An Embedded Project

On

WATER LEVEL INDICATOR USING AURDINO AND ULTRA SONIC SENSOR

By

B. Surendrakumar

Abstract:

water tank overflow is a common problem which leads to the wastage of water. though there are many solutions to it like ball valves which automatically stop the water flow once the tank gets full. but being electronics enthusiastic wouldn't you like an electronic solution for it? so here is a simple and handy that will guide you to make a circuit which will detect the water level and will raise an alarm upon getting the water tank full or a pre-set level. water level indicator is a modern way of measuring the water level using latest technologies like sensors, Arduino. the main aim of the project is to calculate the water level at any instant of time and to buzz the buzzer if the tank is filled completely. I would like to use Arduino and ultrasonic sensor to make it possible. this may be useful to conserve water and helps us not to waste water.

INTRODUCTION:

The facility requirements in many industries, farms, hostels, hotels, offices include an overhead tank for water, which is usually fed through an electric pump that is switched off when the tank is filled up and switched on when it is empty. so, the most common way of knowing when the tank is filled is by observing when it overflows the brim. depending on the type of liquid being handled, overfilling of such a tank could lead to a great liquid material loss ranging in the order of thousands of naira per week depending on the extent of such application. these losses can be prevented if the tank is monitored automatically by incorporating feedback. water level indicator using ultrasonic sensor &Arduino is an amazing and very useful project. the objective of this project is to notify the user the amount of water that is present in the overhead water tank. this project can be further enhanced to control the water level in the tank by turning it on, when the water level is low, and turning it off when the water level is high. thus, the Arduino water level indicator helps in preventing wastage of water in overhead tank. a transmitter circuit and a receiver circuit. the transmitter circuit makes use of an ultrasonic sensor to measure the water level in terms of distance. this data is sent to the receiver circuit using rf communication.

Objective:

The objective of this project is to notify the user the amount of water that is present in the overhead water tank and don’t waste water.

Apparatus:

