8

LD ACC (HL) & LD ACC BCDE

Moves four bytes into ACC via BCDE.

ENTRY: D635 loads BCDE from memory (HL), then...
D638 loads ACC from registers BCDE.

RETURNS with all registers modified.

LD BCDE ACC & LD BCDE (HL)

Loads the registers with four bytes from memory

ENTRY: D643 loads from ACC (sets HL to point to ACC)
D646 loads from area pointed to by HL (which is moved up 4 in the process).

RETURNS with all registers altered.

LD (HL) ACC & LD (HL) (DE)

Moves four bytes from memory pointed to by DE to memory pointed to by HL.

ENTRY: D64F points DE (source) to ACC, then...
D652 moves 4 bytes from (DE) to (HL)

RETURNS with all bar C modified (B is zero, HL & DE increased by 4).

DEC-BCDE

A 32 bit decrement of the BCDE quad register. Returns with Z if the decrement was carried through to the BC (MSB) pair, otherwise with NZ.

ENTRY: D6C3

RETURNS with AF,BC,DE modified, HL unaltered.

PRINT-INTEGER

Prints the ASCII decimal string corresponding to an unsigned binary integer held in HL. It is a complicated routine which I suspect may also cater for floating point numbers, but I havn't worked it all out yet. It uses a table of three byte powers of ten (10e0..10e5) located at D8A3.

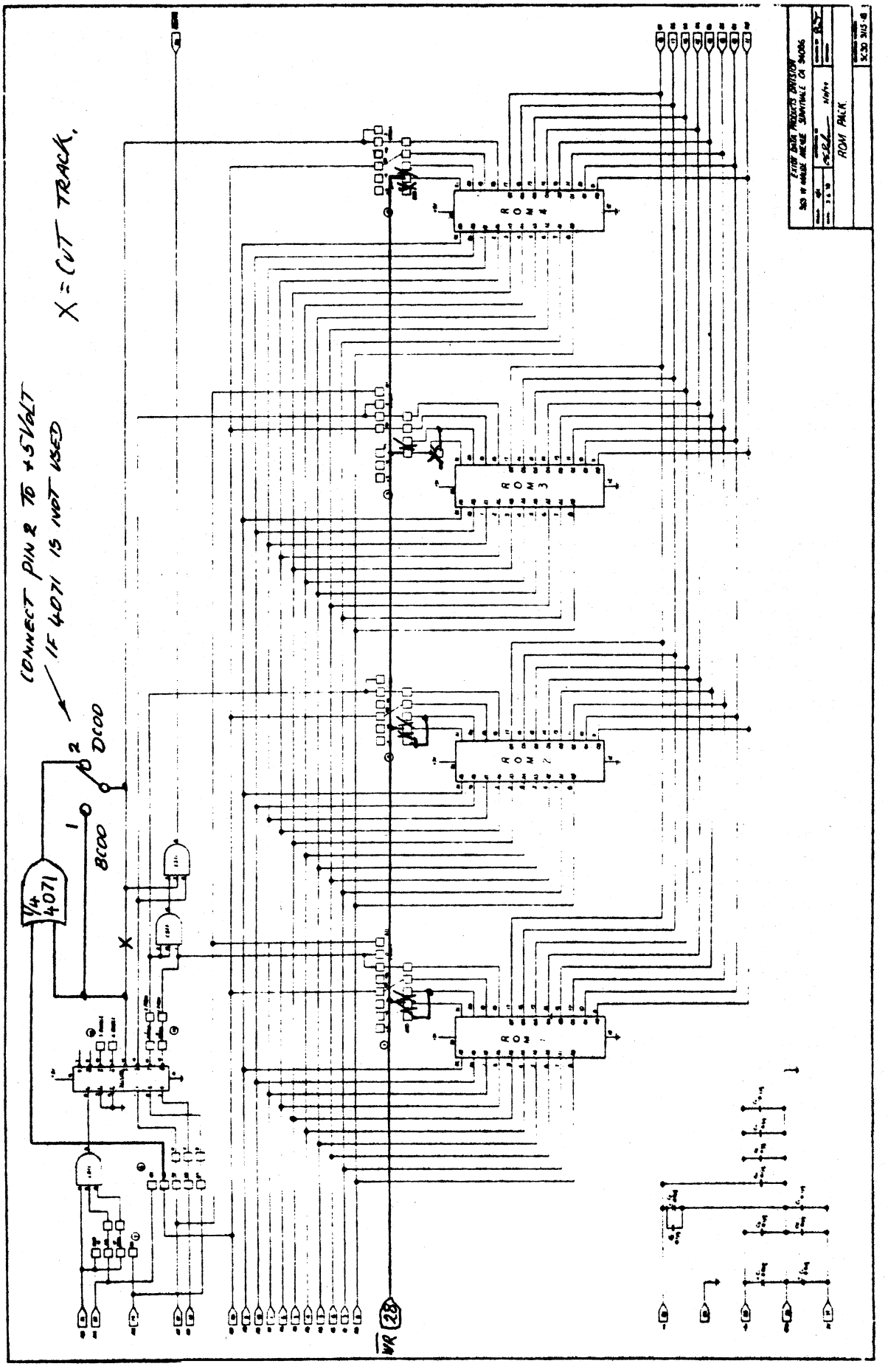
ENTRY: D7BB with 16 bit integer in HL
RETURNS with all registers modified.

KEN GRIMES (QSUG)

BUSINESS APPLICATIONS
***

Mr Maxwell Richardson of 104 Reserve Road, Beaumaris,3193 writes to say that he is interested in conacting other Sorcerer owners in the Melbourne who also use their machine for small business purposes.

Modifications to Exidy Basic ROM Pac to enable use of 6116-2 statac RAMS.



DUTCH SORCERER USERS GROUP...E.S.G.G.

Exidy Sorcerer Gebruikers Groep is the Dutch Sorcerer users group. Since there are probably more Sorcerers in Holland than anywhere else in the world, I thought that the activities of this group would be of interest to our members.

While I was in Amsterdam I called in on Welmoed Jonker who is a member of the five man central committee of ESGG. The group is quite different from others that I have encountered in that it is not a separate entity but rather a subgroup of the Hobby Computer Club, a body which encompasses all personal computerists. When a person joins HCC he registers with a subgroup - eg Apple, TRS-80, etc. There are 760 HCC members who are members of ESGG. HCC publishes a quite professional monthly magazine which covers all machines and is sold at newstands as well as being distributed to all HCC members. The drawback in this arrangement is that very little coverage of any particular microcomputer appears in the magazine. For this reason ESGG has now begun to produce its own newsletter. The circulation is 360. The first two issues are all that I have seen. As one might expect they are of an introductory nature. The content is entirely in Dutch so the newsletter is of limited accessability to most SCUA members. There is a possibility that the more significant technical articles will be translated into English.

