   Wyatt Tack

Bernardo Flores  
   EE 329-01 F'24

Group F

   2024-Oct-7

**EE 329 A3**

This code is designed to validate the use of a simple LCD screen, in which the communication timing was critical. The code in the LCD module works through sending a nibble (half of a byte), with an extra RS bit, of data out on the communication lines, while the enable pin is pulsed to signal the LCD controller that data is being sent through. The timing is critical, as the LCD must be given enough time to read and implement the data before more is sent through. This calls for another module, the delay module, which uses one of the STM32L4 internal clocks to set microsecond timers as delays in the program. This exercise helped with timing-critical bus analyzation to ensure a working device.

**Link to YouTube Presentation:**

<https://youtu.be/KA2iyTTzBKA>

**Obtained Data:**

Figure A3(a): 3x4 Keypad and 16x2 LCD Wiring

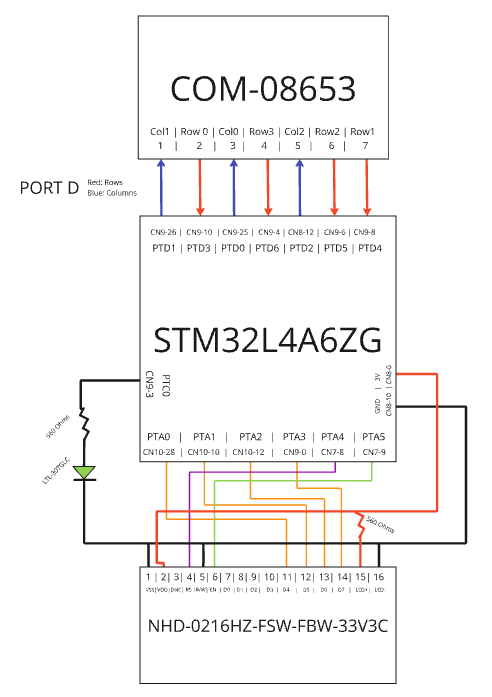
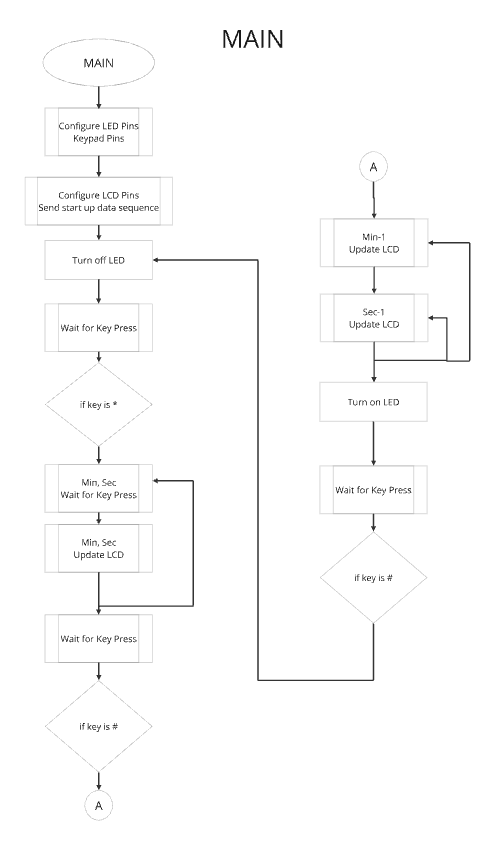
****

Table A3(a): Calibrated Delay Loop Timing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entered In Time |  | **Actual Time** | **Loop Delay Length** | **Delay Loop Iterations** |
| **00:30** |  | 00:30.30 | 930 uS | 1000 |
| **01:30** |  | 1:30.21 | 930 uS | 1000 |
| **03:00** |  | 2:59.18 | 930 uS | 1000 |

**Psuedocode Flow Chart:**



**Formatted Source Code main.h:**

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : main.h

\* project : EE329 Lab A3

\* author : Wyatt Tack (wwt) - wtack@calpoly.edu

\* date : 10/5/2024

\* firmware : ST-Link V1

\* @attention : Copyright (c) 2024 STMicroelectronics. All rights

\* reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* main header for defines for C and stm32 headers/hal

