CSE4342 Embedded Controllers II
CSE5342 Embedded Systems
Fall 2020, Test 2

Name Dijesh Pradhan.

93

Assume that a TM4C123GH6PM controller is used for the following problems. All work must be done on these test pages. Do not write test solutions on the ethics statement page. Calculators, datasheets, notes, and solved problems are allowed on the exam. The use of devices or software that support assembly or compilation of code is not allowed. Phones or communications devices may not be used during the exam.

1. Answer the following questions about <u>your</u> project code (try your best if this step is not complete).

a. Write a function to configure the UART to send DMX512 data.

```
void initUart1()
{
-> UART1_CTL_R = 0;
-> UART1_CC_R = UART_CC_SYSCLK;
-> UART1_IBRD_R = 10;
-> UART1_FBRD_R = 0;
-> UART1_LCRH_R = UART_LCRH_WLEN_8 | UART_LCRH_FEN | UART_LCRH_STP2;
-> UART1_CTL_R = UART_CTL_TXE | UART_CTL_RXE | UART_CTL_UARTEN;
-> UART1_CTL_R |= UART_CTL_EOT;
```

//UART_LCRH_STP2
stop bit 2

-> NVIC_EN0_R |= 1 << (INT_UART1-16);

b. Write a function startDmxTx() and associated variables that starts transmission on the DMX512 bus sending the break and configuring interrupt(s) as needed to continue transmission, including DE and PCTL control.

```
void startDmxTx()
{
  DE = 1;   // PORTC7 configured in init HW.
  GPIO_PORTB_DATA_R = 0;
  phase = 0;
  TIMER1_TAILR_R = 7040;  //176ms
  TIMER1_CTL_R |= TIMER_CTL_TAEN;
  TIMER1_IMR_R = TIMER_IMR_TATOIM;
}

// I have cleared the GPIO_PORTB_PCTL_R in the init UART1();
GPIO_PORTB_PCTL_R &= ~(GPIO_PCTL_PB1_M | GPIO_PCTL_PB0_M);
GPIO_PORTB_PCTL_R |= ~(GPIO_PCTL_PB1_U1TX | GPIO_PCTL_PB0_U1RX);
```

c. Write a function timerIsr() and associated variables that transmits the MAB and start code on the bus, including PCTL register control and configures interrupt(s) as needed to continue transmission.

```
void TimerIIsr()
{
    if (phase == = 0)
      {
       GPIO_PORTB_DATA_R = 0x00000002;
       TIMER1_CTL_R &= ~TIMER_CTL_TAEN;
       TIMER1_TAILR_R = 480;
       TIMER1_CTL_R |= TIMER_CTL_TAEN;
       phase = 1;
      }
```

// I used periodic timer instead of one-shot.

```
else if (phase == = 1)
 {
    TIMER1_IMR_R &= ~TIMER_IMR_TATOIM;
    TIMER1_CTL_R &= ~TIMER_CTL_TAEN;
    TIMER1_TAILR_R = 7040;
 GPIO_PORTB_AFSEL |= (UART1_TX_MASK) UART1_RX_MASK
 GPIO_PORTB_PCTL_R |= (GPIO_PCTL_PB1_U1TX |
                       GPIO_PCTL_PB0_U1RX)
 UART1_DR_R = 0;
 UART1_IMR |= UART_IM_TXIM;
    phase = 2;
 }
TIMER1_ICR_R = TIMER_ICR_TATOCINT;
}
```

d. Write a function uart1TxIsr() and associated variables that transmits the address data and restarts the transmission at the end if run is still active.

```
void UART1ISR()
{
  if((phase-2)<max_add)
   {
     while(UART1_FR_R & UART_FR_TXFF);
     UART1_DR_R = DATA[phase-2];
     phase++;
   }
else if((phase-2) == max_add)
  {
    UART1_IM_R &= ~UART_IM_TXIM;
   GPIO_PORTB_AFSEL_R &= ~(UART1_TX_MASK |
                           UART1_RX_MASK);
   GPIO_PORTB_PCTL_R &= ~(GPIO_PCTL_PB1_M |
                          GPIO_PCTL_PB0_M)
   while(UART1_FR_R & UART_FR_BUSY);
   if (start == 1)
    {
      startDmxTx();
    }
else if (start == 0)
{
  UART1_IM_R &= ~UART_IM_TXIM;
 }
}
}
```

c Write a function timerIsr() and associated variables that transmits the MAB and start code on the bus, including PCTL register control and configures interrupt(s) as needed to continue transmission.

```
void TimerIIsr()
{
    if (phase == 0)
    {
     GPIO_PORTB_DATA_R = 0x00000002;
     TIMER1_CTL_R &= ~TIMER_CTL_TAEN;
     TIMER1_TAILR_R = 480;
     TIMER1_CTL_R |= TIMER_CTL_TAEN;
     phase = 1;
    }

else if (phase == 1)
{
   TIMER1_IMR_R &= ~TIMER_IMR_TATOIM;
   TIMER1_CTL_R &= ~TIMER_CTL_TAEN;
   TIMER1_TAILR_R = 3040;
GPIO_PORTB_AFSEL |= (UART1_TX_MASK | UART1_RX_MASK);
GPIO_PORTB_PCTL_R |= (GPIO_PCTL_PB1_U1TX |
                      GPIO_PCTL_PB0_U1RX);
UART1_DR_R = 0;
UART1_IMR |= UART_IM_TXIM;
  phase = 2;
}
TIMER1_ICR_R = TIMER_ICR_TATOCINT;

}
```

// I used periodic timer instead of one-shot.

AIN3 3.3V

2. Answer the following questions related to the PID controller, as shown in class, assuming an analog feedback on AIN3. The analog feedback is provided by a potentiometer with a free rotational range of 0 to 270 degrees and is powered by a 3.3V supply.

Kp = 300, Ki = 0, Kd = 0, Ko = 0, K = 100, and there is no feedback scaling or integral windup limit.

a. Calculate the instantaneous value of u if the setpoint is 1600 and the current position feedback (y) is 1000.

$Error = y_{setpt} - y = 1600 - 1000 = 600$

$U = \frac{\varepsilon * K_p}{K} = 600 * \frac{300}{100} = 1800$ (As $K_i = 0$, $K_d = 0$)

$\therefore U = 1800$;

b. Calculate the angular displacement error, in degrees, given the conditions of (a).

Error = 600;

$\frac{270}{4096} = 0.0659\ ^\circ/LSB$

Angular Displacement Error = $600 * 0.0659\ ^\circ/LSB$

$= 39.55^\circ$

c. While traveling to 1600, at a position of 1560, the motor stalls. What would be the PWM CMP register and DIRECTION bit values at this point?

$E = 1600 - 1560 = 40.$

$U = 40 * \frac{300}{100} = 120$

$\therefore$ PCM CMP = 120

$\therefore$ DIRECTION = 1

d. After you notice that the motor has stalled, you set Ki to 10. If the motor will start to run at a duty cycle of 30%, how long does it take for the PID software to command such a value?

$K_i = 10$ Duty cycle = 30%.

$U = K_p E + K_i \int_{t=0}^{t_0} E(t)$

$\frac{U}{K} = \left(\frac{40 \times 300 + 10 \times t}{100}\right) \times \frac{30}{100}$ (-1) (0) (-2)

$\frac{3 \times 10 \times t}{1000} = -\frac{40 \times 300}{1000}$ Or, $t = \frac{40 \times 300}{3 \times 10} \times \frac{100}{1} = 400$

3. You are asked to interface an LM34 temperature sensor to the TM4C123GXL board on pin AIN3 of the controller using an circuit that applies voltage offset and gain. The LM34 sensor outputs a voltage of 10 mV / degF. The only range of temperatures to be monitored ranges from 20 to 150 degF.

a. Show all offset and gain blocks necessary to interface the sensor to the controller that maximizes the temperature resolution over the intended frequency range (20 degF → 0V and 150 degF → 3.3V).

