Brandon Wade

CPE301 – SPRING 2016

Design Assignment 2

**DO NOT REMOVE THIS PAGE DURING SUBMISSION:**

The student understands that all required components should be submitted in complete for grading of this assignment.

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **SUBMISSION ITEM** | **COMPLETED (Y/N)** | **MARKS**  **(/MAX)** |
| 0. | COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS |  |  |
| 1. | INITIAL CODE OF TASK 1/A |  |  |
| 2. | INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B |  |  |
| 3. | INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C |  |  |
| 4. | INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D |  |  |
| 5. | INCREMENTAL / DIFFERENTIAL CODE OF TASK 5/E |  |  |
| 6. | SCHEMATICS |  |  |
| 7. | SCREENSHOTS OF EACH TASK OUTPUT |  |  |
| 8. | SCREENSHOT OF EACH DEMO |  |  |
| 9. | VIDEO LINKS OF EACH DEMO |  |  |
| 10. | GOOGLECODE LINK OF THE DA |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 0. | COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS |  |  |

Atmel Studio 7.0

ATMega328P Chip

Breadboard

LED Bar

Green LED

11x Resistors

GND

Resistor Array

R0

R1

R2

R3

R4

R5

R6

R7

R8

R9

R10

Green LED

LED Bar

LED0

LED1

LED2

LED3

LED4

LED5

LED6

LED7

LED8

LED9

ATmega328P

PB0

PB1

PB2

PB3

PB4

PB5

PB6

PB7

PC0

PC4

PC5

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | INITIAL CODE OF TASK 1/A |  |  |

;

; DA2.asm

; Task 1 ASM

; Created: 3/6/2016 11:36:31 AM

; Author : Brandon

;

LDI R16, 0x01

OUT DDRC, R16 ; PortC configured for output

LDI R16, 0x00 ; will hold PortC signals

OUT PORTC, R16 ; initialize PortC to 0

LDI R18, 0x01 ; will be used to toggle PC0

BEGIN:

RCALL DELAY ; wait for .25 seconds

EOR R16, R18 ; toggle signal

OUT PORTC, R16 ; output

RJMP BEGIN ; loop forever

DELAY:

LDI R23, 0x85

STS TCNT1H, R23

LDI R23, 0xEE

STS TCNT1L, R23 ; set initial count of timer

LDI R23, 0x00

STS TCCR1A, R23 ; normal mode

LDI R23, 0x03

STS TCCR1B, R23 ; start timer with 1024 ps

AGAIN:

IN R23, TIFR1 ; check for overflow

SBRS R23, TOV1 ; if overflow, exit loop

RJMP AGAIN ; loop till overflow

LDI R23, 0x00

STS TCCR1B, R23 ; turn off timer

LDI R23, 0x01

OUT TIFR1, R23 ; clear overflow flag

RET

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | INITIAL CODE OF TASK 1/B |  |  |

/\*

\* DA2T1C.c

\*

\* Created: 3/7/2016 10:47:19 AM

\* Author : Brandon

\*/

#define *F\_CPU* 8000000UL

#include <avr/io.h>

#include <util/delay.h>

int main(void)

{

DDRC |= 0x01; // PORTC configured to output mode

PORTC = 0x00; // PORTC initialized to 0

while (1)

{

TCNT1H = 0x85;

TCNT1L = 0xEE; // Set Count

TCCR1A = 0x00; // Normal Mode

TCCR1B = 0x03; // Set Pre Scalar

while((TIFR1 & 0x01) == 0); // Wait for flag

TCCR1A = 0x00;

TIFR1 |= 1<<TOV1; // Clear flag

PORTC ^= 0x01; // toggle PC0

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| 2. | INCREMENTAL CODE OF TASK 2/A |  |  |

;

; DA2.asm

; Task 2 ASM

; Created: 3/6/2016 11:36:31 AM

; Author : Brandon

;

LDI R16, 0xFF

OUT DDRB, R16 ; PortB configured for output

LDI R16, 0x01

OUT DDRC, R16 ; PortC configured for output

LDI R16, 0x00 ; will hold PortC signals

OUT PORTB, R16 ; initialize PortB to 0

OUT PORTC, R16 ; initizlize PortC to 0

LDI R17, 0x00 ; holds current count

LDI R18, 0x01 ; will be used to toggle PC0

BEGIN:

RCALL DELAY ; wait for .25 seconds

EOR R16, R18 ; toggle signal

OUT PORTC, R16 ; output

SBRS R16, 0

RJMP SKIP\_INC ; only increment on rising edge

INC R17 ; increment count

OUT PORTB, R17 ; output

SKIP\_INC:

RJMP BEGIN ; loop forever

DELAY:

LDI R23, 0x85

STS TCNT1H, R23

LDI R23, 0xEE

STS TCNT1L, R23 ; set initial count of timer

LDI R23, 0x00

STS TCCR1A, R23 ; normal mode

LDI R23, 0x03

STS TCCR1B, R23 ; start timer with 1024 ps

AGAIN:

IN R23, TIFR1 ; check for overflow

SBRS R23, TOV1 ; if overflow, exit loop

RJMP AGAIN ; loop till overflow

LDI R23, 0x00

STS TCCR1B, R23 ; turn off timer

LDI R23, 0x01

OUT TIFR1, R23 ; clear overflow flag

RET

|  |  |  |  |
| --- | --- | --- | --- |
| 2. | INCREMENTAL CODE OF TASK 2/B |  |  |

/\*

\* DA2T1C.c

\*

\* Created: 3/7/2016 10:47:19 AM

\* Author : Brandon

\*/

#define *F\_CPU* 8000000UL

#include <avr/io.h>

#include <util/delay.h>

int main(void)

{

DDRC |= 0x01; // PORTC configured to output mode

DDRB |= 0xFF;

PORTC = 0x00; // PORTC initialized to 0

PORTB = 0x00; // PORTB initialized to 0

while (1)

{

TCNT1H = 0x85;

TCNT1L = 0xEE; // set count

TCCR1A = 0x00; // normal mode

TCCR1B = 0x03; // set pre scalar

while((TIFR1 & 0x01) == 0); // wait for flag

TCCR1A = 0x00;

TIFR1 |= 1<<TOV1; // reset flag

PORTC ^= 0x01; // toggle PC0

if(PORTC & 0x01)

PORTB++; // increment count

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| 3. | INCREMENTAL CODE OF TASK 3/A |  |  |

;

; DA2.asm

; Task 3 ASM

; Created: 3/6/2016 11:36:31 AM

; Author : Brandon

;

LDI R16, 0xFF

OUT DDRB, R16 ; PortB configured for output

LDI R16, 0x61

OUT DDRC, R16 ; PortC configured for output

LDI R16, 0x00 ; will hold PortC signals

OUT PORTB, R16 ; initialize PortB to 0

OUT PORTC, R16 ; initizlize PortC to 0

LDI R17, 0x00 ; holds current count

LDI R18, 0x01 ; will be used to toggle PC0

LDI R19, 0x20 ; will be used to toggle PC5

LDI R20, 0x40 ; will be used to toggle PC6

LDI R21, 0x00 ; holds modulo 5

LDI R22, 0x00 ; holds modulo 10

BEGIN:

RCALL DELAY ; wait for .25 seconds

EOR R16, R18 ; toggle signal

OUT PORTC, R16 ; output

SBRS R16, 0

RJMP SKIP\_INC ; only increment on rising edge

INC R17 ; increment count

OUT PORTB, R17 ; output

INC R21 ; increment modulo 5

CPI R21, 5

BRNE SKIP\_INC ; only toggle on 5th rising edge

CLR R21 ; clear modulo 5

EOR R16, R19 ; toggle on 5th rising edge

OUT PORTC, R16 ; output

INC R22 ; increment modulo 10

CPI R22, 2

BRNE SKIP\_INC ; only toggle on 10th rising edge

CLR R22 ; clear modulo 10

EOR R16, R20 ; toggle on 10th rising edge

OUT PORTC, R16 ; output

SKIP\_INC:

RJMP BEGIN ; loop forever

DELAY:

LDI R23, 0x85

STS TCNT1H, R23

LDI R23, 0xEE

STS TCNT1L, R23 ; set initial count of timer

LDI R23, 0x00

STS TCCR1A, R23 ; normal mode

LDI R23, 0x03

STS TCCR1B, R23 ; start timer with 1024 ps

AGAIN:

IN R23, TIFR1 ; check for overflow

SBRS R23, TOV1 ; if overflow, exit loop

RJMP AGAIN ; loop till overflow

LDI R23, 0x00

STS TCCR1B, R23 ; turn off timer

LDI R23, 0x01

OUT TIFR1, R23 ; clear overflow flag

RET

|  |  |  |  |
| --- | --- | --- | --- |
| 3. | INCREMENTAL CODE OF TASK 3/B |  |  |

/\*

\* DA2T1C.c

\*

\* Created: 3/7/2016 10:47:19 AM

\* Author : Brandon

\*/

#define *F\_CPU* 8000000UL

#include <avr/io.h>

#include <util/delay.h>

int main(void)

{

DDRC |= 0x61; // PORTC configured to output mode

DDRB |= 0xFF;

PORTC = 0x00; // PORTC initialized to 0

PORTB = 0x00; // PORTB intiialized to 0

int i = 0;

int j = 0; // helpers to keep track of every 5th and 10th pulse

while (1)

{

TCNT1H = 0x85;