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

#ifndef \_\_MAIN\_H

#define \_\_MAIN\_H

#ifdef \_\_cplusplus

extern "C" {

#endif

#define STAR\_KEY (0x0A)

#define POUND\_KEY (0x0C)

/\* Includes ------------------------------------------------------------------\*/

#include "stm32l4xx\_hal.h"

/\* Exported functions prototypes ---------------------------------------------\*/

void SystemClock\_Config(void);

void Error\_Handler(void);

void Led\_Config(void);

#ifdef \_\_cplusplus

}

#endif

#endif

**Formatted Source Code main.c:**

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : main.c

\* project : EE329 Lab A3

\* author : Wyatt Tack (wwt) - wtack@calpoly.edu

\* date : 10/5/2024

\* firmware : ST-Link V1

\* @attention : Copyright (c) 2024 STMicroelectronics. All rights reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Timer device, uses Keypad to enter numbers for timer, and on pressing #

\* will count down until timer reached, signaling LED until # is pressed again.

\* pressing \* at any time will go to timer enter mode.

\* Keypad defined at GPIOD Col PTD0-2, Row PTD3-6

\* LED defined at GPIOC PTC0

\* LCD defined at GPIOA PTA0-3 (4 bit data), PTA4-5 (RS and EN)

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

#include "main.h"

#include "keypad.h"

#include "lcd.h"

#include "delay.h"

int main(void)

{

//Initialize clock and delay configs

HAL\_Init();

SystemClock\_Config();

SysTick\_Init();

//Initialize keypad and LCD

Keypad\_Config();

LCD\_init();

Led\_Config();

//display:

//"EE 329 A3 TIMER "

//"\*=SET #=GO 00:00"

uint8\_t origin[2] = {0,0};

uint8\_t line2[2] = {0,1};

uint8\_t openingLine1[16] = "EE 329 A3 TIMER ";

uint8\_t openingLine2[16] = "\*=SET #=GO 00:00";

//Initialize variables and cursor locations for each digit

uint8\_t clockMin10[2] = {11,1};

uint8\_t clockMin1[2] = {12,1};

uint8\_t clockSec10[2] = {14,1};

uint8\_t clockSec1[2] = {15,1};

uint8\_t currentKeyValue;

uint8\_t min10, min1;

uint8\_t sec10, sec1;

while (1)

{

// reset LED and poll keypad until \* pressed

currentKeyValue = 0;

LCD\_set\_cursor(origin);

LCD\_write\_string(openingLine1);

LCD\_set\_cursor(line2);

LCD\_write\_string(openingLine2);

GPIOC->BRR = (GPIO\_PIN\_0);

while(Key\_Poll() != STAR\_KEY);

// set time for each digit on clock

min10= 0xF; min1= 0xF; //reset values

sec10= 0xF; sec1= 0xF;

//Set 10\*min

LCD\_set\_cursor(clockMin10);

while(min10 < 0 || min10 > 5) {

if (currentKeyValue == STAR\_KEY) break;

currentKeyValue = Key\_Poll();

min10 = currentKeyValue;

}

LCD\_write\_char(min10+ 0x30);

//Set 1\*min

LCD\_set\_cursor(clockMin1);

while(min1 < 0 || min1 > 9){

if (currentKeyValue == STAR\_KEY) break;

currentKeyValue = Key\_Poll();

min1 = currentKeyValue;

}

LCD\_write\_char(min1+ 0x30);

//Set 10\*sec

LCD\_set\_cursor(clockSec10);

while(sec10 < 0 || sec10 > 5){

if (currentKeyValue == STAR\_KEY) break;

currentKeyValue = Key\_Poll();

sec10 = currentKeyValue;

}

LCD\_write\_char(sec10+ 0x30);

//Set 1\*sec

LCD\_set\_cursor(clockSec1);

while(sec1 < 0 || sec1 > 9){

if (currentKeyValue == STAR\_KEY) break;

currentKeyValue = Key\_Poll();

sec1 = currentKeyValue;

}

LCD\_write\_char(sec1+ 0x30);

//wait for # start key

while(currentKeyValue != POUND\_KEY){

if (currentKeyValue == STAR\_KEY) break;

currentKeyValue = Key\_Poll();

