King Fahd University of Petroleum and Minerals
College of Computer Science and Engineering
Computer Engineering Department

COE 306: INTRODUCTION TO EMBEDDED SYSTEMS

Term 161 (Fall 2016-2017)
Final Exam
Saturday Jan. 14, 2016

Time: 120 minutes, Total Pages: 9

Name:_KEY____________________________ ID:____________________ Section: ________

Notes:

- Do not open the exam book until instructed
- Answer all questions
- All steps must be shown
- Any assumptions made must be clearly stated

Question	Max Points	Score
Q1	27	
Q2	14	
Q3	8	
Q4	11	
Q5	15	
Total	75	

Dr. Aiman El-Maleh

[27 Points]

(Q1) Fill in the blank in each of the following questions:

(1) In a real time operating system, failure to meet hard deadlines causes system failure while failure to meet soft deadlines causes degraded response.

(2) Real time operating systems are based on the concepts of preemption and context switching. Preemption is the ability to interrupt a process to switch to another.

(3) A process can be in one of three states: executing on the CPU, ready to run or waiting for I/O, another process, timer, next period.

(4) The context of a process is the set of registers that define a process.

(5) Techniques for handling race conditions in shared memory that could lead to data inconsistency include atomic test-and-set instruction, semaphore and critical region.

(6) Disadvantages of using shared memory inter-process communication include that it needs concurrency control and lack of data protection from Operating System.

(7) In message passing using pipes, one process writes data into tail end of pipe while another process reads data from head end of the pipe.

(8) Using datagram socket, packets are independent with not necessarily in-order delivery and no guarantee of delivery.

(9) The advantage of serial transmission in comparison to parallel transmission is that it is cheap while the advantage of parallel transmission is that it is fast.

(10) The advantage of using differential signals in serial transmission is that it doubles the signal to noise ratio.

(11) Synchronous (synchronous/asynchronous) transmission is used for high-speed communication between computers.

(12) In simplex transmission, data flow is only in one direction while in duplex transmission, data flow is in both directions simultaneously.

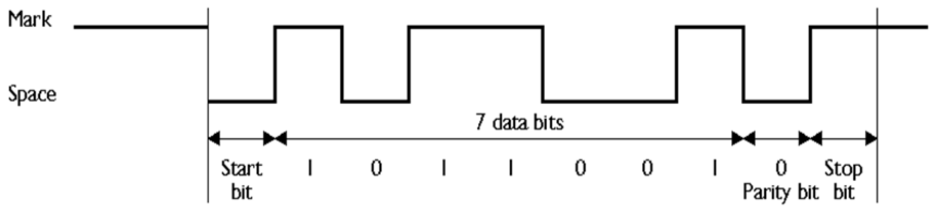
(13) Given 9600 baud rate and 8 voltage levels used for transmitting each symbol, the bit rate is 9600 * 3 = 28800 bps.

(14) Given a protocol with 3 bits of protocol (start, stop and parity), 7 bits of data, 9600 baud rate, and 1 bit per symbol (binary), the information rate is 9600 * 1 * 7/10 = 6720 bps.

(15) SPI has higher (higher/lower) throughput than I^2C.

(16) In SPI, transmission involves two shift registers one in master and one in slave connected in a virtual ring topology.

(17) Using UART, transmitting character 'M'=0x4D with even parity, 7 data bits and one stop bit, the following data waveform is generated:



(18) In UART, a framing error occurs when the stop bit of a received character is a logic 0.

(19) In UART, an overrun error occurs when a new character is assembled while the receiver buffer or FIFO is full.

(20) In UART, a break condition is detected when receiver line is held in the spacing state (all zeroes) for one full character transmission.

