Màster universitari en **Formació del Professorat d'Educació Secundària Obligatòria i Batxillerat, Formació Professional i Ensenyament d'Idiomes**

Treball de fi de màster

Annex 2 – Guia didàctica

Títol: Desenvolupament de material didàctic per a l'aprenentatge del disseny i creació de sistemes informàtics i electrònics

Cognoms: Rivas Contreras

Nom: Santiago

Titulació: Màster en Formació del Professorat d'Educació Secundària Obligatòria i Batxillerat, Formació Professional i Ensenyament d'Idiomes

Especialitat: Tecnologia Industrial

Director/a: Jordi Cosp Vilella

Data de lectura: 17 de Juny de 2020 – 11:10h

ANNEX 2

GUIA DIDÀCTICA

PREFACI

En aquesta guia didàctica pretén ser un ajut al professor per treure el màxim profit de cadascuna de les activitats presentades.

En primer lloc s'explica la dinàmica de treball que caldrà seguir: inicialment cal dividir els alumnes en diferents grups per començar el primer bloc d'activitats. Seguidament es repartiran en tres grups d'experts per fer un segon bloc d'activitats i finalment es tornaran a ajuntar en el grup inicial per a desenvolupar el projecte final.

En cada activitat es detallen quins són els objectius d'aprenentatge, la temporització, si cal una preparació prèvia i quins materials necessitem per dur-la a terme. D'igual forma, es comenten les activitats proposades i s'ofereix un solucionari per a totes aquelles activitats d'ampliació suggerides.

També es fan suggeriments pel desenvolupament de les classes en l'apartat "Consells per a l'aula" on es comenten els possibles problemes en què ens podem trobar, així com aquells aspectes en què cal fer un especial incís, aspectes que cal comentar o evitar, provocar determinades avaries i deixar temes pendents de resoldre per a futures activitats.

NOTA:

Totes les imatges en què no se cita la seva font han estat elaborades per mitjans propis.

Taula de continguts

TAULA DE CONTINGUTS 4

DINÀMICA DE TREBALL. 5
ACTIVITAT 0: PRESENTACIÓ DE COMPONENTS 8
ACTIVITAT 1: FER SERVIR UN PORT DE SORTIDA. 11
ACTIVITAT 2: FER SERVIR UN PORT D'ENTRADA (PART 1) 14
ACTIVITAT 3: FER SERVIR UN PORT D'ENTRADA (PART 2). 16
ACTIVITAT 4: MODULACIÓ PER AMPLE DE POLSOS (PWM) 18
ACTIVITAT 5: SEPARACIÓ DE L'ETAPA DE CONTROL I L'ETAPA DE POTÈNCIA 19
ACTIVITAT 6: CONTROLAR LA VELOCITAT I DIRECCIÓ D'UN MOTOR 21
ACTIVITAT 7: SENSOR D'ULTRASONS 23
ACTIVITAT 8: SENSOR DE GIR I ACCELERACIÓ 25
ACTIVITAT 9: COMUNICACIÓ WEB. INTERNET OF THINGS (IOT) 28
ACTIVITAT 10: PROJECTE. MUNTEM UN VEHICLE AUTÒNOM 34
REFERÈNCIES. 36

Dinàmica de treball.

Puzle d'Aronson

La tècnica del puzle d'Aronson va ser presentada per Elliot Aronson el 1978 al seu llibre *The Jigsaw classroom* després d'haver-la experimentat al llarg de la dècada dels 70.

A la tècnica del puzle d'Aronson, Martínez y Gómez (2010), destaquen diferents objectius tant d'habilitats socials (relació assertiva amb el grup, foment de l'actitud positiva entre els membres de l'equip, desenvolupar la solidaritat i el compromís cívic, atendre la diversitat d'interessos, valors i capacitats de l'alumnat...) com educatius (augment del rendiment acadèmic, millorar l'aprenentatge cooperatiu, aprenentatge significatiu i autodirigit...)

La dinàmica del puzle d'Aronson consisteix, en primer lloc, en explicar al grup quin és l'objectiu final per tal de despertar en ells l'interès i la motivació. Si és la primera vegada, pot ser interessant explicar-los algunes tècniques de treball intel·lectual i habilitats socials per interaccionar amb el grup, alhora que és el mateix grup qui defineix les normes internes de funcionament.

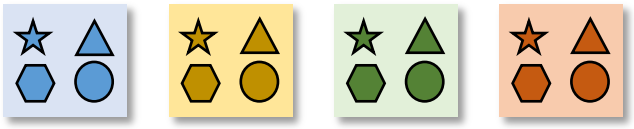
Seguidament es forma el que anomenarem "grups base". Es tracta de dividir la classe en grups heterogenis de quatre o cinc membres. Cal donar temps perquè el grup es conegui, es consolidi i per crear les normes de funcionament. Aquesta consolidació la realitzarem mentre fem les tres primeres activitats.

A continuació es fan públiques les temàtiques en què cal dividir cadascun dels grups bases. Depenent del nombre de temàtiques, els grups assignaran un o dos membres del seu grup per cadascun dels temes que s'hagin d'investigar. Cada membre designat anirà a un nou grup anomenant "grup d'experts" on s'estudiarà a fons la temàtica assignada. Aquests membres s'han de formar en la matèria fins a convertir-se en "experts" per poder transmetre el coneixement adquirit a la resta de membres del grup base. La tornada al grup finalitza amb un informe on s'avalua l'expertesa de cadascun dels membres en totes les matèries.

Fase 1: Creació dels Grups base:

Fase 2: Cada membre d'un grup va a un grup "d'experts" on s'estudia una temàtica concreta:

Fase 3: Un cop formats, els experts tornen als seus grups base:



Aquesta tècnica la farem servir al llarg de tota l'activitat. Hi ha una part introductòria comuna (Activitats 0 a 3), a partir d'aquí es dividiran en 3 grups d'especialitats per tal de fer la resta de pràctiques. Els especialistes del grup 1 faran les activitats 4, 5 i 6 amb l'objectiu de saber controlar un motor de corrent continu. Els especialistes del grup 2 faran les activitats 7 i 8, amb l'objectiu de controlar diferents sensors (ultrasons i giroscopi/acceleròmetre), i finalment, els especialistes del grup 3 s'encarregaran de fer l'activitat 9 d'Internet of Things. D'aquesta manera, es genera la interdependència positiva que és la relació de col·laboració que es genera entre els membres d'un grup de treball quan no poden ser capaços d'arribar als objectius demanats de forma individual si no és amb l'ajut de la resta de l'equip. Aquest efecte positiu multiplica la qualitat dels resultats del grup i fa que el producte final sigui superior a la suma de les parts. (Montanero,2019)

Aprenentatge Basat en Projectes

En el cas d'aquest treball, el punt final del puzle d'Aronson serà, a la vegada, el punt de partida del projecte ABP.

L'aprenentatge basat en problemes (ABP) "és una metodologia d'aprenentatge en la que el punt de partida és un problema o situació que permet a l'estudiant identificar necessitats per comprendre millor el problema o situació. Cal recordar que els problemes són situacions utilitzades com a punt de partida per identificar necessitats d'aprenentatge." [Labrador i Andreu, 2008, traduït del castellà]

Una altra definició de l'ABP pot ser "una estratègia d'instrucció en la qual els estudiants confronten problemes contextualitzats i poc estructurats i cal que s'esforcin per trobar solucions significatives". [Rhem, 2008, traduït de l'anglès]

L'ABP es fa servir per primera vegada a començaments dels anys 70 a l'escola de medicina de la Universitat de McMaster al Canadà. Des de llavors, han estat diversos autors els que l'han estudiat, no només en l'àmbit de la medicina sinó en tota mena d'entorns i disciplines educatives, i han acabat determinant que el nombre òptim d'estudiants per grup es troba entre 4 i 6. De la mateixa manera, indiquen que el professor no ha de tenir un paper directiu sinó que ha de formar part del grup d'aprenentatge. La seva tasca principal és assegurar la progressió dels alumnes i alhora els ha de proporcionar resposta de manera regular i descriptivament sense afegir cap sentit positiu o negatiu a les seves paraules. [Labrador i Andreu, 2008]

No és fins a la darrera pràctica quan entra en joc l'ABP. Als estudiants se'ls dona un enunciat molt clar i simple: construir un vehicle autònom amb connexió IoT. El poden desenvolupar a partir de les eines que han rebut al llarg de les pràctiques anteriors, però se'ls motivarà perquè l'ampliïn i siguin ells mateixos els que acabin de definir l'objectiu del seu treball de forma que puguin incorporar nous dispositius, no treballats fins al moment, per tal de millorar el resultat final.

