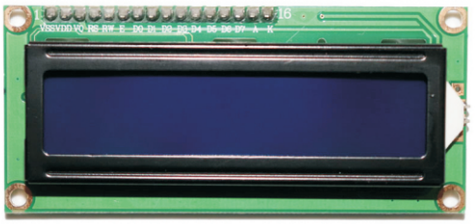
COKOINO

CKM0002

Thank you for choosing a COKOINO product. In order to take full advantage of the features and performance it provides, please read this owner's manual thoroughly, and If you have any questions or find any errors, please let us know. (hhd.cokoino@gmail.com)

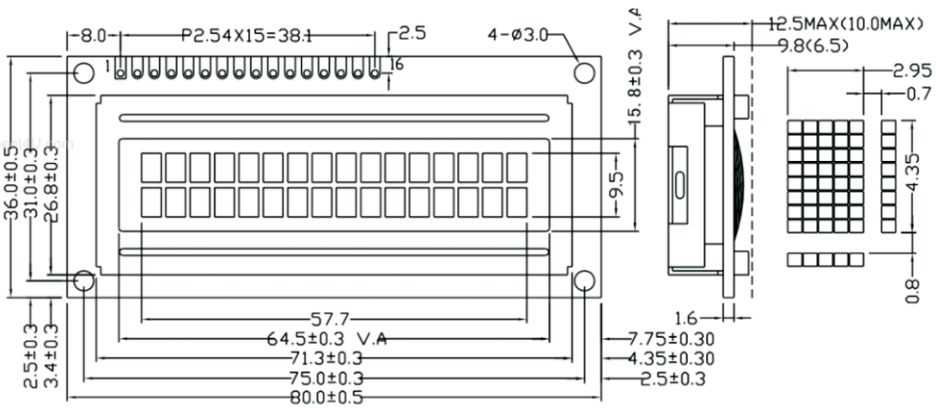


1602 lcd display with i2c interface

User' manual

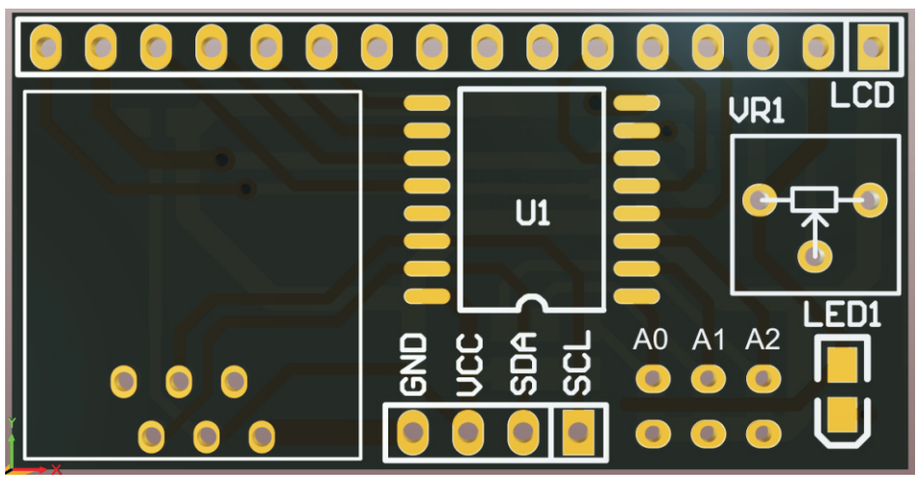
Size:2.4x1'

3. Size (mm)



5. Working Principle

Use the IIC PCF8574T chip to expand 8 pieces IO ports, and use the P4-P7 IO port to connect the 1602 LCD screen D4-D7 to drive the screen. For detailed driving methods, please refer to the provided data sheet.



(1) LCD screen resolution adjustment: The text clarity of the LCD screen can be adjusted by rotating the VR1 potentiometer.