[14 Points]

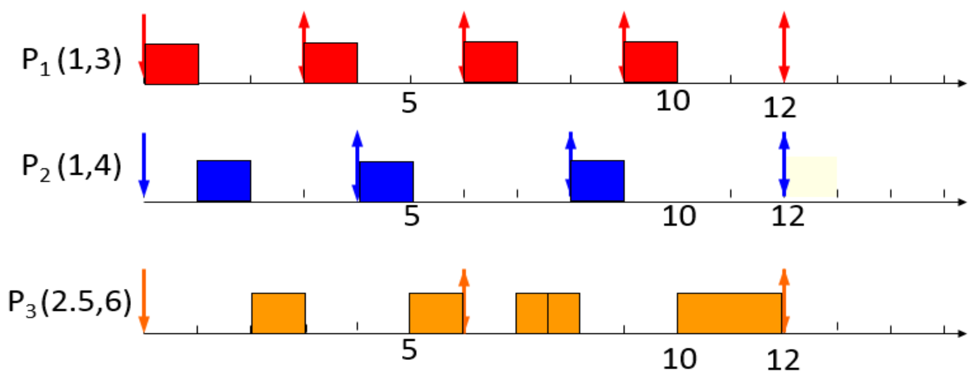
(Q2) Assume that the following periodic processes are ready at time 0. We would like to schedule these processes using RMS and EDF scheduling techniques. In each case, compute the schedule for an interval equal to the least-common multiple of the periods of the processes. If processes have the same priority, schedule processes to minimize preemption and process waiting time. If processes have the same priority under all these criteria then give priority in the following order P1, P2, P3.

Process	**Execution Time**	**Period (Deadline)**
P1	1	3
P2	1	4
P3	2.5	6

(a) Compute CPU utilization.

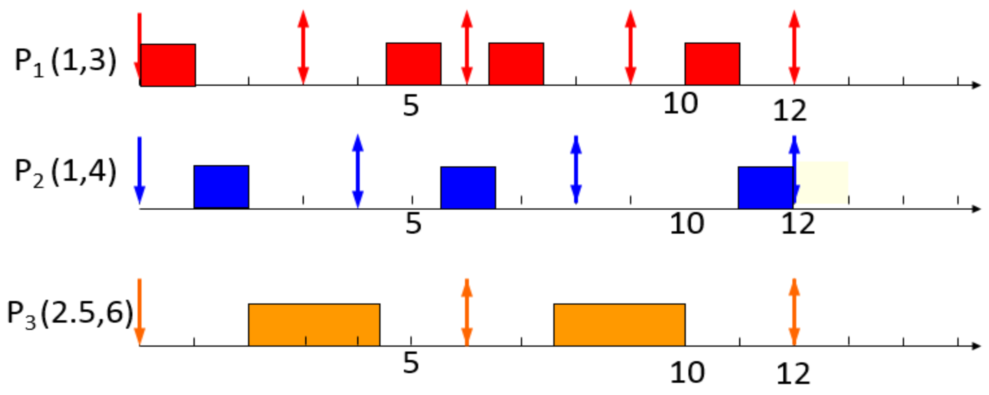
CPU Utilization = 1/3 + 1/4 + 2.5/6 = 1.0 = 100%

(b) Schedule the processes using an RMS policy. If a schedule does not meet the required deadlines for all processes, complete the schedule and indicate this in your solution.



P3 has missed meeting its deadline in the first period.

(c) Schedule the processes using an EDF policy. If a schedule does not meet the required deadlines for all processes, complete the schedule and indicate this in your solution.



(d) Is it possible to find a feasible schedule where all deadlines for all processes are met if the execution time of P2 is increase to 1.5? Justify your answer.

No, it is not possible as CPU utilization will exceed 100%.

[8 Points]

(Q3) Three periodic tasks T1, T2 and T3 are to be executed on a system with a single timer. The timer is configured to expire every 500µs. The task periods are 0.25s for T1, 0.5s for T2 and 1s for T3.

Write a timer interrupt handler (ISR) in C that runs the tasks according to their periods using a single counter. Assume that each task is run by calling a function of its name, e.g. T1().

