General Purpose I/O
Parallel/Serial Ports

96 SHEETS • 5 x 5 QUAD
10⅛ x 7⅞ • 53-110

NATIONAL BLANK BOOK COMPANY, INC.
Holyoke, Massachusetts 01040 • Made in USA

General Purpose I/O Chassy

Contains 4 Dual 16 Bit I/O Parallel Ports
with programmable directions + Microprocessor
control

4 Asynchronous Serial Ports
with Microprocessor control for
Baudrate & Modem control

This unit interfaces with the PDP-11 UNIBUS
and requires 128 contiguous bytes for all ports

Vectors are defined from 400_8 - 574_8

29 Aug 81
ASD

UNIBUS Pin Assignments

AA1	INIT L	BA1	BG6
AA2	(+5)	BA2	(+5)
AB1	INTR L	BB1	BG5
AB2	GND	BB2	GND
AC1	DØØ	BC1	BR5
AC2	GND	BC2	GND
AD1	DØ2	BD1	GND
AD2	DØ1	BD2	BR4
AE1	DØ4	BE1	GND
AE2	DØ3	BE2	BG4
AF1	DØ6	BF1	ACLO
AF2	DØ5	BF2	DCLO
AH1	DØ8	BH1	AØ1
AH2	DØ7	BH2	AØØ
AJ1	D10	BJ1	AØ3
AJ2	DØ9	BJ2	AØ2
AK1	D12	BK1	AØ5
AK2	D11	BK2	AØ4
AL1	D14	BL1	AØ7
AL2	D13	BL2	AØ6
AM1	PA	BM1	AØ9
AM2	D15	BM2	AØ8
AN1	GND	BN1	A11
AN2	PB	BN2	A10
AP1	GND	BP1	A13
AP2	BBSY	BP2	A12
AR1	GND	BR1	A15
AR2	SACK	BR2	A14
AS1	GND	BS1	A17
AS2	NPR	BS2	A16
AT1	GND	BT1	GND
AT2	BR7	BT2	C1
AU1	NPG	BU1	SSYN
AU2	BR6	BU2	C0
AV1	BG7	BV1	MSYN
AV2	GND	BV2	GND

29 Aug 81
ARB

50 Pin Data Cable A

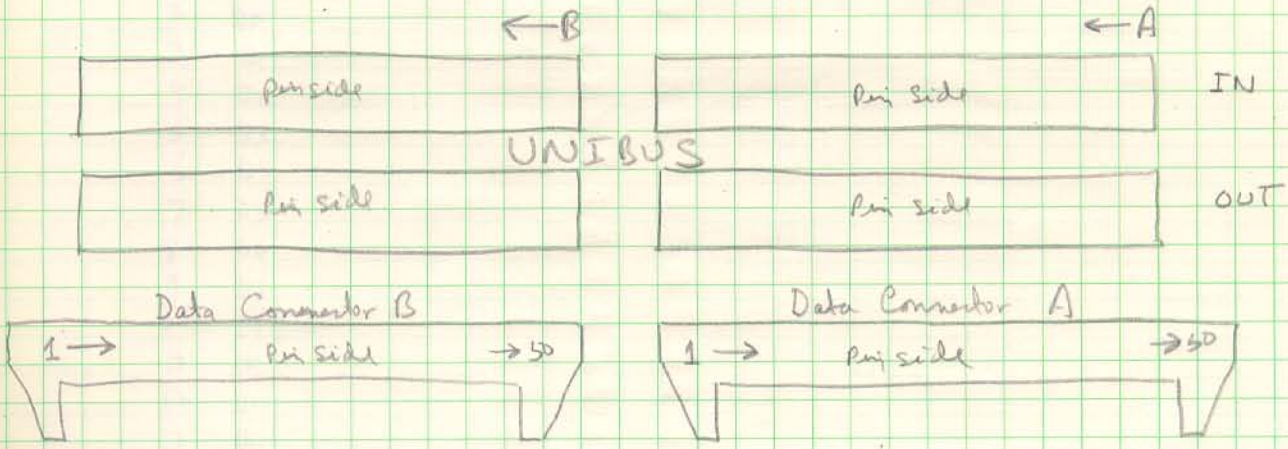
	GND	50	49		GND
	D0	48	47	D1	
	D2	46	45	D3	
	D4	44	43	D5	
	D6	42	41	D7	
	D8	40	39	D9	
	D10	38	37	D11	
	D12	36	35	D13	
	D14	34	33	D15	
	GND	32	31		GND
		30	29	GND	
A2/3	INTR	28	27	GND	
A4/13	INIT	26	25	GND	
A4/15	BBSY	24	23	GND	
A1/3	SACK	22	21	GND	
	:NPR	20	19	GND	
→	NPGI	18	17	GND	
→	NPGO	16	15	GND	
	BR7	14	13	GND	
→	BG7I	12	11	GND	
→	BG7O	10	9	GND	
	BR6	8	7	GND	
→	BG6I	6	5	GND	
→	BG6O	4	3	GND	
	GND	2	1		GND

50 Pin Data Cable B

	GND	50	49	GND	
	BRS	48	47	GND	
	BGSI	46	45	GND	
	BGSO	44	43	GND	
A2/6	BR4	42	41	GND	
A4/11	BG4I	40	39	GND	
A3/2	BG4O	38	37	GND	
	ACLOI	36	35	GND	
	+5V	34	33	GND	
	DCLOI	32	31	GND	
	+5V	30	28	GND	
	A0	28	27	A1	
	A2	26	25	A3	
	A4	24	23	A5	
	A6	22	21	A7	
	A8	20	19	A9	
	A10	18	17	A11	
	A12	16	15	A13	
	A14	14	13	A15	
	A16	12	11	A17	
	GND	10	9	GND	
	C0	8	7	GND	
	C1	6	5	GND	
	SSYN	4	3	GND	
	MSYN	2	1	GND	
			*		

29 Aug 81
ABD

Rear Panel Connector Location —
drawn from inside chassy view



These mount on a PC Board on the Rear Panel

29 Aug 81
ABB

Chassy BUS Connectors PC1

1		GND
2	D0	
3	D1	
4	D2	
5	D3	
6	D4	
7	D5	
8	D6	
9	D7	
10	D8	
11	DA	
12	D10	
13	D11	
14	D12	
15	D13	
16	D14	
17	D15	
18		GND
19		GND
20	~~BCφ~~	
21	$\overline{BC1}$	
22	A0	
23	A1	
24	A2	
25	A3	
26	A4	
27	A5	
28	A6	
29	A7	
30		GND
31	INIT	L
32		GND
33	E(φ2)	
34		GND
35	$\overline{CSD}$	
36		GND

30 Aug 81
ARO

Chassy Power Connector PC2

1		GND	
2		GND	
3		GND	Power GND
4		GND	
5		GND	
6	+5		
7	+5		Power (+5)
8	+5		
9	+5		
10		GND	
11	+12		Power (+12)
12		GND	
13	-12		Power (-12)
14		GND	
15	EX1		
16		GND	
17	EX2		
18		GND	
19	EX3		
20		GND	
21		GND	

Board

P1	H
P1	H
P1	H
P1	H
P1	H
P1	H
P1	H
P1	H
P2	H
P2	H
P2	H
P2	H
P2	H
P2	H
P2	H
P2	H
S1	F
S1	F
S1	F
S1	F
S1	F
S1	F
S1	F
S1	F
S2	F
S2	F
S2	F
S2	F
S2	F
S2	F
S2	F
S2	F

Chassy Interupts PC3 (IRQ chart)

Board #

P1	H8/38	1	PIH1	Parrallel Input #1 Highest Priority
P1	H6/38	2	PIL1	
P1	H8/37	3	POH1	
P1	H6/37	4	POL1	
P1	H3/38	5	PIH2	
P1	H1/38	6	PIL2	
P1	H3/37	7	POH2	
P1	H1/37	8	POL2	
		9	GND	
P2	H8/38	10	PIH3	Parrallel Input #3
P2	H6/38	11	PIL3	
P2	H8/37	12	POH3	
P2	H6/37	13	POL3	
P2	H3/38	14	PIH4	
P2	H1/38	15	PIL4	
P2	H3/37	16	POH4	
P2	H1/37	17	POL4	
		18	GND	
S1	F14/38	19	SIH1	Serial Input #1
S1	F10/38	20	SIL1	
S1	F14/37	21	SOH1	
S1	F10/37	22	SOL1	
S1	F6/38	23	SIH2	
S1	F2/38	24	SIL2	
S1	F6/37	25	SOH2	
S1	F2/37	26	SOL2	
		27	GND	
S2	F14/38	28	SIH3	Serial Input #3
S2	F10/38	29	SIL3	
S2	F14/37	30	SOH3	
S2	F10/37	31	SOL3	
S2	F6/38	32	SIH4	
S2	F2/38	33	SIL4	
S2	F6/37	34	SOH4	
S2	F2/37	35	SOL4	Lowest Priority
		36	GND	

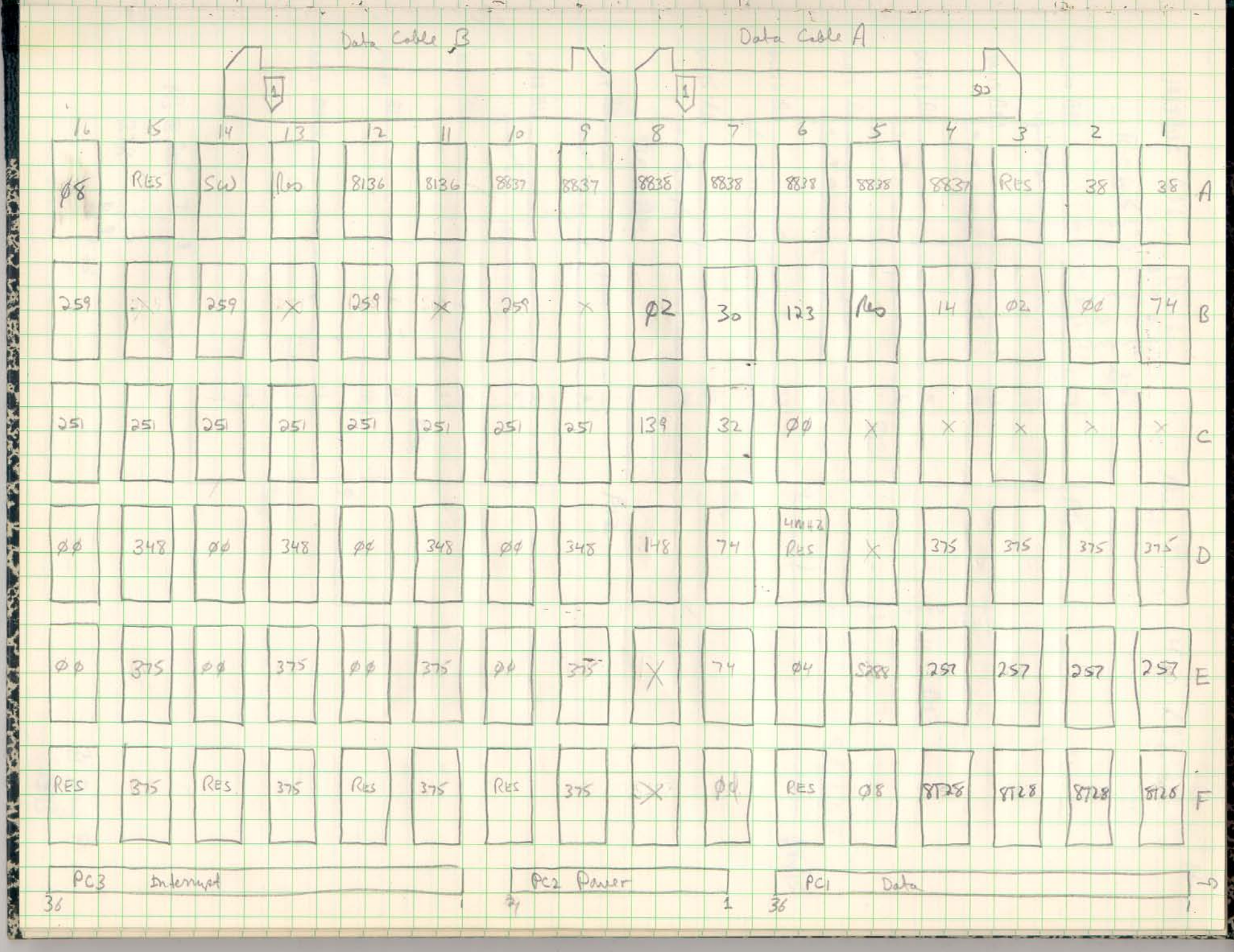
30 Aug 87
ABB

BUS INterface & Interrupt Control Card

1. UNIBUS to Chaosy Bus translator
2. Device Status Registers for each of
   the 4 Parallel I/O Ports and
   4 Serial I/O Ports
3. Interrupt priority logic for 32 levels
   of Interrupting devices (each byte of ports)

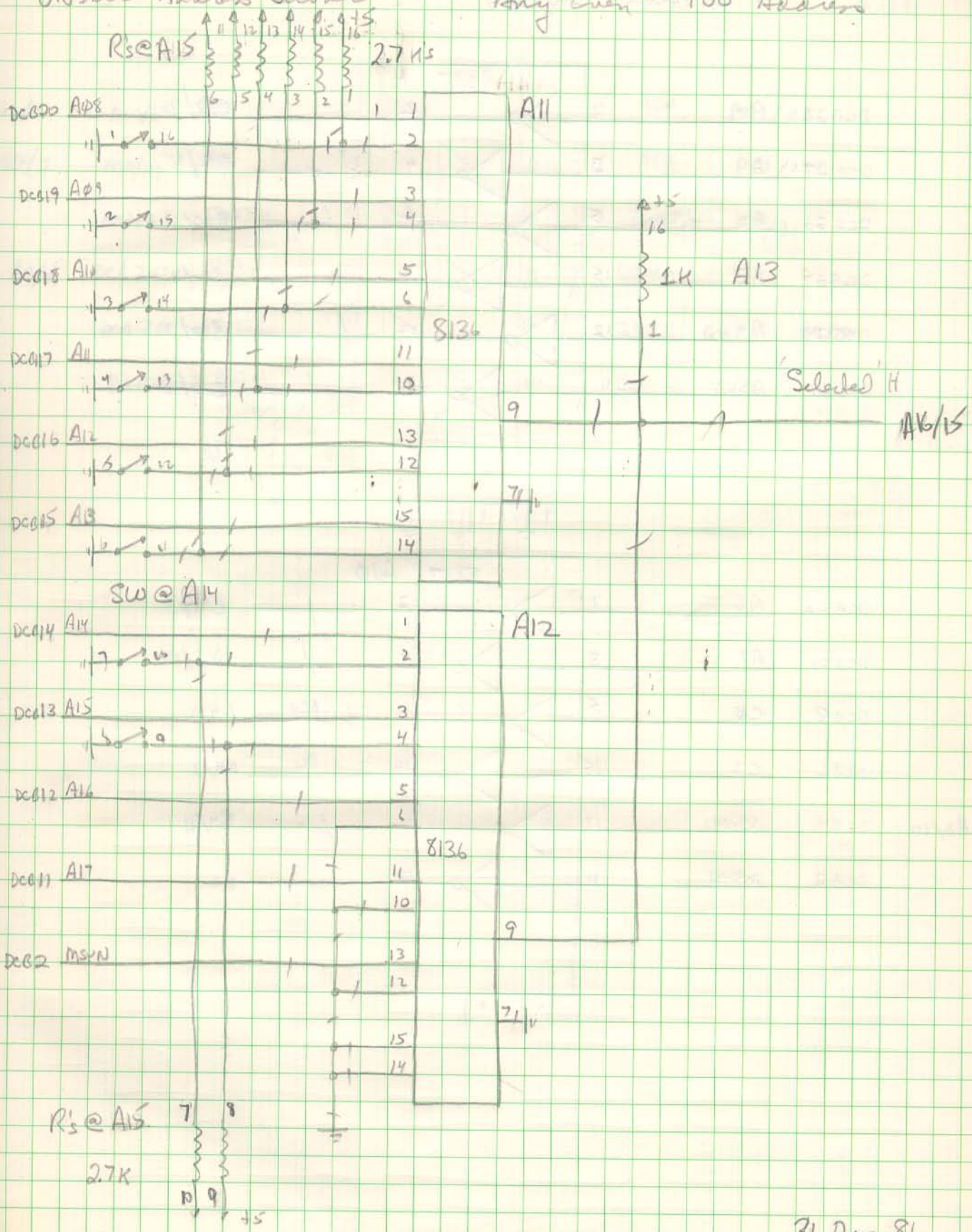
30 Aug 81
ADB

See pgs 28-29 for IC Location Index for this Circuit Card.


Data Cable B
Data Cable A
1
50
16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
A
B
C
D
E
F
PC3 Interrupt
PC2 Power
PC1 Data
36
1
34
1
36
1

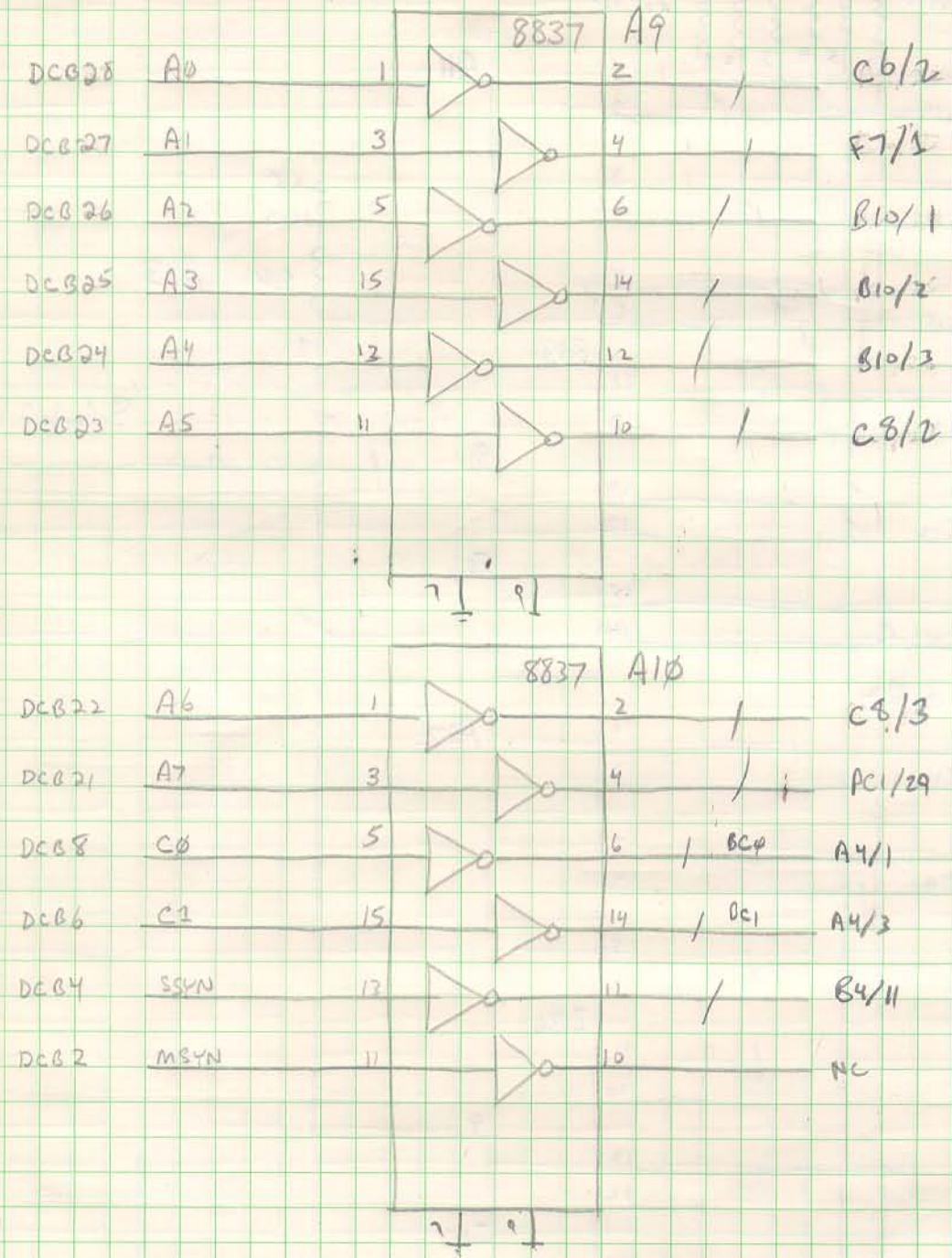
UNIBUS Address Decoder

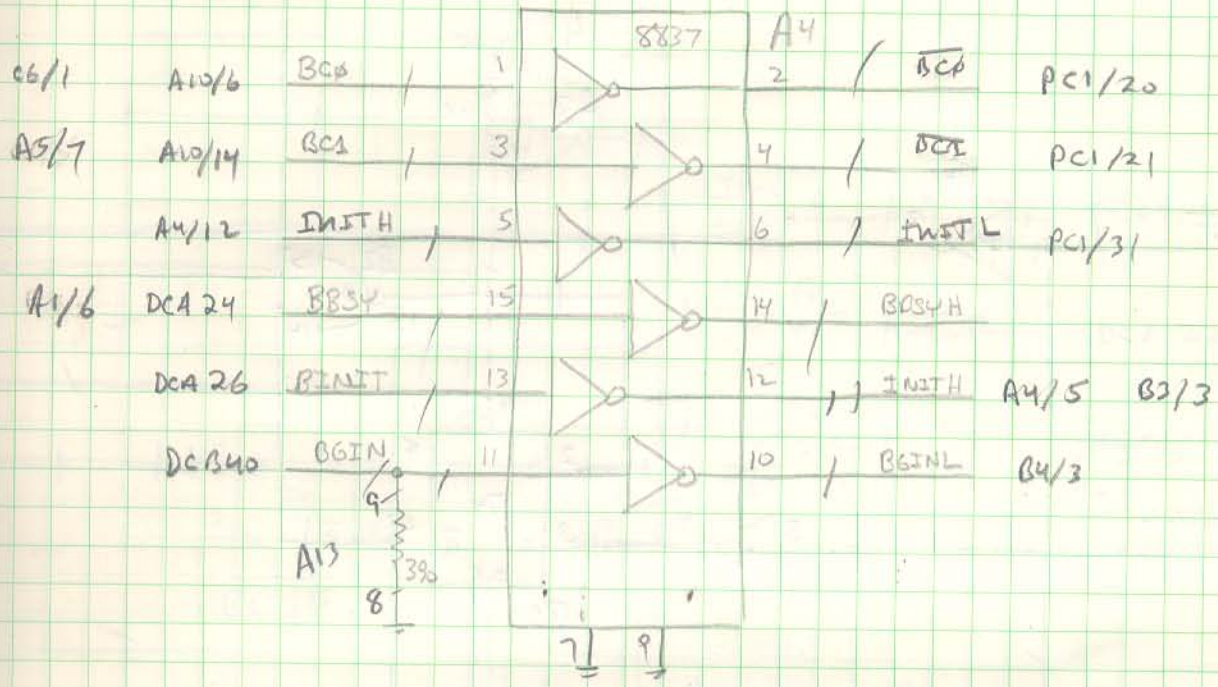
Any Even 400 Address



31 Aug 81
ARS

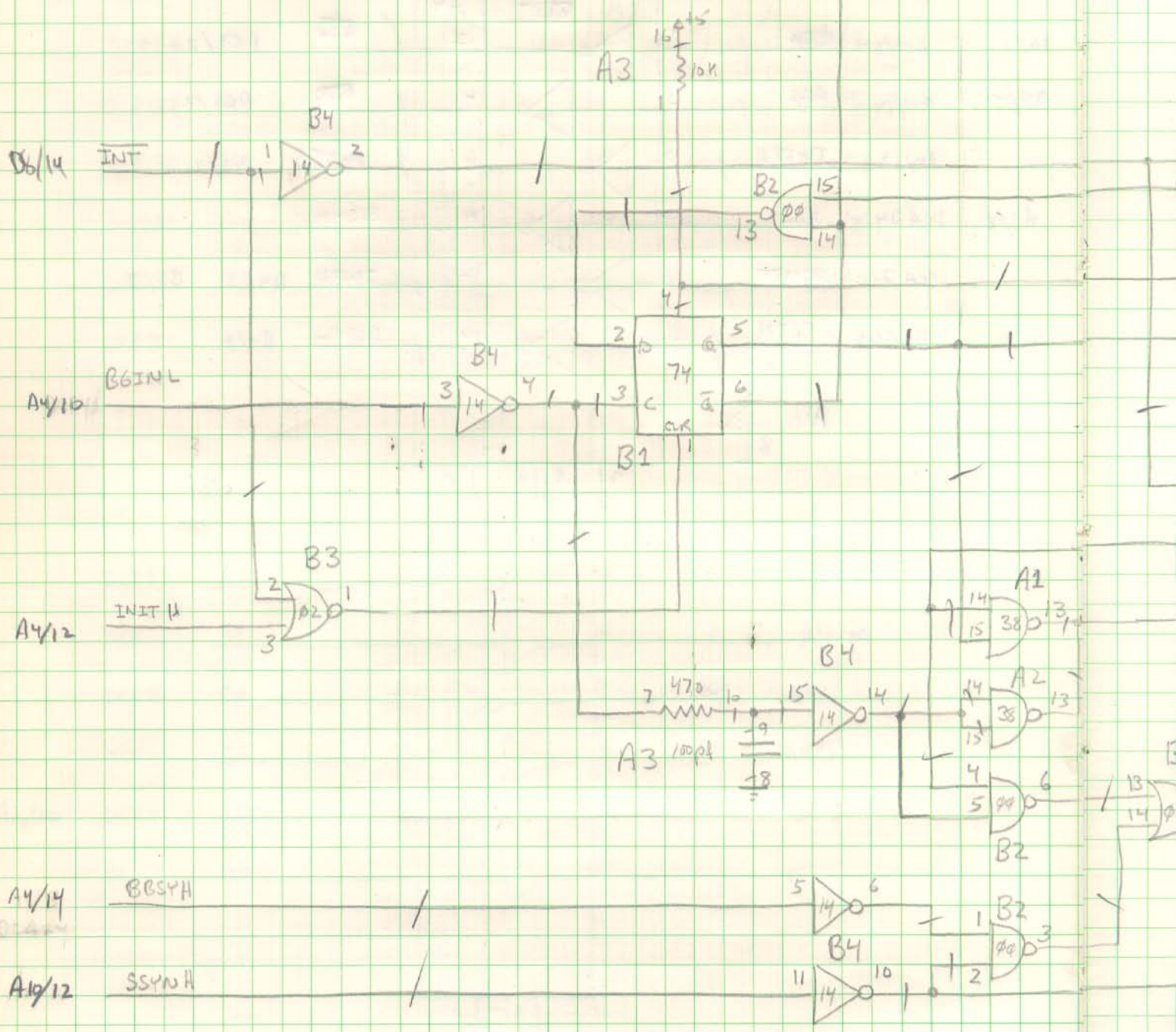
Address & Control Buffers

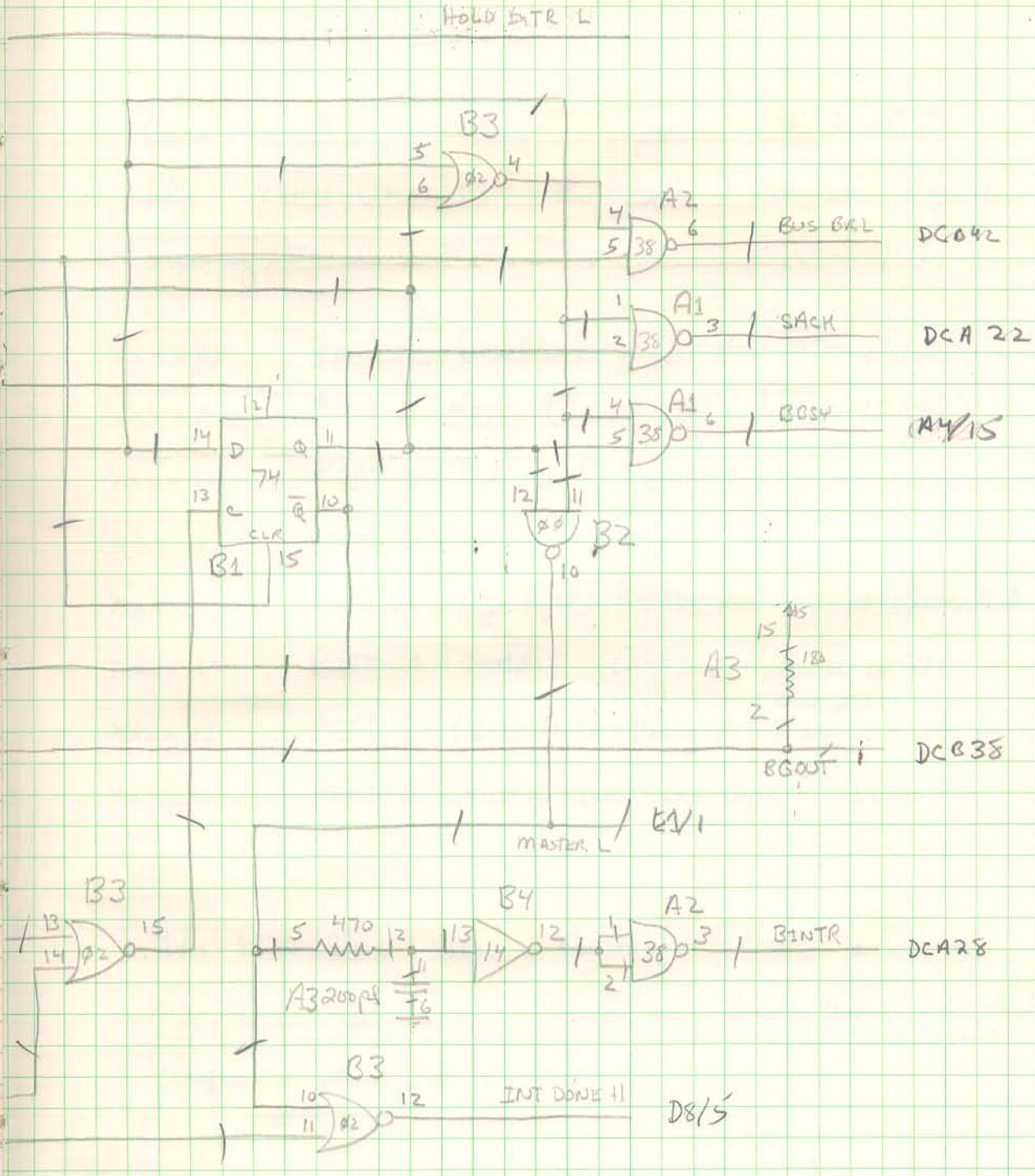



8837
A4
C6/1
A10/6
BC0
1
2
BC0
PC1/20
A5/7
A10/14
BC1
3
4
BC1
PC1/21
A4/12
INITH
5
6
INITL
PC1/31
A1/6
DCA24
BBSY
15
14
BBSYH
DCA26
BINIT
13
12
INITH
A4/5
B3/3
DCB40
BGIN
11
10
BGINL
B4/3
9
A13
390
8
7
9

31 Aug 87
ABB

UNIBUS INTERRUPT CONTROLLER




HOLD INTR L
B3
A2
BUS BRL
DC042
A1
SACK
DCA22
A1
BUSY
A4/15
74
B1
B2
A3
180
BGOUT
DCB38
MASTER L
B3
B4
A2
470
BINTR
DCA28
A3 200pf
B3
INT DONE H
D8/5

2 Sept 81
ARB

General PORT Description -

Each serial/parallel port appears identical to the PDP-11 UNIBUS. The control and data register organization is noted below.

		Interrupt Priority
xxxx ØØ	PORT A (LOW BYTE) READY AND Interrupt enable bits	2
xxxx Ø2	PORT A DATA (Low & High Bytes as 16 Bits)	
xxxx Ø4	PORT B (Low Byte) READY and Interrupt enable Bits	4
xxxx Ø6	PORT B Data (Low & High Bytes as 16 Bits)	
xxxx 1Ø	PORT A (High Byte) READY and Interrupt enable Bits	1
xxxx 12	PORT A CONTROL (Low & High Bytes as 16 Bits)	
xxxx 14	PORT B (High Byte) Ready and Interrupt enable Bits	3
xxxx 16	PORT B CONTROL (Low & High Bytes as 16 Bits)	

The PORT A & B Data & Control Registers are the MC 6821 Registers - These are described in the following pages by the Data sheets.

3 Sept 81
ABB

FIGURE 16 – $\overline{IRQ}$ RELEASE TIME

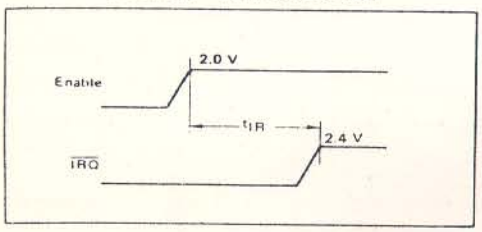
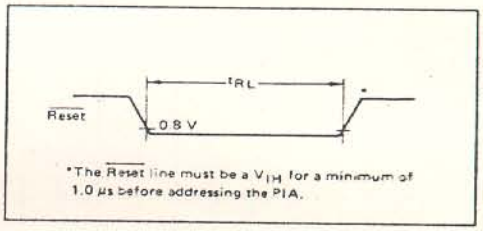
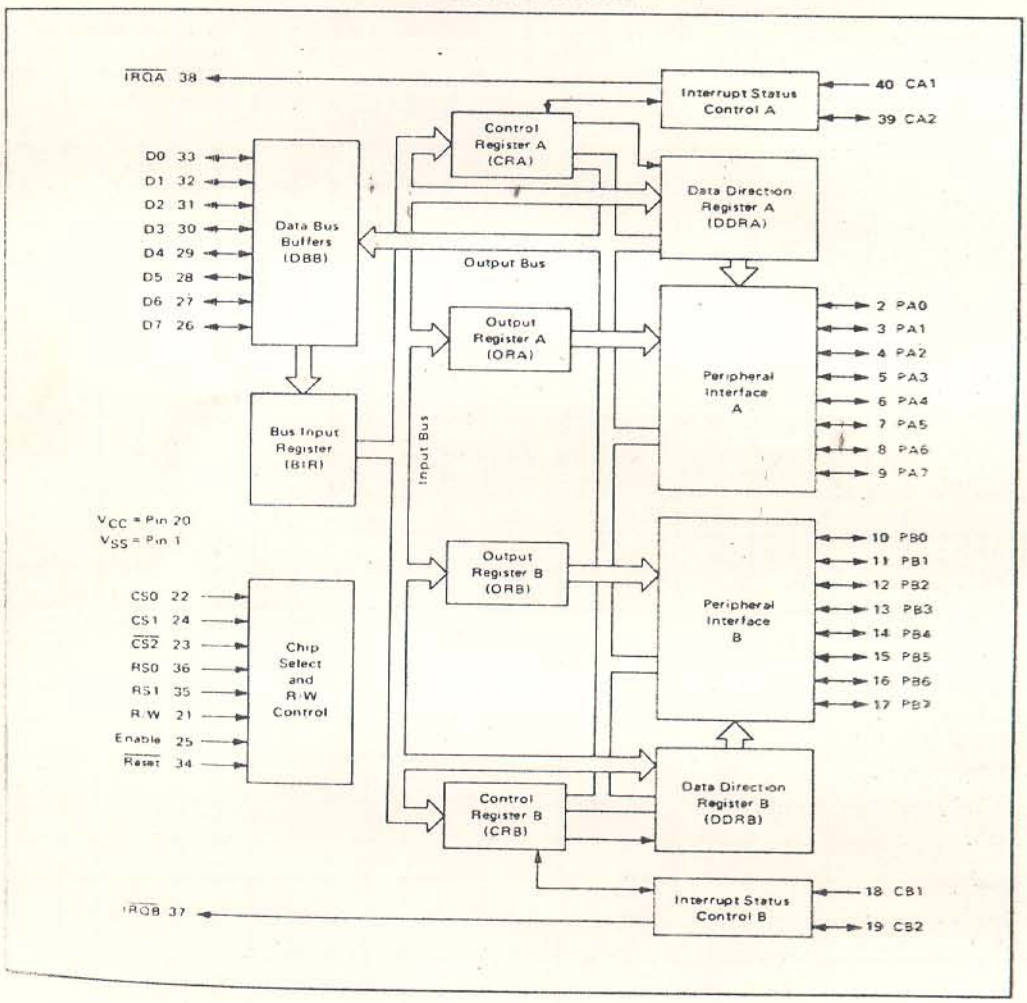


FIGURE 17 – $\overline{RESET}$ LOW TIME



*The $\overline{Reset}$ line must be a V_{IH} for a minimum of 1.0 μs before addressing the PIA.

EXPANDED BLOCK DIAGRAM



PIA INTERFACE SIGNALS FOR MPU

The PIA interfaces to the MC6800 MPU with an eight-bit bi-directional data bus, three chip select lines, two register select lines, two interrupt request lines, read/write line, enable line and reset line. These signals, in conjunction with the MC6800 VMA output, permit the MPU to have complete control over the PIA. VMA should be utilized in conjunction with an MPU address line into a chip select of the PIA.

PIA Bi-Directional Data (D0-D7) – The bi-directional data lines (D0-D7) allow the transfer of data between the MPU and the PIA. The data bus output drivers are three-state devices that remain in the high-impedance (off) state except when the MPU performs a PIA read operation. The Read/Write line is in the Read (high) state when the PIA is selected for a Read operation.

PIA Enable (E) – The enable pulse, E, is the only timing signal that is supplied to the PIA. Timing of all other signals is referenced to the leading and trailing edges of the E pulse. This signal will normally be a derivative of the MC6800 $\phi 2$ Clock.

PIA Read/Write (R/W) – This signal is generated by the MPU to control the direction of data transfers on the Data Bus. A low state on the PIA Read/Write line enables the input buffers and data is transferred from the MPU to the PIA on the E signal if the device has been selected. A high on the Read/Write line sets up the PIA for a transfer of data to the bus. The PIA output buffers are enabled when the proper address and the enable pulse E are present.

$\overline{\text{Reset}}$ – The active low $\overline{\text{Reset}}$ line is used to reset all register bits in the PIA to a logical zero (low). This line can be used as a power-on reset and as a master reset during system operation.

PIA Chip Select (CS0, CS1 and $\overline{\text{CS2}}$) – These three input signals are used to select the PIA. CS0 and CS1 must be high and $\overline{\text{CS2}}$ must be low for selection of the device. Data transfers are then performed under the control of the Enable and Read/Write signals. The chip select lines must be stable for the duration of the E pulse. The device is deselected when any of the chip selects are in the inactive state.

PIA Register Select (RS0 and RS1) – The two register select lines are used to select the various registers inside the PIA. These two lines are used in conjunction with internal Control Registers to select a particular register that is to be written or read.

The register and chip select lines should be stable for the duration of the E pulse while in the read or write cycle.

Interrupt Request ($\overline{\text{IRQA}}$ and $\overline{\text{IRQB}}$) – The active low Interrupt Request lines ($\overline{\text{IRQA}}$ and $\overline{\text{IRQB}}$) act to interrupt the MPU either directly or through interrupt priority circuitry. These lines are "open drain" (no load device on the chip). This permits all interrupt request lines to be tied together in a wire-OR configuration.

Each Interrupt Request line has two internal interrupt flag bits that can cause the Interrupt Request line to go low. Each flag bit is associated with a particular peripheral interrupt line. Also four interrupt enable bits are provided in the PIA which may be used to inhibit a particular interrupt from a peripheral device.

Servicing an interrupt by the MPU may be accomplished by a software routine that, on a prioritized basis, sequentially reads and tests the two control registers in each PIA for interrupt flag bits that are set.

The interrupt flags are cleared (zeroed) as a result of an MPU Read Peripheral Data Operation of the corresponding data register. After being cleared, the interrupt flag bit cannot be enabled to be set until the PIA is deselected during an E pulse. The E pulse is used to condition the interrupt control lines (CA1, CA2, CB1, CB2). When these lines are used as interrupt inputs at least one E pulse must occur from the inactive edge to the active edge of the interrupt input signal to condition the edge sense network. If the interrupt flag has been enabled and the edge sense circuit has been properly conditioned, the interrupt flag will be set on the next active transition of the interrupt input pin.

MC6821 Data Sheets

PIA PERIPHERAL INTERFACE LINES

The PIA provides two 8-bit bi-directional data buses and four interrupt/control lines for interfacing to peripheral devices.

Section A Peripheral Data (PA0-PA7) – Each of the peripheral data lines can be programmed to act as an input or output. This is accomplished by setting a "1" in the corresponding Data Direction Register bit for those lines which are to be outputs. A "0" in a bit of the Data Direction Register causes the corresponding peripheral data line to act as an input. During an MPU Read Peripheral Data Operation, the data on peripheral lines programmed to act as inputs appears directly on the corresponding MPU Data Bus lines. In the input mode the internal pullup resistor on these lines represents a maximum of 1.5 standard TTL loads.

The data in Output Register A will appear on the data lines that are programmed to be outputs. A logical "1" written into the register will cause a "high" on the corresponding data line while a "0" results in a "low". Data in Output Register A may be read by an MPU "Read Peripheral Data A" operation when the corresponding lines are programmed as outputs. This data will be read properly if the voltage on the peripheral data lines is greater than 2.0 volts for a logic "1" output and less than 0.8 volt for a logic "0" output. Loading the output lines such that the voltage on these lines does not reach full voltage causes the data transferred into the MPU on a Read operation to differ from that contained in the respective bit of Output Register A.

Section B Peripheral Data (PB0-PB7) – The peripheral data lines in the B Section of the PIA can be programmed to act as either inputs or outputs in a similar manner to PA0-PA7. However, the output buffers driving these lines differ from those driving lines PA0-PA7. They have three-state capability, allowing them to enter a high impedance state when the peripheral data line is used as an input. In addition, data on the peripheral data lines PB0-PB7 will be read properly from those lines programmed as outputs even if the voltages are below 2.0 volts for a "high". As outputs, these lines are compatible with standard TTL and may also be used as a source of up to 1 milliampere at 1.5 volts to directly drive the base of a transistor switch.

Interrupt Input (CA1 and CB1) – Peripheral Input lines CA1 and CB1 are input only lines that set the interrupt flags of the control registers. The active transition for these signals is also programmed by the two control registers.

Peripheral Control (CA2) – The peripheral control line CA2 can be programmed to act as an interrupt input or as a peripheral control output. As an output, this line is compatible with standard TTL; as an input the internal pullup resistor on this line represents 1.5 standard TTL loads. The function of this signal line is programmed with Control Register A.

Peripheral Control (CB2) – Peripheral Control line CB2 may also be programmed to act as an interrupt input or peripheral control output. As an input, this line has high input impedance and is compatible with standard TTL. As an output it is compatible with standard TTL and may also be used as a source of up to 1 milliampere at 1.5 volts to directly drive the base of a transistor switch. This line is programmed by Control Register B.

INTERNAL CONTROLS

There are six locations within the PIA accessible to the MPU data bus: two Peripheral Registers, two Data Direction Registers, and two Control Registers. Selection of these locations is controlled by the RS0 and RS1 inputs together with bit 2 in the Control Register, as shown in Table 1.

TABLE 1 – INTERNAL ADDRESSING

RS1	RS0	Control Register Bit CRA-2	Control Register Bit CRB-2	Location Selected
0	0	1	X	Peripheral Register A
0	0	0	X	Data Direction Register A
0	1	X	X	Control Register A
1	0	X	1	Peripheral Register B
1	0	X	0	Data Direction Register B
1	1	X	X	Control Register B

X = Don't Care

INITIALIZATION

A low reset line has the effect of zeroing all PIA registers. This will set PA0-PA7, PB0-PB7, CA2 and CB2 as inputs, and all interrupts disabled. The PIA must be configured during the restart program which follows the reset.

Details of possible configurations of the Data Direction and Control Register are as follows.

DATA DIRECTION REGISTERS (DDRA and DDRB)

The two Data Direction Registers allow the MPU to control the direction of data through each corresponding peripheral data line. A Data Direction Register bit set at "0" configures the corresponding peripheral data line as an input; a "1" results in an output.

CONTROL REGISTERS (CRA and CRB)

The two Control Registers (CRA and CRB) allow the MPU to control the operation of the four peripheral control lines CA1, CA2, CB1 and CB2. In addition they allow the MPU to enable the interrupt lines and monitor the status of the interrupt flags. Bits 0 through 5 of the two registers may be written or read by the MPU when the proper chip select and register select signals are applied. Bits 6 and 7 of the two registers are read only and are modified by external interrupts occurring on control lines CA1, CA2, CB1 or CB2. The format of the control words is shown in Table 2

TABLE 2 – CONTROL WORD FORMAT

	7	6	5	4	3	2	1	0
CRA	IRQA1	IRQA2	CA2 Control			DDRA Access	CA1 Control	
CRB	IRQB1	IRQB2	CB2 Control			DDRB Access	CB1 Control	

Data Direction Access Control Bit (CRA-2 and CRB-2) – Bit 2 in each Control register (CRA and CRB) allows selection of either a Peripheral Interface Register or the Data Direction Register when the proper register select signals are applied to RS0 and RS1.

Interrupt Flags (CRA-6, CRA-7, CRB-6, and CRB-7) – The four interrupt flag bits are set by active transitions of signals on the four Interrupt and Peripheral Control lines when those lines are programmed to be inputs. These bits cannot be set directly from the MPU Data Bus and are reset indirectly by a Read Peripheral Data Operation on the appropriate section.

TABLE 3 – CONTROL OF INTERRUPT INPUTS CA1 AND CB1

CRA-1 (CRB-1)	CRA-0 (CRB-0)	Interrupt Input CA1 (CB1)	Interrupt Flag CRA-7 (CRB-7)	MPU Interrupt Request $\overline{IRQA}$ ($\overline{IRQB}$)
0	0	↓ Active	Set high on ↓ of CA1 (CB1)	Disabled — $\overline{IRQ}$ remains high
0	1	↓ Active	Set high on ↓ of CA1 (CB1)	Goes low when the interrupt flag bit CRA-7 (CRB-7) goes high
1	0	↑ Active	Set high on ↑ of CA1 (CB1)	Disabled — $\overline{IRQ}$ remains high
1	1	↑ Active	Set high on ↑ of CA1 (CB1)	Goes low when the interrupt flag bit CRA-7 (CRB-7) goes high

Notes: 1 ↑ indicates positive transition (low to high)

2 ↓ indicates negative transition (high to low)

3 The interrupt flag bit CRA-7 is cleared by an MPU Read of the A Data Register, and CRB-7 is cleared by an MPU Read of the B Data Register

4 If CRA-0 (CRB-0) is low when an interrupt occurs (Interrupt disabled) and is later brought high, $\overline{IRQA}$ ($\overline{IRQB}$) occurs after CRA-0 (CRB-0) is written to a "one".

Control of CA1 and CB1 Interrupt Input Lines (CRA-0, CRB-0, CRA-1, and CRB-1) — The two lowest order bits of the control registers are used to control the interrupt input lines CA1 and CB1. Bits CRA-0 and CRB-0 are used to enable the MPU interrupt signals $\overline{IRQA}$ and $\overline{IRQB}$, respectively. Bits CRA-1 and CRB-1 determine the active transition of the interrupt input signals CA1 and CB1 (Table 3).

TABLE 4 – CONTROL OF CA2 AND CB2 AS INTERRUPT INPUTS
CRA5 (CRB5) is low

CRA-5 (CRB-5)	CRA-4 (CRB-4)	CRA-3 (CRB-3)	Interrupt Input CA2 (CB2)	interrupt Flag CRA-6 (CRB-6)	MPU Interrupt Request $\overline{IRQA}$ ($\overline{IRQB}$)
0	0	0	↓ Active	Set high on ↓ of CA2 (CB2)	Disabled — $\overline{IRQ}$ remains high
0	0	1	↓ Active	Set high on ↓ of CA2 (CB2)	Goes low when the interrupt flag bit CRA-6 (CRB-6) goes high
0	1	0	↑ Active	Set high on ↑ of CA2 (CB2)	Disabled — $\overline{IRQ}$ remains high
0	1	1	↑ Active	Set high on ↑ of CA2 (CB2)	Goes low when the interrupt flag bit CRA-6 (CRB-6) goes high

Notes: 1. ↑ indicates positive transition (low to high)

2. ↓ indicates negative transition (high to low)

3. The interrupt flag bit CRA-6 is cleared by an MPU Read of the A Data Register and CRB-6 is cleared by an MPU Read of the B Data Register

4. If CRA-3 (CRB-3) is low when an interrupt occurs (interrupt disabled) and is later brought high, $\overline{IRQA}$ ($\overline{IRQB}$) occurs after CRA-3 (CRB-3) is written to a "one"

TABLE 5 – CONTROL OF CB2 AS AN OUTPUT
CRB-5 is high

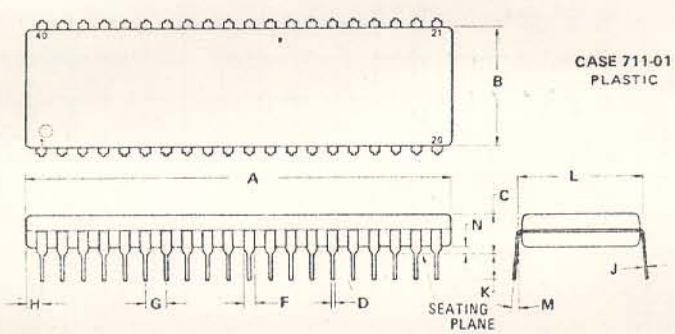
CRB-5	CRB-4	CRB-3	CB2 Cleared	CB2 Set
1	0	0	Low on the positive transition of the first E pulse following an MPU Write "B" Data Register operation	High when the interrupt flag bit CRB-7 is set by an active transition of the CB1 signal
1	0	1	Low on the positive transition of the first E pulse after an MPU Write "B" Data Register operation.	High on the positive edge of the first "E" pulse following an "E" pulse which occurred while the part was deselected
1	1	0	Low when CRB-3 goes low as a result of an MPU Write in Control Register "B"	Always low as long as CRB-3 is low. Will go high on an MPU Write in Control Register "B" that changes CRB-3 to one
1	1	1	Always high as long as CRB-3 is high. Will be cleared when an MPU Write Control Register "B" results in clearing CRB-3 to zero	High when CRB-3 goes high as a result of an MPU Write into Control Register "B".

Control of CA2 and CB2 Peripheral Control Lines (CRA-3, CRA-4, CRA-5, CRB-3, CRB-4, and CRB-5) – Bits 3, 4, and 5 of the two control registers are used to control the CA2 and CB2 Peripheral Control lines. These bits determine if the control lines will be an interrupt input or an output control signal. If bit CRA-5 (CRB-5) is low, CA2 (CB2) is an interrupt input line similar to CA1 (CB1) (Table 4). When CRA-5 (CRB-5) is high, CA2 (CB2) becomes an output signal that may be used to control peripheral data transfers. When in the output mode, CA2 and CB2 have slightly different characteristics (Tables 5 and 6).

TABLE 6 – CONTROL OF CA-2 AS AN OUTPUT
CRA-5 is high

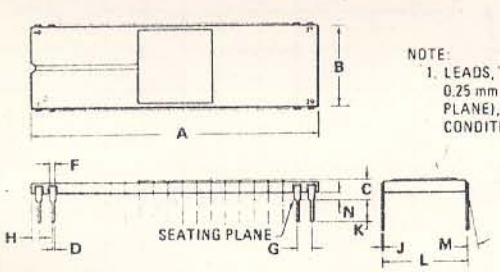
CRA-5	CRA-4	CRA-3	CA2 Cleared	CA2 Set
1	0	0	Low on negative transition of E after an MPU Read "A" Data operation.	High when the interrupt flag bit CRA-7 is set by an active transition of the CA1 signal.
1	0	1	Low on negative transition of E after an MPU Read "A" Data operation.	High on the negative edge of the first "E" pulse which occurs during a deselect.
1	1	0	Low when CRA-3 goes low as a result of an MPU Write to Control Register "A".	Always low as long as CRA-3 is low. Will go high on an MPU Write to Control Register "A" that changes CRA-3 to "one".
1	1	1	Always high as long as CRA-3 is high. Will be cleared on an MPU Write to Control Register "A" that clears CRA-3 to a "zero".	High when CRA-3 goes high as a result of an MPU Write to Control Register "A".

PACKAGE DIMENSIONS



CASE 711-01
PLASTIC

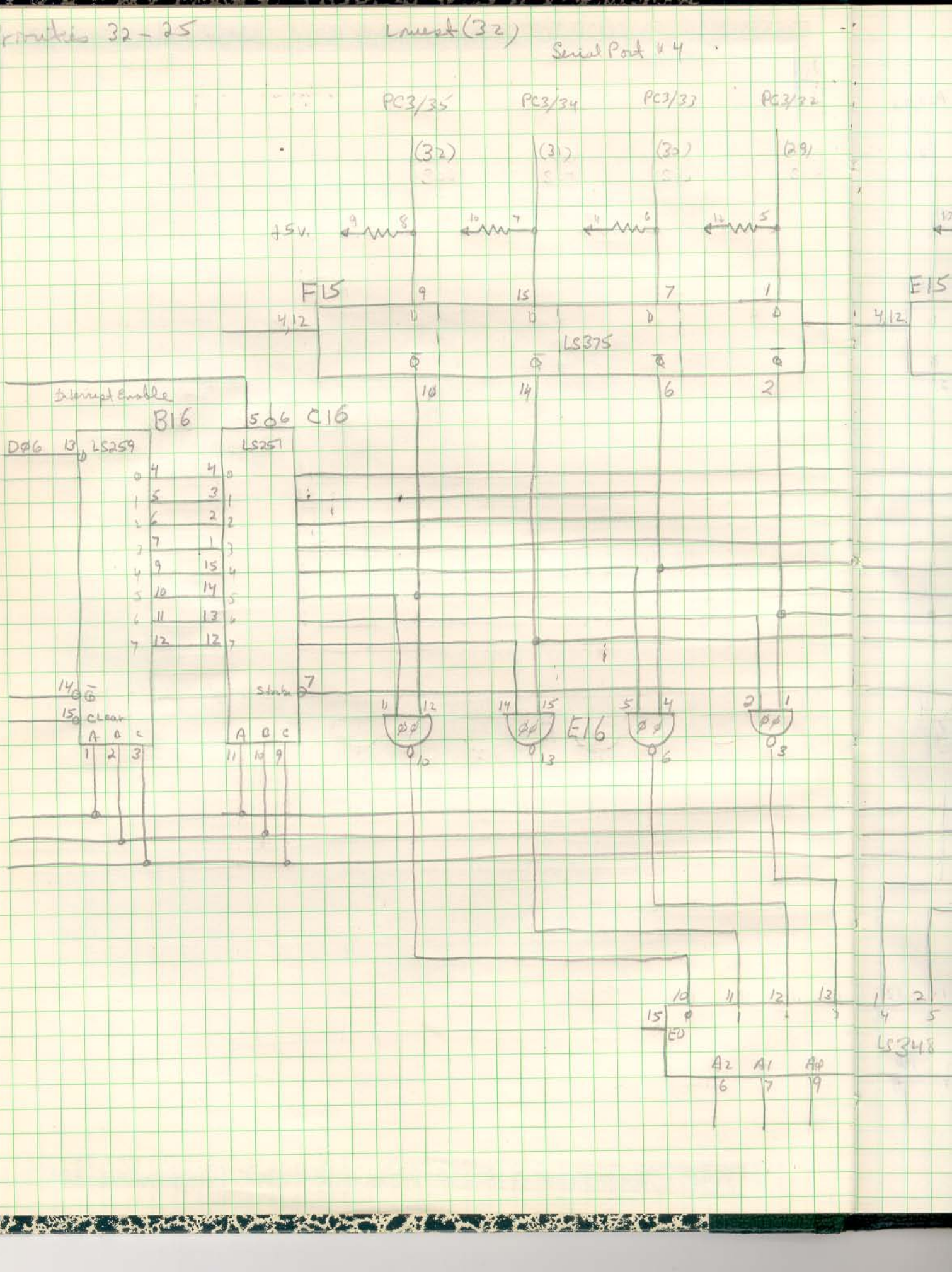
DIM	MILLIMETERS MIN	MILLIMETERS MAX	INCHES MIN	INCHES MAX
A	51.82	52.32	2.040	2.060
B	13.72	14.22	0.540	0.560
C	4.57	5.08	0.180	0.200
D	0.36	0.51	0.014	0.020
F	1.02	1.52	0.040	0.060
G	2.41	2.67	0.095	0.105
H	1.65	2.16	0.065	0.085
J	0.20	0.30	0.008	0.012
K	3.68	4.19	0.145	0.165
L	14.99	15.49	0.590	0.610
M	0°	10°	0°	10°
N	0.51	1.02	0.020	0.040



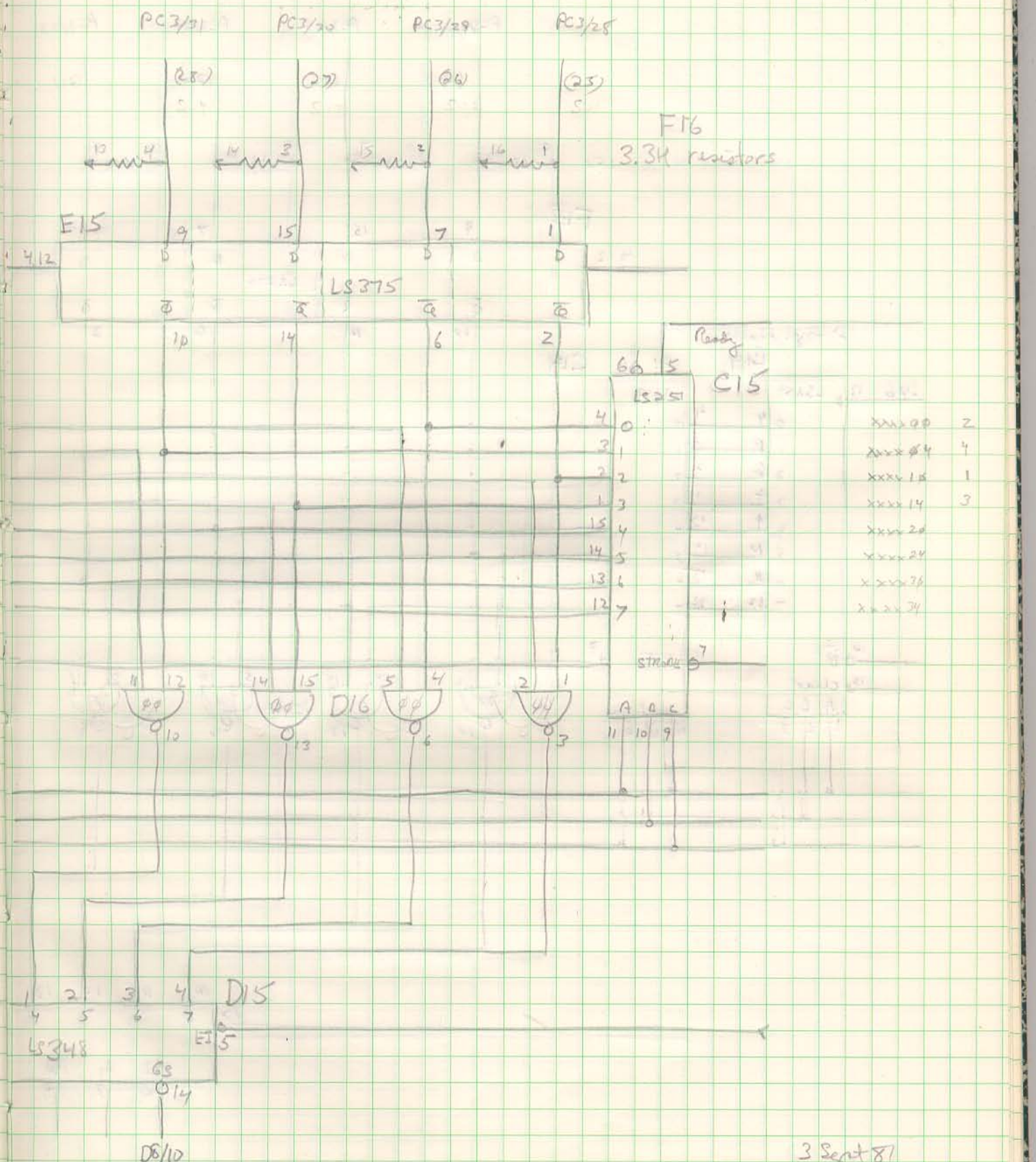
NOTE:
1. LEADS, TRUE POSITIONED WITHIN 0.25 mm (0.010) DIA (AT SEATING PLANE), AT MAX. MAT'L CONDITION.

CASE 715-02
CERAMIC

DIM	MILLIMETERS MIN	MILLIMETERS MAX	INCHES MIN	INCHES MAX
A	50.29	51.31	1.980	2.020
B	14.86	15.62	0.585	0.615
C	2.54	4.19	0.100	0.165
D	0.38	0.53	0.015	0.021
F	0.76	1.40	0.030	0.055
G	2.54 BSC		0.100 BSC	
H	0.76	1.78	0.030	0.070
J	0.20	0.33	0.008	0.013
K	2.54	4.19	0.100	0.165
L	14.60	15.37	0.575	0.605
M	–	10°	–	10°
N	0.51	1.52	0.020	0.060

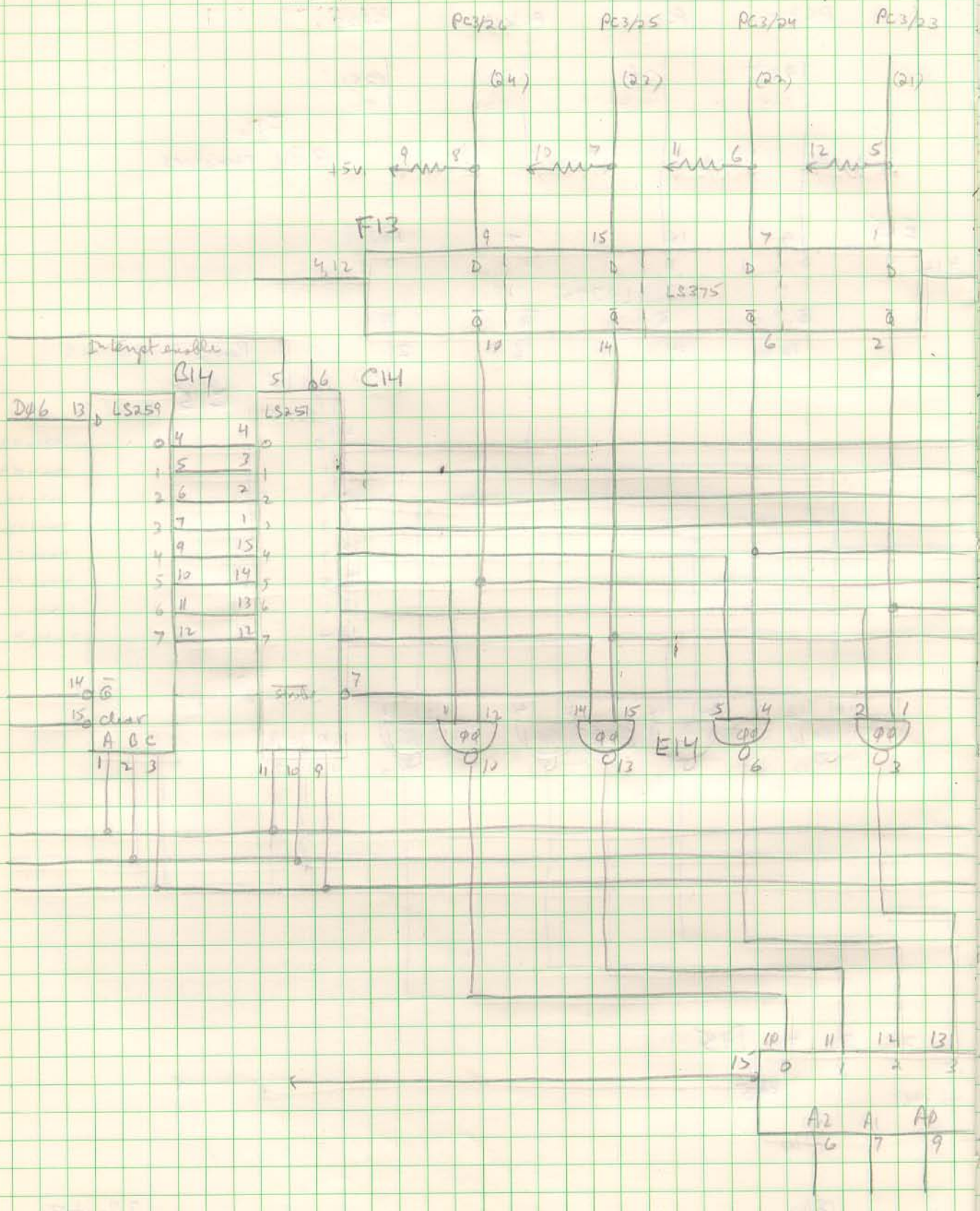

Printer 32-25
Lmsgt (32)
Serial Port 4 4
PC3/35
PC3/34
PC3/33
PC3/32
(32)
(31)
(30)
(29)
+5V.
F15
LS375
E15
4,12
Interrupt Enable
B16
C16
D06
LS259
LS257
Strob
CLear
A
B
C
E16
LS348
A2
A1
A0
EO

Serial Port # 3

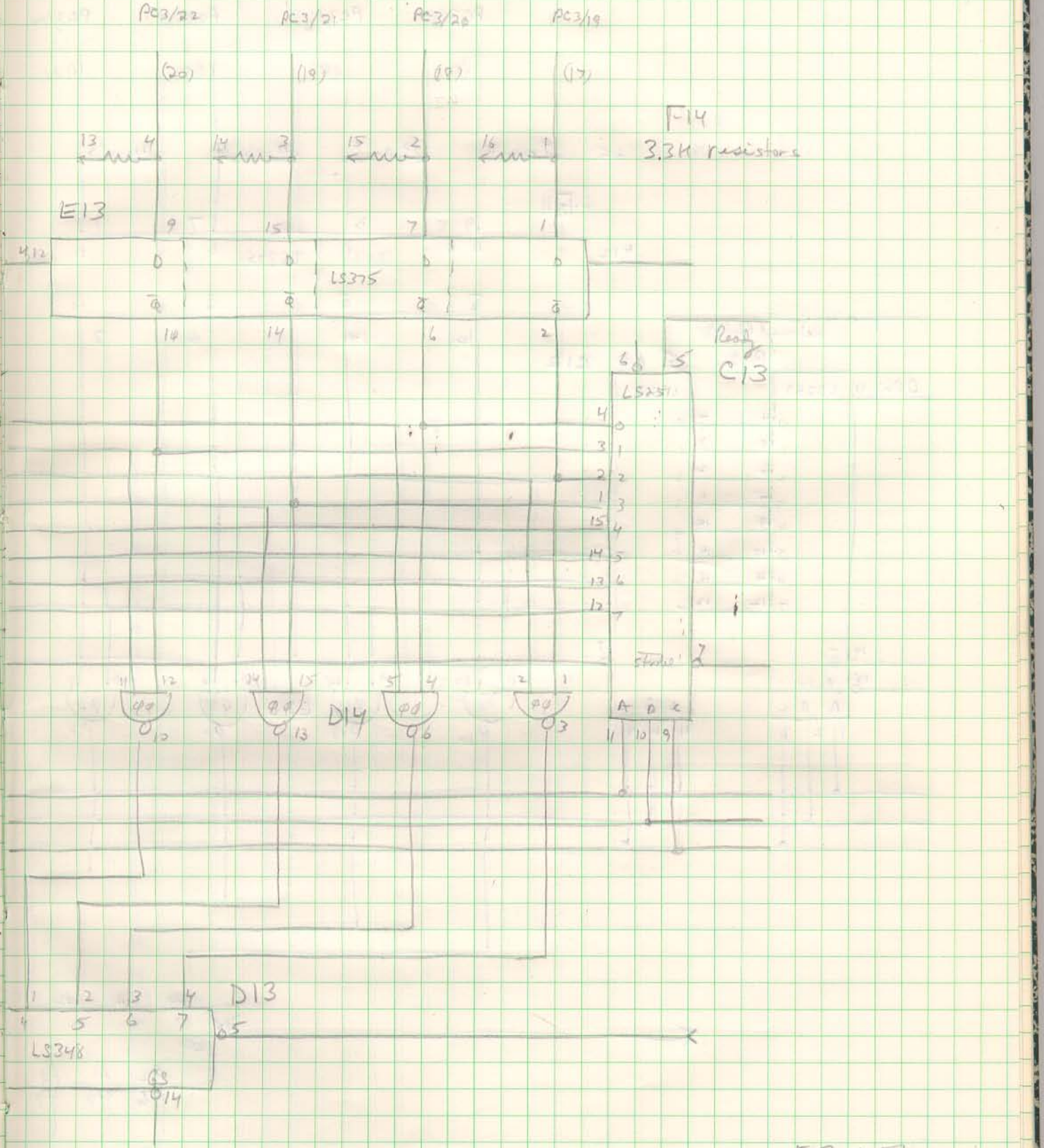


Printer 25-17

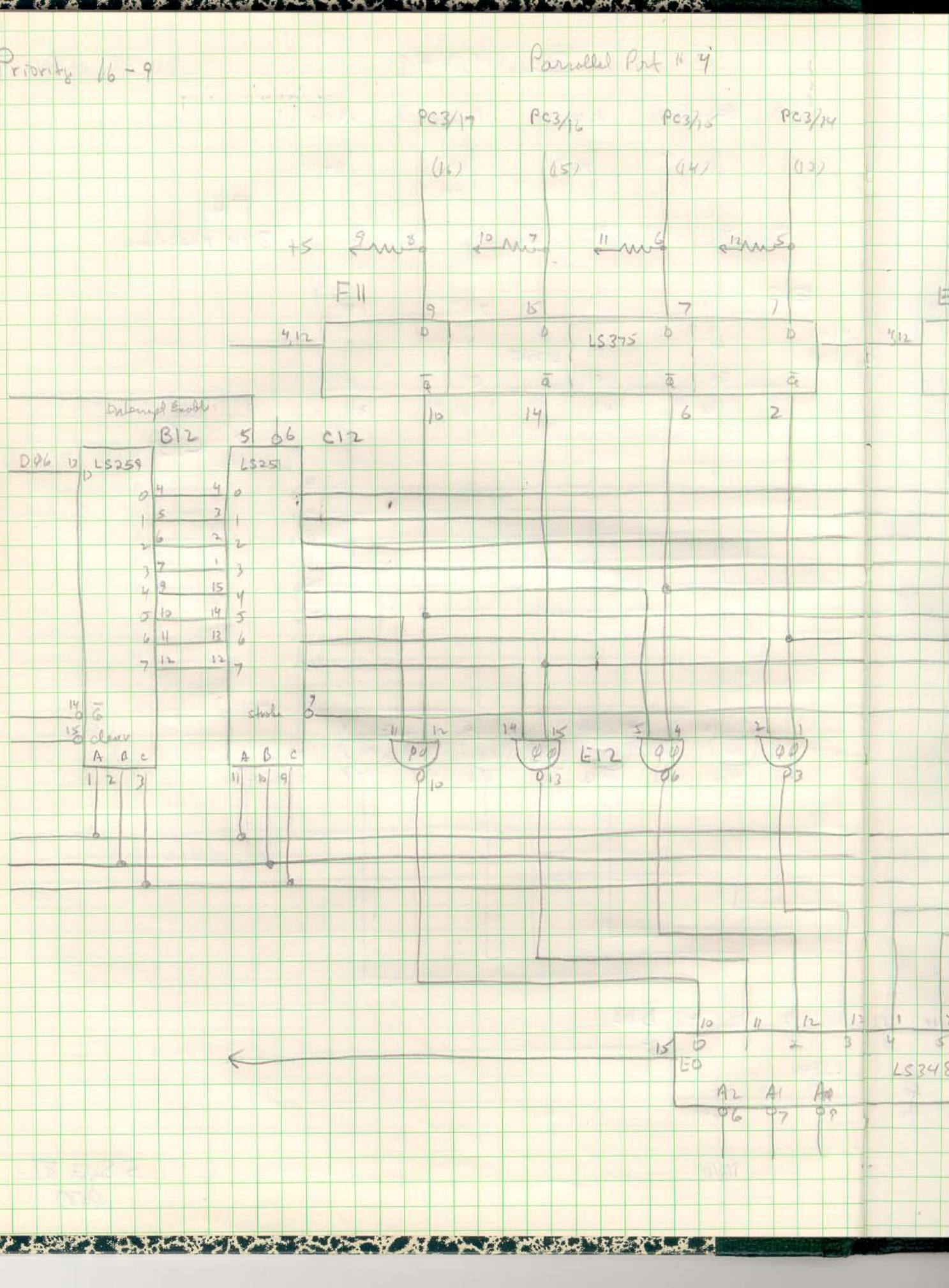
Serial Port #2



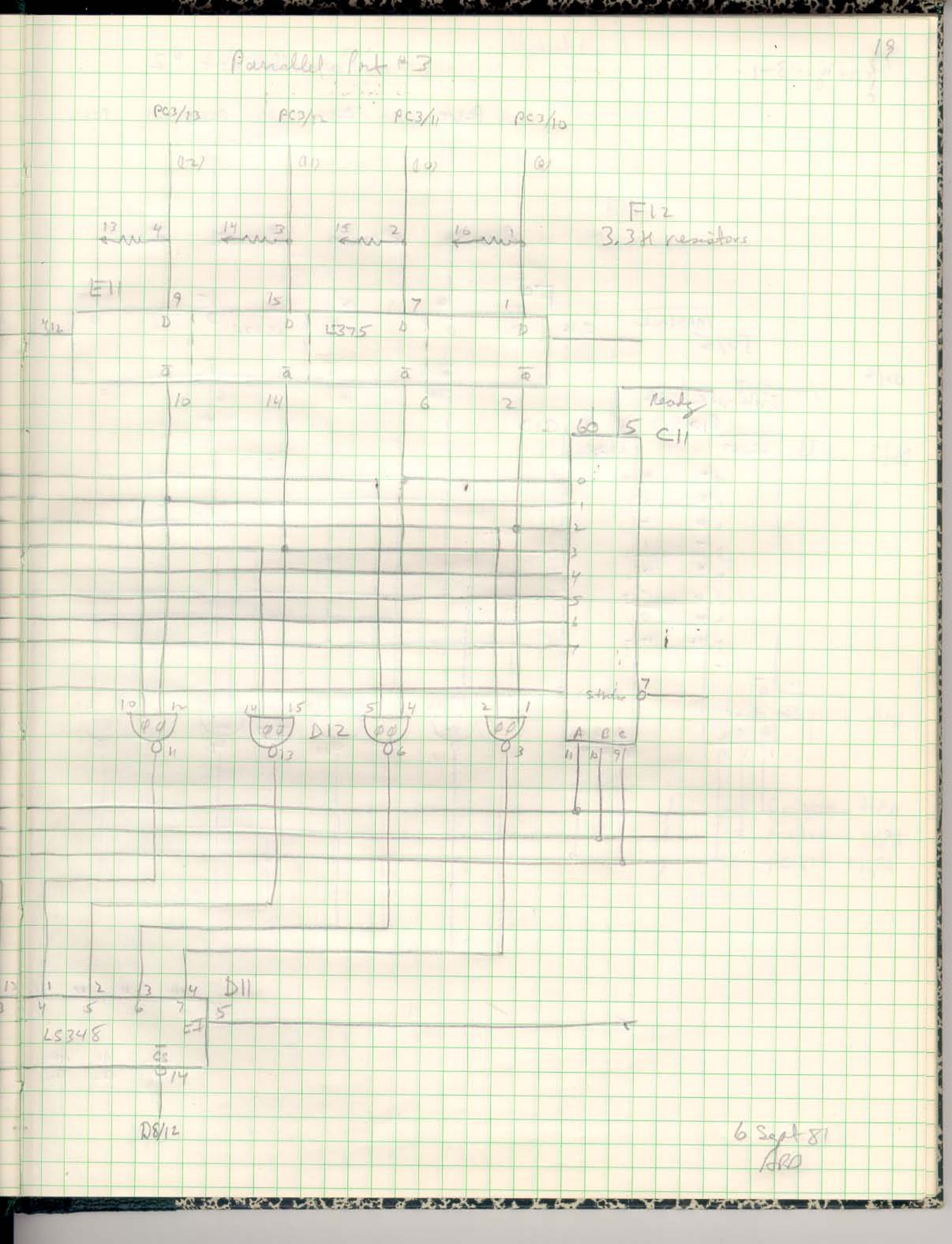
Serial Port #1



5 Sept 81


Priority 16 - 9
Parallel Port # 4
PC3/17
PC3/16
PC3/15
PC3/14
(16)
(15)
(14)
(13)
+5
F11
LS375
4,12
Interrupt Enable
B12
C12
LS259
LS251
DB6
clear
strobe
A
B
C
E12
E0
LS34
A2
A1
A0

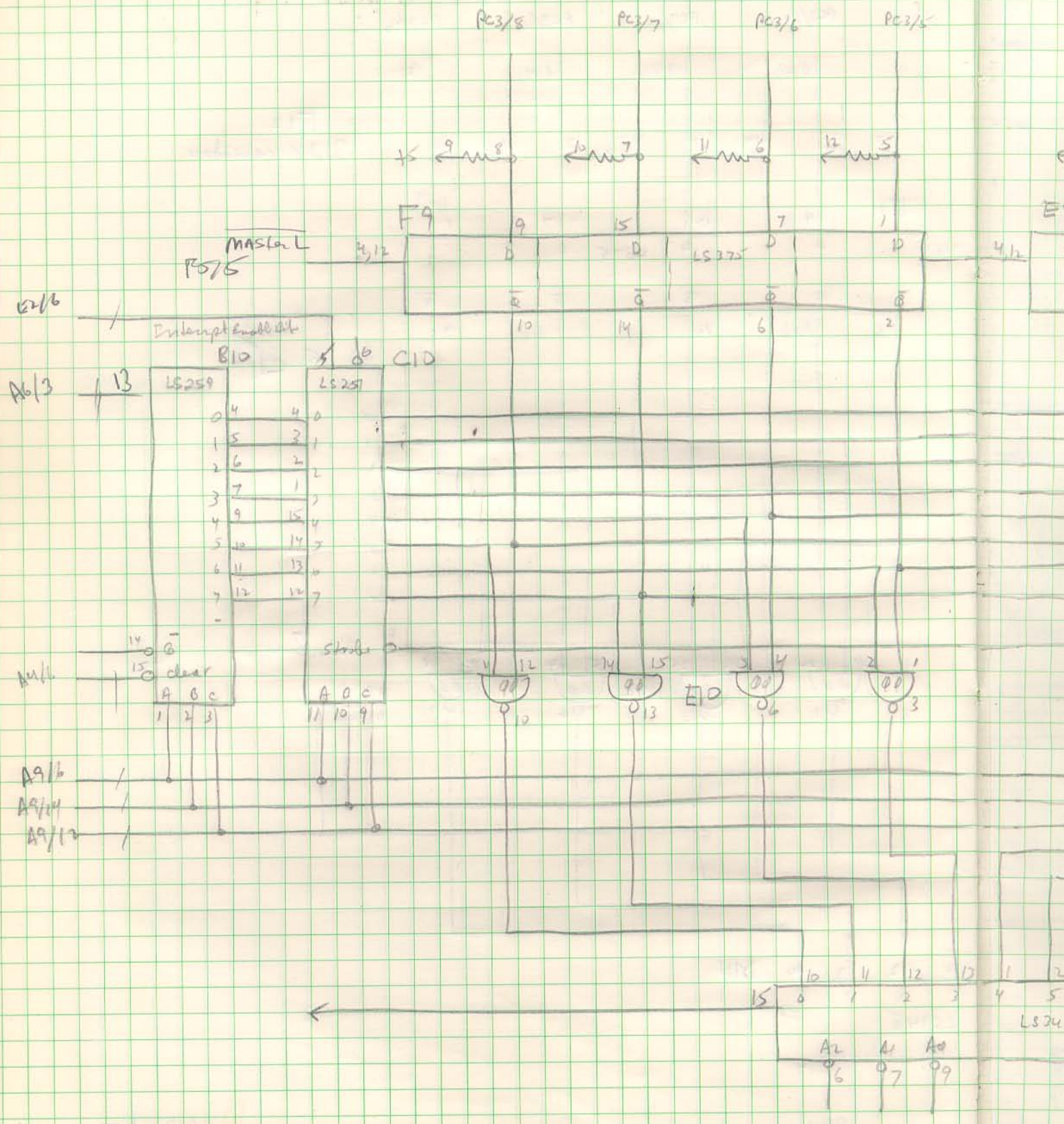
Parallel Port #3



6 Sept 81
ARO

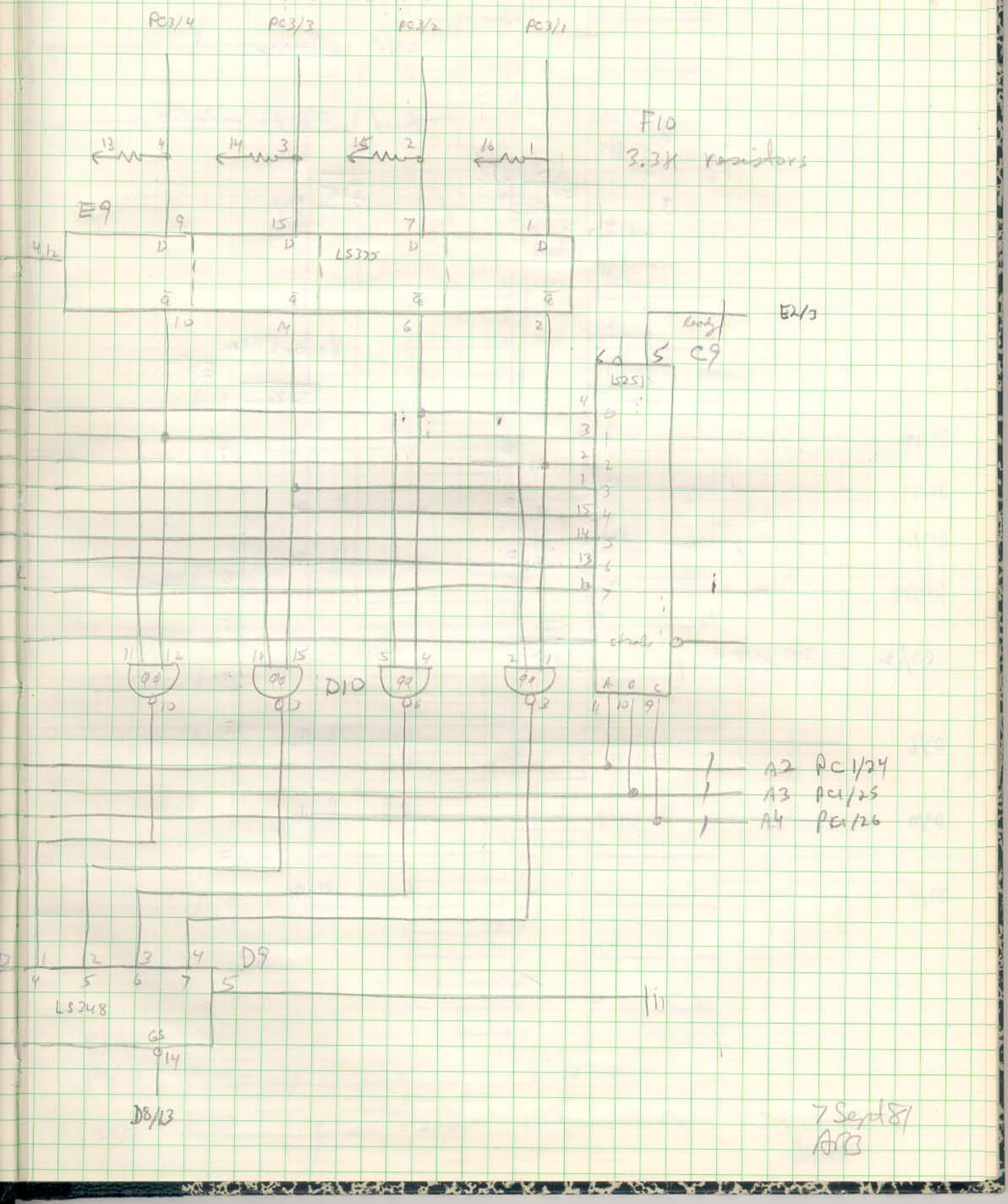
Priority 8-1

Parallel Port #2



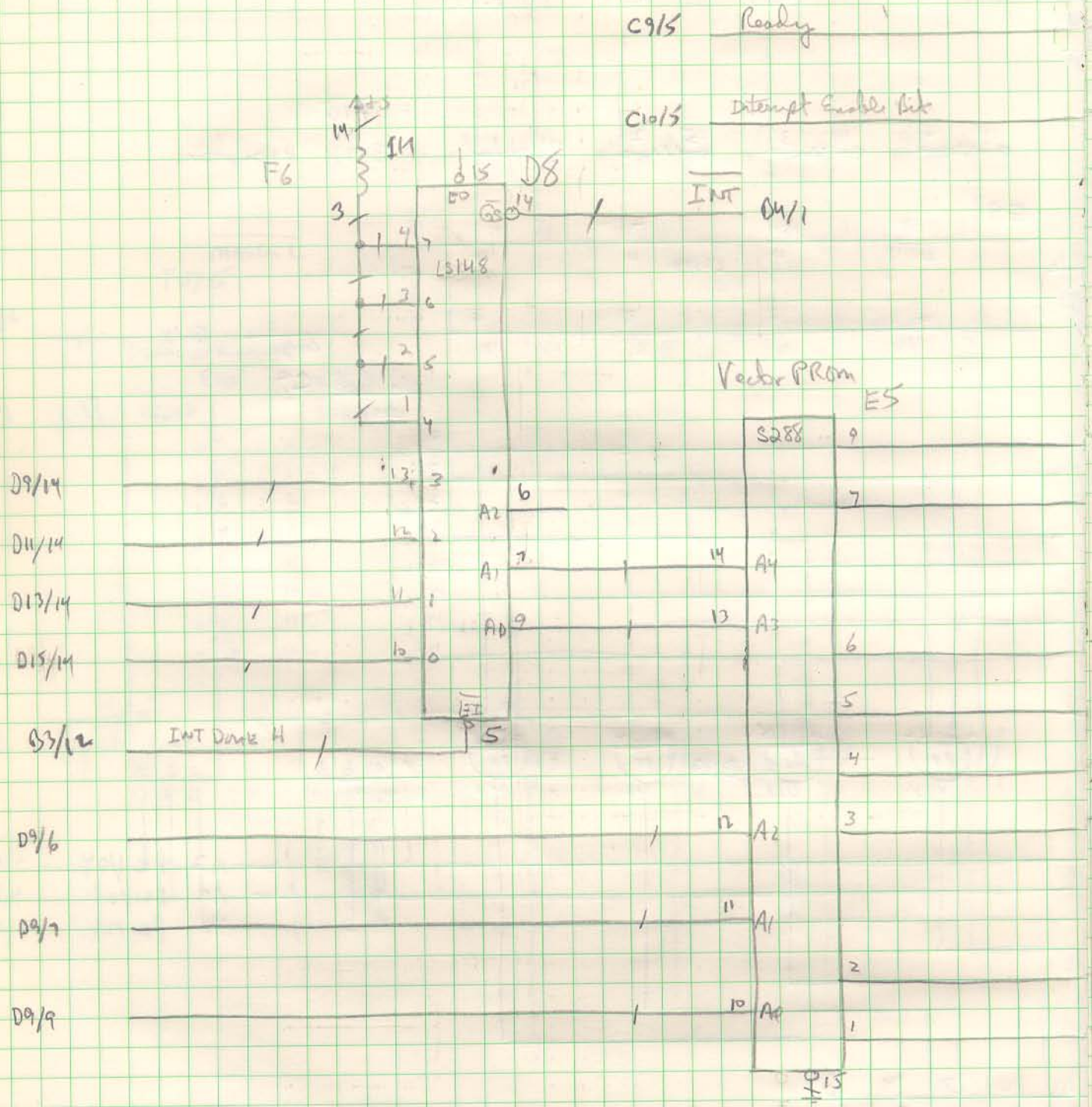
(Highest Priority)

