

Double The Cassette Baud Rate Of Your OSI Superboard II/IP

Mr. James L. Mason
Jacobus, Pennsylvania

I was thrilled when I received my Superboard II. It was my first micro, but being experienced in BASIC Programming (using a phone-linked ASR 33 with GE time-sharing) I had several programs which I was anxious to try. The Superboard performed perfectly. It did everything OSI said it would, however, 2 disadvantages of Superboard soon made themselves apparent. The 25 x 25 character video format was not the easiest to read. Secondly, the baud rate at which programs are saved and loaded from cassette seemed painfully slow. Having a good working background in digital electronics, I thought it might be possible to improve upon these two features. Upon close examination I found the video hardware was too intimate with the software in ROM. Fortunately, modifying the cassette port circuitry was a piece of cake and I was able to cut load and save time by half.

The cassette port utilizes a 6850 programmable Asynchronous Communications Interface Adapter (see figure 1). When using this chip, the communications rate is determined by two things, the frequency of the clock which is applied to the TXCLK and RXCLK pins of the ACIA and the control word which is written into the ACIA's control register. I hypothesized that by doubling the clock frequency I could double the baud rate.

ON the Superboard, a crystal oscillator generates the base timing signal by which the entire board is controlled. This signal drives a synchronous divider chain (see figure 2). The timing signal destined for the ACIA comes off the $\div 32$ tap of this chain. The signal is finally divided by a $\div 24$ circuit composed of U57, U58, and U63. The resultant frequency of 5120 Hz is applied to the TXCLK input of the ACIA. The ACIA must be programmed to utilize a clock frequency either 64, 16, or 1 times the baud rate. The 320 baud rate is realized by program-

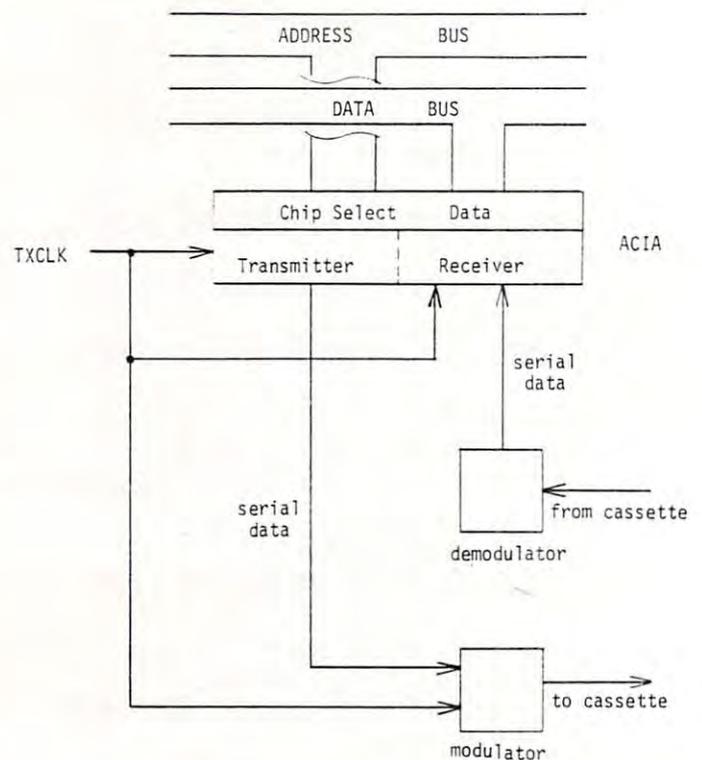


Figure 1. Cassette Port Block Diagram

ming the ACIA for a 16x clock rate.

To obtain a clock rate double of that which is used, I chose to sever the connection between U57 pin 2 and U59 pin 14 (see figure 3), then connect U57 pin 2 to U30 pin 11. I used a switch to maintain compatibility with my old 320 baud tapes.

As far as the ACIA was concerned, the modification was done. However, there is one more block between the ACIA and the cassette machine, namely the Modulator/Demodulator.

The modulator encodes the data in the form of tones. These tones are derived from the TXCLK (see Figure 4A). Since our new TXCLK is twice as fast, our tones will now be 2 times their original frequency. This poses no problem as far as modulation is concerned. It does, however, make a difference on the return trip. U69 determines what will be demodulated as a high or low tone (see figure 4B). A tone coming in will trigger the 74123 one-shot by its rising edge (see figure 5). R57 is adjusted so that U69 will remain triggered until after the falling edge of the high frequency tone but not until the falling edge of a low frequency tone. The falling edge will clock the D flop U63 and propagate the state of U69. Because we now have shorter pulse widths, R57 must be adjusted to allow U69 to time-out during our new low tones. This was very simple to do. I simply saved a program using the new faster baud rate and attempted to load it back. While the program was trying to load, I adjusted R57 while watching the video monitor. I knew I had R57 adjusted properly when the program began appearing on the display, line by line. I experimented with R57 to find

the points where data started to be garbled. The margin was surprisingly wide. Luckily, no software patch had to be made anywhere.

