CPE403 – Advanced Embedded Systems

Design Assignment 2

DO NOT REMOVE THIS PAGE DURING SUBMISSION:

Name: Meral Abu-Jaser

Email: abujaser@unlv.nevada.edu

Github Repository link (root): https://github.com/MeralAbuJaser/Advanced-Embedded-Systems

Youtube Playlist link (root): https://www.youtube.com/playlist?list=PLmRQUGgBgm2dlt37RCWlrSrGj7KxvDKmj

Follow the submission guideline to be awarded points for this Assignment.

Submit the following for all Assignments:

1. In the document, for each task submit the modified or included code (from the base code) with highlights and justifications of the modifications. Also include the comments. If no base code is provided, submit the base code for the first task only.
2. Create a private Github repository with a random name (no CPE/403, Lastname, Firstname). Place all labs under the root folder TIVAC, sub-folder named Assignment1, with one document and one video link file for each lab, place modified c files named as asng_taskxx.c.
3. If multiple c files or other libraries are used, create a folder asng1_t01 and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) with startup_ccs.c and other include files, c) text file with youtube video links (see template).
5. Submit the doc file in canvas before the due date. The root folder of the github assignment directory should have the documentation and the text file with youtube video links.
6. Organize your youtube videos as playlist under the name “cpe403”. The playlist should have the video sequence arranged as submission or due dates.
7. Only submit pdf documents. Do not forget to upload this document in the github repository and in the canvas submission portal.

1. Task 01

```c
#include <stdbool.h>
#include <stdint.h>
#include "inc/hw_i2c.h"
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "inc/hw_gpio.h"
#include "driverlib/i2c.h"
#include "driverlib/sysctl.h"
#include "driverlib/gpio.h"
#include "driverlib/pin_map.h"
#include "stdarg.h"
#include "driverlib/rom.h"
#include "driverlib/sysctl.h"
#include "driverlib/uart.h"

//*****************************************************************************
//
// Define TMP006 I2C Address.
//
//*****************************************************************************
#define TMP006_I2C_ADDRESS      0x41

void initI2C(void);
void readI2C(uint8_t slave_addr, uint8_t reg, int *data);
void I2C0_Send(uint8_t slave_addr, uint8_t num_of_args, ...);
void writeI2C(uint8_t slave_addr, uint8_t reg, uint8_t data);
 void I2C0_read(uint8_t slave_addr, uint8_t *RxData, uint8_t N);
void I2C0_Send16(uint8_t slave_addr, uint8_t pointer_reg, uint16_t TxData);
void I2C0_Read16(uint8_t slave_addr, uint8_t pointer_reg, uint16_t RxData);

void
ConfigureUART(void){
    //
    // Enable the GPIO Peripheral used by the UART.
    //
    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA);
```