TCNT1L = 0xEE; // set count

TCCR1A = 0x00; // normal mode

TCCR1B = 0x03; // set prescalar

while((TIFR1 & 0x01) == 0); // wait for flag

TCCR1A = 0x00;

TIFR1 |= 1<<TOV1; // reset flag

PORTC ^= 0x01; // toggle PC0

if(PORTC & 0x01)

{

PORTB++; // increment count

if(++i == 5)

{

PORTC ^= 0x20; // toggle on 5th

i = 0;

if(++j == 2)

{

PORTC ^= 0x40; // toggle on 10th

j = 0;

}

}

}

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| 4. | INCREMENTAL CODE OF TASK 4/A |  |  |

;

; DA2.asm

; Task 3 ASM

; Created: 3/6/2016 11:36:31 AM

; Author : Brandon

;

.org 0x00

RJMP INIT

.org 0x20

RJMP TCNT1\_overflow ; interrupt

INIT:

LDI R16, 0xFF

OUT DDRB, R16 ; PortB configured for output

LDI R16, 0x61

OUT DDRC, R16 ; PortC configured for output

LDI R16, 0x00 ; will hold PortC signals

OUT PORTB, R16 ; initialize PortB to 0

OUT PORTC, R16 ; initizlize PortC to 0

LDI R17, 0x00 ; holds current count

LDI R18, 0x01 ; will be used to toggle PC0

LDI R19, 0x20 ; will be used to toggle PC5

LDI R20, 0x40 ; will be used to toggle PC6

LDI R21, 0x00 ; holds modulo 5

LDI R22, 0x00 ; holds modulo 10

BEGIN:

LDI R23, 0x85

STS TCNT1H, R23

LDI R23, 0xEE

STS TCNT1L, R23 ; set initial count of timer

LDI R23, 0x00

STS TCCR1A, R23 ; normal mode

LDI R23, 0x03

STS TCCR1B, R23 ; start timer with 1024 ps

LDI R23, 0x01

STS TIMSK1, R23 ; enable timer interrupt

SEI ; enable interrupts

AGAIN:

RJMP AGAIN ; loop forever

TCNT1\_overflow:

LDI R23, 0x01

OUT TIFR1, R23

EOR R16, R18 ; toggle signal

OUT PORTC, R16 ; output

SBRS R16, 0

RJMP SKIP\_INC ; only increment on rising edge

INC R17 ; increment count

OUT PORTB, R17 ; output

INC R21 ; increment modulo 5

CPI R21, 5

BRNE SKIP\_INC ; only toggle on 5th rising edge

CLR R21 ; clear modulo 5

EOR R16, R19 ; toggle on 5th rising edge

OUT PORTC, R16 ; output

INC R22 ; increment modulo 10

CPI R22, 2

BRNE SKIP\_INC ; only toggle on 10th rising edge

CLR R22 ; clear modulo 10

EOR R16, R20 ; toggle on 10th rising edge

OUT PORTC, R16 ; output

SKIP\_INC:

LDI R23, 0x85

STS TCNT1H, R23

LDI R23, 0xEE

STS TCNT1L, R23 ; set initial count of timer

RETI

|  |  |  |  |
| --- | --- | --- | --- |
| 4. | INCREMENTAL CODE OF TASK 4/B |  |  |

/\*

\* DA2T1C.c

\*

\* Created: 3/7/2016 10:47:19 AM

\* Author : Brandon

\*/

#define *F\_CPU* 8000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

int i = 0;

int j = 0; // helpers to keep track of 5th and 10th pulses

ISR(TIMER1\_OVF\_vect)

{

TIFR1 |= 1<<TOV1; // reset flag

PORTC ^= 0x01; // toggle PC0

if(PORTC & 0x01)

{

PORTB++; // inc count

if(++i == 5)

{

PORTC ^= 0x20; // toggle on 5th

i = 0;

if(++j == 2)

{

PORTC ^= 0x40; // toggle on 10th

j = 0;

}

}

}

TCNT1H = 0x85;

TCNT1L = 0xEE; // set count

}

int main(void)

{

DDRC |= 0x61; // PORTC configured to output mode

DDRB |= 0xFF;

PORTC = 0x00; // PORTC initialized to 0

PORTB = 0x00; // PORTB initialized to 0

TCNT1H = 0x85;