Arduino code module IIC address selection, as shown in the following table:

A2	A1	A0	address
1	1	1	0x27
1	1	0	0x26
1	0	1	0x25
1	0	0	0x24
0	1	0	0x23
1	1	1	0x22
0	0	1	0x21
0	0	0	0x20

1. Overview

The COKOINO I2C 1602 LCD screen module is compatible with the Arduino platform, which is mainly used for displaying text and has high definition and stable performance. You can easily fix it to other devices with its 4 positioning holes and combined with the arduino controller board and other electronic modules to DIY various interesting products, such as arduino robot, intelligent clock display, thermometer and hygrometer display, calculator display, etc.

2. Specifications

(1) Working voltage: 4. 5--5. 5V
(2) Working current: 15--20mA
(3) Display style: 5 x 8 (1 Cursor) dots, 16 x 2 Characters
(4) Driver chip: PCF8574T
(5) Communication method: IIC (PCF8574T driver 1602LCD screen)
(6) Working temperature: -20 --- +70 ° C
(7) Size: 80x36mm
Remarks: Distribute PCF8574T data sheet, 1602 screen data sheet.

4. Interface Definition

VCC: DC5V
GND: power ground
SDA: Data line
SCL: clock line
How to use this product: IIC communicationdata.

6. How IIC works

(1)CHARACTERISTICS OF THE I 2 C-BUS

The I 2 C-bus is for 2-way, 2-line communication between different ICs or modules. The two lines are a serial data line (SDA) and a serial clock line (SCL). Both lines must be connected to a positive supply via a pull-up resistor when connected to the output stages of a device. Data transfer may be initiated only when the bus is not busy.

(2)Bit transfer

One data bit is transferred during each clock pulse. The data on the SDA line must remain stable during the HIGH period of the clock pulse as changes in the data line at this time will be interpreted as control signals (see Fig.5).

Start and stop conditions

Both data and clock lines remain HIGH when the bus is not busy. A HIGH-to-LOW transition of the data line, while the clock is HIGH is defined as the start condition (S).A LOW-to-HIGH transition of the data line while the clock is HIGH is defined as the stop condition (P) (see Fig.6).

System configuration

A device generating a message is a 'transmitter', a device receiving is the 'receiver'. The device that controls the message is the 'master' and the devices which are controlled by the master are the 'slaves' (see Fig.7).

Acknowledge

The number of data bytes transferred between the start and the stop conditions from transmitter to receiver is not limited. Each byte of eight bits is followed by one

acknowledge bit (see Fig.8). The acknowledge bit is a HIGH level put on the bus by the transmitter whereas the master generates an extra acknowledge related clock

pulse.

A slave receiver which is addressed must generate an acknowledge after the reception of each byte. Also a master must generate an acknowledge after the reception of each byte that has been clocked out of the slave transmitter. The device that acknowledges has to pull down the SDA line during the acknowledge clock pulse, so that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse, set-up and hold times must be taken into account.

A master receiver must signal an end of data to the transmitter by not generating an acknowledge on the last byte that has been clocked out of the slave. In this event

the transmitter must leave the data line HIGH to enable the master to generate a stop condition.

6. （2）

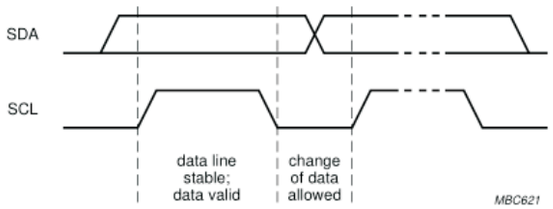


Fig.5 Bit transfer.

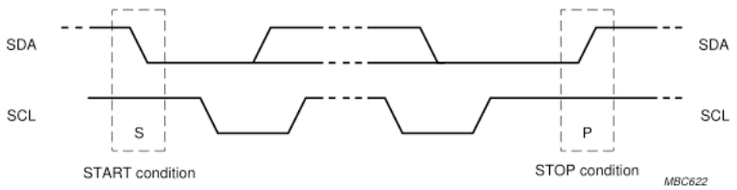


Fig.6 Definition of start and stop conditions.

