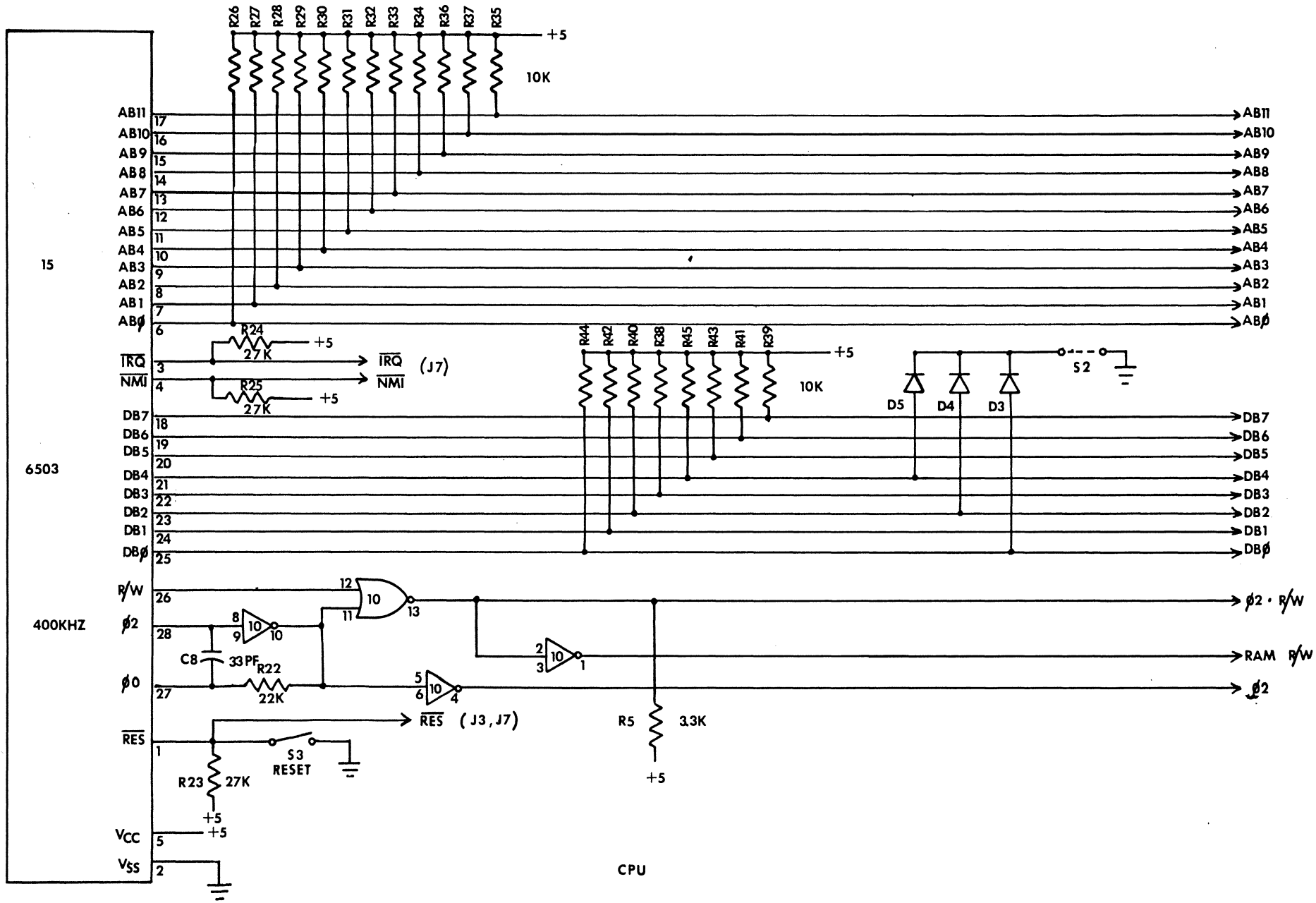
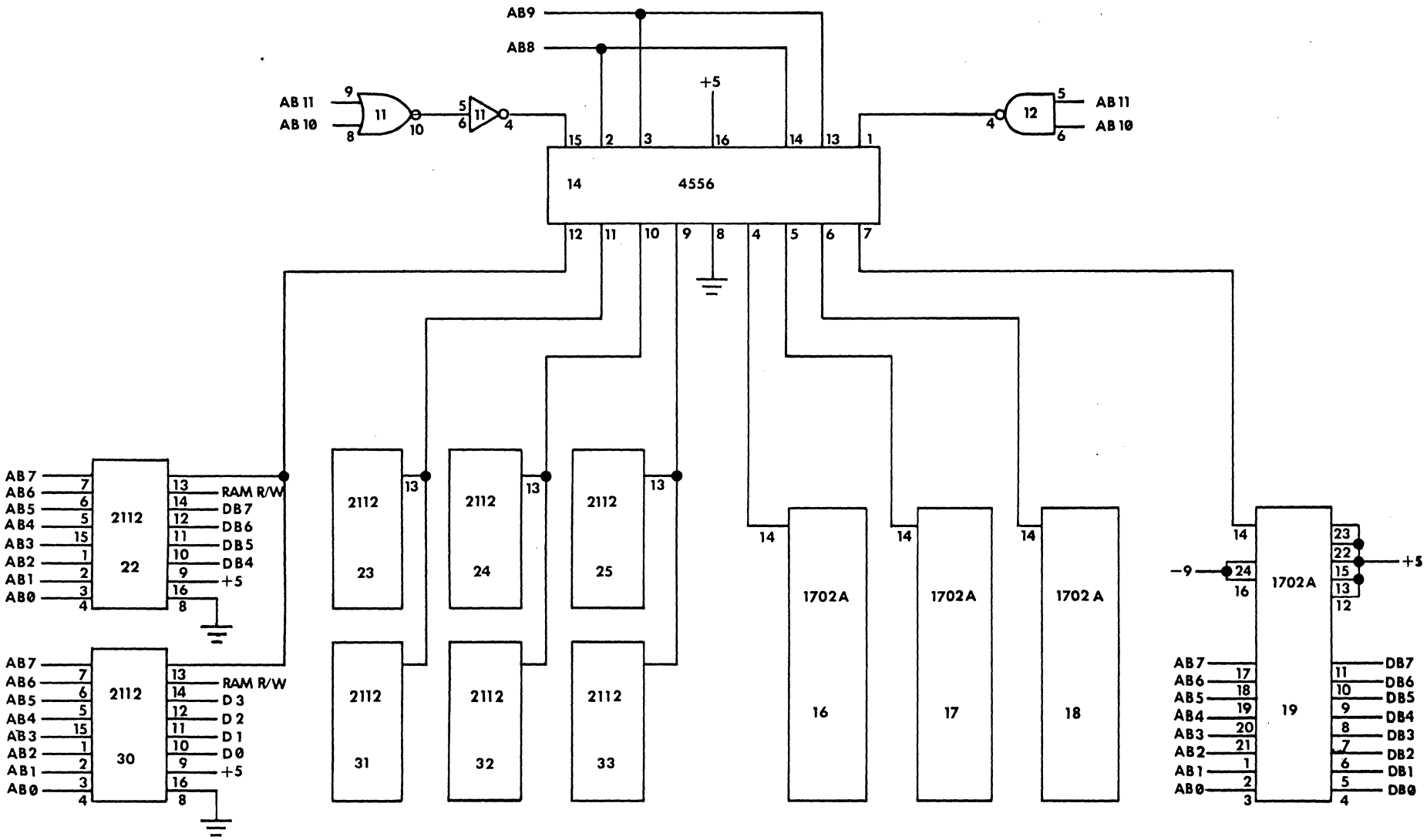

R26 R27 R28 R29 R30 R31 R32 R33 R34 R36 R37 R35
+5
10K
AB11 17
AB10 16
AB9 15
AB8 14
AB7 13
AB6 12
AB5 11
AB4 10
AB3 9
AB2 8
AB1 7
ABØ 6
15
R24 27K +5
IRQ 3
NMI 4
R25 27K +5
IRQ (J7)
NMI
R44 R42 R40 R38 R45 R43 R41 R39
+5
10K
D5 D4 D3
S2
DB7 18
DB6 19
DB5 20
DB4 21
DB3 22
DB2 23
DB1 24
DBØ 25
6503
R/W 26
400KHZ
Ø2 28
C8 33PF
R22 22K
Ø0 27
RES 1
R23 27K
S3 RESET
RES (J3, J7)
R5 3.3K
+5
Vcc 5
Vss 2
CPU
Ø2 · R/W
RAM R/W
Ø2

FIGURE 1



RAM – PROM
MEMORY

FIGURE 2

I/O DECODING

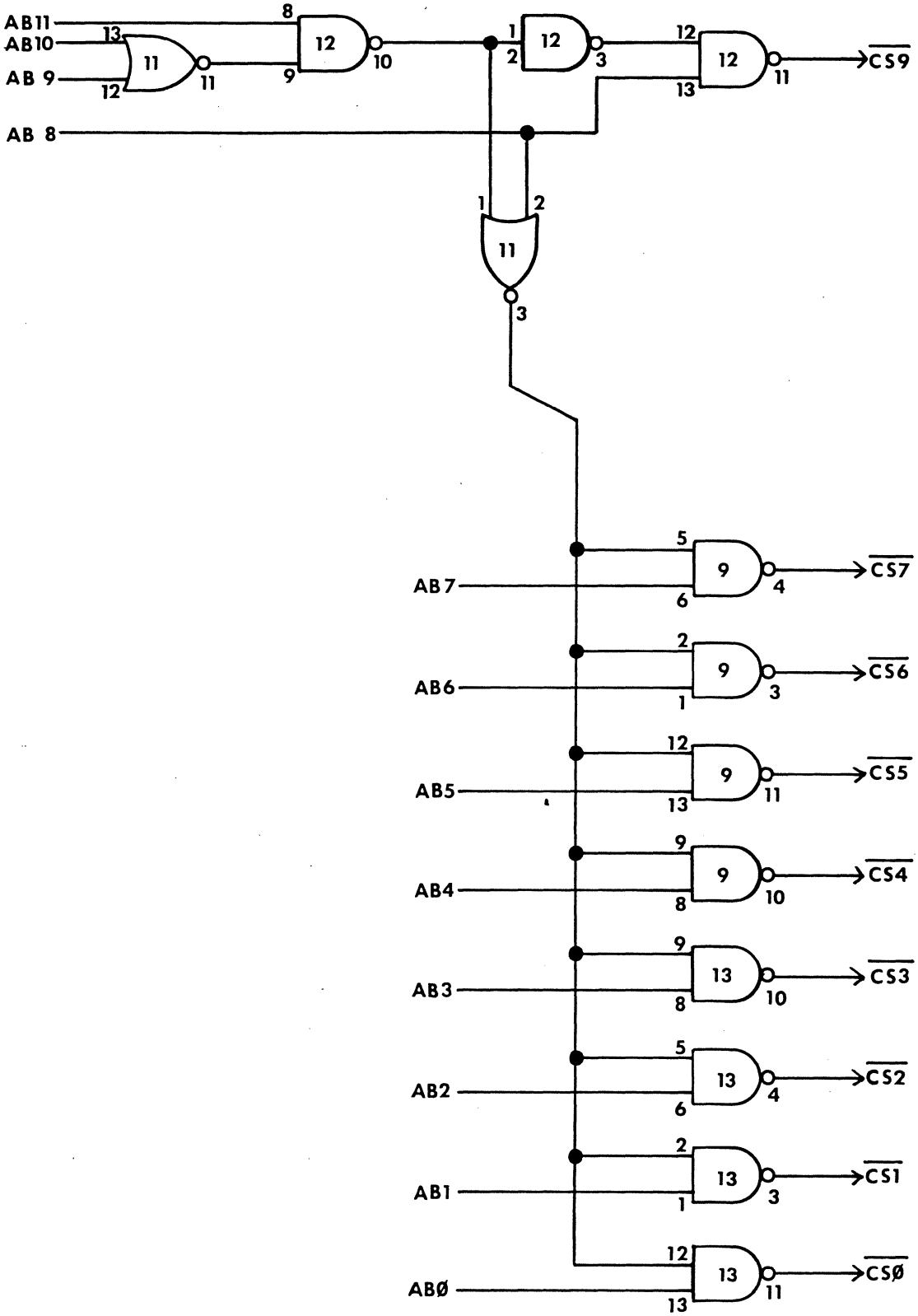


FIGURE 3

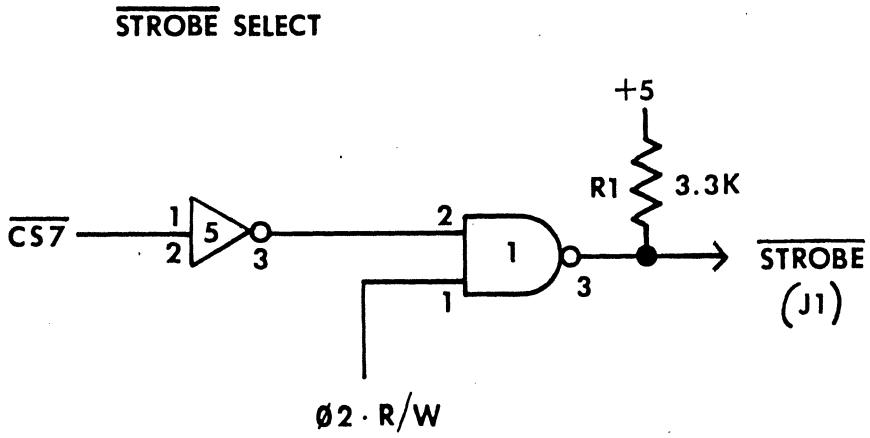


FIGURE 4

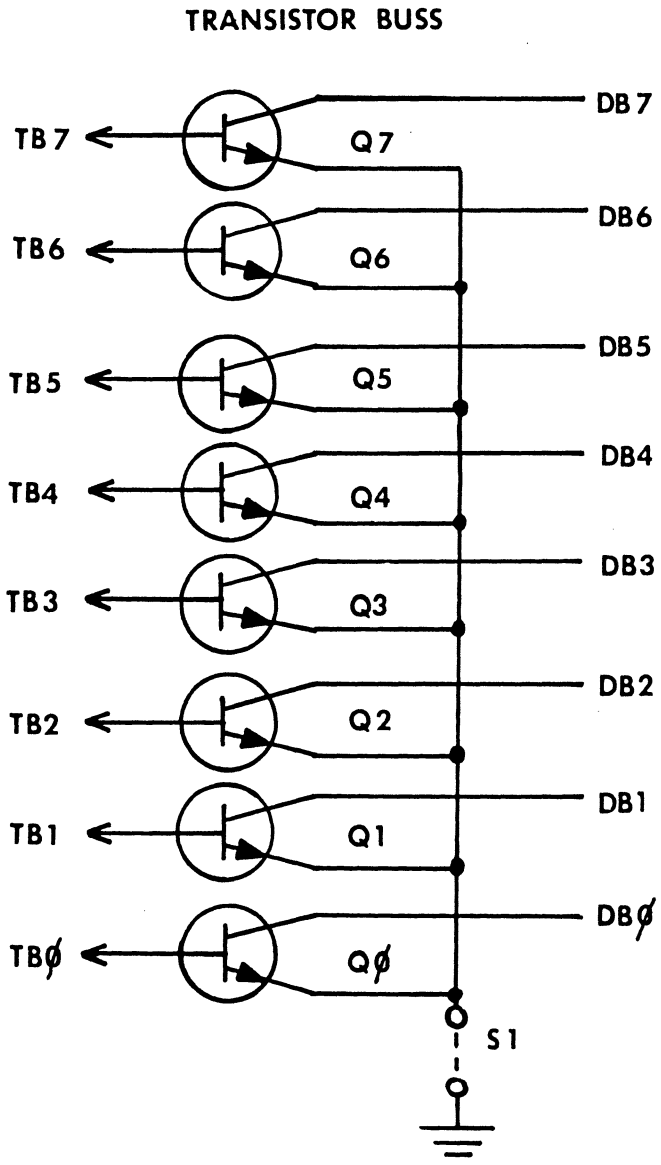
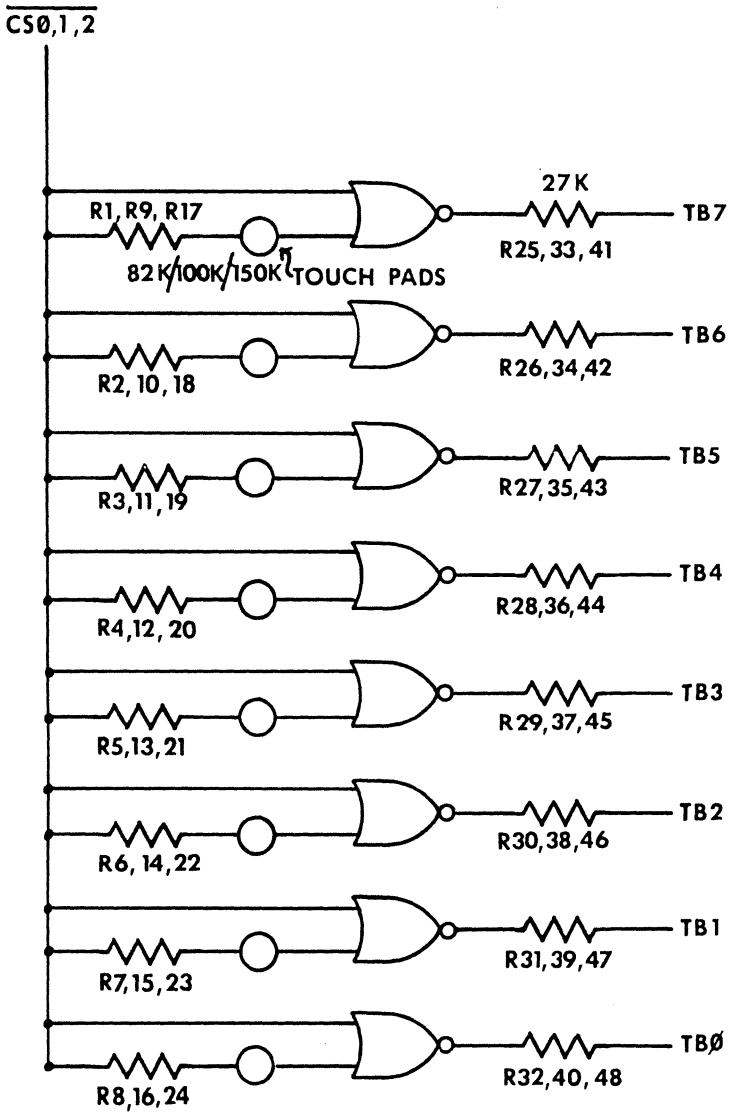
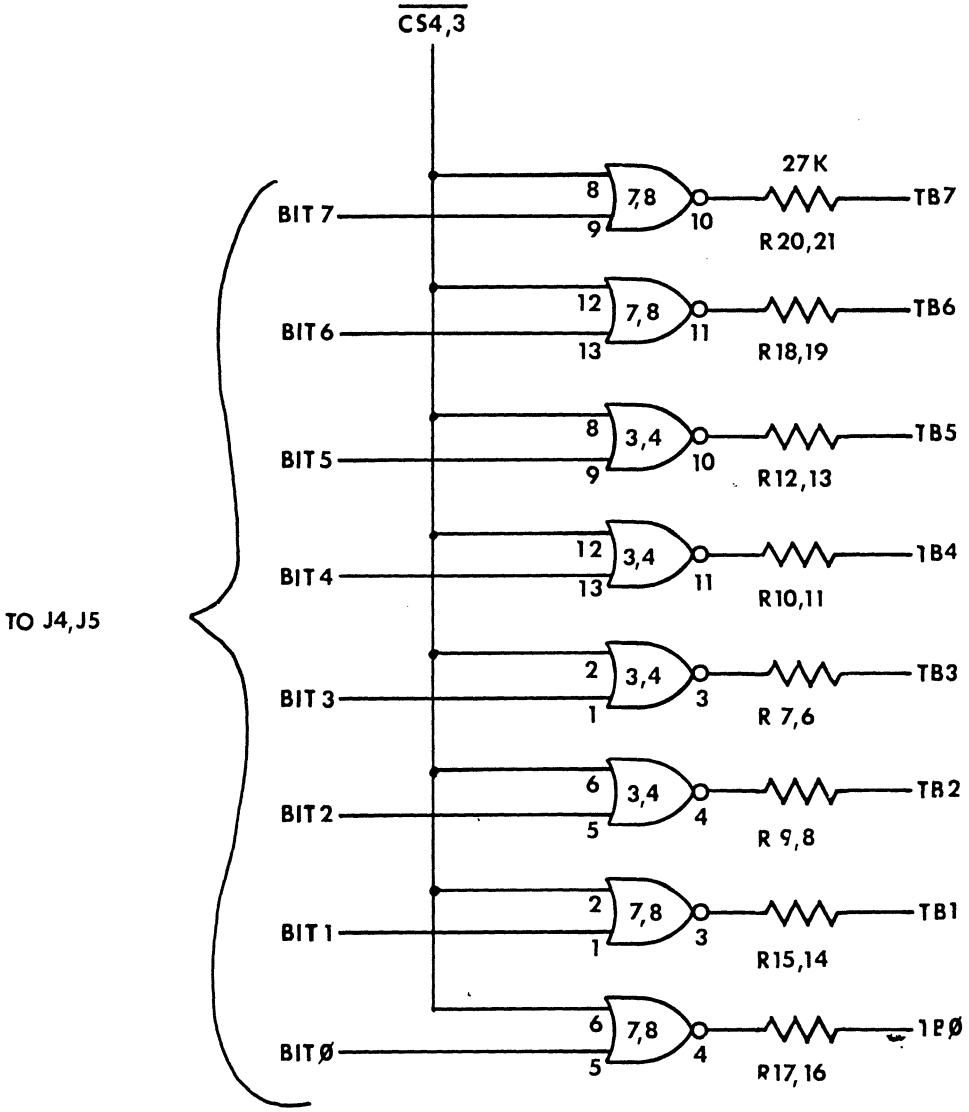


