Programació de Microprocessadors

Challenge

Miquel Rodríguez Juvany

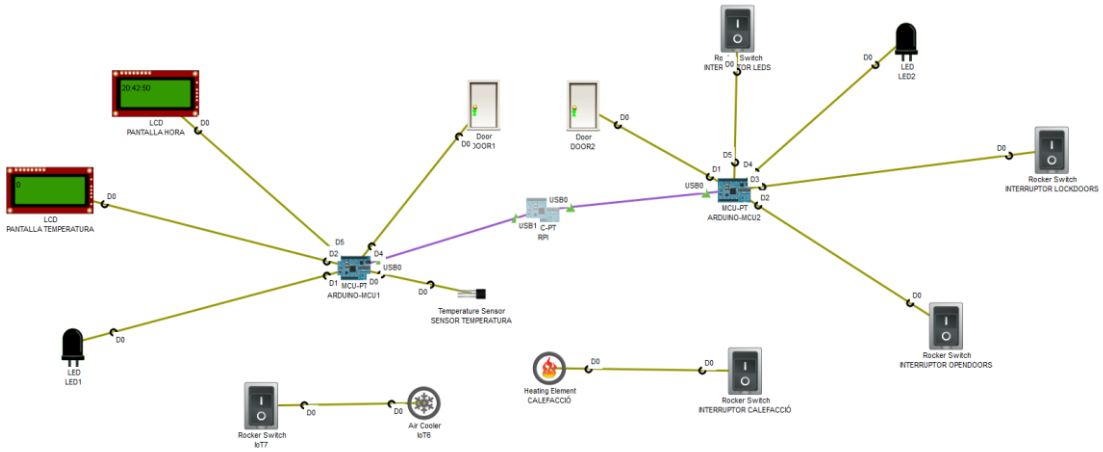
Léonard Janer

Índex

Bibliografía

Janer, L. (17 de Febrer de 2022). *Programació Microprocessadors: Challenge.* Obtenido de aulavirtual.tecnocampus.cat: https://aulavirtual.tecnocampus.cat/pluginfile.php/167010/mod_resource/content/9/PM2022-SESSION12.pdf

Esquema de la Topologia



El circuit està format per una Raspberry Pi que controla tot el circuit a través de dos microcontrol·ladors Arduino connectats per USB.

El primer microcontrol·lador Arduino té connectats 1 LED, dues pantalles, una porta i un sensor de temperatura.

El segon microcontrol·lador Arduino té connectats 1 LED, tres interruptors: un que control·la els LED, dos per gestionar les portes (un bloqueja, l'altre les obre).

Fora del control de la Raspberry Pi hi ha un aparell d'aire condicionat i un calefactor ambdós amb el seu propi interruptor.

La Raspberry Pi està connectada als Arduino amb un cable USB i la resta de dispositius estan connectats amb un "cable personalitzat del Internet of Things".

Els microcontrol·ladors Arduino reben informació dels diferents interruptors del circuit que envien a la Raspberry Pi, la qual s'encarrega de processar la informació i retorna ordres als microcontrol·ladors que són enviades als dispositius corresponents.

Codi MCU1

```
1  from gpio import *
2  from time import *
3  from usb import *
4
5  def loop():
6      hora()
7      temperatura()
8
9  def main():
10     ptmata1 = PTmata(0, 57600)
11     pinMode(2, OUT)
12     pinMode(5, OUT)
13     pinMode(0, IN)
14
15     while True:
16         while ptmata1.inWaiting() > 0:
17             ptmata1.processInput()
18         loop()
19
20 def hora():
21     time_string = strftime("%H:%M:%S", localtime())
22     customWrite(5, time_string)
23
24 def temperatura():
25     temp = analogRead(0)
26     customWrite(2, (temp * 200) / (1023) - 99)
27
28 if __name__ == "__main__":
29     main()
30
```

Es fa un import de les llibreries gpio, time i usb.

Es defineix el loop que executarà les funcions hora i temperatura. La funció hora s'encarrega de guardar la hora local en format Hora, Minuts i Segons i mostrar-la a la pantalla del PORT 5 que seria la pantalla que mostra la hora. La funció temperatura s'encarrega de llegir el que arriba al díode 0, el sensor de temperatura, i mostrar la temperatura (amb una formula que transforma els graus en Celsius) a la pantalla del PORT 2 que és la pantalla que mostra la temperatura.

