what-when-how

In Depth Tutorials and Information

TIME DELAY FOR VARIOUS 8051 CHIPS

SECTION 3.3: TIME DELAY FOR VARIOUS 8051 CHIPS

In the last section we used the DELAY subroutine. In this section we discuss how to generate various time delays and calculate exact delays for the 8051 and DS89C4xO.

Machine cycle for the 8051

The CPU takes a certain number of clock cycles to execute an instruction. In the 8051 family, these clock cycles are referred to as *machine cycles.* Table A-l provides the list of 8051 instructions and their machine cycles. To calculate a time delay, we use this list. In the 8051 family, the length of the machine cycle depends on the frequency of the crystal oscillator connected to the 8051 system. The crystal oscillator, along with on-chip circuitry, provide the clock source for the 8051 CPU (see Chapter 8). The frequency of the crystal connected to the 8051 family can vary from 4 MHz to 30 MHz, depending on the chip rating and manufacturer. Very often the 11.0592 MHz crystal oscillator is used to make the 8051 -based system compatible with the serial port of the IBM PC (see Chapter 10). In the original 8051, one machine cycle lasts 12 oscillator periods. Therefore, to calculate the machine cycle for the 8051, we take 1/12 of the crystal frequency, then take its inverse, as shown in Example 3-13.

Example 3-13

The following shows crystal frequency for three different 8051-based systems. Find the period of the machine cycle in each case.

(a) 11.0592 MHz (b) 16 MHz (c) 20 MHz

Solution:

(a) 11.0592 MHz/12 = 921.6 kHz; machine cycle is 1/921.6 kHz = 1.085 µs (microsecond)

(b) 16 MHz/12 = 1.333 MHz; machine cycle (MC) = 1/1.333 MHz = 0.75 µs

(c) 20 MHz/12 = 1.66 MHz; MC = 1/1.66 MHz = 0.60 µs

Example 3-14

For an 8051 system of 11.0592 MHz, find how long it takes to execute each of the following instructions.

(a) `MOV R3,#55` (b) `DEC R3` (c) `DJNZ R2,target`
(d) `LJMP` (e) `SJMP` (f) `NOP` (no operation)
(g) `MUL AB`

Solution:

The machine cycle for a system of 11.0592 MHz is 1.085 µs as shown in Example 3-13. Table A-1 in Appendix A shows machine cycles for each of the above instructions. Therefore, we have:

Instruction	*Machine cycles*	*Time to execute*
(a) `MOV R3,#55`	1	1×1.085 µs = 1.085 µs
(b) `DEC R3`	1	1×1.085 µs = 1.085 µs
(c) `DJNZ R2,target`	2	2×1.085 µs = 2.17 µs
(d) `LJMP`	2	2×1.085 µs = 2.17 µs
(e) `SJMP`	2	2×1.085 µs = 2.17 µs
(f) `NOP`	1	1×1.085 µs = 1.085 µs
(g) `MUL AB`	4	4×1.085 µs = 4.34 µs

Delay calculation for 8051

As seen in the last section, a delay subroutine consists of two parts: (1) setting a counter, and (2) a loop. Most of the time delay is performed by the body of the loop, as shown in Example 3-15.

Example 3-15

Find the size of the delay in the following program, if the crystal frequency is 11.0592 MHz.

```
         MOV   A,#55H      ;load A with 55H
AGAIN:   MOV   P1,A        ;issue value in reg A to port 1
         ACALL DELAY       ;time delay
         CPL   A           ;complement reg A
         SJMP AGAIN        ;keep doing this indefinitely
;----Time delay
DELAY:   MOV   R3,#200     ;load R3 with 200
HERE:    DJNZ R3,HERE      ;stay here until R3 become 0
         RET               ;return to caller
```

Solution:

From Table A-1 in Appendix A, we have the following machine cycles for each instruction of the DELAY subroutine.

```
                            Machine Cycle
DELAY:   MOV   R3,#200           1
HERE:    DJNZ R3,HERE            2
         RET                     2
```

Therefore, we have a time delay of $[(200 \times 2) + 1 + 2] \times 1.085\ \mu s = 436.255\ \mu s$.

Very often we calculate the time delay based on the instructions inside the loop and ignore the clock cycles associated with the instructions outside the loop.

In Example 3-15, the largest value the R3 register can take is 255; therefore, one way to increase the delay is to use NOP instructions in the loop. NOP, which stands for “no operation,” simply wastes time. This is shown in Example 3-16.

Loop inside loop delay

Another way to get a large delay is to use a loop inside a loop, which is also called a *nested loop.* See Example 3-17.

For an 8051 system of 11.0592 MHz, find the time delay for the following subroutine:

```
                                  Machine Cycle
DELAY:    MOV   R3,#250               1

HERE:     NOP                         1
          NOP                         1
          NOP                         1
          NOP                         1
          DJNZ  R3,HERE               2

          RET                         2
```

Example 3-16

Solution:

The time delay inside the HERE loop is [250(1 + 1 + 1 + 1 + 2)] x 1.085 us =1500 x 1.085 us = 1627.5 us. Adding the two instructions outside the loop we have 1627.5 us + 3xl.085|is= 1630.755 us.