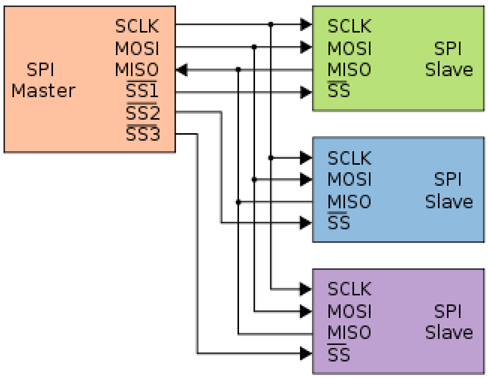
```
void timer_handler() {
  // static variables keep their values across calls
  // a global variable works too
  static int counter = 0;

  counter++;

  if (counter == 500 || counter == 1500) {
          T1();
  }

  if (counter == 1000) {
          T1();
          T2();
  }

  if (counter == 2000) {
          T1();
          T2();
          T3();
          counter = 0;
  }

}
```

[11 Points]

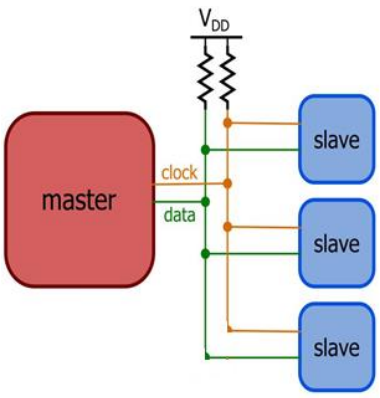
(Q4)

i. It is required to interface a microcontroller as a master to three peripheral devices as slaves. **(6 points)**

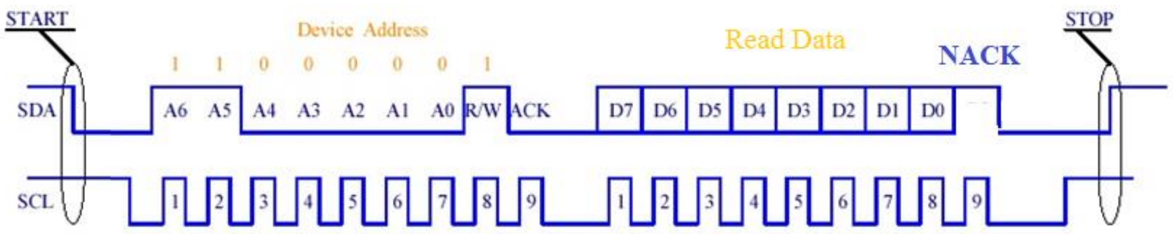
a. Show the block diagram interconnection of the master and slaves using SPI interface.



b. Show the block diagram interconnection of the master and slaves using I^{2}C interface.

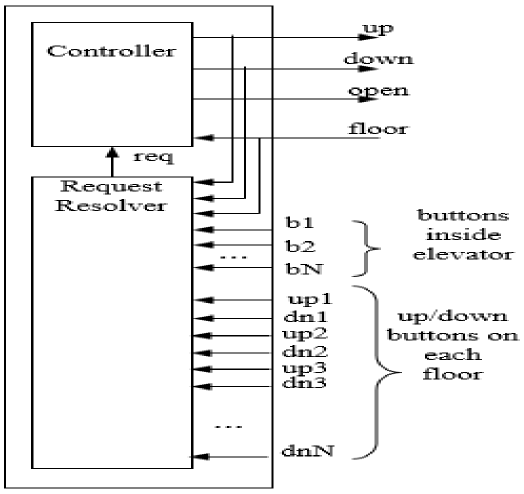


ii. Suppose that a master is interfaced to a slave using I^{2}C interface. Assume that the slave address is 1100000. Show the timing diagram for the data and the clock line for reading a single byte from the slave. **(5 points)**

