Circuit steps program for 8080 debugging

by John F. Wakerly
Stanford University, Stanford, Calif.

Executing a program one step at a time is an important aid to debugging microprocessor systems. There are two basic approaches to providing a single-step capability. One method is to add hardware to provide a software interrupt after each instruction's execution, and the other is to provide a completely hardware-oriented "front panel" to give the capability to execute single instructions or memory cycles under hardware control.

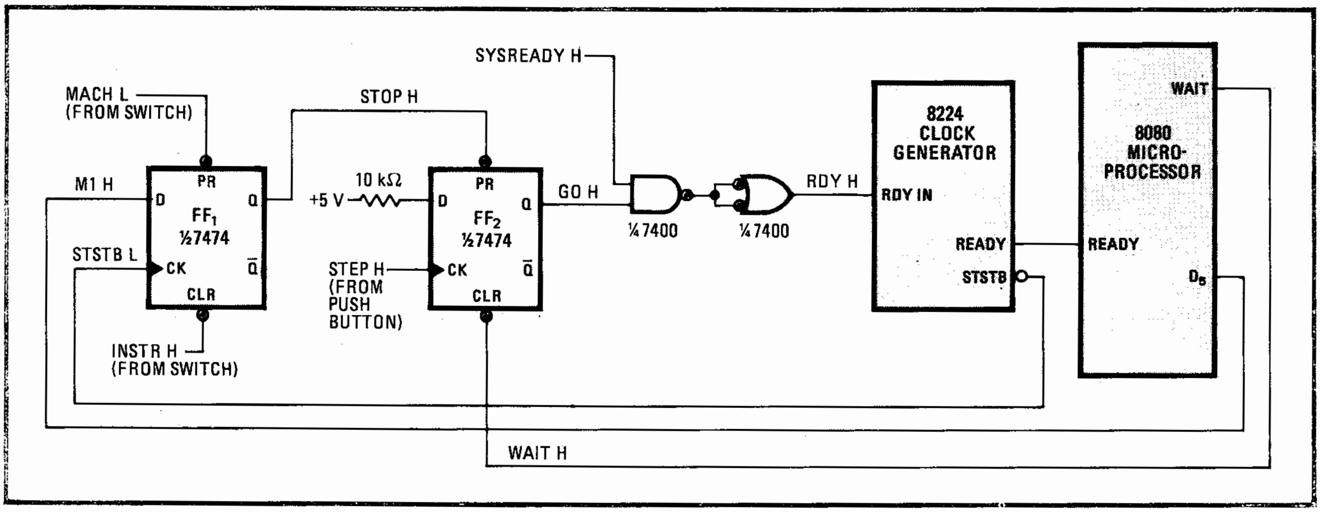
With the first approach, the user can write an interrupt service routine that allows register and memory to be examined at the system teletypewriter or terminal between instruction executions [*Electronics*, June 24, 1976, p. 105]. It takes advantage of the full power and convenience of a software-debugging package, but instructions cannot be single-stepped if interrupts are disabled, and single memory-reference stepping is not possible (there may be several memory references per instruction, and interrupts cannot take place in the middle of an instruction).

The circuit described here is the hardware-implemented "front panel," which enables the user to execute 8080 microprocessor programs either one instruction at a time or one machine cycle (memory or input/output reference) at a time. The circuit uses the READY input of the 8080 to stop the program at each instruction or machine cycle, as selected by the user. A push button then runs the program one step at a time. Between steps, the user can observe the machine state and the current instruction or data on the 8080 data, address, and control lines with a scope, logic probe, or permanent indicator lamps.

The single-step circuit shown in Fig. 1 assumes that an 8224 clock generator is used with the 8080. Without the single-step circuit, a ready signal (SYSREADY H) generated by the memory and I/O systems would be connected to the 8224 RDYIN input. For single-stepping, SYSREADY H (where H denotes active high) is ANDed with the signal GO H. If either signal is low during a memory or I/O reference, the 8080 will go into a wait state until both signals become high. If this circuit is used in a system that does not generate SYSREADY H, then the AND can be removed and GO H connected directly to RDYIN.