ESGG is different in another way from most user groups I have encountered in that they do not hold meetings but rather leave that up to local groups to arrange for themselves. They do however have several areas of co-ordination. One is software. Five tapes of programs are available. These are largely unedited. SCUA has been given unrestricted use of these. The most interesting project of the software group is a version of Pascal for the Sorcerer.
Another area of co-ordination is hardware, and a third is specialised interest groups. We will no doubt be kept informed of the major achievements of ESGG through its newsletter, which we should now be receiving.

Devin Trussell

MICROSOFT BASIC DECODED AND OTHER MYSTERIES

This book, by James Farvour, published by IJG Computer Services, gives a full description of the works of Level II Basic, as implemented on the TRS-80.

TRS-80 code from 0708 to 0A38 is functionally almost identical with Sorcerer code from D39D to D69E. Nearly all the differences are to do with Sorcerer not having integer or double precision variables. Most of the functions, including string functions, are very similar too, but scattered about in different places in the two lots of code. RND is quite different. Other generally matching runs include:

13D2 - 14C9 = D8A3 - D999 153E - 1607 = DA0D - DAD5
1936 - 197C = C2BD - C30B 1A95 - 1C8F " C37E - C529
1C90 - 1C9E = C574 - C582 1CA1 - 1D90 " C62E - C6DC

...and so on, where " means more differences than =, but both include bits that occur in one and not the other.

Bits I still cannot match, nor see what they do include CCA7 - CD43; D785 - D799; and D7F8 - D88F (but D83A = 1307). Bits that astonish me are CC0B (can it ever get here?), D071, and D0BD/C0. Thats all for now!

Andrew Marland (France)

HARDWARE HANDSHAKES WITH THE SORCERER PARRALEL PORT

The Sorcerer Parallel Port has four handshake lines:

A) Output Data Ready (ODR*) and Input Data Accepted (IDA*) are outgoing signals on on pins 3 and 21.

B) Input Data Ready (IDR*) and Output Data Accepted (ODA*) are incoming signals on pins 9 and 2. These also appear as bits 7 & 6 of the Status Port FE (bit 6 is the inverse of ODA*).

Using these signals it should be possible to link two Sorcerers by a parallel cable: connecting ODR* from one machine to IDR* on the other and likewise ODA* to IDA* (see note) ....BUT IT DOESN'T WORK!

This is because of the way the hardware is set up. The ODA* line is connected to the CLEAR pin of the flip-flop which controls the ODA* line. In a typical transmission, when ODA* goes low (indicating acceptance) this resets the ODR* line BUT holds it high so that when the transmitter tries to send a second byte it cannot signal (via ODR* low) to the reciever that it is ready, and the reciever wont let the ODA* line up untill it gets a new ODR*!! So everything hangs up (see figures 1 and 2).

What is needed is a negative pulse on the ODR* line rather than a permanent low.

Likewise one cannot use the hardware handshakes with a Centronics printer, as this also puts a permanent low on the ODA* line, and further it requires a negative strobe on the ODR* line. The Monitor's Centronics driver gets round this by putting a software generated strobe on bit 7 of the output, which is connected to the strobe input of the printer, but this means you cannot send 8 bit graphic data if your printer needs this.

The solution which I have used is to generate suitable pulses with a 74123 Dual-Multivibrator chip. Fig 3 shows the improved signals using the 74123. Fig 4 shows the circuit for a Sorcerer-to-Sorcerer cable, and Fig 5 shows one for an 8 bit Centronics cable. I drilled holes in the plastic backshell of my DB25 plug, mounted the components on the outside and made the connections on the inside - it looks quite neat. In both cases the PARIN and PAROUT routines at E776 and E77F can be used to transfer the data (using bits 6 & 7 of the STATUS port FE for handshaking).

The parallel connection between two Sorcerers is the fastest way of passing data between two machines which use different disk formats. I use a modified version of 'TELNET' (a modem driving program which comes free with the BDS-C compiler) with the I/O redirected to the Parallel Port. This allows a file to be transferred from disk on one machine to disk on the second machine with full handshaking and checksums etc. Both machines spool to and from disk automatically so the file can be larger than memory.

The diagrams on the following page are the figures referred to in the text, and illustrate the electronics involved in the handshaking and other modifications.

Note: '*' indicates Negative logic, i.e. low is true.

KEN GRIMES

Fig 1: The hardware

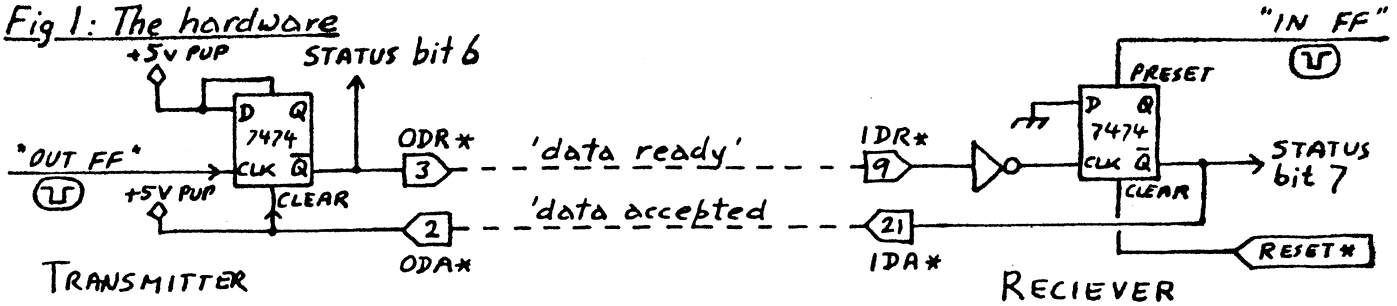


Fig 2: The signals

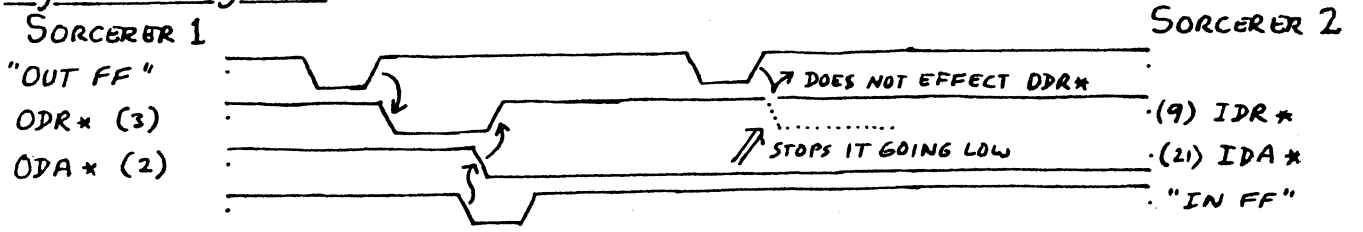


Fig 3: The modified signals with a 74123, as in Fig 4.

