Arduino 1602 LCD KeyPad Shield

From Geeetech Wiki

Contents

Overview

Arduino Keypad LCD Shield V2.0 contains 16*2 LCD with contrast adjustment and backlight. LCD dispays white letters on blue background. It uses only an analog port to input the 5 keys' signal. Here is also a reset button. Still unused IO is prepared for expansion.

Not only it has the contrast adjustment key、backlight optional switch, but also has four directions buttons key、a select key and a reset button.It also has four sensor analog interfaces、ultrasonic sensor interface, bluetooth module interface, APC220 wireless digital module communication interface. They are all independent、easier、convenient for application.

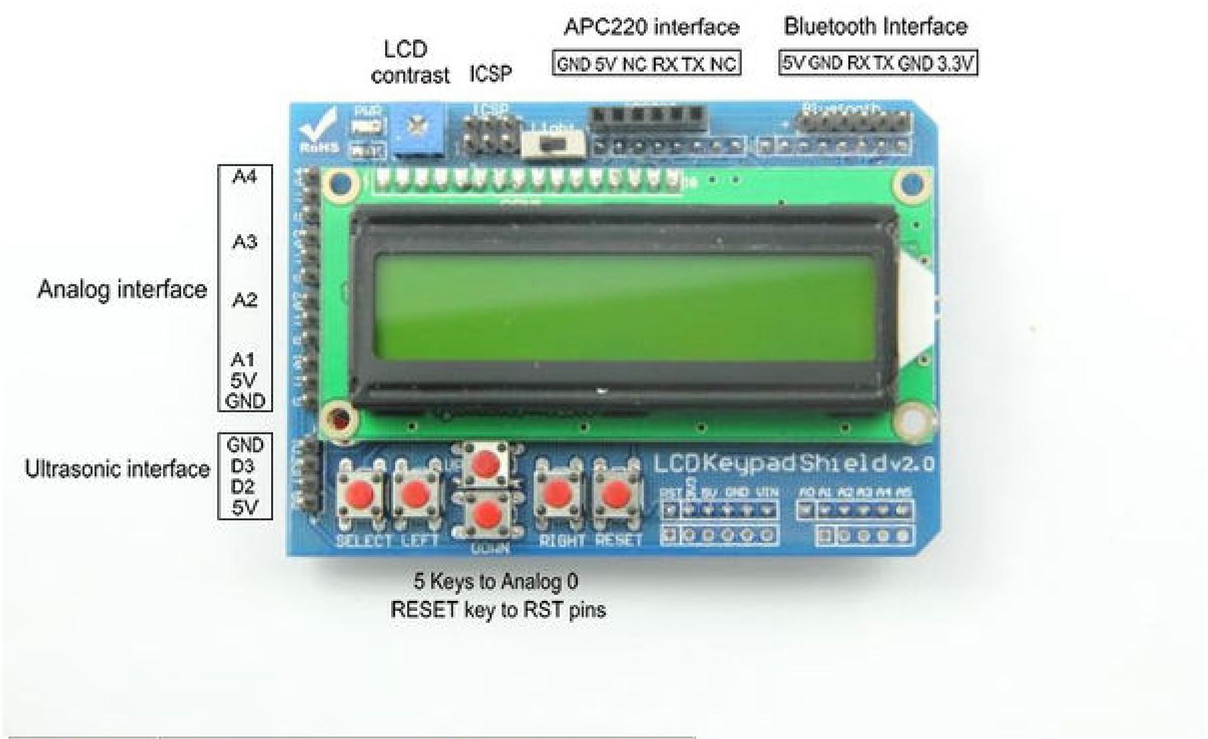
Arduino Keypad LCD Shield V2.0 is very easy and convenient for Sensors to indicate the data records.

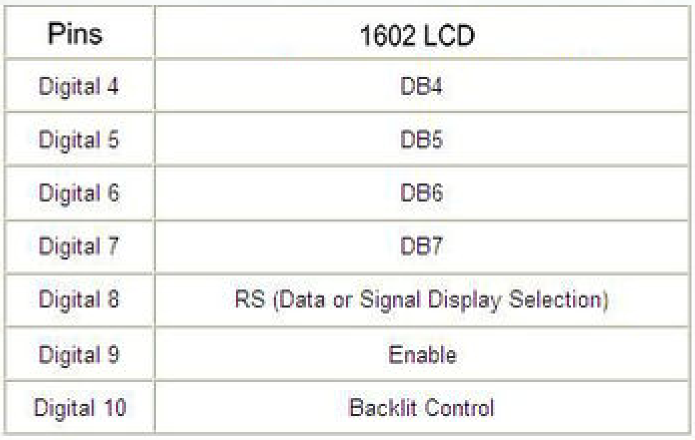
Special Feature

- four sensor analog interfaces
- ultrasonic sensor interface
- bluetooth module interface

- APC220 wireless digital module communication interface
- the contrast adjustment key、backlight optional switch、 four directions buttons key,a select key and a reset button.

Diagram





Pins	1602 LCD
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

Example code

```
//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 since it will be the mo
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 action
{
  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;
    }
}
}
```