Summer Session KINGS 2019

Internet of things Developments using NodeMCU and IoT Platform

Jong-Hyun Kim
jhkim@dit.ac.kr
Dept. of Computer & Information
DONG-EUI INSTITUTE OF TECHNOLOGY

Table of Contents

- Introduction to NodeMCU
- Basic Ardunio Programming on NodeMCU
  - Ardunio Digital I/O Functions
  - Circuit Wiring
  - Simple LED Control
  - Reading Humidity & Temperature Sensor Values
- IoT Platform : Ubidots
- Hands-on Lab 01
  - IoT based Real Time Humidity & Temperature Monitor
- Hands-on Lab 02
  - Smart Switch Controller
- Future Study Topics

IoT Development Toolkits

- Open Source Hardware Platforms

Arduino Uno

Raspberry Pi 3 B+

Beaglebone Black

Intel Edison

Intel Galileo

NodeMCU

NodeMCU/ ESP8266

- NodeMCU : An open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.

- ESP8266 : a low cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability produced by Shanghai-based Chinese manufacturer, Espressif Systems.

Open-source, Interactive, Programmable, Low cost, Simple, Smart, WI-FI enabled

Arduino-like hardware IO

Nodejs style network API

Lowest cost WI-FI

NodeMCU Development Kits

- The Development Kit based on ESP8266
- Integrates GPIO, PWM, I2C, 1-Wire and ADC all in one board
- Growth of ESP8266 modules

NodeMCU : Open Source H/W Platform

NodeMCU GOIP

NodeMCU GOIP

- GPIO 0,2,15 : booting mode setting
- GPIO 1,3 : Serial Communication
- GPIO 6~11 : flash memory
- GPIO 16 : sleep mode
- User available GPIO Pins
  - GPIO 4(D2), GPIO 5(D1), GPIO 12~14(D6,D7,D5)
  - internal LED : GPIO 2(D4)

NodeMCU Development Environments

- LUA : Script language, Interpreter

```
gpio.mode(3, gpio.OUTPUT)
while 1 do
  gpio.write(3, gpio.HIGH)
  tmr.delay(1000000)   -- wait 1,000,000 us = 1 second
  gpio.write(3, gpio.LOW)
  tmr.delay(1000000)   -- wait 1,000,000 us = 1 second
end
```

NodeMCU Development Environments

- Ardunio IDE : CC++, Compiler

Adruino Programming for NodeMCU

- https://www.ardunio.cc

- Connect USB microcable to PC
- Open Device manager of PC, check Silicon Labs CP201x USB to UART Bridge

- Executing Arduino IDE

- Setting Board Manager

- Setting Board Manager : Search “esp8266 by ESP8266 Community”

- Setting Board Manager : Select NodeMCU 1.0(ESP-12E Module)

- Setting serial port

- Setting Board Manager URLs

"**http://arduino.esp8266.com/stable/package_esp8266com_index.json**"

Arduino Programming Basics

Hello World!

```
void setup() {
    // put your setup code here, to run once:
    Serial.begin(115200);
}

void loop() {
    // put your main code here, to run repeatly:
    Serial.print("Hello World\n");
}
```

Arduino Digital I/O functions

- **`pinMode(pin, mode)`**
  - Configures the specified pin to behave either as an input or an output.
- **`digitalWrite(pin, value)`**
  - Write a **HIGH(5V or 3.3V)** or a **LOW(0V or ground)** value to a digital pin
- **`delay(ms)`**
  - Pauses the program for the amount of time (in milliseconds) specified as parameter.
- https://www.arduino.cc/reference/en/

Basic LED Control on NodeMCU

- Perparation materials

NodeMCU

Jump Cables

LED

Register
220 Ω

Internal LED : LED_BUILTIN

Blink_Internal_LED | Arduino 1.8.9 (Windows Store 1.8.21.0)

File Edit Sketch Tools Help

Blink_Internal_LED §

```
//  Note that this sketch uses LED_BUILTIN to find the pin with the internal LED

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);     // Initialize the LED_BUILTIN pin as an output
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, LOW);   // Turn the LED on (Note that LOW is the voltage le
  // but actually the LED is on; this is because
  // it is active low on the ESP-01)
  delay(1000);                      // Wait for a second
  digitalWrite(LED_BUILTIN, HIGH);  // Turn the LED off by making the voltage HIGH
  delay(2000);                      // Wait for two seconds (to demonstrate the active
}
```

Done Saving.