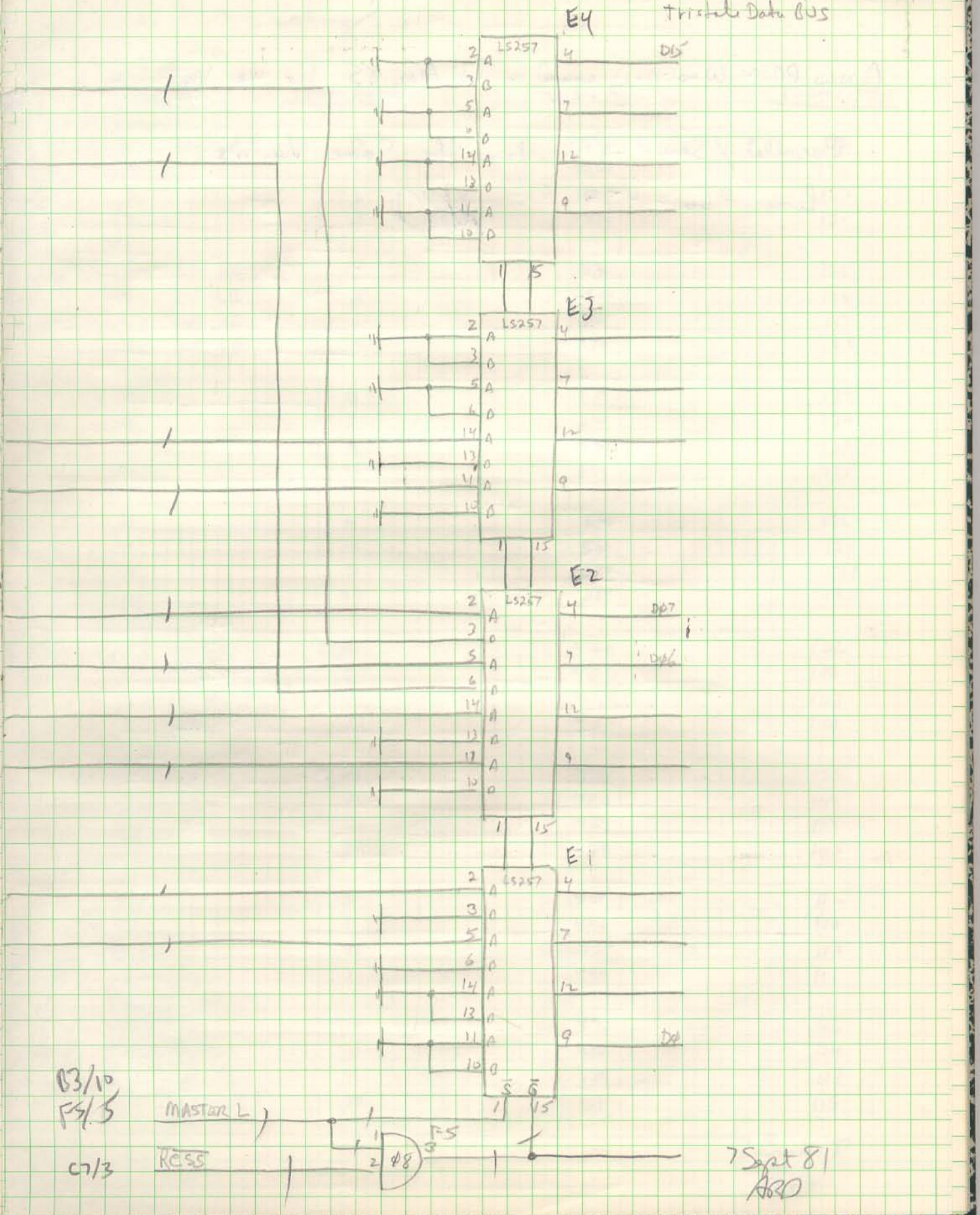
Parallel Port #1



7 Sept 81
ARB

Interrupt Vector & Device Status Logic




E4
Tristate Data BUS
LS257
DB5
E3
LS257
E2
LS257
DB7
DB6
E1
LS257
DB4
MASTER L
RESS
#8
F5
B3/10
F5/5
C7/3
7 Sept 81
ABO

A new PROM was programmed on 31 May 85 for the Vectors -

Parallel / Serial Ports Have the Same Vectors

from 400 - 474

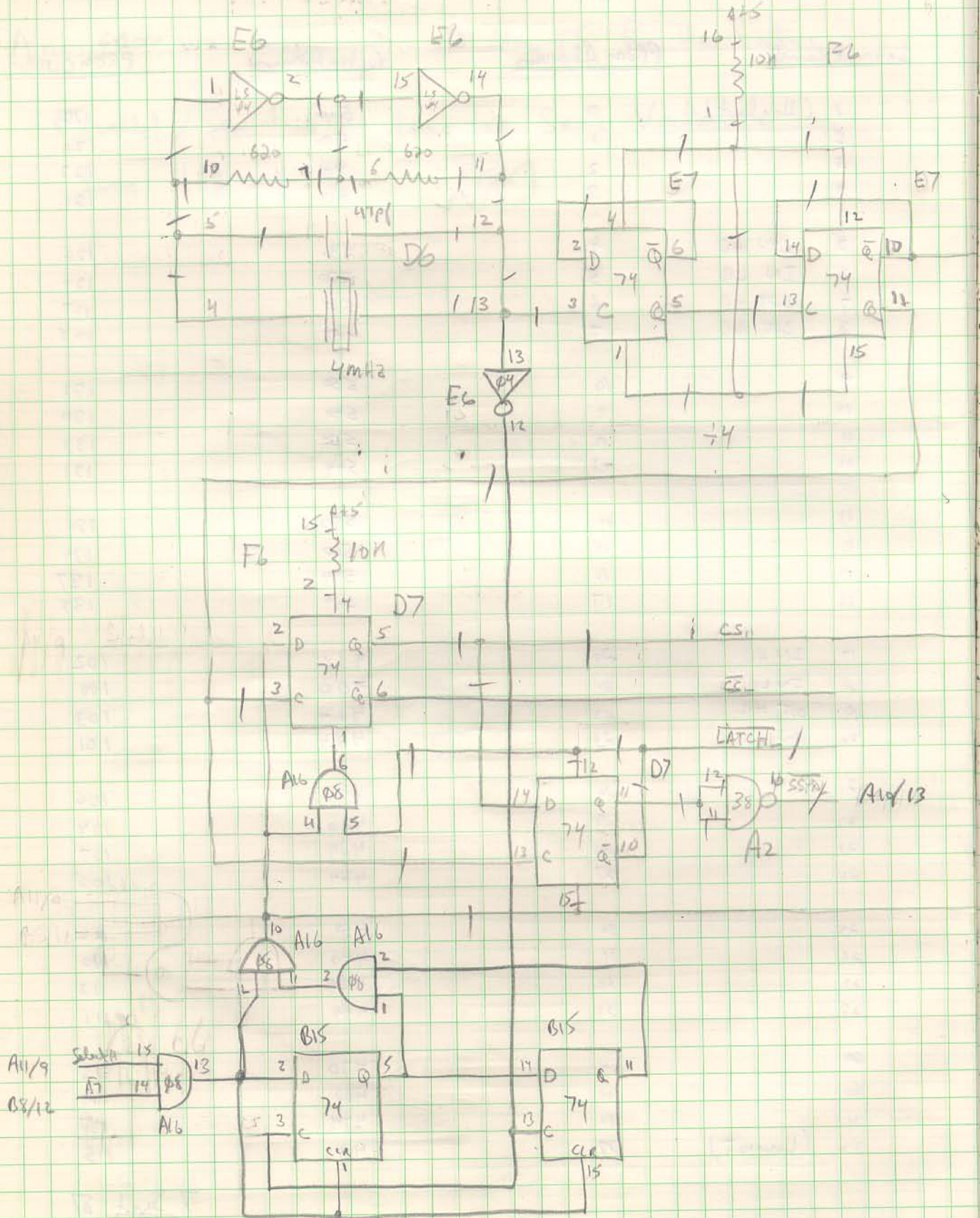
BBD

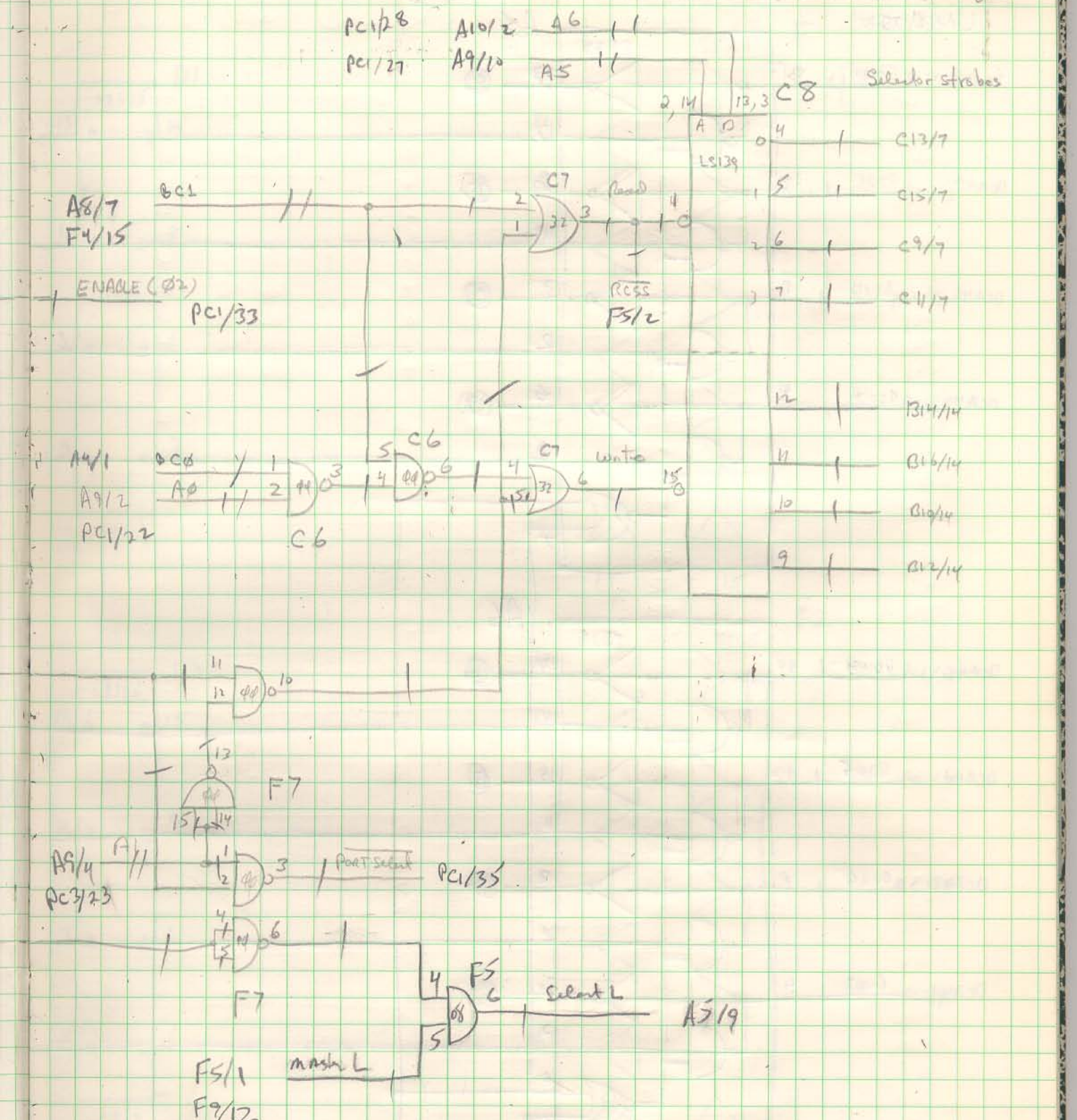
Interrupt PROM Coding

Interrupt Level		PROM Address	Vector Address	PROM OUTPUT
1	(Highest)	0	510	122
2		1	500	120
3		2	504	123
4		3	514	121
5	IN HB	4	530	126
6	IN LB	5	520	124
7	OUT HB	6	534	127
8	OUT LB	7	524	125
9		10	550	132
10		11	540	130
11		12	554	133
12		13	544	131
13		14	570	136
14		15	560	134
15		16	574	137
16		17	564	135
17	IN HB	20	410	102
18	IN LB	21	400	100
19	OUT HB	22	414	103
20	OUT LB	23	404	101
21		24	430	106
22		25	420	104
23		26	434	107
24		27	424	105
25		30	450	112
26		31	440	110
27		32	454	113
28		33	444	111
29		34	470	116
30		35	460	114
31		36	474	117
32	(Lowest)	37	464	115

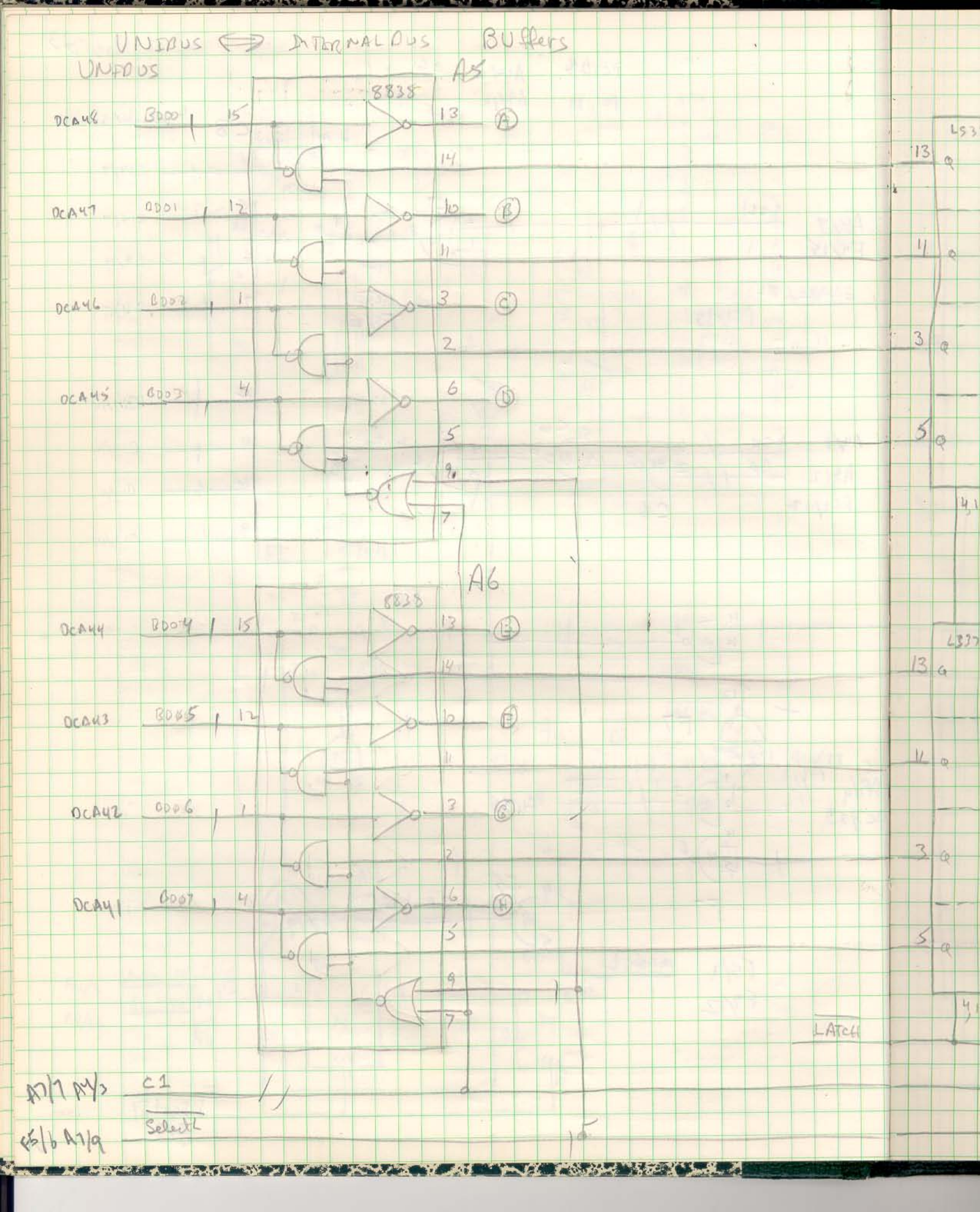
7 Sept 87
ABD

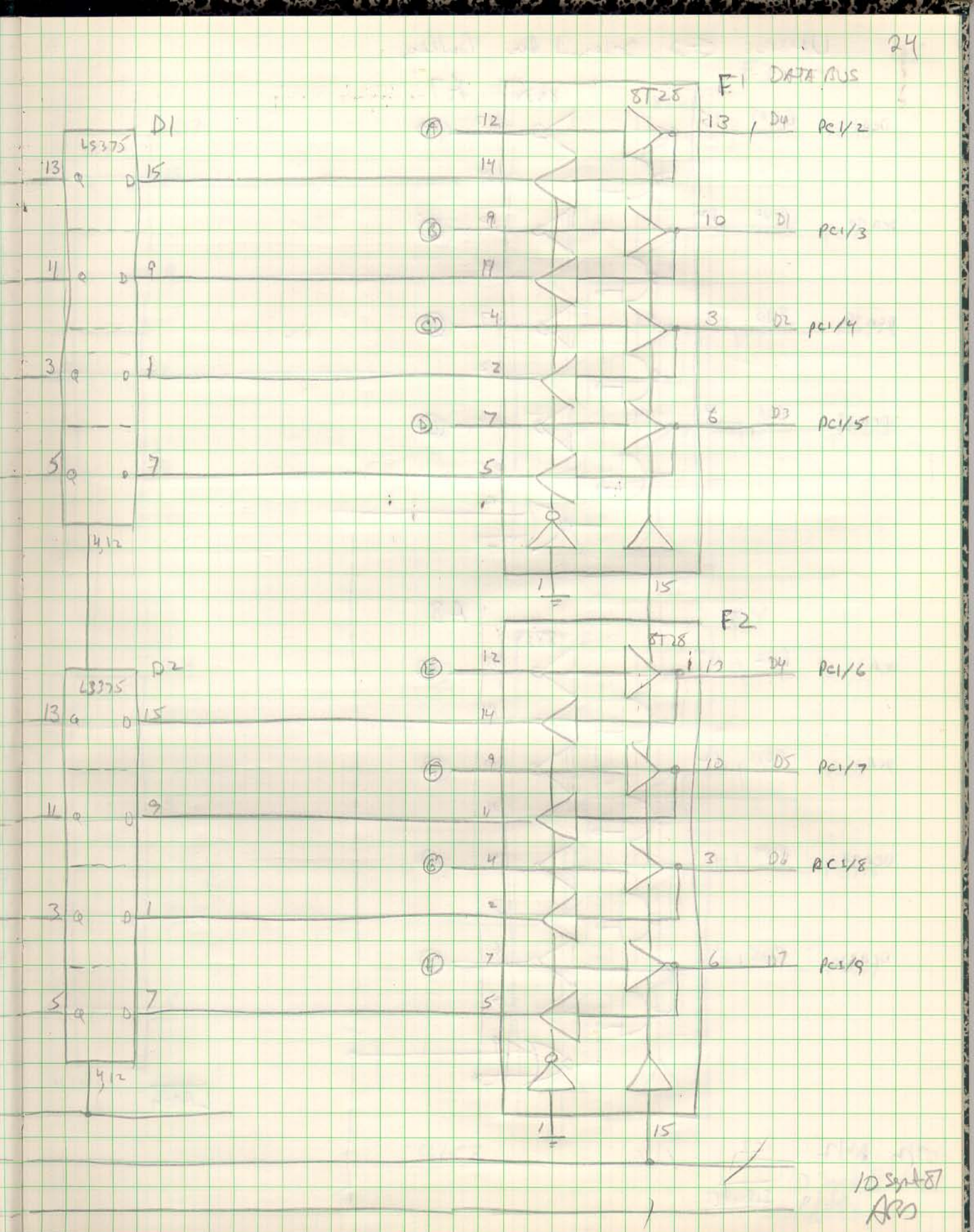
Clock & Select Logic

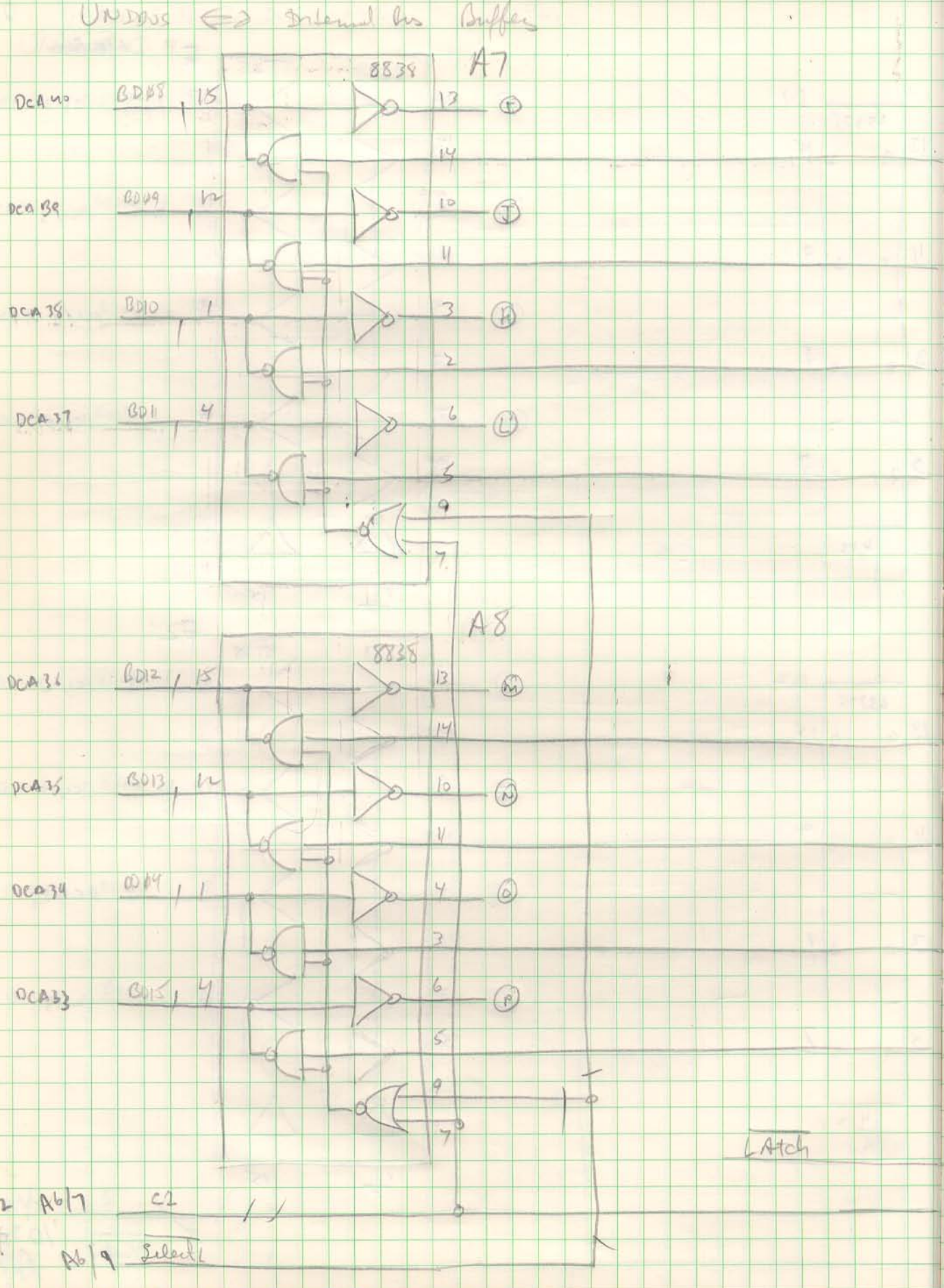


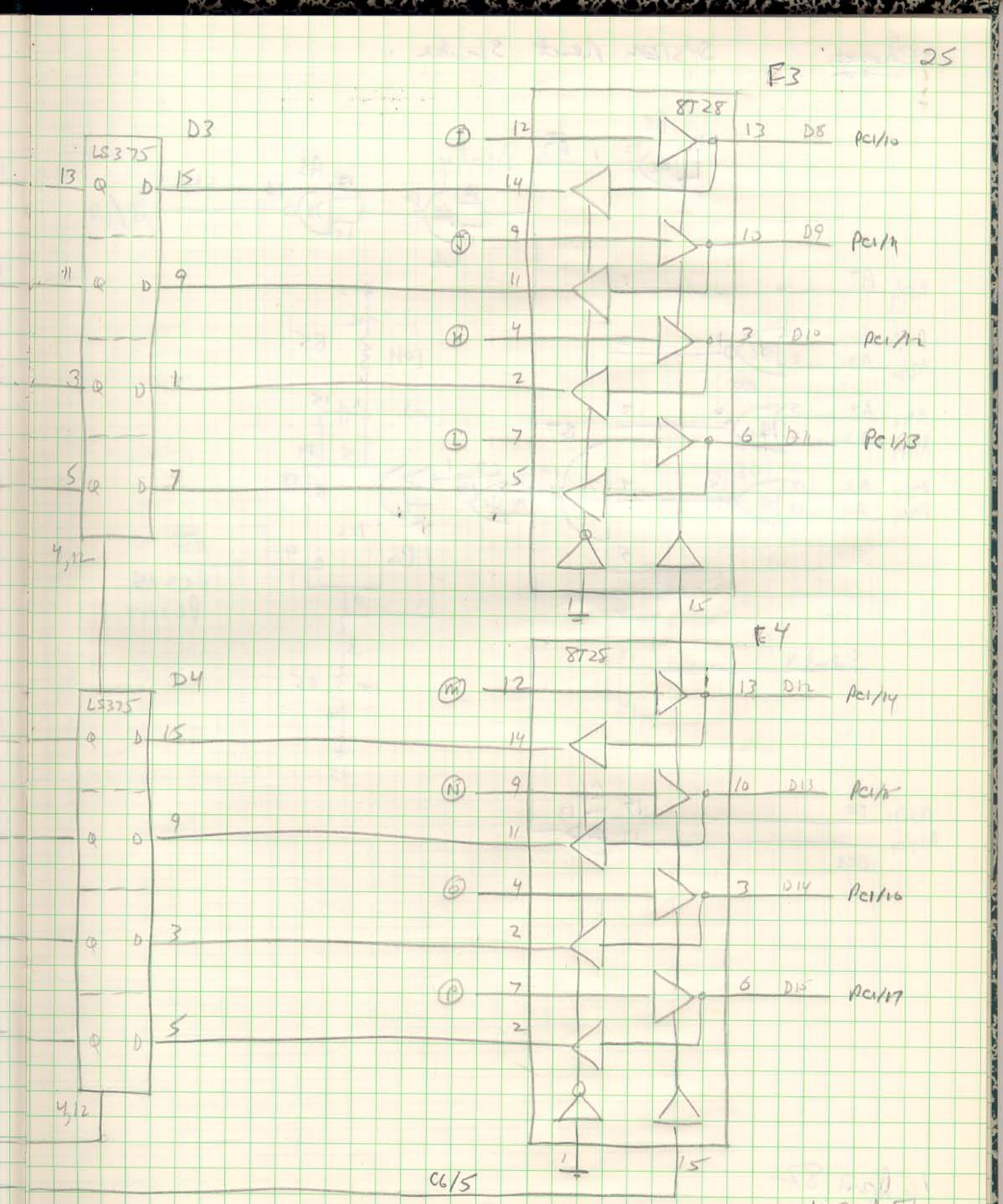

PC1/28
PC1/27
A10/2
A9/10
A6
A5
C8
Selector strobes
LS139
C13/7
C15/7
C9/7
C11/7
B14/14
B16/14
B10/14
B12/14
A8/7
F4/15
BC1
C7
Read
PCSS
F5/2
ENABLE (Ø2)
PC1/33
A4/1
A9/2
PC1/22
BCØ
AØ
CØ
C6
write
F7
A5/4
PC3/23
PORT SELECT
PC1/35
F5/6
Select L
A5/9
F5/1
F9/12
master L

8 Sept 81
AO


UNIBUS ⟺ INTERNAL DUS BUffers
UNIBUS
A5
8838
A6
8838
DCA48
DCA47
DCA46
DCA45
DCA44
DCA43
DCA42
DCA41
BDOO
BDO1
BDO2
BDO3
BDO4
BDO5
BDO6
BDO7
LATCH
Select L

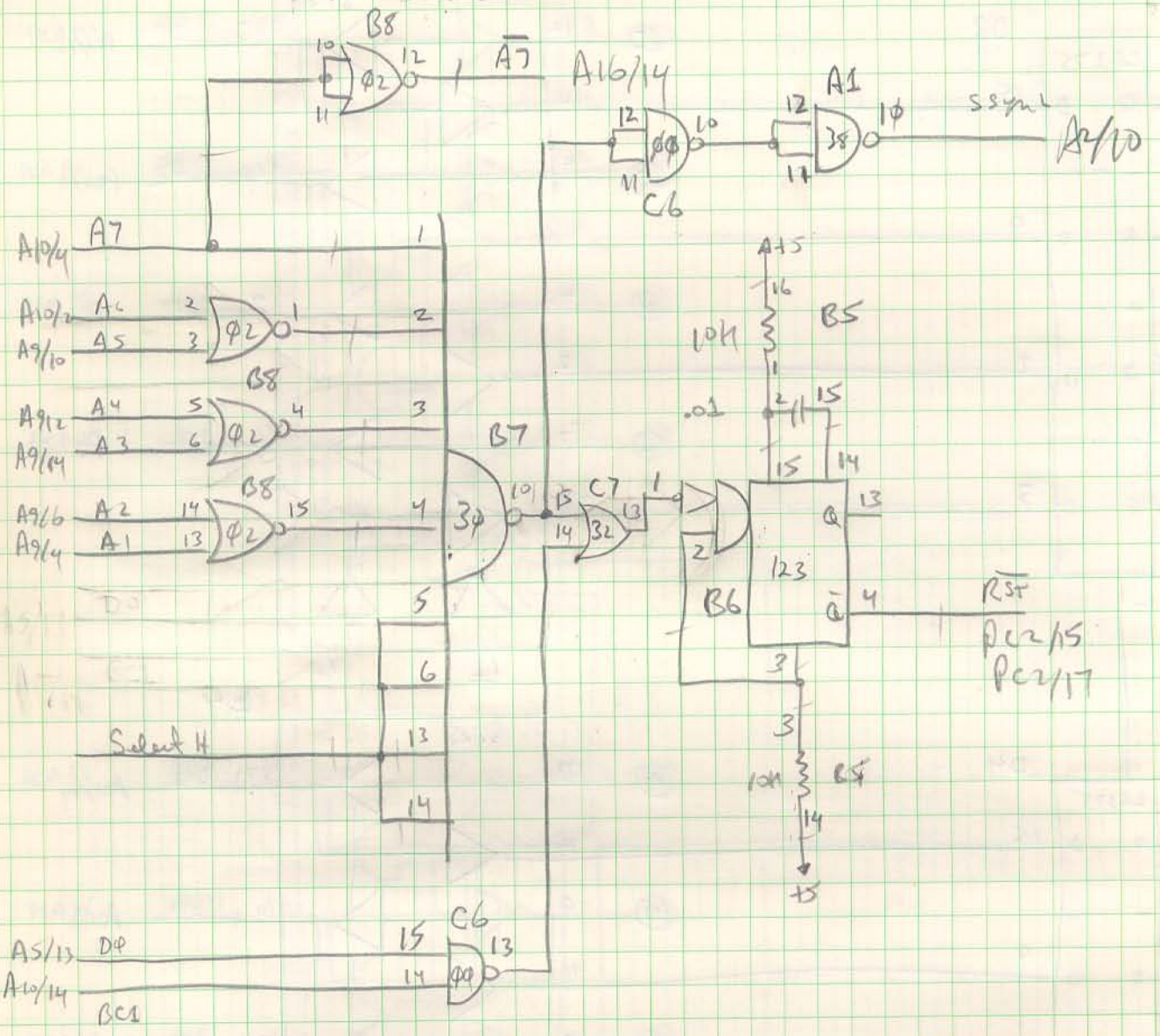

F1 DATA BUS
8T28
D1
LS375
13
15
11
9
3
1
5
7
4,12
D2
LS375
13
15
11
9
3
1
5
7
4,12
A
B
C
D
E
F
G
H
12
14
9
11
4
2
7
5
1
15
13
10
3
6
D4
D1
D2
D3
PC1/2
PC1/3
PC1/4
PC1/5
F2
8T28
12
14
9
11
4
2
7
5
1
15
13
10
3
6
D4
D5
D6
D7
PC1/6
PC1/7
PC1/8
PC1/9
10 Sept 87
ARD


UNIBUS ⟷ Internal Bus Buffer
A7
8838
DCA40
BDB8
15
13
E
14
DCA39
BDB9
12
10
J
11
DCA38
BDB10
1
3
H
2
DCA37
BDB11
4
6
L
5
9
7
A8
8838
DCA36
BDB12
15
13
M
14
DCA35
BDB13
12
10
N
11
DCA34
BDB14
1
4
D
3
DCA33
BDB15
4
6
P
5
9
7
LS373
13
11
3
5
4,12
LS373
4,12
Latch
C1/2 A6/7
C1
A6/9
Select


F3
8T28
D3
LS375
13 Q D 15
11 Q D 9
3 Q D 1
5 Q D 7
4,12
F
12
13
D8
PC1/10
14
V
9
10
D9
PC1/11
11
W
4
3
D10
PC1/12
2
L
7
6
D11
PC1/13
5
1
15
F4
8T28
D4
LS375
M
12
13
D12
PC1/14
14
N
9
10
D13
PC1/15
11
P
4
3
D14
PC1/16
2
R
7
6
D15
PC1/17
2
1
15
C6/5
10 Sept 81
ASB

Change

SYSTEM Reset Strobe



16 April 82

IC Locations

A1	7438	12
A2	7438	12
A3	RES	12
A4	8837	11
A5	8838	24
A6	8838	24
A7	8838	25
A8	8838	25
A9	8837	11
A10	8837	11
A11	8136	10
A12	8136	10
A13	RES	10
A14	SW	10
A15	RES	10
A16	———	

B1	74LS74	12
B2	74LS00	12
B3	74LS02	12
B4	74LS14	12
B5	———	
B6	———	
B7	———	
B8	———	
B9	———	
B10	74LS259	20
B11	———	
B12	74LS259	19
B13	———	
B14	74LS259	18
B15	74LS74	23
B16	74LS259	17

C1	—	
C2	—	
C3	—	
C4	—	
C5	—	
C6	74LS00	23
C7	74LS32	23
C8	74LS139	23
C9	74LS251	20
C10	74LS251	20
C11	74LS251	19
C12	74LS251	19
C13	74LS251	18
C14	74LS251	18
C15	74LS251	17
C16	74LS251	17
D1	74LS375	24
D2	74LS375	24
D3	74LS375	25
D4	74LS375	25
D5	—	
D6	RES - Crystal	23
D7	74LS74	23
D8	74LS148	21
D9	74LS348	20
D10	74LS00	20
D11	74LS348	19
D12	74LS00	19
D13	74LS348	18
D14	74LS00	18
D15	74LS348	17
D16	74LS00	17

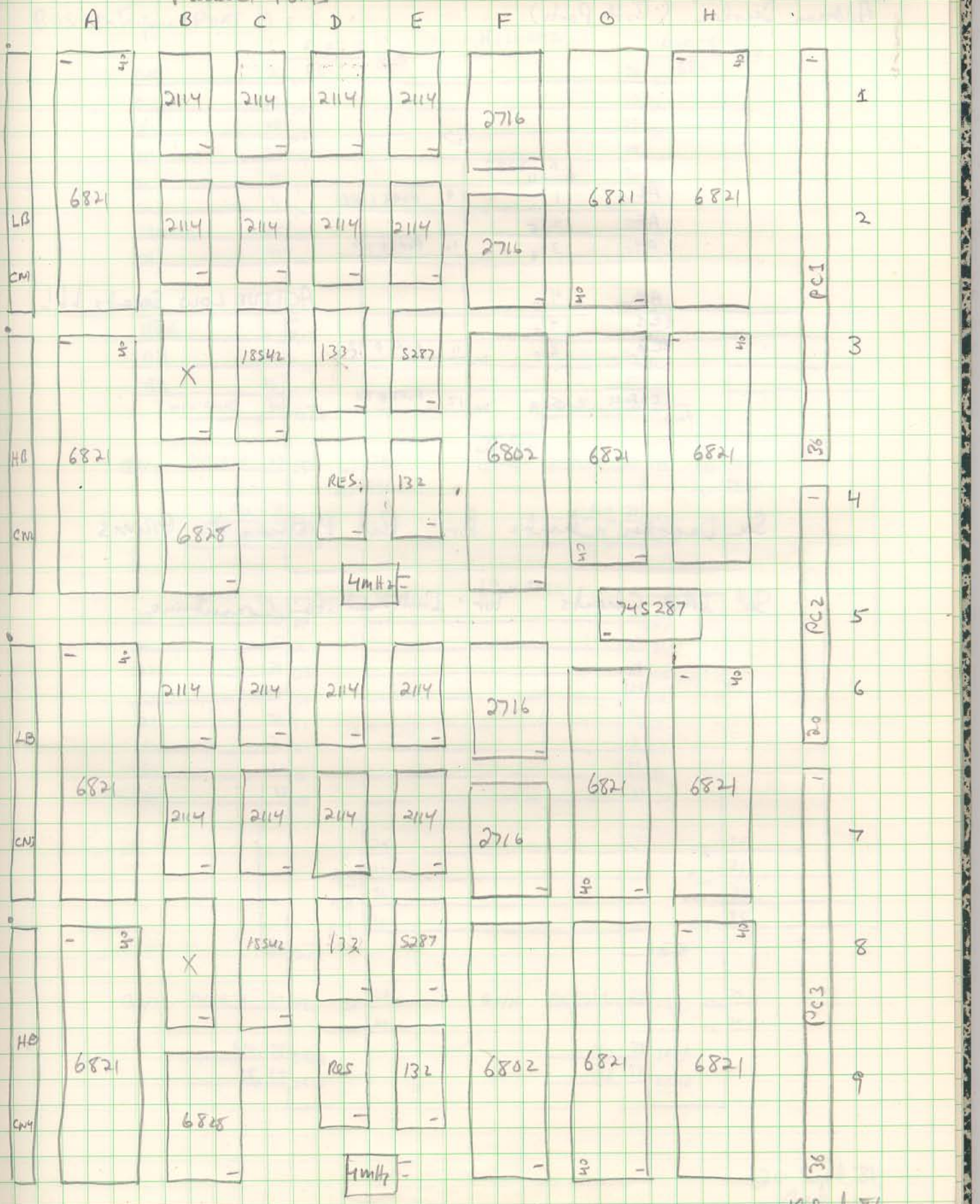
11 Spot 87
ADD

IC Locations

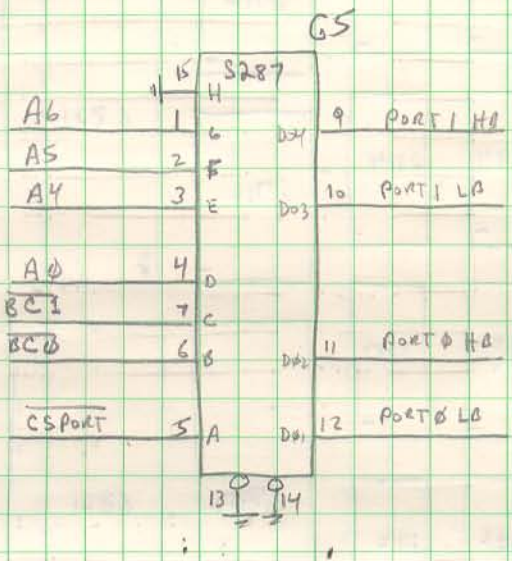
E1	8T28	24
E2	8T28	24
E3	8T28	25
E4	8T28	25
E5	74S288	21
E6	74LS04	23
E7	74LS74	23
E8		
E9	74LS375	20
E10	74LS00	20
E11	74S375	19
E12	74LS00	19
E13	74LS375	18
E14	74LS00	18
E15	74LS375	17
E16	74LS00	17

F1	74LS257	21
F2	74S257	21
F3	74LS257	21
F4	74LS257	21
F5	74LS08	21
F6	RES	21, 23
F7	74LS00	23
F8		
F9	74LS375	20
F10	RES	20
F11	74LS375	19
F12	RES	19
F13	74LS375	18
F14	RES	18
F15	74LS375	17
F16	RES	17

Parallel Ports



Address Decoder (Both Ports)

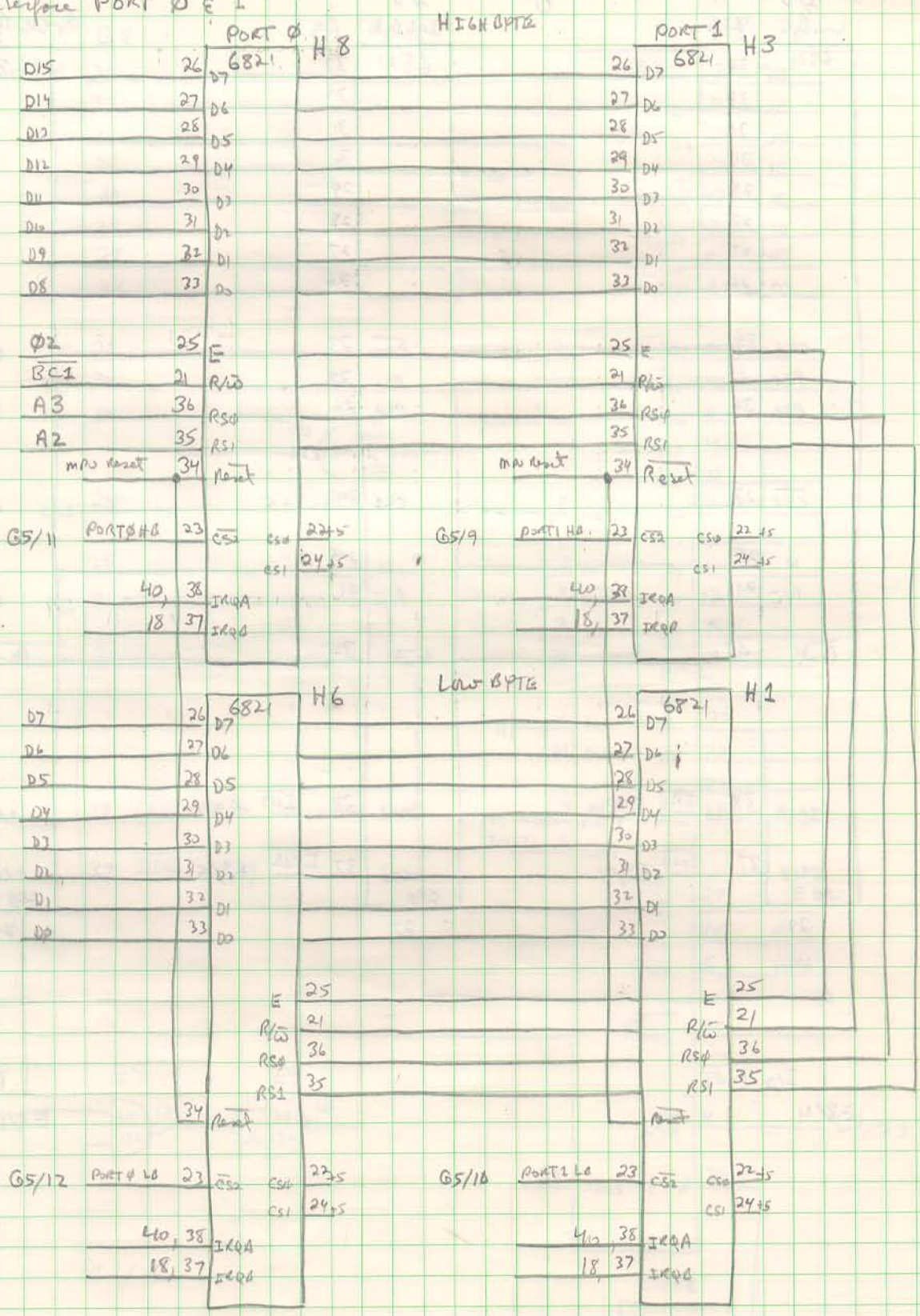


ACTIVE LOW Selects !!!

See Decoder Charts for Bit Patterns for PROMs

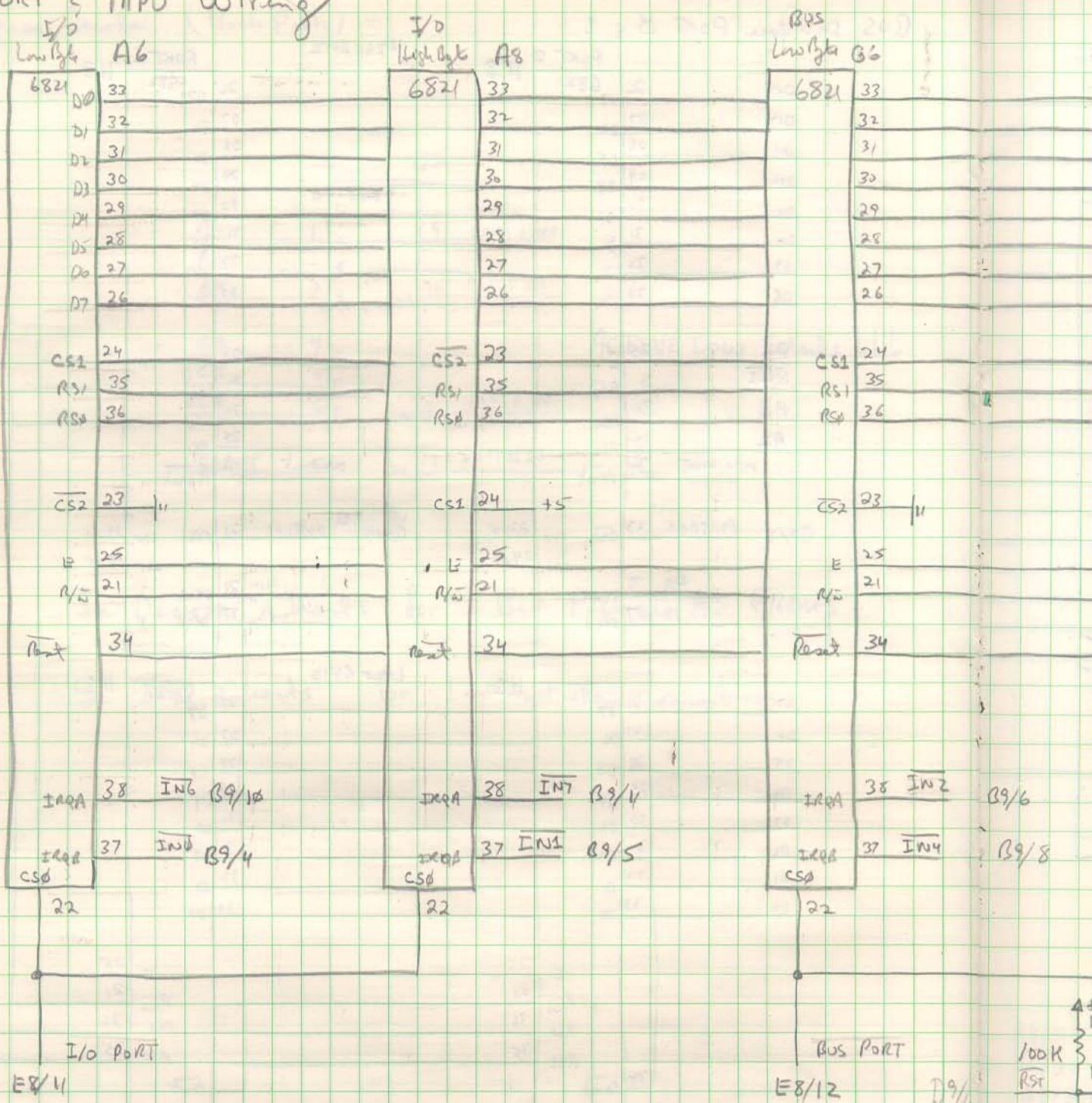
See IRQ Charts for Interrupt Pin Connections

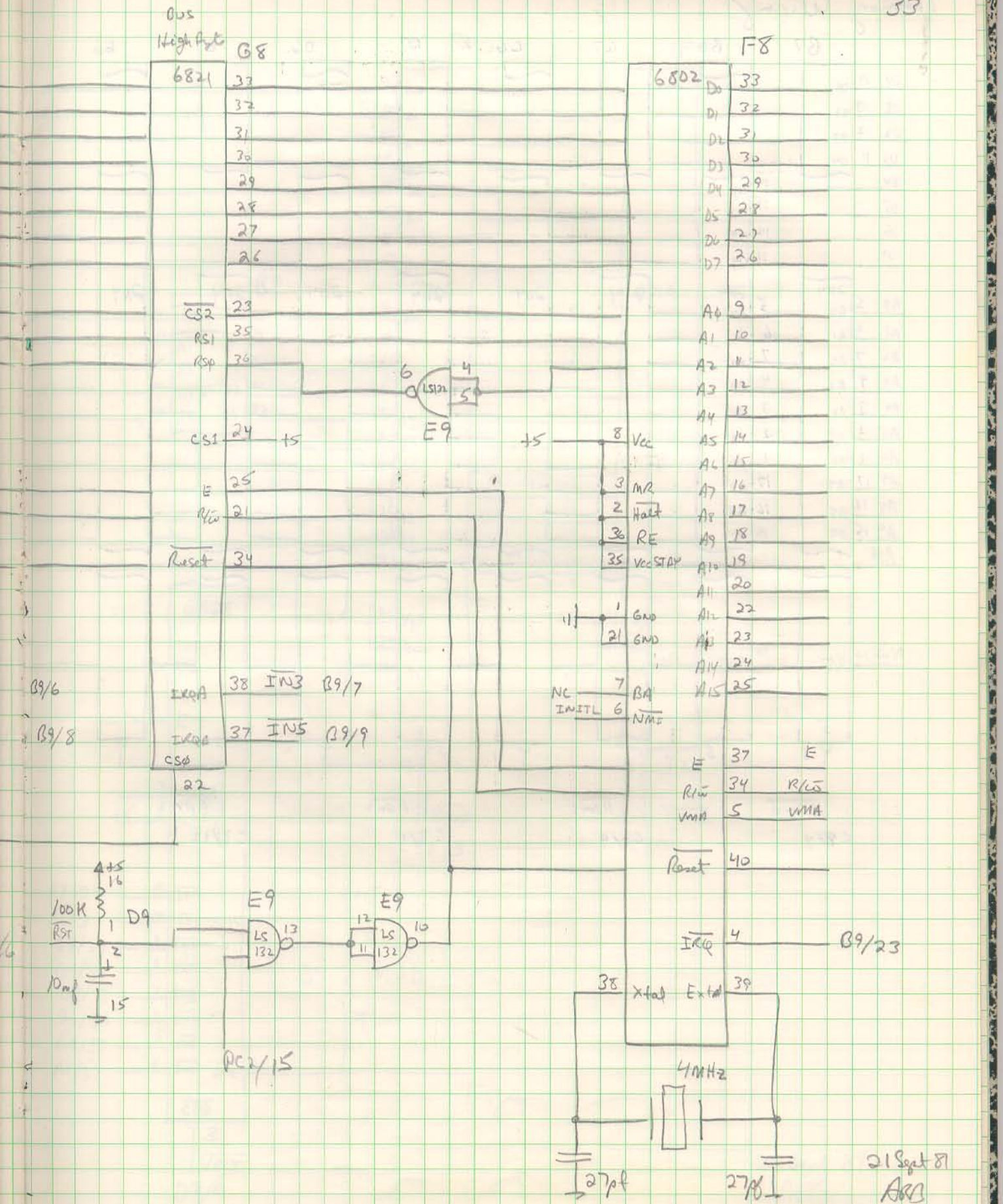
BUS Interface PORT Ø & 1



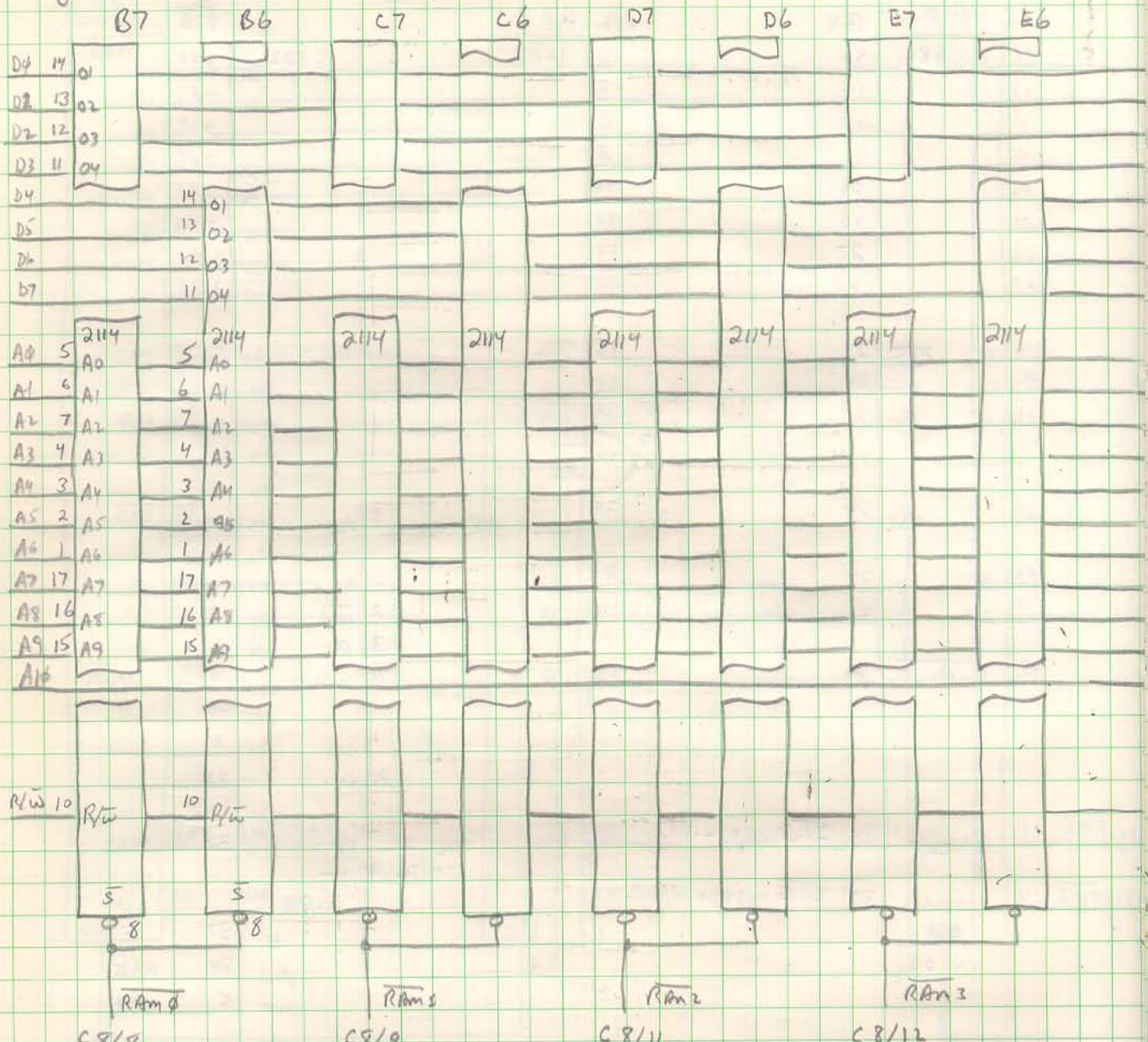
20 Sept 81
ARB

PORT E MPU Wiring

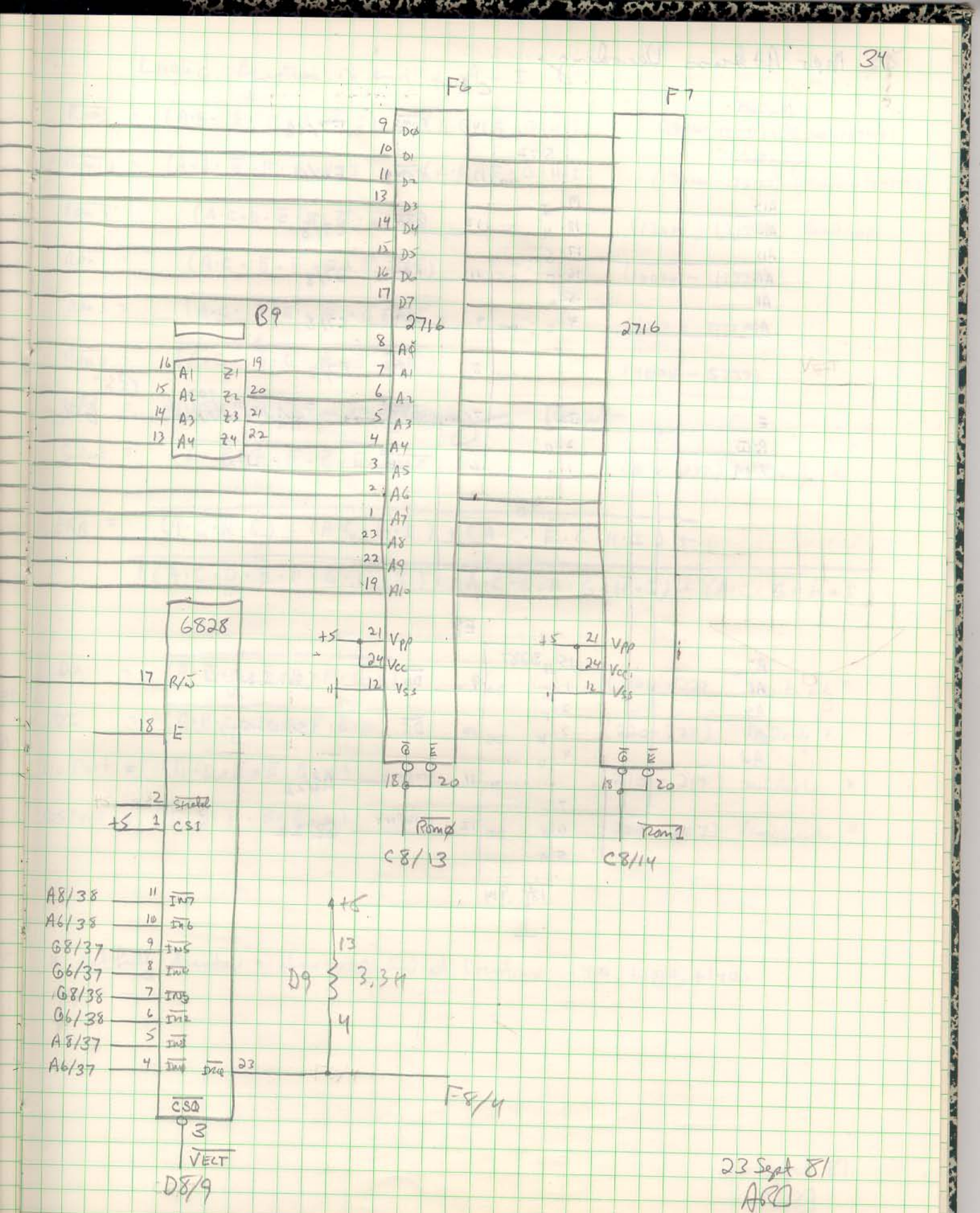



Bus
High Byte
G8
6821
F8
6802
CS2
RS1
RSØ
CS1
E
R/W
Reset
IRQA
IRQB
CSØ
IN3
IN5
B9/6
B9/7
B9/8
B9/9
+5
E9
LS32
LS123
D9
100K
RST
10mf
PC2/15
Vcc
MR
Halt
RE
VccSTBY
GND
NC
INITL
BA
NMI
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
D0
D1
D2
D3
D4
D5
D6
D7
E
R/W
VMA
Reset
IRQ
B9/23
Xtal
Extal
4MHz
27pf
21Sept 81
ABB

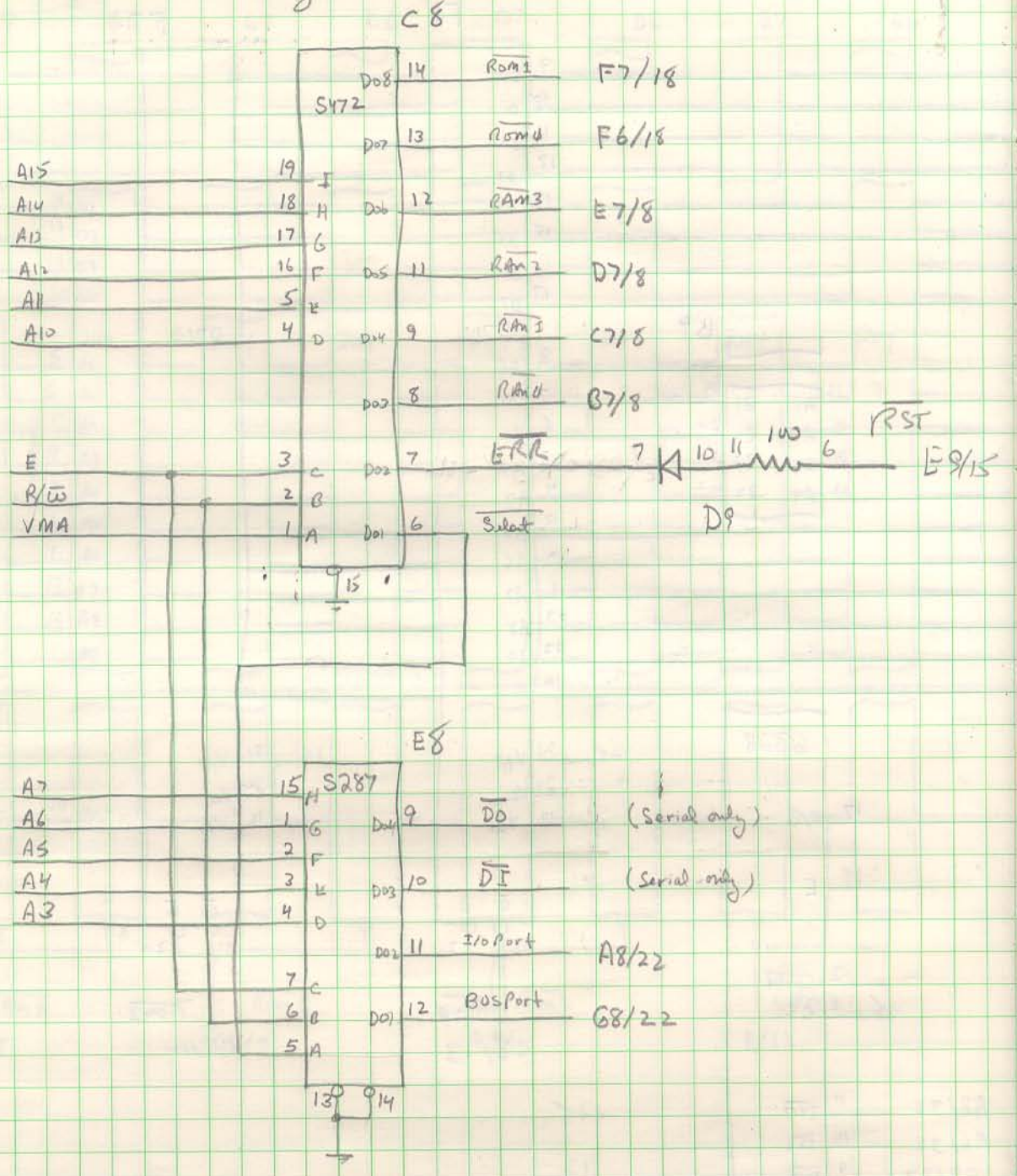
Memory Wiring



A8/38
A6/38
G8/37
G6/37
G8/38
G6/38
A8/37
A6/37


F6
F7
2716
2716
B9
A1
A2
A3
A4
Z1
Z2
Z3
Z4
6828
R/W
E
Strobe
CS1
IN7
IN6
IN5
IN4
IN3
IN2
IN1
IN0
CS0
VECT
D8/9
A8/38
A6/38
G8/37
G6/37
G8/38
G6/38
A8/37
A6/37
D9
3.3K
F-8/4
Rom0
Rom1
C8/13
C8/14
23 Sept 81
AGD

MPU Address Decoding



Logic Equations in terms of A – I

$\overline{ROM1} = \overline{(A\cdot B\cdot \bar{E}\cdot F\cdot G\cdot H\cdot I)} + \overline{(A\cdot B\cdot \bar{F}\cdot G\cdot H\cdot I)}$ (174000-177777) + (160000-167777) Read only

$\overline{ROM0} = \overline{(A\cdot B\cdot \bar{E}\cdot F\cdot G\cdot H\cdot I)} + \overline{(A\cdot B\cdot F\cdot \bar{G}\cdot H\cdot I)}$ (170000-173777) + (150000-157777) Read only

$\overline{RAM3} = \overline{(A\cdot C\cdot D\cdot \bar{E}\cdot F\cdot \bar{G}\cdot \bar{H}\cdot \bar{I})}$ (12000 - 13777) Read/write

$\overline{RAM2} = \overline{(A\cdot C\cdot \bar{D}\cdot \bar{E}\cdot F\cdot \bar{G}\cdot \bar{H}\cdot \bar{I})}$ (10000 - 11777)

$\overline{RAM1} = \overline{(A\cdot C\cdot D\cdot E\cdot \bar{F}\cdot \bar{G}\cdot \bar{H}\cdot \bar{I})}$ (6000 - 7777)

$\overline{RAM0} = \overline{(A\cdot C\cdot \bar{D}\cdot E\cdot \bar{F}\cdot \bar{G}\cdot \bar{H}\cdot \bar{I})}$ (4000 - 5777)

$\overline{ERR}$ = Non Existant Memory Reset

$\overline{Silent} = \overline{(A\cdot \bar{D}\cdot \bar{E}\cdot \bar{F}\cdot \bar{G}\cdot \bar{H}\cdot \bar{I})}$ (0 - 1777) Read/write

$$\overline{ERR} = \overline{(A\cdot C\cdot \bar{H}\cdot I) + (A\cdot C\cdot H\cdot \bar{I}) + (A\cdot C\cdot \bar{F}\cdot \bar{G}\cdot H\cdot I) + (A\cdot C\cdot D\cdot \bar{E}\cdot \bar{F}\cdot \bar{G}\cdot \bar{H}\cdot \bar{I}) + (A\cdot C\cdot E\cdot F\cdot \bar{G}\cdot \bar{H}\cdot I) + (A\cdot C\cdot G\cdot \bar{H}\cdot \bar{I})}$$

$\overline{DO} = \overline{(\bar{A}\cdot \bar{B}\cdot C\cdot \bar{D}\cdot E\cdot \bar{F}\cdot \bar{G}\cdot H)}$ (220 - 227) Write only *

$\overline{DI} = \overline{(\bar{A}\cdot B\cdot C\cdot \bar{D}\cdot E\cdot \bar{F}\cdot \bar{G}\cdot H)}$ (220-227) Read only *

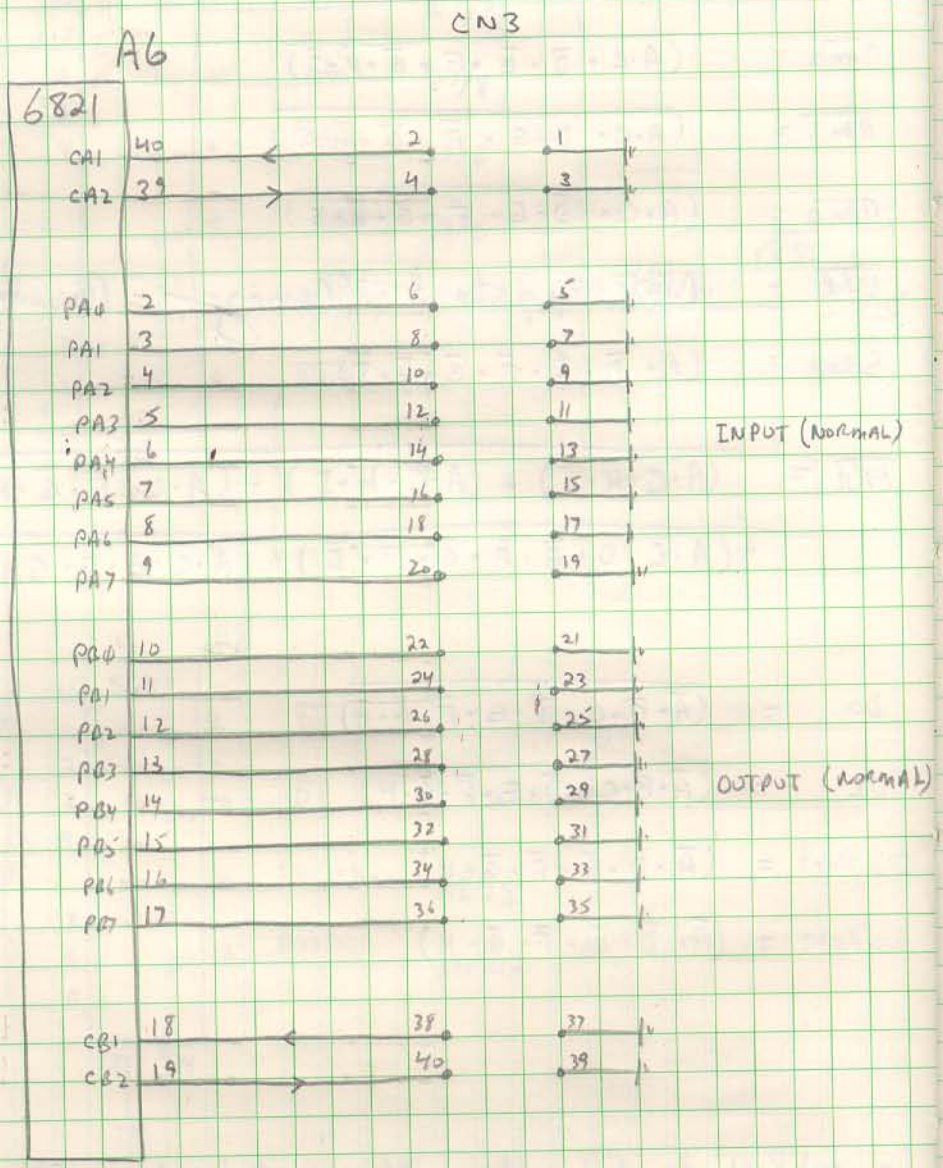
I/O Port = $(\bar{A}\cdot D\cdot \bar{E}\cdot \bar{F}\cdot \bar{G}\cdot H)$ (210 - 217) Read/write *

OS Port = $(\bar{A}\cdot \bar{D}\cdot \bar{E}\cdot \bar{F}\cdot \bar{G}\cdot H)$ (200 - 207) Read/write *

* Not fully decoded - also addressed at Locations +400, +1000, +1400

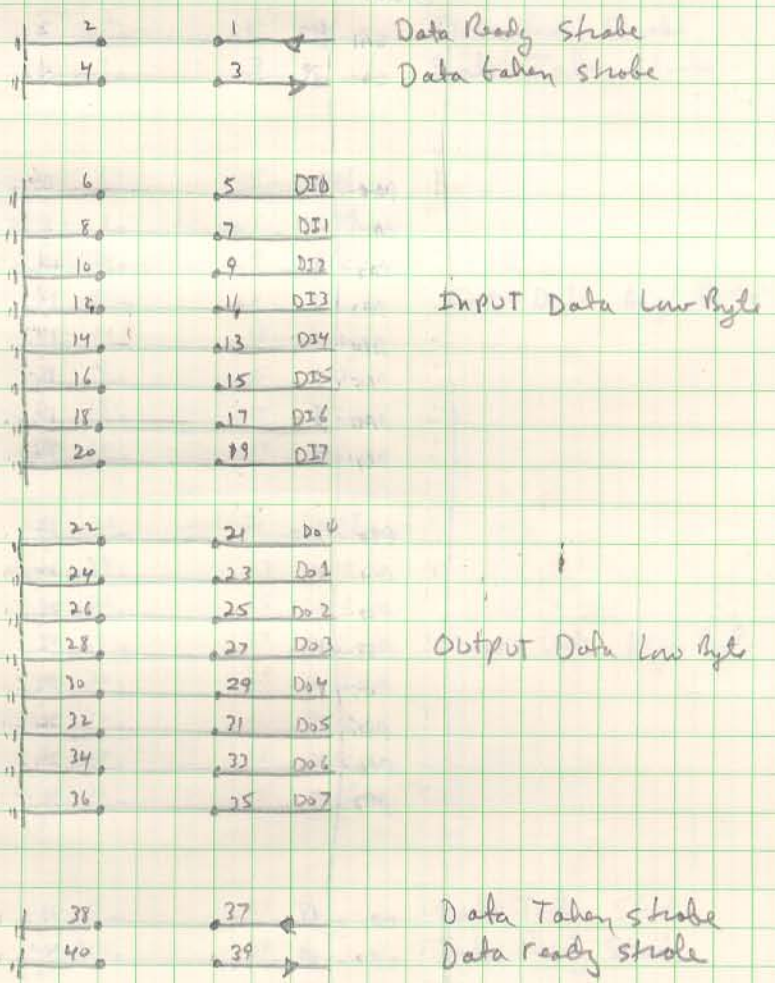
26 Sept 81
AMB

Low Byte I/O Connector Wiring



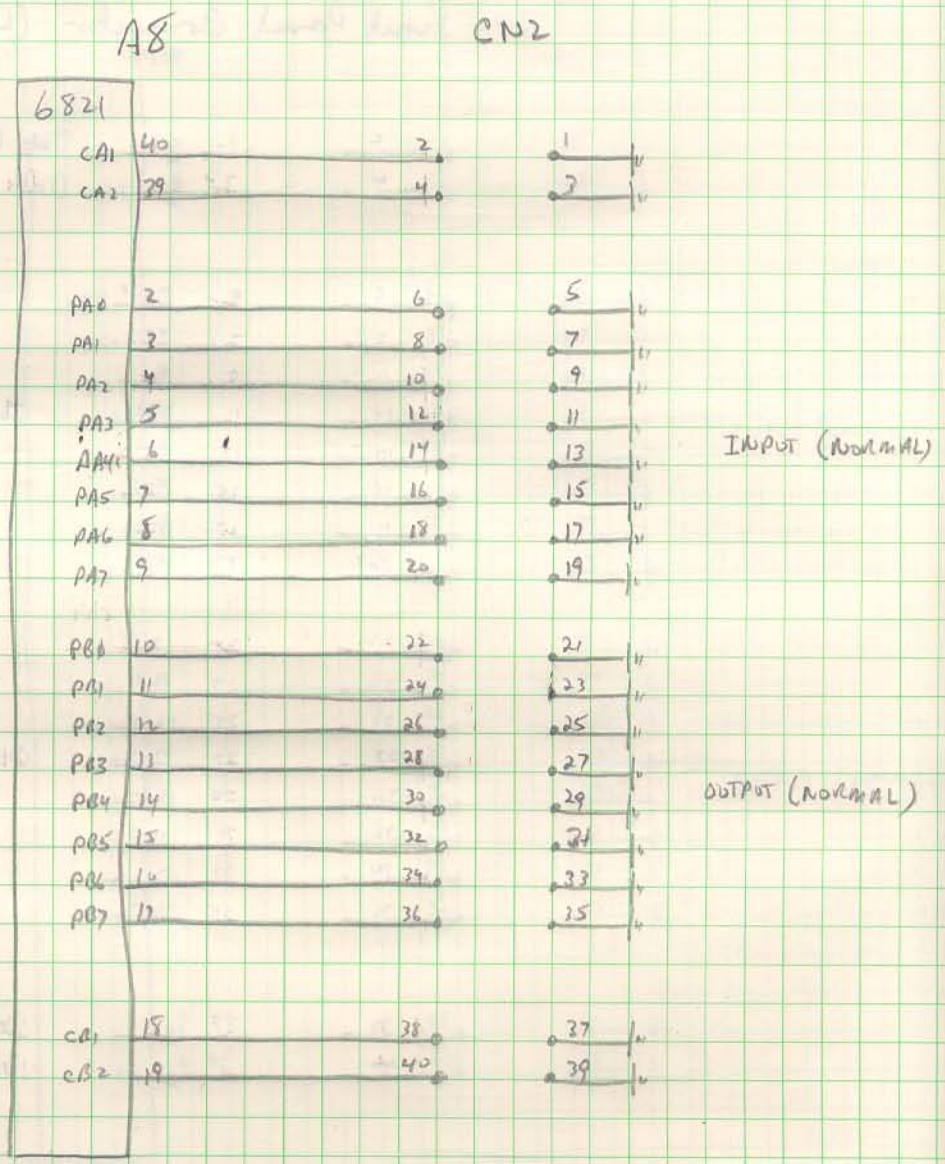
40 Pin Ribbon Connector

Front Panel Connector (Low Byte)



26 Sept 81
ARB

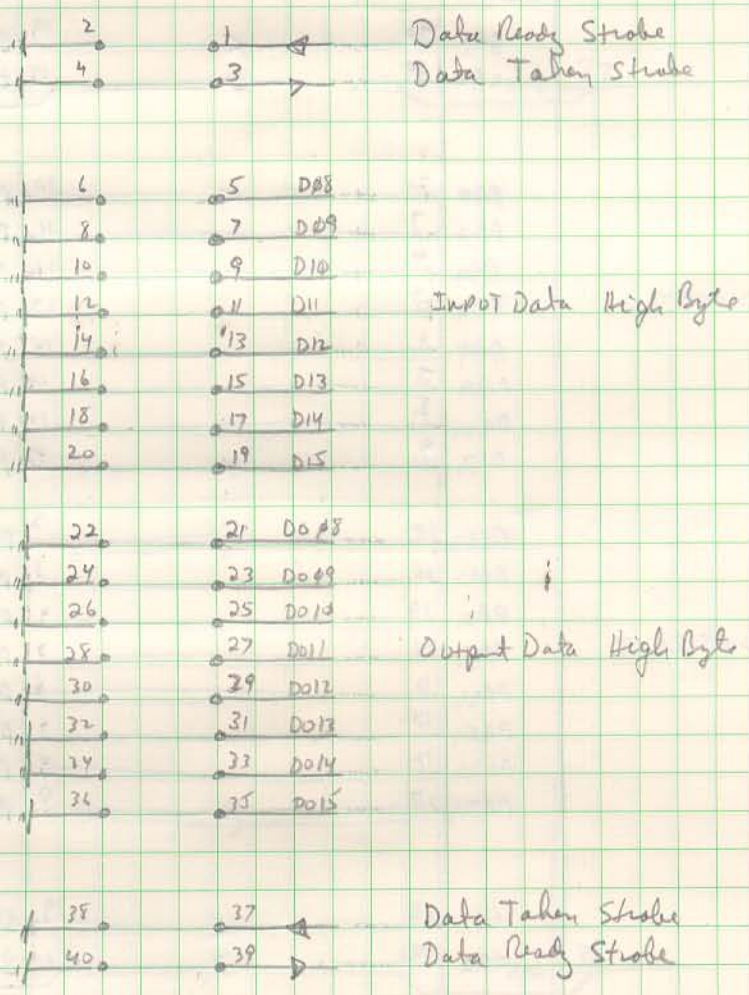
High Byte I/O Connector Wiring



Front Panel I/O Connector Wiring

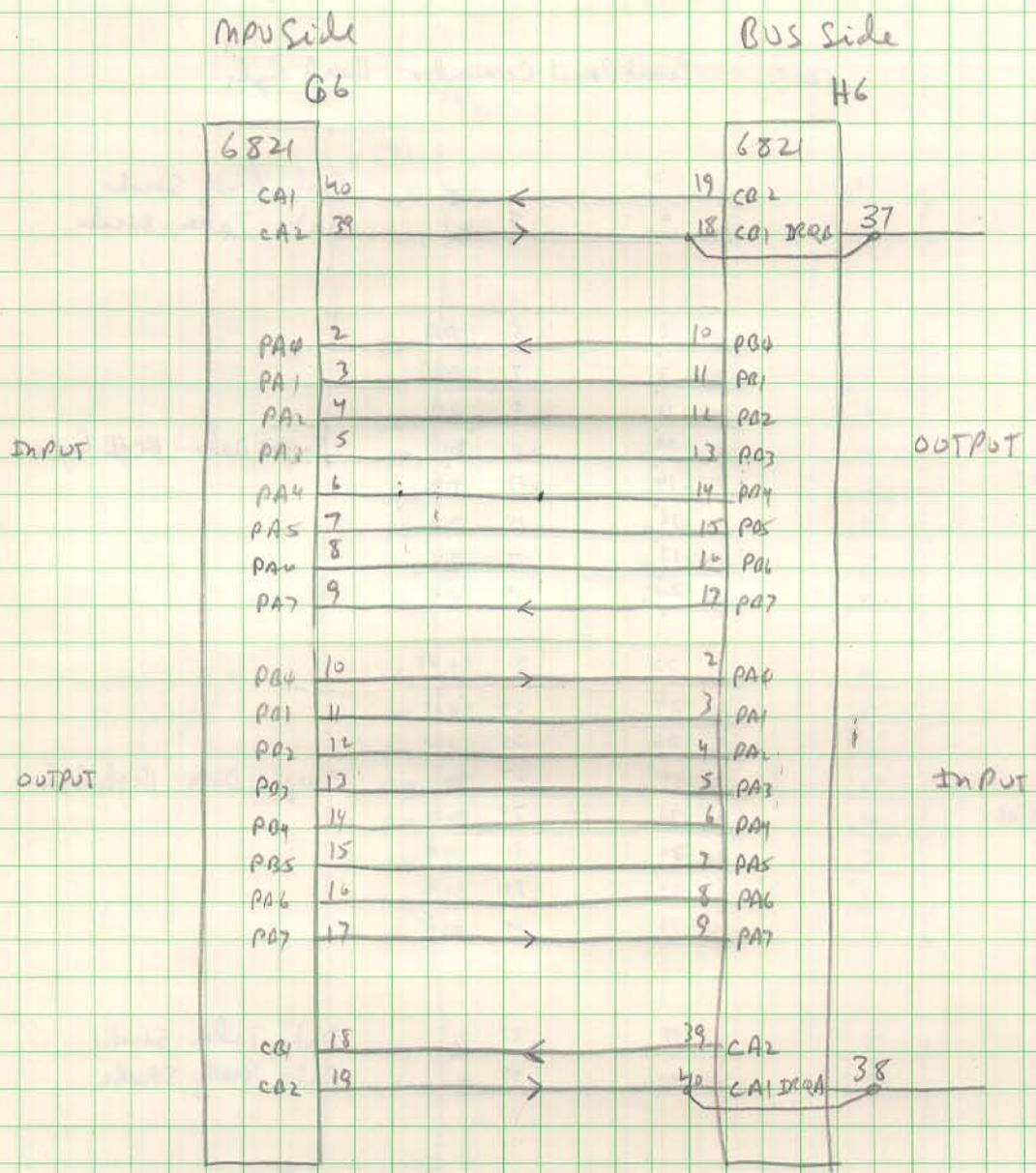
40 pin Ribbon Connector

Front Panel Connector High Byte

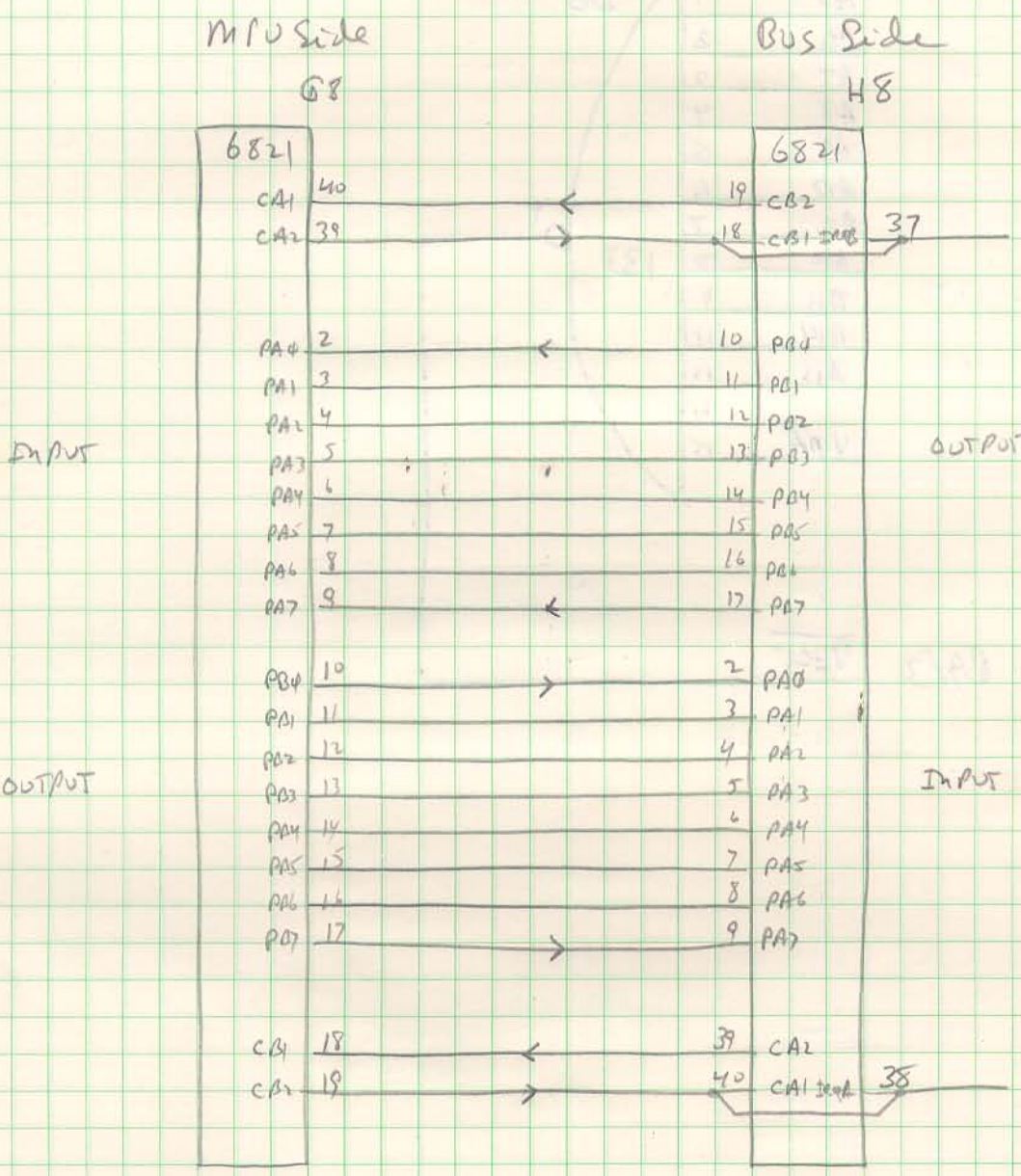


26 Sept 81
ARD

Low Byte BUS Port Wiring

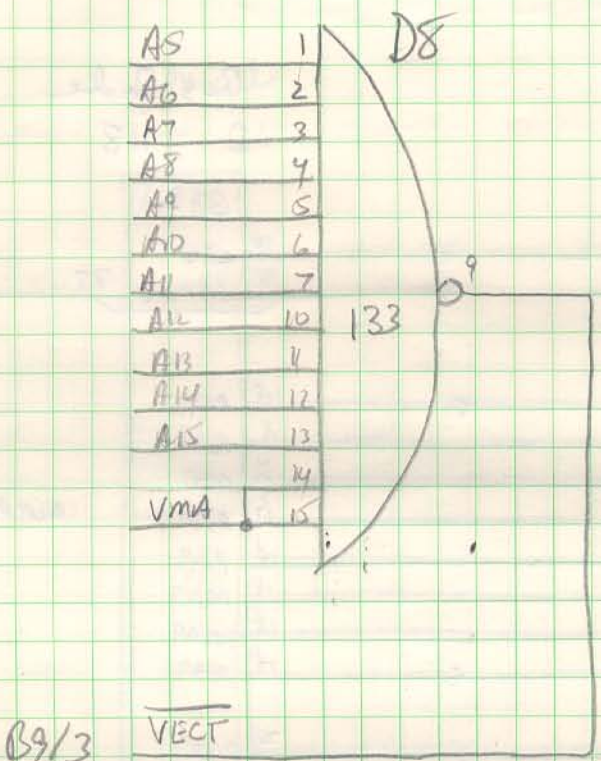


High Byte BUS PORT Wiring



26 Sept 81
ARS

Changes


A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
VMA
1
2
3
4
5
6
7
10
11
12
13
14
15
D8
9
133
B9/3
VECT

Parallel Port IC locations

A1 (A6)	6821	33, 36
A3 (A8)	6821	33, 37
B1 (B6)	2114	34
B2 (B7)	2114	34
B3 (B8)	—	
B4 (B9)	6828	34
C1 (C6)	2114	34
C2 (C7)	2114	34
C3 (C8)	5472 (18542)	35
D1 (D6)	2114	34
D2 (D7)	2114	34
D3 (D8)	—	
D4 (D9)	RES	33

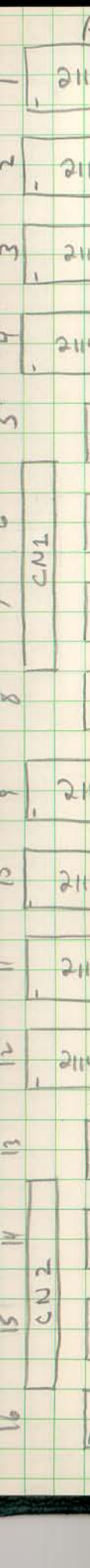
E1 (E6)	2114	34
E2 (E7)	2114	34
E3 (E8)	5287 (14510)	35
E4 (E9)	132	33
F1 (F6)	2716	34
F2 (F7)	2716	34
F3 (F8)	6802	33
G1 (G6)	6821	33, 38
G3 (G8)	6821	33, 38
G5	5287 (14510)	32
H1 (H6)	6821	32, 38
H3 (H8)	6821	32, 38

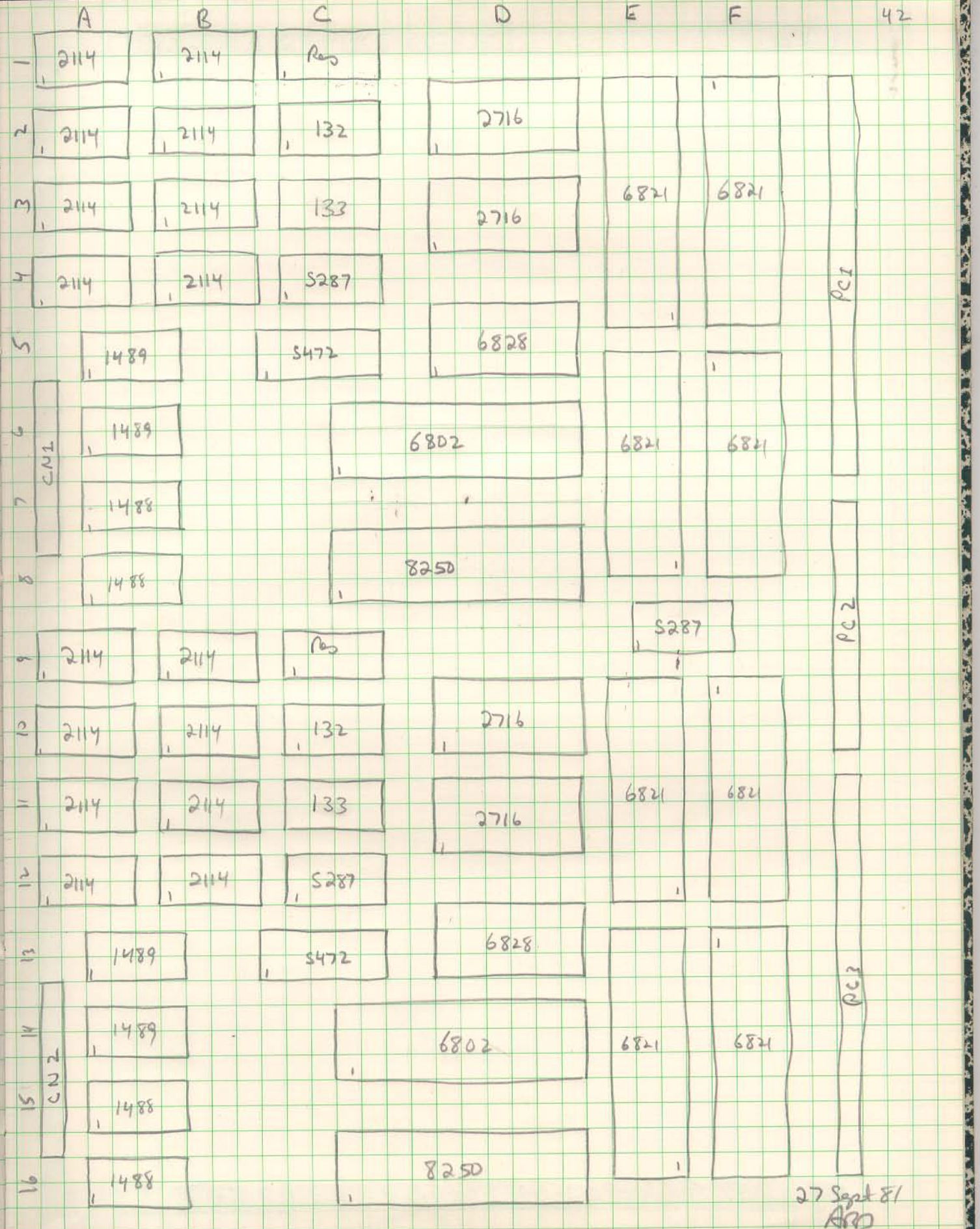
26 Sept 81
AHB

DUAL Serial Port

Programmable Asynchronous Serial Port

Using INS 8250 ACE - See datasheets




A
B
C
D
E
F
42
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
2114
Rom
132
133
5287
1489
1488
2716
6828
6802
8250
6821
CN1
CN2
PC1
PC2
27 Sept 81
ARD

National Semiconductor

Preliminary
JULY 1978

INS8250 Asynchronous Communications Element

General Description

The INS8250 is a programmable Asynchronous Communications Element (ACE) chip contained in a standard 40-pin dual-in-line package. The chip, which is fabricated using N-channel silicon gate technology, functions as a serial data input/output interface in a microcomputer system. The functional configuration of the INS8250 is programmed by the system software via a TRI-STATE® 8-bit bidirectional data bus.