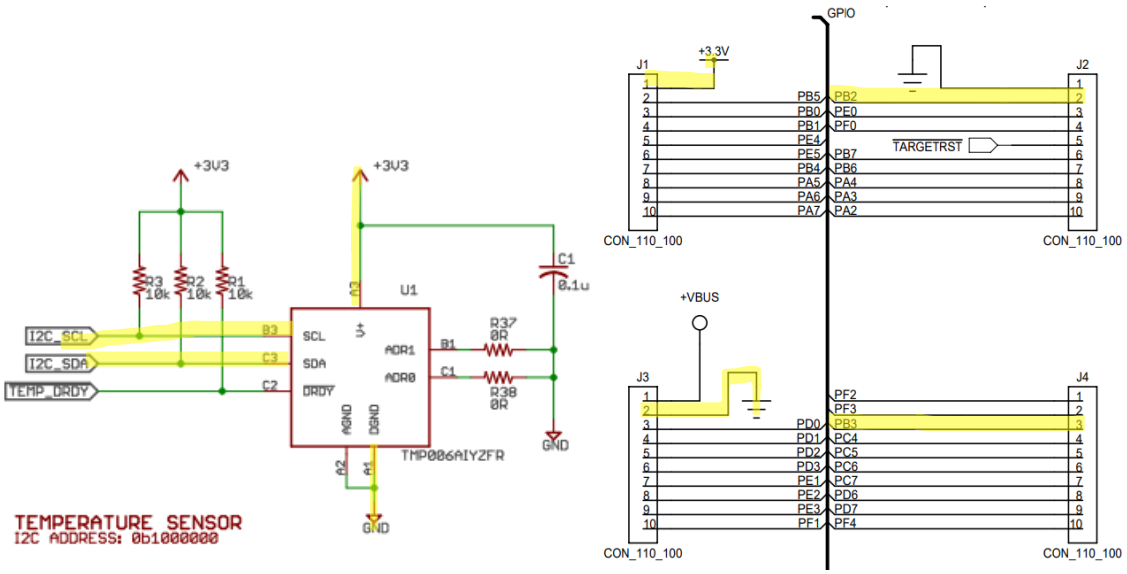
```
    ROM_GPIOPinConfigure(GPIO_PA0_U0RX);
    ROM_GPIOPinConfigure(GPIO_PA1_U0TX);
    ROM_GPIOPinTypeUART(GPIO_PORTA_BASE, GPIO_PIN_0 | GPIO_PIN_1);
    //
    // Use the internal 16MHz oscillator as the UART clock source.
    //
    UARTClockSourceSet(UART0_BASE, UART_CLOCK_PIOSC);
    //
    // Initialize the UART for console I/O.
    //
    UARTStdioConfig(0, 115200, 16000000);
}
int main(void){
    _iq fAmbient, fObject;
    _iq i32IntegerPart;
    _iq i32FractionPart;
    //
    // Initialize the UART.
    //
    //
    // Delay for 10 milliseconds for TMP006 reset to complete.
    // Not explicitly required. Datasheet does not say how long a reset takes.
    //
    ROM_SysCtlDelay(ROM_SysCtlClockGet() / (100 * 3));

    ConfigureUART();
    i32IntegerPart = _IQ(fObject);
    i32FractionPart = _IQ(fObject * 1000.0f);
    i32FractionPart = i32FractionPart - (i32IntegerPart * 1000);
    UARTprintf("TMP006 Temperature:\n", i32IntegerPart, i32FractionPart);
    return 0;
}

void initI2C0(void){
    SysCtlPeripheralEnable(SYSCTL_PERIPH_I2C0);
    SysCtlDelay(3);
     SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOB);
     SysCtlDelay(3);

     GPIOPinConfigure(GPIO_PB2_I2C0SCL);
     GPIOPinConfigure(GPIO_PB3_I2C0SDA);

     GPIOPinTypeI2CSCL(GPIO_PORTB_BASE, GPIO_PIN_2);
     GPIOPinTypeI2C(GPIO_PORTB_BASE, GPIO_PIN_3);

     I2CMasterInitExpClk(I2C0_BASE, SysCtlClockGet(), true);
     //clear I2C FIFOs
     HWREG(I2C0_BASE + I2C_O_FIFOCTL) = 80008000;
  }

  void readI2C(uint8_t slave_addr, uint8_t reg, int *data)
  {
     I2CMasterSlaveAddrSet(I2C0_BASE, slave_addr, true);
     I2CMasterDataPut(I2C0_BASE, reg);
     I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_START);
     while(I2CMasterBusy(I2C0_BASE));
     I2CMasterSlaveAddrSet(I2C0_BASE, slave_addr, true);
     I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_SINGLE_RECEIVE);
     while(I2CMasterBusy(I2C0_BASE));
     *data = I2CMasterDataGet(I2C0_BASE);
  }
```

```
108 // Sends 1 byte over i2c
109 void writeI2C(uint8_t slave_addr, uint8_t reg, uint8_t data)
110 {
111     I2CMasterSlaveAddrSet(I2C0_BASE, slave_addr, false);
112     I2CMasterDataPut(I2C0_BASE, reg);
113     I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_START);
114   while(I2CMasterBusy(I2C0_BASE));
115   I2CMasterDataPut(I2C0_BASE, data);
116   I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_FINISH);
117   while(I2CMasterBusy(I2C0_BASE));
118 }
119
120 //sends an I2C command to the specified slave
121 void I2C0_Send(uint8_t slave_addr, uint8_t num_of_args, ...)
122 {
123    // Tell the master module what address it will place on the bus when
124    // communicating with the slave.
125    I2CMasterSlaveAddrSet(I2C0_BASE, slave_addr, false);
126    //stores list of variable number of arguments
127    va_list vargs;
128    //specifies the va_list to "open" and the last fixed argument
129    //so vargs knows where to start looking
130    va_start(vargs, num_of_args);
131    //put data to be sent into FIFO
132    I2CMasterDataPut(I2C0_BASE, va_arg(vargs, uint32_t));
133    //if there is only one argument, we only need to use the
134    //single send I2C function
135    if(num_of_args == 1)
136    {
137      //Initiate send of data from the MCU
138      I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_SINGLE_SEND);
139      // Wait until MCU is done transferring.
140      while(I2CMasterBusy(I2C0_BASE));
141      //"close" variable argument list
142      va_end(vargs);
143    }
144    //otherwise, we start transmission of multiple bytes on the
145    //I2C bus
146    else
147    {
148      //Initiate send of data from the MCU
149      I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_START);
150      // Wait until MCU is done transferring.
151      while(I2CMasterBusy(I2C0_BASE));
152      //send num_of_args-2 pieces of data, using the
153      //BURST_SEND_CONT command of the I2C module
154      unsigned char i;
155      for(i = 1; i < (num_of_args - 1); i++)
156      {
157        //put next piece of data into I2C FIFO
158        I2CMasterDataPut(I2C0_BASE, va_arg(vargs, uint32_t));
159        //send next data that was just placed into FIFO
160        I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_CONT);
161        // Wait until MCU is done transferring.
162        while(I2CMasterBusy(I2C0_BASE));
163      }
164      //put last piece of data into I2C FIFO
165      I2CMasterDataPut(I2C0_BASE, va_arg(vargs, uint32_t));
166      //send next data that was just placed into FIFO
167      I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_FINISH);
168      // Wait until MCU is done transferring.
169      while(I2CMasterBusy(I2C0_BASE));
170      //"close" variable args list
171      va_end(vargs);
172    }
173 }
```