Al main es declaren els pins d'entrada i sortida i es crida el loop perquè es vagi realitzant el programa sempre mirant l'estat de la resta del circuit.

Codi MCU2

```
from time import *
from usb import *

def main():
    ptmata2 = PTmata(0, 57600)

    while True:
        while ptmata2.inWaiting() > 0:
            ptmata2.processInput()
        ptmata2.readAndReportData()
        delay(1000)

if __name__ == "__main__":
    main()
```

Es fa un import de time i usb.

Es defineix el main senzillament preparant-lo per a rebre informació i deixant un segon de temps per a seguir rebent la informació.

Codi RPI

```
from time import *
from usb import *
from gpio import *

def main():
    ptmata1 = PTmata(1, 57600)
    ptmata1.pinMode(1, OUT)         # LED 1
    ptmata1.pinMode(4, OUT)         # Porta 1

    ptmata2 = PTmata(0, 57600)
    ptmata2.pinMode(1, OUT)         # Porta 2
    ptmata2.pinMode(2, IN)          # Interruptor OPENDOORS
    ptmata2.pinMode(3, IN)          # Interruptor LOCKDOORS
    ptmata2.pinMode(4, OUT)         # LED 2
    ptmata2.pinMode(5, IN)          # Interruptor LEDs

    openD = ptmata2.digitalRead(2)
    lock = ptmata2.digitalRead(3)
    block = False
    i = 1
    while True:
        while ptmata1.inWaiting() > 0:
            ptmata1.processInput()

        while ptmata2.inWaiting() > 0:
            ptmata2.processInput()

        nouOpen = ptmata2.digitalRead(2)
        nouLock = ptmata2.digitalRead(3)
        if nouOpen != openD or nouLock != lock:
            openD = nouOpen
            lock = nouLock
            ptmata1.customWrite(4, str(openD) + "," + str(lock))
            ptmata2.customWrite(1, str(openD) + "," + str(lock))

        led = ptmata2.digitalRead(5)
        if (led == 1023 and block == False):
            if i != 10:
                print(i)
                ptmata1.analogWrite(1, 1000)
                ptmata2.analogWrite(4, 1000)
                i += 1
                sleep(1)
            else:
                block = True
                ptmata1.analogWrite(1, 0)
                ptmata2.analogWrite(4, 0)
        elif (led == 0):
            ptmata1.analogWrite(1, 0)
            ptmata2.analogWrite(4, 0)
            block = False
            i = 1

if __name__ == "__main__":
    main()
```

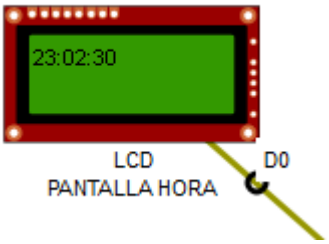
Es fa un import de les llibreries time, usb i gpio.

Al main es defineixen tots els pins d'entrada i sortida dels dos microcontrol·ladors Arduino. Es guarden els estats de lock i de open per a poder comparar a veure si ha canviat. També es llegeix el port del LED i es fa un contador per a comptar els 10 segons que hauria d'estar encès el LED com a màxim. Es vigila també que si no han passat encara els 10 segons però l'interruptor s'apaga, s'han d'apagar també els LED independentment del temps que hagi passat.

ptmata1 serveix per referenciar el MCU1 i ptmata2 serveix per referenciar el MCU2. D'aquesta manera es pot diferenciar l'ús de, per exemple, el port 1 que, en el MCU1 és el LED1 i en el cas del MCU2 és la DOOR2.