FIGURE 5



KEYBOARD

FIGURE 6



INPUT PORTS

3 AND 7, 4 AND 8
PORT 1 PORT 2

FIGURE 7

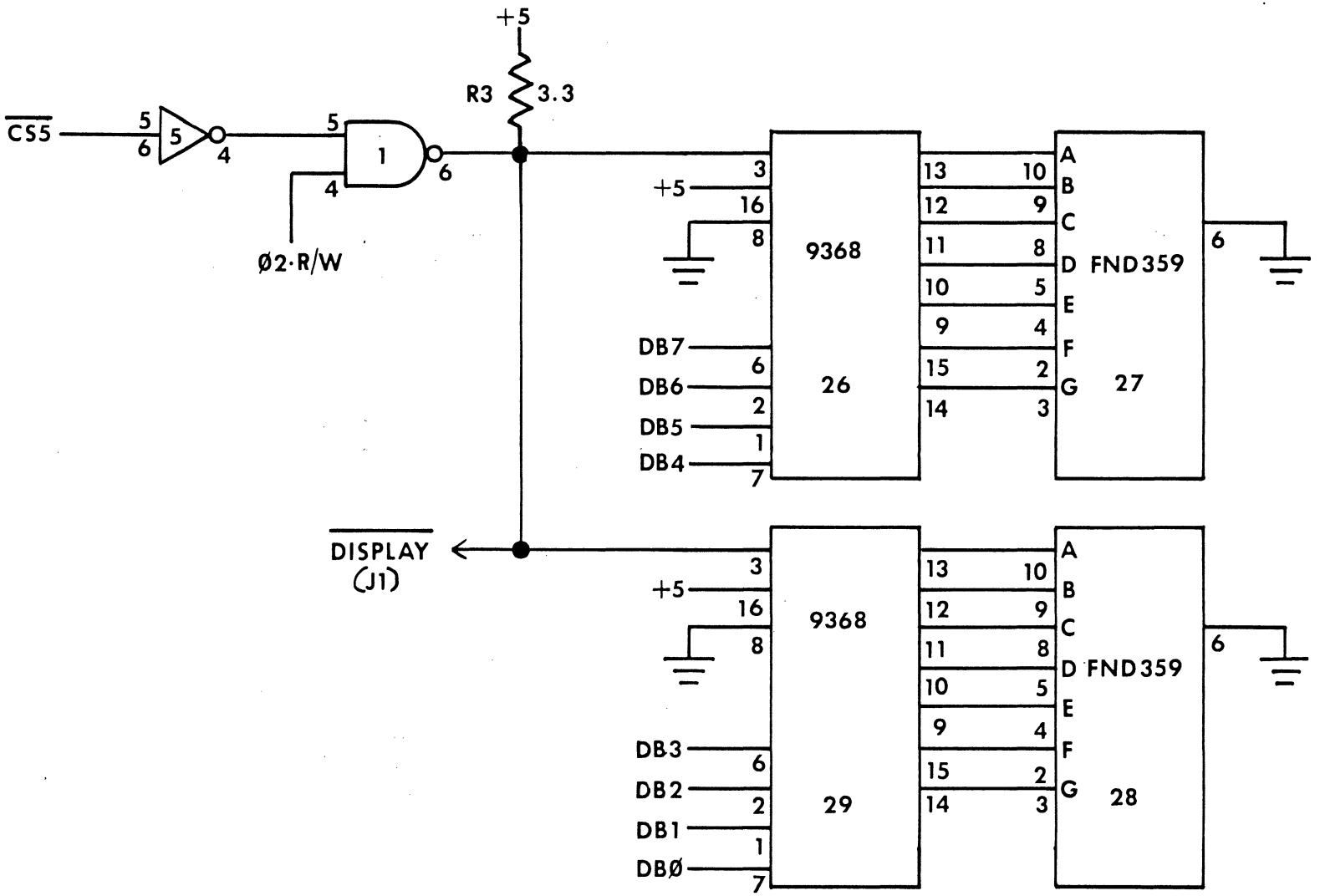

+5
R3
3.3
CS5
Ø2·R/W
DISPLAY
(J1)
9368
26
9368
29
FND359
27
FND359
28
DB7
DB6
DB5
DB4
DB3
DB2
DB1
DBØ
A
B
C
D
E
F
G

FIGURE 8

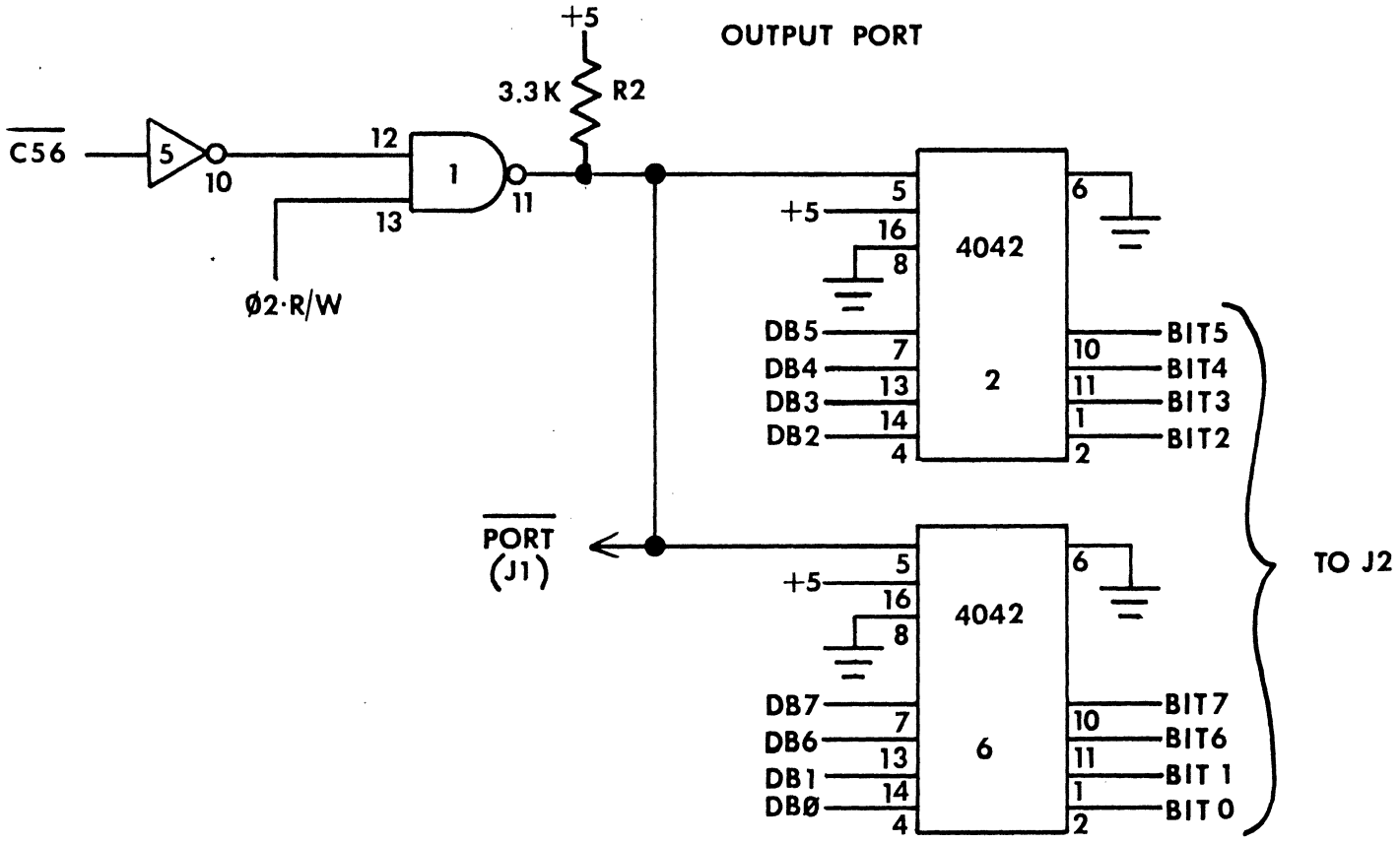

OUTPUT PORT
+5
3.3K
R2
C56
5
10
12
13
1
11
Ø2·R/W
PORT
(J1)
+5
5
16
8
4042
2
6
DB5
DB4
DB3
DB2
7
13
14
4
10
11
1
2
BIT5
BIT4
BIT3
BIT2
4042
6
DB7
DB6
DB1
DBØ
BIT7
BIT6
BIT 1
BIT 0
TO J2

FIGURE 9

Working Notes

Useful Zero Page Locations:

00ED	Monitor stack
00F0	Buffer, LSB (latest entry)
00F1	Buffer
00F2	Buffer
00F3	Buffer
00F4	Buffer
00F5	Buffer, MSB (oldest entry)
00F9	Accumulator
00FA	Y-Register
00FB	X-Register
00FC	Program counter low
00FD	Program counter high
00FE	Stack pointer (user)
00FF	Status register

Vectors:

NMI - 0003
RES - FF48
IRQ - 0000

Break Vector: Store starting at 0000; 4C, C0, FF

Memory Map:

0000-00FF	RAM (IC22, IC30)
0100-01FF	RAM (IC23, IC31)
0200-02FF	RAM (IC24, IC32)
0300-03FF	RAM (IC25, IC33)
0400-07FF	UNOCCUPIED - RAM on Video board
0800-08FF	I/O
0900-09FF	CASSETTE (IC20, IC21) S9
0A00-0BFF	UNOCCUPIED - I/O ports on video board
0B00 - 0BFF	PROM - Video routine
0C00-0CFF	PROM (IC16) - 1702-A Programmer - Operating software, uses shifting ports
0D00-0DFF	PROM (IC17) - Music
0E00-0EFF	PROM (IC18) CASSETTE OPTION
0F00-0FFF	PROM (IC19) MONITOR

I/O Breakdown

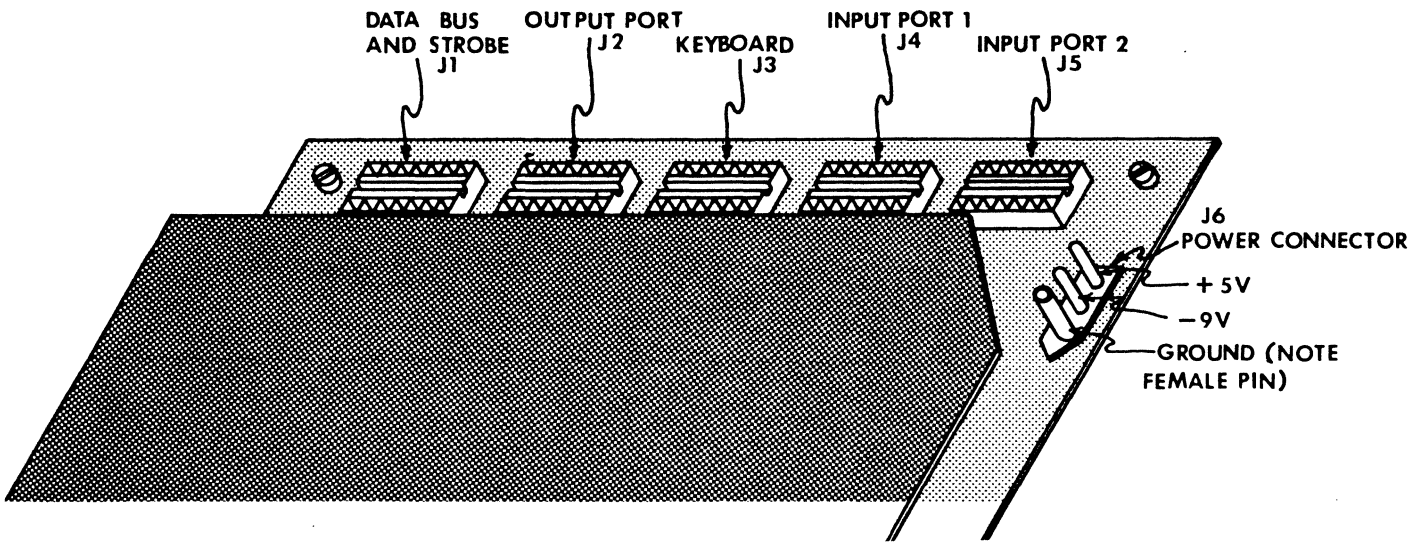
0801	KEYBOARD (IC1, IC2) KEYS 0-7
0802	KEYBOARD (IC3, IC4) KEYS 8-F
0804	KEYBOARD (IC5, IC6) CONTROL KEYS
0808	INPUT PORT 2 (IC4, IC8) J5
0810	INPUT PORT 1 (IC3, IC7) J4
0820	DISPLAY (IC26, IC27, IC28, IC29)
0840	OUTPUT PORT (IC2, IC6) J2
0880	STROBE (IC1) J1

XA0X - output
A1Y - "
A2Y - "

XA3X - input

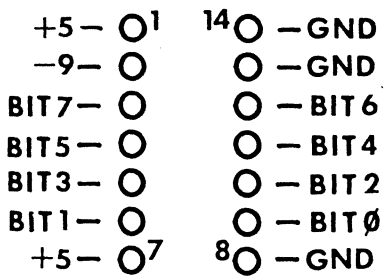
XA4X - interrupt mask port
XA5X - " reset "
XA6X - " status "
XA7X — Video port

System Analysis



OUTPUT PORT (J2)

OUTPUT PORT ADDRESS - x840



The output port is a means of getting data being processed within the computer out to peripheral devices.

The eight output lines (bit 0-bit 7) are all latched and each represents a CMOS output structure.

Included at the output port connector are the system power voltages, +5 volts and -9 volts and gnd.

PROGRAMMING CONSIDERATIONS

The port is memory-mapped, so that any instruction which would ordinarily be used to write data into memory can also be used to write data to the output port.

PROGRAMMING EXAMPLE

```
0020    LOOP   E8   inx; increment count
0021           8E   stx (abs); write result to output port
               40
               A8
0024           4C   jmp LOOP; go to do next
               20
               00
```

ANALYSIS

This short program causes the bits of the output port to count in binary. Bit 0 is the least significant, bit 7 the most significant.

When running, the program increments the X index register by 1 (INX) at location 0020, the STX instruction at location 0021 causes the incremented result in the X register to be "stored" in the output port which occupies memory location x840. The JMP instruction at location 0024 causes the program to loop back to the beginning.

NOTICE TWO THINGS:

1) the location of the output port is listed as x840 where x can be any hexadecimal digit. In this example x is A, but this is arbitrary. Using an oscilloscope you can check that the output lines are counting and that x can be given any value from 0-F without affecting the operation of the program.

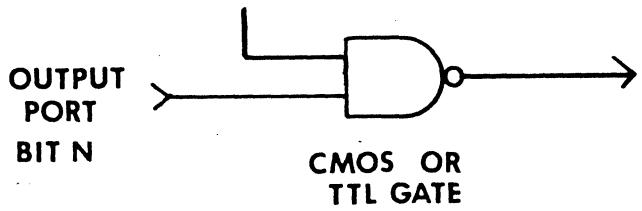
2) because of the pipe-lined architecture of the 650x family of processors, absolute addresses are given LEAST SIGNIFICANT BYTE FIRST. This will be confusing to first-time users of these processors but results in significantly greater processor through-put than would otherwise be possible. (See 6500 PROGRAMMING MANUAL.)

HARDWARE INTERFACING

The easiest situation is interfacing the output port to CMOS logic, which is simply

a matter of tying the output port pin to the input of the CMOS load. Like this:

A

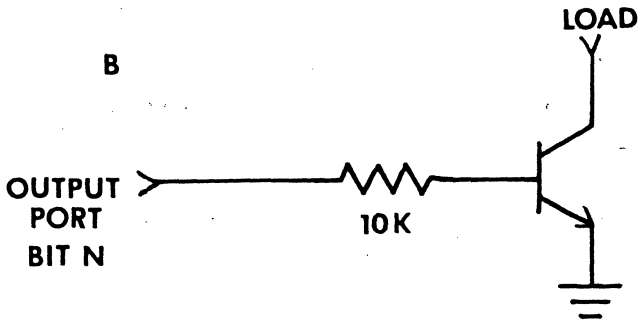


Because of the static nature of these outputs, practically any number of CMOS gates can be driven. (The limiting factor is the risetime of the output as the additional capacitors that the inputs of the gates represent are added.) If you like, and if the specifications of the power supply are not exceeded, power for the peripheral device can be picked up on J2 as are the signal leads.

