CPE301 – SPRING 2019

Design Assignment DA2a

Student Name: Daniel Senda
Student #: 5002362633
Student Email: sendad1@unlv.nevada.edu
Primary Github address: https://github.com/dsenda/Smiles
Directory: DA2a

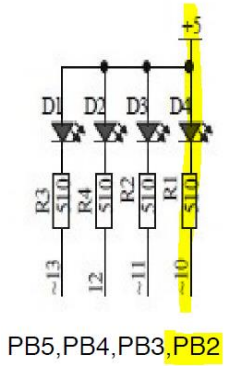
Submit the following for all Labs:

1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

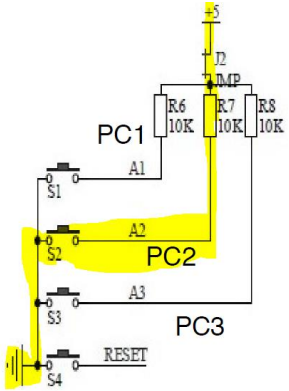
List of Components used:
Atmel Studio 7
ATmega328P Xplained Mini
Arduino Shield.

Block diagram with pins used in the Atmega328P:
Pin 2 of Port B was used to light up LED



PB5,PB4,PB3,PB2

Pin 2 of Port C was used (Switch)



2. INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A

The following is the **first assembly code** that creates a square waveform with period of 0.725s and 60% duty cycle.

```
; assm_code1_waveform.asm
; Daniel Senda
; Square waveform with period of 0.725s and 60% duty cycle.

.org 0                      ;Sets origin of code.
LDI R16, 0b00000100  ;Represents pin 2 of port b (PB2).
OUT DDRB,R16         ;Enables PB2 as output.
LDI R17, 0b00000100  ;Mask used to toggle PB2.
LDI R20,5                   ;Sets clock prescaler to 1024.
STS TCCR1B,R20              ;Sets clock division factor of 32.

main:                               ;Main loop of waveform.
       RCALL delay1         ;Calls delay1 subroutine.
       EOR R17,R16                  ;XOR to toggle LED.
       OUT PORTB,R17        ;Outputs value on LED.
       RCALL timer_reset    ;Calls timer_reset subroutine.
       RCALL delay2         ;Calls delay2 subroutine.
       EOR R17,R16                  ;XOR to toggle LED.
       OUT PORTB,R17        ;Outputs value on LED.
       RCALL timer_reset    ;Calls timer_reset subroutine.
       RJMP main                    ;Jump to top of main and repeat.

timer_reset:         ;This subroutnie resets the timer.
       LDI R20,0x00  ;Sets the counter to 0.
       STS TCNT1H,R20       ;Stores 0 in high byte.
```

```
       STS TCNT1L,R20       ;Stores 0 in low byte.
       RET                         ;Returns to main.

delay1:                            ;Creates a delay of 435ms (Ton value).
       LDS R29,TCNT1H       ;Loads upper byte of counter to R29.
       LDS R28,TCNT1L       ;Loads lower byte of counter to R28.
       CPI R28,0x8C  ;Compares if lower byte of timer is 0x8C.
       BRSH delay1a  ;Branch to delay1a if lower byte has reached desired amount.
       RJMP delay1          ;If not equal, jump back to top of delay1.
delay1a:                   ;Subroutnie to check upper byte of timer.
       CPI R29,0x1A  ;Compares if upper byte of timer is 0x1A.
       BRLT delay1          ;If not equal, jump back to delay1.
       RET                         ;If equal, return to main.

delay2:                            ;Creates a delay of 290ms (Toff value).
       LDS R29,TCNT1H       ;Loads upper byte of counter to R29.
       LDS R28,TCNT1L       ;Loads lower byte of counter to R28.
       CPI R28,0xB2  ;Compares if lower byte of timer is 0xB2.
       BRSH delay2a  ;If lower byte is equal to 0xB2, jump to delay2a.
       RJMP delay2          ;If not equal, jump back to top of delay2.
delay2a:                   ;Subroutine to check upper byte of timer.
       CPI R29,0x11  ;Compares if upper byte of timer is 0x11.
       BRLT delay2          ;If not equal, jump back to delay2.
       RET                         ;If equal, return to main.
```