If machine cycle timing is critical to your system design, make sure that you check the manufacture's data sheets for the device specification. For example, the DS89C420 has 3 machine cycles instead of 2 machine cycles for the RET instruction.

Example 3-17

For a machine cycle of 1.085 μs, find the time delay in the following subroutine.

```
DELAY:                            Machine Cycle
          MOV   R2,#200               1
AGAIN:    MOV   R3,#250               1
HERE:     NOP                         1
          NOP                         1
          DJNZ  R3,HERE               2
          DJNZ  R2,AGAIN              2
          RET                         2
```

For the HERE loop, we have (4 x 250) xl.085 us = 1085 us. The AGAIN loop repeats the HERE loop 200 times; therefore, we have 200 x 1085 us = 217000, if we do not include the overhead. However, the instructions "MOV R3,#250″ and "DJNZ R2, AGAIN" at the beginning and end of the AGAIN loop add (3 x 200 x 1.085 us) = 651 us to the time delay. As a result we have 217000 + 651 = 217651 us = 217.651 milliseconds for total time delay associated with the above DELAY subroutine. Notice that in the case of a nested loop, as in all other time delay loops, the time is approximate since we have ignored the first and last instructions in the subroutine.

Delay calculation for other versions of 8051

In creating a time delay using Assembly language instructions, one must be mindful of two factors that can affect the accuracy of the delay.

products from Philips Semiconductors have the option of using either 6 or 12 clocks per machine cycle. Table 3-2 shows some of the 8051 versions with their machine cycles.

1. **The crystal frequency:** The frequency of the crystal oscillator connected to the XI – X2 input pins is one factor in the time delay calculation. The duration

of the clock period for the machine cycle is a function of this crystal frequen cy.

2. **The 8051 Design:** Since the original 8051 was designed in 1980, both the field of 1C technology and the architectural design of microprocessors have seen great advancements. Due to the limitations of 1C technology and limited CPU design experience at that time, the machine cycle duration was set at 12 clocks. Advances in both 1C technology and CPU design in recent years have made the 1-clock machine cycle a common feature of many new 8051 chips. Indeed, one way to increase the 8051 performance without losing code compatibility with the original 8051 is to reduce the number of clock cycles it takes to exe cute an instruction. For these reasons, the number of machine cycles and the number of clock periods per machine cycle varies among the different versions of the 8051 microcontrollers. While the original 8051 design used 12 clock periods per machine cycle, many of the newer generations of the 8051 use much fewer clocks per machine cycle. For example, the DS5000 uses 4 clock periods per machine cycle while the DS89C4xO uses only one clock per machine cycle. The 8051

TaDie 3-2: Clocks per Machine Cycle (MC) for Various 8051 Versions

Chip/Maker	Clocks per Machine Cycle
AT89C51 Atmel	12
P89C54X2 Philips	6
DS5000 Dallas Semi	4
DS89C420/30/40/50 Dallas Semi	1

Example 3-18

From Table 3-2, find the period of the machine cycle (MC) in each case if XTAL = 11.0592 MHz, and discuss the impact on performance.
(a) AT89C51 (b) P89C54X2 (c) DS5000 (d) DS89C4x0

Solution:

(a) 11.0592 MHz/12 = 921.6 kHz; MC is 1/921.6 kHz = 1.085 µs (microsecond) = 1085 ns
(b) 11.0592 MHz/6 = 1.8432 MHz; MC is 1/1.8432 MHz = 0.5425 µs = 542 ns
(c) 11.0592 MHz/4 = 2.7648 MHz; MC is 1/2.7648 MHz = 0.36 µs = 360 ns
(d) 11.0592 MHz/1 = 11.0592 MHz; MC is 1/11.0592 MHz = 0.0904 µs = 90 ns

This means that if we connect an AT89C51 and a DS89C4x0 to a crystal of the same frequency we get approximately 9 to 10 times performance boost for the DS89C4x0 chip over the AT89C51. See Example 3-20.

Delay calculation for DS89C4xO

In the case of the DS89C4xO, since the number clocks per machine cycle was reduced from 12 to 1, the number of machine cycles used to execute an instruction had to be changed to reflect this reality. Table 3-3 compares the machine cycles for the DS89C4xO and 8051 for some instructions.

Example 3-19

For an AT8051 and DS89C420/30/40/50 system of 11.0592 MHz, find how long it takes to execute each of the following instructions.