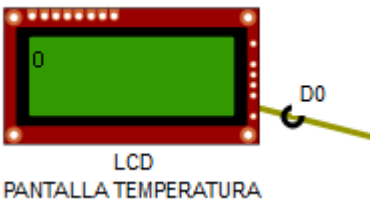
Time Display

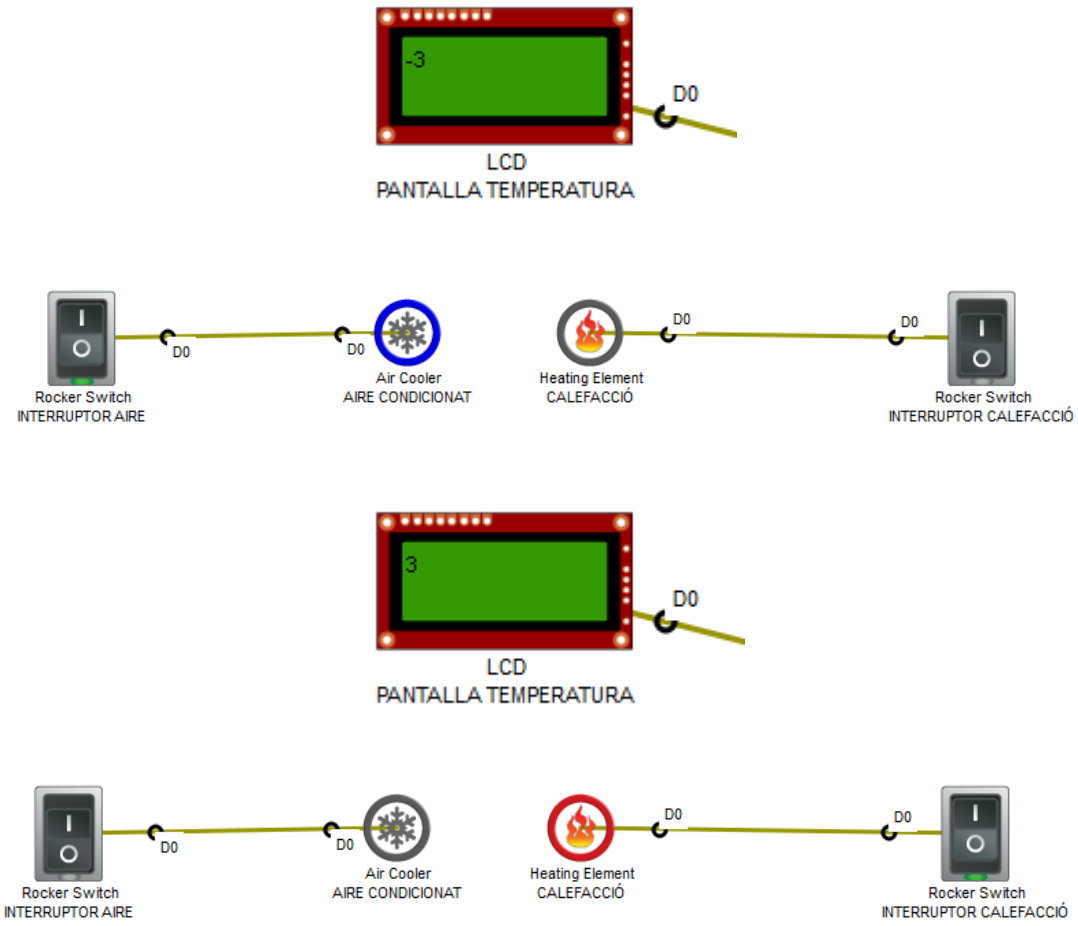
```
def hora():
    time_string = strftime("%H:%M:%S", localtime())
    customWrite(5, time_string)
```



