The 25¢ Apple II Real Time Clock

Erann Gat
Oak Ridge, Tennessee

It is interesting to count the number of features of the Apple II which traditionally require boards full of parts to implement, but are done with only one or two inexpensive chips. For instance, the analog to digital conversion for the game paddles would normally cost at least $25, but is done on the Apple with a single inexpensive timer chip. The refresh for the dynamic memory requires no extra parts at all as this is done by the video circuitry.

This philosophy of doing things the easy way makes one wonder at the prices that are being charged for some of the peripheral boards for the Apple, particularly real time clocks. A search for an easier (and hopefully cheaper) way yielded a clock with good accuracy and any feature found on the more expensive boards, including many extra fringe benefits, with a total cost of between 3 to 25 cents depending on how sophisticated you want it to be.

All About Interrupts

Interrupts are something almost every computer hobbyist has heard of, but most of the information about them is rather cryptic. This section will attempt (note that verb) to clarify how interrupts work because they form the basis of the 25 cent clock.

Here is how an interrupt works: on the 6502 microprocessor there are two pins called IRQ and NMI. IRQ stands for Interrupt ReQuest and NMI stands for Non Maskable Interrupt. When either one of these pins is grounded, the processor finishes the machine language instruction it is currently working on, saves the program counter and processor status register onto the stack, (if you don't know what that means it isn't important) and jumps to a program somewhere in memory called an interrupt handling routine or interrupt handler. It then executes the interrupt handler until it encounters a RTI (ReTurn from Interrupt) instruction. It then restores the status register and program counter to their original values and continues executing the main program at the point where the interrupt occurred.

The main program is not affected by an interrupt except that some time is lost during the interrupt and the main program slows down. How much it slows depends on the length of the interrupt handler.

Now suppose that the interrupt handler was a routine that incremented a memory location and returned. This would then be an interrupt counter; i.e. every time an interrupt occurs, the counter is incremented. Now suppose that a pulse was applied to the interrupt line exactly once each second. Voila! A real time clock that tells time in seconds. This is the idea behind the 25 cent clock.

More About Interrupts

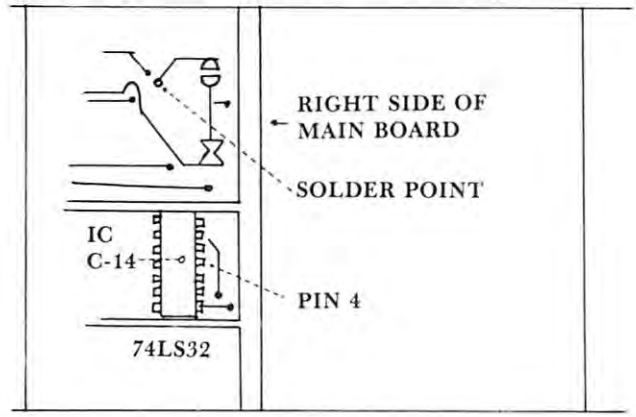
Up until now the 25 cent clock has been discussed in generalities and theories. This section discusses the actual implementation.

First some more facts about interrupts on the 6502: There are two main differences between the IRQ and NMI interrupts. In the 6502 status register there is a flag called interrupt enable. This flag can in effect turn off the IRQ line. If the enable flag is not set, the 6502 will deny Interrupt ReQuests. It will ignore them as if they were not there. On the other hand, NMI cannot be turned off. When a Non Maskable Interrupt occurs, the processor will always act on it and jump to the interrupt handler.

The second difference is that NMI and IRQ have their interrupt handlers at different places in memory. IRQ has another difference in that its interrupt handler is the same routine which handles the BRK instruction. BRK in effect generates a IRQ signal. There is a way to tell IRQ's from BRK's (in fact the Apple monitor does this for you) but this takes up quite a bit of time as well as creating other complications. NMI therefore is more suitable than IRQ for the clock. However, there is no law that says IRQ can't be used.

Next, a signal of known frequency must be found. A time base generator can be used, but at several dollars a piece it would be difficult to stay within the 25¢ budget. An ideal signal can be found in the video circuitry. This signal is the 60 Hz (meaning 60 times each second) pulse which generates the vertical retrace. This signal can be tapped at two locations shown in figure 1. The physical details are discussed in the next section.

