# ACR SUPPLEMENT

to Introductory Manual for Technicians

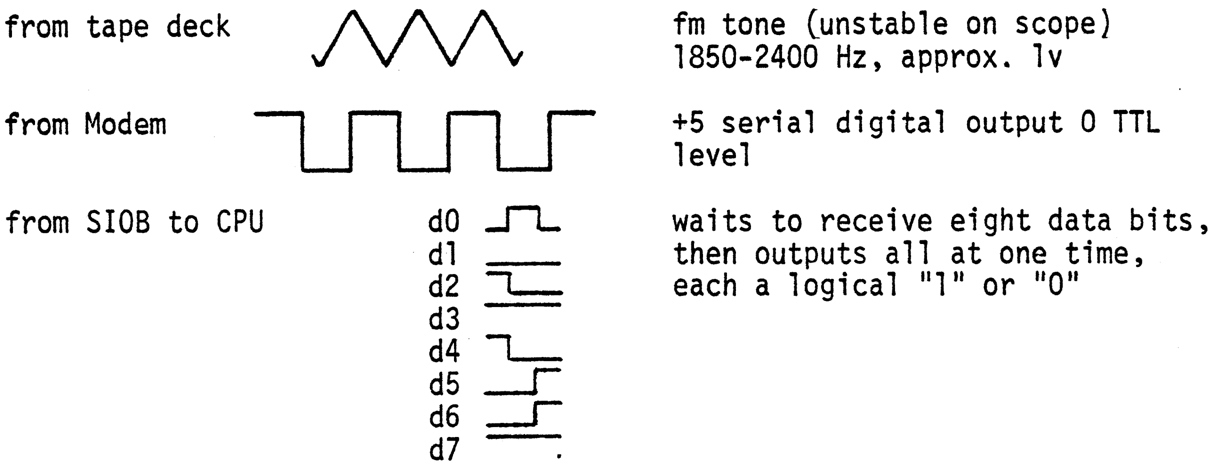
For In-house Use Only

mits

# ACR Supplement to Introductory Manual for Technicians

## Brief Theory of Operation

The 88-ACR consists of two boards which allow an inexpensive cassette recorder to be used as a mass memory storage device. The 88-ACR can read or store data on an audio cassette by recording different frequency tones for the two logic levels. A 2400 Hz tone represents a logical "1", and an 1850 Hz tone represents a logical "0". When the tape is played, it outputs a signal of changing frequency which is similar to frequency shift keying (FSK) transmission. However, the computer cannot use this data; it can only interpret TTL level parallel data. The ACR uses two separate boards to convert the information on the tape into the correct format. The Modem board converts the serial fm data to serial TTL level digital data. The SIOB board then converts it to parallel data.


from tape deck
fm tone (unstable on scope)
1850-2400 Hz, approx. 1v
from Modem
+5 serial digital output 0 TTL level
from SIOB to CPU
d0
d1
d2
d3
d4
d5
d6
d7
waits to receive eight data bits, then outputs all at one time, each a logical "1" or "0"

Each board will convert data in either direction.

Basically, the Modem board consists of two parts: the modulator section and the demodulator section. The modulator section takes serial digital data from the SIOB and modulates it into an fm audio signal that can be recorded onto a tape. The demodulator section inputs data from the tape deck, demodulates the fm signal into digital data, and presents it to the SIOB.

The demodulator section is represented on the top third of the schematic. Data from the tape deck enters from the top left at FSK Play Input. It is filtered through a band-pass and fed into an op-amp (IC A). The output of A is filtered again through another band pass and fed into another op-amp (IC B). The output of B is then fed into IC C, the actual demodulator chip. The output of the chip at pin eight uses a current sink logic. This means that when the output is a logical "0", the chip will ground the line and drop the voltage to zero. When the output is a logical "1", pin 8 will be high impedance, thereby allowing the power supply to pull the line up to +5v. R32 on the extreme right of the diagram is called a "pull-up" resistor for this reason. It is connected to Vcc. Data is jumpered from RS Play Serial Data on the modem to the RSI input on the SIOB board.

SIOB Operation

SIO stands for Serial Input Output. The B is our code and it means the board is TTL (transistor-transistor logic) compatible. The board is designed around a 2502 chip (labelled M on the schematic). This IC is a UART (Universal Asynchronous Receiver-Transmitter). It has the ability to receive serial data, reformat it and output it as parallel data on individual data lines or vice-versa. This IC is very flexible and has many inputs to tell it how it should format the data. It has to be told the number of data bits to receive, the number of stop bits, parity or no parity, etc. Parity is a system for checking the accuracy of transmitted data. (We do not use it here.) Stop bits are bits used to indicate the end of a transmitted byte. We are using one for the ACR. The number of data bits is eight.