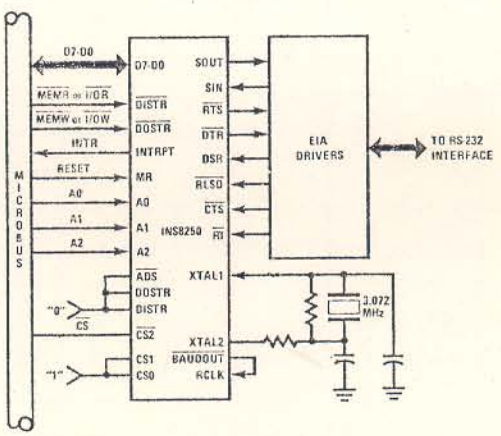
The INS8250 performs serial-to-parallel conversion on data characters received from a peripheral device or a MODEM, and parallel-to-serial conversion on data characters received from the CPU. The CPU can read the complete status of the INS8250 at any time during the functional operation. Status information reported includes the type and condition of the transfer operations being performed by the INS8250, as well as any error conditions (parity, overrun, framing, or break interrupt).

In addition to providing control of asynchronous data communications, the INS8250 includes a programmable Baud Generator that is capable of dividing the timing reference clock input by divisors of 1 to $(2^{16} - 1)$, and producing a 16x clock for driving the internal transmitter logic. Provisions are also included to use this 16x clock to drive the receiver logic. Also included in the INS8250 is a complete MODEM-control capability, and a processor-interrupt system that may be software tailored to the user's requirements to minimize the computing time required to handle the communications link.

Features

- Designed to be Easily Interfaced to Most Popular Microprocessors.
- Adds or Deletes Standard Asynchronous Communication Bits (Start, Stop, and Parity) to or from Serial Data Stream
- Full Double Buffering Eliminates Need for Precise Synchronization
- Independently Controlled Transmit, Receive, Line Status, and Data Set Interrupts
- Programmable Baud Rate Generator Allows Division of Any Input Clock by 1 to $(2^{16} - 1)$ and Generates the Internal 16x Clock
- Independent Receiver Clock Input
- MODEM Control Functions (CTS, RTS, DSR, DTR, RI, and Carrier Detect)
- Fully Programmable Serial-Interface Characteristics
  - 5-, 6-, 7-, or 8-Bit Characters
  - Even, Odd, or No-Parity Bit Generation and Detection
  - 1-, 1½-, or 2-Stop Bit Generation
  - Baud Rate Generation (DC to 56k Baud)
- False Start Bit Detection
- Complete Status Reporting Capabilities
- TRI-STATE TTL Drive Capabilities for Bidirectional Data Bus and Control Bus
- Line Break Generation and Detection
- Internal Diagnostic Capabilities
  - Loopback Controls for Communications Link Fault Isolation
  - Break, Parity, Overrun, Framing Error Simulation
- Full Prioritized Interrupt System Controls
- Single +5-Volt Power Supply
- MICROBUS™* Compatible

INS8250 MICROBUS Configuration



*Trademark, National Semiconductor Corp.

 DA-B25M78/Printed in U.S.A.

Absolute Maximum Ratings

Temperature Under Bias . . . . . 0°C to +70°C
Storage Temperature . . . . . −65°C to +150°C
All Input or Output Voltages with Respect to V_{SS} . . . . −0.5 V to +7.0 V
Power Dissipation . . . . . 400 mW

Absolute maximum ratings indicate limits beyond which permanent damage may occur. Continuous operation at these limits is not intended; operation should be limited to those conditions specified under DC Electrical Characteristics.

DC Electrical Characteristics

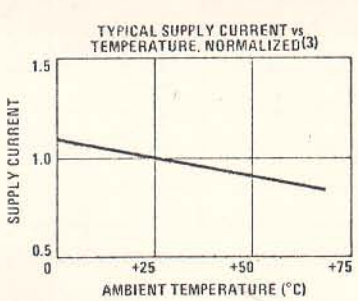
T_A = 0°C to +70°C, V_{CC} = +5 V ± 5%, V_{SS} = 0 V, unless otherwise specified.

Symbol	Parameter	Min	Typ	Max	Units	Test Conditions
V_{ILX}	Clock Input Low Voltage	−0.5		0.8	V	
V_{IHX}	Clock Input High Voltage	2.0		V_{CC}	V	
V_{IL}	Input Low Voltage	−0.5		0.8	V	
V_{IH}	Input High Voltage	2.0		V_{CC}	V	
V_{OL}	Output Low Voltage			0.4	V	I_{OL} = 1.6 mA on all outputs,
V_{OH}	Output High Voltage	2.4			V	I_{OH} = −100 μA
I_{CC} (AV)	Avg Power Supply Current (V_{CC})		65	80	mA	
I_{IL}	Input Leakage			±10	μA	
I_{CL}	Clock Leakage			±10	μA	

Capacitance

T_A = 25°C, V_{CC} = V_{SS} = 0 V

Symbol	Parameter	Min	Typ	Max	Units	Test Conditions
C_{XIN}	Clock Input Capacitance		15	20	pF	
C_{XOUT}	Clock Output Capacitance		20	30	pF	f_c = 1 MHz
C_{IN}	Input Capacitance		6	10	pF	Unmeasured pins returned to V_{SS}
C_{OUT}	Output Capacitance		10	20	pF	



AC Electrical Characteristics

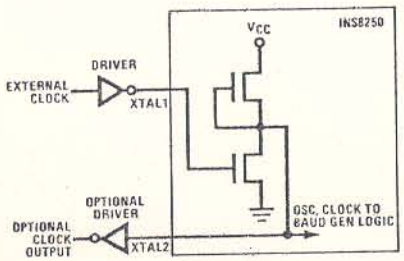
$T_A = 0°C$ to $\pm 70°C$, $V_{CC} = +5\ V \pm 5\%$

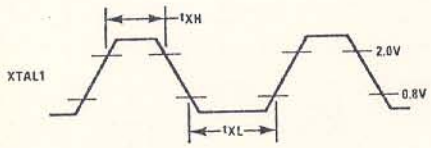
Symbol	Parameter	Min	Max	Units	Test Conditions
t_{AW}	Address Strobe Width	90	—	ns	
t_{AS}	Address Setup Time	110	—	ns	
t_{AH}	Address Hold Time	0	—	ns	
t_{CS}	Chip Select Setup Time	110	—	ns	
t_{CH}	Chip Select Hold Time	0	—	ns	
t_{CSS}	Chip Select Output Delay from Strobe	0	90	ns	-@ 100 pF loading
t_{DID}	$\overline{DISTR}$/DISTR Strobe Delay	0	—	ns	
t_{DIW}	$\overline{DISTR}$/DISTR Strobe Width	175	—	ns	
t_{RC}	Read Cycle Delay	1735	—	ns	
RC	Read Cycle = $t_{AW} + t_{DID} + t_{DIW} + t_{RC}$	2000	—	ns	
t_{DD}	$\overline{DISTR}$/DISTR to Driver Disable Delay	—	150	ns	-@ 100 pF loading
t_{DDD}	Delay from $\overline{DISTR}$/DISTR to Data	—	250	ns	-@ 100 pF loading
t_{HZ}	$\overline{DISTR}$/DISTR to Floating Data Delay	100	—	ns	-@ 100 pF loading
t_{DOD}	$\overline{DOSTR}$/DOSTR Strobe Delay	50	—	ns	
t_{DOW}	$\overline{DOSTR}$/DOSTR Strobe Width	175	—	ns	
t_{WC}	Write Cycle Delay	1785	—	ns	
WC	Write Cycle = $t_{AW} + t_{DOD} + t_{DOW} + t_{WC}$	2100	—	ns	
t_{DS}	Data Setup Time	175	—	ns	
t_{DH}	Data Hold Time	60	—	ns	
t_{CSC}*	Chip Select Output Delay from Select	—	200	ns	-@ 100 pF loading
t_{RA}*	Address Hold Time from $\overline{DISTR}$/DISTR	50	—	ns	
t_{RCS}*	Chip Select Hold Time from $\overline{DISTR}$/DISTR	50	—	ns	
t_{AR}*	$\overline{DISTR}$/DISTR Delay from Address	110	—	ns	
t_{CSR}*	$\overline{DISTR}$/DISTR Delay from Chip Select	110	—	ns	
t_{WA}*	Address Hold Time from $\overline{DOSTR}$/DOSTR	50	—	ns	
t_{WCS}*	Chip Select Hold Time from $\overline{DOSTR}$/DOSTR	50	—	ns	
t_{AW}*	$\overline{DOSTR}$/DOSTR Delay from Address	160	—	ns	
t_{CSW}*	$\overline{DOSTR}$/DOSTR Delay from Select	160	—	ns	
t_{MRW}	Master Reset Pulse Width	25	—	μs	

*Applicable only when $\overline{ADS}$ input is tied permanently low.

AC Electrical Characteristics (cont'd.)

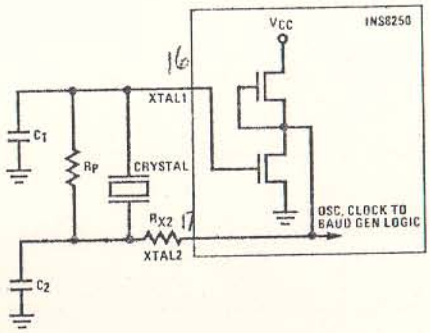
Symbol	Parameter	Min	Max	Units	Test Conditions
BAUD GENERATOR					
N	Baud Rate Divisor	1	$2^{16}-1$		
t_{BLD}	Baud Output Negative Edge Delay		250 typ	ns	100 pF Load
t_{BHD}	Baud Output Positive Edge Delay		250 typ	ns	100 pF Load
t_{LW}	Baud Output Down Time	425 typ		ns	100 pF Load
t_{HW}	Baud Output Up Time	330 typ		ns	100 pF Load
RECEIVER					
t_{SCD}	Delay from RCLK to Sample Time		2 typ	μs	
t_{SINT}	Delay from Stop to Set Interrupt		2 typ	μs	100 pF Load
t_{RINT}	Delay from $\overline{\text{DISTR}}$/DISTR (RD RBR/RDLSR) to Reset Interrupt		1 typ	μs	100 pF Load
TRANSMITTER					
t_{HR}	Delay from $\overline{\text{DOSTR}}$/DOSTR (WR THR) to Reset Interrupt		1 typ	μs	100 pF Load
t_{IRS}	Delay from Initial INTR Reset to Transmit Start		16 typ	$\overline{\text{BAUDOUT}}$ Cycles	
t_{SI}	Delay from Initial Write to Interrupt		24 typ	$\overline{\text{BAUDOUT}}$ Cycles	
t_{SS}	Delay from Stop to Next Start		1 typ	μs	
t_{STI}	Delay from Stop to Interrupt (THRE)		8 typ	$\overline{\text{BAUDOUT}}$ Cycles	
t_{IR}	Delay from $\overline{\text{DISTR}}$/DISTR (RD IIR) to Reset Interrupt (THRE)		1 typ	μs	100 pF Load
MODEM CONTROL					
t_{MDO}	Delay from DOSTR/$\overline{\text{DOSTR}}$ (WR MCR) to Output		1 typ	μs	100 pF Load
t_{SIM}	Delay to Set Interrupt from MODEM Input		1 typ	μs	100 pF Load
t_{RIM}	Delay to Reset Interrupt from $\overline{\text{DISTR}}$/DISTR (RD MSR)		1 typ	μs	100 pF Load





Timing	Min	Units
t_{XH}	140	ns
t_{XL}	140	ns

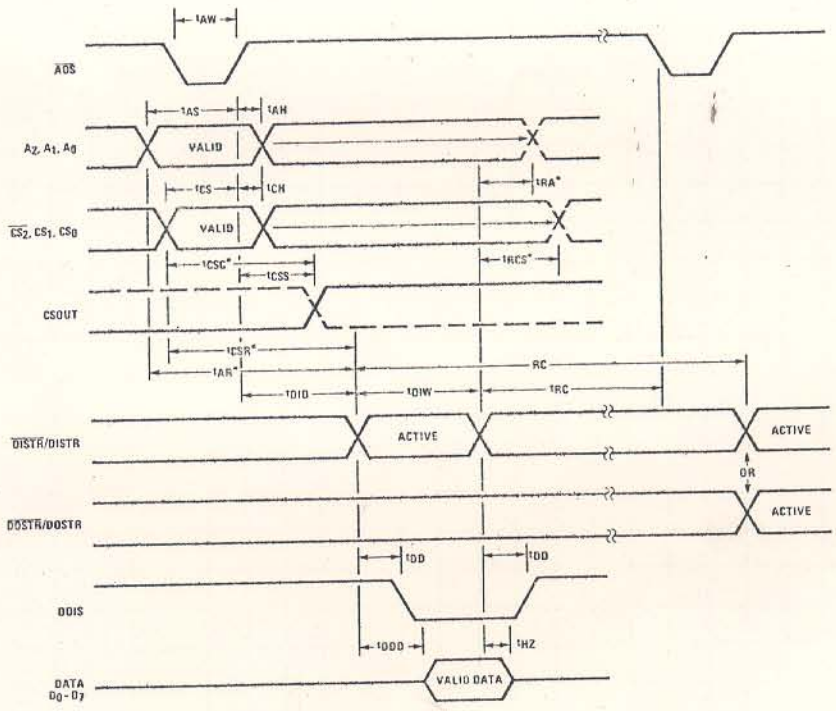
A. External Clock Input (3.1 MHz Max.)



CRYSTAL	RP	RX2	C1	C2
3.1 MHz	1 MΩ	1.5K	10 - 30 pF	40 - 60 pF

B. Typical Crystal Oscillator Network

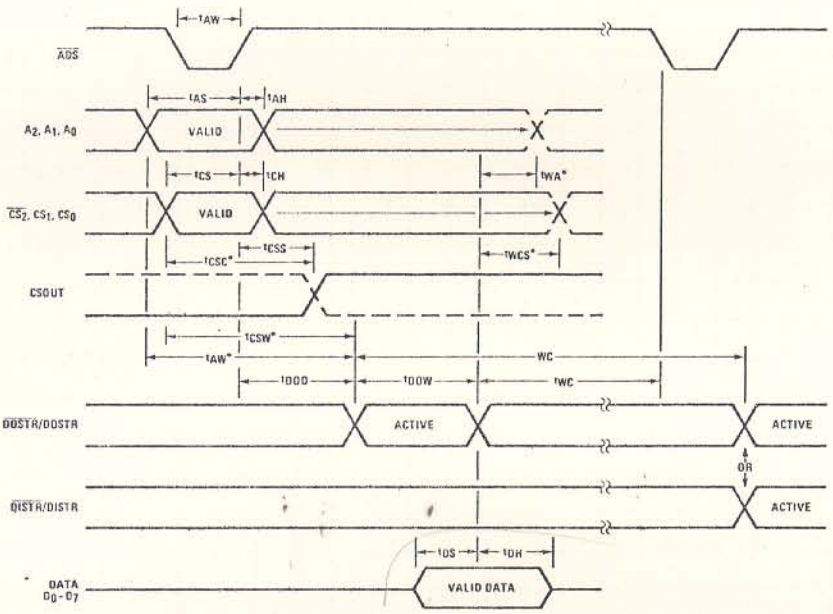
Timing Waveforms



*APPLICABLE ONLY WHEN ADS IS TIED LOW.

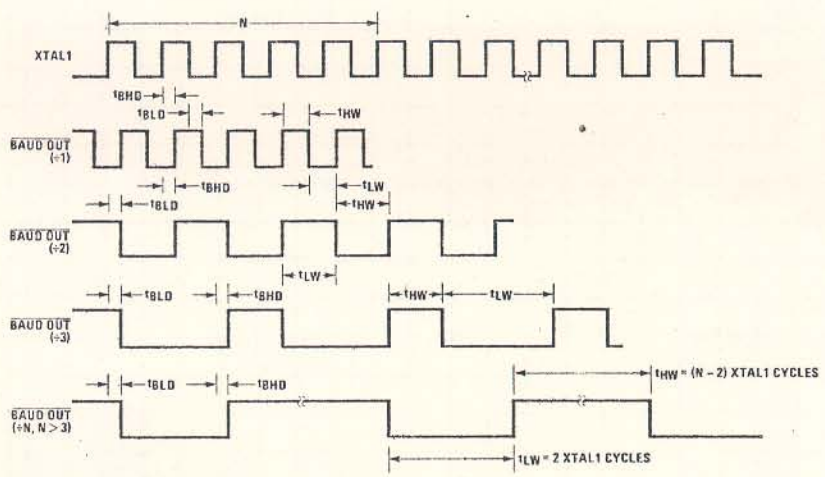
Read Cycle

Timing Waveforms (cont'd.)



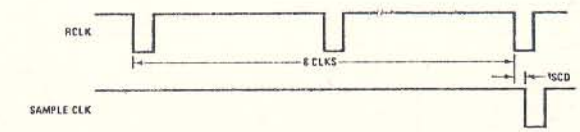
*APPLICABLE ONLY WHEN ADS IS TIED LOW.

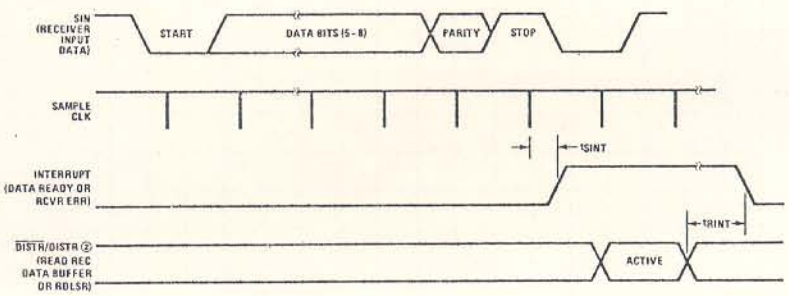
Write Cycle



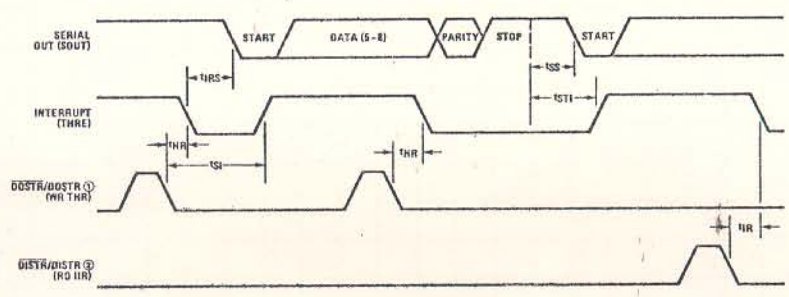
BAUDOUT Timing

Timing Waveforms (cont'd.)

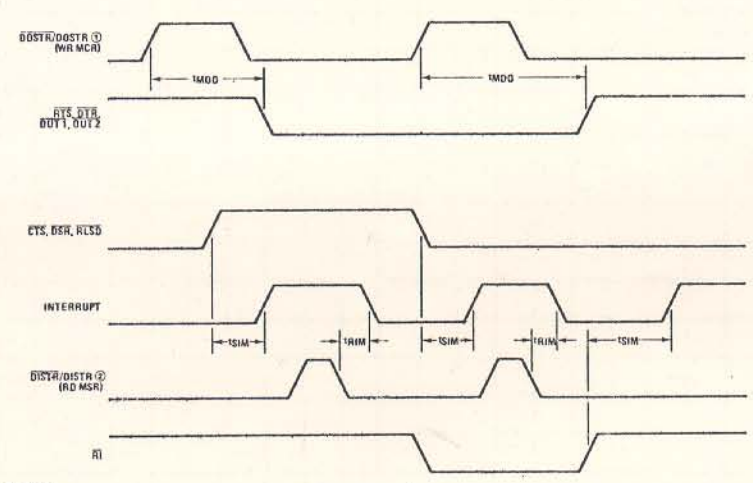




Receiver Timing



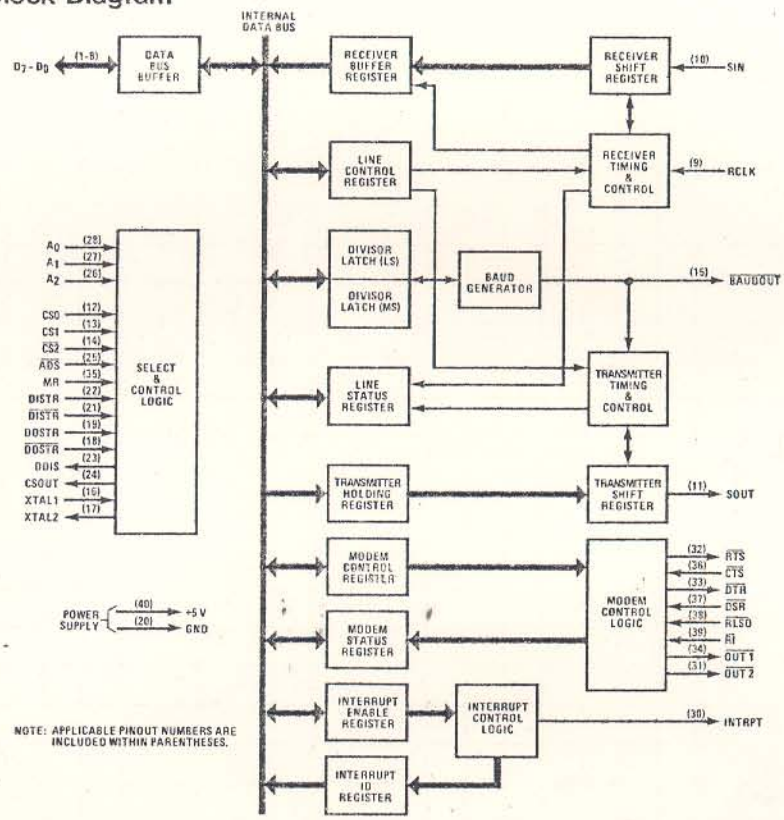
Transmitter Timing



MODEM Controls Timing

NOTES:
① See Write Cycle Timing
② See Read Cycle Timing

INS8250 Block Diagram



INS8250 Functional Pin Description

The following describes the function of all INS8250 input/output pins. Some of these descriptions reference internal circuits.

NOTE

In the following descriptions, a low represents a logic 0 (0 volt nominal) and a high represents a logic 1 (+2.4 volts nominal).

INPUT SIGNALS

Chip Select (CS0, CS1, $\overline{CS2}$), Pins 12 - 14: When CS0 and CS1 are high and $\overline{CS2}$ is low, the chip is selected. Chip selection is complete when the decoded chip select signal is latched with an active (low) Address Strobe ($\overline{ADS}$) input. This enables communication between the INS8250 and the CPU.

Data Input Strobe (DISTR, $\overline{DISTR}$), Pins 22 and 21: When DISTR is high or $\overline{DISTR}$ is low while the chip is selected, allows the CPU to read status information or data from a selected register of the INS8250.

NOTE

Only an active DISTR or $\overline{DISTR}$ input is required to transfer data from the INS8250 during a read operation. Therefore, tie either the DISTR input permanently low or the $\overline{DISTR}$ input permanently high, if not used.

Data Output Strobe (DOSTR, $\overline{DOSTR}$), Pins 19 and 18: When DOSTR is high or $\overline{DOSTR}$ is low while the chip is selected, allows the CPU to write data or control words into a selected register of the INS8250.

NOTE

Only an active DOSTR or $\overline{DOSTR}$ input is required to transfer data to the INS8250 during a write operation. Therefore, tie either the DOSTR input permanently low or the $\overline{DOSTR}$ input permanently high, if not used.

Address Strobe ($\overline{ADS}$), Pin 25: When low, provides latching for the Register Select (A0, A1, A2) and Chip Select (CS0, CS1, $\overline{CS2}$) signals.

NOTE

An active $\overline{ADS}$ input is required when the Register Select (A0, A1, A2) signals are not stable for the duration of a read or write operation. If not required, tie the $\overline{ADS}$ input permanently low.

Register Select (A0, A1, A2), Pins 26 - 28: These three inputs are used during a read or write operation to select an INS8250 register to read from or write into as indicated in the table below. Note that the state of the Divisor Latch Access Bit (DLAB), which is the most significant bit of the Line Control Register, affects the selection of certain INS8250 registers. The DLAB must be set high by the system software to access the Baud Generator Divisor Latches.

DLAB	A_2	A_1	A_0	Register
0	0	0	0	Receiver Buffer (read), Transmitter Holding Register (write)
0	0	0	1	Interrupt Enable
X	0	1	0	Interrupt Identification (read only)
X	0	1	1	Line Control
X	1	0	0	MODEM Control
X	1	0	1	Line Status
X	1	1	0	MODEM Status
X	1	1	1	None
1	0	0	0	Divisor Latch (least significant byte)
1	0	0	1	Divisor Latch (most significant byte)

Master Reset (MR), Pin 35: When high, clears all the registers (except the Receiver Buffer, Transmitter Holding, and Divisor Latches), and the control logic of the INS8250. Also, the state of various output signals (SOUT, INTRPT, $\overline{OUT\,1}$, $\overline{OUT\,2}$, $\overline{RTS}$, $\overline{DTR}$) are affected by an active MR input. (Refer to table 1.)

Receiver Clock (RCLK), Pin 9: This input is the 16x baud rate clock for the receiver section of the chip.

Serial Input (SIN), Pin 10: Serial data input from the communications link (peripheral device, MODEM, or data set).

Clear to Send ($\overline{CTS}$), Pin 36: The $\overline{CTS}$ signal is a MODEM control function input whose condition can be tested by the CPU by reading bit 4 (CTS) of the MODEM Status Register. Bit 0 (DCTS) of the MODEM Status Register indicates whether the $\overline{CTS}$ input has changed state since the previous reading of the MODEM Status Register.

NOTE

Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Data Set Ready ($\overline{DSR}$), Pin 37: When low, indicates that the MODEM or data set is ready to establish the communications link and transfer data with the INS8250. The $\overline{DSR}$ signal is a MODEM-control function input whose condition can be tested by the CPU by reading bit 5 (DSR) of the MODEM Status Register. Bit 1 (DDSR) of the MODEM Status Register indicates whether the $\overline{DSR}$ input has changed state since the previous reading of the MODEM Status Register.

NOTE

Whenever the DSR bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Received Line Signal Detect ($\overline{RLSD}$), Pin 38: When low, indicates that the data carrier has been detected by the MODEM or data set. The $\overline{RLSD}$ signal is a MODEM-control function input whose condition can be tested by the CPU by reading bit 7 (RLSD) of the MODEM Status Register. Bit 3 (DRLSD) of the MODEM Status Register indicates whether the $\overline{RLSD}$ input has changed state since the previous reading of the MODEM Status Register.

NOTE

Whenever the RLSD bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Ring Indicator ($\overline{RI}$), Pin 39: When low, indicates that a telephone ringing signal has been received by the MODEM or data set. The $\overline{RI}$ signal is a MODEM-control function input whose condition can be tested by the CPU by reading bit 6 (RI) of the MODEM Status Register. Bit 2 (TERI) of the MODEM Status Register indicates whether the $\overline{RI}$ input has changed from a low to a high state since the previous reading of the MODEM Status Register.

NOTE

Whenever the RI bit of the MODEM Status Register changes from a high to a low state, an interrupt is generated if the MODEM Status Interrupt is enabled.

V_{CC}, Pin 40: +5-volt supply.

V_{SS}, Pin 20: Ground (0-volt) reference.

OUTPUT SIGNALS

Data Terminal Ready ($\overline{DTR}$), Pin 33: When low, informs the MODEM or data set that the INS8250 is ready to communicate. The DTR output signal can be set to an active low by programming bit 0 (DTR) of the MODEM Control Register to a high level. The $\overline{DTR}$ signal is set high upon a Master Reset operation.

Request to Send ($\overline{RTS}$), Pin 32: When low, informs the MODEM or data set that the INS8250 is ready to transmit data. The $\overline{RTS}$ output signal can be set to an active low by programming bit 1 (RTS) of the MODEM Control Register. The $\overline{RTS}$ signal is set high upon a Master Reset operation.

Output 1 ($\overline{OUT\,1}$), Pin 34: User-designated output that can be set to an active low by programming bit 2 ($\overline{OUT\,1}$) of the MODEM Control Register to a high level. The $\overline{OUT\,1}$ signal is set high upon a Master Reset operation.

Output 2 ($\overline{OUT\,2}$), Pin 31: User-designated output that can be set to an active low by programming bit 3 ($\overline{OUT\,2}$) of the MODEM Control Register to a high level. The $\overline{OUT\,2}$ signal is set high upon a Master Reset operation.

Chip Select Out (CSOUT), Pin 24: When high, indicates that the chip has been selected by active CS0, CS1, and $\overline{CS2}$ inputs. No data transfer can be initiated until the CSOUT signal is a logic 1.

Driver Disable (DDIS), Pin 23: Goes low whenever the CPU is reading data from the INS8250. A high-level DDIS output can be used to disable an external transceiver (if used between the CPU and INS8250 on the D_7-D_0 Data Bus) at all times, except when the CPU is reading data.

Baud Out ($\overline{BAUDOUT}$), Pin 15: 16x clock signal for the transmitter section of the INS8250. The clock rate is equal to the main reference oscillator frequency divided by the specified divisor in the Baud Generator Divisor Latches. The $\overline{BAUDOUT}$ may also be used for the receiver section by tying this output to the RCLK input of the chip.

Interrupt (INTRPT), Pin 30: Goes high whenever any one of the following interrupt types has an active high condition and is enabled via the IER: Receiver Error Flag; Received Data Available; Transmitter Holding Register Empty; and MODEM Status. The INTRPT signal is reset low upon the appropriate interrupt service or a Master Reset operation.

Serial Output (SOUT), Pin 11: Composite serial data output to the communications link (peripheral, MODEM or data set). The SOUT signal is set to the Marking (logic 1) state upon a Master Reset operation.

INPUT/OUTPUT SIGNALS

Data (D_7 - D_0) Bus, Pins 1 - 8: This bus comprises eight TRI-STATE input/output lines. The bus provides bidirectional communications between the INS8250 and the CPU. Data, control words, and status information are transferred via the D_7 - D_0 Data Bus.

External Clock Input/Output (XTAL 1, XTAL 2), Pins 16 and 17: These two pins connect the main timing reference (crystal or signal clock) to the INS8250.

Pin Configuration

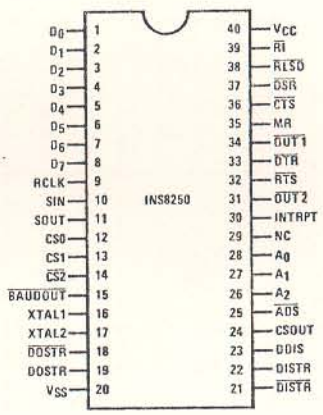


Table 1. ACE Reset Functions

Register/Signal	Reset Control	Reset State
Interrupt Enable Register	Master Reset	All Bits Low (0 - 3 forced and 4 - 7 permanent)
Interrupt Identification Register	Master Reset	Bit 0 is High, Bits 1 and 2 Low Bits 3 - 7 are Permanently Low
Line Control Register	Master Reset	All Bits Low
MODEM Control Register	Master Reset	All Bits Low
Line Status Register	Master Reset	All Bits Low, Except Bits 5 & 6 are High
MODEM Status Register	Master Reset	Bits 0 - 3 Low Bits 4 - 7 — Input Signal
SOUT	Master Reset	High
INTRPT (RCVR Errs)	Read LSR/MR	Low
INTRPT (RCVR Data Ready)	Read RBR/MR	Low
INTRPT (THRE)	Read IIR/Write THR/MR	Low
INTRPT (Modem Status Changes)	Read MSR/MR	Low
$\overline{OUT\ 2}$	Master Reset	High
$\overline{RTS}$	Master Reset	High
$\overline{DTR}$	Master Reset	High
$\overline{OUT\ 1}$	Master Reset	High

INS8250 Accessible Registers

The system programmer may access or control any of the INS8250 registers summarized in table 2 via the CPU. These registers are used to control INS8250 operations and to transmit and receive data.

INS8250 LINE CONTROL REGISTER

The system programmer specifies the format of the asynchronous data communications exchange via the Line Control Register. In addition to controlling the format, the programmer may retrieve the contents of the Line Control Register for inspection. This feature simplifies system programming and eliminates the need for separate storage in system memory of the line characteristics. The contents of the Line Control Register are indicated in table 2 and are described below.

Bits 0 and 1: These two bits specify the number of bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:

Bit 1	Bit 0	Word Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

Bit 2: This bit specifies the number of Stop bits in each transmitted or received serial character. If bit 2 is a logic 0, 1 Stop bit is generated or checked in the transmit or receive data, respectively. If bit 2 is a logic 1 when a 5-bit word length is selected via bits 0 and 1, 1½ Stop bits are generated or checked. If bit 2 is a logic 1 when either a 6-, 7-, or 8-bit word length is selected, 2 Stop bits are generated or checked.

Table 2. Summary of INS8250 Accessible Registers

	Register Address									
	0 DLAB = 0	0 DLAB = 0	1 DLAB = 0	2	3	4	5	6	0 DLAB = 1	1 DLAB = 1
Bit No.	Receiver Buffer Register (Read Only)	Transmitter Holding Register (Write Only)	Interrupt Enable Register	Interrupt Identification Register (Read Only)	Line Control Register	MODEM Control Register	Line Status Register	MODEM Status Register	Divisor Latch (LS)	Divisor Latch (MS)
	RBR	THR	IER	IIR	LCR	MCR	LSR	MSR	DLL	DLM
0	Data Bit 0*	Data Bit 0	Enable Received Data Available Interrupt (ERBFI)	"0" if Interrupt Pending	Word Length Select Bit 0 (WLS0)	Data Terminal Ready (DTR)	Data Ready (DR)	Delta Clear to Send (DCTS)	Bit 0	Bit 8
1	Data Bit 1	Data Bit 1	Enable Transmitter Holding Register Empty Interrupt (ETBEI)	Interrupt ID Bit (0)	Word Length Select Bit 1 (WLS1)	Request to Send (RTS)	Overrun Error (OR)	Delta Data Set Ready (DDSR)	Bit 1	Bit 9
2	Data Bit 2	Data Bit 2	Enable Receiver Line Status Interrupt (ELSI)	Interrupt ID Bit (1)	Number of Stop Bits (STB)	Out 1	Parity Error (PE)	Trailing Edge Ring Indicator (TERI)	Bit 2	Bit 10
3	Data Bit 3	Data Bit 3	Enable MODEM Status Interrupt (EDSSI)	0	Parity Enable (PEN)	Out 2	Framing Error (FE)	Delta Receive Line Signal Detect (DRLSD)	Bit 3	Bit 11
4	Data Bit 4	Data Bit 4	0	0	Even Parity Select (EPS)	Loop	Break Interrupt (BI)	Clear to Send (CTS)	Bit 4	Bit 12
5	Data Bit 5	Data Bit 5	0	0	Stick Parity	0	Transmitter Holding Register Empty (THRE)	Data Set Ready (DSR)	Bit 5	Bit 13
6	Data Bit 6	Data Bit 6	0	0	Set Break	0	Transmitter Shift Register Empty (TSRE)	Ring Indicator (RI)	Bit 6	Bit 14
7	Data Bit 7	Data Bit 7	0	0	Divisor Latch Access Bit (DLAB)	0	0	Received Line Signal Detect (RLSD)	Bit 7	Bit 15

* Bit 0 is the least significant bit. It is the first bit serially transmitted or received.

Bit 3: This bit is the Parity Enable bit. When bit 3 is a logic 1, a Parity bit is generated (transmit data) or checked (receive data) between the last data word bit and Stop bit of the serial data. (The Parity bit is used to produce an even or odd number of 1s when the data word bits and the Parity bit are summed.)

Bit 4: This bit is the Even Parity Select bit. When bit 3 is a logic 1 and bit 4 is a logic 0, an odd number of logic 1s is transmitted or checked in the data word bits and Parity bit. When bit 3 is a logic 1 and bit 4 is a logic 1, an even number of bits is transmitted or checked.

Bit 5: This bit is the Stick Parity bit. When bit 3 is a logic 1 and bit 5 is a logic 1, the Parity bit is transmitted and then detected by the receiver as a logic 0 if bit 4 is a logic 1 or as a logic 1 if bit 4 is a logic 0.

Bit 6: This bit is the Set Break Control bit. When bit 6 is a logic 1, the serial output (SOUT) is forced to the Spacing (logic 0) state and remains there regardless of other transmitter activity. The set break is disabled by setting bit 6 to a logic 0. This feature enables the CPU to alert a terminal in a computer communications system.

Bit 7: This bit is the Divisor Latch Access Bit (DLAB). It must be set high (logic 1) to access the Divisor Latches of the Baud Rate Generator during a Read or Write operation. It must be set low (logic 0) to access the Receiver Buffer, the Transmitter Holding Register, or the Interrupt Enable Register.

INS8250 PROGRAMMABLE BAUD RATE GENERATOR

The INS8250 contains a programmable Baud Rate Generator that is capable of taking any clock input (DC to 3.1 MHz) and dividing it by any divisor from 1 to $(2^{16}-1)$. The output frequency of the Baud Generator is 16x the Baud rate [divisor # = (frequency input) ÷ (baud rate x 16)]. Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization in order to insure desired operation of the Baud Rate Generator. Upon loading either of the Divisor Latches, a 16-bit Baud counter is immediately loaded. This prevents long counts on initial load.

Tables 3 and 4 illustrate the use of the Baud Rate Generator with crystal frequencies of 1.8432 MHz and 3.072 MHz respectively. For baud rates of 38400 and below the error obtained is minimal. The accuracy of the desired baud rate is dependent on the crystal frequency chosen.

NOTE

The maximum operating frequency of the Baud Generator is 3.1 MHz. However, when using divisors of 3 and below, the maximum frequency is equal to the divisor in MHz. For example, if the divisor is 1, then the maximum frequency is 1 MHz. In no case should the data rate be greater than 56k Baud.

LINE STATUS REGISTER

This 8-bit register provides status information to the CPU concerning the data transfer. The contents of the Line Status Register are indicated in table 2 and are described below.

Bit 0: This bit is the receiver Data Ready (DR) indicator. Bit 0 is set to a logic 1 whenever a complete incoming character has been received and transferred into the Receiver Buffer Register. Bit 0 may be reset to a logic 0 either by the CPU reading the data in the Receiver Buffer Register or by writing a logic 0 into it from the CPU.

Bit 1: This bit is the Overrun Error (OE) indicator. Bit 1 indicates that data in the Receiver Buffer Register was not read by the CPU before the next character was transferred into the Receiver Buffer Register, thereby destroying the previous character. The OE indicator is reset whenever the CPU reads the contents of the Line Status Register.

Bit 2: This bit is the Parity Error (PE) indicator. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even-parity-select bit. The PE bit is set to a logic 1 upon detection of a parity error and is reset to a logic 0 whenever the CPU reads the contents of the Line Status Register.

Table 3. Baud Rates Using 1.8432 MHz Crystal

Desired Baud Rate	Divisor Used to Generate 16x Clock	Percent Error Difference Between Desired & Actual
50	2304	–
75	1536	–
110	1047	0.026
134.5	857	0.058
150	768	–
300	384	–
600	192	–
1200	96	–
1800	64	–
2000	58	0.69
2400	48	–
3600	32	–
4800	24	–
7200	16	–
9600	12	–
19200	6	–
38400	3	–
56000	2	2.86

NOTE: 1.8432 MHz is the standard 8080 frequency divided by 10.

Table 4. Baud Rates Using 3.072 MHz Crystal

Desired Baud Rate	Divisor Used to Generate 16x Clock	Percent Error Difference Between Desired & Actual
50	3840	–
75	2560	–
110	1745	0.026
134.5	1428	0.034
150	1280	–
300	640	–
600	320	–
1200	160	–
1800	107	0.312
2000	96	–
2400	80	–
3600	53	0.628
4800	40	–
7200	27	1.23
9600	20	–
19200	10	–
38400	5	–

Bit 3: This bit is the Framing Error (FE) indicator. Bit 3 indicates that the received character did not have a valid Stop bit. Bit 3 is set to a logic 1 whenever the Stop bit following the last data bit or parity bit is detected as a zero bit (Spacing level).

Bit 4: This bit is the Break Interrupt (BI) indicator. Bit 4 is set to a logic 1 whenever the received data input is held in the Spacing (logic 0) state for longer than a full word transmission time (that is, the total time of Start bit + data bits + Parity + Stop bits).

NOTE

Bits 1 through 4 are the error conditions that produce a Receiver Line Status interrupt whenever any of the corresponding conditions are detected.

Bit 5: This bit is the Transmitter Holding Register Empty (THRE) indicator. Bit 5 indicates that the INS8250 is ready to accept a new character for transmission. In addition, this bit causes the INS8250 to issue an interrupt to the CPU when the Transmit Holding Register Empty Interrupt enable is set high. The THRE bit is set to a logic 1 when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to logic 0 concurrently with the loading of the Transmitter Holding Register by the CPU.

Bit 6: This bit is the Transmitter Shift Register Empty (TSRE) indicator. Bit 6 is set to a logic 1 whenever the Transmitter Shift Register is idle. It is reset to logic 0 upon a data transfer from the Transmitter Holding Register to the Transmitter Shift Register. Bit 6 is a read-only bit.

Bit 7: This bit is permanently set to logic 0.

INTERRUPT IDENTIFICATION REGISTER

The INS8250 has an on-chip interrupt capability that allows for complete flexibility in interfacing to all the popular microprocessors presently available. In order to provide minimum software overhead during data character transfers, the INS8250 prioritizes interrupts into four levels. The four levels of interrupt conditions are as follows: Receiver Line Status (priority 1); Received Data Ready (priority 2); Transmitter Holding Register Empty (priority 3); and MODEM Status (priority 4).

Information indicating that a prioritized interrupt is pending and the type of that interrupt are stored in the Interrupt Identification Register (refer to table 5). The Interrupt Identification Register (IIR), when addressed during chip-select time, freezes the highest priority interrupt pending and no other interrupts are acknowledged until the particular interrupt is serviced by the CPU. The contents of the IIR are indicated in table 2 and are described below.

Bit 0: This bit can be used in either a hardwired prioritized or polled environment to indicate whether an interrupt is pending. When bit 0 is a logic 0, an interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine. When bit 0 is a logic 1, no interrupt is pending and polling (if used) continues.

Bits 1 and 2: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in table 5.

Bits 3 through 7: These five bits of the IIR are always logic 0.

Table 5. Interrupt Control Functions

Interrupt Identification Register			Interrupt Set and Reset Functions			
Bit 2	Bit 1	Bit 0	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	0	1	—	None	None	—
1	1	0	Highest	Receiver Line Status	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the Line Status Register
1	0	0	Second	Received Data Available	Receiver Data Available	Reading the Receiver Buffer Register
0	1	0	Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if source of interrupt) or Writing into the Transmitter Holding Register
0	0	0	Fourth	MODEM Status	Clear to Send or Data Set Ready or Ring Indicator or Received Line Signal Detect	Reading the MODEM Status Register

INTERRUPT ENABLE REGISTER

This 8-bit register enables the four types of interrupts of the INS8250 to separately active the chip Interrupt (INTRPT) output signal. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of the Interrupt Enable Register. Similarly, by setting the appropriate bits of this register to a logic 1, selected interrupts can be enabled. Disabling the interrupt system inhibits the Interrupt Identification Register and the active (high) INTRPT output from the chip. All other system functions operate in their normal manner, including the setting of the Line Status and MODEM Status Registers. The contents of the Interrupt Enable Register are indicated in table 2 and are described below.

Bit 0: This bit enables the Received Data Available Interrupt when set to logic 1.

Bit 1: This bit enables the Transmitter Holding Register Empty Interrupt when set to logic 1.

Bit 2: This bit enables the Receiver Line Status Interrupt when set to logic 1.

Bit 3: This bit enables the MODEM Status Interrupt when set to logic 1.

Bits 4 through 7: These four bits are always logic 0.

MODEM CONTROL REGISTER

This 8-bit register controls the interface with the MODEM or data set (or a peripheral device emulating a MODEM). The contents of the MODEM Control Register are indicated in table 2 and are described below.

Bit 0: This bit controls the Data Terminal Ready ($\overline{DTR}$) output. When bit 0 is set to a logic 1, the $\overline{DTR}$ output is forced to a logic 0. When bit 0 is reset to a logic 0, the $\overline{DTR}$ output is forced to a logic 1.

NOTE

The $\overline{DTR}$ output of the INS8250 may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding MODEM or data set.

Bit 1: This bit controls the Request to Send ($\overline{RTS}$) output. Bit 1 affects the $\overline{RTS}$ output in a manner identical to that described above for bit 0.

Bit 2: This bit controls the Output 1 ($\overline{OUT\ 1}$) signal, which is an auxiliary user-designated output. Bit 2 affects the $\overline{OUT\ 1}$ output in a manner identical to that described above for bit 0.

Bit 3: This bit controls the Output 2 ($\overline{OUT\ 2}$) signal, which is an auxiliary user-designated output. Bit 3 affects the $\overline{OUT\ 2}$ output in a manner identical to that described above for bit 0.

Bit 4: This bit provides a loopback feature for diagnostic testing of the INS8250. When bit 4 is set to logic 1, the following occur: the transmitter Serial Output (SOUT) is set to the Marking (logic 1) state; the receiver Serial Input (SIN) is disconnected; the output of the Transmitter Shift Register is "looped back" into the Receiver Shift Register input; the four MODEM Control inputs ($\overline{CTS}$, $\overline{DSR}$, $\overline{RLSD}$, and $\overline{RI}$) are disconnected; and the four MODEM Control outputs ($\overline{DTR}$, $\overline{RTS}$, $\overline{OUT\ 1}$, and $\overline{OUT\ 2}$) are internally connected to the four MODEM Control inputs. In the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit- and receive-data paths of the INS8250.

In the diagnostic mode, the receiver and transmitter interrupts are fully operational. The MODEM Control Interrupts are also operational but the interrupts' sources are now the lower four bits of the MODEM Control Register instead of the four MODEM Control inputs. The interrupts are still controlled by the Interrupt Enable Register.

The INS8250 interrupt system can be tested by writing into the lower six bits of the Line Status Register and the lower four bits of the MODEM Status Register. Setting any of these bits to a logic 1 generates the appropriate interrupt (if enabled). The resetting of these interrupts is the same as in normal INS8250 operation. To return to normal operation, the registers must be reprogrammed for normal operation and then bit 4 of the MODEM Control Register must be reset to logic 0.

Bits 5 through 7: These bits are permanently set to logic 0.

MODEM STATUS REGISTER

This 8-bit register provides the current state of the control lines from the MODEM (or peripheral device) to the CPU. In addition to this current-state information, four bits of the MODEM Status Register provide change information. These bits are set to a logic 1 whenever a control input from the MODEM changes state. They are reset to logic 0 whenever the CPU reads the MODEM Status Register.

The contents of the MODEM Status Register are indicated in table 2 and are described below.

Bit 0: This bit is the Delta Clear to Send (DCTS) indicator. Bit 0 indicates that the $\overline{CTS}$ input to the chip has changed state since the last time it was read by the CPU.

Bit 1: This bit is the Delta Data Set Ready (DDSR) indicator. Bit 1 indicates that the $\overline{DSR}$ input to the chip has changed state since the last time it was read by the CPU.

Bit 2: This bit is the Trailing Edge of Ring Indicator (TERI) detector. Bit 2 indicates that the $\overline{RI}$ input to the chip has changed from an On (logic 1) to an Off (logic 0) condition.

Bit 3: This bit is the Delta Received Line Signal Detector (DRLSD) indicator. Bit 3 indicates that the $\overline{RLSD}$ input to the chip has changed state.

NOTE

Whenever bit 0, 1, 2, or 3 is set to logic 1, a MODEM Status interrupt is generated.

Bit 4: This bit is the complement of the Clear to Send ($\overline{CTS}$) input. If bit 4 (loop) of the MCR is set to a 1, this bit is equivalent to RTS in the MCR.

Bit 5: This bit is the complement of the Data Set Ready ($\overline{DSR}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to DTR in the MCR.

Bit 6: This bit is the complement of the Ring Indicator ($\overline{RI}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 1 in the MCR.

Bit 7: This bit is the complement of the Received Line Signal Detect ($\overline{RLSD}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 2 of the MCR.

Typical Applications

Figures 1 and 2 show how to use the INS8250 chip in an INS8080A system and in a microcomputer system with a high-capacity data bus.

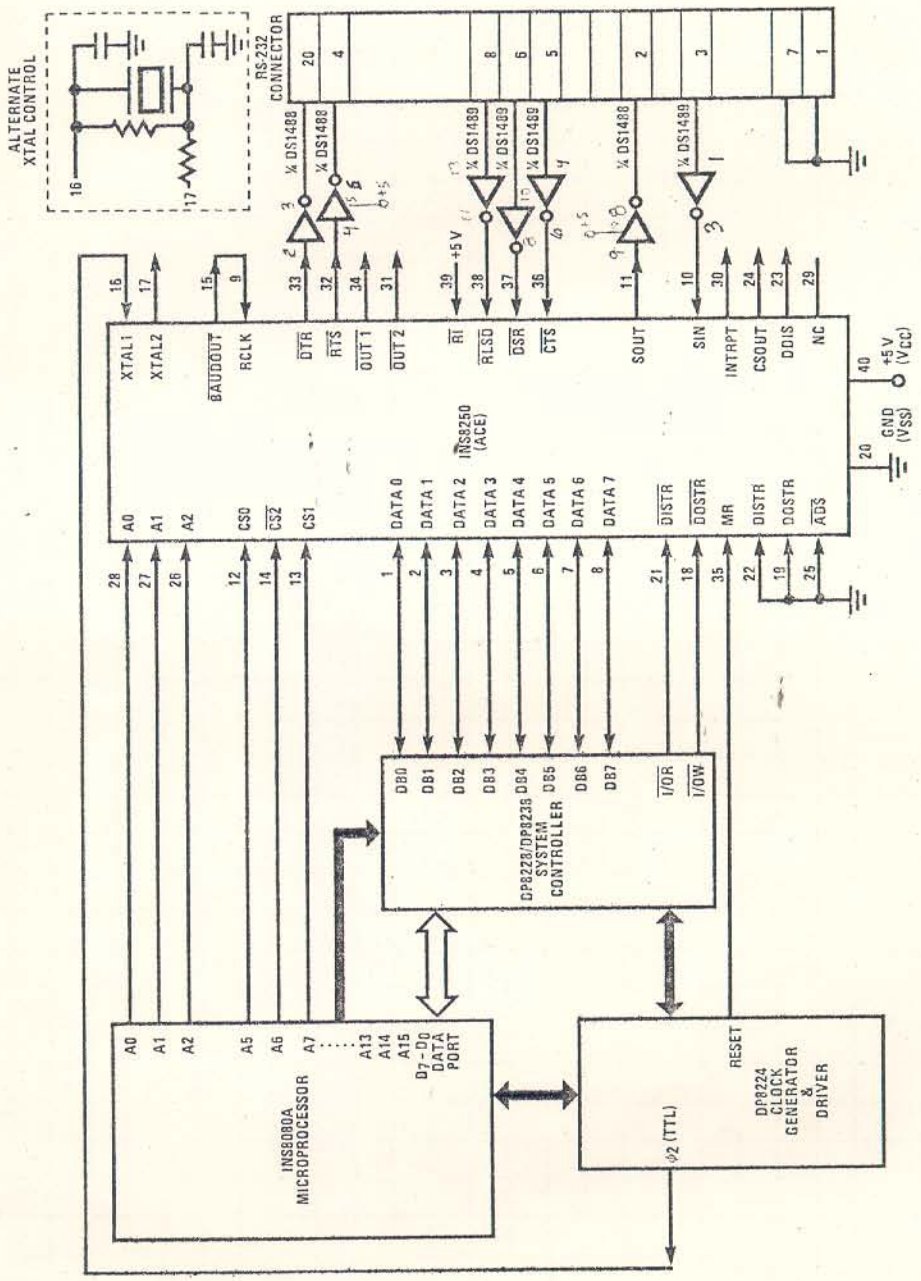


Figure 1. Typical INS8080A/INS8250 RS-232 Terminal Interface

Typical Applications (cont'd)

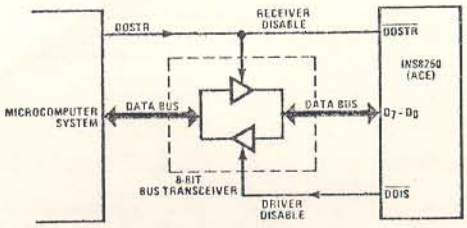
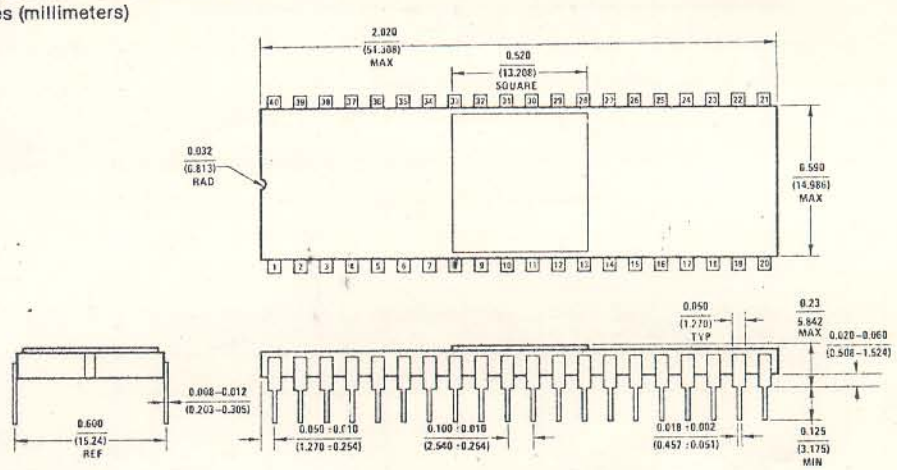


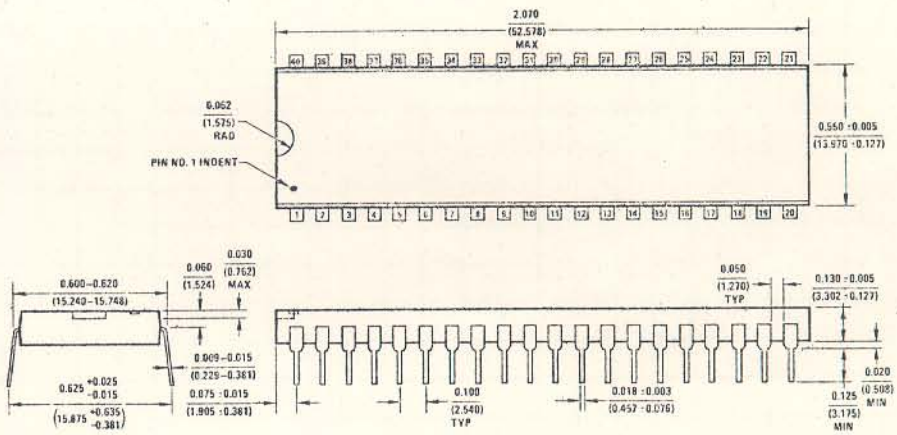
Figure 2. Typical Interface for a High-Capacity Data Bus

Physical Dimensions

inches (millimeters)



Ceramic Dual-In-Line Package (D)
Order Number INS8250D
NS Package Number D40C



Plastic Dual-In-Line Package (N)
Order Number INS8250N
NS Package Number D40A

National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051
Tel.: (408) 737-5000
TWX: (510) 339-9240

National Semiconductor GmbH
8000 München 21
Eisenheimerstrasse 61/2
West Germany
Tel.: 089/9 15027
Telex: 05-22772

NS International Inc., Japan
Miyake Building
1-9 Yotsuya, Shinjuku-ku 160
Tokyo, Japan
Tel.: (03) 355-3711
TWX: 232-2015 NSCJ-J

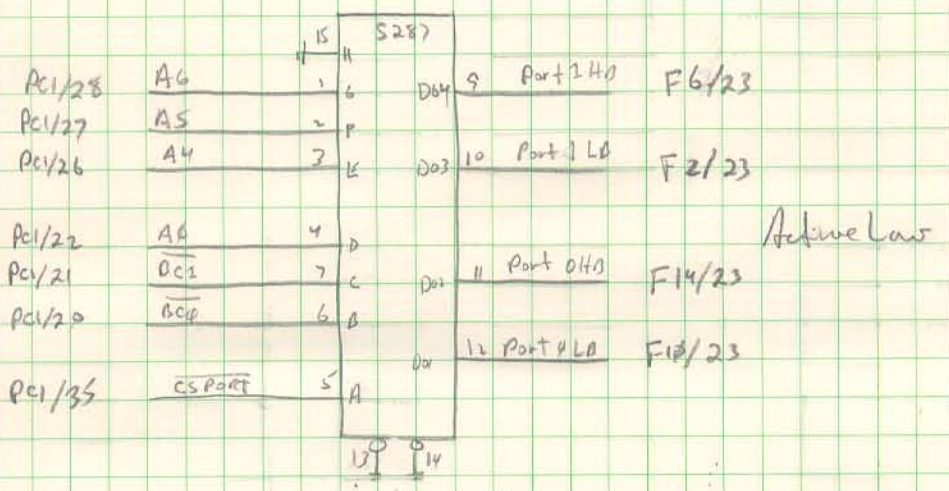
National Semiconductor (Hong Kong) Ltd.
8th Floor,
Cheung Kong Electronic Bldg.
4 Hing Yip Street
Kwun Tong
Kowloon, Hong Kong
Tel.: 3-411241-8
Telex: 73866 NSEHK HX
Cable: NATSEMI

NS Electronics Do Brasil
Avda Brigadeiro Faria Lima 844
11 Andar Conjunto 1104
Jardim Paulistano
Sao Paulo, Brasil
Telex
1121908 CABINE SAO PAULO

NS Electronics Pty. Ltd.
Cnr. Stud Rd. & Mtn. Highway
Bayswater, Victoria 3153
Australia
Tel.: 03-729-6333
Telex: 32096

Address Decoder

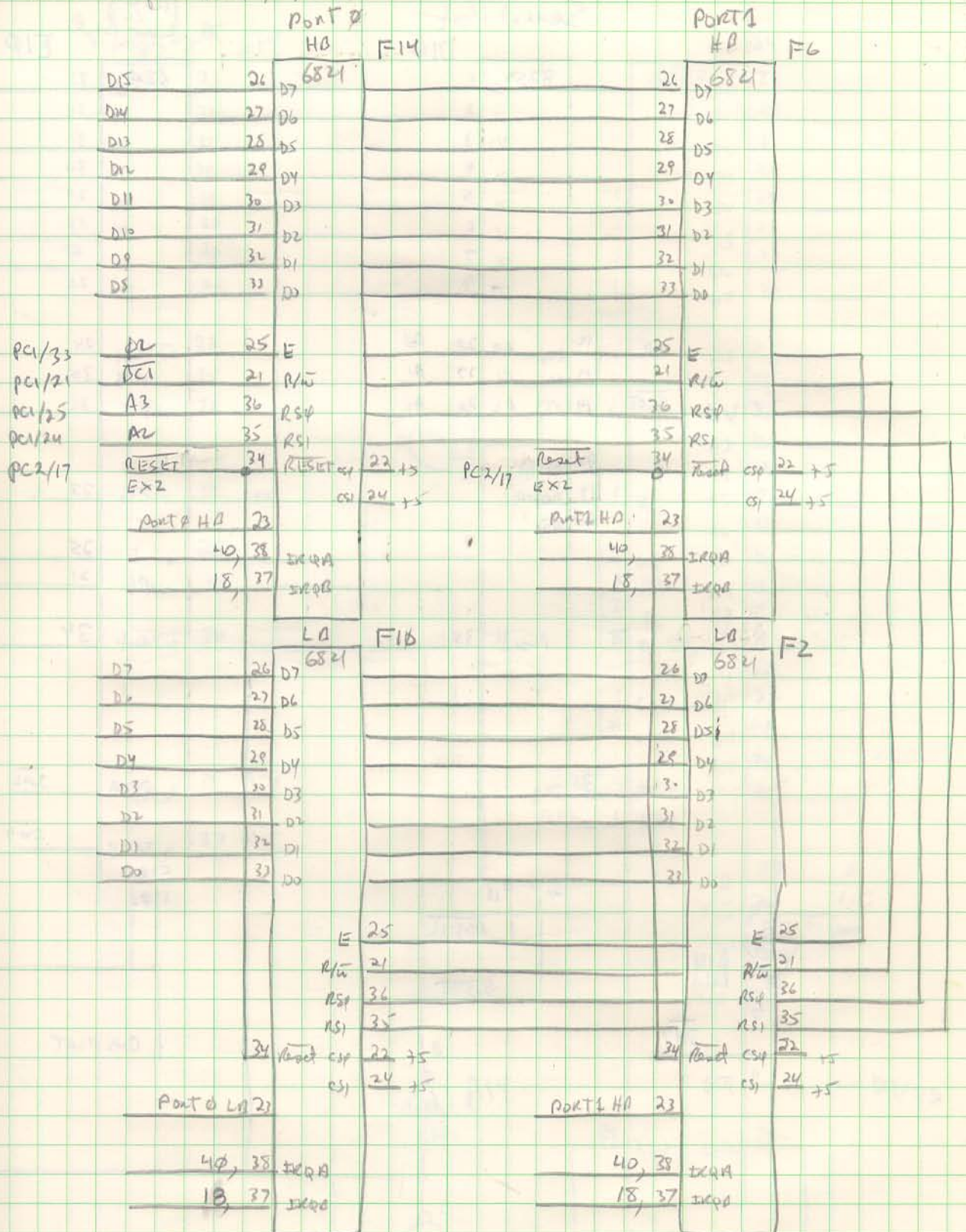
E8



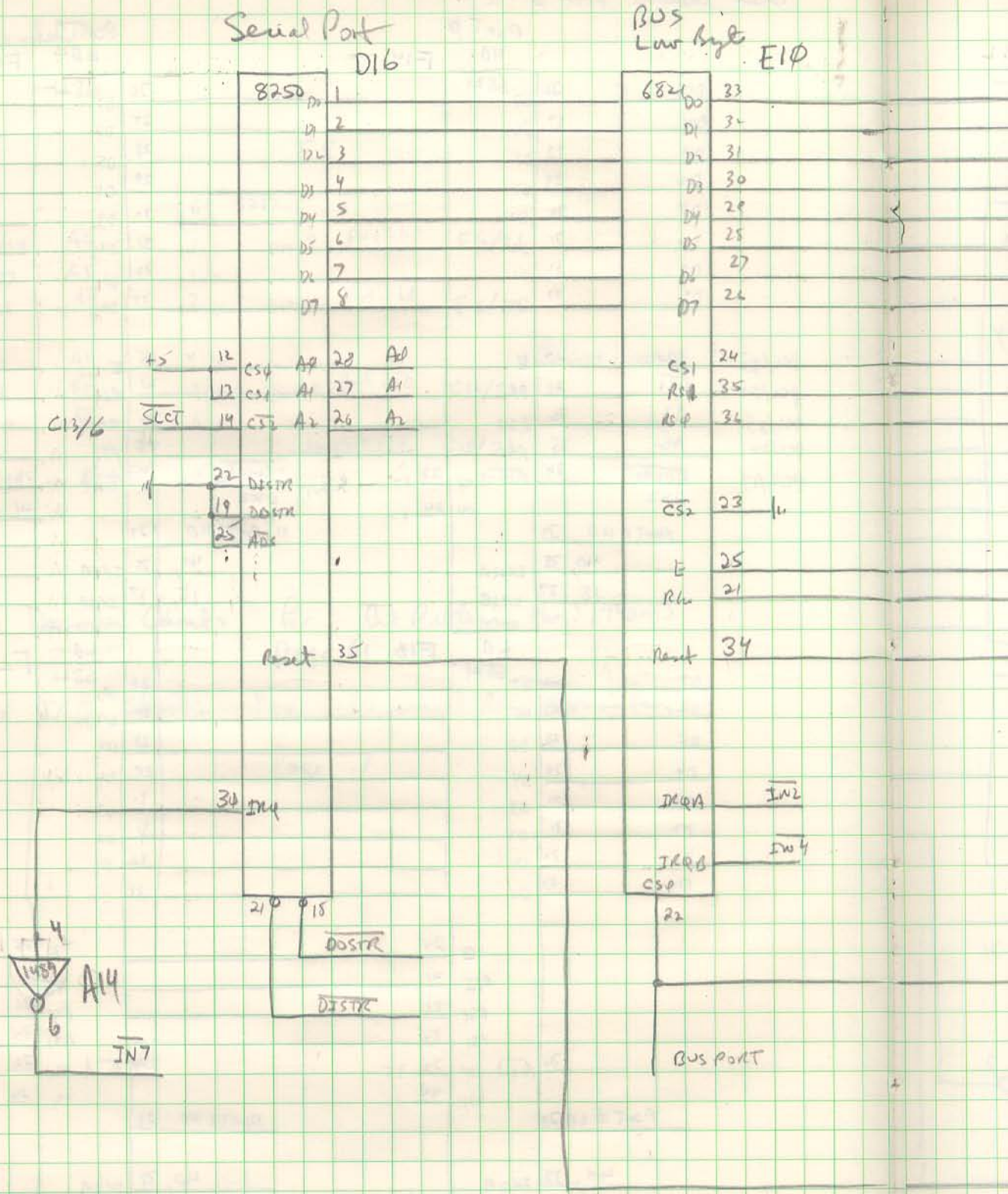
See Decoder Charts For Bit Patterns For PROMs
Page 61

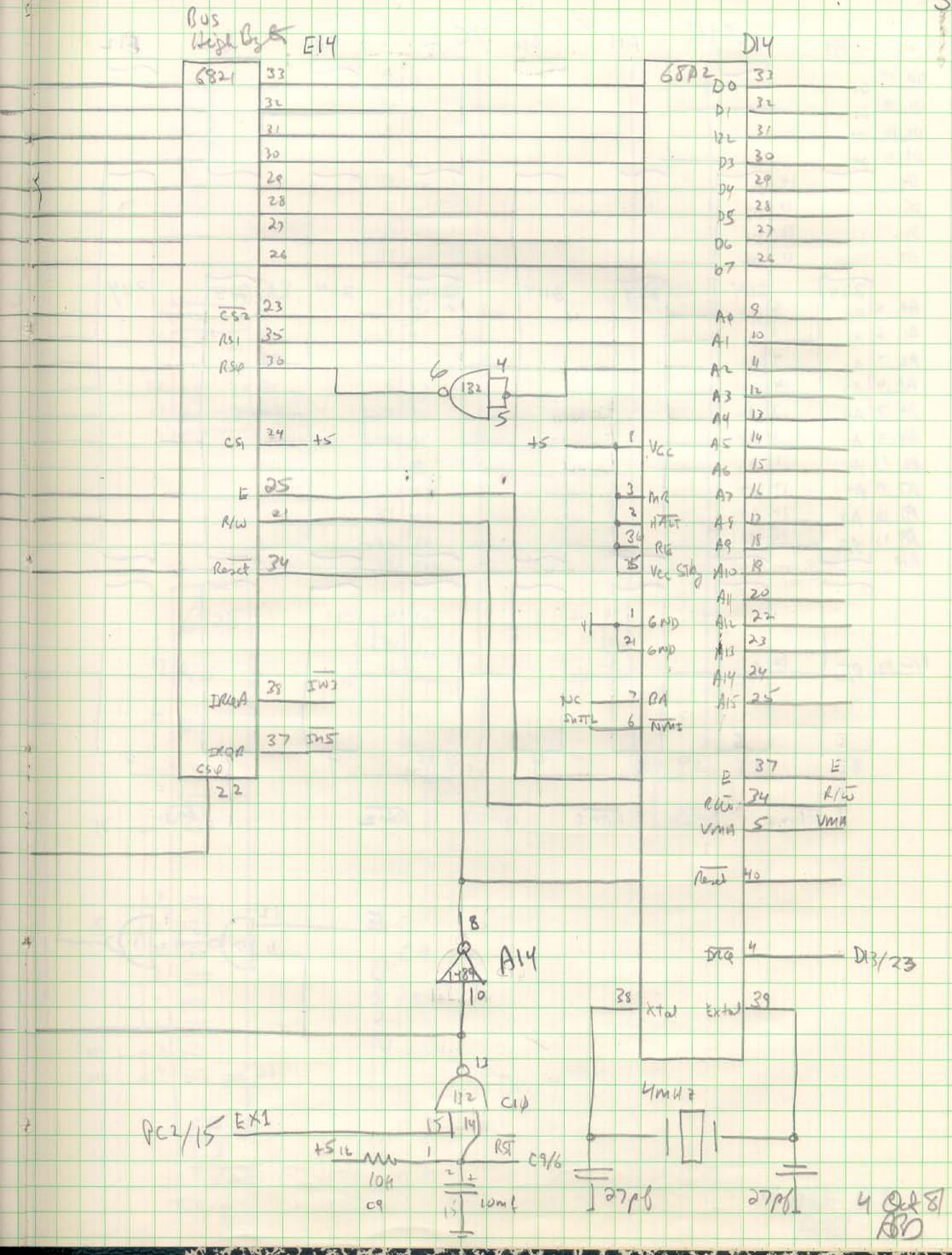
See IRQ Charts For Interrupt Pin Connections
Page 7

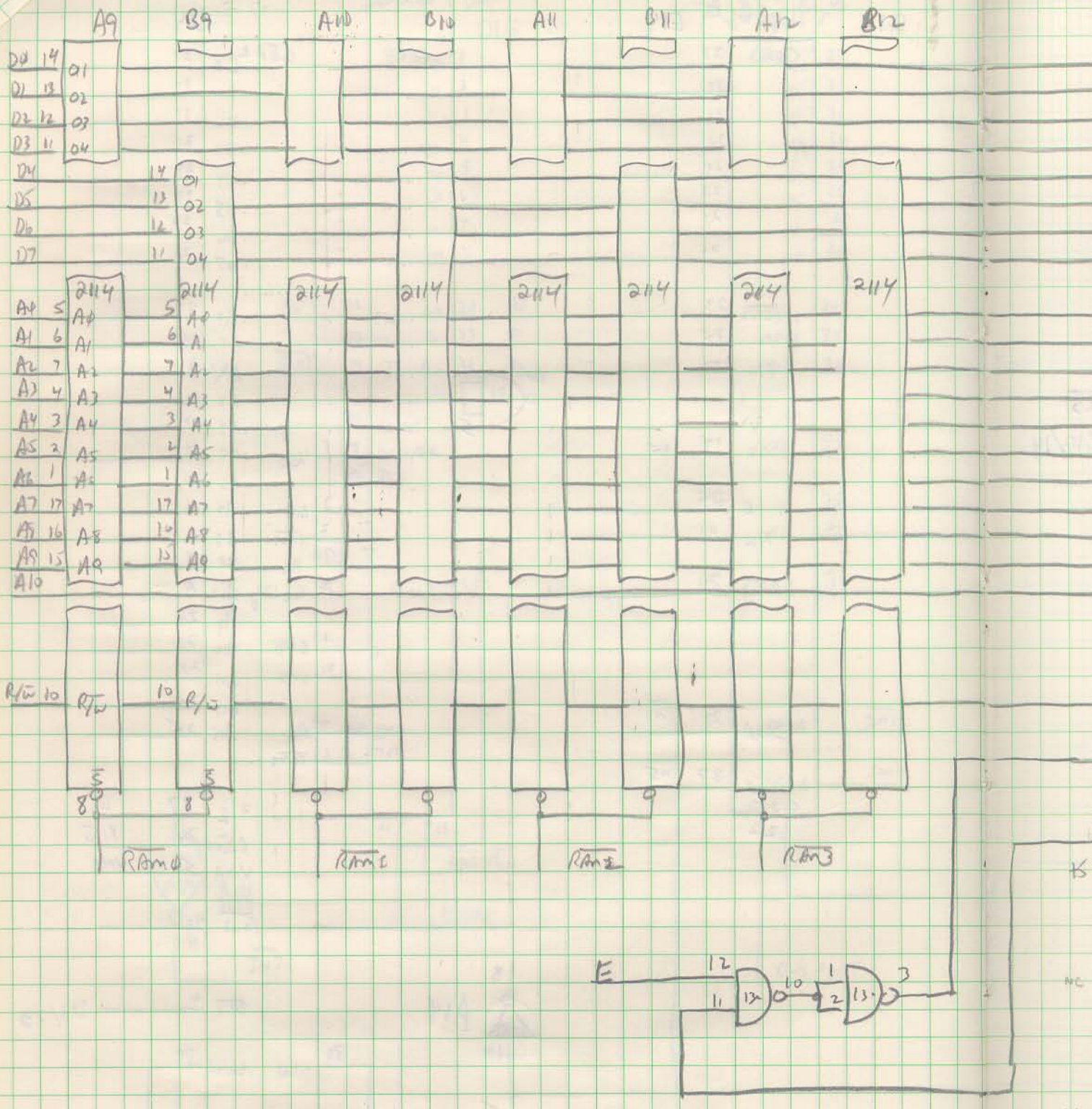
Bus Interface Port 0 & 1

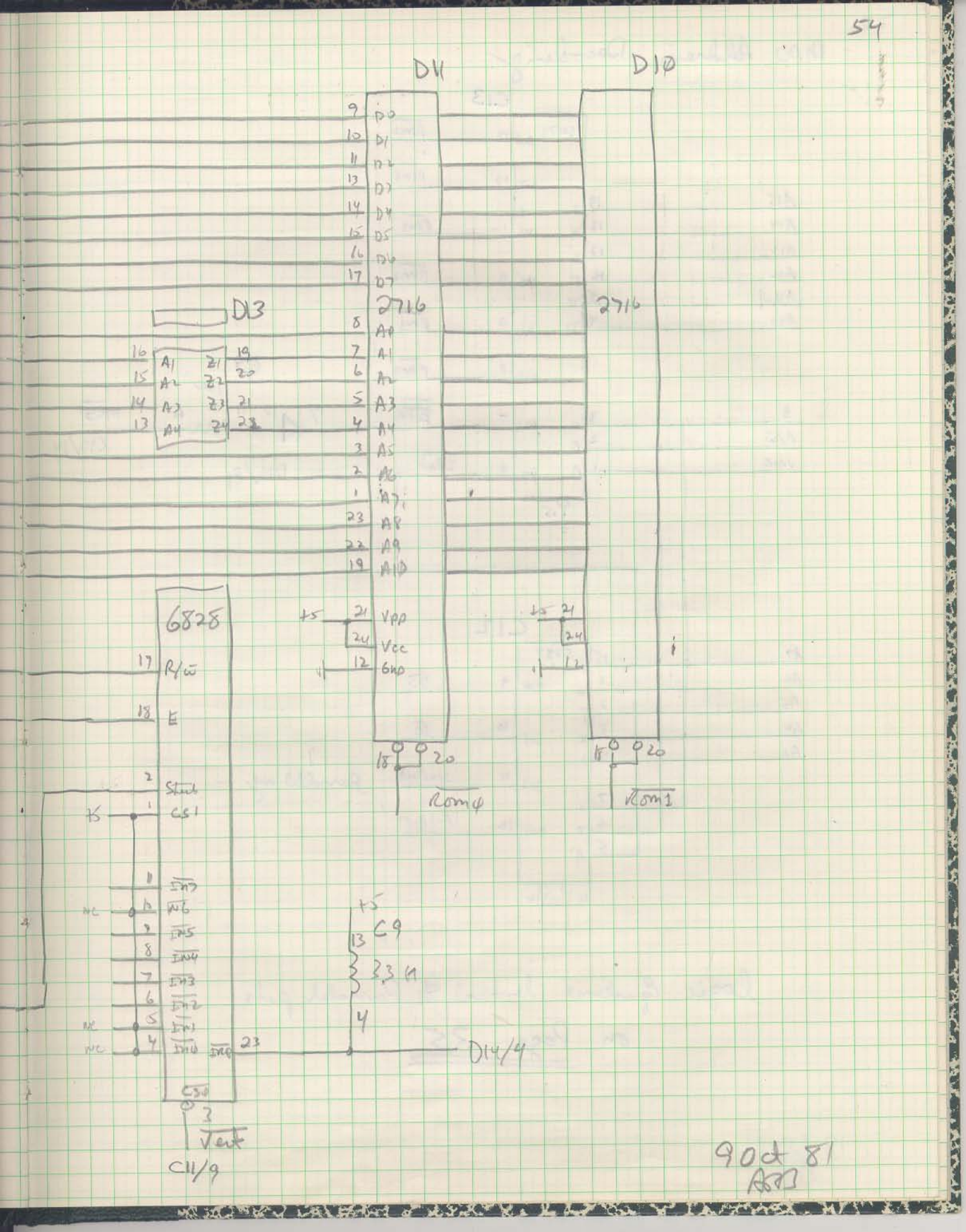


28 Sept 81
ARD


Serial Port
D16
BUS
Low Byte
E10
8250
6824
SLCT
C13/6
Reset
IN4
IN2
IN4
IN7
A14
DOSTR
DISTR
BUS PORT

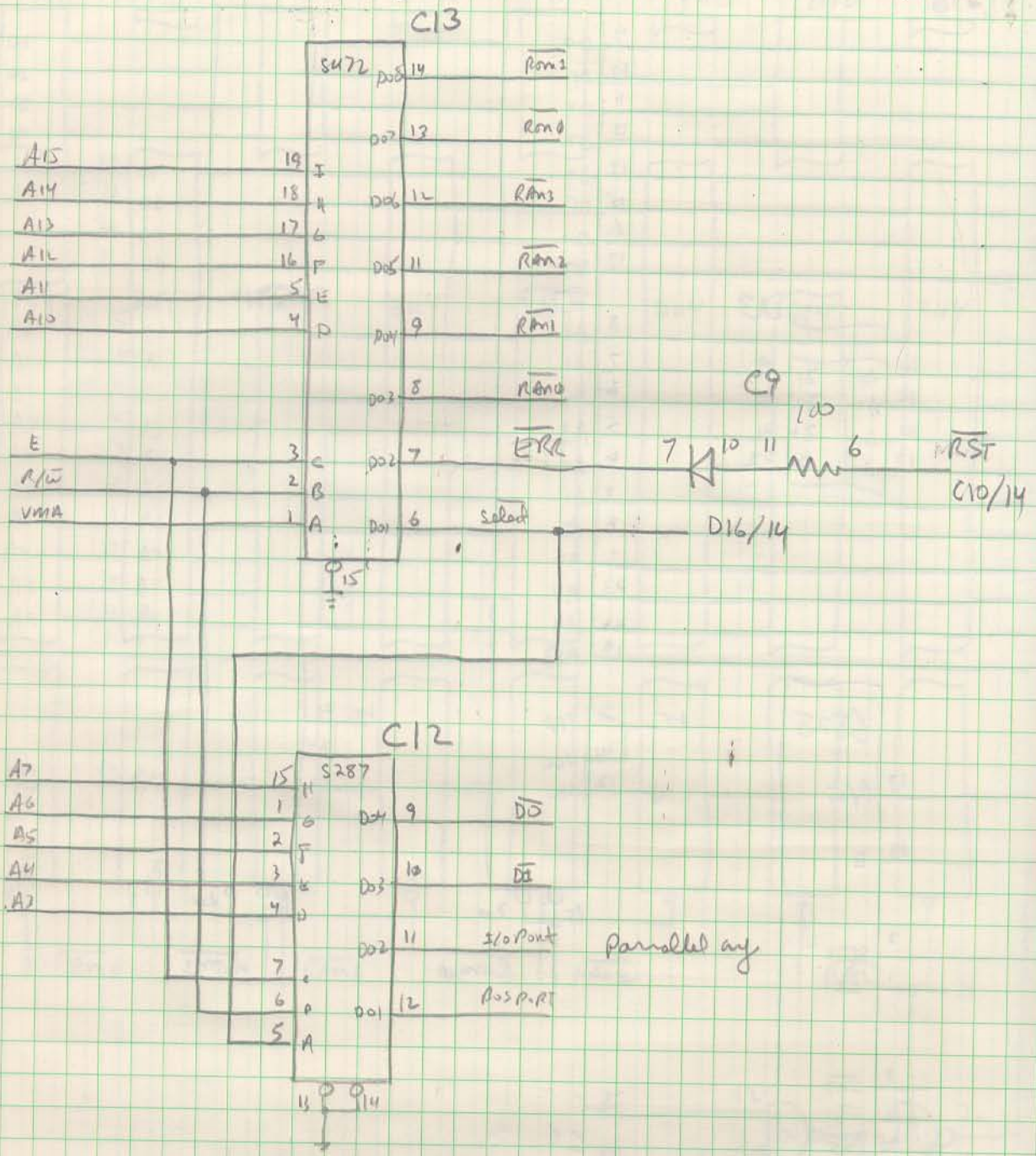

Bus
High Byte
E14
6821
D14
68B2
D0
D1
D2
D3
D4
D5
D6
D7
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
CS2
RS1
RS0
CS1
+5
E
R/W
Reset
IRQA
IRQB
CS0
IW3
IN5
132
+5
Vcc
MR
HALT
RE
Vcc Stby
GND
GND
NC
BA
5+TTL
NMI
E
R/W
VMA
Reset
IRQ
D13/23
A14
1489
Xtal
Extal
4MHz
27pf
27pf
C10
PC2/15
EX1
RST
C9/6
+5
10k
C9
10mf
4 Oct 81
ABD


A9
B9
A10
B10
A11
B11
A12
B12
D0
D1
D2
D3
D4
D5
D6
D7
2114
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
R/W
CS
RAM0
RAM1
RAM2
RAM3
E
12
11
13
10
1
2
13
3


D11
D10
2716
2716
D13
6828
Rom0
Rom1
C9
3.3 M
D14/4
C11/9
Vext
+5
R/W
E
CS1
CS0
VPP
Vcc
GND
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
D0
D1
D2
D3
D4
D5
D6
D7
IN7
IN6
IN5
IN4
IN3
IN2
IN1
IN0
NC