Temperature Display

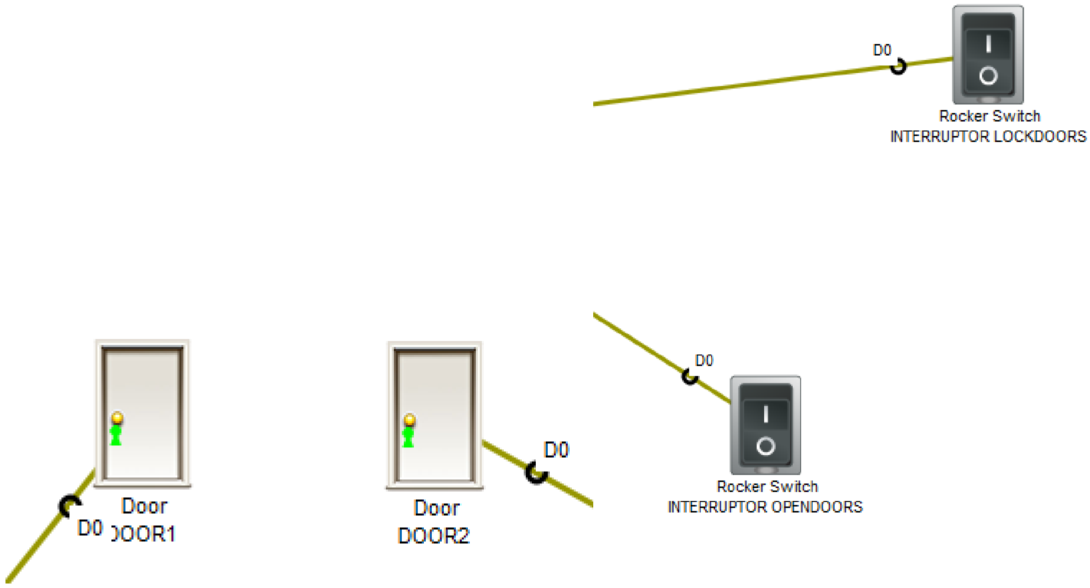
```
def temperatura():
    temp = analogRead(0)
    customWrite(2, (temp * 200) / (1023) - 99)
```