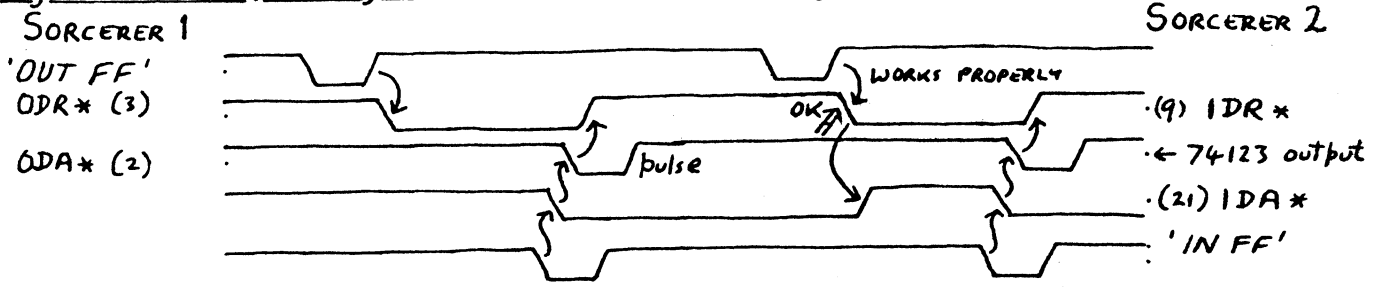


Fig 4: Cable connections for Sorcerer to Sorcerer Link.

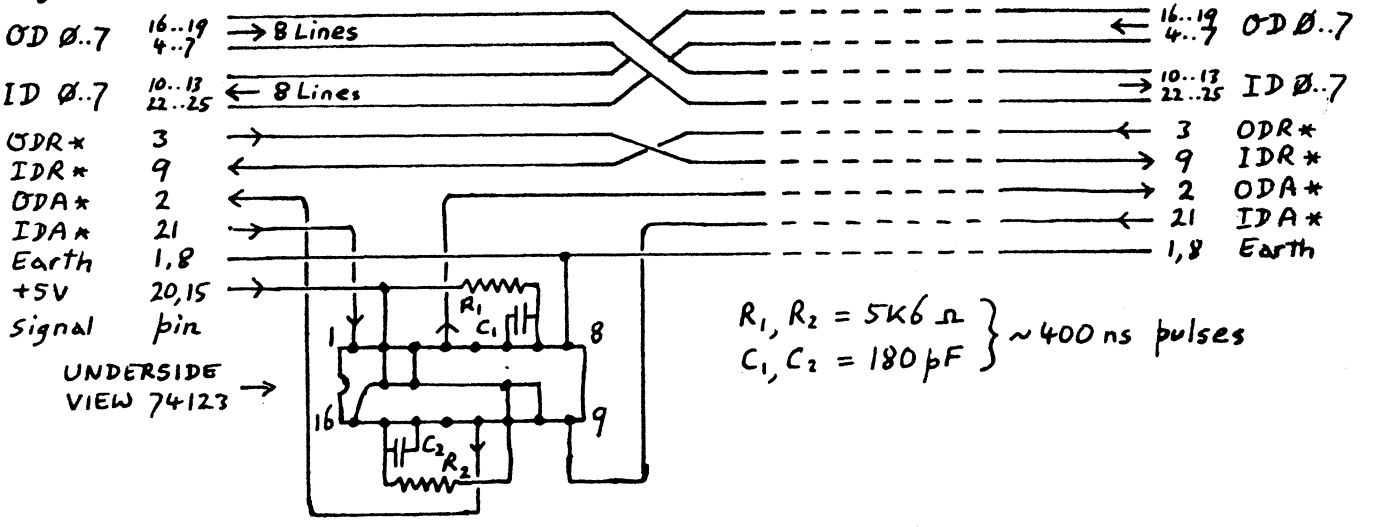
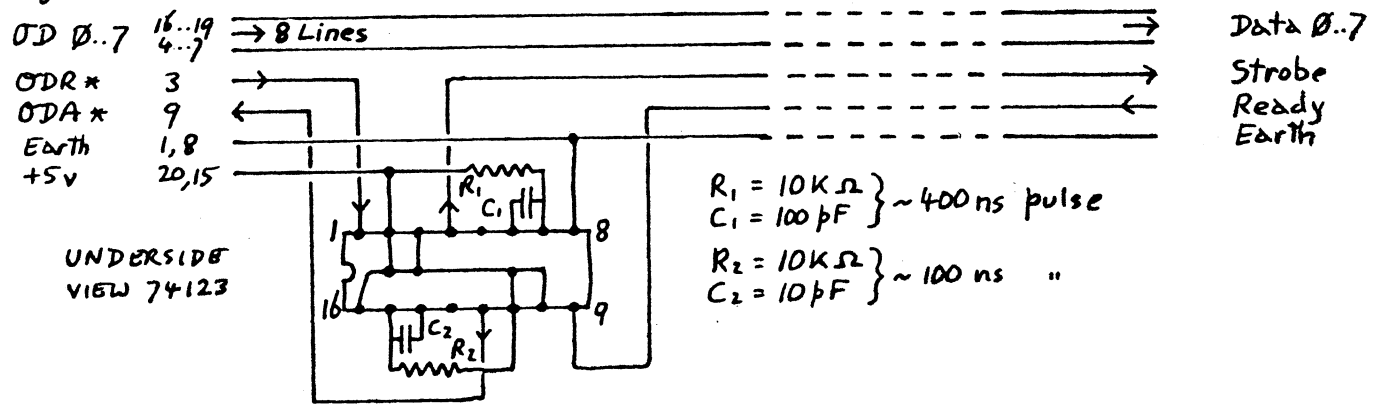
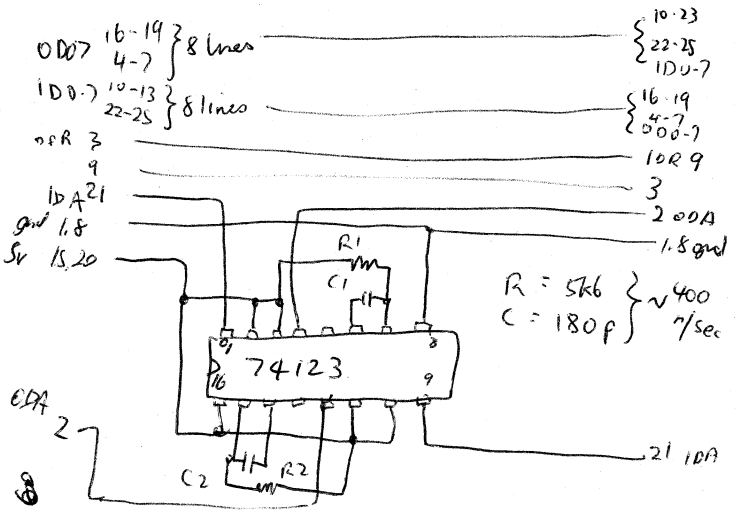