TCNT1L = 0xEE; // set count

TCCR1A = 0x00; // normal mode

TCCR1B = 0x03; // set pre scalar

TIMSK1 = 0x01; // enable timer interrupt

sei(); // enable global interrupts

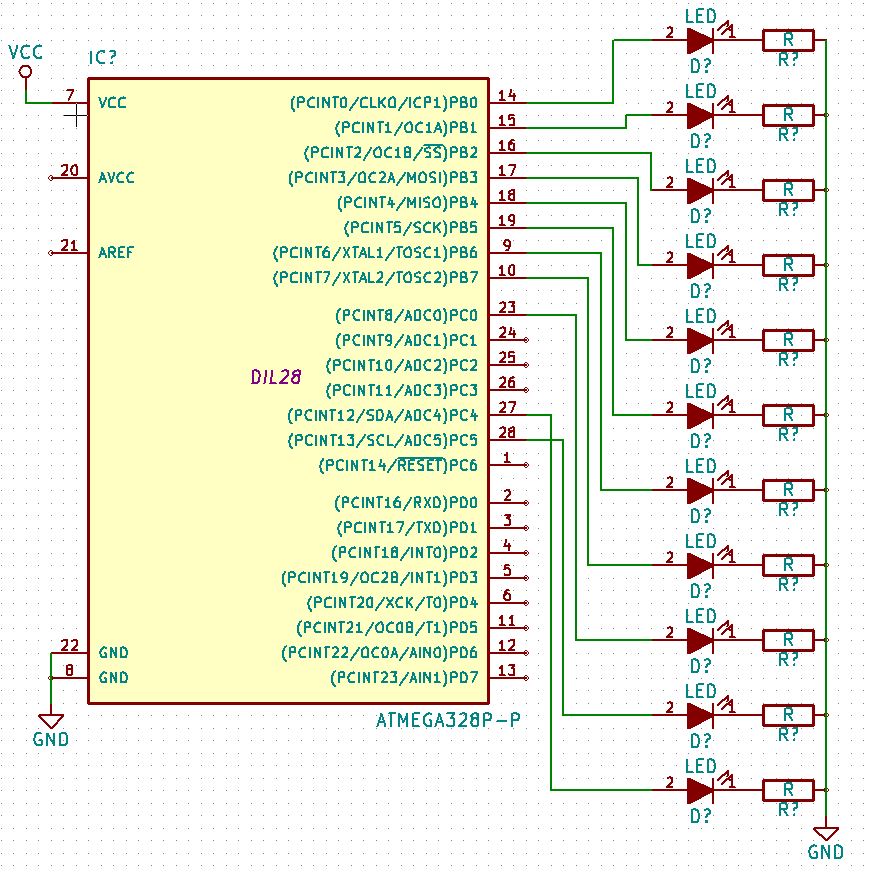
while (1) // wait for interrupts, loop forever

{

}

}

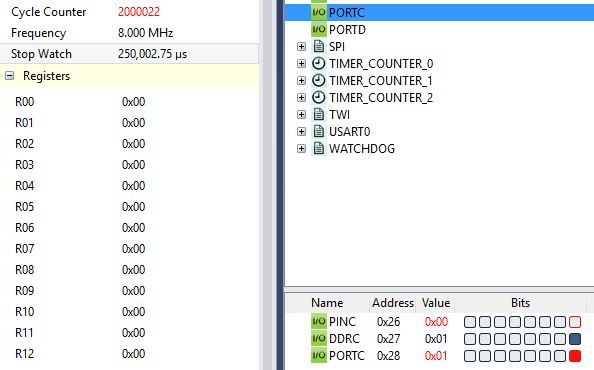
|  |  |  |  |
| --- | --- | --- | --- |
| 6. | SCHEMATICS |  |  |



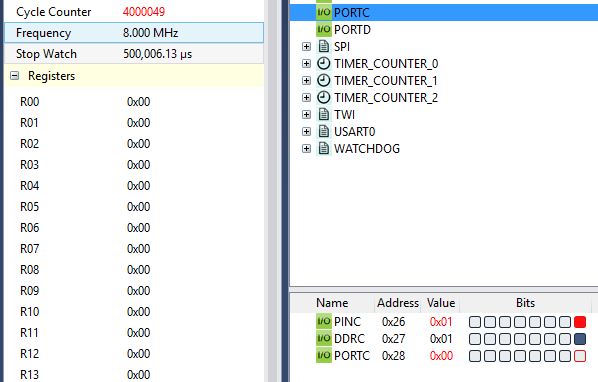
PC6 switched with PC4 in demo only

|  |  |  |  |
| --- | --- | --- | --- |
| 7. | SCREENSHOTS OF EACH TASK OUTPUT |  |  |

TASK 1/A: Half second waveform at PC0 (ASM)

