                          /CODE GTR THAN THAT OF DIGIT
      6220  5235          JMP  READ2      /SO GO TO CHAR HANDLER.
      6221  1334          TAD  SP10       /ADD 10 ("0"=0,"7"=7)
      6222  7510          SPA             /IF AC NEG, THEN CHAR HAS
                                          /CODE LESS THAN THAT OF DIGIT
      6223  5235          JMP  READ2      /SO GO TO CHAR HANDLER
                  /ADD DIGIT TO PARTIAL NUMBER IN WORD
      6224  3020          DCA  TEMP       /SAVE DIGIT
      6225  1024          TAD  WORD       /GET NUMBER SO FAR
      6226  7104          CLL RAL         /SHIFT AC LEFT THREE BITS
      6227  7006          RTL
      6230  1020          TAD  TEMP       /ADD IN NEW DIGIT
      6231  3024          DCA  WORD       /SAVE RESULT IN WORD
                  /CHECK FOR TOO MANY DIGITS, RETURN TO CHAR RDR
      6232  2023          ISZ  TOTE       /5 DIGITS TYPED?
      6233  5204          JMP  READ1      /NO - GET NEXT CHAR
      6234  5337          JMP  ERROR      /YES - GO TO ERROR
                  /CHAR SCANNER
                  /FIND INPUT CHAR IN TABLE1 AND JUMP TO THE
                  /ASSOCIATED ROUTINE IN TABLE2
                  /INITIALHZE SEARCH LOOP
      6235  7200  READ2,  CLA
      6236  1256          TAD  BLIST      /GET PTR TO BEGIN OF TABLE1
      6237  3026          DCA  SPNTER     /INIT PTR INTO TABLE1
                  /SEARCH LOOP
      6240  1426  READ3,  TAD  I SPNTER   /GET CHAR FROM TABLE1
      6241  2026          ISZ  SPNTER     /POINT SPNTER AT NEXT CHAR
      6242  7510          SPA             /SKIP IF NOT END OF TABLE1
                                          /(TABLE1 FOLLOWED BY NEG #)
      6243  5337          JMP  ERROR      /CHAR NOT IN TABLE1 - ERROR
      6244  7041          CIA             /NEGATE TABLE CHAR
      6245  1025          TAD  SCHAR      /ADD INPUT CHAR
      6246  7640          SZA CLA         /SUM ZERO IF CHARS SAME, SKIP
                                          /IF SO
      6247  5240          JMP  READ3      /OTHERWISE, CONTINUE LOOP
                  /JUMP TO ASSOCIATED ROUTINE IN TABLE2
      6250  1026          TAD  SPNTER     /GET PTR INTO TABLE1
                                          /(SPNTR NOW POINTS ONE PAST
                                          / MATCHING CHAR).
      6251  1257          TAD  LTABL      /CONVERT INTO PTR INTO TABLE2
      6252  3020          DCA  TEMP       /PUT TABLE2 ENTRY
      6253  1420          TAD  I TEMP     / INTO AC
      6254  3020          DCA  TEMP       /JUMP TO LOC POINTED TO
      6255  5420          JMP  I TEMP     / BY TABLE ENTRY
      6256  6260  BLIST,  TABLE1          /POINTER TO BEGINNIG OF TABLE1
      6257  0022  LTABL,  TABLE2-TABLE1-1 /CONSTANT TO GET CORRESPONDING
                                          /  LOC IN TABLE2
                  / *** COMMAND SCANNER TABLE ***
                  /EACH ENTRY IN TABLE1 CONTAINS A CHAR.
                  /THE CORRESPONDING ENTRY IN TABLE2 CONTAINS A
                  /PROCEDURE ASSOCIATED WITH THAT CHAR.
      6260  0215  TABLE1, 215     / CR
      6261  0212          212     / LF
      6262  0257          257     / /
      6263  0256          256     / .
      6264  0301          301     / A
      6265  0315          315     / M
      6266  0337          337     / -
      6267  0336          336     /
      6270  0302          302     / B
      6271  0307          307     / G
      6272  0303          303     / C
      6273  0327          327     / W
      6274  0320          320     / P
      6275  0305          305     / E
      6276  0324          324     / T
      6277  0321          321     / Q
      6300  0273          273     / ;
      6301  0314          314     / L
      6302  7777          -1      /TABLE FOLLOWED BY NEG NUMBER
      6303  6616  TABLE2, CRDO
      6304  6625          LFDO
      6305  6601          SLDO
      6306  6654          OCTDO
      6307  6727          ADO
      6310  6730          MDO
      6311  6715          BADO
      6312  6670          UADO
      6313  7215          BDO
      6314  7200          GDO
      6315  7000          CDO
```

```
6316  7303                   WDO
6317  7502                   PDO
6320  7464                   EDO
6321  7471                   TDO
6322  7454                   GDO
6323  7451                   SEMICO
6324  7576                   BIN-2

      6332      *START+332


                / *** CONSTANTS
6332  7510      SP270,  -270
6333  7773      SM5,    -5
6334  0010      SP10,   10
6335  0277      SP277,  277
6336  0377      K0377, 0377;FALSE

      6337      *START+337

                / ***** ERROR HANDLER *****
                /TYPE "?" AND RETURN TO COMMAND SCANNER
6337  7200      ERROR,  CLA
6340  1335              TAD  SP277      /GET QUESTION MARK
6341  4056              CALL
6342  6102              TYPE            /TYPE IT OUT
6343  4056              CALL
6344  6400              CRLF            /TYPE CR,LF
6345  5200              JMP READ




