SESSION I

MICROCOMPUTER-BASED LABORATORIES I

DORIS AARONSON, *New York University, Presider*

Real-time clocks for microcomputers in behavioral research

STEVE GOLDBAND

State University of New York, Buffalo, New York 14226

Real-time clock strategies are described for microcomputers in behavioral research. Software-oriented and hardware-oriented counters are described in both free-running and strobed modes. An example of a clock using the Intel 8253 is given, along with driving software in 8080 assembly code and hardware for the S-100 bus.

A real-time clock is an essential component of any computer installation in which the duration of, or interval between, events is of interest. Many psychological research paradigms amenable to computer control are of this type, including reaction time, memory, psychophysiology, perception, learning, and others. The recent proliferation of relatively inexpensive computers, many of which are based on the S-100 bus configuration, highlights the need for a clock capable of timing events with acceptable precision. The present paper describes the two major strategies for implementing real-time measurement, along with hardware and software examples which implement these strategies.

A real-time clock is a device capable of timing events according to some externally referenced time base. At the heart of a real-time system are a counter and a time base of a specific frequency which increments the counter. The counter may be free-running, in which case elapsed times are determined by subtracting the count at the end of an interval from that at the beginning of an interval and simply read at the end. Each of these counter strategies may be implemented in a hardware-oriented or a software-oriented mode. In the former case, a counter exists in circuits external to the CPU that can be read from programs in the main computer. In the latter case, the counter is implemented in a part of the CPU's memory space and is incremented by software whenever the time base causes an interrupt to occur.

The time base of a counter system typically exists as a hardware device of some kind. In some systems, a

crystal oscillator and divider produce the required pulse at a precise interval. In other cases, the computer's own system clock (usually between 1 and 4 MHz) is divided down by a counter, and the output of the counter forms the time base. The crystal oscillator is a better, though slightly more expensive, choice if extreme accuracy is required. In applications where moderate accuracy is sufficient (say, .1% to .05%), a divider on the system clock performs well at lower cost.

In a hardware-oriented design, the counter is formed by a series of integrated circuit counters arranged so that the "carry" of one drives the next. As many bits of counter are used as required by the longest interval of interest in the smallest necessary time unit. (For example, if one counts intervals up to 1 min with a precision of 1 msec, a counter must contain at least 60,000 possible values. Since $2^{16} = 65536$, a 16-bit counter is necessary.) The counter is driven by a time base with a period equal to the unit of measurement (e.g., 1 msec) and is either free-running or strobed. In order to time several intervals that begin and end at random with respect to one another (such as reaction times of several subjects to several stimuli), a free-running counter can be read and stored in a memory location at the start of each interval, and read and compared to the stored start time as each interval ends. In an interrupt environment, a strobed counter can be configured to interrupt the CPU after a predetermined interval has expired. This frees the CPU for other tasks while awaiting the completion of an event.

A software-oriented counter, on the other hand, exists primarily as a series of memory locations. A time base in hardware interrupts the CPU each time a count takes place, and the interrupt service routine (ISR) increments the value of the memory locations in which the counter resides. The circuitry for the counter is simple (see Figure 1), but the software to drive it can be

Requests for reprints should be addressed to the author, Department of Psychology, State University of New York, 4230 Ridge Lea Road, Buffalo, New York 14226. The author acknowledges the contribution of Robert S. Bundy to the design of the clock board described.

come complex, particularly if it exists in an environment with devices that cannot be interrupted during critical periods [e.g., many disk systems using direct

Figure 1. A circuit to implement a simple interrupt timer.

001	START	PUSH PSW
002		PUSH B
003		PUSH D
004		PUSH H
005		LDA TIME1
006		INR A
007		STA TIME1
008		LDA TIME2
009		INR A
010		STA TIME2
011		POP H
012		POP D
013		POP B
014		POP PSW
015		RET

Figure 2. An example of an interrupt service routine which manages two software timers.

memory access (DMA) and some printers]. Figure 2 gives an example of a simple ISR written in 8080 assembler that might be called on every time base interrupt pulse to update two counters in memory locations TIME1 and TIME2. Using multiple counters to time simultaneous events is a matter of establishing as many memory locations as necessary and incrementing them on each interrupt. Software can also be developed to allow timing of events while the CPU continues useful work. In this case, the ISR checks for a predetermined terminal count when each interrupt occurs. If the count expires, the ISR takes a required action, such as terminating a stimulus presentation; otherwise, it returns control to the main program.

