Embedded Systems Project

Prompt the user to input the number of BCD numbers(max 10). Each number has to be at most 3 digits. Display the numbers entered from the keyboard on the LCD. There should be provision to accept a number or cancel the last entered digit. DISPLAY THE BCD SUM.

Contents:

Team:

- Pratyay Amrit 140953430
- Arpit Garg 140953426
- Arvind Govindarajan 140953454

LOGIC

Global Variables:

 # int logcount – to keep track of where the program control currently is and what to prompt the user to do

 # char c[] – to hold the character array (string) entered by the user as input for it to be sent to the LCD display

 # int top – points to the topmost character in the c[] array

 # int row, col – to store the row and column number of the key pressed at the moment

 # int n – to store the number of numbers (first input by the user)

 # int intsum – stores the sum of all integers entered

Adding and removing to and from c[]:

```
- init c[top] = '\0'; top = 0;

# Adding to c[]
    { c[top++] = '1'; //entered character
       c[top] = '\0'; //appending null character }

# Removing from c[]
    { c[--top] = '\0'; //decrementing position of null }
```

Logic:

```
- get interrupt on rising edge of keypress

# logcount == 0  //get the number of numbers
    - send char[] = {“Number of nos.?”} to row 1
    - move cursor to row 2 and wait for interrupt
    if some number pressed
        - add to c[]
        - display on LCD
    else if “delete” key pressed
        - remove from c[]
        - display on LCD
    else if “enter” key pressed
        - increment logcount
        - send integer equivalent of c[] to n
        - reinitialize c[]

# logcount == 1  //get numbers one by one
    - send char[] = {“Enter numbers”} to row 1
    - move cursor to row 2 and wait for interrupt
    while i is less than n
        if some number pressed
            - add to c[]
            - display on LCD
        else if “delete” key pressed
            - remove from c[]
            - display on LCD
        else if “enter” key pressed
            - increment i
            - add atoi(c) to intsum
            - reinitialize c[]
    //end of while
    - increment logcount

# logcount == 2  //get numbers one by one
    - send char[] = {“BCD Sum is”} to row 1
    - move cursor to row 2
    - convert intsum to string and send to display
```

REQUIREMENTS

Components:

1. ALS-SDA-ARMCTXM3-01	x1
2. Power Supply (+5V)	x1
3. Cross cable for programming and serial communication	x1
4. 10 Core FRC cable	x3
5. USB to B type cable	x1

Hardware Setup:

CND [P0.23 to P0.28] for LCD;

CNC for keyboard [P0.15 to P0.18 cols; P0.19 to P0.22 rows];

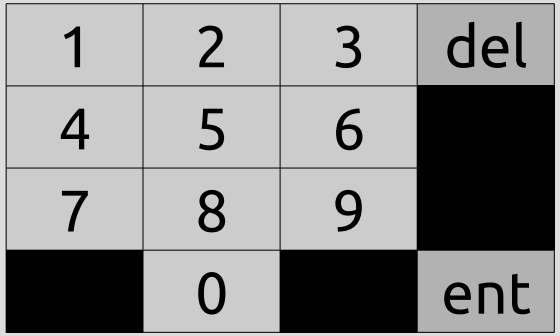
PIN Configuration:

P0.23 – P0.28 as GPIO output

P0.19 – P0.22 as GPIO output

P0.15 – P0.18 as GPIO Interrupt

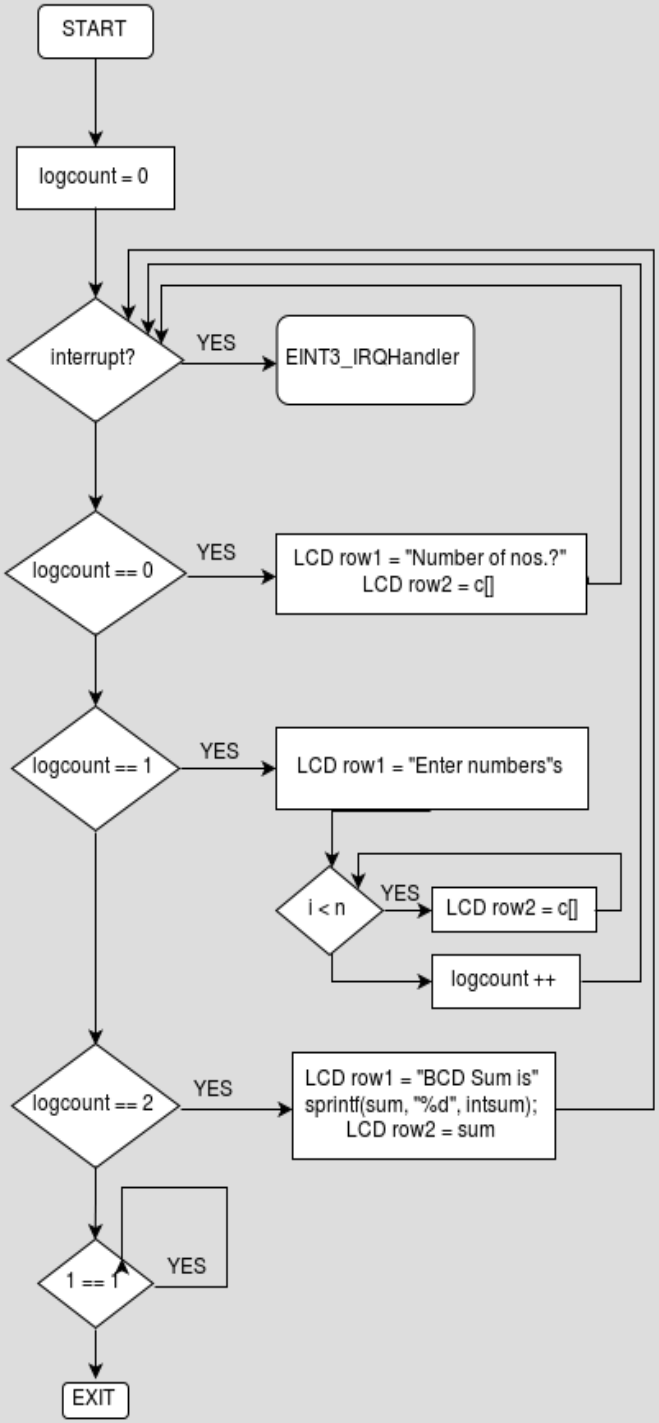
KEYBOARD LAYOUT

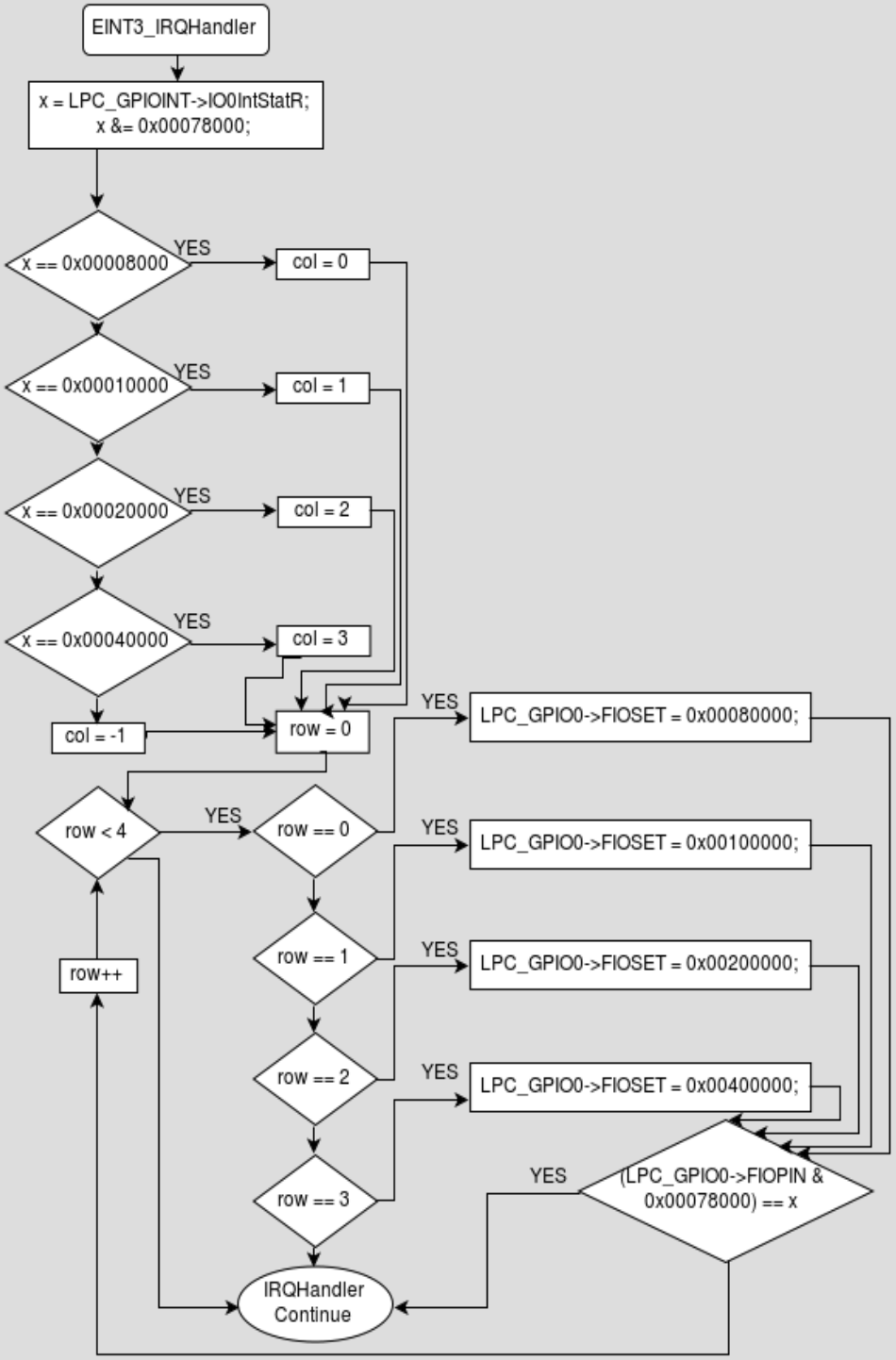


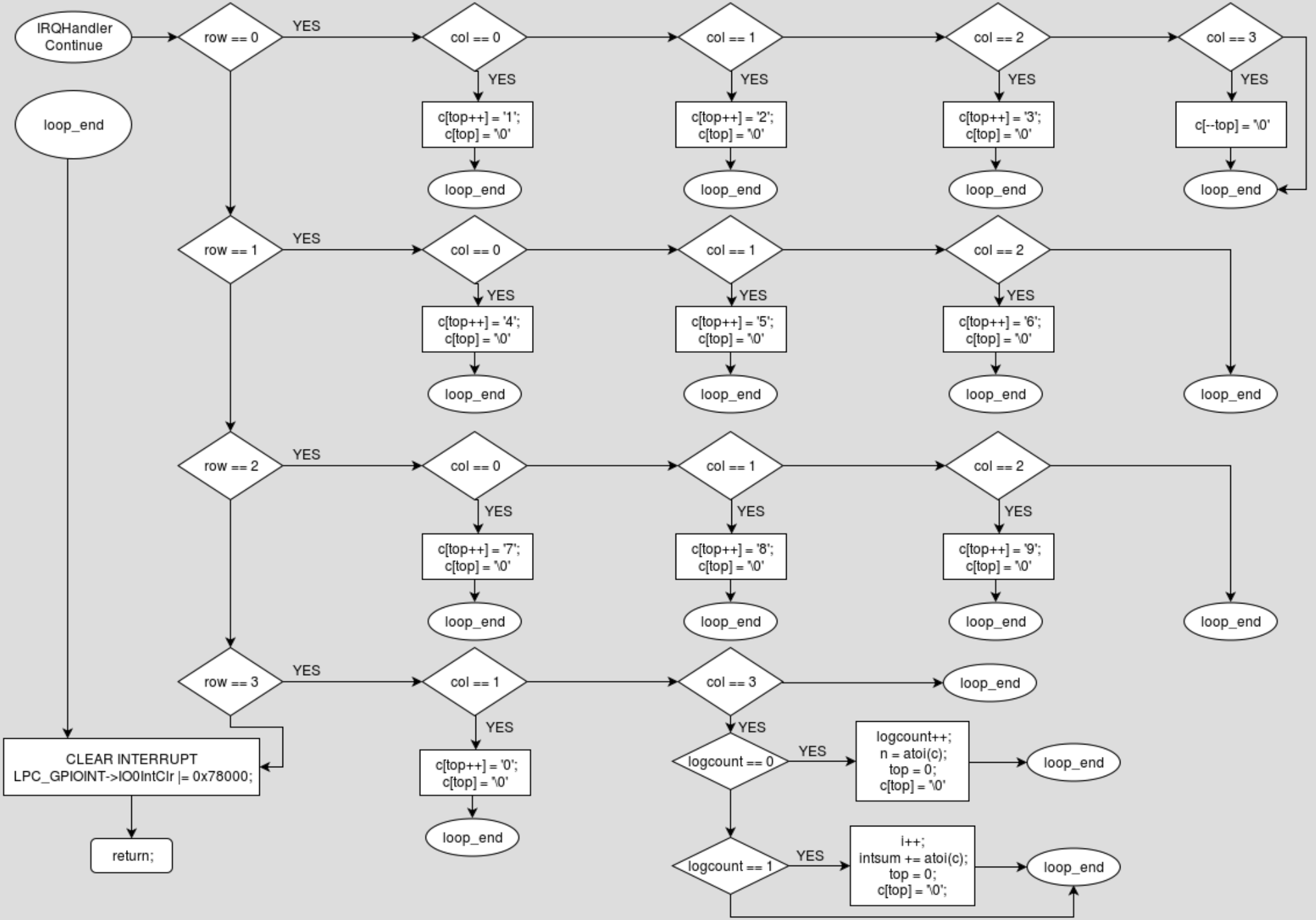
KEY-FUNCTION MAPPING

Row	Col	Action
0	0	Push “1”
0	1	Push “2”
0	2	Push “3”
0	3	Pop
1	0	Push “4”
1	1	Push “5”
1	2	Push “6”
1	3	-x-
2	0	Push “7”
2	1	Push “8”
2	2	Push “9”
2	3	-x-
3	0	-x-
3	1	Push “0”
3	2	-x-
3	3	enter

FLOW CHART


START
logcount = 0
interrupt?
YES
EINT3_IRQHandler
logcount == 0
YES
LCD row1 = "Number of nos.?"
LCD row2 = c[]
logcount == 1
YES
