# IE4-MC Lab Session #2: Preparation sheet

|  |  |
| --- | --- |
| **Surname, First Name** | **Group Team** |
| **1.**  **2.**  **3.** | 1 🞎 A B C D E F  2 🞎  3 🞎 |

This preparation sheet must be filled and uploaded to EMIL by every student before the lab session. Cooperation within the lab team (of 2-3 students) is possible (same solution). However, no cooperation across lab teams.

## Question 1:

Give the 8 x n-matrix and the corresponding hexadecimal value which you want to display on the LED pendulum.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LED7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LED6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LED5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LED4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LED3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LED2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LED1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LED0 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| HEX CODE | 0x00 | 0x00 | 0x00 | 0xFF | 0x18 | 0x18 | 0x18 | 0xFF | 0x00 | 0x00 | 0x00 | 0x00 | 0x00 | 0x3F | 0x48 | 0x88 | 0x48 | 0x3F | 0x00 | 0x00 | 0x00 | 0x00 | 0x00 | 0xF8 | 0x06 | 0x01 | 0x06 | 0xF8 | 0x06 | 0x01 | 0x06 | 0xF8 | 0x00 | 0x00 | 0x00 |

## Question 2:

In order to realize a stable pattern on the LED pendulum, the character string has to be aligned (synchronized) to the turning points indicated by an edge of the -signal.



Give C code that detects the left and the right turning point of the pendulum. Assume that the -signal is connected to PD(0) and that the PORT D has been correctly configured (clock activated, port enabled and directions set).



|  |
| --- |
| unsigned char symbols[] = {0x00, 0x00, 0xFF, 0xFF, 0x18,  0x18, 0x18, 0xFF, 0xFF, 0x00, 0x00,//H  0x07, 0x1C, 0x38, 0x48, 0xC8,  0x48, 0x38, 0x1C, 0x07, 0x00, 0x00,//A  0x08, 0x70, 0x18, 0x06, 0x03,  0x06, 0x18, 0xE0, 0xE0, 0x18,  0x06, 0x03, 0x06, 0x18, 0x70,  0x80, 0x00, 0x00};//W  void main()  {  //Lab2 task 1 - catch rising & falling edges  SYSCTL\_RCGCGPIO\_R |= 1<<4; //activate port D  while(!(SYSCTL\_RCGCGPIO\_R & 0x08)); //wait until port D is activated  GPIO\_PORTD\_DEN\_R |= 0x01; //eanble first bit only  GPIO\_PORTD\_DIR\_R = 0x00; //set the first to 0(means as input to chip)  char edgeDetection = 0;  while(1)  {  if((GPIO\_PORTD\_DATA\_R & 0x01) && !edgeDetection)  {  edgeDetection = 1; //rising edge (LEFT)  }  else if(!(GPIO\_PORTD\_DATA\_R & 0x01) && edgeDetection)  {  edgeDetection = 0; //falling edge (RIGHT)  }  }  } |

## Question 3:

Develop a function **timerConfig(void)** that configures TIMER0A as a 16 bit periodic timer that can represent times up to 10 ms (i.e. does not overflow within 10ms). Give the C-Code:

|  |
| --- |
| //should represent TimerA0 settings -compare, count down, no capture  void timerConfig(void)  {  // configure Timer 0  SYSCTL\_RCGCTIMER\_R |= (1<<0); // timer 0  while(!(SYSCTL\_PRTIMER\_R & (1<<0))); // wait for timer 0 activation  TIMER0\_CTL\_R &= ~0x0001; // disable Timer 0  TIMER0\_CFG\_R = 0x04; // 2 x 16-bit mode  // compare mode, down, periodic: TAMR=0x2 -no match value  TIMER0\_TAMR\_R |= 1<<1; //last 8 bits: 0000 0010  //4th bit = counts down, 2nd bit = 0x02 - periodic    } |

## Question 4:

Develop a function **timerWait(unsigned short usec)** that realizes a delay by Timer 0A and uses the configuration of timerConfig(). The function sets the interval load value, enables the timer, waits for the time-out of the timer, clears the interrupt flags and disables the timer. usec is the value in µs after after which the timer reaches time-out.

|  |
| --- |
| //make a delay of usec in microsec (up to 10ms)  void timerWait(unsigned short usec)  {  timerConfig();  long fCPU = 16 \* pow(10.0, 6);  long pre = ceil((fCPU / pow(2.0, 16)) \* 0.01); //prescaler Tmax = 10ms  TIMER0\_TAPR\_R = pre - 1;  TIMER0\_TAILR\_R = ceil((fCPU / pre) \* (double)usec \* pow(10.0, -6)) - 1; //load value  TIMER0\_CTL\_R |= 0x0001; //start TImer0A  if(!(TIMER0\_RIS\_R & 0x0001) //if the time-out raw interrupt occur 0x0000  {  TIMER0\_ICR\_R |= 1 << 0; //clear TIMER0\_RIS\_R(0) bit  TIMER0\_CTL\_R &= ~0x0001; //disable the Timer0A  }  } |