Fig 5: Cable connections for Centronics printer.




ODO7 16-19 4-7 } 8 lines
IDO-7 10-13 22-25 } 8 lines
10-23 22-25 IDO-7
16-19 4-7 ODO-7
IOR 9
3
2 IOA
1.8 gnd
21 IDA
R1
C1
74123
R = 5k6
C = 180p
~400 m/sec
C2
R2
ECA
2

Z80 CAN ADDRESS 128K

One of the least publicised features of the Z80 is its (somewhat restricted) ability to address 128K of memory.

The secret lies in the IN(C),r and OUT(C),r instructions. When these instructions are executed, the contents of the B register appears on the eight most significant address lines, while the contents of the C register appears on the eight least significant lines. Data is then transferred to or from the specified register 'r' and the specified address, depending on whether the instruction was an IN or an OUT.

Normal 'port' operation uses only the lower eight address lines to select any of 256 ports, and of course the IN and OUT(C) instructions can be used in this way. However because all the address lines are specified by the contents of the BC register, these instructions can actually address 65536 ports!

The main difference between an input/output (I/O) memory access as described, and a normal memory access is in memory selection by the operation of special signals from the Z80 itself, in addition to the address lines.

When a normal memory access is made, pin 19 of the Z80, called MREQ (with a line over it to show that it is active LOW) is energised. If an I/O memory access is made, pin 20, called IOREQ (with a line over it) goes low. Both ports and memory are selected by an address , but ports are enabled by the IOREQ signal, while memory is enabled by the MREQ signal.

A full 64K of memory, completely independent of normal memory, can be addressed by the IN and OUT(C) I/O instructions, which then effectively have the form LD r,(BC)' and LD(BC)',r. The tick ' after (BC) indicates separate memory space.

The system has been further developed by Sony who use a variant in their new SMC-70 computer to address the separate 32K video RAM. In a useful twist, Sony exchange the high and low sets of address lines, so that the upper eight bits are addressed by the C register, and the lower eight bits by the B register. This enables the automatic decrement and increment of the B register in the IND, INDR, INI, INIR, OUTD, OTDR, OUTI, and OTIR instructions to be used, offsetting the time lost in I/O instructions, compared with normal memory access.

In order to enable normal port operations to be used as well as separate memory access, the highest bit of the video address bus (bit 7 in the C register) is used as a switch, so that video RAM is selected when it is 1, and normal port I/O when the bit is 0.

Mr Sinclair has another use for the IN(C),r instruction in the ZX-81 computer, in which the address lines scan the keyboard, and the output appears on the data lines.

Full details of the effect of the 'C' group of Z80 I/O instructions may be found in the Zilog publication 'Z80-Assembly Language Programming Manual' (Zilog, Inc. 1977). It is hard to find things in this book, but once found, the explanation and examples are generally clear and more complete than in most publications.

Ian Macmillan

EASTER IS A DEFINED FUNCTION

Newsletter No. 15 (Item 68 in Yearbook #>) discusses the use of the DEFined function. The following program makes extensive use of DEF FN and guarantees to calculate the date of Easter Sunday in any year from 1583 onwards.

Two functions are defined, FNQ which gives the integer quotient and FNR which gives the remainder.

```
100 REM *** EASTER IS A DEFINED FUNCTION ***
110 PRINT CHR$(12)
120 DEF FNQ(Q)=INT(X/Y):DEF FNR(R)=X-FNQ(Q)*Y
130 INPUT"YEAR AD";X
140 IFX<1583THENPRINT"YEAR TO BE >=1583":GOTO 130
150 X1=X:Y=19:A=FNR(R):Y=100:B=FNQ(Q):C=FNR(R):Y=4:X=B
160 D=FNQ(Q):E=FNR(R):X=8*B+13:Y=25:G=FNQ(Q):X=19*A+B-D-G+15
170 Y=30:H=FNR(R):X=A+11*H:Y=319:MU=FNQ(Q):X=C:Y=4:I=FNQ(Q)
180 K=FNR(R):X=2*E+2*I-K-H+MU+32:Y=7:LAM=FNR(R)
190 X=H-MU+LAM+90:Y=25:N=FNQ(Q)
200 X=H-MU+LAM+N+19:Y=32:P=FNR(R):PRINTCHR$(12)
210 PRINT"IN ANNO DOMINI ";X1;"EASTER SUNDAY IS ";P"/"N"/"X1
220 PRINT: FOR T=1 TO 64:PRINT"-";: NEXT: GOTO 130
```

BRENDAN JOYCE

SCREEN EDITOR...in 20 hex bytes

I entered the 'Screen Editor in 26 Bytes' (SCUA Sept.'82), but found that because my FDS moves the MWA down 256 bytes, the one dependent byte prevented it from working. I therefore set out to make it totally relocatable, and shorter too! The result works on the same principle, but is relocatable, stops syntax errors after pressing CLEAR or LINEFEED, and is only 20 hex bytes long!.

```
            VDEP    EQU     0E9F0H
            KEY     EQU     0EB1CH
CD 1C EB            CALL KEY            ;GET INPUT
FE 03               CP      003H        ;CONTROL C?
28 0F               JR      Z,ZAP-$     ;YES: EXIT
FE 0D               CP      00DH        ;RETURN KEY?
28 0B               JR      Z,ZAP-$     ;YES EXIT
FE 1B               CP      01BH        ;CONTROL CODES?
38 0A               JR      C,ZIP-$     ;YES, FIX 'EM
FE 5C               CP      05CH        ;IS IT '\'?
20 03               JR      NZ,ZAP-$    ;NO, EXIT
FD 7E 67            LD      A,(IY+067H);GET CHAR. FOR BUFFER
FE 00       ZAP:    CP      00H         ;SET FLAGS??
C9                  RET                 ;
CD F0 E9    ZIP:    CALL    VDEP        ;GOTO VIDEO DRIVER
3E 00               LD      A,00H       ;KILL CONTROL CODE
18 F6               JR      ZAP-$       ;EXIT
```