ACTIVITAT 0: Presentació de components

Objectius d'aprenentatge

- Presentem l'ordinador de placa simple Raspberry Pi 4
- Presentem l'entorn de desenvolupament Thonny Python IDE
- Presentem la placa de proves o *protoboard*

Altres objectius

- Explicar la dinàmica de grups del puzle d'Aronson
- Donar la temporització de les activitats
- Formació dels grups base de treball

Temporització

- 2 hores

Preparació prèvia

La preparació prèvia de la Raspberry Pi 4 consisteix en la descàrrega del sistema operatiu a la targeta MicroSD que fa servir la placa com a disc dur i la seva configuració inicial.

Per preparar-la només cal seguir les instruccions que es troben publicades la web oficial de la Raspberry Foundation:

- Descàrrega del sistema operatiu: https://www.raspberrypi.org/downloads/

  En aquest enllaç podeu trobar diferents sistemes operatius disponibles. Aquest treball s'ha fet instal·lant el sistema operatiu Raspbian en la seva versió més completa: Raspbian Buster with desktop and recommended software

- Instal·lació del sistema operatiu:

  https://www.raspberrypi.org/documentation/installation/installing-images/README.md

Inici i comprovació del sistema

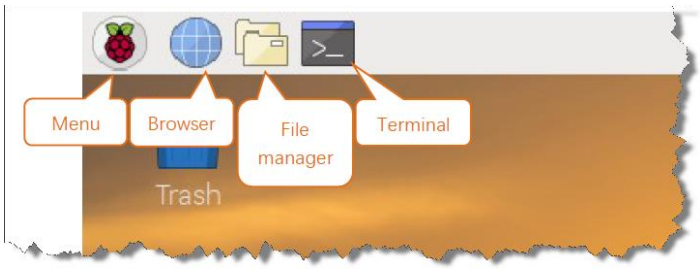
Un cop instal·lat el sistema operatiu, connectarem un teclat, un ratolí i un monitor a la placa per verificar que funciona correctament i procedirem a configurar la connexió sense fils (wifi) o la connexió LAN i l'accés per escriptori remot.

En obrir ens demanarà un usuari i una contrasenya. Per defecte, les credencials inicials són:

- Usuari: pi
- Password: raspberry

Un cop hàgim entrat, veurem l'interface de l'escriptori:

A dalt de tot, a l'esquerra, podem veure les icones principals del menú:



Configuració de la xarxa

Per a configurar la xarxa i establir una IP fixa, podem seguir les instruccions del següent enllaç https://raspberryparanovatos.com/tutoriales/asignar-ip-fija-raspberry-pi/, fer-ho de forma interactiva, o bé mitjançant les següents comandes:

```
sudo nano /etc/dhcpcd.conf
```

I editant el fitxer, canviant els valors destacats en vermell. Recordeu que l'interface eth0 és la connexió a la xarxa per cable Ethernet amb RJ45 i l'interface wlan0 és la connexió per wifi

```
interface eth0
static ip_address=192.168.1.250/24
static routers=192.168.1.1
static domain_name_servers=212.231.6.7 46.6.113.34
```

Habilitar escriptori remot

Per habilitar l'escriptori remot, haurem d'instal·lar prèviament el servei d'escriptori remot xrdp. Per fer-ho cal teclejar la següent comanda:

```
sudo apt-get install xrdp
```

Un cop instal·lat, ens podem connectar des de qualsevol PC amb l'escriptori remot de Windows escrivint directament l'adreça IP que li hem assignat al punt anterior. D'aquesta manera ja no caldrà tenir connectats ni el teclat, ni el ratolí ni el monitor. Com per a qualsevol altra sessió, per iniciar la sessió d'escriptori remot caldrà indicar l'usuari i el password que ja hem comentat al punt anterior.

Instal·lació de llibreries necessàries

Cal instal·lar la llibreria WiringPi. Per fer-ho seguirem els següents passos:

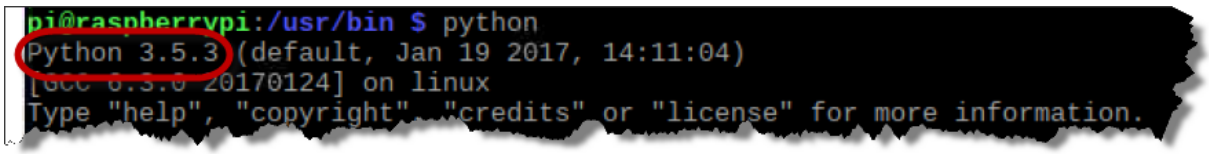
```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install wiringpi
```

Establir Python3

També cal establir Python3 com la versió per defecte de Python:

```
cd /usr/bin
sudo rm python
sudo ln -s python3 python
python
```

La darrera comanda només serveix per verificar que realment la versió 3 és la que tenim per defecte:



Consells per a l'aula

Aquesta sessió consisteix a presentar la placa Raspberry, establir la connexió amb la placa mitjançant l'escriptori remot, accedir a l'entorn d'edició de Python i presentar la placa de proves o *protoboard.*

No es pretén que cap mena d'avaluació però és molt important, ja que estableix les bases per a la resta d'activitats.

Cal explicar:

- la planificació del projecte amb cadascuna de les sessions
- els grups "base" que es faran que han de tenir un mínim de 3 membres i un màxim de 6
- el funcionament dels grups "d'experts" i el retorn a grups base
- la placa Raspberry Pi 4 com connectar-la
- com connectar-se a la Raspberry per escriptori remot
- com entrar a l'entorn de desenvolupament Thonny Python IDE
- la placa de proves o protoboard

ACTIVITAT 1: Fer servir un port de sortida.

Objectius d'aprenentatge

- Fer servir un pin GPIO com a sortida
- Crear un programa senzill amb Python
- Conèixer els nous components: el díode LED, la resistència

Altres objectius

- Comprovar la dinàmica general de les activitats: copiar el programa, executar-lo, analitzar-lo i fer-ne alguna modificació

Temporització

- 2 hores

Consells per a l'aula

Començar explicant la configuració dels pins GPIO i les seves limitacions de tensió i intensitat. Cal insistir també què pot passar si excedim les limitacions (des que els components externs no funcionin fins a fer malbé la placa), i per tant cal estar atents per no sobrepassar-les.

Introduir el component resistència com a limitador bàsic del corrent i explicar el díode LED, vigilant la seva polaritat i posar-lo com a exemple que no podríem encendre gaires LEDs al màxim nivell (20mA) donat que la placa en total pot oferir uns 50mA.

Serà la primera vegada que fan servir la placa de proves o protoboard per la qual cosa és molt important que fem les connexions pas a pas i revisant-les molt bé per tal d'evitar curtcircuits.

Continuar per l'anàlisi del codi i cadascuna de les seves parts. Un cop s'entengui i veient que funciona, passarem a fer l'ampliació.