TTL gates are just as easily driven from the output port, but unfortunately not in unlimited quantities. To be on the safe side, stick to one regular TTL load or two LS TTL loads max.

When interfacing to a discrete transistor, a current-limiting resistor should be put in the line like this:

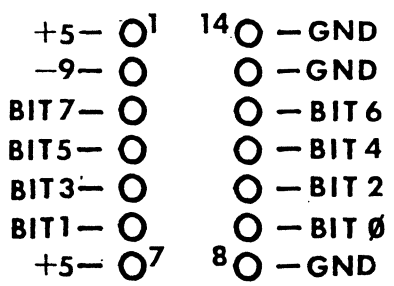
B



If needed, the activating signal that strobes new data into the output port latches, $\overline{\text{(OUTPORT)}}$ is present on pin #10 of the DATA BUSS and-STROBE connector (J1)

INPUT PORTS

PORT #1 (J4) - ADDRESS x810
PORT #2 (J5) - ADDRESS x808



The input ports are means of getting data from the outside world into the computer.

Each input line represents a single CMOS input structure.

Included at the input port connectors are the system supply voltages +5 volts -9 volts and gnd.

PROGRAMMING CONSIDERATIONS

Like the output port, these input ports are memory mapped and any instruction which reads data from a memory location may be used to read the port into the processor.

PROGRAMMING EXAMPLE

```
0020  LOOP  AD  LDA (abs) IN#1     ;read input port
            10
            A8
0023        8D  STA (abs) DSPLY    ;put result in display
            20
            A8
0026        4C  JMP  LOOP          ;do again
            20
            00
```

ANALYSIS

The instruction at location 0020 causes data which is currently being presented to the input port to be read to the processor's accumulator. The next instruction writes this same data to the display. Finally, the jump instruction at 0026 causes the program to loop and start again.

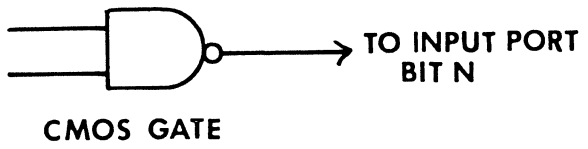
NOTICE ONE THING

Since the input port is a CMOS input, normal precautions should be taken to prevent static damage at these pins; also, if the above program is run without some device connected to the port, some means must be provided to hold the input pins of the port at either ground or supply. Otherwise, normal environmental electromagnetic fields will cause the state of the input lines to be indeterminate. 10K ohm resistors from the pins to either ground or supply (see also HARDWARE INTERFACING) will suffice.

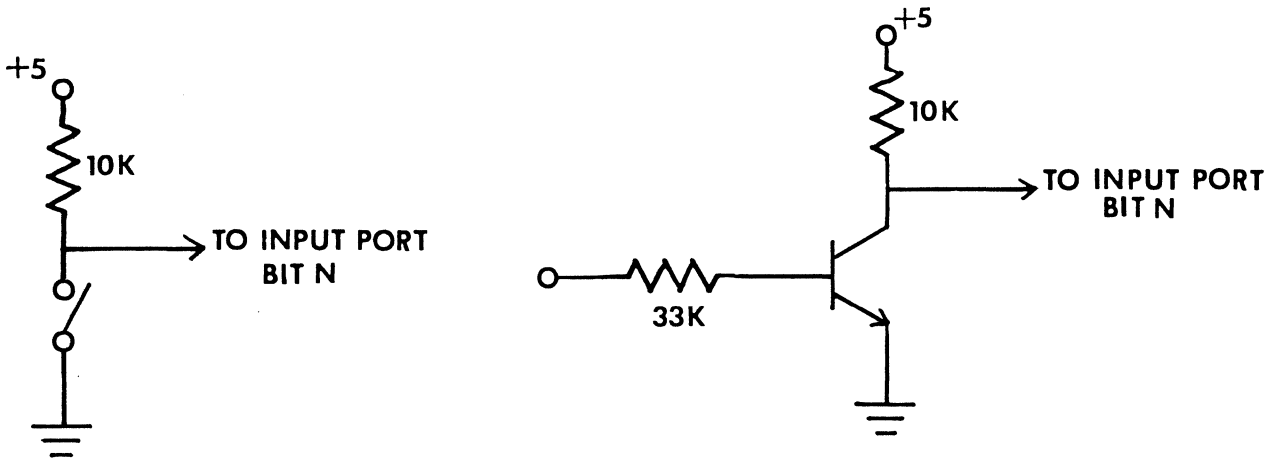
HARDWARE INTERFACING

Being a CMOS input, a variety of devices can supply data to the input ports. The output of another CMOS gate can be connected directly to the port:

A



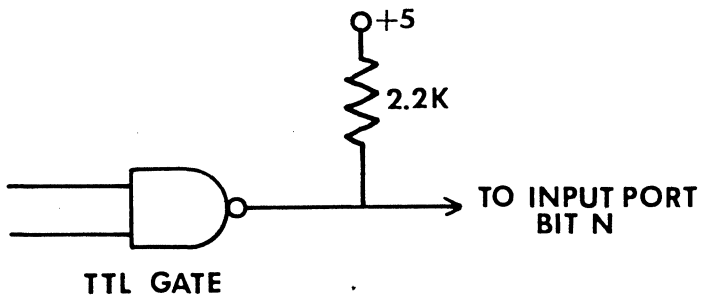
or switches or transistors may be used:



and note that if the transistor or switch above is "on", it represents a 0 input to that pin of the port.

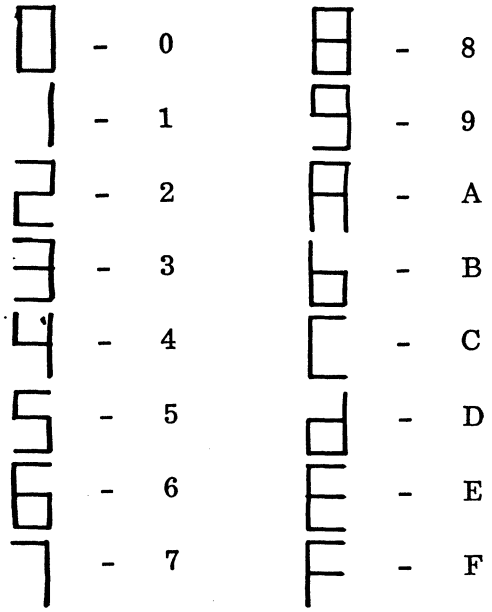
If the output of a TTL gate is being used to drive the input port, a pull-up resistor to supply must be provided:

C



DISPLAY

DISPLAY ADDRESS - x820



The displays consist of two seven-segment displays and associated 9368-type decoders/latches/drivers. The decoder portion of the 9368 takes care of converting a single 4-bit hexadecimal digit input to the appropriate code required to operate the seven-segment displays.

These devices will display all 16 symbols in the hexadecimal character set from 0-F. NOTE that the characters B and D are both displayed as lower case characters (b and d), and that the character 6 is distinguished from the character b by the horizontal "tail" at the top of the 6.

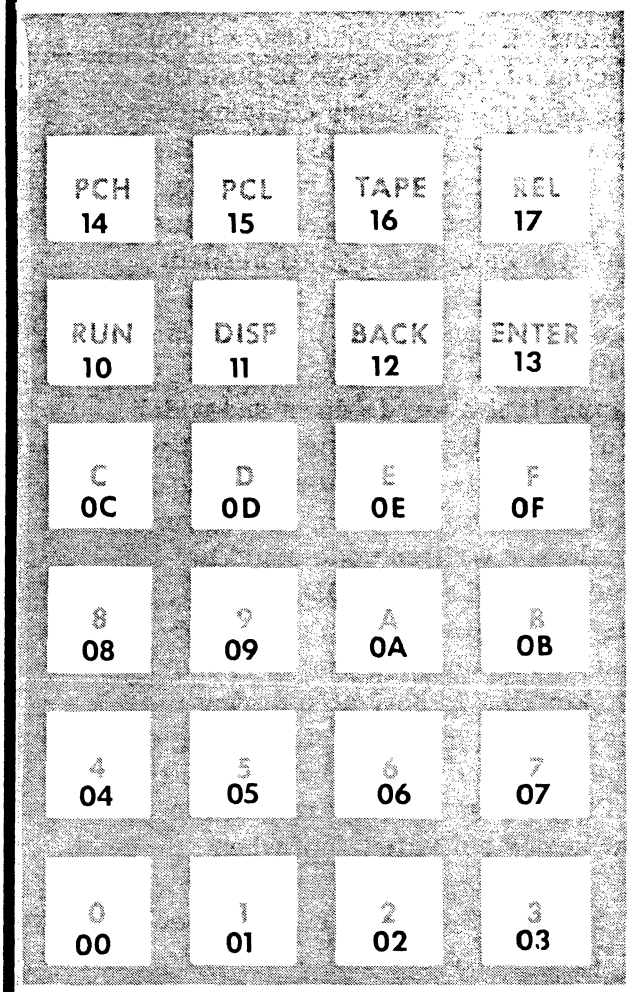
Like other peripheral ports, the displays are memory mapped and any instruction that writes to memory will operate them.

NOTE: it is normal for the 9368 driver ICs to operate at an elevated temperature.

TYPICAL SOFTWARE

(see KEYBOARD section of system analysis for typical programming examples using the displays.)

FIRST RANK (0-7) - address x801
SECOND " (8-15) - address x802
THIRD " (16-23) - address x803
DECODE SUBROUTINE address FF00



The keyboard is used by the monitor for control of user data and program entry as well as operation of the PIEBUG debugging tools, but may also be read by the user's programs employing a variety of techniques.

Because of the capacitive operating principle employed in the 8700 keyboard, this device should provide exceptionally long and trouble-free life.

(FOR EXPLANATION OF KEYBOARD WHEN USED WITH MONITOR, SEE PIEBUG MONITOR.)

USING THE KEYBOARD AS AN INPUT TO USER'S PROGRAMS

There are two ways that the 8700's keyboard may be used to input data to a user's program.

1) Individual "ranks" of keys may be read with any of the statements that read memory locations. For example:

```
0020 LABEL   AD   LDA (abs) A801      ;read first rank
             01
             A8
0023         FO   BEQ LABEL          ;if no key, loop
             FB
```

causes the status of the first 8 keys on the keyboard to be read into the accumulator of the computer (instruction at location 0020). If no keys are being touched when the read operation occurs, the accumulator will be loaded with $00. Under these circumstances, the Branch Not Equal at location 0023 will cause the processor to loop back to the top of the program and read the keyboard again. If a key is being touched when the read operation happens, the accumulator will be loaded with a number that represents the key. Each of the 8 bits in the word that is read represents a key, from $01 (in binary 00000001) for key #0 to $80 (in binary 10000000) for key #7.

While there are circumstances when the above procedure will suffice for inputting data, there will be times when it is most convenient to read not simply one rank of

keys, but rather the whole keyboard.

A new program to do this can of course be written, but under most conditions the effort would be redundant as this program is already a part of the PIEBUG Monitor and written as a subroutine so that it can be easily accessed from user's programs. This subroutine is named DECODE and it lives in the Monitor Prom at address FF00.

Before using this subroutine, there are a few things that you should know about it, like; when called, the routine returns with the number of the key down in both the accumulator and the Y index register, so if either of these registers contains data that will be needed after the keyboard scan, it should be either pushed to the stack or otherwise saved in memory. Similarly, though the X index register doesn't contain any key information when DECODE is exited, its contents are altered by this routine and as with the accumulator and Y register it should be saved (if needed) before entry to DECODE.

If no keys are down, the routine is exited with $18 in A and Y and this fact can of course be used to determine if a key is down or not.

A problem that is just as important as determining that a key is down and which key it is, is to determine whether the key that is down now is the same one that was down the last time through the program. (Otherwise, what is intended as a single keyboard stroke can be interpreted as multiple switch activations, one activation for each pass through the routine). Again, external user written code could be used to perform this task; but, again, it would be redundant as DECODE already indicates whether the key that is currently down is the first activation of that key- or if the key is simply <u>still</u> down. It indicates this by clearing (setting to 0) the Carry Flag in the processor status register; if the key that is activated during the current scan of the keyboard is different from the key that was activated during the last scan. If the same key that was down during the last scan of the keyboard is the same one as is down during this scan, the Carry Flag will not be cleared. Note also that the carry flag is cleared only when a new key is activated, <u>not</u> when a key is released.

The existence of instructions to test the Carry Flag (BCS-Branch if Carry Set- and BCC-Branch if Carry Clear) make the use of this feature exceptionally easy.

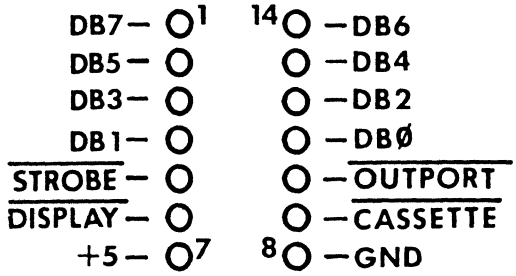
A simple user program to scan the keyboard and display the key that is down could look like this:

```
0020   LOOP     20    JSR  DECODE        ;jump to monitor
                00                       ;keyboard routine
                FF
  23            B0    BCC  LOOP         ;test for new key
                FB
  25            8D    STA  DSP          ;if new key, put
                20                       ;in display and....
                A8
  28            4C    JMP  LOOP         ;begin again
                20
                00
```

It is the op code (BO) at location 0023 and its corresponding operand at the next location that causes the program to skip the display if no key is found down. By replacing these two bytes with NOPs (EA) the program may be modified to display the key number while the key is held down and display 18 (the no-key code) when no keys are pressed.

DATA BUSS and STROBE CONNECTOR (J1)

$\overline{\text{STROBE}}$ - Address x880
$\overline{\text{DISPLAY}}$ - Address x820
$\overline{\text{OUTPORT}}$ - Address x840
CASSETTE-Address x9xx



This connector provides direct access to the data buss as well as a selection of the system peripheral enable signals. Some of the enabling signals are activated when a single address is accessed, others when any one of a group is called for, as summarized below.

Electrical loading is an important consideration in using this connector. Five CMOS loads or one LS TTL is a safe bet, but more than that is on the questionable side. The select lines ($\overline{\text{STROBE}}$, etc) will each drive 4 TTL loads.

The pins labeled DB0-DB7 provide access to the data buss from least significant to most significant respectively.

System +5 volts and ground appear at pins 7 and 8 respectively.

All enable signal lines are memory mapped.

PERIPHERAL ENABLE SIGNALS

$\overline{\text{STROBE}}$ - Provides a low-true signal when any of the following addresses are read from or written to:

x880	x8A0	x8C0	x8E0
x890	x8B0	x8D0	x8F0

$\overline{\text{DISPLAY}}$-This is the select line for the 8700 displays. This line is low-true on a write operation to the address occupied by the displays (x820).

$\overline{\text{OUTPORT}}$-The low-true select line for the output port which lives at address x840 activates on write operations only.

$\overline{\text{CASSETTE}}$-The select line for a contiguous block of 256 addresses from locations x900 - x9FF. Activates on write operations only.

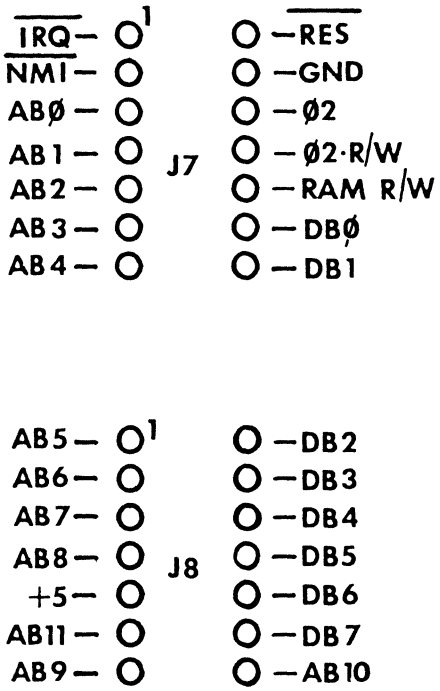
NOTE: All tape dump operations are written to address x900 and this address should be reserved for this operation only. All active addresses above x900 may be used, but if the two relay drivers are used, care must be taken during transfers so that the duty factor of the pulses is not sufficient to close the relays.

EXPANSION CONNECTORS

J7 and J8

EXPANSION CONNECTORS

J7 AND J8



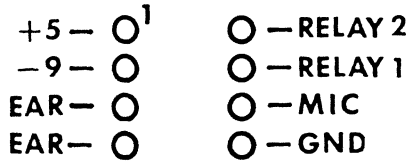
The expansion connectors J7 and J8 provide access to the DATA, ADDRESS, and CONTROL busses of the processor as shown at right.

While these connectors are reserved for future expansions by PAiA, they may be used by the experienced user for system expansion. Appropriate care must be exercised that devices connected to these points do not exceed the loading capabilities of the processor and that appropriate protection against such real-world hazards as overvoltages and transient spikes is provided.