ANTHONY HOWE

WANTED TO BUY

1. Micropolis 1053 Mod.II dual disc drives, and S-100 Expansion unit. Bob Reynolds, 36 Dredge Ave, Moorebank, N.S.W. 2170

2. Sorcerer MK.II, 32-48K, stringy floppy (c/w monitor chips and controller), appropriate operating system, with or without video monitor, cassette, documentation, etc. Ian Thomas, 25 Staley St, Elsternwick, 3185

THEORY OF OPERATION, LOGIC BOARD

Video Clock Generator

The 12.638 MHz crystal,22D-2 and 22D-4 form an oscillator with 22D-6 as a buffer. The flip-flop 22C divides the clock signal in half, providing CLK 6 (6 MHz approximately) and its inversion 5CLK6. The signal 5CLK12 which comes from 22D-6 is further divided by flip flops 22B-5 and 22B-9 to provide the signals CLOCK IN and Ø1 respectively.

Horizontal Sync Generator

22A, 21A and 18A provide the horizontal sync and blanking generation. 22A and 21A form a binary up-counter; the starting count is determined by the state of 18A-6. E1 through E256 are the horizontal scan element counts. 17A-9 shuts off the video during horizontal blanking; 17A-6 shuts off the video when the CPU accesses the screen. 20A is part of the video counter buffer.

Vertical Sync Generator

21B, 20B and 17A-9 work similarly to 22A, 21A and 18A; the start of count depends on the state of 18A-7. L1, L2 and L4 are the line counts of the 8 x 8 character matrices; L8 through L256 are the line counts for the text lines on screen. 16A and 18B-5 latch the vertical sync. 17B is part of the video count buffer. 50 Hz/60Hz operation is selected by multiplexes 13A and 19B.

CPU Clock Generator

The signal CLOCK IN is fed into a divide by two flip-flop 9A and then inverted by 8D then fed into the proccessor.

