***Home Automation System***

***Working of Ultrasonic Sensor – HCSR04:***

Ultrasonic sensors work by emitting sound waves at a frequency too high for humans to hear. They then wait for the sound to be reflected back, calculating distance based on the time required.

Trig (Trigger) Pin is used to trigger the ultrasonic sound pulses. This is an input pin, used to initialize measurement by transmitting ultrasonic waves by keeping this pin high for 10us.

Echo pin produces a pulse when the reflected signal is received. The length of the pulse is proportional to the time it took for the transmitted signal to be detected. This is an output pin, which goes high for a specific time period and it will be equivalent to the duration of the time for the wave to return back to the sensor.

To measure the distance the sound has travelled we use the formula: Distance = (Time x Speed Of Sound) / 2. The "2" is in the formula because the sound has to travel back and forth. First the sound travels away from the sensor, and then it bounces off of a surface and returns back.

***Working of Water Sensor – HCSR04:***

Water sensors detect the presence of water and, when placed in locations where water should not be present, a leak.

The change in resistance corresponds to the distance from the top of the sensor to the surface of the water. The resistance is inversely proportional to the height of the water. The more water the sensor is immersed in, results in better conductivity and will result in a lower resistance. The sensor produces an output voltage according to the resistance, which by measuring we can determine the water level.

***Working of Humidity|Temperature Sensor – DHT11:***

DHT11 is a low-cost digital sensor for sensing temperature and humidity. DHT11 sensor consists of a capacitive humidity sensing element and a thermistor for sensing temperature.

The temperature range of DHT11 is from 0 to 50 degree Celsius with a 2-degree accuracy. Humidity range of this sensor is from 20 to 80% with 5% accuracy.

It has three pins and is mounted to a small PCB. The PCB mounted version is nice because it includes a surface mounted 10K Ohm pull up resistor for the signal line.

The issue that was coming was that it was not determining the dht type and some issues with dht libraries on Arduino IDE too.

***Working of Radio Frequency Identification - RFID:***

RFID or [Radio Frequency Identification](https://en.wikipedia.org/wiki/Radio-frequency_identification) system consists of two main components, a transponder/tag attached to an object to be identified, and a Transceiver also known as interrogator/Reader.

A Reader consists of a Radio Frequency module and an antenna which generates high frequency electromagnetic field. On the other hand, the tag is usually a passive device, meaning it doesn’t contain a battery. Instead it contains a microchip that stores and processes information, and an antenna to receive and transmit a signal.

For example, only the person with the right information on his card is allowed to enter. An RFID system uses:

* **tags** attached to the object to be identified, in this example we have a keychain and an electromagnetic card. Each tag has his own identification (UID).
* two-way radio transmitter-receiver, the**reader,**that send a signal to the tag and read its response.

**Serial Peripheral Interface (SPI)** is an interface bus commonly used to send data between microcontrollers and small peripherals such as shift registers, sensors.

The RC522 module has total 8 pins that interface it to the outside world.

* **MISO** pin and **MOSI** pin is used for SPI communication.
* **SCK**: Serial Clock pin – used to provide clock source.
* **SS**: Acts as Serial input for SPI communication,

***Working of Light Dependent Resistor – LDR:***

An LDR is a component that has a (variable) resistance that changes with the light intensity that falls upon it.

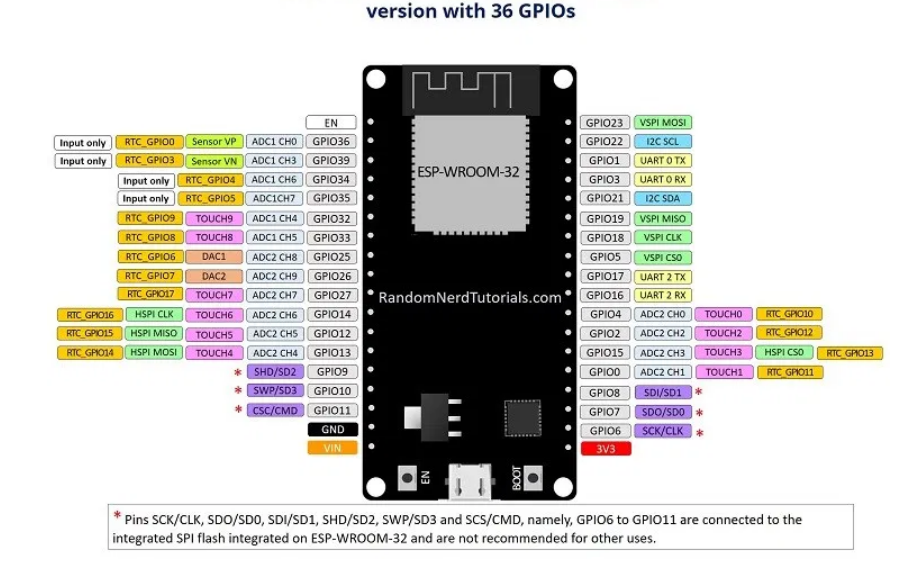
When the light level decreases, the resistance of the LDR increases. As this resistance increases in relation to the other Resistor, which has a fixed resistance, it causes the voltage dropped across the LDR to also increase.

