CME Bulletin 9781

DIGITAL INTERFACING

FOR THE TYPE 7L18 SPECTRUM ANALYZER

The internal data bus of the Tektronix Type 7L18 Spectrum Analyzer plug-in unit may be monitored externally to read out the significant measurement parameters, and the data from display memories A and B. It is also possible to interpose external commands for the control of Time/div, Span/div, Resolution bandwidth, Phase-lock and Band-select functions.

However, external control of Center frequency (a direct analog function not mediated by the 7L18's internal microprocessor) cannot easily be implemented externally, and such an attempt is not recommended.

The 7L18's internal microprocessor operates from internal ROM programs and data tables to read the front-panel control settings and drive the appropriate control circuits, to compute and implement the "auto" functions and to read the center-frequency control voltages and convert this information to frequency values for readout and display. There is no provision for "handshaking" with an external device for output of this information, however: any external monitor must catch this data "on the fly".

There are two "dump" commands which will be recognized by the internal processor to provide a burst of specific data on the 7L18's 4-bit I/O bus.

The first of these, a "dump front panel" command, will trigger a burst of ten 4-bit words describing Center frequency, Vertical mode, Reference level, Span/div and Resolution bandwidth.

The second command will provide a dump of both display memories as a sequence of 2048 4-bit words, issued at the rate of one every 80 (approx.) microseconds. A pair of 4-bit words forms the 8-bit vertical amplitude value (most-significant bits are sent first, then the least-significant bits) for each of the 512 horizontal locations in each display.

In the 7L18, the conversion and storage of data from the analyzer to the display memories takes place via dedicated hardware circuits rather than via the processor. Therefore, it is not possible via the accessible

I/O lines to change the content of the 7L18 display memories -- e.g., for the purpose of displaying externally saved or generated waveforms.

Although it would be possible to construct a dedicated hardware interface circuit to read, assemble and output the panel and display data and to introduce external control commands, the design and layout of such a system in "random" logic is burdensome and inflexible, and would require an inordinate amount of ECB space and power.

Instead, a microprocessor-based interface using just 19 IC packages on an ECB of about 6 x 10 inches (as small as 3.5 x 7" if wire-wrap or point-to-point soldered handwiring is used) is suggested, providing the basic capability of reading and interpreting all internal data transfers on the 7L18 I/O data bus, and communicating with an external controller via a parallel 8-bit duplex or bidirectional data bus with handshake facility. The interface requires +5V at about 1A, easily obtainable from low-cost regulated power-supply assemblies. The interface uses the Motorola MC6802 microprocessor (with on-chip clock and scratchpad RAM), MC6820 PIA's, an Intel 2758 EPROM (1/2 2716. A full 2716 may be used with minor wiring changes), and standard memory and SSI/MSI logic chips. Total parts cost, including power supply, is under $250 at the 1-each distributor price level.

The resulting assembly is then limited only by the EPROM firmware in the number of functions which can be implemented with respect to retrieving and processing available information from the 7L18. Additional hardware would be required, however, to implement a full GPIB (IEE-488) external interface.

The following paragraphs discuss some of the basic constraints imposed by the 7L18, and the approach used in the microprocessor-based interface to deal with them. Whether the user chooses to follow this design or pursue an independent course, the following note should be heeded:

NOTE

(a) The Type 7L18 is not sold with interfacing hardware (connectors and cables) installed, nor are the internal Digital Storage Interface circuits 100% tested for external interfacing timing and logic compatibility. Particularly with 7L18's sold before September 1978, some internal modifications may be required for satisfactory interfacing.

(b) This interfacing information and prefabricated cable assemblies have been offered at no charge to selected 7L18 purchasers who have expressed both a need for the information and confidence in their own electronic design and fabrication capabilities to undertake their own interface construction. No interface parts, boards or EPROM's are available from Tektronix, Inc. The suggested interface has been breadboarded just once on perf-board to demonstrate technical feasibility. The inordinate cost of equipping manufacturing, applications

and service personnel throughout the world with computer hardware and software and training to support such a device precludes any offering of it for sale as a kit, accessory or product by Tektronix, Inc.

(c) No external interface facilities will be supported by Tektronix Service centers. It is recommended that any 7L18 equipped with the front-panel Lemo interconnect have the cable and connector removed before submitting the instrument to a Tektronix Service center for calibration or repair. Otherwise the cable and connector assembly may be removed and discarded by the Service center. Their operations are strictly limited to "as shipped" configurations of Tektronix products.

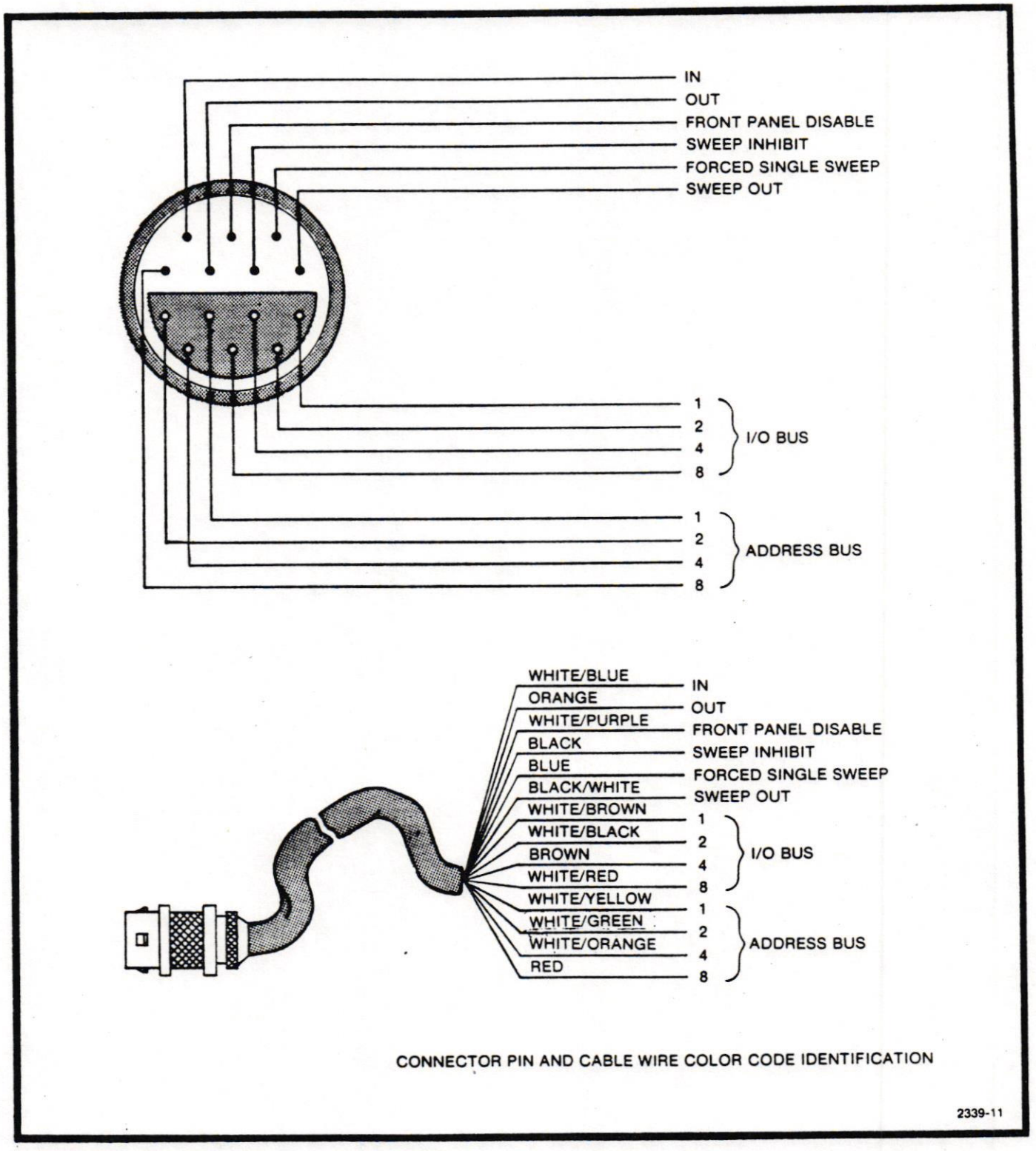


Fig. 1. Option connector pin-out identification and cable wire color code.

GENERAL INTERFACING INFORMATION

1. Access to the 7L18 Data Bus.

The 7L18 microprocessor I/O data bus is accessible at J4225 and J4230 on the 7L18 mother board (See Service Manual schematic ◇28◇ *). By use of "harmonica" connectors and two 7-wire ribbon-cables, this bus may be brought to a front-panel connector in the OPTION position. The panel there is punched to accept a Lemo #RA-C2314 panel-mount 14-contact polarized connector (7 male, 7 female contacts). A mating Lemo #F-C2314 cable-end connector then may be used with a 14-conductor shielded umbilical cable to interconnect with the interface assembly.

> Note: Because the 7L18's I/O address and data lines are un-buffered lines with passive pull-up, the use of a long umbilical will introduce crosstalk and signal-risetime degradation, interfering not only with external operations, but with the internal operation of the 7L18 itself. Two meters should be considered the maximum usable cable length, and one meter or less preferable.

In addition to the I/O bus (4 address lines, 4 data lines, an IN stobe and an OUT strobe), a sweep-ramp output, two sweep-control lines and a FRONT PANEL INHIBIT control line are also available at J4225-J4230, and should be brought to the front-panel connector. The sweep-control and ramp output lines are not used in this interface implementation, however, and may be omitted from the umbilical.

2. Data Timing.

The data lines available are those of the 32 "I/O ports" of the 7L18's microprocessor. The CPU controls the four port-address lines at all times. Sixteen ports are designated INPUT ports, and 16 are designated OUTPUT ports having the same 16 port addresses. Separate IN and OUT lines strobe the data for input or output operations respectively, creating in effect 32 "ports".

For data output operation, the port address lines are set 20 to 120 µs before the data lines are set. Approximately 0.2 µs after the data are valid, a positive-going OUT strobe of 1 to 2 µs duration is generated by the CPU. The trailing edge of the OUT pulse releases the data lines to go high again: valid data are on the line only for 1 to 2 µs (and therefore

*Early interim service manual schematics contain an error: the address bus "2" bit is at pin 1 of J4230; the "4" bit is at pin 7 of J4225, rather than as shown. Later printings show the correct connections.

must be latched externally for recognition by the interface processor). At a time between 2Ø and 32Ø µs after the OUT pulse, the address bus may change.

For data input operations, the address is set up 2Ø to 1ØØ µs before the IN pulse. However, the data are in this case strobed onto the data bus by the IN pulse, and so are not valid until Ø.2 to Ø.5 µs after the leading edge of the IN pulse. The IN pulse is 1 to 2 µs in duration; the data-valid time is therefore only Ø.5 to 1.5 µs. The address lines may change at any time from 32 to 12Ø µs after an IN pulse.

If a single external data latch is used to allow the interface processor time to recognize the I/O pulse and read the data (data latch triggered by the logical OR of the IN and OUT pulses), timing limitations will be approximately as follows:

(a) Time to recognize an address and start to watch for an IN or OUT pulse, 2Ø µs.

(b) Time to read and process data after an IN or OUT pulse, approximately 44 µs.

(c) If data are latched and addresses are not, time to read address and data, approximately 32 µs after an IN pulse.

Some typical timing characteristics are shown in Fig. 2 below.

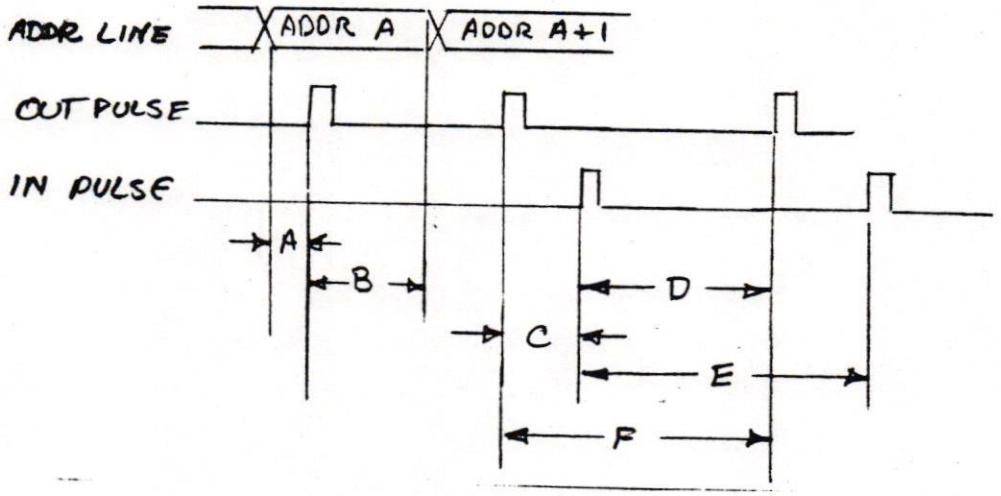


FIGURE 2. 7L18 EVENT TIMING

A: Address change to IN or OUT pulse 2Ø µs min.
B: OUT or IN pulse to address change 3Ø µs min.
C: OUT to IN pulse 44 µs min.)
D: IN to OUT pulse 1ØØ µs min,) Do happen together!
E: IN to IN 15Ø µs min except 8Ø µs in data dump.
F: OUT to OUT 32Ø µs min. except 8Ø µs in data dump.

2.1 Data Sequencing.

Except under specific circumstances, an external device cannot anticipate the exact sequence of addresses and IN/OUT strobes which will occur. In the normal course of operation, the 7L18 processor may sequence through several consecutive addresses, then stop at one address, issue an OUT command, and read in a sequence of words from the same or a different single address. For this reason, much of the data available from the IN and OUT ports is not externally usable unless the external processor can follow each step of the internal processor's program, reading its data and anticipating its decisions. (Note that the externally accessible bus is the 7L18's I/O bus, not the internal processor's address and data buses.)

