CSSE1000/CSSE7035 - Prac 8

AVR I/O - Timer/Counter & Tone Generator

Goal

- Gain more experience with AVR assembly language and C programming
- Learn about using hardware resources on microcontrollers
- Learn more about timers and counters, and their possible applications
- Gain a basic understanding of the compilation process.

Preparation

This prac will involve producing sound (tones) using a pizeo-electric speaker (or buzzer). Changing the voltage across the speaker, causes the piezo element to move. Changing it at some frequency causes the piezo element to vibrate at that frequency and produce sound waves at that frequency. For example, to produce sound at a frequency of 1kHz, we must move the piezo element back **and** forth 1000 times per second, i.e., change the voltage across it 2000 times per second, or once every 500 microseconds. The AVR 8515 we are using runs at 4MHz, i.e. the clock rate is 4 million clock cycles per second, or once every 0.25microseconds. 500 microseconds is equivalent to 2000 clock cycles on our 8515.

We will connect the pizeo speaker across two port pins of the AVR microcontroller (both configured as outputs) and toggle the output on one of the pins (meaning the voltage across the piezo speaker will change between 0 and 5 volts).

1. **Software Delay**

   Write an assembly language program for the AVR8515 Project Board, running at 4MHz, to toggle bit 0 of Port B at a frequency of 1KHz. The piezo speaker will be connected to pins 1 and 0 of Port B, so you will need to set pin 1 of Port B to output a 0. The skeleton code for this program is provided in prac8-1.asm and uses what is called a software delay - i.e. using a loop which counts up to some number. Consider using an exclusive-OR operation to perform the bit toggle. This code also illustrates the use of a procedure call in AVR assembly language.

   Try to determine how long the loop should be to make the delay procedure have a delay of 500 microseconds. (Consider how many cycles each instruction will take - look at the AVR Instruction Set Manual (687kB) or Reference (164kB) to see this.)

   To check this in the AVR simulator, consider using the Processor stopwatch (expand the "Processor" view within the "I/O" Workspace within AVR Studio 4). Add a break point to the code which toggles bit 0 of port B and run the program. The breakpoint should be reached every 500 microseconds (2000 clock cycles at our clock rate of 4MHz). Were your calculations correct? (If your delay procedure was exactly 2000 clock cycles, the time between stops at the breakpoint will be slightly greater than this time given that some time is taken to execute the other instructions in the main loop.)

   Adjust your delay loop so that the bit 0 or port B is toggled (i.e. the breakpoint is reached) every 500 microseconds. (You may need to add some NOP instructions to get the delay exact. NOP instructions do nothing (No Operation), but take one clock cycle to execute.)

   The following registers will be used:

DDRB	Port B Data Direction Register
PORTB	Port B Data Register

2. **Output Compare**

   Read pages 32 to 39 of the AT90S8515 datasheet (2.6MB) to get an overview of how the timers and counters work on the AVR AT90S8515. The 8515 has two built-in timer/counters - one 8-bit (named Timer/Counter0) and one 16-bit (named Timer/Counter1). This program will use the 16-bit (timer/counter1). The counter can be clocked (i.e. increments by 1) from a number of different clock sources (as shown in Table 10, page 37 of the datasheet). These options include counting on every clock cycle, every 8th clock cycle, every 64th clock cycle etc. This counter clock source is determined by the 3 least significant bits of the Timer/Counter1 Control Register B (TCCR1B) - an I/O register described on page 37 of the datasheet. (There is similar clock scaling available for timer 0 - see page 33 of the datasheet.) Timer/Counter 1 also allows the possibility of comparing the count value with some fixed values in the "Output Compare Registers". These 16 bit registers (OCR1A,OCR1B) are considered to be made up of two 8 bit halves (OCR1AH and OCR1AL make up OCR1A, OCR1BH and OCR1BL make up OCR1B). It is possible to configure the timer/counter to toggle, set or reset an output pin when the compare value is reached. This is controlled by the 4 most significant bits of the Timer/Counter1 Control Register A (TCCR1A) - see page 36 of the datasheet. The two output pins are labelled OC1A (shared with port D, bit 5) and OC1B. Another feature of the timer/counter we will use is the ability for it to reset it self when it reaches the value in the "A" output compare register (OCR1A). This is controlled by bit 3 (CTC1 bit) of the TCCR1B register (see page 37 of the datasheet).

   Write an assembly language program for the AVR8515 Project Board, running at 4MHz, to toggle the OC1A pin at a frequency of 1KHz using Output Compare 1. The piezo speaker will be connected to pin 5 (OC1A) and pin 4 of Port D, so the direction for these pins need to be set as output. Also, pin 4 of Port D needs to output a 0. The skeleton code for this program is provided in prac8-2.asm.

