AIR AND SOUND POLLUTION MONITORING SYSTEM

A PROJECT REPORT

ON

"AIR AND SOUND POLLUTION MONITORING SYSTEM"

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Award of

Diploma in Electronics and Communication Engineering

Under the Guidance of

Mr.N.Ramulu

Submitted to

Department of Electronics and Communication Engineering

GOVERNMENT POLYTECHNIC, MASABTANK, HYDERABAD

(2019-2022)

GOVERNMENT POLYTECHNIC MASABTANK

(Recognised by AICTE &Affiliated to SBTET,Hyderabad)

Masabtank, Hyderabad 500028

CERTIFICATE

This is to certify that A.BINDU(19001-EC-203), K.LIKHITHA (19001-EC-219), P.SWETHA(19001-EC-233), M.RUTHIKA (19001-EC-244),C.PARAMESHWARI(19001-EC-246) have successfully completed project work entitled “AIR AND SOUND POLLUTION MONITORING SYSTEM” in partial fulfilment of the requirements for the award of degree of diploma in electronics and communication engineering during academic year 2019-2022

INTERNAL GUIDE

Mr.N.RAMULU
Lecturer in ECE Dept.

HEAD OF DEPT

Mr.M.LAXMAIAH,M.Tech

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

To make an effort over this period can be successful by support of many well wishers. We avail this opportunity to express our gratitude and appreciation to all of them.

We express our thanks to our project guide Mr.N.RAMULU for his valuable and timely suggestions during our project.

We would like to express our profound respect to DR.N.RAJESHWARI DEVI, Principal, government polytechnic college, Masabtank for her useful suggestions during the project work.

We extend our sincere thanks to staff of ECE department and all our friends for their good wishes, their helping hands and constructive criticism, which led the successful completion of my project report.

Finally, we thank all those who directly and indirectly helped us in this regard,we apologize for not listening everyone.

DECLARATION

We declare that the work report in the present this “AIR AND SOUND POLLUTION MONITORING SYSTEM” is a record of work done by us in the department of electronics and communication engineering, government polytechnic college, Masabtank.

No part of this is copied from books or journal or internet and wherever the portion has been taken the same have been duly referred in the text. The report is based on the project work done entirely by us and not copied from other sources.

PROJECT ASSOCIATES

A.BINDU	(19001-EC-203)
K.LIKHITHA	(19001-EC-219)
P.SWETHA	(19001-EC-233)
M.RUTHIKA	(19001-EC-244)
C.PARAMESHWARI	(19001-EC-246)

CONTENTS

ABSTRACT

Every vehicle has its own emission of gases, but the problem occurs when the emission is beyond the standardized values. The primary reason for this breach of emission level being the incomplete combustion of fuel supplied to the engine which is due to the improper maintenance of vehicles. This emission from vehicles cannot be completely avoided, but it definitely can be controlled. This project uses smoke detection and temperature sensors which can detect the abrupt environment condition continuously. A buzzer alert will also be given. This sends a signal to microcontroller. The aim of the project is to monitor the pollutants by using the pollution detection circuit. This pollution control circuit consists of various sensors like smoke sensor, temperature sensor and all of them are integrated and connected to a Controller. It is a real time work where a demo application has been made in which Atmega controller is used and a controller board is made where all these devices get integrated and work accordingly. When the system detects certain threshold pollution level then the buzzer alarm will get triggered to alert the people.

LITERATURE SURVEY

Various research carried out in this area shows the systems and interfaces are continuously developed to adopt required performance in the applications.

Giovanni B. Fioccola et al. have proposed, Arduino based air pollution monitor system. Air contamination is hazard factor for numerous wellbeing conditions including skin and eye disease, disturbance of nose, throat and eyes. It likewise causes genuine conditions like coronary illness, lung malignant growth trouble in breathing and many. At that point put away to cloud stage that keeps up the information accepting from the detecting modules. The idea of observing utilizing a microcontroller which gauges the air signals. Gas sensors are utilized to quantify the contamination level. This information is transferred by means of web and can likewise be checked in cell phones

Arushi Singh et al. have proposed a system which uses sound and air sensors to detect the information continually and afterward transmit the information. The principle goal of undertaking is by utilizing different sensors, GSM/GPRS module and Cloud/server to structure a productive and remote framework to observing the degree of different poisons causing contamination and to limit the impact of these parameters without influencing the common habitat and give live updates to keep away from clashes.

CHAPTER-1

INTRODUCTION

The large majority of today's cars and trucks travel by using internal combustion engines that burn gasoline or other fossil fuels. The process of burning gasoline to power cars and trucks contributes to air pollution by releasing a variety of emissions into the atmosphere. Emissions that are released directly into the atmosphere from the tailpipes of cars and trucks are the primary source of vehicular pollution. The incomplete combustion in the engine of a vehicle leads to emission of different gases contributing to increase in the pollution and adversely affecting the environment. Detection and control of these gases is an important area of work. This emission from vehicles cannot be completely avoided but, it definitely can be controlled. Now a day's accidents are common reason for deaths. These are critical things to control so here we come up with a concept to reduce pollution and detect the location of accident using GPS. As a solution to the above problems we aim to build an automated control system for emission level control of vehicle and accident place detection. Smoke detector is used to detect the carbon percentage in the smoke released by the vehicle due to combustion of fuel in it. Smoke detector is fixed at the end of the exhaust of vehicle from where smoke is released into the environment.

1.1 Problem definition

Air pollution and excessive noise harm our health and our environment. Air pollution mainly stems from industry, transport, energy production and agriculture.

Environmental noise levels are rising in urban areas, mainly as a result of increasing traffic volumes and intensifying industrial and recreational

activities. It is estimated that around 20% of the population of the EU are subjected to noise levels that are considered unacceptable. This can affect quality of life and lead to significant levels of stress, sleep disturbance and adverse health effects, such as cardiovascular problems. Noise also has an Impact on wildlife.

The effects of air pollution could be;

1. Respiratory and heart problems

2. Child health problems

3. Acid rain

4. Effect on wildlife

5. Global warming

1.2 OBJECTIVE:

By incorporating the techniques of sensors and embedded system design, this environmental air monitoring system can provide a convenient and straightforward monitoring method that integrates multiple sensors into a single unit. This is a portable, user-friendly, low power and low cost air monitoring device that can measure environmental air parameters of interest in real-time. This device can be used anywhere to monitor air quality, including outdoors for industrial air pollution near factories and indoors for home, hospital or school use with sensitive populations. The data provided from the device can be used to distinguish between high and low air quality areas.

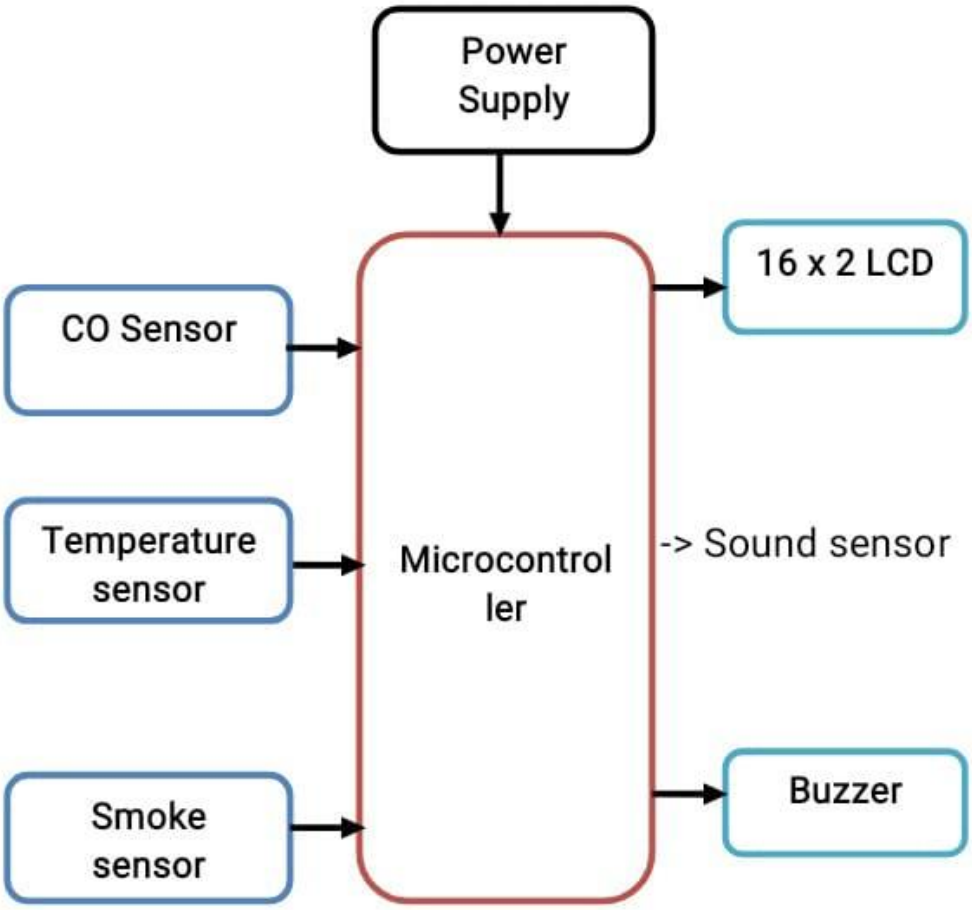
CHAPTER-2

WORKING AND BLOCK DIAGRAM

2.1 Working principle

This Project presents an automatic air pollution detection and alert system using temperature and smoke sensor units. This detection and alert system is composed of a Microcontroller and a display unit. In case of high content of pollution is detected, the information will be available in the controller. This project is designed with Atmega controller. A smoke sensor is included to detect the pollution caused from the vehicle in that case the details will be sent to the microcontroller. A temperature sensor is also interfaced to detect the raise in temperature so that controller will intimate. This is only for shorter distances only.

Block diagram



CHAPTER-3

HARDWARE MODULES

3.1 Atmega 328 controller

Features:

• High Performance, Low Power AVR® 8-Bit Microcontroller

• Advanced RISC Architecture

– 131 Powerful Instructions – Most Single Clock Cycle Execution

– 32 x 8 General Purpose Working Registers

– Fully Static Operation

– On-chip 2-cycle Multiplier

• High Endurance Non-volatile Memory Segments

– 4/8/16/32K Bytes of In-System Self-Programmable Flash program memory

(ATmega48PA/88PA/168PA/328P)

– 256/512/512/1K Bytes EEPROM (ATmega48PA/88PA/168PA/328P)

– 512/1K/1K/2K Bytes Internal SRAM (ATmega48PA/88PA/168PA/328P)

– Write/Erase Cycles: 10,000 Flash/100,000 EEPROM

– Data retention: 20 years at 85°C/100 years at 25°C(1)

– Optional Boot Code Section with Independent Lock Bits

In-System Programming by On-chip Boot Program

True Read-While-Write Operation

– Programming Lock for Software Security

• Peripheral Features

– Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode

– One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode

– Real Time Counter with Separate Oscillator

– Six PWM Channels

• Special Microcontroller Features

– Power-on Reset and Programmable Brown-out Detection

– Internal Calibrated Oscillator

– External and Internal Interrupt Sources

– Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby,

and Extended Standby

• I/O and Packages

– 23 Programmable I/O Lines

– 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF

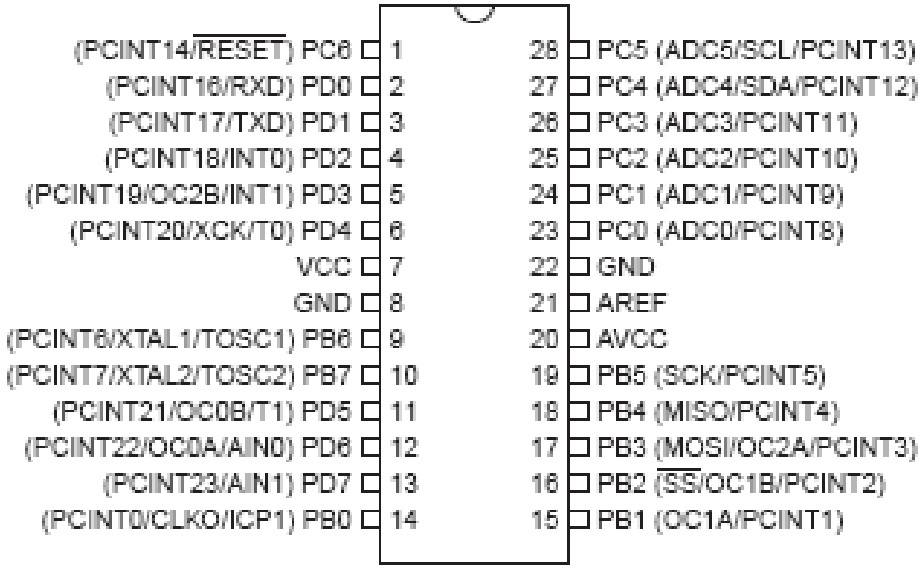
• Operating Voltage:

– 1.8 - 5.5V for ATmega48PA/88PA/168PA/328P

• Temperature Range:

– -40°C to 85°C

PDIP



3.1 Pin Descriptions

3.1.1 VCC Digital supply voltage.

3.1.2 GND Ground.

3.1.3 Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each it). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit. Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7..6 is used as TOSC2..1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set. The various special features of Port B are elaborated in ”Alternate Functions of Port B” on page 82 and ”System Clock and Clock Options” on page 26.

3.1.4 Port C (PC5:0)

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each it). The PC5..0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

3.1.5 PC6/RESET

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C. If the RSTDISBL Fuse is un programmed, PC6 is used as a Reset input. A low level on

this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running.

The minimum pulse length is given in Table 28-3 on page 318. Shorter pulses are not guaranteed to generate a Reset. The various special features of Port C are elaborated in ”Alternate Functions of Port C” on page 85.

3.1.6 Port D (PD7:0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running. .

3.1.7 AVCC

AVCC is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that PC6..4 use digital supply voltage, VCC.

3.1.8 AREF

AREF is the analog reference pin for the A/D Converter.

3.1.9 ADC7:6 (TQFP and QFN/MLF Package Only)

In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

3.2 CO Sensor

A **carbon monoxide detector** or **CO detector** is a device that detects the presence of the carbon monoxide (CO) gas to prevent carbon monoxide poisoning. In the late 1990s Underwriters Laboratories changed the definition of a single station CO detector with a sound device to **carbon monoxide (CO) alarm**.

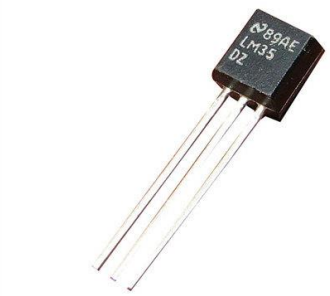
CO is a colorless, tasteless and odorless gas produced by incomplete combustion of carbon-containing materials. It is often referred to as the "silent killer" because it is virtually undetectable by humans.