2.1.1 IN Data.

In order to determine the Time/div setting and to perform substitution of external commands for the five externally-controllable functions, it's necessary to be able to read Port Ø IN through at least Port 4 IN.

Data from Ports Ø through 4 and 8 IN are usually read in consecutive order on 6 consecutive IN pulses: Ø, 1, 2, 3, 4, 8. After reading Port 8 IN, the sequence may be (hex notation): C, D, E, A, F; or C, D, E, A, A, B, B, B, B, B, F. In the case of the longer sequence, OUT pulses (A OUT) are interspersed with the B IN pulses as the internal processor commands the DVM to step through its digits.

Because the external interface processor may jump into the I/O sequence at any point without being able to anticipate the length of the sequence, the simplest algorithm accepts all IN data, storing it in 16 consecutive memory locations according to the address which is on the I/O bus at the time each IN pulse occurs. In the normal operating mode (not during a data dump) there is about 150 µs between consecutive IN pulses -- enough time in which to compute a location and store the data. If a large (redundant) number of IN reads is performed, at least one reading for each port number is guaranteed.

However, this scheme does not provide proper data for Ports 1Ø and 11 (A and B), which may have multi-word data, of which only one word will be saved. Fortunately, the Port 1Ø and Port 11 data are not externally significant. Port 1Ø indicates DVM status; Port 11 outputs DVM voltage readings which will be translated by the internal processor into center-frequency values. The translated information is available via the Front-Panel Data Dump operation.

Port Ø-4 data coding is shown in Appendix B. In the suggested interface firmware, the data are re-grouped to provide just one parameter per word, and are transferred to and from the controller in that form; then re-translated to 7L18 form when used to control front-panel functions. This pre-processing reduces the required bit-manipulation in the controller, and thus facilitates use of high-level language programming of the controller.

2.2 Dump Operations.

The data-dump operations triggered by an external command introduced at Port 15 IN generate a defined sequence of data on the bus.

2.2.1 Panel Dump.

When a Panel Dump command (bit pattern 1xxØ) is recognized at Port 15 IN, the 7L18 processor will respond with ten consecutive 4-bit words via Port 15 OUT, at approximately 8Ø µs intervals. The address lines do not change after the first 15 OUT strobe, so with latched data, the interface processor has about 8Ø µs to read and store each output data word and get back to watch for the next OUT strobe. The available time allows address verification for each word, and any other housekeeping required.

Interpretation of the Panel Dump data is shown in Appendix A.

Typical timing of the Panel Dump sequence is shown in Fig. 3, below.

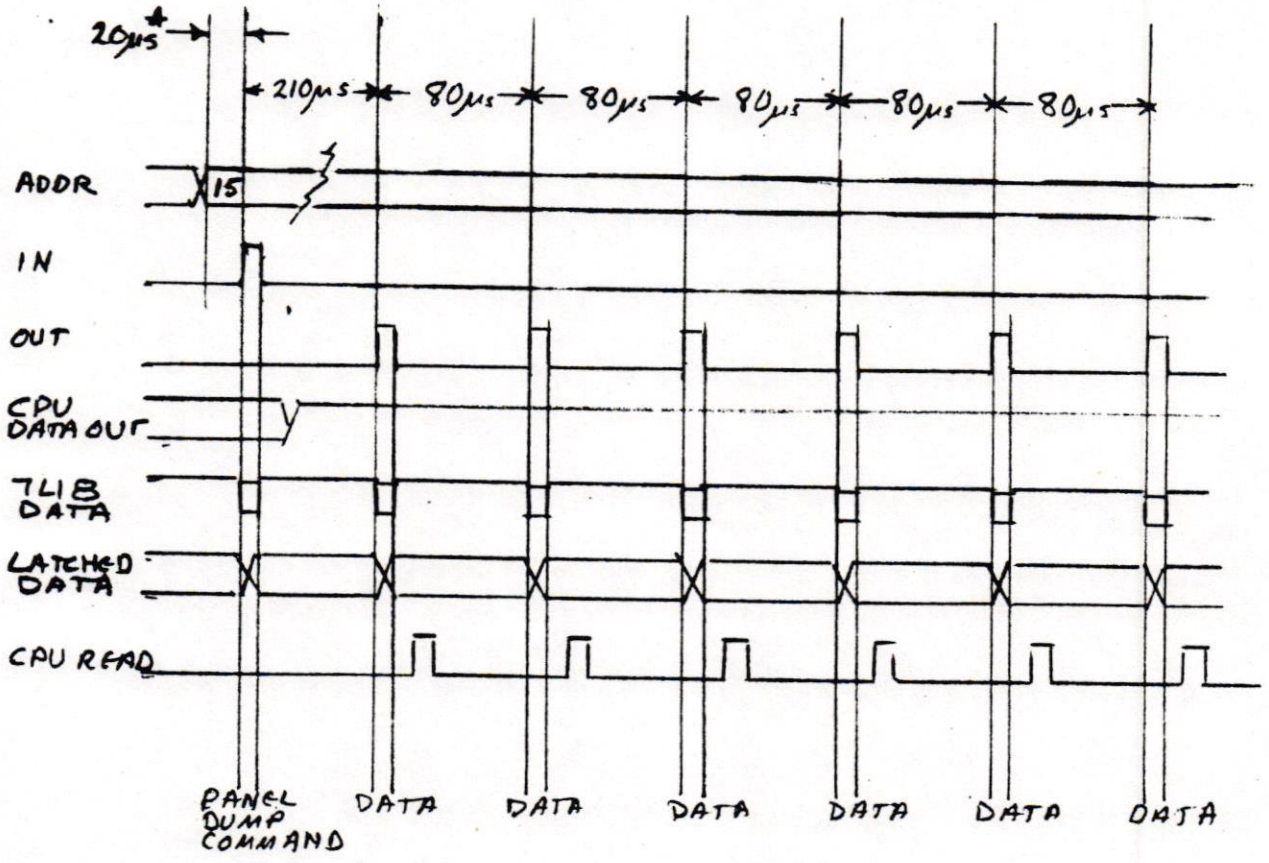


FIGURE 3. TIMING SEQUENCE -- PANEL DUMP

*Planning value (actual may be longer). Others are observed values.

2.2.2 Data Dump.

When a Data Dump command (bit pattern Øxx1) is recognized by the 7L18 processor at Port 15 IN, the processor will respond after about 21Ø µs by switching the address lines to Port 8, and about 2Ø µs later will issue an OUT pulse. After another 5Ø µs there will be a status word strobed at Port 8 IN, followed after about 12Ø µs by the 2Ø48-word display-data stream, starting with the lowest location in Display Memory B (left-hand side of the B display) and continuing through Display Memory A, all strobed at Port 8 IN at about 8Ø µs intervals. When the 1Ø24th word of Display Memory A has been strobed, the 7L18 processor switches the address lines to Port 15 and the OUT pulse strobes a status word (bit pattern ØØØ1 for a successful dump or ØØ1Ø for an abort or "no request").

The 16Ø to 17Ø µs taken in the data dump to output two 4-bit words allows the interface processor time to assemble the two words into one byte before storing in the interface memory, as well as check address validity and provide selective storage of A or B data only, if the interface processor is run at full speed.

If the 7L18 is not in the Store mode, the status word at Port 8 IN after the Dump command will have the least-significant bit high (xxx1), and the 7L18 processor will abort the dump operation immediately, issuing an "abort" status word (ØØ1Ø) via Port 15 OUT, then resume normal operation. By evaluating the Port 8 IN status word, the interface processor can anticipate the abort, and need not monitor Port 15 OUT until after a completed dump.

Typical timing of the start of the Data Dump operation is shown in the sketch on the next page (Fig. 4).

2.2.2.1 Unaddressed Locations.

About 4Ø locations at the start and another 4Ø at the end of each display memory are not addressed during display conversion and storage, and therefore contain random, non-significant data. The suggested interface firmware clears the corresponding interface memory locations after each RECORD operation, to minimize confusion. Some experimentation with the specific 7L18 to be used may suggest using a smaller or a larger number.

2.2.2.2 Scaling.

Neither the vertical (data dump value) nor the horizontal (memory location) information in a data dump is directly scaled in terms of amplitude or frequency (span/div or center frequency). The controller must be programmed to establish both vertical and horizontal calibration using the CAL OUT and other standard waveforms, even to obtain the same accuracy as specified for operator reading of the CRT display. For accuracy approaching the resolution (Ø.5% FS vertical, Ø.25% FS horizontal), calibration for the exact span, center frequency and reference level used will be required.

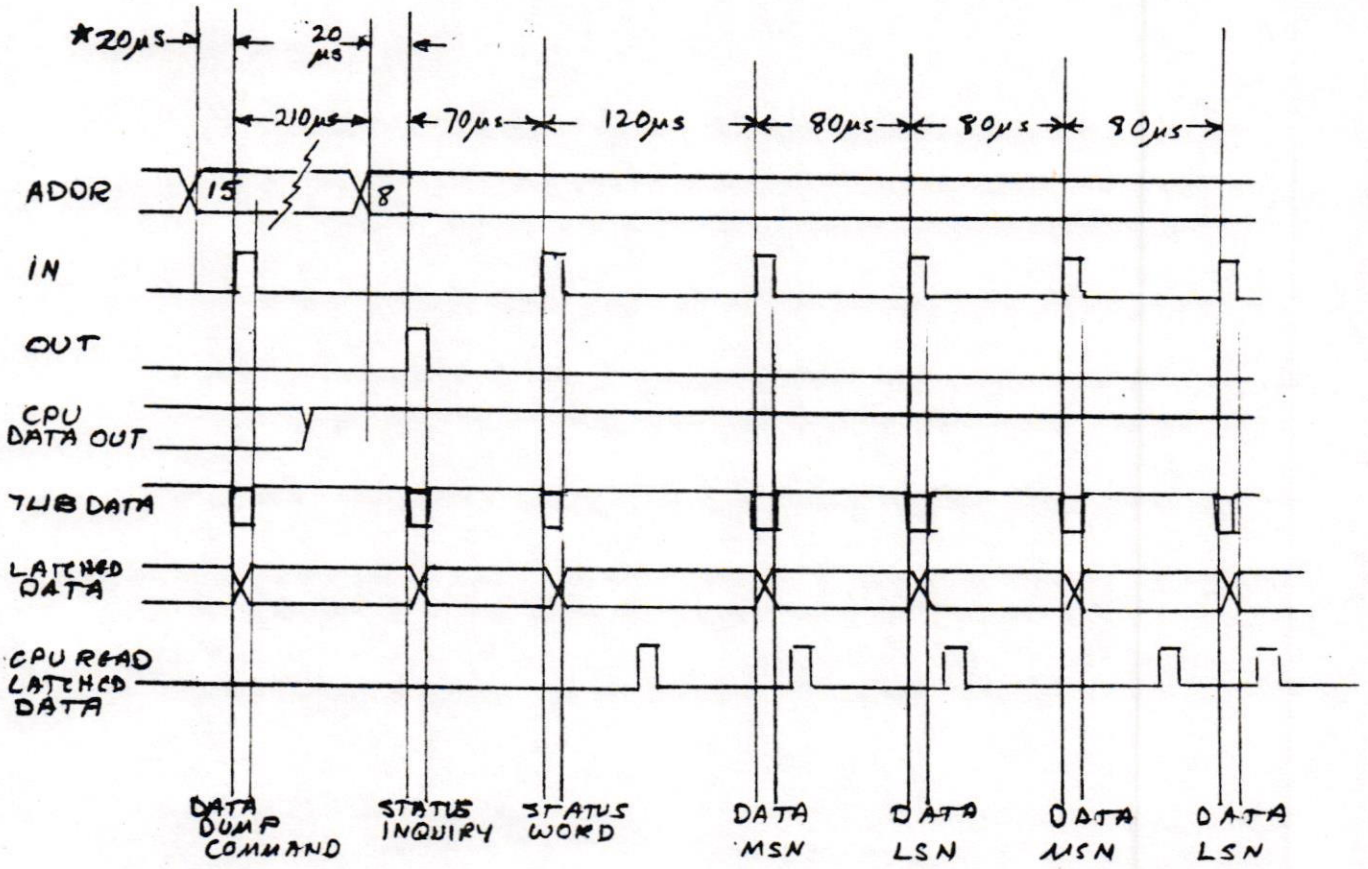


FIGURE 4. TIMING SEQUENCE -- DISPLAY DATA DUMP

*Planning value -- may actually be longer. Others are observed values.

3. Command Input to the 7L18.

The command to dump front-panel control information or display-memory data must be strobed onto the 7L18 data bus by the Port 15 IN pulse, and must not be just "left hanging" on the data bus until a Port 15 IN strobe occurs. Because there is not time (only 20 µs typically) after the Port 15 address is valid to "turn around" a peripheral interface adapter and output the command, special interface hardware is required to strobe the command onto the data bus only when the selected address and the IN pulse occur. Both the command and the address may be preset by the interface processor, after which it need only watch to see that the command actually appears on the bus. Thereafter it will have up to 200 µs to disable the special hardware and start monitoring Port 15 OUT or Port 8 IN for the requested data.

In the suggested configuration, the added hardware consists of a CMOS 4-bit magnitude comparator for address-recognition, and a tri-state CMOS buffer for strobing the command onto the data bus when enabled by the comparator and the IN pulse.

4. Control Output to the 7L18.

Five functions controlled normally by front-panel push-button or rotary switches may be controlled remotely. These are:

Time/div: 20s/div to 1µs/div, Auto, Ext, Manual
Span/div: 0, 5kHz to 500MHz, Identify, Max
Resolution Bandwidth: 30Hz to 3MHz, Auto
Phase-lock: Auto or Off
Band select: 1 to 11

These five front-panel controls are disabled by pulling high on the FPDIS line (nominal pullup current is +5mA to +5V; however, 3V (3mA) is adequate for reliable control). When the controls are disabled, the I/O data lines remain high during the Port 0 through Port 4 IN pulses and may be pulled low by the external interface circuit to effect control of the desired functions.