FIGURE 1

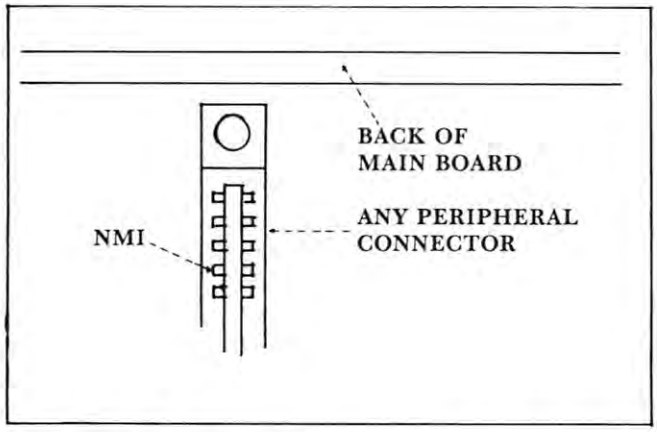


www.commodore.ca

The Three Cent Clock

Implementing the clock in its simplest form involves simply connecting the NMI line to a signal source. On the Apple, the NMI line can be accessed from any of the peripheral slots on the rear of the board. The location of the NMI line is shown in figure 2. The connection can be made using a prototype board or by simply inserting a wire between the metal contact and the plastic housing of the connector.

FIGURE 2



The 60 Hz signal can be accessed in the two locations shown in figure 1. The first place is a small solder filled hole in the board. A wire may be soldered in the hole, or a wire wrap pin may be attached and connected to the NMI line via an alligator clip to make the clock removable. **NOTE; This may void your warranty. Check with your dealer!**

The other connection point does not involve soldering. To make the connection, carefully remove the IC at location C-14. The row and column numbers are marked on the board itself. Then insert a piece of very thin (wire wrap) wire into pin 4 of the socket. (See figure 1.) Now carefully reinsert the IC making sure it is oriented correctly and all the pins are securely seated in the socket.

Before this connection is made an interrupt driver must be entered into memory. If this is not done, the system will crash and RESET will have no effect until the connection is broken.

To get the three cent clock off to a flying start, enter the short program in listing 1. This can be done in the monitor or the mini-assembler. When the program is in memory, connect the interrupt line and watch the upper left hand corner of the screen. If everything was done correctly, the first character on the screen should start changing rapidly. What is happening is that sixty times a second the video circuitry generates a signal which is now being used to generate an interrupt. When an interrupt occurs, the processor starts executing the interrupt handler which is located at 3FB hexadecimal on an Apple. Usually the interrupt handler starts with a jump instruction since there are only five bytes of usable memory at

LISTING 1

```
*3FBL
03FB-   EE 00 04    INC   $0400
03FE-   40          RTI
03FF-   00          BRK
```

3FB, but since this program is so short it can be entered directly at 3FB. The interrupt handler that is now in memory simply increments a memory location and returns to the main program. This is a real time clock. It tells time in sixtieths of a second. Granted, it isn't very useful as it is now, but that will be fixed in a moment.

Now incrementing a memory location on the screen isn't very exciting, but try hitting a few keys. Surprise! They still work. In fact, everything works. Try dumping out some memory or printing something in basic. Everything will work normally and the first character on the screen will go right on counting. WARNING: the disk will **NOT** work. Neither will the tape. This is because the interrupts slow down the main program enough to upset the precise timing required by the disk and tape routines. Having the interrupt connected will also make the bell tone sound peculiar.

To make the clock more useful, enter the three programs in listing 2. The first program is simply a jump instruction to the second program which is a clock routine to drive an hour-minute-second clock. The third program is a basic routine which sets the clock and outputs the time of day. The programs are thoroughly documented so they won't be discussed here.

LISTING 2
PROGRAM #1

```
*3FBL
03FB-   4C 00 03    JMP   $0300
03FE-   00          BRK
03FF-   00          BRK
```

Making It Better or When Is An NMI Really An IRQ?

It should be clear by now that the power of the clock lies in the interrupt driver program, but there are some hardware enhancements that can be made. These extra features will roll the price up to a respectable 25 cents (more or less).