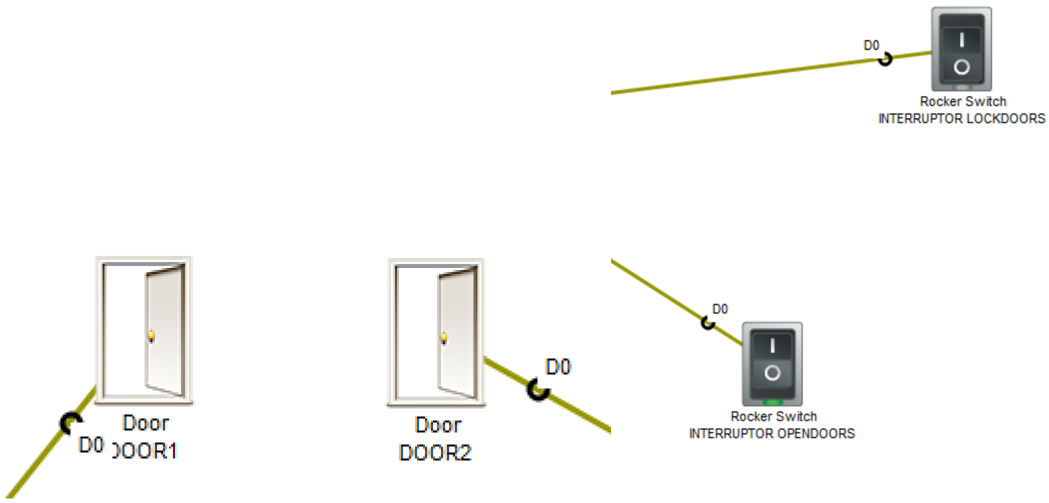


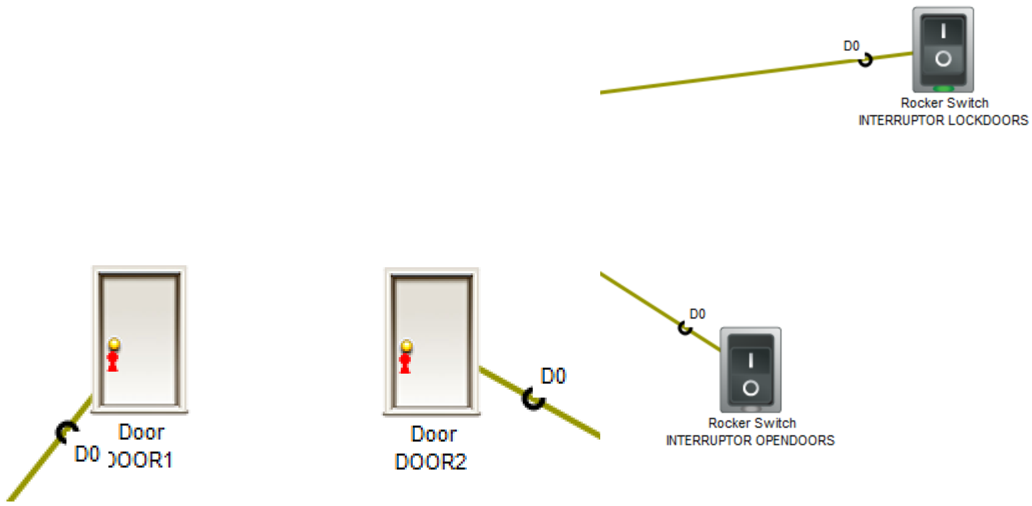


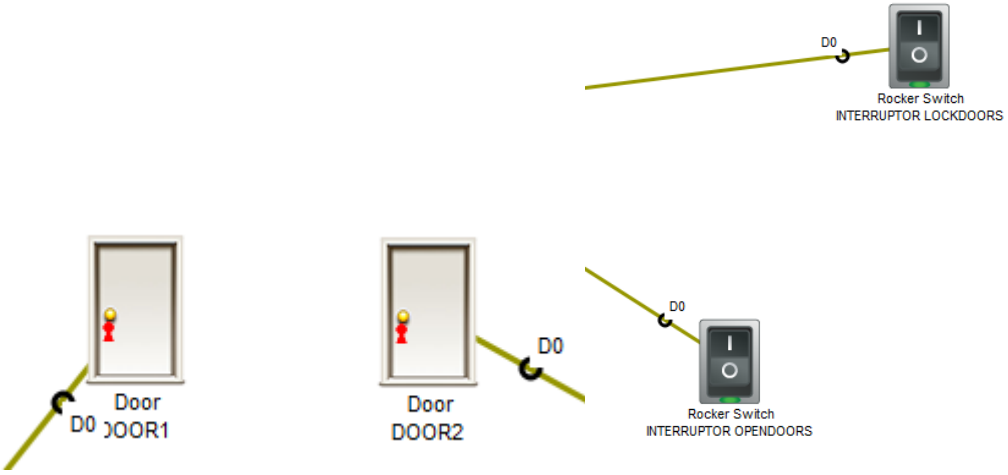
Doors

```
openD = ptmata2.digitalRead(2)
lock = ptmata2.digitalRead(3)
block = False
i = 1
while True:
    while ptmata1.inWaiting() > 0:
        ptmata1.processInput()

    while ptmata2.inWaiting() > 0:
        ptmata2.processInput()

    nouOpen = ptmata2.digitalRead(2)
    nouLock = ptmata2.digitalRead(3)
    if nouOpen != openD or nouLock != lock:
        openD = nouOpen
        lock = nouLock
        ptmata1.customWrite(4, str(openD) + "," + str(lock))
        ptmata2.customWrite(1, str(openD) + "," + str(lock))
```


D0
Rocker Switch
INTERRUPTOR LOCKDOORS
Door
D0
DOOR1
Door
DOOR2
D0
D0
Rocker Switch
INTERRUPTOR OPENDOORS


D0
Rocker Switch
INTERRUPTOR LOCKDOORS
Door
D0
DOOR1
Door
DOOR2
D0
D0
Rocker Switch
INTERRUPTOR OPENDOORS


D0
Rocker Switch
INTERRUPTOR LOCKDOORS
Door
D0
DOOR1
Door
DOOR2
D0
D0
Rocker Switch
INTERRUPTOR OPENDOORS


D0
Rocker Switch
INTERRUPTOR LOCKDOORS
Door
D0)OOR1
Door
DOOR2
D0
D0
Rocker Switch
INTERRUPTOR OPENDOORS

LEDs

```
led = ptmata2.digitalRead(5)
if (led == 1023 and block == False):
    if i != 10:
        print(i)
        ptmata1.analogWrite(1, 1000)
        ptmata2.analogWrite(4, 1000)
        i += 1
        sleep(1)
    else:
        block = True
        ptmata1.analogWrite(1, 0)
        ptmata2.analogWrite(4, 0)
elif (led == 0):
    ptmata1.analogWrite(1, 0)
    ptmata2.analogWrite(4, 0)
    block = False
    i = 1
```