**Arduino**

**Introduction :**

Arduino is an open-source electronics platform based on hardware and software. Arduino are used to read the input in the form of lights, from computer, from finger touch and give the output according to the program written in the IDE. A job can br assigned to your board by a set of instructions. To do so you use the [Arduino programming language](https://www.arduino.cc/en/Reference/HomePage) (based on [Wiring](http://wiring.org.co/)), and [the Arduino Software (IDE)](https://www.arduino.cc/en/Main/Software), based on [Processing](https://processing.org/).

Arduino was designed with the aim for the student that do not have any electronics experience. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments.

**Features of Arduino :** Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step by step instructions of a kit, or sharing ideas online with other members of the Arduino community.

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

* Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50
* Cross-platform - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
* Simple, clear programming environment - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.
* Open source and extensible software - The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.
* Open source and extensible hardware - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the [breadboard version of the module](https://www.arduino.cc/en/Main/Standalone) in order to understand how it works and save money.

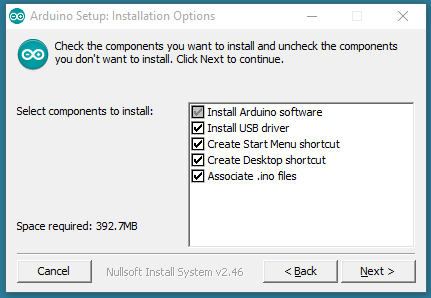
**How do I use Arduino :**

The Arduino Software (IDE) allows you to write programs and upload them to your board. In the [Arduino Software page](https://www.arduino.cc/en/Main/Software) you will find two options:  
  
1. If you have a reliable Internet connection, you should use the [online IDE](https://create.arduino.cc/editor) (Arduino Web Editor). It will allow you to save your sketches in the cloud, having them available from any device and backed up. You will always have the most up-to-date version of the IDE without the need to install updates or community generated libraries.  
  
2. If you would rather work offline, you should use the latest version of the [desktop IDE](https://www.arduino.cc/en/Main/Software#download).

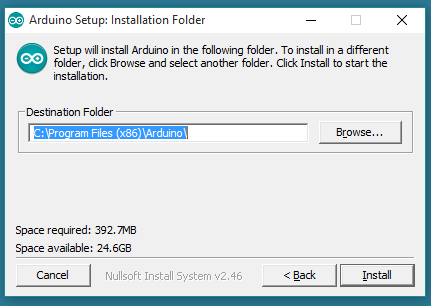
**Installation of Arduino IDE :**

Get the latest version from the [download page](https://www.arduino.cc/en/Main/Software). You can choose between the Installer (.exe) and the Zip packages. We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers. With the Zip package you need to install the drivers manually. The Zip file is also useful if you want to create a[portable installation](https://www.arduino.cc/en/Guide/PortableIDE).

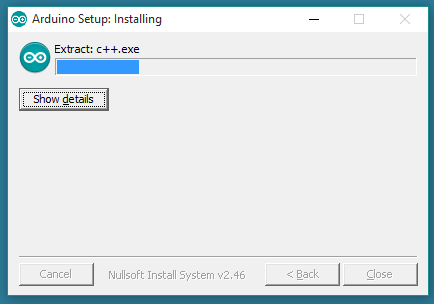
When the download finishes, proceed with the installation and please allow the driver installation process when you get a warning from the operating system.



Choose the components to install



Choose the installation directory (we suggest to keep the default one)



The process will extract and install all the required files to execute properly the Arduino Software (IDE)

**Interfacing of sensors with Arduino board :**

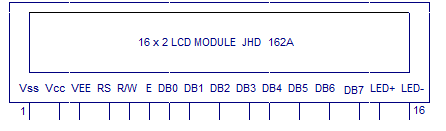
**Interfacing of LCD :**

#### Interfacing  16×2 LCD to Arduino uno

LCD modules form a very important part in many arduino based embedded system designs. So the knowledge on interfacing LCD module to arduino is very essential in designing embedded systems. This section of the article is about interfacing an Arduino to 16×2 LCD. JHD162A is the LCD module used here. JHD162A is a 16×2 LCD module based on the **HD44780 driver from Hitachi**. The JHD162A has 16 pins and can be operated in 4-bit mode (using only 4 data lines) or 8-bit mode (using all 8 data lines). Here we are using the LCD module in 4-bit mode. First, I will show you how to display a plain text messages on the LCD module using arduino and then  I have designed a useful project using LCD and arduino – a **digital thermometer**. Before going in to the details of the project, let’s have a look at the JHD162A LCD module.

#### 16×2 LCD Module Pin Out Diagram

 The JHD162A lcd module has 16 pins and can be operated in 4-bit mode or 8-bit mode. Here we are using the LCD module in 4-bit mode. Before going in to the details of the project, let’s have a look at the JHD162A LCD module.The schematic of a JHD162A LCD pin diagram is given below.