}

//clock loop

//min 10s

for (min10++; min10>0; min10--){

LCD\_set\_cursor(clockMin10);

LCD\_write\_char(min10-1 + 0x30);

//min 1s

for (min1++; min1>0; min1--){

LCD\_set\_cursor(clockMin1);

LCD\_write\_char(min1-1 + 0x30);

//sec 10s

for (sec10++; sec10>0; sec10--){

LCD\_set\_cursor(clockSec10);

LCD\_write\_char(sec10-1 + 0x30);

//sec 1s

for (sec1++; sec1>0; sec1--){

LCD\_set\_cursor(clockSec1);

LCD\_write\_char(sec1-1 + 0x30);

//~1s loop that polls keypad if \* pressed

for (uint32\_t timeDelay = 0; timeDelay < 1000; timeDelay++){

delay\_us(930); //adjust delay based on calibration

if(currentKeyValue == STAR\_KEY) break;

currentKeyValue = Keypad\_WhichKeyIsPressed();

}

}//roll overs from 10s place

sec1 = 9;

if(currentKeyValue == STAR\_KEY) break;

}

sec10 = 5;

if(currentKeyValue == STAR\_KEY) break;

}

min1 = 9;

if(currentKeyValue == STAR\_KEY) break;

}

//timer done turn on LED

GPIOC->BSRR |= (GPIO\_PIN\_0);

//wait till # or \* pressed to reset

while (!(currentKeyValue == STAR\_KEY || currentKeyValue == POUND\_KEY))currentKeyValue = Key\_Poll();

}

}

**(CTD’)**

**Formatted Source Code main.c (CTD’):**

//---------------------------------------------------------------------------

void Led\_Config(void)

{ // configure GPIO pins PC0-3 for:

// output mode, push-pull, no pull up or pull down, high speed

RCC->AHB2ENR |= (RCC\_AHB2ENR\_GPIOCEN);

GPIOC->MODER &= ~(GPIO\_MODER\_MODE0);

GPIOC->MODER |= (GPIO\_MODER\_MODE0\_0);

GPIOC->OTYPER &= ~(GPIO\_OTYPER\_OT0);

GPIOC->PUPDR &= ~(GPIO\_PUPDR\_PUPD0);

GPIOC->OSPEEDR |= (3 << GPIO\_OSPEEDR\_OSPEED0\_Pos);

GPIOC->BRR |= (GPIO\_PIN\_0);

}

// System

void SystemClock\_Config(void)

{

RCC\_OscInitTypeDef RCC\_OscInitStruct = {0};

RCC\_ClkInitTypeDef RCC\_ClkInitStruct = {0};

/\*\* Configure the main internal regulator output voltage

\*/

if (HAL\_PWREx\_ControlVoltageScaling(PWR\_REGULATOR\_VOLTAGE\_SCALE1) != HAL\_OK)

{

Error\_Handler();

}

/\*\* Initializes the RCC Oscillators according to the specified parameters

\* in the RCC\_OscInitTypeDef structure.

\*/

RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_MSI;

RCC\_OscInitStruct.MSIState = RCC\_MSI\_ON;

RCC\_OscInitStruct.MSICalibrationValue = 0;

RCC\_OscInitStruct.MSIClockRange = RCC\_MSIRANGE\_6;

RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_NONE;

if (HAL\_RCC\_OscConfig(&RCC\_OscInitStruct) != HAL\_OK)

{

Error\_Handler();

}

/\*\* Initializes the CPU, AHB and APB buses clocks

\*/

RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_HCLK|RCC\_CLOCKTYPE\_SYSCLK

|RCC\_CLOCKTYPE\_PCLK1|RCC\_CLOCKTYPE\_PCLK2;

RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_MSI;

RCC\_ClkInitStruct.AHBCLKDivider = RCC\_SYSCLK\_DIV1;

RCC\_ClkInitStruct.APB1CLKDivider = RCC\_HCLK\_DIV1;

RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV1;

if (HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_0) != HAL\_OK)

{

Error\_Handler();

}

}

void Error\_Handler(void)

{

\_\_disable\_irq();

while (1)

{

}

}

#ifdef USE\_FULL\_ASSERT

void assert\_failed(uint8\_t \*file, uint32\_t line)