The UART has four control inputs:

16 $\overline{\text{SWE}}$ - Status Word Enable--this allows the status of the UART to be outputted on the data lines to the CPU.

4 $\overline{\text{RDE}}$ - Received Data Enable--this tells the UART to output the received data to the CPU.

23 $\overline{\text{TDS}}$ - Transmit Data Strobe--this signal tells the UART to take data off the bus and transmit it serially.

18 $\overline{\text{RDAV}}$ - Reset Data Available--This resets the data available flip-flop while the CPU is receiving the data.

Logic gates J, G and S control the above commands. The CPU tells the UART if it wants to input or output data by the control signals at left center.

SINP - the CPU wants to input data

SOUT - the CPU wants to output data

$\overline{\text{PWR}}$ - data on the bus is valid, and should now be transmitted.

Every I/O board has two channels. The odd channel is used for data. The even channel is used to tell the CPU the existing conditions in the UART. Both channels use the same data lines. The ACR uses channels 6 and 7.

IC I is an eight-input NAND gate that enables the logic section when address 6 or 7 is on the bus. When strapping the address section, each address line is sent through an inverter when it is set low. When set high, it bypasses the inverter. Therefore, if A7 through A3 are inverted and A2 and A1 are direct when address 006 or 007 is on the bus, all inputs to NAND gate I will be high. Pin 8 will go low and partially enable J1 and J4. A0 is used to select the odd channel (data) or even channel (control), depending on whether it is high or low. The four commands to the UART can be produced by the logical gating of A0, SINP, SOUT and $\overline{\text{PWR}}$, as shown on the schematic. These inputs also control the enable/disable functions for the buffers on the DI (data into the CPU) lines.

The bottom third of the schematic deals with the interrupt capability of the board. This circuitry allows you to interrupt the CPU when data is inputted and/or outputted. These interrupts can be given a relative priority by the strapping at the bottom right of the schematic. The logic in the bottom left portion of the schematic can allow or disallow this capability by software control. This will be discussed in the VI/RTC (Vectored Interrupt/ Real Time Clock) section.

As mentioned before, the SIOB has a control channel that is used to tell the CPU what its current function is and to identify any transmission errors. Each of these indicators is sent over a specific data line when the control channel is called and when $\overline{\text{SWE}}$ (Status Word Enable) is low. The bit definition of these outputs is shown below.

| Data Bit | Logic Low Level | Logic High Level |
|---|---|---|
| 0 | Input device ready (Data is available for computer to input) | Not Ready |
| 1 | Not Used | Not Used |
| 2 | | Parity Error |
| 3 | | Framing Error (data word has no valid stop bit) |
| 4 | | Data Overflow (a new word of data has been received before the previous word was inputted to the accumulator) |
| 5 | Not Used | |
| 6 | Not Used | |
| 7 | Output Device Ready<br>(transmitter buffer is empty) interrupt to occur if interrupt is enabled | Not Ready |

These status bits are very useful for troubleshooting. They will be discussed later in that section.

Inputting Data to the SIOB

Serial data from the Modem appears at the RS1 input (pin 20 of the UART). The input at pin 20 is normally high. When it goes low, a counter (which counts from 1 to 16) is started. This is provided by a clock input at pins 17 and 40. This clock is a negative pulse occurring at 16 times the baud rate. If the input at 20 is still low after eight clock periods (halfway through the incoming start bit), the clock will interpret it as a valid start bit. From then on, each bit is sampled on the eighth clock pulse and loaded into a shift register. After eight bits have been received, it looks for a parity bit, then for a stop bit. If the stop bit is not present, it sets a framing error flag. When the register is full, the data is sent to an output holding register. The Data Available flag goes high, telling the CPU it is ready to send data. The CPU then issues a SINP signal and calls the data channel. This sends $\overline{\text{RDE}}$ and $\overline{\text{RDAV}}$ (pins 4 and 18) low, and the UART outputs the data to the CPU.