Activitat 1.2:

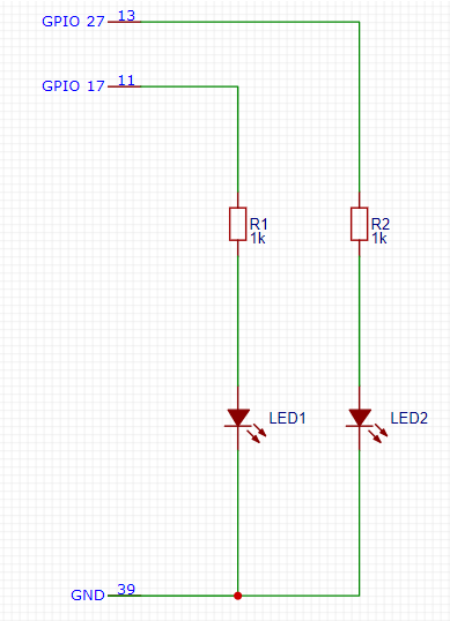
Fer servir dos díodes LED de forma que s'alternin. A partir del circuit i el programa anterior, cal que cada mig segon s'encengui un i s'apagui l'altre.

Els punts principals per fer aquesta activitat són:

- Entendre que caldrà fer servir un altre pin per controlar el segon LED
- Triar el pin adequat
- Fer les connexions correctament pel segon LED
- Fer la programació adequada per alternar els dos LEDs

Solució a l'activitat 1.2

Diagrama



Codi:

```
1.  import RPi.GPIO as GPIO
2.  import time
3.  import sys
4.
5.  PinLED1 = 11     # Farem servir el pin 11 pel LED 1
6.  PinLED2 = 13     # Farem servir el pin 13 pel LED 2
7.
8.  def configuracio_inicial():
9.      GPIO.setmode(GPIO.BOARD)       # Numera els pins segons la posició física
10.     GPIO.setup(PinLED1, GPIO.OUT)  # Estableix el pin del LED 1 com a SORTIDA
11.     GPIO.setup(PinLED2, GPIO.OUT)  # Estableix el pin del LED 2 com a SORTIDA
12.     GPIO.output(PinLED1, GPIO.LOW) # Estableix el valor del pin del LED 1 a BAIX
13.     GPIO.output(PinLED2, GPIO.LOW) # Estableix el valor del pin del LED 2 a BAIX
14.     print ("Fem servir els pins {} i el {}".format(PinLED1, PinLED2) )
15.
16. def proces():
17.     while True:
18.         GPIO.output(PinLED1, GPIO.HIGH)  # Encen el LED 1
19.         GPIO.output(PinLED2, GPIO.LOW)   # Apaga el LED 2
20.         print ('...led1 encés i led2 apagat')
21.         time.sleep(0.5)
22.
23.         GPIO.output(PinLED1, GPIO.LOW)  # Apaga el LED 1
24.         GPIO.output(PinLED2, GPIO.HIGH) # Encen el LED 2
25.         print ('led1 apagat i led2 encés...')
26.         time.sleep(0.5)
27.
28.
29. def destruccio():
30.     GPIO.output(PinLED1, GPIO.LOW)     # Apaga els LEDs
31.     GPIO.output(PinLED2, GPIO.LOW)
32.     GPIO.cleanup()                     # Allibera recursos
33.
34. if __name__ == '__main__':     # El programa comença aquí
35.     configuracio_inicial()
36.     try:
37.         proces()
38.     except KeyboardInterrupt:  # Amb CTRL+C, sortim del programa
39.         destruccio()
```

```
40.     except:
41.         print ("Error inesperat:", sys.exc_info()[0])
42.         destruccio()
43.         raise
44.
```

ACTIVITAT 2: Fer servir un port d'entrada (Part 1)

Objectius d'aprenentatge

- Fer servir un pin GPIO com a entrada
- Conèixer els nous components: el polsador o botó

Altres objectius

- Fer servir estratègies de deducció i resolució de problemes

Temporització

- 2 hores

Consells per a l'aula

Començar explicant la configuració dels pins d'entrada i tornat a fer esment que cal protegir els pins d'entrada dels corrents per no fer malbé la placa.

El botó té molt poca explicació i caldria explicar-ho inicialment com un botó perfecte, sense soroll o rebots, ja que això es provocarà a l'activitat 2.2 i es resoldrà a l'activitat 3

Al final de l'activitat 2.2 cal que deixem oberta la pregunta de per què succeeix i com es resoldria.

Activitat 2.2:

Modificar el programa de forma que en polsar el botó el LED quedi encès i no s'apagui fins que no el tornem a polsar de nou.

Per fer-ho, no cal modificar el circuit, i només caldrà modificar la funció proces() per tal que gestioni els esdeveniments del sistema, els quals seran cíclics.

Cal que forcem un funcionament incorrecte del sistema mitjançant el soroll de botó per tal d'introduir la següent activitat. Ho podrem fer prement i deixant anar el botó ràpidament.

```
45. estatSistema = 0  # Els estats del sistema son cíclics:
46.                   # 0 - led OFF, botó OFF
47.                   # 1 - led ON, botó ON
48.                   # 2 - led ON, botó OFF
49.                   # 3 - led OFF, botó ON
50.
51.
52. def proces():
53.     estatSistema = 0
54.     while True:
55.         if GPIO.input(pinBoto)==GPIO.LOW:   # Si hem pres el botó...
56.             if estatSistema == 0:
57.                 estatSistema = 1
58.             elif estatSistema == 2:
59.                 estatSistema = 3
60.         elif GPIO.input(pinBoto) == GPIO.HIGH: # Si el botó no està pres
61.              if estatSistema == 1:
```

```
62.                 estatSistema = 2
63.             elif estatSistema == 3:
64.                 estatSistema = 0
65.         if (estatSistema == 1 or estatSistema == 2):
66.             GPIO.output(pinLED,GPIO.HIGH)    # Encenem el LED
67.             print ('led on ...')
68.         else :
69.             GPIO.output(pinLED,GPIO.LOW)     # Apaguem el LED
70.             print ('led off ...')
```

ACTIVITAT 3: Fer servir un port d'entrada (Part 2).

Objectius d'aprenentatge

- Gestionar el soroll dels dispositius d'entrada
- Gestionar els ports d'entrada mitjançant el registre d'esdeveniments

Altres objectius

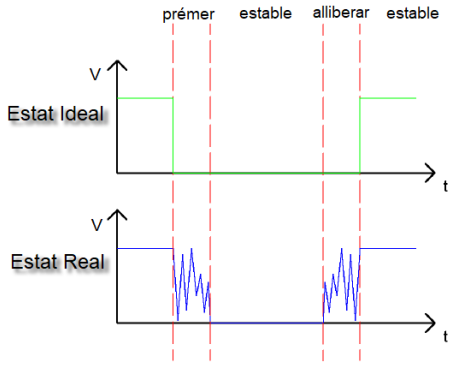
- Entendre que, de vegades, no hi ha solucions perfectes ni universals i que llavors s'ha de buscar una solució de compromís.

Temporització

- 2 hores

Consells per a l'aula

Explicar la diferència entre el senyal real d'un sensor i el senyal ideal. Les transicions entre estats poden tenir soroll o rebots.



Explicar també que un bucle infinit no és la millor forma de controlar un sensor i sempre que sigui possible haurem de programar un esdeveniment sobre el pin que controla el sensor per tal d'optimitzar els recursos.

Activitat 3.2:

Modificar el programa de forma que el LED canviï d'estat quan s'allibera el botó en compte de quan es polsa.

Per fer-ho, només hem de registrar l'esdeveniment amb GPIO.RISING en comptes de GPIO.FALLING

```
14. #Registre de l'event de pulsació del botó
15. GPIO.add_event_detect(pinBoto,GPIO.RISING,callback = eventBoto,bouncetime=300)
```

Activitat 3.3:

Modificar el programa de forma que el LED canviï d'estat tant quan es polsa com quan s'allibera.

Per fer-ho, només hem de registrar l'esdeveniment amb GPIO.BOTH en comptes de GPIO.FALLING o GPIO.RISING

```
14. #Registre de l'event de pulsació del botó
15. GPIO.add_event_detect(pinBoto,GPIO.BOTH,callback = eventBoto,bouncetime=300)
```

Què succeeix si es polsa el botó i es deixa anar molt ràpid? Per què? Juga amb el valor del bouncetime i observa com afecta el funcionament.