90 Oct 81
ASD

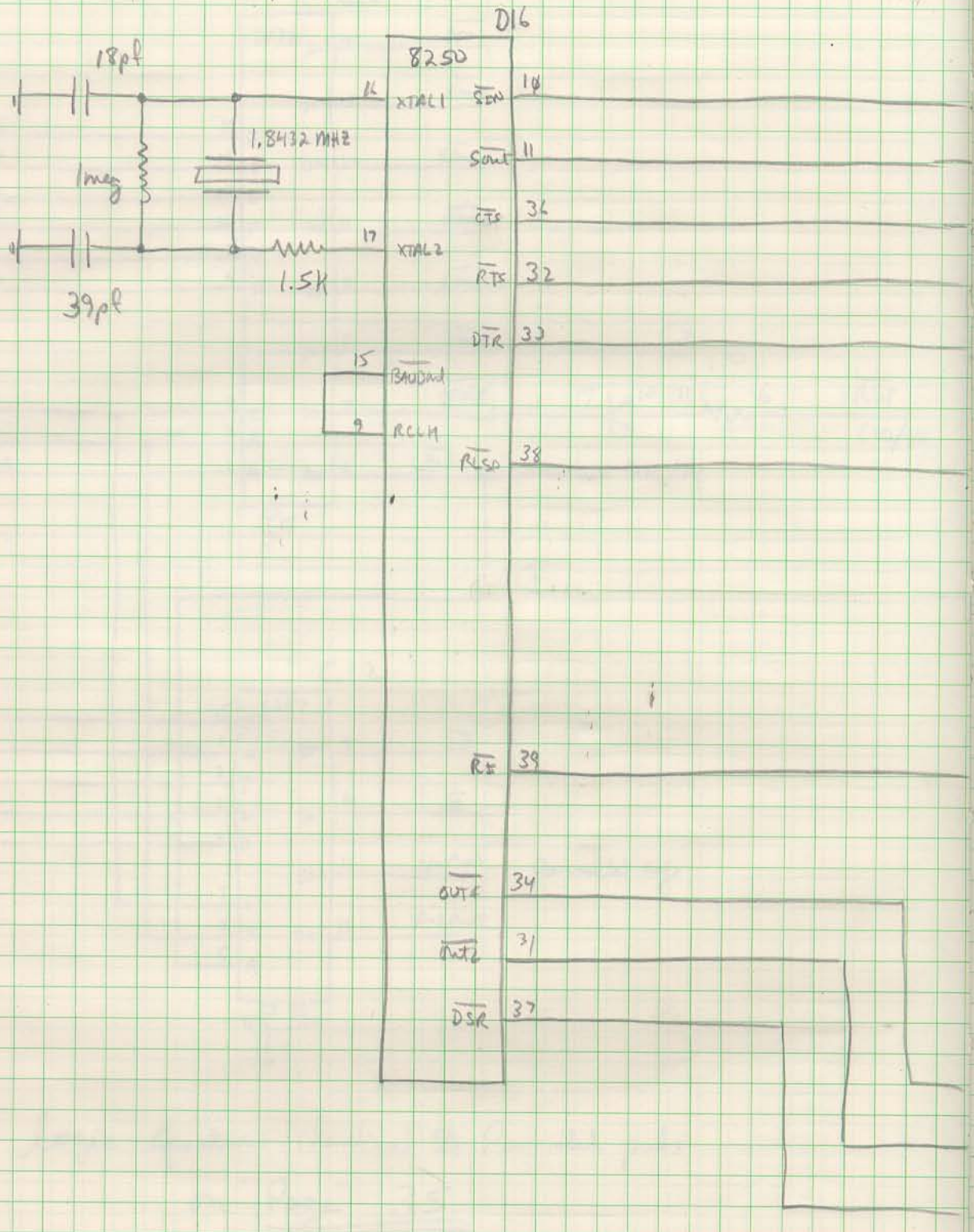
MPU Address Decoding



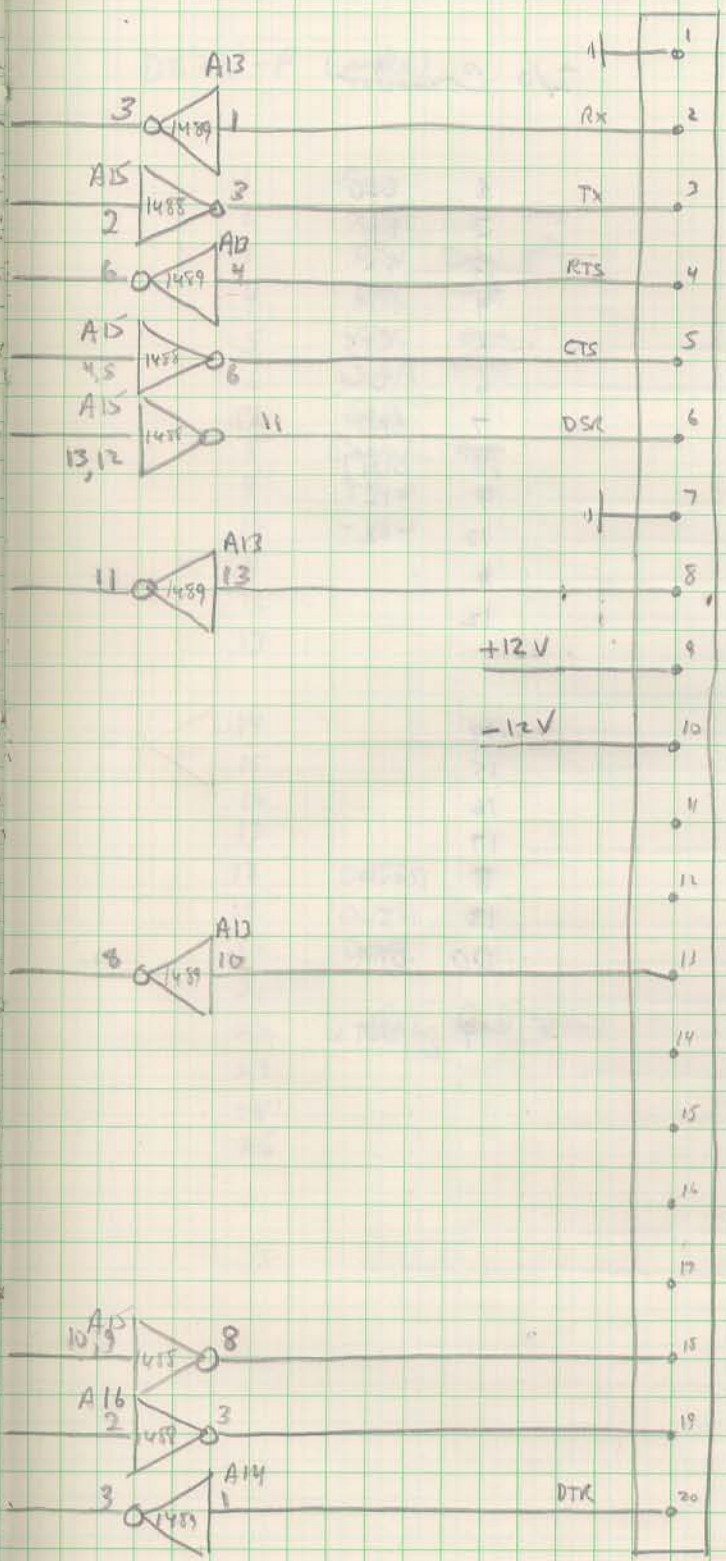
Logic Equations identical to Parallel outs

on Page 35

Serial Port Wiring



CN2 (Serial Board Connector)



11 Oct 81
ABB

RS-232 Connector Connections - (AS MODEM - to terminal)

DB25-S (female)

1	GND	
2	Rec Data	← IN
3	XMIT Data	→
4	RTS	←
5	CTS	→
6	DSR	→
7	GND	
8	Carrier	→
9	+12 V.	→
10	-12 V.	→
11		
12		
13		
14		
15		
16		
17		
18	RSLD	←
19	RI	←
20	DTR	←
21		
22	Ring Ind	→
23		
24		
25		

I/O Connector

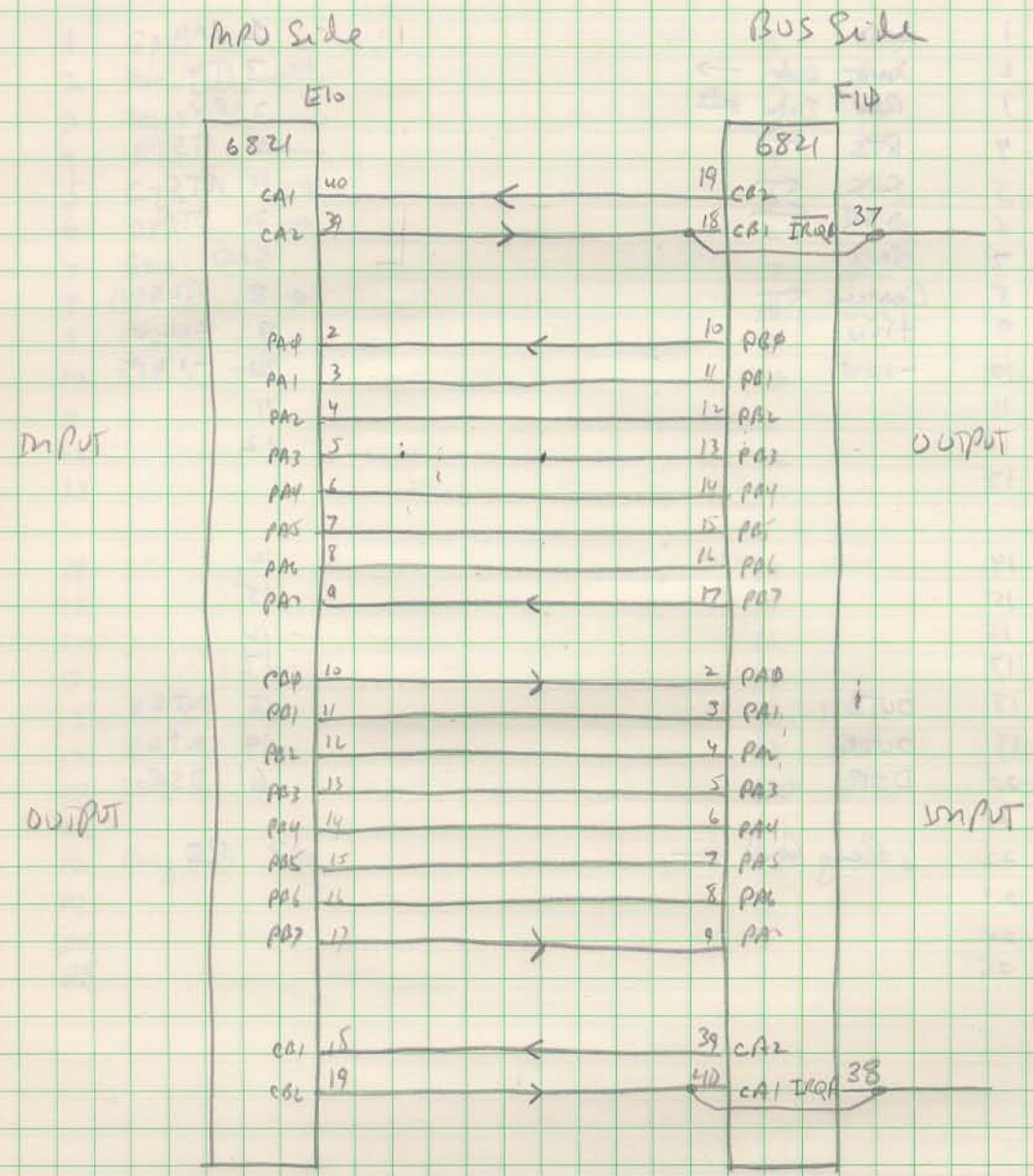
1	GND
2	Rx
3	TX
4	RTS
5	CTS
6	DSR
7	GND
18	OUT1
9	+12
10	-12
11	
12	
14	
15	
16	
17	
8	RSLD
13	RI
20	DTR
19	OUT2

RS-232 Connector Connections — (AS Terminal - to Modem)

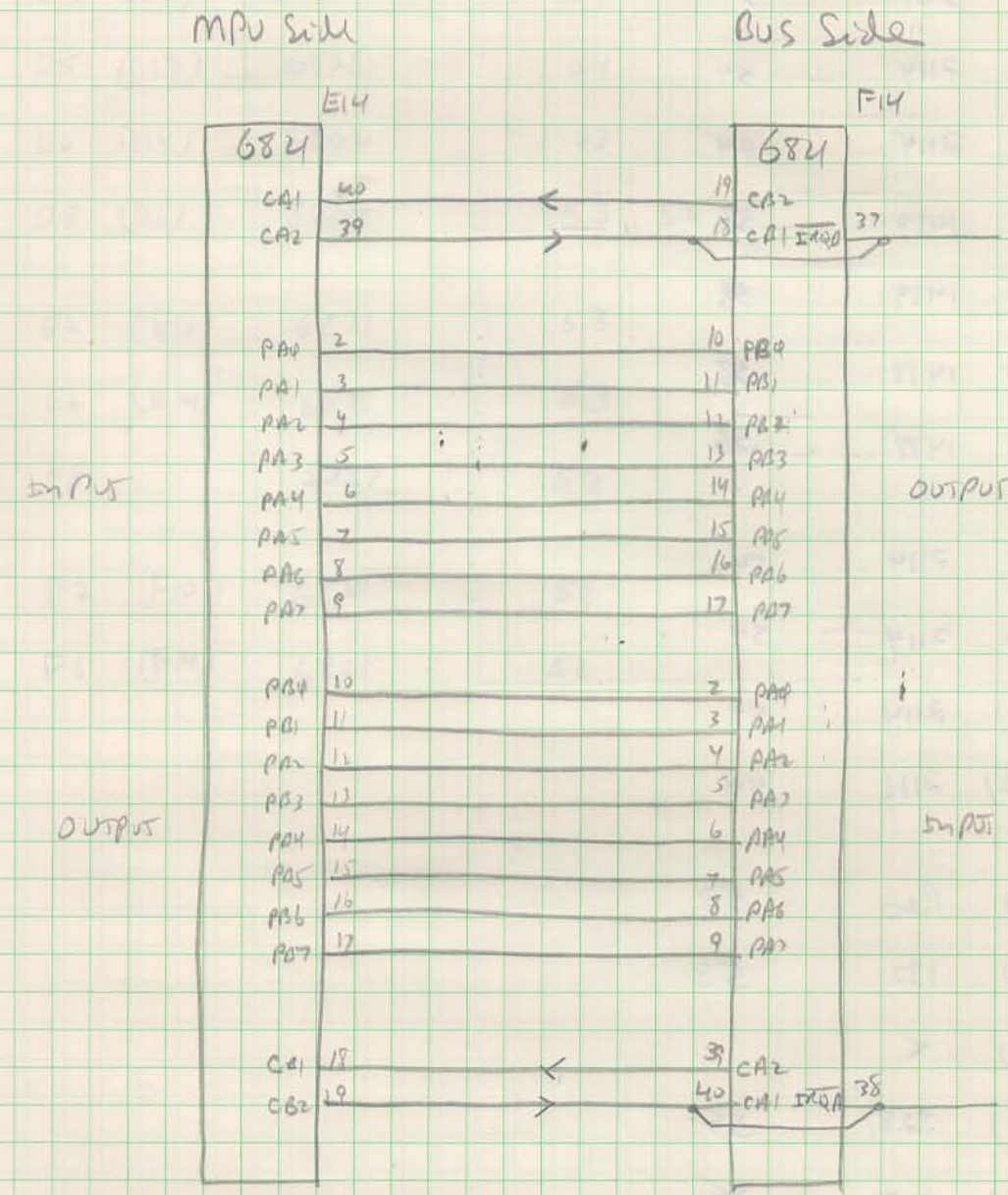
DB25-P (male)			I/O Connector
1	GND		1 GND
2	XMIT Data	→	3 TX
3	RCV Data	←	2 RX
4	RTS	→	5 CTS
5	CTS	←	4 RTS
6	DSR	←	20 DTR
7	GND		GND
8	Carrier	←	8 RLSD
9	+12 V		9 +12 V
10	-12 V		10 -12 V.
11			11
12			12
13			
14			14
15			15
16			16
17			17
18	OUT1		18 Out1
19	OUT2		19 Out2
20	DTR		6 DSR
21			
22	Ring Ind	←	13 RI
23			
24			
25			

12 Oct 81
AGD

Low Byte Bus port wiring



High Byte Bus Port Wiring



13 Oct 81
AAD

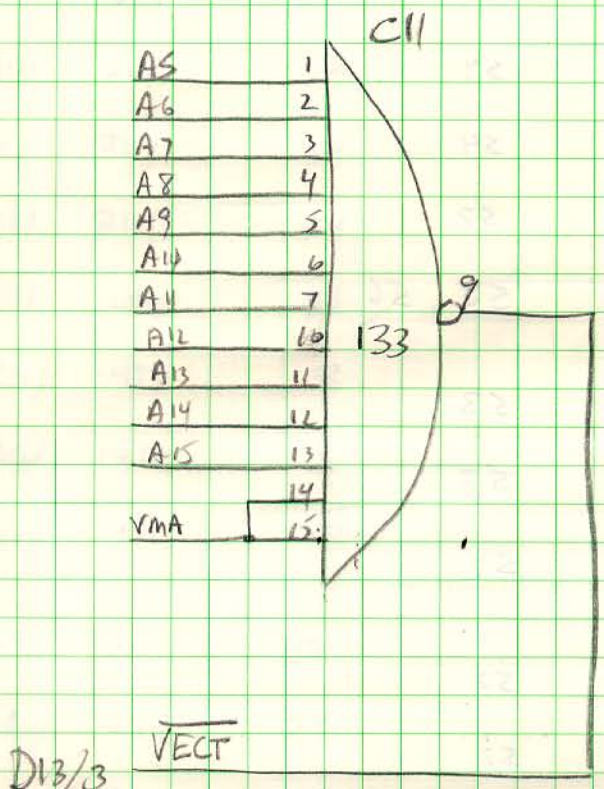
Serial Port IC locations

A1 (A9)	2114	54
A2 (A10)	2114	54
A3 (A11)	2114	54
A4 (A12)	2114	54
A5 (A13)	1489	56
A6 (A14)	1489	56
A7 (A15)	1488	56
A8 (A16)	1488	56
B1 (B9)	2114	54
B2 (B10)	2114	54
B3 (B11)	2114	54
B4 (B12)	2116	54
C1 (C9)	Res	53
C2 (C10)	132	53
C3 (C11)	X	
C4 (C12)	S287	55
C5 (C13)	S472	55

D2	(D10)	2716	54
D3	(D11)	2716	54
D5	(D13)	6828	54
D6	(D14)	6802	53
D8	(D16)	8250	53, 56
E2	(E10)	6821	53
E6	(E14)	6821	53
E8		5287	52
F2	(F10)	6821	52
F6	(F14)	6821	52

changes

6828 Vector Decoder



D13/3

PRom Decoder Charts

Designations	A	B	C	D	E	F	G
	$\overline{CS}$	$\overline{BC\emptyset}$	$\overline{BC1}$	A$\emptyset$	A4	A5	A6

States	$\overline{BC1}$	$\overline{BC\emptyset}$	
	1	1	Read
	1	0	Read
	0	1	Write word
	0	0	Write Byte

PORT LB = C (Read Port)
+ $B \cdot \overline{C}$ (write word port)
+ $\overline{B} \cdot \overline{C} \cdot \overline{D}$ (write Low Byte Port)

PORT HB = C (Read Port)
+ $B \cdot \overline{C}$ (write word Port)
+ $\overline{B} \cdot \overline{C} \cdot D$ (write High Byte Port)

Port Select$_\emptyset$ = $\overline{A} \cdot \overline{E} \cdot \overline{F} \cdot \overline{G}$

$_1$ = $\overline{A} \cdot E \cdot \overline{F} \cdot \overline{G}$

$_2$ = $\overline{A} \cdot \overline{E} \cdot F \cdot \overline{G}$

$_3$ = $\overline{A} \cdot E \cdot F \cdot \overline{G}$

$_4$ = $\overline{A} \cdot \overline{E} \cdot \overline{F} \cdot G$

$_5$ = $\overline{A} \cdot E \cdot \overline{F} \cdot G$

$_6$ = $\overline{A} \cdot \overline{E} \cdot F \cdot G$

$_7$ = $\overline{A} \cdot E \cdot F \cdot G$

PORT ∅ LB = $\overline{(C \cdot \bar{A} \cdot \bar{E} \cdot \bar{F} \cdot \bar{G}) + (B \cdot \bar{C} \cdot \bar{A} \cdot \bar{E} \cdot \bar{F} \cdot \bar{G}) + (\bar{B} \cdot \bar{C} \cdot \bar{D} \cdot \bar{A} \cdot \bar{E} \cdot \bar{F} \cdot \bar{G})}$

⇕ .D.

PORT ∅ HB =

PORT 1 LB = $\overline{(C \cdot \bar{A} \cdot E \cdot \bar{F} \cdot \bar{G}) + (B \cdot \bar{C} \cdot \bar{A} \cdot E \cdot \bar{F} \cdot \bar{G}) + (\bar{B} \cdot \bar{C} \cdot \bar{D} \cdot \bar{A} \cdot E \cdot \bar{F} \cdot \bar{G})}$

⇕ .D.

PORT 1 HB =

other ports repeated

21 Nov 81
ARD

```
R MACRO
SERIAL,SERIAL=M6800,SERIAL,LOADER,6821A,8250A,END
```

```
.IIF    NDF,.M68        .NLIST  ;CROSS-ASSEMBLER NOT LISTED
        ;DURING CHECKOUT DEFINE .M68 TO ENABLE LISTING
.IIF    DF,.M68         .LIST   SRC,SEQ,COM,MD,MC
        ;
        ;
        .TITLE  M6800 CROSS-ASSEMBLER
        ;
.IIF    DF,.M68 .SBTTL  CROSS-ASSEMBLER INTRODUCTION
        ;
.ENABLE ABS
        ;
        ;*************************************************************
        ;*                                                           *
        ;*      MACRO PACKAGE FOR THE MOTOROLA 6800 MICROPROCESSOR   *
        ;*              TO RUN UNDER MACRO 11.                       *
        ;*                                                           *
        ;*      BY ALAN R. BALDWIN                                   *
        ;*                                                           *
        ;*              V03 - OCTOBER 1980                           *
        ;*                                                           *
        ;*************************************************************
        ;
        ;
        ;THE FOLLOWING DIFFERENCES EXIST BETWEEN THIS CROSS-ASSEMBLER
        ;AND MOTOROLA'S M6800 ASSEMBLER
        ;       LABELS MUST TERMINATE WITH A :
        ;       COMMENTS START WITH A ;
        ;       IMMEDIATE MODE IS DENOTED BY A SEPERATE ARGUMENT - #
        ;
        ;
        ;DEFINITION OF ASSEMBLER DIRECTIVES WITH DIFFERENCES
        ;       END - USE       .END    END OF PROGRAM
        ;       EQU - USE  =            EQUATE SYMBOL
        ;       FCB                     FORM SINGLE-BYTE CONSTANT
        ;                               NO MORE THAN 10 ARGUMENTS
        ;       FCC   <ASCII CHRS>      FORM CONSTANT CHARACTERS
        ;       FDB                     FORM DOUBLE-BYTE CONSTANT
        ;                               NO MORE THAN 10 ARGUMENTS
        ;       MON - NOT IMPLEMENTED   RETURN TO MONITOR CONSOLE
        ;       NAM - USE       .SBTTL  PROGRAM NAME
        ;       OPT - NOT IMPLEMENTED   OPTION
        ;       ORG                     ORIGIN
        ;       PAGE - USE      .PAGE   ADVANCE LISTING TO TOP OF PAGE
        ;       RMB                     RESERVE MEMORY BYTES
        ;       SPC - NOT IMPLEMENTED   SPACE N LINES
        ;
        ;
        ;PROCESSOR CONDITION CODE REVIEW
        ;       0 - CARRY BIT (C)
        ;       1 - OVERFLOW BIT (V)
        ;       2 - ZERO BIT (Z)
        ;       3 - NEGATIVE BIT (N)
        ;       4 - INTERRUPT MASK BIT (I)
        ;       5 - HALF CARRY BIT (H)
        ;       6 - ALWAYS 1
        ;       7 - ALWAYS 1
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  SINGLE BYTE 'INHERENT' INSTRUCTIONS
        ;
.MACRO  AINST   H,I
 .MACRO H
 .NLIST SRC
 .BYTE  I
 .LIST  SRC
 .ENDM
.ENDM   AINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
AINST   NOP,    1       ;DO NOTHING
AINST   TAP,    6       ;A TO CC'S
AINST   TPA,    7       ;CC'S TO A
AINST   INX,    10      ;INCREMENT INDEX REGISTER
AINST   DEX,    11      ;DECREMENT INDEX REGISTER
AINST   CLV,    12      ;CLEAR V BIT
```

```
AINST   SEV,    13      ;SET C BIT
AINST   CLC,    14      ;CLEAR C BIT
AINST   SEC,    15      ;SET C BIT
AINST   CLI,    16      ;CLEAR I BIT
AINST   SEI,    17      ;SET I BIT
AINST   SBA,    20      ;ACCA=ACCA-ACCB
AINST   CBA,    21      ;COMPARE ACCA & ACCB
AINST   TAB,    26      ;ACCB=ACCA
AINST   TBA,    27      ;ACCA=ACCB
AINST   DAA,    31      ;DECIMAL ADJUST
AINST   ABA,    33      ;ACCA=ACCA+ACCB
AINST   TSX,    60      ;X=SP+1
AINST   INS,    61      ;SP=SP+1
AINST   PULA,   62      ;PULL A FROM STACK
AINST   PULB,   63      ;PULL B FROM STACK
AINST   DES,    64      ;SP=SP-1
AINST   TXS,    65      ;SP=X-1
AINST   PSHA,   66      ;PUSH A ONTO STACK
AINST   PSHB,   67      ;PUSH B ONTO STACK
AINST   RTS,    71      ;RETURN FROM SUBROUTINE
AINST   RTI,    73      ;RETURN FROM INTERRUPT
AINST   WAI,    76      ;WAIT FOR INTERRUPT
AINST   SWI,    77      ;SOFTWARE INTERRUPT
AINST   NEGA,   100     ;NEGATE A
AINST   COMA,   103     ;COMPLEMENT A
AINST   LSRA,   104     ;LOGICAL SHIFT RIGHT A
AINST   RORA,   106     ;ROTATE RIGHT A
AINST   ASRA,   107     ;ARITHMETIC SHIFT RIGHT A
AINST   ASLA,   110     ;ARITHMETIC SHIFT LEFT A
AINST   ROLA,   111     ;ROTATE LEFT A
AINST   DECA,   112     ;DECREMENT A
AINST   INCA,   114     ;INCREMENT A
AINST   TSTA,   115     ;TEST A
AINST   CLRA,   117     ;CLEAR A
AINST   NEGB,   120     ;NEGATE B
AINST   COMB,   123     ;COMPLEMENT B
AINST   LSRB,   124     ;LOGICAL SHIFT RIGHT B
AINST   RORB,   126     ;ROTATE RIGHT B
AINST   ASRB,   127     ;ARITHMETIC SHIFT RIGHT B
AINST   ASLB,   130     ;ARITHMETIC SHIFT LEFT B
AINST   ROLB,   131     ;ROTATE LEFT B
AINST   DECB,   132     ;DECREMENT B
AINST   INCB,   134     ;INCREMENT B
AINST   TSTB,   135     ;TEST B
AINST   CLRB,   137     ;CLEAR B
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  PUSH AND PULL OPCODES
        ;
.MACRO  PKRNL   I,J
 .IIF   IDN     <J>,A   ...A=0  ;PSH/PUL A
 .IIF   IDN     <J>,B   ...A=1  ;PSH/PUL B
 .BYTE  I+...A          ;INVALID ARGUMENT
.ENDM
        ;
.MACRO  PINST   H,I
 .MACRO H       J
 .NLIST SRC
 PKRNL  I,J
 .LIST  SRC
 .ENDM
.ENDM   PINST
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
PINST   PUL,    62      ;PUL BYTE FROM STACK S=S+1
PINST   PSH,    66      ;PUSH BYTE ONTO STACK S=S-1
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  RELATIVE BRANCH INSTRUCTIONS
        ;
.MACRO  BKRNL   I,J
 ...A=J-.-2
 .IIF   LT,...A+200     .ERROR  ;BRANCH OUT OF RANGE
 .IIF   GE,...A-200     .ERROR  ;BRANCH OUT OF RANGE
 .BYTE  I,...A
.ENDM
```

```
        ;
.MACRO  BINST   H,I
 .MACRO H       J
 .NLIST SRC
 BKRNL  I,J
 .LIST  SRC
 .ENDM
.ENDM   BINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPFERATION
BINST   BRA,    40      ;BRANCH ALWAYS
BINST   BHI,    42      ;BRANCH IF (C=0) AND (Z=0)
BINST   BLS,    43      ;BRANCH IF (C=1) OR (Z=1)
BINST   BCC,    44      ;BRANCH IF (C=0)
BINST   BCS,    45      ;BRANCH IF (C=1)
BINST   BNE,    46      ;BRANCH IF (Z=0)
BINST   BEQ,    47      ;BRANCH IF (Z=1)
BINST   BVC,    50      ;BRANCH IF (V=0)
BINST   BVS,    51      ;BRANCH IF (V=1)
BINST   BPL,    52      ;BRANCH IF (N=0)
BINST   BMI,    53      ;BRANCH IF (N=1)
BINST   BGE,    54      ;BRANCH IF (<N XOR V>=0)
BINST   BLT,    55      ;BRANCH IF (<N XOR V>=1)
BINST   BGT,    56      ;BRANCH IF (Z=0) AND (<N XOR V>=0)
BINST   BLE,    57      ;BRANCH IF (Z=1) OR (<N XOR V>=1)
BINST   BSR,    215     ;BRANCH TO SUBROUTINE
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  INSTRUCTIONS HAVING ONLY ACCX,INDEXED,AND EXTENDED MODES
        ;
        ;
.MACRO  CKRNL   I,J,K
 .IF    NB,<K>          ;TWO ARGUMENTS - THEN INDEXED
  .IIF  DIF     <K>,X   .ERROR  ;INDEX BAD
  .IIF  LT,J    .ERROR  ;NEGATIVE OFFSET
  .BYTE I+40,J          ;OFFSET OUT OF RANGE
  .MEXIT
 .ENDC
 .IF    NB,<J>          ;ONE ARGUMENT - A, B, OR EXTENDED MODE
  ...A=60
  .IIF  IDN     <J>,A   ,...A=0         ;ACCA MODE
  .IIF  IDN     <J>,B   ,...A=20        ;ACCB MODE
  .IIF  NE,...A-60      .BYTE   I+...A
  .IIF  EQ,...A-60      .BYTE   I+...A,J&177400/400,J&377
  .MEXIT
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  CINST   H,I
 .MACRO H       J,K
 .NLIST SRC
 CKRNL  I,J,K
 .LIST  SRC
 .ENDM
.ENDM   CINST
        ;
        ;

    ;MNEMONIC  OPCODE   ;OPERATION
        ;
CINST   NEG,    100     ;NEGATE
CINST   COM,    103     ;COMPLEMENT
CINST   LSR,    104     ;LOGICAL SHIFT RIGHT
CINST   ROR,    106     ;ROTATE RIGHT
CINST   ASR,    107     ;ARITHMETIC SHIFT RIGHT
CINST   ASL,    110     ;ARITHMETIC SHIFT LEFT
CINST   ROL,    111     ;ROTATE LEFT
CINST   DEC,    112     ;DECREMENT
CINST   INC,    114     ;INCREMENT
CINST   TST,    115     ;TEST
CINST   CLR,    117     ;CLEAR
        ;
        ;
```

```
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  JUMP AND JSR INSTRUCTIONS
        ;
.MACRO  DKRNL   I,J,K
 .IF    NB,<K>          ;TWO ARGUMENTS - INDEXED MODE
  .IIF  DIF     <K>,X   .ERROR  ;BAD INDEX
  .IIF  LT,J            .ERROR  ;NEGATIVE OFFSET
  .BYTE I+40,J          ;OFFSET TOO LARGE
  .MEXIT
 .ENDC
 .IF    NB,<J>          ;ONE ARGUMENT - EXTENDED MODE
  .BYTE I+60,J&177400/400,J&377
  .MEXIT
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  DINST   H,I
 .MACRO H       J,K
 .NLIST SRC
 DKRNL  I,J,K
 .LIST  SRC
 .ENDM
.ENDM   DINST
        ;
        ;
     ;MNEMONIC  OPCODE  ;OPERATION
DINST   JMP,    116     ;JUMP
DINST   JSR,    215     ;JUMP TO SUBROUTINE
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  ALL ACCX INSTRUCTIONS
        ;
.MACRO  EKRNL   I,J,K,L
 ...A=-1
 .IIF   IDN     <J>,A   ,...A=0         ;ACCA MODE
 .IIF   IDN     <J>,B   ,...A=100       ;ACCB MODE
  .IF   GE,...A                 ;ACCX MODES
   .IF  NB,<L>                  ;THREE ARGS - IMMEDIATE/INDEXED
    .IF IDN     <K>,#   ;CHECK IMMEDIATE
     .IIF       EQ      <I>-207 ,.ERROR ;STA #
     .BYTE      I+...A,L
     .MEXIT
    .ENDC
    .IF IDN     <L>,X   ;CHECK INDEXED
     .IIF       LT,K    .ERROR  ;NEGATIVE OFFSET
     .BYTE      I+...A+40,K     ;OFFSET TOO LARGE
     .MEXIT
    .ENDC
    .ERROR      ;BAD INSTRUCTION
    .MEXIT
   .ENDC
   .IF  NB,<K>                  ;TWO ARGS - DIRECT/EXTENDED
    .IIF        EQ,K&177400     .BYTE   I+...A+20,K
    .IIF        NE,K&177400     .BYTE   I+...A+60,K&177400/400,K&377
    .MEXIT
   .ENDC
  .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  EINST   H,I
 .MACRO H       J,K,L
 .NLIST SRC
 EKRNL  I,J,K,L
 .LIST  SRC
 .ENDM
.ENDM   EINST
        ;
        ;
     ;MNEMONIC  OPCODE  ;OPERATION
EINST   SUB,    200     ;SUBTRACT
EINST   CMP,    201     ;COMPARE
EINST   SBC,    202     ;SUBTRACT WITH CARRY
```

```
EINST   AND,    204     ;LOGICAL AND
EINST   BIT,    205     ;BIT TEST
EINST   LDA,    206     ;LOAD ACCUMULATOR
EINST   STA,    207     ;STORE ACCUMULATOR
EINST   EOR,    210     ;EXCLUSIVE OR
EINST   ADC,    211     ;ADD WITH CARRY
EINST   ORA,    212     ;LOGICAL OR
EINST   ADD,    213     ;ADD
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  ALL SHORT FORM ACCX INSTRUCTIONS
        ;
.MACRO  SKRNL   I,J,K
 .IF    NB,<K>          ;TWO ARGS - IMMEDIATE/INDEXED
  .IF   IDN     <J>,#   ;CHECK IMMEDIATE
   .IIF EQ      <I>-207 ,.ERROR ;STAA #
   .IIF EQ      <I>-307 ,.ERROR ;STAB #
   .BYTE        I,K
   .MEXIT
  .ENDC
  .IF   IDN     <K>,X   ;CHECK INDEXED
   .IIF LT,J    .ERROR  ;NEGATIVE OFFSET
   .BYTE        I+40,J          ;OFFSET TOO LARGE
   .MEXIT
  .ENDC
  .ERROR        ;BAD INSTRUCTION
  .MEXIT
 .ENDC
 .IF    NB,<J>          ;ONE ARG - DIRECT/EXTENDED
  .IIF  EQ,J&177400     .BYTE   I+20,J
  .IIF  NE,J&177400     .BYTE   I+60,J&177400/400,J&377
  .MEXIT
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  SINST   H,I
 .MACRO H       J,K
 .NLIST SRC
 SKRNL  I,J,K
 .LIST  SRC
 .ENDM
.ENDM   SINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
        ;
SINST   SUBA,   200     ;SUBTRACT
SINST   SUBB,   300
SINST   CMPA,   201     ;COMPARE
SINST   CPMB,   301
SINST   SBCA,   202     ;SUBTRACT WITH CARRY
SINST   SBCB,   302
SINST   ANDA,   204     ;LOGICAL AND
SINST   ANDB,   304
SINST   BITA,   205     ;BIT TEST
SINST   BITB,   305
SINST   LDAA,   206     ;LOAD ACCUMULATOR
SINST   LDAB,   306
SINST   STAA,   207     ;STORE ACCUMULATOR
SINST   STAB,   307
SINST   EORA,   210     ;EXCLUSIVE OR
SINST   EORB,   310
SINST   ADCA,   211     ;ADD WITH CARRY
SINST   ADCB,   311
SINST   ORAA,   212     ;LOGICAL OR
SINST   ORAB,   312
SINST   ADDA,   213     ;ADD
SINST   ADDB,   313
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  STACK AND INDEX REGISTER INSTRUCTIONS
        ;
.MACRO  FKRNL   I,J,K,L
```

```
 .IF    B,<K>           ;ONE ARG - DIRECT/EXTENDED MODE
  .IF   NB,<J>
   .IIF NE,J&177400     .BYTE   I+60,J&177400/400,J&377
   .IIF EQ,J&177400     .BYTE   I+20,J
   .MEXIT
  .ENDC
  .ERROR        ;BAD INSTRUCTION
  .MEXIT
 .ENDC
 .IF    IDN     <J>,#   ;IMMEDIATE MODE
  .IIF  EQ      <I>-217 ,.ERROR ;STS #
  .IIF  EQ      <I>-317 ,.ERROR ;STX #
  .IIF  NB,<L>  .BYTE   I,K,L
  .IIF  B,<L>   .BYTE   I,K&177400/400,K&377
  .MEXIT
 .ENDC
 .IF    B,<L>
  .IF   NB,<K>
   .IF  IDN     <K>,X           ;INDEXED
    .IIF        LT,J    .ERROR  ;NEGATIVE OFFSET
    .BYTE       I+40,J
    .MEXIT
   .ENDC
  .ENDC
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  FINST   H,I
 .MACRO H       J,K,L
 .NLIST SRC
 FKRNL  I,J,K,L
 .LIST  SRC
 .ENDM
.ENDM   FINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
        ;
FINST   CPX,    214     ;COMPARE TO INDEX
FINST   LDS,    216     ;LOAD STACK REGISTER
FINST   LDX,    316     ;LOAD INDEX REGISTER
FINST   STS,    217     ;STORE STACK REGISTER
FINST   STX,    317     ;STORE INDEX REGISTER
        ;
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  ASSEMBLER DIRECTIVES
        ;
.MACRO  FCB     A,B,C,D,E,F,G,H,I,J
.NLIST  SRC
 .IIF   NB,<A>  .BYTE   A
 .IIF   NB,<B>  .BYTE   B
 .IIF   NB,<C>  .BYTE   C
 .IIF   NB,<D>  .BYTE   D
 .IIF   NB,<E>  .BYTE   E
 .IIF   NB,<F>  .BYTE   F
 .IIF   NB,<G>  .BYTE   G
 .IIF   NB,<H>  .BYTE   H
 .IIF   NB,<I>  .BYTE   I
 .IIF   NB,<J>  .BYTE   J
.LIST   SRC
.ENDM   FCB
        ;
.MACRO  FCC     H
.NLIST  SRC
.ASCII  /H/
.LIST   SRC
.ENDM   FCC
        ;
.MACRO  FDB     A,B,C,D,E,F,G,H,I,J
.NLIST  SRC
 .IIF   NB,<A>  .BYTE   A&177400/400,A&377
 .IIF   NB,<B>  .BYTE   B&177400/400,B&377
 .IIF   NB,<C>  .BYTE   C&177400/400,C&377
```

```
 .IIF   NB,<D>  .BYTE   D&177400/400,D&377
 .IIF   NB,<E>  .BYTE   E&177400/400,E&377
 .IIF   NB,<F>  .BYTE   F&177400/400,F&377
 .IIF   NB,<G>  .BYTE   G&177400/400,G&377
 .IIF   NB,<H>  .BYTE   H&177400/400,H&377
 .IIF   NB,<I>  .BYTE   I&177400/400,I&377
 .IIF   NB,<J>  .BYTE   J&177400/400,J&377
.LIST   SRC
.ENDM   FDB
        ;
.MACRO  ORG     H
.NLIST  SRC
.=<H>
.LIST   SRC
.ENDM   ORG
        ;
.MACRO  RMB     H
.NLIST  SRC
.BLKB   H
.LIST   SRC
.ENDM   RMB
        ;
        ;
        ;
.NLIST  TTM     ;PRINTING MODE
.LIST   MD,MEB  ;ENABLE PRINTING OF MACRO EXPANSIONS
.LIST           ;LIST PROGRAM PROPER
        .PAGE
```

```
        .TITLE  SERIAL 'PORT'
        .SBTTL  SERIAL PORT AS SEEN ON THE ARB-11 BUS
        ;
        ;[X XXX XXX XXX XXN NNN]  00    KEYBOARD BUFFER STATUS REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;   0   0   0   0   0   0   0   0      R   I   0   0   0   0   0   0
        ;
        ;       R -     <07:00> DATA READY (R)
        ;                       CLEARED BY READ OF
        ;                       DATA <07:00> OR <15:00> OF (02)
        ;       I -     INTERRUPT ENABLE BIT (R/W)
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  02    FUNCTION/KEYBOARD BUFFERS
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;[ S7  S6  S5  S4  S3  S2  S1  S0 ] [ K7  K6  K5  K4  K3  K2  K1  K0 ]
        ;
        ;       <S7:S0> - FUNCTION DATA READ BUFFER
        ;       <K7:K0> - KEYBOARD INPUT BUFFER
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  04    PRINTER BUFFER STATUS REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;   0   0   0   0   0   0   0   0      R   I   0   0   0   0   0   0
        ;
        ;       R -     <07:00> PRINTER READY (R)
        ;                       CLEARED BY WRITE OF
        ;                       DATA <07:00> OR <15:00> OF (06)
        ;       I -     INTERRUPT ENABLE BIT (R/W)
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  06    FUNCTION/PRINTER BUFFERS
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;[ C4  C3  C2  C1  C0  F2  F1  F0 ] [ P7  P6  P5  P4  P3  P2  P1  P0 ]
        ;
        ;       <P7-P0> -       PRINTER DATA (R/W)
        ;       <F2-F1> -       FUNCTION CODE
        ;       <C4-C0> -       CONTROL BITS (SEE FUNCTIONS)
        ;
        ;
        .PAGE
        ;
        ;[X XXX XXX XXX XXN NNN]  10    FUNCTION INPUT BUFFER STATUS REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;   0   0   0   0   0   0   0   0      R   I   0   0   0   0   0   0
        ;
        ;       R -     <07:00> FUNCTION DATA READY (R)
        ;                       CLEARED BY READ OF
        ;                       DATA <15:08> OR <15:00> OF (02)
        ;       I -     INTERRUPT ENABLE BIT (R/W)
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  12    A-PORT PERIPHERAL CONTROL REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;[ H7  H6  H5  H4  H3  H2  H1  H0 ] [ L7  L6  L5  L4  L3  L2  L1  L0 ]
        ;
        ;       <H7-H0> -       <15:08> PIA CONTROL FOR (02)
        ;       <L7-L0> -       <07:00> PIA CONTROL FOR (02)
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  14    FUNTION CODE STATUS REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
```

```
;   0   0   0   0   0   0   0   0      R   I   0   0   0   0   0   0
;
;       R -     <07:00> FUNCTION CODE DONE (R)
;                       CLEARED BY WRITE OF
;                       DATA <15:08> OR <15:00> OF (06)
;       I -     INTERRUPT ENABLE BIT (R/W)
;
;
;[X XXX XXX XXX XXN NNN]  16    B-PORT PERIPHERAL CONTROL REGISTER
; <15:08> H                          <07:00> L
;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
;   __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
;[ H7  H6  H5  H4  H3  H2  H1  H0 ] [ L7  L6  L5  L4  L3  L2  L1  L0 ]
;
;       <H7-H0> -       <15:08> PIA CONTROL FOR (02)
;       <L7-L0> -       <07:00> PIA CONTROL FOR (02)
;
;
.PAGE
;
;       FUNCTION CODES
;
; F2 F1 F0
;  __ __ __
;  0  0  0      NORMAL SERIAL PORT CONFIGURATION
;               BYTE 02 - KEYBOARD BUFFER
;               BYTE 03 - MODEM STATUS REGISTER (WHEN ENABLED, ELSE=0)
;               BYTE 06 - PRINTER BUFFER
;               BYTE 07 - <07:03> MODEM CONTROL REGISTER
;
;  0  0  1      MODEM CONTROL REGISTER READ/WRITE
;               BYTE 02 - KEYBOARD BUFFER
;               BYTE 03 - CURRENT CONTROL REGISTER
;               BYTE 06 - LOAD CONTROL REGISTER BUFFER
;               BYTE 07 - FUNCTION SELECT
;
;  0  1  0      LINE SERVICES ENABLES
;               BYTE 02 - KEYBOARD BUFFER
;               BYTE 03 - CURRENT ENABLES
;               BYTE 06 - LOAD ENABLES BUFFER
;               BYTE 07 - FUNCTION SELECT
;
;  0  1  1      MODEM SERVICES ENABLES
;               BYTE 02 - KEYBOARD BUFFER
;               BYTE 03 - CURRENT ENABLES
;               BYTE 06 - LOAD ENABLES BUFFER
;               BYTE 07 - FUNCTION SELECT
;
;  1  0  0      LINE CONTROL REGISTER
;               BYTE 02 - KEYBOARD BUFFER
;               BYTE 03 - CURRENT LINE CONTROL
;               BYTE 06 - LOAD CONTROL BUFFER
;               BYTE 07 - FUNCTION SELECT
;
;  1  0  1      LOW BYTE DIVISOR LATCH SELECT
;               BYTE 02 - KEYBOARD BUFFER
;               BYTE 03 - CURRENT DLL
;               BYTE 06 - LOAD DIVISOR BUFFER
;               BYTE 07 - FUNCTION SELECT
;
;  1  1  0      HIGH BYTE DIVISOR LATCH SELECT
;               BYTE 02 - KEYBOARD BUFFER
;               BYTE 03 - CURRENT DLM
;               BYTE 06 - LOAD DIVISOR BUFFER
;               BYTE 07 - FUNCTION SELECT
;
;  1  1  1      ABSOLUTE BINARY LOADER SELECT
;               BYTE 02 - KEYBOARD BUFFER
;               BYTE 03 - CURRENT CHECK SUM
;               BYTE 06 - BINARY INPUT BUFFER
;               BYTE 07 - FUNCTION SELECT
;
.PAGE
;
;       BIT PATTERN DEFINITIONS
;
```

```
;  KEYBOARD / PRINTER  NORMAL MODE
; FUNCTION <F2:F0>=000          WRITE NEW DATA INTO <C7:C3>
;       BITS <C7:C3> CORRESPOND TO BITS <4:0> OF
;       THE MODEM CONTROL REGISTER OF THE INS8250 ACE.
;       BITS <S7:S0> IS THE MODEM STATUS REGISTER OF
;       THE INS8250 ACE.
;
;  MODEM CONTROL REGISTER ACCESS
; FUNCTION <F2:F0>=001          WRITE NEW DATA INTO <P7:P3>
;       BITS <S7:S3> CORRESPOND TO BITS <4:0> OF
;       THE MODEM CONTROL REGISTER OF THE INS8250 ACE.
;
;  LINE INTERRUPT SERVICES ENABLES
; FUNCTION <F2:F0>=010          WRITE NEW DATA INTO <P3:P0>
;       BITS <S3:S0> ARE THE LINE INTERRUPT SERVICE
;       ENABLES CORRESPONDING TO BITS <4:1> OF THE
;       LINE STATUS REGISTER OF THE INS8250 ACE.
;
;  MODEM INTERRUPT SERVICES ENABLES
; FUNCTION <F2:F0>=011          WRITE NEW DATA INTO <P7,P3:P0>
;       BITS <S3:S0> ARE MODEM INTERRUPT SERVICE
;       ENABLES CORRESPONDING TO BITS <3:0> OF THE
;       MODEM STATUS REGISTER OF THE INS8250 ACE.
;       <P7>=1 DISABLES MODEM STATUS TO BYTE 03
;
;  LINE CONTROL REGISTER ACCESS
; FUNCTION <F2:F0>=100          WRITE NEW DATA INTO <P6:P0>
;       BITS <S6:S0> ARE THE CONTENTS OF BITS <6:0>
;       OF THE LINE CONTROL REGISTER OF THE
;       INS8250 ACE.
;
;  LOW BYTE DIVISOR LATCH ACCESS
; FUNCTION <F2:F0>=101          WRITE NEW DATA INTO <P7:P0>
;       BITS <S7:S0> CORRESPOND TO BITS <7:0> OF THE
;       LOW BYTE DIVISOR LATCH OF THE INS8250 ACE.
;
;  HIGH BYTE DIVISOR LATCH ACCESS
; FUNCTION <F2:F0>=110          WRITE NEW DATA INTO <P7:P0>
;       BITS <S7:S0> CORRESPOND TO BITS <7:0> OF THE
;       HIGH BYE DIVISOR LATCH OF THE INS8250 ACE.
;
;  ABSOLUTE BINARY LOADER ACCESS
; FUNCTION <F2:F0>=111          WRITE BINARY INTO <P7:P0>
;       BITS <S7:S0> IS THE CURRENT CHECK SUM RESULT
;       FROM LOADING BINARY DATA
;       ODD JUMP ADDRESS TERMINATES LOADER
;       EVEN JUMP ADDRESS TRANSFERS CONTROL TO JUMP ADDRESS
;
.PAGE
.SBTTL  SOFTWARE DEFINITIONS
;
;         THE SERIAL PORT SOFTWARE REQUIRES
;       THE 16-BIT PIA SOFTWARE PACKAGE '6821A.MAC',
;       THE 8250 ACE PACKAGE '8250A.MAC', AND
;       THE ABSOLUTE BINARY LOADER 'LOADER.MAC'.
;
;
$A.8250 =1
A.8250  =220    ;SERIAL PORT ADDRESS
;
$A.6821 =1
A.6821  =200    ;16-BIT BUS PORT ADDRESS
;
$LOADER =1      ;USE ABSOLUTE LOADER
;
;
PGMSAV  =174000 ;ROM ADDRESS
VARSAV  =0      ;DIRECT VARIABLE SPACE TO 177
;
STACK   =177    ;STACK AREA (64-BYTES)
;
;       INTERRUPT VECTORS
;
.=177740
FDB     SPRIUS  ;OOPS !
FDB     SPRIUS  ;OOPS !
```

```
        FDB     SPRIUS  ;OOPS !
        FDB     SPRIUS  ;OOPS !
IRQ0:   FDB     SPRIUS  ;NO HARDWARE
IRQ1:   FDB     SPRIUS  ;NO HARDWARE
IRQ2:   FDB     A.AL$I  ;A-PORT <7:0> INTERRUPT
IRQ3:   FDB     A.AH$I  ;A-PORT <15:8> INTERRUPT
IRQ4:   FDB     A.BL$I  ;B-PORT <7:0> INTERRUPT
IRQ5:   FDB     A.BH$I  ;B-PORT <15:8> INTERRUPT
IRQ6:   FDB     SPRIUS  ;NO HARDWARE
IRQ7:   FDB     A.82$I  ;8250 ACE INTERRUPT
IRQN:   FDB     SPRIUS  ;NO HARDWARE
SWINT:  FDB     SWIRQ   ;SWI INTERRUPT
NMINT:  FDB     NMIRQ   ;ATTACHED PROCESSOR INIT INTERRUPT
RESINT: FDB     RESTRT  ;POWER UP RESTART VECTOR
        ;
        .PAGE
        .SBTTL  STARTUP AND BACKGROUND PROGRAM
        ;
        .=VARSAV
NMIRQV: .BYTE   0,0     ;INIT INTERRUPT VECTOR
SWIRQV: .BYTE   0,0     ;SWI INTERRUPT VECTOR
S.FNCT: .BYTE   0       ;DISPATCH FUNCTION
MSFLAG: .BYTE   0       ;MODEM STATUS FLAG
        VARSAV=.
        .=PGMSAV
        ;
NMIRQ:  LDX     NMIRQV          ;GET VECTOR
        BEQ     SPRIUS          ;BRANCH IF NOT DEFINED
        JSR     0,X             ;ELSE DO IT
SPRIUS: RTI                     ;HOW DID WE GET HERE ?
        ;
SWIRQ:  LDX     SWIRQV          ;GET VECTOR
        BEQ     1$              ;BRANCH IF NOT DEFINED
        JMP     0,X             ;SPECIAL SWI CALL TO ACCESS
1$:     RTI                     ;INTERRUPT DRIVEN ROUTINES
        ;
RESTRT: LDS     #,STACK         ;SET UP STACK POINTER
        LDX     #,0
        STX     NMIRQV          ;SET NMI FOR NOTHING
        STX     SWIRQV          ;SET SWI FOR NOTHING
        JSR     A.82$B          ;INIT SERIAL PORT BAUD-RATE
        BSR     PINITS          ;AND SERIAL PORT IN GENERAL
        ;
        ;       BACKGROUND PROGRAM
        ;
1$:     LDS     #,STACK         ;SET THE STACK POINTER
        CLR     177740          ;ENABLE ALL INTERRUPTS
        CLI                     ;(VIA 6828 CONTROLLER)
        LDX     #,0             ;LOOP COUNTER
2$:     INX                     ;LOOP HERE 64K TIMES
        BNE     2$
        JSR     A.LNUP          ;RE-ENABLE LINE
        JSR     A.RCUP          ;RE-ENABLE RCVR
        JSR     A.XMUP          ;RE-ENABLE XMTR
        JSR     A.MOUP          ;RE-ENABLE MODEM
        JSR     A.ALUP          ;RE-ENABLE <7:0> INPUT
        JSR     A.AHUP          ;RE-ENABLE <15:8> INPUT
        JSR     A.BLUP          ;RE-ENABLE <7:0> OUTPUT
        JSR     A.BHUP          ;RE-ENABLE <15:8> OUTPUT
        BRA     1$
        ;
        .PAGE
        .SBTTL  PORT INITIALIZATION
        ;
PINITS: JSR     LOAD$I          ;INITIALIZE ABSOLUTE LOADER
        ;
        ;       INSURE A.8250 ACCESS IS NOT TO DLL/DLM
        ;
        JSR     A.LCRR          ;GET STATUS
        AND B   #,177           ;SET DLAB=0
        JSR     A.LCRW          ;SET STATUS
        ;
        ;       SET FUNCTION TO XMTR AND ENABLE ALL LINE / MODEM
        ;
        CLR     S.FNCT          ;XMTR FUNCTION
        LDA B   #,17            ;MODEM ENABLE
```

```
        STA B   A.MNBL
        LDA B   #,36            ;LINE ENABLE
        STA B   A.LNBL
        STA B   MSFLAG          ;AND FLAG
        ;
        ;       INITIALIZE A.8250 SERVICE VECTORS
        ;
        LDX     #,S.MODE        ;A.8250 MODEM VECTOR
        JSR     A.MOSV
        LDX     #,S.RCVR        ;A.8250 RCVR VECTOR
        JSR     A.RCSV
        LDX     #,S.XMTR        ;A.8250 XMTR VECTOR
        JSR     A.XMSV
        LDX     #,A.82$L        ;A.8250 LINE VECTOR
        JSR     A.LNSV
        ;
        ;       SET UP PARALLEL OUTPUT
        ;
        LDX     #,0             ;NO SERVICE
        LDA B   #,44            ;CB2(ON E, LOW), CB1(-, HIGH)
        LDA A   #,74
        JSR     A.BLDF          ;SAVE CONTROL
        LDX     #,0             ;NO SERVICE
        LDA B   #,44            ;CB2(ON E, LOW), CB1(-, HIGH)
        LDA A   #,74
        JSR     A.BHDF          ;SAVE CONTROL
        LDA B   #,377           ;OUTPUT
        JSR     A.BLDR
        LDA B   #,377           ;OUTPUT
        JSR     A.BHDR
        ;
        ;       SET UP PARALLEL INPUT
        ;
        LDX     #,S.INAL        ;SERVICE VECTOR
        LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
        LDA A   #,74
        JSR     A.ALDF          ;SAVE CONTROL
        LDX     #,S.INAH        ;SERVICE VECTOR
        LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
        LDA A   #,74
        JSR     A.AHDF          ;SAVE CONTROL
        LDA B   #,0             ;INPUT
        JSR     A.ALDR
        LDA B   #,0             ;INPUT
        JSR     A.AHDR
        ;
        ;       ENABLE/DISABLE INTERRUPTS
        ;
        JSR     A.ALDN          ;CLEAR, CA2='1'
        JSR     A.ALUP          ;ENABLE
        JSR     A.ALRD+2        ;CLEAR IRQA, CA2='0'
        ;
        JSR     A.AHDN          ;CLEAR, CA2='1'
        JSR     A.AHUP          ;ENABLE
        JSR     A.AHRD+2        ;CLEAR IRQ, CA2='0'
        ;
        JSR     A.BLDN          ;CLEAR, CB2='1'
        JSR     A.BLRD+2        ;CLEAR IRQ, CB2='1'
        JSR     A.BLUP          ;ENABLE
        ;
        JSR     A.BHDN          ;CLEAR, CB2='1'
        JSR     A.BHRD+2        ;CLEAR IRQ, CB2='1'
        JSR     A.BHUP          ;ENABLE
        ;
        JSR     A.MOUP
        JSR     A.XMUP
        JSR     A.RCUP
        JSR     A.LNUP
        ;
        RTS
        ;
        ;       PORT SERVICE ROUTINES
        ;
S.XMTR: LDA A   S.FNCT          ;GET FUNCTION
        CLC                     ;DEFAULT NO DATA
        BNE     1$              ;BRANCH IF NOT XMTR
```

```
        JSR     A.ALUP          ;ENABLE INPUT INTERRUPT
        JSR     A.ALRD          ;READ INPUT DATA
1$:     RTS                     ;SEND TO XMTR
        ;
S.RCVR: JSR     A.BLWT+2        ;TRANSFER CHARACTER TO OUTPUT
        RTS                     ;READY OR NOT !
        ;
S.MODE: LDA A   S.FNCT          ;GET FUNCTION
        BNE     2$              ;BRANCH IF NOT STATUS
        PSH B                   ;SAVE B FOR A MOMENT
        LDA A   MSFLAG          ;CHECK FOR MODEM STATUS DISABLED
        BPL     1$              ;ENABLED - SKIP
        CLR B                   ;CLEAR BYTE
1$:     JSR     A.BHWT+2        ;TRANSFER DATA
        PUL B                   ;RESTORE B
2$:     JSR     A.82$M          ;DO MODEM STATUS
        RTS
        ;
S.INAL: LDA A   S.FNCT          ;GET FUNCTION
        BNE     1$              ;BRANCH IF NOT XMTR
        JSR     A.ALDN          ;DISABLE INPUT INTERRUPT
        JSR     A.XMUP          ;ENABLE A.8250 XMTR
        RTS
1$:     JSR     A.ALRD          ;READ BYTE
        BCC     8$              ;WHAT DATA ?
        LDA A   S.FNCT          ;GET FUNCTION CODE
        DEC A                   ;MODEM CONTROL ?
        BNE     2$              ;BRANCH IF NOT
        AND B   #,370           ;MASK EXTRA
        TBA
        CLC                     ;SHIFT INTO POSITION
        ROR B
        ASR B
        ASR B
        JSR     A.MCRW          ;SAVE NEW CONTROL
        TAB
        BRA     9$
2$:     DEC A                   ;LOAD LINE ENABLES ?
        BNE     3$              ;BRANCH IF NOT
        AND B   #,17            ;MASK EXTRA BITS
        ASL B                   ;SHIFT INTO POSITION
        STA B   A.LNBL          ;SAVE NEW ENABLES
        BRA     9$
3$:     DEC A                   ;LOAD MODEM ENABLES ?
        BNE     4$              ;BRANCH IF NOT
        AND B   #,217           ;MASK EXTRA BITS
        STA B   MSFLAG          ;SAVE DISABLE STATUS
        AND B   #,17            ;MASK FLAG BIT
        STA B   A.MNBL          ;SAVE NEW ENABLES
        LDA B   MSFLAG          ;GET FLAG AND ENABLES
        BRA     9$
4$:     DEC A                   ;LOAD LINE CONTROL REGISTER ?
        BNE     5$              ;BRANCH IF NOT
        AND B   #,177           ;MASK DLAB
        JSR     A.LCRW          ;SAVE NEW VALUE
        BRA     9$
5$:     DEC A                   ;LOAD DLL ?
        BNE     6$              ;BRANCH IF NOT
        JSR     A.DLLW          ;SAVE NEW VALUE
        BRA     9$
6$:     DEC A                   ;LOAD DLM ?
        BNE     7$              ;BRANCH IF NOT
        JSR     A.DLMW          ;SAVE NEW VALUE
        BRA     9$
7$:     DEC A                   ;LOADER ?
        BNE     8$              ;BRANCH IF NOT
        JSR     LOADER          ;AND PROCESS
        BRA     9$
8$:     RTS
9$:     JSR     A.BHWT+2        ;SEND DATA
        RTS
        ;
S.INAH: JSR     A.AHRD          ;READ BYTE
        BCC     8$              ;WHAT DATA ?
        TBA                     ;COPY B TO A
        AND B   #,7             ;MASK EXTRA
```

```
        STA B   S.FNCT          ;NEW FUNCTION
        BNE     1$              ;BRANCH IF NOT CONTROL
        TAB                     ;GET DATA AGAIN
        CLC                     ;POSITION DATA
        ROR B
        ASR B
        ASR B
        JSR     A.MCRW          ;SEND TO A.8250
        CLR B                   ;CLEAR BYTE
        LDA A   MSFLAG          ;MODEM STATUS DISABLED ?
        BMI     9$              ;YES - EXIT
        JSR     A.MSRR          ;GET CURRENT STATUS
        BRA     9$
1$:     DEC B                   ;GET CURRENT MODEM CONTROL ?
        BNE     2$              ;BRANCH IF NOT
        JSR     A.MCRR          ;READ CURRENT SETTING
        ASL B                   ;SHIFT INTO PLACE
        ASL B
        ASL B
        BRA     9$
2$:     DEC B                   ;LINE SERVICE ENABLES ?
        BNE     3$              ;BRANCH IF NOT
        LDA B   A.LNBL          ;ENABLES
        ASR B                   ;SHIFT INTO POSITION
        BRA     9$
3$:     DEC B                   ;MODEM SERVICE ENABLES ?
        BNE     4$              ;BRANCH IF NOT
        LDA B   MSFLAG          ;GET FLAG AND ENABLES
        BRA     9$
4$:     DEC B                   ;LINE CONTROL REGISTER ?
        BNE     5$              ;BRANCH IF NOT
        JSR     A.LCRR          ;GET REGISTER
        AND B   #,177           ;MASK OUT DLAB
        BRA     9$
5$:     DEC B                   ;DLL ACCESS ?
        BNE     6$              ;BRANCH IF NOT
        JSR     A.DLLR          ;READ VALUE
        BRA     9$
6$:     DEC B                   ;DLM ACCESS ?
        BNE     7$              ;BRANCH IF NOT
        JSR     A.DLMR          ;READ VALUE
        BRA     9$
7$:     DEC B                   ;LOADER SELECTED ?
        BNE     8$              ;BRANCH IF NOT
        JSR     LOAD$I          ;ELSE INITIALIZE LOADER
        LDA B   #,0             ;ZERO CHECK SUM
        BRA     9$
8$:     RTS
9$:     JSR     A.BHWT+2        ;SEND
        RTS
        ;
```

```
        .PAGE
        .SBTTL  ABSOLUTE BINARY LOADER
        ;
        ;         THIS LOADER ROUTINE IS AN ADAPTATION OF THE
        ;       'DEC' PAPER TAPE ABSOLUTE BINARY LOADER.
        ;         THE LOADER PROVIDES A MEANS OF LOADING OBJECT
        ;       CODE INTO ANY REGION OF MEMORY AND STARTING
        ;       PROGRAM EXECUTION.
        ;
        ;         AN ODD JUMP ADDRESS TERMINATES LOADER
        ;         AN EVEN JUMP ADDRESS TRANSFERS CONTROL TO THE JUMP ADDRESS
        ;
        .IF     NDF,$LOADER
        VARSAV  =0      ;VARIABLE SPACE
        PGMSAV  =100000 ;PROGRAM SPACE
        .=PGMSAV
        .ENDC
        ;
        PGMSAV=.
        .=VARSAV
LOAD$V: .BYTE   0,0     ;SERVICE ADDRESS
L.BYTC: .BYTE   0,0     ;BYTE COUNTER
L.ADDR: .BYTE   0,0     ;LOAD ADDRESS
L.CKSM: .BYTE   0       ;CHECK SUM
        VARSAV=.
        .=PGMSAV
        ;
        ;       INITIALIZE ENTRY POINT
        ;
LOAD$I: LDX     #,L.LDR0        ;FIRST ENTRY
        STX     LOAD$V
        RTS
        ;
        .PAGE
        .SBTTL  LOADER ROUTINE
        ;
        ;       ENTER WITH DATA IN <B>
        ;       EXITS WITH CURRENT CHECK SUM IN <B>
        ;
LOADER: LDA A   L.CKSM          ;UPDATE CHECK SUM
        ABA
        STA A   L.CKSM
        LDX     L.BYTC          ;UPDATE BYTE COUNT
        DEX
        STX     L.BYTC
        LDX     LOAD$V          ;DISPATCH ADDRESS
        JMP     0,X             ;GO TO IT
L.GBYT: TSX                     ;GET RETURN ADDRESS ON STACK
        LDX     0,X
        STX     LOAD$V          ;SAVE RETURN ADDRESS
        INS                     ;POP RETURN FROM STACK
        INS
        LDA B   L.CKSM          ;RETURN CHECK SUM IN <B>
        RTS                     ;BACK TO CALLER
        ;
L.LDR0: CLR     L.CKSM          ;CLEAR CHECK SUM
        BRA     L.LDR2
L.LDR1: CLR     L.CKSM          ;CLEAR CHECK SUM
        BSR     L.GBYT          ;BACK FOR A BYTE
L.LDR2: DEC B                   ;A '1' ?
        BNE     L.LDR1          ;LOOP UNTIL '1' FOUND
        BSR     L.GBYT          ;BACK FOR A BYTE
        TST B                   ;= '0' ?
        BNE     L.LDR0          ;LOOP BACK UNTIL '1','0' SEQUENCE
        BSR     L.GBYT          ;BACK FOR A BYTE
        STA B   L.ADDR+1        ;NOW SAVE BYTE COUNT
        BSR     L.GBYT          ;BACK FOR A BYTE
        LDA A   L.ADDR+1        ;RETRIEVE LOW ORDER COUNT
        SUB A   #,4             ;CORRECT BYTE COUNT
        SBC B   #,0
        STA A   L.BYTC+1        ;AND SAVE IT
        STA B   L.BYTC
        LDX     L.BYTC          ;L.BYTC=2 ?
        CPX     #,2
        BNE     L.LDR4          ;BRANCH IF NOT
L.LDR3: BSR     L.GBYT          ;BACK FOR A BYTE
```

```
        STA B   L.ADDR+1        ;GET JUMP ADDRESS
        BSR     L.GBYT          ;BACK FOR A BYTE
        STA B   L.ADDR
        BSR     L.GBYT          ;BACK FOR CHECK SUM BYTE !
        TST     L.CKSM          ;CHECK FOR ERROR
        BNE     L.LDR6          ;SKIP ON ERROR
        LDA A   L.ADDR+1        ;CHECK FOR ODD ADDRESS
        BIT A   #,1
        BNE     L.LDR6          ;TERMINATE BY RESTARTING SCAN
        LDX     L.ADDR          ;ELSE GO TO ADDRESS
        JSR     0,X
        BRA     L.LDR6          ;ON QUICK RETURN - CONTINUE
L.LDR4: BSR     L.GBYT          ;BACK FOR A BYTE
        STA B   L.ADDR+1        ;GET LOAD ADDRESS
        BSR     L.GBYT          ;BACK FOR A BYTE
        STA B   L.ADDR
L.LDR5: BSR     L.GBYT          ;BACK FOR A BYTE
        LDX     L.BYTC          ;CHECK BYTE COUNT
        BMI     L.LDR6          ;BRANCH IF DONE
        LDX     L.ADDR          ;GET LOAD ADDRESS
        STA B   0,X             ;STORE DATA
        INX                     ;UPDATE LOAD ADDRESS
        STX     L.ADDR
        BRA     L.LDR5
L.LDR6: BSR     L.GBYT          ;CURRENT BYTE WAS CHECK SUM RESULT
        BRA     L.LDR0          ;START NEW SEQUENCE
        ;
```

```
        .PAGE
        .SBTTL  A.6821 PERIPHERAL INTERFACE SOFTWARE
        ;
        ;         THIS SOFTWARE PACKAGE SUPPORTS THE
        ;       6821 PERIPHERAL INTERFACE ADAPTER (PIA).
        ;         ENTRY POINTS TO THE SOFTWARE ARE DEFINED
        ;       FOR EXTERNAL ACCESS:
        ;
        ;         THE FOLLOWING DATA DIRECTION ROUTINES ARE
        ;       ENTERED WITH
        ;               <B> = DIRECTION CONTROL
        ;       BY A JSR ____
        ;
        ;       A.AHDR --> A-PORT <15:8> DIRECTION
        ;       A.ALDR --> A-PORT <7:0> DIRECTION
        ;       A.BHDR --> B-PORT <15:8> DIRECTION
        ;       A.BLDR --> B-PORT <7:0> DIRECTION
        ;
        ;         THE FOLLOWING CONTROL ROUTINES ARE ENTERED WITH
        ;               <X> = SERVICE ADDRESS
        ;               <B> = INTERRUPT ENABLE CONTROL
        ;               <A> = INTERRUPT DISABLE CONTROL
        ;       BY A JSR ____
        ;
        ;       A.AHDF --> A-PORT <15:8>
        ;       A.ALDF --> A-PORT <7:0>
        ;       A.BHDF --> B-PORT <15:8>
        ;       A.BLDF --> B-PORT <7:0>
        ;
        ;         THE FOLLOWING INTERRUPT ENABLE ROUTINES ARE
        ;       ENTERED BY JSR ____
        ;
        ;       A.AHUP --> A-PORT <15:8> INTERRUPT ENABLE
        ;       A.ALUP --> A-PORT <7:0> INTERRUPT ENABLE
        ;       A.BHUP --> B-PORT <15:0> INTERRUPT ENABLE
        ;       A.BLUP --> B-PORT <7:0> INTERRUPT ENABLE
        ;
        ;         THE FOLLOWING INTERRUPT DISABLE ROUTINES ARE
        ;       ENTERED BY JSR ____
        ;
        ;       A.AHDN --> A-PORT <15:0> INTERRUPT DISABLE
        ;       A.ALDN --> A-PORT <7:0> INTERRUPT DISABLE
        ;       A.BHDN --> B-PORT <15:8> INTERRUPT DISABLE
        ;       A.BLDN --> B-PORT <7:0> INTERRUPT DISABLE
        ;
        ;         THE FOLLOWING ARE INTERRUPT ENTRY POINTS
        ;       FOR THE DUAL PIA PORTS
        ;
        ;       A.AH$I --> A-PORT <15:8> IRQ
        ;
        ;       A.AL$I --> A-PORT <7:0> IRQ
        ;
        ;       A.BH$I --> B-PORT <15:8> IRQ
        ;
        ;       A.BL$L --> B-PORT <7:0> IRQ
        ;
        .PAGE
        ;
        ;         THE FOLLOWING ENTRY POINTS ARE USED TO TRANSFER
        ;       DATA TO THE VARIOUS PIA I/O REGISTERS. THE MAIN
        ;       ENTRY POINT VERIFIES THAT AT LEAST ONE OF THE
        ;       PORT IRQ FLAGS IS SET BEFORE TRANSFERRING DATA.
        ;       IF DATA IS TRANSFERED THEN THE 'C' BIT = 1,
        ;       IF THE PORT WAS NOT READY THEN NO TRANSFER
        ;       IS PERFORMED AND THE 'C' BIT = 0.
        ;         THE ROUTINES MAY BE ENTERED AT THE ENTRY POINT + 2
        ;       IF NO IRQ FLAG CHECKS ARE TO BE MADE.
        ;         BYTE DATA IS PASSED IN <B>.
        ;         WORD DATA IS PASSED IN <B,A>
        ;
        ;       CALL BY JSR ____
        ;
        ;       A.AHRD --> A-PORT <15:8> READ
        ;       A.ALRD --> A-PORT <7:0> READ
        ;       A.AWRD --> A-PORT <15:0> READ
        ;       A.BHRD --> B-PORT <15:8> READ
```

```
;       A.BLRD --> B-PORT <7:0> READ
;       A.BWRD --> B-PORT <15:0> READ
;
;
;       A.AHWT --> A-PORT <15:8> WRITE
;       A.ALWT --> A-PORT <7:0> WRITE
;       A.AWWT --> A-PORT <15:0> WRITE
;       A.BHWT --> B-PORT <15:8> WRITE
;       A.BLWT --> B-PORT <7:0> WRITE
;       A.BWWT --> B-PORT <15:0> WRITE
;
.PAGE
.SBTTL  A.6821 PIA HARDWARE
;
;         THE PIA CONFIGURATION  CONSISTS OF TWO
;       MC6821 PERIPHERAL INTERFACE ADAPTERS (PIA'S)
;       WHICH MAY BE ADDRESSED AS 16-BIT PORTS.
;       ONE PIA UNIT FORMS THE HIGH ORDER PART <15:8>
;       AND THE OTHER THE LOW ORDER PART <7:0>
;
; REGISTER 0
CRAH    =0      ;A-PORT <15:8> CONTROL REGISTER
;
; REGISTER 1
CRAL    =1      ;A-PORT <7:0> CONTROL REGISTER
;
; REGISTER 2
CRBH    =2      ;B-PORT <15:8> CONTROL REGISTER
;
; REGISTER 3
CRBL    =3      ;B-PORT <7:0> CONTROL REGISTER
;
; REGISTER 4
DDRAH   =4      ;CRAH<2>=0, A-PORT <15:8> DATA DIRECTION REGISTER
PRAH    =4      ;CRAH<2>=1, A-PORT <15:8> DATA REGISTER
;
; REGISTER 5
DDRAL   =5      ;CRAL<2>=0, A-PORT <7:0> DATA DIRECTION REGISTER
PRAL    =5      ;CRAL<2>=1, A-PORT <7:0> DATA REGISTER
;
; REGISTER 6
DDRBH   =6      ;CRBH<2>=0, B-PORT <15:8> DATA DIRECTION REGISTER
PRBH    =6      ;CRBH<2>=1, B-PORT <15:8> DATA REGISTER
;
; REGISTER 7
DDRBL   =7      ;CRBL<2>=0, B-PORT <7:0> DATA DIRECTION REGISTER
PRBL    =7      ;CRBL<2>=1, B-PORT <7:0> DATA REGISTER
;
.PAGE
;
;       A-PORT CONTROL REGISTERS
;
; CONTROL REGISTERS CRA_  - READ/WRITE REGISTERS
;                                 CONTROLLING INTERRUPTS AND STROBES
;
; BIT 1 0       CONTROL OF INTERRUPT INPUT CA1
;               ACTIVE TRANSITION
;     _ _       TO SET CRA<7>           /IRQA OUTPUT
;     0 0               (-)             DISABLED
;     0 1               (-)             /CRA<7>
;     1 0               (+)             DISABLED
;     1 1               (+)             /CRA<7>
;       CRA<7> CLEARED BY A READ OF CORRESPONDING PRA_
;       /IRQA FOLLOWS STATE OF /CRA<7> WHEN ENABLED
;
; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
;     _
;     0         ACCESS DATA DIRECTION REGISTERS
;     1         ACCESS PORT DATA REGISTER
;
; BIT 5 4 3     CONTROL OF CA2 AS AN INTERRUPT INPUT
;               ACTIVE TRANSITION
;     _ _ _     TO SET CRA<6>           /IRQB OUTPUT
;     0 0 0             (-)             DISABLED
;     0 0 1             (-)             /CRA<6>
;     0 1 0             (+)             DISABLED
```

```
;     0 1 1             (+)             /CRA<6>
;       CRA<6> CLEARED BY READ OF CORRESPONDING PRA_
;       /IRQB FOLLOWS STATE OF CRA<6> WHEN ENABLED
;
; BIT 5 4 3     CONTROL OF CA2 AS AN OUTPUT
;     _ _ _     CA2 LOW                 CA2 HIGH
;     1 0 0     AFTER READ TO PRA_      WHEN CRA<7> IS SET
;               ON (+) OF NEXT E        BY CA1 TRANSITION
;     1 0 1     AFTER READ TO PRA_      ON SECOND (+) E AFTER
;               ON (+) OF NEXT E        PIA DESELECTED
;     1 1 0     =CRA<3>
;     1 1 1                             =CRA<3>
;
; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CA2, CLEARED BY READ OF PRA_
;
; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CA1, CLEARED BY READ OF PRA_
;
;       DATA DIRECTION REGISTER (DDRA_)
; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
;               BY 0'S AND OUTPUT BITS BY 1'S.
;
;       PERIPHERAL DATA REGISTER (PRA_)
; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGISTERS
;
.PAGE
.SBTTL  A.6821 DETAILS
;
;       B-PORT CONTROL REGISTERS
;
; CONTROL REGISTERS CRB_  - READ/WRITE REGISTERS
;                               CONTROLLING INTERRUPTS AND STROBES
;
; BIT 1 0       CONTROL OF INTERRUPT INPUT CB1
;               ACTIVE TRANSITION
;     _ _       TO SET CRB<7>           /IRQA OUTPUT
;     0 0               (-)             DISABLED
;     0 1               (-)             /CRB<7>
;     1 0               (+)             DISABLED
;     1 1               (+)             /CRB<7>
;       CRB<7> CLEARED BY A READ OF CORRESPONDING PRB_
;       /IRQA FOLLOWS STATE OF /CRB<7> WHEN ENABLED
;
; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
;
;     _
;     0         ACCESS DATA DIRECTION REGISTERS
;     1         ACCESS PORT DATA REGISTER
;
; BIT 5 4 3     CONTROL OF CB2 AS AN INTERRUPT INPUT
;               ACTIVE TRANSITION
;     _ _ _     TO SET CRB<6>           /IRQB OUTPUT
;     0 0 0             (-)             DISABLED
;     0 0 1             (-)             /CRB<6>
;     0 1 0             (+)             DISABLED
;     0 1 1             (+)             /CRB<6>
;       CRB<6> CLEARED BY READ OF CORRESPONDING PRB_
;       /IRQB FOLLOWS STATE OF CRB<6> WHEN ENABLED
;
; BIT 5 4 3     CONTROL OF CB2 AS AN OUTPUT
;     _ _ _     CB2 LOW                 CB2 HIGH
;     1 0 0     AFTER WRITE TO PRB      WHEN CRB<7> IS SET
;               ON (+) OF NEXT E        BY CB1 TRANSITION
;     1 0 1     AFTER WRITE TO PRB      ON SECOND (+) E AFTER
;               ON (+) OF NEXT E        PIA DESELECTED
;     1 1 0     =CRB<3>
;     1 1 1                             =CRB<3>
;
; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CB2, CLEARED BY READ OF PRB_
;
; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CB1, CLEARED BY READ OF PRB_
;
;       DATA DIRECTION REGISTER (DDRB_)
```

```
        ; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
        ;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
        ;               BY 0'S AND OUTPUT BITS BY 1'S.
        ;
        ;       PERIPHERAL DATA REGISTER (PRB_)
        ; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGISTERS
        ;
        .PAGE
        .SBTTL  A.6821 VARIABLES AND POINTERS
        ;
        ;       PORT STATUS AND BUFFERSS
        ;
        .IF     NDF,$A.6821
        A.6821  =       200     ;PIA PORT DEFAULT ADDRESS
        VARSAV  =       0       ;DEFAULT VARIABLE ADDRESS
        PGMSAV  =       100000  ;DEFAULT PROGRAM ADDRESS
        .=PGMSAV
        .ENDC
        ;
        PGMSAV=.
        .=VARSAV
        ;
A.APRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
        .BYTE   0,0     ;<7:0> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
A.BPRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
        .BYTE   0,0     ;<7:0> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
        ;
        VARSAV=.
        .=PGMSAV
        ;
        .PAGE
        .SBTTL  A.6821 A-PORT CONTROL ROUTINES
        ;
        ;       DATA DIRECTION LOAD ROUTINES
        ;       ENTER WITH DIRECTION CODE IN <B>
        ;
A.AHDR: LDA A   CRAH+A.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRAH+A.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRAH+A.6821    ;SET DIRECTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRAH+A.6821     ;AND RESTORE
        RTS
        ;
A.ALDR: LDA A   CRAL+A.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRAL+A.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRAL+A.6821    ;SET DIRCTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRAL+A.6821     ;AND RESTORE
        RTS
        ;
        .PAGE
        ;       CONTROL AND SERVICE VECTOR LOADER
        ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
        ;       'UP' CONTROL IN <B>, AND
        ;       'DN' CONTROL IN <A>
        ;
A.AHDF: STX     A.APRT          ;SAVE SERVICE ROUTINE ADDRESS
        STA B   A.APRT+2        ;'UP' CONTROL
        STA A   A.APRT+3        ;'DN' CONTROL
        RTS
        ;
A.ALDF: STX     A.APRT+4        ;SAVE SERVICE ROUTINE ADDRESS
        STA B   A.APRT+6        ;'UP' CONTROL
        STA A   A.APRT+7        ;'DN' CONTROL
        RTS
        ;
        ;
        ;       INTERRUPT CONTROL LOADERS
```

```
        ;
A.AHUP: LDA B   A.APRT+2        ;'UP' CONTROL
        STA B   CRAH+A.6821
        RTS
        ;
A.ALUP: LDA B   A.APRT+6        ;'UP' CONTROL
        STA B   CRAL+A.6821
        RTS
        ;
A.AHDN: LDA B   A.APRT+3        ;'DN' CONTROL
        STA B   CRAH+A.6821
        RTS
        ;
A.ALDN: LDA B   A.APRT+7        ;'DN' CONTROL
        STA B   CRAL+A.6821
        RTS
        ;
        .PAGE
        .SBTTL  A.6821 B-PORT CONTROL ROUTINES
        ;
        ;       DATA DIRECTION LOAD ROUTINES
        ;       ENTER WITH DIRECTION CODE IN <B>
        ;
A.BHDR: LDA A   CRBH+A.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRBH+A.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRBH+A.6821    ;SET DIRECTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRBH+A.6821     ;AND RESTORE
        RTS
        ;
A.BLDR: LDA A   CRBL+A.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRBL+A.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRBL+A.6821    ;SET DIRCTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRBL+A.6821     ;AND RESTORE
        RTS
        ;
        .PAGE
        ;       CONTROL AND SERVICE VECTOR LOADER
        ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
        ;       'UP' CONTROL IN <B>, AND
        ;       'DN' CONTROL IN <A>
        ;
A.BHDF: STX     A.BPRT          ;SAVE SERVICE ROUTINE ADDRESS
        STA B   A.BPRT+2        ;'UP' CONTROL
        STA A   A.BPRT+3        ;'DN' CONTROL
        RTS
        ;
A.BLDF: STX     A.BPRT+4        ;SAVE SERVICE ROUTINE ADDRESS
        STA B   A.BPRT+6        ;'UP' CONTROL
        STA A   A.BPRT+7        ;'DN' CONTROL
        RTS
        ;
        ;
        ;       INTERRUPT CONTROL LOADERS
        ;
A.BHUP: LDA B   A.BPRT+2        ;'UP' CONTROL
        STA B   CRBH+A.6821
        RTS
        ;
A.BLUP: LDA B   A.BPRT+6        ;'UP' CONTROL
        STA B   CRBL+A.6821
        RTS
        ;
A.BHDN: LDA B   A.BPRT+3        ;'DN' CONTROL
        STA B   CRBH+A.6821
        RTS
        ;
A.BLDN: LDA B   A.BPRT+7        ;'DN' CONTROL
        STA B   CRBL+A.6821
        RTS
        ;
```

```
        .PAGE
        .SBTTL  A.6821 A-PORT TRANSFERS
        ;
        ;       BYTE DATA IS TRANSFERED IN <B>
        ;       WORD DATA IS TRANSFERED IN <B,A>
        ;
A.AHRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRAH+A.6821     ;GET BYTE
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRAH+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
A.ALRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRAL+A.6821     ;GET BYTE
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRG FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
A.AWRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRAH+A.6821     ;GET HIGH BYTE
        LDA A   PRAL+A.6821     ;AND LOW BYTE
        SEC                     ;HAVE WORD
        RTS
2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
        .PAGE
        ;
A.AHWT: BRA     2$              ;TEST ENTRY
1$:     STA B   PRAH+A.6821     ;STORE BYTE
        TST     PRAH+A.6821     ;CLEAR IRQ FLAGS
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRAH+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
A.ALWT: BRA     2$              ;TEST ENTRY
1$:     STA B   PRAL+A.6821     ;STORE BYTE
        TST     PRAL+A.6821     ;CLEAR IRQ FLAGS
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
A.AWWT: BRA     2$              ;TEST ENTRY
1$:     STA B   PRAH+A.6821     ;SEND HIGH BYTE
        STA A   PRAL+A.6821     ;SEND LOW BYTE
        TST     PRAH+A.6821     ;CLEAR IRQ FLAGS
        TST     PRAL+A.6821     ;CLEAR IRQ FLAGS
        SEC                     ;WORD SENT
        RTS
2$:     PSH A                   ;SAVE A
        LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        PUL A                   ;RESTORE A
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
```

```
        .PAGE
        .SBTTL  A.6821 B-PORT TRANSFERS
        ;
        ;       BYTE DATA IS TRANSFERED IN <B>
        ;       WORD DATA IS TRANSFERED IN <B,A>
        ;
A.BHRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRBH+A.6821     ;GET BYTE
        STA B   PRBH+A.6821     ;DO CONTROL
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRBH+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
A.BLRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRBL+A.6821     ;GET BYTE
        STA B   PRBL+A.6821     ;DO CONTROL
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRG FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
A.BWRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRBH+A.6821     ;GET HIGH BYTE
        LDA A   PRBL+A.6821     ;AND LOW BYTE
        STA B   PRBH+A.6821     ;DO CONTROL
        STA A   PRBL+A.6821     ;DO CONTROL
        SEC                     ;HAVE WORD
        RTS
2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
        .PAGE
        ;
A.BHWT: BRA     2$              ;TEST ENTRY
1$:     TST     PRBH+A.6821     ;CLEAR IRQ FLAGS
        STA B   PRBH+A.6821     ;STORE BYTE
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRBH+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
A.BLWT: BRA     2$              ;TEST ENTRY
1$:     TST     PRBL+A.6821     ;CLEAR IRQ FLAGS
        STA B   PRBL+A.6821     ;STORE BYTE
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
A.BWWT: BRA     2$              ;TEST ENTRY
1$:     TST     PRBH+A.6821     ;CLEAR IRQ FLAGS
        TST     PRBL+A.6821     ;CLEAR IRQ FLAGS
        STA B   PRBH+A.6821     ;SEND HIGH BYTE
        STA A   PRBL+A.6821     ;SEND LOW BYTE
        SEC                     ;WORD SENT
        RTS
2$:     PSH A                   ;SAVE A
        LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        PUL A                   ;RESTORE A
```

```
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
        .PAGE
        .SBTTL  A.6821 INTERRUPT SERVICE DISPATCH ROUTINES
        ;
A.AH$I: LDX     A.APRT          ;GET SERVICE ADDRESS
        BNE     A.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRAH+A.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     A.IRQD          ;TERMINATE INTERRUPTS
        ;
A.AL$I: LDX     A.APRT+4        ;GET SERVICE ADDRESS
        BNE     A.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRAL+A.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     A.IRQD          ;TERMINATE INTERRUPTS
        ;
A.BH$I: LDX     A.BPRT          ;GET SERVICE ADDRESS
        BNE     A.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRBH+A.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     A.IRQD          ;TERMINATE INTERRUPTS
        ;
A.BL$I: LDX     A.BPRT+4        ;GET SERVICE ADDRESS
        BNE     A.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRBL+A.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     A.IRQD          ;TERMINATE INTERRUPTS
        ;
A.IRQG: JSR     0,X             ;DO SERVICE ROUTINE
        RTI
        ;
A.IRQD: LDA A   0,X             ;GET CONTROL REGISTER
        AND B   #,376           ;CLEAR CX1 INTERRUPT ENABLE
        BIT B   #,40            ;CX2 INTERRUPT MODE ?
        BNE     1$              ;BRANCH IF NOT
        AND B   #,367           ;CLEAR CX2 INTERRUPT ENABLE
1$:     STA B   0,X             ;SET NEW CONTROL
        RTI
        ;
        .IIF    NDF,$A.6821     .END
```

```
        .PAGE
        .SBTTL  A.8250 ASYNCHRONOUS INTERFACE SOFTWARE
        ;
        ;         THIS SOFTWARE PACKAGE SUPPORTS THE INS8250
        ;       ASYNCHRONOUS COMMUNICATIONS ELEMENT.
        ;       ENTRY POINTS ARE DEFINED HERE FOR EXTERNAL
        ;       ACCESS (VIA JSR ____):
        ;
        ;               A.MOSV --> SAVE 'X' AS 'MODEM' INTERRUPT
        ;                          SERVICE ROUTINE ADDRESS
        ;               A.XMSV --> SAVE 'X' AS 'XMTR' INTERRUPT
        ;                          SERVICE ROUTINE ADDRESS
        ;               A.RCSV --> SAVE 'X' AS 'RCVR' INTERRUPT
        ;                          SERVICE ROUTINE ADDRESS
        ;               A.LNSV --> SAVE 'X' AS 'LINE' INTERRUPT
        ;                          SERVICE ROUTINE ADDRESS
        ;
        ;               A.MOUP --> ENABLE 'MODEM' INTERRUPT
        ;               A.XMUP --> ENABLE 'XMTR' INTERRUPT
        ;               A.RCUP --> ENABLE 'RCVR' INTERRUPT
        ;               A.LNUP --> ENABLE 'LINE' INTERRUPT
        ;
        ;               A.MODN --> DISABLE 'MODEM' INTERRUPT
        ;               A.XMDN --> DISABLE 'XMTR' INTERRUPT
        ;               A.RCDN --> DISABLE 'RCVR' INTERRUPT
        ;               A.LNDN --> DISABLE 'LINE' INTERRUPT
        ;
        ;
        ;         STATUS REGISTER ACCESS TO/FROM <B>
        ;
        ;       A.MSRR --> READ MODEM STATUS REGISTER
        ;       A.MSRW --> WRITE MODEM STATUS REGISTER
        ;
        ;       A.MCRR --> READ MODEM CONTROL REGISTER
        ;       A.MCRW --> WRITE MODEM CONTROL REGISTER
        ;
        ;       A.LSRR --> READ LINE STATUS REGISTER
        ;       A.LSRW --> WRITE LINE STATUS REGISTER
        ;
        ;       A.LCRR --> READ LINE CONTROL REGISTER
        ;       A.LCRW --> WRITE LINE CONTROL REGISTER
        ;
        ;       A.DLLR --> READ DLL
        ;       A.DLLW --> WRITE DLL
        ;
        ;       A.DLMR --> READ DLM
        ;       A.DLMW --> WRITE DLM
        ;
        ;
        .PAGE
        ;
        ;         THE FOLLOWING ENTRY POINT IS VIA INTERRUPT
        ;       AND PROVIDES 8250 ACE SERVICES FOR 'MODEM', 'XMTR',
        ;       'RCVR', AND 'LINE' INTERRUPTS.
        ;
        ;               A.82$I --> MPU INTERRUPT ENTRY POINT
        ;                          FOR 8250 ACE
        ;
        ;
        ;         NOTES ON INTERRUPT SERVICE ROUTINES
        ;
        ;       MODEM --> READS MODEM STATUS REGISTER (MSR) INTO <B> AND
        ;                   IF 'A.MO$V' IS NOT ZERO -
        ;               THEN - CALLS ROUTINE VIA JSR 0,X(=A.MO$V)
        ;               ELSE - NOTHING
        ;
        ;       XMTR --> IF 'A.XM$V' IS NOT ZERO -
        ;               THEN - CALLS ROUTINE VIA JSR 0,X(=A.XM$V)
        ;                       ON RETURN
        ;                       IF C=1 --> TRANSMITS CHARACTER IN <B>
        ;                       IF C=0 --> DISABLES 'XMTR' INTERRUPT
        ;               ELSE - DISABLES 'XMTR' INTERRUPT
        ;
        ;       RCVR --> MOVE INPUT CHARACTER INTO <B>
        ;               IF 'A.RC$V' IS NOT ZERO -
        ;               THEN - CALLS ROUTINE VIA JSR 0,X(=A.RC$V)
```

```
;               ELSE - IGNORES INPUT CHARACTER
;
;       LINE --> READS LINE STATUS REGISTER (LSR) INTO <B> AND
;                IF 'A.LN$V' IS NOT ZERO -
;               THEN - CALLS ROUTINE VIA JSR 0,X(=A.LN$V)
;               ELSE - NOTHING
;
.PAGE
.SBTTL  A.8250 ASYNCHRONOUS HARDWARE
;
; THE ASYNCHRONOUS COMMUNICATION ELEMENT USED BY THE
;PORT IS THE NATIONAL SEMICONDUCTOR TYPE INS8250.
; THE FOLLOWING IS A SUMMARY OF THE REGISTERS AND
;SOME DEFINITIONS.
;
;       DLAB (DATA LATCH ACCES BIT) = 0
;
; REGISTER 0
RBR     =0      ;RECEIVER BUFFER REGISTER (READ ONLY)
THR     =0      ;TRANSMITTER HOLDING REGISTER (WRITE ONLY)
;
; REGISTER 1
IER     =1      ;INTERRUPT ENABLE REGISTER (READ/WRITE)
;
; REGISTER 2
IIR     =2      ;INTERRUPT IDENTIFICATION REGISTER
;
; REGISTER 3
LCR     =3      ;LINE CONTROL REGISTER (READ/WRITE)
;
; REGISTER 4
MCR     =4      ;MODEM CONTROL REGISTER (READ/WRITE)
;
; REGISTER 5
LSR     =5      ;LINE CONTROL STATUS REGISTER (READ/WRITE)
;
; REGISTER 6
MSR     =6      ;MODEM STATUS REGISTER
;
;
;       DLAB (DATA LATCH ACCESS BIT) = 1
;
; REGISTER 0
DLL     =0      ;DIVISOR LATCH LOW BYTE (READ/WRITE)
;
; REGISTER 1
DLM     =1      ;DIVISOR LATCH HIGH BYTE (READ/WRITE)
;
.PAGE
.SBTTL  A.8250 DETAILS
;
; RECEIVER BUFFER REGISTER (RBR) - 8 BIT READ ONLY REGISTER
;               CONTAINING DATA RECEIVED
;
; TRANSMITTER HOLDING REGISTER (THR) - 8 BIT WRITE ONLY REGISTER
;               CONTAINING CHARACTER TO BE TRANSMITTED
;
; INTERRUPT ENABLE REGISTER (IER)
; BIT 0         ENABLE RECEIVED DATA AVAILABLE INTERRUPT
;     1         ENABLE TRANSMITTER HOLDING REGISTER EMPTY INTERRUPT
;     2         ENABLE RECEIVER LINE STATUS INTERRUPT
;     3         ENABLE MODEM STATUS INTERRUPT
;     4-7       0'S
;
; INTERRUPT IDENTIFICATION REGISTER (IIR)
; BIT 0         '0' IF INTERRUPT PENDING
; BIT 1 2
;     _ _
;     0 0       MODEM STATUS INTERRUPT (LOWEST PRIORITY)
;     1 0       TRANSMITTER HOLDING REGISTER EMPTY
;     0 1       RECEIVER DATA AVAILABLE
;     1 1       RECEIVER LINE STATUS (HIGHEST PRIORITY)
;
; LINE CONTROL REGISTER (LCR)
; BIT 0 1       WORD LENGTH SELECT BITS
;     _ _
```

```
;     0 0       5 BITS
;     1 0       6 BITS
;     0 1       7 BITS
;     1 0       8 BITS
;
; BIT 2         STOP BIT SELECT
;
;     _
;     0         1 STOP BIT
;     1         1.5 STOP BITS (5 BITS) / 2 STOP BITS
;
; BIT 3 4 5     PARITY SELECT
;
;     _ _ _
;     0 X X     NO PARITY / NO STICK
;     1 0 0     ODD PARITY
;     1 1 0     EVEN PARITY
;     1 0 1     PARITY = STICK 1
;     1 1 1     PARITY = STICK 0
;
; BIT 6         BREAK CONTROL BIT
;
;     _
;     0         NORMAL OPERATION
;     1         SERIAL OUTPUT FORCED TO SPACING
;
; BIT 7         DIVISOR LATCH ACCESS BIT
;
;     _
;     0         ACCESS RECEIVER BUFFER / TRANSMITTER HOLDING REGISTERS
;     1         ACCESS DIVISOR LATCH REGISTERS
;
.PAGE
;
; MODEM CONTROL REGISTER (MCR)
; BIT 0         DATA TERMINAL READY CONTROL (DSR OUTPUT)
; BIT 1         REQUEST TO SEND CONTROL (CTS OUTPUT)
; BIT 2         OUT 1 CONTROL
; BIT 3         OUT 2 CONTROL
; BIT 4         LOOP CONTROL
;               ENABLES LOOP BACK MODE AND CONNECTS THE FOLLOWING -
;               DTR >> DSR
;               RTS >> CTS
;               OUT1 >> RI
;               OUT2 >> RLSD
;               SO >> SI
; BIT 5-7       0'S
;
; LINE STATUS REGISTER (LSR)
; BIT 0         DATA READY
; BIT 1         OVERRUN ERROR
; BIT 2         PARITY ERROR
; BIT 3         FRAMING ERROR
; BIT 4         BREAK INTERRUPT
; BIT 5         TRANSMITTER HOLDING REGISTER EMPTY
; BIT 6         TRANSMITTER SHIFT REGISTER EMPTY
; BIT 7         0
;
; MODEM STATUS REGISTER (MSR)
; BIT 0         DELTA CLEAR TO SEND
; BIT 1         DELTA DATA SET READY
; BIT 2         TRAILING EDGE RING INDICATOR
; BIT 3         DELTA LINE SIGNAL DETECT
; BIT 4         CLEAR TO SEND (RTS INPUT)
; BIT 5         DATA SET READY (DTR INPUT)
; BIT 6         RING INDICATOR
; BIT 7         RECEIVED LINE SIGNAL DETECT (SIN INPUT)
;
; DIVISOR LATCH (DLL/DLM)
; 16-BIT DIVISOR FOR BAUD RATE SELECTION
; BAUD  COUNT  TIME     @ 1.8432 MHZ
; ____  _____  ____
;  110   1047  9091
;  150    768  6667
;  300    384  3333
;  600    192  1667
; 1200     96   833
; 1800     64   556
; 2400     48   417
; 4800     24   208
```

```
        ; 9600     12   104
        ;
        .PAGE
        .SBTTL  A.8250 TYPICAL RS-232C WIRING CONNECTIONS
        ;
        ;
        ;       1       GROUND
        ;       2       RX      (INPUT)
        ;       3       TX      (OUTPUT)
        ;       4       RTS     (INPUT)
        ;       5       CTS     (OUTPUT)
        ;       6       DSR     (OUTPUT)
        ;       7       GROUND
        ;       8       OUT1    (OUTPUT)
        ;       9       +12 VOLTS SOURCE
        ;       10      -12 VOLTS SOURCE
        ;       11
        ;       12
        ;       13
        ;       14
        ;       15
        ;       16
        ;       17
        ;       18      RLSD    (INPUT)
        ;       19      RI      (INPUT)
        ;       20      DTR     (INPUT)
        ;       21
        ;       22      OUT2    (OUTPUT)
        ;       23
        ;       24
        ;       25
        ;
        ;       JUMPER RSLD(18) TO RX(2) FOR AUTO BAUD-RATE
        ;
        .PAGE
        .SBTTL  A.8250 VARIABLES AND POINTERS
        ;
        ;       PORT STATUS AND BUFFERS
        ;
        .IF     NDF,$A.8250
        A.8250  =       220     ;SERIAL PORT DEFAULT ADDRESS
        VARSAV  =       0       ;DEFAULT VARIABLES ADDRESS
        PGMSAV  =       100000  ;DEFAULT PROGRAM ADDRESS
        .=PGMSAV
        .ENDC
        ;
        PGMSAV=.
        .=VARSAV
        ;
A.MO$V: .BYTE   0,0     ;MODEM DISPATCH VECTOR
A.XM$V: .BYTE   0,0     ;XMTR DISPATCH VECTOR
A.RC$V: .BYTE   0,0     ;RCVR DISPATCH VECTOR
A.LN$V: .BYTE   0,0     ;LINE DISPATCH VECTOR
        ;
A.LNST: .BYTE   0       ;LINE STATUS FLAG
        ;               = 0     NO PROCESS PENDING
        ;               =-1     REQUIRES BAUD RATE PROCESSING
A.ERCT: .BYTE   0       ;LINE ERROR COUNTER
A.LNBL: .BYTE   0       ;'LINE' SERVICE ENABLES
A.MNBL: .BYTE   0       ;'MODEM' SERVICE ENABLES
        ;
A.PBCT: .BYTE   0,0     ;PREVIOUS BAUD COUNT
A.CBCT: .BYTE   0,0     ;CURRENT BAUD COUNT
A.BDIF: .BYTE   0,0     ;COMPUTED DIFFERENCE
A.LWCT: .BYTE   0,0     ;LOWEST DIFFERENCE FOUND
A.BDVS: .BYTE   0,0     ;BAUD RATE DIVISOR
        ;
        VARSAV=.
        .=PGMSAV
        ;
        .PAGE
        .SBTTL  A.8250 INTERRUPT CONTROL ROUTINES
        ;
A.MOSV: STX     A.MO$V          ;SAVE 'MODEM' VECTOR ADDRESS
        RTS
        ;
```

```
A.MOUP: LDA A   IER+A.8250      ;ENABLE MODEM INTERRUPT
        ORA A   #,10
        STA A   IER+A.8250
        RTS
        ;
A.MODN: LDA A   IER+A.8250      ;DISABLE INTERRUPT
        AND A   #,7
        STA A   IER+A.8250
        RTS
        ;
A.XMSV: STX     A.XM$V          ;SAVE 'XMTR' VECTOR ADDRESS
        RTS
        ;
A.XMUP: LDA A   IER+A.8250      ;ENABLE XMTR INTERRUPT
        ORA A   #,2
        STA A   IER+A.8250
        RTS
        ;
A.XMDN: LDA A   IER+A.8250      ;DISABLE INTERRUPT
        AND A   #,15
        STA A   IER+A.8250
        RTS
        ;
        .PAGE
        ;
A.RCSV: STX     A.RC$V          ;SAVE 'RCVR' VECTOR ADDRESS
        RTS
        ;
A.RCUP: LDA A   IER+A.8250      ;ENABLE RCVR INTERRUPT
        ORA A   #,1
        STA A   IER+A.8250
        RTS
        ;
A.RCDN: LDA A   IER+A.8250      ;DISABLE INTERRUPT
        AND A   #,16
        STA A   IER+A.8250
        RTS
        ;
A.LNSV: STX     A.LN$V          ;SAVE 'LINE' INTERRUPT SERVICE
        RTS
        ;
A.LNUP: LDA A   IER+A.8250      ;AND ENABLE LINE INTERRUPT
        ORA A   #,4
        STA A   IER+A.8250
        RTS
        ;
A.LNDN: LDA A   IER+A.8250      ;DISABLE INTERRUPT
        AND A   #,13
        STA A   IER+A.8250
        RTS
        ;
        .PAGE
        .SBTTL  A.8250 STATUS REGISTER ACCESS
        ;
        ;         ACCESS TO THE 8250 STATUS REGISTERS
        ;
        ;       MCR IS THE OUTPUT REGISTER
        ;               DATA FROM <B>
        ;
        ;       MSR IS THE INPUT REGISTER
        ;               DATA PLACED IN <B>
        ;
        ;
A.MSRR: LDA B   MSR+A.8250      ;GET MODEM STATUS
        RTS
        ;
A.MSRW: STA B   MSR+A.8250      ;WRITE TO STATUS
        RTS
        ;
A.MCRR: LDA B   MCR+A.8250      ;READ CONTROL
        RTS
        ;
A.MCRW: STA B   MCR+A.8250      ;STORE CONTROL
        RTS
        ;
A.LSRR: LDA B   LSR+A.8250      ;READ STATUS
```

```
        RTS
        ;
A.LSRW: STA B   LSR+A.8250      ;WRITE TO STATUS
        RTS
        ;
A.LCRR: LDA B   LCR+A.8250      ;READ CONTROL
        RTS
        ;
A.LCRW: STA B   LCR+A.8250      ;WRITE CONTROL
        RTS
        ;
A.DLLR: LDA A   LCR+A.8250      ;SET CONTROL TO ACCESS DLL
        PSH A                   ;SAVE
        ORA A   #,200
        STA A   LCR+A.8250
        LDA B   DLL+A.8250      ;READ DLL
        PUL A                   ;RESTORE LCR
        STA A   LCR+A.8250
        RTS
        ;
A.DLLW: LDA A   LCR+A.8250      ;SET CONTROL TO ACCESS DLL
        PSH A                   ;SAVE
        ORA A   #,200
        STA A   LCR+A.8250
        STA B   DLL+A.8250      ;WRITE DLL
        PUL A                   ;RESTORE LCR
        STA A   LCR+A.8250
        RTS
        ;
A.DLMR: LDA A   LCR+A.8250      ;SET CONTROL TO ACCESS DLM
        PSH A                   ;SAVE
        ORA A   #,200
        STA A   LCR+A.8250
        LDA B   DLM+A.8250      ;READ DLM
        PUL A                   ;RESTORE LCR
        STA A   LCR+A.8250
        RTS
        ;
A.DLMW: LDA A   LCR+A.8250      ;SET CONTROL TO ACCESS DLM
        PSH A                   ;SAVE
        ORA A   #,200
        STA A   LCR+A.8250
        STA B   DLM+A.8250      ;WRITE DLM
        PUL A                   ;RESTORE LCR
        STA A   LCR+A.8250
        RTS
        ;
        .PAGE
        .SBTTL  A.8250 INTERRUPT HANDLER
        ;
A.82$I: LDA A   IIR+A.8250      ;GET INTERRUPT ID
        BIT A   #,1             ;INTERRUPT PENDING ?
        BNE     1$              ;IF NOT - LEAVE
        TAB                     ;COMPUTE JUMP TABLE ENTRY ADDRESS
        ASR B
        ABA
        CLR B
        ADD A   #,2$&377
        ADC B   #,2$&177400/400
        PSH A
        PSH B
        RTS                     ;GO TO JUMP TABLE
1$:     RTI                     ;RETURN FROM INTERRUPT
        ;
2$:     JMP     A.MODE          ;MODEM STATUS INTERRUPT
        JMP     A.XMTR          ;TRANSMITTER INTERRUPT
        JMP     A.RCVR          ;RECEIVER INTERRUPT
        JMP     A.LINE          ;LINE STATUS INTERRUPT
        ;
        .PAGE
        .SBTTL  A.8250 XMTR & RCVR INTERRUPT DISPATCHERS
        ;
A.XMTR: LDA B   LSR+A.8250      ;NEED A CHARACTER ?
        BIT B   #,40
        BEQ     2$              ;BRANCH IF NOT
        LDX     A.XM$V          ;XMTR VECTOR
```

```
        BEQ     1$              ;BETTER HAVE AN ADDRESS !
        JSR     0,X             ;GO TO ROUTINE
        BCC     1$              ;IF NO CHARACTERS (C=0) THEN TERMINATE
        STA B   THR+A.8250      ;ELSE SEND NEW CHARACTER
        JMP     A.82$I          ;GO CHECK FOR OTHER INTERRUPTS
1$:     JSR     A.XMDN          ;KILL TRANSMITTER
2$:     JMP     A.82$I          ;AND CHECK FOR OTHER INTERRUPTS
        ;
A.RCVR: LDA B   LSR+A.8250      ;HAVE A CHARACTER ?
        BIT B   #,1
        BEQ     1$              ;BRANCH IF NOT
        LDA B   RBR+A.8250      ;PLACE CHARACTER IN <B>
        LDX     A.RC$V          ;RCVR VECTOR
        BEQ     1$              ;BETTER HAVE AN ADDRESS !
        JSR     0,X             ;GO TO ROUTINE
1$:     JMP     A.82$I          ;GO CHECK FOR OTHER INTERRUPTS
        ;
        .PAGE
        .SBTTL  A.8250 LINE AND MODEM INTERRUPT DISPATCHERS
        ;
A.LINE: LDA B   LSR+A.8250      ;GET STATUS, CLEAR FLAGS
        BIT B   #,36            ;SOMETHING TO CHECK ?
        BEQ     1$              ;BRANCH IF NOT
        LDX     A.LN$V          ;EXTERNAL VECTOR ?
        BEQ     1$              ;BR IF NOT
        JSR     0,X             ;GO TO ROUTINE
1$:     JMP     A.82$I          ;CHECK FOR OTHER INTERRUPTS
        ;
A.MODE: LDA B   MSR+A.8250      ;GET MODEM STATUS, CLEAR FLAGS
        BIT B   #,17            ;SOMETHING TO CHECK ?
        BEQ     1$              ;BRANCH IF NOT
        LDX     A.MO$V          ;EXTERNAL VECTOR ?
        BEQ     1$              ;BR IF NOT
        JSR     0,X             ;GO TO ROUTINE
1$:     JMP     A.82$I          ;CHECK FOR OTHER INTERRUPTS
        ;
        .PAGE
        .SBTTL  A.8250 STANDARD LINE / MODEM SETTUP
        ;
        ;         USING ROUTINES A.82$L AND A.82$M
        ;       THE ASYNCHRONOUS INTERFACE IS PROGRAMMED
        ;       INITIALLY FOR -
        ;               1 START BIT
        ;               7 DATA BITS
        ;               1 ODD PARITY BIT
        ;               2 STOP BITS
        ;
        ;       ON A 'BREAK' INTERRUPT THE I/O
        ;       IS RESET TO THE ABOVE PARAMETERS
        ;       AND THE BAUD RATE IS DETERMINED DURING THE
        ;       NEXT CHARACTER
        ;
        ;       ON A 'PARITY' INTERRUPT THE I/O
        ;       IS SEQUENCED TO THE NEXT POSSIBLE PARITY
        ;       CONFIGURATION - THESE ARE:
        ;               ODD PARITY
        ;               EVEN PARITY
        ;               STICK '1' PARITY
        ;               STICK '0' PARITY
        ;
        ;       ON A 'FRAMING' INTERRUPT THE I/O
        ;       IS SEQUENCED TO THE NEXT STOP BIT
        ;       CONFIGURATION - 1 OR 2 STOPS
        ;
        ;       IF MORE THAN 20. ERRORS OCCUR THEN A
        ;       'BREAK' INTERRUPT SEQUENCE IS PERFORMED
        ;       TO INITIALIZE THE HARDWARE
        ;
        ;
        ;
        ;         WITH AN EXTERNAL DEVICE ATTACHED TO THE PORT -
        ;       SELECT ONE OF THE FOLLOWING BAUD RATES -
        ;
        ;       110, 150, 300, 600, 1200, 1800, 2400, 4800, OR 9600
        ;
        ;       THE WORD LENGTH MUST BE 7 DATA BITS,
```

```
        ;       ANY PARITY MODE - ODD, EVEN, STICK '1' OR '0',
        ;       AND 1 OR 2 STOP BITS.
        ;
        ;       THE PORT MATCHES YOUR DEVICE'S
        ;       BAUD RATE, PARITY, AND STOPS WHEN THE
        ;       DEVICE TRANSMITS A 'BREAK' FOLLOWED
        ;       BY SEVERAL 'SPACE' CHARACTERS
        ;
        ;       THE PORT IS NOW CONFIGURED FOR YOUR DEVICE.
        ;
        .PAGE
        .SBTTL  A.8250 BAUD RATE TABLE
        ;
A.BTBL: FDB     8.,12.          ;9600 BAUD
        FDB     15.,24.         ;4800 BAUD
        FDB     25.,48.         ;2400 BAUD
        FDB     35.,64.         ;1800 BAUD
        FDB     66.,96.         ;1200 BAUD
        FDB     132.,192.       ; 600 BAUD
        FDB     232.,384.       ; 300 BAUD
        FDB     402.,768.       ; 150 BAUD
        FDB     32767.,1047.    ; 110 BAUD
        ;
        .PAGE
        .SBTTL  A.8250 LINE STATUS HANDLER
        ;
        ;         THIS IS THE STANDARD LINE STATUS ROUTINE
        ;       AND MAY BE INVOKED BY SETTING A.LN$V=A.82$L.
        ;         EACH LINE INTERRUPT FLAG IS COMPARED TO THOSE
        ;       ENABLED CHECKS DEFINED IN A.LNBL (BIT FOR BIT).
        ;         ENTER WITH LSR IN <B>, B IS DESTROYED
        ;
        ;
A.82$L: AND B   A.LNBL          ;MASK WITH ENABLES
        BIT B   #,20            ;A 'BREAK' ?
        BEQ     A.PRTY          ;IF NOT - CHECK PARITY
A.LBRK: LDA B   #,16            ;RESET HARDWARE
        STA B   LCR+A.8250
        LDA B   #,-1            ;SET UP A.LNST
        STA B   A.LNST          ;TO CHECK BAUD RATE
        LDA B   #,21.           ;ERROR COUNT PRESET
        STA B   A.ERCT          ;SAVE IN STATUS TABLE
        RTS
        ;
A.PRTY: BIT B   #,4             ;A 'PARITY' ?
        BEQ     A.FRAM          ;IF NOT - CHECK FRAME
        LDA A   LCR+A.8250      ;GET LINE CONTROL REGISTER
        TAB                     ;SAVE
        AND A   #,317           ;MASK OUT PARITY
        ADD B   #,20            ;UPDATE PARITY SELECT
        AND B   #,60            ;MASK OUT ALL BUT PARITY
        ABA                     ;COMBINE BITS
        STA A   LCR+A.8250      ;STORE AT I/O
        BRA     A.LERR          ;GO FINISH UP
        ;
A.FRAM: BIT B   #,10            ;A 'FRAME' ?
        BEQ     A.OVRN          ;IF NOT - AN OVERRUN ERROR
        LDA B   LCR+A.8250      ;GET CONTROL REGISTER
        EOR B   #,4             ;FLIP/FLOP STOP BIT SELECT
        STA B   LCR+A.8250      ;SAVE AT I/O
A.LERR: DEC     A.ERCT          ;ZERO YET ?
        BLE     A.LBRK          ;IF SO - RESET HARDWARE
A.OVRN:
A.LEND: RTS
        ;
        .PAGE
        .SBTTL  A.8250 MODEM STATUS HANDLER
        ;
        ;         THIS IS THE STANDARD A.8250 MODEM STATUS
        ;       HANDLER. IT MAY BE ACCESSED BY SETTING A.MO$V=A.82$M.
        ;         ONLY THE RLSD BIT IS USED BY THIS ROUTINE
        ;       FOR BAUD-RATE DETERMINATION. AND IS ENABLED
        ;       BY SETTING THE CORRESPONDING BIT IN A.MNBL.
        ;         ENTER WITH MSR IN <B>, B IS DESTROYED
        ;
        ;
```

```
A.MEND: RTS
A.82$M: TBA                     ;COPY B TO A
        AND B   A.MNBL          ;MASK WITH ENABLES
        BIT B   #,10            ;BAUD RATE ENABLED ?
        BEQ     A.MEND          ;EXIT IF NOT
        BIT A   #,200           ;FIRST BIT OF CHARACTER ?
        BEQ     A.MEND          ;IF NOT - LEAVE
        LDA B   A.LNST          ;A BAUD RATE CHECK ?
        BEQ     A.MEND          ;IF NOT - LEAVE
        CLR     A.LNST          ;CLEAR STATUS
        LDX     #,-6000.        ;INITIALIZE COUNTER TO
        STX     A.CBCT          ;ABOUT 12 BIT TIMES @ 110 BAUD
        LDX     #,32767.        ;PRESET LOW COUNT
        STX     A.LWCT
1$:     LDA A   MSR+A.8250      ;CHECK STATUS FOR TRANSITION
        BIT A   #,10
        BNE     3$              ;ON TRANSITION - SKIP
        INC     A.CBCT+1        ;UPDATE COUNTER
        BNE     1$              ;LOOP UNTIL = 0
        INC     A.CBCT
        BNE     1$
2$:     BRA     A.LBRK          ;ELSE TIME RAN OUT
        ;
3$:     LDX     A.CBCT          ;GET CURRENT COUNT
        STX     A.PBCT          ;SAVE IN PREVIOUS COUNT
        ;
4$:     LDA A   MSR+A.8250      ;GET STATUS
        BIT A   #,10            ;CHECK FOR TRANSITION
        BEQ     6$              ;IF NONE - SKIP
        LDA A   A.CBCT+1        ;GET CURRENT COUNT
        LDA B   A.CBCT
        SUB A   A.PBCT+1        ;COMPUTE DIFFERENCE
        SBC B   A.PBCT          ;WITH PREVIOUS COUNT
        STA A   A.BDIF+1        ;SAVE DIFFERENCE
        STA B   A.BDIF
        SUB A   A.LWCT+1        ;DIFFERENCE LESS THAN
        SBC B   A.LWCT          ;CURRENT LOW COUNT ?
        BPL     5$              ;IF NOT - SKIP
        LDX     A.BDIF          ;ELSE SAVE AS NEW LOW COUNT
        STX     A.LWCT
5$:     LDA A   A.CBCT+1        ;UPDATE AND SAVE COUNT
        LDA B   A.CBCT
        STA A   A.PBCT+1
        STA B   A.PBCT
        ADD A   #,3
        ADC B   #,0
        STA A   A.CBCT+1
        STA B   A.CBCT
6$:     INC     A.CBCT+1        ;UPDATE COUNT
        BNE     4$              ;LOOP UNTIL TIME OUT
        INC     A.CBCT
        BNE     4$
        ;
        ;       SCAN BAUD TABLE TO GET BAUD RATE DIVISOR
        ;
        LDX     #,32767.        ;CHECK A.LWCT
        CPX     A.LWCT          ;THE SAME ?
        BEQ     2$              ;IF SO - NO TRANSITION FOUND
        ;
        LDX     #,A.BTBL        ;GET POINTER TO TABLE
7$:     LDA A   A.LWCT+1        ;GET A.LWCT
        LDA B   A.LWCT
        SUB A   1,X             ;CHECK IF LESS THAN TABLE ELEMENT
        SBC B   0,X
        BMI     8$              ;IF SO - SKIP
        INX                     ;ELSE SCAN TABLE
        INX
        INX
        INX
        BRA     7$
        ;
8$:     LDX     2,X             ;GET BAUD RATE DIVISOR
        STX     A.BDVS          ;SAVE DIVISOR
        LDA A   LCR+A.8250      ;GET LINE CONTROL
        ORA A   #,200           ;SET DLAB=1
        STA A   LCR+A.8250
```

```
        LDA B   A.BDVS+1        ;SET BAUD RATE DIVISOR AT I/O
        STA B   DLL+A.8250
        LDA B   A.BDVS
        STA B   DLM+A.8250
        AND A   #,177           ;SET DLAB=0
        STA A   LCR+A.8250
        RTS
        ;
        .PAGE
        .SBTTL  A.8250 INITIALIZATION ROUTINE
        ;
A.82$B: LDA B   #,36            ;SET UP A.LNBL
        STA B   A.LNBL
        LDA B   #,17            ;SET UP A.MNBL
        STA B   A.MNBL
        LDA B   #,-1            ;SET UP A.LNST
        STA B   A.LNST          ;TO CHECK BAUD RATE
        LDA B   #,11            ;ERROR COUNT PRESET
        STA B   A.ERCT          ;IN STATUS TABLE
        LDA B   #,216           ;LINE CONTROL
        STA B   LCR+A.8250
        LDA B   #,12.&377       ;SET UP FOR 9600 BAUD
        STA B   DLL+A.8250
        LDA B   #,12.&177400/400
        STA B   DLM+A.8250
        LDA B   #,16
        STA B   LCR+A.8250
        RTS
        ;
        .IIF    NDF,$A.8250     .END
```

```
        .PAGE
        .SBTTL  THE END
        ;
        .END
```

