Politechnika Świętokrzyska

w Kielcach

Wydział Elektrotechniki, Automatyki i Informatyki

Sprawozdanie 3

Laboratoruim Iot

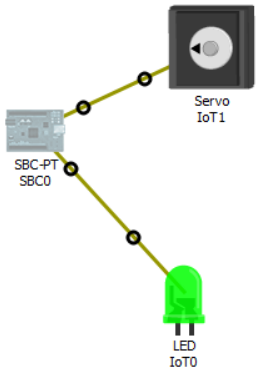
Gałka Karolina

3ID15A

20.11.2018r.

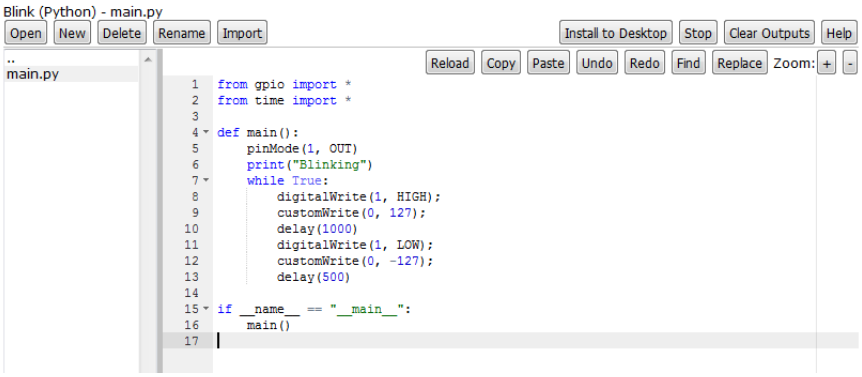
Zadanie 1 Packet Tracer – Simulating IoT Devices

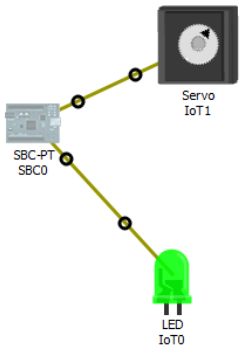
Naszym zadaniem było wykonanie topologii na rysunku poniżej.



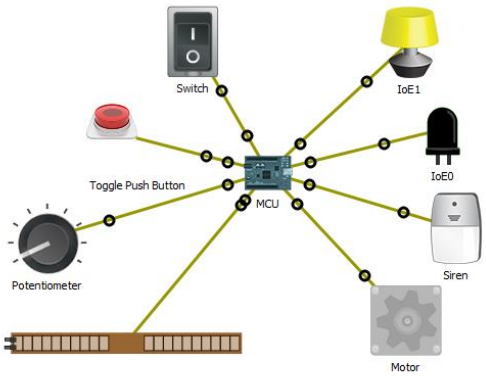
Następnie musieliśmy zmodyfikować plik main.py dopisując dwie linie:

```
customWrite(0, 127);
customWrite(0, -127);
```





Zadanie 2 Sensors and the PT Microcontroller in Packet Tracer 7.1



Naszym zadaniem było przeanalizowanie zasady działania tej topologii. Po lewej stronie znajdują się urządzenia wejściowe, natomiast po prawej wyjściowe. Każde z urządzeń wejściowych oddziałuje na inne urządzenie wyjściowe.

MCU (Python) - main.py

Open | New | Delete | Rename | Import

..
main.py

Reload | Copy | Paste | Undo

```python
from gpio import * # imports all modules in the GPIO library
from time import * # imports all modules in the time library

switchValue = 0 # initialise Switch sensor value global variable to 0
togglePushButtonValue = 0 # initialise Toggle Push Button sensor value global variable to 0
potentiometerValue = 0 # initialise Potentiometer sensor value global variable to 0
flexSensorValue = 0 # initialise Flex Sensor value global variable to 0

def readFromSensors():
    global switchValue # declare switchValue as global
    global togglePushButtonValue # declare togglePushButtonValue as global
    global potentiometerValue # declare potentiometerValue as global
    global flexSensorValue # declare flexSensorValue as global

    switchValue = digitalRead(0) # read Switch sensor value
    togglePushButtonValue = digitalRead(1) # read Toggle Push Button sensor value
    potentiometerValue = analogRead(A0) # read Potentiometer sensor value
    flexSensorValue = analogRead(A1) # read Flex Sensor value

def writeToActuators():
    if (switchValue == HIGH): # evaluates to True if the Switch sensor value is digital HIGH, otherwise false
        customWrite(2, "2") # turn on the Light
    else:
        customWrite(2, "0") # turn off the Light

    if (togglePushButtonValue == HIGH): # evaluates to True if the Toggle Push Button sensor value is digital HIGH, otherwise false
        digitalWrite(3, HIGH) # turn on the LED
    else:
        digitalWrite(3, LOW) # turn off the LED

    if (potentiometerValue > 512): # evaluates to True if the Potentiometer is turned at least half way
        customWrite(4, HIGH) # turn on the Siren
    else:
        customWrite(4, LOW) # turn off the Siren

    if (flexSensorValue > 0): # evaluates to True if the Flex Sensor is bent, otherwise false
        analogWrite(5, flexSensorValue) # turn on the motor with speed equal to the Flex Sensor value
    else:
        analogWrite(5, 0) # turn off the motor

def main(): # defines the main function
    pinMode(0, IN) # sets digital slot 0 (Switch) to input
    pinMode(1, IN) # sets digital slot 1 (Toggle Push Button) to input
    pinMode(2, OUT) # sets digital slot 2 (Light) to output
    pinMode(3, OUT) # sets digital slot 3 (LED) to output
    pinMode(4, OUT) # sets digital slot 4 (Siren) to output

    while True: # loop indefinitely
        readFromSensors() # call the readFromSensors function
        writeToActuators() # call the writeToActuators function
        delay(1000) # delay script execution for 1000 ms

if __name__ == "__main__": # Evaluates to True if this module is the script being executed, otherwise False if this module is being imported into another module
    main() # call the main function
```

Plik main.py zawiera instrukcje odpowiadające ze właściwe działanie tych urządzeń.