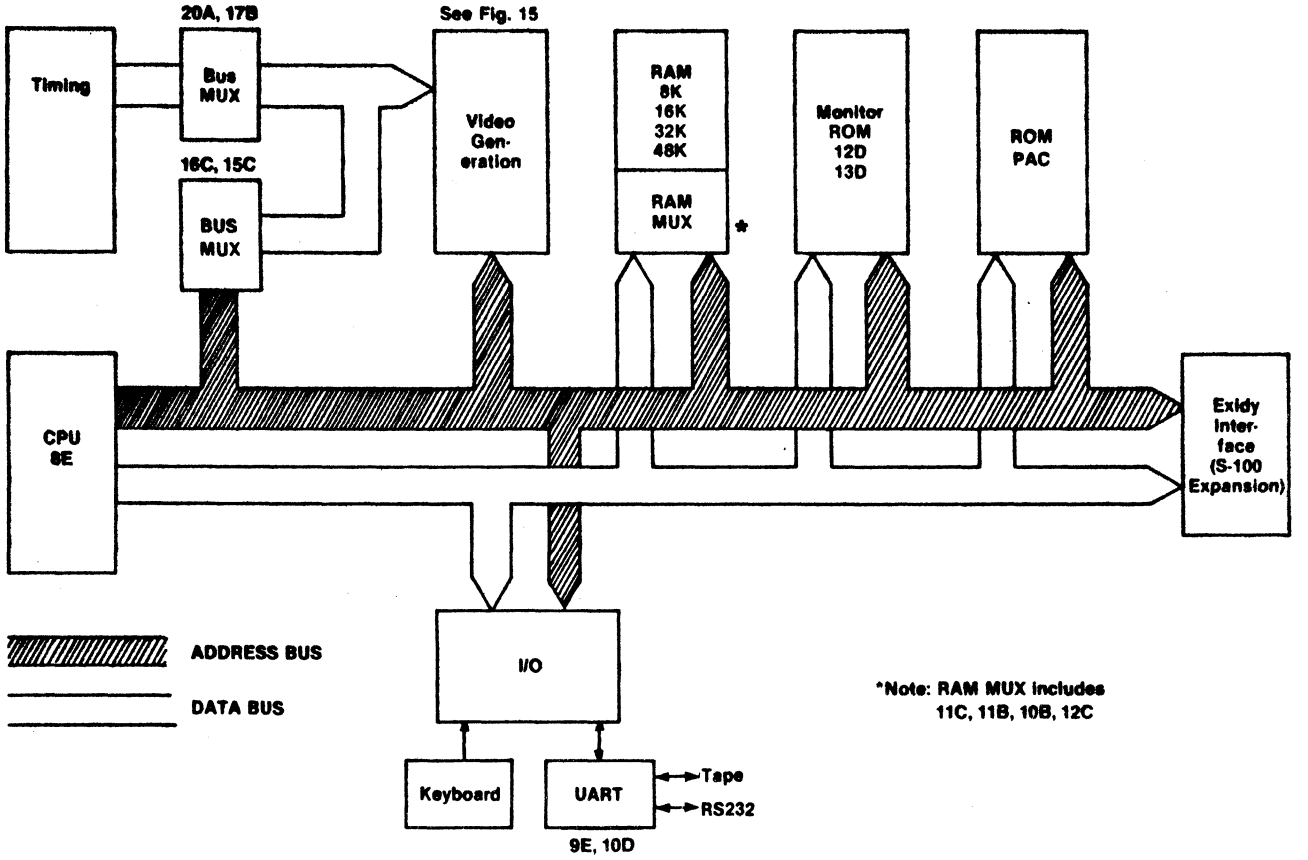


Figure 9 Logic Board Block Diagram

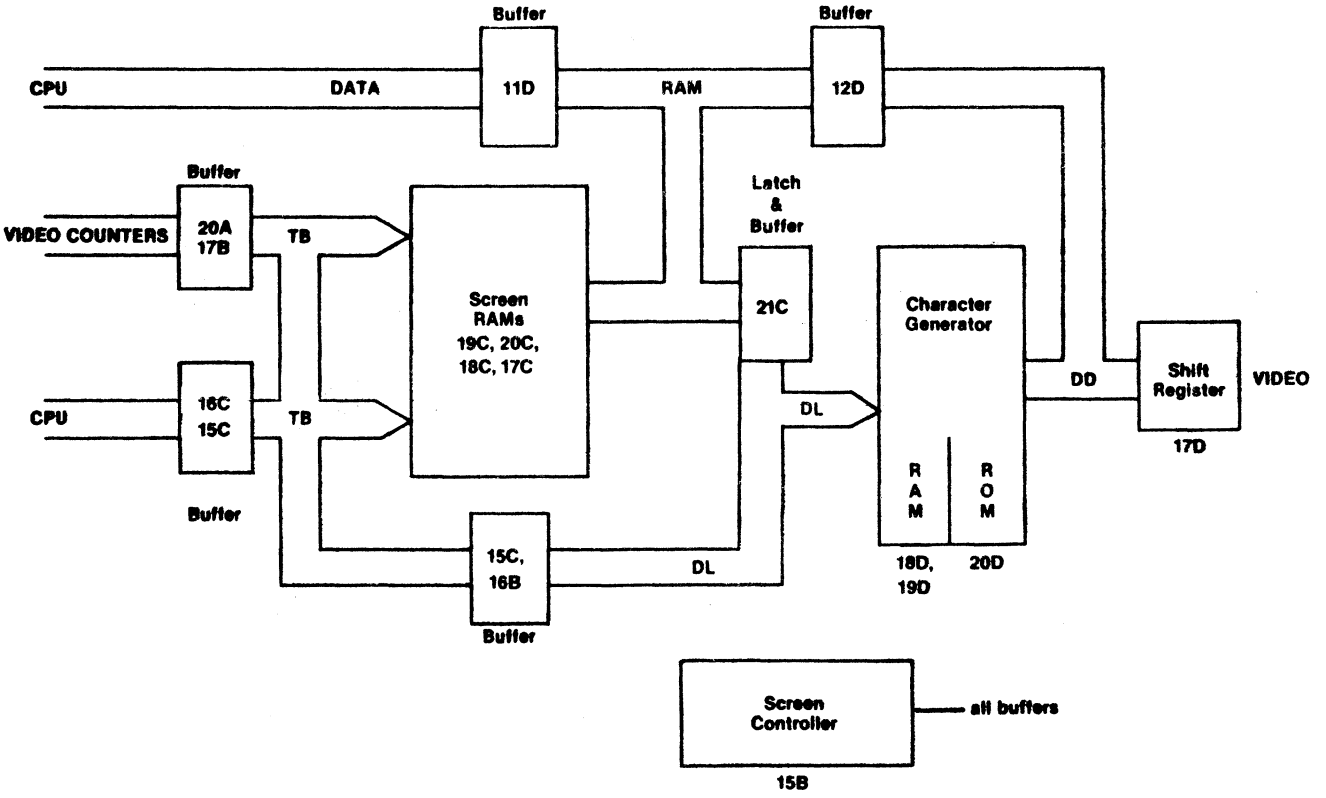


Figure 10 Video Generator Block Diagram

Video Interface

The pre-programmed 6331 PROM 15B is the heart of the screen controller. 15B controls all the buffers on the page, as follows:

- When the CPU is not addressing the screen RAMs, 11D and 12D go high-impedance.
- When the CPU is writing on the screen, the counter buffer (15C and 16B) and the data bus buffer (16C and 15C) go high-impedance.
- When reading the character generator ROM, or reading and writing character generator RAM, an address comes from the CPU through the TB and DL buses; both data bus buffers are on.

CPU Circuitry

The reset circuitry is based on a four-bit counter, 5D. The CPU BUSAK signal is buffered by 10H and fed to all the buffers on the page. At BUSAK, they all go high-impedance; this allows a Direct Memory Access (DMA) into the Sorcerer.

The CPU is a Zilog Z80; refer to the Zilog Z80 CPU Technical Manual, Zilog part number 03-0029-01.

ROM Circuitry

At restart, the Z80 begins executing code at address 0000; however, the Power-On Monitor resides near the top of the memory map. The ROM decode shifts the CPU into the Monitor at reset. The reset signal pulls 12C-5 low; this disables the RAMs and unconditionally enables the ROMs. The first instruction in the Monitor is Jump E062H. For any address from E000H to FFFFH, 10A-6 gives the UP8K signal; address E062H sends 9D-15 low, which sets the latching flip-flop 12C, and enables the bottom RAM.

> NOTE
>
> The RAM is always enabled, except during the first three instruction fetches after a reset.

I/O and UART Circuitry

6D and 7D handle I/O requests. 7D-8 gives the I/O request signal, and enables both halves of 6D. The RD and WR signals are the other enables for the first and second halves of 6D,

respectively. The following I/O port designations come into 6D and A0 and A1:

Signal	Port	A1	A0
UART data (serial interface)	FCH	0	0
UART status	FDH	0	1
Sorcerer housekeeping input	FEH	1	0
Sorcerer user output (parallel interface)	FFH	1	1

The Sorcerer input port and the parallel input port are 3-state buffered by 1D and 8H, respectively. The enable signals for the buffers come from 6D. The Sorcerer output port and user output port are 8-bit latched, by 2D and 9H, respectively. Figure 16 shows the parallel output port timing signals.

The UART is a General Instruments AY-3-1015; refer to the manufacturer's technical publications. 10D buffers the UART output.

Interfaces

Cassette/UART Interface: This circuit communicates with the tape interface board.

Cassette Motor Drivers: 9F and Q2 form a Darlington pair; the reversed diode CR2 is turnoff protection for Q2. 9H, Q3 and CR3 are exactly similar.

Power Supply: The transformer has two primary windings in parallel; to convert the power supply to 220 V input, disconnect the windings and reconnect in series (see Hardware Modifications, 110 V 60 Hz to 220 V 50 Hz).

Exidy Bus Drivers: The CPU Control, Address, and Data signals are bi-directionally buffered by 1H to 5H. The bi-directional buffering allows DMA.

S-100 Control: When anything is happening on the logic-board or the tape interface board, 1F receives an input. This disconnects the S-100 Expansion Unit.

