Electrical and Computer Engineering

EE3613 Processor Systems Laboratory

LAB 11

Timer Programming for AVR microcontroller

Fall 2019

OBJECTIVES:

- To learn how to use C language to program AVR Microcontroller
- To learn how to program ATmega324PB Xplained Pro board using C
- To learn how to use IO ports on the board using C
- To learn Timer programming using C and Assembly

REFERENCES:

Mazidi and Naimi, "The AVR Microcontroller and Embedded Systems," Chapter 9

MATERIALS:

Atmel Studio 7, ATmega324PB Xplained PRO Board

BACKGROUND INFORMATION:

Using C language to program AVR microcontroller is easier and flexible for the programmers even though it takes more code memory space. To use C programming effectively, the programmers must consider the size of the data used in the program and know how to use the function libraries that correspond to the task and the devices used.

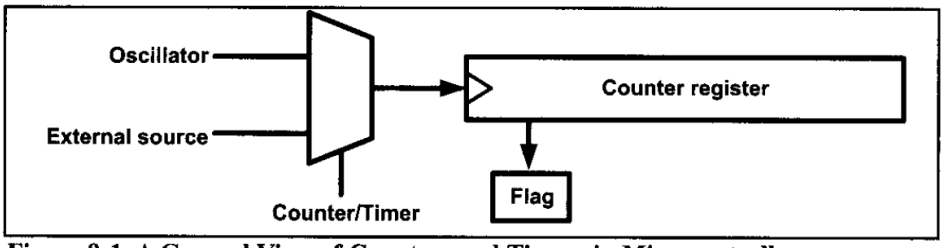

Figure 9-1. A General View of Counters and Timers in Microcontrollers

In this lab, we are going to learn how to use timers and counters. Many applications need to count an event or generate time delays. So, there are counter registers in microcontrollers for this purpose. When we want to count an event, we connect the external event source to the clock pin of the counter register (see **figure 9-1**). When an event occurs externally, the content of the counter is incremented. This way, the counter register represents how many ticks or signal state transitions have occurred from the time we cleared the counter. Since the speed of the microcontroller oscillator is known, we can calculate the tick period, and from the content of the counter register we will know how much time has elapsed.

So, one way to generate a time delay is to clear the counter at the start time and wait until the counter reaches a certain number. For example, consider a microcontroller with an oscillator with frequency of 1MHz; in the microcontroller, the content of the counter register increments once per microsecond. So, if we want a time delay of 100 microseconds, we should clear the counter and wait until it becomes equal to 100.

The AVR microcontrollers usually have 1 to 6 timers (built in) depending on the family member. They are referred to as Timers 0,1,2, … and 6. They can be used as timers to generate time delay or as counter to count events happening outside the microcontroller. In AVR microcontroller, some of the times are 8-bits, some of them are 16-bits. ATmega324PB has two 8-bit timer/counters and three 16-bit timer/counters with separate Prescaler, compare mode and capture mode.

Table 17-9. Waveform Generation Mode Bit Description

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCR0x at	TOV Flag Set on[1][2]
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	-	-	-
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	-	-	-
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Table 17-9 from Atmel datasheet for ATmega324PB shows different mode of operation of the timer/counter. A complete reference of these settings can be found in the datasheet.

Link: http://ww1.microchip.com/downloads/en/DeviceDoc/40001908A.pdf

In this following example, we will toggle all the bits of PORTA continuously with some delay. We will use Timer0, Normal mode and no Prescaler options the generate the delay. You may change the initial timer value to observe the change in time period/frequency. Use oscilloscope probe to connect to any of the PORTA bits to see the waveform.

