8080 *Real-time clocks—how they work and the software requirements.*

D. LARSEN, C. TITUS, R. RONY and J. TITUS*

THERE ARE DIFFERENT TYPES OF REAL-time clocks. Perhaps the simplest type that can be built is a *free-running real-time clock*. Figure 1 shows the schematic diagram. The heart of this circuit is the Mostek Corporation's MK5009, which is driven by a 1-MHz quartz crystal. This device contains several divide-by-10 counters, one of which you can select to drive the output of the MK5009. There are four digital inputs to the device that are used to select the required decade counter. The truth table for these inputs and the resulting output frequencies are shown in Table 1. For input values greater than 1000_2, the MK5009 generates frequencies that are not multiples of ten.

TABLE 1

Inputs				Frequency
D3	D2	D1	D0	
0	0	0	0	1 MHz
0	0	0	1	100 kHz
0	0	1	0	10 kHz
0	0	1	1	1 kHz
0	1	0	0	100 Hz
0	1	0	1	10 Hz
0	1	1	0	1 Hz
0	1	1	1	0.1 Hz
1	0	0	0	0.01 Hz

As shown in Fig. 1, a 4-bit latch (SN7475) is used in the real-time clock interface between the 8080 microcomputer and the MK5009. This latch is used to program the MK5009 for a particular frequency. The output of the MK5009 clocks a D-type flip-flop whose output goes to some additional interrupt-interface hardware. Let's assume that this hardware produces a RST7 instruction when the 8080 acknowledges the interrupt.

Instructions that can be used to program the MK5009 are shown in Table 2. In this program, after the stack pointer is loaded, the A-register is loaded with the value 00000011_2. This value is transferred to output port 305, which is the SN7475 latch in the real-time clock interface. The four least-significant bits of this value are the only bits within the 8-bit byte that are actually latched by the SN7475, and the MK5009 is programmed for 1-kHz operation by this value, 0011_2. Once the MK5009 has been programmed, the interrupt flip-flop (SN7474) is cleared by the second OUT instruction and the interrupt is enabled (EI).

In 1 ms or less, the 8080 is interrupted by the real-time clock. We have assumed that when the real-time clock interrupts the 8080, it is vectored to memory location $000\ 070_8$ by the interrupt hardware. This is where the real-time clock interrupt-service subroutine must be stored in memory. In this subroutine, the 8080 may have to transfer some data between itself and a peripheral device, or it may simply turn some lights and values on or off. However, once the 8080 has performed these tasks, the interrupt flip-flop (the SN7474 IC) shown in Fig 1 is cleared and the interrupt is re-enabled. The 8080 then returns to the program that was interrupted by the real-time clock.

A characteristic of this real-time clock is that it is *free-running,* which means you cannot turn it off or stop it. The clock will always generate a squarewave with a frequency of 1 kHz. The only way that you can prevent the 8080 from being