3.3 Temperature sensor

LM35 is a precision IC temperature sensor with its output proportional to the temperature (in^{o}C). The sensor circuitry is sealed and therefore it is not subjected to oxidation and other processes. With LM35, temperature can be measured more accurately than with a thermistor. It also possess low self heating and does not cause more than 0.1^{o}C temperature rise in still air.

The operating temperature range is from -55°C to 150°C. The output voltage varies by 10mV in response to every ^{o}C rise/fall in ambient temperature, *i.e.*, its scale factor is 0.01V/ ^{o}C.

.

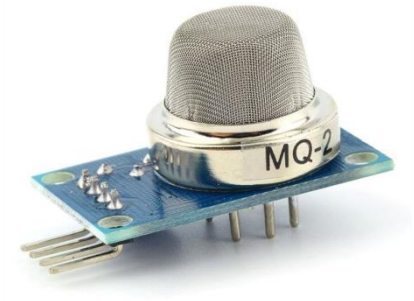


The LM35 does not require any external calibration or trimming and maintains an accuracy of +/-0.4°C at room temperature and +/-0.8°C over a range of 0°C to +100°C.Another important characteristic of the LM35 is that it draws only 60 micro amps from its supply and possesses a low self-heating capability. The LM35 comes in many different packages such as TO-92 plastic transistor-like package, T0-46 metal can transistor-like package,8-lead surface mount SO-8 small outline package.

3.4 Smoke sensor

A **smoke detector** is a device that senses smoke, typically as an indicator of fire. Commercial smoke detectors issue a signal to a fire alarm control panel as part of a fire alarm system

. Smoke can be detected either optically (photoelectric) or by physical process (ionization). Detectors may use one or both sensing methods. Sensitive alarms can be used to detect and deter smoking in banned areas. Smoke detectors in large commercial and industrial buildings are usually connected to a central fire alarm system



.

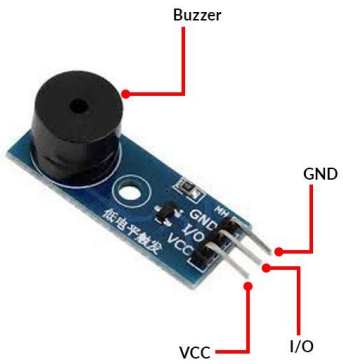
3.5 Sound sensor

Sound sensor is defined as a module that detects the intensity of sound and gives a buzzer if sound is more than threshold value

3.6 Buzzer

An audio signaling device like a beeper or buzzer may be electromechanical or piezoelectric or mechanical type. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren.

An audio signaling device like a beeper or buzzer may be electromechanical or piezoelectric or mechanical type. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren.



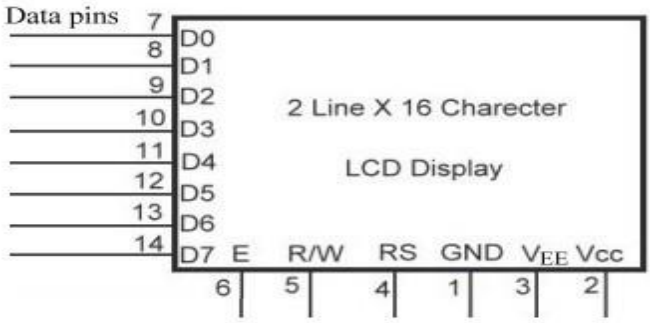
3.7 LCD

LCD is defined as the diode that uses small cells and the ionised gases for production of images. Here in this system we use LCD to display the respective values detected by respective sensors.

Interfacing an LCD to the microcontroller:

LCD pin descriptions:

The LCD discussed in this section has 14 pins. The function of each pin is given in table.



Pin	Symbol	I/O	Description
1	GND	-	Ground
2	Vcc	-	+5V power supply
3	VEE	-	Contrast control
4	RS	I	command/data register selection
5	R/W	I	write/read selection
6	E	I/O	Enable
7	DB0	I/O	The 8-bit data bus
8	DB1	I/O	The 8-bit data bus
9	DB2	I/O	The 8-bit data bus
10	DB3	I/O	The 8-bit data bus
11	DB4	I/O	The 8-bit data bus
12	DB5	I/O	The 8-bit data bus
13	DB6	I/O	The 8-bit data bus
14	DB7	I/O	The 8-bit data bus

Vcc, Vss, and VEE:

While Vcc and Vss provide +5V and ground, respectively, VEE is used for controlling LCD contrast**.**

RS – register select:

There are two very important registers inside the LCD. The RS pin is used for their selection as follows. If RS = 0, the instruction command code register is selected, allowing the user to send a command such as clear display, cursor at home, etc. If RS = 1 the data register is selected, allowing the user to send data to be displayed on the LCD.

R/W – read/write:

R/W input allows the user to write information to the LCD or read information from it. R/W = 1 when reading; R/W =0 when writing.

E – enable:

The enable pin is used by the LCD to latch information presented to its data pins. When data is supplied to data pins, a high to low pulse must be applied to this pin in order for the LCD to latch in the data present at the data pins. This pulse must be a minimum of 450 ns wide.

D0 – D7:

The 8 bit data pins, D0 – D7, are used to send information to the LCD or read the contents of the LCD's internal registers.

To display letters and numbers, we send ASCII codes for the letters A – Z, a – z, and numbers 0 – 9 to these pins while making RS = 1.

There are also instructions command codes that can be sent to the LCD to clear the display or force the cursor to the home position or blink the cursor. Table below lists the instruction command codes.

Code (hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Shift cursor to left
5	Shift display right
6	Shift cursor to right
7	Shift display left
8	Display off, Cursor off
A	Display off, Cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning of 1st line
C0	Force cursor to beginning of 2nd line
38	2 lines and 5x7 matrix

LCD Operation:

In recent years the LCD is finding widespread use replacing LEDs (seven-segment LEDs or other multisegment LEDs). This is due to the following reasons:

1. The declining prices of LCDs.
2. The ability of display numbers, characters, and graphics. This is ain contrast to LEDs, which are limited to numbers and a few characters.
3. Incorporation of a refreshing controller into the LCD, thereby relieving the CPU of the task of refreshing the LCD. In contrast, the LED must be refreshed by the CPU (or in some other way) to keep displaying the data.

3.8 Power Supply

Every electrical and electronic device that we use in our day-to-day life will require a power supply. In general, we use an AC supply of 230V 50Hz, but this power has to be changed into the required form with required values or voltage range for providing power supply to different types of devices. There are various types of power electronic converters such as step-down converter, step-up converter, voltage stabilizer, AC to DC converter, DC to DC converter, DC to AC converter, and so on. For example, consider the microcontrollers that are used frequently for developing many embedded systems' based projects and kits used in real-time applications. These microcontrollers require a 5V DC supply, so the AC 230V needs to be converted into 5V DC using the step-down converter in their power supply circuit.

Power supply circuit, the name itself indicates that this circuit is used to supply the power to other electrical and electronic circuits or devices. There are different types of power supply circuits based on the power they are used to provide for devices. For example, the micro-controller based circuits, usually the 5V DC regulated power supply circuits, are used, which can be designed using different techniques for converting the available 230V AC power to 5V DC power. Generally the converters with output voltage less than the input voltage are called as step-down converters.

Step Down the Voltage Level

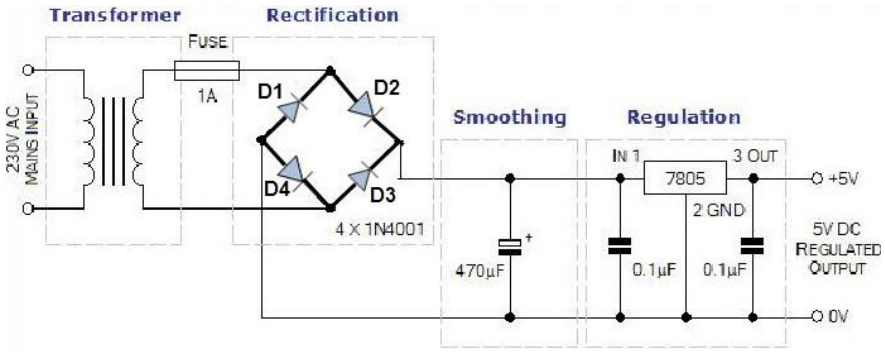
The step-down converters are used for converting the high voltage into low voltage. The converter with output voltage less than the input voltage is called as a step-down converter, and the converter with output voltage greater than the input voltage is called as step-up converter. There are step-up and step-down transformers which are used to step up or step down the voltage levels. 230V AC is converted into 12V AC using a step-down transformer. 12V output of stepdown transformer is an RMS value and its peak value is given by the product of square root of two with RMS value, which is approximately 17V.

Step-down Transformer

Step-down transformer consists of two windings, namely primary and secondary windings where primary can be designed using a less-gauge wire with more number of turns as it is used for carrying low-current high-voltage power, and the secondary winding using a high-gauge wire with less number of turns as it is used for carrying high-current low-voltage power. Transformers works on the principle of Faraday's laws of electromagnetic induction.

Convert AC to DC

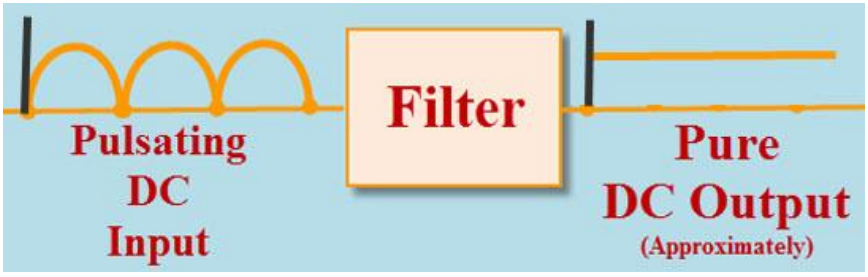
230V AC power is converted into 12V AC (12V RMS value wherein the peak value is around 17V), but the required power is 5V DC; for this purpose, 17V AC power must be primarily converted into DC power then it can be stepped down to the 5V DC. But first and foremost, we must know how to convert AC to DC? AC power can be converted into DC using one of the power electronic converters called as Rectifier. There are different types of rectifiers, such as half-wave rectifier, full-wave rectifier and bridge rectifier. Due to the advantages of the bridge rectifier over the half and full wave rectifier, the bridge rectifier is frequently used for converting AC to DC.



Bridge rectifier consists of four diodes which are connected in the form a bridge. We know that the diode is an uncontrolled rectifier which will conduct only forward bias and will not conduct during the reverse bias. If the diode anode voltage is greater than the cathode voltage then the diode is said to be in forward bias. During positive half cycle, diodes D2 and D4 will conduct and during negative half cycle diodes D1 and D3 will conduct. Thus, AC is converted into DC; here the obtained is not a pure DC as it consists of pulses. Hence, it is called as pulsating DC power. But voltage drop across the diodes is (2*0.7V) 1.4V; therefore, the peak voltage at the output of this retifier circuit is 15V (17-1.4) approx.

Smoothing the Ripples using Filter

15V DC can be regulated into 5V DC using a step-down converter, but before this, it is required to obtain pure DC power. The output of the diode bridge is a DC consisting of ripples also called as pulsating DC. This pulsating DC can be filtered using an inductor filter or a capacitor filter or a resistor-capacitor-coupled filter for removing the ripples. Consider a capacitor filter which is frequently used in most cases for smoothing.



Filter

CHAPTER-4

SOFTWARE REQUIREMENTS

4.1 ARDUINO

Arduino is a computer hardware and software company, project, and user community that designs and manufactures microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical world. The project's products are distributed as open-source hardware and software, which are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form.

The project's board designs use a variety of microprocessors and controllers. These systems provide sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards ("shields") and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, for loading programs from personal computers. The microcontrollers

are mainly programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project.

The Arduino project started in 2005 as a program for students at the Interaction Design Institute Ivrea in Ivrea, Italy, aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats, and motion detectors.

THE KEY FEATURES ARE:

- Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).
- Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.
- Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

Arduino Installation

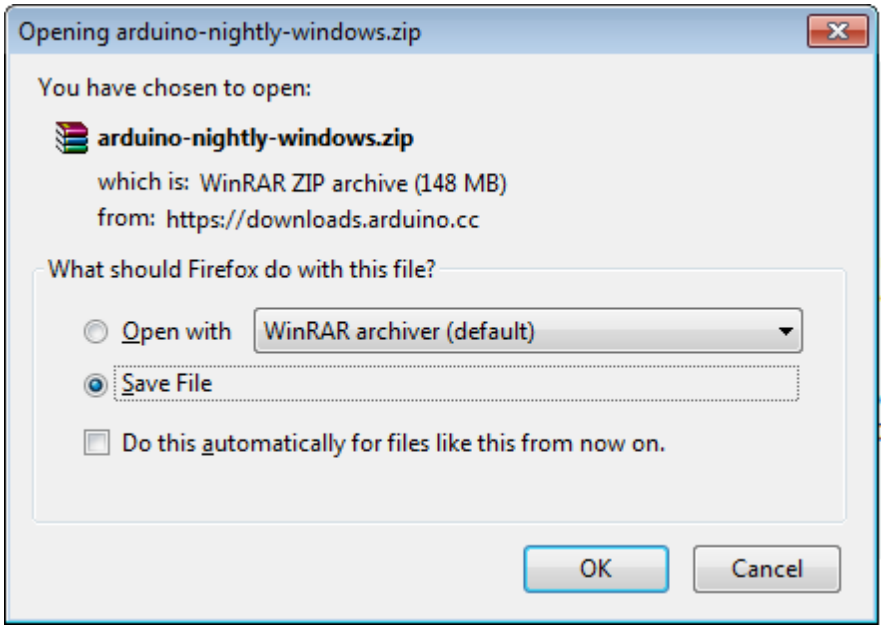
After learning about the main parts of the Arduino UNO board, we are ready to learn how to set up the Arduino IDE. Once we learn this, we will be ready to upload our program onthe Arduino board.In this section, we will learn in easy steps, how to set up the Arduino IDE on our computerand prepare the board to receive the program via USB cable.

Step 1: First you must have your Arduino board (you can choose your favorite board) anda USB cable. In case you use Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega 2560, or Diecimila, you will need a standard USB cable (A plug to B plug), the kind youwould connect to a USB printer as shown in the following image.

In case you use Arduino Nano, you will need an A to Mini-B cable instead as shown in the following image.

Step 2: Download Arduino IDE Software.

You can get different versions of Arduino IDE from the Download page on the Arduino Official website. You must select your software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.