Code 1 (Timer) in C:

```
#include <avr/io.h>

int main(void)
{

DDRA = 0xff;

        while(1)
        {
        // Load TCNT0 with initial timer value (Timer/Counter 0 register)
        TCNT0 = 0xF2; // 0x00 gives 33.6 micro sec, 0xFF gives 1.75 micro sec
        TCCR0B = 0x01; // Timer0, Normal mode, no Prescaler
        while((TIFR0&(1<<TOV0))==0);  //while TIFR0 bit0 is 0
        TCCR0B = 0;
        TIFR0 = (1<<TOV0); // Clear TIFR0 (Timer Flag Register)
        PORTA = ~PORTA;
        }
}
```

Code 1 (Timer) in Assembly:

```
;.INCLUDE "M324pbDEF.INC"
LDI R16,0xFF
OUT DDRA,R16
BEGIN:LDI R20,0xF2
```

```
OUT TCNT0,R20;load timer0
LDI R20,0x01
OUT TCCR0B,R20 ;Timer0,Normal mode,int clk
AGAIN:IN R20,TIFR0;read TIFR
SBRS R20,0 ;Skip if Bit in Register is set if TOV0 is set skip next inst.
RJMP AGAIN
LDI R20,0x0
OUT TCCR0B,R20;stop Timer0
LDI R20,(1<<TOV0);R20 = 0x01
OUT TIFR0,R20;clear TOV0 flag
EOR R17,R16;toggle A5 of R17
OUT PORTA,R17;toggle PA5
RJMP BEGIN
```

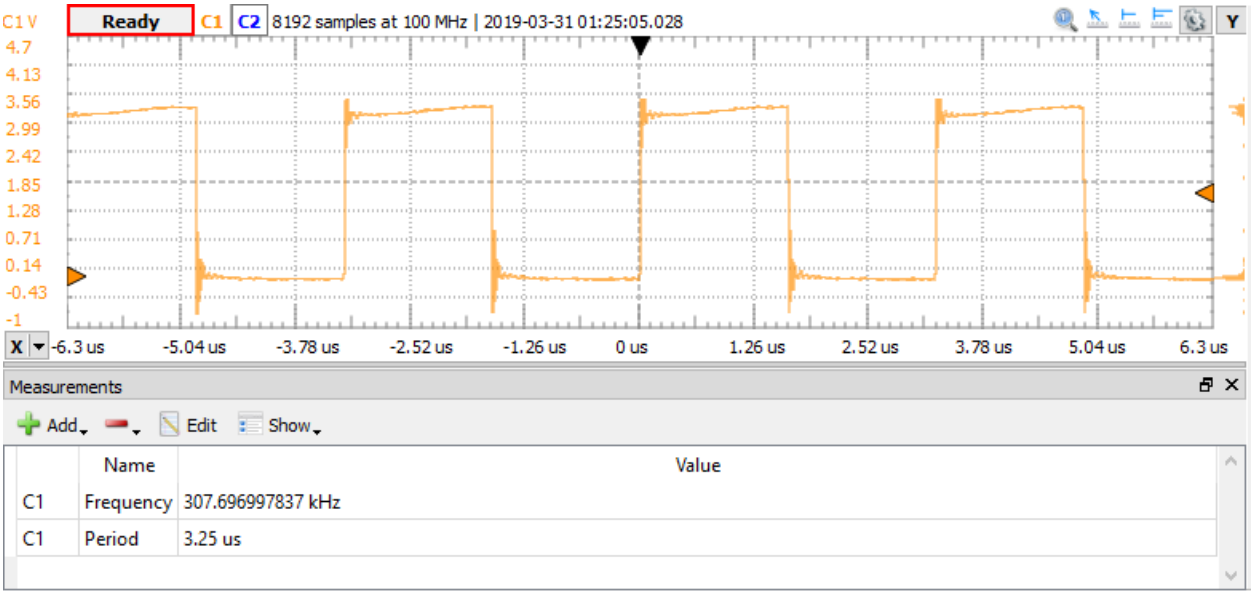

Figure: Square wave output from PORT A on oscilloscope

ACTIVITY

Part I. Programming Timers in C:

a) Program Timer0 in C to generate a square wave of 3kHz. Assume that, XTAL = 16MHz.
b) Program Timer0 in C to generate a square wave of 1kHz. Assume that, XTAL = 16MHz.