Since it is not possible to disable individual front-panel controls (the Port 1 control word, for instance, sets two of the five bits which control Time/div and two of the five bits which control Span/div), the control process must be:

(a) Record the present control words, in the interface memory.
(b) Read out the data to the controller.
(c) Change the required parameters in the controller.
(d) Load the interface memory with the entire set of altered and unaltered commands, from the controller.
(e) Substitute the entire set of new commands from the interface

while holding the FPDIS line high continuously.

Additional hardware will be required for this function, since the data must go onto the 7L18 data bus only during Port Ø through Port 4 IN pulses. Furthermore, the substituted commands must be presented every time the Port Ø through Port 4 IN strobes occur. The internal processor does not implement the commands until after several successive IN readings agree, but they are not thereafter "locked in forever" -- only until such time as other readings are registered consistently for several passes.

In the suggested interface, this command-latching function is performed by a 4-bit-wide read/write memory (a Type 21Ø1 was used, but any memory of 5 x 4 bits capacity with tri-state output could be used). The memory address lines are controlled by the 7L18 I/O address bus; the data-in lines go to the interface processor data bus; the data-out lines go to the 7L18 I/O data bus. The data are written into the control memory by the interface processor, monitoring thr 7L18 address lines, and generating a WRITE to the control memory each time a desired address turns up. The data are read out of the memory by a combination of an IN pulse, a Port address lower than 8, and an enable line from the interface processor.

A latched control line from the interface processor enables the control memory and asserts the FPDIS line to the 7L18 at the same time. Once the interface processor has latched the command on line (and has verified it), it may return to other functions, such as recording data and outputting it to the controller.

5. External Display Generation.

An OUTPUT routine has been incorporated in the firmware to output the B and/or A display data from the interface memory repetitively until interrupted by the controller, for creation of an external real-time display. To generate a horizontal sweep for a CRT display from this data, one of the following may be used:

(a) A conventional oscilloscope time-base, triggered from a retriggerable multivibrator having a 1ØØ us period (effectively recognizing the 6 ms interval between displays) the multivibrator in turn triggered from the $\overline{\text{SDAV}}$ handshake line.

(b) A bucket-and-ladle integerator (operational amplifier, diode and capacitor) pumped by the $\overline{\text{SDAV}}$ pulse and reset by a retriggerable multivibrator as in (a) above, or by a comparator set to the proper peak value.

(c) A 9-bit binary counter and ADC, clocked by the $\overline{\text{SDAV}}$ line (self-resetting).

In all cases, and inexpensive 8-bit DAC may be used with the amplitude data-bytes to generate the Y-axis deflection.

For all of these displays, the controller or the display circuit must complete the SDAV-SDAC handshake to obtain a steady data-stream.

The LOAD operation in the interface firmware (routine to load data from the controller into the interface display-data memories) is included to enhance the utility of the OUTPUT routines, by allowing the controller to introduce arbitrary or reference waveforms for display via the OUTPUT routine (OUTPUT may be commanded to alternate A and B displays).

6. Special Precautions.

6.1 7L18 Lines. As mentioned above, the 7L18 I/O lines are un-buffered. If power to the interface is removed while the 7L18 is operating, the interface will load down the I/O lines and effectively disable the 7L18. This occurrence will not normally damage either the interface or the 7L18, but should be avoided on general principle.

There are no protective circuits on the 7L18 I/O lines. The 7L18 circuits connected to these lines (mostly low-power Schottky TTL devices) are therefore subject to damage from static electricity or fault transients introduced via the umbilical. The user will be required to exercise particular care when connecting and disconnecting interface and controller cables to assure that damaging voltages or currents are notintroduced into the 7L18.

6.2 Interface-to-Controller Lines. In the interest of simplicity, no protective circuits are shown on the 20 lines running between the interface and controller. However, in construction, each line should be clamped at circuit ground and the +5V supply bus by diodes, and at least 100 ohms series resistance be added in each line between the clamps and the off-board connection. The MC6820 PIA is a MOS device and quite susceptible to damage by static discharges or fault transients.

If separate input and output buses are selected, more rugged line-buffers may be added to reduce transient susceptibility. For a bi-directional data line, however, additional steering logic and interface/controller protocols would be required to permit use of transceiver-type buffers.

APPENDIX A

"PANEL DUMP" DATA CODING

Data appear as 1Ø consecutive outputs to Port 15 OUT after a "panel dump" request externally introduced to Port 15 IN.

Words	Parameter	Code
1-5	Center Frequency	BCD, with MSD sent first. Code 1Ø1Ø represents a leading zero (blank). For value in GHz, put decimal point after the second digit (nn.nnn).
6	Vertical Mode	Øxxx 1Ø dB/div 1xxx 2 dB/div or LIN
	Cal/Uncal	xxxØ Reference level is uncalibrated xxx1 Reference level is calibrated
7	Reference Level	Binary ØØØØ-1111. Value in dBm is (code minus 1Ø11) times ten: Code ØØØØ = -11Ø dBm.
8	Span Units	ØØØØ = X 1ØØ Hz ØlØØ = X 1 MHz ØØØ1 = X 1 kHz ØlØ1 = X 1Ø MHz ØØ1Ø = X 1Ø kHz Ø11Ø = X 1ØØ MHz or IDENTIFY ØØ11 = X 1ØØ kHz Ø111 = X Ø Hz or MAX SPAN
9	Span Mantissa	ØØØ1 = 2 or Ø Hz ØØ1Ø = 5 ØØ11 = 1Ø, IDENTIFY or MAX SPAN Note: After decoding the Span Units and Mantissa the controller should normalize the units again (e.g., 1 Mhz, not 1Ø x 1ØØ kHz).
1Ø	*Res. Bandwidth	xØØ1 = 3Ø Hz x1ØØ = 3Ø kHz xØ1Ø = 3ØØ Hz x1Ø1 = 3ØØ kHz xØ11 = 3 kHz x11Ø = 3 MHz
	Auto or Manual	Øxxx Manually selected resolution bandwidth 1xxx Auto bandwidth mode

*Note: This is the preferred source for resolution bandwidth data. The Port 3 IN data will indicate the actual setting only if manually selected, indicating simply "Auto" if Auto mode is selected. The readout here shows both the mode and the value at all times.

APPENDIX B

INPUT PORTS DATA CODING

The suggested interface re-groups Port Ø-1-2 data as shown in the right-hand columns. See following pages for detail coding.

7L18			Interface		
Port	Data	Format	Word	Data	Format
Ø	Time/div exponent	xTTT	Ø	Time/div	ØØØEEEMM
1	Time/div mantissa + Span/div mantissa	TTSS	1	Span/div	ØØØEEEMM
2	Phaselock + Span/Div exponent	PSSS	2	Phase lock	ØØØØPØØØ
3	Band	BBBB	3	Band	ØØØØBBBB
4	Resolution Bandwidth	RRRR	4	Res. b/w	ØØØØRRRR
5)			5	--	ØØØØØØØØ
6)	(Not read)		6	--	ØØØØØØØØ
7)			7	--	ØØØØØØØØ
8	Storage Status	ØØØØ*	8	Stor. Stat.	ØØØØØØØØ
9	(Not read)		9	--	ØØØØØØØØ
1Ø	DVM Status	nnØn**	1Ø	DVM Status	ØØØØ****
11	DVM Data	nnnn***	11	DVM Data	ØØØØ****
12	Vertical Mode	VxxV	12	Vert Mode	ØØØØVxxV
13	RF Attenuators	xAAA	13	RF Atten	ØØØØxAAA
14	IF Gain	GGGG	14	IF Gain	ØØØØGGGG
15	External command input	ExxE	15	Ext	ØØØØ1111

*Always ØØØØ except immediately after request for display data dump.
**Two consecutive readings here if Port 11 is to be read.
***Five consecutive readings here when read.
****Invalid data (only one word saved out of 2 or 5).

DETAIL INPUT PORT DATA CODING

Word Ø in Memory D (From Ports Ø & 1)

TIME/DIV

Format: 8 4 2 1 8 4 2 1 -- (Values for hex notation)
Ø Ø Ø E E E M M

Exponent:

Ø Ø Ø		µs or Auto
Ø Ø 1		x 1Ø µs or Manual
Ø 1 Ø		x 1ØØ µs or External
Ø 1 1		ms
1 Ø Ø		x 1Ø ms
1 Ø 1		x 1ØØ ms
1 1 Ø		s
1 1 1		x 1Ø s

Mantissa:

Ø Ø	Auto, Manual or External
Ø 1	1
1 Ø	2
1 1	5

Word 1 in Memory D (From Ports 1 & 2)

SPAN/DIV

Format: 8 4 2 1 8 4 2 1 -- (Values for hex notation)
Ø Ø Ø E E E M M

Note: SPAN/DIV data are read from a rotary switch using Gray code. Just 23 of the possible 32 codes are used. The controller should use table-lookup to determine values.

Ø Ø Ø Ø Ø	Ø Hz
Ø Ø Ø Ø 1	2ØØ Hz
Ø Ø Ø 1 1	5ØØ Hz
Ø Ø Ø 1 Ø	1 kHz
Ø Ø 1 1 Ø	2 kHz
Ø Ø 1 1 1	5 kHz
Ø Ø 1 Ø 1	1Ø kHz
Ø Ø 1 Ø Ø	2Ø kHz
Ø 1 1 Ø Ø	5Ø kHz
Ø 1 1 Ø 1	1ØØ kHz
Ø 1 1 1 1	2ØØ kHz
Ø 1 1 1 Ø	5ØØ Khz
Ø 1 Ø 1 Ø	1 MHz
Ø 1 Ø 1 1	2 Mhz
Ø 1 Ø Ø 1	5 Mhz
Ø 1 Ø Ø Ø	1Ø Mhz
1 1 Ø Ø Ø	2Ø MHz
1 1 Ø Ø 1	5Ø MHz
1 1 Ø 1 1	1ØØ Mhz
1 1 Ø 1 Ø	2ØØ Mhz
1 1 1 1 Ø	5ØØ MHz
1 1 1 1 1	IDENTIFY
1 1 1 Ø 1	MAX

Word 2 in Memory D (From Port 2) PHASE LOCK

```
                  8 4 2 1  8 4 2 1
        Format:   Ø Ø Ø Ø  P Ø Ø Ø

                           1 Ø Ø Ø  Auto phase-lock.
                           Ø Ø Ø Ø  Phase lock off.
```

Word 3 in Memory D (From Port 3) BAND SWITCH

```
                  8 4 2 1  8 4 2 1
        Format:   Ø Ø Ø Ø  B B B B

                           Ø Ø Ø Ø  "Options"

                           Ø Ø Ø 1  Band 1 (1.5 - 3.5 GHz)
                           . . . .  (Band coding 1-11 in binary code)
                           1 Ø 1 1  Band 11 (3Ø.5 - 6Ø.5 GHz)
```

Word 4 in Memory D (From Port 4) RESOLUTION BANDWIDTH

```
                  8 4 2 1  8 4 2 1
        Format:   Ø Ø Ø Ø  R R R R

                           Ø Ø Ø 1  3Ø Hz
                           Ø Ø 1 Ø  3ØØ Hz
                           Ø Ø 1 1  3 kHz
                           Ø 1 Ø Ø  3Ø kHz
                           Ø 1 Ø 1  3ØØ kHz
                           Ø 1 1 Ø  3 Mhz
                           1 Ø Ø Ø   AUTO
```

Words 5-6-7 in Memory D (From Ports 5, 6, 7) NO DATA

Word 8 in Memory D (From Port 8) STORAGE STATUS

```
                  8 4 2 1  8 4 2 1
        Format:   Ø Ø Ø Ø  S S S S

                           Ø Ø Ø Ø  No dump command received.

                           n n n n
                           | | | └--1 = Non-store mode (abort)
                           | | └----1 = SAVE A is on
                           | └------1 = MAX HOLD is on
                           └--------1 = Data dump OK.
```

Note: Nonzero status word is not normally output during a read of IN ports. It is triggered only by a Port 8 OUT command from the 7L18's processor, which is triggered in turn only by a "data dump" command.

Word 9 in Memory D (From Port 9) NO DATA

Word 1Ø in Memory D (From Port 1Ø) DVM STATUS

```
                8 4 2 1  8 4 2 1
Format:         Ø Ø Ø Ø  S S Ø S
                           | |  └--1 = Counter is at LSB
                           | └------1 = Port 11 data isvoltage value
                           └--------1 = Ready to send
```

Note: Port 1Ø data is not normally usable by an external controller. The status information relates only to the Port 11 data.

Word 11 in Memory D (From Port 11) DVM RAW DATA

```
                8 4 2 1  8 4 2 1
Format:         Ø Ø Ø Ø  D D D D
```

Sequence of 5 BCD digits, LSD first.

Note: Port 11 data is not normally usable by an external controller. For center-frequency data, use the "Panel dump" data.

Word 12 in Memory D (From Port 12) VERT MODE & CAL/UNCAL

```
                8 4 2 1  8 4 2 1
Format:         Ø Ø Ø Ø  M x x C
                         |     |
                         |     ├--1 = Calibrated Reference level
                         |     └--Ø = Uncalibrated*
                         ├--------1 = 2 dB/div or LIN
                         └--------Ø = 1Ø dB/div
```

*Reference Variable control is out of its detent "Calibrated" position. Other possible "uncal" conditions are not flagged here.