[](http://www.circuitstoday.com/wp-content/uploads/2014/06/JHD162A-LCD-module.png)

The name and functions of each pin of the 16×2 LCD module is given below.

**Pin1(Vss)**:Ground pin of the LCD module.

**Pin2(Vcc)**: Power to LCD module (+5V supply is given to this pin)

**Pin3(VEE)**:Contrast adjustment pin. This is done by connecting the ends of a 10K potentimeter to +5V and ground and then connecting the slider pin to the VEE pin. The voltage at the VEE pin defines the contrast. The normal setting is between 0.4 and 0.9V.

**Pin4(RS)**:Register select pin.The JHD162A has two registers namely **command register**and **data register**. Logic HIGH at RS pin selects data register and logic LOW at RS pin selects command register. If we make the RS pin HIGH and feed an input to the data lines (DB0 to DB7), this input will be treated as data to display on LCD screen. If we make the RS pin LOW and feed an input to the data lines, then this will be treated as a command ( a command to be written to LCD controller – like positioning cursor or clear screen or scroll).

**Pin5(R/W)**: Read/Write modes. This pin is used for selecting between read and write modes. Logic HIGH at this pin activates read mode and logic LOW at this pin activates write mode.

**Pin6(E)**: This pin is meant for enabling the LCD module. A HIGH to LOW signal at this pin will enable the module.

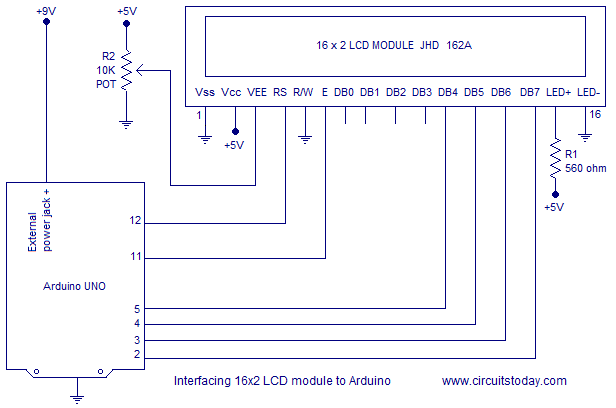
**Pin7(DB0) to Pin14(DB7)**:  These are data pins. The commands and data are fed to the LCD module though these pins.

**Pin15(LED+)**: Anode of the back light LED. When operated on 5V, a 560 ohm resistor should be connected in series to this pin. In arduino based projects the back light LED can be powered from the 3.3V source on the arduino board.

**Pin16(LED-)**: Cathode of the back light LED.

For knowing more about LCD module JHD162A and its pin functions, read this article: [**Interfacing 16×2 LCD and 8051 microcontroller**](http://www.circuitstoday.com/interfacing-16x2-lcd-with-8051). The circuit diagram of interfacing LCD to arduino for displaying a text message is shown below.

Circuit diagram – Arduino to 16×2 LCD Module

[](http://www.circuitstoday.com/wp-content/uploads/2014/06/interfacing-LCD-to-arduino.png)

RS pin of the LCD module is connected to digital pin 12 of the arduino. R/W pin of the LCD is grounded. Enable pin of the LCD module is connected to digital pin 11 of the arduino. In this project, the **LCD module and arduino are interfaced in the 4-bit mode**. This means only four of the digital input lines( DB4 to DB7)  of the LCD are used. This method is very simple, requires less connections and you can almost utilize the full potential of the LCD module. Digital lines DB4, DB5, DB6 and DB7 are interfaced to digital pins 5, 4, 3 and 2 of the Arduino. The 10K potentiometer is used for adjusting the contrast of the display. 560 ohm resistor R1 limits the current through the back light LED. The arduino can be powered through the external power jack provided on the board. +5V required in some other parts of the circuit can be tapped from the 5V source on the arduino board. The arduino can be also powered from the PC through the USB port. The full program for interfacing LCD to arduino is shown below.

##### Program – Arduino to LCD

#include<LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // sets the interfacing pins

void setup()

{

lcd.begin(16, 2); // initializes the 16x2 LCD

}

void loop()

{

lcd.setCursor(0,0); //sets the cursor at row 0 column 0

lcd.print("16x2 LCD MODULE"); // prints 16x2 LCD MODULE

lcd.setCursor(2,1); //sets the cursor at row 1 column 2

lcd.print("HELLO WORLD"); // prints HELLO WORLD

}

**Some Other sensors for Arduino are as follows :**

1.Temperaature humidity sensors

2.Infrared emission sensors

3.Photoresister

4.Ultrasonic Distance sensor

5.Knock Sensor

6.Sound Sensor etc.

**Refrences :**

<https://www.arduino.cc/en/Guide/HomePage>

<http://tutorial45.com/top-used-sensors-for-arduino/>

<http://www.circuitstoday.com/interfacing-lcd-to-arduino>