CPE301 – FALL 2019

Design Assignment 2A

Student Name: Cameron Kirk

Student #: 2000660135

Student Email: kirkc1@ubnlv.nevada.edu

Primary Github address: <https://github.com/kirkster96/submission_da>

Directory:

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

Block diagram with pins used in the Atmega328P

1. **INITIAL CODE OF TASK 1/A**

;

; DA2\_a.asm

;

; Created: 9/30/2019 10:07:47 PM

; Author : Kirks

;

; Replace with your application code

.INCLUDE "M328PBDEF.INC"

ldi r16,0x20

out DDRB,r16

out PortB,r16

Start:

rjmp Start

;Initial code to program the atmega328pb to turn on the LED0

1. **MODIFIED CODE OF TASK 1/A**

;

; DA2\_a.asm

;

; Created: 9/30/2019 10:07:47 PM

; Author : Kirks

;

.INCLUDE "M328PBDEF.INC"

CBI DDRB,7;make PB7 an input (SW0)

SBI DDRB,5

AGAIN: SBIC PINB,7;skip next if PB7 is clear

RJMP OVER;(JMP is OK too)

SBI PortB,5

RJMP AGAIN;we can use JMP too

OVER: CBi PORTB,5

RJMP AGAIN;we can use JMP too

;Modified to incorporate SW0 on board to control the LED0

1. **DEVELOPED CODE OF TASK 1/A**

;

; DA2\_a\_T1\_asm.asm

;

; Created: 10/2/2019 8:46:30 AM

; Author : Kirks

;

; Design a delay subroutine to generate a waveform on PORTB.3 with 40% DC and 0.625 sec period.

;0.25 sec delay. 16MHz clock. 0.0625us/instruction. 4000000 instructions where signal will be high

;6000000 instructions will be low

SBI DDRB,3

start:

CBi PortB,3

ldi R19,50 ; cycle = 1

delay3: ldi R20,255 ;cycle = 1 \* 50

nop ;cycle = 1 \* 50 \*255

delay4: ldi R21,154 ;cycle = 1 \* 50 \* 255

nop ;cycle = 1 \* 50 \* 255 \* 255

nop ;cycle = 1 \* 50 \* 255 \* 255

nop ;cycle = 1 \* 50 \* 255 \* 255

nop ;cycle = 1 \* 50 \* 255 \* 255

nop

nop ;cycle = 1 \* 50 \* 255 \* 255

delay5: dec R21 ;cycle = 1 \* 50 \* 255 \* 255

brne delay5 ;cycle = 2/1 \* 50 \* 255 \* 255 => 50\*255(254\*2)+50\*255(1\*1)

dec R20 ;cycle = 1 \* 50 \* 255

brne delay4 ;cycle = 2/1 \* 50 \* 255=> 50(254\*2)+105(1\*1)

dec R19 ;cycle = 1 \* 50 \* 255

nop

nop

brne delay3 ;cycle = 2/1 \* 50 => (49\*2)+(1\*1)

SBI PORTB,3

ldi R19,50 ; cycle = 1

delay0: ldi R20,255 ;cycle = 1 \* 50

nop ;cycle = 1 \* 50 \*255

delay1: ldi R21,102 ;cycle = 1 \* 50 \* 255

nop ;cycle = 1 \* 50 \* 255 \* 255

nop ;cycle = 1 \* 50 \* 255 \* 255

nop ;cycle = 1 \* 50 \* 255 \* 255

nop ;cycle = 1 \* 50 \* 255 \* 255

nop ;cycle = 1 \* 50 \* 255 \* 255

delay2: dec R21 ;cycle = 1 \* 50 \* 255 \* 255

brne delay2 ;cycle = 2/1 \* 50 \* 255 \* 255 => 50\*255(254\*2)+50\*255(1\*1)

dec R20 ;cycle = 1 \* 50 \* 255

nop

brne delay1 ;cycle = 2/1 \* 50 \* 255=> 50(254\*2)+105(1\*1)

dec R19 ;cycle = 1 \* 50 \* 255

nop

nop

nop

brne delay0 ;cycle = 2/1 \* 50 => (49\*2)+(1\*1)

jmp start

**C code developed based on assembly code**

#define *F\_CPU* 16000000UL

#include <asf.h>

#include <avr/io.h>

#include <util/delay.h>

int main (void)

{

/\* Insert system clock initialization code here (sysclk\_init()). \*/

board\_init();