                /ODT-F  VERSION 5  TAPE 4
                / ***** THIRD ROM PAGE
      6400      *START+400
                / ***** EXAMINE DEPOSIT ROUTINES *****
                /SUBROUTINE TO TYPE CR, LF AND SHUT REGISTER
6400  1257      CRLF,   TAD  TP215      /GET CR
6401  4056              CALL
6402  6102              TYPE            /TYPE IT
6403  1256              TAD  TP212      /GET LF
6404  4056              CALL
6405  6102              TYPE            /TYPE IT
6406  7040              CMA             /SET AC TO -1
6407  3032              DCA  SHUT       /STORE IN SHUT
6410  5461              RETURN          /RETURN
                /SUBROUTINE TO CLOSE REG
                /SEE IF REG ALREADY SHUT
6411  2032      CLOSE,  ISZ  SHUT       /SKIP IF SHUT=-1
6412  7410              SKP
6413  5461              RETURN          /RETURN IF REG ALREADY SHUT
                /SEE IF VALUE TYPED IN
6414  4056              CALL
6415  6423              NTYPED          /SEE IF NUMBER TYPED
6416  5461              RETURN          /RETURN IF NOT
                /STORE TYPED VALUE IN REG
6417  1024              TAD  WORD       /GET NEW VALUE
6420  4056              CALL
6421  6442              SETCAD          /STORE IT IN CAD
6422  5461              RETURN          /RETURN
                /SUBROUTINE TO SEE IF NUMBER HAS BEEN TYPED
                /SKIP ON RETURN IF ANY NUMBER HAS BEEN TYPED SINCE
                /LAST COMMAND
6423  1023      NTYPED, TAD  TOTE       /TOTE IS -5 INITIALLY, AND
6424  7041              CIA             / INCREMENTED ONLY IF A NUMBER
6425  1255              TAD  TM5        / IS TYPED
6426  7640              SZA CLA         /SKIP IF TOTE=-5
6427  2462              ISZ  I STACK    /INCREMENT RETURN ADDR
6430  5461              RETURN
                /SUBROUTINE TO GET CONTENTS OF CURRENT LOCATION
                /NEW CONTENTS RETURNED IN AC
6431  1030      GETCAD, TAD  CADF       /GET FIELD OF CURRENT LOC
6432  1260              TAD  TP6201     /ADD IN "CDF" INSTRUCTION
6433  3073              DCA  DFSET      /STORE IT IN RAM PROGRAM
6434  1261              TAD  PGTCD1     /SET RAM PROG TO RETURN TO GETCD1
6435  3042              DCA  JADR
6436  5073              JMP  DFSET      /EXECUTE RAM PROGRAM - CHANGE FIELD
                                        /TO THAT OF CURRENT LOCATION
6437  1427      GETCD1, TAD  I CAD      /GET CONTENTS OF CURRENT LOC
6440  6201              CDF  F1A        /RESTORE DATA FIELD
6441  5461              RETURN          /RETURN
                /SUBROUTINE TO SET CONTENTS OF CURRENT LOCATION
                /NEW CONTENTS PASSED IN AC
6442  3020      SETCAD, DCA  TEMP       /SAVE AC
6443  1030              TAD  CADF       /GET FIELD OF CURRENT LOC
6444  1260              TAD  TP6201     /ADD IN "CDF" INSTRUCTION
6445  3073              DCA  DFSET      /RTORE IT IN RAM
6446  1262              TAD  PSTCD1     /SET RAM PROG TO RETURN TO SETCD1
6447  3042              DCA  JADR
6450  5073              JMP  DFSET      /EXECUTE RAM PROGRAM - CHANGE FIELD
                                        /TO THAT OF CURRENT LOCATION
6451  1020      SETCD1,  TAD  TEMP      /RESTORE AC
6452  3427              DCA  I CAD      /SET CURRENT LOC TO NEW VALUE
6453  6201              CDF  F1A        /RESTORE DATA FIELD
6454  5461              RETURN          /RETURN
                / *** CONSTANTS
6455  7773      TM5,    -5
6456  0212      TP212,  212
6457  0215      TP215,  215
6460  6201      TP6201, 6201
6461  6437      PGTCD1, GETCD1
6462  6451      PSTCD1, SETCD1
6463  6172      LISN,   SKIP3           / THE TTY LISN ROUTINE FOR THE BIN
6464  7000              NOP             / LOADER
6465  6166              SFLAG1          / SET READER RUN
6466  6172              SKIP3
6467  5266              JMP .-1
6470  6167              CFLAG1
6471  6162              SKPDR
6472  5271              JMP .-1
6473  7200              CLA
6474  6160              RUART
6475  0277              AND TTYM
6476  5461              RETURN
6477  0377      TTYM,   0377

                /ODT-F  VERSION 5  TAPE 5
                / ***** EXAMINE/DEPOSIT ROUTINES - CONTINUED *****
                / ***** FOURTH ROM PAGE
      6600      *START+600
                /TRANSFER ADDRESS TO COMMANAMAND SCANNER
6600  6200      COMM,   READ
                /SLASH HANDLER
                /OPEN LOCATION "CADF"."CAD"
                /IF NUMBER TYPED, SET CAD
6601  1031      SLDO,   TAD CCADF
6602  3030              DCA CADF
6603  4056              CALL
6604  6423              NTYPED          /SEE IF NUMBER TYPED
6605  5210              JMP  SLDO1      /NO NUM TYPED. LEAVE CAD ALONE
6606  1024              TAD  WORD       /ELSE GET NUM TYPED
6607  3027              DCA  CAD        /AND SET CAD TO IT
                /TYPE CONTENTS OF "CADF"."CAD"
6610  3032      SLDO1,  DCA  SHUT       /SET REGISTER STATUS TO "OPEN"
6611  4056              CALL
6612  6431              GETCAD          /GET CONTENTS OF CURRENT LOC
6613  4056              CALL
6614  6107              PNUM            /TYPE IT OUT
6615  5600              JMP  I COMM     /RETURN TO COMMAND SCANNER
                /CR HANDLER
                /CLOSE LOCATION
6616  4056      CRDO,   CALL
6617  6411              CLOSE           /CLOSE LOCATION
6620  1031              TAD CCADF
6621  3030              DCA CADF
6622  4056              CALL
6623  6400              CRLF            /TYPE CR, LF
6624  5600              JMP  I COMM     /GET NEXT COMMAND
                /LF HANDLER
                /CLOSE LOCATION AND OPEN NEXT LOCATION
6625  4056      LFDO,   CALL
6626  6411              CLOSE           /CLOSE LOCATION
6627  1344              TAD  SP215      /GET CR
6630  4056              CALL
6631  6102              TYPE            /TYPE IT
6632  4056              CALL
6633  6102              TYPE            /TYPE A NULL (GIVES TIME FOR CR)
6634  2027              ISZ  CAD        /POINT CAD AT NEXT LOC
6635  7000              NOP             /IN CASE ISZ SKIPS
6636  1030              TAD CADF        /COMPARE CURRENT AND SAVED FIELDS
6637  7041              CIA
6640  1031              TAD CCADF
6641  7650              SNA CLA         /DIFFERENT?
6642  5245              JMP LFDO1       /NO , GO ON
6643  4056              CALL            /YES - PRINT FIELD
6644  7347              PFIELD
                /THE FOLLOWING IS AN ENTRY POINT FOR ANY ROUTINE
                /THAT TYPES OUT WHAT LOCATION IT IS OPENING.
6645  1027      LFDO1,  TAD  CAD        /GET CAD
6646  4056              CALL
6647  6107              PNUM            /TYPE IT OUT
6650  1345              TAD  SP257      /GET ASCII FOR "/"
6651  4056              CALL
6652  6102              TYPE            /TYPE IT OUT
6653  5210              JMP  SLDO1      /REST IS LIKE OPEN REG ROUTINE
                /. HANDLER
                /SET FIELD OF CURRENT LOCATION
6654  1024      DOTDO,  TAD  WORD       /GET FIELD SPEC FROM WORD
6655  7104              CLL RAL         /FIELD SPEC INTO BITS 6-8
6656  7006              RTL
6657  3030              DCA  CADF       /STORE IT AS CURRENT FIELD
6660  1030              TAD CADF
6661  3031              DCA CCADF
6662  1052              TAD DFSAVE      /DFSAVE INITIALIZED?
6663  7700              SMA CLA
6664  5600              JMP I COMM      /YES - RETURN
6665  1030              TAD CADF        /NO - INIT DFSAVE TO CADF
6666  3052              DCA DFSAVE
6667  5600              JMP  I COMM     /GET NEXT COMMAND
                /  HANDLER
                /PRETEND CAD IS A MEMORY REFERENCE INS AND OPEN
                /LOCATION REFERENCED.  IGNORE I BIT.
6670  4056      UADO,   CALL
6671  6411              CLOSE           /CLOSE LOCATION
6672  4056              CALL
6673  6400              CRLF            /TYPE CR,LF
6674  4056              CALL
6675  6431              GETCAD          /GET CONTENTS
6676  3020              DCA  TEMP       /SAVE THEM IN TEMP
6677  1031              TAD CCADF
6700  3030              DCA CADF
6701  1020              TAD  TEMP       /GET BACK CONTENTS
6702  0342              AND  SP177      /ISOLATE PAGE ADDR BITS
6703  3021              DCA  TEMP2      /AND SAVE THEM IN TEMP2
6704  1020              TAD  TEMP       /GET CONTENTS AGAIN
6705  0343              AND  SP200      /ISOLATE PAGE ZERO BIT
6706  7650              SNA CLA         /REFERENCE TO PAGE ZERO?
6707  5312              JMP  UADO1      /YES - SKIP NEXT CODE
6710  1027              TAD  CAD        /NO - GET CURRENT ADR
6711  0351              AND  SP7600     /ISOLATE PAGE NUMBER
6712  1021      UADO1,  TAD  TEMP2      /ADD IN PAGE ADDR
6713  3027              DCA  CAD        /PUT INTO CAD
6714  5245              JMP  LFDO1      /REST LIKE LF
                /  HANDLER
                /OPEN LOC POINTED AT BY CURRENT LOC
6715  4056      BADO,   CALL
6716  6411              CLOSE           /CLOSE CURRENT LOC
6717  4056              CALL
6720  6400              CRLF            /TYPE CR,LF
6721  4056              CALL
6722  6431              GETCAD          /GET CONTENTS OF CURRENT LOC
6723  3027              DCA  CAD        /MAKE IT INTO NEW LOC
6724  1031              TAD CCADF
6725  3030              DCA CADF
6726  5245              JMP  LFDO1      /REST LIKE LF
                / A,M,I HANDLER
                /OPEN LOC CONTAINING AC, MASK, OR INSTRUCTION FIELD
6727  1337      ADO,    TAD ADDR1       /GET REGISTER ADDRESS - ACSAVE
6730  1340      MDO,    TAD ADDR2       /OR MASK
6731  3027              DCA  CAD        /PUT INTO CAD
6732  1030              TAD CADF
6733  3031              DCA CCADF
6734  1341              TAD ADDR4
6735  3030              DCA CADF
6736  5210              JMP  SLDO1      /REST LIKE SLASH
6737  7774      ADDR1,  ACSAVE-MASK
6740  0053      ADDR2,  MASK
6741  0000      ADDR4,  F1A
                / *** CONSTANTS
6742  0177      SP177,  177
6743  0200      SP200,  200
6744  0215      SP215,  215
6745  0257      SP257,  257
```