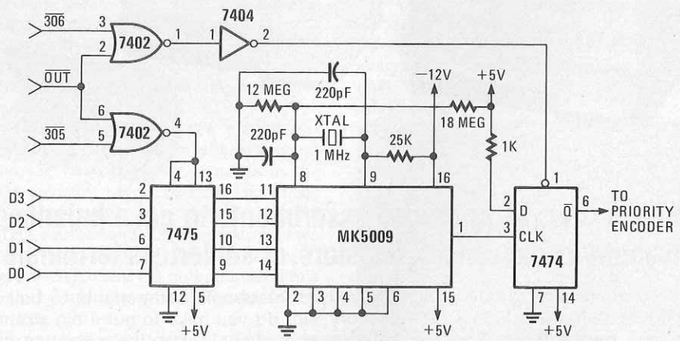

FIG. 1

TABLE 2

```
                *000 000
START,          LXISP     /LOAD THE STACK POINTER WITH A
                STACK     /R/W MEMORY ADDRESS BECAUSE IN-
                0         /TERRUPTS CAN OCCUR.
                MVIA      /THEN LOAD THE A REGISTER WITH
                003       /00000011.
                OUT       /OUTPUT THIS VALUE TO THE
                305       /MK5009 (1 KHZ OPERATION)
                OUT       /CLEAR THE FLIP-FLOP THAT IS
                306       /WIRED TO THE MK5009
                EI        /ENABLE THE INTERRUPT
                •         /THEN EXECUTE THE REMAINDER
                •         /OF THE PROGRAM.
                •

                *000 070
RTCISS,         •         /THE REAL-TIME CLOCK INTERRUPTED
                •         /THE MICROCOMPUTER, SO SERVICE
                •         /SOME OF THE PERIPHERAL DEVICES
                OUT       /THEN CLEAR THE FLIP-FLOP THAT
                306       /CAUSED THE INTERRUPT (MK5009)
                EI        /RE-ENABLE THE INTERRUPT
                RET       /AND RETURN TO THE TASK THAT
                          /WAS INTERRUPTED.
```

*This article is reprinted courtesy American Laboratories. Dr. Rony, Department of Chemical Engineering, and Mr. Larsen, Department of Chemistry, are with the Virginia Polytechnic Institute & State University. Both Dr. C. Titus and Mr. J. Titus are with Tychon, Inc.

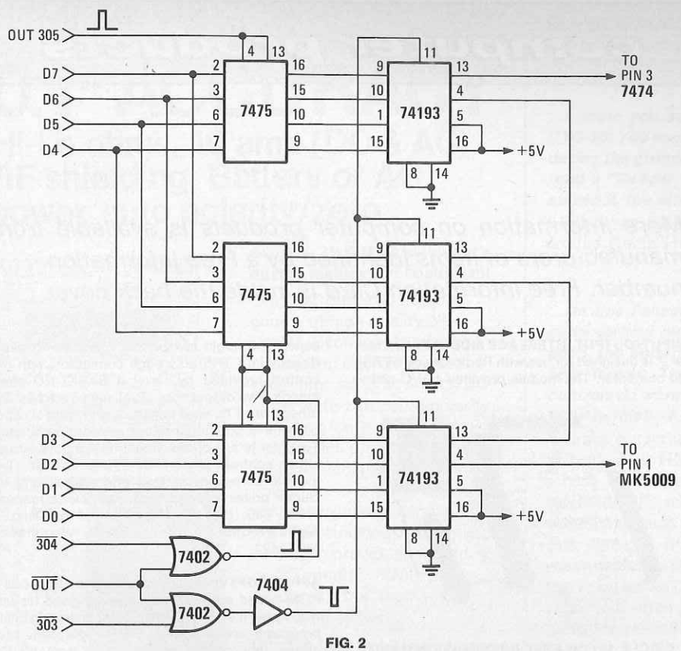

FIG. 2

interrupted by the clock is to disable the interrupt by executing a DI instruction. The limitation of this real-time clock is that it can only be programmed to generate the frequencies listed in Table 1.

How can this device be used to generate an interrupt every 15 or 20 ms? One practical method would be to add some *programmable down-counters* to the real-time clock. You can use the MK5009 to clock these counters, and when they have counted down to 0, the interrupt flip-flop is clocked so that an interrupt occurs. The latches and counters are wired to the 8080 and the MK5009, as shown in Fig. 2. The content of the A-register is latched by this interface when an OUT 304 instruction or an OUT 305 instruction is executed. When an OUT 303 Instruction is executed, the content of the latches is loaded into the counters. Since three 4-bit counters are used, the counters can be loaded with any number between 0 and 111111111111_2. This means that by placing these three counters between the MK5009 and the interrupt flip-flop, the MK5009 can generate a maximum of 4096 output pulses before an interrupt will occur.

Once these counters and latches are added to the interface, you still have to write a program that will *program* the real-time clock for a 15- or 20-ms interval. For a 20-ms interval, the software in Table 3 can be executed. This software loads the counters with the number 000000010100_2 and also programs the MK5009 for 1-kHz operation (1-ms time interval). An OUT 303 instruction must also be executed.

continued on page 92

COMPUTER CORNER

continued from page 87

Are there any other values that could have been used to program the real-time clock for a 20-ms interrupt? Actually, 200_{10} could have been loaded into the counters, and the MK5009 would have been programmed for 10-kHz operation (0.1-ms time interval). A value of 2000_{10} could also be used if the MK5009 is programmed for 100-kHz operation (0.01-ms time interval).