In the system described here, a free-running hardware clock with a time base of 1 msec is used. Sixteen timers, each 32 bits wide, are implemented in software. The time base is derived from the 2.045-MHz system clock by dividing it by 2,045 to yield a 1-msec pulse. While this arrangement does not maximize throughput, it has the virtue of simplicity and low cost. At the same time, the system is powerful enough to allow implementation of rather complex psychophysiological paradigms.

The clock is based on a powerful LSI integrated circuit, the Intel Corporation 8253 (Intel Corporation, 1978). The features of this circuit element make pos-

Figure 3. The circuit diagram for the author's real-time clock board.

sible an inexpensive design with a minimum of support chips. Nevertheless, a range of options is available, many of which are under software control.

The 8253 has capabilities too numerous to describe completely in this space.¹ Basically, however, it consists of three independent 16-bit counters that can operate from dc to 2 MHz. Each counter can be operated in one of several modes, including a simple divide-by-N counter, a preset down counter, a square wave generator, and a software-triggered one-shot. The operation of each counter is determined by writing a "mode" word into an output port, followed optionally by actual values to be used for initial counts or divide rates. Each counter may be reset at any time by writing the new value to an output port, and each may be read by reading the value from a designated input port. Each counter can be operated in binary or binary-coded decimal (BCD), and the counters can be set, read, or reset at any time under software control. Each counter is provided with a "gate" pin, an "output" pin, and a "clock" pin, which allow for flexible hardware configurations.

In the system described here, the counters are configured as a clock by driving them with a time base. The system clock is divided by one of the counters to a 1-kHz rate by configuring the counter with software to be a divide-by-2045 rate generator. The 1-kHz signal is counted by the other two counters, which are ganged together to form a 32-bit timer. The timer can time intervals up to about 49 days with a resolution of 1 msec. The accuracy of the timer is determined, in part, by the correct setting of the divider and, in part, by the resolution required. With a resolution of 1 msec, an optimal division rate yields an error of, at most, one part in 2,045, or about .05%. In applications requiring greater accuracy or precision, an on-board crystal oscillator with a precision crystal can be substituted for the computer's system clock as the time base.

The circuit for this S-100 bus real-time clock is shown in Figure 3. The prototype was constructed on an Artec Corporation² WW-100 general-purpose printed circuit card with wire-wrap techniques. The total component cost for the clock is approximately \$75. Experienced users will find that adapting the circuit to other bus configurations and other processors is a relatively straightforward task. Some advanced applications may require multiple timers, in which case additional 8253s can be added to the board for approximately \$10 each, with little additional support circuitry.

Address lines A7-A2 of the S-100 bus are decoded by U2 to select the four contiguous ports at which the 8253 resides. Lines A0 and A1 are connected directly to the 8253. Read/write control is implemented in NAND gates of U9. During a CPU read operation, SINP and DBIN are active high. One section of U9 detects this condition, enables the output buffers in U5 and U6 and RD on the 8253. During a CPU write operation, SOUT is active high and PWR is active low. PWR is inverted by one section of U9, then gated with

SOUT to enable input buffers U7 and U8 and $\overline{\text{WR}}$ on the 8253. The gate pins of all counters are tied high to enable counting at all times, and the three counters are ganged so that the Phase 02 clock is the input to Counter 0, the output of Counter 0 is the input to Counter 1, and the output of Counter 1 is the input to Counter 2. The output of Counter 1 is inverted before it drives Counter 2 to assure correct counting during carry operations.