```
6746  0260  SP260,  26C
6747  04CC  SP4CO,  4CO
6750  6201  SP6201, 6201
6751  76CC  SP7600, 76CC
            PAUSE
            /CDT-F  VERSICN 5  TAPE 6
            / ***** BREAKPCINT/CCNTINUE RCUTINES *****
            / ***** FIFTH RCM PAGE
      7000  *START+1000
            / C HANDLER
            /CONTINUE EXECUTICN CF PRCGRAM
7000  1031  CCO,    TAD CCADF
7001  303C          DCA CADF
7002  4056          CALL
7003  64CC          CRLF
7004  1024          TAD WCRD          /SET CCNTINUE COUNT TC
7005  704C          CMA               / HTERATE PAST BREAKPCINT
7006  3044          DCA NCCNT         / SPECIFIED NUMBER CF TIMES
            /SIMULATE EXECUTICN CF THE INSTRUCTICN IN
            /LCCATICN CCNTF.CCNT
            /HANDLE ICT CR OPERATE INSTRUCTION
7007  1035  CCNTO,  TAC CCNT          /SET CADF.CAC TO CCNTF.CONT
7010  3027          DCA CAC
7011  1036          TAC IFSAVE        /IFSAVE IS CCNTINUE FHELD
7012  303C          DCA CADF
7013  4056          CALL              /GET CCNTINUE INSTRUCTION
7014  6431          GETCAD
7015  3037          DCA INS           /STORE IT IN INS
7016  2035          ISZ CCNT          /PCINT CONT AT NEXT INS TC EXECUTE
7017  1035          TAC CCNT          /SET ACCR TO CCNT PROG EXEC FROM
7020  3025          DCA JACR1
7021  1037          TAC INS
7022  71CC          CLL
7023  1346          TAC FP2000        /CVERFLOW SETS L FCR ICT CR OPER
7024  7620          SNL CLA
7025  5233          JMP CCNT1         /JMP IF NCT ICT CR CPER
7026  1037          TAD INS /NEWINS  INS
7027  3067          DCA NEWINS
7030  1334          TAC FNWINS        /SET RAM PROGRAM EXECUTION TO
7031  3043          DCA GCACR         /BEGIN FRCM "NEWINS"
7032  574C          JMF I PCNT5        /GC TC EXECUTE RAM
            /PUT EFFECTIVE ADDRESS CF MEMCRY REFERENCE INSTRUCTICN
            /INTO ADCR
7033  1037  CONT1,  TAD INS
7034  0343          ANC FP177         /GET ACCRESS CN PAGE
7035  304C          DCA ADCR
7036  1037          TAC INS           /TEST FOR ZERC PAGE
7037  C344          ANC FP2CC
7040  765C          SNA CLA
7041  5247          JMP CCNT2         /ZERC PAGE - GC CN
7042  704C          CMA               /CURRENT PAGE - GET INS LCCATION
7043  1035          TAC CCNT          /CCNT PCINTS CNE PAST INS LOC
7044  C351          ANC FP76CC        /ISCLATE PAFE NUMBER
7045  104C          TAC ADCR          /ACC TC GET
7046  3040          CCA ADCR          /NEW EFFECTIVE ADDRESS
            /HANDLE INCIRECT REFERENCE
7047  1036  CCNT2,  TAC IFSAVE        /ASSUME EA IS IN INSTRUCTICN FIELD
7050  3041          DCA CF1           / SO SET CF1  IFSAVE
7051  1037          TAD INS           /TEST INS FCR INCIRECT
7052  C345          AND FP4C0
7053  765C          SNA CLA
```

