Final Exam

Date: December 19, 2018

Printed Name: ________________________________ ________________________________
Last, First

Your signature is your promise that you have not cheated and will not cheat on this exam, nor will you help others to cheat on this exam. _You will not reveal the contents of this exam to others who are taking the makeup thereby giving them an undue advantage_:

Signature: ____

Instructions:

- ***Write your UT EID on all pages (at the top) and circle your instructor's name at the bottom.***
- Closed book and closed notes. No books, no papers, no data sheets (other than the last four pages of this Exam)
- No devices other than pencil, pen, eraser (no calculators, no electronic devices), please turn cell phones off.
- Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space (boxes) provided. *Anything outside the boxes will be ignored in grading*.
- You have 180 minutes, so allocate your time accordingly.
- For all questions, unless otherwise stated, find the most efficient (time, resources) solution.
- Unless otherwise stated, make all I/O accesses friendly.
- *Please read the entire exam before starting.* ***See supplement pages for Device I/O registers.***

Problem 1	10	
Problem 2	5	
Problem 3	10	
Problem 4	5	
Problem 5	15	
Problem 6	10	
Problem 7	5	
Problem 8	10	
Problem 9	10	
Problem 10	20	
Total	100	

(10) Problem 1. Give a one-to-three-word answer for each question.

(1) Part a) What data structure do you use to stream data from an ISR to the main program given the situation where data arrives into the ISR bursts but is processed one byte at a time in the main? ……

(1) Part b) With UART transmission we send one start bit, 8 data bits and one stop bit. What term do we use to define these 10 bits? ……………………………………………………………………….…………

(1) Part c) What qualifier do we add to an otherwise local variable (scope within a function) so that the variable is defined in permanently in RAM? ………………………………..……………………………

(1) Part d) What qualifier do we add to an otherwise global variable so that the scope is restricted to software located within that same file? …………………………………………………………………

(1) Part e) What graphical structure describes the modularity of a system, such that circles and rectangles are modules and arrows represent information as it passes from one module to another? ……………………

(1) Part f) What term is used to describe the smallest difference in input voltage that an ADC can reliably distinguish? ………………………………………………………………………………………

(1) Part g) What are the units of electrical power? Give as a one-word answer, and not as a combination of other units. …………………………………………………………………………………

(1) Part h) In a real-time system, it is important to respond to critical events. What is the term used to describe the delay between the time a critical event occurs and the time the event is processed? For example, the time between touching a switch and the time the software recognizes the switch is touched. …………….

(1) Part i) What is the name of the number system where the value 4.125 is represented with the integer 264 (4.125 = 264*0.015625)? …………………………………………………..…………

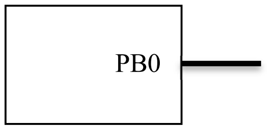
(1) Part j) What is the debugging term used in Lab 3 to store important information into arrays? This Debugging technique can be used to replace printing (printf) information while the program is running. ……..

(5) Problem 2. Consider the sampling rate chosen for the ADC in Lab 7. Give the relationship for the slowest possible sampling rate (f_s, in Hz), given these parameters: ADC range (V, in volts), number of ADC bits (n, in bits, e.g., 12 bits) and rate at which one moves the slide pot (r, in oscillations per sec).

(10) Problem 3: Design a 6-bit DAC connected to Port B using PB5 to PB0. Show the circuit and label all resistors, capacitors and interface chips needed. Make PB0 the least significant bit and make PB5 the most significant bit.

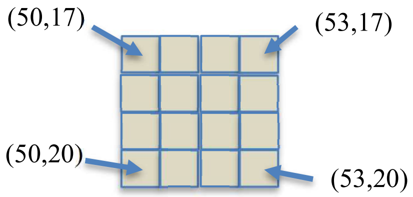
PB5
PB4
PB3
PB2
PB1
PB0
Out

(5) Problem 4. Interface an LED to Port B bit 0 using positive logic. The desired operating point of the LED is 1V and 2 mA. V_{OL} is 0.4V and V_{OH} is 3.1V. Show the circuit and label all resistors, capacitors and interface chips needed. Show the math need to calculate resistor value(s).