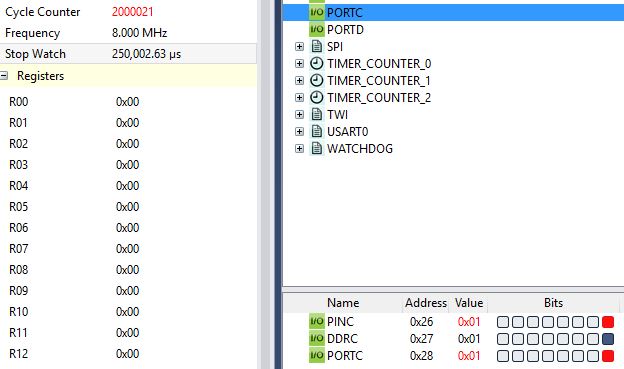


Rising Edge @ .25 seconds

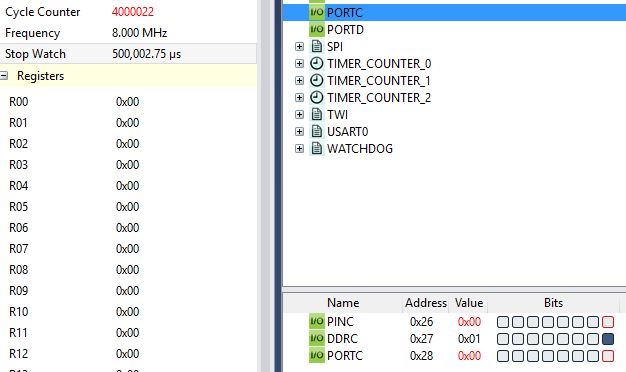


Falling Edge @ .5 seconds

TASK1/B: Half second waveform at PC0 (C)

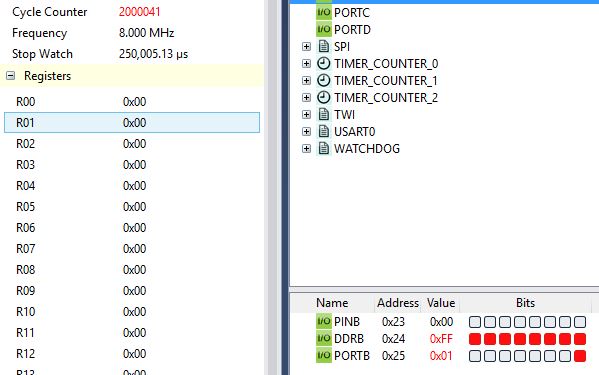


Rising Edge @ .25 seconds

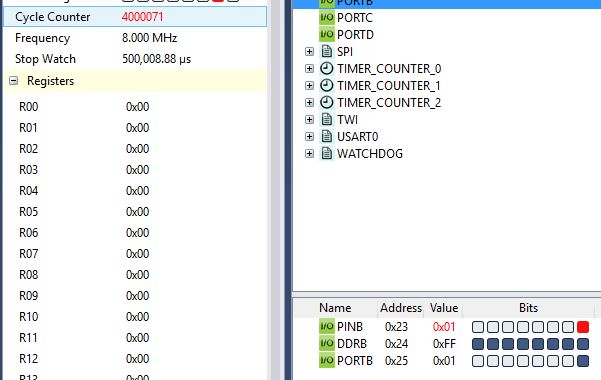


Falling Edge @ .5seconds

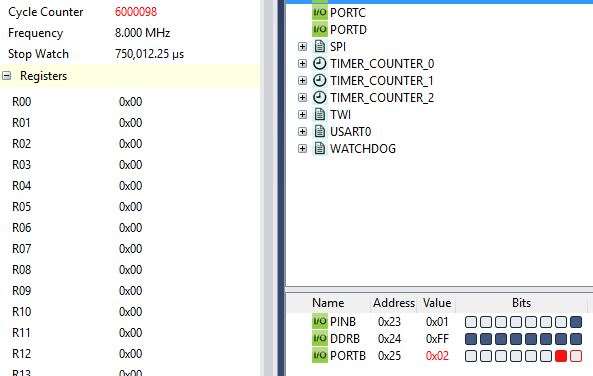
TASK2/A: Count on PORTB (ASM)