SERIAL 'PORT' MACRO V05.06 Thursday 18-Oct-07 19:15

Table of contents

```
      1                                 .TITLE  SERIAL 'PORT'
      2                                 .SBTTL  SERIAL PORT AS SEEN ON THE ARB-11 BUS
      3                                 ;
      4                                 ;[X XXX XXX XXX XXN NNN]  00    KEYBOARD BUFFER STATUS RE
GISTER
      5                                 ; <15:08> H                           <07:00> L
      6                                 ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
      7                                 ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
      8                                 ;   0   0   0   0   0   0   0   0      R   I   0   0   0
  0   0   0
      9                                 ;
     10                                 ;       R -     <07:00> DATA READY (R)
     11                                 ;                       CLEARED BY READ OF
     12                                 ;                       DATA <07:00> OR <15:00> OF (02)
     13                                 ;       I -     INTERRUPT ENABLE BIT (R/W)
     14                                 ;
     15                                 ;
     16                                 ;[X XXX XXX XXX XXN NNN]  02    FUNCTION/KEYBOARD BUFFERS
     17                                 ; <15:08> H                           <07:00> L
     18                                 ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
     19                                 ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
     20                                 ;[ S7  S6  S5  S4  S3  S2  S1  S0 ] [ K7  K6  K5  K4  K3
 K2  K1  K0 ]
     21                                 ;
     22                                 ;       <S7:S0> - FUNCTION DATA READ BUFFER
     23                                 ;       <K7:K0> - KEYBOARD INPUT BUFFER
     24                                 ;
     25                                 ;
     26                                 ;[X XXX XXX XXX XXN NNN]  04    PRINTER BUFFER STATUS REG
ISTER
     27                                 ; <15:08> H                           <07:00> L
     28                                 ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
     29                                 ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
     30                                 ;   0   0   0   0   0   0   0   0      R   I   0   0   0
  0   0   0
```

```
    31                                  ;
    32                                  ;       R -     <07:00> PRINTER READY (R)
    33                                  ;                       CLEARED BY WRITE OF
    34                                  ;                       DATA <07:00> OR <15:00> OF (06)
    35                                  ;       I -     INTERRUPT ENABLE BIT (R/W)
    36                                  ;
    37                                  ;
    38                                  ;[X XXX XXX XXX XXN NNN]  06    FUNCTION/PRINTER BUFFERS
    39                                  ; <15:08> H                          <07:00> L
    40                                  ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
    41                                  ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
    42                                  ;[ C4  C3  C2  C1  C0  F2  F1  F0 ] [ P7  P6  P5  P4  P3
 P2  P1  P0 ]
    43                                  ;
    44                                  ;       <P7-P0> -       PRINTER DATA (R/W)
    45                                  ;       <F2-F1> -       FUNCTION CODE
    46                                  ;       <C4-C0> -       CONTROL BITS (SEE FUNCTIONS)
    47                                  ;
    48                                  ;
```

SERIAL PORT AS SEEN ON THE ARB-11 BUS

```
      1                                 ;
      2                                 ;[X XXX XXX XXX XXN NNN]  10    FUNCTION INPUT BUFFER STA
TUS REGISTER
      3                                 ; <15:08> H                         <07:00> L
      4                                 ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
      5                                 ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
      6                                 ;   0   0   0   0   0   0   0   0      R   I   0   0   0
  0   0   0
      7                                 ;
      8                                 ;       R -     <07:00> FUNCTION DATA READY (R)
      9                                 ;                       CLEARED BY READ OF
     10                                 ;                       DATA <15:08> OR <15:00> OF (02)
     11                                 ;       I -     INTERRUPT ENABLE BIT (R/W)
     12                                 ;
     13                                 ;
     14                                 ;[X XXX XXX XXX XXN NNN]  12    A-PORT PERIPHERAL CONTROL
 REGISTER
     15                                 ; <15:08> H                         <07:00> L
     16                                 ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
     17                                 ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
     18                                 ;[ H7  H6  H5  H4  H3  H2  H1  H0 ] [ L7  L6  L5  L4  L3
 L2  L1  L0 ]
     19                                 ;
     20                                 ;       <H7-H0> -       <15:08> PIA CONTROL FOR (02)
     21                                 ;       <L7-L0> -       <07:00> PIA CONTROL FOR (02)
     22                                 ;
     23                                 ;
     24                                 ;[X XXX XXX XXX XXN NNN]  14    FUNTION CODE STATUS REGIS
TER
     25                                 ; <15:08> H                         <07:00> L
     26                                 ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
     27                                 ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
     28                                 ;   0   0   0   0   0   0   0   0      R   I   0   0   0
  0   0   0
     29                                 ;
```

```
    31                                  ;                       CLEARED BY WRITE OF
    32                                  ;                       DATA <15:08> OR <15:00> OF (06)
    33                                  ;       I -     INTERRUPT ENABLE BIT (R/W)
    34                                  ;
    35                                  ;
    36                                  ;[X XXX XXX XXX XXN NNN]  16    B-PORT PERIPHERAL CONTROL
REGISTER
    37                                  ; <15:08> H                           <07:00> L
    38                                  ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
02  01  00 ]
    39                                  ;  __  __  __  __  __  __  __  __     __  __  __  __  __
__  __  __
    40                                  ;[ H7  H6  H5  H4  H3  H2  H1  H0 ] [ L7  L6  L5  L4  L3
L2  L1  L0 ]
    41                                  ;
    42                                  ;       <H7-H0> -       <15:08> PIA CONTROL FOR (02)
    43                                  ;       <L7-L0> -       <07:00> PIA CONTROL FOR (02)
    44                                  ;
    45                                  ;
```

SERIAL PORT AS SEEN ON THE ARB-11 BUS

```
      1                                     ;
      2                                     ;       FUNCTION CODES
      3                                     ;
      4                                     ; F2 F1 F0
      5                                     ; __ __ __
      6                                     ;  0  0  0      NORMAL SERIAL PORT CONFIGURATION
      7                                     ;               BYTE 02 - KEYBOARD BUFFER
      8                                     ;               BYTE 03 - MODEM STATUS REGISTER (WHEN ENA
BLED, ELSE=0)
      9                                     ;               BYTE 06 - PRINTER BUFFER
     10                                     ;               BYTE 07 - <07:03> MODEM CONTROL REGISTER
     11                                     ;
     12                                     ;  0  0  1      MODEM CONTROL REGISTER READ/WRITE
     13                                     ;               BYTE 02 - KEYBOARD BUFFER
     14                                     ;               BYTE 03 - CURRENT CONTROL REGISTER
     15                                     ;               BYTE 06 - LOAD CONTROL REGISTER BUFFER
     16                                     ;               BYTE 07 - FUNCTION SELECT
     17                                     ;
     18                                     ;  0  1  0      LINE SERVICES ENABLES
     19                                     ;               BYTE 02 - KEYBOARD BUFFER
     20                                     ;               BYTE 03 - CURRENT ENABLES
     21                                     ;               BYTE 06 - LOAD ENABLES BUFFER
     22                                     ;               BYTE 07 - FUNCTION SELECT
     23                                     ;
     24                                     ;  0  1  1      MODEM SERVICES ENABLES
     25                                     ;               BYTE 02 - KEYBOARD BUFFER
     26                                     ;               BYTE 03 - CURRENT ENABLES
     27                                     ;               BYTE 06 - LOAD ENABLES BUFFER
     28                                     ;               BYTE 07 - FUNCTION SELECT
     29                                     ;
     30                                     ;  1  0  0      LINE CONTROL REGISTER
     31                                     ;               BYTE 02 - KEYBOARD BUFFER
     32                                     ;               BYTE 03 - CURRENT LINE CONTROL
     33                                     ;               BYTE 06 - LOAD CONTROL BUFFER
     34                                     ;               BYTE 07 - FUNCTION SELECT
     35                                     ;
```

```
                                        ;  1  0  1      LOW BYTE DIVISOR LATCH SELECT
                                        ;               BYTE 02 - KEYBOARD BUFFER
                                        ;               BYTE 03 - CURRENT DLL
                                        ;               BYTE 06 - LOAD DIVISOR BUFFER
                                        ;               BYTE 07 - FUNCTION SELECT
                                        ;
                                        ;  1  1  0      HIGH BYTE DIVISOR LATCH SELECT
                                        ;               BYTE 02 - KEYBOARD BUFFER
                                        ;               BYTE 03 - CURRENT DLM
                                        ;               BYTE 06 - LOAD DIVISOR BUFFER
                                        ;               BYTE 07 - FUNCTION SELECT
                                        ;
                                        ;  1  1  1      ABSOLUTE BINARY LOADER SELECT
                                        ;               BYTE 02 - KEYBOARD BUFFER
                                        ;               BYTE 03 - CURRENT CHECK SUM
                                        ;               BYTE 06 - BINARY INPUT BUFFER
                                        ;               BYTE 07 - FUNCTION SELECT
                                        ;
```

SERIAL PORT AS SEEN ON THE ARB-11 BUS

```
      1                                     ;
      2                                     ;       BIT PATTERN DEFINITIONS
      3                                     ;
      4                                     ;  KEYBOARD / PRINTER  NORMAL MODE
      5                                     ; FUNCTION <F2:F0>=000          WRITE NEW DATA INTO <C7:C
3>
      6                                     ;       BITS <C7:C3> CORRESPOND TO BITS <4:0> OF
      7                                     ;       THE MODEM CONTROL REGISTER OF THE INS8250 ACE.
      8                                     ;       BITS <S7:S0> IS THE MODEM STATUS REGISTER OF
      9                                     ;       THE INS8250 ACE.
     10                                     ;
     11                                     ;  MODEM CONTROL REGISTER ACCESS
     12                                     ; FUNCTION <F2:F0>=001          WRITE NEW DATA INTO <P7:P
3>
     13                                     ;       BITS <S7:S3> CORRESPOND TO BITS <4:0> OF
     14                                     ;       THE MODEM CONTROL REGISTER OF THE INS8250 ACE.
     15                                     ;
     16                                     ;  LINE INTERRUPT SERVICES ENABLES
     17                                     ; FUNCTION <F2:F0>=010          WRITE NEW DATA INTO <P3:P
0>
     18                                     ;       BITS <S3:S0> ARE THE LINE INTERRUPT SERVICE
     19                                     ;       ENABLES CORRESPONDING TO BITS <4:1> OF THE
     20                                     ;       LINE STATUS REGISTER OF THE INS8250 ACE.
     21                                     ;
     22                                     ;  MODEM INTERRUPT SERVICES ENABLES
     23                                     ; FUNCTION <F2:F0>=011          WRITE NEW DATA INTO <P7,P
3:P0>
     24                                     ;       BITS <S3:S0> ARE MODEM INTERRUPT SERVICE
     25                                     ;       ENABLES CORRESPONDING TO BITS <3:0> OF THE
     26                                     ;       MODEM STATUS REGISTER OF THE INS8250 ACE.
     27                                     ;       <P7>=1 DISABLES MODEM STATUS TO BYTE 03
     28                                     ;
     29                                     ;  LINE CONTROL REGISTER ACCESS
     30                                     ; FUNCTION <F2:F0>=100          WRITE NEW DATA INTO <P6:P
0>
     31                                     ;       BITS <S6:S0> ARE THE CONTENTS OF BITS <6:0>
     32                                     ;       OF THE LINE CONTROL REGISTER OF THE
     33                                     ;       INS8250 ACE.
```

```
     34                                 ;
     35                                 ;  LOW BYTE DIVISOR LATCH ACCESS
     36                                 ; FUNCTION <F2:F0>=101          WRITE NEW DATA INTO <P7:P
0>
     37                                 ;       BITS <S7:S0> CORRESPOND TO BITS <7:0> OF THE
     38                                 ;       LOW BYTE DIVISOR LATCH OF THE INS8250 ACE.
     39                                 ;
     40                                 ;  HIGH BYTE DIVISOR LATCH ACCESS
     41                                 ; FUNCTION <F2:F0>=110          WRITE NEW DATA INTO <P7:P
0>
     42                                 ;       BITS <S7:S0> CORRESPOND TO BITS <7:0> OF THE
     43                                 ;       HIGH BYE DIVISOR LATCH OF THE INS8250 ACE.
     44                                 ;
     45                                 ;  ABSOLUTE BINARY LOADER ACCESS
     46                                 ; FUNCTION <F2:F0>=111          WRITE BINARY INTO <P7:P0>
     47                                 ;       BITS <S7:S0> IS THE CURRENT CHECK SUM RESULT
     48                                 ;       FROM LOADING BINARY DATA
     49                                 ;       ODD JUMP ADDRESS TERMINATES LOADER
     50                                 ;       EVEN JUMP ADDRESS TRANSFERS CONTROL TO JUMP ADDRE
SS
     51                                 ;
```

SOFTWARE DEFINITIONS

```
                                        .SBTTL  SOFTWARE DEFINITIONS
                                        ;
                                        ;         THE SERIAL PORT SOFTWARE REQUIRES
                                        ;       THE 16-BIT PIA SOFTWARE PACKAGE '6821A.MAC',
                                        ;       THE 8250 ACE PACKAGE '8250A.MAC', AND
                                        ;       THE ABSOLUTE BINARY LOADER 'LOADER.MAC'.
                                        ;
                                        ;
000001                                  $A.8250 =1
000220                                  A.8250  =220    ;SERIAL PORT ADDRESS
                                        ;
000001                                  $A.6821 =1
000200                                  A.6821  =200    ;16-BIT BUS PORT ADDRESS
                                        ;
000001                                  $LOADER =1      ;USE ABSOLUTE LOADER
                                        ;
                                        ;
174000                                  PGMSAV  =174000 ;ROM ADDRESS
000000                                  VARSAV  =0      ;DIRECT VARIABLE SPACE TO 177
                                        ;
000177                                  STACK   =177    ;STACK AREA (64-BYTES)
                                        ;
                                        ;       INTERRUPT VECTORS
                                        ;
177740                                  .=177740
177740                                  FDB     SPRIUS  ;OOPS !
   177740     370     006
177742                                  FDB     SPRIUS  ;OOPS !
   177742     370     006
177744                                  FDB     SPRIUS  ;OOPS !
   177744     370     006
177746                                  FDB     SPRIUS  ;OOPS !
   177746     370     006
177750                          IRQ0:   FDB     SPRIUS  ;NO HARDWARE
   177750     370     006
```

```
   177752     370     006
32 177754                      IRQ2:   FDB     A.AL$I  ;A-PORT <7:0> INTERRUPT
   177754     373     230
33 177756                      IRQ3:   FDB     A.AH$I  ;A-PORT <15:8> INTERRUPT
   177756     373     217
34 177760                      IRQ4:   FDB     A.BL$I  ;B-PORT <7:0> INTERRUPT
   177760     373     252
35 177762                      IRQ5:   FDB     A.BH$I  ;B-PORT <15:8> INTERRUPT
   177762     373     241
36 177764                      IRQ6:   FDB     SPRIUS  ;NO HARDWARE
   177764     370     006
37 177766                      IRQ7:   FDB     A.82$I  ;8250 ACE INTERRUPT
   177766     374     123
38 177770                      IRQN:   FDB     SPRIUS  ;NO HARDWARE
   177770     370     006
39 177772                      SWINT:  FDB     SWIRQ   ;SWI INTERRUPT
   177772     370     007
40 177774                      NMINT:  FDB     NMIRQ   ;ATTACHED PROCESSOR INIT INTERRUPT
   177774     370     000
41 177776                      RESINT: FDB     RESTRT  ;POWER UP RESTART VECTOR
   177776     370     016
```

SOFTWARE DEFINITIONS

```
     42                                        ;
```

```
                                        .SBTTL  STARTUP AND BACKGROUND PROGRAM
                                        ;
         000000                         .=VARSAV
000000     000     000          NMIRQV: .BYTE   0,0     ;INIT INTERRUPT VECTOR
000002     000     000          SWIRQV: .BYTE   0,0     ;SWI INTERRUPT VECTOR
000004     000                  S.FNCT: .BYTE   0       ;DISPATCH FUNCTION
000005     000                  MSFLAG: .BYTE   0       ;MODEM STATUS FLAG
         000006                         VARSAV=.
         174000                         .=PGMSAV
                                        ;
174000                          NMIRQ:  LDX     NMIRQV          ;GET VECTOR
   174000     336     000
174002                                  BEQ     SPRIUS          ;BRANCH IF NOT DEFINED
   174002     047     002
174004                                  JSR     0,X             ;ELSE DO IT
   174004     255     000
174006                          SPRIUS: RTI                     ;HOW DID WE GET HERE ?
   174006     073
                                        ;
174007                          SWIRQ:  LDX     SWIRQV          ;GET VECTOR
   174007     336     002
174011                                  BEQ     1$              ;BRANCH IF NOT DEFINED
   174011     047     002
174013                                  JMP     0,X             ;SPECIAL SWI CALL TO ACCESS
   174013     156     000
174015                          1$:     RTI                     ;INTERRUPT DRIVEN ROUTINES
   174015     073
                                        ;
174016                          RESTRT: LDS     #,STACK         ;SET UP STACK POINTER
   174016     216     000     177
174021                                  LDX     #,0
   174021     316     000     000
174024                                  STX     NMIRQV          ;SET NMI FOR NOTHING
   174024     337     000
174026                                  STX     SWIRQV          ;SET SWI FOR NOTHING
```

```
 25 174030                              JSR     A.82$B          ;INIT SERIAL PORT BAUD-RATE
    174030     275     375     254
 26 174033                              BSR     PINITS          ;AND SERIAL PORT IN GENERAL
    174033     215     047
 27                                     ;
 28                                     ;       BACKGROUND PROGRAM
 29                                     ;
 30 174035                      1$:     LDS     #,STACK         ;SET THE STACK POINTER
    174035     216     000     177
 31 174040                              CLR     177740          ;ENABLE ALL INTERRUPTS
    174040     177     377     340
 32 174043                              CLI                     ;(VIA 6828 CONTROLLER)
    174043     016
 33 174044                              LDX     #,0             ;LOOP COUNTER
    174044     316     000     000
 34 174047                      2$:     INX                     ;LOOP HERE 64K TIMES
    174047     010
 35 174050                              BNE     2$
    174050     046     375
 36 174052                              JSR     A.LNUP          ;RE-ENABLE LINE
    174052     275     373     371
```

STARTUP AND BACKGROUND PROGRAM

```
    37 174055                              JSR     A.RCUP          ;RE-ENABLE RCVR
       174055     275     373     350
    38 174060                              JSR     A.XMUP          ;RE-ENABLE XMTR
       174060     275     373     327
    39 174063                              JSR     A.MOUP          ;RE-ENABLE MODEM
       174063     275     373     306
    40 174066                              JSR     A.ALUP          ;RE-ENABLE <7:0> INPUT
       174066     275     372     160
    41 174071                              JSR     A.AHUP          ;RE-ENABLE <15:8> INPUT
       174071     275     372     153
    42 174074                              JSR     A.BLUP          ;RE-ENABLE <7:0> OUTPUT
       174074     275     372     254
    43 174077                              JSR     A.BHUP          ;RE-ENABLE <15:8> OUTPUT
       174077     275     372     247
    44 174102                              BRA     1$
       174102     040     331
    45                                     ;
```

PORT INITIALIZATION

```
      1                                     .SBTTL  PORT INITIALIZATION
      2                                     ;
      3 174104                        PINITS: JSR     LOAD$I          ;INITIALIZE ABSOLUTE LOADER
        174104     275     371     306
      4                                     ;
      5                                     ;       INSURE A.8250 ACCESS IS NOT TO DLL/DLM
      6                                     ;
      7 174107                              JSR     A.LCRR          ;GET STATUS
        174107     275     374     031
      8 174112                              AND B   #,177           ;SET DLAB=0
        174112     304     177
      9 174114                              JSR     A.LCRW          ;SET STATUS
        174114     275     374     034
     10                                     ;
     11                                     ;       SET FUNCTION TO XMTR AND ENABLE ALL LINE / MODEM
     12                                     ;
     13 174117                              CLR     S.FNCT          ;XMTR FUNCTION
        174117     177     000     004
     14 174122                              LDA B   #,17            ;MODEM ENABLE
        174122     306     017
     15 174124                              STA B   A.MNBL
        174124     327     050
     16 174126                              LDA B   #,36            ;LINE ENABLE
        174126     306     036
     17 174130                              STA B   A.LNBL
        174130     327     047
     18 174132                              STA B   MSFLAG          ;AND FLAG
        174132     327     005
     19                                     ;
     20                                     ;       INITIALIZE A.8250 SERVICE VECTORS
     21                                     ;
     22 174134                              LDX     #,S.MODE        ;A.8250 MODEM VECTOR
        174134     316     370     361
     23 174137                              JSR     A.MOSV
        174137     275     373     303
```

```
   174142     316     370     355
25 174145                              JSR     A.RCSV
   174145     275     373     345
26 174150                              LDX     #,S.XMTR        ;A.8250 XMTR VECTOR
   174150     316     370     341
27 174153                              JSR     A.XMSV
   174153     275     373     324
28 174156                              LDX     #,A.82$L        ;A.8250 LINE VECTOR
   174156     316     374     335
29 174161                              JSR     A.LNSV
   174161     275     373     366
30                                     ;
31                                     ;       SET UP PARALLEL OUTPUT
32                                     ;
33 174164                              LDX     #,0             ;NO SERVICE
   174164     316     000     000
34 174167                              LDA B   #,44            ;CB2(ON E, LOW), CB1(-, HIGH)
   174167     306     044
35 174171                              LDA A   #,74
   174171     206     074
36 174173                              JSR     A.BLDF          ;SAVE CONTROL
```

PORT INITIALIZATION

```
           174173     275     372     240
        37 174176                             LDX     #,0             ;NO SERVICE
           174176     316     000     000
        38 174201                             LDA B   #,44            ;CB2(ON E, LOW), CB1(-, HIGH)
           174201     306     044
        39 174203                             LDA A   #,74
           174203     206     074
        40 174205                             JSR     A.BHDF          ;SAVE CONTROL
           174205     275     372     231
        41 174210                             LDA B   #,377           ;OUTPUT
           174210     306     377
        42 174212                             JSR     A.BLDR
           174212     275     372     214
        43 174215                             LDA B   #,377           ;OUTPUT
           174215     306     377
        44 174217                             JSR     A.BHDR
           174217     275     372     177
        45                                    ;
        46                                    ;       SET UP PARALLEL INPUT
        47                                    ;
        48 174222                             LDX     #,S.INAL        ;SERVICE VECTOR
           174222     316     371     003
        49 174225                             LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
           174225     306     045
        50 174227                             LDA A   #,74
           174227     206     074
        51 174231                             JSR     A.ALDF          ;SAVE CONTROL
           174231     275     372     144
        52 174234                             LDX     #,S.INAH        ;SERVICE VECTOR
           174234     316     371     145
        53 174237                             LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
           174237     306     045
        54 174241                             LDA A   #,74
           174241     206     074
        55 174243                             JSR     A.AHDF          ;SAVE CONTROL
```

```
    56 174246                          LDA B   #,0             ;INPUT
       174246     306     000
    57 174250                          JSR     A.ALDR
       174250     275     372     120
    58 174253                          LDA B   #,0             ;INPUT
       174253     306     000
    59 174255                          JSR     A.AHDR
       174255     275     372     103
    60                                 ;
    61                                 ;       ENABLE/DISABLE INTERRUPTS
    62                                 ;
    63 174260                          JSR     A.ALDN          ;CLEAR, CA2='1'
       174260     275     372     172
    64 174263                          JSR     A.ALUP          ;ENABLE
       174263     275     372     160
    65 174266                          JSR     A.ALRD+2        ;CLEAR IRQA, CA2='0'
       174266     275     372     313
    66                                 ;
    67 174271                          JSR     A.AHDN          ;CLEAR, CA2='1'
       174271     275     372     165
    68 174274                          JSR     A.AHUP          ;ENABLE
```

PORT INITIALIZATION

```
        174274     275     372     153
     69 174277                              JSR     A.AHRD+2        ;CLEAR IRQ, CA2='0'
        174277     275     372     275
     70                                     ;
     71 174302                              JSR     A.BLDN          ;CLEAR, CB2='1'
        174302     275     372     266
     72 174305                              JSR     A.BLRD+2        ;CLEAR IRQ, CB2='1'
        174305     275     373     063
     73 174310                              JSR     A.BLUP          ;ENABLE
        174310     275     372     254
     74                                     ;
     75 174313                              JSR     A.BHDN          ;CLEAR, CB2='1'
        174313     275     372     261
     76 174316                              JSR     A.BHRD+2        ;CLEAR IRQ, CB2='1'
        174316     275     373     043
     77 174321                              JSR     A.BHUP          ;ENABLE
        174321     275     372     247
     78                                     ;
     79 174324                              JSR     A.MOUP
        174324     275     373     306
     80 174327                              JSR     A.XMUP
        174327     275     373     327
     81 174332                              JSR     A.RCUP
        174332     275     373     350
     82 174335                              JSR     A.LNUP
        174335     275     373     371
     83                                     ;
     84 174340                              RTS
        174340     071
     85                                     ;
     86                                     ;       PORT SERVICE ROUTINES
     87                                     ;
     88 174341                         S.XMTR: LDA A   S.FNCT          ;GET FUNCTION
        174341     226     004
     89 174343                              CLC                     ;DEFAULT NO DATA
```

```
 90 174344                              BNE     1$              ;BRANCH IF NOT XMTR
    174344     046     006
 91 174346                              JSR     A.ALUP          ;ENABLE INPUT INTERRUPT
    174346     275     372     160
 92 174351                              JSR     A.ALRD          ;READ INPUT DATA
    174351     275     372     311
 93 174354                   1$:        RTS                     ;SEND TO XMTR
    174354     071
 94                                     ;
 95 174355                   S.RCVR:    JSR     A.BLWT+2        ;TRANSFER CHARACTER TO OUTPUT
    174355     275     373     150
 96 174360                              RTS                     ;READY OR NOT !
    174360     071
 97                                     ;
 98 174361                   S.MODE:    LDA A   S.FNCT          ;GET FUNCTION
    174361     226     004
 99 174363                              BNE     2$              ;BRANCH IF NOT STATUS
    174363     046     012
100 174365                              PSH B                   ;SAVE B FOR A MOMENT
    174365     067
101 174366                              LDA A   MSFLAG          ;CHECK FOR MODEM STATUS DISABLED
```

PORT INITIALIZATION

```
        174366     226     005
    102 174370                             BPL     1$              ;ENABLED - SKIP
        174370     052     001
    103 174372                             CLR B                   ;CLEAR BYTE
        174372     137
    104 174373                         1$: JSR     A.BHWT+2        ;TRANSFER DATA
        174373     275     373     127
    105 174376                             PUL B                   ;RESTORE B
        174376     063
    106 174377                         2$: JSR     A.82$M          ;DO MODEM STATUS
        174377     275     375     023
    107 174402                             RTS
        174402     071
    108                                    ;
    109 174403                         S.INAL: LDA A   S.FNCT      ;GET FUNCTION
        174403     226     004
    110 174405                             BNE     1$              ;BRANCH IF NOT XMTR
        174405     046     007
    111 174407                             JSR     A.ALDN          ;DISABLE INPUT INTERRUPT
        174407     275     372     172
    112 174412                             JSR     A.XMUP          ;ENABLE A.8250 XMTR
        174412     275     373     327
    113 174415                             RTS
        174415     071
    114 174416                         1$: JSR     A.ALRD          ;READ BYTE
        174416     275     372     311
    115 174421                             BCC     8$              ;WHAT DATA ?
        174421     044     115
    116 174423                             LDA A   S.FNCT          ;GET FUNCTION CODE
        174423     226     004
    117 174425                             DEC A                   ;MODEM CONTROL ?
        174425     112
    118 174426                             BNE     2$              ;BRANCH IF NOT
        174426     046     015
    119 174430                             AND B   #,370           ;MASK EXTRA
```

```
120 174432                              TBA
    174432     027
121 174433                              CLC                     ;SHIFT INTO POSITION
    174433     014
122 174434                              ROR B
    174434     126
123 174435                              ASR B
    174435     127
124 174436                              ASR B
    174436     127
125 174437                              JSR     A.MCRW          ;SAVE NEW CONTROL
    174437     275     374     020
126 174442                              TAB
    174442     026
127 174443                              BRA     9$
    174443     040     074
128 174445                      2$:     DEC A                   ;LOAD LINE ENABLES ?
    174445     112
129 174446                              BNE     3$              ;BRANCH IF NOT
    174446     046     007
130 174450                              AND B   #,17            ;MASK EXTRA BITS
```

PORT INITIALIZATION

```
        174450     304     017
    131 174452                             ASL B                   ;SHIFT INTO POSITION
        174452     130
    132 174453                             STA B   A.LNBL          ;SAVE NEW ENABLES
        174453     327     047
    133 174455                             BRA     9$
        174455     040     062
    134 174457                         3$: DEC A                   ;LOAD MODEM ENABLES ?
        174457     112
    135 174460                             BNE     4$              ;BRANCH IF NOT
        174460     046     014
    136 174462                             AND B   #,217           ;MASK EXTRA BITS
        174462     304     217
    137 174464                             STA B   MSFLAG          ;SAVE DISABLE STATUS
        174464     327     005
    138 174466                             AND B   #,17            ;MASK FLAG BIT
        174466     304     017
    139 174470                             STA B   A.MNBL          ;SAVE NEW ENABLES
        174470     327     050
    140 174472                             LDA B   MSFLAG          ;GET FLAG AND ENABLES
        174472     326     005
    141 174474                             BRA     9$
        174474     040     043
    142 174476                         4$: DEC A                   ;LOAD LINE CONTROL REGISTER ?
        174476     112
    143 174477                             BNE     5$              ;BRANCH IF NOT
        174477     046     007
    144 174501                             AND B   #,177           ;MASK DLAB
        174501     304     177
    145 174503                             JSR     A.LCRW          ;SAVE NEW VALUE
        174503     275     374     034
    146 174506                             BRA     9$
        174506     040     031
    147 174510                         5$: DEC A                   ;LOAD DLL ?
        174510     112
```

```
        174511     046     005
    149 174513                             JSR     A.DLLW          ;SAVE NEW VALUE
        174513     275     374     054
    150 174516                             BRA     9$
        174516     040     021
    151 174520                         6$: DEC A                   ;LOAD DLM ?
        174520     112
    152 174521                             BNE     7$              ;BRANCH IF NOT
        174521     046     005
    153 174523                             JSR     A.DLMW          ;SAVE NEW VALUE
        174523     275     374     106
    154 174526                             BRA     9$
        174526     040     011
    155 174530                         7$: DEC A                   ;LOADER ?
        174530     112
    156 174531                             BNE     8$              ;BRANCH IF NOT
        174531     046     005
    157 174533                             JSR     LOADER          ;AND PROCESS
        174533     275     371     314
    158 174536                             BRA     9$
        174536     040     001
```

PORT INITIALIZATION

```
    159 174540                          8$:     RTS
        174540     071
    160 174541                          9$:     JSR     A.BHWT+2        ;SEND DATA
        174541     275     373     127
    161 174544                                  RTS
        174544     071
    162                                         ;
    163 174545                          S.INAH: JSR     A.AHRD          ;READ BYTE
        174545     275     372     273
    164 174550                                  BCC     8$              ;WHAT DATA ?
        174550     044     127
    165 174552                                  TBA                     ;COPY B TO A
        174552     027
    166 174553                                  AND B   #,7             ;MASK EXTRA
        174553     304     007
    167 174555                                  STA B   S.FNCT          ;NEW FUNCTION
        174555     327     004
    168 174557                                  BNE     1$              ;BRANCH IF NOT CONTROL
        174557     046     022
    169 174561                                  TAB                     ;GET DATA AGAIN
        174561     026
    170 174562                                  CLC                     ;POSITION DATA
        174562     014
    171 174563                                  ROR B
        174563     126
    172 174564                                  ASR B
        174564     127
    173 174565                                  ASR B
        174565     127
    174 174566                                  JSR     A.MCRW          ;SEND TO A.8250
        174566     275     374     020
    175 174571                                  CLR B                   ;CLEAR BYTE
        174571     137
    176 174572                                  LDA A   MSFLAG          ;MODEM STATUS DISABLED ?
        174572     226     005
```

```
        174574     053     104
    178 174576                             JSR     A.MSRR          ;GET CURRENT STATUS
        174576     275     374     007
    179 174601                             BRA     9$
        174601     040     077
    180 174603                         1$: DEC B                   ;GET CURRENT MODEM CONTROL ?
        174603     132
    181 174604                             BNE     2$              ;BRANCH IF NOT
        174604     046     010
    182 174606                             JSR     A.MCRR          ;READ CURRENT SETTING
        174606     275     374     015
    183 174611                             ASL B                   ;SHIFT INTO PLACE
        174611     130
    184 174612                             ASL B
        174612     130
    185 174613                             ASL B
        174613     130
    186 174614                             BRA     9$
        174614     040     064
    187 174616                         2$: DEC B                   ;LINE SERVICE ENABLES ?
        174616     132
```

PORT INITIALIZATION

```
   188 174617                                   BNE     3$              ;BRANCH IF NOT
       174617     046     005
   189 174621                                   LDA B   A.LNBL          ;ENABLES
       174621     326     047
   190 174623                                   ASR B                   ;SHIFT INTO POSITION
       174623     127
   191 174624                                   BRA     9$
       174624     040     054
   192 174626                           3$:     DEC B                   ;MODEM SERVICE ENABLES ?
       174626     132
   193 174627                                   BNE     4$              ;BRANCH IF NOT
       174627     046     004
   194 174631                                   LDA B   MSFLAG          ;GET FLAG AND ENABLES
       174631     326     005
   195 174633                                   BRA     9$
       174633     040     045
   196 174635                           4$:     DEC B                   ;LINE CONTROL REGISTER ?
       174635     132
   197 174636                                   BNE     5$              ;BRANCH IF NOT
       174636     046     007
   198 174640                                   JSR     A.LCRR          ;GET REGISTER
       174640     275     374     031
   199 174643                                   AND B   #,177           ;MASK OUT DLAB
       174643     304     177
   200 174645                                   BRA     9$
       174645     040     033
   201 174647                           5$:     DEC B                   ;DLL ACCESS ?
       174647     132
   202 174650                                   BNE     6$              ;BRANCH IF NOT
       174650     046     005
   203 174652                                   JSR     A.DLLR          ;READ VALUE
       174652     275     374     037
   204 174655                                   BRA     9$
       174655     040     023
   205 174657                           6$:     DEC B                   ;DLM ACCESS ?
```

```
206 174660                              BNE     7$              ;BRANCH IF NOT
    174660     046     005
207 174662                              JSR     A.DLMR          ;READ VALUE
    174662     275     374     071
208 174665                              BRA     9$
    174665     040     013
209 174667                          7$: DEC B                   ;LOADER SELECTED ?
    174667     132
210 174670                              BNE     8$              ;BRANCH IF NOT
    174670     046     007
211 174672                              JSR     LOAD$I          ;ELSE INITIALIZE LOADER
    174672     275     371     306
212 174675                              LDA B   #,0             ;ZERO CHECK SUM
    174675     306     000
213 174677                              BRA     9$
    174677     040     001
214 174701                          8$: RTS
    174701     071
215 174702                          9$: JSR     A.BHWT+2        ;SEND
    174702     275     373     127
216 174705                              RTS
```

PORT INITIALIZATION

```
        174705     071
    217                                     ;
```

ABSOLUTE BINARY LOADER

```
      2                                     .SBTTL  ABSOLUTE BINARY LOADER
      3                                     ;
      4                                     ;         THIS LOADER ROUTINE IS AN ADAPTATION OF THE
      5                                     ;       'DEC' PAPER TAPE ABSOLUTE BINARY LOADER.
      6                                     ;         THE LOADER PROVIDES A MEANS OF LOADING OBJECT
      7                                     ;       CODE INTO ANY REGION OF MEMORY AND STARTING
      8                                     ;       PROGRAM EXECUTION.
      9                                     ;
     10                                     ;         AN ODD JUMP ADDRESS TERMINATES LOADER
     11                                     ;         AN EVEN JUMP ADDRESS TRANSFERS CONTROL TO THE J
UMP ADDRESS
     12                                     ;
     13                                     .IF     NDF,$LOADER
     14                                     VARSAV  =0      ;VARIABLE SPACE
     15                                     PGMSAV  =100000 ;PROGRAM SPACE
     16                                     .=PGMSAV
     17                                     .ENDC
     18                                     ;
     19         174706                      PGMSAV=.
     20         000006                      .=VARSAV
     21 000006     000     000          LOAD$V: .BYTE   0,0     ;SERVICE ADDRESS
     22 000010     000     000          L.BYTC: .BYTE   0,0     ;BYTE COUNTER
     23 000012     000     000          L.ADDR: .BYTE   0,0     ;LOAD ADDRESS
     24 000014     000                  L.CKSM: .BYTE   0       ;CHECK SUM
     25         000015                      VARSAV=.
     26         174706                      .=PGMSAV
     27                                     ;
     28                                     ;       INITIALIZE ENTRY POINT
     29                                     ;
     30 174706                           LOAD$I: LDX     #,L.LDR0        ;FIRST ENTRY
        174706     316     371     344
     31 174711                               STX     LOAD$V
        174711     337     006
     32 174713                               RTS
        174713     071
     33                                     ;
```

LOADER ROUTINE

```
      1                                     .SBTTL  LOADER ROUTINE
      2                                     ;
      3                                     ;       ENTER WITH DATA IN <B>
      4                                     ;       EXITS WITH CURRENT CHECK SUM IN <B>
      5                                     ;
      6 174714                      LOADER: LDA A   L.CKSM          ;UPDATE CHECK SUM
        174714     226     014
      7 174716                              ABA
        174716     033
      8 174717                              STA A   L.CKSM
        174717     227     014
      9 174721                              LDX     L.BYTC          ;UPDATE BYTE COUNT
        174721     336     010
     10 174723                              DEX
        174723     011
     11 174724                              STX     L.BYTC
        174724     337     010
     12 174726                              LDX     LOAD$V          ;DISPATCH ADDRESS
        174726     336     006
     13 174730                              JMP     0,X             ;GO TO IT
        174730     156     000
     14 174732                      L.GBYT: TSX                     ;GET RETURN ADDRESS ON STACK
        174732     060
     15 174733                              LDX     0,X
        174733     356     000
     16 174735                              STX     LOAD$V          ;SAVE RETURN ADDRESS
        174735     337     006
     17 174737                              INS                     ;POP RETURN FROM STACK
        174737     061
     18 174740                              INS
        174740     061
     19 174741                              LDA B   L.CKSM          ;RETURN CHECK SUM IN <B>
        174741     326     014
     20 174743                              RTS                     ;BACK TO CALLER
        174743     071
```

```
 22 174744                          L.LDR0: CLR     L.CKSM          ;CLEAR CHECK SUM
    174744     177     000     014
 23 174747                                  BRA     L.LDR2
    174747     040     005
 24 174751                          L.LDR1: CLR     L.CKSM          ;CLEAR CHECK SUM
    174751     177     000     014
 25 174754                                  BSR     L.GBYT          ;BACK FOR A BYTE
    174754     215     354
 26 174756                          L.LDR2: DEC B                   ;A '1' ?
    174756     132
 27 174757                                  BNE     L.LDR1          ;LOOP UNTIL '1' FOUND
    174757     046     370
 28 174761                                  BSR     L.GBYT          ;BACK FOR A BYTE
    174761     215     347
 29 174763                                  TST B                   ;= '0' ?
    174763     135
 30 174764                                  BNE     L.LDR0          ;LOOP BACK UNTIL '1','0' SEQUENCE
    174764     046     356
 31 174766                                  BSR     L.GBYT          ;BACK FOR A BYTE
    174766     215     342
 32 174770                                  STA B   L.ADDR+1        ;NOW SAVE BYTE COUNT
```

LOADER ROUTINE

```
        174770     327     013
     33 174772                          BSR     L.GBYT          ;BACK FOR A BYTE
        174772     215     336
     34 174774                          LDA A   L.ADDR+1        ;RETRIEVE LOW ORDER COUNT
        174774     226     013
     35 174776                          SUB A   #,4             ;CORRECT BYTE COUNT
        174776     200     004
     36 175000                          SBC B   #,0
        175000     302     000
     37 175002                          STA A   L.BYTC+1        ;AND SAVE IT
        175002     227     011
     38 175004                          STA B   L.BYTC
        175004     327     010
     39 175006                          LDX     L.BYTC          ;L.BYTC=2 ?
        175006     336     010
     40 175010                          CPX     #,2
        175010     214     000     002
     41 175013                          BNE     L.LDR4          ;BRANCH IF NOT
        175013     046     033
     42 175015                L.LDR3:   BSR     L.GBYT          ;BACK FOR A BYTE
        175015     215     313
     43 175017                          STA B   L.ADDR+1        ;GET JUMP ADDRESS
        175017     327     013
     44 175021                          BSR     L.GBYT          ;BACK FOR A BYTE
        175021     215     307
     45 175023                          STA B   L.ADDR
        175023     327     012
     46 175025                          BSR     L.GBYT          ;BACK FOR CHECK SUM BYTE !
        175025     215     303
     47 175027                          TST     L.CKSM          ;CHECK FOR ERROR
        175027     175     000     014
     48 175032                          BNE     L.LDR6          ;SKIP ON ERROR
        175032     046     043
     49 175034                          LDA A   L.ADDR+1        ;CHECK FOR ODD ADDRESS
        175034     226     013
```

```
   175036     205     001
51 175040                          BNE     L.LDR6          ;TERMINATE BY RESTARTING SCAN
   175040     046     035
52 175042                          LDX     L.ADDR          ;ELSE GO TO ADDRESS
   175042     336     012
53 175044                          JSR     0,X
   175044     255     000
54 175046                          BRA     L.LDR6          ;ON QUICK RETURN - CONTINUE
   175046     040     027
55 175050                 L.LDR4:  BSR     L.GBYT          ;BACK FOR A BYTE
   175050     215     260
56 175052                          STA B   L.ADDR+1        ;GET LOAD ADDRESS
   175052     327     013
57 175054                          BSR     L.GBYT          ;BACK FOR A BYTE
   175054     215     254
58 175056                          STA B   L.ADDR
   175056     327     012
59 175060                 L.LDR5:  BSR     L.GBYT          ;BACK FOR A BYTE
   175060     215     250
60 175062                          LDX     L.BYTC          ;CHECK BYTE COUNT
   175062     336     010
```

LOADER ROUTINE

```
     61 175064                                BMI     L.LDR6          ;BRANCH IF DONE
        175064     053     011
     62 175066                                LDX     L.ADDR          ;GET LOAD ADDRESS
        175066     336     012
     63 175070                                STA B   0,X             ;STORE DATA
        175070     347     000
     64 175072                                INX                     ;UPDATE LOAD ADDRESS
        175072     010
     65 175073                                STX     L.ADDR
        175073     337     012
     66 175075                                BRA     L.LDR5
        175075     040     361
     67 175077                      L.LDR6:   BSR     L.GBYT          ;CURRENT BYTE WAS CHECK SUM RESUL
T
        175077     215     231
     68 175101                                BRA     L.LDR0          ;START NEW SEQUENCE
        175101     040     241
     69                                       ;
```

A.6821 PERIPHERAL INTERFACE SOFTWARE

```
                                .SBTTL  A.6821 PERIPHERAL INTERFACE SOFTWARE
                                ;
                                ;         THIS SOFTWARE PACKAGE SUPPORTS THE
                                ;       6821 PERIPHERAL INTERFACE ADAPTER (PIA).
                                ;         ENTRY POINTS TO THE SOFTWARE ARE DEFINED
                                ;       FOR EXTERNAL ACCESS:
                                ;
                                ;         THE FOLLOWING DATA DIRECTION ROUTINES ARE
                                ;       ENTERED WITH
                                ;               <B> = DIRECTION CONTROL
                                ;       BY A JSR ____
                                ;
                                ;       A.AHDR --> A-PORT <15:8> DIRECTION
                                ;       A.ALDR --> A-PORT <7:0> DIRECTION
                                ;       A.BHDR --> B-PORT <15:8> DIRECTION
                                ;       A.BLDR --> B-PORT <7:0> DIRECTION
                                ;
                                ;         THE FOLLOWING CONTROL ROUTINES ARE ENTERED WITH
                                ;               <X> = SERVICE ADDRESS
                                ;               <B> = INTERRUPT ENABLE CONTROL
                                ;               <A> = INTERRUPT DISABLE CONTROL
                                ;       BY A JSR ____
                                ;
                                ;       A.AHDF --> A-PORT <15:8>
                                ;       A.ALDF --> A-PORT <7:0>
                                ;       A.BHDF --> B-PORT <15:8>
                                ;       A.BLDF --> B-PORT <7:0>
                                ;
                                ;         THE FOLLOWING INTERRUPT ENABLE ROUTINES ARE
                                ;       ENTERED BY JSR ____
                                ;
                                ;       A.AHUP --> A-PORT <15:8> INTERRUPT ENABLE
                                ;       A.ALUP --> A-PORT <7:0> INTERRUPT ENABLE
                                ;       A.BHUP --> B-PORT <15:0> INTERRUPT ENABLE
                                ;       A.BLUP --> B-PORT <7:0> INTERRUPT ENABLE
```

```
38                                      ;         THE FOLLOWING INTERRUPT DISABLE ROUTINES ARE
39                                      ;       ENTERED BY JSR ____
40                                      ;
41                                      ;       A.AHDN --> A-PORT <15:0> INTERRUPT DISABLE
42                                      ;       A.ALDN --> A-PORT <7:0> INTERRUPT DISABLE
43                                      ;       A.BHDN --> B-PORT <15:8> INTERRUPT DISABLE
44                                      ;       A.BLDN --> B-PORT <7:0> INTERRUPT DISABLE
45                                      ;
46                                      ;         THE FOLLOWING ARE INTERRUPT ENTRY POINTS
47                                      ;       FOR THE DUAL PIA PORTS
48                                      ;
49                                      ;       A.AH$I --> A-PORT <15:8> IRQ
50                                      ;
51                                      ;       A.AL$I --> A-PORT <7:0> IRQ
52                                      ;
53                                      ;       A.BH$I --> B-PORT <15:8> IRQ
54                                      ;
55                                      ;       A.BL$L --> B-PORT <7:0> IRQ
56                                      ;
```

A.6821 PERIPHERAL INTERFACE SOFTWARE

```
      1                                 ;
      2                                 ;         THE FOLLOWING ENTRY POINTS ARE USED TO TRANSFER
      3                                 ;       DATA TO THE VARIOUS PIA I/O REGISTERS. THE MAIN
      4                                 ;       ENTRY POINT VERIFIES THAT AT LEAST ONE OF THE
      5                                 ;       PORT IRQ FLAGS IS SET BEFORE TRANSFERRING DATA.
      6                                 ;       IF DATA IS TRANSFERED THEN THE 'C' BIT = 1,
      7                                 ;       IF THE PORT WAS NOT READY THEN NO TRANSFER
      8                                 ;       IS PERFORMED AND THE 'C' BIT = 0.
      9                                 ;         THE ROUTINES MAY BE ENTERED AT THE ENTRY POINT
+ 2
     10                                 ;       IF NO IRQ FLAG CHECKS ARE TO BE MADE.
     11                                 ;         BYTE DATA IS PASSED IN <B>.
     12                                 ;         WORD DATA IS PASSED IN <B,A>
     13                                 ;
     14                                 ;       CALL BY JSR ____
     15                                 ;
     16                                 ;       A.AHRD --> A-PORT <15:8> READ
     17                                 ;       A.ALRD --> A-PORT <7:0> READ
     18                                 ;       A.AWRD --> A-PORT <15:0> READ
     19                                 ;       A.BHRD --> B-PORT <15:8> READ
     20                                 ;       A.BLRD --> B-PORT <7:0> READ
     21                                 ;       A.BWRD --> B-PORT <15:0> READ
     22                                 ;
     23                                 ;
     24                                 ;       A.AHWT --> A-PORT <15:8> WRITE
     25                                 ;       A.ALWT --> A-PORT <7:0> WRITE
     26                                 ;       A.AWWT --> A-PORT <15:0> WRITE
     27                                 ;       A.BHWT --> B-PORT <15:8> WRITE
     28                                 ;       A.BLWT --> B-PORT <7:0> WRITE
     29                                 ;       A.BWWT --> B-PORT <15:0> WRITE
     30                                 ;
```

A.6821 PIA HARDWARE

```
      1                                 .SBTTL  A.6821 PIA HARDWARE
      2                                 ;
      3                                 ;         THE PIA CONFIGURATION  CONSISTS OF TWO
      4                                 ;       MC6821 PERIPHERAL INTERFACE ADAPTERS (PIA'S)
      5                                 ;       WHICH MAY BE ADDRESSED AS 16-BIT PORTS.
      6                                 ;       ONE PIA UNIT FORMS THE HIGH ORDER PART <15:8>
      7                                 ;       AND THE OTHER THE LOW ORDER PART <7:0>
      8                                 ;
      9                                 ; REGISTER 0
     10         000000                 CRAH    =0      ;A-PORT <15:8> CONTROL REGISTER
     11                                 ;
     12                                 ; REGISTER 1
     13         000001                 CRAL    =1      ;A-PORT <7:0> CONTROL REGISTER
     14                                 ;
     15                                 ; REGISTER 2
     16         000002                 CRBH    =2      ;B-PORT <15:8> CONTROL REGISTER
     17                                 ;
     18                                 ; REGISTER 3
     19         000003                 CRBL    =3      ;B-PORT <7:0> CONTROL REGISTER
     20                                 ;
     21                                 ; REGISTER 4
     22         000004                 DDRAH   =4      ;CRAH<2>=0, A-PORT <15:8> DATA DIRECTION
REGISTER
     23         000004                 PRAH    =4      ;CRAH<2>=1, A-PORT <15:8> DATA REGISTER
     24                                 ;
     25                                 ; REGISTER 5
     26         000005                 DDRAL   =5      ;CRAL<2>=0, A-PORT <7:0> DATA DIRECTION R
EGISTER
     27         000005                 PRAL    =5      ;CRAL<2>=1, A-PORT <7:0> DATA REGISTER
     28                                 ;
     29                                 ; REGISTER 6
     30         000006                 DDRBH   =6      ;CRBH<2>=0, B-PORT <15:8> DATA DIRECTION
REGISTER
     31         000006                 PRBH    =6      ;CRBH<2>=1, B-PORT <15:8> DATA REGISTER
     32                                 ;
     33                                 ; REGISTER 7
```

```
    35         000007                          PRBL    =7      ;CRBL<2>=1, B-PORT <7:0> DATA REGISTER
    36                                         ;
```

A.6821 PIA HARDWARE

```
      1                                 ;
      2                                 ;       A-PORT CONTROL REGISTERS
      3                                 ;
      4                                 ; CONTROL REGISTERS CRA_  - READ/WRITE REGISTERS
      5                                 ;                                 CONTROLLING INTERRUPTS AN
D STROBES
      6                                 ;
      7                                 ; BIT 1 0       CONTROL OF INTERRUPT INPUT CA1
      8                                 ;               ACTIVE TRANSITION
      9                                 ;     _ _       TO SET CRA<7>           /IRQA OUTPUT
     10                                 ;     0 0               (-)             DISABLED
     11                                 ;     0 1               (-)             /CRA<7>
     12                                 ;     1 0               (+)             DISABLED
     13                                 ;     1 1               (+)             /CRA<7>
     14                                 ;       CRA<7> CLEARED BY A READ OF CORRESPONDING PRA_
     15                                 ;       /IRQA FOLLOWS STATE OF /CRA<7> WHEN ENABLED
     16                                 ;
     17                                 ; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
     18                                 ;     _
     19                                 ;     0         ACCESS DATA DIRECTION REGISTERS
     20                                 ;     1         ACCESS PORT DATA REGISTER
     21                                 ;
     22                                 ; BIT 5 4 3     CONTROL OF CA2 AS AN INTERRUPT INPUT
     23                                 ;               ACTIVE TRANSITION
     24                                 ;     _ _ _     TO SET CRA<6>           /IRQB OUTPUT
     25                                 ;     0 0 0             (-)             DISABLED
     26                                 ;     0 0 1             (-)             /CRA<6>
     27                                 ;     0 1 0             (+)             DISABLED
     28                                 ;     0 1 1             (+)             /CRA<6>
     29                                 ;       CRA<6> CLEARED BY READ OF CORRESPONDING PRA_
     30                                 ;       /IRQB FOLLOWS STATE OF CRA<6> WHEN ENABLED
     31                                 ;
     32                                 ; BIT 5 4 3     CONTROL OF CA2 AS AN OUTPUT
     33                                 ;     _ _ _     CA2 LOW                 CA2 HIGH
     34                                 ;     1 0 0     AFTER READ TO PRA_      WHEN CRA<7> IS SE
T
```

```
     36                                 ;     1 0 1     AFTER READ TO PRA_      ON SECOND (+) E A
FTER
     37                                 ;               ON (+) OF NEXT E        PIA DESELECTED
     38                                 ;     1 1 0     =CRA<3>
     39                                 ;     1 1 1                             =CRA<3>
     40                                 ;
     41                                 ; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     42                                 ;                       OF CA2, CLEARED BY READ OF PRA_
     43                                 ;
     44                                 ; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     45                                 ;                       OF CA1, CLEARED BY READ OF PRA_
     46                                 ;
     47                                 ;       DATA DIRECTION REGISTER (DDRA_)
     48                                 ; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
     49                                 ;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
     50                                 ;               BY 0'S AND OUTPUT BITS BY 1'S.
     51                                 ;
     52                                 ;       PERIPHERAL DATA REGISTER (PRA_)
     53                                 ; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGIST
ERS
     54                                 ;
```

A.6821 DETAILS

```
      1                                 .SBTTL  A.6821 DETAILS
      2                                 ;
      3                                 ;       B-PORT CONTROL REGISTERS
      4                                 ;
      5                                 ; CONTROL REGISTERS CRB_  - READ/WRITE REGISTERS
      6                                 ;                               CONTROLLING INTERRUPTS AN
D STROBES
      7                                 ;
      8                                 ; BIT 1 0       CONTROL OF INTERRUPT INPUT CB1
      9                                 ;               ACTIVE TRANSITION
     10                                 ;     _ _       TO SET CRB<7>           /IRQA OUTPUT
     11                                 ;     0 0               (-)             DISABLED
     12                                 ;     0 1               (-)             /CRB<7>
     13                                 ;     1 0               (+)             DISABLED
     14                                 ;     1 1               (+)             /CRB<7>
     15                                 ;       CRB<7> CLEARED BY A READ OF CORRESPONDING PRB_
     16                                 ;       /IRQA FOLLOWS STATE OF /CRB<7> WHEN ENABLED
     17                                 ;
     18                                 ; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
     19                                 ;     _
     20                                 ;     0         ACCESS DATA DIRECTION REGISTERS
     21                                 ;     1         ACCESS PORT DATA REGISTER
     22                                 ;
     23                                 ; BIT 5 4 3     CONTROL OF CB2 AS AN INTERRUPT INPUT
     24                                 ;               ACTIVE TRANSITION
     25                                 ;     _ _ _     TO SET CRB<6>           /IRQB OUTPUT
     26                                 ;     0 0 0             (-)             DISABLED
     27                                 ;     0 0 1             (-)             /CRB<6>
     28                                 ;     0 1 0             (+)             DISABLED
     29                                 ;     0 1 1             (+)             /CRB<6>
     30                                 ;       CRB<6> CLEARED BY READ OF CORRESPONDING PRB_
     31                                 ;       /IRQB FOLLOWS STATE OF CRB<6> WHEN ENABLED
     32                                 ;
     33                                 ; BIT 5 4 3     CONTROL OF CB2 AS AN OUTPUT
     34                                 ;     _ _ _     CB2 LOW                 CB2 HIGH
```

```
     36                                 ;              ON (+) OF NEXT E        BY CB1 TRANSITION
     37                                 ;     1 0 1    AFTER WRITE TO PRB      ON SECOND (+) E A
FTER
     38                                 ;              ON (+) OF NEXT E        PIA DESELECTED
     39                                 ;     1 1 0    =CRB<3>
     40                                 ;     1 1 1                            =CRB<3>
     41                                 ;
     42                                 ; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     43                                 ;                       OF CB2, CLEARED BY READ OF PRB_
     44                                 ;
     45                                 ; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     46                                 ;                       OF CB1, CLEARED BY READ OF PRB_
     47                                 ;
     48                                 ;       DATA DIRECTION REGISTER (DDRB_)
     49                                 ; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
     50                                 ;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
     51                                 ;               BY 0'S AND OUTPUT BITS BY 1'S.
     52                                 ;
     53                                 ;       PERIPHERAL DATA REGISTER (PRB_)
     54                                 ; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGIST
ERS
     55                                 ;
```

