HOLY SPIRIT UNIVERSITY OF KASLIK

Faculty of Engineering

Department of Computer Engineering

Dual Seven Segment Display

GEL 575

Presented to:

Dr. Bechara NEHME

Prepared By:

Adib RACHID and Charbel ZALAKET

Problem:

Using the PIC24FJ256DA210 microcontroller, we would like to display the count of two decimal digits on two seven segment displays (one for units and one for tens). The count will start at a number entered by the user using 7 bush buttons connected to PORTA (RA0-RA6). Once the user clicks on an additional push button connected to RA7 the starting value will be read and the count will start. The count will increase to reach 99 and roll over to 0 and increase again to 99. The delay between two count displays is 1 second. Once the RA7 push button is pressed again the count will stop and the microcontroller will wait another input on PORTA and another click on RA7.
The two seven segment displays are connected to PORTB (RB0-RB6) the seven segment displays are controlled via RA9 and RA10. Do your research to find the best switching frequency between the two seven segment displays so that the user will never notice they are switching ON and OFF.

1) Write the program that will answer the needed task

2) Give a the schematic circuit of the system

Solution:

1. The code:

```
/*
 * File:   projet.c
 * Author: adibrachid and charbelzalaket
 *
 * Created on November 9, 2017, 3:46 PM
 */

// PIC24FJ256DA210 Configuration Bit Settings

// 'C' source line config statements

// CONFIG4

// CONFIG3
#pragma config WPFP = WPFP255          // Write Protection Flash Page Segment Boundary (Highest Page (same as page 170))
#pragma config SOSCSEL = SOSC          // Secondary Oscillator Power Mode Select (Secondary oscillator is in Default (high drive strength) Oscillator mode)
#pragma config WUTSEL = LEG            // Voltage Regulator Wake-up Time Select (Default regulator start-up time is used)
#pragma config ALTPMP = ALPMPDIS       // Alternate PMP Pin Mapping (EPMP pins are in default location mode)
#pragma config WPDIS = WPDIS           // Segment Write Protection Disable (Segmented code protection is disabled)
#pragma config WPCFG = WPCFGDIS        // Write Protect Configuration Page Select (Last page (at the top of program memory) and Flash Configuration Words are not write-protected)
#pragma config WPEND = WPENDMEM        // Segment Write Protection End Page Select (Protected code segment upper boundary is at the last page of program memory; the lower boundary is the code page specified by WPFP)

// CONFIG2
#pragma config POSCMOD = HS            // Primary Oscillator Select (HS Oscillator mode is selected)
#pragma config IOL1WAY = ON            // IOLOCK One-Way Set Enable (The IOLOCK bit (OSCCON<6>) can be set once, provided the unlock sequence has been completed. Once set, the Peripheral Pin Select registers cannot be written to a second time.)
#pragma config OSCIOFNC = OFF          // OSCO Pin Configuration (OSCO/CLKO/RC15 functions as CLKO (FOSC/2))
```

```
#pragma config FCKSM = CSDCMD          // Clock Switching and Fail-Safe Clock Monitor (Clock switching and Fail-Safe Clock Monitor are disabled)
#pragma config FNOSC = PRI             // Initial Oscillator Select (Primary Oscillator (XT, HS, EC))
#pragma config PLL96MHZ = ON           // 96MHz PLL Startup Select (96 MHz PLL is enabled automatically on start-up)
#pragma config PLLDIV = DIV12          // 96 MHz PLL Prescaler Select (Oscillator input is divided by 12 (48 MHz input))
#pragma config IESO = ON               // Internal External Switchover (IESO mode (Two-Speed Start-up) is enabled)

// CONFIG1
#pragma config WDTPS = PS32768         // Watchdog Timer Postscaler (1:32,768)
#pragma config FWPSA = PR128           // WDT Prescaler (Prescaler ratio of 1:128)
#pragma config ALTVREF = ALTVREDIS     // Alternate VREF location Enable (VREF is on a default pin (VREF+ on RA9 and VREF- on RA10))
#pragma config WINDIS = OFF            // Windowed WDT (Standard Watchdog Timer enabled,(Windowed-mode is disabled))
#pragma config FWDTEN = ON             // Watchdog Timer (Watchdog Timer is enabled)
#pragma config ICS = PGx1              // Emulator Pin Placement Select bits (Emulator functions are shared with PGEC1/PGED1)
#pragma config GWRP = OFF              // General Segment Write Protect (Writes to program memory are allowed)
#pragma config GCP = OFF               // General Segment Code Protect (Code protection is disabled)
#pragma config JTAGEN = ON             // JTAG Port Enable (JTAG port is enabled)

// #pragma config statements should precede project file includes.
// Use project enums instead of #define for ON and OFF.


#include <xc.h>
#define FCY 4000000UL
#include <libpic30.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <p24FJ256DA210.h>
```

```
int main(int argc, char** argv) {
    //7 push buttons bits
    TRISAbits.TRISA0=1;
    TRISAbits.TRISA1=1;
    TRISAbits.TRISA2=1;
    TRISAbits.TRISA3=1;
    TRISAbits.TRISA4=1;
    TRISAbits.TRISA5=1;
    TRISAbits.TRISA6=1;
    //8th bith to des/activate
    TRISAbits.TRISA7=1;
    //balaillage 2 bits
    TRISAbits.TRISA9=0;
    TRISAbits.TRISA10=0;

    //7 segments
    TRISB=0;
    PORTB=0;
   while(1){
      while(PORTAbits.RA7!=1){
         //wait until user activate bit #7
      }
      int
dec=PORTAbits.RA6*pow(2,6)+PORTAbits.RA5*pow(2,5)+PORTAbits.RA4*pow(2,
4)+PORTAbits.RA3*pow(2,3)+PORTAbits.RA2*pow(2,2)+PORTAbits.RA1*pow(2,1)
+PORTAbits.RA0*pow(2,0);
      if(dec>99)
         dec=99;

      __delay_ms(1000); //to deactivate, user should press A7 t seconds t between 0 and 1
to make it happen between a number transition.
      while(PORTAbits.RA7!=1){
         //wait until user deactivate bit #7
         display(dec);
         dec++;
         if(dec==100)
            dec=0;

      }

   }

    return (EXIT_SUCCESS);
}
```

```
void display(int dec){
   int d1,d2;
   d1=dec%10;
   d2=(dec/10)%10;
   int i=0;
   while(i<20){
      show(d1);
      PORTAbits.RA9=1;
      __delay_ms(25);
      PORTAbits.RA9=0;
      show(d2);
      PORTAbits.RA10=1;
      __delay_ms(25);
      PORTAbits.RA10=0;
      i++;
   }
}
void show(int dec){
   switch(dec){
      case 0:
            PORTB=0x0040;
            break;
      case 1:
            PORTB=0x004F;
            break;
      case 2:
            PORTB=0x0012;
            break;
      case 3:
            PORTB=0x0006;
            break;
      case 4:
            PORTB=0x000D;
            break;
      case 5:
            PORTB=0x0030;
            break;
      case 6:
            PORTB=0x0020;
            break;
      case 7:
            PORTB=0x0031;
```

```
            break;
        case 8:
            PORTB=0x0000;
            break;

        case 9:
            PORTB=0x0004;
            break;
    }
}
```

2. <u>The Schematic Circuit:</u>