Step 3: Power up your board**.**

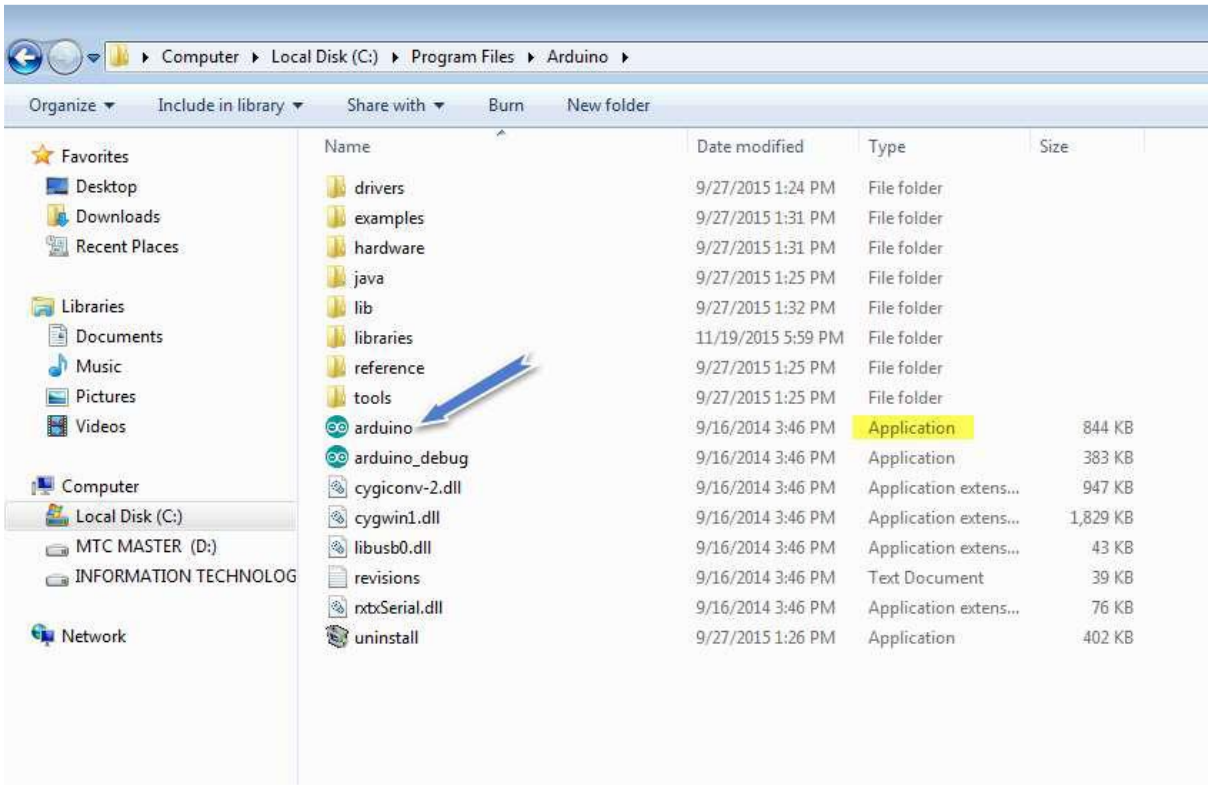
The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If you are using an Arduino Diecimila, you have to make sure that the board is configured to draw power

from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins closest to the USB port.

Connect the Arduino board to your computer using the USB cable. The green power LED (labeled PWR) should glow.

Step 4: Launch Arduino IDE.

After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Doubleclick the icon to start the IDE.

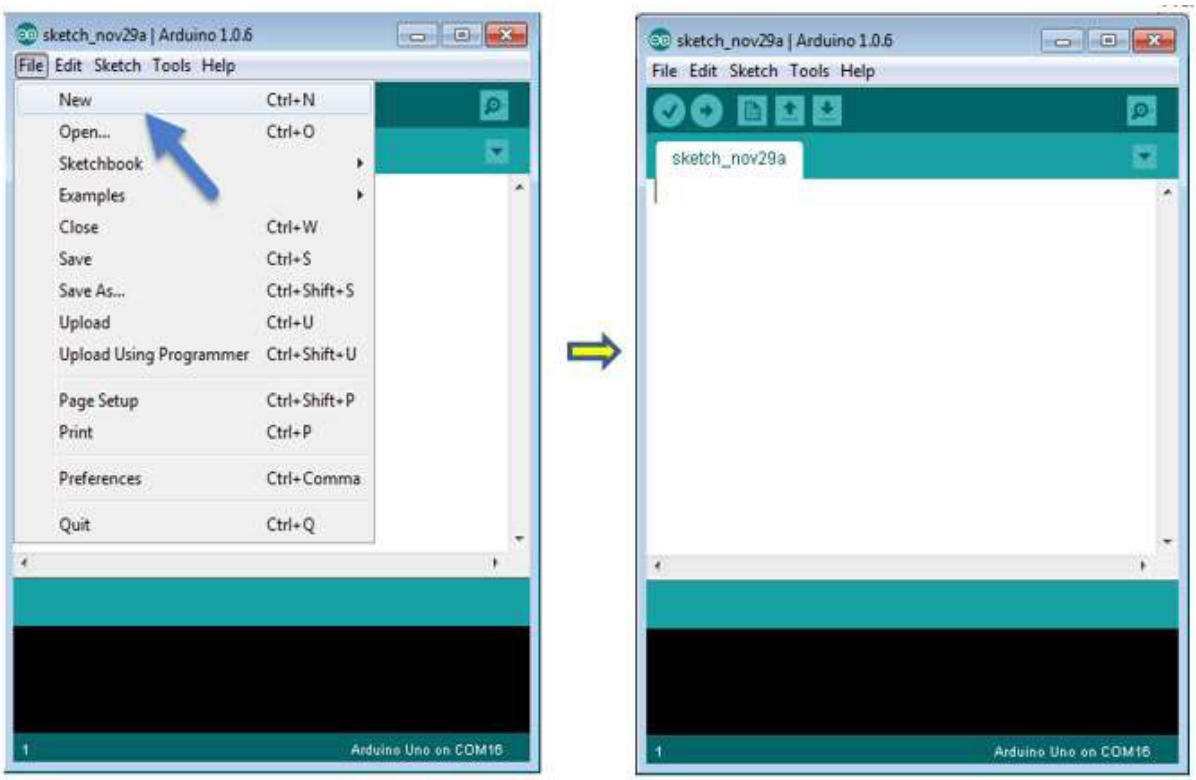


Step 5: Open your first project.

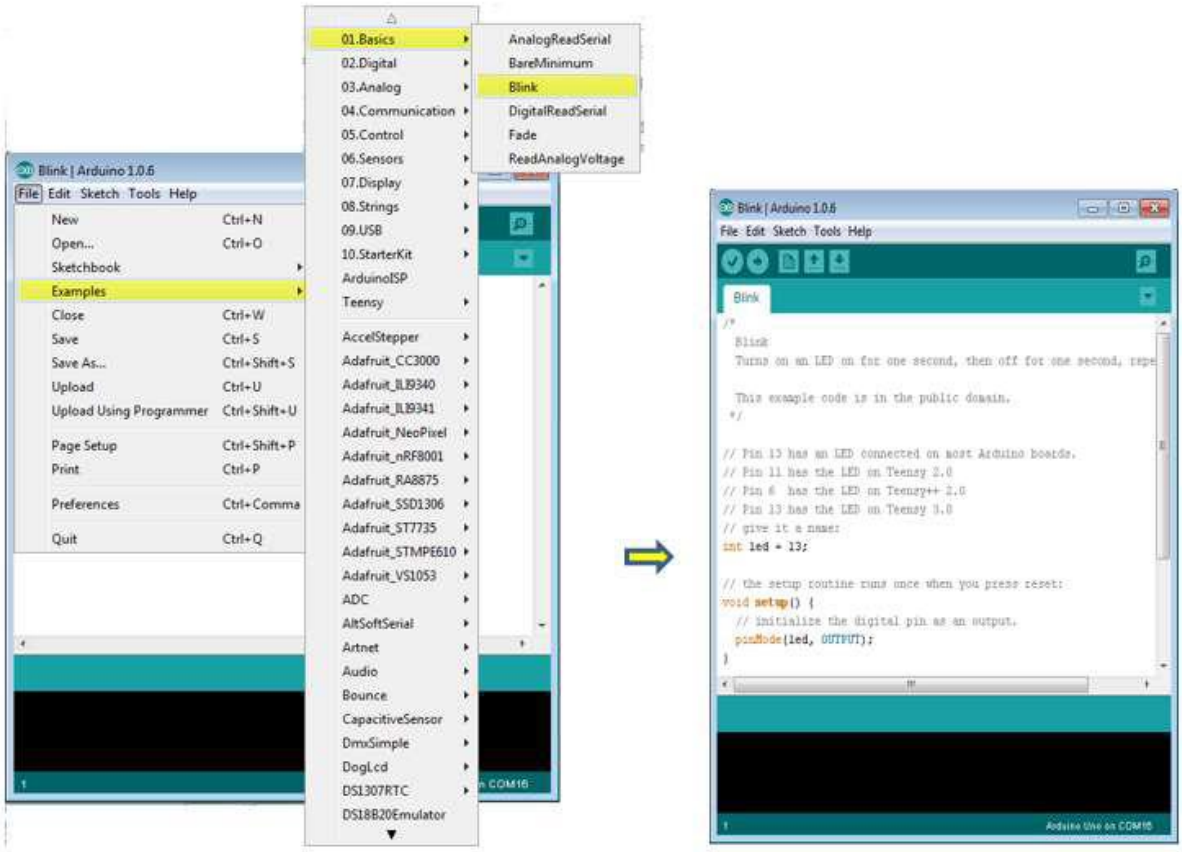
Once the software starts, you have two options:

- Create a new project.
- Open an existing project example.

To create a new project, select File --> New.



To open an existing project example, select File -> Example -> Basics -> Blink.

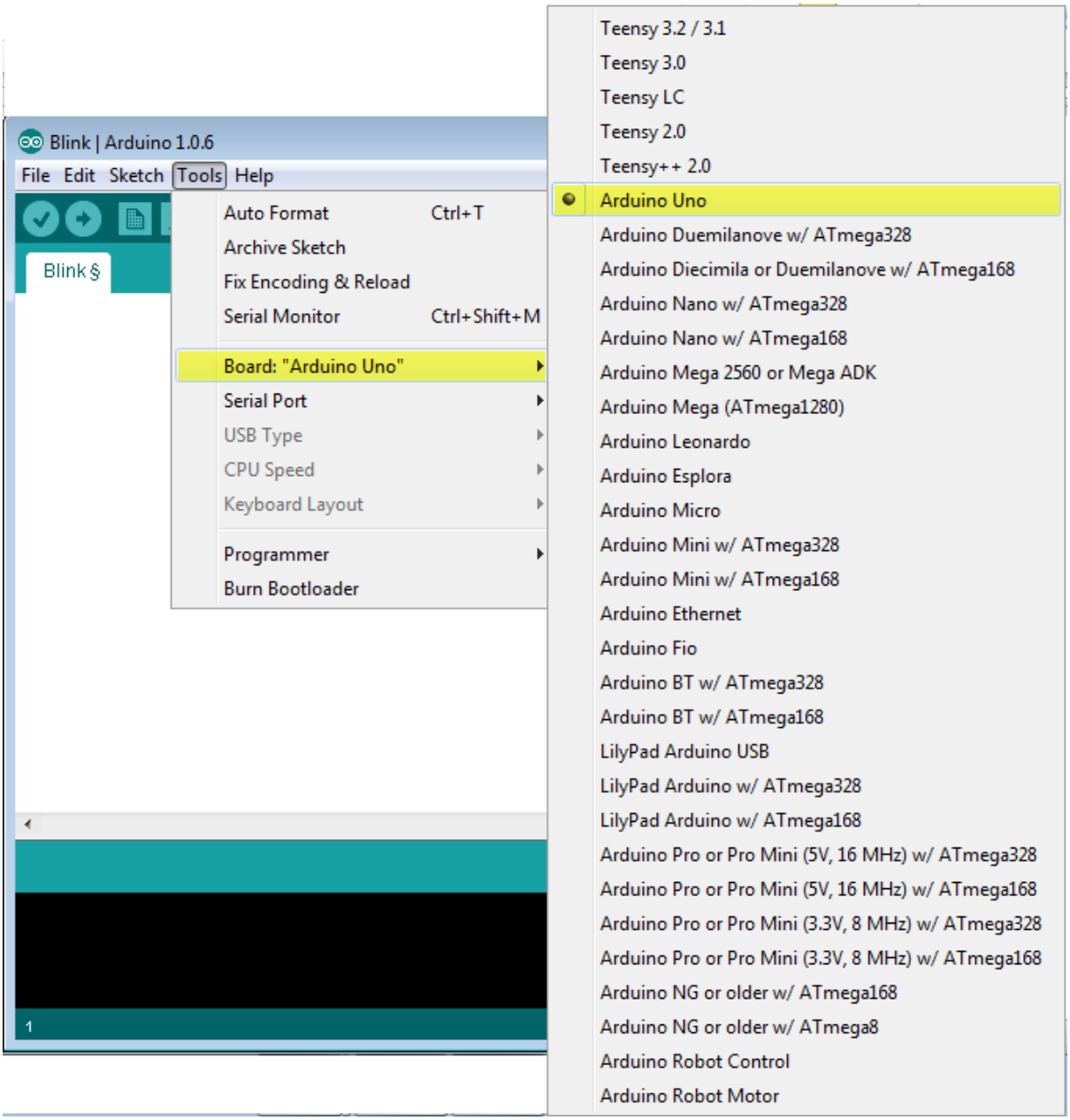


Here, we are selecting just one of the examples with the name Blink. It turns the LED on and off with some time delay. You can select any other example from the list.

Step 6: Select your Arduino board.

To avoid any error while uploading your program to the board, you must select the correct Arduino board name, which matches with the board connected to your computer.