Si el botó es polsa i es deixa anar massa ràpid s'ignora l'alliberament del botó. Això succeeix perquè el paràmetre bouncetime té un valor massa alt. Si juguem amb ell i posem un valor massa baix, llavors pot ser que comencem a tenir compte el soroll. Si el deixem massa alt, pot ser que ignorem un canvi de valor del sensor que passi massa ràpid. Per tant, hem de trobar un punt adequat entre aquests dos casos.

Comentar també que es podria buscar una solució "hardware" fent ús d'un condensador que suavitzes el soroll que es produeix. El problema d'aquesta solució és que alenteix la reacció que tenim al sensor i que, en alguns casos, pot no ser una solució adequada.

ACTIVITAT 4: Modulació per ample de polsos (PWM)

Objectius d'aprenentatge

- Gestionar un dispositiu analògic mitjançant PWM

Altres objectius

- Experimentar i deduir a partir de l'observació

Temporització

- 2 hores

Consells per a l'aula

Donar les pautes bàsiques per explicar la tècnica del PWM (Pulse Width Modulation). Caldria que fessin una petita investigació en grup abans de començar a programar-lo.

Activitat 4.2:

Prova amb diferents freqüències i temps d'espera entre un valor i un altre. Què succeeix amb una freqüència de 20 Hz i un temps d'espera de 10 ms?

Amb una freqüència molt alta, no s'aprecia el canvi d'intensitat. Amb una molt baixa, es veu com fa pampallugues el LED.

ACTIVITAT 5: Separació de l'etapa de control i l'etapa de potència

Objectius d'aprenentatge

- Aprendre a gestionar dispositius que requereixen més potència de la que pot generar un pin GPIO mitjançant la separació de l'etapa de control i l'etapa de potència
- Utilitzar el transistor com amplificador

Altres objectius

- Experimentar i deduir a partir de l'observació

Temporització

- 4 hores

Consells per a l'aula

Explicar la necessitat de separar el control de la potència. No només amb un dispositiu que necessiti més potència de la que pot donar un pin de sortida, sinó fins i tot, un dispositiu de 220V amb un relé.

Explicar com funciona el transistor i donar l'esquema bàsic de l'amplificador amb un transistor NPN.

Explicar els dos tipus de brunzidors: l'actiu i el passiu. L'actiu emet un so constant quan s'aplica tensió. El passiu emet un so en funció de la freqüència del senyal aplicat.

Activitat 5.2:

Sobre el circuit anterior, canviar el brunzidor actiu per un de passiu i prova'l. Què succeeix? Per què?

El brunzidor passiu emet un senyal en funció de la freqüència que li arriba. En aquest cas li està arribant un pols simple que és el senyal que podem realitzar amb un polsador. El so que sentim és la interpretació d'aquest pols simple.

Activitat 5.3:

El brunzidor passiu requereix ser controlat per un PWM i emetrà un senyal en funció de la freqüència. En aquest cas l'ample del pols ens serveix per controlar el volum. Canvia el programa per fer servir el brunzidor passiu.

Es tracta de fer servir el que vam veure a l'activitat 4. En primer lloc hem de crear el pols PWM

```
7.
8.  def configuracio_inicial():
9.      global pols
10. […]
16.     pols = GPIO.PWM(pinBuzzer, 1)
17.     pols.start(0)
18.
```

A continuació controlarem el botó. Si està pres cridarem a la funció que fa sonar el brunzidor i si no està pres, l'aturarem.

```
19. def proces():
20.     while True:
21.         if GPIO.input(pinBoto)==GPIO.LOW:   # Si hem pres el botó...
22.             alarma()
23.             print ('buzzer on ...')
24.         else :
25.             stopAlarma()
26.             print ('buzzer off ...')
27.
```

Seguidament, definim les funcions d'aturar l'alarma i la de fer-la sonar. En aquest cas he posat dos exemples: el primer que simula una sirena i el segon que fa una escala musical.

```
28. def stopAlarma():
29.     pols.stop()
30.
31. def alarma1():
32.     pols.start(50)
33.     for x in range(0,361): #generem una funció sinusoidal
34.         sinVal = math.sin(x * (math.pi / 180.0)) #Calcula el sinus
35.         toneVal = 2000 + sinVal * 500 #Afegeix a una base i l'amplifica
36.         pols.ChangeFrequency(toneVal) #Estableix la frequencia
37.         time.sleep(0.001)
38.
39. def escala_musical():
40.     pols.ChangeFrequency(261.63)    # DO
41.     time.sleep(0.25)
42.     pols.ChangeFrequency(293.66)    # RE
43.     time.sleep(0.25)
44.     pols.ChangeFrequency(329.63)    # MI
45.     time.sleep(0.25)
46.     pols.ChangeFrequency(349.23)    # FA
47.     time.sleep(0.25)
48.     pols.ChangeFrequency(391.99)    # SOL
49.     time.sleep(0.25)
50.     pols.ChangeFrequency(440.00)    # LA
51.     time.sleep(0.25)
52.     pols.ChangeFrequency(493.88)    # SI
53.     time.sleep(0.25)
54.
```

ACTIVITAT 6: Controlar la velocitat i direcció d'un motor

Objectius d'aprenentatge

- Aprendre a gestionar un motor de corrent continu, controlant la seva velocitat i el sentit de gir fent servir la separació de l'etapa de control i l'etapa de potència

Altres objectius

- Conèixer els nous components: Motor, Potenciòmetre, DAC PCF8591 i Driver L293D
- Començar a idear el dispositiu autònom on gestionarem la velocitat i sentit de gir sense intervenció externa

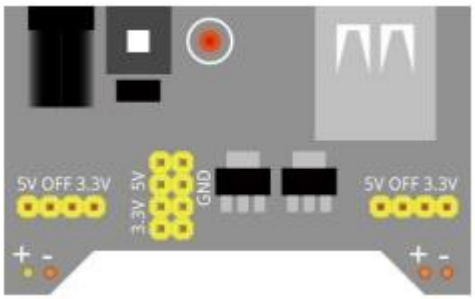
Temporització

- 4 hores

Consells per a l'aula

Explicar cadascun dels nous dispositius i separar les etapes del circuit deixant ben clar per a què serveix cadascun dels blocs: ADC per llegir la posició del potenciòmetre, L293D per controlar el motor i PWM per controlar la velocitat.

Explicar també que l'alimentació serà externa i mai de la Raspberry. Si no es disposa del Breadboard Power Module, ens hauríem de fabricar un de senzill o bé fer servir només l'alimentació de control de la Raspberry Pi (3.3V), però assegurant que la tensió que apliquem al pin 8 (Power VCC for Drivers) prové d'una font externa.



Breadboard Power Module

Explicar cadascuna de les parts del codi i entendre la seva interacció amb el diagrama de blocs.

Activitats

Activitat 6.2:

Fem servir el DAC PCF8591T només per llegir el valor del potenciòmetre i establir el sentit de gir i la velocitat del motor. Com ho faries perquè un motor pogués variar la velocitat i el sentit de marxa sense que ningú actués sobre el potenciòmetre?

Només caldria modificar la funció proces(). En comptes de llegir un valor del port sèrie connectat al ADC el que fem és un bucle que va del valor 0 al 256 seguit d'un altre bucle que va del 256 fins a 0 en sentit descendent.

```
50.      def proces():
51.          while True:
52.              for valor in range(0,256,1):
53.                  print ('ADC Valor : %d'%(valor))
54.                  motor(valor)
55.                  time.sleep(0.01)
56.              for valor in range (256,0,-1):
57.                  print ('ADC Valor : %d'%(valor))
58.                  motor(valor)
59.                  time.sleep(0.01)
```

D'aquesta manera podem comprovar que podem prescindir del potenciòmetre i del DAC en un vehicle autònom, què és l'objectiu final d'aquest projecte.