CASSETTE CONNECTOR (J9)

CASSETTE CONNECTOR

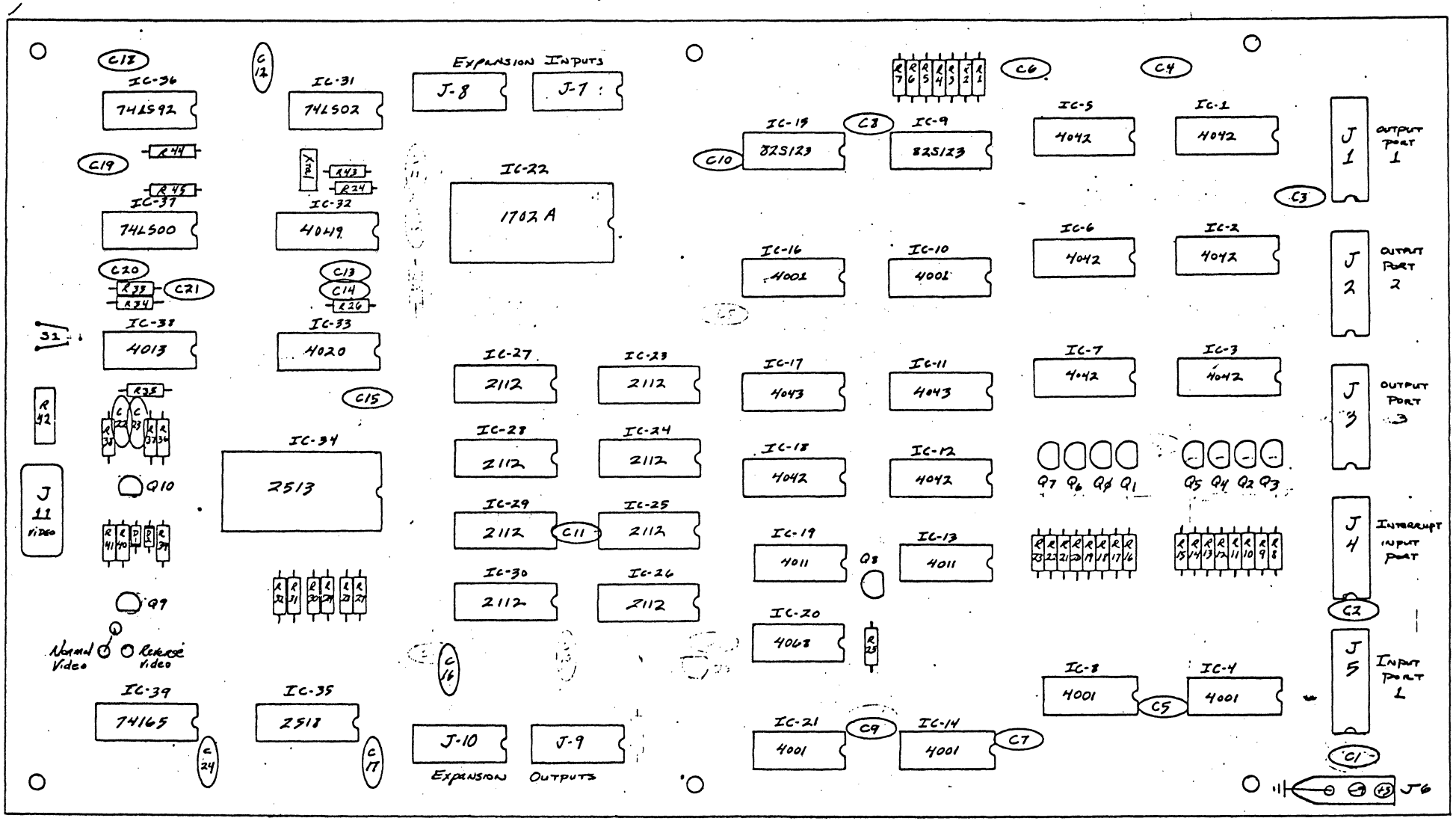
J9



The cassette connector is used in conjunction with the CS-87 option to provide program and data-saving and loading from cassette recorder (see CS-87 Cassette option manual for operating details).

Additionally, this port and its corresponding components provides for a keyboard "beeper" which indicates activation of the control keys of the 8700/A Active Keyboard.

Working Notes



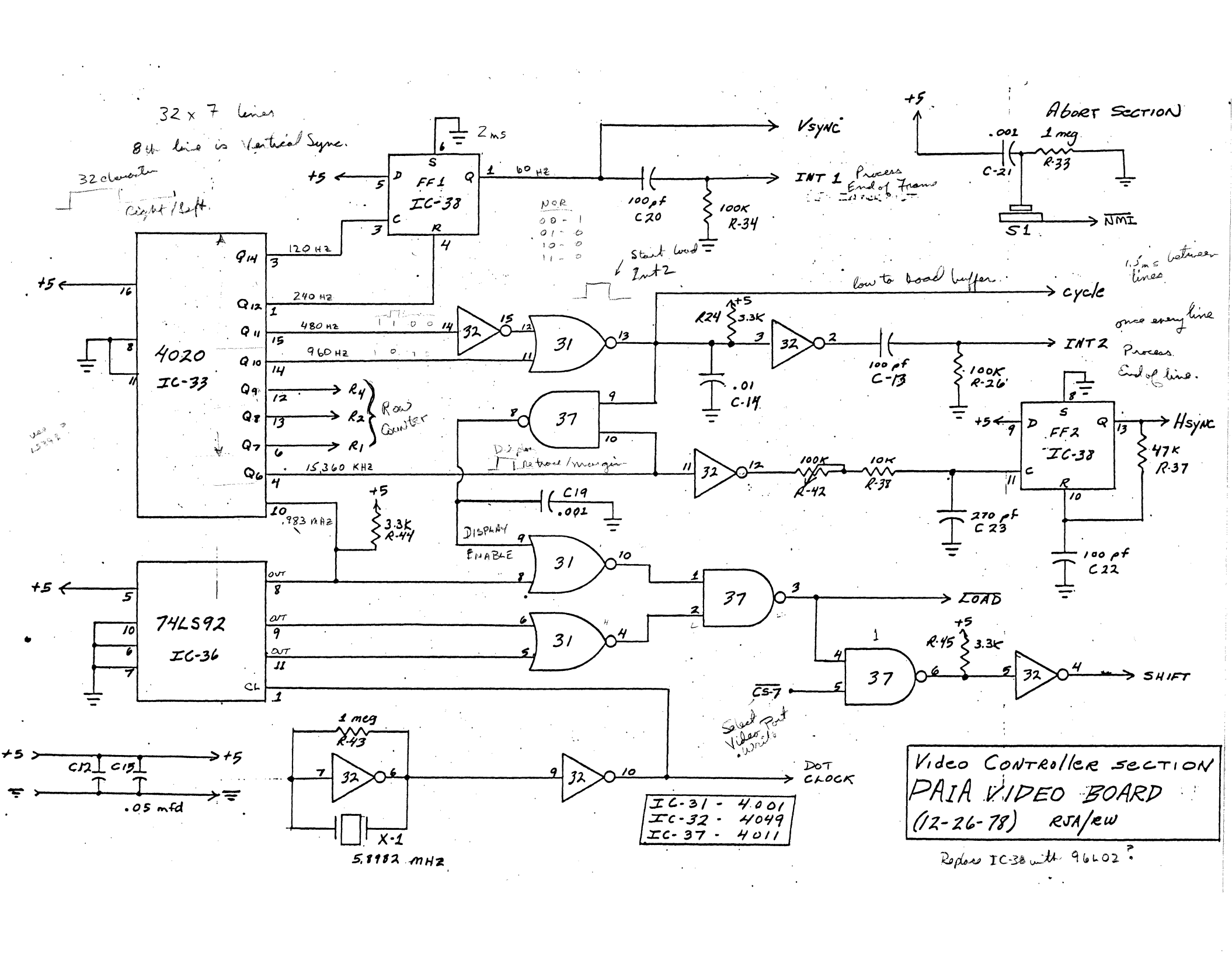
6"

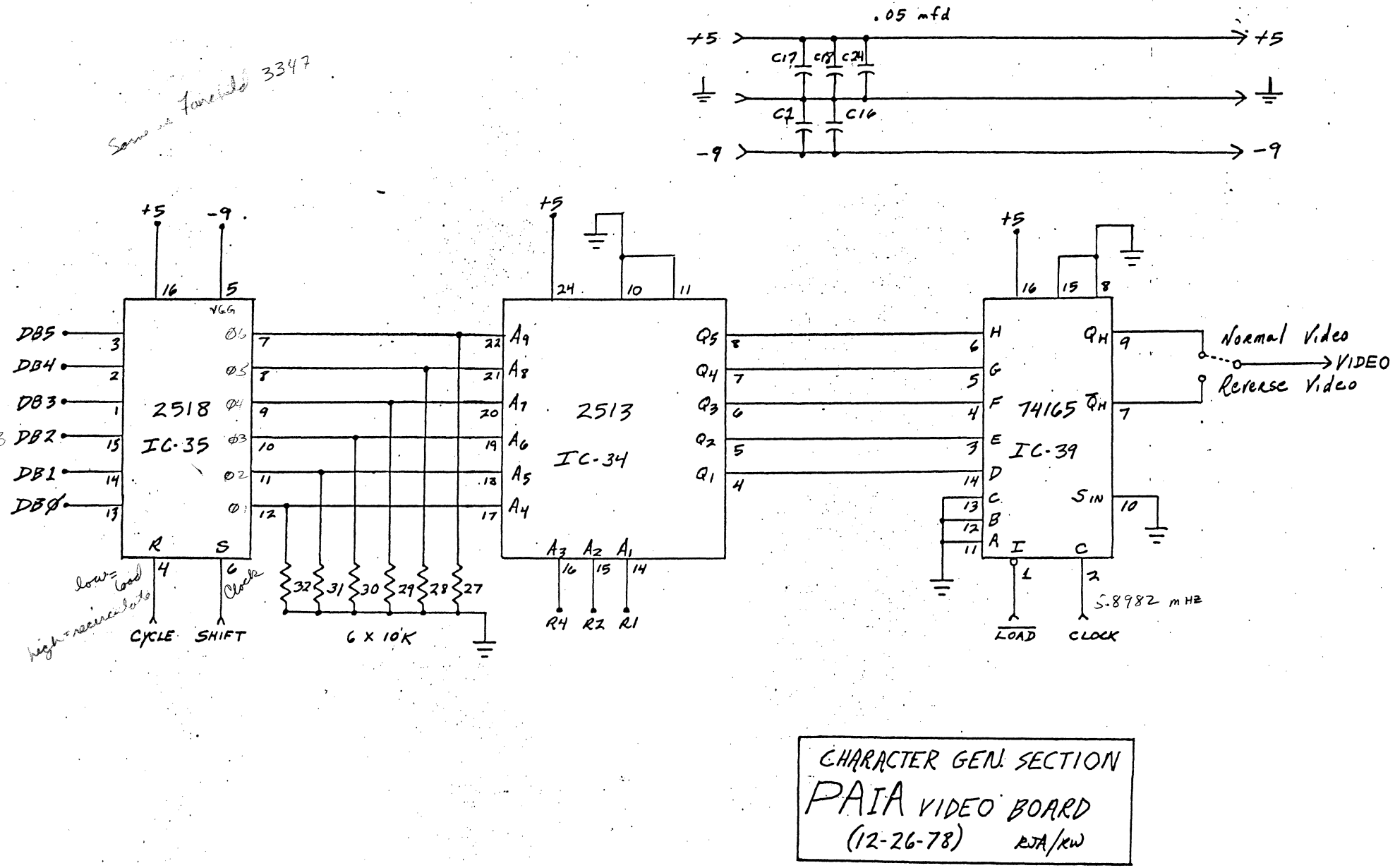
11"

2:1

COMPONENTS LAYOUT
PAIA VIDEO BOARD
(1-1-79) RJA/RW

Ron Hilburn

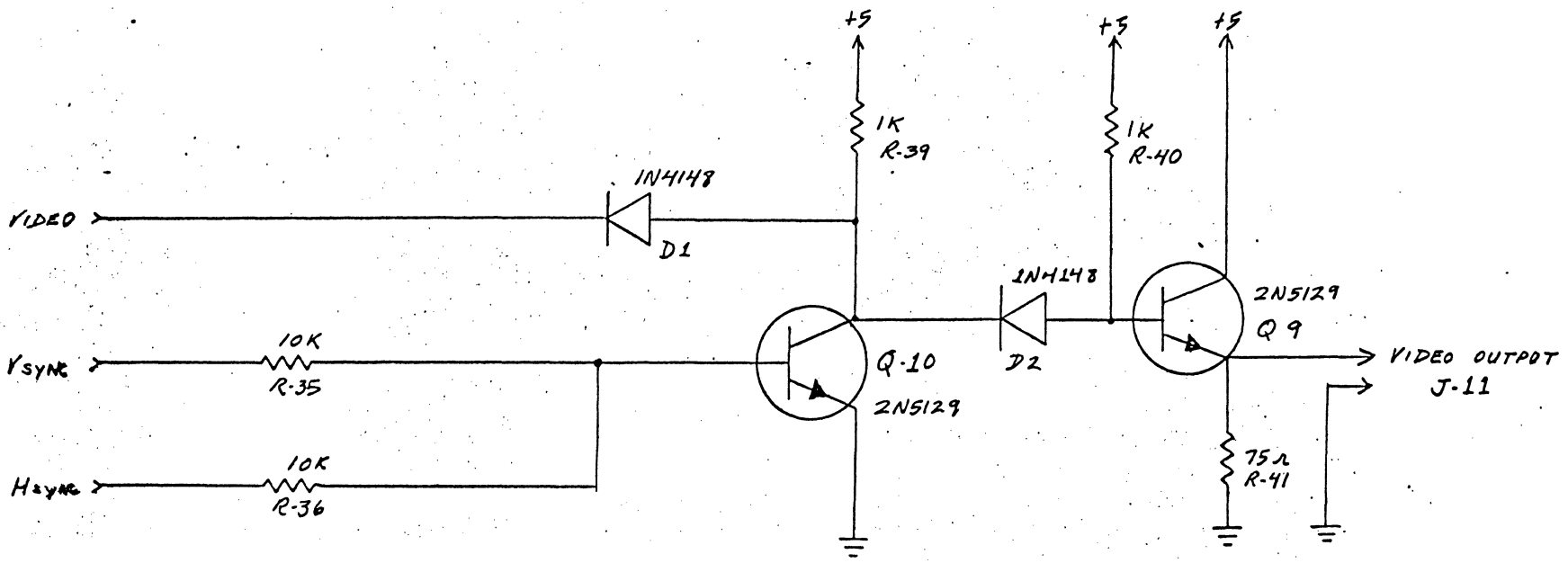

32 x 7 lines
8th line is Vertical Sync.
32 characters
Right/Left
2ms
+5
FF1
IC-38
D
C
S
R
Q
1
60 HZ
VSYNC
INT 1 Process End of Frame
100 pf C20
100K R-34
NOR
00 - 1
01 - 0
10 - 0
11 - 0
Start load Int2
Abort Section
.001
1 meg
R-33
C-21
S1
NMI
4020
IC-33
Q14
Q12
Q11
Q10
Q9
Q8
Q7
Q6
120 HZ
240 HZ
480 HZ
960 HZ
15,360 KHZ
.983 MHZ
R4
R2
R1
Row Counter
low to load buffer
cycle
1.5 ms between lines
once every line
Process End of line.
R24
3.3K
INT2
100 pf C-13
.01 C-14
100K R-26
FF2
IC-38
HSYNC
47K R-37
Retrace/margin
C19
.001
100K
R-42
10K
R-38
270 pf C 23
100 pf C22
3.3K R-44
DISPLAY ENABLE
74LS92
IC-36
OUT
CL
LOAD
R-45
3.3K
SHIFT
CS-7
Select Port Video Write
1 meg
R-43
X-1
5.8982 MHz
C12
C13
.05 mfd
DOT CLOCK
IC-31 - 4001
IC-32 - 4049
IC-37 - 4011
Video Controller Section
PAIA VIDEO BOARD
(12-26-78) RJA/RW
Replace IC-38 with 9602?



