NAMES:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ECE-C304 Lab Assignment # 4

(4A,4B,4C due Mon/Tues of 8th Week)

4-1

What is the frequency of the tachometer signal? \_**6.079 kHz**\_

Calculate RPM of fan (assume 8-pole motor) \_**91.185 rpm**\_

4A-1

What is the frequency of each variable clock?

VC1\_\_**3** **MHz**\_\_ VC2\_\_**1** **MHz**\_\_ VC3\_\_**255** **kHz**\_\_

4A-2

For this fan, a PWM output frequency of 1000 Hz is ideal. Suppose it had been 900 Hz,

what clock value is desired? \_**21.6 MHz**\_

4A-3

How would you generate it?

**Change the VC3 divider to 104 instead of 94.**

4A-4

What is the fan speed in RPM? \_**2175 rpm**\_

4A-5

Lightly touch the fan blades with your finger. What happens to the frequency?

**The frequency decreased**

#include <m8c.h> // part specific constants and macros

#include "PSoCAPI.h" // PSoC API definitions for all User Modules

#include "LCD.h"

#include "DBB00.h"

unsigned int dBounce = 0;

unsigned char pulseWidth = 0xFF;

unsigned char prt1 = 0x00;

void main(void)

{

// Insert your main routine code here.

LCD\_Start();

LCD\_Init();

LCD\_Position(1,4);

DBB00\_WritePulseWidth(pulseWidth);

DBB00\_Start();

LCD\_PrHexByte(DBB00\_bReadPulseWidth());

INT\_MSK0 |= 0x40;

M8C\_EnableGInt;

while(1){

M8C\_Sleep;

prt1 = PRT1DR;

prt1 &= 0x01;

if(prt1 == 0x01){

dBounce++;

if(dBounce == 3){

pulseWidth--;

DBB00\_WritePulseWidth(pulseWidth);

dBounce = 0;

}

}

else

{

dBounce = 0;

}

LCD\_Position(0,4);

LCD\_PrHexByte(dBounce);

LCD\_Position(1,4);

LCD\_PrHexByte(DBB00\_bReadPulseWidth());

}

}

Initials\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Time\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4B-1

What is the period and pulse width of TachOverrideOut?

**Period** = 1000 ms **Pulse Width** = 20 ms

4B-2

The TC output of FanPWM is used to clock TachOverrideOut. Why not use CompareOut?

**Because CompareOut has a different value depending on the pulse width, which changes depending on how long the button has been held down**.

4B-3

Triggering the oscilloscope on the rising edge of TachOverrideOut what is the fan speed for the PWM pulse width values shown in Table 1.14?

|  |  |
| --- | --- |
| Pulse Width | Fan Speed (RPM) |
| 255 | 2586 |
| 224 | 2343 |
| 192 | 2084 |
| 160 | 1786 |
| 128 | 1365 |
| 96 | 1140 |
| 64 | 698 |

4B-4

4B-5 What are the constants m=\_9.81\_ b=\_155\_

Initials\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Time\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4C-1

Describe another way to force FanPWMout high

**Instead of using PWM16 to set R01 high and OR-ing it with the FanPWMout. We can instead set R01 through a interrupt.**

4C-2

Using the assembly language code given in the handout, show the waveforms of PanPWMOut,

TachOverideOut and TachOutput on the oscilloscope. What is your observation?

**When the TachOverride is high, the Tach signal is clearer, the same as in part B**

Initials\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Time\_\_\_\_\_\_\_\_\_\_\_\_\_\_