{

}

#endif

**Formatted Source Code keypad.h:**

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : keypad.h

\* project : EE329 Lab A3

\* author : Wyatt Tack (wwt) - wtack@calpoly.edu

\* date : 9/27/2024

\* firmware : ST-Link V1

\* @attention : Copyright (c) 2024 STMicroelectronics. All rights reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* main header for defines for keypad.h

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

#ifndef INC\_KEYPAD\_H\_

#define INC\_KEYPAD\_H\_

#include "stm32l4xx\_hal.h"

//-------------------------------- KEYPAD Defines ------------------------------

#define COL\_PORT GPIOD

#define COL\_PINS (GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2)

#define ROW\_PORT GPIOD

#define ROW\_PINS (GPIO\_PIN\_3 | GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_6)

#define SETTLE 1900 //defined as time for 20ms loop instruction (\*5.5us)

#define BIT0\_COL GPIO\_PIN\_0 //defined as first bit for columns

#define BIT0\_ROW GPIO\_PIN\_3 //defined as first bit for rows

#define NUM\_COLS 3

#define NUM\_ROWS 4

#define NO\_KEYPRESS 0x0

#define KEY\_ZERO 11

#define CODE\_ZERO 0xF

//--------------------------------- function prototypes ------------------------

void Keypad\_Config(void);

int Keypad\_IsAnyKeyPressed(void);

int Keypad\_WhichKeyIsPressed(void);

uint8\_t Key\_Poll(void);

#endif /\* INC\_KEYPAD\_H\_ \*/

**Formatted Source Code keypad.c:**

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : keypad.c

\* project : EE329 Lab A3

\* author : Wyatt Tack (wwt) - wtack@calpoly.edu

\* date : 10/7/2024

\* firmware : ST-Link V1

\* @attention : Copyright (c) 2024 STMicroelectronics. All rights reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Keypad pulling function source file provided on behalf of the EE329 lab

\* manual. Adapted from EE329 lab manual. Currently attached to GPIO PORT D

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

#include "keypad.h"

// ------------------------------------- modified from excerpt from keypad.c ---

void Keypad\_Config(void){//must be manually changed if separate GPIO port is used

//set for port D as current config

//Port clock initialize

RCC->AHB2ENR |= (RCC\_AHB2ENR\_GPIODEN);

//Column pin initialize - Push Pull, no PU/PD, high speed

COL\_PORT->MODER &= ~(GPIO\_MODER\_MODE0 | GPIO\_MODER\_MODE1

| GPIO\_MODER\_MODE2);

COL\_PORT->MODER |= (GPIO\_MODER\_MODE0\_0 | GPIO\_MODER\_MODE1\_0

| GPIO\_MODER\_MODE2\_0);

COL\_PORT->OTYPER &= ~(GPIO\_OTYPER\_OT0 | GPIO\_OTYPER\_OT1

| GPIO\_OTYPER\_OT2 | GPIO\_OTYPER\_OT3);

COL\_PORT->PUPDR &= ~(GPIO\_PUPDR\_PUPD0 | GPIO\_PUPDR\_PUPD1

| GPIO\_PUPDR\_PUPD2 | GPIO\_PUPDR\_PUPD3);

COL\_PORT->OSPEEDR |= ((GPIO\_OSPEEDR\_OSPEED0) |

(GPIO\_OSPEEDR\_OSPEED1) |

(GPIO\_OSPEEDR\_OSPEED2));

//Row pin initialize - Input, pull down

ROW\_PORT->MODER &= ~(GPIO\_MODER\_MODE3 | GPIO\_MODER\_MODE4

| GPIO\_MODER\_MODE5 | GPIO\_MODER\_MODE6);

ROW\_PORT->PUPDR &= ~(GPIO\_PUPDR\_PUPD3 | GPIO\_PUPDR\_PUPD4

| GPIO\_PUPDR\_PUPD5 | GPIO\_PUPDR\_PUPD6);

ROW\_PORT->PUPDR |= (GPIO\_PUPDR\_PUPD3\_1 | GPIO\_PUPDR\_PUPD4\_1

| GPIO\_PUPDR\_PUPD5\_1 | GPIO\_PUPDR\_PUPD6\_1);