A 7474 edge-triggered D-type flip-flop, FF_2, generates the GO H signal. Proper operation of the circuit depends on the fact that a low input on the PR input of the 7474 produces a high output at Q, regardless of any of the other inputs, including CLR. (If both PR and CLR are low, then both Q and $\overline{Q}$ are high.) Thus, if STOP H is low, GO H is high, and the 8080 executes instructions at full speed. However, the 8080 holds WAIT H low just before every memory or I/O reference; thus, if STOP H is high, FF_2 is cleared by the low signal on WAIT H, the 8080 enters the wait state, and instruction execution is stopped. In the wait state, WAIT H goes high and allows GO H to be set high again by clocking FF_2 with the STEP H input. STEP H is the output of a single-step push button, and instruction execution does not begin until this button is pushed.

If only single-stepping at each machine cycle were desired, STOP H could be obtained from a simple switch to select normal operation or single-stepping. However for single instruction stepping, FF_1 is used to detect the beginning of each instruction cycle—the instruction fetch. At the beginning of each machine cycle, the 8080 places a signal M1 H on data-bus output D_5 which



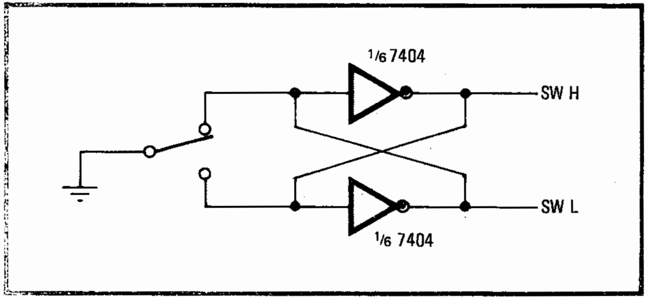
1. Single-step circuit for 8080. This circuit requires only one 7474 dual D flip-flop and half of a 7400 quad NAND package to provide sinç instruction and single-machine-cycle-execution capability for an 8080 microprocessor system. It can be modified for a Zilog Z-80.

indicates the first machine cycle (fetch) of an instruction. This signal is clocked into FF_1 by STSTB L; thus if M1 H is high, STOP H becomes high, and the 8080 is stopped during the instruction fetch. The data bus contains the fetched instruction.

The PR and CLR inputs of FF_1 are used to determine the mode of operation. If MACH L is asserted (low), STOP H is held unconditionally high, and the 8080 stops at each machine cycle. If MACH L is de-asserted and INSTR H is asserted (high), then M1 H is clocked into FF_1 to generate STOP H, and the 8080 stops at every instruction cycle. If both MACH L and INSTR H are de-asserted, then STOP H is held low, and the 8080 operates normally.

If the inputs MACH L, INSTR H, and STEP H are obtained from switches, they should be debounced. Figure 2 shows an economical debouncing circuit that uses a pair of inverters to debounce an single-pole, double-throw switch. The circuit shorts the high output of an inverter to ground for about 20 nanoseconds at each transition, but this short is not harmful.

An advantage of this circuit over conventional cross-couple NAND gates for debouncing is that three switches can be debounced with one 7404, as opposed to two with one 7400. Also, no pull-up resistors are used.



2. Switch-debouncing circuit. Part of a 7404 package debounces spdt switch. Signals are active when switch has position shown.

The single-step circuit can be easily modified for use with the Zilog Z-80 microprocessor. The Z-80 has a separate output pin for $\overline{\text{M}1}$, which should be connected to the D input of FF_1. Since this polarity is the opposite of the 8080 M1 output, the STOP H signal should be obtained from the $\overline{\text{Q}}$ output of FF_1, and the PR and CLR inputs should be reversed. FF_1 should be clocked by the Z-80 input clock Φ, and the RDY H signal should be connected to the $\overline{\text{WAIT}}$ input of the Z-80. Finally, the CLR input of FF_2 should be connected to $\overline{\text{MREQ}+\text{IORQ}}$, which can be obtained from the $\overline{\text{MREQ}}$ and $\overline{\text{IORQ}}$ outputs of the Z-80 with a single 2-input NAND gate (one quarter of a 7400 package). □