LCD row1 = "Enter numbers"s
i < n
YES
LCD row2 = c[]
logcount ++
logcount == 2
YES
LCD row1 = "BCD Sum is"
sprintf(sum, "%d", intsum);
LCD row2 = sum
1 == 1
YES
EXIT


EINT3_IRQHandler
x = LPC_GPIOINT->IO0IntStatR;
x &= 0x00078000;
x == 0x00008000
YES
col = 0
x == 0x00010000
YES
col = 1
x == 0x00020000
YES
col = 2
x == 0x00040000
YES
col = 3
col = -1
row = 0
row < 4
YES
row == 0
YES
LPC_GPIO0->FIOSET = 0x00080000;
row == 1
YES
LPC_GPIO0->FIOSET = 0x00100000;
row == 2
YES
LPC_GPIO0->FIOSET = 0x00200000;
row == 3
YES
LPC_GPIO0->FIOSET = 0x00400000;
row++
(LPC_GPIO0->FIOPIN &
0x00078000) == x
YES
IRQHandler
Continue


IRQHandler
Continue
loop_end
row == 0
YES
col == 0
col == 1
col == 2
col == 3
c[top++] = '1';
c[top] = '\0'
c[top++] = '2';
c[top] = '\0'
c[top++] = '3';
c[top] = '\0'
c[--top] = '\0'
loop_end
row == 1
c[top++] = '4';
c[top] = '\0'
c[top++] = '5';
c[top] = '\0'
c[top++] = '6';
c[top] = '\0'
row == 2
c[top++] = '7';
c[top] = '\0'
c[top++] = '8';
c[top] = '\0'
c[top++] = '9';
c[top] = '\0'
row == 3
c[top++] = '0';
c[top] = '\0'
logcount == 0
logcount++;
n = atoi(c);
top = 0;
c[top] = '\0'
logcount == 1
i++;
intsum += atoi(c);
top = 0;
c[top] = '\0';
CLEAR INTERRUPT
LPC_GPIOINT->IO0IntClr |= 0x78000;
return;