}

// -----------------------------------------------------------------------------

int Keypad\_IsAnyKeyPressed(void) {

// drive all COLUMNS HI; see if any ROWS are HI

// return true if a key is pressed, false if not

COL\_PORT->BSRR = COL\_PINS; // set all columns HI

for ( uint16\_t idx=0; idx<SETTLE; idx++ ) // let it settle

;

if ((ROW\_PORT->IDR & ROW\_PINS) != 0 ) { // got a keypress!

for ( uint16\_t idx=0; idx < SETTLE; idx++ ){

if ((ROW\_PORT->IDR & ROW\_PINS) == 0 ) return( 0 );

}// if key held for 20ms then return 1 (debounce)

return( 1 );

}

else

return( 0 ); // nope.

}

// -----------------------------------------------------------------------------

int Keypad\_WhichKeyIsPressed(void) {

// detect and encode a pressed key at {row,col}

// assumes a previous call to Keypad\_IsAnyKeyPressed() returned TRUE

// verifies the Keypad\_IsAnyKeyPressed() result (no debounce here),

// determines which key is pressed and returns the encoded key ID

int8\_t iRow=0, iCol=0, iKey=0; // keypad row & col index, key ID result

int8\_t bGotKey = 0; // bool for keypress, 0 = no press

COL\_PORT->BSRR = COL\_PINS; // set all columns HI

for ( iRow = 0; iRow < NUM\_ROWS; iRow++ ) { // check all ROWS

if ( ROW\_PORT->IDR & (BIT0\_ROW << iRow) ) { // keypress in iRow!!

COL\_PORT->BRR = ( COL\_PINS ); // set all cols LO

for ( iCol = 0; iCol < NUM\_COLS; iCol++ ) { // 1 col at a time

COL\_PORT->BSRR = ( BIT0\_COL << (iCol) ); // set this col HI

if ( ROW\_PORT->IDR & (BIT0\_ROW << iRow) ) { // keypress in iCol!!

bGotKey = 1;

break; // exit for iCol loop

}

}

if ( bGotKey )

break;

}

}

// encode {iRow,iCol} into LED word : row 1-3 : numeric, ‘1’-’9’

// row 4 : ‘\*’=10, ‘0’=15, ‘#’=12

// no press: send NO\_KEYPRESS

if ( bGotKey ) {

iKey = ( iRow \* NUM\_COLS ) + iCol + 1; // handle numeric keys ...

if ( iKey == KEY\_ZERO ) // works for ‘\*’, ‘#’ too

iKey = CODE\_ZERO;

return( iKey ); // return encoded keypress

}

return( NO\_KEYPRESS ); // unable to verify keypress

}

// -----------------------------------------------------------------------------

uint8\_t Key\_Poll(void) {

// polls key and returns value once key is inputed and let go

uint8\_t currentKeyValue;

while(!Keypad\_IsAnyKeyPressed());

if (Keypad\_WhichKeyIsPressed() > 0 && Keypad\_WhichKeyIsPressed() < 16){

currentKeyValue = Keypad\_WhichKeyIsPressed();

if (currentKeyValue == 0xF ) currentKeyValue = 0; //zero position

}

else

currentKeyValue = (0xF);//error flag

while(Keypad\_IsAnyKeyPressed());

return currentKeyValue;

//wait till key is let go

}

**Formatted Source Code delay.h:**

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : delay.h

\* project : EE329 Lab A3

\* author : Wyatt Tack (wwt) - wtack@calpoly.edu

\* date : 10/1/2024

\* firmware : ST-Link V1

\* @attention : Copyright (c) 2024 STMicroelectronics. All rights reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* main header for defines for delay.h

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

**#ifndef** INC\_DELAY\_H\_

**#define** INC\_DELAY\_H\_

**#include** "stm32l4xx\_hal.h"

**void** **SysTick\_Init**(**void**);

**void** **delay\_us**(**const** uint32\_t time\_us);

**#endif** /\* INC\_DELAY\_H\_ \*/

**Formatted Source Code delay.c:**

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : delay.c

\* project : EE329 Lab A3

\* author : Wyatt Tack (wwt) - wtack@calpoly.edu

\* date : 10/1/2024

\* firmware : ST-Link V1

\* @attention : Copyright (c) 2024 STMicroelectronics. All rights reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Functions for using SysTick clock for software delays. Provided

\* on behalf of EE329 lab manual.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

**#include** "delay.h"

// --------------------------------------------------- delay.c w/o #includes ---

// configure SysTick timer for use with delay\_us().