The following is the **first C code** that accomplishes the same task.

```
// c_code1_waveform.c
// Daniel Senda
// Square waveform with period of 0.725s and 60% duty cycle.

#define              F_CPU  16000000
#include      <avr/io.h>
#include      <util/delay.h>
int num = 1;                       // Used for while loop.

int main(void)
{
   DDRB = 0b100;            // Enables PB2 as output.
   while(num = 1)                  // Loop used to create waveform.
   {
          PORTB = 0xFB;            // Sets PB2 to 0.
          _delay_ms(435);          // Turns LED on for 435ms (Ton).
          PORTB = PORTB^0b100;     // Sets PB2 to 1.
          _delay_ms(290);          // Turns LED off for 290ms (Toff).
   }
   return 0;
}
```

Next is the **second assembly code** that turns on an LED for 1.25 seconds after the pushbutton is pressed.

```
; assm_code2_switched_LEDpulse_1_25s.asm
; Daniel Senda
; Turn on LED for 1.25s after pushbutton is pressed.

.org 0                             ;Sets origin of code.
LDI R16, 0b00000100  ;Represents pin 2 of port b (PB2).
```

```
LDI R21, 0b00000000 ;Represents pin 2 of port c (PC2).
OUT DDRB,R16        ;Enables PB2 as output.
OUT DDRC, R21       ;Enables PC2 as input.
LDI R17, 0b00000100 ;Mask used to toggle PB2.
LDI R20,5                   ;Sets clock prescaler to 1024.
STS TCCR1B,R20              ;Sets clock division factor of 32.
OUT PORTB, R17              ;Resets LED, Turns it off.

main:                               ;Main loop.
      IN R22, PINC          ;Inputs values of PINB onto R22.
      ANDI R22, 0b100               ;Mask, used to isolate the value of PC2.
      CPI R22, 0b100                ;Compares R22 with 0b100
      BREQ main                     ;Branch to top of main if not equal.
      EOR R17,R16                   ;If equal, XOR to toggle LED on.
      OUT PORTB,R17         ;Outputs value on LED.
      RCALL timer_reset     ;Calls timer_reset subroutine.
      RCALL delay1          ;Calls delay1 subroutine.
      RJMP main                     ;Jumps back to top of main.

timer_reset:          ;This subroutnie resets the timer.
      LDI R20,0x00  ;Sets the counter to 0.
      STS TCNT1H,R20        ;Stores 0 in high byte.
      STS TCNT1L,R20        ;Stores 0 in low byte.
      RET                           ;Returns to previous position in main.

delay1:                             ;Creates a delay of 1.25s (Ton value).
      LDS R29,TCNT1H        ;Loads upper byte of counter to R29.
      LDS R28,TCNT1L        ;Loads lower byte of counter to R28.
      CPI R28,0x4A  ;Compares if lower byte of timer is 0x4A.
      BRSH delay1a  ;Branch to delay1a if lower byte has reached desired amount.
      RJMP delay1           ;If not equal, jump back to top of delay1.
delay1a:                    ;Subroutnie to check upper byte of timer.
      CPI R29,0x4C  ;Compares if upper byte of timer is 0x4C.
      BRLT delay1           ;If not equal, jump back to delay1.
      EOR R17,R16           ;If equal, XOR to toggle LED off.
      OUT PORTB,R17 ;Outputs value on LED.
      RET                           ;Returns to previous position in main.
```

Lastly, the following is the **second C code** that does the same task as the second assembly code.

```
// c_code2_switched_LEDpulse_1_25s.c
// Daniel Senda
// Turn on LED for 1.25s after pushbutton is pressed.

#define             F_CPU  16000000
#include      <avr/io.h>
#include      <util/delay.h>
int num = 1;                        // Used for while loop.
int comp;                                 // Used to compare in loop.

int main(void)
{
      DDRB = 0b100;         // Enables PB2 as output.
      DDRC = 0x00;          // Enables PC2 as input.
      while(num = 1)              // Loop used to keep program running.
      {
            comp = PINB & 0b100; // Bit masking.
            if (comp = 0b100)           // If button is pressed, this will be true.
```

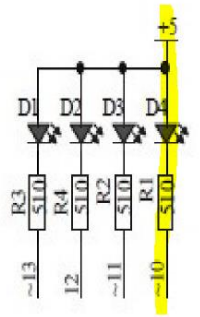
```
                {
                        PORTB = 0xFB;         // Sets PB2 to 0.
                        _delay_ms(1250);      // Turns LED on for 1.25s (Ton).
                        PORTB = PORTB^0b100;// Turns LED back off.
                }
        }
        return 0;
}
```

3. DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A

Not applicable.

