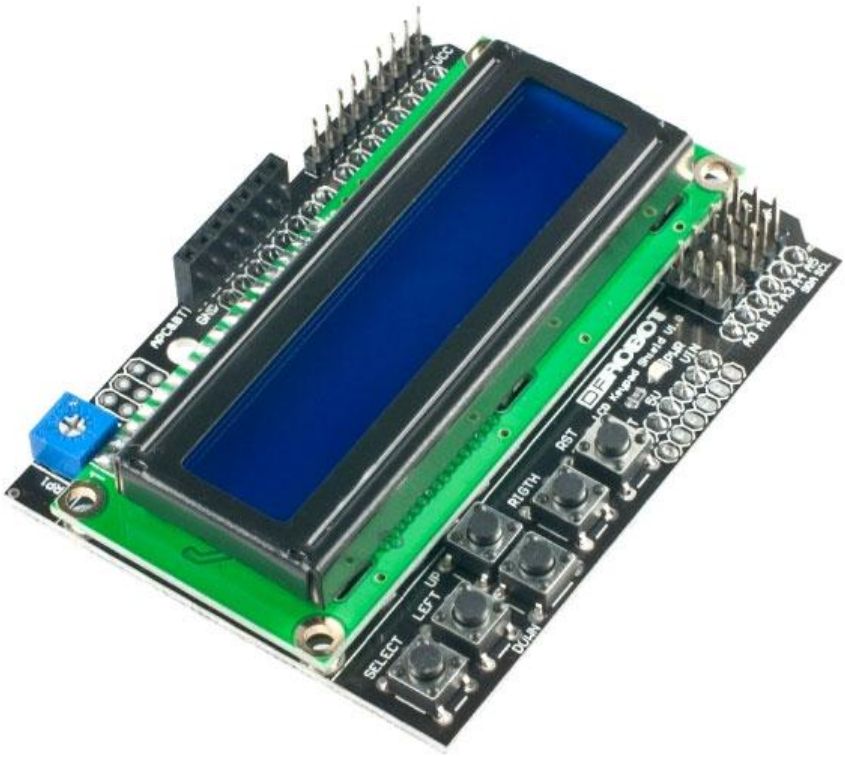
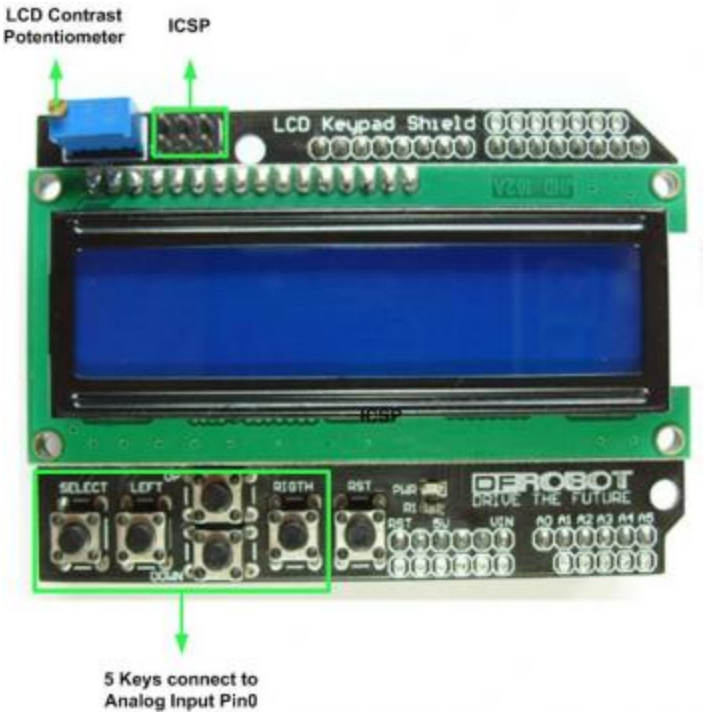
Arduino LCD KeyPad Shield