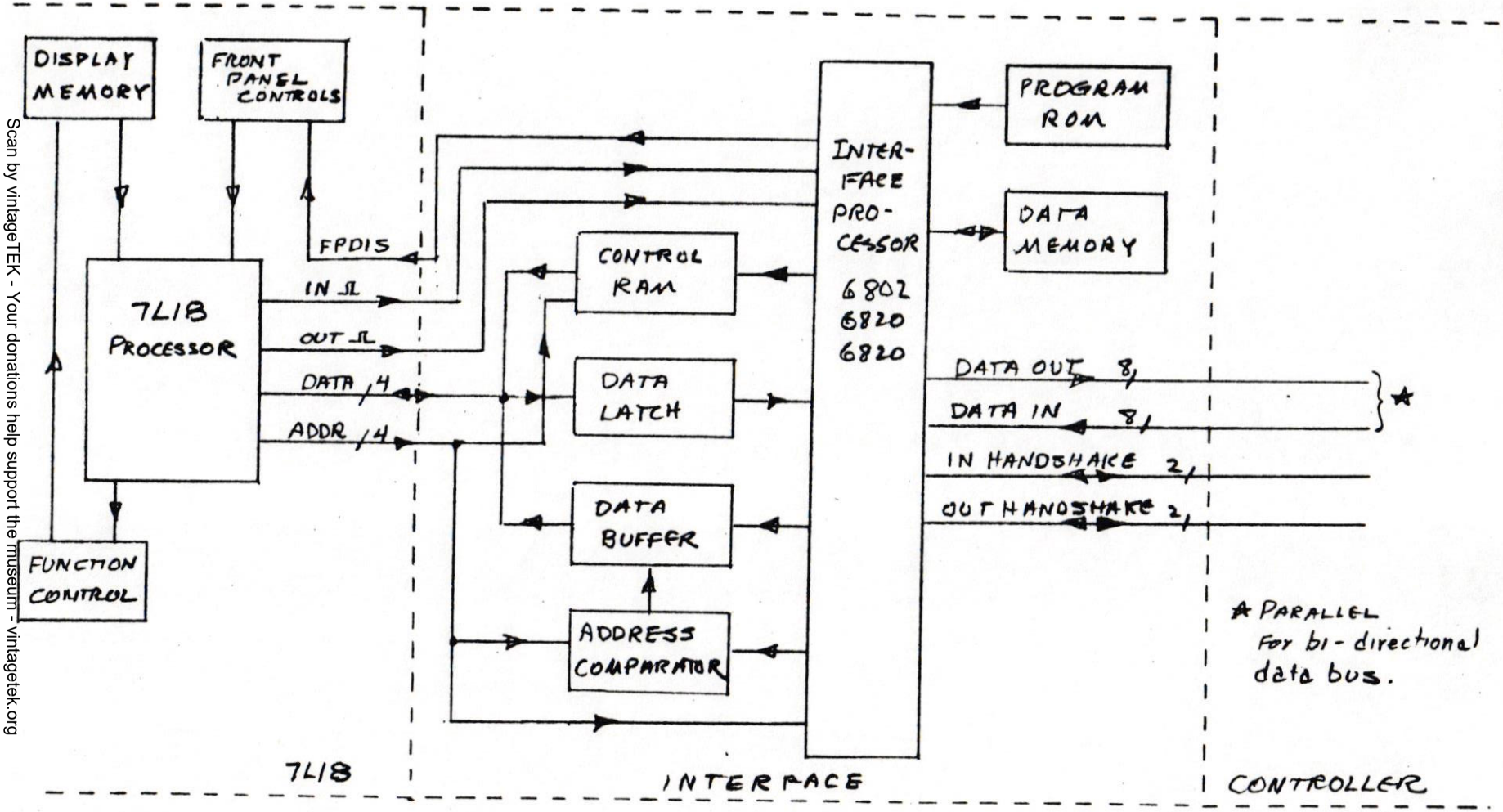
Word 13 in Memory D (From Port 13) RF ATTENUATOR

```
                8 4 2 1  8 4 2 1
Format:         Ø Ø Ø Ø  Ø A A A
```

Ø Ø Ø	Ø dB
Ø Ø 1	1Ø dB
Ø 1 Ø	2Ø dB
Ø 1 1	3Ø dB
1 Ø Ø	4Ø dB
1 Ø 1	5Ø db
1 1 Ø	6Ø dB

Word 14 in Memory D (From Port 14) IF GAIN

Format:	8 4 2 1	8 4 2 1	
	Ø Ø Ø Ø	G G G G	
		1 Ø Ø 1	-1Ø dB
		1 Ø Ø Ø	Ø dB
		Ø 1 1 1	+1Ø dB
		Ø 1 1 Ø	2Ø dB
		Ø 1 Ø 1	3Ø dB
		Ø 1 Ø Ø	4Ø dB
		Ø Ø 1 1	5Ø dB
		Ø Ø 1 Ø	6Ø dB
		Ø Ø Ø 1	7Ø dB
		Ø Ø Ø Ø	8Ø dB



7L18 μP INTERFACE
BLOCK DIAGRAM
9-15-78 GG/CME

7L18 ADDR
7L18 DATA
J20 TO 7L18
U23 4042
LATCH
U26A 1/4 4001
U11 PIA-2 MC6820
OUT
IN
U29F 1/6 80C97
FPDIS
U27 4585
COMPARATOR
A>BI
A=BO
A=BI
A<BI
U28A 1/4 4011
U28B 1/4 4011
DIS4
U29A-D 2/3 80C97
U26B 1/4 4001
U25A 1/4 4011
U21 2101A
OUT DIS
CE2
RAM
WE
CE1
I03 R/W (8n1n)
μP DATA
I02 (8nØ4)
I03 (8n1n)
"E"
U25C 1/4 4011
U26C 1/4 4001
U28D 1/4 4011
U22 2758
ROM 1K x 8
U2-9 8X21L02
RAM 1K x 8
A2 A4
μP 6802
U1
HALT
MR
IRQ
NMI
RESET
4.0 MHz
.001
RESET
47K
(Ø2)
E
A15 A14
VMA
A13 A12 A11 A10
U28C 1/4 4011
U25B,D 1/2 4011
ROMEN
RAMEN
U24A 1/2 4555
DECODER
ROM
I/O
RAM
U26D 1/4 4001
U24B 1/2 4555
IO1(8nØØ)
U12 PIA-1 MC6820
INTER-FACE DATA IN
OPTIONAL BI-DIRECTIONAL BUS.
RDAV
RDAC
SDAC
SDAV
INTER-FACE DATA OUT
R/W
RS0
RS1
CA1
CA2
CB1
CB2
CS2
CS0
PA7
PA0
PB7
PB0
D7
D0

7L18 μP INTERFACE
REV D 9-1-78
GG/CME
C Change U24 (WAS 74LS139) Add U25C, U26D
D Add E pulse to U21 R/W logic

7L18 uP Interface

PARTS LIST

Capacitors

C1	1 ea	2 uF	2Ø%	25V	Tantalum Dip
C2	1 ea	.ØØ1	5%	1ØØV	Ceramic disc

Diodes

CR1	1 ea	Small signal, Silicon, 3ØV	DO-35
CR2-41	4Ø ea	Small signal, Silicon, 3ØV	DO-35 (Protective networks)

Resistors

R1	1 ea	1ØØk	5%	1/4 W	Carbon film
R2	1 ea	47k	5%	1/4 W	Carbon film
R3-22	2Ø ea	1ØØ ohm	5%	1/4 W	Carbon film (Protective networks)

Misc. Electronic

X1	1 ea	Crystal, 4 MHz

Integrated Circuits

U1	1 ea	MC68Ø2	MOS 8-bit Microprocessor
U2-9	8 ea	21LØ2	1k x 1 MOS RAM, 35Ø ns
U11-12	2 ea	MC682Ø or MC6821	MOS Peripheral Interface Adapter
U21	1 ea	21Ø1A	256 x 4 MOS RAM, 35Ø ns
U22	1 ea	2758	1k x 8 EPROM, 5V, MOS
U23	1 ea	4Ø42	Quad latch, CMOS
U24	1 ea	4555	Dual 1 to 4 line decoder, CMOS
U25, 28	2 ea	4Ø11	Quad 2-input NAND gate, CMOS
U26	1 ea	4ØØ1	Quad 2-input NOR gate, CMOS
U27	1 ea	4585	Four-bit magnitude comparator, CMOS
U29	1 ea	8ØC97	Hex tri-state buffer (4 + 2), CMOS

Misc. Hardware and Assemblies

SW1	1 ea	SPST N.O. Pushbutton (RESET)
J2Ø	1 ea	Lemo #RA-C2314 Interface-to-umbilical
J( )	1 ea	DB-25S 25-contact Interface-to-controller
P( )	1 ea	DB-25P 25-contact Interface-to-controller
	3 ea	4Ø-pin sockets
	12 ea	16-pin sockets
	3 ea	14-pin sockets
	1 ea	22-pin socket
	1 ea	ECB
	1 ea	Regulated power-supply: 115VAC to 5V at 2A.

EXTERNAL CONTROLLER COMMANDS

The interface processor responds to single 8-bit commands from the controller as follows:

FORMAT: | 8 4 2 1 Opcode | 8 4 2 1 Operand | -- Values for hex notation

Opcodes: ØØØØ-1111 as shown below.

Operands: Bits in the D,C,B and A positions flag one to four interface memory areas as operands for the command opcode.

Command	Coding	Operation
RECORD	ØØØØ DCBA	Transfer data from 7L18 to interface memory
READ	ØØØ1 DCBA	Interface memory to controller
LOAD	ØØ1Ø DCBA	Controller to interface memory
SUB	ØØ11 1xxx ØØ11 Øx ØØ	Substitute D memory for 7L18 panel controls Re-enable 7L18 panel controls
OUTPUT	ØØ11 ØxAB	Continuous data output until interrupted
(NOP)	Ø1xx xxxx	Reserved for user-defined functions*
(NOP)	1Øxx xxxx	Reserved for user-defined functions*
NOP	11ØØ xxxx	No action (jumps to COMAND)
INZ	11Ø1 xxxx	Resets PIA's, jumps to COMAND
INZ	111Ø xxxx	Resets PIA's, jumps to COMAND
RESET	1111 xxxx	Clears memory, resets CPU & PIA's.

*Firmware as coded returns ENQ on receipt of these instructions.

OPERATIONS

Command	Operation
RECORD (D,C,B,A)	Interface reads data from 7L18. RDAC remains low until operation is finished, or error condition is encountered. Returns status byte on output bus: ACK ($Ø6): Operation complete. ENQ ($Ø5): Bad instruction, or questionable data. NAK ($15): Abort status from 7L18. May not have recorded any data (B or A data only).
READ (D,C,B,A)	Interface reads requested data from interface memory to external device. RDAC goes high as soon as instruction is received. Returns ACK ($Ø6) before outputting data. Returns ENQ ($Ø5) if invalid instruction (i.e., READ ØØ). Returns EOT after last data byte sent.
LOAD (D,C,B,A)	Interface loads selected memory or memories with data from external device. RDAC goes high as soon as instruction is received. Returns ACK (ready to receive data) or ENQ (invalid instruction) when instruction decoded. Then accepts the proper number of bytes for each memory selected, in D,C,B,A order. Issues another ACK when last byte is received.
SUB/OUTPUT	
SUB (Ø)	Restores front-panel control of 7L18; issues ACK when instruction received.
SUB (D)	Sets memory D to control five 7L18 front-panel functions; issues ACK when instruction completed.
OUTPUT (B, A)	Continuous display memory output.

RDAC and ACK issued when instruction is received. If A and/or B is specified, will respond properly. (If only memory C is specified in the operand, the instruction will be treated as SUB (Ø).)

Sets RDAC low, runs through B or A (or B, then A) display memory, sets RDAC high, waits 6 ms, then repeats.

Any command issued to the interface will interrupt the output during the wait period between displays, and then the new command will be read from the bus and be implemented. If a bidirectional bus is used, the external command word will contend with the last display cycle on the bus. To avoid bus contention, the external device should wait for the RDAC line going high before putting the new command on the data bus. The interrupt is implemented by setting the RDAV line low.

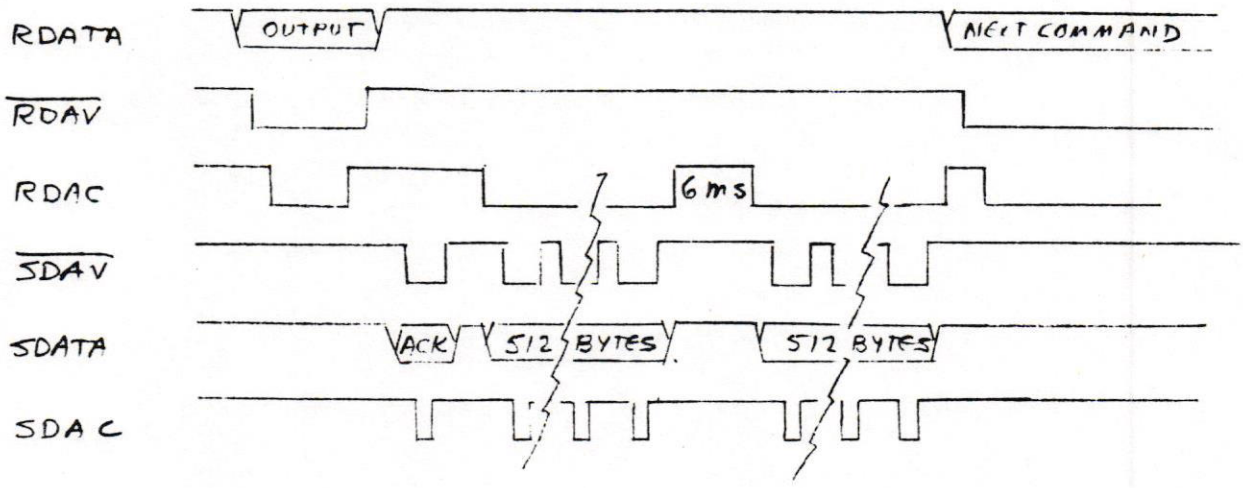
During the display OUTPUT, the external device must supply a SDAC ⌐ for each SDAV ⌙ transition on line. An auto-response circuit may be enabled for this purpose if desired to maximize throughput.

