The Apple II Cassette Interface (2 of 2)

```
Revised:        11/20/84
Security:       Everyone
```

The Apple II Cassette Interface (2 of 2)

===

For reading data, the cassette recorder uses a more complicated input circuit consisting of a 741 operational amplifier configured as a zero crossing detector. Zero crossing detection means that whenever the voltage at the input jack goes from positive to negative (or negative to positive) the output of the amplifier switches from a 1 to a 0 (or 0 to 1). The detector is accessed by any read to address $C060. The sign bit (most significant bit) of the byte read reflects the detector status. The read routines continually EXCLUSIVE ORs this bit with the value most recently read to detect a change in state. The amount of time required to change state indicates the incoming frequency which then is used to determine if a one or a zero has been received. After detecting the first zero crossing at the start of the header, the read routine uses HEADR to generate a 3.5 delay, and then the read routine waits for the sync bit. After HEADR generates the synchronous bit, the read routine reads the data and puts it in the specified memory range.

In using the cassette interface to either read or write, all you need do is specify an address range and execute the read or write subroutine. The address range is stored in four bytes, two for the first address to be saved and two for the last to be saved. In both cases the least significant byte is first.

Commanding the cassette interface:

1. from the monitor:
   If the start is $800 and the end is $9FF, then
   800.9FFW will write the data to the cassette and
   800.9FFR will retrieve it.

2. from machine language:
   Again, if the start is $800 and the end is $9FF then store the address
   range,

   LDA #$00
   STA $3C starting address low
   LDA #$08
   STA $3D starting address high

```
    LDA #$FF
    STA $3E ending address low
    LDA #$09
    STA $3F ending address high
    JSR $FEDC write to block to tape
```

The JSR $FEDC will write to the cassette; JSR $FEFD will read from the cassette.

3. from BASIC:
   First set up the address range. If S = the start and E = the end then from integer BASIC,

```
    POKE 60,S MOD 256
    POKE 61,S / 256
    POKE 62,E MOD 256
    POKE 63,E / 256
```

4. from APPLESOFT,

```
    POKE 60,S - INT(S / 256) * 256
    POKE 61,S / 256
    POKE 62,E - INT(E / 256) * 256
    POKE 63,E / 256
```

   Then, to write out to cassette, use CALL -307; to read in from the cassette, use CALL -259.

Apple Tech Notes

Tech Info Library Article Number:496