ACTIVITAT 7: Sensor d'ultrasons

Objectius d'aprenentatge

- Aprendre a gestionar un sensor d'ultrasons per calcular la distància fins a un obstacle

Altres objectius

- Recordar coneixements de física elemental que poden influir en el desenvolupament del nostre projecte
- Resoldre problemes per la via pràctica
- Pensar com incorporar nous sensors per tal de perfeccionar el nostre projecte

Temporització

- 4 hores

Consells per a l'aula

És important que s'entenguin els principis de la física sobre els que funciona el sensor d'ultrasons i com calculem la distància a un objecte calculant el temps que triguen les ones a anar fins a l'objecte, rebotar i tornar al sensor.

El sensor té un funcionament molt senzill i queda perfectament explicat al diagrama de temps (Fig.13) a la fitxa de l'alumne.

Les dues funcions principals del codi són getSonar i PulseIn. Cal aprofundir en la seva explicació per tal que entenguin el funcionament i siguin capaços de respondre a l'ampliació de les activitats.

Activitats

Activitat 7.2:

Com podem fer per ajustar el sensor de forma que ens detecti els objectes que hi ha en un rang entre 75 y 125 cm?

Aquesta és una activitat per posar a prova el sentit pràctic de resolució de problemes per part dels alumnes. Només ens hem de preocupar de dues coses:

1.- Assegurar-nos que el rang màxim arriba fins als 125 cm (1250 mm)

2.- Ignorar totes aquelles lectures inferiors a 75 cm (750 mm)

I això ho farem modificant el valor de MAX_DISTANCE a 125

```
7.  MAX_DISTANCE = 125          #defineix la distància màxima a mesurar
```

I ignorant els valors llegits que siguin inferiors a 750, comparant el valor llegit amb el límit inferior desitjat que és 750 (línia 40)

```
37. def proces():
38.     while(True):
39.         distancia = getSonar()
40.         if (distancia > 750):
41.             print ("Distància a l'obstacle : %.2f mm"%(distancia))
42.         else:
43.             print ("No s'ha detectat cap obstacle")
44.         time.sleep(1)
```

Activitat 7.3:

Els càlculs de la velocitat de les ones ultrasòniques s'han fet d'acord amb l'assumpció que la velocitat del so és de 343 m/s, però realment la velocitat del so per l'aire depèn, entre altres factors, de la temperatura. On hauríem d'actuar per calibrar correctament el sensor de proximitat? I si volguéssim una calibració automàtica, com ho hauríem de fer?

Per tal de calibrar correctament el sensor de proximitat caldria calcular la velocitat real del so en les condicions ambientals de temperatura i humitat actuals. Un cop tinguéssim aquesta dada, caldria modificar el valor de la constant MICROSEGONS_A_MILIMETRES que ara tenim establert a 0.1715 a conseqüència del càlcul que hem determinat arran de suposar una velocitat del so de 343 m/s

Si volguéssim fer una calibració automàtica, bastaria amb tenir un sensor de temperatura i ajustar la constant amb una senzilla fórmula:

$$vs \approx 331.4 + 0.61 \cdot t$$

On 331,4 és la velocitat de l'aire a 0º C i t és la temperatura de l'aire.

ACTIVITAT 8: Sensor de gir i acceleració

Objectius d'aprenentatge

- Aprendre a gestionar un sensor de gir i acceleració
- Comprendre la necessitat de calibratge d'alguns sensors i fer-ne un

Altres objectius

- Refrescar coneixements de trigonometria que poden influir en el desenvolupament del nostre projecte
- Resoldre problemes per la via pràctica
- Dissenyar una solució per obtenir l'angle de gir sobre l'eix Z

Temporització

- 6 hores

Consells per a l'aula

És important que s'entenguin els principis de la física sobre els que funciona el sensor de gir i acceleració per tal de poder comprendre el funcionament del sensor i poder donar una correcta solució a l'apartat 8.2

El sensor té un funcionament molt senzill si es tenen clars aquests conceptes. Potser no ens dóna les dades del tot elaborades, però ens costarà poc esforç obtenir-les al gust del nostre projecte.

També és important aprendre a interpretar els valors que ens dóna el sensor, entendre les limitacions físiques del mateix i que cal traslladar aquest valor a l'escala seleccionada tenint en compte la precisió que hem configurat.

Activitats

Activitat 8.1:

Implementar el circuit sobre un protoboard. Transcriure el programa i executar-lo verificant que tots els sensors funcionen correctament.

Aquí podem observar la necessitat de realitzar la calibració dels sensors, donat que el giroscopi està donant uns valors propers a zero, però que oscil·len al voltant d'aquest. L'acceleròmetre, en canvi, dóna uns valors propers a zero pels eixos X i Y, i proper a un per a l'eix Z. Tots aquests valors haurien de donar zero en un sensor ideal. L'eix Z és diferent donat que en repòs té una acceleració cap amunt d'1g que es compensa amb la gravetat.

```
Temperatura: 28.20 ºC
---------
Dades del giroscopi
gyro_x: -248  map: -1.8921
gyro_y:   -4  map: -0.0305
gyro_z: -163  map: -1.2436
------------------
Dades de l'acceleròmetre
accel_x:    708  map:  +0.0432
accel_y:   -200  map:  -0.0122
accel_z:  14912  map:  +0.9102
------------------
Dades calculades de rotació
Rotació x:  -2.7180
Rotació y:  -0.7675
Rotació z:  +87.1755
```

Pel que fa a les dades calculades de la rotació, no donen zero donat que el sensor no s'ha calibrat.

Es pot experimentar amb el sensor movent-lo i girant-lo en els tres eixos i comprovant que funciona correctament... excepte per la rotació de l'eix Z que sembla que no fa res. Ho veurem en la següent activitat.

Activitat 8.2:

Què succeeix amb el valor que obtenim per la rotació de l'eix Z? Per què? Com ho podem solucionar?

El càlcul trigonomètric que hem fet per calcular la rotació dels eixos X i Y es fa prenent com a referència el vector de la gravetat. Això no funciona amb un pla perpendicular a l'eix Z, donat que el seu vector de referència no canvia.

Llavors, com podem calcular el gir sobre l'eix Z? Ens caldria un vector de referència que no fos la gravetat i aquest podria ser el nord magnètic. Altres sensors incorporen una brúixola interna i ens podrien donar l'orientació del dispositiu, però aquest no és el cas.

Haurem de calcular el gir acumulat llegint la velocitat de gir de l'eix Z i mesurant el temps entre lectura i lectura. La velocitat de gir s'expressa en º/seg, si ho multipliquem per una mesura de temps en segons, ens quedarà la quantitat de graus que ha girat. Hem de recordar que no ens interessa l'orientació (si està enfocat cap al Nord, o l'Est...) sinó el gir acumulat des que l'hem iniciat. En qualsevol moment podem reinicialitzar el gir acumulat i tornar a acumular des de la posició actual.

A més, per a tenir una mesura més fiable, caldrà fer una calibració inicial del sensor. Tot això ho farem de la següent manera:

Definirem dues variables globals: una per gestionar el gir acumular sobre l'eix z i un altre per calcular el factor de calibració.

```
14. angle_z = 0
15. error_z = 0
```

El calibratge el realitzarem a la configuració inicial o cada vegada que ho estimem oportú. Definirem una funció calibrar_z que farà 300 lectures seguides del registre 0x47 (gyro_z) i n'obtindrà la mitja. Aquest valor se li restarà al valor que s'obtingui cada vegada que llegim aquest registre.

```
55. def calibrar_z():
56.     global error_z
57.     #Calibració de l'eix z
58.     for x in range(0, 300):
59.         error_z = error_z + read_word_2c(0x47)
60.     error_z = error_z/300
```

Cada vegada que llegim el registre ens guardem el temps anterior i el temps actual:

```
81.         temps_anterior = temps_nou
82.         gyro_z = read_word_2c(0x47)
83.         temps_nou = time.time()
```

I calculem l'angle de gir de la darrera lectura i l'acumulem:

```
90.         #Eix Z calculat en base al giroscopi")
91.        angle_z = angle_z + ((gyro_z - error_z) * escala_giro) * (temps_nou - temps_anterior)
```

ACTIVITAT 9: Comunicació web. Internet of Things (IoT)

Objectius d'aprenentatge

- Aprendre a comunicar la Raspberry Pi amb un servidor web per tal d'enviar i rebre informació

Altres objectius

- Conèixer el protocol SOAP
- Conèixer i fer servir els documents XML, els parsers i ser conscients de la seva flexibilitat

Temporització

- 10 hores

Consells per a l'aula

Aquesta activitat no fa servir cap component electrònic i ni tan sols fa servir el port GPIO de la Raspberry. Se centra únicament en la programació d'un client web amb Python per tal de comunicar-se amb un servei web mitjançant el protocol SOAP. A més, caldrà desenvolupar la part del servei web on poder fer la crida que proposarem que sigui amb VisualBasic.NET

A partir d'un webservice senzill (HelloWorld) per provar la connectivitat i la recepció de la resposta es proposarà l'ampliació d'aquest servei i client per tal d'acceptar paràmetres i finalment la creació d'un webservice genèric que rebi un document XML amb les especificacions del treball a realitzar.