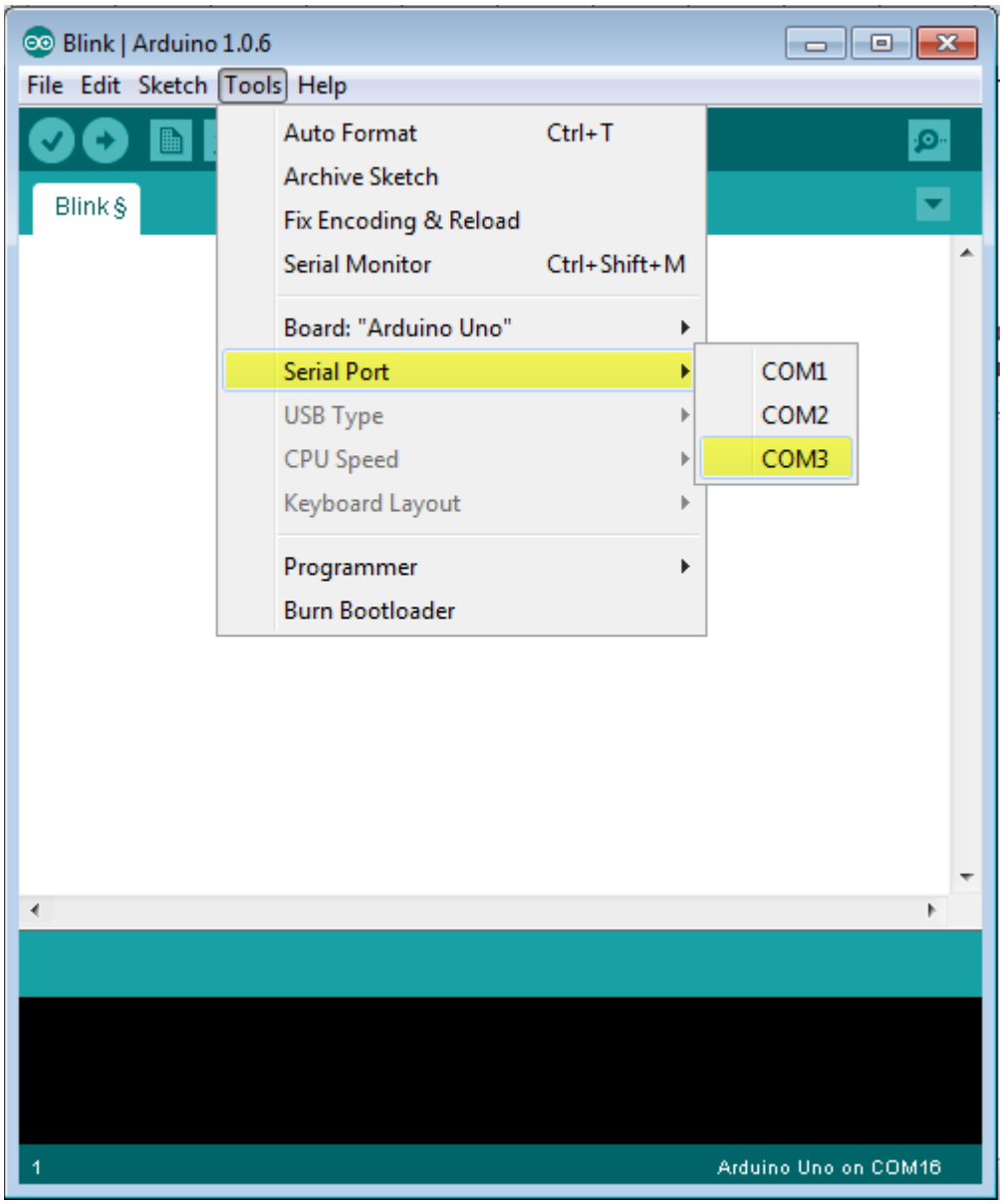
Go to Tools -> Board and select your board.



Here, we have selected Arduino Uno board according to our tutorial, but you must select the name matching the board that you are using.

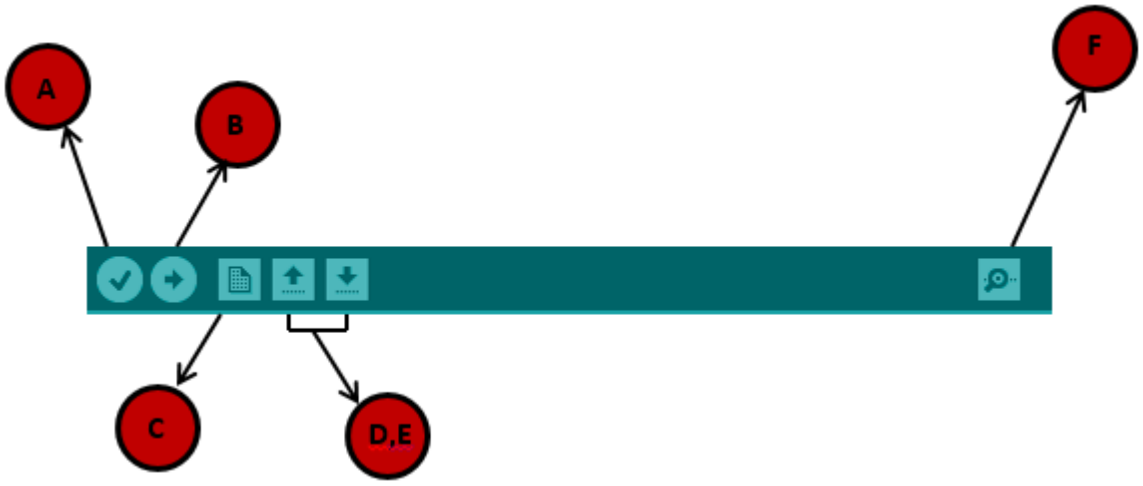
Step 7: Select your serial port.

Select the serial device of the Arduino board. Go to Tools -> Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.



Step 8: Upload the program to your board.

Before explaining how we can upload our program to the board, we must demonstrate the function of each symbol appearing in the Arduino IDE toolbar.



A- Used to check if there is any compilation error.

B- Used to upload a program to the Arduino board.

C- Shortcut used to create a new sketch.

D- Used to directly open one of the example sketch.

E- Used to save your sketch.

F- Serial monitor used to receive serial data from the board and send the serial data to the board.

Now, simply click the "Upload" button in the environment. Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

Note: If you have an Arduino Mini, NG, or other board, you need to press the reset button physically on the board, immediately before clicking the upload button on the Arduino Software.

These instructions mostly show Windows software. Except when indicated, the software (should be) identical on all platforms. Linux will be added once I figure out how to get it working (yay)

Not much is needed for this lesson, just a USB cable and an Arduino. If you have an older Arduino you may also need an LED. Any LED is fine as long as it looks sorta like the photo, with a plastic bulb and two legs.

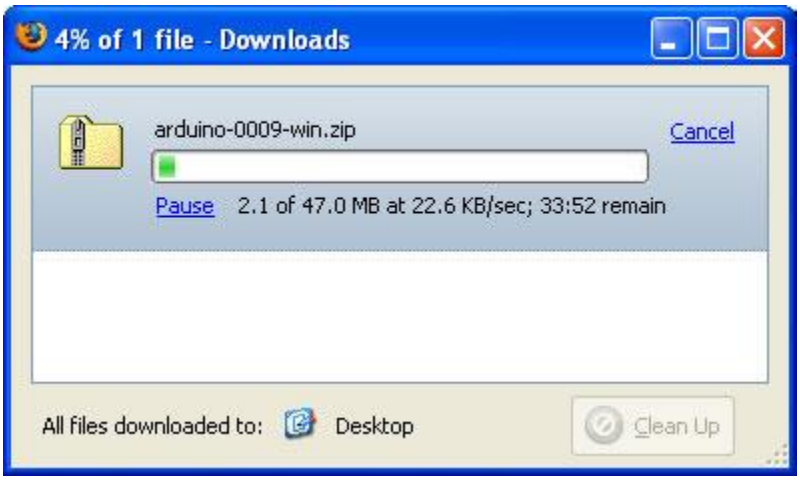
	Assembled Arduino board, preferrably a Diecimila (or whatever the latest version is)	Adafruit	$35
	USB Cable. Standard A-B cable is required. Any length is OK.	Adafruit Or any computer supply store	$5
	LED - **Optional** Nearly any LED is OK, as long as it has two wire legs. This part is only required for **NG rev c** Arduinos (and maybe other older ones). Diecimila Arduino's have this part 'built-in'	Any electronics supply store	$1

Download the Software

The first thing to do is download the Arduino software.

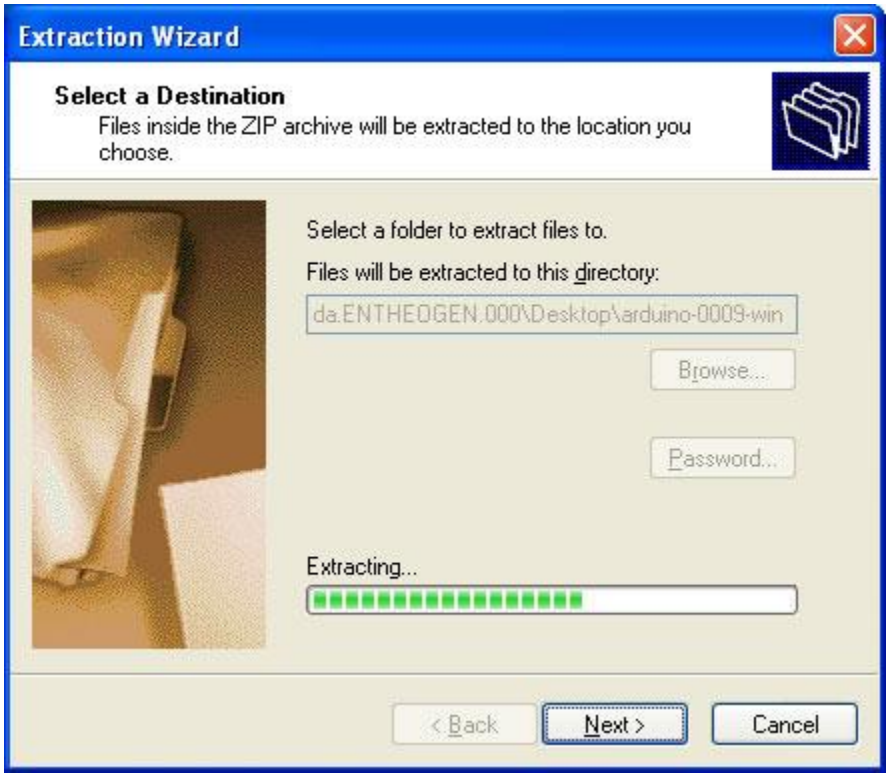
Go to the Arduino Software Download page and grab the right file for your OS. As of Sept 2007 the version is **009** but you should use whatever is most recent.

The packages are quite large, 30-50 MB so it may take a while to finish

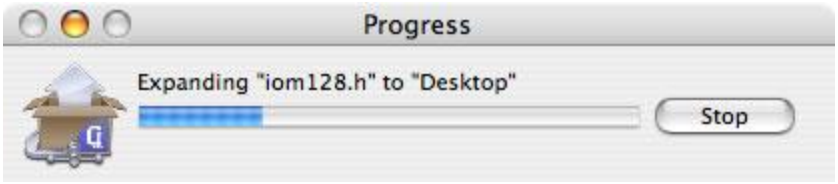


Unpack and Install

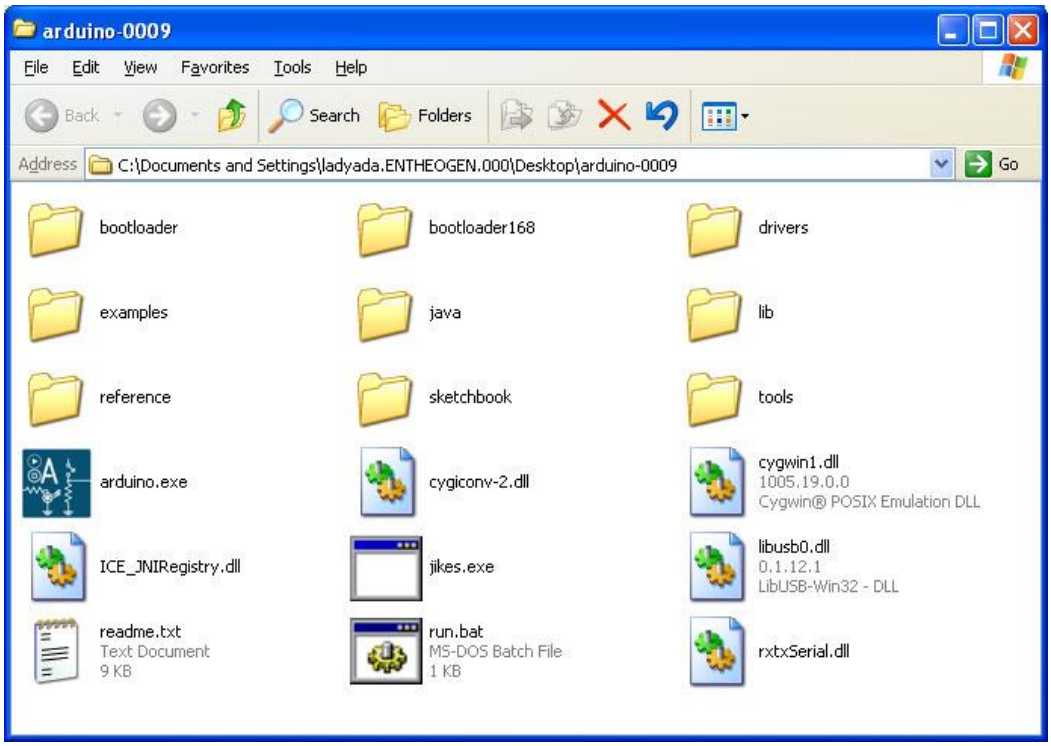
Extract the package onto the Desktop



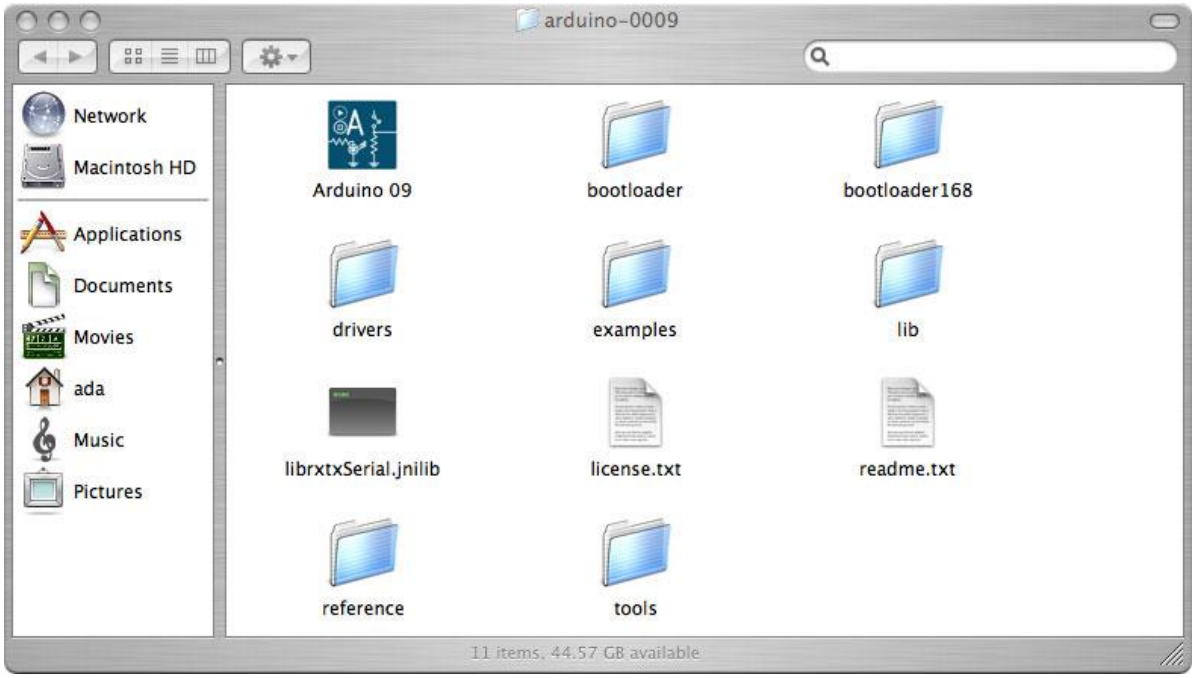
Windows



Mac OS X



Windows



Mac OS X

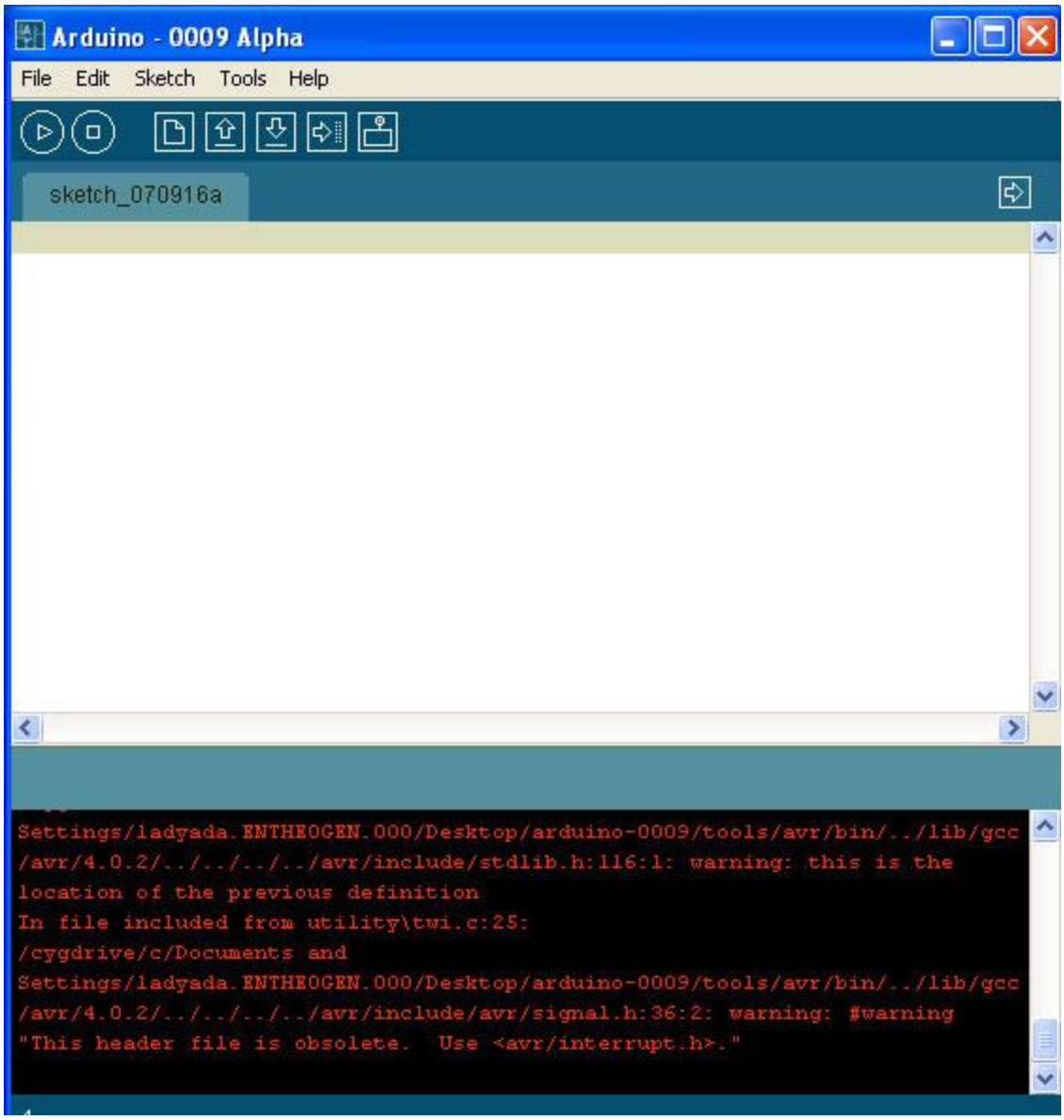
Startup!

Double click the Arduino software icon

Windows

Mac OS X

To open up the workspace



I think I get the red error text shown because I already have Arduino installed. Either way, it isn't a problem if you do or don't see it.

Select chip

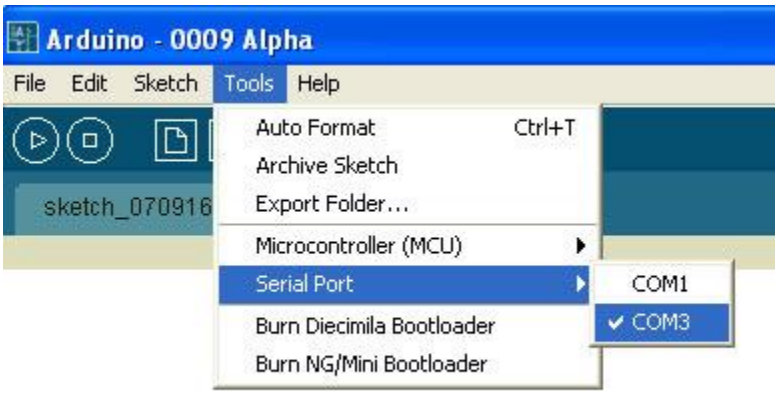
The first step is to configure the Arduino software for the correct chip. Almost all Arduinos use the ATmega168, but there's a chance you have an ATmega8. Look for the chip on the Arduino that looks like this:

If the text says ATMEGA8-16P then you have an **atmega8** chip. If the text says ATMEGA168-20P then you have an**atmega168** chip. If it says "ATMEGA328P-20P" you have an **atmega328p** chip

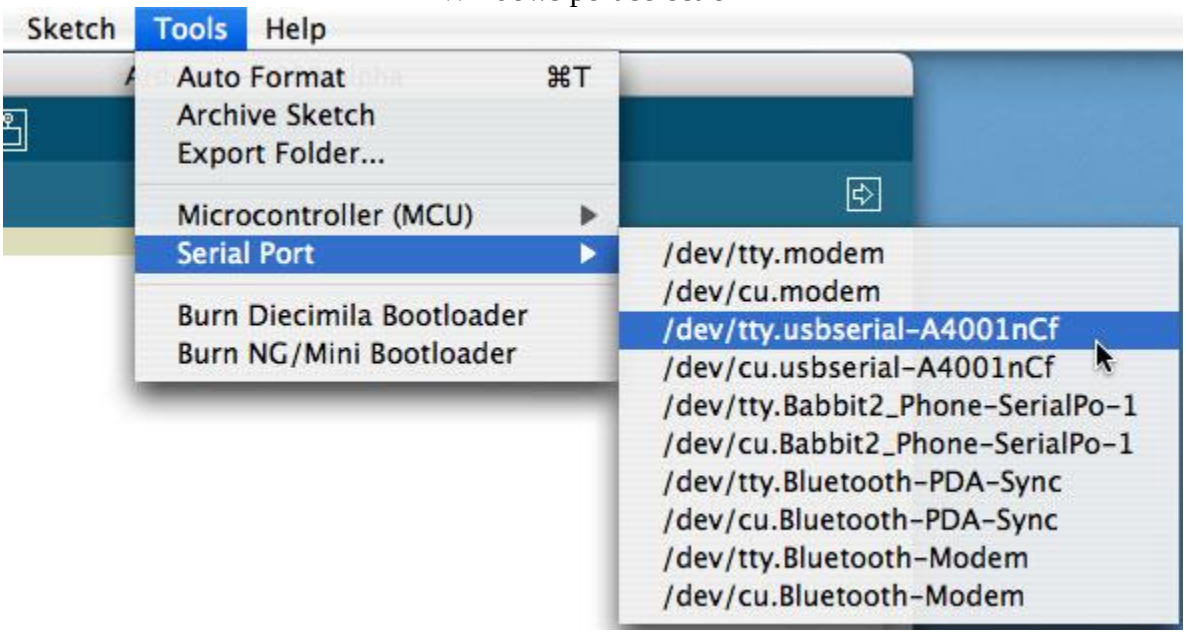
Make sure the correct chip is selected (this picture is really old, will be fixed soon). This preference is saved so you only have to set it once, the program will remember next time it's run.

Select port

Next, its time to configure the Serial Port (also known as the COM Port). Go back to lesson 0 to remind yourself of which port it is. On a PC it will probably be something like **COM3** or **COM4**. On a Mac it will be something like**tty.usbserial-xxxxx**



Windows port selection



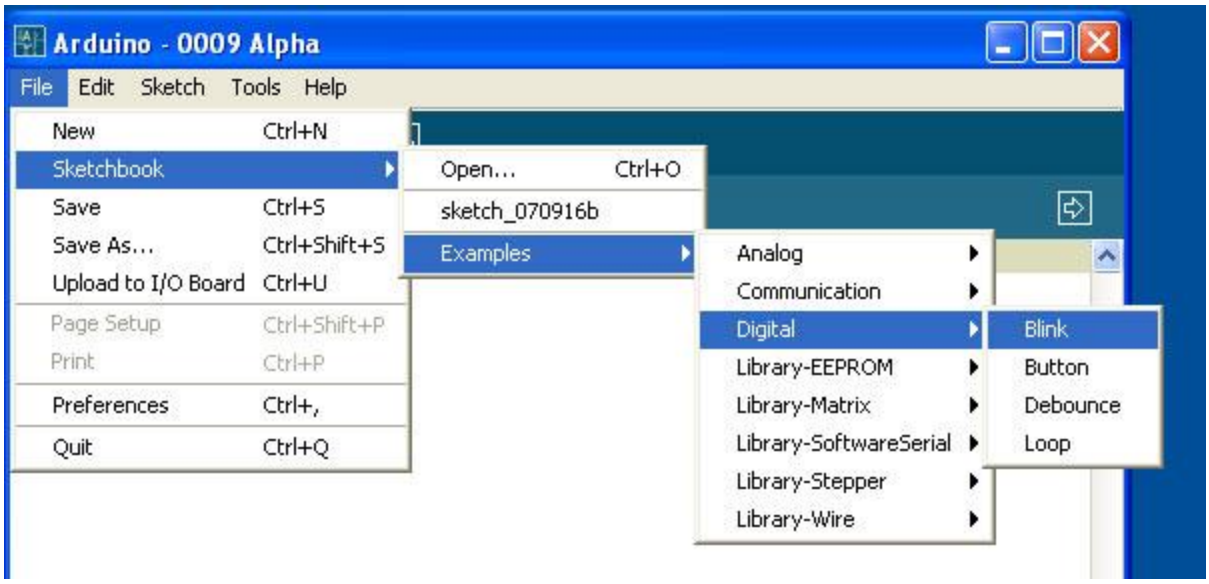
Mac port selection

This preference is saved so you only have to set it once, the program will remember next time it's run.

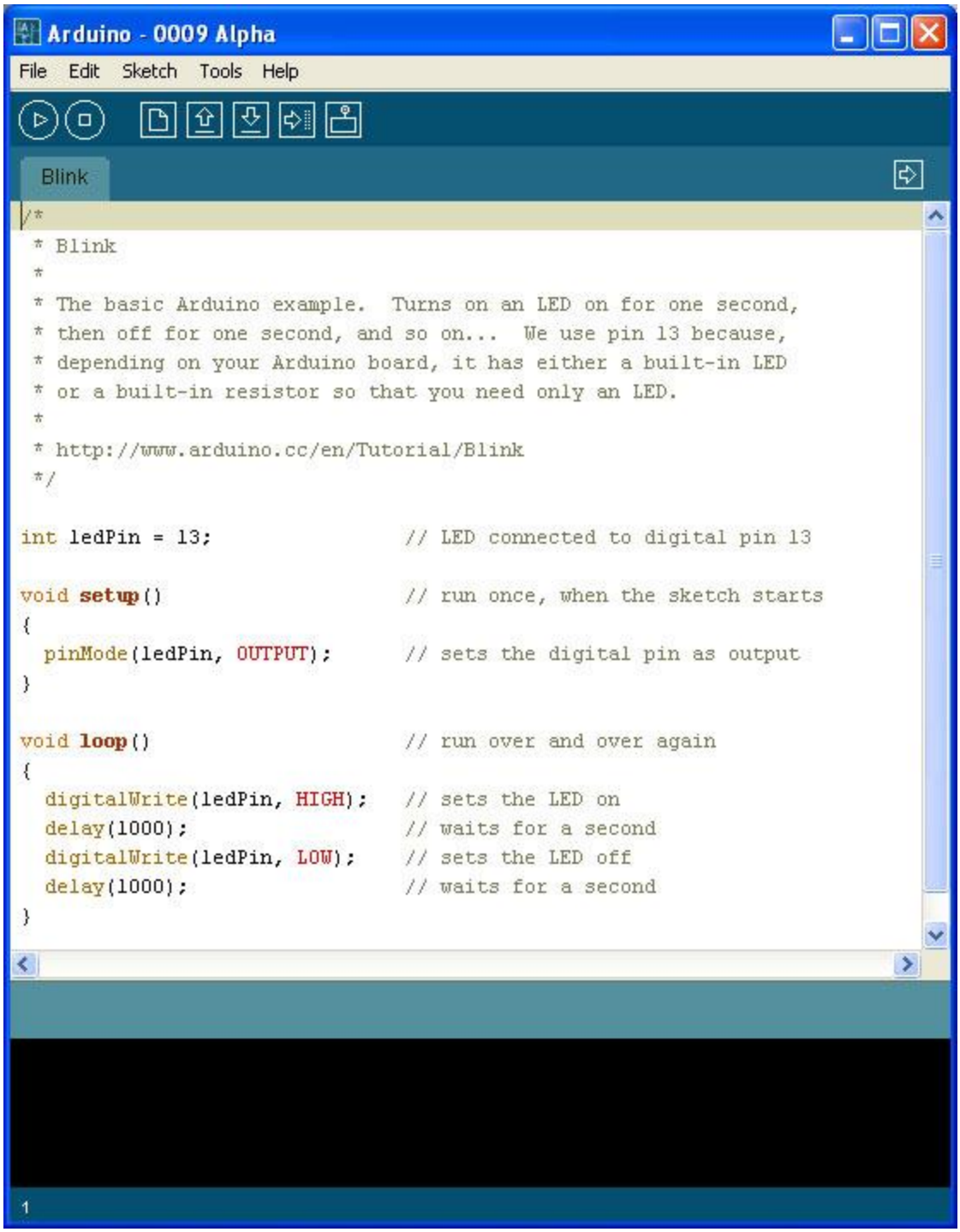
However, if you have multiple Arduino's, they may be assigned difference COM ports. So every time you plug in a new Arduino, double check that the correct port is selected.

Open blink sketch

Sketches are little scripts that you can send to the Arduino to tell it how to act. Let's open up an **Example Sketch**. Go to the **File menu -> Sketchbook -> Examples -> Digital -> Blink**

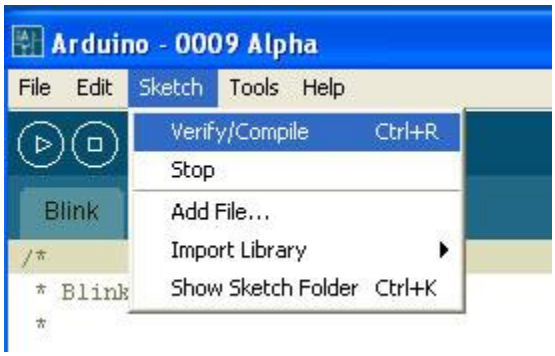


The window should now look like this, with a bunch of text in the formerly empty white space and the tab **Blink** above it

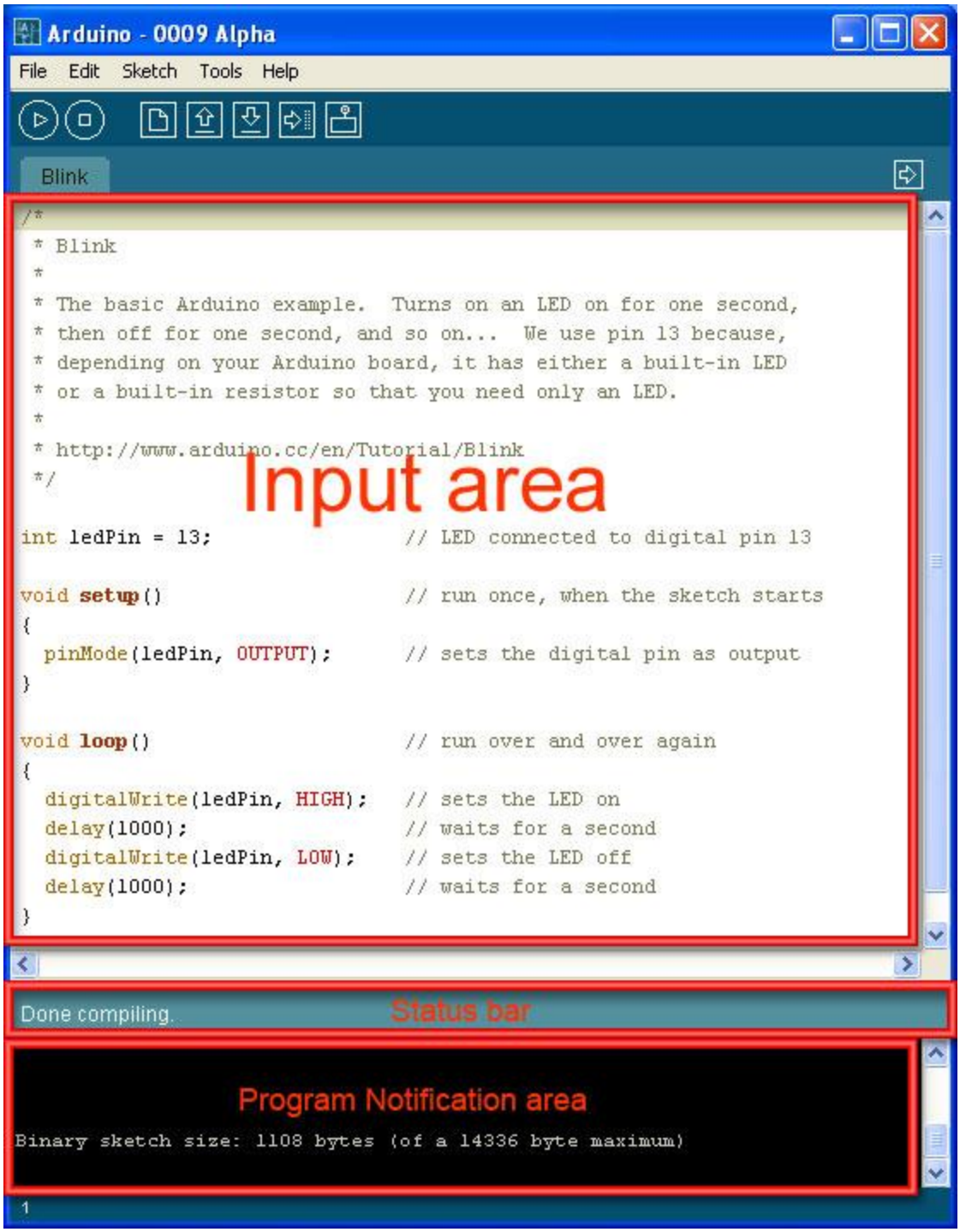


Verify / Compile

The first step to getting a **Sketch** ready for transfer over to the arduino is to **Verify/Compile** it. That means check it over for mistakes (sort of like editing) and then translate it into an application that is compatible with the Arduino hardware.

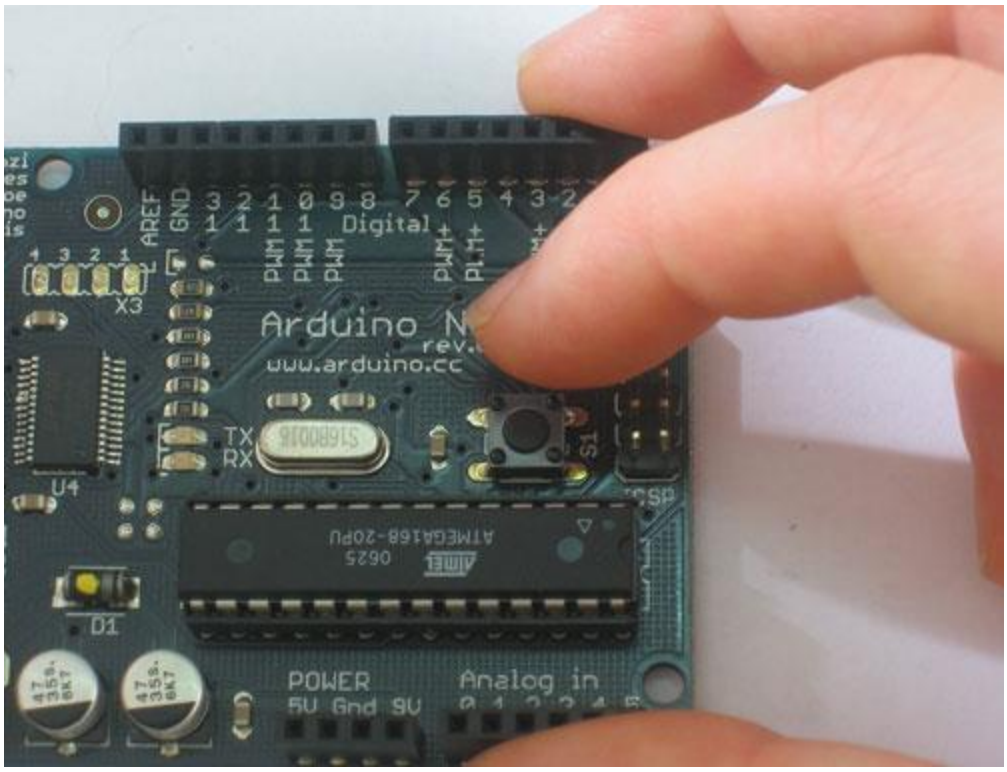


After a few seconds, you should see the message **Done compiling.** in the **Status Bar** and **Binary Sketch Size:** in the**Notification area.** This means the sketch was well-written and is ready for uploading to the Arduino board!



Reset (NG only)

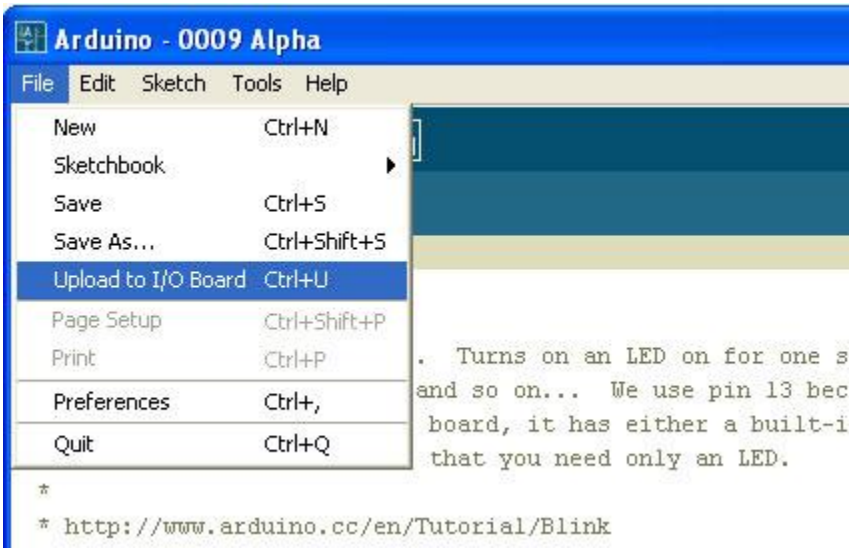
To tell the Arduino that it should prepare itself for a new Sketch upload, you must reset the board. Diecimila Arduino's have built-in auto-reset capability, so you don't need to do anything. Older Arduinos, such as NG, must be manually reset before uploading a sketch. To do that simply press the black button on the right hand side of the board, shown here.



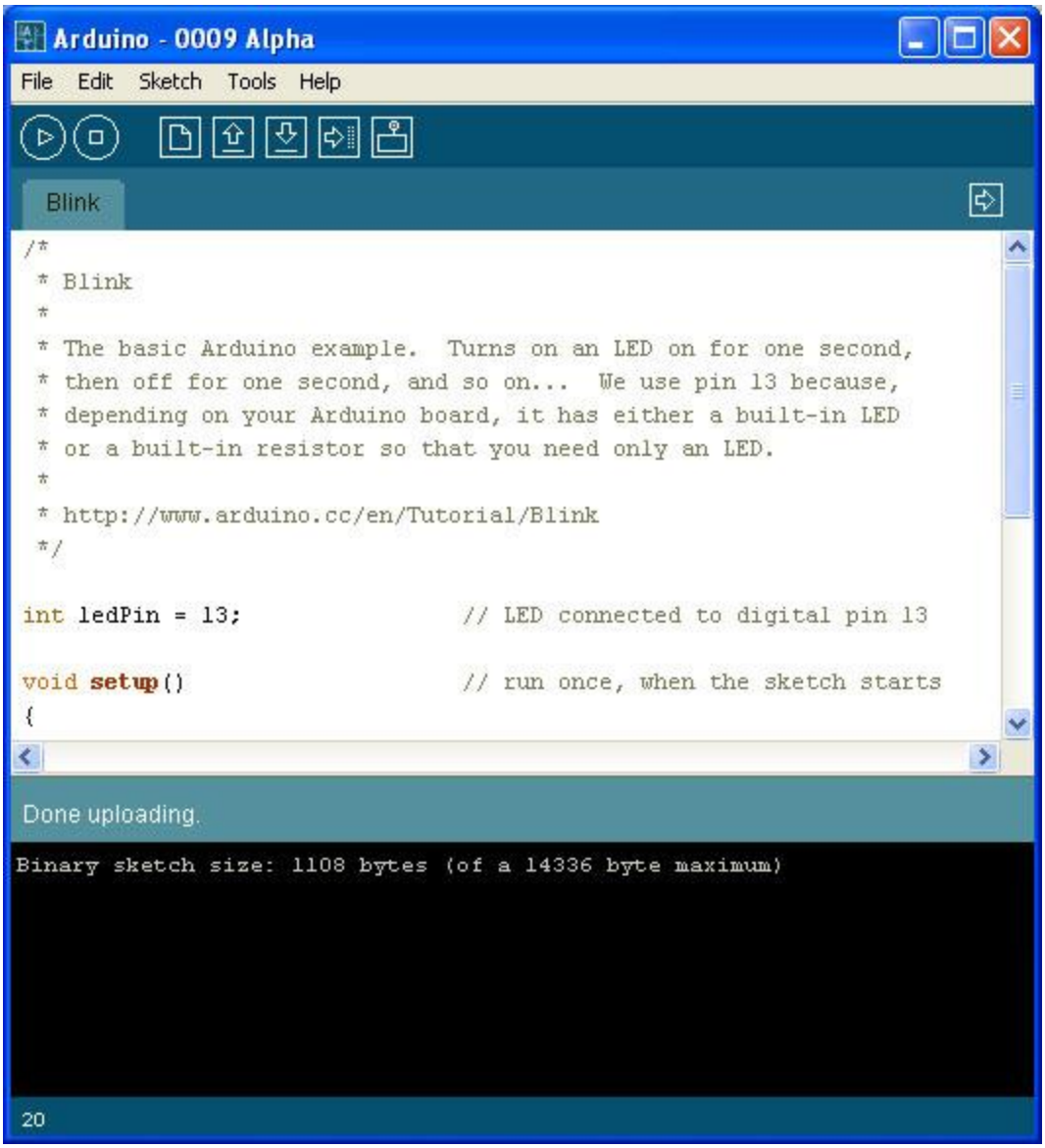
Upload

Now it's time to upload. Make sure the Arduino is plugged in, the green light is on and the correct Serial Port is selected.

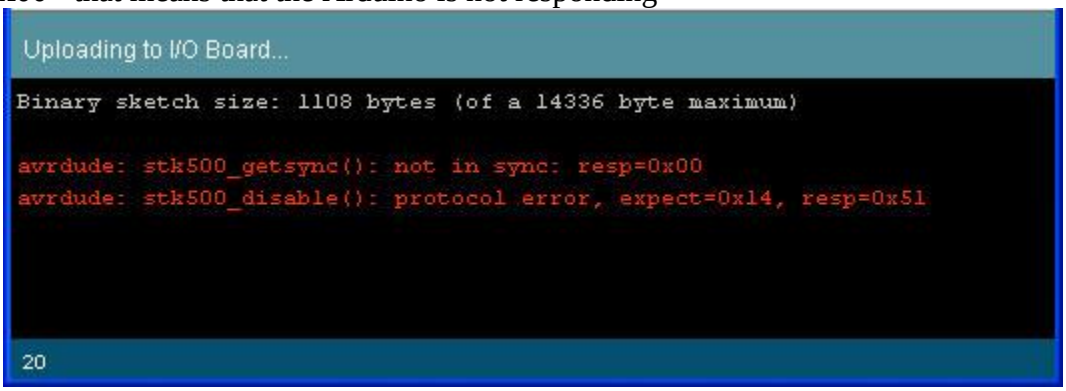
If you have an NG Arduino, press the **Reset Button** now, just before you select the **Upload** menu item.
Select **Upload to I/O Board** from the **File menu**



After a few seconds you should get this screen, with the message **Done uploading.** in the status bar.



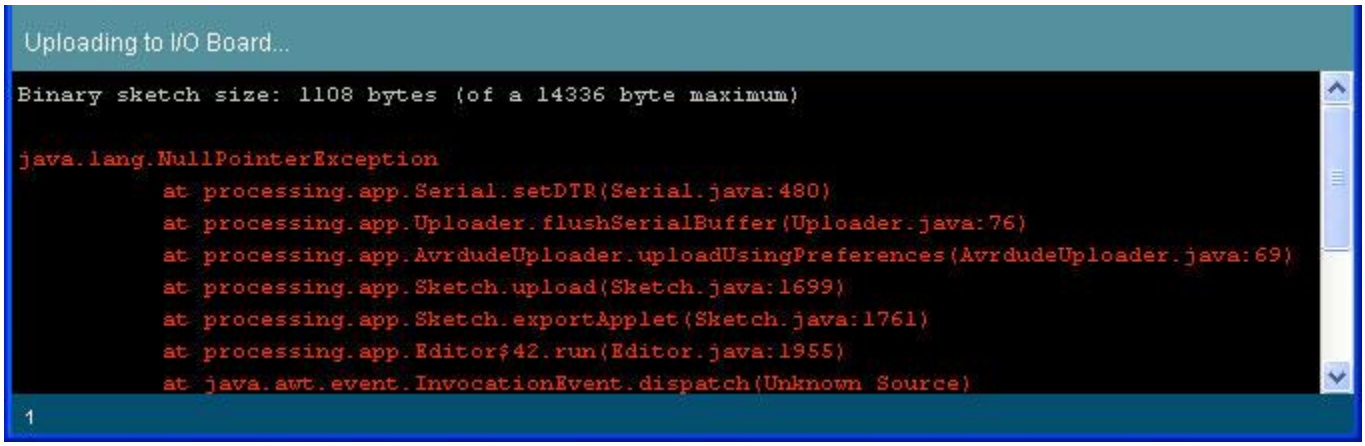
If you get the following error message "**avrdude: stk500_getsync(): not in sync: resp=0x00**" that means that the Arduino is not responding



Then check the following:

- If you have a NG Arduino, did you press reset just before selecting **Upload** menu item?
- Is the correct Serial Port selected?
- Is the correct driver installed?
- Is the chip inserted into the Arduino properly? (If you built your own arduino or have burned the bootloader on yourself)
- Does the chip have the correct bootloader on it? (If you built your own arduino or have burned the bootloader on yourself)

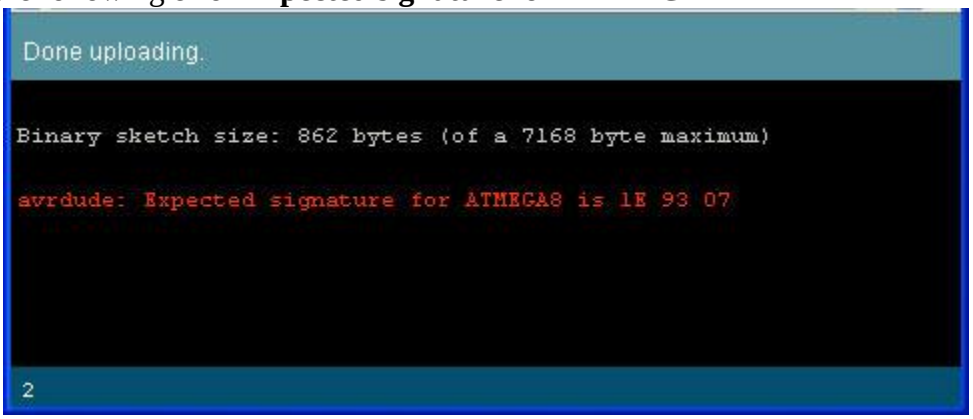
If you get the following error message:



It means you dont have a serial port selected, go back and verify that the correct driver is installed (lesson 0)

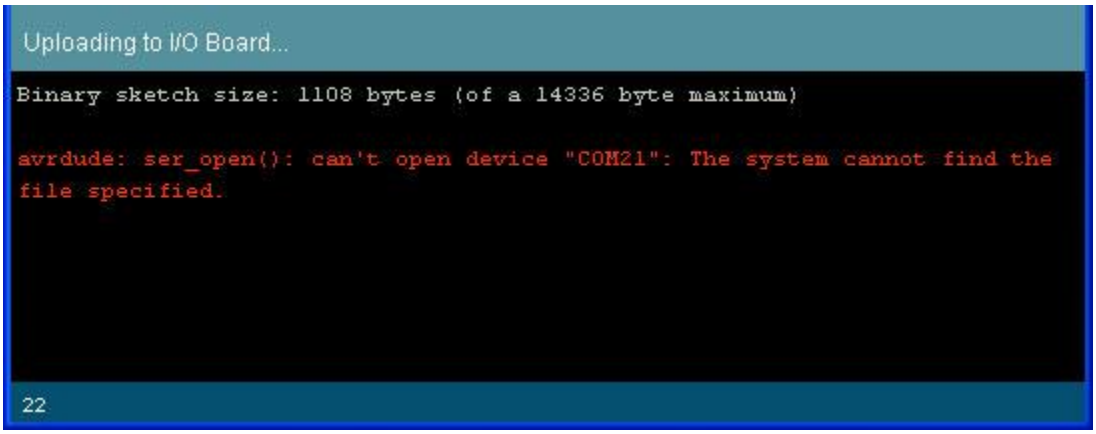
and that you have the correct serial port selected in the menu.

If you get the following error **Expected signature for ATMEGA**

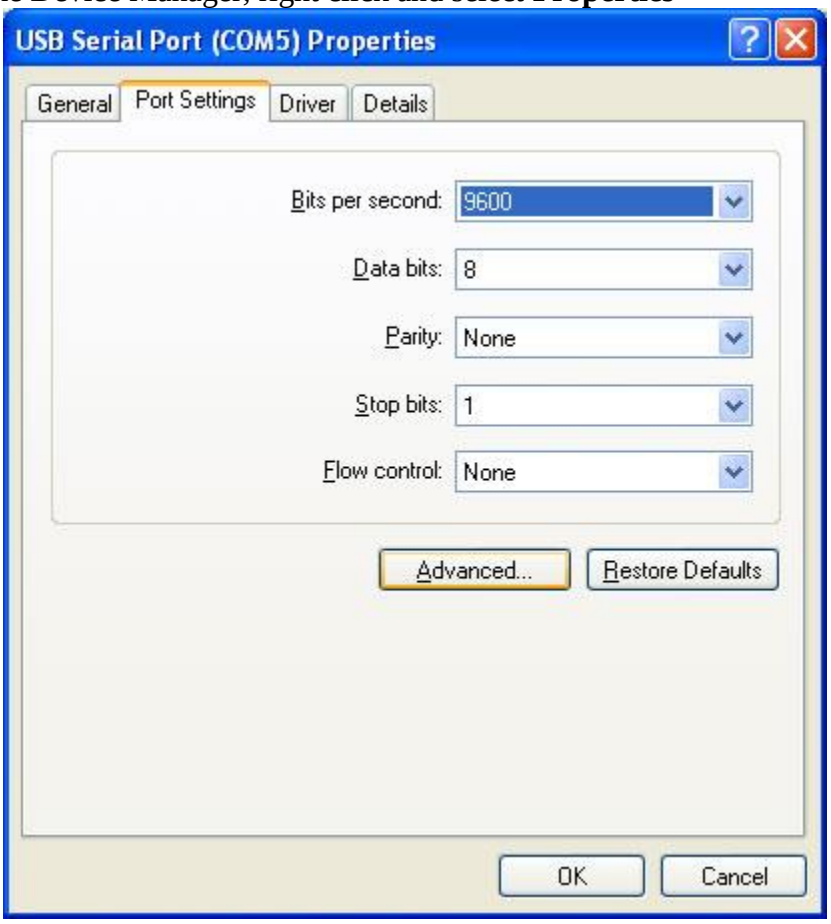


Then you have either the incorrect chip selected in the **Tools** menu or the wrong bootloader burned onto the chip

If you get the following error: **can't open device "COM10": The system cannot find the file specified** (under Windows, COM port value may vary)



It means that you have too many COM ports (maybe you've got 9 Arduinos?) You should make sure that the port is numbered as low as possible. You can use a program like FTClean to clear out old COM ports you aren't using anymore. Once you've cleaned out the ports, you'll have to reinstall the driver again (see lesson 0).
Alternately, if you're sure that the ports are not used for something else but are left over from other USB devices, you can simply change the COM port using the **Device Manager**. Select the USB device in the Device Manager, right click and select **Properties**



Then click **Advanced**... and in the next window change the COM port to something like **COM4** or **COM5**. Don't forget to select the new port name in the Arduino software. The lower port names may say **(in use**) but as long as the other USB devices aren't plugged in, it shouldn't be a problem. This is a little riskier than just using FTClean...

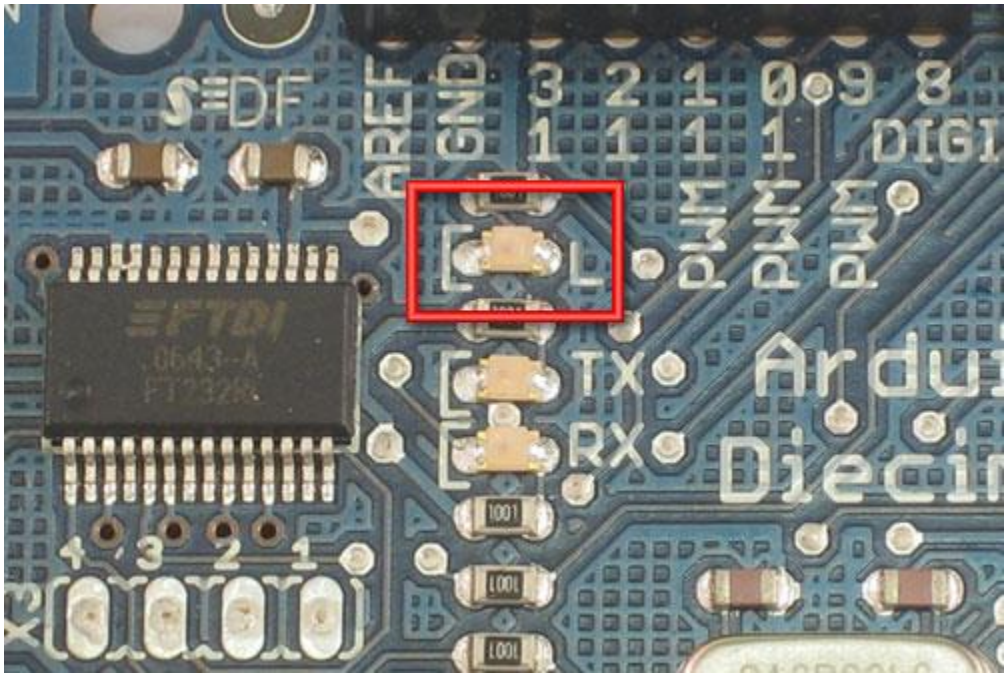
Video of all steps

Here is a video showing the timing of the steps described so far.

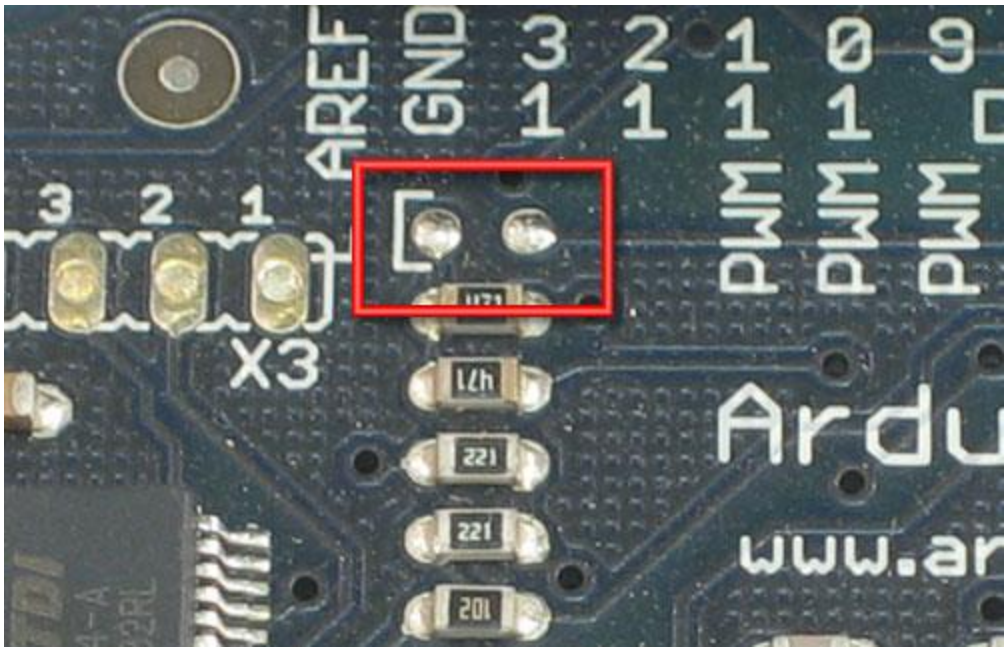
Insert LED (NG Arduinos)

Some older Arduinos don't have a built in LED, its easy to tell if yours does or not

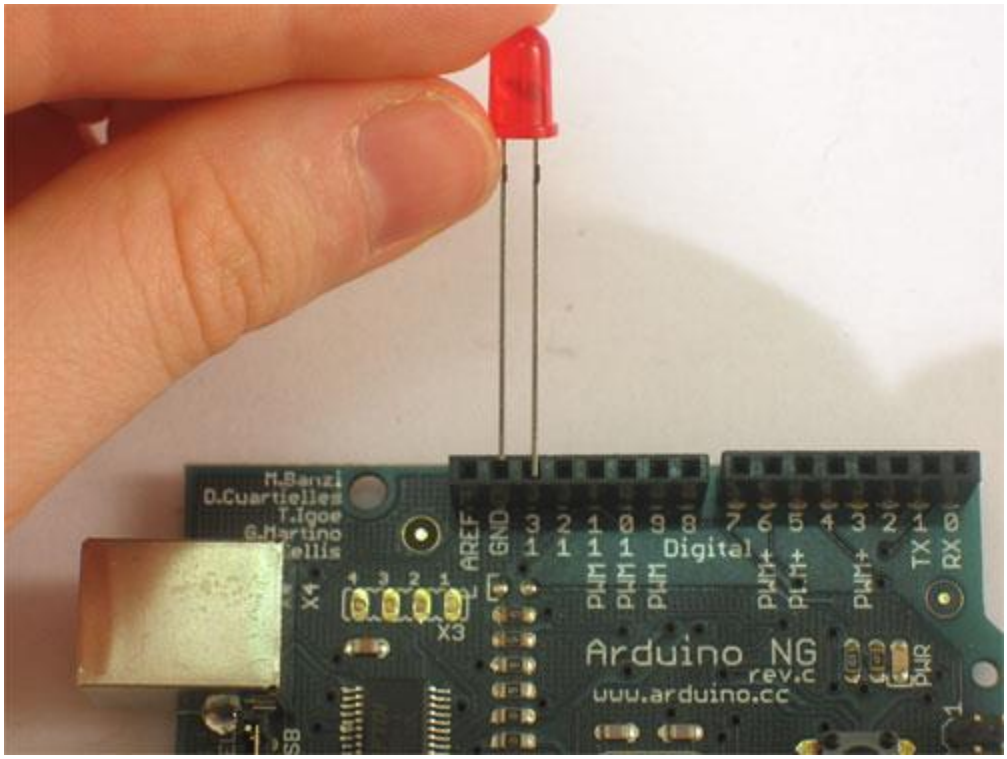
If you have a Diecimila or other Arduino with a built in LED you will see a translucent part as shown



If you have an NG rev C or other Arduino without an LED, the translucent part will not be there, and instead you will see two silver dots



If you don't have an LED, you'll need to add your own. Any LED will do, as long as it has two legs and kinda looks like the one shown here. LEDs are **directional** components. That means if you put it in backwards it will not work! To help you put the LED in right, the LED factory cuts the legs at different lengths. The longer leg goes in the hole marked **13**and the shorter one goes in the hole marked **GND**



4.2 Embedded C

An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions, sometimes with real-time computing constraints. It is usually embedded as part of a complete device including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, can do many different tasks depending on programming. Embedded systems have become very important today as they control many of the common devices we use.

Since the embedded system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product, or increasing the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically,embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

In general, "embedded system" is not an exactly defined term, as many systems have some element of programmability. For example, Handheld computers share some elements with embedded systems — such as the operating systems and microprocessors which power them — but are not truly embedded

systems, because they allow different applications to be loaded and peripherals to be connected.

An embedded system is some combination of computer hardware and software, either fixed in capability or programmable, that is specifically designed for a particular kind of application device. Industrial machines, automobiles, medical equipment, cameras, household appliances, airplanes, vending machines, and toys (as well as the more obvious cellular phone and PDA) are among the myriad possible hosts of an embedded system. Embedded systems that are programmable are provided with a programming interface, and embedded systems programming is a specialized occupation.

Certain operating systems or language platforms are tailored for the embedded market, such as Embedded Java and Windows XP Embedded. However, some low-end consumer products use very inexpensive microprocessors and limited storage, with the application and operating system both part of a single program. The program is written permanently into the system's memory in this case, rather than being loaded into RAM (random access memory), as programs on a personal computer are.

Applications of embedded system:

We are living in the Embedded World. You are surrounded with many embedded products and your daily life largely depends on the proper functioning of these gadgets. Television, Radio, CD player of your living room, Washing Machine or Microwave Oven in your kitchen, Card readers, Access Controllers, Palm devices of your work space enable you to do many of your tasks very effectively. Apart from all these, many controllers embedded in your car take care

of car operations between the bumpers and most of the times you tend to ignore all these controllers.

4.3 USB Programmer

The USB interface is achieved by using an atmega8 processor and the rest is done in firmware.

Some of the features include:

Allows you to read or write the microcontroller EEPROM, firmware, fuse bits and lock bits
Support for Windows, Mac OS X and Linux (will work on Windows 8.1)
5 KB/sec maximum write speed
Software controlled SCK option to support targets with low clock speed (< 1.5MHz)
10 pin ISP interface (conforms to standard ISP 10-pin pinout)
The latest Window Drivers are fully signed, so you can use them on Windows Vista and above without any issues. The driver will work on both 32 and 64 bit platforms. On Linux and Mac OS X no kernel driver is required, just use AVRdude and specify the correct port.

The programmer will work with a wide variety of Atmel AVR microcontrollers including the Atmega8a and Atmega168a. A full list is available on the specifications tab. The programmer will also work with a variety of software including AVRdude. AVRdude is available for many platforms.

4.4 Proteus

The **Proteus Design Suite** is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronic design engineers and technicians to create schematics and electronic prints for manufacturing printed circuit boards.

The Proteus Design Suite is a Windows application for schematic capture, simulation, and PCB (Printed Circuit Board) layout design. It can be purchased in many configurations, depending on the size of designs being produced and the requirements for microcontroller simulation. All PCB Design products include an autorouter and basic mixed mode SPICE simulation capabilities.

Schematic Capture

Schematic capture in the Proteus Design Suite is used for both the simulation of designs and as the design phase of a PCB layout project. It is therefore a core component and is included with all product configurations.

Microcontroller Simulation

The micro-controller simulation in Proteus works by applying either a hex file or a debug file to the microcontroller part on the schematic. It is then co-simulated along with any analog and digital electronics connected to it. This enables its use in a broad spectrum of project prototyping in areas such as motor control, temperature control and user interface design. It also finds use in the general hobbyist community and, since no hardware is required, is convenient to use as a training or teaching tool. Support is available for co-simulation of:

- Microchip Technologies PIC10, PIC12, PIC16, PIC18, PIC24, dsPIC33 microcontrollers
- Atmel AVR (and Arduino), 8051 and ARM Cortex-M3 microcontrollers
- NXP 8051, ARM7, ARM Cortex-M0 and ARM Cortex-M3 microcontrollers
- Texas Instruments MSP430, PICCOLO DSP and ARM Cortex-M3 microcontrollers
- Parallax Basic Stamp, Freescale HC11, 8086 microcontrollers

PCB Design

The PCB Layout module is automatically given connectivity information in the form of a netlist from the schematic capture module. It applies this information, together with the user specified design rules and various design automation tools, to assist with error free board design. PCB's of up to 16 copper layers can be produced with design size limited by product configuration.

3D Verification

The 3D Viewer module allows the board under development to be viewed in 3D together with a semi-transparent height plane that represents the boards enclosure. STEP output can then be used to transfer to mechanical CAD software such as Solidworks or Autodesk for accurate mounting and positioning of the board.

4.5 Program

```
#include <LiquidCrystal.h>

LiquidCrystallcd(7, 6, 5, 4,3,2);


void setup()

{

Serial.begin(9600);

lcd.begin(16, 2);




pinMode(A0,INPUT);

pinMode(A1,INPUT);
```

```
pinMode(A2,INPUT);
pinMode(8,INPUT);
pinMode(11,OUTPUT);

lcd.setCursor(0, 0);
lcd.print("Air & Sound Poll");
lcd.setCursor(0, 1);
lcd.print(" Monitor System ");
delay(3000);
lcd.clear();
 }

void loop()
{

 int sensorValue1 = 0;
  sensorValue1 = analogRead(A0);
delay(10);
  sensorValue1 = analogRead(A0);
delay(10);
  sensorValue1 = analogRead(A0);
delay(10);
```

```
Serial.println("AQI:");
Serial.println(sensorValue1);
lcd.setCursor(0, 0);
lcd.print("Temp:");
lcd.print(sensorValue1/2);
lcd.print("  ");
delay(2000);

 int sensorValue2 = 0;
  sensorValue2 = analogRead(A1);
delay(10);

Serial.println("Pol:");
Serial.println(sensorValue2);
lcd.setCursor(9, 0);
lcd.print("CO:");
lcd.print(sensorValue2-70);
lcd.print("     ");
delay(2000);

int sensorValue3 = 0;
  sensorValue3 = analogRead(A2);
delay(10);
```

```
  sensorValue3 = analogRead(A2);
delay(10);
  sensorValue3 = analogRead(A2);
delay(10);

Serial.println("Pol:");
Serial.println(sensorValue3);
lcd.setCursor(0, 1);
lcd.print("Smoke:");
lcd.print(sensorValue3-70);
lcd.print(" ");
delay(2000);

   if(digitalRead(8)==LOW)
   {
digitalWrite(11, HIGH);
lcd.setCursor(10, 1);
lcd.print("S: ON ");
delay(2000);
   }
   if(digitalRead(8)==HIGH)
   {
digitalWrite(11, LOW);
```

```
lcd.setCursor(10, 1);
lcd.print("S: OFF");
delay(2000);
   }
   if((sensorValue1/2)>=35)
   {
digitalWrite(11, HIGH);
lcd.setCursor(0, 1);
lcd.print("High Temperature        ");
   }
    if((sensorValue2-60)>=150)
   {
digitalWrite(11, HIGH);
lcd.setCursor(0, 1);
lcd.print("CO Level Exceeded        ");
   }
   if((sensorValue3-70)>=250)
   {
digitalWrite(11, HIGH);
lcd.setCursor(0, 1);
lcd.print("High Smoke          ");
   }
}
```

CHAPTER 5

5.1 Applications

- To estimate the pollution
- Indoor air quality monitoring
- To make data available for common man
- Industrial perimeter monitoring
- To set a danger limit on that server and inform authorities to take future actions for well being
- At small level ,in schools and colleges we can use this device
- Activities like shooting, open air events, football and cricket matches

5.2 Advantages

- Sensors are easily available
- Sensors are effortlessly accessible
- Detecting of wide range of gases
- Simple device
- Compact device
- It is easy to handle
- Sensors have long life time
- Low cost
- Data can be used to control pollution

5.3 Limitations

Can be used for short distances only

Only 3 sensors are used

Humidity should be less than 95%

Accurate measure of contaminant gases cannot be detected in ppm

5.4 Future scope

In the future we modify the system to notify a user about the air quality and noise level it reaches beyond permissible level through sms or app

GPS can be used and we can monitor the air and sound pollution level at any place

CHAPTER 6

CONCLUSION

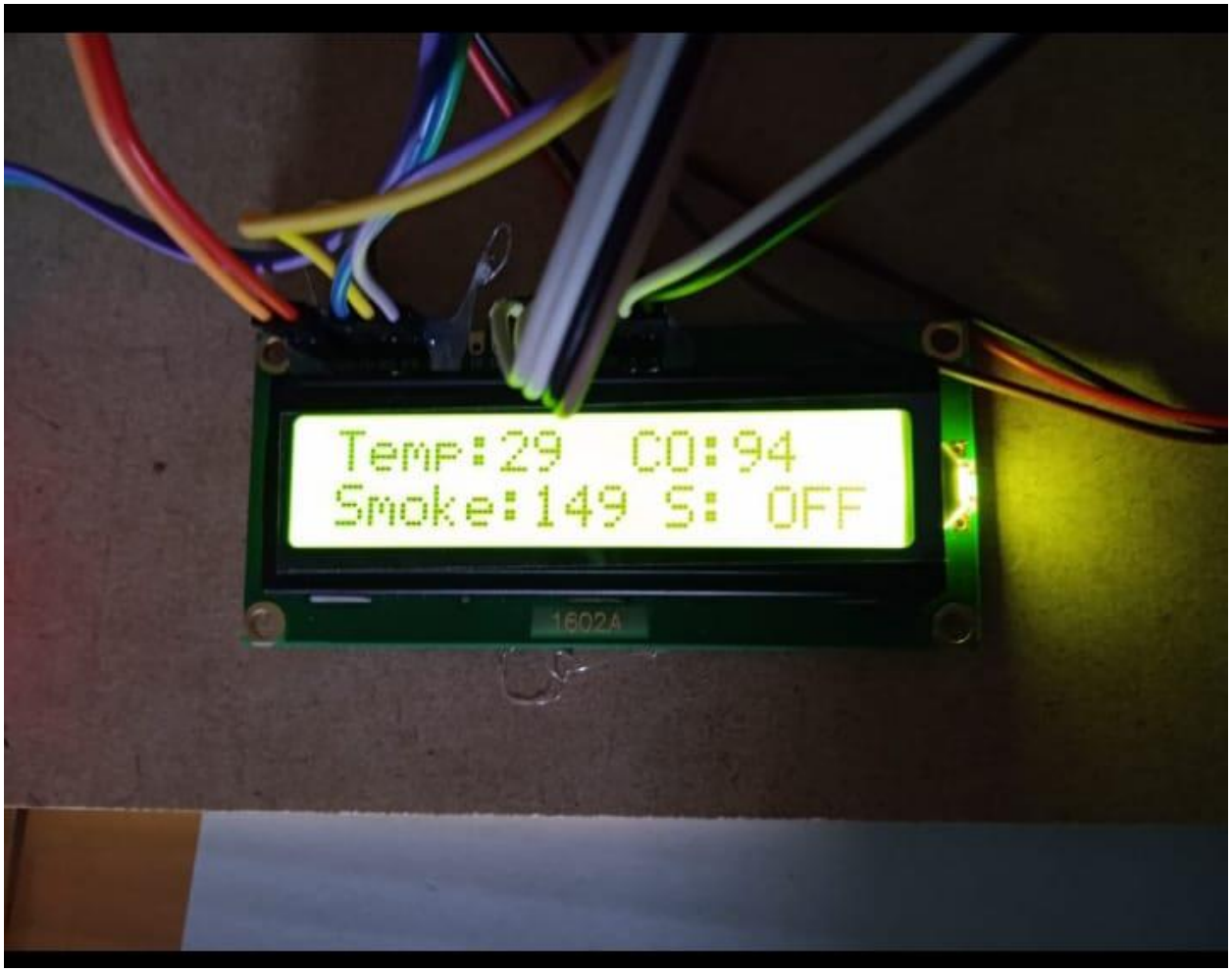
The concept of detecting the level of Pollution and indicating it is implemented. There is an increase in the level of Pollution over the last couple of decades, leading to several Environmental problems. There will be a huge population, who do not take care of the pollution from their vehicles seriously, which has already resulted in several environmental problems such as Ozone layer depletion and so on. Hence this system will be highly beneficial in curbing this problem.

By using this project each and every variation we can analyze and inform nearby people in time. We can also analyze data from home using things peak.

The most important factor of this system is small, cost efficient and portable. This system is fully helpful to save lives and overcome all environment related problem

RESULT

The air and sound pollution monitoring system is designed to monitor the air quality using co sensor, temperature sensor, sound sensor, smoke sensor. The result will be displayed on LCD and if the value exceeds the threshold value buzzer gets ON.



REFERENCES

https://www.ijert.org/air-and-sound-pollution-monitoring-system-using-cloud-computing

https://www.researchgate.net/publication/353287494

https://www.twinschip.com/USB_programmer_Atmel_AVR_Controllers

https://www.europarl.europa.eu/factsheets/en/sheet/75/air-and-noise-pollution

GUIDE HELP

BOOKS

INTERNET

PROJECT WORK TEAM BIODATA

S.NO	STUDENT NAME	PIN NO	ADDRESS	CONTACT NO.	E-MAIL
1	A.BINDU	19001-EC-203	2-2-128/38, Vijaypuri colony, Uppal,hyd	6302333020	binduanchula@gmail.com
2	K.LIKHITHA	19001-EC-219	2-3-512/134/1/b, Bapunagar, Amberpet	7396634726	likhithareddy185@gmail.com
3	P.SWETHA	19001-EC-233	3-14-36/7/30/1 Viveknagar,Ramanthapur	9100523787	paakaswetha@gmail.com
4	M.RUTHIKA	19001-EC-244	42-13, AnjaiahNagar, Jagadgiri gutta, Hyderabad	9059214466	mamidalaruthika2004@gmail.com
5	C.PARAMESHWARI	19001-EC-246	4-1-522\A,MIG colony, Vikarabad	6309132855	Parameshwaripandu06@gmail.com