Industrial IoT Minor

Name: Kshitij Kumar Sharma   Roll No: 1905514   Branch: CSE

Experiment - 2

AIM: To interface analog voltage input to NodeMCU.

OBJECTIVE: To write a code in Arduino IDE and interface LM35.

THEORY:
The circuit connections are made as follows:
- ➢ Pin 1 of the LM35 goes into +3v of the NodeMCU.
- ➢ Pin 2 of the LM35 goes into Analog Pin A0 of the NodeMCU.
- ➢ Pin 3 of the LM35 goes into Ground Pin (GND) of the NodeMCU.

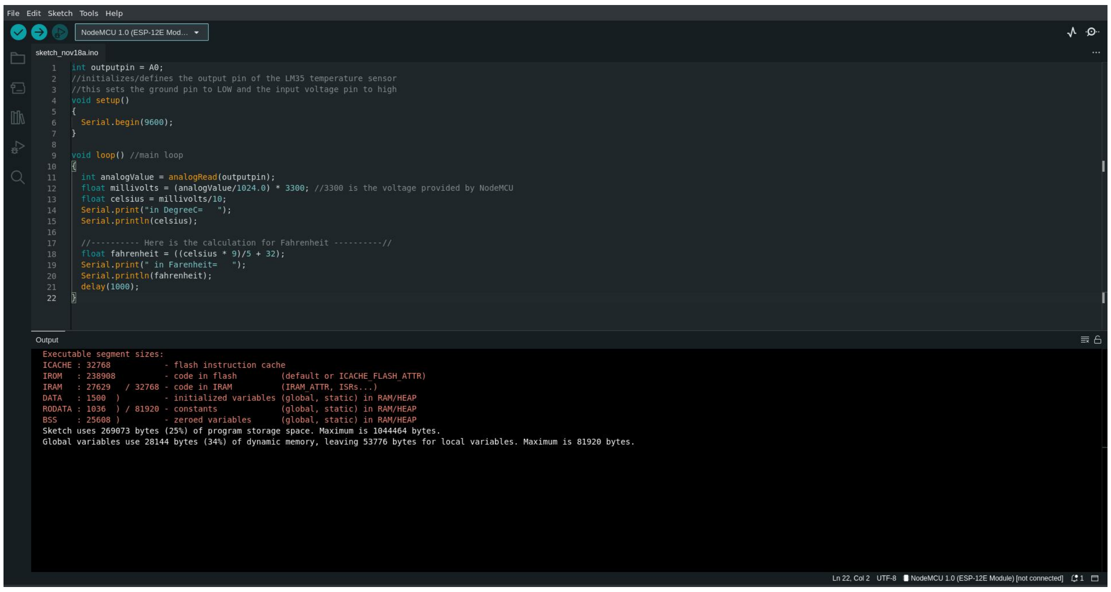
Before getting the Celsius reading of the temperature, the analog output voltage from LM35 must first be read from the Vout pin of LM35. This will be the raw value divided by 1024 times 3300. It is divided by 1024 because a span of 1024 occupies 3.3v. Here we get the ratio of the raw value to the full span of 1024 and then multiply it by 3300 to get the millivolt value. Since the output pin can give out a maximum of 3.3 volts (1024), 1024 represents the possible range it can give out.

SIMULATION CODE:

```
int outputpin = A0;
//initializes/defines the output pin of the LM35 temperature sensor
//this sets the ground pin to LOW and the input voltage pin to high
void setup()
{
Serial.begin(9600);
}

void loop() //main loop
{
int analogValue = analogRead(outputpin);
float millivolts = (analogValue/1024.0) * 3300;
//3300 is the voltage provided by NodeMCU
float celsius = millivolts/10;
Serial.print("in DegreeC= ");
Serial.println(celsius);
//---------- Here is the calculation for Fahrenheit ----------//
float fahrenheit = ((celsius * 9)/5 + 32);
Serial.print(" in Farenheit= ");
Serial.println(fahrenheit);
delay(1000);
}
```

SIMULATION RESULT:

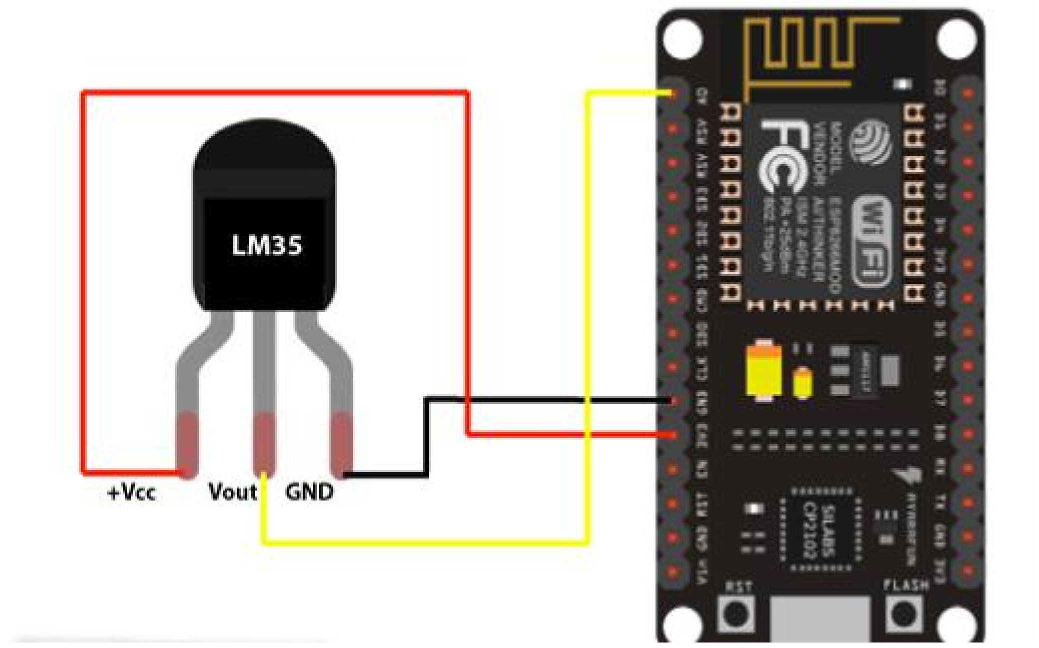

File Edit Sketch Tools Help
NodeMCU 1.0 (ESP-12E Mod...
sketch_nov18a.ino
Output
Ln 22, Col 2 UTF-8 NodeMCU 1.0 (ESP-12E Module) [not connected] 1

```cpp
int outputpin = A0;
//initializes/defines the output pin of the LM35 temperature sensor
//this sets the ground pin to LOW and the input voltage pin to high
void setup()
{
  Serial.begin(9600);
}

void loop() //main loop
{
  int analogValue = analogRead(outputpin);
  float millivolts = (analogValue/1024.0) * 3300; //3300 is the voltage provided by NodeMCU
  float celsius = millivolts/10;
  Serial.print("in DegreeC=   ");
  Serial.println(celsius);

  //---------- Here is the calculation for Fahrenheit ----------//
  float fahrenheit = ((celsius * 9)/5 + 32);
  Serial.print(" in Farenheit=   ");
  Serial.println(fahrenheit);
  delay(1000);
}
```

```
Executable segment sizes:
ICACHE : 32768           - flash instruction cache
IROM   : 238908          - code in flash         (default or ICACHE_FLASH_ATTR)
IRAM   : 27629   / 32768 - code in IRAM          (IRAM_ATTR, ISRs...)
DATA   : 1500  )         - initialized variables (global, static) in RAM/HEAP
RODATA : 1036  ) / 81920 - constants             (global, static) in RAM/HEAP
BSS    : 25608 )         - zeroed variables      (global, static) in RAM/HEAP
Sketch uses 269073 bytes (25%) of program storage space. Maximum is 1044464 bytes.
Global variables use 28144 bytes (34%) of dynamic memory, leaving 53776 bytes for local variables. Maximum is 81920 bytes.
```



CONCLUSION: After performing this experiment we were able to log the temperature in Arduino IDE’s serial monitor.