The first add-on is a sophicitcated piece of hardware called a switch. This is used to make easier the task of turning the interrupts on and off. The switch is installed so that it breaks the connection from the 60 Hz signal. Personal experience has shown that flipping a switch makes a more dignified display than pulling a wire in and out.

The second modification is a bit more complicated. (Seriously.) This modification allows the computer to control the interrupts via one of the annunciator outputs on the game I/O connector. The

www.commodore.ca

LISTING 2
PROGRAM #2

```
*300LL                  CLOCK
0300-   85 05           STA   $05       SAVE A AND X
0302-   86 06           STX   $06
0304-   A9 3C           LDA   #$3C      A=60 DECIMAL   X=0
0306-   A2 00           LDX   #$00
0308-   E6 04           INC   $04       COUNT 1/60 SECOND
030A-   C5 04           CMP   $04        FULL SECOND YET?
030C-   D0 22           BNE   $0330        IF NO THEN RESTORE REGISTERS & RETURN
030E-   86 04           STX   $04       RESET 1/60 SECONDS
0310-   E6 03           INC   $03       COUNT 1 SECOND
0312-   C5 03           CMP   $03        1 MINUTE YET?
0314-   D0 1A           BNE   $0330
0316-   86 03           STX   $03
0318-   E6 02           INC   $02       MINUTES
031A-   C5 02           CMP   $02
031C-   D0 12           BNE   $0330
031E-   86 02           STX   $02
0320-   A9 0D           LDA   #$0D      SET A=# HOURS IN 1 DAY PLUS 1
0322-   E6 01           INC   $01       HOURS
0324-   C5 01           CMP   $01       FULL DAY?
0326-   D0 08           BNE   $0330
0328-   E8              INX             IF YES SET HOURS TO 1
0329-   86 01           STX   $01
032B-   A5 05           LDA   $05       RESTORE REGISTERS
032D-   A6 06           LDX   $06
032F-   40              RTI
0330-   A5 05           LDA   $05       RESTORE THEM HERE TOO
0332-   A6 06           LDX   $06
0334-   40              RTI
0335-   00              BRK
0336-   00              BRK
0337-   00              BRK
0338-   00              BRK
0339-   00              BRK
```

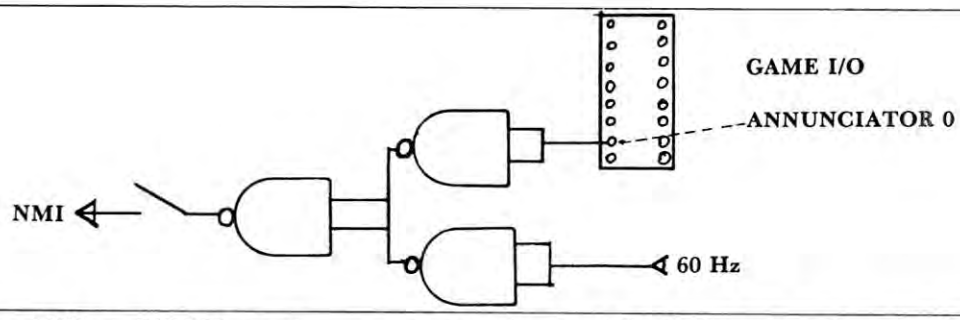
LISTING 2
PROGRAM #3

```
>LIST CLOCK DRIVER
  5 PRINT CHR$(4);"BLOAD CLOCK"
  7 POKE 1020,0: POKE 1021,3: REM  SET INTERRUPT
    VECTOR
 10 INPUT "INPUT TIME -->",H,M,S
 15 REM  SET CLOCK
 20 POKE 1,H
 30 POKE 2,M
 40 POKE 3,S
 43 POKE 4,0
 45 A= PEEK (-16296): REM  TURN CLOCK ON
 47 INPUT "12 OR 24 HOUR CLOCK",A: POKE 801,A+1
 48 REM  SEE LISTING   FOR EXPLAINATION OF LINE 47
 50 CALL -936: REM  CLEAR SCREEN
 60 VTAB 10: PRINT "
 61 REM  ERASE OLD TIME
 70 VTAB 10: TAB 10
 75 REM  DISPLAY CURRENT TIME
 80 PRINT PEEK (1);":";
 81 REM  HOURS
 90 IF PEEK (2)<10 THEN PRINT "0";: PRINT PEEK (2),
 91 REM  MINUTES
100 PRINT PEEK (3), PEEK (4): GOTO 60
110 REM  SECONDS AND 1/60 SECONDS
```




