Quang Nguyen

CPE 409 Lab 8

# Goals

* To write code to improve the existing LCD library
* To learn how to improve embedded systems’ code by eliminating delay loop
* Read timing diagram and a write software in accordance to timing requirements

# Equipment used

## Hardware

* Microchip Explorer 16 board
* PIC kit 3

## Software

* MPLAB X IDE 2.00

# Design Specifications

* Must eliminate delay loop in lcd\_cmd() and lcd\_data().
* Must write code that will meet timing requirements for LCD driver

# Design

* checkBusy() function was used to replace the delay loops used by lcd\_cmd() and lcd\_data()
  + Refer to figure 1 for the flow diagram of checkBusy().

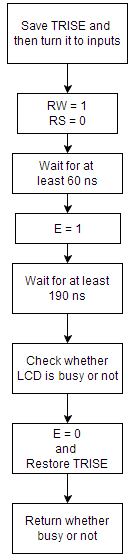


Figure 1: Flow diagram of checkBusy()

# Verification

* The program of lab 7 was verified to function correctly using the improved library.

# Questions

**Why is it much better to call checkbusy at the beginning of the lcd\_data and lcd\_cmd functions than at the end of the function (that is enter a loop after writing that checks for busy and when it is not busy return from the function).**

* It is not a good idea to use checkBusy() at the end of the function because doing so would be a waste of processor’s cycle. The program might want to perform another task right after executing lcd\_data or lcd\_cmd and the LCD might be ready for new data by the next time that processor wants to execute the function again.

# Conclusions and Limitations

* No limitations were found and the program works just as expected.

# Programming Code

void lcd\_cmd( char cmd ) // subroutiune for lcd commands

{

while(checkBusy());

// TRISD &= 0xFF00; // ensure RD0 - RD7 are outputs

DATA &= 0xFF00; // prepare RD0 - RD7

DATA |= cmd; // command byte to lcd

RW = 0; // ensure RW is 0

RS = 0;

E = 1; // toggle E line

Nop();

Nop();

Nop();

Nop();

E = 0;

// Delay(Delay\_5mS\_Cnt); // 5ms delay

}

void lcd\_data( char data ) // subroutine for lcd data

{

while(checkBusy());

// TRISD &= 0xFF00; // ensure RD0 - RD7 are outputs

RW = 0; // ensure RW is 0

RS = 1; // assert register select to 1

DATA &= 0xFF00; // prepare RD0 - RD7

DATA |= data; // data byte to lcd

E = 1;

Nop();

Nop();

Nop();

E = 0; // toggle E signal

RS = 0; // negate register select to 0

// Delay\_Us( Delay200uS\_count ); // 200uS delay

// Delay\_Us( Delay200uS\_count ); // 200uS delay

}

void puts\_lcd( unsigned char \*data, unsigned char count )

{

while ( count )

{

lcd\_data( \*data++ );

count --;

}

}

char checkBusy()

{

char Result = 0;

//Config PORTE to inputs

// Store the current tri-state first though

int TrisETemp = TRISE;

TRISE = TRISE | 0x00FF;

// Set RW and RS to request a read

RW = 1;

RS = 0;

//According to the data sheet, we need to wait for 60 ns

//At instruction frequency of 16 MHz, each instrustion is 62.25 ns

Nop();

// Set E to high to start the read

E = 1;

//According to the data sheet, we need to wait for 190 ns

//At instruction frequency of 16 MHz, each instrustion is 62.25 ns

// 190 / 62.25 = 3.05. Therefore we should wait for 4 instructions to be safe

Nop();

Nop();

Nop();

Nop();

// Read the PORTE bus

// According to datasheet, the busy bit is located on the last bit of the data

Result = PORTE & 0x80;

// Pull E low

E = 0;

TRISE = TrisETemp;

return Result;

}