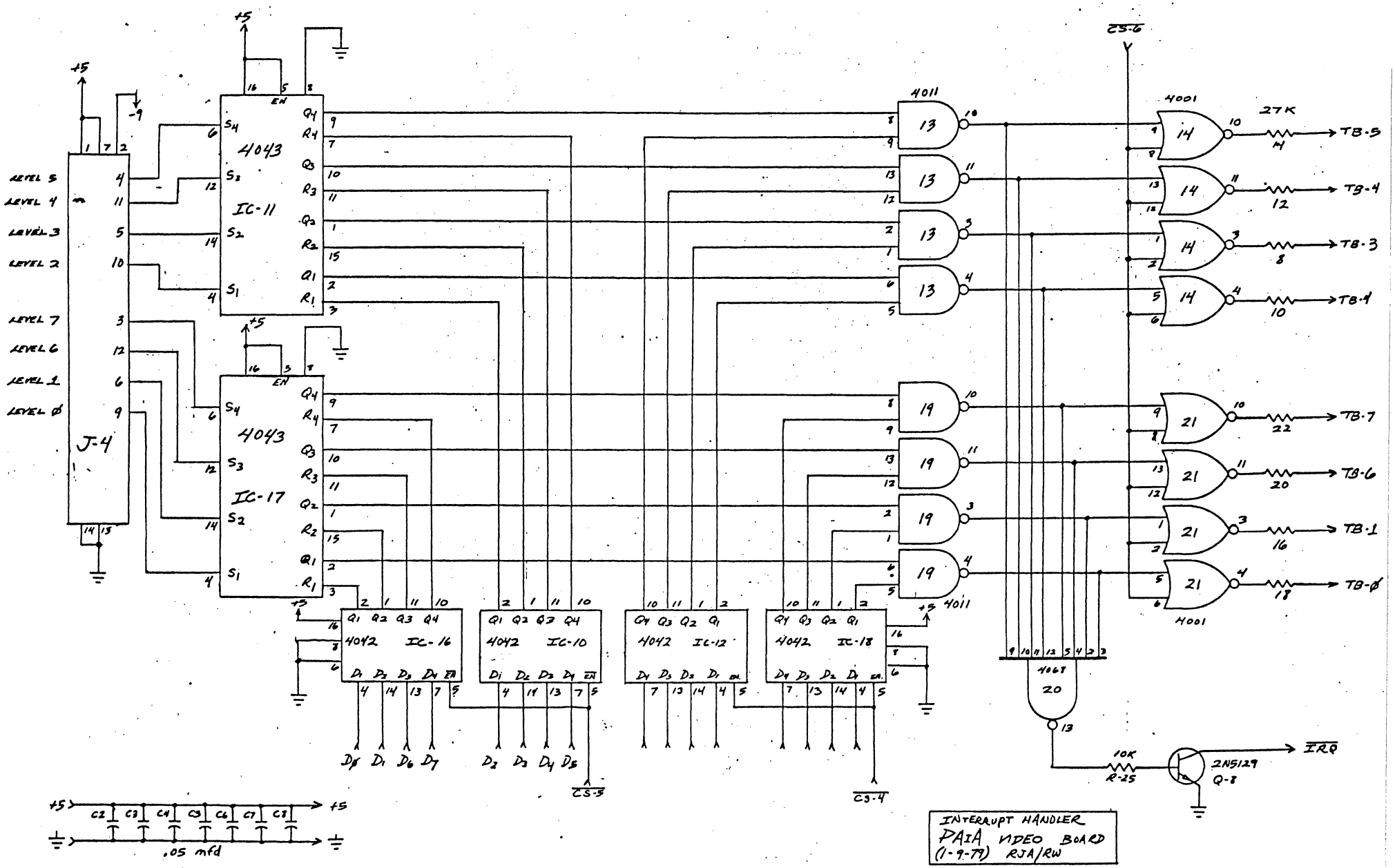
CHARACTER GEN. SECTION

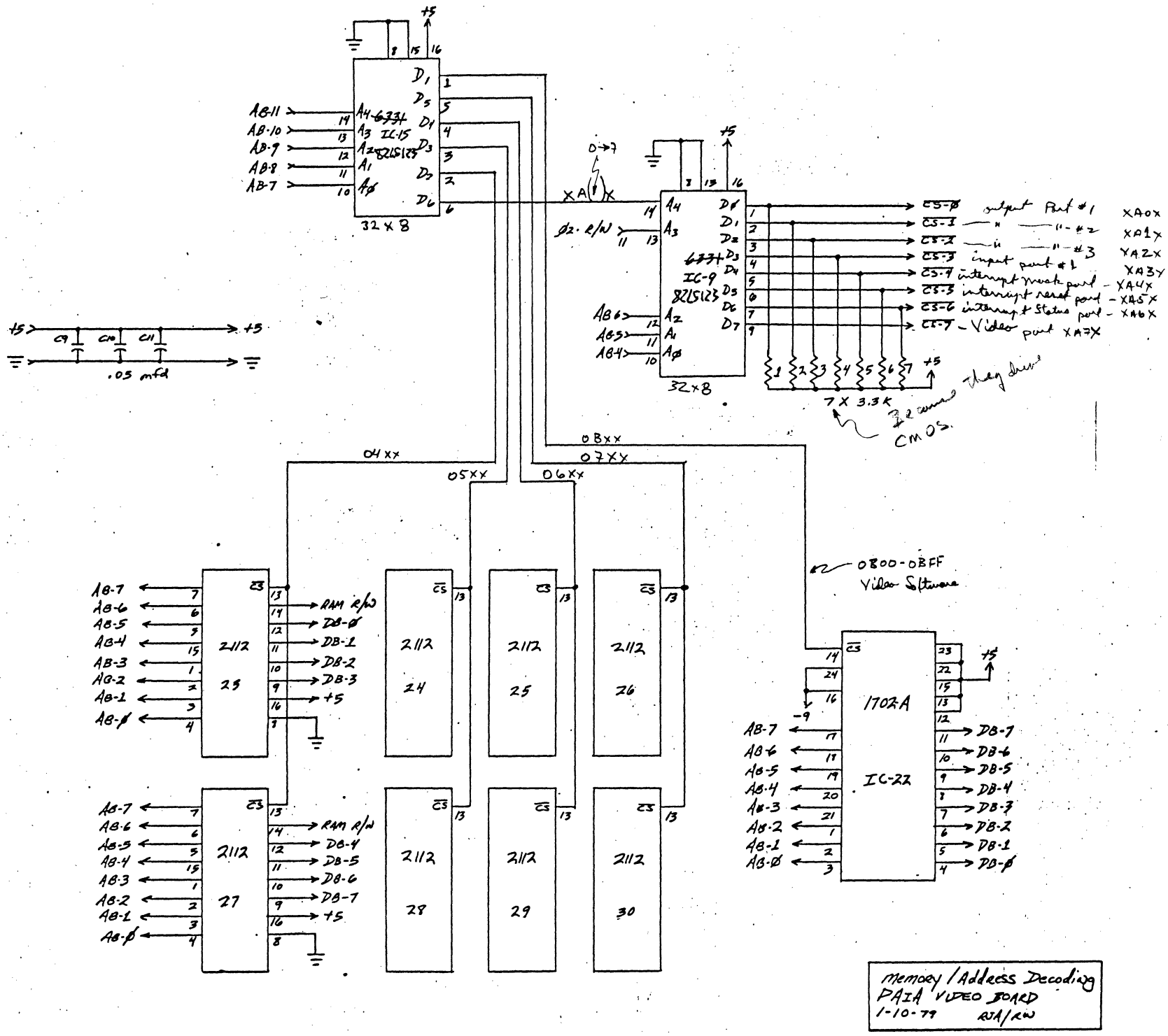
PAIA VIDEO BOARD

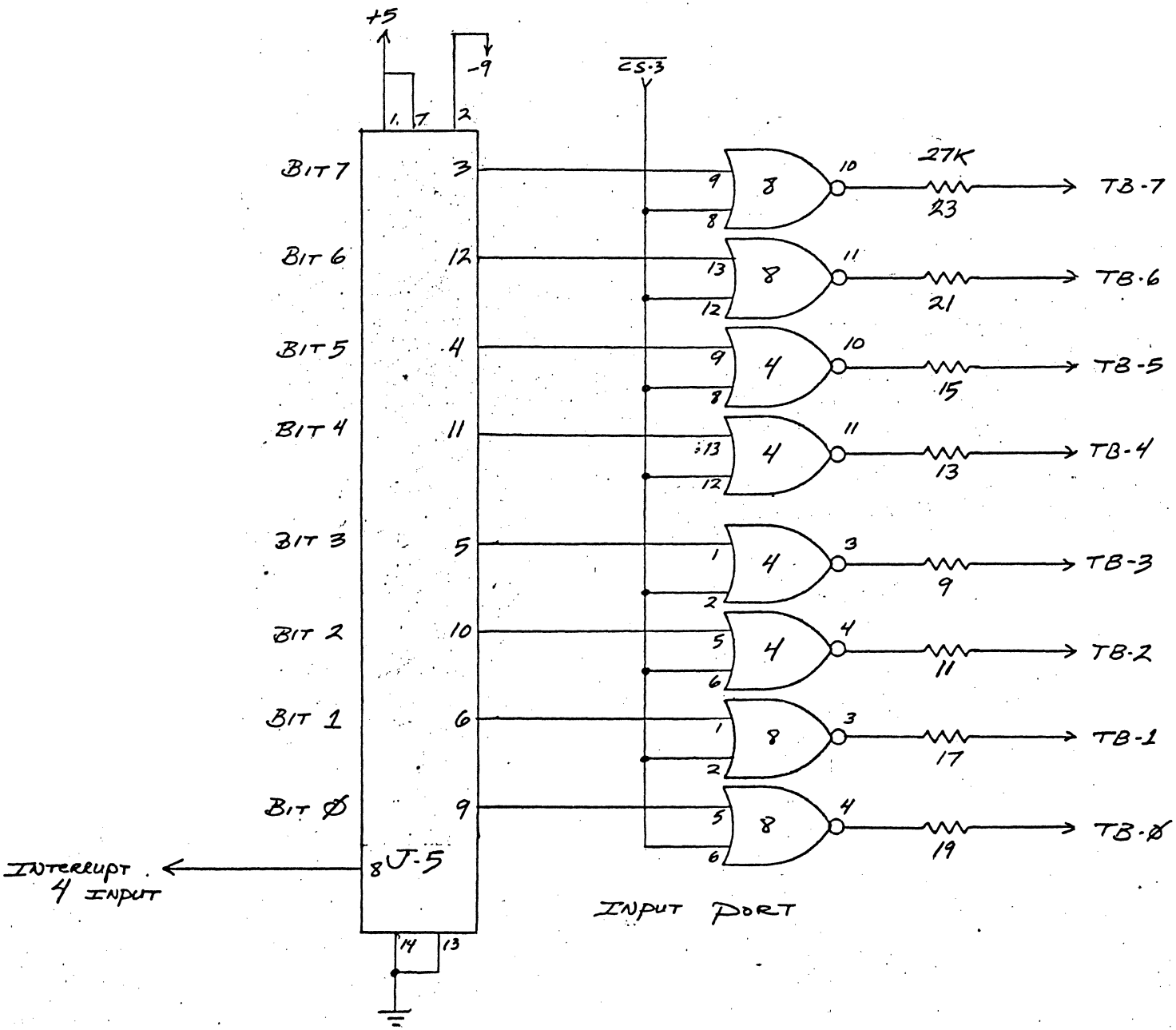
(12-26-78) RJA/RW



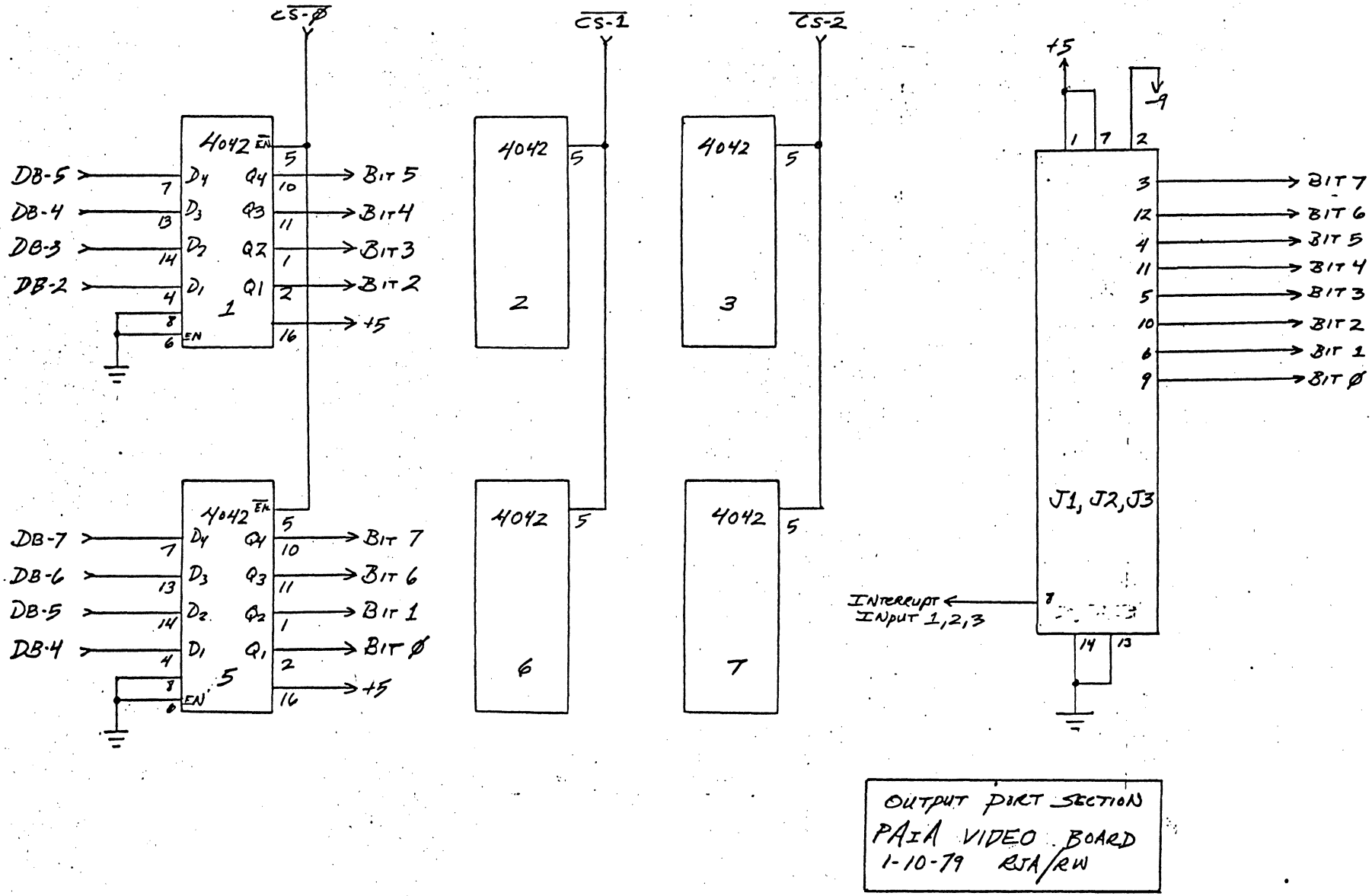
VIDEO MIXER SECTION
PAIA VIDEO BOARD
(12-26-78) RJA/RW


INTERRUPT HANDLER
PAIA VIDEO BOARD
(1-9-79) RJA/RW


+5
AB-11
AB-10
AB-9
AB-8
AB-7
A4 6331
A3 IC-15
A2 82S123
A1
A0
D1
D3
D5
D7
D6
32 x 8
0→7
XA(1)X
A4
A3
Ø2 · R/W
A2
A1
A0
6331
IC-9
82S123
AB-6
AB-5
AB-4
D0
D1
D2
D3
D4
D5
D6
D7
32x8
CS-0 output Port #1 XA0X
CS-1 " " " #2 XA1X
CS-2 " " " #3 XA2X
CS-3 input port #1 XA3X
CS-4 interrupt mask port - XA4X
CS-5 interrupt reset port - XA5X
CS-6 interrupt status port - XA6X
CS-7 - Video port XA7X
1 2 3 4 5 6 7
7 X 3.3K
32 used they drive CMOS
C9 C10 C11
.05 mfd
04XX
05XX
06XX
07XX
0BXX
0800-0BFF
Video Software
2112 25
2112 24
2112 26
2112 27
2112 28
2112 29
2112 30
CS
RAM R/W
DB-0
DB-1
DB-2
DB-3
DB-4
DB-5
DB-6
DB-7
1702A
IC-22
Memory / Address Decoding
PAIA VIDEO BOARD
1-10-79 RJA/RW

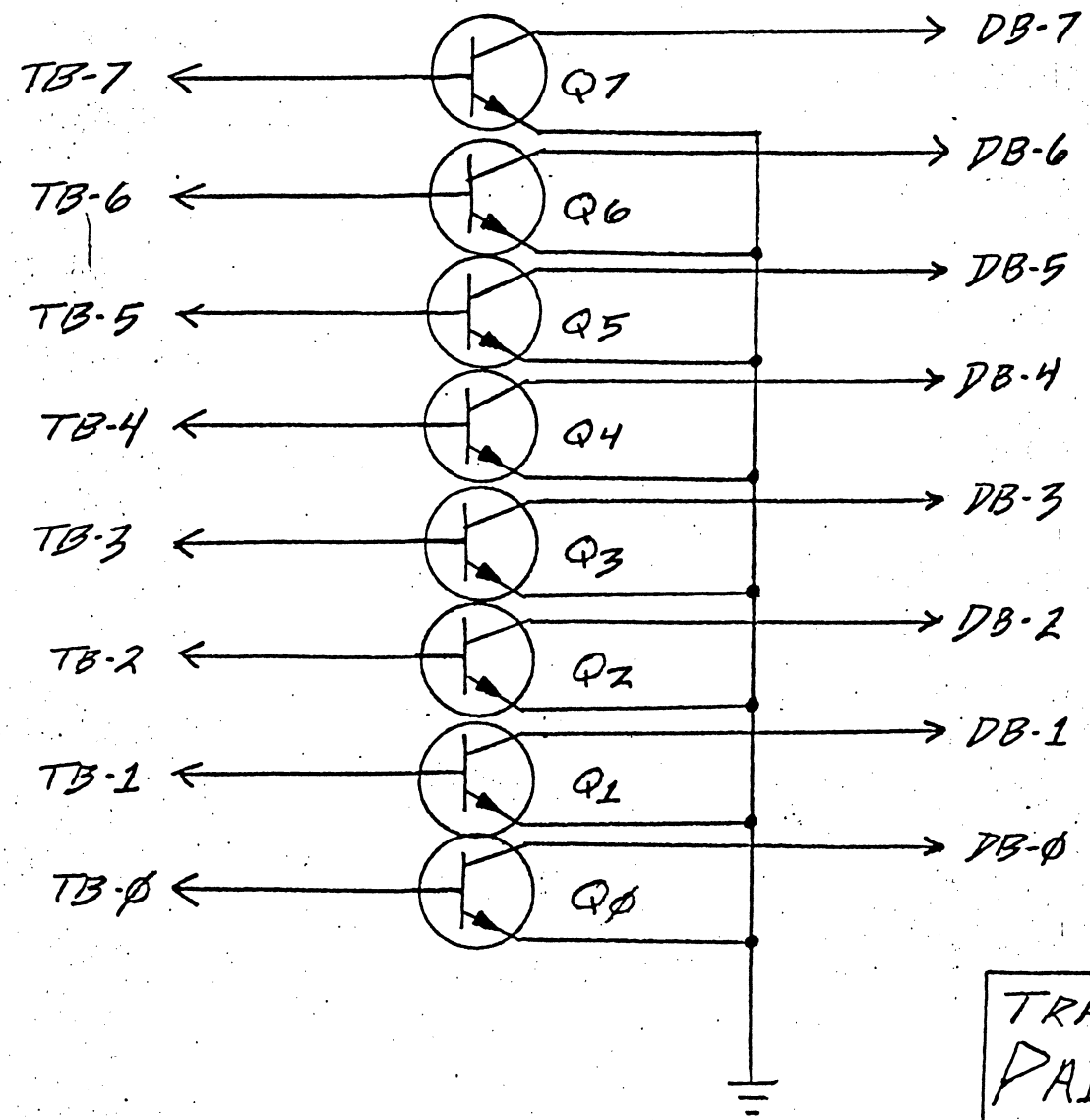


INPUT PORT SECTION
PAIA VIDEO BOARD
1-10-79 RJA/RW



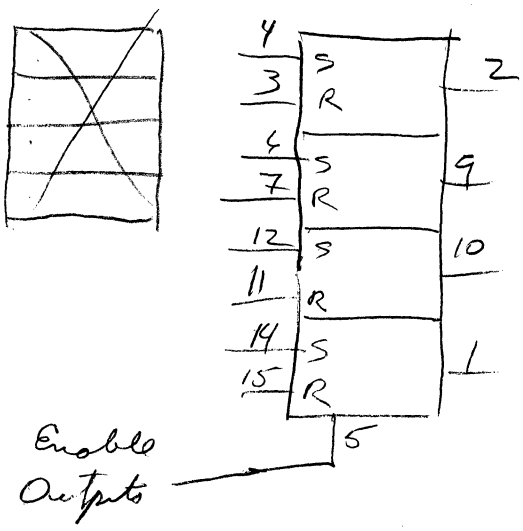
OUTPUT PORT SECTION
PAIA VIDEO BOARD
1-10-79 RJA/RW

2N5129

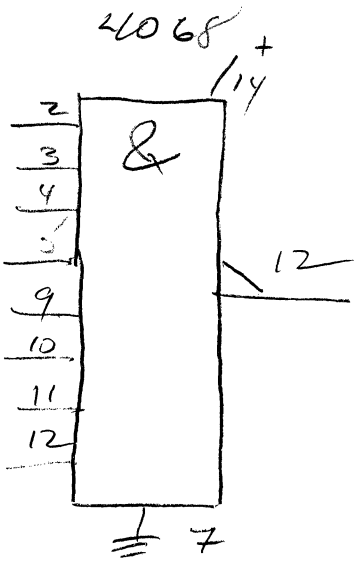


TRANSISTOR BUSS
PAIA VIDEO BOARD
3-12-79 RJA/RW

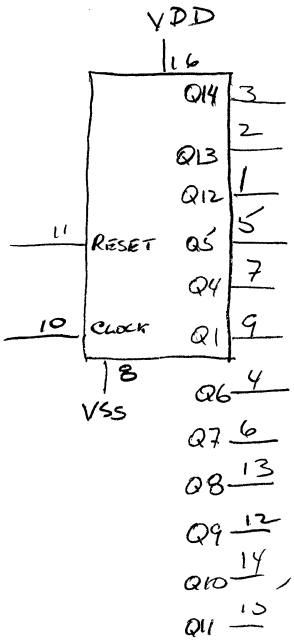
4043 – Quad R-S latch



4068



4020 – 14 Bit Counter 13MHz at 13 volts 1MHz MAX at 5V



Reset High – all outputs low min 320 NS.