```
7054  5276          JMP CCNT3         /CIRECT - GC CN
7055  1052          TAD CFSAVE        /INDIRECT - SC EA IS IN DATA FIELC
7056  3041          CCA CF1           / SC SCT CF1  CFSAVE
7057  1040          TAC ACCR          /SET  CAD  ADCR
7060  3027          CCA CAC
            /HANDLE AUTO-INCREMENT
7061  104C          TAC ACCR          /SEE IF ADCR IS AUTO-INC REGISTER
7062  0352          ANC FP7770
7063  1342          TAD FM10
7064  764C          SZA CLA
7065  5273          JMP CCNT2A        /NCT AUTO-INCR - GC CN
7066  4056          CALL              /AUTC-INCR - GET REGISTER VALUE
7067  6431          GETCAC
7070  70C1          IAC               /INCREMENT IT
7071  4056          CALL              /ANC RESTCRE
7072  6442          SETCAD
            /FINISH HANCLING INDIRECT
7073  4056  CONT2A, CALL              /GET CCNTENTS OF ACCR
7074  6431          GETCAD
7075  3C4C          DCA ADCR          /AND MAKE THAT NEW ADCR
            /HANCLE JMS, JMF INSTRUCTICNS
7076  1037  CCNT3,  TAD INS           /SEE IF INS IS JMP CR JMS
7077  71CC          CLL
7100  035C          AND FP7COC
7101  1347          TAC FP40CC
7102  742C          SNL               /L=1 IF INS IS JMS CR JMP
7103  5321          JMP CCNT4         /NCT JMS CR JMP - GC CN
7104  764C          SZA CLA           /JMS CR JMP - AC=0 IF INS IS JMS
7105  5314          JMP CCNT3A        /NCT JMS - GO ON
7106  1040          TAD ADCR          /JMS - EMULATE JMS
7107  3027          DCA CAC
7110  1035          TAC CCNT
7111  4056          CALL
7112  6442          SETCAD            /SET RETURNRURN ACCRESS
7113  204C          ISZ ADCR          /PCINT ADDR AT SUBRCUTINE BCDY
7114  104C  CCNT3A, TAC ACCR          /GET JUMP DESTINATICN
7115  3025          DCA JADR1         /AND SET IT
7116  1335          TAD FIFSET        /START RAM PRCG EXECUTION
7117  3043          DCA GOADR         /FRCM IFSET
7120  5740          JMP I PCNT5       /GC TC EXECUTE RAM PRCG
            /HANDLE AND, TAC, CCA, ISZ INSTRUCTICNS
7121  7200  CCNT4,  CLA
7122  1037          TAD INS           /GET INS
7123  C350          AND FP7CCC        /ISCLATE OPCCCE
7124  1337          TAD IADCR
7125  3067          DCA NEWINS        /SET  NEWINS  OPCCCE I ADCR
7126  1041          TAD CF1
7127  1341          TAC CDFO
7130  3066          DCA DFSET1        /SET  DFSET1  CDF CF1
7131  1336          TAC PCFST1        /SET RAM PROG EXECUTICN
7132  3043          CCA GCACR         / TO BEGIN FRCM DFSET1
7133  5740          JMP I PCNT5       /GC TC EXECUTE RAM PRCG
            / *** CONSTANTS
7134  0067  PNWINS, NEWINS
7135  0072  PIFSET, IFSET
7136  0066  PCFST1, DFSET1
7137  0440  IADCR,  400 ADCR
7140  7400  PCNT5,  CONT5
7141  62C1  CCFC,   CDF
7142  7770  FM1C,   -1C
7143  C177  FF177,  177
```

```
7144  0200  FP2C0,  200
7145  0400  FP400,  400
7146  2000  FP2C00200C, 20C0
7147  40C0  FP4000, 4000
7150  7000  FP7C00, 7000
7151  76C0  FP7600, 7600
7152  777C  FP7770, 777C
            PAUSE
            /ODT-F  VERSICN 5  TAPE 7
            / ***** BREAKPCINT/CCNTINUE RCUTINES - CCNTINUED *****
            / ***** SIXTH RCM PAGE
      72C0  *START+1200
            / G HANDLER
            /GC TO A SPECIFIEC LCCATICN
7200  1031  GCO,    TAD CCADF
7201  303C          DCA CACF
7202  4056          CALL
7203  640C          CRLF
7204  704C          CMA               /SET CCNTINUE COUNT TC ZERC
7205  3044          CCA NCCNT
7206  1024          TAD WCRC          /SET ACCR TC CCNTINUE PROGRAM
7207  3025          CCA JACR1         / EXECUTICN FRCM
7210  1030          TAD CACF          /SET FIELC TC CONTINUE PRCGRAM
7211  3036          DCA IFSAVE        / EXECUTICN FRCM
7212  1365          TAD FIFSTA        /SET RAM PRCGRAM EXECUTION TO
7213  3043          CCA GCACR         / BEGIN AT MFSET
7214  5767          JMF I FCCNT5      /GC TC EXECUTE RAM PRCGRAM
            / B HANDLER
            /SET BREAKPCINT AT A SPECIFIEC LOCATICN
7215  1031  BCO,    TAC CCACF
7216  303C          DCA CACF
7217  4056          CALL              /TYPE CR,LF
7220  640C          CRLF
7221  4056          CALL              /SEE IF VALUE SPECIFIED
7222  6423          NTYPEC
7223  5231          JMP BDC1          /VALTE SPECIFIED - GC CN
7224  1024          TAC WCRC          /SET BREAKPCINT ACCRESS
7225  3033          DCA TRAC
7226  1030          TAC CACF          /ANC FIELD
7227  3034          DCA TRACF
7230  5770          JMF I PREAC       /GET NEXT COMMAND
7231  1371  BDO1,   TAD CCTLCC        /NC VALUE SPECIFIEC -
7232  3033          DCA TRAC          / CLEAR BREAKPCINT BY SETTING
7233  10CC          TAD  F1A          / IT WITHIN CCT-F
7234  3034          CCA TRACF
7235  577C          JMF I PREAC       /GET NEW CCMMAND
            /BREAKPCINT RETURN
            /ENTRY TO ODT UPCN ENCCUNTERING BREAKPCINT
            /SAVE STUFF THEN GC TC CCMMANC SCANNER
7236  3047  BKPT,   DCA ACSAVE        /SAVE AC
7237  7CC4          RAL               /SAVE LINK
7240  305C          DCA LSAVE
7241  77C1          ACL               /SAVE MC
7242  3051          CCA MCSAVE
7243  6214          RCF               /SAVE FROGRAM CATA FIELD
7244  3052          CCA CFSAVE
7245  62C1          CDF F1A           /SET CCT-F F-F DATA FIELD
7246  1034          TAD TRADF         /SAVE PROGRAM INSTRUCTION FIELC
7247  3036          DCA IFSAVE
7250  1033          TAD TRAC          /ANC BREAKPCINT LOCATION
7251  3035          DCA CCNT          /FCR CCNTINUE
```