// warning: breaks HAL\_delay() by disabling interrupts for shorter delay timing.

**void** **SysTick\_Init**(**void**) {

SysTick->CTRL |= (SysTick\_CTRL\_ENABLE\_Msk | // enable SysTick Timer

SysTick\_CTRL\_CLKSOURCE\_Msk); // select CPU clock

SysTick->CTRL &= ~(SysTick\_CTRL\_TICKINT\_Msk); // disable interrupt

}

// delay in microseconds using SysTick timer to count CPU clock cycles

// do not call with 0 : error, maximum delay.

// careful calling with small nums : results in longer delays than specified:

// e.g. @4MHz, delay\_us(1) = 10=15 us delay.

**void** **delay\_us**(**const** uint32\_t time\_us) {

// set the counts for the specified delay

SysTick->LOAD = (uint32\_t)((time\_us \* (SystemCoreClock / 1000000)) - 1);

SysTick->VAL = 0; // clear timer count

SysTick->CTRL &= ~(SysTick\_CTRL\_COUNTFLAG\_Msk); // clear count flag

**while** (!(SysTick->CTRL & SysTick\_CTRL\_COUNTFLAG\_Msk)); // wait for flag

}

**Formatted Source Code lcd.h:**

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : lcd.h

\* project : EE329 Lab A3

\* author : Wyatt Tack (wwt) - wtack@calpoly.edu

\* date : 10/1/2024

\* firmware : ST-Link V1

\* @attention : Copyright (c) 2024 STMicroelectronics. All rights reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* main header for defines for lcd.h

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

**#ifndef** INC\_LCD\_H\_

**#define** INC\_LCD\_H\_

**#include** "delay.h"

**#include** "stm32l4xx\_hal.h"

**#define** LCD\_MODER (0x03FFF)

**#define** LCD\_MODER\_0 (0x01555)

**#define** LCD\_OTYPER (0x07F)

**#define** LCD\_PUPDR (0x03FFF)

**#define** LCD\_OSPEEDR (0x03FFF)

**#define** LCD\_PORT GPIOA

**#define** LCD\_EN GPIO\_PIN\_5 //Pin 5

**#define** LCD\_RS GPIO\_PIN\_4 //Pin 4

**#define** LCD\_DATA\_BITS (GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3)

//Pins 11-14

//pull down R/W (set as only outputs)

**void** **LCD\_init**( **void** );

**void** **LCD\_pulse\_ENA**( **void** );

**void** **LCD\_4b\_command**( uint8\_t command );

**void** **LCD\_command**( uint8\_t command );

**void** **LCD\_write\_char**( uint8\_t letter );

**void** **LCD\_set\_cursor**( uint8\_t position[2]);

**void** **LCD\_write\_string**( uint8\_t writeData[] );

**#endif** /\* INC\_LCD\_H\_ \*/

**Formatted Source Code lcd.c:**

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : lcd.c

\* project : EE329 Lab A3

\* author : Wyatt Tack (wwt) - wtack@calpoly.edu

\* date : 10/1/2024

\* firmware : ST-Link V1

\* @attention : Copyright (c) 2024 STMicroelectronics. All rights reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Functions for interfacing and communicating to LCD display through

\* nibble mode. Provided on behalf of EE329 lab manual. Configured to

\* be wired through GPIO PORT A Pin0-3 as D4-D7; Pin4 as RS; Pin5 as EN

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

**#include** "lcd.h"

// ------------------------------------------------------ excerpt from lcd.c ---

**void** **LCD\_init**( **void** ) {

//Port clock initialize

RCC->AHB2ENR |= (RCC\_AHB2ENR\_GPIOAEN);

//LCD pin initialize - Push Pull, no PU/PD, high speed

LCD\_PORT->MODER &= ~(LCD\_MODER);

LCD\_PORT->MODER |= (LCD\_MODER\_0);

LCD\_PORT->OTYPER &= ~(LCD\_OTYPER);