A.6821 VARIABLES AND POINTERS

```
                                        .SBTTL  A.6821 VARIABLES AND POINTERS
                                        ;
                                        ;       PORT STATUS AND BUFFERSS
                                        ;
                                        .IF     NDF,$A.6821
                                        A.6821  =       200     ;PIA PORT DEFAULT ADDRESS
                                        VARSAV  =       0       ;DEFAULT VARIABLE ADDRESS
                                        PGMSAV  =       100000  ;DEFAULT PROGRAM ADDRESS
                                        .=PGMSAV
                                        .ENDC
                                        ;
        175103                          PGMSAV=.
        000015                          .=VARSAV
                                        ;
000015    000     000           A.APRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
000017    000     000                   .BYTE   0,0     ;'UP','DN' CONTROL
000021    000     000                   .BYTE   0,0     ;<7:0> SERVICE VECTOR
000023    000     000                   .BYTE   0,0     ;'UP','DN' CONTROL
000025    000     000           A.BPRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
000027    000     000                   .BYTE   0,0     ;'UP','DN' CONTROL
000031    000     000                   .BYTE   0,0     ;<7:0> SERVICE VECTOR
000033    000     000                   .BYTE   0,0     ;'UP','DN' CONTROL
                                        ;
        000035                          VARSAV=.
        175103                          .=PGMSAV
                                        ;
```

A.6821 A-PORT CONTROL ROUTINES

```
      1                                     .SBTTL  A.6821 A-PORT CONTROL ROUTINES
      2                                     ;
      3                                     ;       DATA DIRECTION LOAD ROUTINES
      4                                     ;       ENTER WITH DIRECTION CODE IN <B>
      5                                     ;
      6 175103                        A.AHDR: LDA A   CRAH+A.6821     ;GET CURRENT CONTROL
        175103     226     200
      7 175105                                PSH A                   ;AND SAVE
        175105     066
      8 175106                                AND A   #,373           ;MASK DDR BIT
        175106     204     373
      9 175110                                STA A   CRAH+A.6821     ;ACCESS DIRECTION REGISTER
        175110     227     200
     10 175112                                STA B   DDRAH+A.6821    ;SET DIRECTIONS
        175112     327     204
     11 175114                                PUL A                   ;GET SAVED CONTROL
        175114     062
     12 175115                                STA A   CRAH+A.6821     ;AND RESTORE
        175115     227     200
     13 175117                                RTS
        175117     071
     14                                      ;
     15 175120                        A.ALDR: LDA A   CRAL+A.6821     ;GET CURRENT CONTROL
        175120     226     201
     16 175122                                PSH A                   ;AND SAVE
        175122     066
     17 175123                                AND A   #,373           ;MASK DDR BIT
        175123     204     373
     18 175125                                STA A   CRAL+A.6821     ;ACCESS DIRECTION REGISTER
        175125     227     201
     19 175127                                STA B   DDRAL+A.6821    ;SET DIRCTIONS
        175127     327     205
     20 175131                                PUL A                   ;GET SAVED CONTROL
        175131     062
     21 175132                                STA A   CRAL+A.6821     ;AND RESTORE
```

```
    22 175134                           RTS
       175134     071
    23                                  ;
```

A.6821 A-PORT CONTROL ROUTINES

```
                                        ;       CONTROL AND SERVICE VECTOR LOADER
                                        ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
                                        ;       'UP' CONTROL IN <B>, AND
                                        ;       'DN' CONTROL IN <A>
                                        ;
 175135                         A.AHDF: STX     A.APRT          ;SAVE SERVICE ROUTINE ADDRESS
 175135     337     015
 175137                                 STA B   A.APRT+2        ;'UP' CONTROL
 175137     327     017
 175141                                 STA A   A.APRT+3        ;'DN' CONTROL
 175141     227     020
 175143                                 RTS
 175143     071
                                        ;
 175144                         A.ALDF: STX     A.APRT+4        ;SAVE SERVICE ROUTINE ADDRESS
 175144     337     021
 175146                                 STA B   A.APRT+6        ;'UP' CONTROL
 175146     327     023
 175150                                 STA A   A.APRT+7        ;'DN' CONTROL
 175150     227     024
 175152                                 RTS
 175152     071
                                        ;
                                        ;
                                        ;       INTERRUPT CONTROL LOADERS
                                        ;
 175153                         A.AHUP: LDA B   A.APRT+2        ;'UP' CONTROL
 175153     326     017
 175155                                 STA B   CRAH+A.6821
 175155     327     200
 175157                                 RTS
 175157     071
                                        ;
 175160                         A.ALUP: LDA B   A.APRT+6        ;'UP' CONTROL
 175160     326     023
```

```
   175162     327     201
25 175164                              RTS
   175164     071
26                                     ;
27 175165                      A.AHDN: LDA B   A.APRT+3        ;'DN' CONTROL
   175165     326     020
28 175167                              STA B   CRAH+A.6821
   175167     327     200
29 175171                              RTS
   175171     071
30                                     ;
31 175172                      A.ALDN: LDA B   A.APRT+7        ;'DN' CONTROL
   175172     326     024
32 175174                              STA B   CRAL+A.6821
   175174     327     201
33 175176                              RTS
   175176     071
34                                     ;
```

A.6821 B-PORT CONTROL ROUTINES

```
      1                                     .SBTTL  A.6821 B-PORT CONTROL ROUTINES
      2                                     ;
      3                                     ;       DATA DIRECTION LOAD ROUTINES
      4                                     ;       ENTER WITH DIRECTION CODE IN <B>
      5                                     ;
      6 175177                        A.BHDR: LDA A   CRBH+A.6821     ;GET CURRENT CONTROL
        175177     226     202
      7 175201                                PSH A                   ;AND SAVE
        175201     066
      8 175202                                AND A   #,373           ;MASK DDR BIT
        175202     204     373
      9 175204                                STA A   CRBH+A.6821     ;ACCESS DIRECTION REGISTER
        175204     227     202
     10 175206                                STA B   DDRBH+A.6821    ;SET DIRECTIONS
        175206     327     206
     11 175210                                PUL A                   ;GET SAVED CONTROL
        175210     062
     12 175211                                STA A   CRBH+A.6821     ;AND RESTORE
        175211     227     202
     13 175213                                RTS
        175213     071
     14                                      ;
     15 175214                        A.BLDR: LDA A   CRBL+A.6821     ;GET CURRENT CONTROL
        175214     226     203
     16 175216                                PSH A                   ;AND SAVE
        175216     066
     17 175217                                AND A   #,373           ;MASK DDR BIT
        175217     204     373
     18 175221                                STA A   CRBL+A.6821     ;ACCESS DIRECTION REGISTER
        175221     227     203
     19 175223                                STA B   DDRBL+A.6821    ;SET DIRCTIONS
        175223     327     207
     20 175225                                PUL A                   ;GET SAVED CONTROL
        175225     062
     21 175226                                STA A   CRBL+A.6821     ;AND RESTORE
```

```
    22 175230                                RTS
       175230     071
    23                                       ;
```

A.6821 B-PORT CONTROL ROUTINES

```
      1                                     ;       CONTROL AND SERVICE VECTOR LOADER
      2                                     ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
      3                                     ;       'UP' CONTROL IN <B>, AND
      4                                     ;       'DN' CONTROL IN <A>
      5                                     ;
      6 175231                        A.BHDF: STX     A.BPRT          ;SAVE SERVICE ROUTINE ADDRESS
        175231     337     025
      7 175233                                STA B   A.BPRT+2        ;'UP' CONTROL
        175233     327     027
      8 175235                                STA A   A.BPRT+3        ;'DN' CONTROL
        175235     227     030
      9 175237                                RTS
        175237     071
     10                                     ;
     11 175240                        A.BLDF: STX     A.BPRT+4        ;SAVE SERVICE ROUTINE ADDRESS
        175240     337     031
     12 175242                                STA B   A.BPRT+6        ;'UP' CONTROL
        175242     327     033
     13 175244                                STA A   A.BPRT+7        ;'DN' CONTROL
        175244     227     034
     14 175246                                RTS
        175246     071
     15                                     ;
     16                                     ;
     17                                     ;       INTERRUPT CONTROL LOADERS
     18                                     ;
     19 175247                        A.BHUP: LDA B   A.BPRT+2        ;'UP' CONTROL
        175247     326     027
     20 175251                                STA B   CRBH+A.6821
        175251     327     202
     21 175253                                RTS
        175253     071
     22                                     ;
     23 175254                        A.BLUP: LDA B   A.BPRT+6        ;'UP' CONTROL
        175254     326     033
```

```
   175256     327     203
25 175260                              RTS
   175260     071
26                                     ;
27 175261                         A.BHDN: LDA B   A.BPRT+3        ;'DN' CONTROL
   175261     326     030
28 175263                              STA B   CRBH+A.6821
   175263     327     202
29 175265                              RTS
   175265     071
30                                     ;
31 175266                         A.BLDN: LDA B   A.BPRT+7        ;'DN' CONTROL
   175266     326     034
32 175270                              STA B   CRBL+A.6821
   175270     327     203
33 175272                              RTS
   175272     071
34                                     ;
```

A.6821 A-PORT TRANSFERS

```
                                        .SBTTL  A.6821 A-PORT TRANSFERS
                                        ;
                                        ;       BYTE DATA IS TRANSFERED IN <B>
                                        ;       WORD DATA IS TRANSFERED IN <B,A>
                                        ;
175273                          A.AHRD: BRA     2$              ;TEST ENTRY
  175273    040     004
175275                          1$:     LDA B   PRAH+A.6821     ;GET BYTE
  175275    326     204
175277                                  SEC                     ;HAVE BYTE
  175277    015
175300                                  RTS
  175300    071
175301                          2$:     LDA A   CRAH+A.6821     ;CHECK FOR IRQ FLAG
  175301    226     200
175303                                  BIT A   #,300
  175303    205     300
175305                                  BNE     1$              ;BRANCH IF FLAG SET
  175305    046     366
175307                                  CLC                     ;NO DATA
  175307    014
175310                                  RTS
  175310    071
                                        ;
175311                          A.ALRD: BRA     2$              ;TEST ENTRY
  175311    040     004
175313                          1$:     LDA B   PRAL+A.6821     ;GET BYTE
  175313    326     205
175315                                  SEC                     ;HAVE BYTE
  175315    015
175316                                  RTS
  175316    071
175317                          2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRG FLAG
  175317    226     201
175321                                  BIT A   #,300
```

```
 22 175323                              BNE     1$              ;BRANCH IF FLAG SET
    175323     046     366
 23 175325                              CLC                     ;NO DATA
    175325     014
 24 175326                              RTS
    175326     071
 25                                     ;
 26 175327                      A.AWRD: BRA     2$              ;TEST ENTRY
    175327     040     006
 27 175331                      1$:     LDA B   PRAH+A.6821     ;GET HIGH BYTE
    175331     326     204
 28 175333                              LDA A   PRAL+A.6821     ;AND LOW BYTE
    175333     226     205
 29 175335                              SEC                     ;HAVE WORD
    175335     015
 30 175336                              RTS
    175336     071
 31 175337                      2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
    175337     226     201
 32 175341                              BIT A   #,300
    175341     205     300
```

A.6821 A-PORT TRANSFERS

```
     33 175343                              BNE     1$              ;BRANCH IF FLAG SET
        175343     046     364
     34 175345                              CLC                     ;NO DATA
        175345     014
     35 175346                              RTS
        175346     071
     36                                     ;
```

A.6821 A-PORT TRANSFERS

```
      1                                         ;
      2 175347                        A.AHWT: BRA     2$              ;TEST ENTRY
        175347     040     007
      3 175351                        1$:     STA B   PRAH+A.6821     ;STORE BYTE
        175351     327     204
      4 175353                                TST     PRAH+A.6821     ;CLEAR IRQ FLAGS
        175353     175     000     204
      5 175356                                SEC                     ;BYTE SENT
        175356     015
      6 175357                                RTS
        175357     071
      7 175360                        2$:     LDA A   CRAH+A.6821     ;CHECK FOR IRQ FLAG
        175360     226     200
      8 175362                                BIT A   #,300
        175362     205     300
      9 175364                                BNE     1$              ;BRANCH IF FLAG SET
        175364     046     363
     10 175366                                CLC                     ;DATA NOT SENT
        175366     014
     11 175367                                RTS
        175367     071
     12                                         ;
     13 175370                        A.ALWT: BRA     2$              ;TEST ENTRY
        175370     040     007
     14 175372                        1$:     STA B   PRAL+A.6821     ;STORE BYTE
        175372     327     205
     15 175374                                TST     PRAL+A.6821     ;CLEAR IRQ FLAGS
        175374     175     000     205
     16 175377                                SEC                     ;BYTE SENT
        175377     015
     17 175400                                RTS
        175400     071
     18 175401                        2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
        175401     226     201
     19 175403                                BIT A   #,300
```

```
 20 175405                                BNE     1$              ;BRANCH IF FLAG SET
    175405     046     363
 21 175407                                CLC                     ;DATA NOT SENT
    175407     014
 22 175410                                RTS
    175410     071
 23                                       ;
 24 175411                        A.AWWT: BRA     2$              ;TEST ENTRY
    175411     040     014
 25 175413                        1$:     STA B   PRAH+A.6821     ;SEND HIGH BYTE
    175413     327     204
 26 175415                                STA A   PRAL+A.6821     ;SEND LOW BYTE
    175415     227     205
 27 175417                                TST     PRAH+A.6821     ;CLEAR IRQ FLAGS
    175417     175     000     204
 28 175422                                TST     PRAL+A.6821     ;CLEAR IRQ FLAGS
    175422     175     000     205
 29 175425                                SEC                     ;WORD SENT
    175425     015
 30 175426                                RTS
    175426     071
```

A.6821 A-PORT TRANSFERS

```
     31 175427                          2$:     PSH A                   ;SAVE A
        175427     066
     32 175430                                  LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
        175430     226     201
     33 175432                                  BIT A   #,300
        175432     205     300
     34 175434                                  PUL A                   ;RESTORE A
        175434     062
     35 175435                                  BNE     1$              ;BRANCH IF FLAG SET
        175435     046     354
     36 175437                                  CLC                     ;DATA NOT SENT
        175437     014
     37 175440                                  RTS
        175440     071
     38                                         ;
```

A.6821 B-PORT TRANSFERS

```
                                        .SBTTL  A.6821 B-PORT TRANSFERS
                                        ;
                                        ;       BYTE DATA IS TRANSFERED IN <B>
                                        ;       WORD DATA IS TRANSFERED IN <B,A>
                                        ;
175441                          A.BHRD: BRA     2$              ;TEST ENTRY
  175441    040     006
175443                          1$:     LDA B   PRBH+A.6821     ;GET BYTE
  175443    326     206
175445                                  STA B   PRBH+A.6821     ;DO CONTROL
  175445    327     206
175447                                  SEC                     ;HAVE BYTE
  175447    015
175450                                  RTS
  175450    071
175451                          2$:     LDA A   CRBH+A.6821     ;CHECK FOR IRQ FLAG
  175451    226     202
175453                                  BIT A   #,300
  175453    205     300
175455                                  BNE     1$              ;BRANCH IF FLAG SET
  175455    046     364
175457                                  CLC                     ;NO DATA
  175457    014
175460                                  RTS
  175460    071
                                        ;
175461                          A.BLRD: BRA     2$              ;TEST ENTRY
  175461    040     006
175463                          1$:     LDA B   PRBL+A.6821     ;GET BYTE
  175463    326     207
175465                                  STA B   PRBL+A.6821     ;DO CONTROL
  175465    327     207
175467                                  SEC                     ;HAVE BYTE
  175467    015
175470                                  RTS
```

```
 22 175471                      2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRG FLAG
    175471     226     203
 23 175473                              BIT A   #,300
    175473     205     300
 24 175475                              BNE     1$              ;BRANCH IF FLAG SET
    175475     046     364
 25 175477                              CLC                     ;NO DATA
    175477     014
 26 175500                              RTS
    175500     071
 27                                     ;
 28 175501                      A.BWRD: BRA     2$              ;TEST ENTRY
    175501     040     012
 29 175503                      1$:     LDA B   PRBH+A.6821     ;GET HIGH BYTE
    175503     326     206
 30 175505                              LDA A   PRBL+A.6821     ;AND LOW BYTE
    175505     226     207
 31 175507                              STA B   PRBH+A.6821     ;DO CONTROL
    175507     327     206
 32 175511                              STA A   PRBL+A.6821     ;DO CONTROL
    175511     227     207
```

A.6821 B-PORT TRANSFERS

```
     33 175513                              SEC                     ;HAVE WORD
        175513     015
     34 175514                              RTS
        175514     071
     35 175515                         2$:  LDA A   CRBL+A.6821      ;CHECK FOR IRQ FLAG
        175515     226     203
     36 175517                              BIT A   #,300
        175517     205     300
     37 175521                              BNE     1$              ;BRANCH IF FLAG SET
        175521     046     360
     38 175523                              CLC                     ;NO DATA
        175523     014
     39 175524                              RTS
        175524     071
     40                                     ;
```

A.6821 B-PORT TRANSFERS

```
      1                                         ;
      2 175525                        A.BHWT: BRA     2$              ;TEST ENTRY
        175525     040     007
      3 175527                        1$:     TST     PRBH+A.6821     ;CLEAR IRQ FLAGS
        175527     175     000     206
      4 175532                                STA B   PRBH+A.6821     ;STORE BYTE
        175532     327     206
      5 175534                                SEC                     ;BYTE SENT
        175534     015
      6 175535                                RTS
        175535     071
      7 175536                        2$:     LDA A   CRBH+A.6821     ;CHECK FOR IRQ FLAG
        175536     226     202
      8 175540                                BIT A   #,300
        175540     205     300
      9 175542                                BNE     1$              ;BRANCH IF FLAG SET
        175542     046     363
     10 175544                                CLC                     ;DATA NOT SENT
        175544     014
     11 175545                                RTS
        175545     071
     12                                         ;
     13 175546                        A.BLWT: BRA     2$              ;TEST ENTRY
        175546     040     007
     14 175550                        1$:     TST     PRBL+A.6821     ;CLEAR IRQ FLAGS
        175550     175     000     207
     15 175553                                STA B   PRBL+A.6821     ;STORE BYTE
        175553     327     207
     16 175555                                SEC                     ;BYTE SENT
        175555     015
     17 175556                                RTS
        175556     071
     18 175557                        2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
        175557     226     203
     19 175561                                BIT A   #,300
```

```
 20 175563                              BNE     1$              ;BRANCH IF FLAG SET
    175563     046     363
 21 175565                              CLC                     ;DATA NOT SENT
    175565     014
 22 175566                              RTS
    175566     071
 23                                     ;
 24 175567                      A.BWWT: BRA     2$              ;TEST ENTRY
    175567     040     014
 25 175571                      1$:     TST     PRBH+A.6821     ;CLEAR IRQ FLAGS
    175571     175     000     206
 26 175574                              TST     PRBL+A.6821     ;CLEAR IRQ FLAGS
    175574     175     000     207
 27 175577                              STA B   PRBH+A.6821     ;SEND HIGH BYTE
    175577     327     206
 28 175601                              STA A   PRBL+A.6821     ;SEND LOW BYTE
    175601     227     207
 29 175603                              SEC                     ;WORD SENT
    175603     015
 30 175604                              RTS
    175604     071
```

A.6821 B-PORT TRANSFERS

```
     31 175605                          2$:     PSH A                   ;SAVE A
        175605     066
     32 175606                                  LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
        175606     226     203
     33 175610                                  BIT A   #,300
        175610     205     300
     34 175612                                  PUL A                   ;RESTORE A
        175612     062
     35 175613                                  BNE     1$              ;BRANCH IF FLAG SET
        175613     046     354
     36 175615                                  CLC                     ;DATA NOT SENT
        175615     014
     37 175616                                  RTS
        175616     071
     38                                         ;
```

A.6821 INTERRUPT SERVICE DISPATCH ROUTINES

```
                                        .SBTTL  A.6821 INTERRUPT SERVICE DISPATCH ROUTINES
                                        ;
175617                          A.AH$I: LDX     A.APRT          ;GET SERVICE ADDRESS
  175617     336     015
175621                                  BNE     A.IRQG          ;BRANCH IF DEFINED
  175621     046     040
175623                                  LDX     #,CRAH+A.6821   ;INTERRUPT CONTROL ADDRESS
  175623     316     000     200
175626                                  BRA     A.IRQD          ;TERMINATE INTERRUPTS
  175626     040     036
                                        ;
175630                          A.AL$I: LDX     A.APRT+4        ;GET SERVICE ADDRESS
  175630     336     021
175632                                  BNE     A.IRQG          ;BRANCH IF DEFINED
  175632     046     027
175634                                  LDX     #,CRAL+A.6821   ;INTERRUPT CONTROL ADDRESS
  175634     316     000     201
175637                                  BRA     A.IRQD          ;TERMINATE INTERRUPTS
  175637     040     025
                                        ;
175641                          A.BH$I: LDX     A.BPRT          ;GET SERVICE ADDRESS
  175641     336     025
175643                                  BNE     A.IRQG          ;BRANCH IF DEFINED
  175643     046     016
175645                                  LDX     #,CRBH+A.6821   ;INTERRUPT CONTROL ADDRESS
  175645     316     000     202
175650                                  BRA     A.IRQD          ;TERMINATE INTERRUPTS
  175650     040     014
                                        ;
175652                          A.BL$I: LDX     A.BPRT+4        ;GET SERVICE ADDRESS
  175652     336     031
175654                                  BNE     A.IRQG          ;BRANCH IF DEFINED
  175654     046     005
175656                                  LDX     #,CRBL+A.6821   ;INTERRUPT CONTROL ADDRESS
  175656     316     000     203
```

```
   175661     040     003
22                                 ;
23 175663                  A.IRQG: JSR     0,X             ;DO SERVICE ROUTINE
   175663     255     000
24 175665                          RTI
   175665     073
25                                 ;
26 175666                  A.IRQD: LDA A   0,X             ;GET CONTROL REGISTER
   175666     246     000
27 175670                          AND B   #,376           ;CLEAR CX1 INTERRUPT ENABLE
   175670     304     376
28 175672                          BIT B   #,40            ;CX2 INTERRUPT MODE ?
   175672     305     040
29 175674                          BNE     1$              ;BRANCH IF NOT
   175674     046     002
30 175676                          AND B   #,367           ;CLEAR CX2 INTERRUPT ENABLE
   175676     304     367
31 175700                  1$:     STA B   0,X             ;SET NEW CONTROL
   175700     347     000
32 175702                          RTI
   175702     073
```

SERIAL 'PORT' MACRO V05.06 Thursday 18-Oct-07 19:15 Page 25-1

A.6821 INTERRUPT SERVICE DISPATCH ROUTINES

```
     33                                     ;
     34                                     .IIF    NDF,$A.6821     .END
```

```
                                .SBTTL  A.8250 ASYNCHRONOUS INTERFACE SOFTWARE
                                ;
                                ;         THIS SOFTWARE PACKAGE SUPPORTS THE INS8250
                                ;       ASYNCHRONOUS COMMUNICATIONS ELEMENT.
                                ;       ENTRY POINTS ARE DEFINED HERE FOR EXTERNAL
                                ;       ACCESS (VIA JSR ____):
                                ;
                                ;               A.MOSV --> SAVE 'X' AS 'MODEM' INTERRUPT
                                ;                                  SERVICE ROUTINE ADDRESS
                                ;               A.XMSV --> SAVE 'X' AS 'XMTR' INTERRUPT
                                ;                                  SERVICE ROUTINE ADDRESS
                                ;               A.RCSV --> SAVE 'X' AS 'RCVR' INTERRUPT
                                ;                                  SERVICE ROUTINE ADDRESS
                                ;               A.LNSV --> SAVE 'X' AS 'LINE' INTERRUPT
                                ;                                  SERVICE ROUTINE ADDRESS
                                ;
                                ;               A.MOUP --> ENABLE 'MODEM' INTERRUPT
                                ;               A.XMUP --> ENABLE 'XMTR' INTERRUPT
                                ;               A.RCUP --> ENABLE 'RCVR' INTERRUPT
                                ;               A.LNUP --> ENABLE 'LINE' INTERRUPT
                                ;
                                ;               A.MODN --> DISABLE 'MODEM' INTERRUPT
                                ;               A.XMDN --> DISABLE 'XMTR' INTERRUPT
                                ;               A.RCDN --> DISABLE 'RCVR' INTERRUPT
                                ;               A.LNDN --> DISABLE 'LINE' INTERRUPT
                                ;
                                ;
                                ;         STATUS REGISTER ACCESS TO/FROM <B>
                                ;
                                ;       A.MSRR --> READ MODEM STATUS REGISTER
                                ;       A.MSRW --> WRITE MODEM STATUS REGISTER
                                ;
                                ;       A.MCRR --> READ MODEM CONTROL REGISTER
                                ;       A.MCRW --> WRITE MODEM CONTROL REGISTER
                                ;
```

```
38                                      ;       A.LSRW --> WRITE LINE STATUS REGISTER
39                                      ;
40                                      ;       A.LCRR --> READ LINE CONTROL REGISTER
41                                      ;       A.LCRW --> WRITE LINE CONTROL REGISTER
42                                      ;
43                                      ;       A.DLLR --> READ DLL
44                                      ;       A.DLLW --> WRITE DLL
45                                      ;
46                                      ;       A.DLMR --> READ DLM
47                                      ;       A.DLMW --> WRITE DLM
48                                      ;
49                                      ;
```

A.8250 ASYNCHRONOUS INTERFACE SOFTWARE

```
      1                                         ;
      2                                         ;         THE FOLLOWING ENTRY POINT IS VIA INTERRUPT
      3                                         ;       AND PROVIDES 8250 ACE SERVICES FOR 'MODEM', 'XMTR
',
      4                                         ;       'RCVR', AND 'LINE' INTERRUPTS.
      5                                         ;
      6                                         ;               A.82$I --> MPU INTERRUPT ENTRY POINT
      7                                         ;                                  FOR 8250 ACE
      8                                         ;
      9                                         ;
     10                                         ;         NOTES ON INTERRUPT SERVICE ROUTINES
     11                                         ;
     12                                         ;       MODEM --> READS MODEM STATUS REGISTER (MSR) INTO
<B> AND
     13                                         ;                 IF 'A.MO$V' IS NOT ZERO -
     14                                         ;               THEN - CALLS ROUTINE VIA JSR 0,X(=A.MO$V)
     15                                         ;               ELSE - NOTHING
     16                                         ;
     17                                         ;       XMTR --> IF 'A.XM$V' IS NOT ZERO -
     18                                         ;               THEN - CALLS ROUTINE VIA JSR 0,X(=A.XM$V)
     19                                         ;                       ON RETURN
     20                                         ;                       IF C=1 --> TRANSMITS CHARACTER IN
 <B>
     21                                         ;                       IF C=0 --> DISABLES 'XMTR' INTERR
UPT
     22                                         ;               ELSE - DISABLES 'XMTR' INTERRUPT
     23                                         ;
     24                                         ;       RCVR --> MOVE INPUT CHARACTER INTO <B>
     25                                         ;               IF 'A.RC$V' IS NOT ZERO -
     26                                         ;               THEN - CALLS ROUTINE VIA JSR 0,X(=A.RC$V)
     27                                         ;               ELSE - IGNORES INPUT CHARACTER
     28                                         ;
     29                                         ;       LINE --> READS LINE STATUS REGISTER (LSR) INTO <B
> AND
     30                                         ;                IF 'A.LN$V' IS NOT ZERO -
     31                                         ;               THEN - CALLS ROUTINE VIA JSR 0,X(=A.LN$V)
     32                                         ;               ELSE - NOTHING
     33                                         ;
```

A.8250 ASYNCHRONOUS HARDWARE

```
                                .SBTTL  A.8250 ASYNCHRONOUS HARDWARE
                                ;
                                ; THE ASYNCHRONOUS COMMUNICATION ELEMENT USED BY THE
                                ;PORT IS THE NATIONAL SEMICONDUCTOR TYPE INS8250.
                                ; THE FOLLOWING IS A SUMMARY OF THE REGISTERS AND
                                ;SOME DEFINITIONS.
                                ;
                                ;       DLAB (DATA LATCH ACCES BIT) = 0
                                ;
                                ; REGISTER 0
000000                          RBR     =0      ;RECEIVER BUFFER REGISTER (READ ONLY)
000000                          THR     =0      ;TRANSMITTER HOLDING REGISTER (WRITE ONLY
)
                                ;
                                ; REGISTER 1
000001                          IER     =1      ;INTERRUPT ENABLE REGISTER (READ/WRITE)
                                ;
                                ; REGISTER 2
000002                          IIR     =2      ;INTERRUPT IDENTIFICATION REGISTER
                                ;
                                ; REGISTER 3
000003                          LCR     =3      ;LINE CONTROL REGISTER (READ/WRITE)
                                ;
                                ; REGISTER 4
000004                          MCR     =4      ;MODEM CONTROL REGISTER (READ/WRITE)
                                ;
                                ; REGISTER 5
000005                          LSR     =5      ;LINE CONTROL STATUS REGISTER (READ/WRITE
)
                                ;
                                ; REGISTER 6
000006                          MSR     =6      ;MODEM STATUS REGISTER
                                ;
                                ;
                                ;       DLAB (DATA LATCH ACCESS BIT) = 1
                                ;
```

```
    36         000000                          DLL     =0      ;DIVISOR LATCH LOW BYTE (READ/WRITE)
    37                                         ;
    38                                         ; REGISTER 1
    39         000001                          DLM     =1      ;DIVISOR LATCH HIGH BYTE (READ/WRITE)
    40                                         ;
```

A.8250 DETAILS

```
      1                                 .SBTTL  A.8250 DETAILS
      2                                 ;
      3                                 ; RECEIVER BUFFER REGISTER (RBR) - 8 BIT READ ONLY REGIST
ER
      4                                 ;               CONTAINING DATA RECEIVED
      5                                 ;
      6                                 ; TRANSMITTER HOLDING REGISTER (THR) - 8 BIT WRITE ONLY R
EGISTER
      7                                 ;               CONTAINING CHARACTER TO BE TRANSMITTED
      8                                 ;
      9                                 ; INTERRUPT ENABLE REGISTER (IER)
     10                                 ; BIT 0         ENABLE RECEIVED DATA AVAILABLE INTERRUPT
     11                                 ;     1         ENABLE TRANSMITTER HOLDING REGISTER EMPTY
 INTERRUPT
     12                                 ;     2         ENABLE RECEIVER LINE STATUS INTERRUPT
     13                                 ;     3         ENABLE MODEM STATUS INTERRUPT
     14                                 ;     4-7       0'S
     15                                 ;
     16                                 ; INTERRUPT IDENTIFICATION REGISTER (IIR)
     17                                 ; BIT 0         '0' IF INTERRUPT PENDING
     18                                 ; BIT 1 2
     19                                 ;     _ _
     20                                 ;     0 0       MODEM STATUS INTERRUPT (LOWEST PRIORITY)
     21                                 ;     1 0       TRANSMITTER HOLDING REGISTER EMPTY
     22                                 ;     0 1       RECEIVER DATA AVAILABLE
     23                                 ;     1 1       RECEIVER LINE STATUS (HIGHEST PRIORITY)
     24                                 ;
     25                                 ; LINE CONTROL REGISTER (LCR)
     26                                 ; BIT 0 1       WORD LENGTH SELECT BITS
     27                                 ;     _ _
     28                                 ;     0 0       5 BITS
     29                                 ;     1 0       6 BITS
     30                                 ;     0 1       7 BITS
     31                                 ;     1 0       8 BITS
     32                                 ;
     33                                 ; BIT 2         STOP BIT SELECT
     34                                 ;     _
```

```
     35                                 ;     0         1 STOP BIT
     36                                 ;     1         1.5 STOP BITS (5 BITS) / 2 STOP BITS
     37                                 ;
     38                                 ; BIT 3 4 5     PARITY SELECT
     39                                 ;     _ _ _
     40                                 ;     0 X X     NO PARITY / NO STICK
     41                                 ;     1 0 0     ODD PARITY
     42                                 ;     1 1 0     EVEN PARITY
     43                                 ;     1 0 1     PARITY = STICK 1
     44                                 ;     1 1 1     PARITY = STICK 0
     45                                 ;
     46                                 ; BIT 6         BREAK CONTROL BIT
     47                                 ;     _
     48                                 ;     0         NORMAL OPERATION
     49                                 ;     1         SERIAL OUTPUT FORCED TO SPACING
     50                                 ;
     51                                 ; BIT 7         DIVISOR LATCH ACCESS BIT
     52                                 ;     _
     53                                 ;     0         ACCESS RECEIVER BUFFER / TRANSMITTER HOLD
ING REGISTERS
     54                                 ;     1         ACCESS DIVISOR LATCH REGISTERS
     55                                 ;
```

A.8250 DETAILS

```
      1                                 ;
      2                                 ; MODEM CONTROL REGISTER (MCR)
      3                                 ; BIT 0         DATA TERMINAL READY CONTROL (DSR OUTPUT)
      4                                 ; BIT 1         REQUEST TO SEND CONTROL (CTS OUTPUT)
      5                                 ; BIT 2         OUT 1 CONTROL
      6                                 ; BIT 3         OUT 2 CONTROL
      7                                 ; BIT 4         LOOP CONTROL
      8                                 ;               ENABLES LOOP BACK MODE AND CONNECTS THE F
OLLOWING -
      9                                 ;               DTR >> DSR
     10                                 ;               RTS >> CTS
     11                                 ;               OUT1 >> RI
     12                                 ;               OUT2 >> RLSD
     13                                 ;               SO >> SI
     14                                 ; BIT 5-7       0'S
     15                                 ;
     16                                 ; LINE STATUS REGISTER (LSR)
     17                                 ; BIT 0         DATA READY
     18                                 ; BIT 1         OVERRUN ERROR
     19                                 ; BIT 2         PARITY ERROR
     20                                 ; BIT 3         FRAMING ERROR
     21                                 ; BIT 4         BREAK INTERRUPT
     22                                 ; BIT 5         TRANSMITTER HOLDING REGISTER EMPTY
     23                                 ; BIT 6         TRANSMITTER SHIFT REGISTER EMPTY
     24                                 ; BIT 7         0
     25                                 ;
     26                                 ; MODEM STATUS REGISTER (MSR)
     27                                 ; BIT 0         DELTA CLEAR TO SEND
     28                                 ; BIT 1         DELTA DATA SET READY
     29                                 ; BIT 2         TRAILING EDGE RING INDICATOR
     30                                 ; BIT 3         DELTA LINE SIGNAL DETECT
     31                                 ; BIT 4         CLEAR TO SEND (RTS INPUT)
     32                                 ; BIT 5         DATA SET READY (DTR INPUT)
     33                                 ; BIT 6         RING INDICATOR
     34                                 ; BIT 7         RECEIVED LINE SIGNAL DETECT (SIN INPUT)
     35                                 ;
```

```
                                        ; DIVISOR LATCH (DLL/DLM)
                                        ; 16-BIT DIVISOR FOR BAUD RATE SELECTION
                                        ; BAUD  COUNT  TIME     @ 1.8432 MHZ
                                        ; ____  _____  ____
                                        ;  110   1047  9091
                                        ;  150    768  6667
                                        ;  300    384  3333
                                        ;  600    192  1667
                                        ; 1200     96   833
                                        ; 1800     64   556
                                        ; 2400     48   417
                                        ; 4800     24   208
                                        ; 9600     12   104
                                        ;
```

A.8250 TYPICAL RS-232C WIRING CONNECTIONS

```
                                        .SBTTL  A.8250 TYPICAL RS-232C WIRING CONNECTIONS
                                        ;
                                        ;
                                        ;       1       GROUND
                                        ;       2       RX      (INPUT)
                                        ;       3       TX      (OUTPUT)
                                        ;       4       RTS     (INPUT)
                                        ;       5       CTS     (OUTPUT)
                                        ;       6       DSR     (OUTPUT)
                                        ;       7       GROUND
                                        ;       8       OUT1    (OUTPUT)
                                        ;       9       +12 VOLTS SOURCE
                                        ;       10      -12 VOLTS SOURCE
                                        ;       11
                                        ;       12
                                        ;       13
                                        ;       14
                                        ;       15
                                        ;       16
                                        ;       17
                                        ;       18      RLSD    (INPUT)
                                        ;       19      RI      (INPUT)
                                        ;       20      DTR     (INPUT)
                                        ;       21
                                        ;       22      OUT2    (OUTPUT)
                                        ;       23
                                        ;       24
                                        ;       25
                                        ;
                                        ;       JUMPER RSLD(18) TO RX(2) FOR AUTO BAUD-RATE
                                        ;
```

A.8250 VARIABLES AND POINTERS

```
                                        .SBTTL  A.8250 VARIABLES AND POINTERS
                                        ;
                                        ;       PORT STATUS AND BUFFERS
                                        ;
                                        .IF     NDF,$A.8250
                                        A.8250  =       220     ;SERIAL PORT DEFAULT ADDRESS
                                        VARSAV  =       0       ;DEFAULT VARIABLES ADDRESS
                                        PGMSAV  =       100000  ;DEFAULT PROGRAM ADDRESS
                                        .=PGMSAV
                                        .ENDC
                                        ;
         175703                         PGMSAV=.
         000035                         .=VARSAV
                                        ;
 000035     000     000         A.MO$V: .BYTE   0,0     ;MODEM DISPATCH VECTOR
 000037     000     000         A.XM$V: .BYTE   0,0     ;XMTR DISPATCH VECTOR
 000041     000     000         A.RC$V: .BYTE   0,0     ;RCVR DISPATCH VECTOR
 000043     000     000         A.LN$V: .BYTE   0,0     ;LINE DISPATCH VECTOR
                                        ;
 000045     000                 A.LNST: .BYTE   0       ;LINE STATUS FLAG
                                        ;               = 0     NO PROCESS PENDING
                                        ;               =-1     REQUIRES BAUD RATE PROCESSING
 000046     000                 A.ERCT: .BYTE   0       ;LINE ERROR COUNTER
 000047     000                 A.LNBL: .BYTE   0       ;'LINE' SERVICE ENABLES
 000050     000                 A.MNBL: .BYTE   0       ;'MODEM' SERVICE ENABLES
                                        ;
 000051     000     000         A.PBCT: .BYTE   0,0     ;PREVIOUS BAUD COUNT
 000053     000     000         A.CBCT: .BYTE   0,0     ;CURRENT BAUD COUNT
 000055     000     000         A.BDIF: .BYTE   0,0     ;COMPUTED DIFFERENCE
 000057     000     000         A.LWCT: .BYTE   0,0     ;LOWEST DIFFERENCE FOUND
 000061     000     000         A.BDVS: .BYTE   0,0     ;BAUD RATE DIVISOR
                                        ;
         000063                         VARSAV=.
         175703                         .=PGMSAV
                                        ;
```

A.8250 INTERRUPT CONTROL ROUTINES

```
      1                                     .SBTTL  A.8250 INTERRUPT CONTROL ROUTINES
      2                                     ;
      3 175703                      A.MOSV: STX     A.MO$V          ;SAVE 'MODEM' VECTOR ADDRESS
        175703     337     035
      4 175705                             RTS
        175705     071
      5                                     ;
      6 175706                      A.MOUP: LDA A   IER+A.8250      ;ENABLE MODEM INTERRUPT
        175706     226     221
      7 175710                             ORA A   #,10
        175710     212     010
      8 175712                             STA A   IER+A.8250
        175712     227     221
      9 175714                             RTS
        175714     071
     10                                     ;
     11 175715                      A.MODN: LDA A   IER+A.8250      ;DISABLE INTERRUPT
        175715     226     221
     12 175717                             AND A   #,7
        175717     204     007
     13 175721                             STA A   IER+A.8250
        175721     227     221
     14 175723                             RTS
        175723     071
     15                                     ;
     16 175724                      A.XMSV: STX     A.XM$V          ;SAVE 'XMTR' VECTOR ADDRESS
        175724     337     037
     17 175726                             RTS
        175726     071
     18                                     ;
     19 175727                      A.XMUP: LDA A   IER+A.8250      ;ENABLE XMTR INTERRUPT
        175727     226     221
     20 175731                             ORA A   #,2
        175731     212     002
     21 175733                             STA A   IER+A.8250
```

```
 22 175735                                RTS
    175735     071
 23                                       ;
 24 175736                      A.XMDN: LDA A   IER+A.8250      ;DISABLE INTERRUPT
    175736     226     221
 25 175740                                AND A   #,15
    175740     204     015
 26 175742                                STA A   IER+A.8250
    175742     227     221
 27 175744                                RTS
    175744     071
 28                                       ;
```

A.8250 INTERRUPT CONTROL ROUTINES

```
      1                                         ;
      2 175745                        A.RCSV: STX     A.RC$V          ;SAVE 'RCVR' VECTOR ADDRESS
        175745     337     041
      3 175747                                RTS
        175747     071
      4                                         ;
      5 175750                        A.RCUP: LDA A   IER+A.8250      ;ENABLE RCVR INTERRUPT
        175750     226     221
      6 175752                                ORA A   #,1
        175752     212     001
      7 175754                                STA A   IER+A.8250
        175754     227     221
      8 175756                                RTS
        175756     071
      9                                         ;
     10 175757                        A.RCDN: LDA A   IER+A.8250      ;DISABLE INTERRUPT
        175757     226     221
     11 175761                                AND A   #,16
        175761     204     016
     12 175763                                STA A   IER+A.8250
        175763     227     221
     13 175765                                RTS
        175765     071
     14                                         ;
     15 175766                        A.LNSV: STX     A.LN$V          ;SAVE 'LINE' INTERRUPT SERVICE
        175766     337     043
     16 175770                                RTS
        175770     071
     17                                         ;
     18 175771                        A.LNUP: LDA A   IER+A.8250      ;AND ENABLE LINE INTERRUPT
        175771     226     221
     19 175773                                ORA A   #,4
        175773     212     004
     20 175775                                STA A   IER+A.8250
        175775     227     221
```

A.8250 STATUS REGISTER ACCESS

```
                                        .SBTTL  A.8250 STATUS REGISTER ACCESS
                                        ;
                                        ;         ACCESS TO THE 8250 STATUS REGISTERS
                                        ;
                                        ;       MCR IS THE OUTPUT REGISTER
                                        ;               DATA FROM <B>
                                        ;
                                        ;       MSR IS THE INPUT REGISTER
                                        ;               DATA PLACED IN <B>
                                        ;
                                        ;
176007                          A.MSRR: LDA B   MSR+A.8250      ;GET MODEM STATUS
176007  326     226
                                        RTS
176011  071
                                        ;
176012                          A.MSRW: STA B   MSR+A.8250      ;WRITE TO STATUS
176012  327     226
                                        RTS
176014  071
                                        ;
176015                          A.MCRR: LDA B   MCR+A.8250      ;READ CONTROL
176015  326     224
                                        RTS
176017  071
                                        ;
176020                          A.MCRW: STA B   MCR+A.8250      ;STORE CONTROL
176020  327     224
                                        RTS
176022  071
                                        ;
176023                          A.LSRR: LDA B   LSR+A.8250      ;READ STATUS
176023  326     225
                                        RTS
176025  071
```

A.8250 STATUS REGISTER ACCESS

```
     39 176044                              STA A   LCR+A.8250
        176044     227     223
     40 176046                              LDA B   DLL+A.8250      ;READ DLL
        176046     326     220
     41 176050                              PUL A                   ;RESTORE LCR
        176050     062
     42 176051                              STA A   LCR+A.8250
        176051     227     223
     43 176053                              RTS
        176053     071
     44                                     ;
     45 176054                      A.DLLW: LDA A   LCR+A.8250      ;SET CONTROL TO ACCESS DLL
        176054     226     223
     46 176056                              PSH A                   ;SAVE
        176056     066
     47 176057                              ORA A   #,200
        176057     212     200
     48 176061                              STA A   LCR+A.8250
        176061     227     223
     49 176063                              STA B   DLL+A.8250      ;WRITE DLL
        176063     327     220
     50 176065                              PUL A                   ;RESTORE LCR
        176065     062
     51 176066                              STA A   LCR+A.8250
        176066     227     223
     52 176070                              RTS
        176070     071
     53                                     ;
     54 176071                      A.DLMR: LDA A   LCR+A.8250      ;SET CONTROL TO ACCESS DLM
        176071     226     223
     55 176073                              PSH A                   ;SAVE
        176073     066
     56 176074                              ORA A   #,200
        176074     212     200
     57 176076                              STA A   LCR+A.8250
```

A.8250 STATUS REGISTER ACCESS

```
     69 176120                                STA A   LCR+A.8250
        176120     227     223
     70 176122                                RTS
        176122     071
     71                                       ;
```

A.8250 INTERRUPT HANDLER

```
                                        .SBTTL  A.8250 INTERRUPT HANDLER
                                        ;
176123                          A.82$I: LDA A   IIR+A.8250      ;GET INTERRUPT ID
  176123    226     222
176125                                  BIT A   #,1             ;INTERRUPT PENDING ?
  176125    205     001
176127                                  BNE     1$              ;IF NOT - LEAVE
  176127    046     013
176131                                  TAB                     ;COMPUTE JUMP TABLE ENTRY ADDRESS
  176131    026
176132                                  ASR B
  176132    127
176133                                  ABA
  176133    033
176134                                  CLR B
  176134    137
176135                                  ADD A   #,2$&377
  176135    213     145
176137                                  ADC B   #,2$&177400/400
  176137    311     374
176141                                  PSH A
  176141    066
176142                                  PSH B
  176142    067
176143                                  RTS                     ;GO TO JUMP TABLE
  176143    071
176144                          1$:     RTI                     ;RETURN FROM INTERRUPT
  176144    073
                                        ;
176145                          2$:     JMP     A.MODE          ;MODEM STATUS INTERRUPT
  176145    176     374     252
176150                                  JMP     A.XMTR          ;TRANSMITTER INTERRUPT
  176150    176     374     161
176153                                  JMP     A.RCVR          ;RECEIVER INTERRUPT
  176153    176     374     212
```

A.8250 XMTR & RCVR INTERRUPT DISPATCHERS

```
      1                                         .SBTTL  A.8250 XMTR & RCVR INTERRUPT DISPATCHERS
      2                                         ;
      3 176161                          A.XMTR: LDA B   LSR+A.8250      ;NEED A CHARACTER ?
        176161     326     225
      4 176163                                  BIT B   #,40
        176163     305     040
      5 176165                                  BEQ     2$              ;BRANCH IF NOT
        176165     047     020
      6 176167                                  LDX     A.XM$V          ;XMTR VECTOR
        176167     336     037
      7 176171                                  BEQ     1$              ;BETTER HAVE AN ADDRESS !
        176171     047     011
      8 176173                                  JSR     0,X             ;GO TO ROUTINE
        176173     255     000
      9 176175                                  BCC     1$              ;IF NO CHARACTERS (C=0) THEN TERM
INATE
        176175     044     005
     10 176177                                  STA B   THR+A.8250      ;ELSE SEND NEW CHARACTER
        176177     327     220
     11 176201                                  JMP     A.82$I          ;GO CHECK FOR OTHER INTERRUPTS
        176201     176     374     123
     12 176204                          1$:     JSR     A.XMDN          ;KILL TRANSMITTER
        176204     275     373     336
     13 176207                          2$:     JMP     A.82$I          ;AND CHECK FOR OTHER INTERRUPTS
        176207     176     374     123
     14                                         ;
     15 176212                          A.RCVR: LDA B   LSR+A.8250      ;HAVE A CHARACTER ?
        176212     326     225
     16 176214                                  BIT B   #,1
        176214     305     001
     17 176216                                  BEQ     1$              ;BRANCH IF NOT
        176216     047     010
     18 176220                                  LDA B   RBR+A.8250      ;PLACE CHARACTER IN <B>
        176220     326     220
     19 176222                                  LDX     A.RC$V          ;RCVR VECTOR
        176222     336     041
```

A.8250 LINE AND MODEM INTERRUPT DISPATCHERS

```
                                        .SBTTL  A.8250 LINE AND MODEM INTERRUPT DISPATCHERS
                                        ;
176233                          A.LINE: LDA B   LSR+A.8250      ;GET STATUS, CLEAR FLAGS
  176233    326     225
176235                                  BIT B   #,36            ;SOMETHING TO CHECK ?
  176235    305     036
176237                                  BEQ     1$              ;BRANCH IF NOT
  176237    047     006
176241                                  LDX     A.LN$V          ;EXTERNAL VECTOR ?
  176241    336     043
176243                                  BEQ     1$              ;BR IF NOT
  176243    047     002
176245                                  JSR     0,X             ;GO TO ROUTINE
  176245    255     000
176247                              1$: JMP     A.82$I          ;CHECK FOR OTHER INTERRUPTS
  176247    176     374     123
                                        ;
176252                          A.MODE: LDA B   MSR+A.8250      ;GET MODEM STATUS, CLEAR FLAGS
  176252    326     226
176254                                  BIT B   #,17            ;SOMETHING TO CHECK ?
  176254    305     017
176256                                  BEQ     1$              ;BRANCH IF NOT
  176256    047     006
176260                                  LDX     A.MO$V          ;EXTERNAL VECTOR ?
  176260    336     035
176262                                  BEQ     1$              ;BR IF NOT
  176262    047     002
176264                                  JSR     0,X             ;GO TO ROUTINE
  176264    255     000
176266                              1$: JMP     A.82$I          ;CHECK FOR OTHER INTERRUPTS
  176266    176     374     123
                                        ;
```

A.8250 STANDARD LINE / MODEM SETTUP

```
                                .SBTTL  A.8250 STANDARD LINE / MODEM SETTUP
                                ;
                                ;         USING ROUTINES A.82$L AND A.82$M
                                ;       THE ASYNCHRONOUS INTERFACE IS PROGRAMMED
                                ;       INITIALLY FOR -
                                ;               1 START BIT
                                ;               7 DATA BITS
                                ;               1 ODD PARITY BIT
                                ;               2 STOP BITS
                                ;
                                ;       ON A 'BREAK' INTERRUPT THE I/O
                                ;       IS RESET TO THE ABOVE PARAMETERS
                                ;       AND THE BAUD RATE IS DETERMINED DURING THE
                                ;       NEXT CHARACTER
                                ;
                                ;       ON A 'PARITY' INTERRUPT THE I/O
                                ;       IS SEQUENCED TO THE NEXT POSSIBLE PARITY
                                ;       CONFIGURATION - THESE ARE:
                                ;               ODD PARITY
                                ;               EVEN PARITY
                                ;               STICK '1' PARITY
                                ;               STICK '0' PARITY
                                ;
                                ;       ON A 'FRAMING' INTERRUPT THE I/O
                                ;       IS SEQUENCED TO THE NEXT STOP BIT
                                ;       CONFIGURATION - 1 OR 2 STOPS
                                ;
                                ;       IF MORE THAN 20. ERRORS OCCUR THEN A
                                ;       'BREAK' INTERRUPT SEQUENCE IS PERFORMED
                                ;       TO INITIALIZE THE HARDWARE
                                ;
                                ;
                                ;
                                ;         WITH AN EXTERNAL DEVICE ATTACHED TO THE PORT -
                                ;       SELECT ONE OF THE FOLLOWING BAUD RATES -
```

A.8250 BAUD RATE TABLE

```
      1                                     .SBTTL  A.8250 BAUD RATE TABLE
      2                                     ;
      3 176271                        A.BTBL: FDB     8.,12.          ;9600 BAUD
        176271     000     010
        176273     000     014
      4 176275                             FDB     15.,24.         ;4800 BAUD
        176275     000     017
        176277     000     030
      5 176301                             FDB     25.,48.         ;2400 BAUD
        176301     000     031
        176303     000     060
      6 176305                             FDB     35.,64.         ;1800 BAUD
        176305     000     043
        176307     000     100
      7 176311                             FDB     66.,96.         ;1200 BAUD
        176311     000     102
        176313     000     140
      8 176315                             FDB     132.,192.       ; 600 BAUD
        176315     000     204
        176317     000     300
      9 176321                             FDB     232.,384.       ; 300 BAUD
        176321     000     350
        176323     001     200
     10 176325                             FDB     402.,768.       ; 150 BAUD
        176325     001     222
        176327     003     000
     11 176331                             FDB     32767.,1047.    ; 110 BAUD
        176331     177     377
        176333     004     027
     12                                     ;
```

A.8250 LINE STATUS HANDLER

```
 1                                      .SBTTL  A.8250 LINE STATUS HANDLER
 2                                      ;
 3                                      ;         THIS IS THE STANDARD LINE STATUS ROUTINE
 4                                      ;       AND MAY BE INVOKED BY SETTING A.LN$V=A.82$L.
 5                                      ;         EACH LINE INTERRUPT FLAG IS COMPARED TO THOSE
 6                                      ;       ENABLED CHECKS DEFINED IN A.LNBL (BIT FOR BIT).
 7                                      ;         ENTER WITH LSR IN <B>, B IS DESTROYED
 8                                      ;
 9                                      ;
10 176335                       A.82$L: AND B   A.LNBL          ;MASK WITH ENABLES
   176335     324     047
11 176337                               BIT B   #,20            ;A 'BREAK' ?
   176337     305     020
12 176341                               BEQ     A.PRTY          ;IF NOT - CHECK PARITY
   176341     047     015
13 176343                       A.LBRK: LDA B   #,16            ;RESET HARDWARE
   176343     306     016
14 176345                               STA B   LCR+A.8250
   176345     327     223
15 176347                               LDA B   #,-1            ;SET UP A.LNST
   176347     306     377
16 176351                               STA B   A.LNST          ;TO CHECK BAUD RATE
   176351     327     045
17 176353                               LDA B   #,21.           ;ERROR COUNT PRESET
   176353     306     025
18 176355                               STA B   A.ERCT          ;SAVE IN STATUS TABLE
   176355     327     046
19 176357                               RTS
   176357     071
20                                      ;
21 176360                       A.PRTY: BIT B   #,4             ;A 'PARITY' ?
   176360     305     004
22 176362                               BEQ     A.FRAM          ;IF NOT - CHECK FRAME
   176362     047     016
23 176364                               LDA A   LCR+A.8250      ;GET LINE CONTROL REGISTER
```

A.8250 LINE STATUS HANDLER

```
     35 176410                                EOR B   #,4             ;FLIP/FLOP STOP BIT SELECT
        176410     310     004
     36 176412                                STA B   LCR+A.8250      ;SAVE AT I/O
        176412     327     223
     37 176414                          A.LERR: DEC     A.ERCT          ;ZERO YET ?
        176414     172     000     046
     38 176417                                BLE     A.LBRK          ;IF SO - RESET HARDWARE
        176417     057     322
     39 176421                          A.OVRN:
     40 176421                          A.LEND: RTS
        176421     071
     41                                       ;
```

A.8250 MODEM STATUS HANDLER

```
      1                                     .SBTTL  A.8250 MODEM STATUS HANDLER
      2                                     ;
      3                                     ;         THIS IS THE STANDARD A.8250 MODEM STATUS
      4                                     ;       HANDLER. IT MAY BE ACCESSED BY SETTING A.MO$V=A.8
2$M.
      5                                     ;         ONLY THE RLSD BIT IS USED BY THIS ROUTINE
      6                                     ;       FOR BAUD-RATE DETERMINATION. AND IS ENABLED
      7                                     ;       BY SETTING THE CORRESPONDING BIT IN A.MNBL.
      8                                     ;         ENTER WITH MSR IN <B>, B IS DESTROYED
      9                                     ;
     10                                     ;
     11 176422                      A.MEND: RTS
        176422     071
     12 176423                      A.82$M: TBA                     ;COPY B TO A
        176423     027
     13 176424                              AND B   A.MNBL          ;MASK WITH ENABLES
        176424     324     050
     14 176426                              BIT B   #,10            ;BAUD RATE ENABLED ?
        176426     305     010
     15 176430                              BEQ     A.MEND          ;EXIT IF NOT
        176430     047     370
     16 176432                              BIT A   #,200           ;FIRST BIT OF CHARACTER ?
        176432     205     200
     17 176434                              BEQ     A.MEND          ;IF NOT - LEAVE
        176434     047     364
     18 176436                              LDA B   A.LNST          ;A BAUD RATE CHECK ?
        176436     326     045
     19 176440                              BEQ     A.MEND          ;IF NOT - LEAVE
        176440     047     360
     20 176442                              CLR     A.LNST          ;CLEAR STATUS
        176442     177     000     045
     21 176445                              LDX     #,-6000.        ;INITIALIZE COUNTER TO
        176445     316     350     220
     22 176450                              STX     A.CBCT          ;ABOUT 12 BIT TIMES @ 110 BAUD
        176450     337     053
     23 176452                              LDX     #,32767.        ;PRESET LOW COUNT
```

A.8250 MODEM STATUS HANDLER

```
     35 176503                              STX     A.PBCT          ;SAVE IN PREVIOUS COUNT
        176503     337     051
     36                                     ;
     37 176505                        4$:   LDA A   MSR+A.8250      ;GET STATUS
        176505     226     226
     38 176507                              BIT A   #,10            ;CHECK FOR TRANSITION
        176507     205     010
     39 176511                              BEQ     6$              ;IF NONE - SKIP
        176511     047     046
     40 176513                              LDA A   A.CBCT+1        ;GET CURRENT COUNT
        176513     226     054
     41 176515                              LDA B   A.CBCT
        176515     326     053
     42 176517                              SUB A   A.PBCT+1        ;COMPUTE DIFFERENCE
        176517     220     052
     43 176521                              SBC B   A.PBCT          ;WITH PREVIOUS COUNT
        176521     322     051
     44 176523                              STA A   A.BDIF+1        ;SAVE DIFFERENCE
        176523     227     056
     45 176525                              STA B   A.BDIF
        176525     327     055
     46 176527                              SUB A   A.LWCT+1        ;DIFFERENCE LESS THAN
        176527     220     060
     47 176531                              SBC B   A.LWCT          ;CURRENT LOW COUNT ?
        176531     322     057
     48 176533                              BPL     5$              ;IF NOT - SKIP
        176533     052     004
     49 176535                              LDX     A.BDIF          ;ELSE SAVE AS NEW LOW COUNT
        176535     336     055
     50 176537                              STX     A.LWCT
        176537     337     057
     51 176541                        5$:   LDA A   A.CBCT+1        ;UPDATE AND SAVE COUNT
        176541     226     054
     52 176543                              LDA B   A.CBCT
        176543     326     053
```

A.8250 MODEM STATUS HANDLER

```
     65                                     ;
     66 176573                              LDX     #,32767.        ;CHECK A.LWCT
        176573     316     177     377
     67 176576                              CPX     A.LWCT          ;THE SAME ?
        176576     234     057
     68 176600                              BEQ     2$              ;IF SO - NO TRANSITION FOUND
        176600     047     275
     69                                     ;
     70 176602                              LDX     #,A.BTBL        ;GET POINTER TO TABLE
        176602     316     374     271
     71 176605                         7$:  LDA A   A.LWCT+1        ;GET A.LWCT
        176605     226     060
     72 176607                              LDA B   A.LWCT
        176607     326     057
     73 176611                              SUB A   1,X             ;CHECK IF LESS THAN TABLE ELEMENT
        176611     240     001
     74 176613                              SBC B   0,X
        176613     342     000
     75 176615                              BMI     8$              ;IF SO - SKIP
        176615     053     006
     76 176617                              INX                     ;ELSE SCAN TABLE
        176617     010
     77 176620                              INX
        176620     010
     78 176621                              INX
        176621     010
     79 176622                              INX
        176622     010
     80 176623                              BRA     7$
        176623     040     360
     81                                     ;
     82 176625                         8$:  LDX     2,X             ;GET BAUD RATE DIVISOR
        176625     356     002
     83 176627                              STX     A.BDVS          ;SAVE DIVISOR
        176627     337     061
```

A.8250 INITIALIZATION ROUTINE

```
      1                                     .SBTTL  A.8250 INITIALIZATION ROUTINE
      2                                     ;
      3 176654                        A.82$B: LDA B   #,36            ;SET UP A.LNBL
        176654     306     036
      4 176656                                STA B   A.LNBL
        176656     327     047
      5 176660                                LDA B   #,17            ;SET UP A.MNBL
        176660     306     017
      6 176662                                STA B   A.MNBL
        176662     327     050
      7 176664                                LDA B   #,-1            ;SET UP A.LNST
        176664     306     377
      8 176666                                STA B   A.LNST          ;TO CHECK BAUD RATE
        176666     327     045
      9 176670                                LDA B   #,11            ;ERROR COUNT PRESET
        176670     306     011
     10 176672                                STA B   A.ERCT          ;IN STATUS TABLE
        176672     327     046
     11 176674                                LDA B   #,216           ;LINE CONTROL
        176674     306     216
     12 176676                                STA B   LCR+A.8250
        176676     327     223
     13 176700                                LDA B   #,12.&377       ;SET UP FOR 9600 BAUD
        176700     306     014
     14 176702                                STA B   DLL+A.8250
        176702     327     220
     15 176704                                LDA B   #,12.&177400/400
        176704     306     000
     16 176706                                STA B   DLM+A.8250
        176706     327     221
     17 176710                                LDA B   #,16
        176710     306     016
     18 176712                                STA B   LCR+A.8250
        176712     327     223
     19 176714                                RTS
```

THE END

```
      2                                       .SBTTL  THE END
      3                                       ;
      4         000001                        .END
```