www.commodore.ca

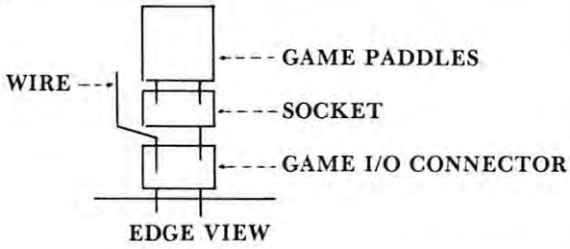
FIGURE 3



only extra part required is a 7400 or 74LS00 nand gate. It is wired according to figure 3 using a prototype board, an off-board wire wrap socket, or the breadboard area on the Apple board.Even the revision 1 boards have room for two IC's in the right hand corner under the keyboard. NOTE: To wire the modification in this way requires removal of the Apple board and will probably void your warranty. Check with your local dealer.

The connection to the game I/O connector is made using a piece of stiff wire such as the lead of a small resistor. This wire is inserted into the connector and bent as shown in figure 4. A 16 pin IC socket with one pin clipped to accommodate the wire is inserted over that and the game paddles are plugged into that socket. Many connections can be made to the game connector in this manner without having to clip pins off of the game paddles.

FIGURE 4



The Disadvantages

Unfortunately, every silver lining comes equipped with a cloud and the 25 cent clock is no exception. The main problem is that the disk and tape will not work, as well as other programs which involve precise timing. The interrupts must be disabled, either manually or under program control, while such programs are running.

Another hitch is in the computer control circuit itself. When an Apple is turned on, the annunciator outputs are high (logic 1) so this has been made to disable the interrupts. An autostart rom however, turns all the annunciators to logic 0. Before this happens all the annunciators are still at logic 1 for a few milliseconds so inverting the signal from the annunciator will still leave the interrupts enabled for enough time to cause an interrupt and a system crash. Therefore, the interrupts must be disabled manually upon power up with an autostart rom.

Another problem is that the bell tone sounds raspy. This isn't serious, but it can get on your nerves after a while. It doesn't make a good way to check if interrupts are enabled.

The final problem is that the clock seems to lose about ten seconds each hour. This can be remedied by adding ten seconds to the seconds counter each hour.

Fringe Benefits

The 25 cent clock is remarkably user proof. The NMI line doesn't require debouncing, and resetting the comupter doesn't interfere with its operation either (unless the reset key is held down for a long time).

The two main dangers of system crashes are working on the interrupt handler while interrupts are enabled, and not saving registers. THIS IS IMPORTANT!!! You must save each register you intend to modify. If you do not you will get very mysterious results. You can save registers in memory or you can push them onto the stack. There is also a routine to save and restore all registers in the monitor.

Once these restrictions have been met, the 25 cent clock opens a vast new horizon of features that would cost tens of dollars if bought from vendors. The price you pay is speed. The longer the interrupt routine, the slower the computer runs. This is not a severe handicap. The clock routine does not slow the computer down enough to be perceived, even when the interrupts are switched on and off for comparison. In order to slow the computer down by even one percent it requires a one hundred instruction routine.

Some things that can be done include:

Control Of Computer Speed Using Game Paddles: have the interrupt driver pause according to the position of a game paddle to give control of listing speed, how fast a program runs, etc.

Keyboard Buffering: have the interrupt routine sample the keyboard and store any keypresses in a buffer to give storage of multiple keypresses while something else is going on.

Mixing Display Modes: sixty times a second switch to another display mode to mix text and graphics, or mix two graphics modes for extra colors.

The possibilities are endless. You can even run two programs at once using the interrupt. The twenty-five cent Apple II real time clock is a lot more than just a clock, it's a cheap way of doing a lot of expensive things, right in line with Apple tradition.

www.commodore.ca