Clock No Change

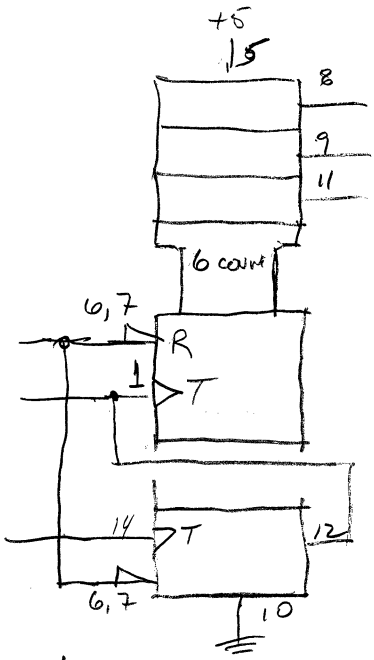
Clock – Count 140 NS minimum at +5V

4001 – Quad 2 input NOR

1/2 – 3 5/6 – 4 8/9 – 10 12/13 – 11

Unused inputs to VSS

LS92

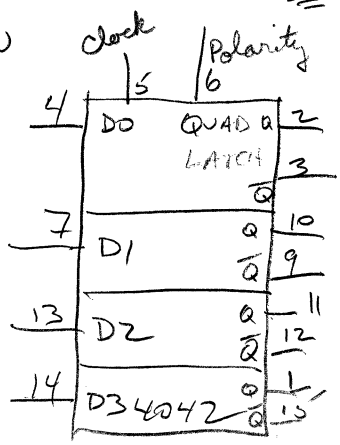


4011 – Same as 4001 but a NAND

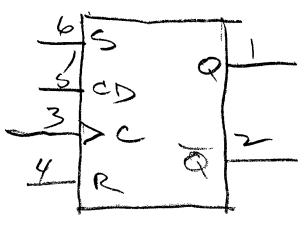
4049 – Hex inverter

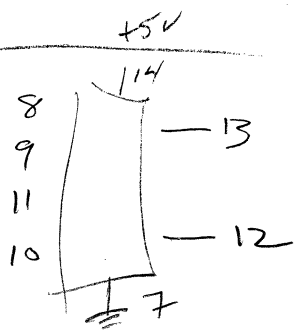
3–2
5–4
7–6
9–10
11–12
14–15

Polarity low/clock low
high/clock high



4013





IC 9

PAIA VIDEO BOARD

1-15-79

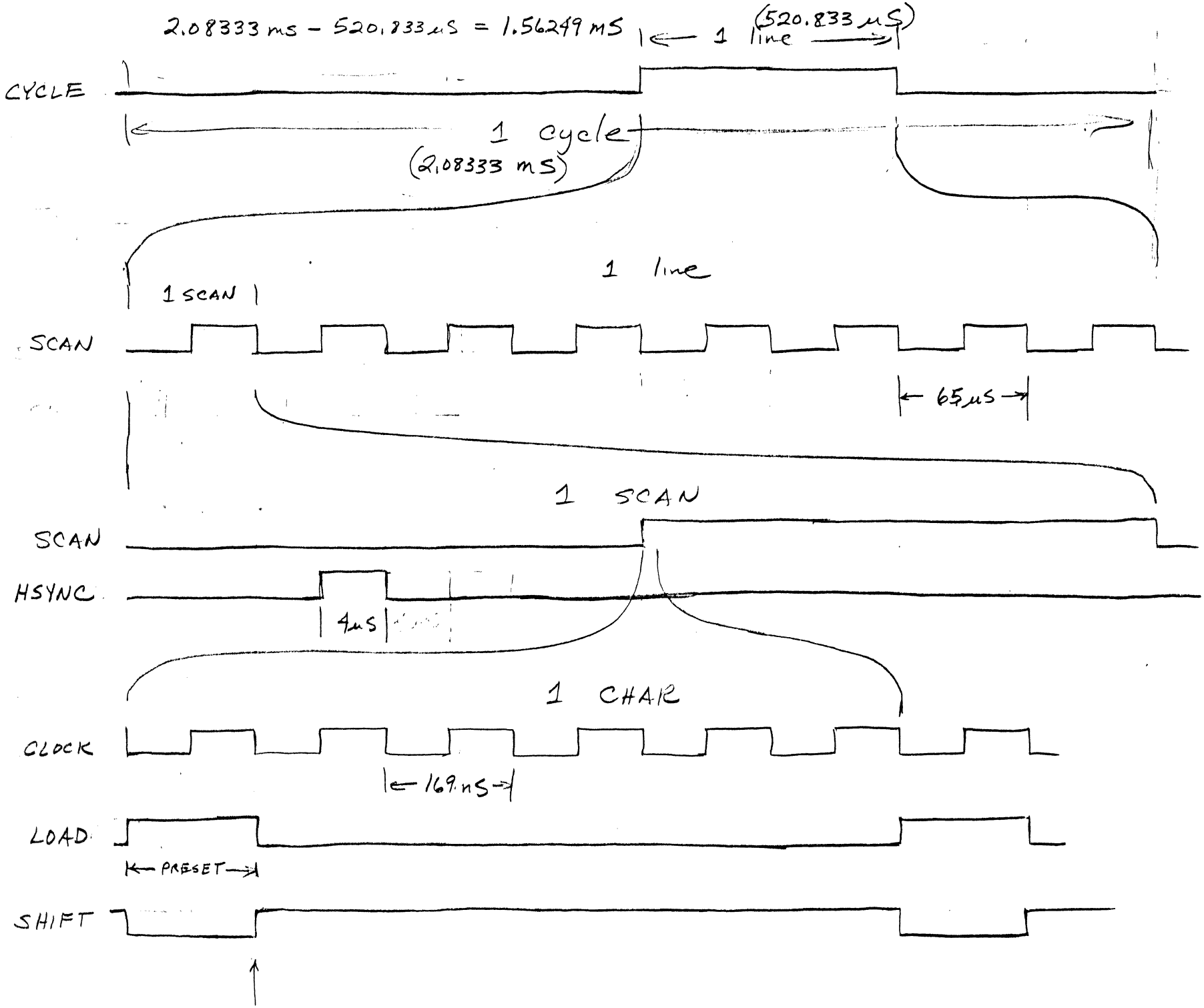
ADDR							PROCESSOR ADDR	DATA								
		I/O ENABLE	Φ2 · R/W	A86	A85	A84	MIDDLE DIGIT OF THREE		OUTPUT PORT 1	OUTPUT PORT 2	OUTPUT PORT 3	INPUT PORT	INTERRUPT MASK	INTERRUPT RESETS	INTERRUPT STATUS	VIDEO
OCTAL		A_4	A_3	A_2	A_1	A_0	HEX	HEX	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
00		0	0	0	0	0	0									
01		0	0	0	0	1	1									
02		0	0	0	1	0	2									
03		0	0	0	1	1	3		1	1	1	0	1	1	1	1
04		0	0	1	0	0	4									
05		0	0	1	0	1	5									
06		0	0	1	1	0	6		1	1	1	1	1	1	0	1
07		0	0	1	1	1	7									
10		0	1	0	0	0	0		0	1	1	1	1	1	1	1
11		0	1	0	0	1	1		1	0	1	1	1	1	1	1
12		0	1	0	1	0	2		1	1	0	1	1	1	1	1
13		0	1	0	1	1	3									
14		0	1	1	0	0	4		1	1	1	1	0	1	1	1
15		0	1	1	0	1	5		1	1	1	1	1	0	1	1
16		0	1	1	1	0	6									
17		0	1	1	1	1	7		1	1	1	1	1	1	1	0
20		1	0	0	0	0	0									
21		1	0	0	0	1	1									
22		1	0	0	1	0	2									
23		1	0	0	1	1	3									
24		1	0	1	0	0	4									
25		1	0	1	0	1	5									
26		1	0	1	1	0	6									
27		1	0	1	1	1	7									
30		1	1	0	0	0	0									
31		1	1	0	0	1	1									
32		1	1	0	1	0	2									
33		1	1	0	1	1	3									
34		1	1	1	0	0	4									
35		1	1	1	0	1	5									
36		1	1	1	1	0	6									
37		1	1	1	1	1	7									

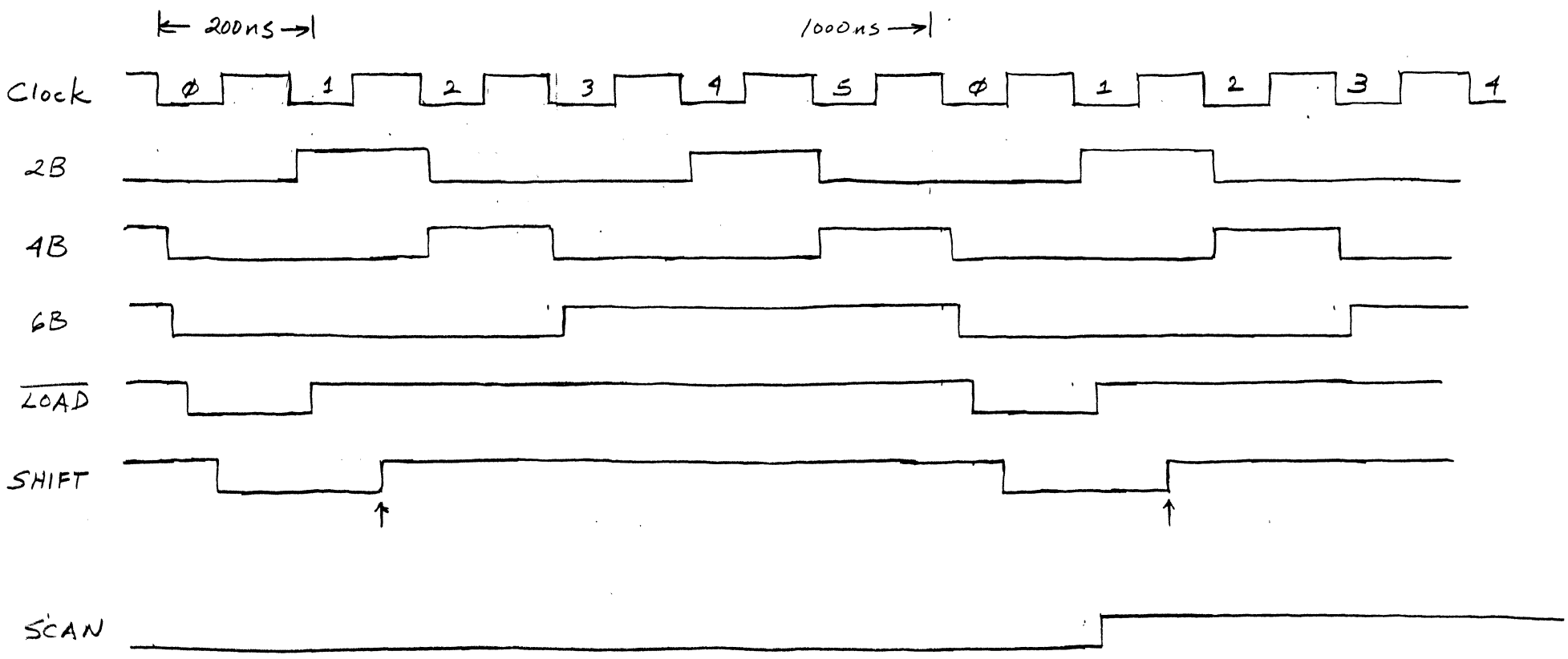
IC 15

PAIA VIDEO BOARD

1-15-79

ADDR												DATA									
OCTAL	HEX	A_4	A_3	A_2	A_1	A_0	ABII	AB10	AB9	AB8	AB7	PROCESSOR ADDR 1ST TWO DIGITS OUT OF THREE	HEX	HEX	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	PROM 7 RAM 6 RAM 5 RAM 4 I/O
00	00	0	0	0	0	0						00	FF	1									
01	01	0	0	0	0	1						08	FF	1									
02	02	0	0	0	1	0						10	FF	1									
03	03	0	0	0	1	1						18	FF	1									
04	04	0	0	1	0	0						20	FF	1									
05	05	0	0	1	0	1						28	FF	1									
06	06	0	0	1	1	0						30	FF	1									
07	07	0	0	1	1	1						38	FF	1									
10	08	0	1	0	0	0						40	F7	1	1	1	1	0	1	1	1		
11	09	0	1	0	0	1						48	F7	1	1	1	1	0	1	1	1		
12	0A	0	1	0	1	0						50	EF	1	1	1	0	1	1	1	1		
13	0B	0	1	0	1	1						58	EF	1	1	1	0	1	1	1	1		
14	0C	0	1	1	0	0						60	DF	1	1	0	1	1	1	1	1		
15	0D	0	1	1	0	1						68	DF	1	1	0	1	1	1	1	1		
16	0E	0	1	1	1	0						70	BF	1	0	1	1	1	1	1	1		
17	0F	0	1	1	1	1						78	BF	1	0	1	1	1	1	1	1		
20	10	1	0	0	0	0						80	FF	1									
21	11	1	0	0	0	1						88	FF	1									
22	12	1	0	0	1	0						90	FF	1									
23	13	1	0	0	1	1						98	FF	1									
24	14	1	0	1	0	0						A0	FB	1	1	1	1	1	0	1	1		
25	15	1	0	1	0	1						A8	FF	1									
26	16	1	0	1	1	0						B0	7F	0	1	1	1	1	1	1	1		
27	17	1	0	1	1	1						B8	7F	0	1	1	1	1	1	1	1		
30	18	1	1	0	0	0						C0	FF	1									
31	19	1	1	0	0	1						C8	FF	1									
32	1A	1	1	0	1	0						D0	FF	1									
33	1B	1	1	0	1	1						D8	FF	1									
34	1C	1	1	1	0	0						E0	FF	1									
35	1D	1	1	1	0	1						E8	FF	1									
36	1E	1	1	1	1	0						F0	FF	1									
37	1F	1	1	1	1	1						F8	FF	1									


2.08333 mS - 520.833 µS = 1.56249 mS
1 line (520.833 µS)
CYCLE
1 cycle (2.08333 mS)
1 line
1 SCAN
SCAN
65µS
1 SCAN
SCAN
HSYNC
4µS
1 CHAR
CLOCK
169 nS
LOAD
PRESET
SHIFT


← 200ns →
1000ns →
Clock
Ø
1
2
3
4
5
Ø
1
2
3
4
2B
4B
6B
LOAD
SHIFT
SCAN

signetics

HEX 32-HEX 40-BIT STATIC SHIFT REGISTERS

2518
2519

SILICON GATE MOS 2500 SERIES

DESCRIPTION

These Signetics 2500 Series Hex 32 and 40-bit recirculating static shift registers consists of enhancement mode P-channel silicon gate MOS devices integrated on a single monolithic chip. Internal recirculation logic plus TTL/DTL level clock signals are provided for maximum interfacing capability.

FEATURES

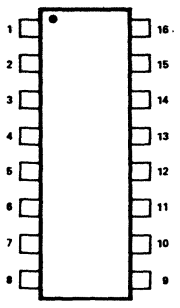
- TYPICAL CLOCK AND DATA RATE = 3MHz
- TTL/DTL COMPATIBLE CLOCK (SINGLE) PROVIDES EXTREMELY LOW CLOCK CAPACITANCE
- RECIRCULATION PATH ON CHIP
- TWO BIT LENGTHS AVAILABLE
- SINGLE-ENDED (BARE DRAIN) BUFFERS
- TTL, DTL COMPATIBLE SIGNALS
- STANDARD PACKAGE – 16 PIN DIP
- SIGNETICS P-MOS SILICON GATE PROCESS TECHNOLOGY

APPLICATIONS

LOW COST SEQUENTIAL ACCESS MEMORIES
LOW COST STATIC BUFFER MEMORIES
CRT REFRESH MEMORIES – LINE STORAGE
LINE PRINTERS
CARD EQUIPMENT BUFFERS

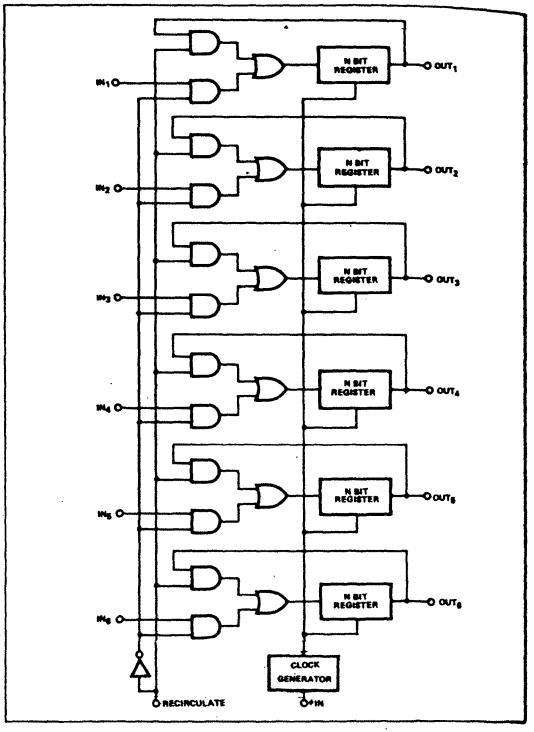
PIN CONFIGURATION (Top View)

B/I PACKAGE



1. IN_4
2. IN_5
3. IN_6
4. Recirculate
5. V_{GG}
6. Clock
7. OUT_6
8. OUT_5
16. V_{CC}
15. IN_3
14. IN_2
13. IN_1
12. OUT_1
11. OUT_2
10. OUT_3
9. OUT_4

BLOCK DIAGRAM



TRUTH TABLE

RECIRCULATE	INPUT	FUNCTION
1	0	Recirculate
1	1	Recirculate
0	0	"0" is Written
0	1	"1" is Written

PART IDENTIFICATION TABLE

PART NUMBER	BIT LENGTH	PACKAGE
2518B	HEX 32	16-Pin Silicone DIP
2518I	HEX 32	16-Pin Ceramic DIP
2519B	HEX 40	16-Pin Silicone DIP
2519I	HEX 40	16-Pin Ceramic DIP

MAXIMUM GUARANTEED RATINGS (1)

Operating Temperature (2)	0°C to +70°C
Storage Temperature	-65°C to +150°C
Package Power Dissipation at T_A = 70°C	640 mW
Data and Clock Input Voltages and Supply Voltages with Respect to V_{CC}	+0.3V to -20V

NOTES:

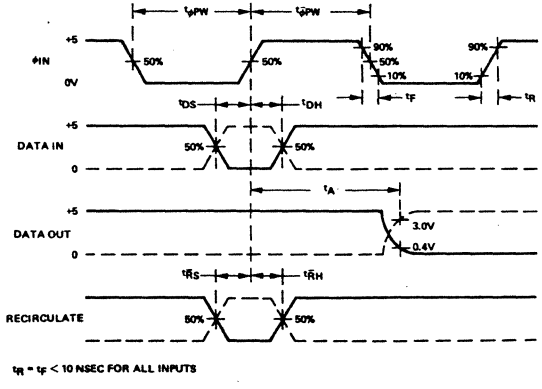
1. Stresses above those listed under "Maximum Guaranteed Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specification is not implied.
2. For operating at elevated temperatures the device must be derated based on a 150°C maximum junction temperature and a thermal resistance of 125°C C/W junction to ambient.
3. All inputs are protected against static charge.
4. Parameters are valid over operating temperature range unless specified.
5. All voltage measurements are referenced to ground.
6. Manufacturer reserves the right to make design and process changes and improvements.
7. Typical values are at +25°C and nominal supply voltages.
8. V_{CC} tolerance is ±5% Any variation in actual V_{CC} will be tracked directly by V_{IL}, V_{IH} and V_{OH} which are stated for a V_{CC} of exactly 5 volts.
9. V_{OL} is dependent on R_L and characteristics of driven gate.