Count = 1 at first rising edge

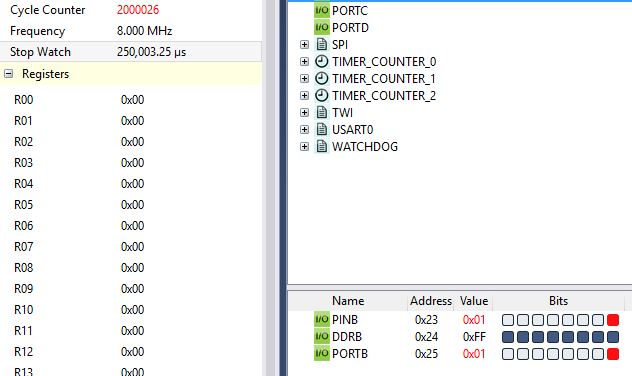


Count = 1 at first falling edge

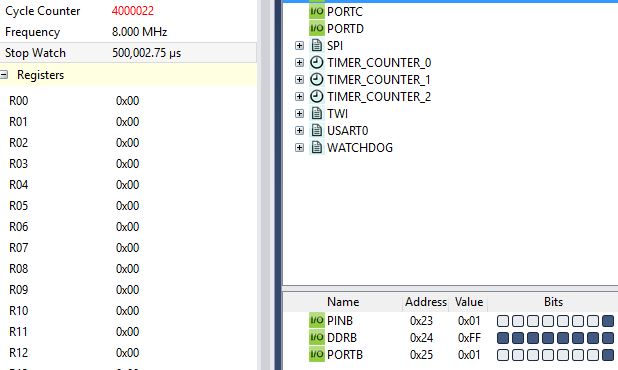


Count = 2 at second rising edge

TASK2/B: Count on PORTB (C)



Count = 1 at first rising edge

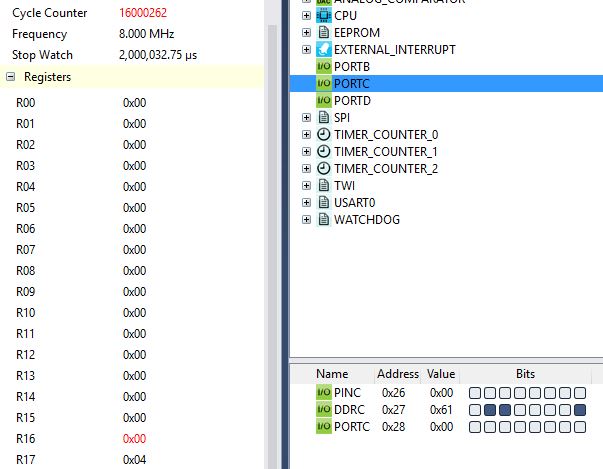


Count = 1 at first falling edge

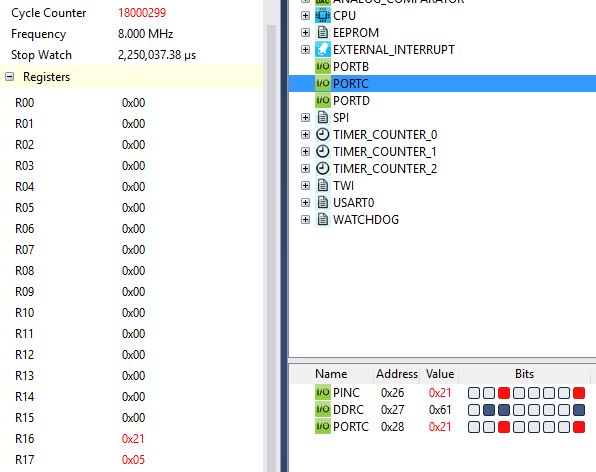


Count = 2 at second rising edge

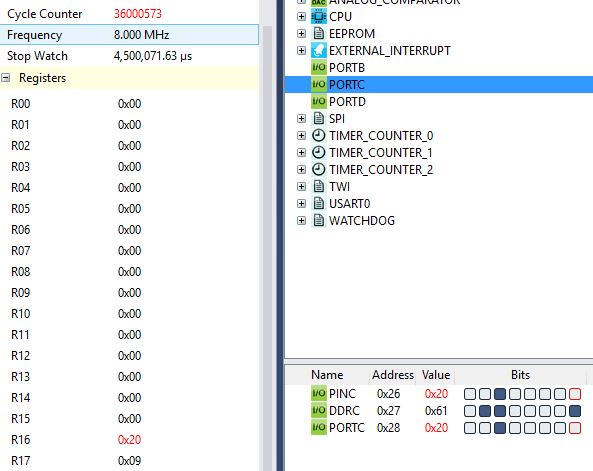
TASK3/A: Toggle PC5 and PC6 on 5th and 10th rising edges (ASM)



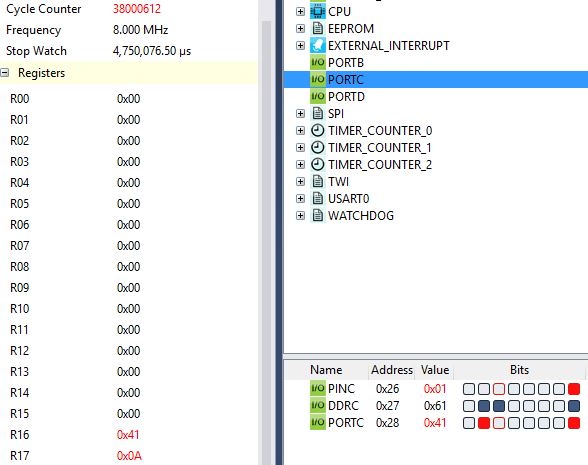
Pre Fifth Rising Edge, R17 holds current count



