

Design Assignment DA2b

Student Name: Daniel Senda

Student #: 5002362633

Student Email: sendad1@unlv.nevada.edu

Primary Github address: <https://github.com/dsenda/Smiles>

Directory: DA2b

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:

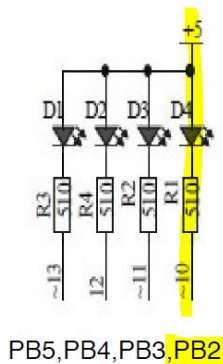
Atmel Studio 7

ATmega328P Xplained Mini

Arduino Shield.

Block diagram with pins used in the Atmega328P:

Pin 2 of Port B was used to light up LED



2. INITIAL/MODIFIED/DEVELOPED CODE OF TASK DA2A

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

```
; asmm_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 subrountie 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:                                ;Subroutine 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 subrountie 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:              ;Subrountie 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 DA2b from TASK DA2a

The design assignment 2b, the student had to modify the code from design assignment 2a. The code had to make use of INT0 (PD2 pin) as an interrupt mechanism.

The following is the **assembly code** that achieves the task successfully.

```

; DA2b.asm
; Daniel Senda
; Turn LED on for 1.25s after GPIO external pin interrupt is set.

JMP MAIN          ;Jumps to main.
.ORG 0x02         ;Location for external interrupt 0.
JMP EX0_ISR       ;Jumps to EX0_ISR.

MAIN:
    LDI R20,HIGH(RAMEND);Loads the high byte of RAMEND(the address of the last memory
                        ;location in SRAM) into R20.
    OUT SPH,R20      ;Stores the high byte of RAMEND onto the Stack Pointer High
                        ;register.
    LDI R20,LOW(RAMEND) ;Loads the low byte of RAMEND into R20.
    OUT SPL,R20      ;(Initializes stack)Stores the low byte of RAMEND onto the
                        ;Stack Pointer Low register.
    LDI R20,0x2      ;Loads a value of 0b10 into R20.
    STS EICRA,R20    ;Sets the EICRA(External Interrupt Control Register A) to
                        ;falling edge of INT0.
    LDI R18,0xFF      ;Loads 0xFF into R18, used to turn all LEDs off.
    OUT DDRB,R18      ;Sets port B pins as outputs.
    LDI R19,0xFB      ;Loads 0xFB into R19, used to turn LED on.
    OUT PORTB,R18     ;Turns all LEDs off.
    SBI PORTD,2       ;Pull-up activated.
    LDI R20,1<<INT0   ;Enables external interrupt 0 INT0.
    OUT EIMSK,R20     ;Stores the value R20 into EIMSK(External Interrupt Mask Reg).
    SEI              ;(Enables interrupts)Sets the Global Interrupt flag I in SREG.
HERE:JMP HERE        ;Waits for interrupt here (Wait loop).

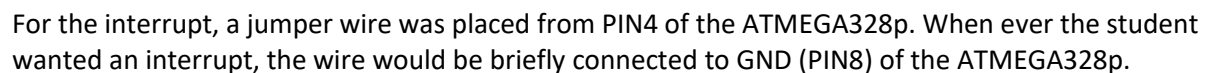
EX0_ISR:
    OUT PORTB,R19     ;Turns on LED connected to PB2.
    ldi R21, 51       ;Loads 51 to R21.
    ldi R22, 187      ;Loads 187 to R22.
    ldi R23, 224      ;Loads 224 to R23.
Loop1:               ;Delay of 1.250s.
    dec R23           ;Decrements R23.
    brne Loop1        ;Branches if not equal.
    dec R22           ;Decrements R22.
    brne Loop1        ;Branches if not equal.
    dec R21           ;Decrements R21.
    brne Loop1        ;Branches if not equal.
    OUT PORTB,R18     ;Loads the value of R21 onto PORTB.
    RETI             ;Returns from the interrupt.

```

The following is the **C code** that does the required tasks successfully.

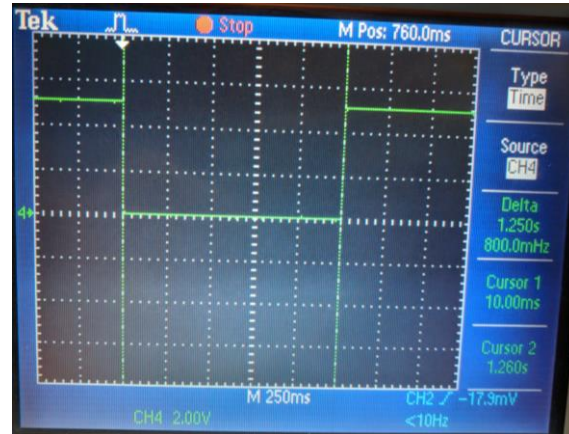
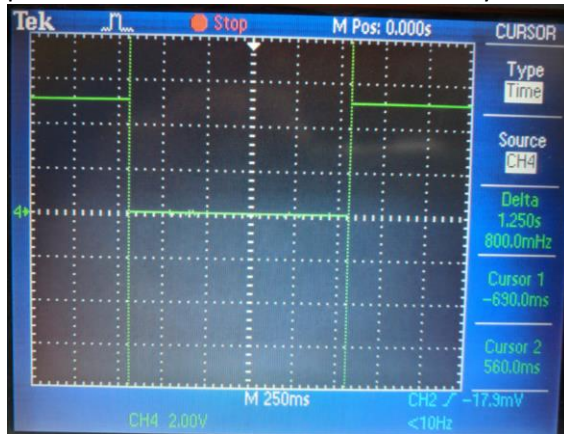
4. SCHEMATICS

ATmega8/48/88/168/328 DIP pinout



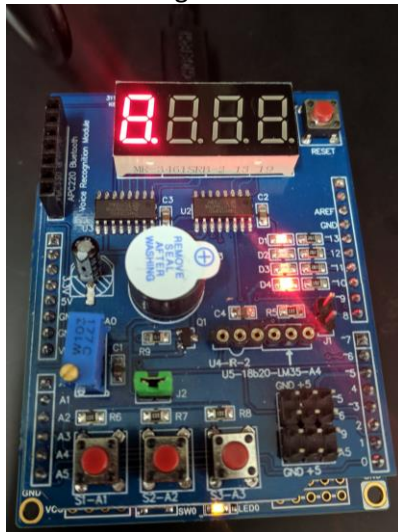
5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

The first picture is of the C code results showing that the LED turn on for exactly 1.25s. The second picture shows the same for the assembly code.



6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

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



7. VIDEO LINKS OF EACH DEMO

<https://youtu.be/jOsiyyCp6Es> (All of Demos are in one video.)

8. GITHUB LINK OF THIS DA

<https://github.com/dsenda/Smiles/tree/master/DA2b>

Student Academic Misconduct Policy

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

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

Daniel Senda