DC CHARACTERISTICS

T_A = 0°C to +70°C; V_{CC} = +5V (8); V_{GG} = -12V ± 5% unless otherwise noted. (Notes: 3,4,5,6,7)

SYMBOL	TEST	MIN	TYP	MAX	UNIT	CONDITIONS
I_{LI}	INPUT LOAD CURRENT		10	500	nA	V_{in} = 5.5V, T_A = 25°C
I_{LO}	OUTPUT LEAKAGE CURRENT		10	1000	nA	T_A = 25°C
I_{LC}	CLOCK LEAKAGE CURRENT		10	500	nA	V_{ILC} = GND, T_A = 25°C
I_{GG}	POWER SUPPLY CURRENT		16	25	mA	CONTINUOUS OPERATION T_A = 25°C F = 2MHz
V_{IL}	INPUT "LOW" VOLTAGE			1.05	V	
V_{IH}	INPUT "HIGH" VOLTAGE	3.2		5.3	V	
V_{ILC}	CLOCK INPUT "LOW" VOLTAGE			1.05	V	
V_{IHC}	CLOCK INPUT "HIGH" VOLTAGE	3.2		5.3	V	

TIMING DIAGRAM

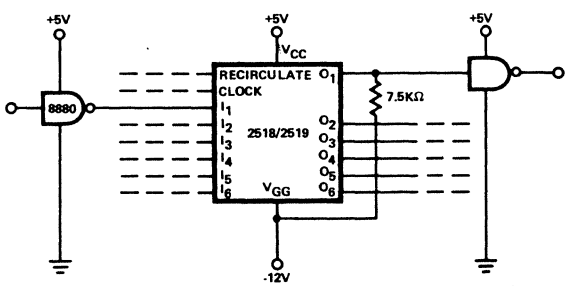


$t_R = t_F < 10$ NSEC FOR ALL INPUTS

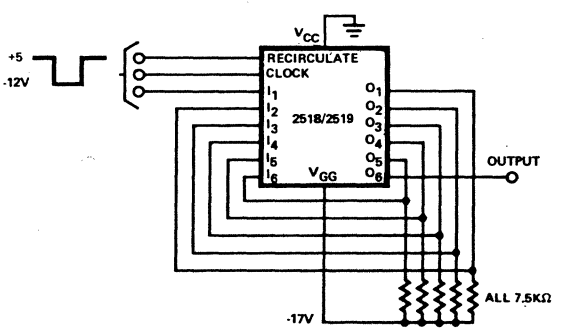
Note: Input rise and fall times: 10nsec. Output load is 1 TTL gate.

AC CHARACTERISTICS $T_A = 0°C$ to $+70°C$, $V_{CC} = +5V$;(8) $V_{GG} = -12V$ +5%, $V_{ILC} = 0.4V$ to $4.0V$

SYMBOL	TEST	MIN	TYP	MAX	UNIT	CONDITIONS
FREQUENCY	CLOCK REP RATE	DC	3	2	MHz	See Max Frequency Curve
$t_{\phi PW}$	CLOCK PULSE WIDTH	.300		100	μsec	
$t_{\overline{\phi}PW}$	CLOCK PULSE WIDTH	.200		DC	μsec	
t_R, t_F	CLOCK PULSE TRANSITION			5	μsec	
t_{DS}	DATA WRITE (SET-UP) TIME	100			nsec	
t_{DH}	DATA TO CLOCK HOLD TIME	50			nsec	
t_A	CLOCK TO DATA OUT DELAY		300	350	nsec	
t_{RS}	RECIRCULATE SET-UP TIME	150			ns	
t_{RH}	RECIRCULATE HOLD TIME	50			ns	
$t_{\overline{\phi}PW}$	CLOCK PULSE WIDTH	.200		DC	μsec	
C_{in}	INPUT CAPACITANCE		5	7	pF	@ 1MHz; $V_{in} = V_{CC}$; V_{AC} = 25mV p-p
$C\phi$	CLOCK CAPACITANCE		6	7	pF	@ 1MHz; $V_\phi = V_{CC}$; V_{AC} = 25mV p-p
V_{OL}	OUTPUT "LOW" VOLTAGE		0.4		V	Note 9
V_{OH}	OUTPUT "HIGH" VOLTAGE	3.6			V	$R_L = 7.5K\Omega$ to V_{GG}



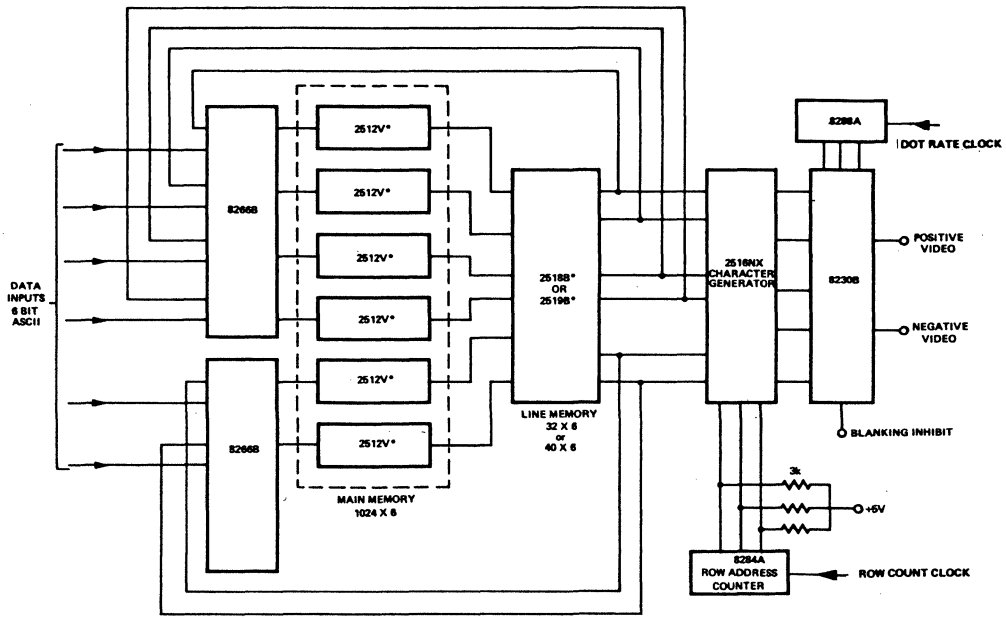
TTL INTERFACE



MOS INTERFACE

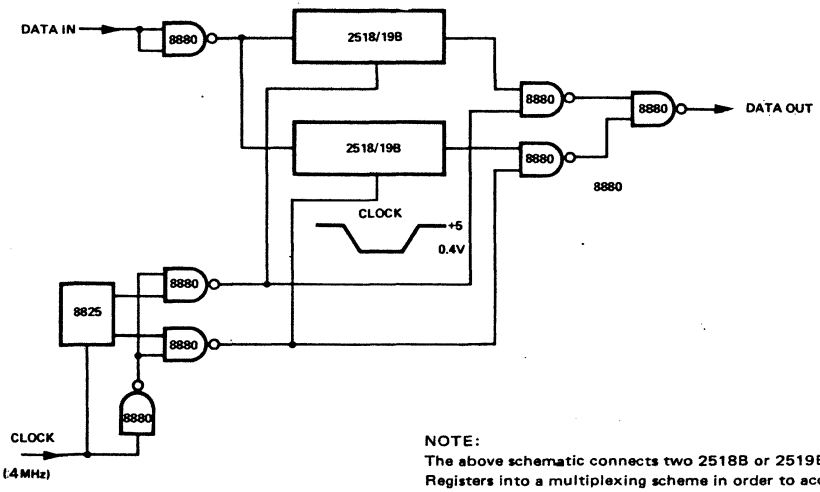
APPLICATIONS DATA

32 or 40 POSITION CRT DISPLAY MEMORY SYSTEM



*These registers include internal recirculates. Two 8266B multiplexers are used for system recirculate.

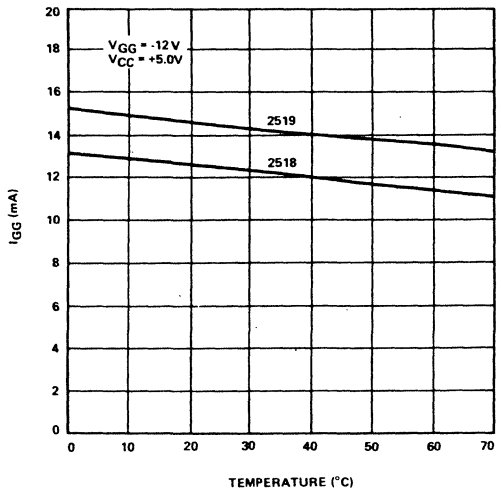
MULTIPLEXING LINE MEMORY REGISTERS AT 4MHz DATA RATE



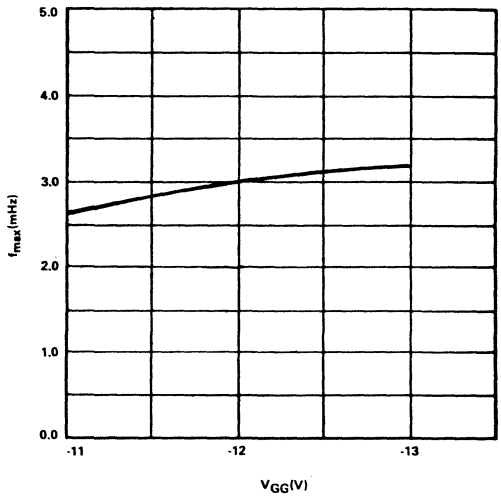
NOTE:
The above schematic connects two 2518B or 2519B Hex Shift Registers into a multiplexing scheme in order to accomplish a 64 or 80 character/line display at 4MHz data rate..

