**Major Components :**

1.Ultrasonic sonar sensor(HC-SR04)

2.Aurdino Uno

3.Serial monitoring

4.IDE

5.Servo MG 996R

**Description of each component is given below:**

***Ultrasonic Sonar sensor(HC-SR04):***

Ultrasonic sonar sensor is defined as that measures the distance by using ultrasonic wave . The sensor head emits an ultrasonic wave and receives the waves reflected back from the target .Ultrasonic sensor measures the distance to the target by measuring the time between the emission and reception .It has four pins for Ground ,Trigger ,Echo and VCC.

**Pin Configuration:**

1. VCC: The Vcc pin powers the sensor, typically with +5V

2. Trigger: Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.

3.Echo: Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.

4.Ground: This pin is connected to the Ground of the system.

**Features of HC-SR04 Ultrasonic sensor:**

* Operating voltage: +5V
* Theoretical  Measuring Distance: 2cm to 450cm
* Practical Measuring Distance: 2cm to 80cm
* Accuracy: 3mm
* Measuring angle covered: <15°
* Operating Current: <15mA
* Operating Frequency: 40Hz

**Working of HC-SR04 Ultrasonic Sensor:**

**HC-SR04 Ultrasonic (US) sensor** is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple high school formula that

**Distance = Speed × Time**

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module .

**How to use HC-SR04 Ultrasonic sonar sensor:**

**HC-SR04 distance sensor** is commonly used with both microcontroller and microprocessor platforms like Arduino, ARM, PIC, Raspberry Pie etc. The following guide is universally since it has to be followed irrespective of the type of computational device used.

  Power the Sensor using a regulated +5V through the Vcc ad Ground pins of the sensor. The current consumed by the sensor is less than 15mA and hence can be directly powered by the on board 5V pins (If available). The Trigger and the Echo pins are both I/O pins and hence they can be connected to I/O pins of the microcontroller. To start the measurement, the trigger pin has to be made high for 10uS and then turned off. This action will trigger an ultrasonic wave at frequency of 40Hz from the transmitter and the receiver will wait for the wave to return. Once the wave is returned after it getting reflected by any object the Echo pin goes high for a particular amount of time which will be equal to the time taken for the wave to return back to the sensor.

The amount of time during which the Echo pin stays high is measured by the MCU/MPU as it gives the information about the time taken for the wave to return back to the Sensor.

**Some application of HC- SR04 sonar sensor:**

* Used to avoid and detect obstacles with robots like biped robot, obstacle avoider robot, path finding robot etc.
* Used to measure the distance within a wide range of 2cm to 400cm
* Can be used to map the objects surrounding the sensor by rotating it
* Depth of certain places like wells, pits etc can be measured since the waves can penetrate through water

#**Aurdino Uno:**

The Arduino UNO is an open-source microcontroller board based on the [Microchip](https://en.wikipedia.org/wiki/Microchip_Technology) [ATmega328P](https://en.wikipedia.org/wiki/ATmega328P) microcontroller and developed by [Arduino.cc](https://en.wikipedia.org/wiki/Arduino). The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.[[1]](https://en.wikipedia.org/wiki/Arduino_Uno#cite_note-Makerspace-1) The board has 14 Digital pins, 6 Analog pins, and programmable with the [Arduino IDE](https://en.wikipedia.org/wiki/Arduino#Software) (Integrated Development Environment) via a type B USB cable. It can be powered by a USB cable or by an external 9 volt battery, though it accepts voltages between 7 and 20 volts. It is also similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a [Creative Commons](https://en.wikipedia.org/wiki/Creative_Commons) Attribution Share-Alike 2.5 license and is available on the Arduino website. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases.[The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform. The ATmega328 on the Arduino Uno comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware program.

**Technical Specifcation of Aurdino Uno:**

* [Microcontroller](https://en.wikipedia.org/wiki/Microcontroller): [Microchip](https://en.wikipedia.org/wiki/Microchip_Technology) [ATmega328P](https://en.wikipedia.org/wiki/ATmega328P)
* Operating Voltage: 5 Volt
* Input Voltage: 7 to 20 Volts
* Digital I/O Pins: 14 (of which 6 provide PWM output)
* Analog Input Pins: 6
* DC Current per I/O Pin: 20 mA
* DC Current for 3.3V Pin: 50 mA
* [Flash Memory](https://en.wikipedia.org/wiki/Flash_Memory): 32 KB of which 0.5 KB used by [bootloader](https://en.wikipedia.org/wiki/Booting#BOOT-LOADER)
* [SRAM](https://en.wikipedia.org/wiki/Static_random-access_memory): 2 KB
* [EEPROM](https://en.wikipedia.org/wiki/EEPROM): 1 KB
* Clock Speed: 16 MHz
* Length: 68.6 mm
* Width: 53.4 mm
* Weight: 25 g

**General pin Functions:**

* **LED**: There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
* **VIN**: The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
* **5V**: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.
* **3V3**: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
* **GND**: Ground pins.
* **IOREF**: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.
* **Reset**: Typically used to add a reset button to shields which block the one on the board.

**Special pin Functions:**

* **Serial**: pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
* **External Interrupts**: pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
* **PWM**(**P**ulse **W**idth **M**odulation) 3, 5, 6, 9, 10, and 11 Can provide 8-bit PWM output with the analogWrite() function.
* **SPI**(**S**erial **P**eripheral **I**nterface): 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
* **TWI**(**T**wo **W**ire **I**nterface): A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.
* **AREF**(**A**nalog REFerence): Reference voltage for the analog input.

**Automatic Software Reset:**

Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed tdata (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after the connection is opened.

**Pin description of Aurdino Uno:**

|  |  |  |
| --- | --- | --- |
| **Pin Category** | **Pin Name** | **Details** |
| **Power** | **Vin, 3.3V, 5V, GND** | **Vin: Input voltage to Arduino when using an external power source.**  **5V: Regulated power supply used to power microcontroller and other components on the board.**  **3.3V: 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA.**  **GND: ground pins.** |
| **Reset** | **Reset** | **Resets the microcontroller.** |
| **Analog Pins** | **A0 – A5** | **Used to provide analog input in the range of 0-5V** |
| **Input/Output Pins** | **Digital Pins 0 - 13** | **Can be used as input or output pins.** |
| **Serial** | **0(Rx), 1(Tx)** | **Used to receive and transmit TTL serial data.** |
| **External Interrupts** | **2, 3** | **To trigger an interrupt.** |
| **PWM** | **3, 5, 6, 9, 11** | **Provides 8-bit PWM output.** |
| **SPI** | **10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)** | **Used for SPI communication.** |
| **Inbuilt LED** | **13** | **To turn on the inbuilt LED.** |
| **TWI** | **A4 (SDA), A5 (SCA)** | **Used for TWI communication.** |
| **AREF** | **AREF** | **To provide reference voltage for input voltage** |

**Programming of Aurdino Uno:**

Once arduino IDE is installed on the computer, connect the board with computer using USB cable. Now open the arduino IDE and choose the correct board by selecting Tools>Boards>Arduino/Genuino Uno, and choose the correct Port by selecting Tools>Port. Arduino Uno is programmed using Arduino programming language based on Wiring. To get it started with Arduino Uno board and blink the built-in LED, load the example code by selecting Files>Examples>Basics>Blink. Once the example code (also shown below) is loaded into your IDE, click on the ‘upload’ button given on the top bar. Once the upload is finished, you should see the Arduino’s built-in LED blinking.  Below is the example code for blinking

// the setup function runs once when you press reset or power the board

void setup() {

// initialize digital pin LED\_BUILTIN as an output.

pinMode(LED\_BUILTIN, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)

delay(1000); // wait for a second

digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW

delay(1000); // wait for a second

}

\*\*\*Finally added a picture of aurdino uno and ultrasonic sonar sensor with proper pin configuration.

**Serial Monitoring :**

**Serial Monitor** is an invaluable tool for all software and hardware developers working with serial ports. The full set of features and maximum product functionality will save a lot of your time while working with serial port data and protocols.

Serial Port Monitor allows you to capture, display, analyze, record and replay all serial port data exchanged between the Windows application and the serial device. It can be successfully used in application development, device driver or serial hardware development and offers the powerful platform for effective coding, testing and optimization.

It supports custom protocols. You may use this rs232 analyzer as serial port sniffer for capturing and parsing any serial protocol data packets according to built-in or your own protocol definitions.

This nonintrusive software serial protocol analyzer, which is developed by professionals in the area of serial port communications, will help you to save time, money and precious nerves, as it already helped [our partners](https://www.hhdsoftware.com/Company/partners-and-customers).

**Step of using serial monitor:**

*Step 1*

TO use the Serial Monitor to debug Arduino Software Sketches or to view data sent by a working Sketch. We must have an Arduino connected by USB to our computer to be able to activate the Serial Monitor

*Step 2*

*Look at the serial monitor window:*

* - The small upper box is where you can type in characters (hit or click "Send")
* - The larger area (Corner can be dragged to enlarge) is where characters sent From Arduino will be displayed.- At the bottom are two pulldowns:
* - One sets the "line ending" that will be sent to Arduino when you or click Send
* - The other sets the Baud Rate for communications. (If this does not match the value set up in your sketch in Setup, characters will be unreadable). Example: Serial.begin(9600); Some sketches or other applications may use a different Baud Rate.

*Step 3*

If we are testing a new sketch we may need to know what's happening when we try to run it. But**"Software Is Invisible ! "**. So we need the tell the software to tell we what it's doing, and sometimes the value of changing variables. we do this my using the Serial Monitor and adding code to our sketch to send characters that we can see.

**SETUP:**  
In Setup you need to begin Serial Communications and set the Baud Rate (speed) that data will be transferred at.

**LOOP:**  
Here you can print helpful info to the Serial Monitor.

IDE:

Integrated development environments are designed to maximize programmer productivity by providing tight-knit components with similar [user interfaces](https://en.wikipedia.org/wiki/User_interface). IDEs present a single program in which all development is done. This program typically provides many features for authoring, modifying, compiling, deploying and debugging software. This contrasts with software development using unrelated tools, such as [vi](https://en.wikipedia.org/wiki/Vi), [GCC](https://en.wikipedia.org/wiki/GNU_Compiler_Collection) or [make](https://en.wikipedia.org/wiki/Make_(software)).

One aim of the IDE is to reduce the configuration necessary to piece together multiple development utilities, instead providing the same set of capabilities as a cohesive unit. Reducing that setup time can increase developer productivity, in cases where learning to use the IDE is faster than manually integrating all of the individual tools. Tighter integration of all development tasks has the potential to improve overall productivity beyond just helping with setup tasks. For example, code can be continuously parsed while it is being edited, providing instant feedback when syntax errors are introduced.

Some IDEs are dedicated to a specific [programming language](https://en.wikipedia.org/wiki/Programming_language), allowing a feature set that most closely matches the [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm) of the language. However, there are many multiple-language IDEs.

While most modern IDEs are graphical, text-based IDEs such as [Turbo Pascal](https://en.wikipedia.org/wiki/Turbo_Pascal) were in popular use before the widespread availability of windowing systems like [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) and the [X Window System](https://en.wikipedia.org/wiki/X_Window_System)(X11). They commonly use function keys or [hotkeys](https://en.wikipedia.org/wiki/Keyboard_shortcut) to execute frequently used commands or macros.

**Servo MG 996R:**

The MG996R is essentially an upgraded version of the famous MG995 servo, and features upgraded shock-proofing and a redesigned PCB and IC control system that make it much more accurate than its predecessor.The gearing and motor have also been upgraded to improve dead bandwidth and centering.This high-torque standard servo can rotate approximately 120 degrees (60 in each direction).

Specifications:

* Operating speed: 0.17 s/60º (4.8 V), 0.14 s/60º (6 V)
* Running Current 500 mA – 900mA
* Operating voltage: 4.8 V a 7.2 V
* Stall torque: 9.4 kgf·cm (4.8 V ), 11 kgf·cm (6 V)
* Stall Current 2.5 A (6V)
* Dead band width: 5 µs
* Stable and shock proof double ball bearing design
* Temperature range: 0 ºC – 55ºC

The process of connecting MG-996R Servo motor:

**Hardware and Software Required**

* MG996R Servo motor
* Arudino Uno
* Arduino IDE(1.0.6V)

**Hardware connections**

The MG996R servo motor has 3 wire interface and the connections should made as follows:

* Orange wire-5V
* Brown wire-Ground
* Yellow wire-digital pin 9

**Program for MG996R servo motor**

After making the connections upload the program given below.As soon as the program has uploaded,the motor will start to rotate with 15ms delay.

#include <Servo.h>

Servo myservo; // create servo object to control a servo

// a maximum of eight servo objects can be created

int pos = 0; // variable to store the servo position

void setup() {

myservo.attach(9); // attaches the servo on pin 9 to the servo object

}

void loop() {

for(pos = 0; pos < 180; pos += 1)

{

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

for(pos = 180; pos>=1; pos-=1)

{

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

}

**Working principle of Aurdino Based Radar System:**

Initially, upload the code to Arduino after making the connections. We can observe the servo sweeping from 00 to 1800 and again back to 00. Since the Ultrasonic Sensor is mounted over the Servo, it will also participate in the sweeping action.

Now, open the processing application and paste the above given sketch. In the Processing Sketch, make necessary changes in the COM Port selection and replace it with the COM Port number to which your Arduino is connected to.

If we note the Processing Sketch, We have used the output display size as 1280×720 (assuming almost all computers now-a-days have a minimum resolution of 1366×768) and made calculation with respect to this resolution.

In the future, We will upload a new Processing sketch where we can enter the desired resolution (like 1920×1080) and all the calculations will be automatically adjusted to this resolution.

Now, run the sketch in the Processing and if everything goes well, a new Processing window opens up like the one shown below. A Graphical representation of the data from the Ultrasonic Sensor is represented in a Radar type display. If the Ultrasonic Sensor detects any object within its range, the same will be displayed graphically on the screen.