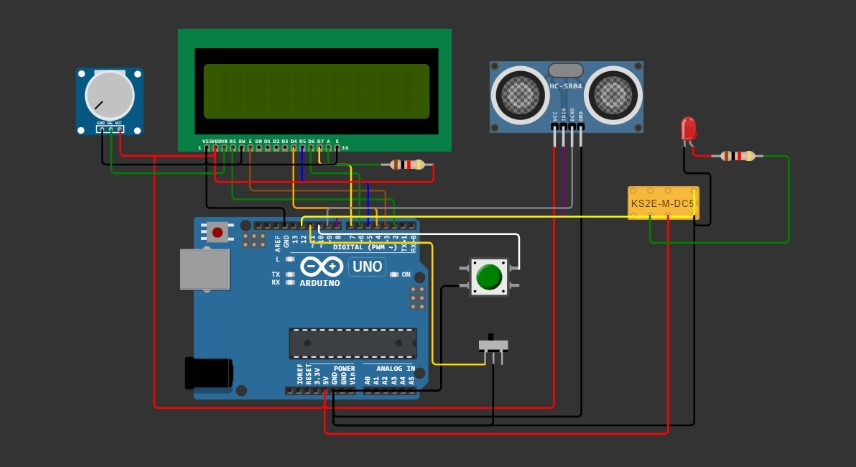
**PHASE 4**

**SMART WATER SYSTEM**

**CIRCUIT SIMULATION**



**CODE**

The code is written for an Arduino platform to control a water level monitoring system.

#include <EEPROM.h>

#include <LiquidCrystal.h>

LiquidCrystal lcd(2,3,4,5,6,7);

long duration, inches;

int set\_val,percentage;

bool state,pump;

void setup() {

  lcd.begin(16, 2);

  lcd.print("WATER LEVEL:");

  lcd.setCursor(0, 1);

  lcd.print("PUMP:OFF MANUAL");

  pinMode(8, OUTPUT);

  pinMode(9, INPUT);

  pinMode(10, INPUT\_PULLUP);

  pinMode(11, INPUT\_PULLUP);

  pinMode(12, OUTPUT);

   set\_val=**EEPROM**.read(0);

   if(set\_val>150)set\_val=150;

}

void loop() {

   digitalWrite(3, LOW);

   delayMicroseconds(2);

   digitalWrite(8, HIGH);

   delayMicroseconds(10);

   digitalWrite(8, LOW);

   duration = pulseIn(9, HIGH);

   inches = microsecondsToInches(duration);

   percentage=(set\_val-inches)\*100/set\_val;

   lcd.setCursor(12, 0);

   if(percentage<0)percentage=0;

   lcd.print(percentage);

   lcd.print("%   ");

   if(percentage<30&digitalRead(11))pump=1;

   if(percentage>99)pump=0;

   digitalWrite(12,!pump);

   lcd.setCursor(5, 1);

   if(pump==1)lcd.print("ON ");

   else if(pump==0) lcd.print("OFF");

    lcd.setCursor(9, 1);

    if(!digitalRead(11))lcd.print("MANUAL");

    else lcd.print("AUTO   ");

    if(!digitalRead(10)&!state&digitalRead(11)){

      state=1;

      set\_val=inches;

**EEPROM**.write(0, set\_val);

      }

     if(!digitalRead(10)&!state&!digitalRead(11)){

        state=1;

        pump=!pump;

      }

    if(digitalRead(10))state=0;

    delay(500);