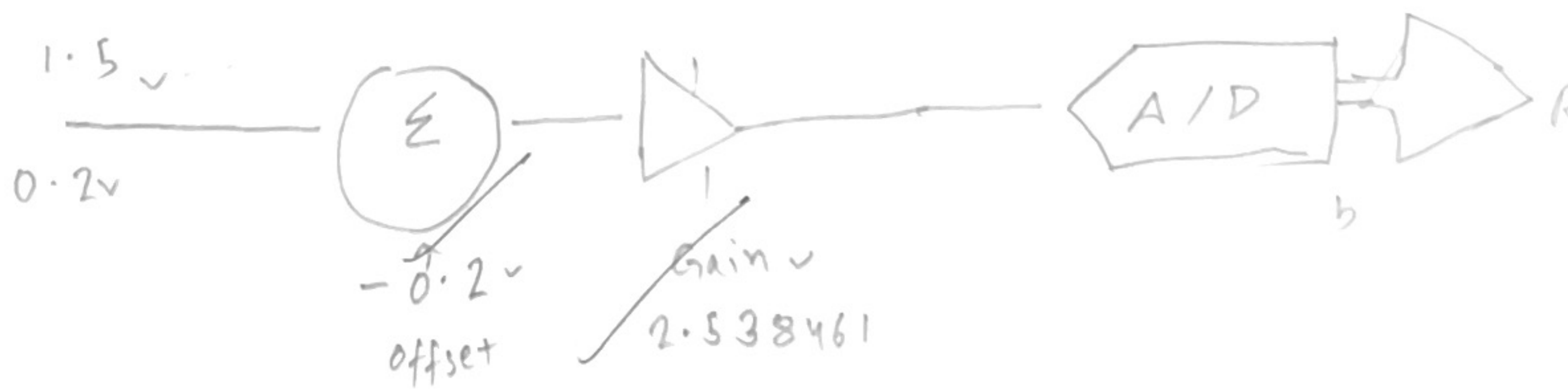
Output voltage = $\frac{10\,mv}{deg F}$ = $\frac{0.01\,V}{deg F}$

So V_{in} will be As Range is from 20 degF to 150 degF

0.2 V – 1.5 V

Vin	ofs	gain	Vadc	R
1.5	1.3	2.538	3.3v	4095
0.2	0	0	0	0.

Gain = $\frac{3.3}{1.3}$ = 2.538461538



b. What is the resolution in degF / LSb?

Resolution = $\frac{150-20}{4096}$ = 0.03173 degF/LSB

c. Write a function, float getTemperature() that returns the temperature measured by the LM34 sensor.

```
float getTemperature()
{
  int16_t temp = 0;
  int16_t Raw = 0;
   Raw = readAdcSs3();
   temp = Raw * 0.03173;
  return temp;
}
```

(-4)

4. Answer the following questions related to the bootloader.

a. What is function of the following line of code from the hex file:

:00000001FF

This line of code indicates that it is the end of the file & there is no more data as 01 is the type for end.

b. How does the bootloader program know the interrupt vector table is not in the correct location?

In the .hex file if the lower 16bits of address is 0000 the it shows that it is trying to load the IVT in the Bootloade IVT. The LLAAMTTCDD..DTC. The AAAA cannot be 0000. This is set in the .cmd file to let program know where to start IVT.

c. What is the initial value of the PC that will be loaded when the bootloaded runs this program as represented in this first line of a hex file:

:20100000004002041140000471300004913000048130000481300004813000000000008A
... SP PC

The initial value of the PC is 0X 00 00 14 41

d. Using the data in (c), what is the initial value of the SP that will be loaded when this bootloader runs this program:

The initial value of the SP is 0X 20 00 04 00

5. Answer the following questions related to GPIO on the RPi 3b+

a. Suppose an LED is connected to the RPi GPIO number 14 so that it is on when the pin is high. Using the virtual file system (sysfs), show the commands from the bash shell required to turn on this LED.

echo 14 > export
echo out > direction
echo 1 > value

b. How was the value of 0xB4 calculated for the mmap call?

The value of 0xB4 for the mmap call is generated from the BCM2835 data sheet where the registers start from 0x7E200000 to 0x7E2000B0 & the last registers occupies till 0x7E2000B4 so the required space is 0xB4.

c. What is the physical memory address of /sys/gpiomem in memory?

The physical memory address of /sys/gpiomem in mem is

0x7E200000.