DDRB |= (1<<3); //output PB3

PORTB |= (0<<3);//output PB3 set to zero initial

while(1){

PORTB |= (1<<2); //output PB2 on

*\_delay\_ms*(250);

PORTB &= ~(1<<2); // off\*/

*\_delay\_ms*(375);

}

return 1;

}

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

.INCLUDE "M328PBDEF.INC"

ldi r16,0x00

out DDRB, r16

out DDRC, r16

out DDRD, r16

out PortB,r16

out PortD,r16

out PortC,r16

CBI DDRC,3;make PC3 an input

CBI portC,3;make PC3 pull up resistor to active high

SBI DDRB,2

AGAIN: SBIC PINC,3;skip next if PC is clear

RJMP OVER;(JMP is OK too)

CBi PortB,2

RJMP AGAIN;we can use JMP too

OVER: SBI PORTB,2

RJMP AGAIN;we can use JMP too

/\*Code has been modified to use the Aurdino shield and the ports and pins have been changed to control what the assignment asks for\*/

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

;

; DA2\_a.asm

;

; Created: 9/30/2019 10:07:47 PM

; Author : Kirks

;

.INCLUDE "M328PBDEF.INC"

JMP start

DelaySubroutine:

ldi R19,108 ; cycle = 1

delay0: ldi R20,255 ;cycle = 1 \* 108

nop ;cycle = 1 \* 108

delay1: ldi R21,255 ;cycle = 1 \* 108 \* 255

nop ;cycle = 1 \* 108 \* 255

nop ;cycle = 1 \* 108 \* 255

nop ;cycle = 1 \* 108 \* 255

nop ;cycle = 1 \* 108 \* 255

nop ;cycle = 1 \* 108 \* 255

delay2: dec R21 ;cycle = 1 \* 108 \* 255 \* 255

brne delay2 ;cycle = 2/1 \* 108 \* 255 \* 255 => 105\*255(254\*2)+105\*255(1\*1)

dec R20 ;cycle = 1 \* 108 \* 255

brne delay1 ;cycle = 2/1 \* 108 \* 255=> 105(254\*2)+105(1\*1)

dec R19 ;cycle = 1 \* 108 \* 255

brne delay0 ;cycle = 2/1 \* 108 \* 255=> 105(254\*2)+105(1\*1)

ret ;cycle = 1

;1.33 sec delay. 16MHz clock. 0.0625us/instruction. 21280000 instructions is 1.33 seconds

start:

ldi r16,0x00

out DDRB, r16

out DDRC, r16

out DDRD, r16

out PortB,r16

out PortD,r16

out PortC,r16

CBI DDRC,3;make PC3 an input

CBI portC,3;make PC3 pull up resistor to active high

SBI DDRB,2

AGAIN: SBIC PINC,3;skip next if PC is clear

RJMP OVER;(JMP is OK too)

CBi PortB,2

call DelaySubroutine

RJMP AGAIN;we can use JMP too

OVER: SBI PORTB,2

RJMP AGAIN;we can use JMP too

**/\*final assembly code for task 2\*/**

**C code developed based on assembly code**

#define *F\_CPU* 16000000UL

#include <asf.h>

#include <avr/io.h>

#include <util/delay.h>

int main (void)

{

/\* Insert system clock initialization code here (sysclk\_init()). \*/

board\_init();

DDRB |= (1<<2); //output PB2

PORTB |= (0<<2);//output PB2 set to zero initial

DDRC |= (0<<3); // PC3 input

PORTC |= (1<<3);// PC3 input pullup resistor enable

while(1){

if(PINC & (1<<3)){

PORTB |= (1<<2); //output PB2 on

}

else if(PINC & (0<<3))

break;

else{

*\_delay\_ms*(1330);

PORTB &= ~(1<<2);} // off

}

return 1;

}

1. **VIDEO LINKS OF EACH DEMO**

Task 1 Assembly solution:

<https://drive.google.com/file/d/10wNXyd5wdjafd2GqtwRw9RYlxUSKP8H1/view?usp=sharing>

Task 2 C Solution:

<https://drive.google.com/file/d/10wsMakrtE0EElqiX2TiWnw_v6Vc6CQFg/view?usp=sharing>

Task 2 Assembly Solution:

<https://drive.google.com/file/d/10RkP0iyYxXMdQiIQ6YuEc0KZ19zHuEEZ/view?usp=sharing>

1. **GITHUB LINK OF THIS DA**

**Student Academic Misconduct Policy**

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

“This assignment submission is my own, original work”.

Cameron Kirk