LCD\_PORT->PUPDR &= ~(LCD\_PUPDR);

LCD\_PORT->OSPEEDR |= (LCD\_OSPEEDR);

delay\_us( 80000 ); // power-up wait 80 ms

LCD\_PORT->ODR &= ~(LCD\_RS); // clear RS bit

**for** ( **int** idx = 0; idx < 3; idx++ ) { // wake up 1,2,3: DATA = 0011 XXXX

LCD\_4b\_command( 0x30 );// HI 4b of 8b cmd, low nibble = X

delay\_us( 5000 );

}

LCD\_4b\_command( 0x20 ); // fcn set #4: 4b cmd set 4b mode - next 0x28:2-line

delay\_us(3000);

LCD\_command( 0x28 ); // fcn set 4b mode and 2x28 line

delay\_us( 300 );

LCD\_command( 0x10 ); //Set cursor

delay\_us( 300 );

LCD\_command( 0x0F ); // display, cursor, blink on

delay\_us( 300 );

LCD\_command(0x06); //Entry mode

delay\_us( 300 );

LCD\_command(0x01); //clear display

delay\_us( 300 );

LCD\_command(0x80); // set cursor home

delay\_us( 300 );

}

**void** **LCD\_pulse\_ENA**( **void** ) {

// ENAble line sends command on falling edge

// set to restore default then clear to trigger

LCD\_PORT->ODR |= ( LCD\_EN ); // ENABLE = HI

delay\_us( 25 ); // TDDR > 320 ns

LCD\_PORT->ODR &= ~( LCD\_EN ); // ENABLE = LOW

delay\_us( 20 ); // low values flakey, see A3:p.1

}

**void** **LCD\_4b\_command**( uint8\_t command ) {

// LCD command using high nibble only - used for 'wake-up' 0x30 commands

LCD\_PORT->ODR &= ~( LCD\_DATA\_BITS ); // clear DATA bits

LCD\_PORT->ODR |= ( command >> 4 ); // DATA = command

delay\_us( 15 );

LCD\_pulse\_ENA( );

}

**void** **LCD\_command**( uint8\_t command ) {

// send command to LCD in 4-bit instruction mode

// HIGH nibble then LOW nibble, timing sensitive

LCD\_PORT->ODR &= ~( LCD\_DATA\_BITS ); // isolate cmd bits

LCD\_PORT->ODR |= ( (command>>4) & LCD\_DATA\_BITS ); // HIGH shifted low

delay\_us( 15 );

LCD\_pulse\_ENA( ); // latch HIGH NIBBLE

LCD\_PORT->ODR &= ~( LCD\_DATA\_BITS ); // isolate cmd bits

LCD\_PORT->ODR |= ( command & LCD\_DATA\_BITS ); // LOW nibble

delay\_us( 15 );

LCD\_pulse\_ENA( ); // latch LOW NIBBLE

}

**void** **LCD\_write\_char**( uint8\_t letter ) {

// calls LCD\_command() w/char data; assumes all ctrl bits set LO in LCD\_init()

LCD\_PORT->ODR |= (LCD\_RS); // RS = HI for data to address

delay\_us( 15 );

LCD\_command( letter ); // character to print

LCD\_PORT->ODR &= ~(LCD\_RS); // RS = LO

}

**void** **LCD\_set\_cursor**( uint8\_t position[2]) {

// calls LCD\_command to change cursor position

//position formatted as [row,col] (zero indexed)

//sets ddram address for cursor set

LCD\_PORT->ODR &= ~(LCD\_RS);

uint8\_t ddRamAdd = (40 \* position[1]);

ddRamAdd |= (0x80);

LCD\_command(ddRamAdd);

delay\_us( 500 );

**for** (**int** col = 0; col < position[0]; col++){

LCD\_command( 0x14 );

delay\_us( 300 );

}

//set address as RS\_Low, data as [0x80 | address]

//address defined as 0x00-0x0F, 0x40-0x4F

}

**void** **LCD\_write\_string**( uint8\_t writeData[]) {

// calls LCD\_write\_char in row long for loop

**for** (uint8\_t indexCol = 0; indexCol < 16; indexCol++ ){

LCD\_write\_char(writeData[indexCol]);

delay\_us(60);

}

}