1.aurdino board uno

2.ultra-sonic sensor

3.resistor

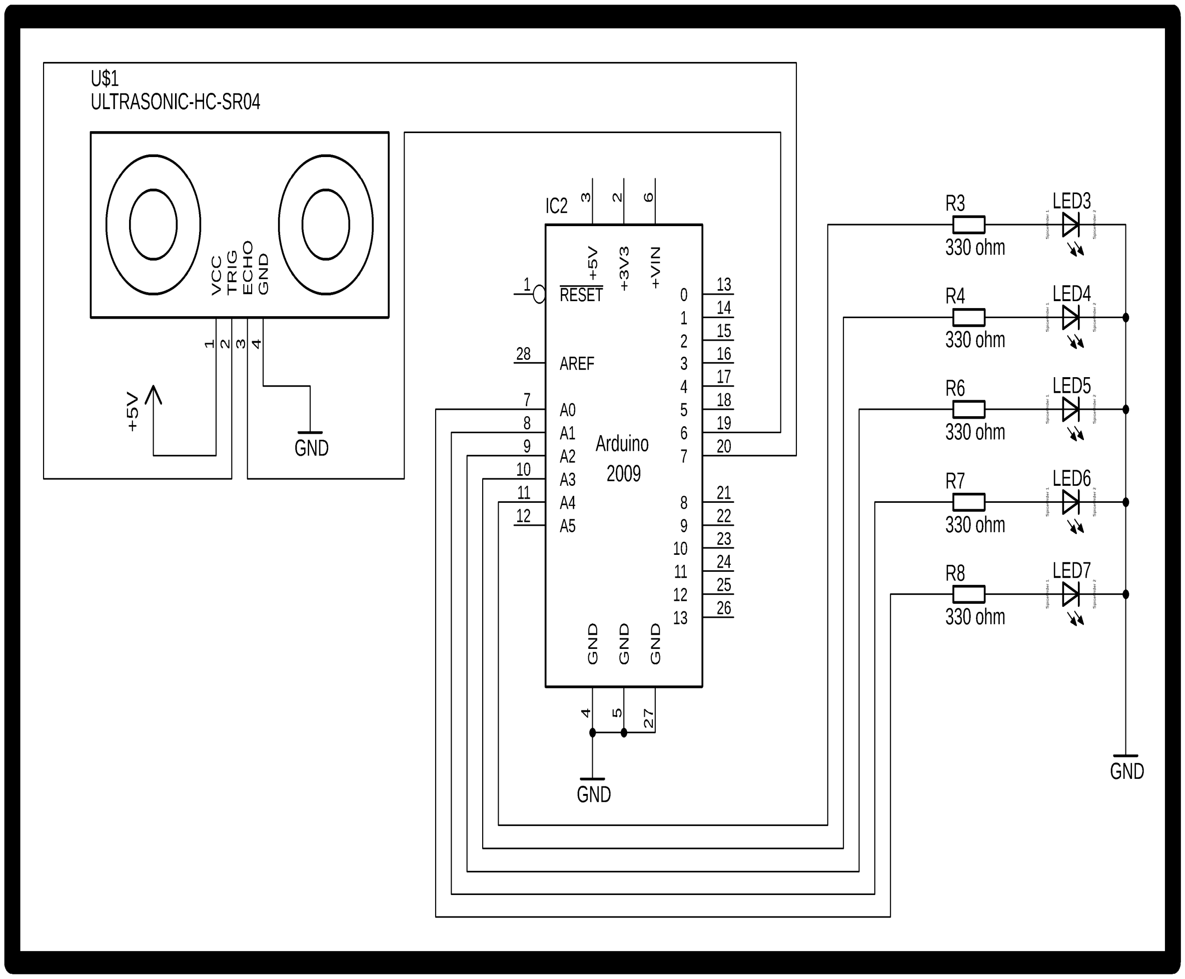
4.jumper wires

5.led

6.Water tank

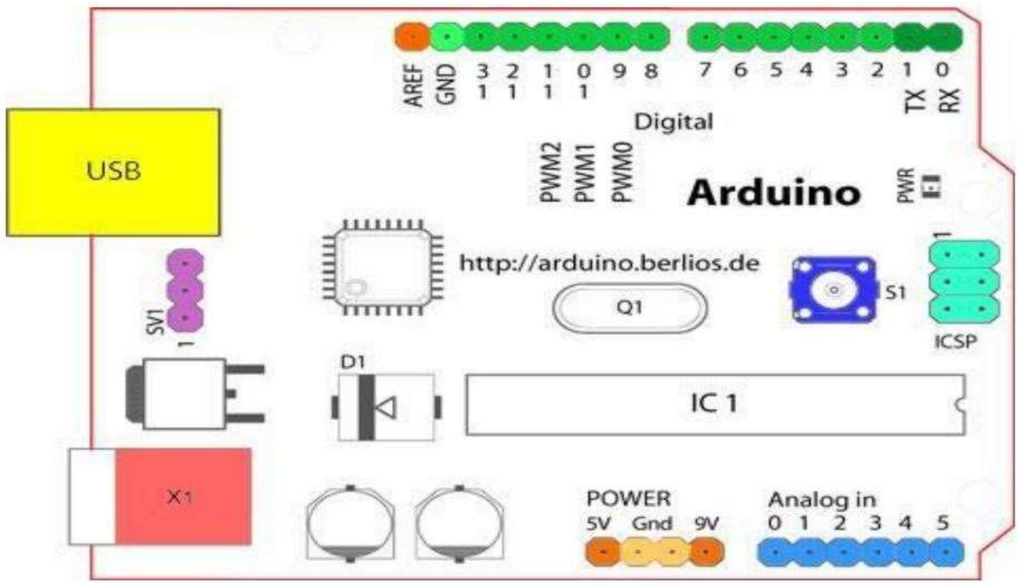
7.bread board

Block Diagram:



INTRODUCTION TO THE ARDUINO BOARD:

looking at the board from the top down, this is an outline of what you will see (parts of the board you might interact with in the course of normal use are highlighted):