When the real-time clock interrupts the 8080, the 8080 services some peripheral devices and then clears the interrupt flip-flop (OUT 306). The count is then transferred from the latches to the counters when the OUT 303 instructon is executed. This must be done because the counters, after counting to zero, are next decremented to 11111111111_2. Therefore, if the counters were not reloaded, the 8080 would be interrupted 4096 clock pulses later, rather than the count in the latches. After re-enabling the interrupt, the 8080 returns to the interrupted task.

Adding the three 4-bit counters to the real-time clock interface has increased its capabilities. Intervals of from 1 μs to 4.096×10^5 seconds can be timed.

Another real-time clock application is as a *time-of-day clock*. This type of clock is simply a peripheral device, or a series of memory locations, in which the current time of day is stored and updated. The time can be updated every second or $^1/_{100}$ second, depending on the hardware, software and uses of the clock.

One method of constructing a time-of-day clock would be to program the MK5009 for 1-Hz operation and then wire the output of the MK5009 to a *counter chain* or *divider chain*. These chains consist of a divide-by-10 counter and a divide-by-6 counter for seconds, a divide-by-10 counter and a divide-by-6 counter for minutes, and a divide-by-4 counter and a divide-by-6 counter for hours. Instead of using a divide-by-4 counter and a divide-by-6 counter for a 24-hour format, a divide-by-4 and a divide-by-3 counter could be used for a 12-hour format. The counter outputs could be wired through tri-state interface devices to the microcomputer's data bus. The microcomputer would then have to execute some accumulator I/O or memory-mapped I/O instructions to read the time from the time-of-day clock. By using this method, *no* interrupts are required and the software instructions for reading the time are very simple. Additional instructions could be added so that you can enter a time into the microcomputer via a CRT or teletypewriter. This time would then be written out to the time-of-day clock, so that it is programmed for the correct time when the microcomputer is started. This can only be done if programmable counters are used in the time-of-day clock interface. Of course, an MK5009 does not have to be used as a 1-Hz clock source. A 60-Hz signal could be derived from the 110- to 220-VAC power lines and then be divided by 60 before being applied to the counter chain. **R-E**

TABLE 3

```
                *000 000
START,          LXISP
                STACK
                0
                MVIA
                003
                OUT
                305
                MVIA
                024
                OUT
                304
                OUT
                306
                OUT
                303
                EI
                •
                •
                •

                *000 070
RTCISS,         •
                •
                •
                OUT
                306
                OUT
                303
                EI
                RET
```

```
/LOAD THE STACK POINTER WITH A
/R/W MEMORY ADDRESS BECAUSE IN-
/TERRUPTS CAN OCCUR.
/THEN LOAD THE A REGISTER WITH
/0000 0011, TO PROGRAM THE MOST
/SIGNIFICANT COUNTER WITH 0
/WHEN THE MK5009 (1 KHZ).
/THEN LOAD THE A REGISTER WITH
/00010100 SO THAT THE TWO LEAST
/SIGNIFICANT COUNTERS ARE
/LOADED WITH DECIMAL 20.
/CLEAR THE INTERRUPT FLIP-FLOP

/THEN TRANSFER THE CONTENT OF
/THE LATCHES TO THE COUNTERS.
/ENABLE THE INTERRUPT AND
/THEN EXECUTE THE REMAINDER
/OF THE PROGRAM.


/THE REAL-TIME CLOCK INTERRUPTED
/THE MICROCOMPUTER, SO SERVICE
/SOME OF THE PERIPHERAL DEVICES
/THEN CLEAR THE FLIP-FLOP THAT
/CAUSED THE INTERRUPT AND
/RELOAD THE COUNTERS WITH
/THE CONTENT OF THE LATCHES.
/RE-ENABLE THE INTERRUPT
/AND RETURN TO THE TASK THAT
/WAS INTERRUPTED.
```