The maximum data output rate (5Ø usec or shorter SDAC response) is a little over 15,ØØØ bytes/sec or a 512-byte display in about 33 ms. With a 6-ms wait between displays, the refresh rate is about 25 hz for a single-memory display, or about 13 hz if both A and B memories are being output.

-o-

D Memory: 16 bytes, front-panel settings.
C Memory: 1Ø bytes, "panel dump" data.
B Memory: 512 bytes, display memory B.
A Memory: 512 bytes, display memory A.

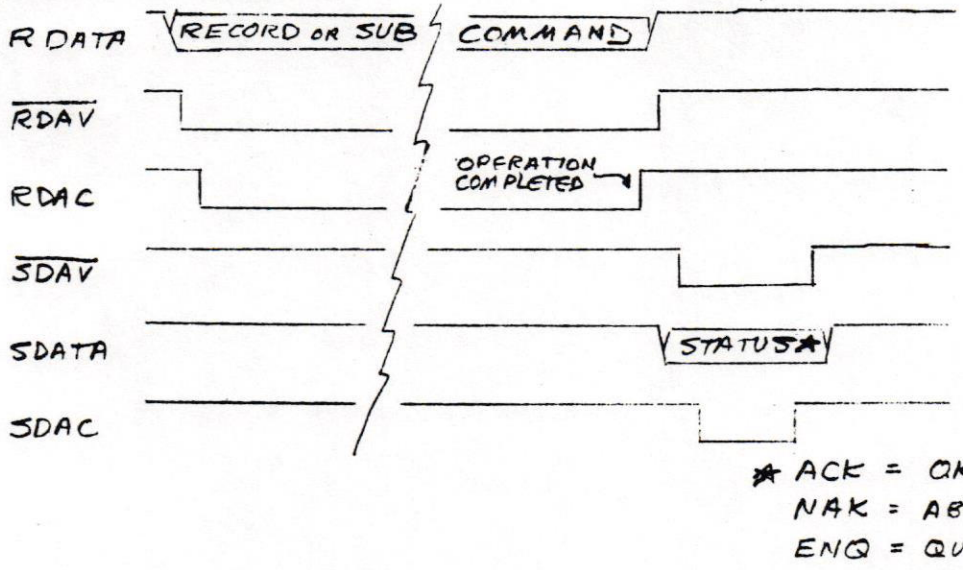
If more than one memory is referred to in a command, the order of reference is always D,C,B,A (READ, LOAD, OUTPUT commands).



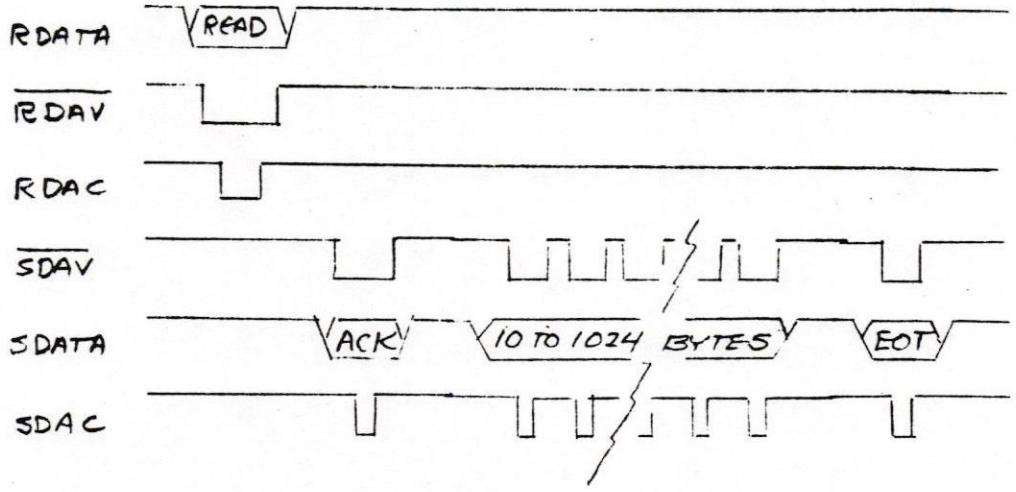
HANDSHAKE DETAIL FOR OUTPUT OPERATION

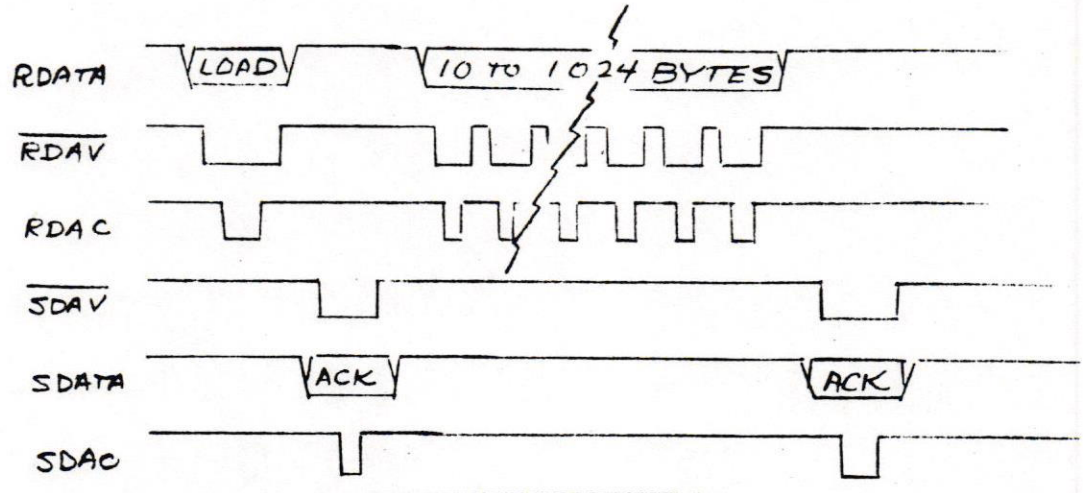
Output continues recurrently until controller interrupts with new command during the 6ms interval between displays, when RDAC is high. RDAC signal can be used to reset the display-generator horizontal deflection system.

7L18 uP Interface



* ACK = OK.
NAK = ABORT
ENQ = QUERY

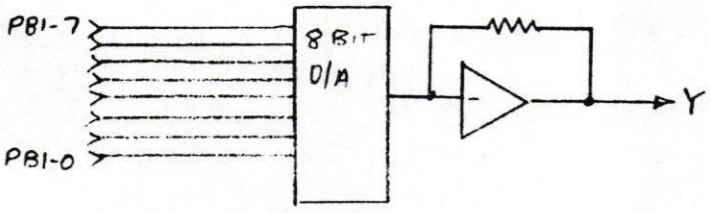




HANDSHAKE DETAIL FOR RECORD, SUB, READ & LOAD

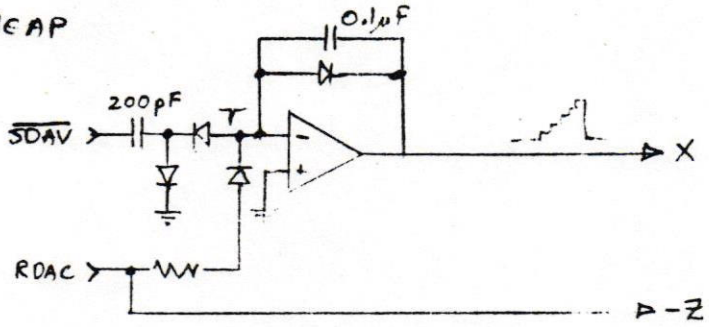
RDATA, $\overline{\text{RDAV}}$ and SDAC signals originate from controller.

Y-AXIS:

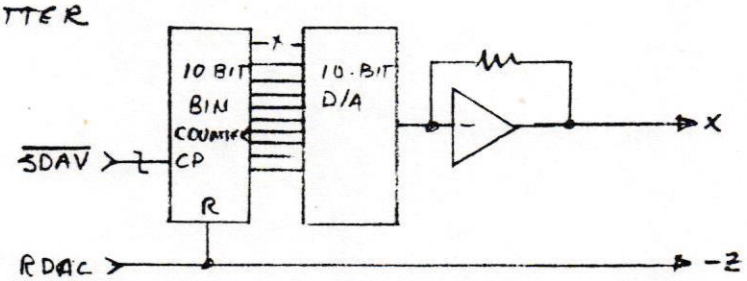


XBZ

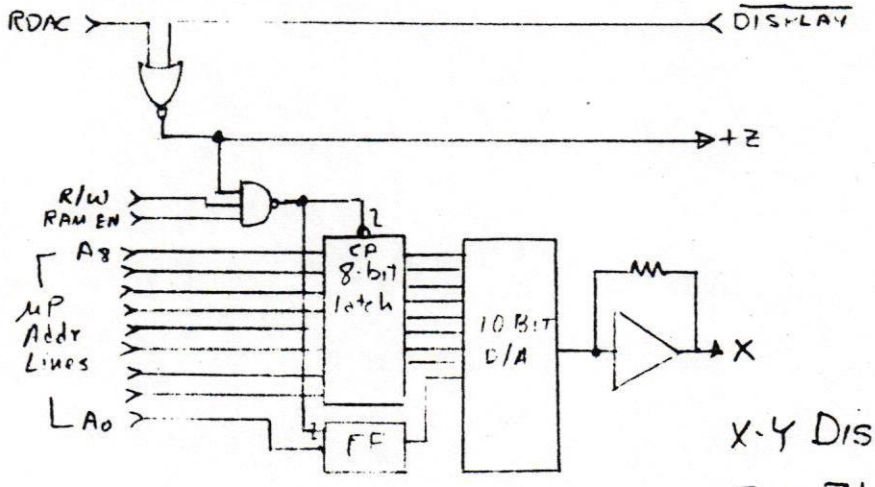
(1) CHEAP



(2) BETTER



(3) MORE ELABORATE



X-Y DISPLAY DRIVERS
From 7L18 μP INTERFACE
GG/CME 7-14-78

```
*
*
*IF7L18:  Program to interface and handle data to and from
*Tektronix Type 7L18 Spectrum Analyzer Plug In Unit.
*
*Program is for Type 68Ø2 MPU with program ROM at
*addresses $FCØØ-$FFFF; I/O at $8ØØØ-$8Ø1Ø; RAM at
*$4ØØØ-$43FF and scratchpad (in CPU) at $ØØØØ-ØØ7F.
*
*
```



```
*
*
*
*PAGE Ø REGISTERS:
*
*
ØØ2Ø   FC       COMND   RMB             Input command.
ØØ21   ØØ       FUNCT   RMB     2       Location in jump-table.
ØØ22   FF
ØØ23   FF               RMB             Reserved.
ØØ24   ØØ       BLANK   RMB             Bytes to be blanked in A & B mem
ØØ25   74       TEMD    RMB             Temporary data storage.
ØØ26   ØØ       IOWORD  RMB             PIA2B I/O pattern.
ØØ27   42               RMB             Reserved.
ØØ28   ØØ       MEMAD   RMB     2       First location for read/write.
ØØ29   ØØ
ØØ2A   3Ø       MEMLOC  RMB     2       Current working location.
ØØ2B   13
ØØ2C   ØØ       MEMEND  RMB     2       Last loc + 1 for read/write.
ØØ2D   FF
ØØ2E   ØØ       IOBASE  RMB     2       1st I/O location (set by IOSET).
ØØ2F   Ø2
*
*
ØØ3Ø   ØØ       STACK   RMB     $2Ø     Stack area $ØØ4F to $ØØ3Ø
*
ØØ5Ø   ØØ       EMEM    RMB     $1Ø     Raw data for D memory.
*
ØØ6Ø   ØØ       DMEM    RMB     $1Ø     IN Port data, re-grouped.
*
ØØ7Ø   ØØ       CMEM    RMB     $1Ø     Panel Dump data Memory C.
*
ØØ8Ø   ØØ       BMEM    EQU     $4ØØØ   Location for B Display Data
ØØ8Ø   ØØ       AMEM    EQU     $42ØØ   Location for A Display Data
*
ØØ8Ø   ØØ               ORG     $FCØØ   Start of interface program.
```

```
*RESET is entry point after power-on or restart.
*
FCØØ   Ø1       RESET   NOP                 Re-sync for runaway.
FCØ1   Ø1               NOP
FCØ2   Ø1               NOP
FCØ3   CE ØØ2Ø          LDX    #$ØØ2Ø       Start of buffer area.
FCØ6   6F ØØ    CLRM    CLR    $ØØ,X        Clear all RAM except $ØØØØ-ØØ1F.
FCØ8   Ø8               INX                 Step.
FCØ9   8C 8ØØØ          CPX    #$8ØØØ       At I/O area yet?
FCØC   26 F8            BNE    CLRM         If not, loop.
*
*IOSET is reentry point for NMI or SWI.
*
FCØE   CE 8ØØ8  IOSET   LDX    #$8ØØ8       Top of PIA group.
FC11   C6 Ø8            LDA B  #$Ø8         Will be clearing 8 locations.
FC13   Ø9       CLRP    DEX                 Clear control registers first.
FC14   6F ØØ            CLR    $ØØ,X
FC16   5A               DEC B
FC17   2E FA            BGT    CLRP
FC19   DF 2E            STX    IOBASE       XR now at $8ØØØ.
FC1B   86 3E            LDA A  #$3E         CB-2 high; CB-1 + transition.
FC1D   A7 Ø7            STA A  $Ø7,X        Disables control RAM.
FC1F   86 16            LDA A  #$16         C()-2 open; C()-1 + transition.
FC21   A7 Ø3            STA A  $Ø3,X        PIA1B control register.
FC23   A7 Ø5            STA A  $Ø5,X        PIA2A control register.
FC25   C6 28            LDA B  #$28         Bytes in display to be cleared.
FC27   D7 24            STA B  BLANK        Page Ø reference for RECORD.
*
*COMND routine inputs command words via PIA1-A;
*finds proper routine in INSTBL jump-table.
*
FC29   Ø1       COMAND  NOP                 Optional breakaway, if needed.
FC2A   Ø1               NOP
FC2B   Ø1               NOP
FC2C   ØF               SEI                 No interrupts.
FC2D   8E ØØ4F          LDS    #$ØØ4F       Reset Stack Pointer.
FC3Ø   DE 2E            LDX    IOBASE       Get start of I/O area.
FC32   86 14            LDA A  #$14         Lock DDR; CA-2 open; CA-1 - edge
FC34   A7 Ø1            STA A  $Ø1,X        PIA1A (receive).
FC36   C6 34            LDA B  #$34         Code to set CA-2 low (RDAC).
FC38   6D Ø1            TST    $Ø1,X        Watch for input IRQ.
FC3A   2A FC            BPL    *-$Ø2        None? Loop.
FC3C   E7 Ø1    COMND1  STA B  $Ø1,X        RDAC low (CA-2).
FC3E   A6 ØØ            LDA A  $ØØ,X        Get command.
FC4Ø   97 2Ø            STA A  COMND        Save it on Page Ø.
FC42   84 FØ            AND A  #$FØ         Scrub off operand.
FC44   44               LSR A               Starts as CCCC ØØØØ.
FC45   44               LSR A
```

```
FC46   44              LSR A                 Ends as ØØØC CCCØ (opcode x 2).
FC47   CE FFEØ         LDX    #INSTBL        S. A. of jump table.
FC4A   DF 21           STX    FUNCT          For manipulation.
FC4C   9B 22           ADD A  FUNCT + 1      Opcode X 2 + LSB
FC4E   97 22           STA A  FUNCT + 1      Position in table.
FC5Ø   DE 21           LDX    FUNCT          Location into XR.
FC52   EE ØØ           LDX    $ØØ,X          Get routine address there.
FC54   AD ØØ           JSR    $ØØ,X          Do the routine.
FC56   2Ø D1           BRA    COMAND         Get next command.
FC58   Ø1              NOP
*
*ERROR outputs a NAK via PIA1B to the external device.
*
FC59   86 15    ERROR  LDA A  #$15           ASCII 'NAK': aborted operation.
*
*STATO  outputs one byte from ACC A via PIA1B;
*then 'turns around' PIA1B to open-circuit state again.
*
FC5B   BD FE76  STATO  JSR    FOUT           'First output' routine.
FC5E   6D Ø3           TST    $Ø3,X          Wait for +SDAC from other device
FC6Ø   2A FC           BPL    *-$Ø2
FC62   6F Ø3           CLR    $Ø3,X          Open DDR for PIA1B
FC64   6F Ø2           CLR    $Ø2,X          Change back to 8 inputs.
FC66   39              RTS                   Return.
*
*ERRORQ sends back ENQ to external device.
*
FC67   86 Ø5    ERRORQ LDA A  #$Ø5           ASCII ENQ: Command error.
FC69   2Ø FØ           BRA    STATO          Output to external device.
*
*ACK sends command acknowledgement to external device.
*
FC6B   86 Ø6    ACK    LDA A  #$Ø6           ASCII ACK: "Roger".
FC6D   2Ø EC           BRA    STATO
*
*DONE sends EOT to external device after READ.
*
FC6F   86 Ø4    DONE   LDA A  #$Ø4           ASCII EOT: "All done".
FC71   2Ø E8           BRA    STATO
*
*RECORD gets 7L18 data from IN bus, Panel Dump or
*Display memory, loads into interface memory.
*
FC73   96 2Ø    RECORD LDA A  COMND           Get command.
FC75   84 ØF           AND A  #$ØF           Scrub opcode, get operand.
FC77   27 EE           BEQ    ERRORQ         Record ØØ?  Invalid command.
FC79   Ø1              NOP
FC7A   Ø1              NOP
```

```
FC7B   Ø1              NOP              (Optional breakaway).
FC7C   DE 2E           LDX    IOBASE
FC7E   E6 Ø7           LDA B  $Ø7,X     Get PIA2B control code.
FC8Ø   C4 FB           AND B  #$FB      Scrub DDR bit.
FC82   E7 Ø7           STA B  $Ø7,X     Back into PIA2B control register
FC84   CA Ø6           ORA B  #$Ø6      DDR lock; CB-1 + transition.
FC86   37              PSH B            Save that code for later.
FC87   D6 26           LDA B  IOWORD    I/O pattern for PIA2B.
FC89   C4 FØ           AND B  #$FØ      No outputs at 4 low-order bits.
FC8B   E7 Ø6           STA B  $Ø6,X     Set data-direction register.
FC8D   33              PUL B            Get saved code.
FC8E   E7 Ø7           STA B  $Ø7,X     Lock up DDR.
FC9Ø   85 Ø8           BIT A  #$Ø8      Check operand: record D?
FC92   27 5D           BEQ    CREC      If not, check for C.
FC94   C6 1Ø    DREC   LDA B  #$1Ø      Recording D...16 bytes.
FC96   CE ØØ5Ø         LDX    #EMEM     E buffer for raw D data.
FC99   DF 28           STX    MEMAD     Start location
FC9B   DF 2A           STX    MEMLOC    Working location
FC9D   6F ØØ    CLRE   CLR    $ØØ,X     Clear out any old data.
FC9F   Ø8              INX              Step.
FCAØ   5A              DEC B            Count.
FCA1   2E FA           BGT    CLRE      Loop until done.
FCA3   C6 4Ø           LDA B  #$4Ø      To assure catching all ports.
FCA5   DE 2E    NEXTD  LDX    IOBASE    Set XR for I/O operations.
FCA7   A6 Ø6           LDA A  $Ø6,X     Clear any old IRQ's.
FCA9   A6 Ø7           LDA A  $Ø7,X     Look for IRQ (IN pulse).
FCAB   2A FC           BPL    *-$Ø2     Loop until IN detected.
FCAD   A6 Ø4           LDA A  $Ø4,X     Get address & data.
FCAF   36              PSH A            Save it.
FCBØ   44              LSR A            Shift address down to low-order.
FCB1   44              LSR A
FCB2   44              LSR A
FCB3   44              LSR A
FCB4   9B 29           ADD A  MEMAD + 1 Add to start of buffer (LSB).
FCB6   97 2B           STA A  MEMLOC + 1 Working location.
FCB8   DE 2A           LDX    MEMLOC    Get location in XR.
FCBA   32              PUL A            Recover data.
FCBB   84 ØF           AND A  #$ØF      Scrub off address.
FCBD   A7 ØØ           STA A  $ØØ,X     Into buffer.
FCBF   5A              DEC B            Operation counter.
FCCØ   2E E3           BGT    NEXTD     Loop until done.
FCC2   DE 28           LDX    MEMAD     $ØØ5Ø, E memory.
FCC4   A6 ØØ           LDA A  $ØØ,X     Get Time/Div Exponent
FCC6   84 Ø7           AND A  #$Ø7      Scrub extraneous data.
FCC8   48              ASL A            Shift left (also clears Carry).
FCC9   48              ASL A            Now ØØØE EEØØ
FCCA   E6 Ø1           LDA B  $Ø1,X     Time/Div mantissa.
FCCC   C4 ØC           AND B  #$ØC      Scrub other data.
```

```
FCCE   Ø1              NOP              ØØØØ MMØØ
FCCF   54              LSR B            Shift down
FCDØ   54              LSR B            Now ØØØØ ØØMM
FCD1   1B              ABA              Assemble in A: ØØØE EEMM.
FCD2   A7 1Ø           STA A  $1Ø,X     Into D register at $ØØ6Ø.
FCD4   A6 Ø2           LDA A  $Ø2,X     Get Span/div exponent.
FCD6   84 Ø7           AND A  #$Ø7      Scrub other data bit.
FCD8   48              ASL A            Shift.
FCD9   48              ASL A            Now ØØØE EEØØ.
FCDA   E6 Ø1           LDA B  $Ø1,X     Mantissas.
FCDC   C4 Ø3           AND B  #$Ø3      Delete Time/div data.
FCDE   1B              ABA              Assemble Span/div in A.
FCDF   A7 11           STA A  $11,X     Into D register at $ØØ61.
FCE1   A6 Ø2           LDA A  $Ø2,X     Phase-lock code.
FCE3   84 Ø8           AND A  #$Ø8      One-bit code: ØØØØ PØØØ.
FCE5   A7 12           STA A  $12,X     Into D register.
FCE7   C6 ØD           LDA B  #$ØD      13 more bytes in buffer.
FCE9   A6 Ø3   TRANSD  LDA A  $Ø3,X     Move other data to D register.
FCEB   A7 13           STA A  $13,X
FCED   Ø8              INX              Step
FCEE   5A              DEC B            Count
FCEF   2E F8           BGT    TRANSD    Loop
*
*CREC obtains a dump of front-panel control information.
*
FCF1   96 2Ø   CREC    LDA A  COMND     Get original command.
FCF3   85 Ø4           BIT A  #$Ø4      Check for "C" bit.
FCF5   27 25           BEQ    ABQP      None? Check for A or B.
FCF7   CE ØØ7Ø         LDX    #CMEM     Get S.A. of C register.
FCFA   DF 28           STX    MEMAD     Start-point reference.
FCFC   DF 2A           STX    MEMLOC    Working location.
FCFE   86 F8           LDA A  #$F8      Port 15, command 1ØØØ.
FDØØ   8D 1C           BSR    COMOUT    Output command to 7L18.
FDØ2   C6 ØA           LDA B  #$ØA      Will be getting 1Ø output words.
FDØ4   DE 2E   NEXTC   LDX    IOBASE    Start of I/O area.
FDØ6   A6 Ø5   OUTCK   LDA A  $Ø5,X     Watch for OUT pulse.
FDØ8   2A FC           BPL    OUTCK     None? Loop.
FDØA   A6 Ø4           LDA A  $Ø4,X     Get address & data; clear IRQ.
FDØC   81 FØ           CMP A  #$FØ      Check address: 15?
FDØE   25 F6           BCS    OUTCK     If not, get next.
FD1Ø   DE 2A           LDX    MEMLOC    Addr = 15. Get register location
FD12   84 ØF           AND A  #$ØF      Scrub address data.
FD14   A7 ØØ           STA A  $ØØ,X     Store data.
FD16   Ø8              INX              Next location.
FD17   DF 2A           STX    MEMLOC    For next operation.
FD19   5A              DEC B            Count
FD1A   2E E8           BGT    NEXTC     Loop
FD1C   2Ø 57   ABQP    BRA    ABQ       Done. Go to A - B check.
```

```
*
*COMOUT outputs command in low-order bits of ACC A to 7L18
*to IN Port address contained in high-order bits of ACC A.
*Specialized interface hardware puts data on-line to the
*7L18 when the 7L18 address bus matches the ACC A address.
*
FD1E   97 25    COMOUT  STA A  TEMD        Save command.
FD2Ø   DE 2E            LDX    IOBASE      Get start of I/O area in XR.
FD22   86 3E            LDA A  #$3E        Code for CA-2 high.
FD24   A7 Ø5            STA A  $Ø5,X       Disable comparator via PIA2A
fd26   C6 FF            LDA B  #$FF        Code for 8 outputs.
FD28   8D 21            BSR    BSET        Set PIA2B control.
