CSSE1000/CSSE7035 - Prac 9

AVR Interrupts & Serial I/O

Goal

- Gain more experience with AVR assembly language and C programming.
- Become familiar with assembly and C interrupt based programming

See the following topics:

- Preparation
- Procedure
- Equipment
- References

Back to Lab Experiment Index

Preparation

Stopwatch

You are to design a stopwatch on the AVR board. The stopwatch has:

- A 2 digit seven segment display for displaying the current time
- A push button (P) for stopping and starting the stopwatch

The system operates as follows: When the system is reset (I.E. the board is programmed or the reset button is pressed) the displayed time is cleared to 00. When the push button (P) is pressed the stopwatch starts counting up once every 10th of a second in decimal. i.e. the left hand digit on the display will display seconds elapsed, while the right hand digit displays 10ths of seconds. When P is pressed again, the stopwatch pauses at its current value. The stopwatch can be started and stopped again by continually pressing P. Once the timer reaches 9.9 seconds it stops counting and continues to display "99" until reset.

Complete this specification as you see fit and if needed consult the newsgroup to resolve any issues.

Base your code on the interrupts example given in lecture 9. Use a timer with an interrupt to measure the 0.1 second intervals and update the timer. Use an external interrupt to detect the push button press.

Implement your stopwatch in both assembler and C code. Compile your C program and compare the generated list file to your assembly program. Be prepared to answer questions about your code from the tutors.

(Optional prep) You may also wish to prepare a program for part 2 below.

Procedure

1. Stopwatch

Test both your C and assembler stopwatch programs and note your result in your workbook. Compare your design with your partners design. Show one of your stopwatch programs to the

marker (start programming part 2 if you are waiting for any length of time).

2. Interrupt-based Serial Input/Output

Modify the skeleton code in prac9-1.c to create an interrupt-based C language program to read characters from the AVR UART (serial port) and echo them back to the serial port except if a number (0 to 9) is received as input. In this case a word description is output ("zero" to "nine" - without the quotation marks) instead. This program illustrates interrupt-based serial input/output using the UART (Universal Asynchronous Receiver and Transmitter, i.e. serial port) and also has code which implements a circular buffer.

The following registers will be used:

UDR	UART Data Register
UCR	UART Control Register
UBRR	UART Baud Rate Register

Compile 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. Ensure a serial cable is connected between the computer and the serial port of the project board.
iii. Switch on the power supply to the project board and logic workstation.
iv. Press the red reset button on the project board.
v. Start a terminal emulator program on the PC (e.g. HyperTerminal, available in Start Menu -> Programs -> Accessories -> Communications -> HyperTerminal)
vi. **Configure HyperTerminal with the following settings:**
New Connection Name: Anything you like (e.g. prac9-2) (OK)
Connect via: COM1 (OK)
Port Settings:
Bits per second: 19200 (the same as defined in the AVR program)
Data bits: 8
Parity: None
Stop bits: 1
Flow control: None
vii. **Press the red reset button on the project board.** (Do this after HyperTerminal has been started - this ensures the terminal is appropriately initialised.)
viii. Type characters into the terminal window and see that they are echoed back (or words are echoed back in the case of numbers being typed).

You may want to try recompiling your program using a higher UART speed (baud rate) and see if you can communicate successfully with HyperTerminal (make sure you change the baud rate within HyperTerminal also). As described on pages 57 and 58 of the AT90S8515 datasheet, with a 4MHz clock rate, the AVR is unable to precisely match higher baud rates, e.g. if you set UBRR to 1, the AVR will use a baud rate of 125000baud - which is probably not close enough to the standard 115200baud to enable successful communication. Try it and see!

For our purposes, you can think of a "baud" as a bit per second.

Show your working serial I/O program to the tutor.

3. Tutor task

The tutor will give you an advanced programming task involving both serial communications and timer or external interrupts.

Assessment

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

- Preparation - Programming task completed and documented. Comparison exercise completed and documented. Student is able to answer questions about their preparation if required. - ***1 Mark***
- Demonstration - Both stopwatch and serial I/O programs demonstrated - ***1 Mark***
- Documentation - Experimental process documented in workbook with an appropriate conclusion. - ***1 Mark***
- Tutor Task - Tutor task completed, demonstrated and documented in workbook - ***1 Mark***

Equipment

- Computer
- 12V Power Supply
- AVR8515 Project Board with STK200 Download Cable
- Piezo Speaker
- Logic Workstation
- Hookup Wire
- Serial Cable to connect the AVR8515 Project Board to the PC

References

- Atmel AVR Resources