STARTING CLOCKWISE FROM THE TOP CENTER: -

- analog reference pin (orange)
- digital ground (light green)
- digital pins 2-13 (green)
- digital pins 0-1/serial in/out - Tx/rx (dark green) - these pins cannot be used for digital i/o (digital read and digital write) if you are also using serial communication (e.g., serial. Begin).
- reset button - s1 (dark blue) xi
- in-circuit serial programmer (blue-green)
- analog in pins 0-5 (light blue) ϖ power and ground pins (power: orange, grounds: light orange)
- external power supply in (9-12vdc) - x1 (pink)
- toggles external power and usb power (place jumper on two pins closest to desired supply) - sv1 (purple)
- usb (used for uploading sketches to the board and for serial communication between the board and the computer; can be used to power the board) (yellow).
- **Digital Pins: -**

in addition to the specific functions listed below, the digital pins on an Arduino board can be used for general purpose input and output viathe pinmode (), digitalread (), and digital write() comm ands. each pin has an internal pull-up resistor which can be turned on and off using digital write () (w/ a value of high or low, respectively) when the pin is configured as an input. the maximum current per pin is 40 ma. . serial: 0 (rx) and 1 (tx). used to receive (rx) and transmit (tx) ttl serial data. on the Arduino decimal, these pins are connected to the corresponding pins of the ftdi usb-to-ttl serial chip. on the Arduino but, they are connected to the corresponding pins of the wt11 bluetooth module. on the arduino mini and lilypad arduino, they are intended for use with an external ttl serial module (e.g. the mini-usb adapter). • external interrupts: 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. see the attachinterrupt() function for details. • pwm: 3, 5, 6, 9, 10, and 11. provide 8-bit pwm output with the analogwrite() function. on boards with an atmega8, pwm output is available only on pins 9, 10, and 11.

Analog pins:

In addition to the specific functions listed below, the analog input pins support 10-bit analogto-digital conversion (adc) using the analogread() function. most of the analog inputs can also be used as digital pins: analog input 0 as digital pin 14 through analog input 5 as digital pin 19. analog inputs 6 and 7 (present on the mini and bt) cannot be used as digital pins.

• i 2 c: 4 (sda) and 5 (scl). support i2 c (twi) communication using the wire library (documentation on the wiring website).

POWER PINS: -

• vin (sometimes labelled "9v"). the input voltage to the arduino 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. note that different boards accept different input voltages ranges, please see the documentation for your board. also note that the lilypad has no vin pin and accepts only a regulated input.

• 5v. the regulated power supply used to power the microcontroller and other components on the board. this can come either from vin via an on-board regulator, or be supplied by usb or another regulated 5v supply.

• 3v3. (diecimila-only) a 3.3 volt supply generated by the on-board ftdi chip.

• gnd. ground pins.

OTHER PINS:-

• aref. reference voltage for the analog inputs. not currently supported by the arduino software.

• reset. (diecimila-only) bring this line low to reset the microcontroller. typically used to add a reset button to shields which block the one on the board. xiii the text of the arduino getting started guide is licensed under a creative commons attributionsharealike 3.0 license. code samples in the guide are released into the public domain.

APPLICATIONS: -

• a handheld game console based on arduino

• arduinome, a midi controller device that mimics the monome

• ardupilot, drone software and hardware

• ardusat, a cubesat based on arduino. Xiv

• c-stem studio, a platform for hands-on integrated learning of computing, science, technology, engineering, and mathematics (c-stem) with robotics.

• data loggers for scientific research.

• obduino, a trip computer that uses the on-board diagnostics interface found in most modern cars

• openevse an open-source electric vehicle charger

• xod, a visual programming language for arduino

SOFTWARE:-

a program for arduino hardware may be written in any programming language with compilers that produce binary machine code for the target processor. atmel provides a development environment for their 8-bit avr and 32-bit arm cortex-m based microcontrollers: avr studio (older) and atmel studio (newer).

Arduino UNO:

The uno is a great choice for your first arduino. it's got everything you need to get started, and nothing you don't. it has 14 digital input/output pins (of which 6 can be used as pwm outputs), 6 analog inputs, a usb connection, a power jack, a reset button and more. it contains everything needed to support the microcontroller; simply connect it to a computer with a usb cable or power it with a ac-to-dc adapter or battery to get started.

Resistor:

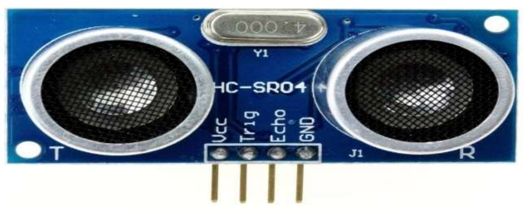
A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses. High-power resistors that can dissipate many watts of electrical power as heat, may be used as part of motor controls, in power distribution systems, or as test loads for generators. Fixed resistors have resistances that only change slightly with temperature, time or operating voltage.