```
7252  1033          TAD TRAC          /RESTCRE BREAKPOINT LCCATION
7253  3027          CCA CAC           / CCNTENTS
7254  1034          TAD TRACF
7255  3030              CCA CACF
7256  1045          TAD KEEP
7257  4056          CALL
7260  6442          SETCAC
7261  1030          TAD CACF          /INITIALIZE CCADF
7262  3031          DCA CCACF
7263  2044          ISZ NCCNT         /CCNTINUE CCUNT OVER?
7264  5766          JMP I FCCNTO      /NC - CONTINUE
7265  4056          CALL              /TYPE CUT BKPT FIELC
7266  7347          PFIELC
7267  1033          TAD TRAC          /ANC LCCATICN
7270  4056          CALL
7271  61C7          PNUM
7272  1361          TAC AP250         /TYPE "("
7273  4056          CALL
7274  6102          TYPE
7275  1047          TAD ACSAVE        /TYPE CUT AC
7276  4056          CALL
7277  61C7          PNUM
7300  4056          CALL              /TYPE CR, LF
7301  640C          CRLF
7302  577C          JMP I PREAC       /GET NEXT CCMMAND
            / ***** WCRD SEARCH RCUTINE ENE *****
            / W HANDLER
            /SEARCH BETWEEN LIMLC ANC LIMHI FCR WCRDS THAT
            /MATCH SPECIFIEC NUMBER IN MASKED BITS
7303  1031  WCO,    TAD CCACF
7304  303C          DCA CACF
7305  4056          CALL              /TYPE CR,LF
7306  64CC          CRLF
7307  1024          TAC WCRC          /MASK SPECIFIEC NUMBER
7310  0053          AND MASK
7311  3024          CCA WCRC
7312  1054          TAD LIMLC         /INITIALIZE CAC AS MEMCRY
7313  3027          DCA CAC           / POINTER FOR SEARCH
7314  4056  WDO1,   CALL              /GET MEMORY WCRD
7315  6431          GETCAC
7316  0053          AND MASK          /MASK IT
7317  7041          CIA               /CCMPARE TC WCRD
7320  1024          TAD WCRC
7321  764C          SZA CLA           /ECUAL?
7322  5337          JMP WCC2          /NC CNC - GET NEXT
7323  1027          TAC CAC           /YES - PRINT LOC
7324  4056          CALL
7325  61C7          PNUM
7326  1363          TAC AP257         /PRINT "/"
7327  4056          CALL
7330  6102          TYPE
7331  4056          CALL              /PRINT CCNTENTS
7332  6431          GETCAD
7333  4056          CALL
7334  61C7          PNUM
7335  4056          CALL              /TYPE CR, LF
7336  64CC          CRLF
7337  1027  WDO2,   TAD CAC           /GET LCC
7340  2027          ISZ CAC           /PCINT AT NEXT LCC
7341  70CC          NOP               /IN CASE ISZ SKIPS
7342  7041          CIA               /CCMPARE LCC WITH LIMHI
```

```
1090    7657  4056   RUN,         CALL
1091    7660  6463                LISN
1092    7661  3053                DCA HOLD
1093    7662  1053                TAD HOLD
1094    7663  4056                CALL
1095    7664  6102                TYPE
1096    7665  1053                TAD HOLD
1097    7666  1356                TAD KRUB
1098    7667  7700                SMA CLA
1099    7670  5216                JMP BEG+1
1100    7671  5257                JMP RUN
1101
1102    7672  2072   MORE,        ISZ DATA2
1103    7673  5277                JMP DL2
1104
1105    7674  1053                TAD HOLD
1106    7675  3071                DCA THIRD
1107    7676  5215                JMP BEG
1108
1109    7677  1067   DL2,         TAD FIRST
1110    7700  7002                BSW
1111    7701  1070                TAD SEC
1112    7702  3454                DCA I SAVPC
1113    7703  7000                NOP
1114    7704  2054                ISZ SAVPC
1115    7705  7300                CLA CLL
1116
1117    7706  7300   BACK,        CLA CLL
1118    7707  1071                TAD THIRD
1119    7710  3067                DCA FIRST
1120    7711  1053                TAD HOLD
1121    7712  3070                DCA SEC
1122    7713  5213                JMP BEGG
1123
1124    7714  1067   PCL,         TAD FIRST
1125    7715  0351                AND PMSK
1126    7716  7002                BSW
1127    7717  3054                DCA SAVPC
1128    7720  3067                DCA FIRST
1129    7721  1053                TAD HOLD
1130    7722  3071                DCA THIRD
1131    7723  7040                CMA
1132    7724  5215                JMP BEG
1133
1134    7725  3066   LTC,         DCA LT
1135    7726  1020                TAD LAST
1136    7727  1352                TAD KCH8
1137    7730  7650                SNA CLA
1138    7731  5213                JMP BEGG
1139    7732  1067                TAD FIRST
1140    7733  0351                AND PMSK
1141    7734  7002                BSW
1142    7735  1070                TAD SEC
1143    7736  7041                CIA
1144    7737  1046                TAD CHKSUM
1145    7740  7041                CIA
1146    7741  3046                DCA CHKSUM
1147    7742  1067                TAD FIRST
1148    7743  0351                AND PMSK
1149    7744  1046                TAD CHKSUM
1150    7745  1070                TAD SEC


1151    7746  3047                DCA ACSAVE
1152    7747  5750                JMP I .+1
1153    7750  6727                ADD
1154
1155
1156    7751  0077   PMSK,        0077
1157    7752  7600   KCH8,        7600
1158    7753  0100   KLONG,       0100
1159    7754  0200   K200,        0200
1160    7755  7500   KFD,         7500
1161    7756  7401   KRUB,        7401
1162    7757  0102   K102,        0102
1163
1164
1165          7772   *START+1772
1166
1167    7772  7301                CLA CLL IAC        / SET THE AC TO 0001
1168    7773  6400                6400               / CLEAR THE INTERCEPT JR.
1169                                                 / DISPLAY
1170    7774  6402                6402               / TURN OF THE CP TIMER
1171    7775  5776                JMP I .+1          / GO TO THE START OF THE
1172    7776  6000                INIT               / PROGRAM
1173
1174    7777  5372                JMP .-5            / ENTRY POINT FOR CP REQUEST
1175
1176
1177
1178