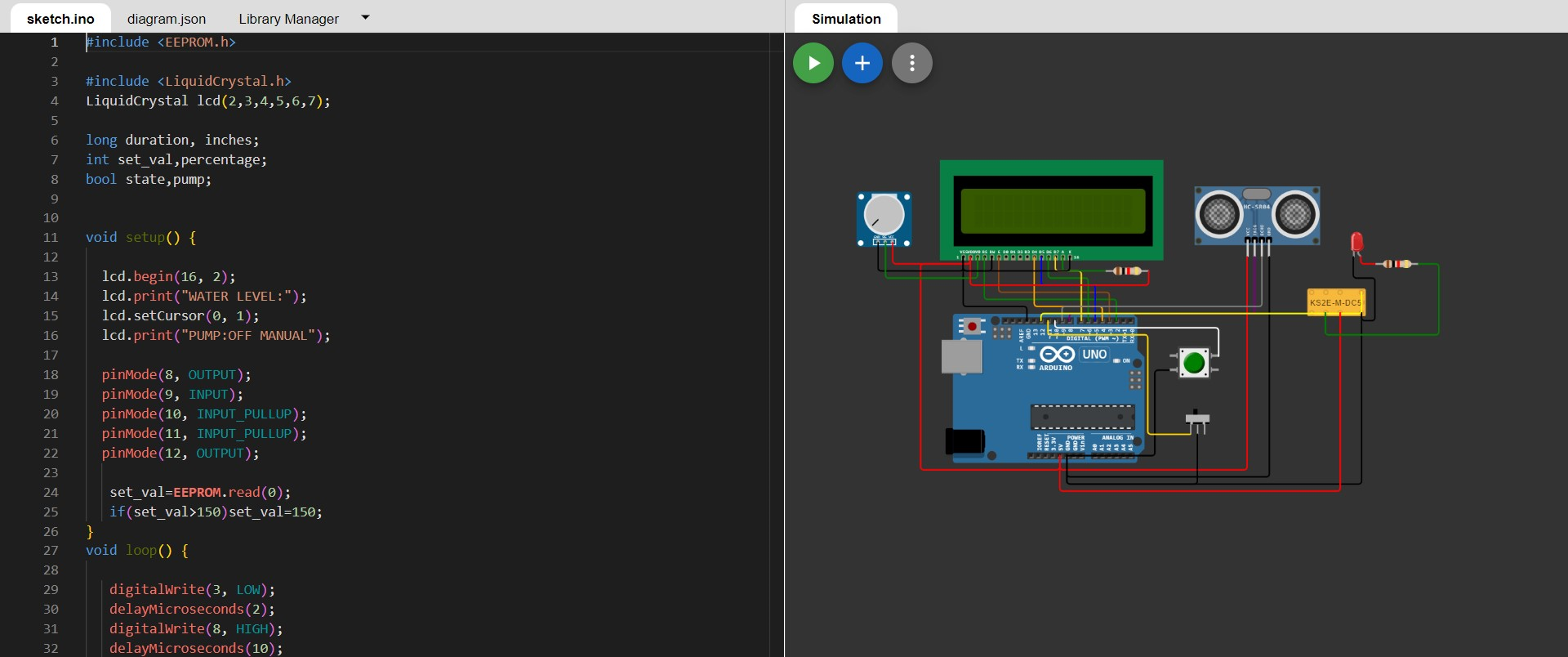
}

long microsecondsToInches(long microseconds) {