Activitats

Activitat 9.1:

Transcriure els programes i executar-lo verificant el correcte funcionament.

Activitat 9.2:

Programa un servei web que rebi dos paràmetres com a nombres enters, els sumi i torni el resultat com un nombre enter. Fes també el programa en Python per tal de fer la crida a aquest servei web i rebre el resultat.

Caldrà afegir el nou mètode al servei web:

```
1.      <WebMethod()>
2.      Public Function Test_Suma(x As Integer, y As Integer) As Integer
3.          Return x + y
4.      End Function
5.
```

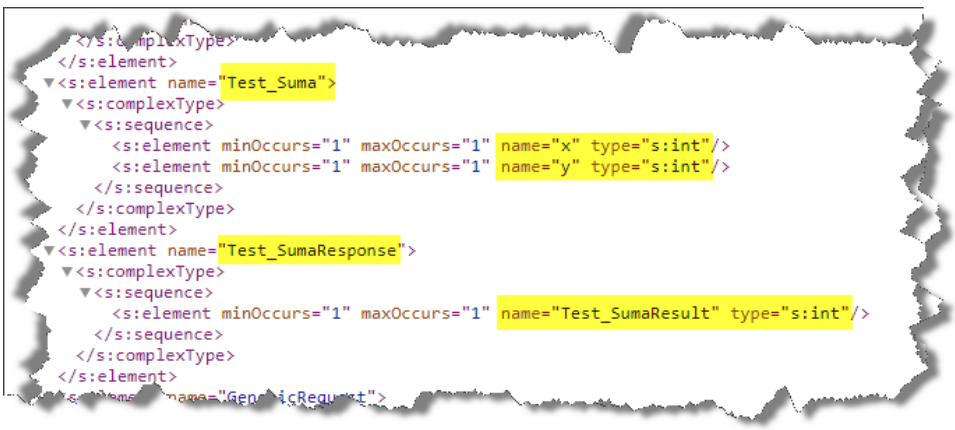
I adaptar el client en Python:

```
1.  import requests
2.  import sys
```

```
from xml.dom.minidom import parse, parseString

endpoint = "http://192.168.1.37:8080/WSRequest/RaspberryRequest.asmx?wsdl"
headers = {'content-type': 'application/soap+xml'}
body="""<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-
envelope">
 <soap12:Body>
   <Test Suma xmlns="http://mytests.cat/">
     <x>10</x>
     <y>25</y>
   </Test_Suma>
   </soap12:Body>
</soap12:Envelope>"""


if __name__ == '__main__':      # El programa comença aquí
    try:

        print(body)

        print ("***Resposta***")
        response = requests.post(endpoint, data=body, headers=headers)
        x=response.content
        x=str(x,"UTF-8")
        print(x)

        print ("Parse...")
        dom = parseString(x)

        print ("getElements...")
        itemlist = dom.getElementsByTagName('Test_SumaResult')
        print("Quantitat d'elements: " + str(len(itemlist)))

        print ("itemList de Test_SumaResult...")
        print(itemlist[0])
        print("item.nodeValue: " + itemlist[0].childNodes[0].nodeValue)

    except:
        print ("Error inesperat:", sys.exc_info()[0])

```

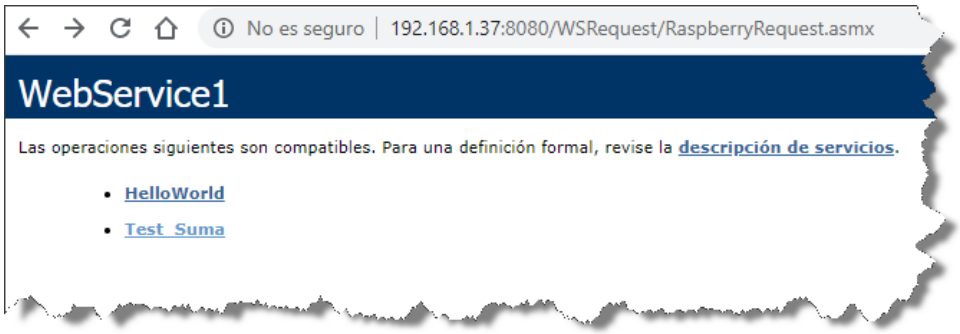
En aquest programa podem veure que bàsicament canvien dues coses: el contingut del "body" que ara s'adapta a la crida del mètode "Test_Suma" i als seus paràmetres i, per una altra banda, que la resposta l'hem d'anar a buscar a l'element "Test_SumaResult".

Com podem saber l'estructura que requereix un nou mètode? Hi ha dues maneres: la primera és consultar el fitxer wsdl que tenim indicat a la constant *endpoint*. Només cal que posem aquesta URL a un navegador web i veurem la definició del webservice:



A la definició del webservice, haurem d'identificar el mètode i la seva resposta i anar desxifrant cadascun dels seus paràmetres i tipus.

Una altra forma més senzilla d'esbrinar-ho és anar al navegador, copiar la URL que hem definit a la constant *endpoint*, però sense el `?wsdl` . Aquesta acció ens mostrarà una pantalla com la següent:



Aquí haurem de seleccionar el mètode que volem consultar, fem clic en ell i ens mostrarà un exemple tant del document d'entrada com del de sortida amb què treballa aquest mètode:

Activitat 9.3:

Programar un servei web genèric. Aquest servei web ha de rebre com a únic paràmetre un document XML. Aquest document XML ha de contenir la comanda a realitzar i els paràmetres que siguin necessaris. Com a resultat retornarà un altre document XML amb el resultat de l'operació (OK o KO), una descripció de l'error si ha anat malament o bé el resultat demanat si ha funcionat correctament.

En primer lloc, hem de definir el missatge genèric que li volem passar. En aquest cas és molt senzill i només volem que ens faci la suma de dos nombres enters:

```
<missatge>
    <operacio>Sumar</operacio>
    <x>12</x>
    <y>16</y>
</missatge>
```

A continuació, i d'acord amb al missatge genèric que hem definit abans, definirem el mètode web per rebre les crides genèriques. L'únic paràmetre que rep és un string que esperem que sigui un document XML amb l'estructura anterior:

```
<WebMethod()>
Public Function GenericRequest(strMessage As String) As String
    Dim strResult As String = ""

    Try
        Dim doc As New XmlDocument
        Dim oNode As XmlNode

        doc.LoadXml(strMessage)
        oNode = doc.SelectSingleNode("missatge/operacio")

        If oNode Is Nothing Then
            strResult = Utils.EscriuResposta("KO", "No hi ha operació al missatge")
        ElseIf oNode.InnerText = "Sumar" Then
            Dim x, y As Integer
            x = CInt(doc.SelectSingleNode("missatge/x").InnerText)
            y = CInt(doc.SelectSingleNode("missatge/y").InnerText)
            strResult = Utils.EscriuResposta("OK", Test_Suma(x, y))
        End If
    Catch e As Exception
        strResult = Utils.EscriuResposta("KO", e.Message)
    End Try

    Return strResult
End Function
```