Tips: Use NI Virtual Bench to see the waveforms. Take necessary screenshots and measurements for the report.

The report should contain for Part I:

- Explanation of all functions used in Activity (a) and (b)
- Necessary modifications you needed to make on the example code for this task
- Flow chart of the code operation (same for both tasks)
- Screenshots of the port outputs on oscilloscope in the results section
- Complete code in the appendix and txt file

Part II. Programming Timers in Assembly:

a) Program Timer0 in assembly to generate a square wave of 3kHz. Assume that, XTAL = 16MHz.
b) Program Timer0 in assembly to generate a square wave of 1kHz. Assume that, XTAL = 16MHz.

Tips: Use NI Virtual Bench to see the waveforms. Take necessary screenshots and measurements for the report.

<u>The report should contain for Part II:</u>

- Explanation of all functions used in Activity (a) and (b)
- Necessary modifications you needed to make on the example code for this task
- Flow chart of the code operation (same for both tasks)
- Screenshots of the port outputs on oscilloscope in the results section
- Complete code in the appendix and txt file

REPORT

Lab 11 requires the full-length report. The report must include the following sections and details:
1. Cover page (5pts): your information must be shown
2. Introduction (15pts)
   - Objectives
   - Any related knowledge about Timer/Counter and the programming to use them
3. Procedure (30pts)
   - Flowchart: Activity (only for Part I – (a) and (b))
   - Method: description of the given tasks based on your understanding and your approaches to make the solutions / Explanation of the required procedure and the expected result including the calculation for the timer after you follow the procedure
4. Results (30pts)
   - Screenshots and Description: you must explain the result with your own words and figures
   - Screenshots of measurement using an oscilloscope

5. Discussion (20pts)
   - Verification of the results with the expected result from the calculation in the procedure part. Also, compare the results of C program and of Assembly.
6. Conclusion (10pts)
   - Summary of key concepts of this lab
7. Appendix (10pts – code here and code text file submission both to get full credit)
   - Code for Part I – (a) and (b)
   - Code for Part II – (a) and (b)

Your codes must be saved in a .txt file, too. Then, submit the .txt file to the LAB 11 Code submission part separately. The lab report must be submitted by November 21, 2019, 11:59 pm (section 1) / November 23, 2019, 11:59 pm (section 2).

IMPORTANT: DO NOT COPY FROM LAB MANUAL OR ANY OTHER SOURCES >>

ECE3613 Processor System Laboratory Rubric
Lab #: 10
Section: 001 / 002
Name: _____

Report Section	Part	Contents	Full Points	Earned Points	Comment
Cover Page			5		Lab report Cover page
Introduction			15		Objective (5pts) and background information about Timer/Counter (10pts)
Procedure	I	Flowchart	5		
		Method	10		• Outline of the steps do complete the activities(5 pts) • Description of the code lines that are related to the timer setup and calculation for the timer (10 pts)
	II	Flowchart	5		
		Method	10		• Outline of the steps do complete the activities(5 pts) • Description of the code lines that are related to the timer setup and calculation for the timer (10 pts)
	Subtotal		**50**		
Result	I	Result	15		• Screenshot of the timer result in stopwatch (5pts) • Screenshot of the oscilloscope reading (10pts) **Note:** Explanation of the result must be presented
	II	Result	15		• Screenshot of the timer result in stopwatch (5pts) • Screenshot of the oscilloscope reading (10pts) **Note:** Explanation of the result must be presented
	Subtotal		**30**		
Discussion		Evaluation	10		Result Verification and Analysis of Activity (a) and (b) by comparison with the calculation (expected value)

		Evaluation	10		Result Verification and Analysis of the result from C and Assembly code
Conclusion			10		Summary of the lab
Appendix	1	Code	5		Full comments, text file submission
	2	Code	5		Full comments, text file submission
	Subtotal		**40**		
Total			**120**		