Symbol table

```
A.AHDF  175135          A.BL$I  175652          A.MNBL  000050          DDRAL = 000005          MCR   = 0
00004

A.AHDN  175165          A.BPRT  000025          A.MODE  176252          DDRBH = 000006          MSFLAG  0
00005

A.AHDR  175103          A.BTBL  176271          A.MODN  175715          DDRBL = 000007          MSR   = 0
00006

A.AHRD  175273          A.BWRD  175501          A.MOSV  175703          DLL   = 000000          NMINT   1
77774

A.AHUP  175153          A.BWWT  175567          A.MOUP  175706          DLM   = 000001          NMIRQ   1
74000

A.AHWT  175347          A.CBCT  000053          A.MO$V  000035          IER   = 000001          NMIRQV  0
00000

A.AH$I  175617          A.DLLR  176037          A.MSRR  176007          IIR   = 000002          PGMSAV= 1
75703

A.ALDF  175144          A.DLLW  176054          A.MSRW  176012          IRQN    177770          PINITS  1
74104

A.ALDN  175172          A.DLMR  176071          A.OVRN  176421          IRQ0    177750          PRAH  = 0
00004

A.ALDR  175120          A.DLMW  176106          A.PBCT  000051          IRQ1    177752          PRAL  = 0
00005

A.ALRD  175311          A.ERCT  000046          A.PRTY  176360          IRQ2    177754          PRBH  = 0
00006

A.ALUP  175160          A.FRAM  176402          A.RCDN  175757          IRQ3    177756          PRBL  = 0
00007

A.ALWT  175370          A.IRQD  175666          A.RCSV  175745          IRQ4    177760          RBR   = 0
00000

A.AL$I  175630          A.IRQG  175663          A.RCUP  175750          IRQ5    177762          RESINT  1
77776

A.APRT  000015          A.LBRK  176343          A.RCVR  176212          IRQ6    177764          RESTRT  1
74016

A.AWRD  175327          A.LCRR  176031          A.RC$V  000041          IRQ7    177766          SPRIUS  1
74006

A.AWWT  175411          A.LCRW  176034          A.XMDN  175736          LCR   = 000003          STACK = 0
00177

A.BDIF  000055          A.LEND  176421          A.XMSV  175724          LOADER  174714          SWINT   1
77772

A.BDVS  000061          A.LERR  176414          A.XMTR  176161          LOAD$I  174706          SWIRQ   1
74007

A.BHDF  175231          A.LINE  176233          A.XMUP  175727          LOAD$V  000006          SWIRQV  0
00002

A.BHDN  175261          A.LNBL  000047          A.XM$V  000037          LSR   = 000005          S.FNCT  0
00004

A.BHDR  175177          A.LNDN  176000          A.6821= 000200          L.ADDR  000012          S.INAH  1
74545

A.BHRD  175441          A.LNST  000045          A.82$B  176654          L.BYTC  000010          S.INAL  1
74403
```

```
A.BHWT  175525          A.LNUP  175771          A.82$L  176335          L.GBYT  174732          S.RCVR  1
74355

A.BH$I  175641          A.LN$V  000043          A.82$M  176423          L.LDR0  174744          S.XMTR  1
74341

A.BLDF  175240          A.LSRR  176023          A.8250= 000220          L.LDR1  174751          THR   = 0
00000

A.BLDN  175266          A.LSRW  176026          CRAH  = 000000          L.LDR2  174756          VARSAV= 0
00063

A.BLDR  175214          A.LWCT  000057          CRAL  = 000001          L.LDR3  175015          $A.682= 0
00001

A.BLRD  175461          A.MCRR  176015          CRBH  = 000002          L.LDR4  175050          $A.825= 0
00001

A.BLUP  175254          A.MCRW  176020          CRBL  = 000003          L.LDR5  175060          $LOADE= 0
00001

A.BLWT  175546          A.MEND  176422          DDRAH = 000004          L.LDR6  175077          ...A  = 0
00100



. ABS.  177776    000   (RW,I,GBL,ABS,OVR)

        000000    001   (RW,I,LCL,REL,CON)

Errors detected:  0


*** Assembler statistics



Work  file  reads: 0

Work  file writes: 0

Size of work file: 14766 Words  ( 58 Pages)

Size of core pool: 18176 Words  ( 71 Pages)

Operating  system: RT-11


Elapsed time: 00:00:02.23

SERIAL,SERIAL=M6800,SERIAL,LOADER,6821A,8250A,END
```

```
R MACRO
PARLEL,PARLEL=M6800,PARLEL,LOADER,6821A,6821B,END
```

```
.IIF    NDF,.M68        .NLIST  ;CROSS-ASSEMBLER NOT LISTED
        ;DURING CHECKOUT DEFINE .M68 TO ENABLE LISTING
.IIF    DF,.M68         .LIST   SRC,SEQ,COM,MD,MC
        ;
        ;
        .TITLE  M6800 CROSS-ASSEMBLER
        ;
.IIF    DF,.M68 .SBTTL  CROSS-ASSEMBLER INTRODUCTION
        ;
.ENABLE ABS
        ;
        ;*************************************************************
        ;*                                                           *
        ;*      MACRO PACKAGE FOR THE MOTOROLA 6800 MICROPROCESSOR   *
        ;*              TO RUN UNDER MACRO 11.                       *
        ;*                                                           *
        ;*      BY ALAN R. BALDWIN                                   *
        ;*                                                           *
        ;*              V03 - OCTOBER 1980                           *
        ;*                                                           *
        ;*************************************************************
        ;
        ;
        ;THE FOLLOWING DIFFERENCES EXIST BETWEEN THIS CROSS-ASSEMBLER
        ;AND MOTOROLA'S M6800 ASSEMBLER
        ;       LABELS MUST TERMINATE WITH A :
        ;       COMMENTS START WITH A ;
        ;       IMMEDIATE MODE IS DENOTED BY A SEPERATE ARGUMENT - #
        ;
        ;
        ;DEFINITION OF ASSEMBLER DIRECTIVES WITH DIFFERENCES
        ;       END - USE       .END    END OF PROGRAM
        ;       EQU - USE  =            EQUATE SYMBOL
        ;       FCB                     FORM SINGLE-BYTE CONSTANT
        ;                               NO MORE THAN 10 ARGUMENTS
        ;       FCC   <ASCII CHRS>      FORM CONSTANT CHARACTERS
        ;       FDB                     FORM DOUBLE-BYTE CONSTANT
        ;                               NO MORE THAN 10 ARGUMENTS
        ;       MON - NOT IMPLEMENTED   RETURN TO MONITOR CONSOLE
        ;       NAM - USE       .SBTTL  PROGRAM NAME
        ;       OPT - NOT IMPLEMENTED   OPTION
        ;       ORG                     ORIGIN
        ;       PAGE - USE      .PAGE   ADVANCE LISTING TO TOP OF PAGE
        ;       RMB                     RESERVE MEMORY BYTES
        ;       SPC - NOT IMPLEMENTED   SPACE N LINES
        ;
        ;
        ;PROCESSOR CONDITION CODE REVIEW
        ;       0 - CARRY BIT (C)
        ;       1 - OVERFLOW BIT (V)
        ;       2 - ZERO BIT (Z)
        ;       3 - NEGATIVE BIT (N)
        ;       4 - INTERRUPT MASK BIT (I)
        ;       5 - HALF CARRY BIT (H)
        ;       6 - ALWAYS 1
        ;       7 - ALWAYS 1
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  SINGLE BYTE 'INHERENT' INSTRUCTIONS
        ;
.MACRO  AINST   H,I
 .MACRO H
 .NLIST SRC
 .BYTE  I
 .LIST  SRC
 .ENDM
.ENDM   AINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
AINST   NOP,    1       ;DO NOTHING
AINST   TAP,    6       ;A TO CC'S
AINST   TPA,    7       ;CC'S TO A
AINST   INX,    10      ;INCREMENT INDEX REGISTER
AINST   DEX,    11      ;DECREMENT INDEX REGISTER
AINST   CLV,    12      ;CLEAR V BIT
```

```
AINST   SEV,    13      ;SET C BIT
AINST   CLC,    14      ;CLEAR C BIT
AINST   SEC,    15      ;SET C BIT
AINST   CLI,    16      ;CLEAR I BIT
AINST   SEI,    17      ;SET I BIT
AINST   SBA,    20      ;ACCA=ACCA-ACCB
AINST   CBA,    21      ;COMPARE ACCA & ACCB
AINST   TAB,    26      ;ACCB=ACCA
AINST   TBA,    27      ;ACCA=ACCB
AINST   DAA,    31      ;DECIMAL ADJUST
AINST   ABA,    33      ;ACCA=ACCA+ACCB
AINST   TSX,    60      ;X=SP+1
AINST   INS,    61      ;SP=SP+1
AINST   PULA,   62      ;PULL A FROM STACK
AINST   PULB,   63      ;PULL B FROM STACK
AINST   DES,    64      ;SP=SP-1
AINST   TXS,    65      ;SP=X-1
AINST   PSHA,   66      ;PUSH A ONTO STACK
AINST   PSHB,   67      ;PUSH B ONTO STACK
AINST   RTS,    71      ;RETURN FROM SUBROUTINE
AINST   RTI,    73      ;RETURN FROM INTERRUPT
AINST   WAI,    76      ;WAIT FOR INTERRUPT
AINST   SWI,    77      ;SOFTWARE INTERRUPT
AINST   NEGA,   100     ;NEGATE A
AINST   COMA,   103     ;COMPLEMENT A
AINST   LSRA,   104     ;LOGICAL SHIFT RIGHT A
AINST   RORA,   106     ;ROTATE RIGHT A
AINST   ASRA,   107     ;ARITHMETIC SHIFT RIGHT A
AINST   ASLA,   110     ;ARITHMETIC SHIFT LEFT A
AINST   ROLA,   111     ;ROTATE LEFT A
AINST   DECA,   112     ;DECREMENT A
AINST   INCA,   114     ;INCREMENT A
AINST   TSTA,   115     ;TEST A
AINST   CLRA,   117     ;CLEAR A
AINST   NEGB,   120     ;NEGATE B
AINST   COMB,   123     ;COMPLEMENT B
AINST   LSRB,   124     ;LOGICAL SHIFT RIGHT B
AINST   RORB,   126     ;ROTATE RIGHT B
AINST   ASRB,   127     ;ARITHMETIC SHIFT RIGHT B
AINST   ASLB,   130     ;ARITHMETIC SHIFT LEFT B
AINST   ROLB,   131     ;ROTATE LEFT B
AINST   DECB,   132     ;DECREMENT B
AINST   INCB,   134     ;INCREMENT B
AINST   TSTB,   135     ;TEST B
AINST   CLRB,   137     ;CLEAR B
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  PUSH AND PULL OPCODES
        ;
.MACRO  PKRNL   I,J
 .IIF   IDN     <J>,A   ...A=0  ;PSH/PUL A
 .IIF   IDN     <J>,B   ...A=1  ;PSH/PUL B
 .BYTE  I+...A          ;INVALID ARGUMENT
.ENDM
        ;
.MACRO  PINST   H,I
 .MACRO H       J
 .NLIST SRC
 PKRNL  I,J
 .LIST  SRC
 .ENDM
.ENDM   PINST
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
PINST   PUL,    62      ;PUL BYTE FROM STACK S=S+1
PINST   PSH,    66      ;PUSH BYTE ONTO STACK S=S-1
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  RELATIVE BRANCH INSTRUCTIONS
        ;
.MACRO  BKRNL   I,J
 ...A=J-.-2
 .IIF   LT,...A+200     .ERROR  ;BRANCH OUT OF RANGE
 .IIF   GE,...A-200     .ERROR  ;BRANCH OUT OF RANGE
 .BYTE  I,...A
.ENDM
```

```
        ;
.MACRO  BINST   H,I
 .MACRO H       J
 .NLIST SRC
 BKRNL  I,J
 .LIST  SRC
 .ENDM
.ENDM   BINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPFERATION
BINST   BRA,    40      ;BRANCH ALWAYS
BINST   BHI,    42      ;BRANCH IF (C=0) AND (Z=0)
BINST   BLS,    43      ;BRANCH IF (C=1) OR (Z=1)
BINST   BCC,    44      ;BRANCH IF (C=0)
BINST   BCS,    45      ;BRANCH IF (C=1)
BINST   BNE,    46      ;BRANCH IF (Z=0)
BINST   BEQ,    47      ;BRANCH IF (Z=1)
BINST   BVC,    50      ;BRANCH IF (V=0)
BINST   BVS,    51      ;BRANCH IF (V=1)
BINST   BPL,    52      ;BRANCH IF (N=0)
BINST   BMI,    53      ;BRANCH IF (N=1)
BINST   BGE,    54      ;BRANCH IF (<N XOR V>=0)
BINST   BLT,    55      ;BRANCH IF (<N XOR V>=1)
BINST   BGT,    56      ;BRANCH IF (Z=0) AND (<N XOR V>=0)
BINST   BLE,    57      ;BRANCH IF (Z=1) OR (<N XOR V>=1)
BINST   BSR,    215     ;BRANCH TO SUBROUTINE
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  INSTRUCTIONS HAVING ONLY ACCX,INDEXED,AND EXTENDED MODES
        ;
        ;
.MACRO  CKRNL   I,J,K
 .IF    NB,<K>          ;TWO ARGUMENTS - THEN INDEXED
  .IIF  DIF     <K>,X   .ERROR  ;INDEX BAD
  .IIF  LT,J    .ERROR  ;NEGATIVE OFFSET
  .BYTE I+40,J          ;OFFSET OUT OF RANGE
  .MEXIT
 .ENDC
 .IF    NB,<J>          ;ONE ARGUMENT - A, B, OR EXTENDED MODE
  ...A=60
  .IIF  IDN     <J>,A   ,...A=0         ;ACCA MODE
  .IIF  IDN     <J>,B   ,...A=20        ;ACCB MODE
  .IIF  NE,...A-60      .BYTE   I+...A
  .IIF  EQ,...A-60      .BYTE   I+...A,J&177400/400,J&377
  .MEXIT
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  CINST   H,I
 .MACRO H       J,K
 .NLIST SRC
 CKRNL  I,J,K
 .LIST  SRC
 .ENDM
.ENDM   CINST
        ;
        ;

    ;MNEMONIC  OPCODE   ;OPERATION
        ;
CINST   NEG,    100     ;NEGATE
CINST   COM,    103     ;COMPLEMENT
CINST   LSR,    104     ;LOGICAL SHIFT RIGHT
CINST   ROR,    106     ;ROTATE RIGHT
CINST   ASR,    107     ;ARITHMETIC SHIFT RIGHT
CINST   ASL,    110     ;ARITHMETIC SHIFT LEFT
CINST   ROL,    111     ;ROTATE LEFT
CINST   DEC,    112     ;DECREMENT
CINST   INC,    114     ;INCREMENT
CINST   TST,    115     ;TEST
CINST   CLR,    117     ;CLEAR
        ;
        ;
```

```
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  JUMP AND JSR INSTRUCTIONS
        ;
.MACRO  DKRNL   I,J,K
 .IF    NB,<K>          ;TWO ARGUMENTS - INDEXED MODE
  .IIF  DIF     <K>,X   .ERROR  ;BAD INDEX
  .IIF  LT,J            .ERROR  ;NEGATIVE OFFSET
  .BYTE I+40,J          ;OFFSET TOO LARGE
  .MEXIT
 .ENDC
 .IF    NB,<J>          ;ONE ARGUMENT - EXTENDED MODE
  .BYTE I+60,J&177400/400,J&377
  .MEXIT
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  DINST   H,I
 .MACRO H       J,K
 .NLIST SRC
 DKRNL  I,J,K
 .LIST  SRC
 .ENDM
.ENDM   DINST
        ;
        ;
     ;MNEMONIC  OPCODE  ;OPERATION
DINST   JMP,    116     ;JUMP
DINST   JSR,    215     ;JUMP TO SUBROUTINE
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  ALL ACCX INSTRUCTIONS
        ;
.MACRO  EKRNL   I,J,K,L
 ...A=-1
 .IIF   IDN     <J>,A   ,...A=0         ;ACCA MODE
 .IIF   IDN     <J>,B   ,...A=100       ;ACCB MODE
  .IF   GE,...A                 ;ACCX MODES
   .IF  NB,<L>                  ;THREE ARGS - IMMEDIATE/INDEXED
    .IF IDN     <K>,#   ;CHECK IMMEDIATE
     .IIF       EQ      <I>-207 ,.ERROR ;STA #
     .BYTE      I+...A,L
     .MEXIT
    .ENDC
    .IF IDN     <L>,X   ;CHECK INDEXED
     .IIF       LT,K    .ERROR  ;NEGATIVE OFFSET
     .BYTE      I+...A+40,K     ;OFFSET TOO LARGE
     .MEXIT
    .ENDC
    .ERROR      ;BAD INSTRUCTION
    .MEXIT
   .ENDC
   .IF  NB,<K>                  ;TWO ARGS - DIRECT/EXTENDED
    .IIF        EQ,K&177400     .BYTE   I+...A+20,K
    .IIF        NE,K&177400     .BYTE   I+...A+60,K&177400/400,K&377
    .MEXIT
   .ENDC
  .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  EINST   H,I
 .MACRO H       J,K,L
 .NLIST SRC
 EKRNL  I,J,K,L
 .LIST  SRC
 .ENDM
.ENDM   EINST
        ;
        ;
     ;MNEMONIC  OPCODE  ;OPERATION
EINST   SUB,    200     ;SUBTRACT
EINST   CMP,    201     ;COMPARE
EINST   SBC,    202     ;SUBTRACT WITH CARRY
```

```
EINST   AND,    204     ;LOGICAL AND
EINST   BIT,    205     ;BIT TEST
EINST   LDA,    206     ;LOAD ACCUMULATOR
EINST   STA,    207     ;STORE ACCUMULATOR
EINST   EOR,    210     ;EXCLUSIVE OR
EINST   ADC,    211     ;ADD WITH CARRY
EINST   ORA,    212     ;LOGICAL OR
EINST   ADD,    213     ;ADD
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  ALL SHORT FORM ACCX INSTRUCTIONS
        ;
.MACRO  SKRNL   I,J,K
 .IF    NB,<K>          ;TWO ARGS - IMMEDIATE/INDEXED
  .IF   IDN     <J>,#   ;CHECK IMMEDIATE
   .IIF EQ      <I>-207 ,.ERROR ;STAA #
   .IIF EQ      <I>-307 ,.ERROR ;STAB #
   .BYTE        I,K
   .MEXIT
  .ENDC
  .IF   IDN     <K>,X   ;CHECK INDEXED
   .IIF LT,J    .ERROR  ;NEGATIVE OFFSET
   .BYTE        I+40,J          ;OFFSET TOO LARGE
   .MEXIT
  .ENDC
  .ERROR        ;BAD INSTRUCTION
  .MEXIT
 .ENDC
 .IF    NB,<J>          ;ONE ARG - DIRECT/EXTENDED
  .IIF  EQ,J&177400     .BYTE   I+20,J
  .IIF  NE,J&177400     .BYTE   I+60,J&177400/400,J&377
  .MEXIT
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  SINST   H,I
 .MACRO H       J,K
 .NLIST SRC
 SKRNL  I,J,K
 .LIST  SRC
 .ENDM
.ENDM   SINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
        ;
SINST   SUBA,   200     ;SUBTRACT
SINST   SUBB,   300
SINST   CMPA,   201     ;COMPARE
SINST   CPMB,   301
SINST   SBCA,   202     ;SUBTRACT WITH CARRY
SINST   SBCB,   302
SINST   ANDA,   204     ;LOGICAL AND
SINST   ANDB,   304
SINST   BITA,   205     ;BIT TEST
SINST   BITB,   305
SINST   LDAA,   206     ;LOAD ACCUMULATOR
SINST   LDAB,   306
SINST   STAA,   207     ;STORE ACCUMULATOR
SINST   STAB,   307
SINST   EORA,   210     ;EXCLUSIVE OR
SINST   EORB,   310
SINST   ADCA,   211     ;ADD WITH CARRY
SINST   ADCB,   311
SINST   ORAA,   212     ;LOGICAL OR
SINST   ORAB,   312
SINST   ADDA,   213     ;ADD
SINST   ADDB,   313
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  STACK AND INDEX REGISTER INSTRUCTIONS
        ;
.MACRO  FKRNL   I,J,K,L
```

```
 .IF    B,<K>           ;ONE ARG - DIRECT/EXTENDED MODE
  .IF   NB,<J>
   .IIF NE,J&177400     .BYTE   I+60,J&177400/400,J&377
   .IIF EQ,J&177400     .BYTE   I+20,J
   .MEXIT
  .ENDC
  .ERROR        ;BAD INSTRUCTION
  .MEXIT
 .ENDC
 .IF    IDN     <J>,#   ;IMMEDIATE MODE
  .IIF  EQ      <I>-217 ,.ERROR ;STS #
  .IIF  EQ      <I>-317 ,.ERROR ;STX #
  .IIF  NB,<L>  .BYTE   I,K,L
  .IIF  B,<L>   .BYTE   I,K&177400/400,K&377
  .MEXIT
 .ENDC
 .IF    B,<L>
  .IF   NB,<K>
   .IF  IDN     <K>,X           ;INDEXED
    .IIF        LT,J    .ERROR  ;NEGATIVE OFFSET
    .BYTE       I+40,J
    .MEXIT
   .ENDC
  .ENDC
 .ENDC
 .ERROR ;BAD INSTRUCTION
.ENDM
        ;
.MACRO  FINST   H,I
 .MACRO H       J,K,L
 .NLIST SRC
 FKRNL  I,J,K,L
 .LIST  SRC
 .ENDM
.ENDM   FINST
        ;
        ;
    ;MNEMONIC  OPCODE   ;OPERATION
        ;
FINST   CPX,    214     ;COMPARE TO INDEX
FINST   LDS,    216     ;LOAD STACK REGISTER
FINST   LDX,    316     ;LOAD INDEX REGISTER
FINST   STS,    217     ;STORE STACK REGISTER
FINST   STX,    317     ;STORE INDEX REGISTER
        ;
        ;
        ;
.IIF    DF,.M68 .PAGE
.IIF    DF,.M68 .SBTTL  ASSEMBLER DIRECTIVES
        ;
.MACRO  FCB     A,B,C,D,E,F,G,H,I,J
.NLIST  SRC
 .IIF   NB,<A>  .BYTE   A
 .IIF   NB,<B>  .BYTE   B
 .IIF   NB,<C>  .BYTE   C
 .IIF   NB,<D>  .BYTE   D
 .IIF   NB,<E>  .BYTE   E
 .IIF   NB,<F>  .BYTE   F
 .IIF   NB,<G>  .BYTE   G
 .IIF   NB,<H>  .BYTE   H
 .IIF   NB,<I>  .BYTE   I
 .IIF   NB,<J>  .BYTE   J
.LIST   SRC
.ENDM   FCB
        ;
.MACRO  FCC     H
.NLIST  SRC
.ASCII  /H/
.LIST   SRC
.ENDM   FCC
        ;
.MACRO  FDB     A,B,C,D,E,F,G,H,I,J
.NLIST  SRC
 .IIF   NB,<A>  .BYTE   A&177400/400,A&377
 .IIF   NB,<B>  .BYTE   B&177400/400,B&377
 .IIF   NB,<C>  .BYTE   C&177400/400,C&377
```

```
 .IIF   NB,<D>  .BYTE   D&177400/400,D&377
 .IIF   NB,<E>  .BYTE   E&177400/400,E&377
 .IIF   NB,<F>  .BYTE   F&177400/400,F&377
 .IIF   NB,<G>  .BYTE   G&177400/400,G&377
 .IIF   NB,<H>  .BYTE   H&177400/400,H&377
 .IIF   NB,<I>  .BYTE   I&177400/400,I&377
 .IIF   NB,<J>  .BYTE   J&177400/400,J&377
.LIST   SRC
.ENDM   FDB
        ;
.MACRO  ORG     H
.NLIST  SRC
.=<H>
.LIST   SRC
.ENDM   ORG
        ;
.MACRO  RMB     H
.NLIST  SRC
.BLKB   H
.LIST   SRC
.ENDM   RMB
        ;
        ;
        ;
.NLIST  TTM     ;PRINTING MODE
.LIST   MD,MEB  ;ENABLE PRINTING OF MACRO EXPANSIONS
.LIST           ;LIST PROGRAM PROPER
        .PAGE
```

```
        .TITLE  PARALLLEL 'PORT'
        .SBTTL  PARALLEL PORT AS SEEN ON THE ARB-11 BUS
        ;
        ;[X XXX XXX XXX XXN NNN]  00    INPUT BUFFER STATUS REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;   0   0   0   0   0   0   0   0      R   I   0   0   0   0   0   0
        ;
        ;       R -     <07:00> DATA READY (R)
        ;                       CLEARED BY READ OF
        ;                       DATA <07:00> OR <15:00> OF (02)
        ;       I -     INTERRUPT ENABLE BIT (R/W)
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  02    INPUT BUFFER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;
        ; [15-00]       16-BIT INPUT BUFFER
        ; [15-08]       CHECK SUM (FUNCTION=7)
        ;
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  04    OUTPUT BUFFER STATUS REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;   0   0   0   0   0   0   0   0      R   I   0   0   0   0   0   0
        ;
        ;       R -     <07:00> DATA REQUEST (R)
        ;                       CLEARED BY WRITE OF
        ;                       DATA <07:00> OR <15:00> OF (06)
        ;       I -     INTERRUPT ENABLE BIT (R/W)
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  06    OUTPUT BUFFER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;
        ;
        ; [15-00]       16-BIT OUTPUT BUFFER
        ; [15-08]       8-BIT FUNCTION CODE (ONLY IF UPPER BYTE INTERRUPT
        ;                                    IS ENABLED DURING WRTE)
        ; [07-00]       8-BIT BINARY LOADER BUFFER (FUNCTION=7)
        ;
        ;
        .PAGE
        ;
        ;[X XXX XXX XXX XXN NNN]  10    FUNCTION STATUS REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;   0   0   0   0   0   0   0   0      R   I   0   0   0   0   0   0
        ;
        ;       R -     <07:00> FUNCTION DATA READY (R)
        ;                       CLEARED BY READ OF
        ;                       DATA <15:08> OR <15:00> OF (02)
        ;       I -     INTERRUPT ENABLE BIT (R/W)
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  12    A-PORT PERIPHERAL CONTROL REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;[ H7  H6  H5  H4  H3  H2  H1  H0 ] [ L7  L6  L5  L4  L3  L2  L1  L0 ]
        ;
        ;       <H7-H0> -       <15:08> PIA CONTROL FOR (02)
        ;       <L7-L0> -       <07:00> PIA CONTROL FOR (02)
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  14    FUNCTION STATUS
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
```

```
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;   0   0   0   0   0   0   0   0      R   I   0   0   0   0   0   0
        ;
        ;       R -     FUNCTION CODE READY
        ;                       CLEARED BY WRITE OF
        ;                       DATA <15:08> OR <15:00> OF (06)
        ;       I -     INTERRUPT ENABLE BIT (R/W)
        ;
        ;
        ;[X XXX XXX XXX XXN NNN]  16    B-PORT PERIPHERAL CONTROL REGISTER
        ; <15:08> H                          <07:00> L
        ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03  02  01  00 ]
        ;  __  __  __  __  __  __  __  __     __  __  __  __  __  __  __  __
        ;[ H7  H6  H5  H4  H3  H2  H1  H0 ] [ L7  L6  L5  L4  L3  L2  L1  L0 ]
        ;
        ;       <H7-H0> -       <15:08> PIA CONTROL FOR (06)
        ;       <L7-L0> -       <07:00> PIA CONTROL FOR (06)
        ;
        ;
        .PAGE
        .SBTTL  SOFTWARE DEFINITIONS
        ;
        ;         THE PARALLEL PORT SOFTWARE REQUIRES
        ;       THE 16-BIT PIA SOFTWARE PACKAGE '6821A.MAC',
        ;       THE 16-BIT PIA SOFTWARE PACKAGE '6821B.MAC', AND
        ;       THE ABSOLUTE BINARY LOADER 'LOADER.MAC'.
        ;
        ;
        $A.6821 =1
        A.6821  =200    ;16-BIT BUS PORT ADDRESS
        ;
        $B.6821 =1
        B.6821  =210    ;16-BIT I/O PORT ADDRESS
        ;
        $LOADER =1      ;USE ABSOLUTE LOADER
        ;
        $FAST   =1      ;DIRECT VERSION
        ;
        PGMSAV  =174000 ;ROM ADDRESS
        VARSAV  =0      ;DIRECT VARIABLE SPACE TO 177
        ;
        STACK   =177    ;STACK AREA (64-BYTES)
        ;
        ;       INTERRUPT VECTORS
        ;
        .=177740
        FDB     SPRIUS  ;OOPS !
        FDB     SPRIUS  ;OOPS !
        FDB     SPRIUS  ;OOPS !
        FDB     SPRIUS  ;OOPS !
IRQ0:   FDB     B.BL$I  ;I/O B-PORT <7:0> INTERRUPT
IRQ1:   FDB     B.BH$I  ;I/O B-PORT <15:8> INTERRUPT
IRQ2:   FDB     A.AL$I  ;BUS A-PORT <7:0> INTERRUPT
IRQ3:   FDB     A.AH$I  ;BUS A-PORT <15:8> INTERRUPT
IRQ4:   FDB     A.BL$I  ;BUS B-PORT <7:0> INTERRUPT
IRQ5:   FDB     A.BH$I  ;BUS B-PORT <15:8> INTERRUPT
IRQ6:   FDB     B.AL$I  ;I/O A-PORT <7:0> INTERRUPT
IRQ7:   FDB     B.AH$I  ;I/O A-PORT <15:8> INTERRUPT
IRQN:   FDB     SPRIUS  ;NO HARDWARE
SWINT:  FDB     SWIRQ   ;SWI INTERRUPT
NMINT:  FDB     NMIRQ   ;ATTACHED PROCESSOR INIT INTERRUPT
RESINT: FDB     RESTRT  ;POWER UP RESTART VECTOR
        ;
        .PAGE
        .SBTTL  STARTUP AND BACKGROUND PROGRAM
        ;
        .=VARSAV
NMIRQV: .BYTE   0,0     ;INIT INTERRUPT VECTOR
SWIRQV: .BYTE   0,0     ;SWI INTERRUPT VECTOR
P.FNCT: .BYTE   0       ;FUNCTION CODE
        VARSAV=.
        .=PGMSAV
        ;
NMIRQ:  LDX     NMIRQV          ;GET VECTOR
        BEQ     SPRIUS          ;BRANCH IF NOT DEFINED
        JSR     0,X             ;ELSE DO IT
```

```
SPRIUS: RTI                     ;HOW DID WE GET HERE ?
        ;
SWIRQ:  LDX     SWIRQV          ;GET SWI VECTOR
        BEQ     1$              ;BRANCH IF NOT DEFINED
        JMP     0,X             ;SPECIAL SWI CALL TO ACCESS
1$:     RTI                     ;INTERRUPT DRIVEN ROUTINES
        ;
RESTRT: LDS     #,STACK         ;SET UP STACK POINTER
        LDX     #,0
        STX     SWIRQV          ;SET SWI FOR NOTHING
        STX     NMIRQV          ;SET NMI FOR NOTHING
        BSR     PINITS          ;INITIALLIZE THE PORT
        ;
        ;       BACKGROUND PROGRAM
        ;
1$:     LDS     #,STACK         ;SET THE STACK POINTER
        CLR     177740          ;ENABLE ALL INTERRUPTS
        CLI                     ;(VIA 6828 CONTROLLER)
        LDX     #,0             ;LOOP COUNTER
2$:     INX                     ;LOOP HERE 64K TIMES
        BNE     2$
        JSR     A.ALUP          ;RE-ENABLE <7:0> INPUT
        JSR     A.AHUP          ;RE-ENABLE <15:8> INPUT
        JSR     A.BLUP          ;RE-ENABLE <7:0> OUTPUT
        JSR     A.BHDN          ;DISABLE <15:8> OUTPUT
        JSR     B.ALUP          ;RE-ENABLE <7:0> INPUT
        JSR     B.AHUP          ;RE-ENABLE <15:8> INPUT
        JSR     B.BLUP          ;RE-ENABLE <7:0> OUTPUT
        JSR     B.BHDN          ;DISABLE <15:8> OUTPUT
        BRA     1$
        ;
        .PAGE
        .SBTTL  PORT INITIALIZATION
        ;
PINITS: JSR     LOAD$I          ;INITIALIZE ABSOLUTE LOADER
        ;
        CLR     P.FNCT          ;INSURE NORMAL PORT
        ;
        ;       SET UP BUS-PORT PARALLEL OUTPUT
        ;
        LDX     #,P.BSBL        ;SERVICE VECTOR
        LDA B   #,45            ;CB2(ON E, LOW), CB1(-, HIGH)
        LDA A   #,74
        JSR     A.BLDF          ;SAVE CONTROL
        LDX     #,0             ;NO SERVICE
        LDA B   #,44            ;CB2(ON E, LOW), CB1(-, HIGH)
        LDA A   #,74
        JSR     A.BHDF          ;SAVE CONTROL
        LDA B   #,377           ;OUTPUT
        JSR     A.BLDR
        LDA B   #,377           ;OUTPUT
        JSR     A.BHDR
        ;
        ;       SET UP BUS-PORT PARALLEL INPUT
        ;
        LDX     #,P.BSAL        ;SERVICE VECTOR
        LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
        LDA A   #,74
        JSR     A.ALDF          ;SAVE CONTROL
        LDX     #,P.BSAH        ;SERVICE VECTOR
        LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
        LDA A   #,74
        JSR     A.AHDF          ;SAVE CONTROL
        LDA B   #,0             ;INPUT
        JSR     A.ALDR
        LDA B   #,0             ;INPUT
        JSR     A.AHDR
        ;
        ;       SET UP I/O-PORT PARALLEL OUTPUT
        ;
        LDX     #,P.IOBL        ;SERVICE VECTOR
        LDA B   #,45            ;CB2(ON E, LOW), CB1(-, HIGH)
        LDA A   #,74
        JSR     B.BLDF          ;SAVE CONTROL
        LDX     #,0             ;NO SERVICE
        LDA B   #,44            ;CB2(ON E, LOW), CB1(-, HIGH)
```

```
        LDA A   #,74
        JSR     B.BHDF          ;SAVE CONTROL
        LDA B   #,377           ;OUTPUT
        JSR     B.BLDR
        LDA B   #,377           ;OUTPUT
        JSR     B.BHDR
        ;
        ;       SET UP I/O-PORT PARALLEL INPUT
        ;
        LDX     #,P.IOAL        ;SERVICE VECTOR
        LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
        LDA A   #,74
        JSR     B.ALDF          ;SAVE CONTROL
        LDX     #,0             ;NO SERVICE
        LDA B   #,44            ;CA2(ON E, LOW), CA1(-, HIGH)
        LDA A   #,74
        JSR     B.AHDF          ;SAVE CONTROL
        LDA B   #,0             ;INPUT
        JSR     B.ALDR
        LDA B   #,0             ;INPUT
        JSR     B.AHDR
        ;
        ;       ENABLE/DISABLE INTERRUPTS
        ;
        ;BUS-PORT
        JSR     A.ALDN          ;CLEAR, CA2='1'
        JSR     A.ALUP          ;ENABLE
        JSR     A.ALRD+2        ;CLEAR IRQA, CA2='0'
        ;
        JSR     A.AHDN          ;CLEAR, CA2='1'
        JSR     A.AHUP          ;ENABLE
        JSR     A.AHRD+2        ;CLEAR IRQ, CA2='0'
        ;
        JSR     A.BLDN          ;CLEAR, CB2='1'
        JSR     A.BLRD+2        ;CLEAR IRQ, CB2='1'
        JSR     A.BLUP          ;ENABLE
        ;
        JSR     A.BHDN          ;CLEAR, CB2='1'
        JSR     A.BHRD+2        ;CLEAR IRQ, CB2='1'
        ;
        ;I/O-PORT
        JSR     B.AHDN          ;CLEAR, CA2='1'
        JSR     B.ALDN
        JSR     B.AHUP          ;ENABLE
        JSR     B.ALUP
        JSR     B.AWRD+2        ;CLEAR IRQ, CA2'S='0'
        ;
        JSR     B.BHDN          ;DISABLE
        JSR     B.BLDN
        JSR     B.BWRD+2        ;CLEAR IRQ, CB2'S='1'
        JSR     B.BLUP          ;ENABLE
        ;
        RTS
        ;
        .IF     NDF,$FAST
        ;       PORT SERVICE ROUTINES
        ;
        ; ARB-11 TRANSFER TO I/O-PORT OUTPUT
        ;  LOW  BYTE 'READY' WHEN PORT REQUESTS DATA
        ;  I/O-PORT LOW BYTE CA2='0' WHEN DATA IS AT THE PORT
        ;
P.BSAL: LDA B   P.FNCT          ;CHECK  FUNCTION
        CMP B   #,7             ;LOADER ?
        BNE     1$              ;BRANCH IF NOT
        JSR     A.ALRD          ;GET BYTE
        JSR     LOADER          ;GO PROCESS BYTE
        JSR     A.BHWT+2        ;WRITE CHECK SUM
        RTS
1$:     JSR     A.ALDN          ;HOLD CA2='1'
        JSR     B.BLDN          ;RESTROBE CONTROL
        JSR     B.BLUP
        JSR     A.AWRD+2        ;READ WORD DATA
        JSR     B.BWWT+2        ;TRANSFER READY OR NOT !
        JSR     A.ALUP          ;LEAVE CA2='1'
        RTS
        ;
```

```
P.IOBL: JSR     A.ALRD+2        ;CA2='0', PORT READY
        JSR     B.BLDN          ;DISABLE CONTROL
        JSR     B.BLRD+2        ;CLEAR INTERRUPT
        JSR     B.BLUP          ;RE-ENABLE
        RTS
        ;
        ;
        ; I/O-PORT INPUT TRANSFER TO ARB-11
        ;  LOW  BYTE 'READY' WHEN DATA IS AVAILABLE
        ;  I/O-PORT LOW  BYTE CA2='0' WHEN I/O-PORT DATA HAS BEEN READ
        ;  I/O-PORT HIGH BYTE CA2='0' WHEN DATA IS TAKEN BY ARB-11
        ;
P.IOAL: JSR     B.AHDN          ;HOLD CA2='1'
        JSR     B.AWRD+2        ;READ I/O-PORT DATA
        JSR     A.BWWT+2        ;TRANSFER READY OR NOT !
        JSR     B.AHUP          ;LEAVE CA2='1'
        RTS
        ;
P.BSBL: JSR     B.AHDN          ;CA2='1'
        JSR     B.AHUP
        JSR     B.AHRD+2        ;CA2='0', DATA TAKEN
        JSR     A.BLDN          ;DISABLE CONTROL
        JSR     A.BLRD+2        ;CLEAR INTERRUPT
        JSR     A.BLUP          ;RE-ENABLE
        RTS
        ;
        .ENDC
        .IF     DF,$FAST
        ;       PORT SERVICE ROUTINES
        ;
        ; ARB-11 TRANSFER TO I/O-PORT OUTPUT
        ;  LOW  BYTE 'READY' WHEN PORT REQUESTS DATA
        ;  I/O-PORT LOW BYTE CA2='0' WHEN DATA IS AT THE PORT
        ;
P.BSAL: LDA B   P.FNCT          ;CHECK  FUNCTION
        CMP B   #,7             ;LOADER ?
        BNE     1$              ;BRANCH IF NOT
        JSR     A.ALRD          ;GET BYTE
        JSR     LOADER          ;GO PROCESS BYTE
        JSR     A.BHWT+2        ;WRITE CHECK SUM
        RTS
1$:     LDA B   A.APRT+7        ;HOLD CA2='1'
        STA B   CRAL+A.6821
        LDA B   B.BPRT+7        ;RESTROBE CONTROL
        STA B   CRBL+B.6821
        LDA B   B.BPRT+6
        STA B   CRBL+B.6821
        LDA B   PRBH+B.6821     ;CLEAR FLAGS
        LDA B   PRAH+A.6821     ;READ AND TRANSFER WORD
        STA B   PRBH+B.6821
        LDA B   PRBL+B.6821
        LDA B   PRAL+A.6821
        STA B   PRBL+B.6821
        LDA B   A.APRT+6        ;LEAVE CA2='1'
        STA B   CRAL+A.6821
        RTS
        ;
P.IOBL: LDA B   PRAL+A.6821     ;CA2='0', PORT READY
        LDA B   PRBL+B.6821     ;CLEAR INTERRUPT
        RTS
        ;
        ;
        ; I/O-PORT INPUT TRANSFER TO ARB-11
        ;  LOW  BYTE 'READY' WHEN DATA IS AVAILABLE
        ;  I/O-PORT LOW  BYTE CA2='0' WHEN I/O-PORT DATA HAS BEEN READ
        ;  I/O-PORT HIGH BYTE CA2='0' WHEN DATA IS TAKEN BY ARB-11
        ;
P.IOAL: LDA B   B.APRT+3        ;HOLD CA2='1'
        STA B   CRAH+B.6821
        LDA B   PRBH+A.6821     ;CLEAR FLAGS
        LDA B   PRAH+B.6821     ;AND TRANSFER WORD
        STA B   PRBH+A.6821
        LDA B   PRBL+A.6821
        LDA B   PRAL+B.6821
        STA B   PRBL+A.6821
        LDA B   B.APRT+2        ;LEAVE CA2='1'
```

```
        STA B   CRAH+B.6821
        RTS
        ;
P.BSBL: LDA B   B.APRT+3        ;CA2='1'
        STA B   CRAH+B.6821
        LDA B   B.APRT+2
        STA B   CRAH+B.6821
        LDA B   PRAH+B.6821     ;CA2='0', DATA TAKEN
        LDA B   PRBL+A.6821     ;CLEAR INTERRUPT
        RTS
        ;
        .ENDC
        ;
P.BSAH: JSR     A.AHRD          ;READ CONTROL CODE
        STA B   P.FNCT          ;SAVE FUNCTION
        CMP B   #,7             ;LOADER ?
        BNE     1$              ;BRANCH IF NOT
        JSR     LOAD$I          ;ELSE INITIALIZE LOADER
        LDA B   #,0             ;ZERO CHECK SUM
        JSR     A.BHWT+2        ;WRITE CHECK SUM
1$:     RTS
        ;
```

```
        .PAGE
        .SBTTL  ABSOLUTE BINARY LOADER
        ;
        ;         THIS LOADER ROUTINE IS AN ADAPTATION OF THE
        ;       'DEC' PAPER TAPE ABSOLUTE BINARY LOADER.
        ;         THE LOADER PROVIDES A MEANS OF LOADING OBJECT
        ;       CODE INTO ANY REGION OF MEMORY AND STARTING
        ;       PROGRAM EXECUTION.
        ;
        ;         AN ODD JUMP ADDRESS TERMINATES LOADER
        ;         AN EVEN JUMP ADDRESS TRANSFERS CONTROL TO THE JUMP ADDRESS
        ;
        .IF     NDF,$LOADER
        VARSAV  =0      ;VARIABLE SPACE
        PGMSAV  =100000 ;PROGRAM SPACE
        .=PGMSAV
        .ENDC
        ;
        PGMSAV=.
        .=VARSAV
LOAD$V: .BYTE   0,0     ;SERVICE ADDRESS
L.BYTC: .BYTE   0,0     ;BYTE COUNTER
L.ADDR: .BYTE   0,0     ;LOAD ADDRESS
L.CKSM: .BYTE   0       ;CHECK SUM
        VARSAV=.
        .=PGMSAV
        ;
        ;       INITIALIZE ENTRY POINT
        ;
LOAD$I: LDX     #,L.LDR0        ;FIRST ENTRY
        STX     LOAD$V
        RTS
        ;
        .PAGE
        .SBTTL  LOADER ROUTINE
        ;
        ;       ENTER WITH DATA IN <B>
        ;       EXITS WITH CURRENT CHECK SUM IN <B>
        ;
LOADER: LDA A   L.CKSM          ;UPDATE CHECK SUM
        ABA
        STA A   L.CKSM
        LDX     L.BYTC          ;UPDATE BYTE COUNT
        DEX
        STX     L.BYTC
        LDX     LOAD$V          ;DISPATCH ADDRESS
        JMP     0,X             ;GO TO IT
L.GBYT: TSX                     ;GET RETURN ADDRESS ON STACK
        LDX     0,X
        STX     LOAD$V          ;SAVE RETURN ADDRESS
        INS                     ;POP RETURN FROM STACK
        INS
        LDA B   L.CKSM          ;RETURN CHECK SUM IN <B>
        RTS                     ;BACK TO CALLER
        ;
L.LDR0: CLR     L.CKSM          ;CLEAR CHECK SUM
        BRA     L.LDR2
L.LDR1: CLR     L.CKSM          ;CLEAR CHECK SUM
        BSR     L.GBYT          ;BACK FOR A BYTE
L.LDR2: DEC B                   ;A '1' ?
        BNE     L.LDR1          ;LOOP UNTIL '1' FOUND
        BSR     L.GBYT          ;BACK FOR A BYTE
        TST B                   ;= '0' ?
        BNE     L.LDR0          ;LOOP BACK UNTIL '1','0' SEQUENCE
        BSR     L.GBYT          ;BACK FOR A BYTE
        STA B   L.ADDR+1        ;NOW SAVE BYTE COUNT
        BSR     L.GBYT          ;BACK FOR A BYTE
        LDA A   L.ADDR+1        ;RETRIEVE LOW ORDER COUNT
        SUB A   #,4             ;CORRECT BYTE COUNT
        SBC B   #,0
        STA A   L.BYTC+1        ;AND SAVE IT
        STA B   L.BYTC
        LDX     L.BYTC          ;L.BYTC=2 ?
        CPX     #,2
        BNE     L.LDR4          ;BRANCH IF NOT
L.LDR3: BSR     L.GBYT          ;BACK FOR A BYTE
```

```
        STA B   L.ADDR+1        ;GET JUMP ADDRESS
        BSR     L.GBYT          ;BACK FOR A BYTE
        STA B   L.ADDR
        BSR     L.GBYT          ;BACK FOR CHECK SUM BYTE !
        TST     L.CKSM          ;CHECK FOR ERROR
        BNE     L.LDR6          ;SKIP ON ERROR
        LDA A   L.ADDR+1        ;CHECK FOR ODD ADDRESS
        BIT A   #,1
        BNE     L.LDR6          ;TERMINATE BY RESTARTING SCAN
        LDX     L.ADDR          ;ELSE GO TO ADDRESS
        JSR     0,X
        BRA     L.LDR6          ;ON QUICK RETURN - CONTINUE
L.LDR4: BSR     L.GBYT          ;BACK FOR A BYTE
        STA B   L.ADDR+1        ;GET LOAD ADDRESS
        BSR     L.GBYT          ;BACK FOR A BYTE
        STA B   L.ADDR
L.LDR5: BSR     L.GBYT          ;BACK FOR A BYTE
        LDX     L.BYTC          ;CHECK BYTE COUNT
        BMI     L.LDR6          ;BRANCH IF DONE
        LDX     L.ADDR          ;GET LOAD ADDRESS
        STA B   0,X             ;STORE DATA
        INX                     ;UPDATE LOAD ADDRESS
        STX     L.ADDR
        BRA     L.LDR5
L.LDR6: BSR     L.GBYT          ;CURRENT BYTE WAS CHECK SUM RESULT
        BRA     L.LDR0          ;START NEW SEQUENCE
        ;
```

```
        .PAGE
        .SBTTL  A.6821 PERIPHERAL INTERFACE SOFTWARE
        ;
        ;         THIS SOFTWARE PACKAGE SUPPORTS THE
        ;       6821 PERIPHERAL INTERFACE ADAPTER (PIA).
        ;         ENTRY POINTS TO THE SOFTWARE ARE DEFINED
        ;       FOR EXTERNAL ACCESS:
        ;
        ;         THE FOLLOWING DATA DIRECTION ROUTINES ARE
        ;       ENTERED WITH
        ;               <B> = DIRECTION CONTROL
        ;       BY A JSR ____
        ;
        ;       A.AHDR --> A-PORT <15:8> DIRECTION
        ;       A.ALDR --> A-PORT <7:0> DIRECTION
        ;       A.BHDR --> B-PORT <15:8> DIRECTION
        ;       A.BLDR --> B-PORT <7:0> DIRECTION
        ;
        ;         THE FOLLOWING CONTROL ROUTINES ARE ENTERED WITH
        ;               <X> = SERVICE ADDRESS
        ;               <B> = INTERRUPT ENABLE CONTROL
        ;               <A> = INTERRUPT DISABLE CONTROL
        ;       BY A JSR ____
        ;
        ;       A.AHDF --> A-PORT <15:8>
        ;       A.ALDF --> A-PORT <7:0>
        ;       A.BHDF --> B-PORT <15:8>
        ;       A.BLDF --> B-PORT <7:0>
        ;
        ;         THE FOLLOWING INTERRUPT ENABLE ROUTINES ARE
        ;       ENTERED BY JSR ____
        ;
        ;       A.AHUP --> A-PORT <15:8> INTERRUPT ENABLE
        ;       A.ALUP --> A-PORT <7:0> INTERRUPT ENABLE
        ;       A.BHUP --> B-PORT <15:0> INTERRUPT ENABLE
        ;       A.BLUP --> B-PORT <7:0> INTERRUPT ENABLE
        ;
        ;         THE FOLLOWING INTERRUPT DISABLE ROUTINES ARE
        ;       ENTERED BY JSR ____
        ;
        ;       A.AHDN --> A-PORT <15:0> INTERRUPT DISABLE
        ;       A.ALDN --> A-PORT <7:0> INTERRUPT DISABLE
        ;       A.BHDN --> B-PORT <15:8> INTERRUPT DISABLE
        ;       A.BLDN --> B-PORT <7:0> INTERRUPT DISABLE
        ;
        ;         THE FOLLOWING ARE INTERRUPT ENTRY POINTS
        ;       FOR THE DUAL PIA PORTS
        ;
        ;       A.AH$I --> A-PORT <15:8> IRQ
        ;
        ;       A.AL$I --> A-PORT <7:0> IRQ
        ;
        ;       A.BH$I --> B-PORT <15:8> IRQ
        ;
        ;       A.BL$L --> B-PORT <7:0> IRQ
        ;
        .PAGE
        ;
        ;         THE FOLLOWING ENTRY POINTS ARE USED TO TRANSFER
        ;       DATA TO THE VARIOUS PIA I/O REGISTERS. THE MAIN
        ;       ENTRY POINT VERIFIES THAT AT LEAST ONE OF THE
        ;       PORT IRQ FLAGS IS SET BEFORE TRANSFERRING DATA.
        ;       IF DATA IS TRANSFERED THEN THE 'C' BIT = 1,
        ;       IF THE PORT WAS NOT READY THEN NO TRANSFER
        ;       IS PERFORMED AND THE 'C' BIT = 0.
        ;         THE ROUTINES MAY BE ENTERED AT THE ENTRY POINT + 2
        ;       IF NO IRQ FLAG CHECKS ARE TO BE MADE.
        ;         BYTE DATA IS PASSED IN <B>.
        ;         WORD DATA IS PASSED IN <B,A>
        ;
        ;       CALL BY JSR ____
        ;
        ;       A.AHRD --> A-PORT <15:8> READ
        ;       A.ALRD --> A-PORT <7:0> READ
        ;       A.AWRD --> A-PORT <15:0> READ
        ;       A.BHRD --> B-PORT <15:8> READ
```

```
;       A.BLRD --> B-PORT <7:0> READ
;       A.BWRD --> B-PORT <15:0> READ
;
;
;       A.AHWT --> A-PORT <15:8> WRITE
;       A.ALWT --> A-PORT <7:0> WRITE
;       A.AWWT --> A-PORT <15:0> WRITE
;       A.BHWT --> B-PORT <15:8> WRITE
;       A.BLWT --> B-PORT <7:0> WRITE
;       A.BWWT --> B-PORT <15:0> WRITE
;
.PAGE
.SBTTL  A.6821 PIA HARDWARE
;
;         THE PIA CONFIGURATION  CONSISTS OF TWO
;       MC6821 PERIPHERAL INTERFACE ADAPTERS (PIA'S)
;       WHICH MAY BE ADDRESSED AS 16-BIT PORTS.
;       ONE PIA UNIT FORMS THE HIGH ORDER PART <15:8>
;       AND THE OTHER THE LOW ORDER PART <7:0>
;
; REGISTER 0
CRAH    =0      ;A-PORT <15:8> CONTROL REGISTER
;
; REGISTER 1
CRAL    =1      ;A-PORT <7:0> CONTROL REGISTER
;
; REGISTER 2
CRBH    =2      ;B-PORT <15:8> CONTROL REGISTER
;
; REGISTER 3
CRBL    =3      ;B-PORT <7:0> CONTROL REGISTER
;
; REGISTER 4
DDRAH   =4      ;CRAH<2>=0, A-PORT <15:8> DATA DIRECTION REGISTER
PRAH    =4      ;CRAH<2>=1, A-PORT <15:8> DATA REGISTER
;
; REGISTER 5
DDRAL   =5      ;CRAL<2>=0, A-PORT <7:0> DATA DIRECTION REGISTER
PRAL    =5      ;CRAL<2>=1, A-PORT <7:0> DATA REGISTER
;
; REGISTER 6
DDRBH   =6      ;CRBH<2>=0, B-PORT <15:8> DATA DIRECTION REGISTER
PRBH    =6      ;CRBH<2>=1, B-PORT <15:8> DATA REGISTER
;
; REGISTER 7
DDRBL   =7      ;CRBL<2>=0, B-PORT <7:0> DATA DIRECTION REGISTER
PRBL    =7      ;CRBL<2>=1, B-PORT <7:0> DATA REGISTER
;
.PAGE
;
;       A-PORT CONTROL REGISTERS
;
; CONTROL REGISTERS CRA_  - READ/WRITE REGISTERS
;                                 CONTROLLING INTERRUPTS AND STROBES
;
; BIT 1 0       CONTROL OF INTERRUPT INPUT CA1
;               ACTIVE TRANSITION
;     _ _       TO SET CRA<7>           /IRQA OUTPUT
;     0 0               (-)             DISABLED
;     0 1               (-)             /CRA<7>
;     1 0               (+)             DISABLED
;     1 1               (+)             /CRA<7>
;       CRA<7> CLEARED BY A READ OF CORRESPONDING PRA_
;       /IRQA FOLLOWS STATE OF /CRA<7> WHEN ENABLED
;
; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
;     _
;     0         ACCESS DATA DIRECTION REGISTERS
;     1         ACCESS PORT DATA REGISTER
;
; BIT 5 4 3     CONTROL OF CA2 AS AN INTERRUPT INPUT
;               ACTIVE TRANSITION
;     _ _ _     TO SET CRA<6>           /IRQB OUTPUT
;     0 0 0             (-)             DISABLED
;     0 0 1             (-)             /CRA<6>
;     0 1 0             (+)             DISABLED
```

```
;     0 1 1             (+)             /CRA<6>
;       CRA<6> CLEARED BY READ OF CORRESPONDING PRA_
;       /IRQB FOLLOWS STATE OF CRA<6> WHEN ENABLED
;
; BIT 5 4 3     CONTROL OF CA2 AS AN OUTPUT
;     _ _ _     CA2 LOW                 CA2 HIGH
;     1 0 0     AFTER READ TO PRA_      WHEN CRA<7> IS SET
;               ON (+) OF NEXT E        BY CA1 TRANSITION
;     1 0 1     AFTER READ TO PRA_      ON SECOND (+) E AFTER
;               ON (+) OF NEXT E        PIA DESELECTED
;     1 1 0     =CRA<3>
;     1 1 1                             =CRA<3>
;
; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CA2, CLEARED BY READ OF PRA_
;
; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CA1, CLEARED BY READ OF PRA_
;
;       DATA DIRECTION REGISTER (DDRA_)
; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
;               BY 0'S AND OUTPUT BITS BY 1'S.
;
;       PERIPHERAL DATA REGISTER (PRA_)
; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGISTERS
;
.PAGE
.SBTTL  A.6821 DETAILS
;
;       B-PORT CONTROL REGISTERS
;
; CONTROL REGISTERS CRB_  - READ/WRITE REGISTERS
;                               CONTROLLING INTERRUPTS AND STROBES
;
; BIT 1 0       CONTROL OF INTERRUPT INPUT CB1
;               ACTIVE TRANSITION
;     _ _       TO SET CRB<7>           /IRQA OUTPUT
;     0 0               (-)             DISABLED
;     0 1               (-)             /CRB<7>
;     1 0               (+)             DISABLED
;     1 1               (+)             /CRB<7>
;       CRB<7> CLEARED BY A READ OF CORRESPONDING PRB_
;       /IRQA FOLLOWS STATE OF /CRB<7> WHEN ENABLED
;
; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
;
;     _
;     0         ACCESS DATA DIRECTION REGISTERS
;     1         ACCESS PORT DATA REGISTER
;
; BIT 5 4 3     CONTROL OF CB2 AS AN INTERRUPT INPUT
;               ACTIVE TRANSITION
;     _ _ _     TO SET CRB<6>           /IRQB OUTPUT
;     0 0 0             (-)             DISABLED
;     0 0 1             (-)             /CRB<6>
;     0 1 0             (+)             DISABLED
;     0 1 1             (+)             /CRB<6>
;       CRB<6> CLEARED BY READ OF CORRESPONDING PRB_
;       /IRQB FOLLOWS STATE OF CRB<6> WHEN ENABLED
;
; BIT 5 4 3     CONTROL OF CB2 AS AN OUTPUT
;     _ _ _     CB2 LOW                 CB2 HIGH
;     1 0 0     AFTER WRITE TO PRB      WHEN CRB<7> IS SET
;               ON (+) OF NEXT E        BY CB1 TRANSITION
;     1 0 1     AFTER WRITE TO PRB      ON SECOND (+) E AFTER
;               ON (+) OF NEXT E        PIA DESELECTED
;     1 1 0     =CRB<3>
;     1 1 1                             =CRB<3>
;
; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CB2, CLEARED BY READ OF PRB_
;
; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CB1, CLEARED BY READ OF PRB_
;
;       DATA DIRECTION REGISTER (DDRB_)
```

```
        ; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
        ;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
        ;               BY 0'S AND OUTPUT BITS BY 1'S.
        ;
        ;       PERIPHERAL DATA REGISTER (PRB_)
        ; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGISTERS
        ;
        .PAGE
        .SBTTL  A.6821 VARIABLES AND POINTERS
        ;
        ;       PORT STATUS AND BUFFERSS
        ;
        .IF     NDF,$A.6821
        A.6821  =       200     ;PIA PORT DEFAULT ADDRESS
        VARSAV  =       0       ;DEFAULT VARIABLE ADDRESS
        PGMSAV  =       100000  ;DEFAULT PROGRAM ADDRESS
        .=PGMSAV
        .ENDC
        ;
        PGMSAV=.
        .=VARSAV
        ;
A.APRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
        .BYTE   0,0     ;<7:0> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
A.BPRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
        .BYTE   0,0     ;<7:0> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
        ;
        VARSAV=.
        .=PGMSAV
        ;
        .PAGE
        .SBTTL  A.6821 A-PORT CONTROL ROUTINES
        ;
        ;       DATA DIRECTION LOAD ROUTINES
        ;       ENTER WITH DIRECTION CODE IN <B>
        ;
A.AHDR: LDA A   CRAH+A.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRAH+A.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRAH+A.6821    ;SET DIRECTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRAH+A.6821     ;AND RESTORE
        RTS
        ;
A.ALDR: LDA A   CRAL+A.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRAL+A.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRAL+A.6821    ;SET DIRCTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRAL+A.6821     ;AND RESTORE
        RTS
        ;
        .PAGE
        ;       CONTROL AND SERVICE VECTOR LOADER
        ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
        ;       'UP' CONTROL IN <B>, AND
        ;       'DN' CONTROL IN <A>
        ;
A.AHDF: STX     A.APRT          ;SAVE SERVICE ROUTINE ADDRESS
        STA B   A.APRT+2        ;'UP' CONTROL
        STA A   A.APRT+3        ;'DN' CONTROL
        RTS
        ;
A.ALDF: STX     A.APRT+4        ;SAVE SERVICE ROUTINE ADDRESS
        STA B   A.APRT+6        ;'UP' CONTROL
        STA A   A.APRT+7        ;'DN' CONTROL
        RTS
        ;
        ;
        ;       INTERRUPT CONTROL LOADERS
```

```
        ;
A.AHUP: LDA B   A.APRT+2        ;'UP' CONTROL
        STA B   CRAH+A.6821
        RTS
        ;
A.ALUP: LDA B   A.APRT+6        ;'UP' CONTROL
        STA B   CRAL+A.6821
        RTS
        ;
A.AHDN: LDA B   A.APRT+3        ;'DN' CONTROL
        STA B   CRAH+A.6821
        RTS
        ;
A.ALDN: LDA B   A.APRT+7        ;'DN' CONTROL
        STA B   CRAL+A.6821
        RTS
        ;
        .PAGE
        .SBTTL  A.6821 B-PORT CONTROL ROUTINES
        ;
        ;       DATA DIRECTION LOAD ROUTINES
        ;       ENTER WITH DIRECTION CODE IN <B>
        ;
A.BHDR: LDA A   CRBH+A.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRBH+A.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRBH+A.6821    ;SET DIRECTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRBH+A.6821     ;AND RESTORE
        RTS
        ;
A.BLDR: LDA A   CRBL+A.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRBL+A.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRBL+A.6821    ;SET DIRCTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRBL+A.6821     ;AND RESTORE
        RTS
        ;
        .PAGE
        ;       CONTROL AND SERVICE VECTOR LOADER
        ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
        ;       'UP' CONTROL IN <B>, AND
        ;       'DN' CONTROL IN <A>
        ;
A.BHDF: STX     A.BPRT          ;SAVE SERVICE ROUTINE ADDRESS
        STA B   A.BPRT+2        ;'UP' CONTROL
        STA A   A.BPRT+3        ;'DN' CONTROL
        RTS
        ;
A.BLDF: STX     A.BPRT+4        ;SAVE SERVICE ROUTINE ADDRESS
        STA B   A.BPRT+6        ;'UP' CONTROL
        STA A   A.BPRT+7        ;'DN' CONTROL
        RTS
        ;
        ;
        ;       INTERRUPT CONTROL LOADERS
        ;
A.BHUP: LDA B   A.BPRT+2        ;'UP' CONTROL
        STA B   CRBH+A.6821
        RTS
        ;
A.BLUP: LDA B   A.BPRT+6        ;'UP' CONTROL
        STA B   CRBL+A.6821
        RTS
        ;
A.BHDN: LDA B   A.BPRT+3        ;'DN' CONTROL
        STA B   CRBH+A.6821
        RTS
        ;
A.BLDN: LDA B   A.BPRT+7        ;'DN' CONTROL
        STA B   CRBL+A.6821
        RTS
        ;
```

```
        .PAGE
        .SBTTL  A.6821 A-PORT TRANSFERS
        ;
        ;       BYTE DATA IS TRANSFERED IN <B>
        ;       WORD DATA IS TRANSFERED IN <B,A>
        ;
A.AHRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRAH+A.6821     ;GET BYTE
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRAH+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
A.ALRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRAL+A.6821     ;GET BYTE
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRG FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
A.AWRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRAH+A.6821     ;GET HIGH BYTE
        LDA A   PRAL+A.6821     ;AND LOW BYTE
        SEC                     ;HAVE WORD
        RTS
2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
        .PAGE
        ;
A.AHWT: BRA     2$              ;TEST ENTRY
1$:     STA B   PRAH+A.6821     ;STORE BYTE
        TST     PRAH+A.6821     ;CLEAR IRQ FLAGS
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRAH+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
A.ALWT: BRA     2$              ;TEST ENTRY
1$:     STA B   PRAL+A.6821     ;STORE BYTE
        TST     PRAL+A.6821     ;CLEAR IRQ FLAGS
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
A.AWWT: BRA     2$              ;TEST ENTRY
1$:     STA B   PRAH+A.6821     ;SEND HIGH BYTE
        STA A   PRAL+A.6821     ;SEND LOW BYTE
        TST     PRAH+A.6821     ;CLEAR IRQ FLAGS
        TST     PRAL+A.6821     ;CLEAR IRQ FLAGS
        SEC                     ;WORD SENT
        RTS
2$:     PSH A                   ;SAVE A
        LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        PUL A                   ;RESTORE A
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
```

```
        .PAGE
        .SBTTL  A.6821 B-PORT TRANSFERS
        ;
        ;       BYTE DATA IS TRANSFERED IN <B>
        ;       WORD DATA IS TRANSFERED IN <B,A>
        ;
A.BHRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRBH+A.6821     ;GET BYTE
        STA B   PRBH+A.6821     ;DO CONTROL
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRBH+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
A.BLRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRBL+A.6821     ;GET BYTE
        STA B   PRBL+A.6821     ;DO CONTROL
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRG FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
A.BWRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRBH+A.6821     ;GET HIGH BYTE
        LDA A   PRBL+A.6821     ;AND LOW BYTE
        STA B   PRBH+A.6821     ;DO CONTROL
        STA A   PRBL+A.6821     ;DO CONTROL
        SEC                     ;HAVE WORD
        RTS
2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
        .PAGE
        ;
A.BHWT: BRA     2$              ;TEST ENTRY
1$:     TST     PRBH+A.6821     ;CLEAR IRQ FLAGS
        STA B   PRBH+A.6821     ;STORE BYTE
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRBH+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
A.BLWT: BRA     2$              ;TEST ENTRY
1$:     TST     PRBL+A.6821     ;CLEAR IRQ FLAGS
        STA B   PRBL+A.6821     ;STORE BYTE
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
A.BWWT: BRA     2$              ;TEST ENTRY
1$:     TST     PRBH+A.6821     ;CLEAR IRQ FLAGS
        TST     PRBL+A.6821     ;CLEAR IRQ FLAGS
        STA B   PRBH+A.6821     ;SEND HIGH BYTE
        STA A   PRBL+A.6821     ;SEND LOW BYTE
        SEC                     ;WORD SENT
        RTS
2$:     PSH A                   ;SAVE A
        LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        PUL A                   ;RESTORE A
```

```
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
        .PAGE
        .SBTTL  A.6821 INTERRUPT SERVICE DISPATCH ROUTINES
        ;
A.AH$I: LDX     A.APRT          ;GET SERVICE ADDRESS
        BNE     A.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRAH+A.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     A.IRQD          ;TERMINATE INTERRUPTS
        ;
A.AL$I: LDX     A.APRT+4        ;GET SERVICE ADDRESS
        BNE     A.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRAL+A.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     A.IRQD          ;TERMINATE INTERRUPTS
        ;
A.BH$I: LDX     A.BPRT          ;GET SERVICE ADDRESS
        BNE     A.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRBH+A.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     A.IRQD          ;TERMINATE INTERRUPTS
        ;
A.BL$I: LDX     A.BPRT+4        ;GET SERVICE ADDRESS
        BNE     A.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRBL+A.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     A.IRQD          ;TERMINATE INTERRUPTS
        ;
A.IRQG: JSR     0,X             ;DO SERVICE ROUTINE
        RTI
        ;
A.IRQD: LDA A   0,X             ;GET CONTROL REGISTER
        AND B   #,376           ;CLEAR CX1 INTERRUPT ENABLE
        BIT B   #,40            ;CX2 INTERRUPT MODE ?
        BNE     1$              ;BRANCH IF NOT
        AND B   #,367           ;CLEAR CX2 INTERRUPT ENABLE
1$:     STA B   0,X             ;SET NEW CONTROL
        RTI
        ;
        .IIF    NDF,$A.6821     .END
```

```
        .PAGE
        .SBTTL  B.6821 PERIPHERAL INTERFACE SOFTWARE
        ;
        ;         THIS SOFTWARE PACKAGE SUPPORTS THE
        ;       6821 PERIPHERAL INTERFACE ADAPTER (PIA).
        ;         ENTRY POINTS TO THE SOFTWARE ARE DEFINED
        ;       FOR EXTERNAL ACCESS:
        ;
        ;         THE FOLLOWING DATA DIRECTION ROUTINES ARE
        ;       ENTERED WITH
        ;               <B> = DIRECTION CONTROL
        ;       BY A JSR ____
        ;
        ;       B.AHDR --> A-PORT <15:8> DIRECTION
        ;       B.ALDR --> A-PORT <7:0> DIRECTION
        ;       B.BHDR --> B-PORT <15:8> DIRECTION
        ;       B.BLDR --> B-PORT <7:0> DIRECTION
        ;
        ;         THE FOLLOWING CONTROL ROUTINES ARE ENTERED WITH
        ;               <X> = SERVICE ADDRESS
        ;               <B> = INTERRUPT ENABLE CONTROL
        ;               <A> = INTERRUPT DISABLE CONTROL
        ;       BY A JSR ____
        ;
        ;       B.AHDF --> A-PORT <15:8>
        ;       B.ALDF --> A-PORT <7:0>
        ;       B.BHDF --> B-PORT <15:8>
        ;       B.BLDF --> B-PORT <7:0>
        ;
        ;         THE FOLLOWING INTERRUPT ENABLE ROUTINES ARE
        ;       ENTERED BY JSR ____
        ;
        ;       B.AHUP --> A-PORT <15:8> INTERRUPT ENABLE
        ;       B.ALUP --> A-PORT <7:0> INTERRUPT ENABLE
        ;       B.BHUP --> B-PORT <15:0> INTERRUPT ENABLE
        ;       B.BLUP --> B-PORT <7:0> INTERRUPT ENABLE
        ;
        ;         THE FOLLOWING INTERRUPT DISABLE ROUTINES ARE
        ;       ENTERED BY JSR ____
        ;
        ;       B.AHDN --> A-PORT <15:0> INTERRUPT DISABLE
        ;       B.ALDN --> A-PORT <7:0> INTERRUPT DISABLE
        ;       B.BHDN --> B-PORT <15:8> INTERRUPT DISABLE
        ;       B.BLDN --> B-PORT <7:0> INTERRUPT DISABLE
        ;
        ;         THE FOLLOWING ARE INTERRUPT ENTRY POINTS
        ;       FOR THE DUAL PIA PORTS
        ;
        ;       B.AH$I --> A-PORT <15:8> IRQ
        ;
        ;       B.AL$I --> A-PORT <7:0> IRQ
        ;
        ;       B.BH$I --> B-PORT <15:8> IRQ
        ;
        ;       B.BL$L --> B-PORT <7:0> IRQ
        ;
        .PAGE
        ;
        ;         THE FOLLOWING ENTRY POINTS ARE USED TO TRANSFER
        ;       DATA TO THE VARIOUS PIA I/O REGISTERS. THE MAIN
        ;       ENTRY POINT VERIFIES THAT AT LEAST ONE OF THE
        ;       PORT IRQ FLAGS IS SET BEFORE TRANSFERRING DATA .
        ;       IF DATA IS TRANSFERED THEN THE 'C' BIT = 1,
        ;       IF THE PORT WAS NOT READY THEN NO TRANSFER
        ;       IS PERFORMED AND THE 'C' BIT = 0.
        ;         THE ROUTINES MAY BE ENTERED AT THE ENTRY POINT + 2
        ;       IF NO IRQ FLAG CHECKS ARE TO BE MADE.
        ;         BYTE DATA IS PASSED IN <B>.
        ;         WORD DATA IS PASSED IN <B,A>
        ;
        ;       CALL BY JSR ____
        ;
        ;       B.AHRD --> A-PORT <15:8> READ
        ;       B.ALRD --> A-PORT <7:0> READ
        ;       B.AWRD --> A-PORT <15:0> READ
        ;       B.BHRD --> B-PORT <15:8> READ
```

```
;       B.BLRD --> B-PORT <7:0> READ
;       B.BWRD --> B-PORT <15:0> READ
;
;
;       B.AHWT --> A-PORT <15:8> WRITE
;       B.ALWT --> A-PORT <7:0> WRITE
;       B.AWWT --> A-PORT <15:0> WRITE
;       B.BHWT --> B-PORT <15:8> WRITE
;       B.BLWT --> B-PORT <7:0> WRITE
;       B.BWWT --> B-PORT <15:0> WRITE
;
.PAGE
.SBTTL  B.6821 PIA HARDWARE
;
;         THE PIA CONFIGURATION  CONSISTS OF TWO
;       MC6821 PERIPHERAL INTERFACE ADAPTERS (PIA'S)
;       WHICH MAY BE ADDRESSED AS 16-BIT PORTS.
;       ONE PIA UNIT FORMS THE HIGH ORDER PART <15:8>
;       AND THE OTHER THE LOW ORDER PART <7:0>
;
; REGISTER 0
CRAH    =0      ;A-PORT <15:8> CONTROL REGISTER
;
; REGISTER 1
CRAL    =1      ;A-PORT <7:0> CONTROL REGISTER
;
; REGISTER 2
CRBH    =2      ;B-PORT <15:8> CONTROL REGISTER
;
; REGISTER 3
CRBL    =3      ;B-PORT <7:0> CONTROL REGISTER
;
; REGISTER 4
DDRAH   =4      ;CRAH<2>=0, A-PORT <15:8> DATA DIRECTION REGISTER
PRAH    =4      ;CRAH<2>=1, A-PORT <15:8> DATA REGISTER
;
; REGISTER 5
DDRAL   =5      ;CRAL<2>=0, A-PORT <7:0> DATA DIRECTION REGISTER
PRAL    =5      ;CRAL<2>=1, A-PORT <7:0> DATA REGISTER
;
; REGISTER 6
DDRBH   =6      ;CRBH<2>=0, B-PORT <15:8> DATA DIRECTION REGISTER
PRBH    =6      ;CRBH<2>=1, B-PORT <15:8> DATA REGISTER
;
; REGISTER 7
DDRBL   =7      ;CRBL<2>=0, B-PORT <7:0> DATA DIRECTION REGISTER
PRBL    =7      ;CRBL<2>=1, B-PORT <7:0> DATA REGISTER
;
.PAGE
;
;       A-PORT CONTROL REGISTERS
;
; CONTROL REGISTERS CRA_  - READ/WRITE REGISTERS
;                                 CONTROLLING INTERRUPTS AND STROBES
;
; BIT 1 0       CONTROL OF INTERRUPT INPUT CA1
;               ACTIVE TRANSITION
;     _ _       TO SET CRA<7>           /IRQA OUTPUT
;     0 0               (-)             DISABLED
;     0 1               (-)             /CRA<7>
;     1 0               (+)             DISABLED
;     1 1               (+)             /CRA<7>
;       CRA<7> CLEARED BY A READ OF CORRESPONDING PRA_
;       /IRQA FOLLOWS STATE OF /CRA<7> WHEN ENABLED
;
; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
;     _
;     0         ACCESS DATA DIRECTION REGISTERS
;     1         ACCESS PORT DATA REGISTER
;
; BIT 5 4 3     CONTROL OF CA2 AS AN INTERRUPT INPUT
;               ACTIVE TRANSITION
;     _ _ _     TO SET CRA<6>           /IRQB OUTPUT
;     0 0 0             (-)             DISABLED
;     0 0 1             (-)             /CRA<6>
;     0 1 0             (+)             DISABLED
```

```
;     0 1 1             (+)             /CRA<6>
;       CRA<6> CLEARED BY READ OF CORRESPONDING PRA_
;       /IRQB FOLLOWS STATE OF CRA<6> WHEN ENABLED
;
; BIT 5 4 3     CONTROL OF CA2 AS AN OUTPUT
;     _ _ _     CA2 LOW                 CA2 HIGH
;     1 0 0     AFTER READ TO PRA_      WHEN CRA<7> IS SET
;               ON (+) OF NEXT E        BY CA1 TRANSITION
;     1 0 1     AFTER READ TO PRA_      ON SECOND (+) E AFTER
;               ON (+) OF NEXT E        PIA DESELECTED
;     1 1 0     =CRA<3>
;     1 1 1                             =CRA<3>
;
; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CA2, CLEARED BY READ OF PRA_
;
; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CA1, CLEARED BY READ OF PRA_
;
;       DATA DIRECTION REGISTER (DDRA_)
; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
;               BY 0'S AND OUTPUT BITS BY 1'S.
;
;       PERIPHERAL DATA REGISTER (PRA_)
; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGISTERS
;
.PAGE
.SBTTL  B.6821 DETAILS
;
;       B-PORT CONTROL REGISTERS
;
; CONTROL REGISTERS CRB_  - READ/WRITE REGISTERS
;                               CONTROLLING INTERRUPTS AND STROBES
;
; BIT 1 0       CONTROL OF INTERRUPT INPUT CB1
;               ACTIVE TRANSITION
;     _ _       TO SET CRB<7>           /IRQA OUTPUT
;     0 0               (-)             DISABLED
;     0 1               (-)             /CRB<7>
;     1 0               (+)             DISABLED
;     1 1               (+)             /CRB<7>
;       CRB<7> CLEARED BY A READ OF CORRESPONDING PRB_
;       /IRQA FOLLOWS STATE OF /CRB<7> WHEN ENABLED
;
; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
;
;     _
;     0         ACCESS DATA DIRECTION REGISTERS
;     1         ACCESS PORT DATA REGISTER
;
; BIT 5 4 3     CONTROL OF CB2 AS AN INTERRUPT INPUT
;               ACTIVE TRANSITION
;     _ _ _     TO SET CRB<6>           /IRQB OUTPUT
;     0 0 0             (-)             DISABLED
;     0 0 1             (-)             /CRB<6>
;     0 1 0             (+)             DISABLED
;     0 1 1             (+)             /CRB<6>
;       CRB<6> CLEARED BY READ OF CORRESPONDING PRB_
;       /IRQB FOLLOWS STATE OF CRB<6> WHEN ENABLED
;
; BIT 5 4 3     CONTROL OF CB2 AS AN OUTPUT
;     _ _ _     CB2 LOW                 CB2 HIGH
;     1 0 0     AFTER WRITE TO PRB      WHEN CRB<7> IS SET
;               ON (+) OF NEXT E        BY CB1 TRANSITION
;     1 0 1     AFTER WRITE TO PRB      ON SECOND (+) E AFTER
;               ON (+) OF NEXT E        PIA DESELECTED
;     1 1 0     =CRB<3>
;     1 1 1                             =CRB<3>
;
; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CB2, CLEARED BY READ OF PRB_
;
; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRANSITION
;                       OF CB1, CLEARED BY READ OF PRB_
;
;       DATA DIRECTION REGISTER (DDRB_)
```

```
        ; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
        ;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
        ;               BY 0'S AND OUTPUT BITS BY 1'S.
        ;
        ;       PERIPHERAL DATA REGISTER (PRB_)
        ; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGISTERS
        ;
        .PAGE
        .SBTTL  B.6821 VARIABLES AND POINTERS
        ;
        ;       PORT STATUS AND BUFFERSS
        ;
        .IF     NDF,$B.6821
        B.6821  =       210     ;PIA PORT DEFAULT ADDRESS
        VARSAV  =       0       ;DEFAULT VARIABLE ADDRESS
        PGMSAV  =       100000  ;DEFAULT PROGRAM ADDRESS
        .=PGMSAV
        .ENDC
        ;
        PGMSAV=.
        .=VARSAV
        ;
B.APRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
        .BYTE   0,0     ;<7:0> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
B.BPRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
        .BYTE   0,0     ;<7:0> SERVICE VECTOR
        .BYTE   0,0     ;'UP','DN' CONTROL
        ;
        VARSAV=.
        .=PGMSAV
        ;
        .PAGE
        .SBTTL  B.6821 A-PORT CONTROL ROUTINES
        ;
        ;       DATA DIRECTION LOAD ROUTINES
        ;       ENTER WITH DIRECTION CODE IN <B>
        ;
B.AHDR: LDA A   CRAH+B.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRAH+B.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRAH+B.6821    ;SET DIRECTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRAH+B.6821     ;AND RESTORE
        RTS
        ;
B.ALDR: LDA A   CRAL+B.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRAL+B.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRAL+B.6821    ;SET DIRCTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRAL+B.6821     ;AND RESTORE
        RTS
        ;
        .PAGE
        ;       CONTROL AND SERVICE VECTOR LOADER
        ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
        ;       'UP' CONTROL IN <B>, AND
        ;       'DN' CONTROL IN <A>
        ;
B.AHDF: STX     B.APRT          ;SAVE SERVICE ROUTINE ADDRESS
        STA B   B.APRT+2        ;'UP' CONTROL
        STA A   B.APRT+3        ;'DN' CONTROL
        RTS
        ;
B.ALDF: STX     B.APRT+4        ;SAVE SERVICE ROUTINE ADDRESS
        STA B   B.APRT+6        ;'UP' CONTROL
        STA A   B.APRT+7        ;'DN' CONTROL
        RTS
        ;
        ;
        ;       INTERRUPT CONTROL LOADERS
```

```
        ;
B.AHUP: LDA B   B.APRT+2        ;'UP' CONTROL
        STA B   CRAH+B.6821
        RTS
        ;
B.ALUP: LDA B   B.APRT+6        ;'UP' CONTROL
        STA B   CRAL+B.6821
        RTS
        ;
B.AHDN: LDA B   B.APRT+3        ;'DN' CONTROL
        STA B   CRAH+B.6821
        RTS
        ;
B.ALDN: LDA B   B.APRT+7        ;'DN' CONTROL
        STA B   CRAL+B.6821
        RTS
        ;
        .PAGE
        .SBTTL  B.6821 B-PORT CONTROL ROUTINES
        ;
        ;       DATA DIRECTION LOAD ROUTINES
        ;       ENTER WITH DIRECTION CODE IN <B>
        ;
B.BHDR: LDA A   CRBH+B.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRBH+B.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRBH+B.6821    ;SET DIRECTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRBH+B.6821     ;AND RESTORE
        RTS
        ;
B.BLDR: LDA A   CRBL+B.6821     ;GET CURRENT CONTROL
        PSH A                   ;AND SAVE
        AND A   #,373           ;MASK DDR BIT
        STA A   CRBL+B.6821     ;ACCESS DIRECTION REGISTER
        STA B   DDRBL+B.6821    ;SET DIRCTIONS
        PUL A                   ;GET SAVED CONTROL
        STA A   CRBL+B.6821     ;AND RESTORE
        RTS
        ;
        .PAGE
        ;       CONTROL AND SERVICE VECTOR LOADER
        ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
        ;       'UP' CONTROL IN <B>, AND
        ;       'DN' CONTROL IN <A>
        ;
B.BHDF: STX     B.BPRT          ;SAVE SERVICE ROUTINE ADDRESS
        STA B   B.BPRT+2        ;'UP' CONTROL
        STA A   B.BPRT+3        ;'DN' CONTROL
        RTS
        ;
B.BLDF: STX     B.BPRT+4        ;SAVE SERVICE ROUTINE ADDRESS
        STA B   B.BPRT+6        ;'UP' CONTROL
        STA A   B.BPRT+7        ;'DN' CONTROL
        RTS
        ;
        ;
        ;       INTERRUPT CONTROL LOADERS
        ;
B.BHUP: LDA B   B.BPRT+2        ;'UP' CONTROL
        STA B   CRBH+B.6821
        RTS
        ;
B.BLUP: LDA B   B.BPRT+6        ;'UP' CONTROL
        STA B   CRBL+B.6821
        RTS
        ;
B.BHDN: LDA B   B.BPRT+3        ;'DN' CONTROL
        STA B   CRBH+B.6821
        RTS
        ;
B.BLDN: LDA B   B.BPRT+7        ;'DN' CONTROL
        STA B   CRBL+B.6821
        RTS
        ;
```

```
        .PAGE
        .SBTTL  B.6821 A-PORT TRANSFERS
        ;
        ;       BYTE DATA IS TRANSFERED IN <B>
        ;       WORD DATA IS TRANSFERED IN <B,A>
        ;
B.AHRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRAH+B.6821     ;GET BYTE
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRAH+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
B.ALRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRAL+B.6821     ;GET BYTE
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRAL+B.6821     ;CHECK FOR IRG FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
B.AWRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRAH+B.6821     ;GET HIGH BYTE
        LDA A   PRAL+B.6821     ;AND LOW BYTE
        SEC                     ;HAVE WORD
        RTS
2$:     LDA A   CRAL+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
        .PAGE
        ;
B.AHWT: BRA     2$              ;TEST ENTRY
1$:     STA B   PRAH+B.6821     ;STORE BYTE
        TST     PRAH+B.6821     ;CLEAR IRQ FLAGS
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRAH+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
B.ALWT: BRA     2$              ;TEST ENTRY
1$:     STA B   PRAL+B.6821     ;STORE BYTE
        TST     PRAL+B.6821     ;CLEAR IRQ FLAGS
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRAL+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
B.AWWT: BRA     2$              ;TEST ENTRY
1$:     STA B   PRAH+B.6821     ;SEND HIGH BYTE
        STA A   PRAL+B.6821     ;SEND LOW BYTE
        TST     PRAH+B.6821     ;CLEAR IRQ FLAGS
        TST     PRAL+B.6821     ;CLEAR IRQ FLAGS
        SEC                     ;WORD SENT
        RTS
2$:     PSH A                   ;SAVE A
        LDA A   CRAL+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        PUL A                   ;RESTORE A
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
```

```
        .PAGE
        .SBTTL  B.6821 B-PORT TRANSFERS
        ;
        ;       BYTE DATA IS TRANSFERED IN <B>
        ;       WORD DATA IS TRANSFERED IN <B,A>
        ;
B.BHRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRBH+B.6821     ;GET BYTE
        STA B   PRBH+B.6821     ;DO CONTROL
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRBH+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
B.BLRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRBL+B.6821     ;GET BYTE
        STA B   PRBL+B.6821     ;DO CONTROL
        SEC                     ;HAVE BYTE
        RTS
2$:     LDA A   CRBL+B.6821     ;CHECK FOR IRG FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
B.BWRD: BRA     2$              ;TEST ENTRY
1$:     LDA B   PRBH+B.6821     ;GET HIGH BYTE
        LDA A   PRBL+B.6821     ;AND LOW BYTE
        STA B   PRBH+B.6821     ;DO CONTROL
        STA A   PRBL+B.6821     ;DO CONTROL
        SEC                     ;HAVE WORD
        RTS
2$:     LDA A   CRBL+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;NO DATA
        RTS
        ;
        .PAGE
        ;
B.BHWT: BRA     2$              ;TEST ENTRY
1$:     TST     PRBH+B.6821     ;CLEAR IRQ FLAGS
        STA B   PRBH+B.6821     ;STORE BYTE
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRBH+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
B.BLWT: BRA     2$              ;TEST ENTRY
1$:     TST     PRBL+B.6821     ;CLEAR IRQ FLAGS
        STA B   PRBL+B.6821     ;STORE BYTE
        SEC                     ;BYTE SENT
        RTS
2$:     LDA A   CRBL+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
B.BWWT: BRA     2$              ;TEST ENTRY
1$:     TST     PRBH+B.6821     ;CLEAR IRQ FLAGS
        TST     PRBL+B.6821     ;CLEAR IRQ FLAGS
        STA B   PRBH+B.6821     ;SEND HIGH BYTE
        STA A   PRBL+B.6821     ;SEND LOW BYTE
        SEC                     ;WORD SENT
        RTS
2$:     PSH A                   ;SAVE A
        LDA A   CRBL+B.6821     ;CHECK FOR IRQ FLAG
        BIT A   #,300
        PUL A                   ;RESTORE A
```

```
        BNE     1$              ;BRANCH IF FLAG SET
        CLC                     ;DATA NOT SENT
        RTS
        ;
        .PAGE
        .SBTTL  B.6821 INTERRUPT SERVICE DISPATCH ROUTINES
        ;
B.AH$I: LDX     B.APRT          ;GET SERVICE ADDRESS
        BNE     B.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRAH+B.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     B.IRQD          ;TERMINATE INTERRUPTS
        ;
B.AL$I: LDX     B.APRT+4        ;GET SERVICE ADDRESS
        BNE     B.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRAL+B.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     B.IRQD          ;TERMINATE INTERRUPTS
        ;
B.BH$I: LDX     B.BPRT          ;GET SERVICE ADDRESS
        BNE     B.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRBH+B.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     B.IRQD          ;TERMINATE INTERRUPTS
        ;
B.BL$I: LDX     B.BPRT+4        ;GET SERVICE ADDRESS
        BNE     B.IRQG          ;BRANCH IF DEFINED
        LDX     #,CRBL+B.6821   ;INTERRUPT CONTROL ADDRESS
        BRA     B.IRQD          ;TERMINATE INTERRUPTS
        ;
B.IRQG: JSR     0,X             ;DO SERVICE ROUTINE
        RTI
        ;
B.IRQD: LDA A   0,X             ;GET CONTROL REGISTER
        AND B   #,376           ;CLEAR CX1 INTERRUPT ENABLE
        BIT B   #,40            ;CX2 INTERRUPT MODE ?
        BNE     1$              ;BRANCH IF NOT
        AND B   #,367           ;CLEAR CX2 INTERRUPT ENABLE
1$:     STA B   0,X             ;SET NEW CONTROL
        RTI
        ;
        .IIF    NDF,$B.6821     .END
```

```
        .PAGE
        .SBTTL  THE END
        ;
        .END
```

PARALLLEL 'PORT' MACRO V05.06 Thursday 18-Oct-07 19:15

Table of contents

```
      1                                 .TITLE  PARALLLEL 'PORT'
      2                                 .SBTTL  PARALLEL PORT AS SEEN ON THE ARB-11 BUS
      3                                 ;
      4                                 ;[X XXX XXX XXX XXN NNN]  00    INPUT BUFFER STATUS REGIS
TER
      5                                 ; <15:08> H                           <07:00> L
      6                                 ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
      7                                 ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
      8                                 ;   0   0   0   0   0   0   0   0      R   I   0   0   0
  0   0   0
      9                                 ;
     10                                 ;       R -     <07:00> DATA READY (R)
     11                                 ;                       CLEARED BY READ OF
     12                                 ;                       DATA <07:00> OR <15:00> OF (02)
     13                                 ;       I -     INTERRUPT ENABLE BIT (R/W)
     14                                 ;
     15                                 ;
     16                                 ;[X XXX XXX XXX XXN NNN]  02    INPUT BUFFER
     17                                 ; <15:08> H                           <07:00> L
     18                                 ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
     19                                 ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
     20                                 ;
     21                                 ; [15-00]       16-BIT INPUT BUFFER
     22                                 ; [15-08]       CHECK SUM (FUNCTION=7)
     23                                 ;
     24                                 ;
     25                                 ;
     26                                 ;[X XXX XXX XXX XXN NNN]  04    OUTPUT BUFFER STATUS REGI
STER
     27                                 ; <15:08> H                           <07:00> L
     28                                 ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
     29                                 ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
     30                                 ;   0   0   0   0   0   0   0   0      R   I   0   0   0
  0   0   0
```

```
    32                                  ;       R -     <07:00> DATA REQUEST (R)
    33                                  ;                       CLEARED BY WRITE OF
    34                                  ;                       DATA <07:00> OR <15:00> OF (06)
    35                                  ;       I -     INTERRUPT ENABLE BIT (R/W)
    36                                  ;
    37                                  ;
    38                                  ;[X XXX XXX XXX XXN NNN]  06    OUTPUT BUFFER
    39                                  ; <15:08> H                          <07:00> L
    40                                  ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
    41                                  ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
    42                                  ;
    43                                  ;
    44                                  ; [15-00]       16-BIT OUTPUT BUFFER
    45                                  ; [15-08]       8-BIT FUNCTION CODE (ONLY IF UPPER BYTE I
NTERRUPT
    46                                  ;                                       IS ENABLED DURING
 WRTE)
    47                                  ; [07-00]       8-BIT BINARY LOADER BUFFER (FUNCTION=7)
    48                                  ;
    49                                  ;
```

PARALLEL PORT AS SEEN ON THE ARB-11 BUS

```
     1                                  ;
     2                                  ;[X XXX XXX XXX XXN NNN]  10    FUNCTION STATUS REGISTER
     3                                  ; <15:08> H                          <07:00> L
     4                                  ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
     5                                  ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
     6                                  ;   0   0   0   0   0   0   0   0      R   I   0   0   0
  0   0   0
     7                                  ;
     8                                  ;       R -     <07:00> FUNCTION DATA READY (R)
     9                                  ;                       CLEARED BY READ OF
    10                                  ;                       DATA <15:08> OR <15:00> OF (02)
    11                                  ;       I -     INTERRUPT ENABLE BIT (R/W)
    12                                  ;
    13                                  ;
    14                                  ;[X XXX XXX XXX XXN NNN]  12    A-PORT PERIPHERAL CONTROL
 REGISTER
    15                                  ; <15:08> H                          <07:00> L
    16                                  ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
    17                                  ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
    18                                  ;[ H7  H6  H5  H4  H3  H2  H1  H0 ] [ L7  L6  L5  L4  L3
 L2  L1  L0 ]
    19                                  ;
    20                                  ;       <H7-H0> -       <15:08> PIA CONTROL FOR (02)
    21                                  ;       <L7-L0> -       <07:00> PIA CONTROL FOR (02)
    22                                  ;
    23                                  ;
    24                                  ;[X XXX XXX XXX XXN NNN]  14    FUNCTION STATUS
    25                                  ; <15:08> H                          <07:00> L
    26                                  ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
 02  01  00 ]
    27                                  ;  __  __  __  __  __  __  __  __     __  __  __  __  __
 __  __  __
    28                                  ;   0   0   0   0   0   0   0   0      R   I   0   0   0
  0   0   0
    29                                  ;
    30                                  ;       R -     FUNCTION CODE READY
```

```
    32                                  ;                       DATA <15:08> OR <15:00> OF (06)
    33                                  ;       I -     INTERRUPT ENABLE BIT (R/W)
    34                                  ;
    35                                  ;
    36                                  ;[X XXX XXX XXX XXN NNN]  16    B-PORT PERIPHERAL CONTROL
REGISTER
    37                                  ; <15:08> H                           <07:00> L
    38                                  ;[ 15  14  13  12  11  10  09  08 ] [ 07  06  05  04  03
02  01  00 ]
    39                                  ;  __  __  __  __  __  __  __  __     __  __  __  __  __
__  __  __
    40                                  ;[ H7  H6  H5  H4  H3  H2  H1  H0 ] [ L7  L6  L5  L4  L3
L2  L1  L0 ]
    41                                  ;
    42                                  ;       <H7-H0> -       <15:08> PIA CONTROL FOR (06)
    43                                  ;       <L7-L0> -       <07:00> PIA CONTROL FOR (06)
    44                                  ;
    45                                  ;
```

SOFTWARE DEFINITIONS

```
      1                                     .SBTTL  SOFTWARE DEFINITIONS
      2                                     ;
      3                                     ;         THE PARALLEL PORT SOFTWARE REQUIRES
      4                                     ;       THE 16-BIT PIA SOFTWARE PACKAGE '6821A.MAC',
      5                                     ;       THE 16-BIT PIA SOFTWARE PACKAGE '6821B.MAC', AND
      6                                     ;       THE ABSOLUTE BINARY LOADER 'LOADER.MAC'.
      7                                     ;
      8                                     ;
      9         000001                      $A.6821 =1
     10         000200                      A.6821  =200    ;16-BIT BUS PORT ADDRESS
     11                                     ;
     12         000001                      $B.6821 =1
     13         000210                      B.6821  =210    ;16-BIT I/O PORT ADDRESS
     14                                     ;
     15         000001                      $LOADER =1      ;USE ABSOLUTE LOADER
     16                                     ;
     17         000001                      $FAST   =1      ;DIRECT VERSION
     18                                     ;
     19         174000                      PGMSAV  =174000 ;ROM ADDRESS
     20         000000                      VARSAV  =0      ;DIRECT VARIABLE SPACE TO 177
     21                                     ;
     22         000177                      STACK   =177    ;STACK AREA (64-BYTES)
     23                                     ;
     24                                     ;       INTERRUPT VECTORS
     25                                     ;
     26         177740                      .=177740
     27 177740                              FDB     SPRIUS  ;OOPS !
        177740     370     006
     28 177742                              FDB     SPRIUS  ;OOPS !
        177742     370     006
     29 177744                              FDB     SPRIUS  ;OOPS !
        177744     370     006
     30 177746                              FDB     SPRIUS  ;OOPS !
        177746     370     006
     31 177750                      IRQ0:   FDB     B.BL$I  ;I/O B-PORT <7:0> INTERRUPT
```

```
    32 177752                          IRQ1:   FDB     B.BH$I  ;I/O B-PORT <15:8> INTERRUPT
       177752     374     275
    33 177754                          IRQ2:   FDB     A.AL$I  ;BUS A-PORT <7:0> INTERRUPT
       177754     373     064
    34 177756                          IRQ3:   FDB     A.AH$I  ;BUS A-PORT <15:8> INTERRUPT
       177756     373     053
    35 177760                          IRQ4:   FDB     A.BL$I  ;BUS B-PORT <7:0> INTERRUPT
       177760     373     106
    36 177762                          IRQ5:   FDB     A.BH$I  ;BUS B-PORT <15:8> INTERRUPT
       177762     373     075
    37 177764                          IRQ6:   FDB     B.AL$I  ;I/O A-PORT <7:0> INTERRUPT
       177764     374     264
    38 177766                          IRQ7:   FDB     B.AH$I  ;I/O A-PORT <15:8> INTERRUPT
       177766     374     253
    39 177770                          IRQN:   FDB     SPRIUS  ;NO HARDWARE
       177770     370     006
    40 177772                          SWINT:  FDB     SWIRQ   ;SWI INTERRUPT
       177772     370     007
    41 177774                          NMINT:  FDB     NMIRQ   ;ATTACHED PROCESSOR INIT INTERRUPT
       177774     370     000
    42 177776                          RESINT: FDB     RESTRT  ;POWER UP RESTART VECTOR
```

PARALLLEL 'PORT' MACRO V05.06 Thursday 18-Oct-07 19:15 Page 4-1

SOFTWARE DEFINITIONS

```
        177776     370     016

     43                                        ;
```

```
                                        .SBTTL  STARTUP AND BACKGROUND PROGRAM
                                        ;
         000000                         .=VARSAV
000000     000     000          NMIRQV: .BYTE   0,0     ;INIT INTERRUPT VECTOR
000002     000     000          SWIRQV: .BYTE   0,0     ;SWI INTERRUPT VECTOR
000004     000                  P.FNCT: .BYTE   0       ;FUNCTION CODE
         000005                         VARSAV=.
         174000                         .=PGMSAV
                                        ;
174000                          NMIRQ:  LDX     NMIRQV          ;GET VECTOR
   174000     336     000
174002                                  BEQ     SPRIUS          ;BRANCH IF NOT DEFINED
   174002     047     002
174004                                  JSR     0,X             ;ELSE DO IT
   174004     255     000
174006                          SPRIUS: RTI                     ;HOW DID WE GET HERE ?
   174006     073
                                        ;
174007                          SWIRQ:  LDX     SWIRQV          ;GET SWI VECTOR
   174007     336     002
174011                                  BEQ     1$              ;BRANCH IF NOT DEFINED
   174011     047     002
174013                                  JMP     0,X             ;SPECIAL SWI CALL TO ACCESS
   174013     156     000
174015                          1$:     RTI                     ;INTERRUPT DRIVEN ROUTINES
   174015     073
                                        ;
174016                          RESTRT: LDS     #,STACK         ;SET UP STACK POINTER
   174016     216     000     177
174021                                  LDX     #,0
   174021     316     000     000
174024                                  STX     SWIRQV          ;SET SWI FOR NOTHING
   174024     337     002
174026                                  STX     NMIRQV          ;SET NMI FOR NOTHING
   174026     337     000
```

```
   174030     215     047
25                                 ;
26                                 ;       BACKGROUND PROGRAM
27                                 ;
28 174032                     1$:     LDS     #,STACK         ;SET THE STACK POINTER
   174032     216     000     177
29 174035                             CLR     177740          ;ENABLE ALL INTERRUPTS
   174035     177     377     340
30 174040                             CLI                     ;(VIA 6828 CONTROLLER)
   174040     016
31 174041                             LDX     #,0             ;LOOP COUNTER
   174041     316     000     000
32 174044                     2$:     INX                     ;LOOP HERE 64K TIMES
   174044     010
33 174045                             BNE     2$
   174045     046     375
34 174047                             JSR     A.ALUP          ;RE-ENABLE <7:0> INPUT
   174047     275     372     014
35 174052                             JSR     A.AHUP          ;RE-ENABLE <15:8> INPUT
   174052     275     372     007
36 174055                             JSR     A.BLUP          ;RE-ENABLE <7:0> OUTPUT
```

PARALLLEL 'PORT' MACRO V05.06 Thursday 18-Oct-07 19:15 Page 5-1

STARTUP AND BACKGROUND PROGRAM

```
        174055     275     372     110
     37 174060                              JSR     A.BHDN          ;DISABLE <15:8> OUTPUT
        174060     275     372     115
     38 174063                              JSR     B.ALUP          ;RE-ENABLE <7:0> INPUT
        174063     275     373     214
     39 174066                              JSR     B.AHUP          ;RE-ENABLE <15:8> INPUT
        174066     275     373     207
     40 174071                              JSR     B.BLUP          ;RE-ENABLE <7:0> OUTPUT
        174071     275     373     310
     41 174074                              JSR     B.BHDN          ;DISABLE <15:8> OUTPUT
        174074     275     373     315
     42 174077                              BRA     1$
        174077     040     331
     43                                     ;
```

PORT INITIALIZATION

```
      1                                     .SBTTL  PORT INITIALIZATION
      2                                     ;
      3 174101                        PINITS: JSR     LOAD$I          ;INITIALIZE ABSOLUTE LOADER
        174101     275     371     142
      4                                     ;
      5 174104                              CLR     P.FNCT          ;INSURE NORMAL PORT
        174104     177     000     004
      6                                     ;
      7                                     ;       SET UP BUS-PORT PARALLEL OUTPUT
      8                                     ;
      9 174107                              LDX     #,P.BSBL        ;SERVICE VECTOR
        174107     316     371     103
     10 174112                              LDA B   #,45            ;CB2(ON E, LOW), CB1(-, HIGH)
        174112     306     045
     11 174114                              LDA A   #,74
        174114     206     074
     12 174116                              JSR     A.BLDF          ;SAVE CONTROL
        174116     275     372     074
     13 174121                              LDX     #,0             ;NO SERVICE
        174121     316     000     000
     14 174124                              LDA B   #,44            ;CB2(ON E, LOW), CB1(-, HIGH)
        174124     306     044
     15 174126                              LDA A   #,74
        174126     206     074
     16 174130                              JSR     A.BHDF          ;SAVE CONTROL
        174130     275     372     065
     17 174133                              LDA B   #,377           ;OUTPUT
        174133     306     377
     18 174135                              JSR     A.BLDR
        174135     275     372     050
     19 174140                              LDA B   #,377           ;OUTPUT
        174140     306     377
     20 174142                              JSR     A.BHDR
        174142     275     372     033
     21                                     ;
```

```
23                                      ;
24 174145                               LDX     #,P.BSAL        ;SERVICE VECTOR
   174145     316     370     374
25 174150                               LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
   174150     306     045
26 174152                               LDA A   #,74
   174152     206     074
27 174154                               JSR     A.ALDF          ;SAVE CONTROL
   174154     275     372     000
28 174157                               LDX     #,P.BSAH        ;SERVICE VECTOR
   174157     316     371     120
29 174162                               LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
   174162     306     045
30 174164                               LDA A   #,74
   174164     206     074
31 174166                               JSR     A.AHDF          ;SAVE CONTROL
   174166     275     371     371
32 174171                               LDA B   #,0             ;INPUT
   174171     306     000
33 174173                               JSR     A.ALDR
   174173     275     371     354
```

```
PARALLLEL 'PORT'       MACRO V05.06  Thursday 18-Oct-07 19:15  Page 6-1

PORT INITIALIZATION


     34 174176                              LDA B   #,0             ;INPUT
        174176     306     000
     35 174200                              JSR     A.AHDR
        174200     275     371     337
     36                                     ;
     37                                     ;       SET UP I/O-PORT PARALLEL OUTPUT
     38                                     ;
     39 174203                              LDX     #,P.IOBL        ;SERVICE VECTOR
        174203     316     371     051
     40 174206                              LDA B   #,45            ;CB2(ON E, LOW), CB1(-, HIGH)
        174206     306     045
     41 174210                              LDA A   #,74
        174210     206     074
     42 174212                              JSR     B.BLDF          ;SAVE CONTROL
        174212     275     373     274
     43 174215                              LDX     #,0             ;NO SERVICE
        174215     316     000     000
     44 174220                              LDA B   #,44            ;CB2(ON E, LOW), CB1(-, HIGH)
        174220     306     044
     45 174222                              LDA A   #,74
        174222     206     074
     46 174224                              JSR     B.BHDF          ;SAVE CONTROL
        174224     275     373     265
     47 174227                              LDA B   #,377           ;OUTPUT
        174227     306     377
     48 174231                              JSR     B.BLDR
        174231     275     373     250
     49 174234                              LDA B   #,377           ;OUTPUT
        174234     306     377
     50 174236                              JSR     B.BHDR
        174236     275     373     233
     51                                     ;
     52                                     ;       SET UP I/O-PORT PARALLEL INPUT
     53                                     ;
     54 174241                              LDX     #,P.IOAL        ;SERVICE VECTOR
```

```
 55 174244                              LDA B   #,45            ;CA2(ON E, LOW), CA1(-, HIGH)
    174244     306     045
 56 174246                              LDA A   #,74
    174246     206     074
 57 174250                              JSR     B.ALDF          ;SAVE CONTROL
    174250     275     373     200
 58 174253                              LDX     #,0             ;NO SERVICE
    174253     316     000     000
 59 174256                              LDA B   #,44            ;CA2(ON E, LOW), CA1(-, HIGH)
    174256     306     044
 60 174260                              LDA A   #,74
    174260     206     074
 61 174262                              JSR     B.AHDF          ;SAVE CONTROL
    174262     275     373     171
 62 174265                              LDA B   #,0             ;INPUT
    174265     306     000
 63 174267                              JSR     B.ALDR
    174267     275     373     154
 64 174272                              LDA B   #,0             ;INPUT
    174272     306     000
 65 174274                              JSR     B.AHDR
```

PORT INITIALIZATION

```
   174274     275     373     137
66                                 ;
67                                 ;       ENABLE/DISABLE INTERRUPTS
68                                 ;
69                                 ;BUS-PORT
70 174277                          JSR     A.ALDN          ;CLEAR, CA2='1'
   174277     275     372     026
71 174302                          JSR     A.ALUP          ;ENABLE
   174302     275     372     014
72 174305                          JSR     A.ALRD+2        ;CLEAR IRQA, CA2='0'
   174305     275     372     147
73                                 ;
74 174310                          JSR     A.AHDN          ;CLEAR, CA2='1'
   174310     275     372     021
75 174313                          JSR     A.AHUP          ;ENABLE
   174313     275     372     007
76 174316                          JSR     A.AHRD+2        ;CLEAR IRQ, CA2='0'
   174316     275     372     131
77                                 ;
78 174321                          JSR     A.BLDN          ;CLEAR, CB2='1'
   174321     275     372     122
79 174324                          JSR     A.BLRD+2        ;CLEAR IRQ, CB2='1'
   174324     275     372     317
80 174327                          JSR     A.BLUP          ;ENABLE
   174327     275     372     110
81                                 ;
82 174332                          JSR     A.BHDN          ;CLEAR, CB2='1'
   174332     275     372     115
83 174335                          JSR     A.BHRD+2        ;CLEAR IRQ, CB2='1'
   174335     275     372     277
84                                 ;
85                                 ;I/O-PORT
86 174340                          JSR     B.AHDN          ;CLEAR, CA2='1'
   174340     275     373     221
87 174343                          JSR     B.ALDN
```

```
    88 174346                          JSR     B.AHUP          ;ENABLE
       174346     275     373     207
    89 174351                          JSR     B.ALUP
       174351     275     373     214
    90 174354                          JSR     B.AWRD+2        ;CLEAR IRQ, CA2'S='0'
       174354     275     373     365
    91                                 ;
    92 174357                          JSR     B.BHDN          ;DISABLE
       174357     275     373     315
    93 174362                          JSR     B.BLDN
       174362     275     373     322
    94 174365                          JSR     B.BWRD+2        ;CLEAR IRQ, CB2'S='1'
       174365     275     374     137
    95 174370                          JSR     B.BLUP          ;ENABLE
       174370     275     373     310
    96                                 ;
    97 174373                          RTS
       174373     071
    98                                 ;
    99                                 .IF     NDF,$FAST
   100                                 ;       PORT SERVICE ROUTINES
```

PORT INITIALIZATION

```
    101                                 ;
    102                                 ; ARB-11 TRANSFER TO I/O-PORT OUTPUT
    103                                 ;  LOW  BYTE 'READY' WHEN PORT REQUESTS DATA
    104                                 ;  I/O-PORT LOW BYTE CA2='0' WHEN DATA IS AT THE PORT
    105                                 ;
    106                         P.BSAL: LDA B   P.FNCT          ;CHECK  FUNCTION
    107                                 CMP B   #,7             ;LOADER ?
    108                                 BNE     1$              ;BRANCH IF NOT
    109                                 JSR     A.ALRD          ;GET BYTE
    110                                 JSR     LOADER          ;GO PROCESS BYTE
    111                                 JSR     A.BHWT+2        ;WRITE CHECK SUM
    112                                 RTS
    113                         1$:     JSR     A.ALDN          ;HOLD CA2='1'
    114                                 JSR     B.BLDN          ;RESTROBE CONTROL
    115                                 JSR     B.BLUP
    116                                 JSR     A.AWRD+2        ;READ WORD DATA
    117                                 JSR     B.BWWT+2        ;TRANSFER READY OR NOT !
    118                                 JSR     A.ALUP          ;LEAVE CA2='1'
    119                                 RTS
    120                                 ;
    121                         P.IOBL: JSR     A.ALRD+2        ;CA2='0', PORT READY
    122                                 JSR     B.BLDN          ;DISABLE CONTROL
    123                                 JSR     B.BLRD+2        ;CLEAR INTERRUPT
    124                                 JSR     B.BLUP          ;RE-ENABLE
    125                                 RTS
    126                                 ;
    127                                 ;
    128                                 ; I/O-PORT INPUT TRANSFER TO ARB-11
    129                                 ;  LOW  BYTE 'READY' WHEN DATA IS AVAILABLE
    130                                 ;  I/O-PORT LOW  BYTE CA2='0' WHEN I/O-PORT DATA HAS BEEN
 READ
    131                                 ;  I/O-PORT HIGH BYTE CA2='0' WHEN DATA IS TAKEN BY ARB-1
1
    132                                 ;
    133                         P.IOAL: JSR     B.AHDN          ;HOLD CA2='1'
    134                                 JSR     B.AWRD+2        ;READ I/O-PORT DATA
```

```
   136                                  JSR     B.AHUP          ;LEAVE CA2='1'
   137                                  RTS
   138                                  ;
   139                          P.BSBL: JSR     B.AHDN          ;CA2='1'
   140                                  JSR     B.AHUP
   141                                  JSR     B.AHRD+2        ;CA2='0', DATA TAKEN
   142                                  JSR     A.BLDN          ;DISABLE CONTROL
   143                                  JSR     A.BLRD+2        ;CLEAR INTERRUPT
   144                                  JSR     A.BLUP          ;RE-ENABLE
   145                                  RTS
   146                                  ;
   147                                  .ENDC
   148                                  .IF     DF,$FAST
   149                                  ;       PORT SERVICE ROUTINES
   150                                  ;
   151                                  ; ARB-11 TRANSFER TO I/O-PORT OUTPUT
   152                                  ;  LOW  BYTE 'READY' WHEN PORT REQUESTS DATA
   153                                  ;  I/O-PORT LOW BYTE CA2='0' WHEN DATA IS AT THE PORT
   154                                  ;
   155 174374                   P.BSAL: LDA B   P.FNCT          ;CHECK  FUNCTION
       174374     326     004
   156 174376                           CMP B   #,7             ;LOADER ?
```

PORT INITIALIZATION

```
        174376     301     007
    157 174400                             BNE     1$              ;BRANCH IF NOT
        174400     046     012
    158 174402                             JSR     A.ALRD          ;GET BYTE
        174402     275     372     145
    159 174405                             JSR     LOADER          ;GO PROCESS BYTE
        174405     275     371     150
    160 174410                             JSR     A.BHWT+2        ;WRITE CHECK SUM
        174410     275     372     363
    161 174413                             RTS
        174413     071
    162 174414                         1$: LDA B   A.APRT+7        ;HOLD CA2='1'
        174414     326     023
    163 174416                             STA B   CRAL+A.6821
        174416     327     201
    164 174420                             LDA B   B.BPRT+7        ;RESTROBE CONTROL
        174420     326     053
    165 174422                             STA B   CRBL+B.6821
        174422     327     213
    166 174424                             LDA B   B.BPRT+6
        174424     326     052
    167 174426                             STA B   CRBL+B.6821
        174426     327     213
    168 174430                             LDA B   PRBH+B.6821     ;CLEAR FLAGS
        174430     326     216
    169 174432                             LDA B   PRAH+A.6821     ;READ AND TRANSFER WORD
        174432     326     204
    170 174434                             STA B   PRBH+B.6821
        174434     327     216
    171 174436                             LDA B   PRBL+B.6821
        174436     326     217
    172 174440                             LDA B   PRAL+A.6821
        174440     326     205
    173 174442                             STA B   PRBL+B.6821
        174442     327     217
```

```
        174444     326     022
    175 174446                             STA B   CRAL+A.6821
        174446     327     201
    176 174450                             RTS
        174450     071
    177                                    ;
    178 174451                         P.IOBL: LDA B   PRAL+A.6821     ;CA2='0', PORT READY
        174451     326     205
    179 174453                             LDA B   PRBL+B.6821     ;CLEAR INTERRUPT
        174453     326     217
    180 174455                             RTS
        174455     071
    181                                    ;
    182                                    ;
    183                                    ; I/O-PORT INPUT TRANSFER TO ARB-11
    184                                    ;  LOW  BYTE 'READY' WHEN DATA IS AVAILABLE
    185                                    ;  I/O-PORT LOW  BYTE CA2='0' WHEN I/O-PORT DATA HAS BEEN
 READ
    186                                    ;  I/O-PORT HIGH BYTE CA2='0' WHEN DATA IS TAKEN BY ARB-1
1
    187                                    ;
    188 174456                         P.IOAL: LDA B   B.APRT+3        ;HOLD CA2='1'
        174456     326     037
```