Post Fifth Rising Edge, R17 holds current count

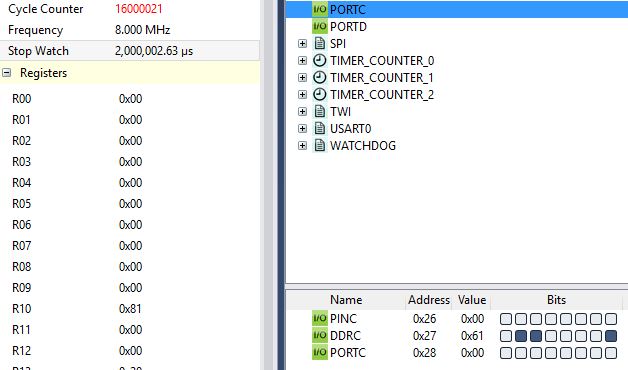


Pre Tenth Rising Edge, R17 holds current count

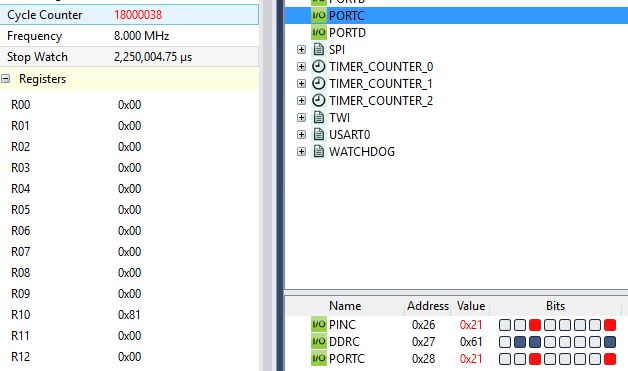


Post Tenth Rising Edge

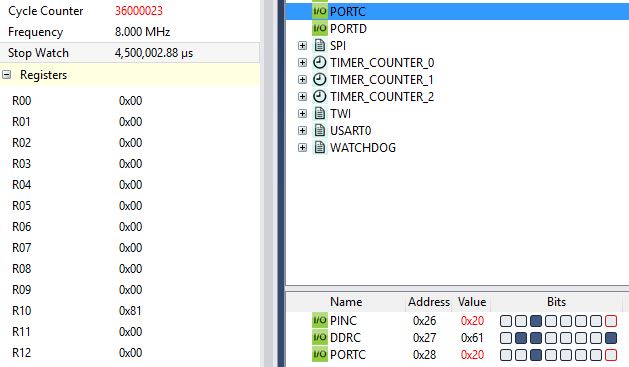
TASK3/B: Toggle PC5 and PC6 on 5th and 10th rising edges (C)



Pre Fifth Rising Edge



Post Fifth Rising Edge

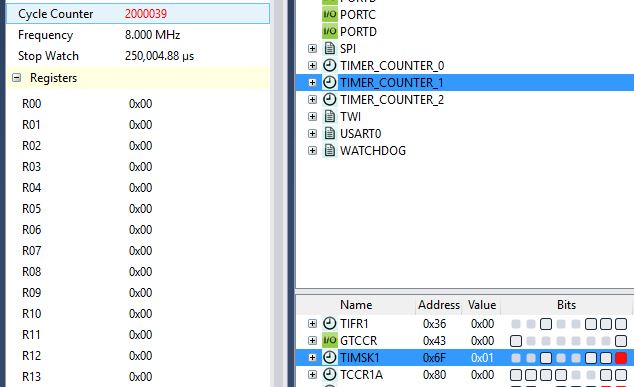


Pre Tenth Rising Edge



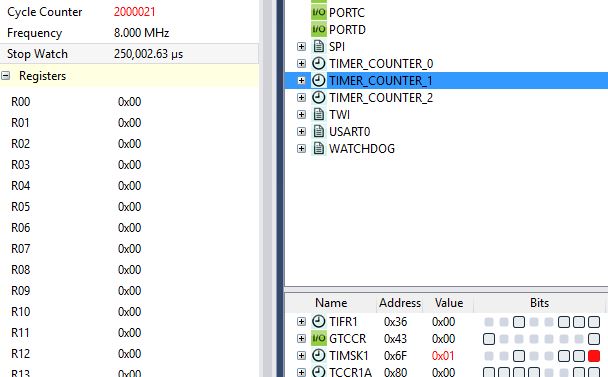
Post Tenth Rising Edge

TASK4/A: Interrupt Enabled (ASM)