Assembly language software to drive the clock is shown in Figure 4. The initialization phase is the most

```

0001 * THESE ARE THE DRIVER ROUTINES FOR THE
0002 * 8253 COUNTER CHIP, WITH THE INPUT OF COUNTER
0003 * ZERO TIED TO 02 CLOCK (2.045 MHZ), AND COUNTERS 1 AND
0004 * 2 GANGED AS HIGHER ORDER COUNTERS. IN THIS CONFIGURATION
0005 * THE INPUT TO COUNTER 0 IS DIVIDED BY 2045 TO YIELD A
0006 * 1 KHZ TIMEBASE FOR COUNTERS 1 AND 2, WHICH ARE FREE
0007 * RUNNING. THE COUNTER IS ADDRESSED TO PORTS E0-E3.
0008 *
0009 CLK0 EQU 0E0H THE 8253 IS ADDRESSED TO E0-E3
0010 CLK1 EQU CLK0+1
0011 CLK2 EQU CLK0+2
0012 MODE EQU CLK0+3
0013 *
0014 * JUMP TABLE BEGINS HERE
0015 J1 JMP INIT INITIALIZE COUNTERS
0016 J2 JMP RD1 READ CLOCK 1, CONTENTS IN REGISTERS BC
0017 J3 JMP RD2 READ CLOCK 2, CONTENTS IN REGISTERS BC
0018 * END OF TABLE
0019 *
0020 INIT MVI A,084H MODE WORD FOR COUNTER 2 (MSB)
0021 OUT MODE
0022 MVI A,0FULL COUNT
0023 OUT CLK2
0024 OUT CLK2
0025 MVI A,34H LOAD MODE WORD FOR COUNTER 0, MODE 2,
0026 OUT MODE
0027 MVI A,02 SHORT COUNT FOR INITIAL CLEAR OPERATION
0028 OUT CLK0 LOAD LSB
0029 MVI A,00 SHORT COUNT FOR CLEAR
0030 OUT CLK0 LOAD MSB
0031 MVI A,74H MODE WORD FOR COUNTER 1, MODE 2, BINARY
0032 OUT MODE
0033 MVI A,2 COUNTER SET SHORT COUNT
0034 OUT CLK1 LOAD LSB
0035 MVI A,0
0036 OUT CLK1 LOAD MSB
0037 NOP TIMER COUNTS OUT TIME HERE
0038 NOP
0039 NOP
0040 * NOW SET COUNTER VALUES ACTUALLY REQUIRED
0041 MVI A,34H DIVIDE BY 2045 IN COUNTER 0
0042 OUT MODE
0043 MVI A,74H
0044 OUT MODE
0045 MVI A,0FDH LSB OF 2045 DIVIDER 0
0046 OUT CLK0
0047 MVI A,7 MSB OF 2045 DIVIDER
0048 OUT CLK0
0049 MVI A,0
0050 OUT CLK1 MAXIMUM COUNT (65536) FOR COUNTER 1
0051 OUT CLK1
0052 RET
0053 *
0054 * READ CLK1
0055 RD1 MVI A,40H SET MODE FOR LATCHED READ
0056 OUT MODE
0057 IN CLK1
0058 MOV B,A SAVE MSB IN B
0059 IN CLK1
0060 MOV C,A LSB IN C
0061 RET
0062 *
0063 * READ CLK2
0064 RD2 MVI A,80H
0065 OUT MODE
0066 IN CLK2
0067 MOV B,A
0068 IN CLK2
0069 MOV C,A
0070 RET
0071 *END*

```

Figure 4. Assembly language routines to initialize and read the clock in Figure 3. The board is assumed to be addressed to I/O ports E0-E3.

complex, and is necessary to determine the mode of operation of each counter and any initial counts or rates that are required. This routine need be executed only once, but, if desirable, several alternate initialization routines may be used dynamically during a program to change the definition of one or more counters. Reading the value of a section of the counter is conceptually straightforward, although in practice it can be complicated by the need to handle as many as 4 bytes of data. An interesting feature of the 8253 is its ability to latch a count independently of reading it. This makes it possible to "freeze" the value of a 32-bit timer and read it at a later point without being concerned about intervening counts.

In the initialization routine, Counters 0 and 1 are first set to divide by 2, and then reset to divide by 2,045 and 65,536, respectively. This is done because after a new or initial count is loaded into a counter, one clock pulse on that counter is necessary to institute the count. This scheme guarantees that each counter receives several clock pulses after it is loaded and before its value must be valid.

The real-time clock serves as the basis for a series of timers implemented in software that work by storing an initial time and subtracting a later time from the stored value to determine an elapsed interval. Thus the 8253

is used as a running clock that is read as needed. The chip itself, however, allows very powerful functions to be carried out, such as interrupt on terminal count, hardware-triggered timing, event counting, etc. Circuit designers are referred to the full specifications for further information.

SUMMARY

A board has been described which fills the need of many laboratory applications for a real-time clock. It provides a highly flexible configuration that is adaptable to virtually any laboratory need for timing or counting. The board can be built by an inexperienced user with readily available and inexpensive parts.

REFERENCE

INTEL CORPORATION. *The data catalog*, 1978. Santa Clara, Calif: Author, 1978.

NOTES

1. The interested reader is referred to the full specifications in the Intel Data Catalog, Intel Corporation, 1978.
2. Artec Electronics Corporation, 605 Old Country Road, San Carlos, California 94070.