   This program will use the 16-bit timer/counter - in this case it will be set to count on every clock cycle, starting at value 0. We also configure the timer so that when it reaches the compare value (1999) we toggle the OC1A pin (pin 5 of port D) as well as start counting from 0 again. This will cause the OC1A pin to toggle every 2000 clock cycles (i.e. 500microseconds) resulting in an output signal with a period of 1000 microseconds (1 millisecond), i.e. a frequency of 1kHz. (The compare value is 1999 so that the count pattern repeats every 2000 clock cycles - the counter will count 0, 1, 2, ... 1998, 1999, 0, 1, ...) This counting and output pin toggling will be completely under hardware control - the software will be doing nothing during this time.

   The following registers will be used. The skeleton code contains page references to the datasheet regarding the use of these registers.

DDRD	Port D Data Direction Register
PORTD	Port D Data Register
TCCR1A	Timer/Counter 1 Control Register A
TCCR1B	Timer/Counter 1 Control Register B
OCR1AH	Timer/Counter 1 Output Compare Register (High Byte)
OCR1AL	Timer/Counter 1 Output Compare Register (Low Byte)

Optional Prep: Read through the instructions below and prepare a program for Part 3.

Procedure

Part 1 - Software Delay

Assemble the program that you have written and and program it into the device on the AVR8515 Project Board.

With your program in the device on the project board, carry out the following steps:

i. Switch off the power supply to the project board and logic workstation.
ii. **Connect the piezo speaker to pins 1 and 0 of Port B.**
iii. Switch on the power supply to the project board and logic workstation.

Part 2 - Output Compare

Assemble the program that you have written and and program it into the device on the AVR8515 Project Board.

With your program in the device on the project board, carry out the following steps:

i. Switch off the power supply to the project board and logic workstation.
ii. **Connect the piezo speaker to pins 5 and 4 of Port D**.
iii. Switch on the power supply to the project board and logic workstation.

Part 3 - Tone Generator in C (using Output Compare)

Write and test a C version of the program in part 2 (above). i.e. Write a C language program for the AVR8515 Project Board, running at 4MHz, to toggle the OC1A pin at a frequency of 1KHz using Output Compare 1. The skeleton code for this program is provided in prac8-3.c.

When we're interacting with hardware (timer/counters in this case) at the lowest level, C programs do similar things to assembly language programs, however we can use assignment statements to directly interact with IO registers and it is possible to treat 16 bit registers like OCR1A as a single entity rather than having to consider it as two halves.

Demonstrate your design as working to the tutor.

Part 4 - Analysis of Compiled C Programs

Before a C program can be implemented on the AVR it must be translated to assembly language. This process is know as **compilation.** When AVR builds your C programs, the compiled assembly program is available for viewing as part of the **list file** in the default directory. The list file is created every time you build the project.

Build your part 3 program and navigate to the list file (file extension is .lss) in the default directory (the same directory as the hex file). Here you should see your C code interleaved with the assembly implementation. You may need to scroll down to see it, i.e. look for (or search for) the section labeled *main* to see your code and its assembly implementation. Ignore the other sections of the list file for the moment, we will explain these concepts in a future lecture.

Compare the generated assembly for your part 3 program to your hand coded assembly for part 2. Note any similarities and differences. Write up your comparison in your workbook.

Tutor Task

The tutor will ask you specific questions about the differences between your compiled and hand written C programs and what possible factors may affect the compilation process. You will need to answer these questions in your workbook.

Challenge Tasks (Complete after you have been marked off)

1. Modify your answer to part 3 above so that the sound frequency can be adjusted by the user pressing one of three push buttons (0, 1 and 2) connnected to pins 0, 1 and 2 of port C. For example, if pushbutton 0 is pushed (i.e. 0), then the sound frequency output should be 2kHz. If pushbutton 1 is pushed, then the sound frequency output should be 1kHz.
2. Implement Part 1 (software timer/tone generator) using a C program instead of assembler. What are the complications of this? Discuss with the tutor at the end of the session or on the newsgroup.

Assessment

This practical is marked out of 4 and worth 2% of your mark for CSSE1000:

- Preparation - Programming task completed and documented - ***1 Mark***
- Demonstration - C tone-generator (Part 3) demonstrated. Part marks given if an earlier exercise is demonstrated - ***1 Mark***
- Documentation - Experimental process documented in workbook with an appropriate conclusion. Comparison exercise (Part 4) completed. - ***1 Mark***
- Tutor Task - Tutor task completed and questions answered in workbook - ***1 Mark***

Equipment

- Computer
- 12V Power Supply
- AVR8515 Project Board with STK200 Download Cable
- Piezo Speaker
- Logic Workstation
- Hookup Wire

References

- Atmel AVR Resources