Outputting Data Through the SIOB

When the computer is turned on, the $\overline{\text{POC}}$ clears the registers through pin 21 (master reset) and puts the UART into an idle state. When the UART is ready to input a byte for transmission, it will set pin 22 TBMT (transmit buffer empty) high. When the CPU sees this, it will output data onto the DO lines. The CPU sends out a SOUT and $\overline{\text{PWR}}$ signal and calls the data channel. These signals are gated to become a negative going pulse to pin 23 ($\overline{\text{TDS}}$, Transmit Data Strobe). On the leading edge of this pulse the UART will input data from the DO lines to a holding register. On the trailing edge of the pulse the data is moved to the transmitter register where the start and stop bits are added and transmission is started. The data is outputted as serial TTL level data from pin 25 (TSO, Transmitter Serial output). This output appears at pin 5 of the molex connector. From there it is jumpered to the XS Record Serial Data input to the Modem.

## Recording Data Through the Modem

The modulator section of the modem is shown on the bottom third of the schematic. The input labelled $\overline{FT}$ is a 2MHz clock input. The modem divides this frequency down to 2400 Hz to indicate a logical "1" or to 1850 Hz to indicate a logical "0". The 2MHz signal is presented to the clock inputs of ICs J, K and E. Note that J3 and J4 are the jumper connections that have to be hooked together. Data from the UART appears at XS on the extreme right. The two NOR gates labelled G are used as inverters. The first gate supplies $\overline{\text{data}}$; the second supplies data. ICs J and K are synchronous 4-bit counters. Since the carry output of J is connected to the enable input of K, they can be thought of as eight flip-flops in series. Since each flip-flop will divide the signal by 2, eight in series will divide the signal by 256. IC E is strapped to divide by eight: 2MHz/ 8 = 250KHz. To arrive at 2400 Hz and 1850 Hz the signal must be divided by 104 and 135 at ICs J and K. Since the counters can only divide by 256, they can be started at a count other than 0. They can be wired to load the flip-flop with any starting count when they get a load pulse. Since their highest count is 255, 104 and 135 are subtracted from 255 which leaves 151 and 120 for the start count. Then, depending on whether they get a logical "1" or "0", they will give the proper division of the 2MHz signal. The carry output is sent to IC E and divided by eight. Output data is shown on the right side of the schematic. A TTL modulated square wave is at H8. Our recorder cannot accept this because it wants to see a MIC level signal in the audio range. A square wave contains harmonics in the megahertz band.

R50 and R51 act as a voltage divider to reduce the output to a few hundred millivolts. R51 and C22 integrate the signal into a sawtooth. It is then outputted to the tape deck MIC input. (Note that R50 should be 22K rather than 220K.)

## Alignment of the Modem

Every cassette player and tape has slightly different speed and head specifications. Therefore, the ACR must be realigned whenever the deck or tape is changed. One method is shown below.

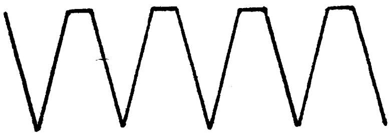
1. Check strapping as shown at the back of this section.
2. Plug ACR into chassis on top of an extender card (SIOB side to your right).
3. Insert mini-plug into earphone (plug-in) jack and the other end into the ACR cable going to pin 2.
4. Plug ACR cable into the ACR board. Pin 4 is ground.
5. Insert test side of 3.2 8K BASIC tape into player and rewind.
6. Turn on computer. Reset and deposit the following program.

   333

   007 (window program which displays input data on the data lights)

   Examine zero, then single step twice.
7. Attach the clip-on probe of the scope to pin 10 of the ACR Molex connector (pin furthest from you). This is the output of the demodulator.
8. Set scope for 2v/cm and 1 msec/cm.
9. Play tape with volume set for 8 or 9.
10. Adjust R29 (port furthest from you) for an equal duty cycle. Each alternation should be about 3.2 msec. Data lights should display 1 25's (01 010 101).
11. Put scope probe into the test point hole (on Modem side of board near R29) labelled TP. Set scope for 5v/cm and 2 msec/cm.

12. Adjust the next port towards you for maximum gain and flat positive peaks.



13. Adjust the port closest to you to do the same for the negative peaks.
14. Advance the tape to the second test pattern (175's or 256's) and check to see that you get a stable display on the data lights. This completes the alignment procedure.