(a) `MOV R3,#55` (b) `DEC R3` (c) `DJNZ R2,target`
(d) `LJMP` (e) `SJMP` (f) `NOP` (no operation)
(g) `MUL AB`

Solution:

The machine cycle time for the AT8951 and DS89C420/30 was shown in Example 3-18. Table 3-3 shows machine cycles for each of the above instructions. Therefore, we have:

Instruction	*AT8051*	*DS89C420/30/40/50*
(a) MOV R3,#55	1×1085 ns = 1085 ns	2×90 ns = 180 ns
(b) DEC R3	1×1085 ns = 1085 ns	1×90 ns = 90 ns
(c) DJNZ R2,..	2×1085 ns = 2170 ns	4×90 ns = 360 ns
(d) LJMP	2×1085 ns = 2170 ns	3×90 ns = 270 ns
(e) SJMP	2×1085 ns = 2170 ns	3×90 ns = 270 ns
(f) NOP	1×1085 ns = 1085 ns	1×90 ns = 90 ns
(g) MUL AB	4×1085 ns = 4340 ns	9×90 ns = 810 ns

Table 3-3: Comoarison of 8051 and DS89C4xO Machine Cvcles

Instruction	8051	DS89C4x0
MOV R3,#value	1	2
DEC Rx	1	1
DJNZ	2	4
LJMP	2	3
SJMP	2	3
NOP	1	1
MUL AB	4	9

Example 3-20

Find the time delay for the loop section of the following subroutine if it is run on a DS89C420/30 chip, assuming a crystal frequency of 11.0592 MHz.

```
                                DS89C420/30 Machine Cycle
DELAY:     MOV  R3,#250

HERE:      NOP                  1
           NOP                  1
           NOP                  1
           NOP                  1
           DJNZ R3,HERE         4

           RET
```

Solution:

The time delay inside the HERE loop is $[250(1 + 1 + 1 + 1 + 4)] \times 90$ ns $= 2000 \times 90$ ns $= 180$ µs. Comparing this with Example 3-16, we see DS89C4x0 is about 9 times faster. (1627 µs / 180 µs = 9)

Example 3-21

Write a program to toggle all the bits of P1 every 200 ms. Assume that the crystal frequency is 11.0592 MHz, and that the system is using the AT89C51.

Solution:

```
;Tested for AT89C51 of 11.0592 MHz.

           MOV    A,#55H
AGAIN:     MOV    P1,A
           ACALL  DELAY
           CPL    A
           SJMP   AGAIN

;----Time delay
DELAY:     MOV    R5,#2
HERE1:     MOV    R4,#180
HERE2:     MOV    R3,#255
HERE3:     DJNZ   R3,HERE3
           DJNZ   R4,HERE2
           DJNZ   R5,HERE1
           RET
```

$2 \times 180 \times 255 \times 2$ MC $\times 1.085$ µs $= 199{,}206$ µs

Write a program to toggle all the bits of P1 every 200 ms. Assume crystal frequency is 11.0592 MHz and the system is using DS89C420/30/40/50.

Solution:

```
;Tested for DS89C420 of 11.0592 MHz.
          MOV   A,#55H
AGAIN:    MOV   P1,A
          ACALL DELAY_200m
          CPL   A
          SJMP  AGAIN


;----Time delay
DELAY_200m:
          MOV   R5,#9
HERE1:    MOV   R4,#242
HERE2:    MOV   R3,#255
HERE3:    DJNZ  R3,HERE3
          DJNZ  R4,HERE2
          DJNZ  R5,HERE1
          RET
```

Delay $9 \times 242 \times 255 \times 4$ MC $\times$ 90 ns = 199,940 µs

Use an oscilloscope to measure the system square wave period to verify delay.

Example 3-22

From the above discussion we conclude that use of the instruction in generating time delay is not the most reliable method. To get more accurate time delay we use timers as described in Chapter 9. Meanwhile, to get an accurate time delay for a given 8051 microcontroller, we must use an oscilloscope to measure the exact time delay.

SJMP to itself using $ sign

In cases where there is no monitor program, we need to short jump to itself in order to keep the microcontroller busy. A simple way of doing that is to use the $ sign. That means in place of this

```
HERE:     SJMP HERE
```

we can use the following:

```
          SJMP $
```

SUMMARY

The flow of a program proceeds sequentially, from instruction to instruction, unless a control transfer instruction is executed. The various types of control transfer instructions in Assembly language include conditional and unconditional jumps, and call instructions.

The looping action in 8051 Assembly language is performed using a special instruction, which decrements a counter and jumps to the top of the loop if the counter is not zero. Other jump instructions jump conditionally, based on the value of the carry flag, the accumulator, or bits of the I/O port. Unconditional jumps can be long or short, depending on the relative value of the target address. Special attention must be given to the effect of LCALL and ACALL instructions on the stack.

Next post: I/O PORT PROGRAMMING

Previous post: 8051 CALL INSTRUCTIONS

- **Related Links**
  - 8051 Microcontroller
    - 8051 MICROCONTROLLERS
    - MICROCONTROLLERS AND EMBEDDED PROCESSORS
    - OVERVIEW OF THE 8051 FAMILY
    - 8051 ASSEMBLY LANGUAGE PROGRAMMING
    - INSIDE THE 8051

- **:: Search WWH ::**