Aquest mètode busca el contingut missatge/operació dins del document XML, i en funció del contingut farà una cosa o una altra. Es basa en una llibreria d'utilitats "Utils" que hem definit així:

```
Public Class Utils

  Private Shared Function FormatResposta() As String
     Return "<?xml version=""1.0"" ?>" &
            "<Resposta>" &
                 "<Resultat>#RESP#</Resultat>" &
                 "<Descripcio>#DESC#</Descripcio>" &
            "</Resposta>"
  End Function

  Public Shared Function EscriuResposta(strResult As String,
strDescription As String) As String
     Dim strResposta As String = FormatResposta()
     Try
        If strResult.ToUpper = "OK" Then
           strResposta = strResposta.Replace("#RESP#", "OK").Replace("#DESC#",
strDescription)
        ElseIf strResult.ToUpper = "KO" Then
           strResposta = strResposta.Replace("#RESP#", "KO").Replace("#DESC#",
strDescription)
        Else
```

```
            strResposta = strResposta.Replace("#RESP#", "KO").Replace("#DESC#", "Codi de resultat incorrecte:" &
                                strResult & " Descripció: " & strDescription)
        End If
    Catch ex As Exception
        strResposta = FormatResposta().Replace("#RESP#", "KO").Replace("#DESC#", ex.Message)
    End Try
    Return strResposta
  End Function

End Class
```

Conté l'estructura d'un missatge genèric que anirà omplint amb els valors que necessiti a cada moment.

I finalment el codi en Phyton que consumeix aquest nou webservice:

```
import requests
import sys
from xml.dom.minidom import parse, parseString

endpoint = "http://192.168.1.37:8080/WSRequest/RaspberryRequest.asmx?wsdl"

body ="""<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
 <soap12:Body>
   <GenericRequest xmlns="http://mytests.cat/">
     <strMessage><![CDATA[MISSATGE]]></strMessage>
   </GenericRequest>
   </soap12:Body>
</soap12:Envelope>"""

missatge = """<missatge>
                <operacio>Sumar</operacio>
                <x>12</x>
                <y>16</y>
              </missatge>"""

headers = {'content-type': 'application/soap+xml'}
body = body.replace("MISSATGE",missatge)

if __name__ == '__main__':     # El programa comença aquí
    try:

        print(body)

        print ("-----")
        response = requests.post(endpoint, data=body, headers=headers)
        x=response.content
        x=str(x,"UTF-8")
        print(x)

        print ("Parse...")
        dom = parseString(x)

        print ("getElements...")
        itemlist = dom.getElementsByTagName('GenericRequestResult')
        print(len(itemlist))

        print ("itemList de GenericRequestResult...")
        print(itemlist[0])
        print("item.nodeValue" + itemlist[0].childNodes[0].nodeValue)

        dom2 = parseString(itemlist[0].childNodes[0].nodeValue)

        print("---- Resultat -----")
        itemlist = dom2.getElementsByTagName('Resultat')
        print("Resultat: " + itemlist[0].childNodes[0].nodeValue)

```

```
53.         print("---- Descripció -----")
54.         itemlist = dom2.getElementsByTagName('Descripcio')
55.         print("Descripció: " + itemlist[0].childNodes[0].nodeValue)
56.     except:
57.         print ("Error inesperat:", sys.exc_info()[0])
```

El tractament de la resposta no varia gaire respecte del que ja teníem anteriorment. Només cal buscar el valor en el node corresponent.

El que sí que varia una mica, però és fonamental, és que en enviar un document XML com a paràmetre cal incloure dins d'una secció CDATA per evitar que es tracti el XML que es vol fer arribar com a paràmetre. Això ho podem veure a la línia 11

```
7.  <soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-
    envelope">
8.   <soap12:Body>
9.     <GenericRequest xmlns="http://mytests.cat/">
10.      <strMessage><![CDATA[MISSATGE]]></strMessage>
11.    </GenericRequest>
12.    </soap12:Body>
13. </soap12:Envelope>"""
14.
```

Cal fer veure als alumnes la flexibilitat que dóna definir la interfície en un document XML de cara a ampliacions o actualitzacions en la capa del servidor: convé que el protocol es mantingui congelat per afavorir la comptabilitat amb dispositius antics. Això ens ho pot donar d'una manera molt fàcil fer servir un mètode genèric i anar canviant la definició del document, si és que cal, per a noves funcions.

Com a ampliació i per a reforçar la idea d'aquesta flexibilitat, es pot demanar als estudiants que facin el desenvolupament de la funció "Resta" mitjançant la funció "GenericRequest" o bé que defineixin un nou mètode webservice "Resta" i que comparin la facilitat de manteniment de cadascuna de les dues estratègies.

ACTIVITAT 10: Projecte. Muntem un vehicle autònom

Objectius d'aprenentatge

- Recopilar tots els coneixements adquirits al llarg d'aquestes pràctiques en un únic projecte
- Dissenyar un vehicle que funcioni de manera autònoma

Enunciat

Amb l'ajuda d'un ordinador Raspberry Pi i amb el coneixement adquirit durant les activitats 1 a 9 cal que dissenyeu, per grups, un vehicle que funcioni de manera autònoma i que tingui, com a mínim, les següents característiques:

- Cal que avanci a tota velocitat fins que detecti un obstacle a menys d'un metre
- Quan hagi detectat un obstacle a menys d'un metre, ha de reduir la velocitat fins que es trobi a menys de 5 cm de l'obstacle
- Quan es trobi a 5 cm de l'obstacle cal que giri 90º a la dreta i segueixi avançant a tota velocitat
- En tot moment ha d'anar enviant dades al servidor de
    - Temperatura
    - Distància a l'obstacle
    - Potència aplicada als motors (en percentatge)
- En qualsevol moment pot rebre ordres des del servidor que cal complir immediatament:
    - GD: Gira a la dreta 90º
    - GE: Gira a l'esquerra 90º
    - START: Funcionament normal en mode autònom
    - STOP: Aturar-se. Només queda a l'espera de comandes des del servidor
    - HIGH: 100 % de potència màxima als motors
    - LOW: 75% de potència màxima als motors

A més del vehicle autònom cal programar:

- Un servidor web que centralitzi les peticions dels vehicles
- Un client que mostri les darreres dades que ha enviat un vehicle i que permeti enviar ordres cap al vehicle.

Temporització

- 40 hores

Consells per a l'aula

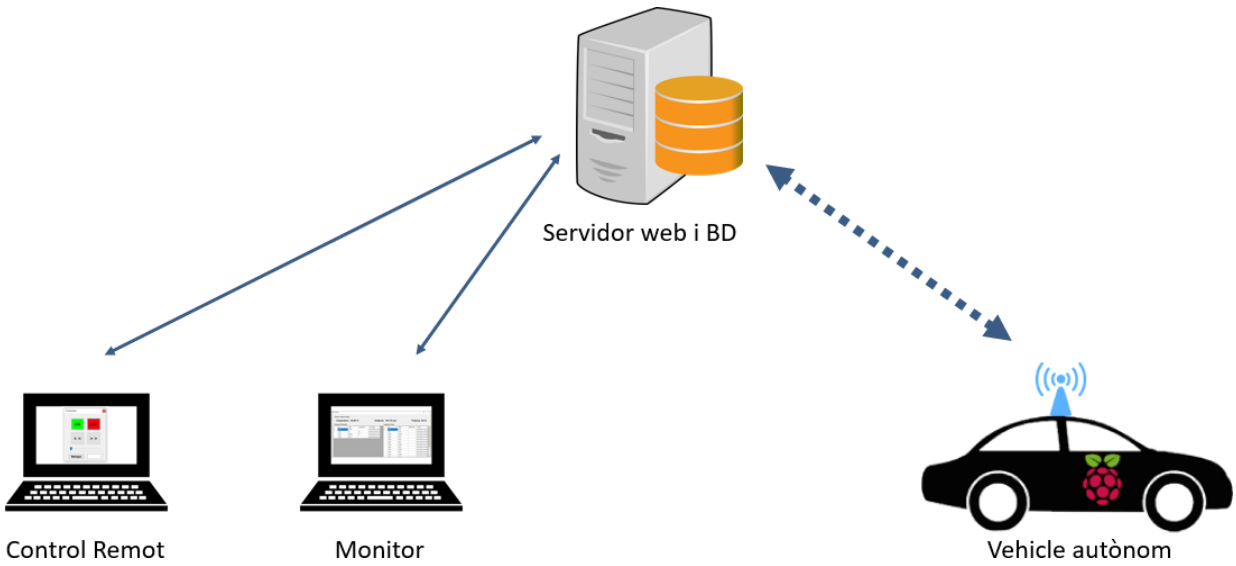
- Proveir un xassís on poder implantar la Raspberry i els circuits. Proposta: https://www.amazon.es/gp/product/B07F73738Z/ref=ppx_yo_dt_b_asin_title_o05_s00?ie=UTF8&psc=1
- Cal pensar com s'alimentarà la Raspberry en mode autònom. Proposta: Carregador extern de mòbil. Amb un de 2000 mAh s'ha aconseguit una autonomia de més d'1h.