Resistors are common elements of electrical networks and electronic circuits and are ubiquitous in electronic equipment. Practical resistors as discrete components can be composed of various compounds and forms. Resistors are also implemented within integrated circuits.

WATER TANK: -

A water tank is a container for storing water. Water tanks are used to provide storage of water for use in many applications, drinking water, irrigation agriculture, fire suppression, agricultural farming, both for plants and livestock, chemical manufacturing, food preparation as well as many other.

HC-SR04 Ultrasonic Sensor:



HC-SR04 SENSOR FEATURES:

- operating voltage: +5v
- theoretical measuring distance: 2cm to 450cm
- practical measuring distance: 2cm to 80cm
- accuracy: 3mm
- measuring angle covered:

APPLICATIONS:

- 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

Code:

```
#define trigpin 7
#define echopin 6
int led1 = A0;
int led2 = A1;
int led3 = A2;
int led4 = A3;
int led5 = A4;
void setup()
{
```

```
Serial.begin(9600);
pinMode(trigpin, OUTPUT);
pinMode(echopin, INPUT);
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);
  pinMode(led4, OUTPUT);
  pinMode(led5, OUTPUT);

  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, LOW);
  digitalWrite(led4, LOW);
  digitalWrite(led5, LOW);
delay(1000);
}

void loop()
{
 int duration, distance;
 digitalWrite(trigpin, HIGH);

delayMicroseconds(1000);
digitalWrite(trigpin, LOW);



duration = pulseIn(echopin,HIGH);
```

```
distance = ( duration / 2) / 29.1;
Serial.println("cm:");
Serial.println(distance);



if(  (distance > 0) && (distance <= 10)   )
{
  digitalWrite(led1, HIGH);
  digitalWrite(led2, HIGH);
  digitalWrite(led3, HIGH);
  digitalWrite(led4, HIGH);
  digitalWrite(led5, HIGH);
} else
if(  (distance > 10) && (distance <= 20)  )
{


  digitalWrite(led1, LOW);
  digitalWrite(led2, HIGH);
  digitalWrite(led3, HIGH);
  digitalWrite(led4, HIGH);
  digitalWrite(led5, HIGH);


} else

if(  (distance > 20) && (distance <= 30)  )
{
  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, HIGH);
  digitalWrite(led4, HIGH);
  digitalWrite(led5, HIGH);
} else

if(  (distance > 30) && (distance <= 40)  )
{

  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, LOW);
  digitalWrite(led4, HIGH);
  digitalWrite(led5, HIGH);
} else

if(  (distance > 50) && (distance <= 60)  )
{

  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, LOW);
  digitalWrite(led4, LOW);
  digitalWrite(led5, HIGH);
} else
```

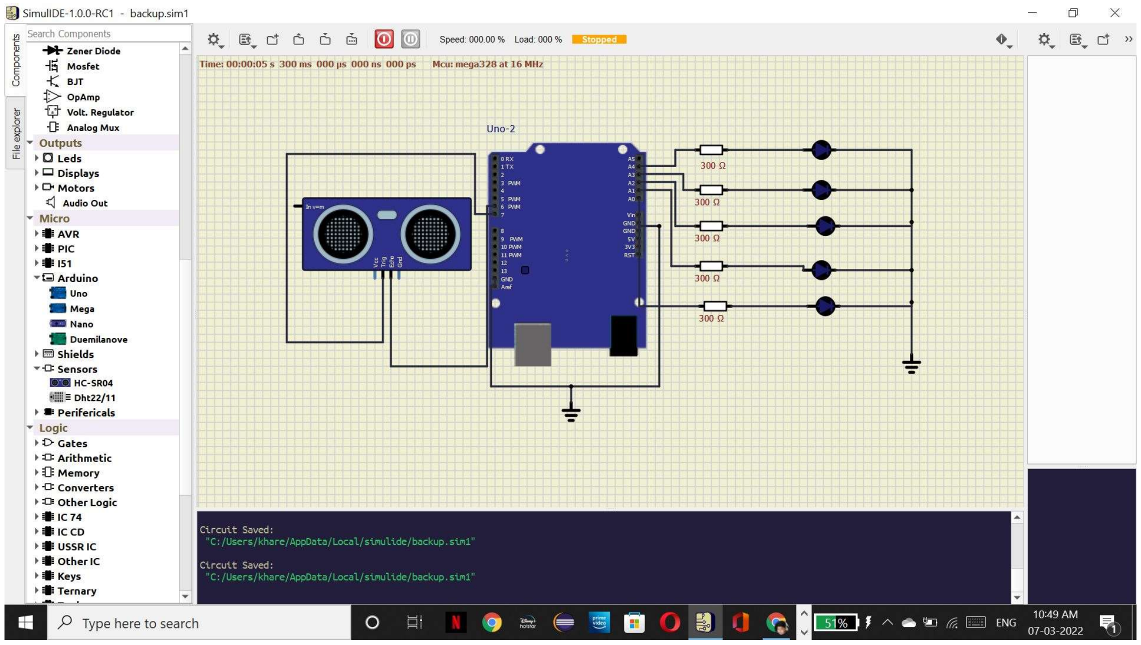
```
if(  distance > 60 )
{


  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, LOW);
  digitalWrite(led4, LOW);
  digitalWrite(led5, LOW);
}


}
```