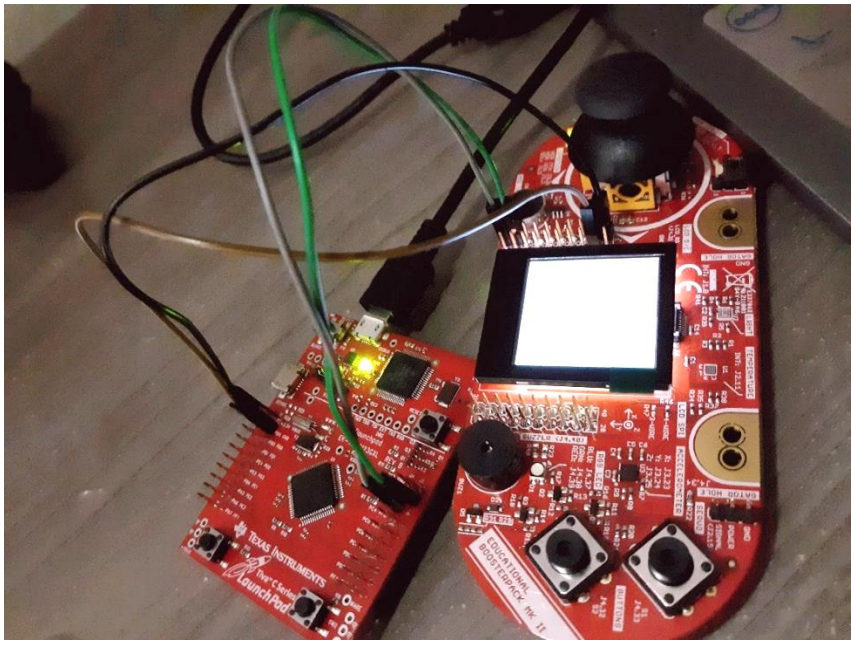
```
175 void I2C0_read(uint8_t slave_addr, uint8_t *RxData, uint8_t N)
176 {
177   uint8_t i;
178
179   I2CMasterSlaveAddrSet(I2C0_BASE, slave_addr, true);
180   while (I2CMasterBusy(I2C0_BASE));
181
182   if (N==1)
183   {
184    I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_SINGLE_RECEIVE);
185    while (I2CMasterBusy(I2C0_BASE));
186    RxData[0]=I2CMasterDataGet(I2C0_BASE);
187    while (I2CMasterBusy(I2C0_BASE));
188  }
189  else
190  {
191    I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_RECEIVE_START);
192    while (I2CMasterBusy(I2C0_BASE));
193    RxData[0]=I2CMasterDataGet(I2C0_BASE);
194    while (I2CMasterBusy(I2C0_BASE));
195
196    for (i=1;i<(N-1);i++)
197    {
198     I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_RECEIVE_CONT);
199     while (I2CMasterBusy(I2C0_BASE));
200     RxData[i]=I2CMasterDataGet(I2C0_BASE);
201     while (I2CMasterBusy(I2C0_BASE));
202   }
203
204   I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_RECEIVE_FINISH);
205   while (I2CMasterBusy(I2C0_BASE));
206   RxData[N-1]=I2CMasterDataGet(I2C0_BASE);
207   while (I2CMasterBusy(I2C0_BASE));
208 }
209}
212void I2C0_Send16(uint8_t slave_addr, uint8_t pointer_reg, uint16_t TxData)
213{
214   uint8_t data;
215   I2CMasterSlaveAddrSet(I2C0_BASE, slave_addr, false);
216   I2CMasterDataPut(I2C0_BASE, pointer_reg);
217   I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_START);
218   while(I2CMasterBusy(I2C0_BASE));
219   // MSB First
220   data = (uint8_t)((TxData >> 8) & 0x00FF);
221   I2CMasterDataPut(I2C0_BASE, data);
222   while(I2CMasterBusy(I2C0_BASE));
223   //LSB Later
224   data = (uint8_t)(TxData  & 0x00FF);
225   I2CMasterDataPut(I2C0_BASE, data);
226   I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_FINISH);
227   while(I2CMasterBusy(I2C0_BASE));
228}
```

```
230 //sends an I2C command to the specified slave
231 void I2C0_Read16(uint8_t slave_addr, uint8_t pointer_reg, uint16_t RxData )
232 {
233     uint8_t data;
234     I2CMasterSlaveAddrSet(I2C0_BASE, slave_addr, true);
235     I2CMasterDataPut(I2C0_BASE, pointer_reg);
236     I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_START);
237     while(I2CMasterBusy(I2C0_BASE));
238     I2CMasterSlaveAddrSet(I2C0_BASE, slave_addr, true);
239     I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_RECEIVE_CONT);
240     while(I2CMasterBusy(I2C0_BASE));
241     //MSB first
242     data = I2CMasterDataGet(I2C0_BASE);
243     RxData = (uint16_t)(data << 8);
244     I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_RECEIVE_CONT);
245     while(I2CMasterBusy(I2C0_BASE));
246     //LSB later
247     data = I2CMasterDataGet(I2C0_BASE);
248     RxData |= (uint16_t)(data);
249     I2CMasterControl(I2C0_BASE, I2C_MASTER_CMD_BURST_SEND_FINISH);
250     while(I2CMasterBusy(I2C0_BASE));
251 }
```