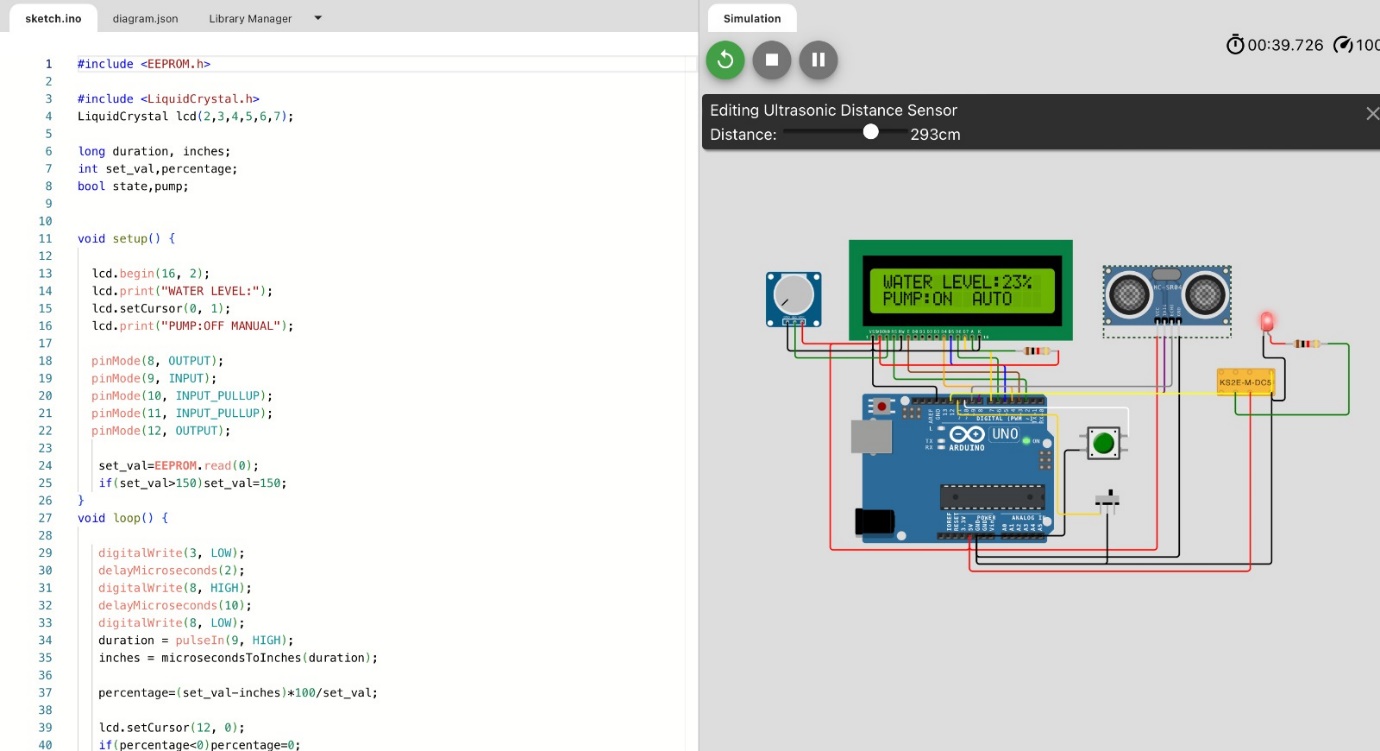
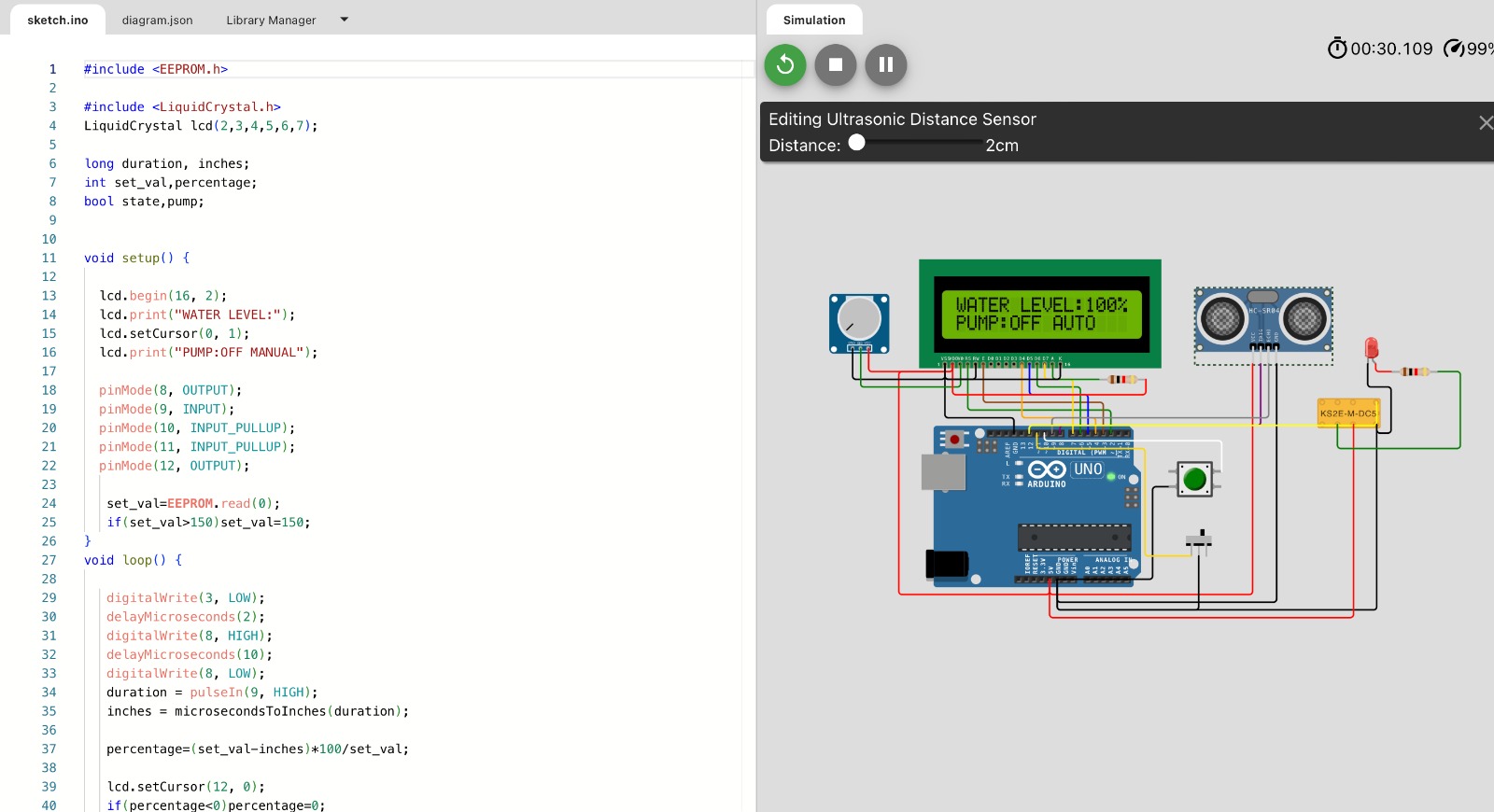
   return microseconds / 74 / 2;

