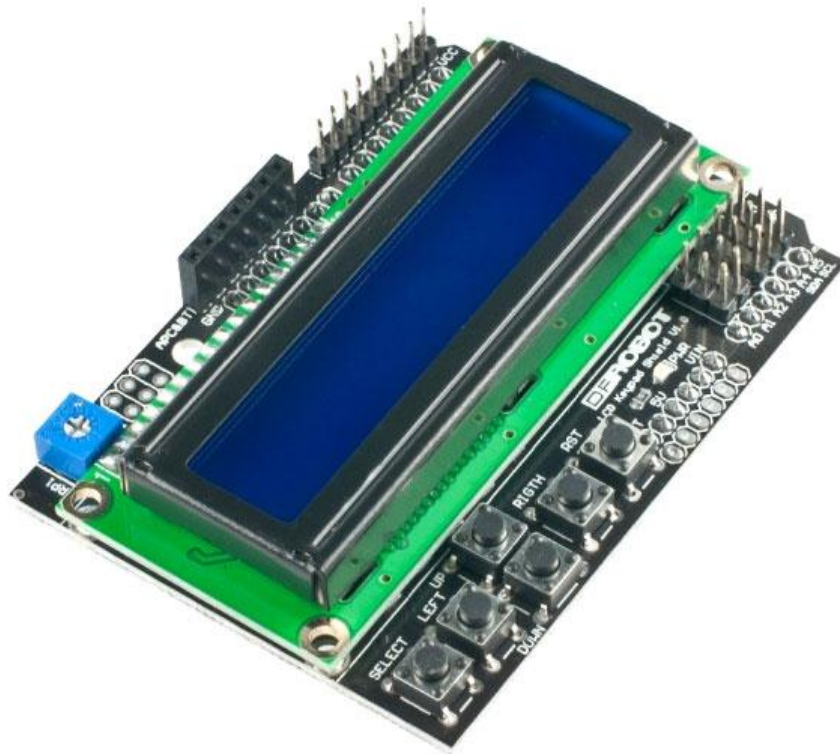


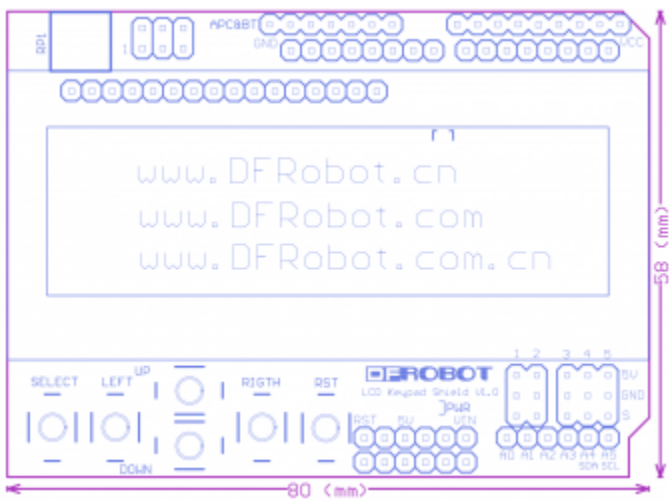
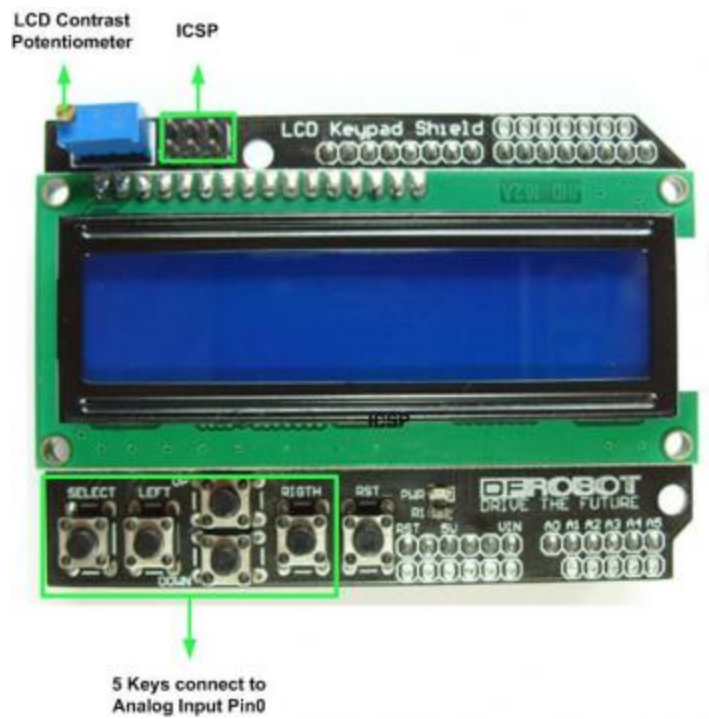
Arduino LCD Keypad Shield

Introduction



The LCD Keypad shield is developed for Arduino compatible boards, to provide a user-friendly interface that allows users to go through the menu, make selections etc. It consists of a 16x2 white character blue backlight LCD. The keypad consists of 5 keys — select, up, right, down and left. To save the digital IO pins, the keypad interface uses only one ADC channel. The key value is read through a 5 stage voltage divider.