How to Load BASIC

1. Normally you will use an 8K 3.2 BASIC tape to test out an ACR.
2. Align ACR if necessary.
3. Install at least 8K of memory starting at 0 and strapped continuously.
4. Set up your terminal with the proper I/O card.
5. Deposit ACR bootstrap which is listed at the back of this manual. If you have a multiboot PROM card, merely address the starting location of the ACR boot (see table at the back of manual).
6. Install tape (BASIC side up) and rewind to beginning.
7. If you have loaded the boot by hand, examine address 000012 and single step twice. This is the window program and will allow you to see the 256's written on tape before BASIC starts. If you are using a PROM bootstrap, find 333, 007 on the PROM and single step twice from 333.
8. Start tape.
9. After a few seconds, the data lights should show a stable display of 256 (10 101 110).
10. Quickly examine address 0.
11. Lift switch A15. If using a 2SIO for an I/O board, also lift A11.
12. Hit RUN.

13. After about 20 seconds the address lights will "jump". Address lights A3, A4 and A6 should go out.

14. After a few minutes, the terminal should say "memory size?"

15. Hit return key on terminal.

16. It will then ask you if you want certain functions "Want Sin, Cos, Atan?"

17. Answer Y for yes; N for no.

18. When it prints "OK", you are ready to write a test program.

    If the terminal starts printing out C's (ccccccc, etc.) while it is loading, a "checksum error" is indicated. This is an error in transmission of data, possibly a bad tape. If it prints out M's (mmmm, etc.), a memory error is indicated. One of your memory boards is probably bad.

<u>Saving Programs through the ACR</u>

The ACR can be used to store programs by utilizing the CSAVE command. When checking out any ACR, the following should be done several times.

Procedure:

1. Write a simple test program.

   Example:
   ```
   10 Print "ABCDEFGHIJK . . ."
   20 GOTO 10
   ```

2. Run program to see if it works.

3. "Control C" to stop program.

4. Type C SAVE A (any label can be used).

5. Connect cable from the record out jack of ACR cable to "MIC" or "record in" jack of the tape deck.

6. Install blank cassette in ACR. If there are two holes in the back edge of the cassette, they should be covered with tape. (This system prevents accidental erasures of BASIC).

7. Rewind tape.
8. Press play and record. Let it run past leader or about 15 seconds.
9. Press return key on terminal.
10. When program has been stored, it will print "OK".
11. Reconnect cable so that earphone-out on deck is connected to play-in on ACR.
12. Type NEW on terminal and hit return. This will erase the program from memory.
13. Rewind tape.
14. Type CLOAD A (or whatever you have called your program) and hit RETURN.
15. Play tape.
16. The terminal should respond "OK".
17. Type RUN and hit return.
18. Your program should run on the terminal.
19. Repeat several times.

If you type CLOAD (A), hit return and nothing happens; it either did not save or load properly. Your computer will continue to wait for A to load indefinitely. To bring the computer back to reality, stop it (examine ØØØ,ØØØ) and hit run. Now try to find something you did wrong and try it again from the top.

ACR Troubleshooting Hints

1. 3.2 BASIC incorporates a larger frequency swing in the data stored on the cassettes. All the older ACR's were modified to correctly interpret these cassettes. A list of these modifications is located in the back of this manual. Check to see that these modifications are correct on all boards. If the modifications have not been made, find out which revision of software the customer is using. A revised board will usually work with 3.1 BASIC, however, an old board will not work at

all with 3.2 BASIC.

2. If you have trouble getting a stable 125 on the data lights while running a test tape, check out R29 which may have collected dirt inside. Ohm it out over the entire range to see if it has a smooth response. Check the 5 μf capacitor for leakage.
3. If you are not receiving any data on the data lights after entering a window program, check the following:
   a. regulated voltages
   b. Baud rate--Look at pin 40 of the UART for a negative going 2.3-3.0 microsecond pulse every 208 microseconds.
   c. Play 125's from the test tape and check pin 8 of the XR-210 for a good square wave output (-12v to +5v). If there is no output, go to the test point labelled TP and look for a 14v p-p fm signal. By signal tracing around these points, you should be able to isolate a failure. If the problem is centered around the SR-210, refer to the table of waveforms in the ACR manual.
   d. If you have good output from the XR-210 and good continuity to pin 20 of the UART (IC M on the SIOB side), refer to the SIO section of this manual.
   e. If you have good input to the XR-210 and bad output, change the chip. You may have to try a few chips to find one that works reliably with your board.
4. If you cannot save programs:
   a. On the bottom of the modem there are a number of holes for interconnection wires to the SIOB. The second hole from the right (labelled GND) and the third from the right should both have wires going to the ground connection (pin 4) of the 10-pin molex plug.

