Wyatt Tack
Keanu Lam
EE 329-01 F'24
Group P
2024-Sept-28

EE 329 A2

This code is designed to validate the use of a keypad, using the same LED array used in A2. The code functions through polling to determine if a key is pressed, and if so, goes into a subroutine to determine which key is pressed by scrolling through each column, then each row. This program was helpful for organizing file structure in C, making various source and header files, as well as writing multiple #define statements to make code more legible.

Link to YouTube Presentation:

https://youtu.be/oWzriSy2BYs

Obtained Data:

Figure A2(a): 3x4 Keypad and 4xLED Wiring

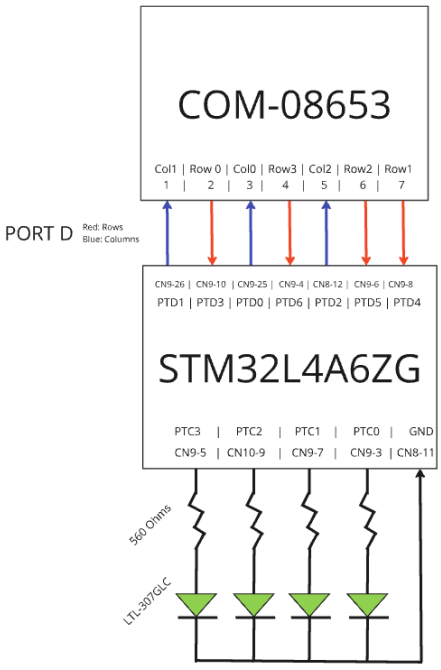


Table A2(a): Pull-Up/Pull-Down resistances found in pg.176 of STM32 Datasheet

Resistance	**Min**	**Nominal**	**Max**
Pull Up	25 kΩ	40 kΩ	55 kΩ
Pull Down	25 kΩ	40 kΩ	55 kΩ

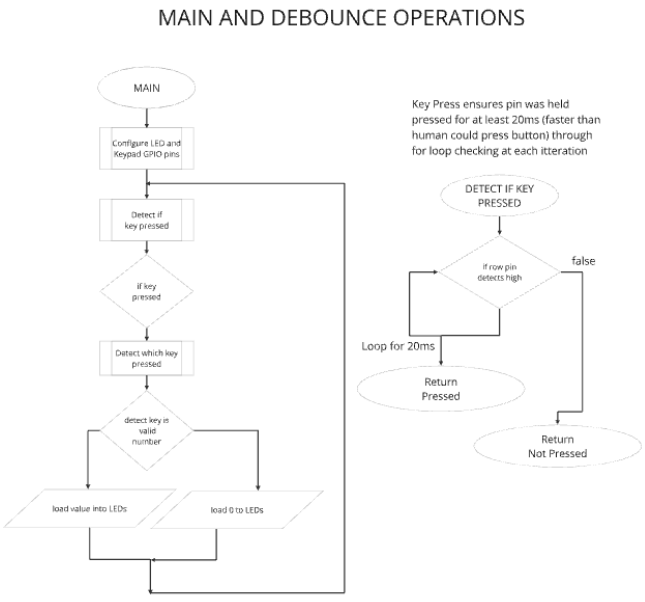
Psuedocode Flow Chart:

Main:

initializeLEDGPIO()
initializeKeypadGPIO()
loop
    ifKeyPressed
        detectWhichKeyPressed
            ifNotValid → display zero
            ifValid → display key pressed

ifKeyPressed:

columns = high
poll rows
    if row high
        read high for 10ms (debounce)
            if high for 10ms → return pressed
            else → return not pressed



Formatted Source Code main.h:

```
------------------------------------------------------------------------------
main.h
------------------------------------------------------------------------------

/**
  ******************************************************************************
  * @file               : main.h
  * project             : EE329 Lab A2
  * 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 C and stm32 headers/hal
  *
  ******************************************************************************
  */

#ifndef __MAIN_H
#define __MAIN_H

#ifdef __cplusplus
extern "C" {
#endif

/* Includes ------------------------------------------------------------------*/
#include "stm32l4xx_hal.h"

/* Exported functions prototypes ---------------------------------------------*/
void SystemClock_Config(void);
void Error_Handler(void);


#ifdef __cplusplus
}
#endif

#endif
```