- Cal pensar com s'alimentaran els motors. Proposta: bateria 9V externa exclusiva per als motors. Amb 5V els motors s'aconsegueixen moure en buit, però no amb el pes del xassís, bateries, circuits i la placa Raspberry.
- Cal pensar a economitzar energia dels motors:
    - Fer servir només dos motors
    - Fer servir l'acceleròmetre per arrencar a plena potència, però baixar-la un cop hem iniciat el moviment, cosa que també facilitarà la frenada

Solució

A l'annex 3 podem trobar el codi font amb una proposta de solució per al vehicle autònom i la resta de sistemes involucrats.

El disseny de blocs de tot el sistema seria el següent:



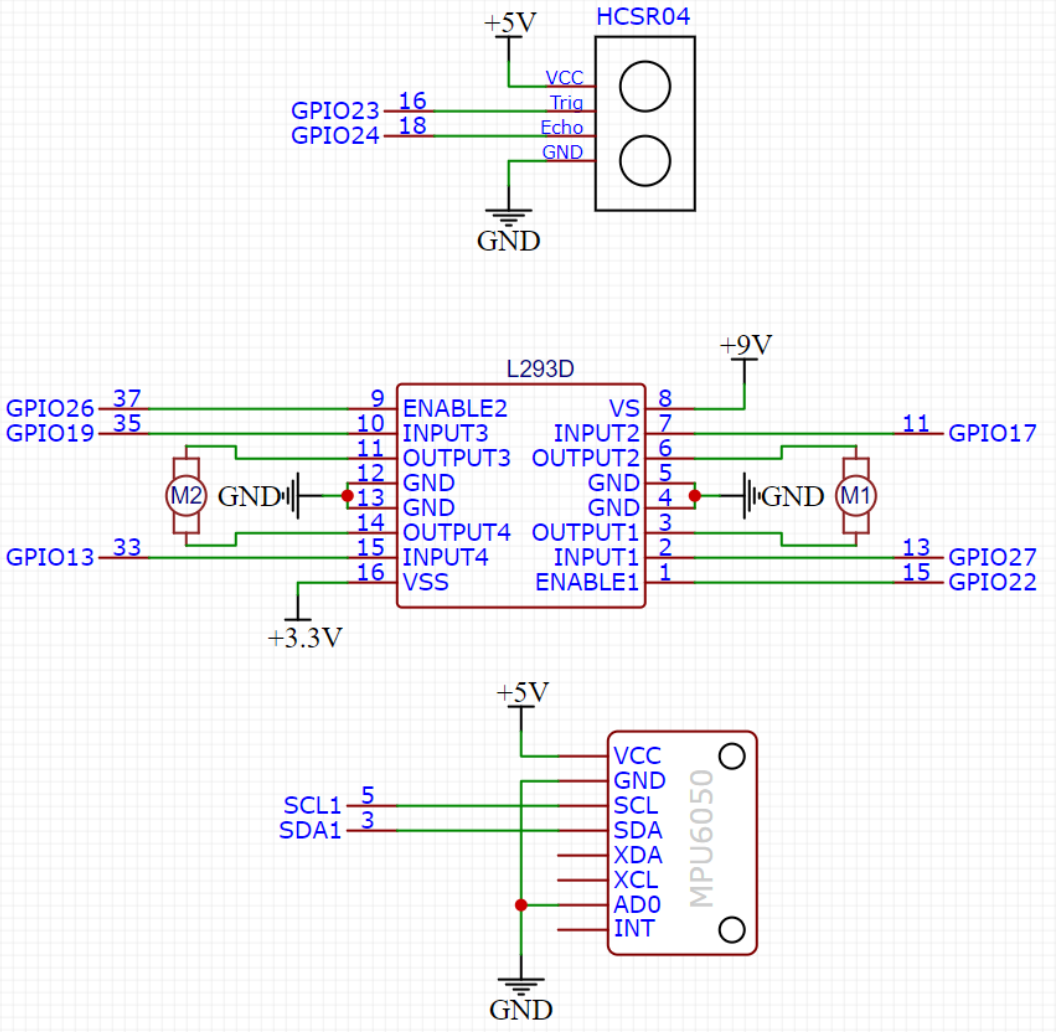
El vehicle autònom es connecta al servidor a través de la seva connexió sense fils i envia tant les peticions de noves ordres, com les dades dels sensors. El servidor centralitza els registres que envia el vehicle en una base de dades i alhora respon amb les tasques que estiguin pendents de realitzar-se. Una vegada que el vehicle hagi executat les tasques les marcarà com fetes amb un nou missatge que enviarà cap al servidor web.

Per una altra banda, tindrem dos programes client que es connectaran al servidor. És important destacar que els programes client no es connecten mai de manera directa al vehicle, sinó que ho fan sempre a través del servidor. El programa de control remot envia ordres al vehicle i el monitor de l'estat del vehicle mostra els darrers valors que ha enviat de cadascun dels sensors i la llista de tasques realitzades i pendents.

El codi que podem trobar en l'Annex 3 (codi font) per aquesta solució es troba estructurat de la següent forma:

- /Python:
  - /Activitat10.py : Programa principal del vehicle autònom
  - /Activitat10_IoT.py : Mòdul de comunicacions IoT del vehicle autònom
  - /Activitat10_Motor.py : Mòdul de control dels motors del vehicle autònom
  - /Activitat10_Sensor.py: Mòdul de control dels sensors del vehicle autònom
- /VB.NET:
  - ws-control-raspberry.sln : Solució que conté tots els projectes
  - /Controlador: Projecte del client de control remot
  - /Monitor: Projecte del client de monitoratge del sistema
  - /ws-control-raspberry: Projecte del servidor web
  - /TestWSRaspberry: Projecte client per fer proves del web service
- /BaseDades:
  - RaspberryDB.bak: Còpia de seguretat de la BD (SQL Server 2019)

Circuit:



Referències.

Labrador Piquer, M.J., Andreu Andrés, M.A. Metodologías Activas. Universitat Politècnica de València – Ediciones UPV (2008) Disponible a: <http://www.upv.es/diaal/publicaciones/Andreu-Labrador12008_Libro Metodologias_Activas.pdf>

Martínez, J. y Gómez, F. (2010) La técnica puzzle de Aronson: descripción y desarrollo. En Arnaiz, P.; Hurtado, Mª.D. y Soto, F.J. (Coords.) 25 Años de Integración Escolar en España: Tecnología e Inclusión en el ámbito educativo, laboral y comunitario. Murcia: Consejería de Educación, Formación y Empleo. Disponible a <https://diversidad.murciaeduca.es/tecnoneet/2010/docs/jmartinez.pdf>

Montanero Fernández, M. (2019). Métodos pedagógicos emergentes para un nuevo siglo ¿Qué hay realmente de innovación? Teoría de la Educación. Revista Interuniversitaria , 31 (1), 5. Disponible a <https://doi.org/10.14201/teri.19758>