Check this connection.

b. Deposit and run the following program:

```
333   input data to accumulator
377   from sense switches
323   output accumulator
007   to channel 7 (ACR data channel)
303   jump
000   to
000   address zero
```

This program continually outputs data from sense switches A15-A8 to the modulator section of the modem. With this program running, you can signal trace the circuit.

1. Check IC H, pin 8, for a square wave output. You should be able to modulate the output by changing the sense switch patterns.
2. Check point RO if there is an output at H8. You should see a modulated triangular wave. This is a very weak signal, approximately 100 mv. Note that R50 should be 22K ohm, and not 220K ohm as shown on the schematic. R50 is a voltage divider and integrating circuit. You should be feeding into a high impedance MIC input which demands a low level input.
3. If there is no output, check for a 2MHz input to J2, K2 and E14. If there is no 2MHz input, check for a low at G11. Do you have continuity from G12 to SØ on the SIOB side? Are ICs J, K and E dividing the signal?

Proper strapping of the SIOB Card:

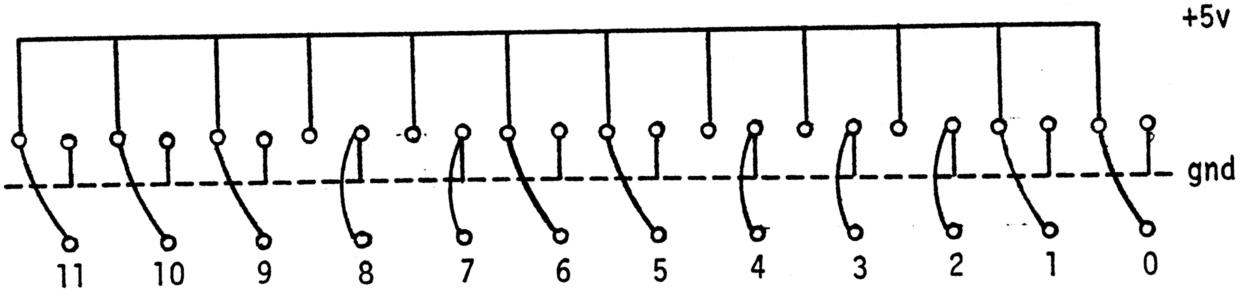
1. The ACR should be at channels 6 and 7. Therefore, look at the address straps at left center. I1 should be connected to A1, I2 should be at A2, and all others should be strapped low.

$A2 = 2^2 = 4$

$A1 = 2^1 = \underline{2}$

$\phantom{A1 = 2^1 = }6$

2. The ACR operates at 300 baud. Notice the straps at the bottom right corner of the board. The diagram below shows the correct connections for a Rev. 1 SIOB board.



3. The following jumpers should be installed on the board:

   -v to -v

   0 to 0

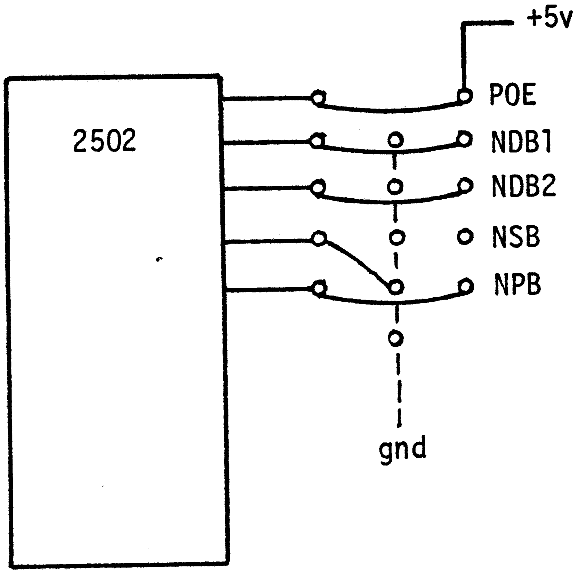
   I to I

   C to C

   PC to PC

   The interrupt jumper (INT) should not be connected.