PORT INITIALIZATION

```
   189 174460                              STA B   CRAH+B.6821
       174460     327     210
   190 174462                              LDA B   PRBH+A.6821     ;CLEAR FLAGS
       174462     326     206
   191 174464                              LDA B   PRAH+B.6821     ;AND TRANSFER WORD
       174464     326     214
   192 174466                              STA B   PRBH+A.6821
       174466     327     206
   193 174470                              LDA B   PRBL+A.6821
       174470     326     207
   194 174472                              LDA B   PRAL+B.6821
       174472     326     215
   195 174474                              STA B   PRBL+A.6821
       174474     327     207
   196 174476                              LDA B   B.APRT+2        ;LEAVE CA2='1'
       174476     326     036
   197 174500                              STA B   CRAH+B.6821
       174500     327     210
   198 174502                              RTS
       174502     071
   199                                     ;
   200 174503                      P.BSBL: LDA B   B.APRT+3        ;CA2='1'
       174503     326     037
   201 174505                              STA B   CRAH+B.6821
       174505     327     210
   202 174507                              LDA B   B.APRT+2
       174507     326     036
   203 174511                              STA B   CRAH+B.6821
       174511     327     210
   204 174513                              LDA B   PRAH+B.6821     ;CA2='0', DATA TAKEN
       174513     326     214
   205 174515                              LDA B   PRBL+A.6821     ;CLEAR INTERRUPT
       174515     326     207
   206 174517                              RTS
       174517     071
```

```
  208                                   .ENDC
  209                                   ;
  210 174520                    P.BSAH: JSR     A.AHRD          ;READ CONTROL CODE
      174520     275     372     127
  211 174523                            STA B   P.FNCT          ;SAVE FUNCTION
      174523     327     004
  212 174525                            CMP B   #,7             ;LOADER ?
      174525     301     007
  213 174527                            BNE     1$              ;BRANCH IF NOT
      174527     046     010
  214 174531                            JSR     LOAD$I          ;ELSE INITIALIZE LOADER
      174531     275     371     142
  215 174534                            LDA B   #,0             ;ZERO CHECK SUM
      174534     306     000
  216 174536                            JSR     A.BHWT+2        ;WRITE CHECK SUM
      174536     275     372     363
  217 174541                         1$:     RTS
      174541     071
  218                                   ;
```

ABSOLUTE BINARY LOADER

```
      2                                     .SBTTL  ABSOLUTE BINARY LOADER
      3                                     ;
      4                                     ;         THIS LOADER ROUTINE IS AN ADAPTATION OF THE
      5                                     ;       'DEC' PAPER TAPE ABSOLUTE BINARY LOADER.
      6                                     ;         THE LOADER PROVIDES A MEANS OF LOADING OBJECT
      7                                     ;       CODE INTO ANY REGION OF MEMORY AND STARTING
      8                                     ;       PROGRAM EXECUTION.
      9                                     ;
     10                                     ;         AN ODD JUMP ADDRESS TERMINATES LOADER
     11                                     ;         AN EVEN JUMP ADDRESS TRANSFERS CONTROL TO THE J
UMP ADDRESS
     12                                     ;
     13                                     .IF     NDF,$LOADER
     14                                     VARSAV  =0      ;VARIABLE SPACE
     15                                     PGMSAV  =100000 ;PROGRAM SPACE
     16                                     .=PGMSAV
     17                                     .ENDC
     18                                     ;
     19         174542                      PGMSAV=.
     20         000005                      .=VARSAV
     21 000005     000     000          LOAD$V: .BYTE   0,0     ;SERVICE ADDRESS
     22 000007     000     000          L.BYTC: .BYTE   0,0     ;BYTE COUNTER
     23 000011     000     000          L.ADDR: .BYTE   0,0     ;LOAD ADDRESS
     24 000013     000                  L.CKSM: .BYTE   0       ;CHECK SUM
     25         000014                      VARSAV=.
     26         174542                      .=PGMSAV
     27                                     ;
     28                                     ;       INITIALIZE ENTRY POINT
     29                                     ;
     30 174542                           LOAD$I: LDX     #,L.LDR0        ;FIRST ENTRY
        174542     316     371     200
     31 174545                               STX     LOAD$V
        174545     337     005
     32 174547                               RTS
        174547     071
     33                                     ;
```

LOADER ROUTINE

```
      1                                     .SBTTL  LOADER ROUTINE
      2                                     ;
      3                                     ;       ENTER WITH DATA IN <B>
      4                                     ;       EXITS WITH CURRENT CHECK SUM IN <B>
      5                                     ;
      6 174550                      LOADER: LDA A   L.CKSM          ;UPDATE CHECK SUM
        174550     226     013
      7 174552                              ABA
        174552     033
      8 174553                              STA A   L.CKSM
        174553     227     013
      9 174555                              LDX     L.BYTC          ;UPDATE BYTE COUNT
        174555     336     007
     10 174557                              DEX
        174557     011
     11 174560                              STX     L.BYTC
        174560     337     007
     12 174562                              LDX     LOAD$V          ;DISPATCH ADDRESS
        174562     336     005
     13 174564                              JMP     0,X             ;GO TO IT
        174564     156     000
     14 174566                      L.GBYT: TSX                     ;GET RETURN ADDRESS ON STACK
        174566     060
     15 174567                              LDX     0,X
        174567     356     000
     16 174571                              STX     LOAD$V          ;SAVE RETURN ADDRESS
        174571     337     005
     17 174573                              INS                     ;POP RETURN FROM STACK
        174573     061
     18 174574                              INS
        174574     061
     19 174575                              LDA B   L.CKSM          ;RETURN CHECK SUM IN <B>
        174575     326     013
     20 174577                              RTS                     ;BACK TO CALLER
        174577     071
```

```
 22 174600                          L.LDR0: CLR     L.CKSM          ;CLEAR CHECK SUM
    174600     177     000     013
 23 174603                                  BRA     L.LDR2
    174603     040     005
 24 174605                          L.LDR1: CLR     L.CKSM          ;CLEAR CHECK SUM
    174605     177     000     013
 25 174610                                  BSR     L.GBYT          ;BACK FOR A BYTE
    174610     215     354
 26 174612                          L.LDR2: DEC B                   ;A '1' ?
    174612     132
 27 174613                                  BNE     L.LDR1          ;LOOP UNTIL '1' FOUND
    174613     046     370
 28 174615                                  BSR     L.GBYT          ;BACK FOR A BYTE
    174615     215     347
 29 174617                                  TST B                   ;= '0' ?
    174617     135
 30 174620                                  BNE     L.LDR0          ;LOOP BACK UNTIL '1','0' SEQUENCE
    174620     046     356
 31 174622                                  BSR     L.GBYT          ;BACK FOR A BYTE
    174622     215     342
 32 174624                                  STA B   L.ADDR+1        ;NOW SAVE BYTE COUNT
```

LOADER ROUTINE

```
        174624     327     012
     33 174626                          BSR     L.GBYT          ;BACK FOR A BYTE
        174626     215     336
     34 174630                          LDA A   L.ADDR+1        ;RETRIEVE LOW ORDER COUNT
        174630     226     012
     35 174632                          SUB A   #,4             ;CORRECT BYTE COUNT
        174632     200     004
     36 174634                          SBC B   #,0
        174634     302     000
     37 174636                          STA A   L.BYTC+1        ;AND SAVE IT
        174636     227     010
     38 174640                          STA B   L.BYTC
        174640     327     007
     39 174642                          LDX     L.BYTC          ;L.BYTC=2 ?
        174642     336     007
     40 174644                          CPX     #,2
        174644     214     000     002
     41 174647                          BNE     L.LDR4          ;BRANCH IF NOT
        174647     046     033
     42 174651                L.LDR3:   BSR     L.GBYT          ;BACK FOR A BYTE
        174651     215     313
     43 174653                          STA B   L.ADDR+1        ;GET JUMP ADDRESS
        174653     327     012
     44 174655                          BSR     L.GBYT          ;BACK FOR A BYTE
        174655     215     307
     45 174657                          STA B   L.ADDR
        174657     327     011
     46 174661                          BSR     L.GBYT          ;BACK FOR CHECK SUM BYTE !
        174661     215     303
     47 174663                          TST     L.CKSM          ;CHECK FOR ERROR
        174663     175     000     013
     48 174666                          BNE     L.LDR6          ;SKIP ON ERROR
        174666     046     043
     49 174670                          LDA A   L.ADDR+1        ;CHECK FOR ODD ADDRESS
        174670     226     012
```

```
   174672     205     001
51 174674                          BNE     L.LDR6          ;TERMINATE BY RESTARTING SCAN
   174674     046     035
52 174676                          LDX     L.ADDR          ;ELSE GO TO ADDRESS
   174676     336     011
53 174700                          JSR     0,X
   174700     255     000
54 174702                          BRA     L.LDR6          ;ON QUICK RETURN - CONTINUE
   174702     040     027
55 174704                  L.LDR4: BSR     L.GBYT          ;BACK FOR A BYTE
   174704     215     260
56 174706                          STA B   L.ADDR+1        ;GET LOAD ADDRESS
   174706     327     012
57 174710                          BSR     L.GBYT          ;BACK FOR A BYTE
   174710     215     254
58 174712                          STA B   L.ADDR
   174712     327     011
59 174714                  L.LDR5: BSR     L.GBYT          ;BACK FOR A BYTE
   174714     215     250
60 174716                          LDX     L.BYTC          ;CHECK BYTE COUNT
   174716     336     007
```

LOADER ROUTINE

```
     61 174720                                BMI     L.LDR6          ;BRANCH IF DONE
        174720     053     011
     62 174722                                LDX     L.ADDR          ;GET LOAD ADDRESS
        174722     336     011
     63 174724                                STA B   0,X             ;STORE DATA
        174724     347     000
     64 174726                                INX                     ;UPDATE LOAD ADDRESS
        174726     010
     65 174727                                STX     L.ADDR
        174727     337     011
     66 174731                                BRA     L.LDR5
        174731     040     361
     67 174733                      L.LDR6: BSR     L.GBYT          ;CURRENT BYTE WAS CHECK SUM RESUL
T
        174733     215     231
     68 174735                                BRA     L.LDR0          ;START NEW SEQUENCE
        174735     040     241
     69                                       ;
```

```
                                        .SBTTL  A.6821 PERIPHERAL INTERFACE SOFTWARE
                                        ;
                                        ;         THIS SOFTWARE PACKAGE SUPPORTS THE
                                        ;       6821 PERIPHERAL INTERFACE ADAPTER (PIA).
                                        ;         ENTRY POINTS TO THE SOFTWARE ARE DEFINED
                                        ;       FOR EXTERNAL ACCESS:
                                        ;
                                        ;         THE FOLLOWING DATA DIRECTION ROUTINES ARE
                                        ;       ENTERED WITH
                                        ;               <B> = DIRECTION CONTROL
                                        ;       BY A JSR ____
                                        ;
                                        ;       A.AHDR --> A-PORT <15:8> DIRECTION
                                        ;       A.ALDR --> A-PORT <7:0> DIRECTION
                                        ;       A.BHDR --> B-PORT <15:8> DIRECTION
                                        ;       A.BLDR --> B-PORT <7:0> DIRECTION
                                        ;
                                        ;         THE FOLLOWING CONTROL ROUTINES ARE ENTERED WITH
                                        ;               <X> = SERVICE ADDRESS
                                        ;               <B> = INTERRUPT ENABLE CONTROL
                                        ;               <A> = INTERRUPT DISABLE CONTROL
                                        ;       BY A JSR ____
                                        ;
                                        ;       A.AHDF --> A-PORT <15:8>
                                        ;       A.ALDF --> A-PORT <7:0>
                                        ;       A.BHDF --> B-PORT <15:8>
                                        ;       A.BLDF --> B-PORT <7:0>
                                        ;
                                        ;         THE FOLLOWING INTERRUPT ENABLE ROUTINES ARE
                                        ;       ENTERED BY JSR ____
                                        ;
                                        ;       A.AHUP --> A-PORT <15:8> INTERRUPT ENABLE
                                        ;       A.ALUP --> A-PORT <7:0> INTERRUPT ENABLE
                                        ;       A.BHUP --> B-PORT <15:0> INTERRUPT ENABLE
                                        ;       A.BLUP --> B-PORT <7:0> INTERRUPT ENABLE
```

```
38                                      ;         THE FOLLOWING INTERRUPT DISABLE ROUTINES ARE
39                                      ;       ENTERED BY JSR ____
40                                      ;
41                                      ;       A.AHDN --> A-PORT <15:0> INTERRUPT DISABLE
42                                      ;       A.ALDN --> A-PORT <7:0> INTERRUPT DISABLE
43                                      ;       A.BHDN --> B-PORT <15:8> INTERRUPT DISABLE
44                                      ;       A.BLDN --> B-PORT <7:0> INTERRUPT DISABLE
45                                      ;
46                                      ;         THE FOLLOWING ARE INTERRUPT ENTRY POINTS
47                                      ;       FOR THE DUAL PIA PORTS
48                                      ;
49                                      ;       A.AH$I --> A-PORT <15:8> IRQ
50                                      ;
51                                      ;       A.AL$I --> A-PORT <7:0> IRQ
52                                      ;
53                                      ;       A.BH$I --> B-PORT <15:8> IRQ
54                                      ;
55                                      ;       A.BL$L --> B-PORT <7:0> IRQ
56                                      ;
```

A.6821 PERIPHERAL INTERFACE SOFTWARE

```
                                        ;
                                        ;         THE FOLLOWING ENTRY POINTS ARE USED TO TRANSFER
                                        ;       DATA TO THE VARIOUS PIA I/O REGISTERS. THE MAIN
                                        ;       ENTRY POINT VERIFIES THAT AT LEAST ONE OF THE
                                        ;       PORT IRQ FLAGS IS SET BEFORE TRANSFERRING DATA.
                                        ;       IF DATA IS TRANSFERED THEN THE 'C' BIT = 1,
                                        ;       IF THE PORT WAS NOT READY THEN NO TRANSFER
                                        ;       IS PERFORMED AND THE 'C' BIT = 0.
                                        ;         THE ROUTINES MAY BE ENTERED AT THE ENTRY POINT
+ 2
                                        ;       IF NO IRQ FLAG CHECKS ARE TO BE MADE.
                                        ;         BYTE DATA IS PASSED IN <B>.
                                        ;         WORD DATA IS PASSED IN <B,A>
                                        ;
                                        ;       CALL BY JSR ____
                                        ;
                                        ;       A.AHRD --> A-PORT <15:8> READ
                                        ;       A.ALRD --> A-PORT <7:0> READ
                                        ;       A.AWRD --> A-PORT <15:0> READ
                                        ;       A.BHRD --> B-PORT <15:8> READ
                                        ;       A.BLRD --> B-PORT <7:0> READ
                                        ;       A.BWRD --> B-PORT <15:0> READ
                                        ;
                                        ;
                                        ;       A.AHWT --> A-PORT <15:8> WRITE
                                        ;       A.ALWT --> A-PORT <7:0> WRITE
                                        ;       A.AWWT --> A-PORT <15:0> WRITE
                                        ;       A.BHWT --> B-PORT <15:8> WRITE
                                        ;       A.BLWT --> B-PORT <7:0> WRITE
                                        ;       A.BWWT --> B-PORT <15:0> WRITE
                                        ;
```

A.6821 PIA HARDWARE

```
      1                                 .SBTTL  A.6821 PIA HARDWARE
      2                                 ;
      3                                 ;         THE PIA CONFIGURATION  CONSISTS OF TWO
      4                                 ;       MC6821 PERIPHERAL INTERFACE ADAPTERS (PIA'S)
      5                                 ;       WHICH MAY BE ADDRESSED AS 16-BIT PORTS.
      6                                 ;       ONE PIA UNIT FORMS THE HIGH ORDER PART <15:8>
      7                                 ;       AND THE OTHER THE LOW ORDER PART <7:0>
      8                                 ;
      9                                 ; REGISTER 0
     10         000000                 CRAH    =0      ;A-PORT <15:8> CONTROL REGISTER
     11                                 ;
     12                                 ; REGISTER 1
     13         000001                 CRAL    =1      ;A-PORT <7:0> CONTROL REGISTER
     14                                 ;
     15                                 ; REGISTER 2
     16         000002                 CRBH    =2      ;B-PORT <15:8> CONTROL REGISTER
     17                                 ;
     18                                 ; REGISTER 3
     19         000003                 CRBL    =3      ;B-PORT <7:0> CONTROL REGISTER
     20                                 ;
     21                                 ; REGISTER 4
     22         000004                 DDRAH   =4      ;CRAH<2>=0, A-PORT <15:8> DATA DIRECTION
REGISTER
     23         000004                 PRAH    =4      ;CRAH<2>=1, A-PORT <15:8> DATA REGISTER
     24                                 ;
     25                                 ; REGISTER 5
     26         000005                 DDRAL   =5      ;CRAL<2>=0, A-PORT <7:0> DATA DIRECTION R
EGISTER
     27         000005                 PRAL    =5      ;CRAL<2>=1, A-PORT <7:0> DATA REGISTER
     28                                 ;
     29                                 ; REGISTER 6
     30         000006                 DDRBH   =6      ;CRBH<2>=0, B-PORT <15:8> DATA DIRECTION
REGISTER
     31         000006                 PRBH    =6      ;CRBH<2>=1, B-PORT <15:8> DATA REGISTER
     32                                 ;
     33                                 ; REGISTER 7
```

```
    35          000007                          PRBL    =7      ;CRBL<2>=1, B-PORT <7:0> DATA REGISTER
    36                                          ;
```

A.6821 PIA HARDWARE

```
                                        ;
                                        ;       A-PORT CONTROL REGISTERS
                                        ;
                                        ; CONTROL REGISTERS CRA_  - READ/WRITE REGISTERS
                                        ;                                CONTROLLING INTERRUPTS AN
D STROBES
                                        ;
                                        ; BIT 1 0       CONTROL OF INTERRUPT INPUT CA1
                                        ;               ACTIVE TRANSITION
                                        ;     _ _       TO SET CRA<7>           /IRQA OUTPUT
                                        ;     0 0               (-)             DISABLED
                                        ;     0 1               (-)             /CRA<7>
                                        ;     1 0               (+)             DISABLED
                                        ;     1 1               (+)             /CRA<7>
                                        ;       CRA<7> CLEARED BY A READ OF CORRESPONDING PRA_
                                        ;       /IRQA FOLLOWS STATE OF /CRA<7> WHEN ENABLED
                                        ;
                                        ; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
                                        ;     _
                                        ;     0         ACCESS DATA DIRECTION REGISTERS
                                        ;     1         ACCESS PORT DATA REGISTER
                                        ;
                                        ; BIT 5 4 3     CONTROL OF CA2 AS AN INTERRUPT INPUT
                                        ;               ACTIVE TRANSITION
                                        ;     _ _ _     TO SET CRA<6>           /IRQB OUTPUT
                                        ;     0 0 0             (-)             DISABLED
                                        ;     0 0 1             (-)             /CRA<6>
                                        ;     0 1 0             (+)             DISABLED
                                        ;     0 1 1             (+)             /CRA<6>
                                        ;       CRA<6> CLEARED BY READ OF CORRESPONDING PRA_
                                        ;       /IRQB FOLLOWS STATE OF CRA<6> WHEN ENABLED
                                        ;
                                        ; BIT 5 4 3     CONTROL OF CA2 AS AN OUTPUT
                                        ;     _ _ _     CA2 LOW                 CA2 HIGH
                                        ;     1 0 0     AFTER READ TO PRA_      WHEN CRA<7> IS SE
T
```

```
     36                                 ;     1 0 1     AFTER READ TO PRA_      ON SECOND (+) E A
FTER
     37                                 ;               ON (+) OF NEXT E        PIA DESELECTED
     38                                 ;     1 1 0     =CRA<3>
     39                                 ;     1 1 1                             =CRA<3>
     40                                 ;
     41                                 ; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     42                                 ;                       OF CA2, CLEARED BY READ OF PRA_
     43                                 ;
     44                                 ; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     45                                 ;                       OF CA1, CLEARED BY READ OF PRA_
     46                                 ;
     47                                 ;       DATA DIRECTION REGISTER (DDRA_)
     48                                 ; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
     49                                 ;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
     50                                 ;               BY 0'S AND OUTPUT BITS BY 1'S.
     51                                 ;
     52                                 ;       PERIPHERAL DATA REGISTER (PRA_)
     53                                 ; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGIST
ERS
     54                                 ;
```

A.6821 DETAILS

```
      1                                 .SBTTL  A.6821 DETAILS
      2                                 ;
      3                                 ;       B-PORT CONTROL REGISTERS
      4                                 ;
      5                                 ; CONTROL REGISTERS CRB_  - READ/WRITE REGISTERS
      6                                 ;                               CONTROLLING INTERRUPTS AN
D STROBES
      7                                 ;
      8                                 ; BIT 1 0       CONTROL OF INTERRUPT INPUT CB1
      9                                 ;               ACTIVE TRANSITION
     10                                 ;     _ _       TO SET CRB<7>           /IRQA OUTPUT
     11                                 ;     0 0               (-)             DISABLED
     12                                 ;     0 1               (-)             /CRB<7>
     13                                 ;     1 0               (+)             DISABLED
     14                                 ;     1 1               (+)             /CRB<7>
     15                                 ;       CRB<7> CLEARED BY A READ OF CORRESPONDING PRB_
     16                                 ;       /IRQA FOLLOWS STATE OF /CRB<7> WHEN ENABLED
     17                                 ;
     18                                 ; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
     19                                 ;     _
     20                                 ;     0         ACCESS DATA DIRECTION REGISTERS
     21                                 ;     1         ACCESS PORT DATA REGISTER
     22                                 ;
     23                                 ; BIT 5 4 3     CONTROL OF CB2 AS AN INTERRUPT INPUT
     24                                 ;               ACTIVE TRANSITION
     25                                 ;     _ _ _     TO SET CRB<6>           /IRQB OUTPUT
     26                                 ;     0 0 0             (-)             DISABLED
     27                                 ;     0 0 1             (-)             /CRB<6>
     28                                 ;     0 1 0             (+)             DISABLED
     29                                 ;     0 1 1             (+)             /CRB<6>
     30                                 ;       CRB<6> CLEARED BY READ OF CORRESPONDING PRB_
     31                                 ;       /IRQB FOLLOWS STATE OF CRB<6> WHEN ENABLED
     32                                 ;
     33                                 ; BIT 5 4 3     CONTROL OF CB2 AS AN OUTPUT
     34                                 ;     _ _ _     CB2 LOW                 CB2 HIGH
```

```
     36                                 ;              ON (+) OF NEXT E        BY CB1 TRANSITION
     37                                 ;     1 0 1    AFTER WRITE TO PRB      ON SECOND (+) E A
FTER
     38                                 ;              ON (+) OF NEXT E        PIA DESELECTED
     39                                 ;     1 1 0    =CRB<3>
     40                                 ;     1 1 1                            =CRB<3>
     41                                 ;
     42                                 ; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     43                                 ;                       OF CB2, CLEARED BY READ OF PRB_
     44                                 ;
     45                                 ; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     46                                 ;                       OF CB1, CLEARED BY READ OF PRB_
     47                                 ;
     48                                 ;       DATA DIRECTION REGISTER (DDRB_)
     49                                 ; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
     50                                 ;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
     51                                 ;               BY 0'S AND OUTPUT BITS BY 1'S.
     52                                 ;
     53                                 ;       PERIPHERAL DATA REGISTER (PRB_)
     54                                 ; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGIST
ERS
     55                                 ;
```

A.6821 VARIABLES AND POINTERS

```
                                        .SBTTL  A.6821 VARIABLES AND POINTERS
                                        ;
                                        ;       PORT STATUS AND BUFFERSS
                                        ;
                                        .IF     NDF,$A.6821
                                        A.6821  =       200     ;PIA PORT DEFAULT ADDRESS
                                        VARSAV  =       0       ;DEFAULT VARIABLE ADDRESS
                                        PGMSAV  =       100000  ;DEFAULT PROGRAM ADDRESS
                                        .=PGMSAV
                                        .ENDC
                                        ;
        174737                          PGMSAV=.
        000014                          .=VARSAV
                                        ;
000014    000     000           A.APRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
000016    000     000                   .BYTE   0,0     ;'UP','DN' CONTROL
000020    000     000                   .BYTE   0,0     ;<7:0> SERVICE VECTOR
000022    000     000                   .BYTE   0,0     ;'UP','DN' CONTROL
000024    000     000           A.BPRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
000026    000     000                   .BYTE   0,0     ;'UP','DN' CONTROL
000030    000     000                   .BYTE   0,0     ;<7:0> SERVICE VECTOR
000032    000     000                   .BYTE   0,0     ;'UP','DN' CONTROL
                                        ;
        000034                          VARSAV=.
        174737                          .=PGMSAV
                                        ;
```

A.6821 A-PORT CONTROL ROUTINES

```
                                        .SBTTL  A.6821 A-PORT CONTROL ROUTINES
                                        ;
                                        ;       DATA DIRECTION LOAD ROUTINES
                                        ;       ENTER WITH DIRECTION CODE IN <B>
                                        ;
174737                          A.AHDR: LDA A   CRAH+A.6821     ;GET CURRENT CONTROL
174737  226  200
174741                                  PSH A                   ;AND SAVE
174741  066
174742                                  AND A   #,373           ;MASK DDR BIT
174742  204  373
174744                                  STA A   CRAH+A.6821     ;ACCESS DIRECTION REGISTER
174744  227  200
174746                                  STA B   DDRAH+A.6821    ;SET DIRECTIONS
174746  327  204
174750                                  PUL A                   ;GET SAVED CONTROL
174750  062
174751                                  STA A   CRAH+A.6821     ;AND RESTORE
174751  227  200
174753                                  RTS
174753  071
                                        ;
174754                          A.ALDR: LDA A   CRAL+A.6821     ;GET CURRENT CONTROL
174754  226  201
174756                                  PSH A                   ;AND SAVE
174756  066
174757                                  AND A   #,373           ;MASK DDR BIT
174757  204  373
174761                                  STA A   CRAL+A.6821     ;ACCESS DIRECTION REGISTER
174761  227  201
174763                                  STA B   DDRAL+A.6821    ;SET DIRCTIONS
174763  327  205
174765                                  PUL A                   ;GET SAVED CONTROL
174765  062
174766                                  STA A   CRAL+A.6821     ;AND RESTORE
```

```
    22 174770                           RTS
       174770     071
    23                                  ;
```

A.6821 A-PORT CONTROL ROUTINES

```
                                        ;       CONTROL AND SERVICE VECTOR LOADER
                                        ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
                                        ;       'UP' CONTROL IN <B>, AND
                                        ;       'DN' CONTROL IN <A>
                                        ;
 174771                         A.AHDF: STX     A.APRT          ;SAVE SERVICE ROUTINE ADDRESS
 174771  337  014
 174773                                 STA B   A.APRT+2        ;'UP' CONTROL
 174773  327  016
 174775                                 STA A   A.APRT+3        ;'DN' CONTROL
 174775  227  017
 174777                                 RTS
 174777  071
                                        ;
 175000                         A.ALDF: STX     A.APRT+4        ;SAVE SERVICE ROUTINE ADDRESS
 175000  337  020
 175002                                 STA B   A.APRT+6        ;'UP' CONTROL
 175002  327  022
 175004                                 STA A   A.APRT+7        ;'DN' CONTROL
 175004  227  023
 175006                                 RTS
 175006  071
                                        ;
                                        ;
                                        ;       INTERRUPT CONTROL LOADERS
                                        ;
 175007                         A.AHUP: LDA B   A.APRT+2        ;'UP' CONTROL
 175007  326  016
 175011                                 STA B   CRAH+A.6821
 175011  327  200
 175013                                 RTS
 175013  071
                                        ;
 175014                         A.ALUP: LDA B   A.APRT+6        ;'UP' CONTROL
 175014  326  022
```

```
   175016     327     201
25 175020                              RTS
   175020     071
26                                     ;
27 175021                        A.AHDN: LDA B   A.APRT+3        ;'DN' CONTROL
   175021     326     017
28 175023                              STA B   CRAH+A.6821
   175023     327     200
29 175025                              RTS
   175025     071
30                                     ;
31 175026                        A.ALDN: LDA B   A.APRT+7        ;'DN' CONTROL
   175026     326     023
32 175030                              STA B   CRAL+A.6821
   175030     327     201
33 175032                              RTS
   175032     071
34                                     ;
```

A.6821 B-PORT CONTROL ROUTINES

```
                                        .SBTTL  A.6821 B-PORT CONTROL ROUTINES
                                        ;
                                        ;       DATA DIRECTION LOAD ROUTINES
                                        ;       ENTER WITH DIRECTION CODE IN <B>
                                        ;
175033                          A.BHDR: LDA A   CRBH+A.6821     ;GET CURRENT CONTROL
175033  226  202
175035                                  PSH A                   ;AND SAVE
175035  066
175036                                  AND A   #,373           ;MASK DDR BIT
175036  204  373
175040                                  STA A   CRBH+A.6821     ;ACCESS DIRECTION REGISTER
175040  227  202
175042                                  STA B   DDRBH+A.6821    ;SET DIRECTIONS
175042  327  206
175044                                  PUL A                   ;GET SAVED CONTROL
175044  062
175045                                  STA A   CRBH+A.6821     ;AND RESTORE
175045  227  202
175047                                  RTS
175047  071
                                        ;
175050                          A.BLDR: LDA A   CRBL+A.6821     ;GET CURRENT CONTROL
175050  226  203
175052                                  PSH A                   ;AND SAVE
175052  066
175053                                  AND A   #,373           ;MASK DDR BIT
175053  204  373
175055                                  STA A   CRBL+A.6821     ;ACCESS DIRECTION REGISTER
175055  227  203
175057                                  STA B   DDRBL+A.6821    ;SET DIRCTIONS
175057  327  207
175061                                  PUL A                   ;GET SAVED CONTROL
175061  062
175062                                  STA A   CRBL+A.6821     ;AND RESTORE
```

```
    22 175064                               RTS
       175064     071
    23                                      ;
```

A.6821 B-PORT CONTROL ROUTINES

```
 1                                      ;       CONTROL AND SERVICE VECTOR LOADER
 2                                      ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
 3                                      ;       'UP' CONTROL IN <B>, AND
 4                                      ;       'DN' CONTROL IN <A>
 5                                      ;
 6 175065                       A.BHDF: STX     A.BPRT          ;SAVE SERVICE ROUTINE ADDRESS
   175065     337     024
 7 175067                               STA B   A.BPRT+2        ;'UP' CONTROL
   175067     327     026
 8 175071                               STA A   A.BPRT+3        ;'DN' CONTROL
   175071     227     027
 9 175073                               RTS
   175073     071
10                                      ;
11 175074                       A.BLDF: STX     A.BPRT+4        ;SAVE SERVICE ROUTINE ADDRESS
   175074     337     030
12 175076                               STA B   A.BPRT+6        ;'UP' CONTROL
   175076     327     032
13 175100                               STA A   A.BPRT+7        ;'DN' CONTROL
   175100     227     033
14 175102                               RTS
   175102     071
15                                      ;
16                                      ;
17                                      ;       INTERRUPT CONTROL LOADERS
18                                      ;
19 175103                       A.BHUP: LDA B   A.BPRT+2        ;'UP' CONTROL
   175103     326     026
20 175105                               STA B   CRBH+A.6821
   175105     327     202
21 175107                               RTS
   175107     071
22                                      ;
23 175110                       A.BLUP: LDA B   A.BPRT+6        ;'UP' CONTROL
   175110     326     032
```

```
   175112     327     203
25 175114                              RTS
   175114     071
26                                     ;
27 175115                         A.BHDN: LDA B   A.BPRT+3        ;'DN' CONTROL
   175115     326     027
28 175117                              STA B   CRBH+A.6821
   175117     327     202
29 175121                              RTS
   175121     071
30                                     ;
31 175122                         A.BLDN: LDA B   A.BPRT+7        ;'DN' CONTROL
   175122     326     033
32 175124                              STA B   CRBL+A.6821
   175124     327     203
33 175126                              RTS
   175126     071
34                                     ;
```

A.6821 A-PORT TRANSFERS

```
                                        .SBTTL  A.6821 A-PORT TRANSFERS
                                        ;
                                        ;       BYTE DATA IS TRANSFERED IN <B>
                                        ;       WORD DATA IS TRANSFERED IN <B,A>
                                        ;
175127                          A.AHRD: BRA     2$              ;TEST ENTRY
175127  040     004
175131                          1$:     LDA B   PRAH+A.6821     ;GET BYTE
175131  326     204
175133                                  SEC                     ;HAVE BYTE
175133  015
175134                                  RTS
175134  071
175135                          2$:     LDA A   CRAH+A.6821     ;CHECK FOR IRQ FLAG
175135  226     200
175137                                  BIT A   #,300
175137  205     300
175141                                  BNE     1$              ;BRANCH IF FLAG SET
175141  046     366
175143                                  CLC                     ;NO DATA
175143  014
175144                                  RTS
175144  071
                                        ;
175145                          A.ALRD: BRA     2$              ;TEST ENTRY
175145  040     004
175147                          1$:     LDA B   PRAL+A.6821     ;GET BYTE
175147  326     205
175151                                  SEC                     ;HAVE BYTE
175151  015
175152                                  RTS
175152  071
175153                          2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRG FLAG
175153  226     201
175155                                  BIT A   #,300
```

```
 22 175157                              BNE     1$              ;BRANCH IF FLAG SET
    175157     046     366
 23 175161                              CLC                     ;NO DATA
    175161     014
 24 175162                              RTS
    175162     071
 25                                     ;
 26 175163                      A.AWRD: BRA     2$              ;TEST ENTRY
    175163     040     006
 27 175165                      1$:     LDA B   PRAH+A.6821     ;GET HIGH BYTE
    175165     326     204
 28 175167                              LDA A   PRAL+A.6821     ;AND LOW BYTE
    175167     226     205
 29 175171                              SEC                     ;HAVE WORD
    175171     015
 30 175172                              RTS
    175172     071
 31 175173                      2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
    175173     226     201
 32 175175                              BIT A   #,300
    175175     205     300
```

A.6821 A-PORT TRANSFERS

```
     33 175177                              BNE     1$              ;BRANCH IF FLAG SET
        175177     046     364
     34 175201                              CLC                     ;NO DATA
        175201     014
     35 175202                              RTS
        175202     071
     36                                     ;
```

A.6821 A-PORT TRANSFERS

```
     1                                          ;
     2 175203                         A.AHWT: BRA     2$              ;TEST ENTRY
       175203     040     007
     3 175205                         1$:     STA B   PRAH+A.6821     ;STORE BYTE
       175205     327     204
     4 175207                                 TST     PRAH+A.6821     ;CLEAR IRQ FLAGS
       175207     175     000     204
     5 175212                                 SEC                     ;BYTE SENT
       175212     015
     6 175213                                 RTS
       175213     071
     7 175214                         2$:     LDA A   CRAH+A.6821     ;CHECK FOR IRQ FLAG
       175214     226     200
     8 175216                                 BIT A   #,300
       175216     205     300
     9 175220                                 BNE     1$              ;BRANCH IF FLAG SET
       175220     046     363
    10 175222                                 CLC                     ;DATA NOT SENT
       175222     014
    11 175223                                 RTS
       175223     071
    12                                          ;
    13 175224                         A.ALWT: BRA     2$              ;TEST ENTRY
       175224     040     007
    14 175226                         1$:     STA B   PRAL+A.6821     ;STORE BYTE
       175226     327     205
    15 175230                                 TST     PRAL+A.6821     ;CLEAR IRQ FLAGS
       175230     175     000     205
    16 175233                                 SEC                     ;BYTE SENT
       175233     015
    17 175234                                 RTS
       175234     071
    18 175235                         2$:     LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
       175235     226     201
    19 175237                                 BIT A   #,300
```

```
 20 175241                              BNE     1$              ;BRANCH IF FLAG SET
    175241     046     363
 21 175243                              CLC                     ;DATA NOT SENT
    175243     014
 22 175244                              RTS
    175244     071
 23                                     ;
 24 175245                      A.AWWT: BRA     2$              ;TEST ENTRY
    175245     040     014
 25 175247                      1$:     STA B   PRAH+A.6821     ;SEND HIGH BYTE
    175247     327     204
 26 175251                              STA A   PRAL+A.6821     ;SEND LOW BYTE
    175251     227     205
 27 175253                              TST     PRAH+A.6821     ;CLEAR IRQ FLAGS
    175253     175     000     204
 28 175256                              TST     PRAL+A.6821     ;CLEAR IRQ FLAGS
    175256     175     000     205
 29 175261                              SEC                     ;WORD SENT
    175261     015
 30 175262                              RTS
    175262     071
```

A.6821 A-PORT TRANSFERS

```
    31 175263                      2$:     PSH A                   ;SAVE A
       175263     066
    32 175264                              LDA A   CRAL+A.6821     ;CHECK FOR IRQ FLAG
       175264     226     201
    33 175266                              BIT A   #,300
       175266     205     300
    34 175270                              PUL A                   ;RESTORE A
       175270     062
    35 175271                              BNE     1$              ;BRANCH IF FLAG SET
       175271     046     354
    36 175273                              CLC                     ;DATA NOT SENT
       175273     014
    37 175274                              RTS
       175274     071
    38                                     ;
```

A.6821 B-PORT TRANSFERS

```
                                        .SBTTL  A.6821 B-PORT TRANSFERS
                                        ;
                                        ;       BYTE DATA IS TRANSFERED IN <B>
                                        ;       WORD DATA IS TRANSFERED IN <B,A>
                                        ;
175275                          A.BHRD: BRA     2$              ;TEST ENTRY
  175275    040     006
175277                          1$:     LDA B   PRBH+A.6821     ;GET BYTE
  175277    326     206
175301                                  STA B   PRBH+A.6821     ;DO CONTROL
  175301    327     206
175303                                  SEC                     ;HAVE BYTE
  175303    015
175304                                  RTS
  175304    071
175305                          2$:     LDA A   CRBH+A.6821     ;CHECK FOR IRQ FLAG
  175305    226     202
175307                                  BIT A   #,300
  175307    205     300
175311                                  BNE     1$              ;BRANCH IF FLAG SET
  175311    046     364
175313                                  CLC                     ;NO DATA
  175313    014
175314                                  RTS
  175314    071
                                        ;
175315                          A.BLRD: BRA     2$              ;TEST ENTRY
  175315    040     006
175317                          1$:     LDA B   PRBL+A.6821     ;GET BYTE
  175317    326     207
175321                                  STA B   PRBL+A.6821     ;DO CONTROL
  175321    327     207
175323                                  SEC                     ;HAVE BYTE
  175323    015
175324                                  RTS
```

```
 22 175325                      2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRG FLAG
    175325     226     203
 23 175327                              BIT A   #,300
    175327     205     300
 24 175331                              BNE     1$              ;BRANCH IF FLAG SET
    175331     046     364
 25 175333                              CLC                     ;NO DATA
    175333     014
 26 175334                              RTS
    175334     071
 27                                     ;
 28 175335                      A.BWRD: BRA     2$              ;TEST ENTRY
    175335     040     012
 29 175337                      1$:     LDA B   PRBH+A.6821     ;GET HIGH BYTE
    175337     326     206
 30 175341                              LDA A   PRBL+A.6821     ;AND LOW BYTE
    175341     226     207
 31 175343                              STA B   PRBH+A.6821     ;DO CONTROL
    175343     327     206
 32 175345                              STA A   PRBL+A.6821     ;DO CONTROL
    175345     227     207
```

A.6821 B-PORT TRANSFERS

```
     33 175347                              SEC                     ;HAVE WORD
        175347     015
     34 175350                              RTS
        175350     071
     35 175351                        2$:   LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
        175351     226     203
     36 175353                              BIT A   #,300
        175353     205     300
     37 175355                              BNE     1$              ;BRANCH IF FLAG SET
        175355     046     360
     38 175357                              CLC                     ;NO DATA
        175357     014
     39 175360                              RTS
        175360     071
     40                                     ;
```

A.6821 B-PORT TRANSFERS

```
 1                                          ;
 2 175361                         A.BHWT: BRA     2$              ;TEST ENTRY
   175361     040     007
 3 175363                         1$:     TST     PRBH+A.6821     ;CLEAR IRQ FLAGS
   175363     175     000     206
 4 175366                                 STA B   PRBH+A.6821     ;STORE BYTE
   175366     327     206
 5 175370                                 SEC                     ;BYTE SENT
   175370     015
 6 175371                                 RTS
   175371     071
 7 175372                         2$:     LDA A   CRBH+A.6821     ;CHECK FOR IRQ FLAG
   175372     226     202
 8 175374                                 BIT A   #,300
   175374     205     300
 9 175376                                 BNE     1$              ;BRANCH IF FLAG SET
   175376     046     363
10 175400                                 CLC                     ;DATA NOT SENT
   175400     014
11 175401                                 RTS
   175401     071
12                                          ;
13 175402                         A.BLWT: BRA     2$              ;TEST ENTRY
   175402     040     007
14 175404                         1$:     TST     PRBL+A.6821     ;CLEAR IRQ FLAGS
   175404     175     000     207
15 175407                                 STA B   PRBL+A.6821     ;STORE BYTE
   175407     327     207
16 175411                                 SEC                     ;BYTE SENT
   175411     015
17 175412                                 RTS
   175412     071
18 175413                         2$:     LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
   175413     226     203
19 175415                                 BIT A   #,300
```

```
 20 175417                              BNE     1$              ;BRANCH IF FLAG SET
    175417     046     363
 21 175421                              CLC                     ;DATA NOT SENT
    175421     014
 22 175422                              RTS
    175422     071
 23                                     ;
 24 175423                      A.BWWT: BRA     2$              ;TEST ENTRY
    175423     040     014
 25 175425                      1$:     TST     PRBH+A.6821     ;CLEAR IRQ FLAGS
    175425     175     000     206
 26 175430                              TST     PRBL+A.6821     ;CLEAR IRQ FLAGS
    175430     175     000     207
 27 175433                              STA B   PRBH+A.6821     ;SEND HIGH BYTE
    175433     327     206
 28 175435                              STA A   PRBL+A.6821     ;SEND LOW BYTE
    175435     227     207
 29 175437                              SEC                     ;WORD SENT
    175437     015
 30 175440                              RTS
    175440     071
```

A.6821 B-PORT TRANSFERS

```
 31 175441                      2$:     PSH A                   ;SAVE A
    175441     066
 32 175442                              LDA A   CRBL+A.6821     ;CHECK FOR IRQ FLAG
    175442     226     203
 33 175444                              BIT A   #,300
    175444     205     300
 34 175446                              PUL A                   ;RESTORE A
    175446     062
 35 175447                              BNE     1$              ;BRANCH IF FLAG SET
    175447     046     354
 36 175451                              CLC                     ;DATA NOT SENT
    175451     014
 37 175452                              RTS
    175452     071
 38                                     ;
```

A.6821 INTERRUPT SERVICE DISPATCH ROUTINES

```
                                        .SBTTL  A.6821 INTERRUPT SERVICE DISPATCH ROUTINES
                                        ;
175453                          A.AH$I: LDX     A.APRT          ;GET SERVICE ADDRESS
  175453    336     014
175455                                  BNE     A.IRQG          ;BRANCH IF DEFINED
  175455    046     040
175457                                  LDX     #,CRAH+A.6821   ;INTERRUPT CONTROL ADDRESS
  175457    316     000     200
175462                                  BRA     A.IRQD          ;TERMINATE INTERRUPTS
  175462    040     036
                                        ;
175464                          A.AL$I: LDX     A.APRT+4        ;GET SERVICE ADDRESS
  175464    336     020
175466                                  BNE     A.IRQG          ;BRANCH IF DEFINED
  175466    046     027
175470                                  LDX     #,CRAL+A.6821   ;INTERRUPT CONTROL ADDRESS
  175470    316     000     201
175473                                  BRA     A.IRQD          ;TERMINATE INTERRUPTS
  175473    040     025
                                        ;
175475                          A.BH$I: LDX     A.BPRT          ;GET SERVICE ADDRESS
  175475    336     024
175477                                  BNE     A.IRQG          ;BRANCH IF DEFINED
  175477    046     016
175501                                  LDX     #,CRBH+A.6821   ;INTERRUPT CONTROL ADDRESS
  175501    316     000     202
175504                                  BRA     A.IRQD          ;TERMINATE INTERRUPTS
  175504    040     014
                                        ;
175506                          A.BL$I: LDX     A.BPRT+4        ;GET SERVICE ADDRESS
  175506    336     030
175510                                  BNE     A.IRQG          ;BRANCH IF DEFINED
  175510    046     005
175512                                  LDX     #,CRBL+A.6821   ;INTERRUPT CONTROL ADDRESS
  175512    316     000     203
```

```
   175515     040     003
22                                 ;
23 175517                  A.IRQG: JSR     0,X             ;DO SERVICE ROUTINE
   175517     255     000
24 175521                          RTI
   175521     073
25                                 ;
26 175522                  A.IRQD: LDA A   0,X             ;GET CONTROL REGISTER
   175522     246     000
27 175524                          AND B   #,376           ;CLEAR CX1 INTERRUPT ENABLE
   175524     304     376
28 175526                          BIT B   #,40            ;CX2 INTERRUPT MODE ?
   175526     305     040
29 175530                          BNE     1$              ;BRANCH IF NOT
   175530     046     002
30 175532                          AND B   #,367           ;CLEAR CX2 INTERRUPT ENABLE
   175532     304     367
31 175534                  1$:     STA B   0,X             ;SET NEW CONTROL
   175534     347     000
32 175536                          RTI
   175536     073
```

PARALLLEL 'PORT' MACRO V05.06 Thursday 18-Oct-07 19:15 Page 23-1

A.6821 INTERRUPT SERVICE DISPATCH ROUTINES

```
     33                                     ;
     34                                     .IIF    NDF,$A.6821     .END
```

B.6821 PERIPHERAL INTERFACE SOFTWARE

```
                                .SBTTL  B.6821 PERIPHERAL INTERFACE SOFTWARE
                                ;
                                ;         THIS SOFTWARE PACKAGE SUPPORTS THE
                                ;       6821 PERIPHERAL INTERFACE ADAPTER (PIA).
                                ;         ENTRY POINTS TO THE SOFTWARE ARE DEFINED
                                ;       FOR EXTERNAL ACCESS:
                                ;
                                ;         THE FOLLOWING DATA DIRECTION ROUTINES ARE
                                ;       ENTERED WITH
                                ;               <B> = DIRECTION CONTROL
                                ;       BY A JSR ____
                                ;
                                ;       B.AHDR --> A-PORT <15:8> DIRECTION
                                ;       B.ALDR --> A-PORT <7:0> DIRECTION
                                ;       B.BHDR --> B-PORT <15:8> DIRECTION
                                ;       B.BLDR --> B-PORT <7:0> DIRECTION
                                ;
                                ;         THE FOLLOWING CONTROL ROUTINES ARE ENTERED WITH
                                ;               <X> = SERVICE ADDRESS
                                ;               <B> = INTERRUPT ENABLE CONTROL
                                ;               <A> = INTERRUPT DISABLE CONTROL
                                ;       BY A JSR ____
                                ;
                                ;       B.AHDF --> A-PORT <15:8>
                                ;       B.ALDF --> A-PORT <7:0>
                                ;       B.BHDF --> B-PORT <15:8>
                                ;       B.BLDF --> B-PORT <7:0>
                                ;
                                ;         THE FOLLOWING INTERRUPT ENABLE ROUTINES ARE
                                ;       ENTERED BY JSR ____
                                ;
                                ;       B.AHUP --> A-PORT <15:8> INTERRUPT ENABLE
                                ;       B.ALUP --> A-PORT <7:0> INTERRUPT ENABLE
                                ;       B.BHUP --> B-PORT <15:0> INTERRUPT ENABLE
                                ;       B.BLUP --> B-PORT <7:0> INTERRUPT ENABLE
```

```
38                                      ;         THE FOLLOWING INTERRUPT DISABLE ROUTINES ARE
39                                      ;       ENTERED BY JSR ____
40                                      ;
41                                      ;       B.AHDN --> A-PORT <15:0> INTERRUPT DISABLE
42                                      ;       B.ALDN --> A-PORT <7:0> INTERRUPT DISABLE
43                                      ;       B.BHDN --> B-PORT <15:8> INTERRUPT DISABLE
44                                      ;       B.BLDN --> B-PORT <7:0> INTERRUPT DISABLE
45                                      ;
46                                      ;         THE FOLLOWING ARE INTERRUPT ENTRY POINTS
47                                      ;       FOR THE DUAL PIA PORTS
48                                      ;
49                                      ;       B.AH$I --> A-PORT <15:8> IRQ
50                                      ;
51                                      ;       B.AL$I --> A-PORT <7:0> IRQ
52                                      ;
53                                      ;       B.BH$I --> B-PORT <15:8> IRQ
54                                      ;
55                                      ;       B.BL$L --> B-PORT <7:0> IRQ
56                                      ;
```

B.6821 PERIPHERAL INTERFACE SOFTWARE

```
      1                                     ;
      2                                     ;         THE FOLLOWING ENTRY POINTS ARE USED TO TRANSFER
      3                                     ;       DATA TO THE VARIOUS PIA I/O REGISTERS. THE MAIN
      4                                     ;       ENTRY POINT VERIFIES THAT AT LEAST ONE OF THE
      5                                     ;       PORT IRQ FLAGS IS SET BEFORE TRANSFERRING DATA .
      6                                     ;       IF DATA IS TRANSFERED THEN THE 'C' BIT = 1,
      7                                     ;       IF THE PORT WAS NOT READY THEN NO TRANSFER
      8                                     ;       IS PERFORMED AND THE 'C' BIT = 0.
      9                                     ;         THE ROUTINES MAY BE ENTERED AT THE ENTRY POINT
+ 2
     10                                     ;       IF NO IRQ FLAG CHECKS ARE TO BE MADE.
     11                                     ;         BYTE DATA IS PASSED IN <B>.
     12                                     ;         WORD DATA IS PASSED IN <B,A>
     13                                     ;
     14                                     ;       CALL BY JSR ____
     15                                     ;
     16                                     ;       B.AHRD --> A-PORT <15:8> READ
     17                                     ;       B.ALRD --> A-PORT <7:0> READ
     18                                     ;       B.AWRD --> A-PORT <15:0> READ
     19                                     ;       B.BHRD --> B-PORT <15:8> READ
     20                                     ;       B.BLRD --> B-PORT <7:0> READ
     21                                     ;       B.BWRD --> B-PORT <15:0> READ
     22                                     ;
     23                                     ;
     24                                     ;       B.AHWT --> A-PORT <15:8> WRITE
     25                                     ;       B.ALWT --> A-PORT <7:0> WRITE
     26                                     ;       B.AWWT --> A-PORT <15:0> WRITE
     27                                     ;       B.BHWT --> B-PORT <15:8> WRITE
     28                                     ;       B.BLWT --> B-PORT <7:0> WRITE
     29                                     ;       B.BWWT --> B-PORT <15:0> WRITE
     30                                     ;
```

B.6821 PIA HARDWARE

```
                                .SBTTL  B.6821 PIA HARDWARE
                                ;
                                ;         THE PIA CONFIGURATION  CONSISTS OF TWO
                                ;       MC6821 PERIPHERAL INTERFACE ADAPTERS (PIA'S)
                                ;       WHICH MAY BE ADDRESSED AS 16-BIT PORTS.
                                ;       ONE PIA UNIT FORMS THE HIGH ORDER PART <15:8>
                                ;       AND THE OTHER THE LOW ORDER PART <7:0>
                                ;
                                ; REGISTER 0
000000                          CRAH    =0      ;A-PORT <15:8> CONTROL REGISTER
                                ;
                                ; REGISTER 1
000001                          CRAL    =1      ;A-PORT <7:0> CONTROL REGISTER
                                ;
                                ; REGISTER 2
000002                          CRBH    =2      ;B-PORT <15:8> CONTROL REGISTER
                                ;
                                ; REGISTER 3
000003                          CRBL    =3      ;B-PORT <7:0> CONTROL REGISTER
                                ;
                                ; REGISTER 4
000004                          DDRAH   =4      ;CRAH<2>=0, A-PORT <15:8> DATA DIRECTION
REGISTER
000004                          PRAH    =4      ;CRAH<2>=1, A-PORT <15:8> DATA REGISTER
                                ;
                                ; REGISTER 5
000005                          DDRAL   =5      ;CRAL<2>=0, A-PORT <7:0> DATA DIRECTION R
EGISTER
000005                          PRAL    =5      ;CRAL<2>=1, A-PORT <7:0> DATA REGISTER
                                ;
                                ; REGISTER 6
000006                          DDRBH   =6      ;CRBH<2>=0, B-PORT <15:8> DATA DIRECTION
REGISTER
000006                          PRBH    =6      ;CRBH<2>=1, B-PORT <15:8> DATA REGISTER
                                ;
                                ; REGISTER 7
```

```
    35          000007                          PRBL    =7      ;CRBL<2>=1, B-PORT <7:0> DATA REGISTER
    36                                          ;
```

B.6821 PIA HARDWARE

```
      1                                 ;
      2                                 ;       A-PORT CONTROL REGISTERS
      3                                 ;
      4                                 ; CONTROL REGISTERS CRA_  - READ/WRITE REGISTERS
      5                                 ;                                 CONTROLLING INTERRUPTS AN
D STROBES
      6                                 ;
      7                                 ; BIT 1 0       CONTROL OF INTERRUPT INPUT CA1
      8                                 ;               ACTIVE TRANSITION
      9                                 ;     _ _       TO SET CRA<7>           /IRQA OUTPUT
     10                                 ;     0 0               (-)             DISABLED
     11                                 ;     0 1               (-)             /CRA<7>
     12                                 ;     1 0               (+)             DISABLED
     13                                 ;     1 1               (+)             /CRA<7>
     14                                 ;       CRA<7> CLEARED BY A READ OF CORRESPONDING PRA_
     15                                 ;       /IRQA FOLLOWS STATE OF /CRA<7> WHEN ENABLED
     16                                 ;
     17                                 ; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
     18                                 ;     _
     19                                 ;     0         ACCESS DATA DIRECTION REGISTERS
     20                                 ;     1         ACCESS PORT DATA REGISTER
     21                                 ;
     22                                 ; BIT 5 4 3     CONTROL OF CA2 AS AN INTERRUPT INPUT
     23                                 ;               ACTIVE TRANSITION
     24                                 ;     _ _ _     TO SET CRA<6>           /IRQB OUTPUT
     25                                 ;     0 0 0             (-)             DISABLED
     26                                 ;     0 0 1             (-)             /CRA<6>
     27                                 ;     0 1 0             (+)             DISABLED
     28                                 ;     0 1 1             (+)             /CRA<6>
     29                                 ;       CRA<6> CLEARED BY READ OF CORRESPONDING PRA_
     30                                 ;       /IRQB FOLLOWS STATE OF CRA<6> WHEN ENABLED
     31                                 ;
     32                                 ; BIT 5 4 3     CONTROL OF CA2 AS AN OUTPUT
     33                                 ;     _ _ _     CA2 LOW                 CA2 HIGH
     34                                 ;     1 0 0     AFTER READ TO PRA_      WHEN CRA<7> IS SE
T
```

```
     36                                 ;     1 0 1     AFTER READ TO PRA_      ON SECOND (+) E A
FTER
     37                                 ;               ON (+) OF NEXT E        PIA DESELECTED
     38                                 ;     1 1 0     =CRA<3>
     39                                 ;     1 1 1                             =CRA<3>
     40                                 ;
     41                                 ; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     42                                 ;                       OF CA2, CLEARED BY READ OF PRA_
     43                                 ;
     44                                 ; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     45                                 ;                       OF CA1, CLEARED BY READ OF PRA_
     46                                 ;
     47                                 ;       DATA DIRECTION REGISTER (DDRA_)
     48                                 ; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
     49                                 ;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
     50                                 ;               BY 0'S AND OUTPUT BITS BY 1'S.
     51                                 ;
     52                                 ;       PERIPHERAL DATA REGISTER (PRA_)
     53                                 ; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGIST
ERS
     54                                 ;
```

B.6821 DETAILS

```
      1                                 .SBTTL  B.6821 DETAILS
      2                                 ;
      3                                 ;       B-PORT CONTROL REGISTERS
      4                                 ;
      5                                 ; CONTROL REGISTERS CRB_  - READ/WRITE REGISTERS
      6                                 ;                               CONTROLLING INTERRUPTS AN
D STROBES
      7                                 ;
      8                                 ; BIT 1 0       CONTROL OF INTERRUPT INPUT CB1
      9                                 ;               ACTIVE TRANSITION
     10                                 ;     _ _       TO SET CRB<7>           /IRQA OUTPUT
     11                                 ;     0 0               (-)             DISABLED
     12                                 ;     0 1               (-)             /CRB<7>
     13                                 ;     1 0               (+)             DISABLED
     14                                 ;     1 1               (+)             /CRB<7>
     15                                 ;       CRB<7> CLEARED BY A READ OF CORRESPONDING PRB_
     16                                 ;       /IRQA FOLLOWS STATE OF /CRB<7> WHEN ENABLED
     17                                 ;
     18                                 ; BIT 2         DATA / DATA DIRECTION ACCESS CONTROL BIT
     19                                 ;     _
     20                                 ;     0         ACCESS DATA DIRECTION REGISTERS
     21                                 ;     1         ACCESS PORT DATA REGISTER
     22                                 ;
     23                                 ; BIT 5 4 3     CONTROL OF CB2 AS AN INTERRUPT INPUT
     24                                 ;               ACTIVE TRANSITION
     25                                 ;     _ _ _     TO SET CRB<6>           /IRQB OUTPUT
     26                                 ;     0 0 0             (-)             DISABLED
     27                                 ;     0 0 1             (-)             /CRB<6>
     28                                 ;     0 1 0             (+)             DISABLED
     29                                 ;     0 1 1             (+)             /CRB<6>
     30                                 ;       CRB<6> CLEARED BY READ OF CORRESPONDING PRB_
     31                                 ;       /IRQB FOLLOWS STATE OF CRB<6> WHEN ENABLED
     32                                 ;
     33                                 ; BIT 5 4 3     CONTROL OF CB2 AS AN OUTPUT
     34                                 ;     _ _ _     CB2 LOW                 CB2 HIGH
```

```
     36                                 ;              ON (+) OF NEXT E        BY CB1 TRANSITION
     37                                 ;     1 0 1    AFTER WRITE TO PRB      ON SECOND (+) E A
FTER
     38                                 ;              ON (+) OF NEXT E        PIA DESELECTED
     39                                 ;     1 1 0    =CRB<3>
     40                                 ;     1 1 1                            =CRB<3>
     41                                 ;
     42                                 ; BIT 6         IRQB2 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     43                                 ;                       OF CB2, CLEARED BY READ OF PRB_
     44                                 ;
     45                                 ; BIT 7         IRQB1 INTERRUPT FLAG - SET BY ACTIVE TRAN
SITION
     46                                 ;                       OF CB1, CLEARED BY READ OF PRB_
     47                                 ;
     48                                 ;       DATA DIRECTION REGISTER (DDRB_)
     49                                 ; BIT 0-7       EACH DDR BIT CORRESPONDS TO A PERIPHERAL
     50                                 ;               REGISTER BIT. INPUT BITS ARE PROGRAMMED
     51                                 ;               BY 0'S AND OUTPUT BITS BY 1'S.
     52                                 ;
     53                                 ;       PERIPHERAL DATA REGISTER (PRB_)
     54                                 ; BIT 0-7       8-BIT READ/WRITE INPUT/OUTPUT DATA REGIST
ERS
     55                                 ;
```

B.6821 VARIABLES AND POINTERS

```
 1                                      .SBTTL  B.6821 VARIABLES AND POINTERS
 2                                      ;
 3                                      ;       PORT STATUS AND BUFFERSS
 4                                      ;
 5                                      .IF     NDF,$B.6821
 6                                      B.6821  =       210     ;PIA PORT DEFAULT ADDRESS
 7                                      VARSAV  =       0       ;DEFAULT VARIABLE ADDRESS
 8                                      PGMSAV  =       100000  ;DEFAULT PROGRAM ADDRESS
 9                                      .=PGMSAV
10                                      .ENDC
11                                      ;
12         175537                       PGMSAV=.
13         000034                       .=VARSAV
14                                      ;
15 000034     000     000          B.APRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
16 000036     000     000                  .BYTE   0,0     ;'UP','DN' CONTROL
17 000040     000     000                  .BYTE   0,0     ;<7:0> SERVICE VECTOR
18 000042     000     000                  .BYTE   0,0     ;'UP','DN' CONTROL
19 000044     000     000          B.BPRT: .BYTE   0,0     ;<15:8> SERVICE VECTOR
20 000046     000     000                  .BYTE   0,0     ;'UP','DN' CONTROL
21 000050     000     000                  .BYTE   0,0     ;<7:0> SERVICE VECTOR
22 000052     000     000                  .BYTE   0,0     ;'UP','DN' CONTROL
23                                      ;
24         000054                       VARSAV=.
25         175537                       .=PGMSAV
26                                      ;
```

B.6821 A-PORT CONTROL ROUTINES

```
 1                                      .SBTTL  B.6821 A-PORT CONTROL ROUTINES
 2                                      ;
 3                                      ;       DATA DIRECTION LOAD ROUTINES
 4                                      ;       ENTER WITH DIRECTION CODE IN <B>
 5                                      ;
 6 175537                       B.AHDR: LDA A   CRAH+B.6821     ;GET CURRENT CONTROL
   175537     226     210
 7 175541                               PSH A                   ;AND SAVE
   175541     066
 8 175542                               AND A   #,373           ;MASK DDR BIT
   175542     204     373
 9 175544                               STA A   CRAH+B.6821     ;ACCESS DIRECTION REGISTER
   175544     227     210
10 175546                               STA B   DDRAH+B.6821    ;SET DIRECTIONS
   175546     327     214
11 175550                               PUL A                   ;GET SAVED CONTROL
   175550     062
12 175551                               STA A   CRAH+B.6821     ;AND RESTORE
   175551     227     210
13 175553                               RTS
   175553     071
14                                      ;
15 175554                       B.ALDR: LDA A   CRAL+B.6821     ;GET CURRENT CONTROL
   175554     226     211
16 175556                               PSH A                   ;AND SAVE
   175556     066
17 175557                               AND A   #,373           ;MASK DDR BIT
   175557     204     373
18 175561                               STA A   CRAL+B.6821     ;ACCESS DIRECTION REGISTER
   175561     227     211
19 175563                               STA B   DDRAL+B.6821    ;SET DIRCTIONS
   175563     327     215
20 175565                               PUL A                   ;GET SAVED CONTROL
   175565     062
21 175566                               STA A   CRAL+B.6821     ;AND RESTORE
```

B.6821 A-PORT CONTROL ROUTINES

```
 1                                      ;       CONTROL AND SERVICE VECTOR LOADER
 2                                      ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
 3                                      ;       'UP' CONTROL IN <B>, AND
 4                                      ;       'DN' CONTROL IN <A>
 5                                      ;
 6 175571                       B.AHDF: STX     B.APRT          ;SAVE SERVICE ROUTINE ADDRESS
   175571     337     034
 7 175573                               STA B   B.APRT+2        ;'UP' CONTROL
   175573     327     036
 8 175575                               STA A   B.APRT+3        ;'DN' CONTROL
   175575     227     037
 9 175577                               RTS
   175577     071
10                                      ;
11 175600                       B.ALDF: STX     B.APRT+4        ;SAVE SERVICE ROUTINE ADDRESS
   175600     337     040
12 175602                               STA B   B.APRT+6        ;'UP' CONTROL
   175602     327     042
13 175604                               STA A   B.APRT+7        ;'DN' CONTROL
   175604     227     043
14 175606                               RTS
   175606     071
15                                      ;
16                                      ;
17                                      ;       INTERRUPT CONTROL LOADERS
18                                      ;
19 175607                       B.AHUP: LDA B   B.APRT+2        ;'UP' CONTROL
   175607     326     036
20 175611                               STA B   CRAH+B.6821
   175611     327     210
21 175613                               RTS
   175613     071
22                                      ;
23 175614                       B.ALUP: LDA B   B.APRT+6        ;'UP' CONTROL
   175614     326     042
```

B.6821 B-PORT CONTROL ROUTINES

```
                                        .SBTTL  B.6821 B-PORT CONTROL ROUTINES
                                        ;
                                        ;       DATA DIRECTION LOAD ROUTINES
                                        ;       ENTER WITH DIRECTION CODE IN <B>
                                        ;
175633                          B.BHDR: LDA A   CRBH+B.6821     ;GET CURRENT CONTROL
175633  226  212
175635                                  PSH A                   ;AND SAVE
175635  066
175636                                  AND A   #,373           ;MASK DDR BIT
175636  204  373
175640                                  STA A   CRBH+B.6821     ;ACCESS DIRECTION REGISTER
175640  227  212
175642                                  STA B   DDRBH+B.6821    ;SET DIRECTIONS
175642  327  216
175644                                  PUL A                   ;GET SAVED CONTROL
175644  062
175645                                  STA A   CRBH+B.6821     ;AND RESTORE
175645  227  212
175647                                  RTS
175647  071
                                        ;
175650                          B.BLDR: LDA A   CRBL+B.6821     ;GET CURRENT CONTROL
175650  226  213
175652                                  PSH A                   ;AND SAVE
175652  066
175653                                  AND A   #,373           ;MASK DDR BIT
175653  204  373
175655                                  STA A   CRBL+B.6821     ;ACCESS DIRECTION REGISTER
175655  227  213
175657                                  STA B   DDRBL+B.6821    ;SET DIRCTIONS
175657  327  217
175661                                  PUL A                   ;GET SAVED CONTROL
175661  062
175662                                  STA A   CRBL+B.6821     ;AND RESTORE
```

B.6821 B-PORT CONTROL ROUTINES

```
                                        ;       CONTROL AND SERVICE VECTOR LOADER
                                        ;       ENTER WITH SERVICE ROUTINE ADDRESS IN <X>
                                        ;       'UP' CONTROL IN <B>, AND
                                        ;       'DN' CONTROL IN <A>
                                        ;
 175665                         B.BHDF: STX     B.BPRT          ;SAVE SERVICE ROUTINE ADDRESS
 175665  337  044
                                        STA B   B.BPRT+2        ;'UP' CONTROL
 175667  327  046
                                        STA A   B.BPRT+3        ;'DN' CONTROL
 175671  227  047
                                        RTS
 175673  071
                                        ;
 175674                         B.BLDF: STX     B.BPRT+4        ;SAVE SERVICE ROUTINE ADDRESS
 175674  337  050
                                        STA B   B.BPRT+6        ;'UP' CONTROL
 175676  327  052
                                        STA A   B.BPRT+7        ;'DN' CONTROL
 175700  227  053
                                        RTS
 175702  071
                                        ;
                                        ;
                                        ;       INTERRUPT CONTROL LOADERS
                                        ;
 175703                         B.BHUP: LDA B   B.BPRT+2        ;'UP' CONTROL
 175703  326  046
                                        STA B   CRBH+B.6821
 175705  327  212
                                        RTS
 175707  071
                                        ;
 175710                         B.BLUP: LDA B   B.BPRT+6        ;'UP' CONTROL
 175710  326  052
```

B.6821 A-PORT TRANSFERS

```
                                        .SBTTL  B.6821 A-PORT TRANSFERS
                                        ;
                                        ;       BYTE DATA IS TRANSFERED IN <B>
                                        ;       WORD DATA IS TRANSFERED IN <B,A>
                                        ;
175727                          B.AHRD: BRA     2$              ;TEST ENTRY
  175727    040     004
175731                          1$:     LDA B   PRAH+B.6821     ;GET BYTE
  175731    326     214
175733                                  SEC                     ;HAVE BYTE
  175733    015
175734                                  RTS
  175734    071
175735                          2$:     LDA A   CRAH+B.6821     ;CHECK FOR IRQ FLAG
  175735    226     210
175737                                  BIT A   #,300
  175737    205     300
175741                                  BNE     1$              ;BRANCH IF FLAG SET
  175741    046     366
175743                                  CLC                     ;NO DATA
  175743    014
175744                                  RTS
  175744    071
                                        ;
175745                          B.ALRD: BRA     2$              ;TEST ENTRY
  175745    040     004
175747                          1$:     LDA B   PRAL+B.6821     ;GET BYTE
  175747    326     215
175751                                  SEC                     ;HAVE BYTE
  175751    015
175752                                  RTS
  175752    071
175753                          2$:     LDA A   CRAL+B.6821     ;CHECK FOR IRG FLAG
  175753    226     211
175755                                  BIT A   #,300
```

PARALLLEL 'PORT' MACRO V05.06 Thursday 18-Oct-07 19:15 Page 34-1

B.6821 A-PORT TRANSFERS

```
    33 175777                           BNE     1$              ;BRANCH IF FLAG SET
       175777     046     364
    34 176001                           CLC                     ;NO DATA
       176001     014
    35 176002                           RTS
       176002     071
    36                                  ;
```

B.6821 A-PORT TRANSFERS

```
     1                                          ;
     2 176003                         B.AHWT: BRA     2$              ;TEST ENTRY
       176003     040     007
     3 176005                         1$:     STA B   PRAH+B.6821     ;STORE BYTE
       176005     327     214
     4 176007                                 TST     PRAH+B.6821     ;CLEAR IRQ FLAGS
       176007     175     000     214
     5 176012                                 SEC                     ;BYTE SENT
       176012     015
     6 176013                                 RTS
       176013     071
     7 176014                         2$:     LDA A   CRAH+B.6821     ;CHECK FOR IRQ FLAG
       176014     226     210
     8 176016                                 BIT A   #,300
       176016     205     300
     9 176020                                 BNE     1$              ;BRANCH IF FLAG SET
       176020     046     363
    10 176022                                 CLC                     ;DATA NOT SENT
       176022     014
    11 176023                                 RTS
       176023     071
    12                                          ;
    13 176024                         B.ALWT: BRA     2$              ;TEST ENTRY
       176024     040     007
    14 176026                         1$:     STA B   PRAL+B.6821     ;STORE BYTE
       176026     327     215
    15 176030                                 TST     PRAL+B.6821     ;CLEAR IRQ FLAGS
       176030     175     000     215
    16 176033                                 SEC                     ;BYTE SENT
       176033     015
    17 176034                                 RTS
       176034     071
    18 176035                         2$:     LDA A   CRAL+B.6821     ;CHECK FOR IRQ FLAG
       176035     226     211
    19 176037                                 BIT A   #,300
```

B.6821 A-PORT TRANSFERS

```
    31 176063                      2$:     PSH A                   ;SAVE A
       176063     066
    32 176064                              LDA A   CRAL+B.6821     ;CHECK FOR IRQ FLAG
       176064     226     211
    33 176066                              BIT A   #,300
       176066     205     300
    34 176070                              PUL A                   ;RESTORE A
       176070     062
    35 176071                              BNE     1$              ;BRANCH IF FLAG SET
       176071     046     354
    36 176073                              CLC                     ;DATA NOT SENT
       176073     014
    37 176074                              RTS
       176074     071
    38                                     ;
```

B.6821 B-PORT TRANSFERS

```
                                        .SBTTL  B.6821 B-PORT TRANSFERS
                                        ;
                                        ;       BYTE DATA IS TRANSFERED IN <B>
                                        ;       WORD DATA IS TRANSFERED IN <B,A>
                                        ;
176075                          B.BHRD: BRA     2$              ;TEST ENTRY
176075  040     006
176077                          1$:     LDA B   PRBH+B.6821     ;GET BYTE
176077  326     216
176101                                  STA B   PRBH+B.6821     ;DO CONTROL
176101  327     216
176103                                  SEC                     ;HAVE BYTE
176103  015
176104                                  RTS
176104  071
176105                          2$:     LDA A   CRBH+B.6821     ;CHECK FOR IRQ FLAG
176105  226     212
176107                                  BIT A   #,300
176107  205     300
176111                                  BNE     1$              ;BRANCH IF FLAG SET
176111  046     364
176113                                  CLC                     ;NO DATA
176113  014
176114                                  RTS
176114  071
                                        ;
176115                          B.BLRD: BRA     2$              ;TEST ENTRY
176115  040     006
176117                          1$:     LDA B   PRBL+B.6821     ;GET BYTE
176117  326     217
176121                                  STA B   PRBL+B.6821     ;DO CONTROL
176121  327     217
176123                                  SEC                     ;HAVE BYTE
176123  015
176124                                  RTS
```

B.6821 B-PORT TRANSFERS

```
     33 176147                              SEC                     ;HAVE WORD
        176147     015
     34 176150                              RTS
        176150     071
     35 176151                         2$:  LDA A   CRBL+B.6821      ;CHECK FOR IRQ FLAG
        176151     226     213
     36 176153                              BIT A   #,300
        176153     205     300
     37 176155                              BNE     1$               ;BRANCH IF FLAG SET
        176155     046     360
     38 176157                              CLC                     ;NO DATA
        176157     014
     39 176160                              RTS
        176160     071
     40                                     ;
```

B.6821 B-PORT TRANSFERS

```
     1                                          ;
     2 176161                         B.BHWT: BRA     2$              ;TEST ENTRY
       176161     040     007
     3 176163                         1$:     TST     PRBH+B.6821     ;CLEAR IRQ FLAGS
       176163     175     000     216
     4 176166                                 STA B   PRBH+B.6821     ;STORE BYTE
       176166     327     216
     5 176170                                 SEC                     ;BYTE SENT
       176170     015
     6 176171                                 RTS
       176171     071
     7 176172                         2$:     LDA A   CRBH+B.6821     ;CHECK FOR IRQ FLAG
       176172     226     212
     8 176174                                 BIT A   #,300
       176174     205     300
     9 176176                                 BNE     1$              ;BRANCH IF FLAG SET
       176176     046     363
    10 176200                                 CLC                     ;DATA NOT SENT
       176200     014
    11 176201                                 RTS
       176201     071
    12                                          ;
    13 176202                         B.BLWT: BRA     2$              ;TEST ENTRY
       176202     040     007
    14 176204                         1$:     TST     PRBL+B.6821     ;CLEAR IRQ FLAGS
       176204     175     000     217
    15 176207                                 STA B   PRBL+B.6821     ;STORE BYTE
       176207     327     217
    16 176211                                 SEC                     ;BYTE SENT
       176211     015
    17 176212                                 RTS
       176212     071
    18 176213                         2$:     LDA A   CRBL+B.6821     ;CHECK FOR IRQ FLAG
       176213     226     213
    19 176215                                 BIT A   #,300
```

PARALLLEL 'PORT' MACRO V05.06 Thursday 18-Oct-07 19:15 Page 37-1

B.6821 B-PORT TRANSFERS

```
    31 176241                       2$:     PSH A                   ;SAVE A
       176241     066
    32 176242                               LDA A   CRBL+B.6821     ;CHECK FOR IRQ FLAG
       176242     226     213
    33 176244                               BIT A   #,300
       176244     205     300
    34 176246                               PUL A                   ;RESTORE A
       176246     062
    35 176247                               BNE     1$              ;BRANCH IF FLAG SET
       176247     046     354
    36 176251                               CLC                     ;DATA NOT SENT
       176251     014
    37 176252                               RTS
       176252     071
    38                                      ;
```

B.6821 INTERRUPT SERVICE DISPATCH ROUTINES

```
                                        .SBTTL  B.6821 INTERRUPT SERVICE DISPATCH ROUTINES
                                        ;
176253                          B.AH$I: LDX     B.APRT          ;GET SERVICE ADDRESS
  176253    336     034
176255                                  BNE     B.IRQG          ;BRANCH IF DEFINED
  176255    046     040
176257                                  LDX     #,CRAH+B.6821   ;INTERRUPT CONTROL ADDRESS
  176257    316     000     210
176262                                  BRA     B.IRQD          ;TERMINATE INTERRUPTS
  176262    040     036
                                        ;
176264                          B.AL$I: LDX     B.APRT+4        ;GET SERVICE ADDRESS
  176264    336     040
176266                                  BNE     B.IRQG          ;BRANCH IF DEFINED
  176266    046     027
176270                                  LDX     #,CRAL+B.6821   ;INTERRUPT CONTROL ADDRESS
  176270    316     000     211
176273                                  BRA     B.IRQD          ;TERMINATE INTERRUPTS
  176273    040     025
                                        ;
176275                          B.BH$I: LDX     B.BPRT          ;GET SERVICE ADDRESS
  176275    336     044
176277                                  BNE     B.IRQG          ;BRANCH IF DEFINED
  176277    046     016
176301                                  LDX     #,CRBH+B.6821   ;INTERRUPT CONTROL ADDRESS
  176301    316     000     212
176304                                  BRA     B.IRQD          ;TERMINATE INTERRUPTS
  176304    040     014
                                        ;
176306                          B.BL$I: LDX     B.BPRT+4        ;GET SERVICE ADDRESS
  176306    336     050
176310                                  BNE     B.IRQG          ;BRANCH IF DEFINED
  176310    046     005
176312                                  LDX     #,CRBL+B.6821   ;INTERRUPT CONTROL ADDRESS
  176312    316     000     213
```

PARALLLEL 'PORT' MACRO V05.06 Thursday 18-Oct-07 19:15 Page 38-1

B.6821 INTERRUPT SERVICE DISPATCH ROUTINES

```
     33                                     ;
     34                                     .IIF    NDF,$B.6821     .END
```

THE END

```
      2                                       .SBTTL  THE END
      3                                       ;
      4         000001                        .END
```

Symbol table

```
A.AHDF  174771          A.BLRD  175315          B.BHDF  175665          DDRBL = 000007          NMIRQV  0
00000

A.AHDN  175021          A.BLUP  175110          B.BHDN  175715          IRQN    177770          PGMSAV= 1
75537

A.AHDR  174737          A.BLWT  175402          B.BHDR  175633          IRQ0    177750          PINITS  1
74101

A.AHRD  175127          A.BL$I  175506          B.BHRD  176075          IRQ1    177752          PRAH  = 0
00004

A.AHUP  175007          A.BPRT  000024          B.BHUP  175703          IRQ2    177754          PRAL  = 0
00005

A.AHWT  175203          A.BWRD  175335          B.BHWT  176161          IRQ3    177756          PRBH  = 0
00006

A.AH$I  175453          A.BWWT  175423          B.BH$I  176275          IRQ4    177760          PRBL  = 0
00007

A.ALDF  175000          A.IRQD  175522          B.BLDF  175674          IRQ5    177762          P.BSAH  1
74520

A.ALDN  175026          A.IRQG  175517          B.BLDN  175722          IRQ6    177764          P.BSAL  1
74374

A.ALDR  174754          A.6821= 000200          B.BLDR  175650          IRQ7    177766          P.BSBL  1
74503

A.ALRD  175145          B.AHDF  175571          B.BLRD  176115          LOADER  174550          P.FNCT  0
00004

A.ALUP  175014          B.AHDN  175621          B.BLUP  175710          LOAD$I  174542          P.IOAL  1
74456

A.ALWT  175224          B.AHDR  175537          B.BLWT  176202          LOAD$V  000005          P.IOBL  1
74451

A.AL$I  175464          B.AHRD  175727          B.BL$I  176306          L.ADDR  000011          RESINT  1
77776

A.APRT  000014          B.AHUP  175607          B.BPRT  000044          L.BYTC  000007          RESTRT  1
74016

A.AWRD  175163          B.AHWT  176003          B.BWRD  176135          L.CKSM  000013          SPRIUS  1
74006

A.AWWT  175245          B.AH$I  176253          B.BWWT  176223          L.GBYT  174566          STACK = 0
00177

A.BHDF  175065          B.ALDF  175600          B.IRQD  176322          L.LDR0  174600          SWINT   1
77772

A.BHDN  175115          B.ALDN  175626          B.IRQG  176317          L.LDR1  174605          SWIRQ   1
74007

A.BHDR  175033          B.ALDR  175554          B.6821= 000210          L.LDR2  174612          SWIRQV  0
00002

A.BHRD  175275          B.ALRD  175745          CRAH  = 000000          L.LDR3  174651          VARSAV= 0
00054

A.BHUP  175103          B.ALUP  175614          CRAL  = 000001          L.LDR4  174704          $A.682= 0
00001

A.BHWT  175361          B.ALWT  176024          CRBH  = 000002          L.LDR5  174714          $B.682= 0
00001
```

```
A.BLDF  175074          B.APRT  000034          DDRAH = 000004          NMINT   177774          $LOADE= 0
00001

A.BLDN  175122          B.AWRD  175763          DDRAL = 000005          NMIRQ   174000          ...A  = 0
00100

A.BLDR  175050          B.AWWT  176045          DDRBH = 000006


. ABS.  177776    000   (RW,I,GBL,ABS,OVR)

        000000    001   (RW,I,LCL,REL,CON)

Errors detected:  0


*** Assembler statistics



Work  file  reads: 0

Work  file writes: 0

Size of work file: 14766 Words  ( 58 Pages)

Size of core pool: 18176 Words  ( 71 Pages)

Operating  system: RT-11


Elapsed time: 00:00:02.03

PARLEL,PARLEL=M6800,PARLEL,LOADER,6821A,6821B,END
```

```
        .TITLE  PORTS INITIALIZATION
        ;
        ;WRITTEN BY
        ;       ALAN R. BALDWIN
        ;       PHYSICS DEPARTMENT
        ;       KENT STATE UNIVERSITY
        ;       KENT, OHIO  44242
        ;       4-APRIL-1983

        .MCALL  .EXIT

        .PAGE
        .SBTTL  SERIAL PORT INITIALIZATION TABLES

SPORT1: .WORD   175400          ;BASE ADDRESS
        ; SET 7 DATA-BITS / 2 STOP-BITS / EVEN PARITY
        ; 19200 BAUD / MODEM STATUS DISABLED
        .WORD   1000,1600,2406,3000,2036
        .WORD   0
        ; LOAD CONTROL BYTE
        .WORD   0
        ; DISABLE CONTROL TRANSFER
        .WORD   36054,175416
        ; END OF SERIAL PORT 1 INITIALIZATION
        .WORD   0,0

SPORT2: .WORD   175420          ;BASE ADDRESS
        ; AUTO-BAUD RATE AND PARAMETERS ENABLED
        ; SET 7 DATA-BITS / 2 STOP-BITS / EVEN PARITY
        ; 9600 BAUD
        .WORD   1017,1417,2414,3000,2036
        .WORD   0
        ; LOAD CONTROL BYTE
        .WORD   0
        ; DISABLE CONTROL TRANSFER
        .WORD   36054,175436
        ; END OF SERIAL PORT 2 INITIALIZATION
        .WORD   0,0

SPORT3: .WORD   175440          ;BASE ADDRESS
        ; AUTO-BAUD RATE AND PARAMETERS ENABLED
        ; SET 7 DATA-BITS / 2 STOP-BITS / EVEN PARITY
        ; 9600 BAUD
        .WORD   1017,1417,2414,3000,2036
        .WORD   0
        ; LOAD CONTROL BYTE
        .WORD   0
        ; DISABLE CONTROL TRANSFER
        .WORD   36054,175456
        ; END OF SERIAL PORT 3 INITIALIZATION
        .WORD   0,0

SPORT4: .WORD   175460          ;BASE ADDRESS
        ; AUTO-BAUD RATE AND PARAMETERS ENABLED
        ; SET 7 DATA-BITS / 2 STOP-BITS / EVEN PARITY
        ; 9600 BAUD
        .WORD   1017,1417,2414,3000,2036
        .WORD   0
        ; LOAD CONTROL BYTE
        .WORD   0
        ; DISABLE CONTROL TRANSFER
        .WORD   36054,175476
        ; END OF SERIAL PORT 4 INITIALIZATION
        .WORD   0,0

        .PAGE
        .SBTTL  PARALLEL PORT INITIALIZATION TABLES

PPORT1: .WORD   175500          ;BASE ADDRESS
        ; USE INTERNAL SET UP
        .WORD   0
        ; DISABLE CONTROL INTERRUPTS
        .WORD   36054,175512,36054,175516
        ; END PARALLEL PORT 1 INITIALIZATION
        .WORD   0,0
```

```
PPORT2: .WORD   175520          ;BASE ADDRESS
        ; USE INTERNAL SET UP
        .WORD   0
        ; DISABLE CONTROL INTERRUPTS
        .WORD   36054,175532,36054,175536
        ; END PARALLEL PORT 2 INITIALIZATION
        .WORD   0,0

PPORT3: .WORD   175540          ;BASE ADDRESS
        ; USE INTERNAL SET UP
        .WORD   0
        ; DISABLE CONTROL INTERRUPTS
        .WORD   36054,175552,36054,175556
        ; END PARALLEL PORT 3 INITIALIZATION
        .WORD   0,0

PPORT4: .WORD   175560          ;BASE ADDRESS
        ; USE INTERNAL SET UP
        .WORD   0
        ; DISABLE CONTROL INTERRUPTS
        .WORD   36054,175572,36054,175576
        ; END PARALLEL PORT 4 INITIALIZATION
        .WORD   0,0


        .PAGE
        .SBTTL  INITIALIZATION CALLS

START:  MOV     #177777,175600  ;RESET ALL PORTS

        MOV     #8.,R0          ;WAIT LOOP
1$:     CLR     R1
2$:     SOB     R1,2$
        SOB     R0,1$

        JSR     R5,SERIAL       ;INIT SERIAL PORT #1
        SPORT1
        JSR     R5,SERIAL       ;INIT SERIAL PORT #2
        SPORT2
        JSR     R5,SERIAL       ;INIT SERIAL PORT #3
        SPORT3
        JSR     R5,SERIAL       ;INIT SERIAL PORT #4
        SPORT4
        JSR     R5,PARLEL       ;INIT PARALLEL PORT #1
        PPORT1
        JSR     R5,PARLEL       ;INIT PARALLEL PORT #2
        PPORT2
        JSR     R5,PARLEL       ;INIT PARALLEL PORT #3
        PPORT3
        JSR     R5,PARLEL       ;INIT PARALLEL PORT #4
        PPORT4
        .EXIT

        .PAGE
        .SBTTL  SERIAL PORT LOADER

SERIAL: MOV     (R5)+,R0        ;POINTER TO TABLE
        MOV     (R0)+,R2        ;BASE ADDRESS
        MOV     #177777,R1

        BIC     R1,0(R2)        ;CLEAR INTERRUPT ENABLES
        BIC     R1,4(R2)
        BIC     R1,10(R2)
        BIC     R1,14(R2)

        MOV     #34070,12(R2)   ;SET CONTROL - DISABLE TRANSFERS
        MOV     #34070,16(R2)

        BIC     R1,2(R2)        ;SET FOR INPUT
        BIS     R1,6(R2)        ;SET FOR OUTPUT

        MOV     #26054,12(R2)   ;SET CONTROL - ENABLE TRANSFERS
        MOV     #26054,16(R2)

1$:     TSTB    14(R2)          ;TEST TRANSFER CONTROL READY
        BPL     1$
```

```
        TSTB    4(R2)           ;TEST DATA PORT READY
        BPL     1$
        MOV     (R0)+,R1
        BEQ     2$              ;FINISHED ON ZERO
        MOV     R1,6(R2)        ;LOAD DATA
        BR      1$              ;AND LOOP

2$:     MOV     (R0)+,R1        ;GET BYTE CONTROL
        MOVB    R1,7(R2)        ;AND LOAD

3$:     MOV     (R0)+,R1        ;DATA
        MOV     (R0)+,R2        ;ADDRESS
        BEQ     4$              ;ADDRESS=0, THEN SECTION DONE
        MOV     R1,(R2)         ;LOAD DATA @R2
        BR      3$              ;AND LOOP

4$:     RTS     R5              ;FINISHED

        .PAGE
        .SBTTL  PARALLEL PORT INITIALIZATION

PARLEL: MOV     (R5)+,R0        ;POINTER TO TABLE
        MOV     (R0)+,R2        ;BASE ADDRESS
        MOV     #177777,R1

        BIC     R1,0(R2)        ;CLEAR INTERRUPT ENABLES
        BIC     R1,4(R2)
        BIC     R1,10(R2)
        BIC     R1,14(R2)

        MOV     #34070,12(R2)   ;SET CONTROL - DISABLE TRANSFERS
        MOV     #34070,16(R2)

        BIC     R1,2(R2)        ;SET FOR INPUT
        BIS     R1,6(R2)        ;SET FOR OUTPUT

        MOV     #26054,12(R2)   ;SET CONTROL - ENABLE TRANSFERS
        MOV     #26054,16(R2)

2$:     TSTB    14(R2)          ;TEST TRANSFER CONTROL READY
        BPL     2$
        MOV     (R0)+,R1
        BEQ     3$              ;END ON ZERO
        MOV     R1,6(R2)        ;LOAD DATA
        BR      2$              ;AND LOOP

3$:     MOV     (R0)+,R1        ;DATA
        MOV     (R0)+,R2        ;ADDRESS
        BEQ     4$              ;ADDRESS=0, THEN SECTION DONE
        MOV     R1,(R2)         ;LOAD DATA @R2
        BR      3$              ;AND LOOP

4$:     RTS     R5              ;FINISHED


        .END    START
```