CHARACTERISTIC CURVES

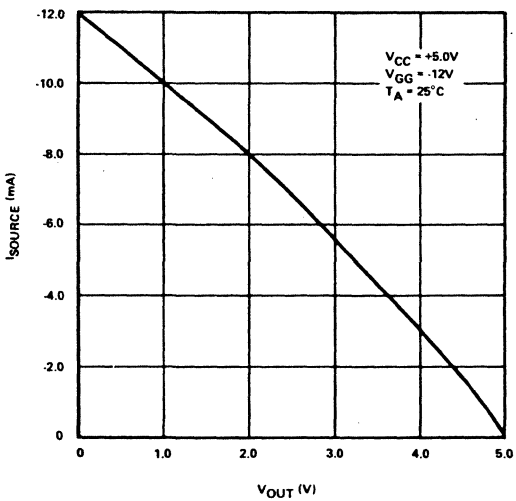
I_{GG} VERSUS TEMPERATURE



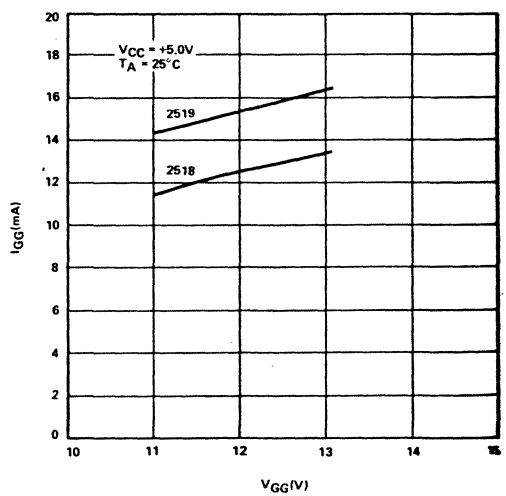
MAXIMUM SHIFT FREQUENCY VERSUS V_{GG}



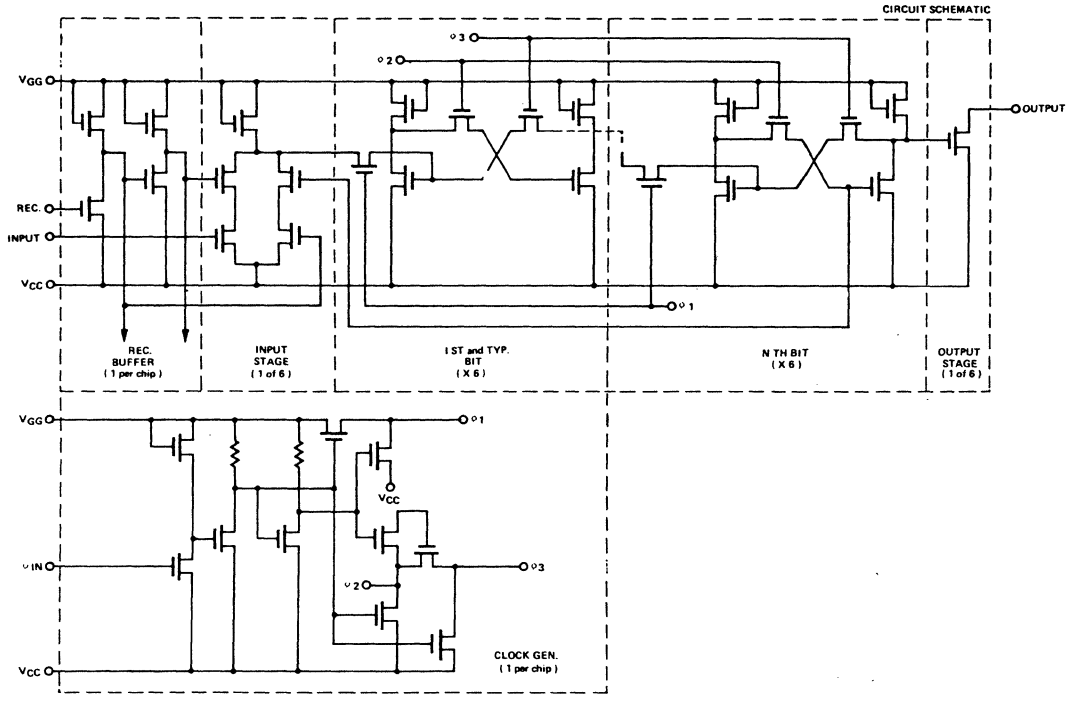
I_{SOURCE} VERSUS V_{OUT}



I_{GG} VERSUS V_{GG}



CIRCUIT SCHEMATIC



```
A

KIMASM

LINE #  LØC     CØDE       LINE

 0010  0200                BRKVEC =$102C
 0020  0200                VECTR0 =$1032
 0030  0200                VECTR1 =$1035
 0040  0200                VECTR2 =$1038
 0050  0200                VECTR3 =$103B
 0060  0200                VECTR4 =$103E
 0070  0200                VECTR5 =$1041
 0080  0200                VECTR6 =$1044
 0090  0200                VECTR7 =$1047
 0100  0200                LØØP   =$10B5
 0110  0200
 0120  0200
 0130  0200
 0140  0200                ;PAIA "VIDEØ WINDØW" SØFTWARE
 0150  0200
 0160  0200
 0170  0200
 0180  0200                ;  1-19-79  20:00
 0190  0200
 0200  0200                VIDEØ  =$0A70          ;PØRT ADDR FØR VIDEØ BØA
 0210  0200                KEYBRD =$0A30          ;ASCII KEYBØARD PØRT
 0220  0200                BASE   =$1700          ;DISPLAY PAGE BASE
 0230  0200                CURSER =$EC            ;CURSER PØSITIØN
 0240  0200                CURFLG =$EB            ;CURSER ØN-ØFF; 0=ØN
 0250  0200                JIFFY  =$EA            ;1/60 ØF A SECØND
 0260  0200                SECØND =$E9
 0270  0200                MINUTE =$E8
 0280  0200                HØUR   =$E7
 0290  0200                FLASHR =$E6            ;CURSER FLASH STATUS
 0300  0200                CLKFLG =$E5            ;CLØCK FLAG, 0=ØFF
 0310  0200                YTEMP  =$E4
 0320  0200                XTEMP  =$E3
 0330  0200                BUFFER =$F0            ;MØNITØR KEYBØARD BUFFER
 0340  0200                GETKEY =$0F1F          ;MØNITØR RØUTINE
 0350  0200                SHIFT  =$0F34          ;MØNITØR RØUTINE
 0360  0200                DISPLY =$0820          ;LED DISPLAYS
 0370  0200                ISTATS =$0A60          ;INTERRUPT STATUS
 0380  0200                IRESET =$0A50          ;INTERRUPT RESETS
 0390  0200                IMASK  =$0A40          ;INTERRUPT MASK
 0400  0200                KEYBUF =$E2            ;KEYSTRØKE BUFFER
 0410  0200                KEYFLG =$E1            ;KEY FLAG, $80=KEY
 0420  0200
 0430  0200                      *=$1000
```

```
0440  1000
0450  1000                ;INTERRUPT SERVICE RØUTINE
0460  1000                ;-------------------------
0470  1000
0480  1000  48            INT     PHA             ;SAVE A
0490  1001  8A                    TXA
0500  1002  48                    PHA             ;SAVE X
0510  1003
0520  1003  BA                    TSX
0530  1004  E8                    INX
0540  1005  E8                    INX
0550  1006  E8                    INX
0560  1007  BD 00 01              LDA $0100,X     ;GET P-REG
0570  100A  29 10                AND #%00010000  ;BREAK FLAG
0580  100C  D0 1E                BNE BRKVEC      ;BRA IF BREAK
0590  100E
0600  100E  AD 60 0A              LDA ISTATS      ;GET INTERRUPT STATUS
0610  1011  6A                    RØR A
0620  1012  90 1E                BCC VECTR0
0630  1014  6A                    RØR A
0640  1015  90 1E                BCC VECTR1
0650  1017  6A                    RØR A
0660  1018  90 1E                BCC VECTR2
0670  101A  6A                    RØR A
0680  101B  90 1E                BCC VECTR3
0690  101D  6A                    RØR A
0700  101E  90 1E                BCC VECTR4
0710  1020  6A                    RØR A
0720  1021  90 1E                BCC VECTR5
0730  1023  6A                    RØR A
0740  1024  90 1E                BCC VECTR6
0750  1026  6A                    RØR A
0760  1027  90 1E                BCC VECTR7
0770  1029  20 ** **              JSR ERRØR       ;PØINT ØF NØ RETURN
0780  102C
0790  102C  68            BRKVEC  PLA             ;RESTØRE X
0800  102D  AA                    TAX
0810  102E  68                    PLA             ;RESTØRE A
0820  102F  4C C0 0F              JMP $0FC0       ;MØNITØR BREAK RØUTINE
0830  1032
0840  1032  20 ** **      VECTR0  JSR ERRØR       ;INTERRUPT 0
0850  1035  4C ** **      VECTR1  JMP INT1        ;VIDEØ, END ØF FRAME
0860  1038  4C ** **      VECTR2  JMP INT2        ;VIDEØ, END ØF LINE
0870  103B  4C ** **      VECTR3  JMP INT3        ;ASCII KEYBØARD
0880  103E  20 ** **      VECTR4  JSR ERRØR
0890  1041  20 ** **      VECTR5  JSR ERRØR
0900  1044  20 ** **      VECTR6  JSR ERRØR
0910  1047  20 ** **      VECTR7  JSR ERRØR
0920  104A
0930  104A  08            ERRØR   PHP             ;SIMULATE BREAK
0940  104B  4C C0 0F              JMP $0FC0       ;MØNITØR BREAK RØUTINE
0950  104E
```

```
0960  104E
0970  104E                ;END ØF FRAME SERVICE RØUTINE
0980  104E                ;----------------------------
0990  104E  98            INT1    TYA
1000  104F  48                    PHA             ;SAVE Y
1010  1050
1020  1050  A9 20                 LDA #32
1030  1052  8D B6 10              STA LØØP+1      ;RESET DISPLAY BASE ADDR
1040  1055  A9 06                 LDA #6
1050  1057  8D B7 10              STA LØØP+2
1060  105A
1070  105A  A2 04                 LDX #4
1080  105C  F8                    SED
1090  105D  38                    SEC
1100  105E  B5 E6          NEXT    LDA HØUR-1,X
1110  1060  69 00                 ADC #0
1120  1062  C9 60                 CMP #$60
1130  1064  90 ** **              BCC SKIP
1140  1067  A9 00                 LDA #0
1150  1069  95 E6          SKIP    STA HØUR-1,X
1160  106B  CA                    DEX
1170  106C  D0 F0                 BNE NEXT
1180  106E  C9 24                 CMP #$24
1190  1070  D0 ** **              BNE SKIP1
1200  1073  A9 00                 LDA #0
1210  1075  85 E7                 STA HØUR
1220  1077
1230  1077  C6 E6          SKIP1   DEC FLASHR
1240  1079  D0 ** **              BNE SKIP3
1250  107C  A9 10                 LDA #16
1260  107E  85 E6                 STA FLASHR
1270  1080
1280  1080  A5 EB          SKIP3   LDA CURFLG
1290  1082  D0 ** **              BNE SKIP2
1300  1085  A6 EC                 LDX CURSER
1310  1087  A5 E6                 LDA FLASHR
1320  1089  C9 08                 CMP #8
1330  108B  A9 20                 LDA #$20        ;CURSER ØFF
1340  108D  B0 ** **              BCS STØRE
1350  1090  A9 5F                 LDA #$5F        ;CURSER ØN
1360  1092  9D 00 17    STØRE     STA BASE,X
1370  1095
1380  1095  A5 E5          SKIP2   LDA CLKFLG
1390  1097  F0 ** **              BEQ SKIP5
1400  109A  20 ** **              JSR PCLØCK      ;PRINT CLØCK
1410  109D
1420  109D  A9 02          SKIP5   LDA #%00000010
1430  109F  8D 50 0A              STA IRESET      ;ACK INTERRUPT
1440  10A2  A9 00                 LDA #0
1450  10A4  8D 50 0A              STA IRESET
1460  10A7  68                    PLA
1470  10A8  A8                    TAY             ;RESTØRE Y
1480  10A9  68                    PLA
1490  10AA  AA                    TAX             ;RESTØRE X
1500  10AB  68                    PLA             ;RESTØRE A
1510  10AC  40                    RTI             ;RETURN
1520  10AD
```

```
1530  10AD
1540  10AD                    ;END ØF LINE SERVICE RØUTINE
1550  10AD                    ;---------------------------
1560  10AD
1570  10AD  AD B6 10    INT2  LDA L0ØP+1
1580  10B0  F0 ** **          BEQ SKIP6        ;BRA IF LAST LINE DISPLE
1590  10B3
1600  10B3  A2 E0             LDX #$E0
1610  10B5  BD 00 10    L0ØP  LDA $1000,X      ;GET CHARACTER
1620  10B8  8D 70 0A          STA VIDEØ        ;SEND IT TØ VIDEØ BØARD
1630  10BB  E8               INX              ;LAST CHAR ØN LINE?
1640  10BC  D0 F7             BNE L0ØP         ;BRANCH IF NØT
1650  10BE
1660  10BE  D8               CLD
1670  10BF  18               CLC
1680  10C0  AD B6 10          LDA L0ØP+1       ;BASE ADDR, LØW BYTE
1690  10C3  69 20             ADC #32          ;SETUP FØR NEXT LINE
1700  10C5  8D B6 10          STA L0ØP+1
1710  10C8
1720  10C8  A9 04        SKIP6 LDA #%00000100
1730  10CA  8D 50 0A          STA IRESET       ;ACK INTERRUPT
1740  10CD  A9 00             LDA #0
1750  10CF  8D 50 0A          STA IRESET
1760  10D2  68               PLA
1770  10D3  AA               TAX              ;RESTØRE X
1780  10D4  68               PLA              ;RESTØRE A
1790  10D5  40               RTI              ;RETURN
1800  10D6
1810  10D6
1820  10D6                    ;---------------------------------------
1830  10D6
1840  10D6                    *=$0200
1850  0200
1860  0200  4C ** **          JMP TEST
1870  0203
1880  0203  A2 E0        INIT  LDX #$E0
1890  0205  A9 20             LDA #$20
1900  0207  9D FF 16    L0ØP4 STA BASE-1,X     ;CLEAR SCREEN
1910  020A  CA               DEX
1920  020B  D0 FA             BNE L0ØP4
1930  020D  86 EC             STX CURSER
1940  020F  86 EB             STX CURFLG       ;ENABLE CURSER
1950  0211  86 E4             STX YTEMP
1960  0213  8E 50 0A          STX IRESET
1970  0216  86 E1             STX KEYFLG
1975  0218  86 E5             STX CLKFLG       ;TURN ØFF CLØCK PRINT
1980  021A  A9 15             LDA #$15
1990  021C  85 E3             STA XTEMP
2000  021E  A9 0E             LDA #%00001110
2010  0220  8D 40 0A          STA IMASK        ;INTERRUPT MASK
2020  0223  58               CLI
2030  0224  60               RTS
2040  0225
2050  ****  ** **
```

```
2050  0225  A6 EC       ØUTCHA LDX CURSER
2060  0227  48                 PHA
2070  0228  A9 20               LDA #$20
2080  022A  85 EB               STA CURFLG       ;TURN ØFF CURSER
2090  022C  9D 00 17            STA BASE,X       ;REMØVE CURSER
2100  022F  68                 PLA
2110  0230  C9 0D               CMP #$0D
2120  0232  F0 ** **            BEQ CR
2130  0235  C9 0A               CMP #$0A
2140  0237  F0 ** **            BEQ LF
2150  023A  C9 7F               CMP #$7F         ;BACKSPACE (RUBØUT KEY)
2160  023C  F0 ** **            BEQ BACKSP
2170  023F  C9 03               CMP #$03         ;CLEAR SCREEN (CNTRL-C)
2180  0241  F0 ** **            BEQ CLEAR
2190  0244  C9 11               CMP #$11         ;CLØCK ØN (CNTRL-Q)
2200  0246  F0 ** **            BEQ CLKØN
2210  0249  C9 13               CMP #$13         ;CLØCK ØFF (CNTRL-S)
2220  024B  F0 ** **            BEQ CLKØFF
2230  024E
2240  024E  9D 00 17            STA BASE,X       ;DISPLAY CHAR
2250  0251  E8                 INX
2260  0252  86 EC               STX CURSER
2270  0254  E0 E0               CPX #$E0         ;END ØF DISPLAY PAGE?
2280  0256  90 ** **            BCC RET          ;RETURN IF NØT
2290  0259  8A                 TXA              ;CURSER
2300  025A  D8                 CLD
2310  025B  38                 SEC
2320  025C  E9 20               SBC #32          ;SET CURSER TØ BEG ØF LI
2330  025E  85 EC               STA CURSER
2340  0260  D0 ** **            BNE SCRØLL       ;BRANCH ALWAYS
2350  0263
2360  0263  8A           CR    TXA              ;CURSER
2370  0264  29 E0               AND #%11100000
2380  0266  85 EC               STA CURSER
2390  0268  AA                 TAX
2400  0269
2410  0269  8A           LF    TXA              ;CURSER
2420  026A  C9 C0               CMP #$C0         ;CURSER ØN LAST LINE?
2430  026C  B0 ** **            BCS SCRØLL       ;BRANCH IF SØ
2440  026F  18                 CLC
2450  0270  D8                 CLD
2460  0271  69 20               ADC #32          ;CURSER TØ NEXT LINE
2470  0273  85 EC               STA CURSER
2480  0275  D0 ** **            BNE RET          ;BRANCH ALWAYS
2490  0278
2500  0278  8A           BACKSP TXA              ;CURSER
2510  0279  F0 ** **            BEQ RET          ;BRA IF CURSER AT BEGIN
2520  027C  CA                 DEX
2530  027D  86 EC               STX CURSER
2540  027F  4C ** **            JMP RET
2550  0282
```

```
2560  0282  20 03 02    CLEAR  JSR INIT          ;CLEAR SCREEN
2570  0285  60                 RTS
2580  0286
2590  0286  A9 01        CLKØN  LDA #1
2600  0288  D0 ** **            BNE SET
2610  028B  A9 00        CLKØFF LDA #0
2620  028D  85 E5        SET    STA CLKFLG
2630  028F  4C ** **            JMP RET
2640  0292
2650  0292  A2 3F        SCRØLL LDX #63
2660  0294  BD E0 16     LØØP1  LDA BASE-32,X
2670  0297  9D C0 16            STA BASE-64,X
2680  029A  E8                  INX
2690  029B  D0 F7               BNE LØØP1
2700  029D
2710  029D  A2 1F,               LDX #31
2720  029F  A9 20                LDA #$20
2730  02A1  9D C0 17     LØØP2  STA BASE+$C0,X  ;CLEAR BØTTØM LINE
2740  02A4  CA                  DEX
2750  02A5  10 FA               BPL LØØP2
2760  02A7
2770  02A7  A9 00        RET    LDA #0
2780  02A9  85 EB               STA CURFLG      ;TURN ØN CURSER
2790  02AB  60                  RTS
2800  02AC
2810  02AC                     ;-----------------------------------------------
2820  02AC
2830  02AC  20 03 02     TEST   JSR INIT        ;INITIALIZE
2840  02AF  20 ** **     LØØPX  JSR KEY         ;GETKEY
2850  02B2  20 25 02            JSR ØUTCHA
2860  02B5  4C AF 02            JMP LØØPX
2870  02B8
2880  02B8  A5 E1        KEY    LDA KEYFLG
2890  02BA  10 FC               BPL KEY         ;WAIT FØR KEYSTRØKE
2895  02BC  A5 E2               LDA KEYBUF
2900  02BE  8D 20 08            STA DISPLY      ;CHAR TØ LED DISP
2910  02C1  A0 00               LDY #0
2920  02C3  84 E1               STY KEYFLG      ;RESET KEYFLAG
2930  02C5  60                  RTS
2940  02C6
2950  02C6  AD 30 0A     INT3   LDA KEYBRD      ;GET CHARACTER
2960  02C9  85 E2               STA KEYBUF      ;SAVE IN KEY BUFFER
2970  02CB  A9 80               LDA #$80
2980  02CD  85 E1               STA KEYFLG      ;SET KEYFLAG
2990  02CF
3000  02CF  A9 08               LDA #%00001000
3010  02D1  8D 50 0A            STA IRESET
3020  02D4  A9 00               LDA #0
3030  02D6  8D 50 0A            STA IRESET
3040  02D9  68                  PLA
3050  02DA  AA                  TAX             ;RESTØRE X
3060  02DB  68                  PLA             ;RESTØRE A
3070  02DC  40                  RTI
```

```
3080  02DD
3090  02DD  29 0F       HEXASC AND #%00001111
3100  02DF  D8          .      CLD
3110  02E0  C9 0A              CMP #$0A
3120  02E2  90 ** **          BCC SKIP4
3130  02E5  69 06              ADC #6
3140  02E7  18          SKIP4  CLC
3150  02E8  69 30              ADC #$30
3160  02EA  60                 RTS
3170  02EB
3180  02EB  48          BYTASC PHA              ;A=MSD, Y=LSD
3190  02EC  20 DD 02           JSR HEXASC
3200  02EF  A8                 TAY
3210  02F0  68                 PLA
3220  02F1  4A                 LSR A
3230  02F2  4A                 LSR A
3240  02F3  4A                 LSR A
3250  02F4  4A                 LSR A
3260  02F5  20 DD 02           JSR HEXASC
3270  02F8  60                 RTS
3280  02F9
3290  02F9  A9 20       PCLØCK LDA #$20         ;PRINT CLØCK
3300  02FB  A2 10              LDX #$10
3310  02FD  9D 00 17    LØØP5  STA BASE,X
3320  0300  E8                 INX
3330  0301  E0 15              CPX #$15
3340  0303  D0 F8              BNE LØØP5
3350  0305
3360  0305  A4 E4              LDY YTEMP
3370  0307  A6 E3              LDX XTEMP
3380  0309  B9 E7 00           LDA HØUR,Y
3390  030C  20 EB 02           JSR BYTASC
3400  030F  9D 00 17           STA BASE,X
3410  0312  E8                 INX
3420  0313  98                 TYA
3430  0314  9D 00 17           STA BASE,X
3440  0317  A4 E4              LDY YTEMP
3450  0319  C0 03              CPY #3
3460  031B  F0 ** **           BEQ RETN
3470  031E  C8                 INY
3480  031F  E8                 INX
3490  0320  A9 3A              LDA #':
3500  0322  9D 00 17           STA BASE,X
3510  0325  E8                 INX
3520  0326  86 E3              STX XTEMP
3530  0328  84 E4              STY YTEMP
3540  032A  60                 RTS
3550  032B
3560  032B  A9 15       RETN   LDA #$15
3570  032D  85 E3              STA XTEMP
3580  032F  A9 00              LDA #0
3590  0331  85 E4              STA YTEMP
3600  0333  60                 RTS
```

```
3610  0334
3620  0334                ;-----------------------------------------
3630  0334
3635  0334  20 03 02     MØNITR JSR INIT
3640  0337  A9 0D        DISP   LDA #$0D
3650  0339  20 ** **             JSR ØUT
3660  033C  20 1F 0F     LØØP7  JSR GETKEY
3670  033F  C9 10                CMP #$10
3680  0341  B0 ** **             BCS CØNTRL
3690  0344  20 34 0F             JSR SHIFT
3700  0347  A5 F0                LDA BUFFER
3710  0349  8D 20 08             STA DISPLY
3720  034C  20 DD 02             JSR HEXASC
3730  034F  20 ** **             JSR ØUT
3740  0352  4C 3C 03             JMP LØØP7
3750  0355
3760  0355  A9 0D        CØNTRL LDA #$0D
3770  0357  20 ** **             JSR ØUT
3780  035A  A9 0D        NEXT1  LDA #$0D
3790  035C  20 ** **             JSR ØUT
3800  035F  A5 F3                LDA BUFFER+3    ;BEG ADDR MSB
3810  0361  20 ** **             JSR PRTBYT
3820  0364  A5 F2                LDA BUFFER+2    ;BEG ADDR LSB
3830  0366  20 ** **             JSR PRTBYT
3840  0369
3850  0369  A9 20        LØØP6  LDA #$20
3860  036B  20 ** **             JSR ØUT
3870  036E  A0 00                LDY #0
3880  0370  B1 F2                LDA (BUFFER+2),Y
3890  0372  20 ** **             JSR PRTBYT
3900  0375
3910  0375  A5 F2                LDA BUFFER+2    ;LAST ADDR?
3920  0377  C5 F0                CMP BUFFER
3930  0379  D0 ** **             BNE SKIP8
3940  037C  A5 F3                LDA BUFFER+3
3950  037E  C5 F1                CMP BUFFER+1
3960  0380  F0 B5                BEQ DISP
3970  0382
3980  0382  E6 F2        SKIP8  INC BUFFER+2    ;INC ADDR
3990  0384  D0 ** **             BNE SKIP7
4000  0387  E6 F3                INC BUFFER+3
4010  0389
4020  0389  A5 F2        SKIP7  LDA BUFFER+2
4030  038B  29 07                AND #%00000111
4040  038D  F0 CB                BEQ NEXT1
4050  038F  D0 D8                BNE LØØP6
4060  0391
4070  0391  20 EB 02     PRTBYT JSR BYTASC
4080  0394  20 ** **             JSR ØUT
```

```
4090  0397  98              TYA
4100  0398
4110  0398  48        ØUT   PHA
4120  0399  98              TYA
4130  039A  48              PHA             ;Y
4140  039B  A2 02           LDX #2
4150  039D  A0 00           LDY #0
4160  039F  88        DLY   DEY
4170  03A0  D0 FD           BNE DLY
4180  03A2  CA              DEX
4190  03A3  D0 FA           BNE DLY
4200  03A5  68              PLA             ;Y
4210  03A6  A8              TAY
4220  03A7  68              PLA             ;A
4230  03A8  4C 25 02        JMP ØUTCHA
4240  03AB
4250  03AB                  .END
```

ERRØRS = 0000

SYMBØL TABLE

BRKVEC	102C	VECTR0	1032	VECTR1	1035	VECTR2	1038
VECTR3	103B	VECTR4	103E	VECTR5	1041	VECTR6	1044
VECTR7	1047	LØØP	10B5	VIDEØ	0A70	KEYBRD	0A30
BASE	1700	CURSER	00EC	CURFLG	00EB	JIFFY	00EA
SECØND	00E9	MINUTE	00E8	HØUR	00E7	FLASHR	00E6
CLKFLG	00E5	YTEMP	00E4	XTEMP	00E3	BUFFER	00F0
GETKEY	0F1F	SHIFT	0F34	DISPLY	0820	ISTATS	0A60
IRESET	0A50	IMASK	0A40	KEYBUF	00E2	KEYFLG	00E1
INT	1000	ERRØR	104A	INT1	104E	INT2	10AD
INT3	02C6	NEXT	105E	SKIP	1069	SKIP1	1077
SKIP3	1080	SKIP2	1095	STØRE	1092	SKIP5	109D
PCLØCK	02F9	SKIP6	10C8	TEST	02AC	INIT	0203
LØØP4	0207	ØUTCHA	0225	CR	0263	LF	0269
BACKSP	0278	CLEAR	0282	CLKØN	0286	CLKØFF	028B
RET	02A7	SCRØLL	0292	SET	028D	LØØP1	0294
LØØP2	02A1	LØØPX	02AF	KEY	02B8	HEXASC	02DD
SKIP4	02E7	BYTASC	02EB	LØØP5	02FD	RETN	032B
MØNITR	0334	DISP	0337	ØUT	0398	LØØP7	033C
CØNTRL	0355	NEXT1	035A	PRTBYT	0391	LØØP6	0369
SKIP8	0382	SKIP7	0389	DLY	039F		

END ØF ASSEMBLY

```
?S
1200 2D39 0428
?
```