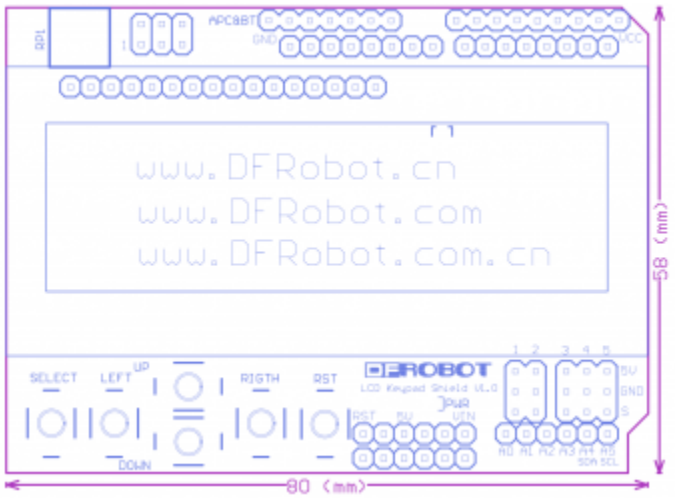
Introduction



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 1602 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

```
//
#include <LCD4Bit_mod.h>
//create object to control an LCD.
//number of lines in display=1
LCD4Bit_mod lcd = LCD4Bit_mod(2);
//Key message
char msgs[5][15] = {"Right Key OK ",
                    "Up Key OK    ",
                    "Down Key OK  ",
                    "Left Key OK  ",
                    "Select Key OK"};
int adc_key_val[5] ={30, 150, 360, 535, 760 };
int NUM_KEYS = 5;
int adc_key_in;
int key=-1;
int oldkey=-1;
void setup() {
  pinMode(13, OUTPUT);  //we'll use the debug LED to output a heartbeat

  lcd.init();
  //optionally, now set up our application-specific display settings, overriding whatev
lcd did in lcd.init()
  //lcd.commandWrite(0x0F);//cursor on, display on, blink on.  (nasty!)
   lcd.clear();
  lcd.printIn("KEYPAD testing... pressing");
}
void loop()
{
adc_key_in = analogRead(0);    // read the value from the sensor
digitalWrite(13, HIGH);
key = get_key(adc_key_in);    // convert into key press
    if (key != oldkey) // if keypress is detected
    {
    delay(50);      // wait for debounce time
    adc_key_in = analogRead(0);    // read the value from the sensor
    key = get_key(adc_key_in);    // convert into key press
    if (key != oldkey)
    {
      oldkey = key;
      if (key >=0){
      lcd.cursorTo(2, 0);  //line=2, x=0
    lcd.printIn(msgs[key]);
      }
    }
  }
  digitalWrite(13, LOW);
}
// Convert ADC value to key number
int get_key(unsigned int input)
{   int k;
    for (k = 0; k < NUM_KEYS; k++)
    {
```

```
        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

```
//Sample using LiquidCrystal library
#include <LiquidCrystal.h>

/*********************************************************

This program will test the LCD panel and the buttons
Mark Bramwell, July 2010

*********************************************************/

// select the pins used on the LCD panel
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

// define some values used by the panel and buttons
int lcd_key     = 0;
int adc_key_in  = 0;
#define btnRIGHT  0
#define btnUP     1
#define btnDOWN   2
#define btnLEFT   3
#define btnSELECT 4
#define btnNONE   5

// read the buttons
int read_LCD_buttons()
{
 adc_key_in = analogRead(0);      // read the value from the sensor
 // my buttons when read are centered at these valies: 0, 144, 329, 504, 741
 // we add approx 50 to those values and check to see if we are close
 if (adc_key_in > 1000) return btnNONE; // We make this the 1st option for speed reasons
will be the most likely result
 if (adc_key_in < 50)   return btnRIGHT;
 if (adc_key_in < 195)  return btnUP;
 if (adc_key_in < 380)  return btnDOWN;
 if (adc_key_in < 555)  return btnLEFT;
 if (adc_key_in < 790)  return btnSELECT;
 return btnNONE;  // when all others fail, return this...
}

void setup()
{
 lcd.begin(16, 2);              // start the library
 lcd.setCursor(0,0);
 lcd.print("Push the buttons"); // print a simple message
}

void loop()
{
```

```
  lcd.setCursor(9,1);            // move cursor to second line "1" and 9 spaces over
  lcd.print(millis()/1000);      // display seconds elapsed since power-up


  lcd.setCursor(0,1);            // move to the begining of the second line
  lcd_key = read_LCD_buttons();  // read the buttons

  switch (lcd_key)               // depending on which button was pushed, we perform an
  {
    case btnRIGHT:
      {
      lcd.print("RIGHT ");
      break;
      }
    case btnLEFT:
      {
      lcd.print("LEFT   ");
      break;
      }
    case btnUP:
      {
      lcd.print("UP    ");
      break;
      }
    case btnDOWN:
      {
      lcd.print("DOWN  ");
      break;
      }
    case btnSELECT:
      {
      lcd.print("SELECT");
      break;
      }
      case btnNONE:
      {
      lcd.print("NONE  ");
      break;
      }
 }

}
```