END OF PASS 2

   0 ERRORS DETECTED
SYMBOL TABLE

ACSAVE 0047   ACTEMP 0063   ADDR1  6737   ADDR2  6740   ADDR4  6741   ADDR   0040   ADD    6727   AP250  7361
AP256  7362   AP257  7363   AP260  7364   BACK   7706   BACC   6715   BDC1   7231   BDC    7215   BEGG   7613
BEG    7615   BIN    7600   BKPT   7236   BLIST  6256   BM100  7553   BP177  7555   BP200  7556   BP300  7557
BP77   7554   CADF   0030   CAD    0027   CALL1  0056   CALL   4056   CDFC   7141   CDC    7000   CFLAG1 6167
CFLAG3 6177   CHKSUM 0046   CLOSE  6411   COMM   6600   CONTC  7007   CONT1  7033   CONT2A 7073   CONT2  7047
CONT3A 7114   CONT3  7076   CONT4  7121   CONT5  7400   CONT   0035   CRDC   6616   CRLF   6400   DATA2  0072
DFSAVE 0052   DFSET1 0066   DFSET  0073   DF1    0041   DL2    7677   DOTDC  6654   EDD    7464   ERROR  6337
FIRST  0067   FM10   7142   FP177  7143   FP2000 7146   FP200  7144   FP4000 7147   FP400  7145   FP7000 7150
FP7600 7151   FP7770 7152   F1A    0000   F1     0000   GOC    7200   GETCAD 6431   GETCD1 6437   GDADR  0043
HOLD   0053   IADDR  7137   IFSAVE 0036   IFSET  0072   INIT1  6040   INIT   6000   INS    0037   JADR1  0025
JADR   0042   KCH8   7752   KCRA   6056   KCRB   6057   KEEP   0045   KFD    7755   KLING  7647   KLONG  7753
KRUB   7756   KTTY   6060   KC377  6336   K102   7757   K200   7754   LAST   0020   LFDC1  6645   LFDC   6625
LIMHI  0055   LIMLO  0054   LISN   6463   LSAVE  0050   LTABL  6257   LTC    7725   LT     0066   MASK   0053
MDO    6730   MORE   7672   MQSAVE 0051   M4     6134   NCCNT  0044   NEWINS 0067   NTYPED 6423   OCADF  0031
ODTLOC 7371   PBCDF  7446   PBCIF  7445   PBREAD 7560   PCL2   7653   PCL    7714   PCNT5  7140   PCONTC 7366
PCONT5 7367   PCREAD 6044   PC2    0073   PDFST1 7136   PDC1   7511   PDC    7502   PFIELD 7347   PGTCD1 6461
PIFSET 7135   PIFSTA 7365   PMSK   7751   PNUM2  6114   PNUM   6107   PNWINS 7134   POPJ   6072   PREAD  7370
PSTCD1 6462   PUNCH1 7543   PUNCH  7526   PUSHJ  6061   PZPAT  7450   P240   6137   P260   6140   P7     6135
QDC    7454   RAM1   6051   RAM2   6052   RAM3   6053   READ1  6204   READ2  6235   READ3  6240   READ   6200
RETRN1 0061   RETURN 5461   RUART  6160   RUN    7657   SAVPC  0054   SCHAR  0025   SEC    0070   SEMIDO 7451
SETCAD 6442   SETCD1 6451   SFLAG1 6166   SFLAG3 6176   SHUT   0032   SKIP3  6172   SKIP4  6173   SKPDR  6162
SKPTBR 6163   SLOO1  6610   SLOO   6601   SM270  6332   SM5    6333   SPNTER 0026   SP10   6334   SP177  6742
SP200  6743   SP215  6744   SP257  6745   SP260  6746   SP277  6335   SP400  6747   SP6201 6750   SP7600 6751
STACK1 6050   STACK1 0074   STACK  0062   START  6000   SUB1   6045   SUB2   6046   SUB3   6047   TABLE1 6260
TABLE2 6303   TDC1   7473   TDC    7471   TEMP2  0021   TEMP   0020   THIRD  0071   TM5    6455   TOTE   0023
TP212  6456   TP215  6457   TP6201 6460   TRADF  0034   TRAC   0033   TRAP1  7443   TRAP2  7444   TRAP3  6054
TRAP4  6055   TRAP   7447   TTYM   6477   TYPE   6102   UACC1  6712   UACC   6670   WCRA   6165   WCRB   6175