(15) Problem 5. Consider a game that has 50 boxes. There is an array specifying the current status of each box. Each box is 4 by 4 pixels, and has an (x,y) coordinate, a velocity, a direction, and a life parameter. You may assume the `box` array has been populated with data before your function is called. The figure on the right shows one example box at (x,y)=(50,20)

```
typedef enum {dead,alive} status_t;
struct abox {
  int16_t x;        // x coordinate, in pixels
  int16_t y;        // y coordinate, in pixels
  int16_t velocity; // velocity, in pixels/frame
  int16_t angle;    // direction, in degrees
  status_t life;};  // dead or alive
typedef abox box_t;
```



Each box has 16 pixels in the game world, occupying the square space from (x,y) to (x+3, y-3). *Implement a C function* that searches to see if two alive boxes are overlapping (the location of any of the 16 pixels of one box is equal to any of the 16 pixels of another box). If two alive boxes occupy overlapping space, set the life parameter of both boxes to dead. Do not worry about 3 or more boxes overlapping the same space.

```
void Search(box_t box[]){ // 50 elements
```

(10) Problem 6. Draw the state transition graph for a Moore FSM used to control 6 taillights on a car. There are two inputs and 6 outputs. If the input is 0, the output is 0. If the input is 1 (turn right), the output cycles through the values 4 2 1 every ½ second. If the input is 2 (turn left), the output cycles through the values 8,16,32 every ½ second. If the input is 3 (brake) the output is 63. Each state has a name, an output, a dwell time, and multiple arrows to next states. In a STG you can assign the symbol X for an arrow to mean "for all possible input values".

(5) Problem 7. Assume the UART0 has been initialized for busy-wait synchronization. Design a C function with these four steps

1) Wait for new serial port input (RXFE is bit 2 of UART0->STAT)
2) Read the new 8-bit ASCII character data, read UART0->RXDATA
3) Echo the data by transmitting the same 8-bit data just received, write UART0->TXDATA
4) Return by value the one character received.

Show what you would place in the .h file, including comments

Show what you would place in the .c file

(10) Question 8. The subroutine **mySub** uses a call by value parameter passed on the stack. There are no return parameters. Call by value means the data itself is pushed on the stack. This is not AAPCS compliant. A typical calling sequence is

```
      .text
stuff: .long  123        // 32-bit constant
start: LDR  R0,=stuff
       LDR  R0,[R0]
       PUSH {R0}         // the value of the input parameter is pushed
       MOVS R0,#0        // no cheating, parameter not in R0, on stack
       BL   mySub
       ADD  SP,SP,#4     // discard parameter
```

The subroutine allocates one 32-bit local variable, **i**, and uses SP stack pointer addressing to access the local variable and the parameter. The binding for these two are

```
.equ in, [          ]  //binding for 32-bit value that is the input parameter

.equ i,  [          ] //binding for 32-bit local variable
mySub: PUSH {R4-R7,LR}
    [              ]//allocate i

//--------start of body-------------------
    LDR  R4,[SP,#in]  //Reg R11 is the input parameter value
    STR  R4,[SP,#i]   //save parameter into local i
//--------end of body---------------------
    [              ]//deallocate i

    POP  {R4-R7,PC}
```

In the boxes provided, show the binding for **in**, the binding for the local variable **i**, the assembly instruction(s) to allocate **i**, and the assembly instruction(s) to deallocate **i**.

(10) Question 9: Write C code to maintain the elapsed time in minutes. I.e., increment the global variable **Time** once a minute. Include both the initialization (arm and enable interrupts), and the ISR (maintain **Time**). Do not worry about priority. Assume the bus clock is 16 MHz. You may add additional variables of whatever type you wish. Note that 2^24=16,777,216.

```
uint32_t Time; // in minutes
// interrupts when counter goes from 1 to 0
void SysTick_Init(void){
  SysTick->CTRL  = 0x00;      // disable during initialization

  SysTick->LOAD  =

  SysTick->VAL   = 0;         // clear count, cause reload
  SysTick->CTRL  = 0x07;      // Enable SysTick IRQ and SysTick Timer
}
void SysTick_Handler(void){









}
```

(20) Question 10: You are asked to implement a simple postfix calculator as a subroutine in assembly language. The input (call by reference in R0) to your function is a null-terminated character string with the following 12 valid characters 0,1,2,3,4,5,6,7,8,9,+,and*. You may assume all strings are valid and calculate exactly one 32-bit output value. For example

`"5"`	returns 5
`"79+"`	returns 7+9 = 16
`"58*1+"`	returns (5*8)+1 = 41
`"92*7+4*52+*"`	returns (((9*2)+7)*4)*(5+2)) = ((18+7)*4)*7) = ((25*4)*7) = (100*7) =700

The basic idea is to fetch a character from the string:

- if it is a + or * operator, pop two numbers from the stack, unsigned 32-bit operate, and push the result
- if it is a digit, push the value (0 to 9) of the digit as a 32-bit value onto the stack
- if it is the null termination, pop one 32-bit value from the stack and return that value in R0

```
//input: R0 points to the string to process
//output: R0 contains the 32-bit value
Calc:
```