}

**WOKWI SIMULATION INTERFACE**



**SIMULATION OUTPUT**



**CODE EXPLANATION**

**Including Libraries**

**#include <LiquidCrystal.h>**

* The code begins by including the **LiquidCrystal** library, which is used to control a character LCD display.

**Initializing LCD and Variables**

   LiquidCrystal lcd(2, 3, 4, 5, 6, 7);

   long duration, inches;

   int set\_val, percentage;

   bool state, pump;

* An instance of the **LiquidCrystal** class is created, indicating the pins used to connect to the LCD.
* **duration** and **inches** are used to store the duration of a pulse and the corresponding distance in inches.
* **set\_val** stores a set water level value (a reference point for the desired water level).
* **percentage** is used to calculate the percentage of the current water level compared to the set value.
* **state** and **pump** are Boolean flags for system state and the pump's status.

**Setup Function**

void setup() {

       // ... (initialize LCD and pins)

   }

* In the **setup()** function, the code initializes the LCD, sets up input and output pins and reads a stored value from EEPROM, which is the reference water level.

**Loop Function**

 void loop() {

       // ... water level measurement, calculations, and control

   }

* The main logic of the code is in the **loop()** function.

**Water Level Measurement**

* It triggers an ultrasonic distance sensor connected to **pin 9** to measure the distance to the water surface.
* The duration of the pulse is measured and converted to inches using the **microsecondsToInches** function.

**Calculating Water Level Percentage**

* The code calculates the water level percentage using the difference between the set value and the actual measurement.

**Displaying Water Level and Pump Status on LCD**

* The calculated percentage is displayed on the LCD.
* The code determines whether the pump should be on based on the water level and whether it's in manual or automatic mode.
* The pump's status is displayed on the LCD.

**Controlling the Pump**

* The pump is controlled based on the water level. If the water level is below 30% and in automatic mode (as determined by pin 11), the pump is turned on.
* If the water level is above 99%, the pump is turned off.

**User Interaction**

* The code checks for user interaction via a button connected to pin 10. If the button is pressed and released, it stores the current water level as the set value in EEPROM.
* If the button is pressed and released in manual mode (determined by pin 11), it toggles the pump status.

**Delay**

* There is a delay of 500 milliseconds before the loop repeats.

**Summary**

This code controls a water level monitoring and pump control system. It uses an ultrasonic sensor to measure the water level, displays the water level percentage and pump status on an LCD, and allows user interaction to set the desired water level and control the pump manually. The pump operates automatically to maintain the water level within a specified range.