2. Block diagram and/or Schematics showing the components, pins used, and interface.



3. Screenshots of the IDE, physical setup, debugging process - Provide screenshot of successful compilation, screenshots of registers, variables, graphs, etc.



```
Building target: "I2C.out"
Invoking: ARM Linker
"C:/ti/ccs1010/ccs/tools/compiler/ti-cgt-arm_20.2.1.LTS/bin/armcl" -mv7M4 --code_state=16 --float_support=FPv4SPD16
-me -O2 --advice:power=all --define=ccs="ccs" --define=PART_TM4C123GH6PM --define=TARGET_IS_TM4C123_RB1 -g --gcc
--diag_warning=225 --diag_wrap=off --display_error_number --gen_func_subsections=on --abi=eabi --ual -z
-m"project0_ccs.map" --heap_size=0 --stack_size=256 -i"C:/ti/ccs1010/ccs/tools/compiler/ti-cgt-arm_20.2.1.LTS/lib"
-i"C:/ti/ccs1010/ccs/tools/compiler/ti-cgt-arm_20.2.1.LTS/include" --reread_libs --diag_wrap=off
--display_error_number --warn_sections --xml_link_info="I2C_linkInfo.xml" --rom_model -o "I2C.out" "./I2C.obj"
"./startup_ccs.obj" "./uartstdio.obj" "../project0_ccs.cmd"  -llibc.a
-l"C:/ti/tivaware_c_series_2_1_4_178/driverlib/ccs/Debug/driverlib.lib"
<Linking>
remark #10371-D: (ULP 1.1) Detected no uses of low power mode state changing instructions
Finished building target: "I2C.out"

"C:/ti/ccs1010/ccs/utils/tiobj2bin/tiobj2bin" "I2C.out" "I2C.bin"
"C:/ti/ccs1010/ccs/tools/compiler/ti-cgt-arm_20.2.1.LTS/bin/armofd"
"C:/ti/ccs1010/ccs/tools/compiler/ti-cgt-arm_20.2.1.LTS/bin/armhex" "C:/ti/ccs1010/ccs/utils/tiobj2bin/mkhex4bin"


**** Build Finished ****
```

Debug

I2C [Code Composer Studio - Device Debugging]

Stellaris In-Circuit Debug Interface_0/CORTEX_M4_0 (Running)

Getting Started | Console | exit.c | I2C.c | temperature_tmp006.c

```
91    if (__TI_cleanup_ptr)  (*__TI_cleanup_ptr)();
92
93    _unlock();
94    abort();
95}
96
97
98
99
100/*****************************************************************************/
101/* ABORT - ABNORMAL PROGRAM TERMINATION.  CURRENTLY JUST HALTS EXECUTION.   */
102/*****************************************************************************/
103__attribute__((section(".text:abort")))
104void abort(void)
105{
106#if defined(EMBED_CIO_BP)
107    __asm("         .global C$$EXITE");
108#if defined(__32bis__)
109    __asm("C$$EXITE:.word 0xDEFED0FE");
110#else
111    __asm("   .align  4");
```

Terminal | Registers

COM3

```
TMP006 Temperature:
```

4. Declaration

I understand the Student Academic Misconduct Policy - http://studentconduct.unlv.edu/misconduct/policy.html

"This assignment submission is my own, original work".
Meral Abu-Jaser

Dr. Venki,

I highly appreciate that you gave an extend due date even after 2 weeks of its original submission, unfortunately, I was unable to get it to work. I find it quite difficult to get help remotely.

However, I am trying my best to stay consistent, respect the due dates and to always attend your live lectures.