FD2A   96 25            LDA A  TEMD        Get command (addr + data)
FD2C   A7 Ø6            STA A  $Ø6,X       PIA2B data register.
FD2E   C6 ØF            LDA B  #$ØF        Highest code for Address Ø.
FD3Ø   A6 Ø6            LDA A  $Ø6,X       Clear any IN IRQ's.
FD32   E1 Ø4            CMP B  $Ø4,X       Check for a "Ø" address.
FD34   25 FA            BCS    *-$Ø4       $1Ø or higher? Keep looking.
FD36   86 36            LDA A  #$36        Code to set CA-2 low.
FD38   A7 Ø5            STA A  $Ø5,X       Enable address comparator.
FD3A   96 25            LDA A  TEMD        Get command again.
FD3C   E6 Ø7    COMCHK  LDA B  $Ø7,X       Watch for IN pulse.
FD3E   2A FC            BPL    COMCHK      Loop until IN IRQ.
FD4Ø   E6 Ø6            LDA B  $Ø6,X       Got an IN. Clear IRQ.
FD42   A1 Ø4            CMP A  $Ø4,X       Check address & data.
FD44   26 F6            BNE    COMCHK      Hasn't shown yet? Get next IN.
FD46   86 3E    COMEX   LDA A  #$3E        Code to disable comparator.
FD48   A7 Ø5            STA A  $Ø5,X       CA-2 high via PIA2A control.
FD4A   5F               CLR B              ØØ for 8 inputs.
FD4B   A6 Ø7            LDA A  $Ø7,X       Get PIA2B control register.
FD4D   84 FB            AND A  #$FB        Clear the DDR bit.
FD4F   A7 Ø7            STA A  $Ø7,X       Opens DDR.
FD51   E7 Ø6            STA B  $Ø6,X       Set I/O pattern.
FD53   8A Ø4            ORA A  #$Ø4        Set DDR bit.
FD55   A7 Ø7            STA A  $Ø7,X       Lock up DDR.
FD57   A6 Ø6            LDA A  $Ø6,X       Clear any IN IRQ.
FD59   39               RTS                Return.
*
*MPREP sets up Page Ø registers for RECORD and OUTPUT routines
*
FD5A   CE 42ØØ  MPREP   LDX    #AMEM       Start of A register.
FD5D   85 Ø1            BIT A  #$Ø1        ACC A has command word.
FD5F   27 Ø3            BEQ    *+$Ø5       Not recording A? Skip next.
FD61   CE 43FF          LDX    #$43FF      Will do A. Get end of A.
FD64   DF 2C            STX    MEMEND      Set terminating address.
FD66   CE 4ØØØ          LDX    #BMEM       Start of B register.
FD69   DF 2A            STX    MEMLOC      1st working address always.
FD6B   85 Ø2            BIT A  #$Ø2        Is B bit in command?
```

```
FD6D   26 Ø3            BNE    *+$Ø5          If so, skip next.
FD6F   CE 42ØØ          LDX    #AMEM          A only. Get start of A.
FD72   DF 28            STX    MEMAD          Set operation-start register.
FD74   39               RTS                   That's all.
*
*ABQ checks for A or B RECORD operation.
*
FD75   96 2Ø    ABQ     LDA A  COMND          Get command word.
FD77   85 Ø3            BIT A  #$Ø3           Recording A or B?
FD79   26 Ø3            BNE    DUMPC          If either, do it.
FD7B   7E FC6B          JMP    ACK            Else, terminate operations.
*
*DUMPC sets up for display memory A or B recording.
*
FD7E   8D DA    DUMPC   BSR    MPREP          Set up pointers per command word
FDØØ   86 F1            LDA A  #$F1           Port 15, command ØØØ1.
FD82   8D 9A            BSR    COMOUT         COMOUT sets XR to IOBASE.
FD84   A6 Ø5            LDA A  $Ø5,X          Watch for OUT pulse.
FD86   2A FC            BPL    *-$Ø2          Loop until OUT detected.
FD88   A6 Ø4            LDA A  $Ø4,X          Clear the OUT IRQ.
FD8A   E6 Ø7            LDA B  $Ø7,X          Now, watch for IN pulse.
FD8C   2A FC            BPL    *-$Ø2          Loop until next IN.
FD8E   E6 Ø6            LDA B  $Ø6,X          Clear that IRQ.
FD9Ø   A6 Ø4            LDA A  $Ø4,X          Get the IN data.
FD92   16               TAB                   Save a copy in B.
FD93   C4 FØ            AND B  #$FØ           Scrub off the data.
FD95   C1 8Ø            CMP B  #$8Ø           Check the address.
FD97   27 Ø7            BEQ    READS          Port 8 IN? All's well.
FD99   C1 FØ            CMP B  #$FØ           Port 15 IN?
FD9B   27 E1            BEQ    DUMPC          Command ignored? Try again.
FD9D   7E FC67  ERRQP   JMP    ERRORQ         Not 8 or 15? Abort.
FDAØ   85 Ø1    READS   BIT A  #$Ø1           Check status byte from 8-IN.
FDA2   27 Ø3            BEQ    ABSTOR         Non-store bit low? OK.
FDA4   7E FC59  ERRORP  JMP    ERROR          Non-store bit high.  Abort.
*
*ABSTOR inputs and assembles Display A or Display B memory
*data from 7L18 and stores in Registers B or A as full bytes.
*
FDA7   5F       ABSTOR  CLR B                 B is "save" flag: Ø = no-save.
FDA8   DE 2E    ABSTO1  LDX    IOBASE         Reset XR (needed in loop).
FDAA   A6 Ø7    ABSTO2  LDA A  $Ø7,X          Look for IN pulse.
FDAC   2A FC            BPL    ABSTO2         No IRQ? Loop.
FDAE   A6 Ø6            LDA A  $Ø6,X          Clear IRQ.
FDBØ   A6 Ø4            LDA A  $Ø4,X          Get address and data.
FDB2   97 25            STA A  TEMD           Save it.
FDB4   84 FØ            AND A  #$FØ           Scrub the data.
FDB6   81 8Ø            CMP A  #$8Ø           Check the address.
FDB8   26 EA            BNE    ERRQP          Not 8? Something's wrong.
```

```
FDBA   96 25          LDA A  TEMD        Address OK. Recover data.
FDBC   48             ASL A              Shift data to MSD position.
FDBD   48             ASL A
FDBE   48             ASL A
FDBF   48             ASL A              Now DDDD ØØØØ.
FDCØ   97 25          STA A  TEMD        Park it.
FDC2   A6 Ø7          LDA A  $Ø7,X       Look for next IN pulse.
FDC4   2A FC          BPL    *-$Ø2       None yet? Loop.
FDC6   A6 Ø6          LDA A  $Ø6,X       Clear IRQ.
FDC8   A6 Ø4          LDA A  $Ø4,X       Get address & data.
FDCA   84 ØF          AND A  #$ØF        Chop off address.
FDCC   9B 25          ADD A  TEMD        Assemble with MSD.
FDCE   DE 2A          LDX    MEMLOC      Get working location.
FDDØ   9C 28          CPX    MEMAD       At start-recording point?
FDD2   26 Ø1          BNE    *+$Ø3       If not, skip next.
FDD4   5A             DEC B              Set "save" flag.
FDD5   5D             TST B              Check "save" flag.
FDD6   27 Ø2          BEQ    *+$Ø4       Ø? Skip the Store operation.
FDD8   A7 ØØ          STA A  $ØØ,X       Data into register.
FDDA   Ø8             INX                Next location.
FDDB   DF 2A          STX    MEMLOC      For next go 'round.
FDDD   9C 2C          CPX    MEMEND      $42ØØ or $43FF.
FDDF   27 C6          BEQ    ABSTOR      Clear B; don't store next block.
FDE1   8C 44ØØ        CPX    #$44ØØ      Past end of registers?
FDE4   26 C2          BNE    ABSTO1      If not, keep on truckin'.
FDE6   DE 2E  ABDONE  LDX    IOBASE      Hit $44ØØ. End of operations.
FDE8   E6 Ø5  ENDCK   LDA B  $Ø5,X       Watch for an OUT pulse.
FDEA   2A FC          BPL    ENDCK       No IRQ? Loop.
FDEC   A6 Ø4          LDA A  $Ø4,X       Get OUT data.
FDEE   81 FØ          CMP A  #$FØ        Address 15?
FDFØ   25 F6          BCS    ENDCK       If not, keep looking.
FDF2   D6 24          LDA B  BLANK       Get # of invalid data locations.
FDF4   27 1C          BEQ    ABEXIT      None? Do nothing.
FDF6   5Ø             NEG B              Get complement.
FDF7   D7 2B          STA B  MEMLOC + 1  S.A. for end-register clearing.
FDF9   D7 2D          STA B  MEMEND + 1  Also for A register.
FDFB   C6 41          LDA B  #$41        2nd-half addresses, B register.
FDFD   D7 2A          STA B  MEMLOC      MEMLOC now $41nn.
FDFF   CB Ø2          ADD B  #$Ø2        Now $43.
FEØ1   D7 2C          STA B  MEMEND      MEMEND now $43nn.
FEØ3   CE 4ØØØ        LDX    #BMEM       Start point for clearing.
FEØ6   8D 15          BSR    CLREND      Clear (BLANK) bytes, start of B.
FEØ8   DE 2A          LDX    MEMLOC      (BLANK) bytes from end of B.
FEØA   8D 11          BSR    CLREND      Clear end of B register.
FEØC   8D ØF          BSR    CLREND      Now the start of A register.
FEØE   DE 2C          LDX    MEMEND      Now the end of A register.
FE1Ø   8D ØB          BSR    CLREND      Do it.
FE12   81 F2  ABEXIT  CMP A  #$F2        Check that "15 OUT" word.
```

```
FE14   27 8E            BEQ    ERRORP        'Abort' code? Issue a NAK.
FE16   81 F1            CMP A  #$F1          Check for "OK" code.
FE18   26 83            BNE    ERRQP         Not OK?  Send out ENQ.
FE1A   7E FC6B          JMP    ACK           Was "OK". Output ACK; return.
*
*CLREND blanks (BLANK) consecutive bytes, per XR.
*
FE1D   D6 24    CLREND  LDA B  BLANK         Get # of invalid data locations.
FE1F   6F ØØ    CLRE1   CLR    $ØØ,X         Clear 1 location
FE21   Ø8               INX                  Step
FE22   5A               DEC B                Count
FE23   2E FA            BGT    CLRE1         Loop
FE25   39               RTS                  Return.
FE26   Ø1               NOP
FE27   Ø1               NOP
FE28   Ø1               NOP
*
*READ outputs content of Registers D, C, B and/or A to
*the external device via PIA1B, per external device command.
*
FE29   D6 2Ø    READ    LDA B  COMND         Get command.
FE2B   C5 ØF            BIT B  #$ØF          Check for operand.
FE2D   26 Ø3            BNE    *+$Ø5         Got one? OK.
FE2F   7E FC67          JMP    ERRORQ        Operand Ø. Invalid command.
FE32   86 Ø6            LDA A  #$Ø6          ASCII 'ACK'.
FE34   8D 4Ø            BSR    FOUT          RDAC +; set outputs; issue ACK.
FE36   96 2Ø            LDA A  COMND         Get command.
FE38   85 Ø8            BIT A  #$Ø8          Check for 'D' bit.
FE3A   27 Ø7            BEQ    CREAD         None? Try C.
*
*DREAD outputs 16 bytes from D register.
*
FE3C   C6 1Ø    DREAD   LDA B  #$1Ø          Byte count (16).
FE3E   CE ØØ6Ø          LDX    #DMEM         S. A. of D register.
FE41   8D 46            BSR    CONTO         Output translated buffer.
*
*CREAD outputs 1Ø bytes from C register (panel dump data).
*
FE43   96 2Ø            LDA A  COMND         Get command.
FE45   85 Ø4            BIT A  #$Ø4          Check for 'C' bit.
FE47   27 Ø7            BEQ    BREAD         None? Try B.
FE49   C6 ØA            LDA B  #$ØA          Byte count (1Ø).
FE4B   CE ØØ7Ø          LDX    #CMEM         S. A. of C register.
FE4E   8D 39            BSR    CONTO         Output 1Ø bytes.
*
*BREAD outputs data from B register $4ØØØ-$41FF (512 bytes).
*
FE5Ø   96 2Ø    BREAD   LDA A  COMND         Get command.
```

```
FE52   85 Ø2          BIT A  #$Ø2        Check for 'B' bit.
FE54   27 ØA          BEQ    AREAD       None? Try A.
FE56   CE 42ØØ        LDX    #AMEM       S. A. of A register.
FE59   DF 2C          STX    MEMEND      Limit for output.
FE5B   CE 4ØØØ        LDX    #BMEM       Start point.
FE5E   8D 3Ø          BSR    MEMO        Output the data.
*
*AREAD outputs data from A register $42ØØ-43FF (512 bytes).
*
FE6Ø   96 2Ø    AREAD LDA A  COMND        Get command word.
FE62   85 Ø1          BIT A  #$Ø1        Check for 'A' bit.
FE64   27 ØA          BEQ    EXREAD      None? Exit.
FE66   CE 44ØØ        LDX    #$44ØØ      End of A register + 1.
FE69   DF 2C          STX    MEMEND      Limit.
FE6B   CE 42ØØ        LDX    #AMEM       Start.
FE6E   8D 2Ø          BSR    MEMO        Output data.
*
FE7Ø   7E FC6F  EXREAD JMP    DONE        Output EOT; return to COMAND.
FE73   Ø1              NOP
FE74   Ø1              NOP
FE75   Ø1              NOP
*
*FOUT releases RDAC; sets up 8 outputs at PIA1B;
*outputs character in ACC A; leaves outputs set at PIA1B.
*
FE76   DE 2E    FOUT   LDX    IOBASE      Get start of I/O area.
FE78   C6 14           LDA B  #$14        CA-2 open; CA-1 - transition.
FE7A   E7 Ø1           STA B  $Ø1,X       PIA1A control: release RDAC.
FE7C   C6 FF    FOUT1  LDA B  #$FF        Code for 8 outputs.
FE7E   6F Ø3           CLR    $Ø3,X       Open DDR on B side.
FE8Ø   E7 Ø2           STA B  $Ø2,X       Set outputs.
FE82   C6 26           LDA B  #$26        Code for CB-2: /SDAV after write
FE84   E7 Ø3           STA B  $Ø3,X       Lock DDR; CB-1 + edge.
FE86   A7 Ø2           STA A  $Ø2,X       Output character; set /SDAV.
FE88   39              RTS                /SDAV will be released by CB-1 +
*
*CONTO outputs (ACC B) characters per XR to external device.
*
FE89   8D ØD    CONTO  BSR    OUT1        Output character in ACC A.
FE8B   Ø8              INX                Step
FE8C   5A              DEC B              Count
FE8D   2E FA           BGT    CONTO       Loop
FE8F   39              RTS                Return.
*
*MEMO outputs data to external device, per XR up to MEMEND.
*
FE9Ø   8D Ø6    MEMO   BSR    OUT1        Output 1 character.
FE92   Ø8              INX                Step
```

```
FE93   9C 2C            CPX    MEMEND       At limit?