15 NodeMCU 1.0 (ESP-12E Module) on COM13

Circuit Wiring

- LED_BUILITIN -> D4

Operation of Internal LED : D4 pin

LED Control : D2 pin

LED Control : D2 pin

Blin_LED_D2 | Arduino 1.8.9 (Windows Store 1.8.21.0)

File Edit Sketch Tools Help

Blin_LED_D2 §

```
// Note that this sketch uses LED PIn D2(GPIO4)
#define LED_PIN D2
void setup() {
  pinMode(LED_PIN, OUTPUT);     // Initialize the LED_BUILTIN pin as an output
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_PIN, LOW);   // Turn the LED on (Note that LOW is the voltage level
  delay(1000);                      // Wait for a second
  digitalWrite(LED_PIN, HIGH);  // Turn the LED off by making the voltage HIGH
  delay(2000);                      // Wait for two seconds (to demonstrate the active
}
```

Done uploading.

```
Sketch uses 258604 bytes (24%) of program storage space. Maximum is 1044464 bytes.
Global variables use 26696 bytes (32%) of dynamic memory, leaving 55224 bytes for local va
Uploading 262752 bytes from C:\Users\user\AppData\Local\Temp\arduino_build_53403/Blin_LED_
..................................................................................... [ 31% ]
..................................................................................... [ 62% ]
..................................................................................... [ 93% ]
...................                                                                 [ 100% ]
```

1 NodeMCU 1.0 (ESP-12E Module) on COM13

Operation of LED : D2 pin

Reading Humidity and Temperature on NodeMCU

- Preparation materials

NodeMCU

Jump Cables

DHT11 Sensor

16x2 LCD Module

Circuit Wiring

DHT Sensor Library(1)

- Menu -> Sketch -> Include -> Library -> Manage Libraries

Adafruit Unified Sensor Library(2)

- Menu -> Sketch -> Include -> Library -> Manage Libraries

Sketch

Temp_Hum01 | Arduino 1.8.9 (Windows Store 1.8.21.0)

File Edit Sketch Tools Help

Temp_Hum01

```
#include <DHT.h>
#define DHTPIN D4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  dht.begin();
}

void loop() {
  // put your main code here, to run repeatedly:
  float temp = dht.readTemperature();
  float humi = dht.readHumidity();
  Serial.print("Temp: ");
  Serial.print(temp);
  Serial.print("  Humi: ");
  Serial.println(humi);
  delay(1000);
}
```

Done uploading.

```
Sketch uses 265380 bytes (25%) of program storage space. Maximum is 1044464 bytes.
Global variables use 26832 bytes (32%) of dynamic memory, leaving 55088 bytes for local va
Uploading 269520 bytes from C:\Users\user\AppData\Local\Temp\arduino_build_824906/Temp_Hum
..................................................................................... [ 30% ]
..................................................................................... [ 60% ]
..................................................................................... [ 90% ]
.......................                                                               [ 100% ]
```

19 NodeMCU 1.0 (ESP-12E Module) on COM13

COM13

```
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 45.00
Temp: 27.00  Humi: 45.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
Temp: 27.00  Humi: 46.00
```

16x2 I2C LCD Display

OLED Display

LCD I2C Pin connection

LCD	NodeMCU
GND	GND
VCC	3.3V
SDA	D2
SCL	D1

Circuit Wiring

16x2 LCD Cursor Matrix

LiquidCrystal-I2C-library

- Create a new folder called "LiquidCrystal_I2C" under the folder named "libraries" in your Arduino sketchbook folder.
- Create the folder "libraries" in case it does not exist yet. Place all the files in the "LiquidCrystal_I2C" folder.

  - https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library

Sketch

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

#include <DHT.h>

#define DHTPIN D4

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

LiquidCrystal_I2C lcd(0x27,16,2);

void setup() {

  Serial.begin(115200);

  dht.begin();

  // lcd init

  lcd.begin();

  // lcd back light on

  lcd.backlight();

  //lcd.print("Test");

}
```

```
void loop() {
  float temp = dht.readTemperature();
  float humi = dht.readHumidity();
  Serial.print("Temp: ");
  Serial.print(temp);
  Serial.print("  Humi: ");
  Serial.println(humi);

  // LCD Display
  // LCD Cursor change to (0,0)
  lcd.setCursor(0,0);
  lcd.print("Temp: ");
  lcd.print(temp);

  // LCD Cursor change to (0,1)
  lcd.setCursor(0,1);
  lcd.print("Humi: ");
  lcd.print(humi);

  delay(1000);
}
```

Ubidots : IoT Development Platform

- Ubidots Education : https://ubidots.com/education/
  - Sign up ->

https://ubidots.com/platform/

IoT and Cloud tools to build your business

Merge the physical and digital worlds into deployable, end-user ready IoT Applications that deliver insights and solve problems.

Lab 01 : IoT based Real Time Humidity/ Temperature Monitor

Ubidots Library(1)

- Download the UbidotsMicroESP8266 library
  - https://github.com/ubidots/ubidots-esp8266
- Select Download ZIP file

Ubidots Library(2)

- Click on Sketch -> Include Library -> Add .ZIP library
- Select the .ZIP file of UbidotsMicro8266 and the “Accept”
- Close the Ardunio IDE and open it again.

Copy your Ubidots Token

Sending DHT11 Sensor Values to Ubidots Cloud

- Input code, upload and execute

```
#include <DHT.h>
#include "UbidotsMicroESP8266.h"
#define TOKEN "xxxxx" // put here your Ubidits token
#define WIFISSID "melon"
#define PASSWORD "deitcs3217"
#define DHTPIN D4
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

