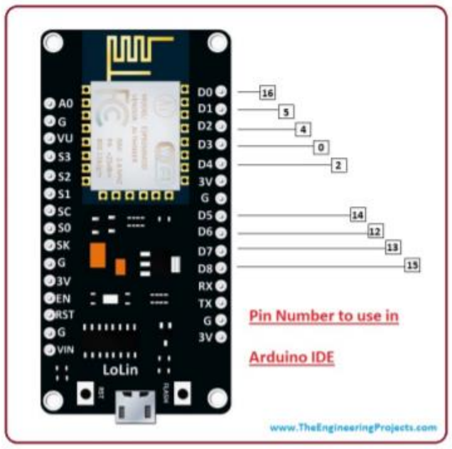
Internet of Things: Lab 3

Working with NodeMCU pins:

NodeMCU has labels for its pin and there are numbers that correspond to Arduino pins (in case you wish to use numbers). The table and figure illustrates these.

NodeMCU pin	Arduino pin number(to be used in your code)
D0	16
D1	5
D2	4
D3	0
D4	2
D5	14
D6	12
D7	13
D8	15
D9	3
D10	1



So to use pin D0 of NodeMCU you may choose to use Pin 16 in Arduino IDE code or simply D0.

Exercise 1 (Ungraded)

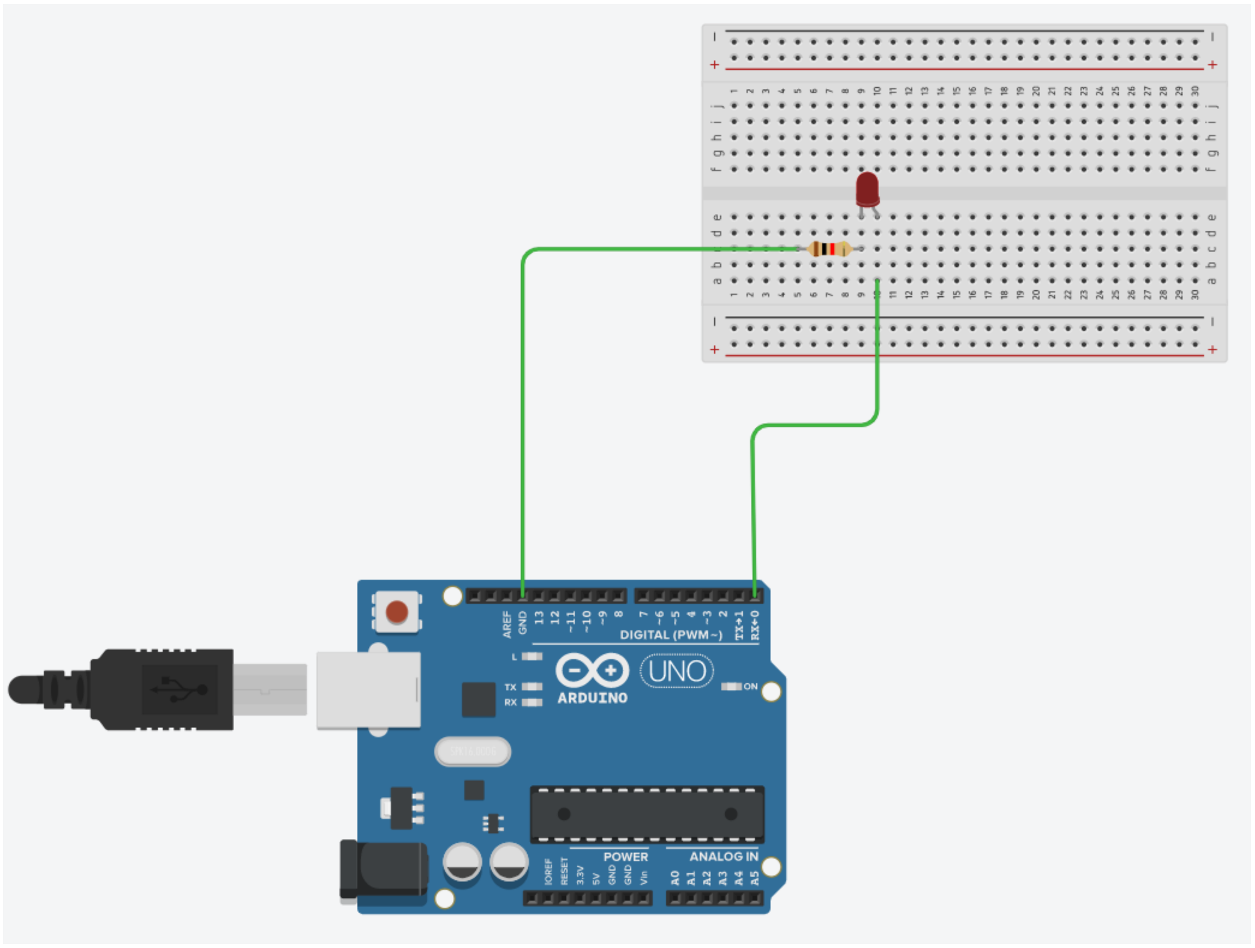
Test the basic blink code by going to File →Examples→ESP8266→Blink

Select the Blink Example code and upload it on the board.