THEORY OF OPERATION, TAPE INTERFACE

General

The tape interface translates between the UART data format (non-return to zero) and the tape cassette format (frequency shift). The frequency-shift format uses a high frequency for logic 1, and a low frequency for logic 0 (see Figure 13). At 1200 baud, a logic 1 is 1 cycle of 1200 Hz and a logic 0 is 1/2 cycle of 600 Hz; at 300 baud, a logic 1 is 8 cycles of 2400 Hz and a logic 0 is 4 cycles of 1200 Hz. In both cases, the time required to transmit a logic 1 is the same as the time to transmit a logic 0.

The interface also adjusts the output signal levels to approximately 250 mV p-p for the tape recorder AUX input and approximately 50 mV p-p for the tape MIC input. A jumper at board location 12H allows a 4 V p-p signal instead, for digital recorders (jumper points A, C).

Manchester Encoder

Flip-flop 16F synchronizes the input data with the 1200 Hz clock, triggering on the positive edge of the clock pulse. The signal is inverted in passing through 16F, but is otherwise unchanged. 13F6 and 13F7 frequency encode the data, giving a high frequency for logic 1 and a low frequency for logic 0.

Level Adjustor/Pulse Shaper

C78 rounds the corners of the square waves (audio recorders don't like square waves); C79, C80, C81 and C82 are DC isolators. The output voltage jumper (location 12H) is part of this circuit.

Clock Selector

This circuit selects a clock rate for the manchester encoder, dependent on the selected baud rate.

Frequency Divider (x 1/55)

13H and 14H form a six stage binary up-counter, which counts from 9 to 64 and then sends a lock pulse from 14H-13 and a carry from 14H-15 to 16E-9; the carry starts the count cycle.

