**Part 1. Preliminary Design Report: Developing Codes (30 points: 3 X 10 points)**

CS224

Section No.: 05

Spring 2018

Lab No.: 07

Asuman Aydın/21502604

.

a) Research and read about SFRs. Explain the differences between TRISx, PORTx, LATx and ODCx ports. Specify the special function registers (SFRs) for the I/O device(s) involved in Part2.a and Part2.b.

TRISx: Data Direction register : controls the bits whether it is 1 which is input or 0 which is output.

PORTx: I/O Port register : it reaches the data comes from I/O, reads it while writing in to the data latch.

LATx: I/O Latch register : it evaluates the control and takes the problems that can happen while reading-writing stages.

ODCx ports: Open-drain Control Register: it determines if the pin has normal digit output or open-drain output.

b) Give the C code for Part2.a, with comments, an explanatory header, well-chosen identifiers and good use of spacing and layout to make your program self-documenting.

/\*

\* Project name:

    Led\_Blinking (The simplest simple example)

\* Copyright:

    (c) Mikroelektronika, 2012.

\* Revision History:

    20120810:

      - initial release (FJ);

\* Description:

    Simple "Hello world" example for the world of PIC32 MCUs;

\* Test configuration:

    MCU:            P32MX795F512L

                     http://ww1.microchip.com/downloads/en/DeviceDoc/61156F.pdf

    Dev.Board:       EasyPIC Fusion v7 - ac:LED

                     http://www.mikroe.com/easypic-fusion/

    Oscillator:      XT-PLL, 80.0000MHz

    Ext. Modules:    None.

    SW:            mikroC PRO for PIC32

                     http://www.mikroe.com/mikroc/pic32/

\* NOTES:

    - Turn ON PORTA, B, C, D, F & G LEDS at SW15. (board specific)

\*/

unsigned char reverse(PORTD);

void main() {

 AD1PCFG = 0xFFFF;      // Configure AN pins as digital I/O

 JTAGEN\_bit = 0;        // Disable JTAG

 TRISD = 0;             // Initialize PORTD as output

 TRISF = 1;

 LATD = 255;

 LATF = 255;

//normal displayin in pushing first button

 if(PORTFbits.RF0 == 0)

{

    Delay\_ms(100);

    if(PORTFbits.RF0 == 0)

    {

    do{

         LATD = PORTD--;

         Delay\_ms(100);

         } while(1);

   }

 }

 //reverse displayin in pushing second button

 else if(PORTFbits.RF1 == 0)

 {

      Delay\_ms(100);

      reverse(PORTD);

    if(PORTFbits.RF1 == 0)

    {

    do{

         LATD = PORTD--;

         Delay\_ms(100);

         } while(1);

   }

 }

else if(PORTFbits.RF1 == 0 && PORTFbits.RF0 == 0)

 //doesnot show anything

 else if PORTFbits.RF1 == 0 && PORTFbits.RF0 == 0)

 {

   }

 }

 //time getting faster while displayin in pushing fourth button

  else if(PORTFbits.RF3 == 0)

 {

      Delay\_ms(1000);

    if(PORTFbits.RF3 == 0)

    {

    do{

         LATD = PORTD--;

         Delay\_ms(1000);

         } while(1);

   }

 }

 //time slowing  in pushing third button

 else if(PORTFbits.RF2 == 0)

 {

      Delay\_ms(1000);

    if(PORTFbits.RF2 == 0)

    {

    do{

         LATD = PORTD--;

         Delay\_ms(500);

         } while(1);

   }

 }

 else if(PORTFbits.RF3 == 0 && PORTFbits.RF2 == 0)

{

    Delay\_ms(100);

Char I = 500;

    if(PORTFbits.RF3 == 0 && PORTFbits.RF2 == 0)

    {

    do{

         LATD = PORTD--;

         Delay\_ms(i);

i = i +i;

         } while(1);

   }

}

unsigned char reverse(PORTD)

{

char i= 7; //size-1

        char reversed = 1; //number

char number = reversed;

number = number >>1;

         PORTD = 0;

        //shifting 8 times to reverse and store the value in portd

      While(number)

{

          PORTD = reversed;

             Delay\_ms(100);

             reversed = reversed <<1;

reversed |= num &1; //or

num >>= 1;

i--;

         }

reversed = reversed << count;

return reversed;

}

c) Give the C code for Part2.b, with comments, an explanatory header, well-chosen identifiers and good use of spacing and layout to make your program self-documenting.

/\*

\* Project name:

    Led\_Blinking (The simplest simple example)

\* Copyright:

    (c) Mikroelektronika, 2012.

\* Revision History:

    20120810:

      - initial release (FJ);

\* Description:

    Simple "Hello world" example for the world of PIC32 MCUs;

\* Test configuration:

    MCU:            P32MX795F512L

                     http://ww1.microchip.com/downloads/en/DeviceDoc/61156F.pdf

    Dev.Board:       EasyPIC Fusion v7 - ac:LED

                     http://www.mikroe.com/easypic-fusion/

    Oscillator:      XT-PLL, 80.0000MHz

    Ext. Modules:    None.

    SW:            mikroC PRO for PIC32

                     http://www.mikroe.com/mikroc/pic32/

\* NOTES:

    - Turn ON PORTA, B, C, D, F & G LEDS at SW15. (board specific)

\*/

unsigned char reverse(PORTD);

void main() {

 AD1PCFG = 0xFFFF;      // Configure AN pins as digital I/O

 JTAGEN\_bit = 0;        // Disable JTAG

 TRISD = 0;             // Initialize PORTD as output

 TRISF = 1;

 LATD = 255;

 LATF = 255;

//normal displayin in pushing first button

 if(PORTFbits.RF0 == 0)

{

    Delay\_ms(100);

    if(PORTFbits.RF0 == 0)

    {

    do{

         LATD = PORTD--;

         Delay\_ms(100);

         } while(1);

   }

 }

 //reverse displayin in pushing second button

 else if(PORTFbits.RF1 == 0)

 {

      Delay\_ms(100);

      reverse(PORTD);

    if(PORTFbits.RF1 == 0)

    {

    do{

         LATD = PORTD--;

         Delay\_ms(100);

         } while(1);

   }

 }

 //time slowing  in pushing third button

 else if(PORTFbits.RF2 == 0)

 {

      Delay\_ms(1000);

    if(PORTFbits.RF2 == 0)

    {

    do{

         LATD = PORTD--;

         Delay\_ms(500);

         } while(1);

   }

 }

 //time getting faster while displayin in pushing fourth button

  else if(PORTFbits.RF3 == 0)

 {

      Delay\_ms(1000);

    if(PORTFbits.RF3 == 0)

    {

    do{

         LATD = PORTD--;

         Delay\_ms(1000);

         } while(1);

   }

 }

}

unsigned char reverse(PORTD)

{

char i= 7; //size-1

        char reversed = 1; //number

char number = reversed;

number = number >>1;

         PORTD = 0;

        //shifting 8 times to reverse and store the value in portd

      While(number)

{

          PORTD = reversed;

             Delay\_ms(100);

             reversed = reversed <<1;

reversed |= num &1; //or

num >>= 1;

i--;

         }

reversed = reversed << count;

return reversed;

}