The on-board LED should start to blink. That means you successfully programmed the board. The on-board LED is connected to pin D0 of NodeMCU.

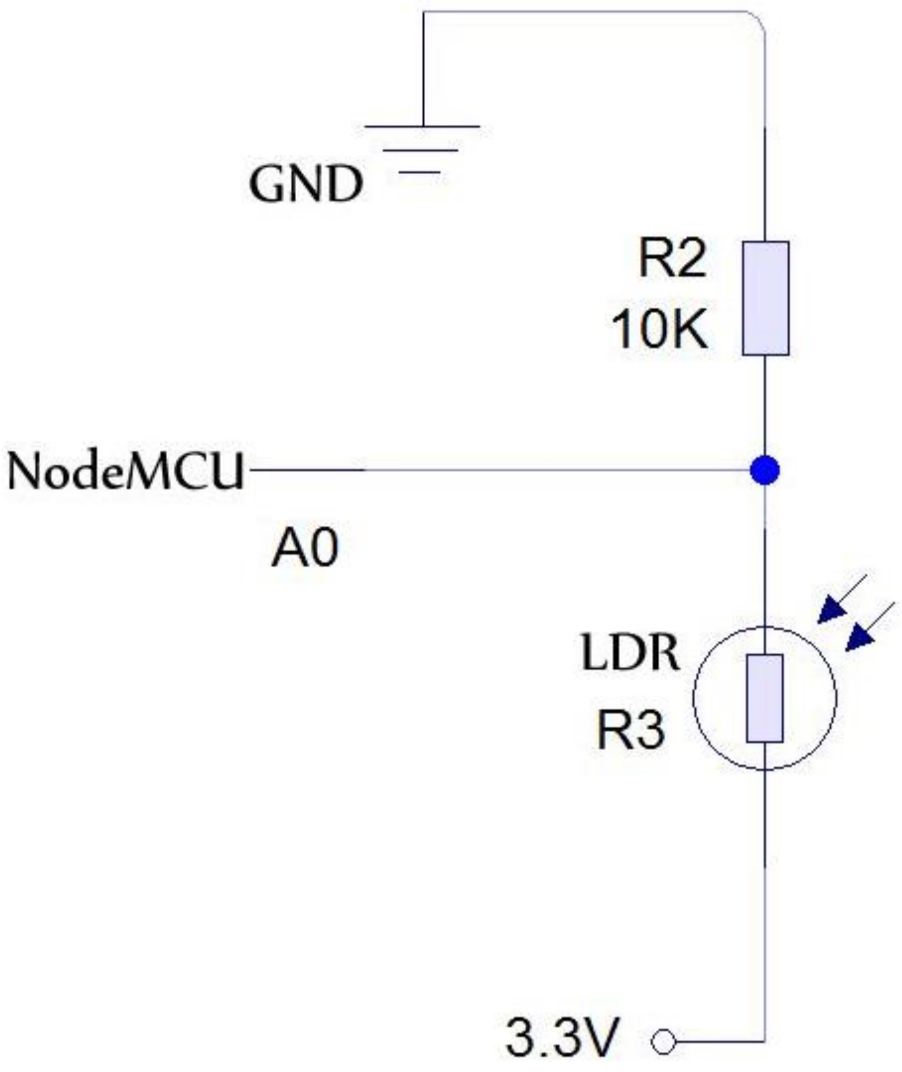
Exercise 2 – Working with an LED

Modify the blink code to use pin D0. Put D0 in the code and connect to pin D0. Connect an LED and a 1kΩ resister to pin D0. Ensure it blinks appropriately.



Exercise 3 – Working with an LDR

Since the output of an LDR is analog in nature, we connect it to an analog pin. Connect it to pin A0. Wire up your circuit like the schematic below.



Use the code below:

```
void setup() {

        Serial.begin(9600);   // initialize serial communication at 9600

}


void loop() {

        int sensorValue = analogRead(A0);   // read the input on analog pin 0

        Serial.println(sensorValue );   // print out the sensor value you read
        float voltage = sensorValue * (5.0 / 1023.0);   // Convert the analog readin
g (which goes from 0 - 1023) to a voltage (0 - 5V)

        Serial.println(voltage);   // print out the value you read

}
```

View the result from the serial monitor.