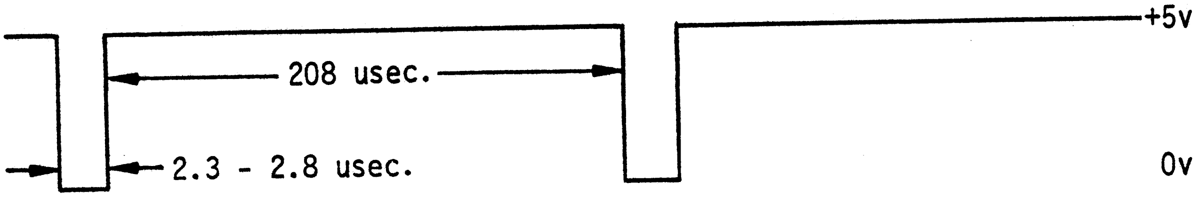
4. On the right side of IC M five jumpers should be connected as shown below.



## SIOB Troubleshooting

Routine checks:

1. Make sure all strapping is as shown on diagrams.
2. Is the board a Rev. 1? If not, modify according to the following sheet.
3. Check to see if R4 is a 7.5K resistor.
4. Make sure interrupt jumper is not installed.
5. Check +5 and -12 supplies.
6. Check the clock input at M-40.



If you cannot input data, deposit the bootstrap and play 125's from the test side of the tape.

1. Do you set an input at pin 20 of UART?
2. Is D8 pulsing low to enable data in tri-state buffers?
3. Is pin 4 of UART ($\overline{\text{RDS}}$) pulsing low to enable data transfer?
4. Is pin 18 pulsing low to enable status transfer to the CPU?
5. Is pin 19 (DA) pulsing high to tell the CPU that data is available?
6. Are pins 15, 14 and 13 (OVR, FE, PE) staying low to indicate that there are no transmission errors?

If just one bit is not being transmitted, you may not notice it playing 125's. Try playing 256's also. If you see a bit missing on the data window, check the corresponding buffer and gate, if any.

If you cannot transmit data, load the output from the sense switches program listed earlier. Single step through the program. Check for the following:

1. To output pin 23 ($\overline{\text{TDS}}$) must go low
2. Pin 22 (TBMT) should go high after an output sequence

3. When running the program, you should see continuous data-out at pin 25 (TSO)

If you are not getting proper command signals to the UART, it is a simple matter to trace back through the logic gates to find the cause.

There is an esoteric "bug" in the timing you should know about. An instruction address after an input instruction pair must not be the address of any I/O port being used in a system. The following program cannot occur:

| address | data byte |
|---|---|
| 000 | xxy (don't cares) |
| 001 | xxx |
| 002 | xxy |
| 003 | xxx |
| 004 | xxx |
| 005 | 333 |
| 006 | 006 or 007 |
| 007 | xxx |

When the CPU is trying to input data and the address changes to 007, it will reset the UART in the middle of a byte and cause that byte to get lost.

## ACR Bootstrap

| Address | Data |
|---|---|
| 000 | 041 |
| 1 | 256 for 3.2 BASIC; 175 for 3.1 |
| 2 | 037 for 8K BASIC; 017 for 4K; 057 for Extended |
| 3 | 061 |
| 4 | 022 |
| 5 | 000 |
| 6 | 333 |
| 7 | 006 |
| 010 | 017 |
| 1 | 330 |
| 2 | 333 |
| 3 | 007 |
| 4 | 275 |
| 5 | 310 |
| 6 | 055 |
| 7 | 167 |
| 020 | 300 |
| 1 | 351 |
| 2 | 003 |
| 3 | 000 |

## Window Program

| | |
|---|---|
| 000 | 333 |
| 001 | 006 for control channel; 007 for data channel |

examine 000

single step twice

## Rev. 0, SIOB Modifications

See sheet labelled SIOC Rev. 0 Modification. For the SIOB only the bottom paragraph is applicable.

Check your board to see that pin 6 of IC O is connected to pin 11 of ICs P, Q and R. If it is not, install jumpers.

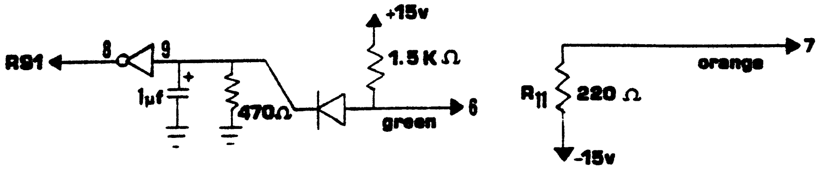
# SIOC REV 0 MOD

In house, the SIOC Rev 0 boards underwent two significant changes. One change, the more apparent of the two, concerns the location of the interface connections between the TTY and the board as well as the nature of the interface electronically. First the position of the pins between the computer and the TTY will be redefined to be more compatible to the Rev 1 cabling. The color code is arbitrary but provided in order that the client is aware of MITS standards.