FE95   26 F9            BNE    MEMO         If not loop.
FE97   39               RTS                 At limit. Return.
*
*OUT1 outputs 1 character per XR, via PIA1B.
*
FE98   A6 ØØ    OUT1    LDA A  $ØØ,X        Get character.
FE9A   DF 2A            STX    MEMLOC       Save XR value.
FE9C   DE 2E            LDX    IOBASE       Get start of I/O area.
FE9E   6D Ø3            TST    $Ø3,X        Check for +SDAC from previous.
FEAØ   2A FC            BPL    *-$Ø2        Wait for it if not here.
FEA2   A7 Ø2            STA A  $Ø2,X        Output PIA1B data.
FEA4   A6 Ø2            LDA A  $Ø2,X        Clear IRQ.
FEA6   DE 2A            LDX    MEMLOC       Recover XR.
FEA8   39               RTS                 Return.
*
*LOAD inputs data from external device via PIA1A,
*to be stored in interface memory registers.
*
FEA9   D6 2Ø    LOAD    LDA B  COMND        Get command word.
FEAB   C5 ØF            BIT B  #$ØF         Check operand.
FEAD   26 Ø3            BNE    *+$Ø5        Not Ø? Go.
FEAF   7E FC67          JMP    ERRORQ       Invalid command. Output ENQ.
FEB2   8D 72            BSR    ACKP         Output ACK.
*
*DLOAD accepts 16 bytes for D register.
*
FEB4   96 2Ø    DLOAD   LDA A  COMND        Get command.
FEB6   85 Ø8            BIT A  #$Ø8         Check for 'D' bit.
FEB8   27 Ø7            BEQ    CLOAD        None?  Check C.
FEBA   C6 1Ø            LDA B  #$1Ø         Byte count.
FEBC   CE ØØ6Ø          LDX    #DMEM        S. A. of Register D.
FEBF   8D 2F            BSR    CONTLD       Input 16 bytes.
*
*CLOAD accepts 1Ø bytes for C register.
*
FEC1   96 2Ø    CLOAD   LDA A  COMND        Get command.
FEC3   85 Ø4            BIT A  #$Ø4         Check for 'C' bit.
FEC5   27 Ø7            BEQ    BLOAD        No C?  Try B.
FEC7   C6 ØA            LDA B  #$ØA         Count for 1Ø bytes.
FEC9   CE Ø07Ø          LDX    #CMEM        Get S. A. of register.
FECC   8D 22            BSR    CONTLD       Input the data.
*
*BLOAD accepts 512 bytes for storage in B register.
*
FECE   96 2Ø    BLOAD   LDA A  COMND        Get command.
FEDØ   85 Ø2            BIT A  #$Ø2         Look for 'B' bit.
FED2   27 ØA            BEQ    ALOAD        None?  Try A.
```

```
FED4   CE 42ØØ          LDX    #AMEM        Limit for input.
FED7   DF 2C            STX    MEMEND
FED9   CE 4ØØØ          LDX    #BMEM        Start point.
FEDC   8D 1B            BSR    MEMLD        Input data.
*
*ALOAD accepts 512 bytes for storage in A register.
*
FEDE   96 2Ø    ALOAD   LDA A  COMND        Get command.
FEEØ   85 Ø1            BIT A  #$Ø1         Check for 'A' bit.
FEE2   27 ØA            BEQ    ENDLD        None?  You're done.
FEE4   CE 44ØØ          LDX    #$44ØØ       Limit.
FEE7   DF 2C            STX    MEMEND
FEE9   CE 42ØØ          LDX    #AMEM        Start point.
FEEC   8D ØB            BSR    MEMLD        Load $42ØØ-$43FF.
*
FEEE   2Ø 36    ENDLD   BRA    ACKP         Output ACK; back to COMAND.
*
*CONTLD stores (ACC B) bytes per XR from external device.
*
FEFØ   8D 11    CONTLD  BSR    IN1          Get input byte.
FEF2   A7 ØØ            STA A  $ØØ,X        Store per XR.
FEF4   Ø8               INX                 Step
FEF5   5A               DEC B               Count
FEF6   2E F8            BGT    CONTLD       Loop
FEF8   39               RTS                 Return.
*
*MEMLD stores data per XR up to MEMEND, from external device.
*
FEF9   8D Ø8    MEMLD   BSR    IN1          Get input data byte.
FEFB   A7 ØØ            STA A  $ØØ,X        Store per XR.
FEFD   Ø8               INX                 Step
FEFE   9C 2C            CPX    MEMEND       At limit?
FFØØ   26 F7            BNE    MEMLD        If not, loop.
FFØ2   39               RTS                 Done.
*
*IN1 inputs 1 character from external device via PIA1A.
*
FFØ3   DF 2A    IN1     STX    MEMLOC       Save XR.
FFØ5   DE 2E            LDX    IOBASE       Start of I/O area.
FFØ7   86 14            LDA A  #$14         Code for CA-2 open circuit.
FFØ9   A7 Ø1            STA A  $Ø1,X        Release RDAC.
FFØB   86 34            LDA A  #$34         For RDAC low; CB-1 - edge.
FFØD   6D Ø1            TST    $Ø1,X        Look for /RDAV.
FFØF   2A FC            BPL    *-$Ø2        Loop until received.
FF11   A7 Ø1            STA A  $Ø1,X        Set RDAC low.
FF13   A6 ØØ            LDA A  $ØØ,X        Get input character.
FF15   DE 2A            LDX    MEMLOC       Recover XR.
FF17   39               RTS                 Return.
```

```
*
*SUB restores front-panel control of Time/Div, Span/Div, Phase
*Lock, Resolution Bandwidth and Band (SUB Ø), or puts these
*functions under Control RAM, loaded from Register D (SUB D).
*
FF18   96 2Ø     SUB     LDA A  COMND     Get command.
FF1A   85 ØB             BIT A  #$ØB      Get operand; ignore any C bit.
FF1C   26 ØB             BNE    PLOAD     D, A or B? Skip next.
FF1E   DE 2E             LDX    IOBASE    SUB Ø = restore front-panel.
FF2Ø   E6 Ø7             LDA B  $Ø7,X     Get PIA2B control register.
FF22   CA 38             ORA B  #$38      Set CB-2 bits high.
FF24   E7 Ø7             STA B  $Ø7,X     CB-2 high; control RAM disabled.
FF26   7E FC6B   ACKP    JMP    ACK       Issue ACK; return to COMAND.
*
*PLOAD loads Control RAM with contents of Register D; disables
*five 7L18 front-panel controls and enables Control RAM.
*
FF29   85 Ø8     PLOAD   BIT A  #$Ø8      Check command for SUB D bit.
FF2B   26 Ø7             BNE    PLOAD1    Got one? Do it.
FF2D   7E FFB3           JMP    OUTPUT    No SUB D. Do OUTPUT.
FF3Ø   Ø1                NOP
FF31   Ø1                NOP
FF32   Ø1                NOP
FF33   Ø1                NOP
FF34   CE ØØ5Ø   PLOAD1  LDX    #EMEM     Raw-data register.
FF37   5F                CLR B             B will get mantissa.
FF38   A6 1Ø             LDA A  $1Ø,X     Time/Div from D register
FF3A   46                ROR A             Was: ØØØE EEMM
FF3B   56                ROR B             Mantissa bit from Carry into B.
FF3C   46                ROR A             Shift again, mantissa to Carry.
FF3D   56                ROR B             A: ØØØØ ØEEE; B: MMØØ ØØØØ.
FF3E   54                LSR B             Shift B data to LSD
FF3F   54                LSR B
FF4Ø   54                LSR B
FF41   54                LSR B             B now ØØØØ MMØØ
FF42   A7 ØØ             STA A  $ØØ,X     Time/div exponent in E register.
FF44   A6 11             LDA A  $11,X     Get Span/div: ØØØE EEMM
FF46   84 Ø3             AND A  #$Ø3      Scrub exponent.
FF48   1B                ABA               Assemble mantissas: ØØØØ TTSS
FF49   A7 Ø1             STA A  $Ø1,X     Into E register.
FF4B   A6 11             LDA A  $11,X     Get Span/div again.
FF4D   44                LSR A             Shift out mantissa.
FF4E   44                LSR A             A now: ØØØØ ØEEE
FF4F   AB 12             ADD A  $12,X     Add in phase-lock bit.
FF51   A7 Ø2             STA A  $Ø2,X     Into E register.
FF53   A6 13             LDA A  $13,X     Resolution bandwidth code.
FF55   A7 Ø3             STA A  $Ø3,X     Intoo E register.
FF57   A6 14             LDA A  $14,X     Band switch code.
```

```
FF59   A7 Ø4          STA A  $Ø4,X     Band code into E register.
FF5B   CE ØØ5Ø PLOAD2 LDX    #EMEM     Reload XR for repeat ops.
FF5E   DF 2A          STX    MEMLOC    Set working location.
FF6Ø   C6 Ø5          LDA B  #$Ø5      For transfer of 5 words.
FF62   A6 Ø4    PSHD  LDA A  $Ø4,X     Get data (in inverse order).
FF64   36             PSH A             Onto stack.
FF65   Ø9             DEX               Step
FF66   5A             DEC B             Count
FF67   2E F9          BGT    PSHD       Loop
FF69   DE 2E          LDX    IOBASE     ACC B is now Ø.
FF6B   D7 25    PLOOP STA B  TEMD       B is address counter.
FF6D   A6 Ø4    PLOOP1 LDA A $Ø4,X      Get address & data.
FF6F   84 FØ          AND A  #$FØ       Scrub data.
FF71   E6 Ø4    PLOOP2 LDA B $Ø4,X      Address and data in B.
FF73   C4 FØ          AND B  #$FØ       Clear data.
FF75   11             CBA               Looking for address change.
FF76   27 F9          BEQ    PLOOP2     No change? Keep looking.
FF78   D1 25          CMP B  TEMD       Got a change. Right address?
FF7A   26 F1          BNE    PLOOP1     Not the one we want. Start over.
FF7C   32             PUL A             Right address. Get PSH'd data.
FF7D   A7 1Ø          STA A  $1Ø,X      Address $8n1n enables RAM /WRITE
FF7F   D6 25          LDA B  TEMD       Get preset address.
FF81   CB 1Ø          ADD B  #$1Ø       Next address (in MSD).
FF83   C1 5Ø          CMP B  #$5Ø       Up to address 5?
FF85   25 E4          BCS    PLOOP      If not, go again.
FF87   C6 36          LDA B  #$36       Code for CB-2 low; CB-1 + edge.
FF89   E7 Ø7          STA B  $Ø7,X      To PIA2B. Will enable RAM.
FF8B   5F       CONF  CLR B             Start confirmation routine.
FF8C   DE 2E    CONF1 LDX    IOBASE     Start of I/O area.
FF8E   D7 25          STA B  TEMD       Address $ØØ-$4Ø (Port 1-5).
FF9Ø   A6 Ø6    CONF2 LDA A  $Ø6,X      Clear IRQ.
FF92   A6 Ø7          LDA A  $Ø7,X      Watch for IN pulse.
FF94   2A FC          BPL    *-$Ø2      Loop till you get one.
FF96   A6 Ø4          LDA A  $Ø4,X      Get IN data.
FF98   16             TAB               Replicate in B.
FF99   C4 FØ          AND B  #$FØ       Scrub off data.
FF9B   C1 25          CMP B  #TEMD      Check address.
FF9D   26 F1          BNE    CONF2      Not the one? Loop.
FF9F   DE 2A          LDX    MEMLOC     Got a match. Get E register.
FFA1   84 ØF          AND A  #$ØF       Scrub off address.
FFA3   A1 ØØ          CMP A  $ØØ,X      Compare to E register data.
FFA5   26 B4          BNE    PLOAD2     No match?  Start all over.
FFA7   Ø8             INX               Match.  Step to next E reg loc.
FFA8   DF 2A          STX    MEMLOC     For next operation.
FFAA   CB 1Ø          ADD B  #$1Ø       Step address.
FFAC   C1 5Ø          CMP B  #$5Ø       Past $4Ø?
FFAE   25 EØ          BCS    CONF1      If not, keep checking.
FFBØ   7E FC6B        JMP    ACK        All done & confirmed.
```

```
*
*OUTPUT provides continuous output of B and/or A register data
*to external device via PIA1B, with 6 ms wait between repeats.
*
FFB3   BD FD5A  OUTPUT  JSR    MPREP      Set MEMAD & MEMLOC per ACC A.
FFB6   86 Ø6            LDA A  #$Ø6       ASCII 'ACK'.
FFB8   BD FE76          JSR    FOUT       Output ACK; leave PIA1B set.
FFBB   A6 ØØ            LDA A  $ØØ,X      Clear any old RDAV IRQ.
FFBD   C6 34    NEXOUT  LDA B  #$34       Code for RDAC low, CA-1 -edge.
FFBF   E7 Ø1            STA B  $Ø1,X      Into PIA1A control.
FFC1   DE 28            LDX    MEMAD      Starting point.
FFC3   BD FE9Ø          JSR    MEMO       Output memory up to MEMEND.
FFC6   DE 2E            LDX    IOBASE     Start of I/O area.
FFC8   86 14            LDA A  #$14       Code for RDAC high, CA-1 -edge.
FFCA   5F               CLR B             Counter. Ø = 256.
FFCB   5A       WAIT    DEC B             Counter
FFCC   27 EF            BEQ    NEXOUT     Done? Run another display.
FFCE   A7 Ø1            STA A  $Ø1,X      Set RDAC high. (Helps fill time)
FFDØ   6D Ø1            TST    $Ø1,X      Check for interrupt.
FFD2   2A F7            BPL    WAIT       Not interrupt? Loop for 5.86 ms.
FFD4   6F Ø3    EXOUT   CLR    $Ø3,X      Got interrupt. Open DDR.
FFD6   6F Ø2            CLR    $Ø2,X      8 inputs on B side.
FFD8   8E ØØ4F          LDS    #$ØØ4F     Reset Stack Pointer.
FFDB   C6 34            LDA B  #$34       PIA code for COMND1.
FFDD   7E FC3C          JMP    COMND1     Read new command.
*
*
*
FFEØ   FC       INSTBL  FDB    $FC73      RECORD
FFE1   73
FFE2   FE               FDB    $FE29      READ
FFE3   29
FFE4   FE               FDB    $FEA9      LOAD
FFE5   A9
FFE6   FF               FDB    $FF18      SUB/OUTPUT
FFE7   18
FFE8   FC               FDB    ERRORQ     NOP (Optional instructions)
FFE9   67
FFEA   FC               FDB    ERRORQ     NOP
FFEB   67
FFEC   FC               FDB    ERRORQ     NOP
FFED   67
FFEE   FC               FDB    ERRORQ     NOP
FFEF   67
FFFØ   FC               FDB    ERRORQ     NOP
FFF1   67
FFF2   FC               FDB    ERRORQ     NOP
FFF3   67
```

```
FFF4  FC            FDB   ERRORQ   NOP
FFF5  67
FFF6  FC            FDB   ERRORQ   NOP
FFF7  67
FFF8  FC      IRQ   FDB   COMAND   Maskable interrupt (not used).
FFF9  29
FFFA  FC      SWI   FDB   IOSET    Software interrupt (not used).
FFFB  ØE
FFFC  FC      NMI   FDB   IOSET    Non-maskable interrupt.
FFFD  ØE
FFFE  FC      RESTAR FDB  RESET    Power-on reset.
FFFF  ØØ
```