EE443 - Embedded Systems

Experiment 3

Edge Detection and Delay Generation

Purpose: Detect signal transitions at the MCU port inputs to control the LED rotation (as opposed to the level-sensitive control in the previous experiment). Use delay functions for periodic operations.

Programming Hints

It is always a good programming practice to read an input port only once before performing a series of operations on it. The input value must be stored in a variable, and the stored value must be used in all calculations or status checks in an execution cycle. Otherwise, the program may produce unexpected results if the external inputs change between the operations. As an example, consider the algorithm given below:

```
while (1)
{ if (PORTA == 0) then  // read PORTA here
  { <execute task-1>
    ---
  }
  if (PORTA == 1) then  // read PORTA again
  { <execute task-2>
    ---
  }
};
```

If the **PORTA** status changes between the two "**if (PORTA ==...**" statements, then the program may make inconsistent decisions in a loop cycle. Unless there are specific conditions that require repeated checks on the **PORTA** status, the following algorithm should be used to obtain consistent results.

```
while (1)
{ PORTAsave = PORTA ; // read PORTA only once
  if (PORTAsave == 0) then  // use the stored value
  { <execute task-1>
    ---
  }
  if (PORTAsave == 1) then  // use the same stored value
  { <execute task-2>
    ---
  }
};
```

The programming mistakes like the one described above usually cause occasional errors that are hard to detect. These errors may cause serious problems even if they occur once in a million cycles.

Preliminary Work

1. Device an algorithm to detect **1-to-0** transitions either at pin-0 or pin-1 of **port-D** of ATmega328p. The algorithm will be repeated in the main loop reading the port status once in every loop cycle. You must compare the current port bits with the port bits read in the previous loop cycle.

Procedure

1. Create a new project directory for Experiment 3 on your hard disk following the suggestions given in Experiment 1.

2. Open the design file for Experiment 1, **Exp1_LED.dsn**, in **ISIS**, and save it with a different name in the new project directory using the "**File>Save Design As...**" menu option.

3. Start **Code::blocks**, and create a new AVR project in the Experiment 3 directory following the guidelines given for Experiment 1. Modify the program you wrote for Experiment 2, so that the lighted LED position will shift one step immediately after a **1-to-0** transition at pin-0 of port-D (i.e. every time **SW0** is pressed). The rotation direction is determined by **SW1** as it was specified for Experiment 2.

- If **SW1** is pressed then LED light rotates **right**:
- If **SW1** is not pressed then LED light rotates **left**:

Compile your program and debug if necessary.

4. Open the **Edit Properties** window for **ATMEGA328P** microcontroller on the ISIS schematic and select the **.hex** file generated by the compiler for Experiment-3. Test your program looking at the animated simulation results, and debug it, if necessary.

5. Modify the program again as described below.

- SW0 output will be checked once every 1 ms.
- LED light will move one step left or right (according to **SW1** position) every time **SW0** is pressed.
- If the **SW0** output remains at **0** for longer than **250 ms**, then the LED light rotation will repeat, one step at every 250 ms.

You can use the WinAVR library function, **`_delay_ms(1)`**, to set the loop repetition rate. Test your program looking at the animated simulation results, and debug if necessary.

6. Measure the time for each LED rotation step. Keep the LED light rotating for at least 10 full cycles, and find the elapsed time looking at the animation time index at the bottom of the ISIS window. Calculate the time for one step, and make the necessary corrections to obtain the specified timing.

7. Write a C function that returns after **1 ms** delay when it is called. Modify your main program to call your delay function instead of the WinAVR library function. Measure the time for each LED rotation step with your delay function.

Note that, WinAVR code optimizer eliminates the "**for**" loops that does not produce any results. The delay loop should contain a statement, such as:

```
if (PINB == 0xFF)  PORTB = 0x00;
```

As an alternative, you can also use the **NOP** (No Operation) instruction by entering an inline assembly statement in the C code:

```
__asm__ volatile ( "NOP" );
```

8. Open the extended listing (**.lss**) file generated by the compiler and locate the assembly code of your delay function. Calculate the number of clock cycles required to complete one cycle of the delay loop referring to the **Instruction Set Summary** given on page-522 of ATmega328 datasheet. Make the necessary corrections to obtain **1 ms** delay and measure the LED rotation step time again.