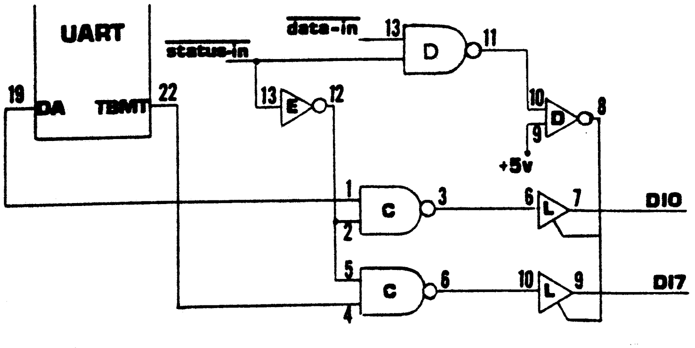
| Molex Conn on Board | Color | Male & Female 25-Pin Conn. | Term Lugs in TTY(ASR-33) |
|---|---|---|---|
| 4 (GnD) | Black | 2 | 6 |
| 5 (Tran) | Red | 3 | 7 |
| 6 (Rcv) | Green | 4 | 3 |
| 7 (Rcv) | Orange | 5 | 4 |

It might also be noted that the SIOC board is capable of only functioning with full duplex transmission. That is, a White-Blue lead and a Brown-Yellow lead should be moved from term lugs 4 and 3 respectively to lug 5. Unless some major change has been made in the interface electronics the SIOC board will have difficulty in functioning with a 60mA current loop in the TTY. The desired 20MA loop can be obtained by 1) moving a Purple lead from lug 8 to lug 9. 2) Moving the left-most terminal on R1 (a 4 connection resistor located about 8" back from the line/local switch on the base plate) to the left-most terminal connection of R1 (from the 3rd connection to the 4th connection).

The Rev 0 modified electronic interface and the Rev 1 interface appear below:



The second modification of the Rev 0 board (called hardware interrupt) is in the status provided to the data bus by the UART. The new board will pull DI0 low on the DA high condition (input status) and will pull DI7 low on favorable output status. The logic follows.



# New Audio Modulation Method for ACR

As evidence that we at MITS listen to our customers, we are improving the 88-ACR read and write performance. The changes described below will allow the 88-ACR to accept 2.75 times wider speed variation when demodulating tapes written with the new method. Also, demodulation (reading) of tapes written by the old method will be the same as before.

I Purpose: Make reading and writing of data on audio tapes less susceptible to errors due to speed variations, and to make adjustment of R29 (phase locked loop center frequency adjust) less critical.

II Method: Change modulator frequencies from 2225Hz/2025Hz-(200 Hz difference) to 2400Hz/1850Hz-(550 Hz difference). This change keeps the center frequency at 2125Hz, allowing the 88-ACR to demodulate (read) either type of modulation.

III Modifications to 88-ACR Modem Boards in the field:

A) Modulator - Change jumpers as follows:

1. Remove jumpers #1 & 2.
2. Connect pins 3, 4, and 5 of IC "J" together.
3. Change jumper #3 from 3B to 2A.
4. Change jumper #4 from 4B to 4A.
5. Disconnect pins 5 and 6 of IC "K" from ground (unsolder and bend out of board).
6. Connect pins 4 and 5 of IC "K" together.
7. Change jumper #5 from 5B to 2A.
8. Connect pin 6 of IC "K" to point 5A.
9. Change jumper #7 from 7B to 7A.

NOTE: The "B" row of jumper points is closest to edge of Modem Board, the "A" row of jumper points is closest to the row of numbered jumper wires (see schematic diagram in manual).

This changes the modulation frequencies to:

LOGIC 1 = 2404 Hz ± 1 Hz
LOGIC Ø = 1852 Hz ± 1 Hz

(measured at IC "H"-8)

B) Demodulator: Change R28 to 3.3K ohms, or parallel a 5.6K ohm resistor with the existing 8.2K ohm resistor.

This change increases the lock range of the phase locked loop (IC "C") for the wider frequency spread of the new modulation method. It does not affect demodulation of tapes previously recorded with the old frequencies (2225/2025 Hz).

This change allows tape speed variations between writing and reading of over 3% without readjustment of R29 (if demodulating tapes written with the new method).