This resistor works on the principle of **photo conductivity**. It is nothing but, when the light falls on its surface, then the material conductivity reduces and also the electrons in the valence band of the device are excited to the conduction band. These photons in the incident light must have energy greater than the band gap of the semiconductor material. This makes the electrons to jump from the valence band to conduction.

***Hardware***

***Wiring and LEDs (Inputs & Outputs)***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sensors** | **Pin Configuration** | **Cable Color** | | **Output** |
| **With Esp32** | **With sensors** |  |
| **Humidity/**  **Temperature** | Data Pin: 15 | Data Pin: Purple | Data Pin: Brown | **Humidity**  Pin 33  LED: Red  **Temperature**  Pin 32  LED: Green |
| **Water Level** | Power Pin: 17 (TX2)- UART  Signal Pin: 36 (INPUT ONLY) | Power Pin: Yellow  Signal Pin: Green | GND: Brown  Power Pin: Red  Signal Pin: Orange | Pin 5  LED: Yellow |
| **LDR** | Terminal 1: Vcc  Terminal 2: Pin 27  (10ohms Resistor) | Terminal 2: Green | Terminal 2: Green | Pin 26  LED: Red |
| **Ultrasonic** | Trigger Pin: 13  Echo Pin: 12 | Trigger: Yellow  Echo: Green | VCC: Yellow  Trigger: Green  Echo: Blue  GND: Purple | Pin 25  Buzzer |
| **RFID** |  |  |  |  |



* 34 Programmable GPIOs
* 18 12-bit ADC Channels
* 2 8-bit DAC Channels
* 16 PWM Channels
* 3 UART Interfaces
* 3 SPI Interfaces
* 2 I2C Interfaces
* 2 I2S Interfaces
* 10 Capacitive Touch Sensing GPIOs
* 16 RTC GPIOs
* GPIOs 34 to 39 are GPIs – input only pins. These pins don’t have internal pull-up or pull-down resistors. They can’t be used as outputs, so use these pins only as inputs
* GPIO 6 to GPIO 11 are exposed in some ESP32 development boards. However, these pins are connected to the integrated SPI flash on the ESP-WROOM-32 chip and are not recommended for other uses. So, don’t use these pins in your projects:

Those internal touch sensors are connected to these GPIOs:

* T0 (GPIO 4)
* T1 (GPIO 0)
* T2 (GPIO 2)
* T3 (GPIO 15)
* T4 (GPIO 13)
* T5 (GPIO 12)
* T6 (GPIO 14)
* T7 (GPIO 27)
* T8 (GPIO 33)
* T9 (GPIO 32)

The ESP32 has 18 x 12 bits ADC input channels (while the [ESP8266 only has 1x 10 bits ADC](https://randomnerdtutorials.com/esp8266-adc-reading-analog-values-with-nodemcu/)). These are the GPIOs that can be used as ADC and respective channels:

* ADC1\_CH0 (GPIO 36)
* ADC1\_CH1 (GPIO 37)
* ADC1\_CH2 (GPIO 38)
* ADC1\_CH3 (GPIO 39)
* ADC1\_CH4 (GPIO 32)
* ADC1\_CH5 (GPIO 33)
* ADC1\_CH6 (GPIO 34)
* ADC1\_CH7 (GPIO 35)
* ADC2\_CH0 (GPIO 4)
* ADC2\_CH1 (GPIO 0)
* ADC2\_CH2 (GPIO 2)
* ADC2\_CH3 (GPIO 15)
* ADC2\_CH4 (GPIO 13)
* ADC2\_CH5 (GPIO 12)
* ADC2\_CH6 (GPIO 14)
* ADC2\_CH7 (GPIO 27)
* ADC2\_CH8 (GPIO 25)
* ADC2\_CH9 (GPIO 26)

There are 2 x 8 bits DAC channels on the ESP32 to convert digital signals into analog voltage signal outputs. These are the DAC channels:

* DAC1 (GPIO25)
* DAC2 (GPIO26)

GPIO 6 – 11 are not recommended to use as inputs or outputs.

**I2C uses 2 wires.**

HC-SR04 (Ultrasonic) uses I2C to communicate with host controller.

* Stands for **Inter-integrated-circuit** (I2C)
* It is a serial communications protocol similarly to UART. However, it is not used for PC-device communication but instead with modules and sensors.
* It is a simple, bidirectional two-wire synchronous serial bus and requires only two wires to transmit information between devices connected to the bus.
* Communication method used by sensors to transmit their data to micro-controllers.
* I2C communication protocol uses two wires to share information. One is used for the clock signal (**SCL**) and the other is used to send and receive data (**SDA**).
* SDA: GPIO 21 on esp 32
* SCL: GPIO 22 on esp 32

DHT 11 uses one-wire protocol having single data line.