Formatted Source Code main.c:

```
/**
  ******************************************************************************
  * @file           : main.c
  * project                                       : EE329 Lab A2
  * author                                        : Wyatt Tack (wwt) - wtack@calpoly.edu
  * date                                          : 9/27/2024
  * firmware                          : ST-Link V1
  * @attention                        : Copyright (c) 2024 STMicroelectronics. All rights reserved.
  ******************************************************************************
  *
  *                Takes in input from 3x4 (0-9 *#) keypad, then writes the hex value of the
  *                number to the LED array. LEDs display 0 if glitch, LEDs display 0xF for 0,
  *                0xA for *, and 0xC for #
  *
  *                Keypad defined at GPIOD Col PTC0-2, Row PTC3-6
  *                LEDs defined at GPIOC PTC0-3
  *
  ******************************************************************************
  */

#include "main.h"
#include "keypad.h"
void Led_Config(void);


int main(void)
{

                // Initialize system clock, keypad registers, led registers, and held value
  HAL_Init();
  SystemClock_Config();
  Keypad_Config();
  Led_Config();
  //uint8_t currentKeyValue = 0;
  //GPIOC->BSRR = GPIO_PIN_0;
  while (1)
  {
                    if (Keypad_IsAnyKeyPressed())
                    {

                                    if (Keypad_WhichKeyIsPressed() > 0 && Keypad_WhichKeyIsPressed() < 16){
                                                    //currentKeyValue = Keypad_WhichKeyIsPressed();
                                                    //GPIOC->ODR = (currentKeyValue);
                                                    GPIOC->ODR = Keypad_WhichKeyIsPressed();
                                    }
                                    else
                                                    GPIOC->ODR = (0x0);

                    }
  }

}
/********************************* ledConfig ********************************/
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 | GPIO_MODER_MODE1
                                                                                                                                    | GPIO_MODER_MODE2 | GPIO_MODER_MODE3);
                    GPIOC->MODER    |=  (GPIO_MODER_MODE0_0 | GPIO_MODER_MODE1_0                                                    | GPIO_MODER_MODE2_0 | GPIO_MODER_MODE3_0);
                    GPIOC->OTYPER   &= ~(GPIO_OTYPER_OT0 | GPIO_OTYPER_OT1                                                          | GPIO_OTYPER_OT2 | GPIO_OTYPER_OT3);
                    GPIOC->PUPDR    &= ~(GPIO_PUPDR_PUPD0 | GPIO_PUPDR_PUPD1                                                       | GPIO PUPDR PUPD2 | GPIO PUPDR PUPD3);
                    GPIOC->OSPEEDR |=  ((3 << GPIO_OSPEEDR_OSPEED0_Pos) |
                                                    (3 << GPIO_OSPEEDR_OSPEED1_Pos) |
                                                    (3 << GPIO_OSPEEDR_OSPEED2_Pos) |
                                                    (3 << GPIO_OSPEEDR_OSPEED3_Pos));
                    GPIOC->BRR = (GPIO_PIN_0 | GPIO_PIN_1                                                   | GPIO_PIN_2 | GPIO_PIN_3);
}


// 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:

```
--------------------------------------------------------------------------------
keypad.h
--------------------------------------------------------------------------------

/**
  ******************************************************************************
  * @file           : keypad.h
  * project                    : EE329 Lab A2
  * 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);


#endif /* INC_KEYPAD_H_ */
```

Formatted Source Code keypad.c:

```
/**
  ******************************************************************************
  * @file           : keypad.c
  * project                                  : EE329 Lab A2
  * author                                   : Wyatt Tack (wwt) - wtack@calpoly.edu
  * date                                     : 9/27/2024
  * 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.
  *
  ******************************************************************************
  */
#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
}
```