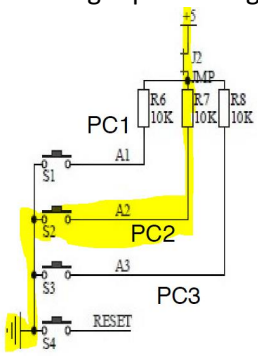
4. SCHEMATICS

For the first assembly and C code, the following schematic was used. PINB2 would go low to turn on the LED for 435ms and would go high to turn of the LED for 290ms.



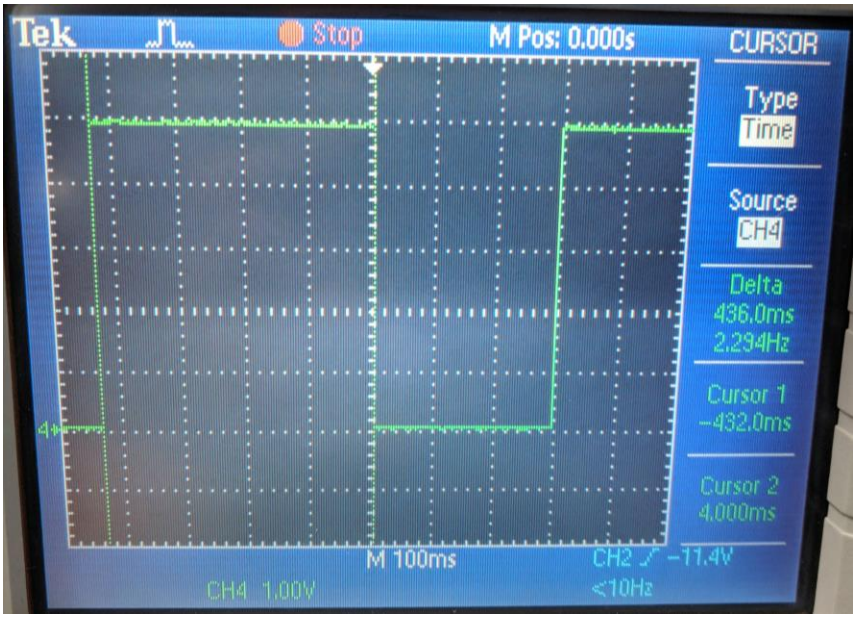
PB5,PB4,PB3,PB2

For the second assembly and C code, the same LED schematic was used as shown previously. In addition, the following switch was used, PINC2, with the jumper(J2) in place. When the switch would get pressed, the PC2 would go low and would in result turn on LED D4 for 1.25s. Then it would shut-off and would wait to get pressed again.



5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

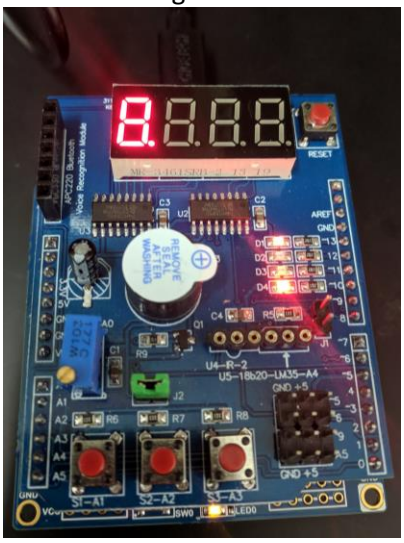
The following is a picture of the waveform that was created by the first assembly code. As can be seen, the output is almost exactly a perfect square waveform with period of 0.725s and 60% duty cycle.



The first C code also had an almost perfect waveform and came out to look exactly as the picture above.

6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

The follwowing is a screenshot of the board set-up. The shield sits on top of the xplainned mini.



7. VIDEO LINKS OF EACH DEMO

https://youtu.be/qSdWzfpsD6U (All of Demos are in one video.)

8. GITHUB LINK OF THIS DA

https://github.com/dsenda/Smiles/tree/master/DA2a

Student Academic Misconduct Policy

http://studentconduct.unlv.edu/misconduct/policy.html

"*This assignment submission is my own, original work*".

Daniel Senda