If you use a switch in your mod, remember you will have to readjust R57 each time you change baud rates. I see no reason why a DPDT switch couldn't be used to switch in a different resistor value for R57 along with switching the clock rate.

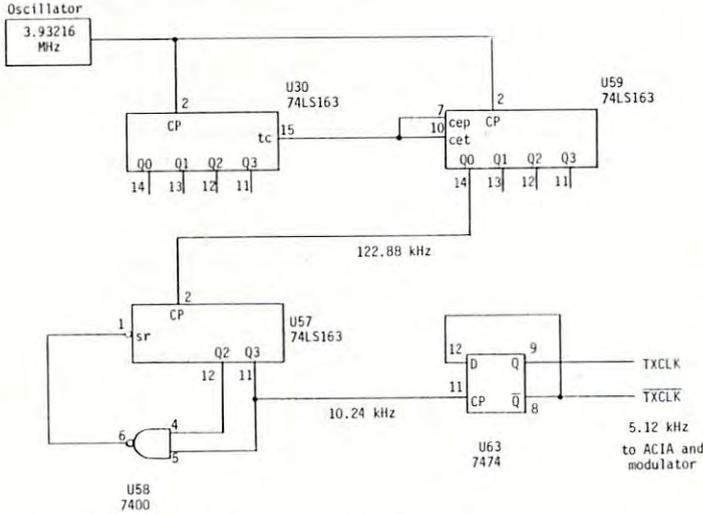


Figure 2. TXCLK Generation

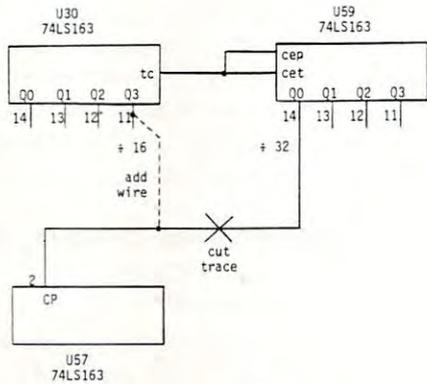


Figure 3. Installing The Modification

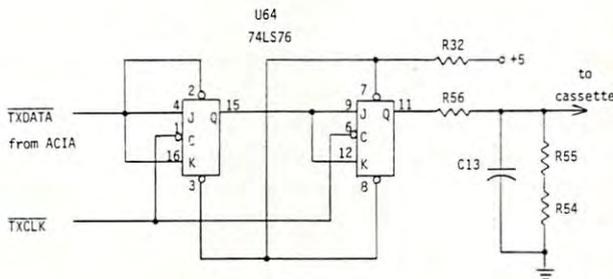


Figure 4A. Modulator

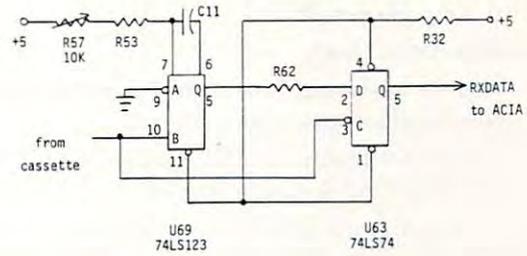


Figure 4B. Demodulator

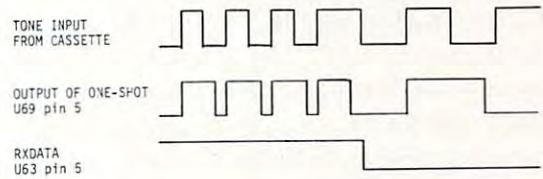


Figure 5. Demodulator Timing

Review

HEXDOS 2.3: A Disk Operating System For The OSI C1P or Superboard II

Ronald C. Whitaker
Salt Lake City, Utah

The day I received my OSI disk drive and 610 Expander board, I hooked them to my C1P and my homebuilt power supply, turned them on, and pushed "D" to boot up the disk. OSI's Pico-Dos came with the drive and expander board and booted up OK but would only allow me to load eight programs of up to 8K each. This was faster than cassette to be sure, but definitely lacking the features I wanted in a disk operating system. I lacked funds for OSI's OS65D and the additional 4K of memory it required to run on my system. The future looked dismal, indeed!

The day was saved by a single stroke of good fortune. Several months earlier a local dealer had loaned me a catalog from "The 6502 Program Exchange". While oriented mostly toward Apple and single board systems, they did have a few OSI compatible programs listed. One of these was HEXDOS 2.3 for the C1P and Superboard II by Steve Hendrix. The features promised by the catalog sounded too good to be true! These included:

1. An operating system and directory which occupied only the first two tracks on the disk