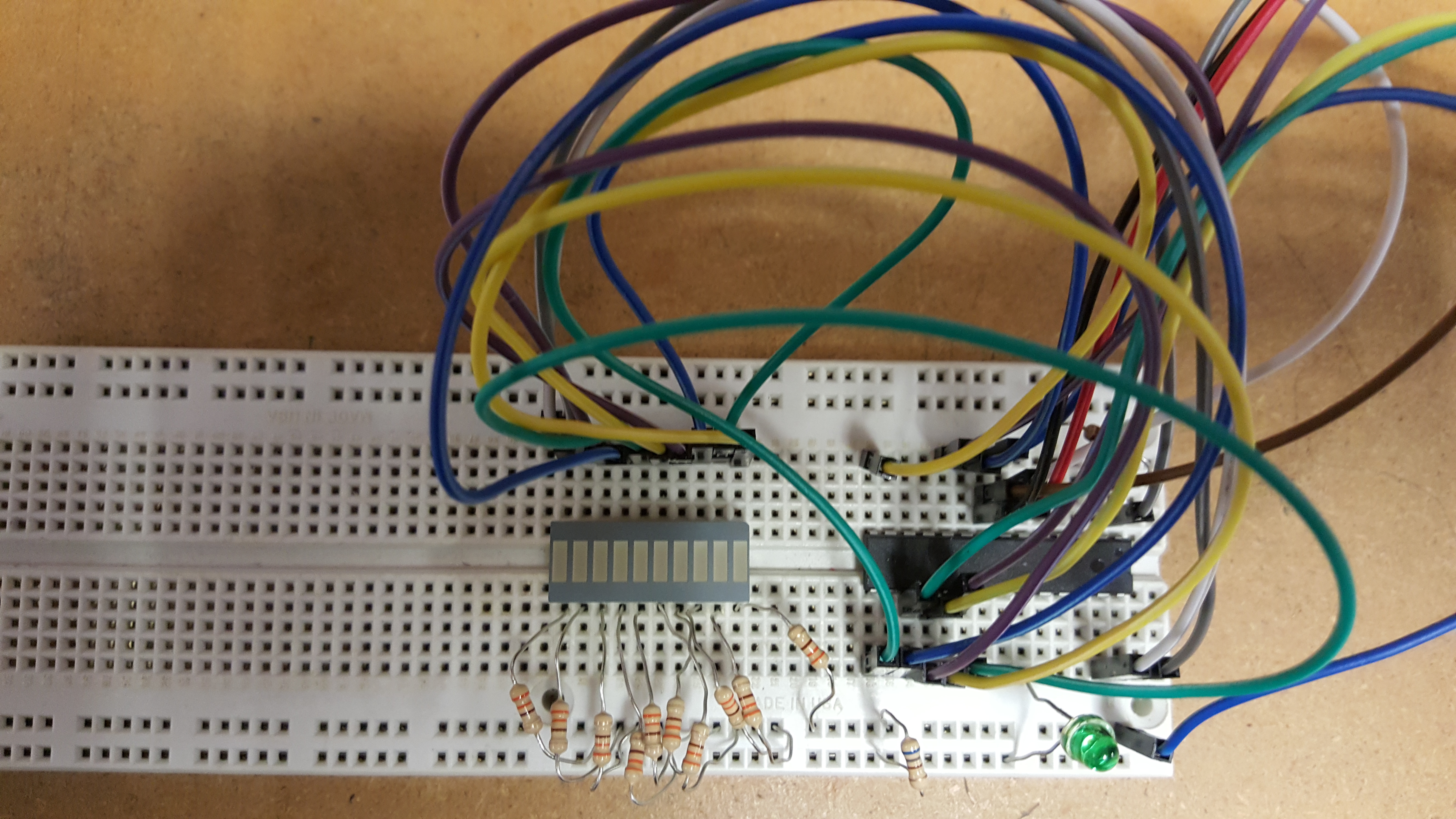
Interrupt every .25 seconds

TASK4/B: Interrupt Enabled (C)



Interrupt every .25 seconds

|  |  |  |  |
| --- | --- | --- | --- |
| 8. | SCREENSHOT OF EACH DEMO |  |  |



LED bar on left, with left most LEDs connected to PC4, and PC5. Right most LEDs connected to PORTB. Green LED is connected to PC0. Setup used for each task. Video demonstrations linked below.

|  |  |  |  |
| --- | --- | --- | --- |
| 9. | VIDEO LINKS OF EACH DEMO |  |  |
| Demo ASM - <https://www.youtube.com/playlist?list=PLp82ERNjD3ZY7RiQE7akGRd7dOT7NSAHC>  Demo C - <https://www.youtube.com/playlist?list=PLp82ERNjD3ZZF0pcgVJH0VDHRdgkJ1ooZ> | | | |
| 10. | GOOGLECODE LINK OF THE DA |  |  |
| <https://github.com/wadeb1/KF3HF6ZFMP.git> | | | |

**Student Academic Misconduct Policy**

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

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

Brandon Wade