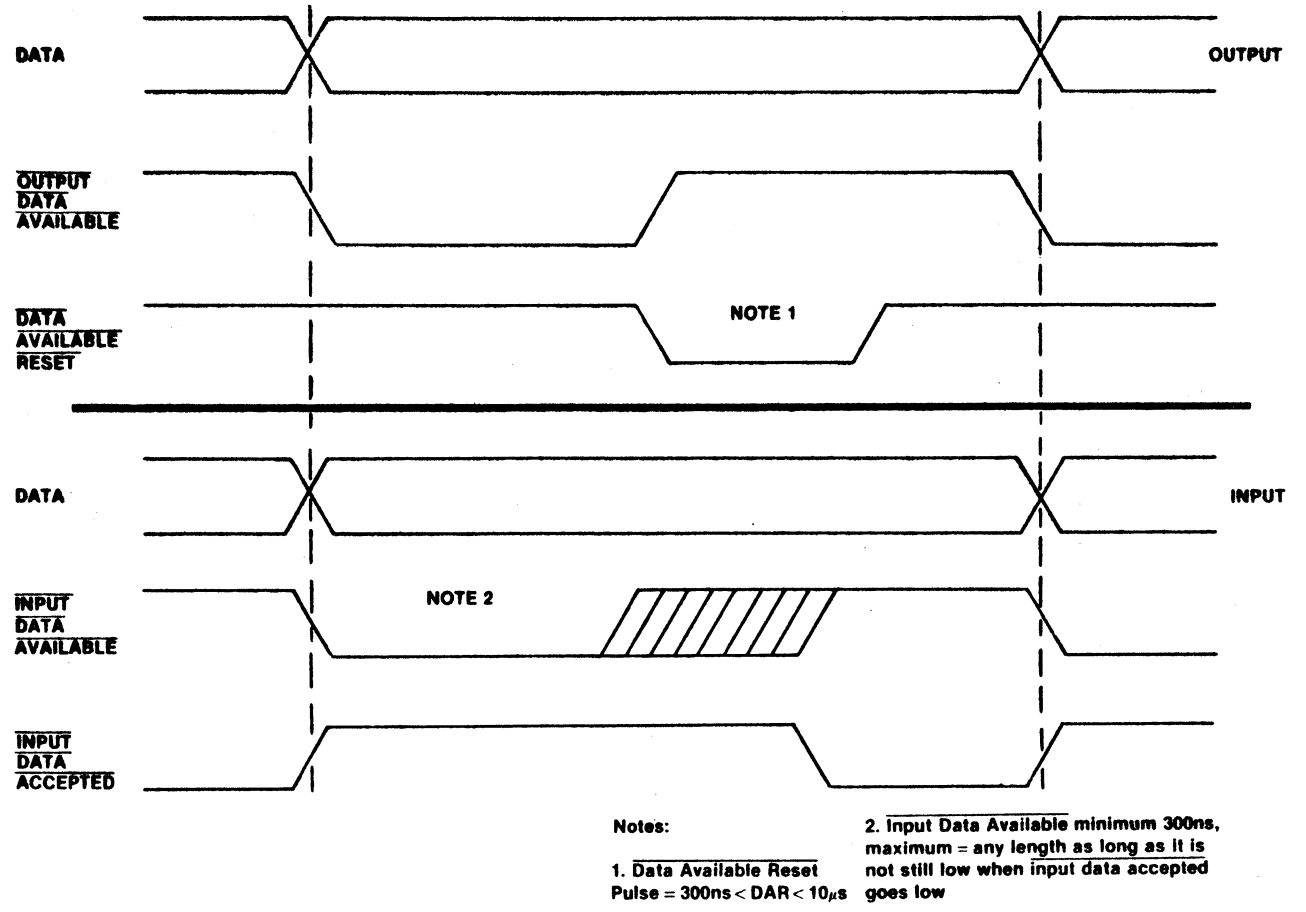


Figure 11 Parallel Port Timing System

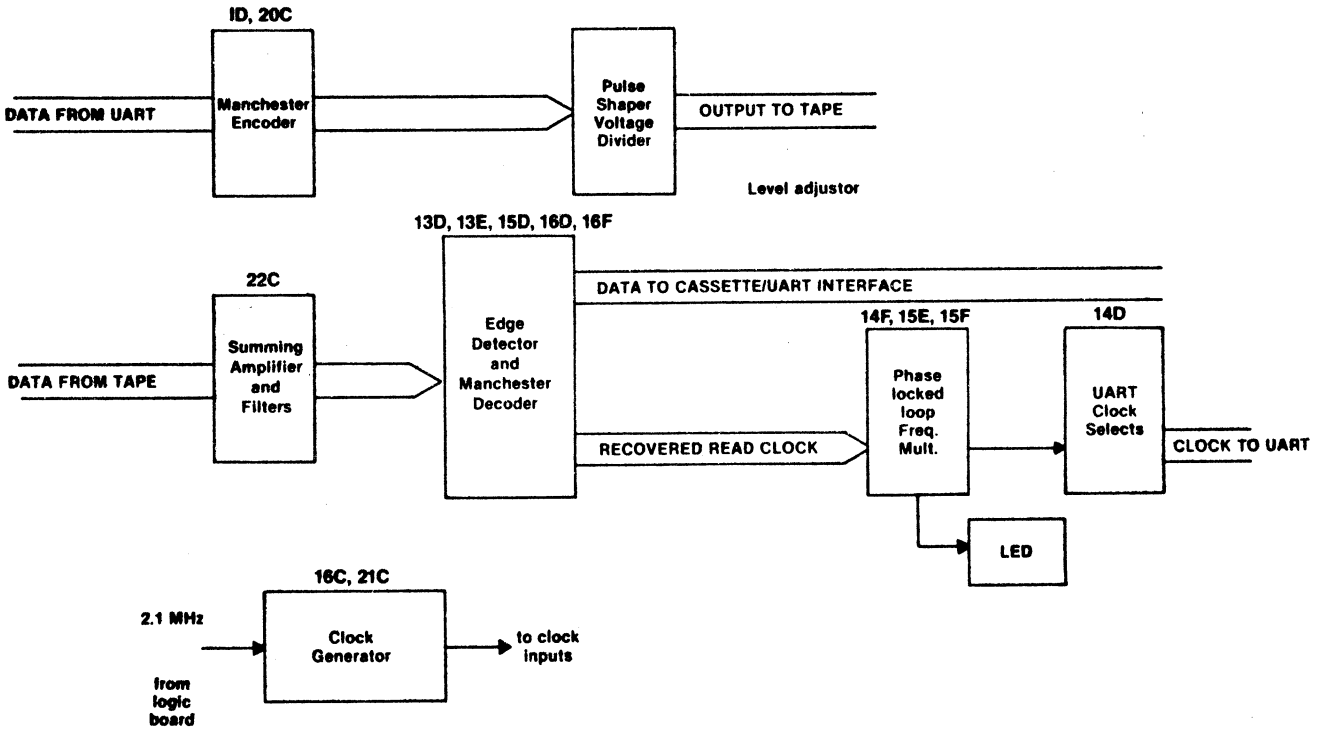


Figure 12 Tape Interface Block Diagram

Clock Pulse Generator

This circuit is simply a five stage binary up-counter.

Summing Amplifier/Low Pass Filter

This circuit allows input from two tape units, by preventing the non-operating unit from disturbing the signal of the operating unit. The output is 62% of the input signal, partially filtered. C77 is a DC isolater; C60 and the parallel 130K ohm resistor (R35) form a 3700 Hz low-pass filter. The 3.3 Megohm resistor (R34) biases 12F-7 to the center of its range (+4.5 V to -5 V) so that the positive and negative portions of the wave are equally clipped.

Second-Order Linear-Phase High Pass Filter

This circuit removes flutter. C57, C58, R31and R33 determine the high-pass cutoff at 300 Hz.

3400 Hz Low-Pass Filter

This filter reduces high-frequency noise and prevents op-amp oscillation. C62 is a DC isolator.

Comparator/Low-Pass Filter

This filter converts the analog input signal to digital pulses. C65 and R39 reduce frequencies above 480 Hz. Any input signal above 0 V is converted to approximately +5 V at output; any input signal below 0 V is converted to approximately 0 V at output.

Bi-Directional Edge Detector

This circuit provides a positive-going pulse for each transition at the input. The Schmidt-trigger inverter 13D reduces signal noise and squares the pulse edges. 13D-4, 13D-6, R20, and C41 form a delay line, with a delay on the order of a few microseconds. 13E is an exclusive-OR gate which provides a high-going signal during the time the input signal has <u>not</u> passed through the delay line.

Manchester Decoder

13E-6 is used as an inverter, to force CASSREAD high when RS232 is chosen. 16D-5 is set by the pulses from the bi-directional edge detector and produces a recovered clock signal. 15D is a binary up-counter which functions as one-shot to reset 16D-1.

15D counts from 4 to 15 before firing--this determines a critical period during which 16D-9 is either set or reset, and thus determines a maximum pulse width. Pulses narrower than this max width are considered 1s; wider pulses are 0s. 16D synchronizes the 16D-9 signal with 16D-5's recovered cl signal.

Frequency Multiplier (x 8 or x 16)

14F is a phase-locked loop and 15E is a frequency-divider, 14F has voltage controlled oscillator, which is adjusted so that the signal out of 15E equals the recovered clock (that is, the signal out of 14F-4 is adjusted so that the signal into 14F-14 equals the signal in at 14F-3). 14F locks onto the recovered clock and provides a clock signal for the UART, at 16 times the data rate. 15F selects the working frequencies, depending on the chosen baud rate. C66 and its 150K resistor. form a low-pass filtered feedback loop for the voltage controlled oscillator; C67, VR1 (location 15H) and the 68Kand 100K (pin 1) resistors set the center frequency and frequency range.

Sync Indicator

13E9 acts as a buffer. 14F-1 is high if the phase-locked loop is in sync. C84 filters out small pulses.

UART Clock Selector

This circuit selects working clock signals based on choice of RS232 or cassette, and baud rate.

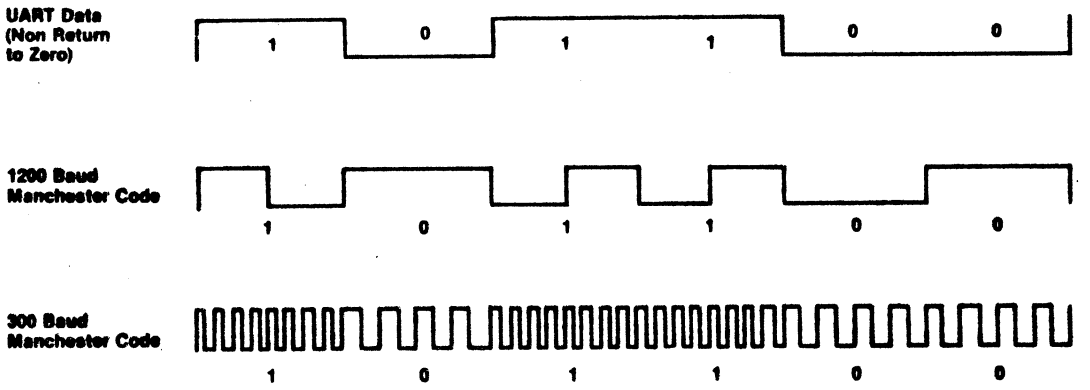


Figure 13 UART and Cassette Data Formats