WDO1   7314   WDO2   7337   WDO    7303   WORD   0024   WTTY   6171   WUART  6161   WVR    6174   ZPAT1  0064
ZPAT   0065
```

APPENDIX C

INTERSIL 6960-SAMPLR BOARD

NOTE: 1. Square pads are used for pin 1 of all integrated circuits, the emitter of all transistors, the cathode of all diodes, and the common terminal of all switches.

2. Push button switches are assumed to have common as one end terminal, the center terminal normally open and the other end terminal normally closed (C & K type 8121). Toggle switches are assumed to have common in the center (C & K type 7101).

JUMPER OPTIONS

J1	CO (6100) to VCC
J2	C2 (6100) to VCC
J3	INTREQ (6100) to VCC
J4	CPREQ (6100) to VCC
J5	DMAREQ (6100) to VCC
J6	WAIT (6100) to VCC
J7	**(Deleted)**
J8	SENSE4 (6101) to GND
J9	PRIN (6101) to GND
J10	INTGNT (6101) to GND
J11	SEL4 (6101) to GND
J12	SEL4 (6101) to VCC
J13	SEL7 (6101) to VCC
J14	SEL7 (6101) to GND
J15	Deleted
J16	Deleted
J17	OSC IN (6100) to pin 3 IC35 (74C00)
J18	OSC OUT (6100) to pin 11 IC35 (74C00)
J19	TRC (6402) to pin 2 (7213)
J20	RRC (6402) to pin 2 (7213)
J21	RESET (6100) to pin 1 IC26 (74C74)
J22	XTB (6100) to S16 common
J23	WAIT (6100) to pin 6 IC26 (74C74)
J24	IFETCH (6100) to pin 1 IC16 (74C00)
J25	TRO (6402/03) to RS-232-C XMT level shifter
J26	TRO (6402/03) to 20 mA loop XMT level shifter
J27	GND to 20 mA loop XMT level shifter
J28	RRI (6402/03) to RS-232-C RCVE level shifter
J29	RRI (6402/03) to 20 mA loop RCVE level shifter
J30	FLAG1 (6101) to 20 mA RDR RUN loop driver
J31	GND to -12V point of 20 mA loop interface
J32	-12V terminal to -12V point of 20 mA loop interface
J33	Pin 1 of EIA connector to GND

JUMPER STRAPPING

OPTION 1 - BASIC MICROCOMPUTER (See Figure 1)

Insert the following jumpers: J1, J2, J3, J4, J5, J6, J8, J9, J10, J11, J13

Note: J11 and J12 should NEVER be in circuit at the same time.

Similarly, J13 and J14 should NEVER be in circuit at the same time.

If both jumpers were in, a direct short circuit between VCC and GND would result.

XTAL 11 and XTAL 12 should be used.

OPTION 2 - 20 mA CURRENT LOOP INTERFACE (See Figure 2)

Insert jumpers J26 and J29

OPTION 3 - PROGRAM CONTROLLED TELETYPE PAPER TAPE READER

Insert jumper J30.

OPTION 4 - EIA RS-232-C INTERFACE (See Figure 3)

Insert jumpers J25, J28, J27.

Note: Options 2 and 4 are mutually exclusive, so insert and delete jumpers as necessary.

OPTION 5 - IM6402 UART TIMING (See Figure 4)

Use XTAL14 (delete XTAL21), insert jumpers J19 and J20

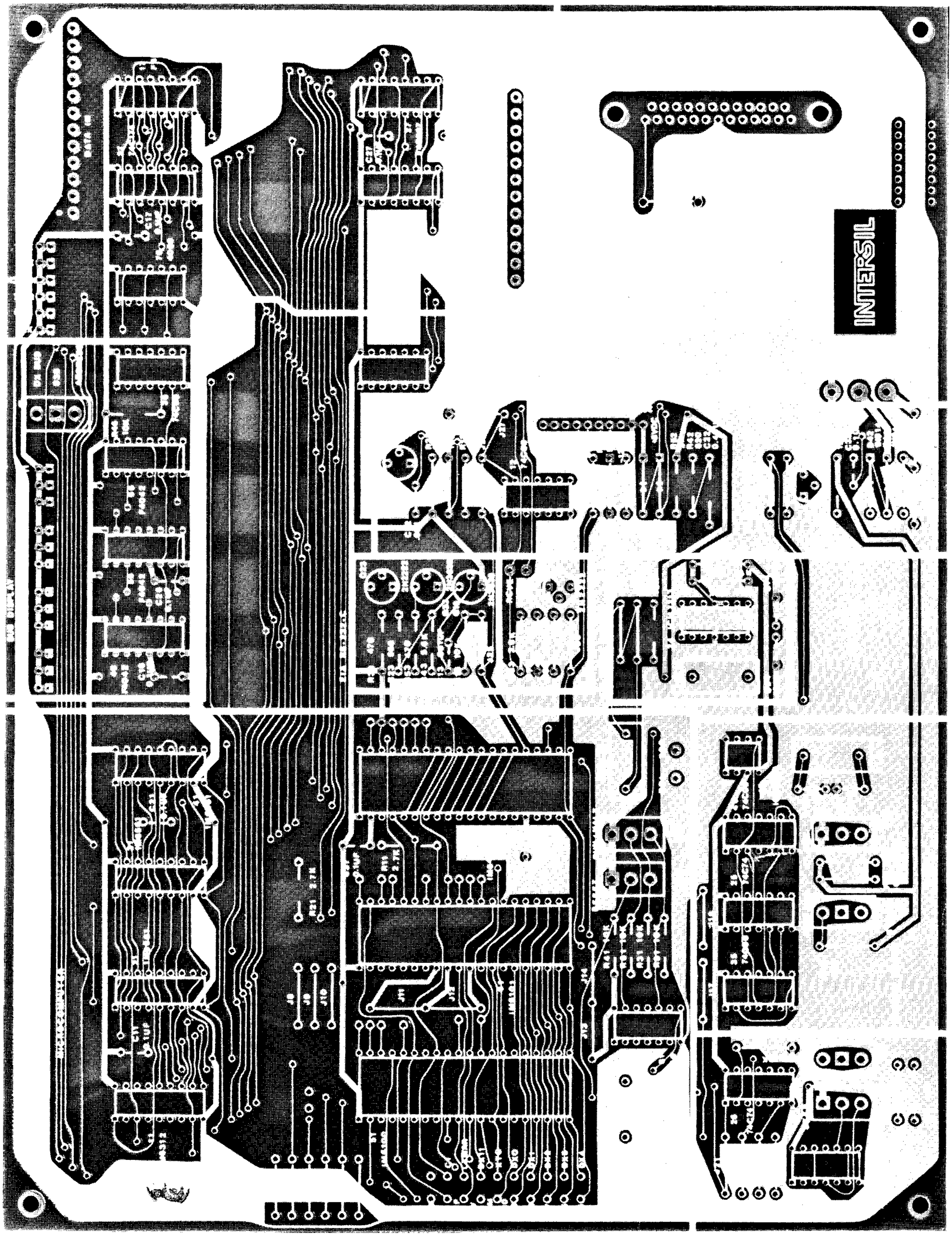
OPTION 6- SINGLE CLOCK OPERATION (See Figure 5)

Use XTAL15 (delete XTAL11), insert jumpers J17 and J18

OPTION 7 - SINGLE STEPPING OPERATION (See Figure 6)

Delete jumper J6 and add jumpers J21, J22, J23 and J24

APPENDIX C
6960 SAMPLR BOARD



APPENDIX D
TELETYPE MODIFICATIONS FOR THE INTERCEPT SYSTEM

The Intersil INTERCEPT systems have been designed to be used in conjunction with a Model ASR-33 Teletype. Before attempting to use your system inspect your Teletype for the following modifications and additions. If they have not yet been performed, you must complete them before using INTERCEPT.

To check for, or make, these modifications remove the cover of the Teletype. Loosen the three thumb screws in the back and remove the Platen that holds the roll of paper, the Mode Switch knob and the Face Plate. Remove the small screw on the Reader cover and the four screws under the Face Plate. You should now be able to lift the cover off. Use Figure D-1 to locate the various parts located below.

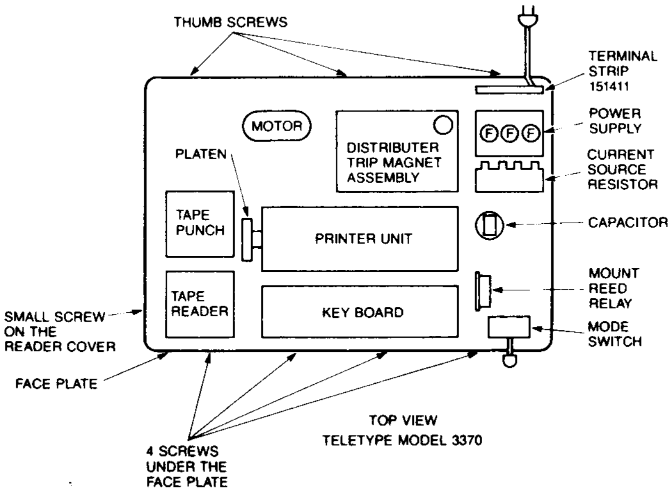


FIGURE D-1

The modifications are:

CURRENT LOOPS CHANGED FROM 60 TO 20 MILLIAMPS