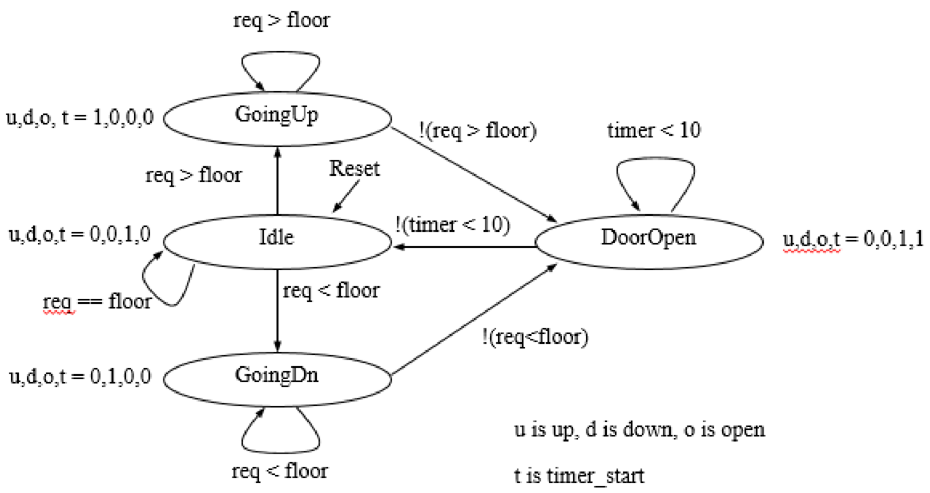


[15 Points]

(Q5) An elevator system of an N-Story building is composed of two units, a controller unit and a request resolver unit as shown below. The request resolver unit receives N buttons inside the elevator to indicate the desired floor. It also receives two buttons from each floor one for going up and another for going down requests. In addition, it receives a floor input indicating the current floor level and the two signals up and down indicating the direction of movement of the elevator. The controller unit receives the requested floor from the request resolver unit and a floor input indicating the current floor level. It generates three control signals as outputs: up, down and open. The signals up and down control the direction of movement of the elevator while the open signal control the elevator door.



The system will move the elevator either up or down to reach the requested floor. Once at the requested floor, the elevator door is opened for at least 10 seconds, and is kept open until the requested floor changes. It must be ensured that the elevator door is never open while the elevator is moving.
The state machine diagram for the elevator controller unit is given below. In the FSM, the symbols U, D, O, T represent elevator going up, elevator going down, door is open, and start timer, respectively.



Write a C program that implements this state machine of the elevator controller. Assume that any change in the inputs **req** and **floor** will cause interrupts to update their values. You do not need to take care of that. Use Timer0 to implement the required timing requirement. Show the code for TIMER0_IRQHandler(). To use Timer0 interrupt, you need to use NVIC_EnableIRQ(1). Note that for TC registers to start counting, the least significant bit of the Timer Control Register (`LPC_TIM0->TCR`) must be set. To clear the MR0 interrupt flag, set the least significant bit in the Interrupt Register (`LPC_TIM0->IR`). Information on how timer actions using the MCR register are given below:

MCR bit	**Bit value = 1**	**Bit value = 0**
0	Enable timer interrupt	Disable timer interrupt
1	Reset TC	Disable this feature
2	Stop TC	Disable this feature

Timer0 match and MCR registers are `LPC_TIM0->MR0` and `LPC_TIM0->MCR`. Assume that the clock frequency is 25 MHZ.

```
#define IDLE         0
#define GOINGUP      1
#define GOINGDN      2
#define DOOROPEN 3

int timer=0;
int req=0, floor=0;
int up, down, open, timer_start;

void UnitControl() {

  NVIC_EnableIRQ(1);

  int state = IDLE;

  while (1) {
    switch (state) {
      IDLE: up=0; down=0; open=1; timer_start=0;
        if  (req==floor) {state = IDLE;}
        if  (req > floor) {state = GOINGUP;}
        if  (req < floor) {state = GOINGDN;}
        break;

     GOINGUP: up=1; down=0; open=0; timer_start=0;
        if  (req > floor) {state = GOINGUP;}
        if  (!(req>floor)) {state = DOOROPEN;}
        break;

     GOINGDN: up=0; down=1; open=0; timer_start=0;
        if  (req < floor) {state = GOINGDN;}
        if  (!(req<floor)) {state = DOOROPEN;}
        break;
```

```
         DOOROPEN: up=0; down=0; open=1;
             if (timer_start==0){
                 timer=0; timer_start=1;
                 LPC_TIM0->TCR |= 1;
                 LPC_TIM0->MR0 = 250000000; // clk frequency = 25 MHZ
                 LPC_TIM0->MCR |= 5;
             }
            if (!timer) {state = DOOROPEN;}
            else {state = IDLE;}
            break;
      }
   }
}

void TIMER0_IRQHandler() {
   timer=1;
   LPC_TIM0->IR |= 1;
}
```