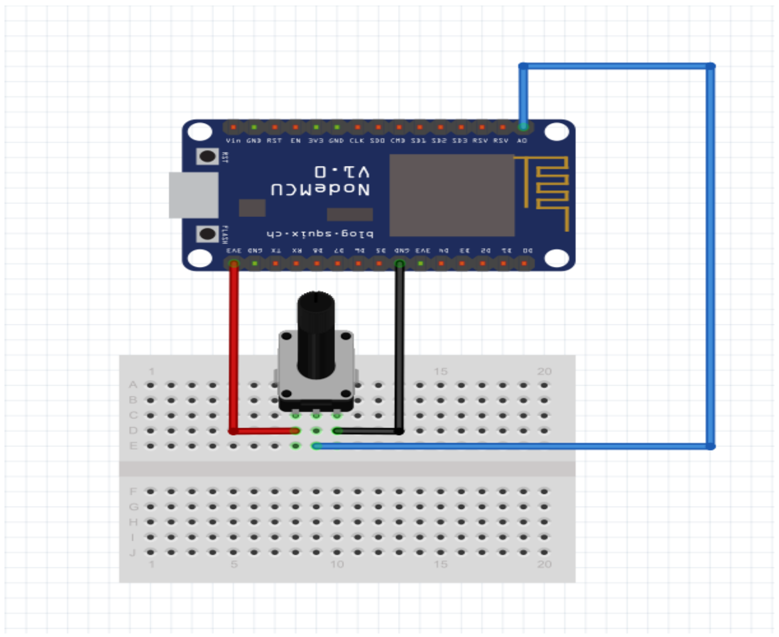
Exercise 4

Modify the code in Exercise 2 to print ldr values

a. Greater than 350
b. Less than 300

Exercise 5 – Working with the potentiometer

Do the connection below and use the potentiometer to control the brightness of the inbuilt led.



```
void setup()

{

  Serial.begin(115200);
  pinMode(A0, INPUT);

  pinMode(LED_BUILTIN, OUTPUT);

}

void loop()

{

  int potentiometer = analogRead(A0);

  Serial.println(potentiometer);

  analogWrite(LED_BUILTIN, potentiometer);

}
```

Task – Take Home

1. Research on how to use the temperature and humidity sensor with the NodeMCU. Write a code to print the temperature and humidity values greater than certain thresholds to the serial monitor.
2. Work with the ultrasonic level sensor. Use the serial monitor to print distance values greater than 10cm.
3. Combine different sensors to work together. Use DHT21/DHT21, LDR, and LED. Write a single code to do the following:
    - **a.** Blink the LED when the LDR value is greater than 200.
    - **b.** Blink the LED when the temperature is less than 33°C.
    - **c.** Print the LDR value, temperature values, and LED status (ON/OFF) on the serial monitor.

Submission: Take a **1min** working video (let us hear your voice describing the work) for question 3. Zip the **video, schematics, and code** and submit on canvas.

Appendix

Installation instructions:

The NodeMCU platform is a microcontroller that can be programmed with the Arduino IDE. It has the ESP8266 wireless module that provides WiFi connectivity already included.

Setting up ESP8266 (NodeMCU) in Arduino.

- In the Arduino IDE, Open **preferences** window from Arduino IDE. Go to File -> Preferences.
- Paste "**http://arduino.esp8266.com/stable/package_esp8266com_index.json**" into Additional Board Manager URLs field and click the "OK" button
- Open Boards Manager. Go to Tools -> Board -> Boards Manager
- Search ESP8266 board menu and install "esp8266 platform"
- Choose your ESP8266 board from Tools > Board > Generic ESP8266 Module
- A number of further settings may be needed. See ***configuration*** section below. If the board is not recognized, see the ***driver installation*** sections below.
- Restart the IDE

Driver installation: Version A: (the smaller narrow board)

To recognize board, install CP210X Drivers. Visit the link below and download the version compatible to your device.

Download from : https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers

On Mac, was identified as /dev/cu.SLAB_USBtoUART

Driver installation: Version B: (the wider Board)

Some boards have a different driver: Driver for Lolin nodeMCU appears to be CH340.[1]

For windows, use the Device Manager and update the driver from:

http://www.wch.cn/download/CH341SER_EXE.html

On Mac: it may be attached as a serial device in the device list(/dev/tty.wchusbserial*). a restart may be needed.

Configuration

After driver installation, some parameters may need to be configured. Look under the tools menu. Select the correct board , Choose NodeMCU 1.0 (ESP-12E Module). The generic ESP8266 also works.

After selecting the board use the settings below :-

- Flash Size : "4M (3M SPIFFS)" (from 512 SPIFFS)
- Debug Port : "Disabled"
- Debug Level: "None"
- IWIP Variant: "V2 Lower Memory"
- CPU Frequency: "80Mhz"

- Upload Speed: "921600"
- Erase Flash: "Sketch On"
- Port : "COM port available" (where the device is connected should show up)

Now you can upload your sketch on the board.

Test the basic blink code by going to File→Examples→ESP8266→Blink

Potential issues:

There may be compilation issues eg an error such as "python3/3.7.2-post1/python3: no such file or directory Error compiling for board NodeMCU 1.0 (ESP-12E Module).

```
Used:
/Users/xxxx/Library/Arduino15/packages/esp8266/hardware/esp8266/2.6.3/libraries/ESP8266WiFi
fork/exec  /Users/xxxx/Library/Arduino15/packages/esp8266/tools/python3/3.7.2-post1/python3: no
such file or directory
Error compiling for board NodeMCU 1.0 (ESP-12E Module).
```

This can be fixed by installing python 3. You may alternatively create a symbolic link to the python 3 location[2].