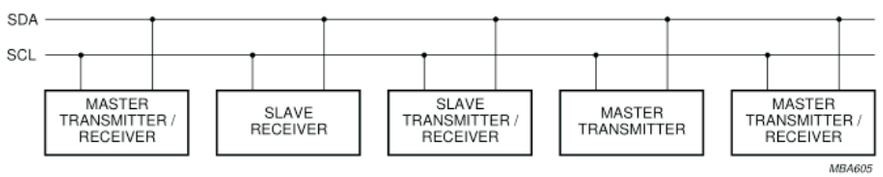


Fig.7 System configuration.

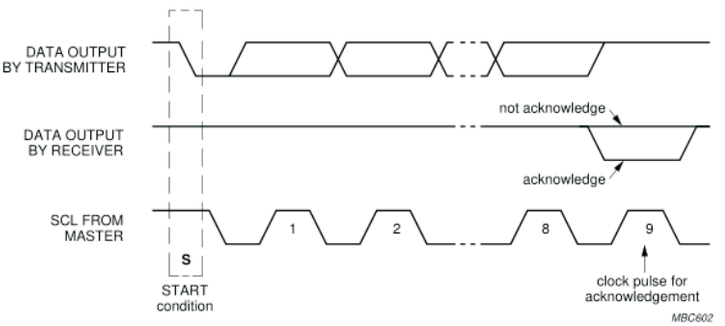
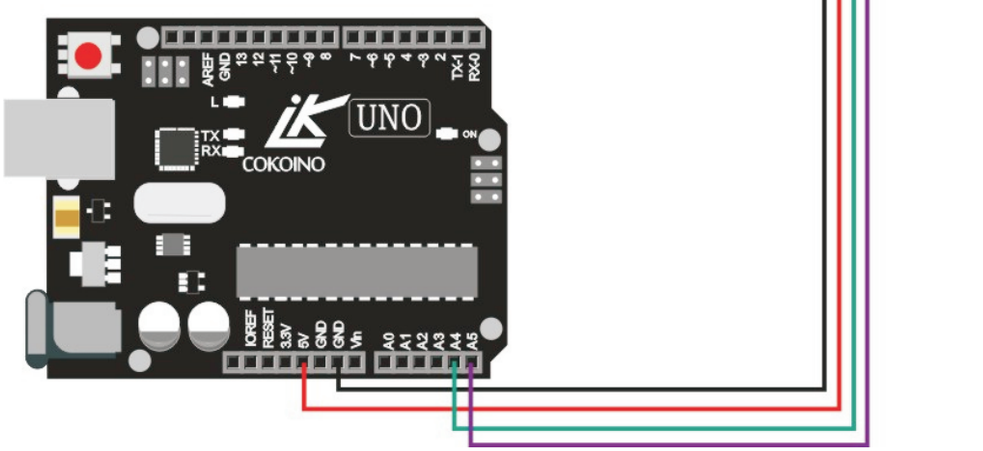


Fig.8 Acknowledgment on the I²C-bus.

7. Wiring Diagram

UNO R3	LCD
GND	G
5V	V
A4	SDA
A5	SCL



8.Arduino Test Code

```
#include <Wire.h> //IIC Communication Library File
#include <LiquidCrystal_I2C.h> //1602 screen IIC communication library file
LiquidCrystal_I2C lcd(0x27, 16, 2); // instantiate an LCD class with address 0x27
Void setup() //Set the parameter function, only run once after the program starts.
{
    Lcd.init(); //Initialize 1602LCD screen
    Lcd.backlight(); //Off 1602 LCD screen backlight
}
Void loop () / / main loop function, the program will continue to run in this function after this function is executed

{
    lcd.setCursor(2, 0); //The cursor is set to the 3rd position on the 1st line.
    Lcd.print("Hello world!"); //Show "Hello world!" from the 3rd position on the 1st line
    lcd.setCursor(0, 1); //The cursor is set to the first position on the 2nd line.
    Lcd.print("COKOINO Arduino!"); //Show "COKOINO Arduino!" from the 1st position on the 2nd line

}
```

Note: Before uploading the code, you must copy the LiquidCrystal_I2C.h library file in the lib folder to the libraries folder in the Arduino IDE directory.

9.Test Results

Connect the line according to the above picture, upload the code through arduino IDE, after power-on, you can see the following picture: