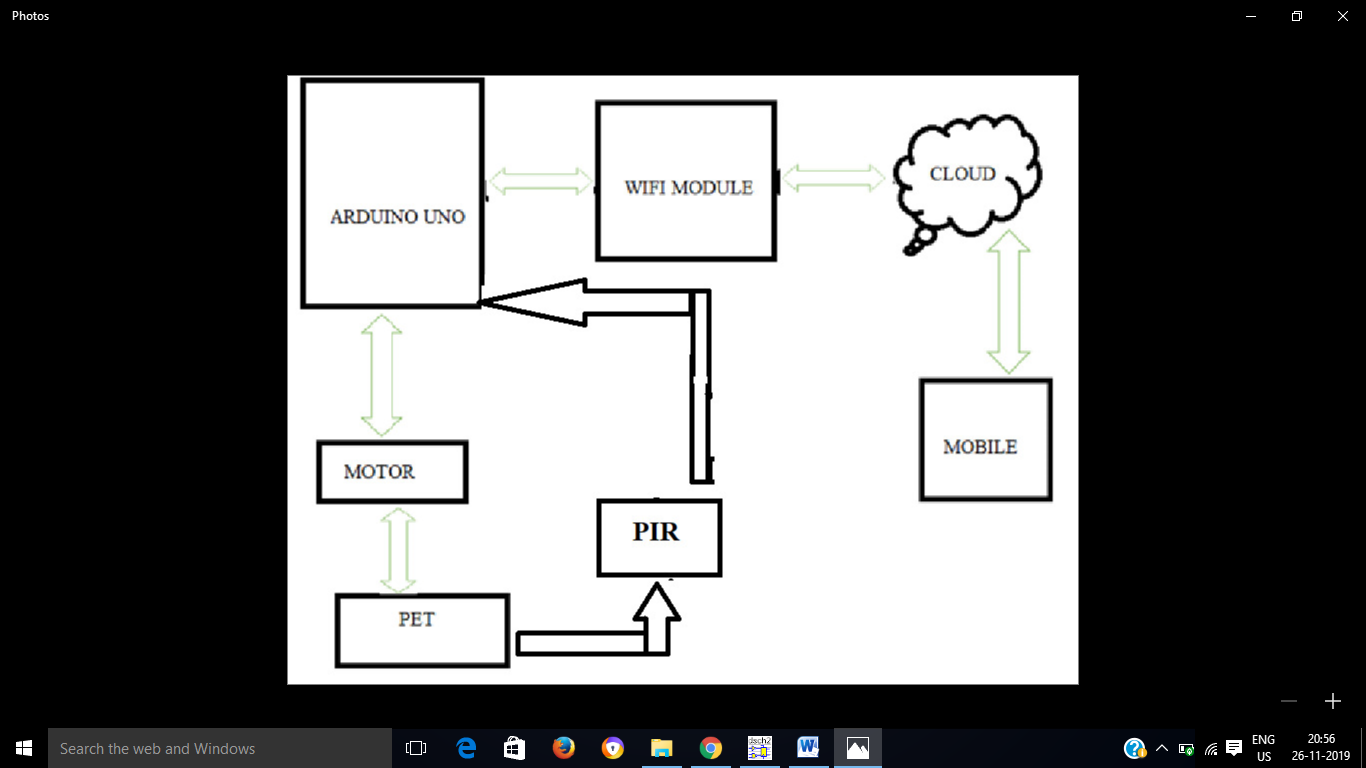
|  |  |  |
| --- | --- | --- |
| S.NO | CONTENTS | PAGE NO |
| 1 | ABSTRACT | 1 |
| 2 | INTRODUCTION | 2 |
| 3 | BLOCK DIAGRAM | 3 |
| 4 | BLOCK DIAGRAM DESCRIPTION | 4 |
| 5 | COMPONENTS USED | 5 |
| 6 | HARDWARE DESCRIPTION | 6-17 |
|  | 6.1 ARDUINO UNO | 6 |
|  | 6.2 SERVO MOTOR | 10 |
|  | 6.3 Wall Adapter Power supply | 11 |
|  | 6.4 ESP8266 WiFi Module | 12 |
|  | 6.5 RESISTORS , JUMPER WIRES AND PIR | 14-17 |
| 7 | WORKING PRINCIPLE | 18 |
| 8 | CIRCUIT DIAGRAM | 20 |
| 9 | SOFTWARE DESCRIPTION | 21-33 |
|  | 9.1 ARDUINO IDE and CLOUD | 21 |
|  | 9.2 SOFTWARE DESIGN | 31 |
| 10 | HARDWARE PHOTOGRAPH | 34 |
| 11 | APPLICATIONS | 35 |
| 12 | ADVANTAGES | 36 |
| 13 | FUTURE SCOPE | 37 |
| 14 | CONCLUSION | 38 |
| 15 | REFERENCES | 39 |

Pets need special treatment and special care. Due to nowadays busy life style, this task is not as simple as it used to be. The goal of this work is to introduce, design and implement a smart pet system. The interaction between human and physical devices and devices in the real world is gaining more attention, and re-quires a natural and intuitive methodology to employ. According to this idea and living well, life has been a growing demand. Thus, how to raise pets in an easy way has been the main issue recently. This study examines the ability of computation, communication, and control technologies to improve human interaction with pets by the technology of the Internet of Things. This work addresses the improvement through the pet application of the ability of location-awareness, and to help pet owners raise their pet on the activity and eating control easily. Our study not only presents the key improvement of the pet monitor system involved in the ideas of the Internet of Things, but also meets the demands of pet owners, who are out for works without any trouble. The objective is to allow pet owners to automate simple things, like monitoring, and feeding controls. Implementing smart pet houses will assure pets owners an increased comfort and peace of mind especially when pets are unattended.

The main aim of the project is to get knowledge of Internet of Things and developing simple projects which are useful in our daily life activities. Here, we are willing to construct an entity where it provides food to the pet accordingly which can be controlled by the owner of the pet. This project provides the owner to feed their pet from anywhere by simply connecting mobile or pc to internet.

Nowadays most of us are fascinated to have pets at their home. But these pets have to be taken care properly. Their feeding on time is an important task as they become part of our family. But in our busy schedule we fail to pay attention on our pet thus it doesn’t get proper food on time. Pet Feeding System consists food storage, servo motor, feeding bowl, etc. It also features Arduino to control the operations using internet from anywhere. It is also possible to make more hi-tech by adding cameras and audio box to check activities of pet and talk to it. Pet feeding system features a machine which can feed pets (e.g. dogs) in absence of his master. By using machine master don’t have to stay with his pet every time to feed it and he gets liberty to do his other works outside without caring about his pet.





The block diagram consits of Arduino Uno board, servo motor,mobile and a wifi module.

* We control whole operation from our mobile.
* Microcontroller is connected with wifi module (ESP 8266) in order to Feed the pet
* By connecting ESP 8266 with wifi we control mobile phone

|  |  |  |
| --- | --- | --- |
| **S.NO** | **NAME OF THE COMPONETNT** | **SPECIFICATION** |
| **1.** | ARDUINO UNO |  |
| **2.** | SERVO MOTOR |  |
| **3.** | POWER SUPPLY | WALL ADAPTER 12V |
| **4.** | WIFI MODULE | ESP 8266 |
| **5.** | RESISTORS | 1K |
| **6.** | USB CABLE |  |
| **7.** | PIR SENSOR |  |
| **8.** | PCB BOARD |  |

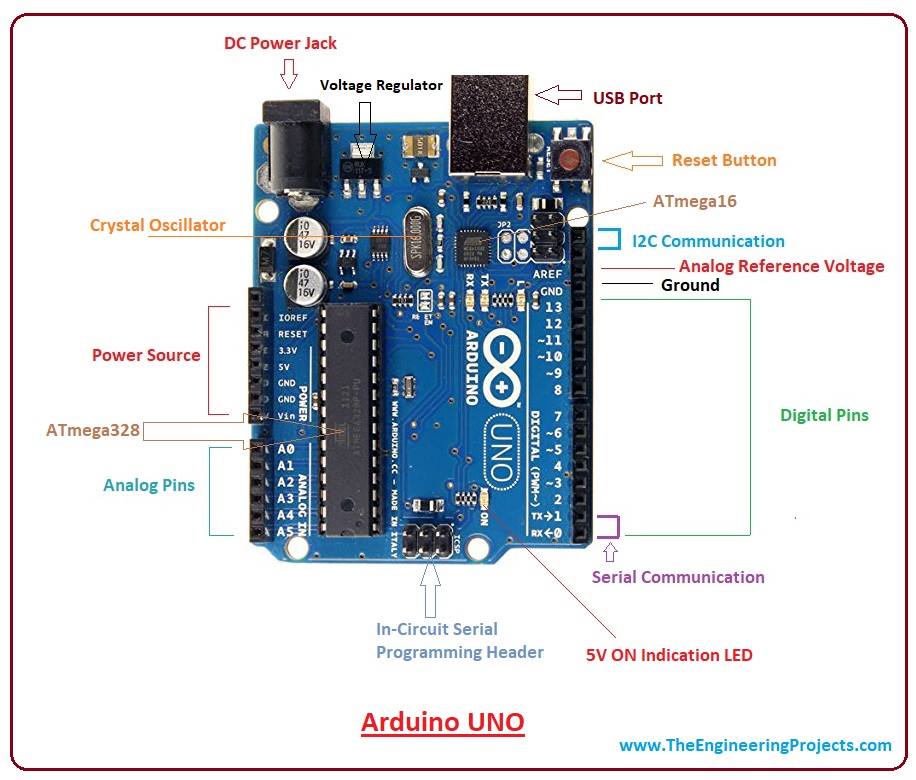
**HARDWARE DESCRIPTION**

**6.1 Arduino UNO:**

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programmed (referred to as a microcontroller) and ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

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 – BOARD DESCRIPTION:**

1) **Power USB**

Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection.

2) **Power (Barrel Jack)**

Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack.

3) **Voltage Regulator**

The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.

4) **Crystal Oscillator**

The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.

5) **Arduino Reset**

You can reset your Arduino board, i.e., starts your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button on the board. Second, you can connect an external reset button to the Arduino pin labeled RESET (5).

6) **Pins (3.3, 5, GND, Vin)**

* 3.3V : Supply 3.3 output volt
* 5V : Supply 5 output volt
* Most of the components used with Arduino board works fine with 3.3 volt and 5 volt.
* GND (Ground): There are several GND pins on the Arduino, any of which can be used to ground your circuit.
* Vin : This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.

7) **Analog pins**

The Arduino UNO board has five analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.

8) **Main microcontroller**

Each Arduino board has its own microcontroller. You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE.

9) **ICSP pin**

Mostly, ICSP is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.

10) **Power LED indicator**

This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.

11) **TX and RX LEDs**

On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led. The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process

12) **Digital I / O**

The Arduino UNO board has 14 digital I/O pins (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM.

13) **AREF**

AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Specification:

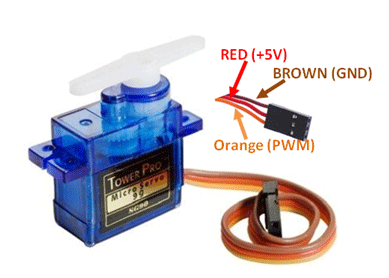
|  |  |
| --- | --- |
| Microcontroller | ATmega328 |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limits) | 6-20V |
| Digital I/O Pins | 14 (of which 6 provide PWM output) |
| Analog Input Pins | 6 |
| DC Current per I/O Pin | 40 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (ATmega328) of which 0.5 KB used by boot loader |
| SRAM | 2 KB (ATmega328) |
| EEPROM | 1 KB (ATmega328) |
| Clock Speed | 16 MHz |
| Length | 68.6 mm |

**6.2 Servo motor:**

A **servo motor** is an electrical device which can push or rotate an object with great precision. If you want to rotate and object at some specific angles or distance, then you use servo motor. It is just made up of simple motor which run through **servo mechanism**. If motor is used is DC powered then it is called DC servo motor, and if it is AC powered motor then it is called AC servo motor. We can get a very high torque servo motor in a small and light weight packages.

A servo consists of a Motor (DC or AC), a potentiometer, gear assembly and a controlling circuit. First of all we use gear assembly to reduce RPM and to increase torque of motor. Say at initial position of servo motor shaft, the position of the potentiometer knob is such that there is no electrical signal generated at the output port of the potentiometer. Now an electrical signal is given to another input terminal of the error detector amplifier. Now difference between these two signals, one comes from potentiometer and another comes from other source, will be processed in feedback mechanism and output will be provided in term of error signal. This error signal acts as the input for motor and motor starts rotating. Now motor shaft is connected with potentiometer and as motor rotates so the potentiometer and it will generate a signal. So as the potentiometer’s angular position changes, its output feedback signal changes. After sometime the position of potentiometer reaches at a position that the output of potentiometer is same as external signal provided. At this condition, there will be no output signal from the amplifier to the motor input as there is no difference between external applied signal and the signal generated at potentiometer, and in this situation motor stops rotating.

The Servo Motor basically consists of a DC Motor, a Gear system, a position sensor and a control circuit. [The DC motors get powered from a battery and run at high speed and low torque.](https://www.edgefxkits.com/four-quadrant-dc-motor-speed-control-with-microcontroller)The Gear and shaft assembly connected to the DC motors lower this speed into sufficient speed and higher torque. The position sensor senses the position of the shaft from its definite position and feeds the information to the control circuit. The control circuit accordingly decodes the signals from the position sensor and compares the actual position of the motors with the desired position and accordingly controls the direction of rotation of the DC motor to get the required position. The Servo Motor generally requires DC supply of 4.8V to 6 V.



**6.3 Wall Adapter Power supply:**

12V power supplies (or 12VDC power supplies) are one of the most common power supplies in use today. In general, a 12VDC output is obtained from a 120VAC or 240VAC input using a combination of transformers, diodes and transistors. 12V power supplies can be of two types: 12V regulated power supplies, and 12V unregulated power supplies.12V regulated power supplies come in three styles: Switching regulated AC to DC, Linear regulated AC to DC, and Switching regulated DC to DC.



**6.4 ESP8266 WiFi Module:**

This is Wi-Fi serial transceiver module, based on ESP8266 SoC., The SOC has Integrated TCP/IP protocolstack ESP8266 is a highly integrated chip designed for the needs of a new connected world. It offers a complete and self-contained Wi-Fi networking solution, allowing it to either host the application or to offload all Wi-Fi networking functions from another application processor.

ESP8266 has powerful on-board processing and storage capabilities that allow it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. Its high degree of on-chip integration allows for minimal external circuitry, and the entire solution, including front-end module, is designed to occupy minimal PCB area.

**Specification:**  
• 802.11 b/g/n

• Wi-Fi Direct (P2P), soft-AP

• Integrated TCP/IP protocol stack

• Integrated TR switch

• Integrated PLLs, regulators, DCXO and power management units

•+19.5dBm output power in 802.11b mode  
• Power down leakage current of <10Ua

• Integrated low power 32-bit CPU could be used as application processor

**Power**

* VCC-3.0-3.6V
* Standby ~ 0.9uA
* Running ~60-215mA,
* Average ~ 80Ma

**I/O Features**

* Integrated TCP/IP
* Integrated TR switch, LNA, balun

**Basic Connection**

* VCC - 3.3V
* GND - GND
* TX - RX on ARDUINO
* RX - TX on ARDUINO
* Chip Enable - 3.3V

**Default Baud Rate**

* 115200

**LEDs**

* Red: Power
* Blue: TX

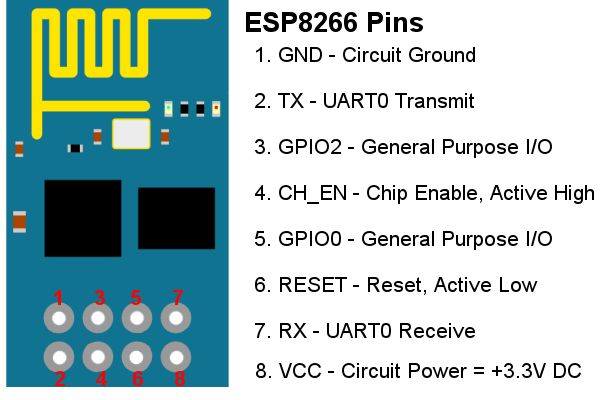
**Wi-Fi Features**

* 802.11 b/g/n
* 2.4GHz
* WPA/WPA2
* Wi-Fi Direct
* +20dBm output power (802.11b)

**Memory/Speed Features**

* 80MHz
* 64KB instruction RAM
* 96KB data RAM
* 64K boot ROM
* 1MB\* Flash Memory

**PIN ASSIGNMENT:**



**6.5 Resistor:**

A resistor is a [passive](https://en.wikipedia.org/wiki/Passivity_(engineering)) [two-terminal](https://en.wikipedia.org/wiki/Terminal_(electronics)) [electrical component](https://en.wikipedia.org/wiki/Electronic_component) that implements [electrical resistance](https://en.wikipedia.org/wiki/Electrical_resistance) as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, [bias](https://en.wikipedia.org/wiki/Biasing) active elements, and terminate [transmission lines](https://en.wikipedia.org/wiki/Transmission_line), among other uses. High-power resistors that can dissipate many [watts](https://en.wikipedia.org/wiki/Watt) of electrical power as heat, may be used as part of motor controls, in power distribution systems, or as test loads for [generators](https://en.wikipedia.org/wiki/Electric_generator). Fixed resistors have resistances that only change slightly with temperature, time or operating voltage. Variable resistors can be used to adjust circuit elements (such as a volume control or a lamp dimmer), or as sensing devices for heat, light, humidity, force, or chemical activity.

Resistors are common elements of [electrical networks](https://en.wikipedia.org/wiki/Electrical_network) and [electronic circuits](https://en.wikipedia.org/wiki/Electronic_circuit) and are ubiquitous in [electronic equipment](https://en.wikipedia.org/wiki/Electronics). Practical resistors as discrete components can be composed of various compounds and forms. Resistors are also implemented within [integrated circuits](https://en.wikipedia.org/wiki/Integrated_circuits).

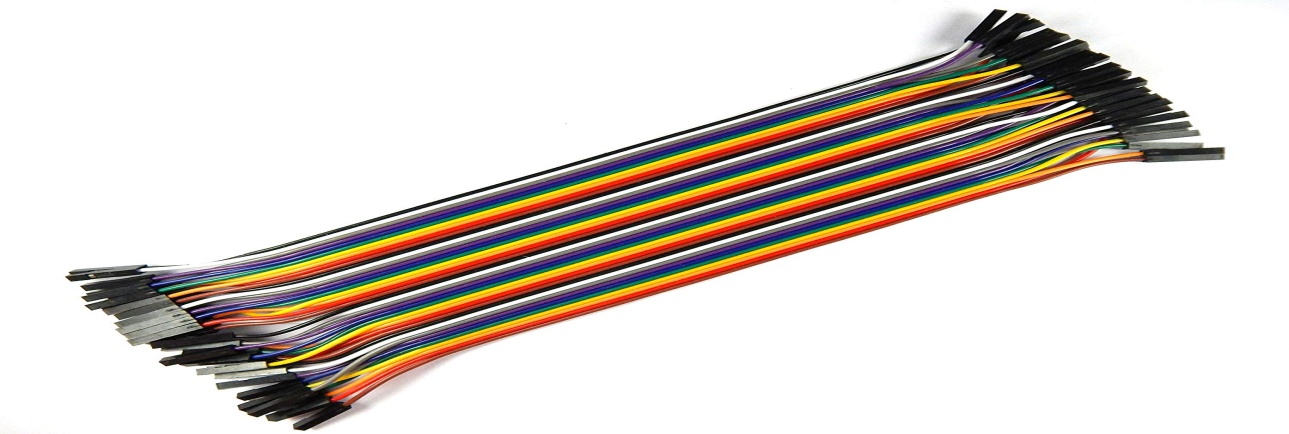
The electrical function of a resistor is specified by its resistance: common commercial resistors are manufactured over a range of more than nine [orders of magnitude](https://en.wikipedia.org/wiki/Orders_of_magnitude). The nominal value of the resistance falls within the [manufacturing tolerance](https://en.wikipedia.org/wiki/Engineering_tolerance#Electrical_component_tolerance), indicated on the component.



JUMPERWIRES:

A jump wire (also known as jumper wire, or jumper) is an [electrical wire](https://en.wikipedia.org/wiki/Electrical_wire), or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a [breadboard](https://en.wikipedia.org/wiki/Breadboard) or other prototype or test circuit, internally or with other equipment or components, without soldering.

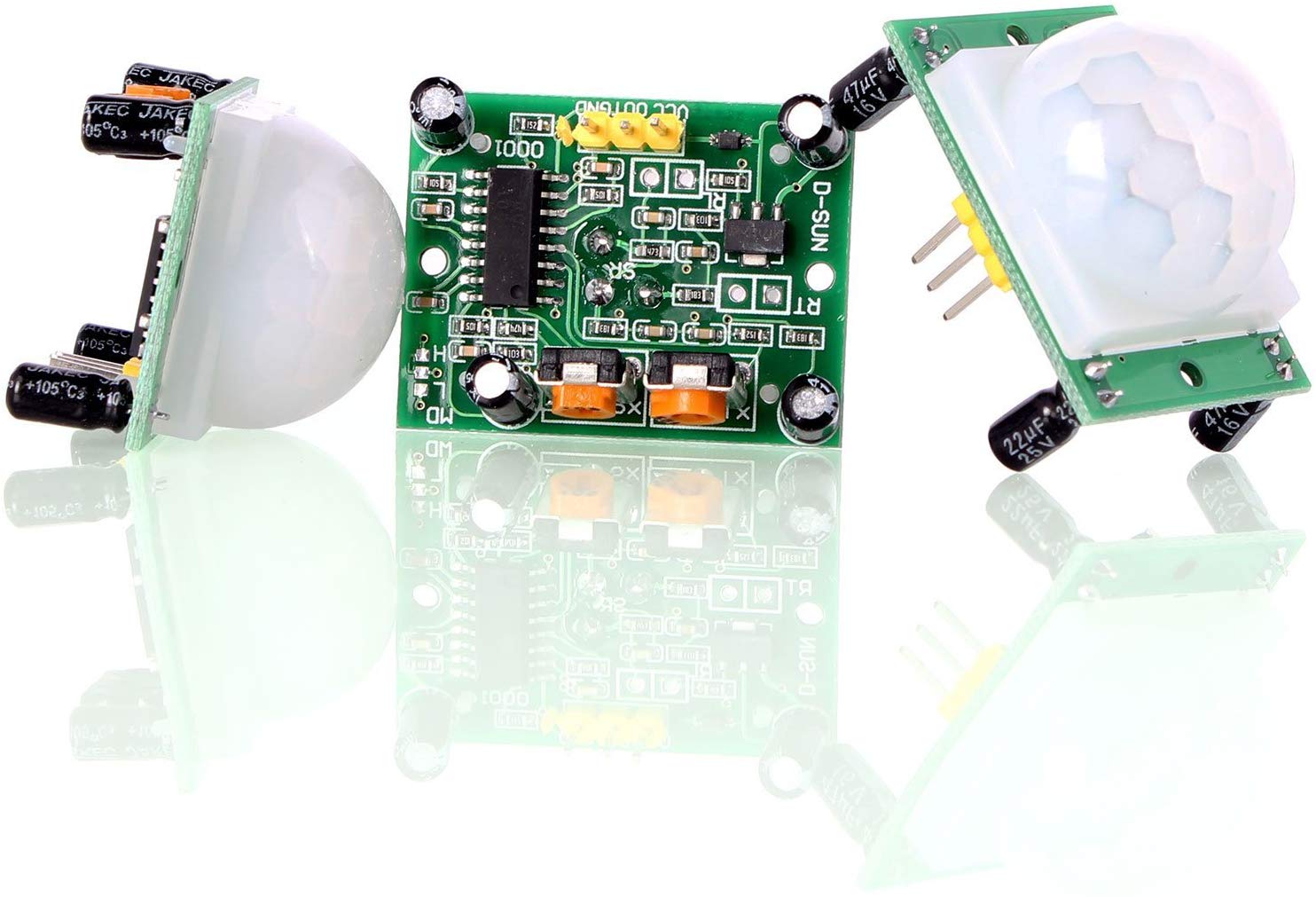
Individual jump wires are fitted by inserting their "end connectors" into the slots provided in a breadboard, the [header connector](https://en.wikipedia.org/wiki/Pin_header#Header_connector) of a circuit board, or a piece of test equipment.

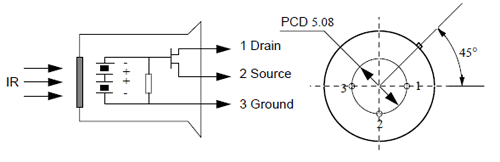


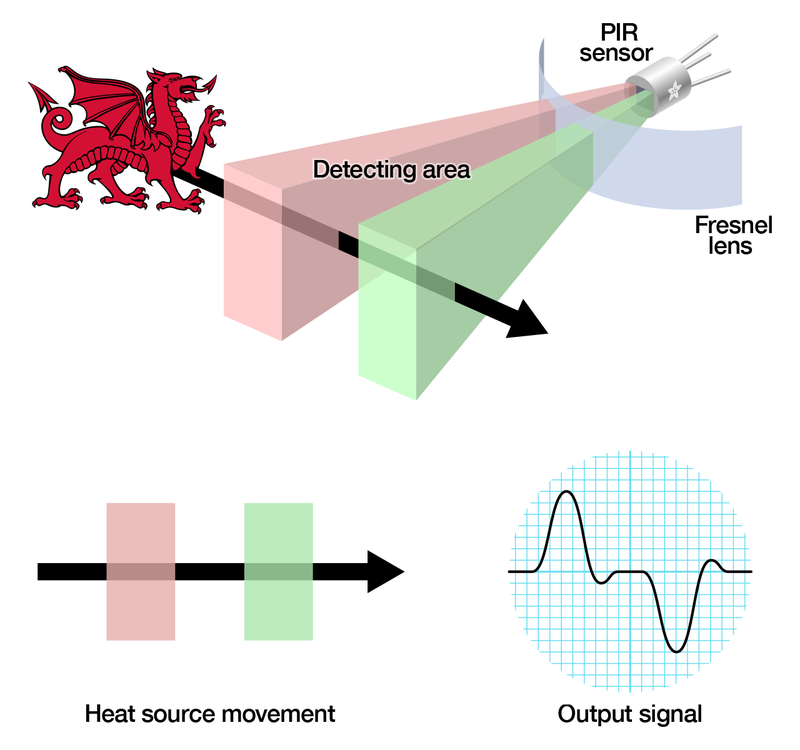
**PIR :**

PIR sensors are more complicated than many of the other sensors explained in these tutorials (like photocells, FSRs and tilt switches) because there are multiple variables that affect the sensors input and output. To begin explaining how a basic sensor works, we'll use this rather nice diagram

The PIR sensor itself has two slots in it, each slot is made of a special material that is sensitive to IR. The lens used here is not really doing much and so we see that the two slots can 'see' out past some distance (basically the sensitivity of the sensor). When the sensor is idle, both slots detect the same amount of IR, the ambient amount radiated from the room or walls or outdoors. When a warm body like a human or animal passes by, it first intercepts one half of the PIR sensor, which causes a positive differential change between the two halves. When the warm body leaves the sensing area, the reverse happens, whereby the sensor generates a negative differential change. These change pulses are what is detected.







**WORKING PRINCIPLE**

* There are many ways to implement a pet feeder: you can set it to fill up the bowl at a certain time, you can command it to fill up whenever it gets empty, or maybe to give your dog food after they follow a set of orders that you taught them.
* In addition, we also decided to add the option to control the pet feeder from our mobile phone, using a pre-defined dashboard made with Remote XY.
* First we have to connect our Wifi module to internet in-order to control using internet from anywhere.
* When a instruction given through mobile by connecting wirelessly to the hardware, according the food is provides to the pet.
* Here, wifi plays a mojor role where the food equipment is controlled using servo motor.
* We use a PIR sensor which detects the living things whenever it detects it then automatically we feed the pet.



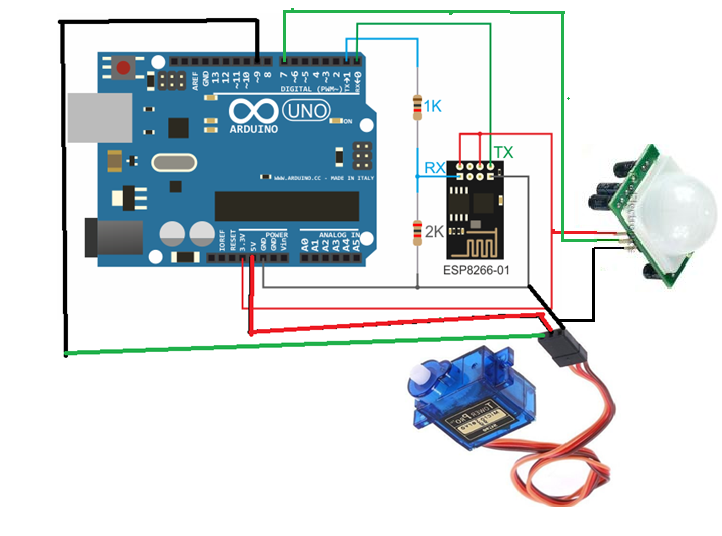
**Flow chart:**

***MOBILE***

**FEEDING**

***SERVO MOTOR***

***FEED THE PET***



**SOFTWARE DESCRIPTION**

**9.1 ARDUINO IDE:**

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

**Writing Sketches:**

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

|  |  |
| --- | --- |
| H:\recent\Arduino - Environment_files\play.png | Verify  Checks your code for errors compiling it. |
| H:\recent\Arduino - Environment_files\export.png | Upload  Compiles your code and uploads it to the configured board. See [uploading](https://www.arduino.cc/en/Guide/Environment#uploading) below for details.  Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer" |

|  |  |
| --- | --- |
| H:\recent\Arduino - Environment_files\new.png | New  Creates a new sketch. |
| H:\recent\Arduino - Environment_files\open.png | Open  Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.  Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbook menu instead. |
| H:\recent\Arduino - Environment_files\save.png | Save  Saves your sketch. |
| H:\recent\Arduino - Environment_files\serial_monitor.png | Serial Monitor  Opens the [serial monitor](https://www.arduino.cc/en/Guide/Environment#serialmonitor). |

Additional commands are found within the five menus: File, Edit, Sketch, Tools, and Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

**UPLOADING:**

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The [boards](https://www.arduino.cc/en/Guide/Environment#boards) are described below. On the Mac, the serial port is probably something like /dev/tty.usbmodem241 (for a Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty.USA19QW1b1P1.1 (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be /dev/ttyACMx, /dev/ttyUSBx or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the File menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino boot-loader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The boot-loader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The boot-loader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

**Libraries:**

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more #include statements at the top of the sketch and compiles the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #include statements from the top of your code.

There is a [list of libraries](https://www.arduino.cc/en/Reference/Libraries) in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch. See these [instructions for installing a third-party library](https://www.arduino.cc/en/Guide/Libraries).

**Third-Party Hardware:**

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, boot-loaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

For details on creating packages for third-party hardware, see the [Arduino IDE 1.5 3rd party Hardware specification](https://github.com/arduino/Arduino/wiki/Arduino-IDE-1.5-3rd-party-Hardware-specification).

**Serial Monitor:**

Displays serial data being sent from the Arduino or Genuino board (USB or serial board). To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down that matches the rate passed to Serial. begin in your sketch. Note that on Windows, Mac or Linux, the Arduino or Genuino board will reset (rerun your sketch execution to the beginning) when you connect with the serial monitor.

You can also talk to the board from Processing, Flash, MaxMSP, etc. (see the [interfacing page](http://www.arduino.cc/playground/Main/Interfacing) for details).

**Preferences:**

Some preferences can be set in the preferences dialog (found under the Arduino menu on the Mac, or File on Windows and Linux). The rest can be found in the preferences file, whose location is shown in the preference dialog.

**Language Support:**

Since version 1.0.1, the Arduino Software (IDE) has been translated into 30+ different languages. By default, the IDE loads in the language selected by your operating system. (Note: on Windows and possibly Linux, this is determined by the locale setting which controls currency and date formats, not by the language the operating system is displayed in.)

If you would like to change the language manually, start the Arduino Software (IDE) and open the Preferences window. Next to the Editor Language there is a dropdown menu of currently supported languages. Select your preferred language from the menu, and restart the software to use the selected language. If your operating system language is not supported, the Arduino Software (IDE) will default to English.

You can return the software to its default setting of selecting its language based on your operating system by selecting System Default from the Editor Language drop-down. This setting will take effect when you restart the Arduino Software (IDE). Similarly, after changing your operating system's settings, you must restart the Arduino Software (IDE) to update it to the new default language.

**Boards:**

The board selection has two effects: it sets the parameters (e.g. CPU speed and baud rate) used when compiling and uploading sketches; and sets and the file and fuse settings used by the burn bootloader command. Some of the board definitions differ only in the latter, so even if you've been uploading successfully with a particular selection you'll want to check it before burning the bootloader. You can find a comparison table between the various boards [here](https://www.arduino.cc/en/Products/Compare).

Arduino Software (IDE) includes the built in support for the boards in the following list, all based on the AVR Core. The [Boards Manager](https://www.arduino.cc/en/Guide/Cores) included in the standard installation allows to add support for the growing number of new boards based on different cores like Arduino Due, Arduino Zero, Edison, Galileo and so on.

**Cloud (remote xy):**

RemoteXY is easy way to make and use a mobile graphical user interface for controller boards to control via smartphone or tablet. The system includes:

* Editor of mobile graphical interfaces for controller boards, located on the site [remotexy.com](http://remotexy.com/ru/)
* Mobile app RemoteXY that allows to connect to the controller and control it via graphical interface. [Download app](http://remotexy.com/en/download/).

Distinctive features:

* The interface structure is stored in the controller. When connected, there is no interaction with servers to download the interface. The interface structure is downloaded to the mobile application from the controller.
* One mobile application can manage all your devices. The number of devices is not limited.

Connection between the controller and the mobile device using:

* Bluetooth;
* WiFi client and access point;
* Ethernet by IP or URL;
* Internet from anywhere through the cloud server.
* USB (Android only that support USB OTG);

The source code generator have support next controllers:

* Arduino UNO, Arduino MEGA, Arduino Leonardo, Arduino Pro Mini, Arduino Nano, Arduino MICRO;
* WeMos D1, WeMos D1 R2, WeMos D1 mini;
* NodeMCU V2, NodeMCU V3;
* The AirBoard;
* ChipKIT UNO32, ChipKIT uC32, ChipKIT Max32;

Supported communication modules:

* Bluetooth HC-05, HC-06 or compatible;
* WiFi ESP8266;
* Ethernet Shield W5100;

Supported IDE:

* Arduino IDE;
* FLProg IDE;
* MPIDE;

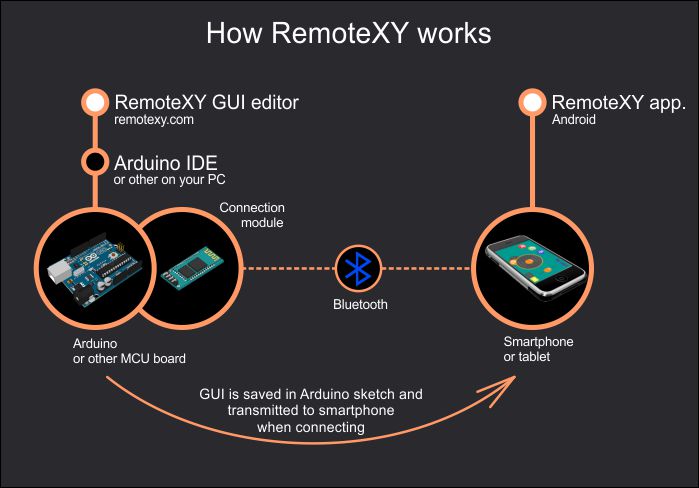
Supported mobile OS:

* Android;
* iOS;

RemoteXY is easy way to make a unique graphical interface to control microcontroller device via mobile application, Arduino for example.

RemoteXY allows:

* To develop any graphical management interface, using the control, display and decoration elements any combination thereof. You can develop the graphical interface for any task, placing the elements on the screen using the online editor. Online editor posted on the website [remotexy.com](http://remotexy.com/en/).
* After the development of the graphical interface, you get the source code for the microcontroller that implements your interface. The source code provides a structure for interaction between your program with the controls and display. Thus you can easily integrate the control system into your task for which you are developing the device.
* To manage microcontroller device using your smartphone or tablet with the graphical interface. For manage used mobile application RemoteXY.
* Using one mobile application, you can manage a large number of devices with different graphical management interfaces. As the interface description is stored on board the microcontroller device.



**9.2 Software design:**

#define REMOTEXY\_MODE\_\_ESP8266\_HARDSERIAL\_CLOUD

#include <RemoteXY.h>

#define REMOTEXY\_SERIAL Serial

#define REMOTEXY\_SERIAL\_SPEED 115200

#define REMOTEXY\_WIFI\_SSID "mahidhar"

#define REMOTEXY\_WIFI\_PASSWORD "123456789"

#define REMOTEXY\_CLOUD\_SERVER "cloud.remotexy.com"

#define REMOTEXY\_CLOUD\_PORT 6376

#define REMOTEXY\_CLOUD\_TOKEN "16774b7c8eea670926ca20529973fa2a"

#pragma pack(push, 1)

uint8\_t RemoteXY\_CONF[] =

{ 255,1,0,0,0,26,0,8,31,1,

2,1,7,33,52,22,6,24,31,31,

70,69,69,68,73,78,71,0,70,69,

69,68,0 };

struct {

uint8\_t FEED;

uint8\_t connect\_flag;

} RemoteXY;

#pragma pack(pop)

#include <Servo.h>

Servo servo;

int pos = 0;

int x=7;

void setup()

{

RemoteXY\_Init ();

pinMode(x,INPUT);

}

void loop()

{

Int PIR=0;

PIR=digitalRead(x);

RemoteXY\_Handler ();

if(RemoteXY.FEED==1 || PIR==HIGH)

{ servo.attach(9);

if(pos<120){

for (pos = 20; pos <120; pos += 1) {

servo.write(pos);

delay(30);

}

}

else

servo.write(120);

}

else{

for ( ; pos >= 20; pos -= 1) {

servo.write(pos);

delay(30);

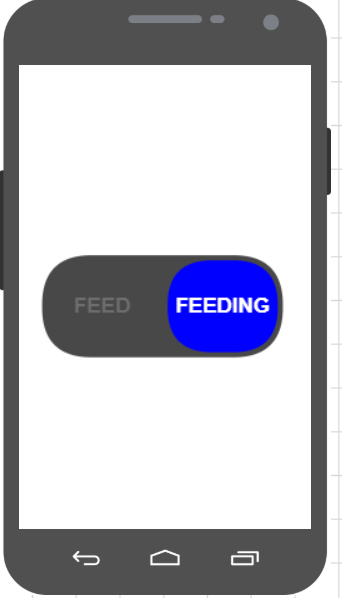
}

servo.detach();

}

}

**HARDWARE PHOTOGRAPH:**



**CONROLLING FROM MOBILE USER INTERFACE**



**PET FEEDER CONROLLING THROUGH INTERNET**

**APPLICATIONS:**

* This project can feed the dogs in kennels.
* It can play a major role in pet stores.
* It is helpful to feed K-9 dogs(Trained police dogs)
* It is used to feed animals in Zoo.
* It makes easy to feed pet animals in daily life.
* It is helpful to feed birds sanctuaries.
* This project can meet the needs of Blue Cross Society for feeding animals.

**ADVANTAGES:**

* Less Cost
* Saves time
* No need of monitoring
* Control from anywhere in the world
* Time to time feed
* Instead user feeds at particular time, user can set some particular time
* Pets can be monitored (using Arduino ATMEGA 328P) and accordingly quantity of food can be served.
* IoT is a platform which can embed both software and hardware. It is obvious from that IoT is an efficient way to access data.
* Automatic feeding can be done by connecting Android application with server and Dog Feeder through WiFi communication.
* Hence, a pet feeder equipment is build for feeding pet on time.
* It is ease to the user to feed from anyway.

**REFERENCES**

* <https://www.arduino.cc/reference/en/>
* <https://create.arduino.cc/projecthub/circuito-io-team/iot-pet-feeder-10a4f3>
* [http://remotexy.com](http://remotexy.com/)
* <https://www.esp8266.com/>
* <https://github.com/esp8266/esp8266-wiki/wiki>