The Current Source Resistor must be changed from 750 ohms to 1450 ohms. This is accomplished by moving the BLUE wire from Terminal #3 to Terminal #4 of the large power resistor shown in Figure D-2. The receiver current level is changed by moving the PURPLE wire of Terminal #8 on Terminal Strip 151411 to Terminal #9 on the same strip. Terminal Strip 151411 is shown in Figure D-3 with Terminal #1 at the far left.

TELETYPE WIRED FOR FULL DUPLEX OPERATION

The half duplex wiring must be changed by moving the BROWN/YELLOW wire from Terminal #3 to Terminal #5 and the WHITE/BLUE wire from Terminal #4 to Terminal #5 on Terminal Strip 151411.

THE READER RUN RELAY ADDED

The Reader circuit should have a 12 volt relay inserted to allow program control of the Reader. This Relay is shown along with the mode switch in Figure D-4. Mount the Relay with two 6-32 screws on the available bracket. A schematic diagram for the Relay and its connections is shown in Figure D-6. Locate the BROWN wire coming from the Distributor Trip Magnet which is connected to terminal J4—Pin 11 as shown in Figure D-5. Cut this BROWN wire and connect to the wire marked BROWN on the Relay circuit (note that this leaves J4—Pin 11 with no connection). Connect the wire marked LINE to terminal L1 and the wire marked LOCAL to terminal N of the mode switch as in Figure D-6. A preassembled Reader Relay Card is available from Intersil Inc., Model # 6908-RELAY.

LEVEL 8 OPTION WIRED TO 'ALWAYS MARK'

The level 8 option must be changed from parity to 'ALWAYS MARK'. This causes the keyboard to always output a 1 for the 8th bit, and the Reader to read the 8th bit as it was written. Locate the Left Contact Block and the Right Contact Block as shown in Figure D-7. It may be necessary to remove a clear plastic shield to gain access to the Left Contact Block. On the Left Contact Block remove the RED/GREEN wire from the upper left contact, leave the RED/GREEN wire open and connect the GREEN wire to the upper left contact. On the Right Contact Block connect the GREEN wire to the upper left contact. For a detailed reference see Teletype keyboard schematic 9334WD.

CONNECT CPUTTY OUTPUTS TO THE TELETYPE

The TTY outputs of the CPUTTY board are connected to Terminal Strip 151411 and the relay as shown in Figures D-6 and D-8.

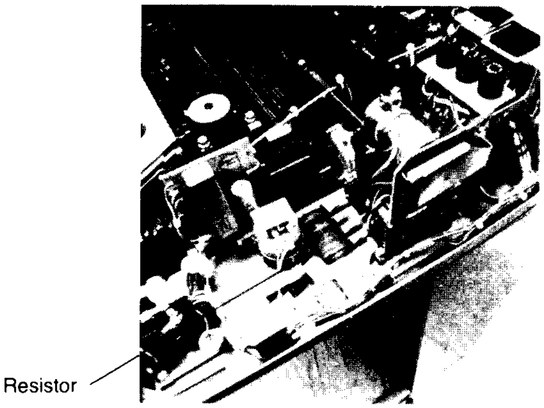


FIGURE D-2
Current Loop Resistor

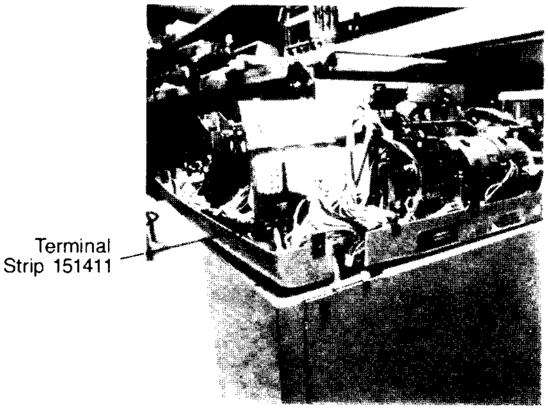


FIGURE D-3
Terminal Strip

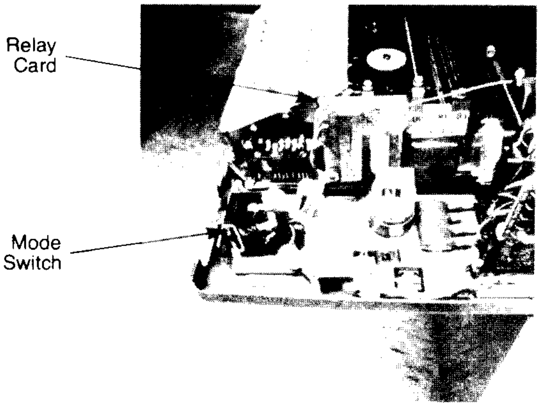


FIGURE D-4
Relay Card

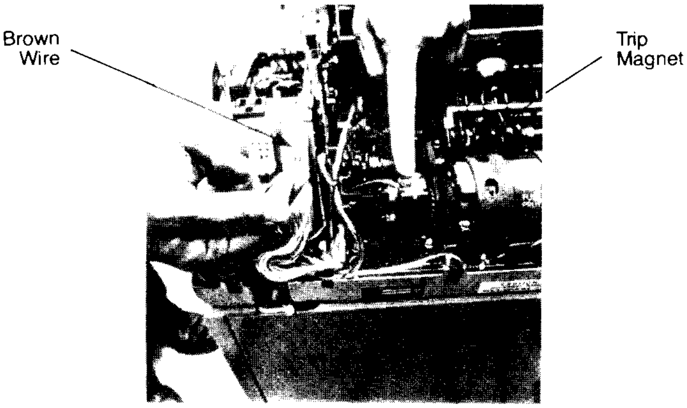


FIGURE D-5
Distributor Trip Magnet

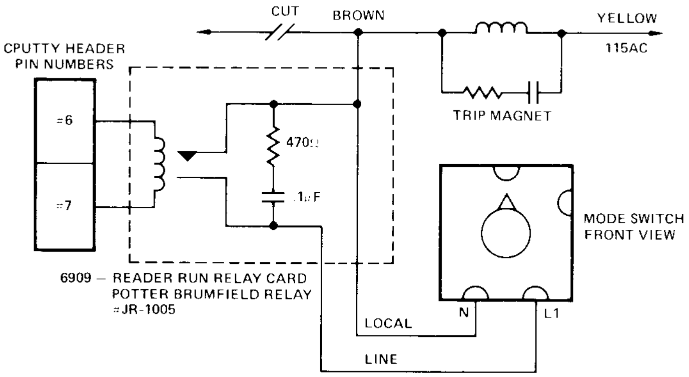


FIGURE D-6
Reader Relay Circuit

INTERSIL

10710 N. Tantau Ave., Cupertino, California 95014, (408) 996-5000, TWX 910-338-0228

Intersil cannot assume responsibility for use of any circuitry described other than circuitry entirely embodied in an Intersil product. No other circuit patent licenses are implied. Intersil reserves the right to change without notice at any time the circuitry and specifications.

6801 SAMPLER KIT MANUAL

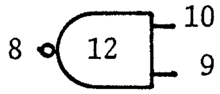
-ERRATA-

The following changes will be required in order to use the new ROM ODT : IM6312-001 marked 63S003 (in place of 63S000):

Figure 1 page 2: "SEL 4" (pin 9) should be shown to ground, not + 5V. "SEL 7" (pin 14) should be shown to + 5V, not ground.

Page 5: The first paragraph should read,"The PIE is wired to respond to the select code 00111 on SEL3 - SEL7 ...".

Page 6: The "pin-out" on the 74C00 should read as follows:



Page 33: OPTION 1, 'Note: Should read, "XTAL 11 and XTAL 21 should be used.".

Note: The description of the jumpers used, shown on page 33, is correct.