Output:

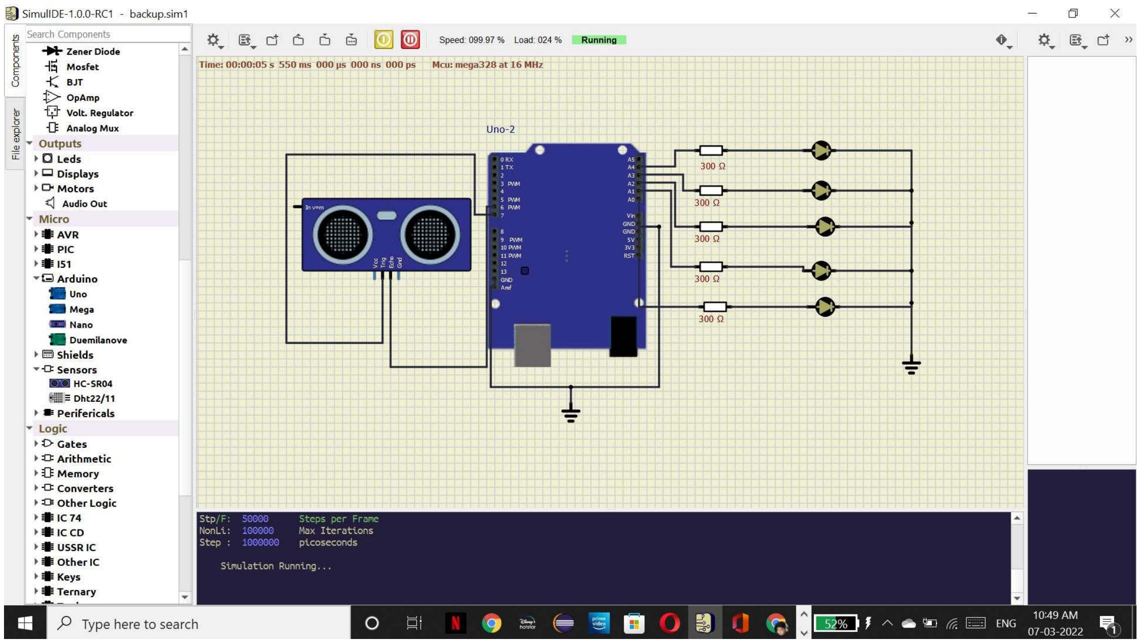
Before running:



```
Circuit Saved:
 "C:/Users/khare/AppData/Local/simulide/backup.sim1"

Circuit Saved:
 "C:/Users/khare/AppData/Local/simulide/backup.sim1"
```

After running:



```
Stp/F:  50000      Steps per Frame
NonLi:  100000     Max Iterations
Step :  1000000    picoseconds

    Simulation Running...
```

CONCLUSION:

In this project we came to know the working of Arduino, its hardware / software features and its applications as to where it is currently being used. We have also learnt how to write sketches for Arduino in its own IDE (software). Developing new ideas with Arduino is endless. The possibilities of using an Arduino to learn and develop new ideas are infinite. Though it does have its own limitations, it is a great tool that can be used in learning.