Ubidots client(TOKEN);

void setup() {
  Serial.begin(115200);
  dht.begin();
  client.wifiConnection(WIFISSID, PASSWORD);
}
```

```
void loop() {
  float temp = dht.readTemperature();
  float humi = dht.readHumidity();
  Serial.print("Temp: ");
  Serial.print(temp);
  Serial.print("  Humi: ");
  Serial.println(humi);
  client.add("Temperature", temp);
  client.add("Humidity", humi);
  client.sendAll(true);
  delay(1000);
}
```

Devices Creation : ESP8266

- Wait for the device(ESP8266) to appear

Devices Status(1)

Devices Status(2)

Dashboards Creation : Add new widget

Dashboard Creation : Add Variable

Dashboard Creation: Select a Device, a Varibale

Dashboards : Chart, Table -> Historical Data

Dashboards : Indicator -> Gauge

Lab 02 : Smart Light Controller

Circuit Wiring

Input Code, upload & Execute

```
#include "UbidotsMicroESP8266.h"

#define DEVICE  "led-control"  // Put here your Ubidots
device label

#define VARIABLE  "led-value"  // Put here your Ubidots
variable label

 // Put here your Ubidots TOKEN

#define TOKEN  "A1E-btnEp0we6d27oA5sRLjlexPvxL02ro"

#define WIFISSID "melon"

#define PASSWORD "deitcs3217"

#define LED_PIN D2

Ubidots client(TOKEN);

void setup() {

    Serial.begin(115200);

    client.wifiConnection(WIFISSID, PASSWORD);

    pinMode(LED_PIN, OUTPUT);

    digitalWrite(LED_PIN, 1);

}
```

```
void loop() {

    float value = client.getValueWithDevice(DEVICE, VARIABLE);

    if (value != ERROR_VALUE){

      Serial.print(F("value obtained: "));

      Serial.println(value);

      if(value == 1) digitalWrite(LED_PIN, HIGH);

      else digitalWrite(LED_PIN, LOW);

    }else{

      Serial.println(F("Error getting value"));

    }

    delay(1000);

}
```

Device Creation

- MyDevices -> My Data Source -> SmartLEDController

Setting Variables

Dashboard : Add widgets

Dashboard : Select a Device, a Variable

Dashboard Creation : Control -> Switch

Dashboard : Smart Switch Controller

Serial Monitor of LED Control

COM4

```
Getting your variable:
value obtained: 0.00
UbidotsESP8266/1.3|LV|A1E-btnEp0we6d27oA5sRLjlexPvxL02ro|SmartLED:ledValue|end
Getting your variable:
value obtained: 0.00
UbidotsESP8266/1.3|LV|A1E-btnEp0we6d27oA5sRLjlexPvxL02ro|SmartLED:ledValue|end
Getting your variable:
value obtained: 1.00
UbidotsESP8266/1.3|LV|A1E-btnEp0we6d27oA5sRLjlexPvxL02ro|SmartLED:ledValue|end
Getting your variable:
value obtained: 0.00
UbidotsESP8266/1.3|LV|A1E-btnEp0we6d27oA5sRLjlexPvxL02ro|SmartLED:ledValue|end
Getting your variable:
value obtained: 1.00
UbidotsESP8266/1.3|LV|A1E-btnEp0we6d27oA5sRLjlexPvxL02ro|SmartLED:ledValue|end
Getting your variable:
value obtained: 0.00
UbidotsESP8266/1.3|LV|A1E-btnEp0we6d27oA5sRLjlexPvxL02ro|SmartLED:ledValue|end
Getting your variable:
value obtained: 0.00
UbidotsESP8266/1.3|LV|A1E-btnEp0we6d27oA5sRLjlexPvxL02ro|SmartLED:ledValue|end
Getting your variable:
value obtained: 0.00
UbidotsESP8266/1.3|LV|A1E-btnEp0we6d27oA5sRLjlexPvxL02ro|SmartLED:ledValue|end
Getting your variable:
value obtained: 1.00
```

Operation of IoT Devices

Mobile Application : Ubidots Explorer

- https://play.google.com/store/apps/details?id=com.ubidots.ubiapp

Future Study Topics

- IoT Communication Protocols

- Use Ubidots MQTT…
  - https://github.com/ubidots/ubidots-mqtt-esp
  - https://ubidots.com/blog/temperature-control-with-ubidots/

- Big Data Analytics, Machine Learning
- Apply IoT to field areas
  - Smart Farm
  - Smart Home
  - Smart Healthcare
  - Smart Transportation
  - Smart Energy…

Program Sources

- GitHub : https://github.com/kings-iot-2019

References

1. https://www.theengineeringprojects.com/2018/10/introduction-to-nodemcu-v3.html
2. http://pdacontrolen.com/tutorial-platform-iot-ubidots-esp8266-sensor-dht11/
3. https://www.hackster.io/ubidots/projects
4. https://ubidots.com/education/
5. https://help.ubidots.com/