IV Other Circuitry Changes Recommended for the 88-ACR.

A) Change C18 (was 5 µf electrolytic) to a 1 µf mylar or non-polarity sensitive capacitor. This prevents breakdown of C18 when reverse biased (no carrier).

B) Use the old C18 (5 µf electrolytic) to add a 5 µf capacitor: + end to IC "C" pin 9 end of R30, -- end to -12 volts. This helps stabilize adjustment of R29.

C) Change R32 to 8.2K (use old R38) and change Z1 (12 volt zener) to a 3.3K resistor. This allows the P. L. L. output (IC "C", pin 8) to pull down point "RS" to a valid logic Ø even if the system negative voltage supply is low.

D) Remove diode D4. This allows reading and writing of tapes simultaneously.

E) Optional - For indication of the carrier (2K Hz tones) a L. E. D. may be wired to points "A" and "K" on the Modem Board. Remove the jumper wire from "A" to "K", and connect the LED anode to "A", the cathode to "K". When the carrier is being received, the LED forward current is about 10MA. Use a red LED only--1.7 volts forward drop.

V Effective Date of Change

A) All COMTER II units, all assembled 88-ACR's and all repaired 88-ACR's shipped from MITS after March 1, 1976, contain the modification described above.

B) All 88-ACR kits shipped after March 15, 1976, contain the modification described above.

C) All ALTAIR BASIC and Package I cassette tapes will be made with the new modulation technique starting April 5, 1976.

VI Converting Old Tapes to the New Modulation Method:
Although it is not necessary, you may wish to convert existing tapes to the new form. To do this, you need two tape recorders and:

A) Modify your 88-ACR board as indicated, including Step IV-D.

B) Identify the slower of the two tape recorders, and use it for playback of your existing tape during transfer. The play machine should be slightly slow to prevent the inputting of data faster than it can be outputted. Connect the slower machine to the "PLAY IN" circuit, and adjust R29 for the proper pattern.

C) Connect the other tape recorder to the "RECORD OUT" circuit and use it for recording the new tape.

D) Use the following program to transfer data:

| Address | Octal Code | Mnemonic |
|---|---|---|
| 000,000 | 333 | IN |
| 1 | 006 | |
| 2 | 017 | RRC |
| 3 | 332 | JC |
| 4 | 000 | |
| 5 | 000 | |
| 6 | 333 | IN |
| 7 | 007 | |
| 10 | 323 | OUT |
| 11 | 007 | |
| 12 | 303 | JMP |
| 13 | 000 | |
| 14 | 000 | |

-continued

# 88 ACR USER NOTES

by Tom Durston

Here are more helpful hints for those Altair users having difficulties loading MITS software on cassette tape.

1. Try using lower volume settings on your tape recorder during playback. Sometimes noise generated in recorders playing at maximum volume can cause errors in data. We have found that in most recorders volume settings as low as 1/3 of maximum are satisfactory.

2. If you have trouble obtaining a proper "JUMP" of address lights when beginning a bootstrap load, or you don't want to wait the 15 seconds between starting the tape playing and depressing the run switch, try this 9 step program in addition to the bootstrap loader.

This program tests for the leader bit pattern that is recorded before the checksum loader at the beginning of MITS software. The program will loop at the high addresses until the leader byte is found (10-15 seconds after start of tape) and then jump to the bootstrap loader at ØØØ,ØØØ. Approximately 10 seconds later the address lights change again, indicating proper loading of the software (for version 3.2 A3, A4, & A6 off).

1) Deposit bootstrap loader.
2) Deposit leader detector.
3) Examine 001,000.
4) Start tape and depress "RUN" on Altair.
5) 25 seconds later, Altair should jump , indicating proper loading of data.

Bootstrap Leader Detector

| TAG | MNEMONIC | ADDRESS | OCTAL CODE | EXPLANATION |
|---|---|---|---|---|
| START | IN | 001,000 | 333 | input data |
| | | 1 | 007 | from ACR |
| | CPI | 2 | 376 | compare data byte to |
| | | 3 | 256 | leader byte for version 3.2<br>(175 for 3.1) (same as bootstrap<br>location 1) |
| | JNZ | 4 | 302 | jump if data ≠ 256 (or 175) |
| | | 5 | 000 | to "start" |
| | | 6 | 001 | |
| | JMP | 7 | 303 | jump to bootstrap loader |
| | | 001,010 | 000 | if data = 256 (or 175) |
| | | 11 | 000 | |