Diagram



Pin Allocation

Pin	Function
Analog 0	Button (select, up, right, down and left)
Digital 4	DB4
Digital 5	DB5
Digital 6	DB6
Digital 7	DB7
Digital 8	RS (Data or Signal Display Selection)
Digital 9	Enable
Digital 10	Backlit Control

Sample Code

Example use of LCD4Bit_mod library

```
//
1  #include <LCD4Bit_mod.h>
2  //create object to control an LCD.
3  //number of lines in display=1
4  LCD4Bit_mod lcd = LCD4Bit_mod(2);
5  //Key message
6  charmsgs[5][15] = {"Right Key OK ",
7                    "Up Key OK   ",
8                    "Down Key OK ",
9                    "Left Key OK  ",
10                   "Select Key OK"};
11
12  int adc_key_val[5] = {30, 150, 360, 535, 760 };
13  int NUM_KEYS = 5;
14  int adc_key_in;
15  int key=-1;
16  int oldkey=-1;
17  void setup() {
18      pinMode(13, OUTPUT); //we'll use the debug LED to output a heartbeat
19
20      lcd.init();
21      //optionally, now set up our application-specific display settings, overriding whatever
22      lcd did in lcd.init()
23      //lcd.commandWrite(0x0F);//cursor on, display on, blink on. (nasty!)
24      lcd.clear();
25      lcd.printIn("KEYPAD testing... pressing");
26  }
27
28  void loop()
29  {
30      adc_key_in = analogRead(0); // read the value from the sensor
31      digitalWrite(13, HIGH);
32      key = get_key(adc_key_in); // convert into key press
33      if (key != oldkey) // if keypress is detected
34      {
35          delay(50); // wait for debounce time
36          adc_key_in = analogRead(0); // read the value from the sensor
37          key = get_key(adc_key_in); // convert into key press
38          if (key != oldkey)
39          {
40              oldkey = key;
41              if (key >=0){
42                  lcd.cursorTo(2, 0); //line=2, x=0
43                  lcd.printIn(msgs[key]);
44              }
45          }
46      }
47      digitalWrite(13, LOW);
48  }
49
50  // Convert ADC value to key number
51  int get_key(unsigned int input)
52  {
53      int k;
54      for (k = 0; k < NUM_KEYS; k++)
55      {
```

```

        if (input < adc_key_val[k])
        { return k; }
    }
    if (k >= NUM_KEYS)
        k = -1; // No valid key pressed
    return k;
}

```

Example use of LiquidCrystal library

```

1 //Sample using LiquidCrystal library
2 #include <LiquidCrystal.h>
3
4 /*****
5
6 This program will test the LCD panel and the buttons
7 Mark Bramwell, July 2010
8
9 *****/
10
11 // select the pins used on the LCD panel
12 LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
13
14 // define some values used by the panel and buttons
15
16 int lcd_key = 0;
17 int adc_key_in = 0;
18 #define btnRIGHT 0
19 #define btnUP 1
20 #define btnDOWN 2
21 #define btnLEFT 3
22 #define btnSELECT 4
23 #define btnNONE 5
24
25 // read the buttons
26
27 int read_LCD_buttons()
28 {
29     {
30         adc_key_in = analogRead(0); // read the value from the sensor
31         // my buttons when read are centered at these values: 0, 144, 329, 504, 741
32         // we add approx 50 to those values and check to see if we are close
33         if (adc_key_in > 1000) return btnNONE; // We make this the 1st option for speed reasons
34         // will be the most likely result
35         if (adc_key_in < 50) return btnRIGHT;
36         if (adc_key_in < 195) return btnUP;
37         if (adc_key_in < 380) return btnDOWN;
38         if (adc_key_in < 555) return btnLEFT;
39         if (adc_key_in < 790) return btnSELECT;
40         return btnNONE; // when all others fail, return this...
41     }
42 }
43
44 void setup()
45 {
46     {
47         lcd.begin(16, 2); // start the library
48         lcd.setCursor(0,0);
49         lcd.print("Push the buttons"); // print a simple message
50     }
51 }
52
53 void loop()
54 {
55

```

```

56  lcd.setCursor(9,1);           // move cursor to second line "1" and 9 spaces over
57  lcd.print(millis()/1000);     // display seconds elapsed since power-up
58
59
60  lcd.setCursor(0,1);           // move to the begining of the second line
61  lcd_key = read_LCD_buttons(); // read the buttons
62
63  switch (lcd_key)              // depending on which button was pushed, we perform an
64  {
65      case btnRIGHT:
66          {
67              lcd.print("RIGHT ");
68              break;
69          }
70      case btnLEFT:
71          {
72              lcd.print("LEFT  ");
73              break;
74          }
75      case btnUP:
76          {
77              lcd.print("UP   ");
78              break;
79          }
80      case btnDOWN:
81          {
82              lcd.print("DOWN ");
83              break;
84          }
85      case btnSELECT:
86          {
87              lcd.print("SELECT");
88              break;
89          }
90      case btnNONE:
91          {
92              lcd.print("NONE  ");
93              break;
94          }
95  }
96
97  }

```