TITLE:SMART WATER FOUNTAIN

PHASE 4:DEVELOPMENT.

Student Name:K.Santhosh

Reg_No:510421106042

Creating a real-time smart water fountain platform involves a combination of front end and backend technologies. Here’s a simplified outline using C and C++ and python programming with wi-fi connection for the front end and Node.

- PYTHON:

```
# Setup GPIO

GPIO.setmode(GPIO.BCM)

Water_sensor_pin = 17

Pump_pin = 18
```

~~~
GPIO.setup(water_sensor_pin, GPIO.IN)
GPIO.setup(pump_pin, GPIO.OUT)
  While True:
    If GPIO.input(water_sensor_pin) == GPIO.LOW:
      Print("Water level is low. Turning on the pump.")
      GPIO.output(pump_pin, GPIO.HIGH)
    Else:
      Print("Water level is sufficient. Turning off the pump.")
      GPIO.output(pump_pin, GPIO.LOW)

    Time.sleep(5)  # Check water level every 5 seconds

Except KeyboardInterrupt:
  Print("Exiting…")
  GPIO.cleanup()
```
~~~

- C++

```
#include <Arduino.h>
```

```
// Define pins for water pump and water level sensor.
Const int PUMP_PIN = 8;
Const int SENSOR_PIN = 9;
Void setup() {
  pinMode(PUMP_PIN, OUTPUT);
  pinMode(SENSOR_PIN, INPUT);
  Serial.begin(9600);
}
Void turnOnPump() {
  digitalWrite(PUMP_PIN, HIGH);
}
Void turnOffPump() {
  digitalWrite(PUMP_PIN, LOW);
}
Bool isWaterLevelLow() {
  Return digitalRead(SENSOR_PIN) == LOW;
}
Void loop() {
  If (isWaterLevelLow()) {
    Serial.println("Water level is low. Turning on the pump.");
```

~~~~
    turnOnPump();

    Delay(5000); // Run the pump for 5 seconds (adjust as needed).

    turnOffPump();

  }

  Delay(1000); // Check water level periodically.

}

```
~~~~

❖ C PROGRAM

~~~~
#include <stdio.h>

#include <stdlib.h>

#include <wiringPi.h>  // You'll need to install WiringPi library for Raspberry Pi or similar.

// Define GPIO pins for water pump and water level sensor.

#define PUMP_PIN 17

#define SENSOR_PIN 18



Void setup() {

    wiringPiSetupGpio();  // Initialize the WiringPi library.



    pinMode(PUMP_PIN, OUTPUT);

    pinMode(SENSOR_PIN, INPUT);
~~~~

```
}

Void turnOnPump() {
    digitalWrite(PUMP_PIN, HIGH);
}

Void turnOffPump() {
    digitalWrite(PUMP_PIN, LOW);
}

Int isWaterLevelLow() {
    Return digitalRead(SENSOR_PIN) == LOW;
}

Int main() {
    If (wiringPiSetupGpio() == -1) {
        Fprintf(stderr, “Unable to initialize WiringPi. Exiting.\n”);
        Return 1;
    }
    Setup();


    While (1) {
      If (isWaterLevelLow()) {
        turnOnPump();
        delay(5000);  // Run the pump for 5 seconds (adjust as needed).
        turnOffPump();
      }
    }


    Return 0;
}
```

- **MICROPROCESSOR PROGRAM:**

~~~
```assembly
ORG 0x1000     ; Set the origin address


PUMP_PIN  EQU P1.0   ; Define pump control pin
SENSOR_PIN  EQU P1.1 ; Define water level sensor pin


MAIN:          ; Main program
~~~

```
  MOV P1, #0xFF   ; Set P1 as output

  MOV P2, #0x00   ; Set P2 as input


LOOP:

  CLR A          ; Clear accumulator

  MOV A, P2       ; Read the state of the water level sensor

  CJNE A, #0, WATER_LOW ; If water level is low, jump to WATER_LOW

  SJMP CONTINUE


WATER_LOW:

  SETB P1.0       ; Turn on the water pump

  ACALL DELAY     ; Delay for 5 seconds

  CLR P1.0        ; Turn off the water pump


CONTINUE:

  ACALL DELAY     ; Delay before checking water level again

  SJMP LOOP       ; Repeat the loop


DELAY:

  MOV R5, #100    ; Load R5 with 100
```

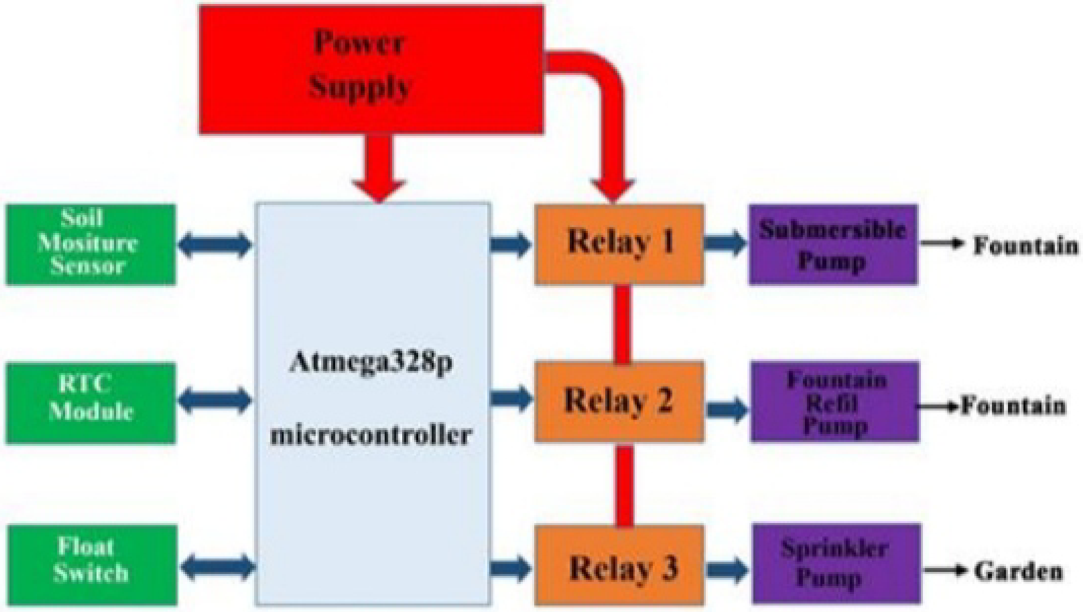
```
DELAY_LOOP:

   DJNZ R5, DELAY_LOOP ; Decrement R5, repeat until R5 is zero

   RET

END
```

BLOCK DIAGRAM:



COUNCLUSION:

In conclusion, the smart water fountain represents an innovative and efficient solution for promoting hydration and environmental sustainability. Its advanced features, such as automated refilling, water quality monitoring, and user-friendly mobile app control, make it a valuable addition to both public spaces and homes. By encouraging healthier hydration habits and reducing water wastage, these smart fountains contribute to a more sustainable and well-connected future.