French team 2016

Project :

Imad BELARBI
Alexandre FERREIRA
Djoselie OSSIE
Greg LEDAIN

SPARTAN SUPERWAY

International Institute of Sustainable Transportation

San jose CA , 2016 SUMMER

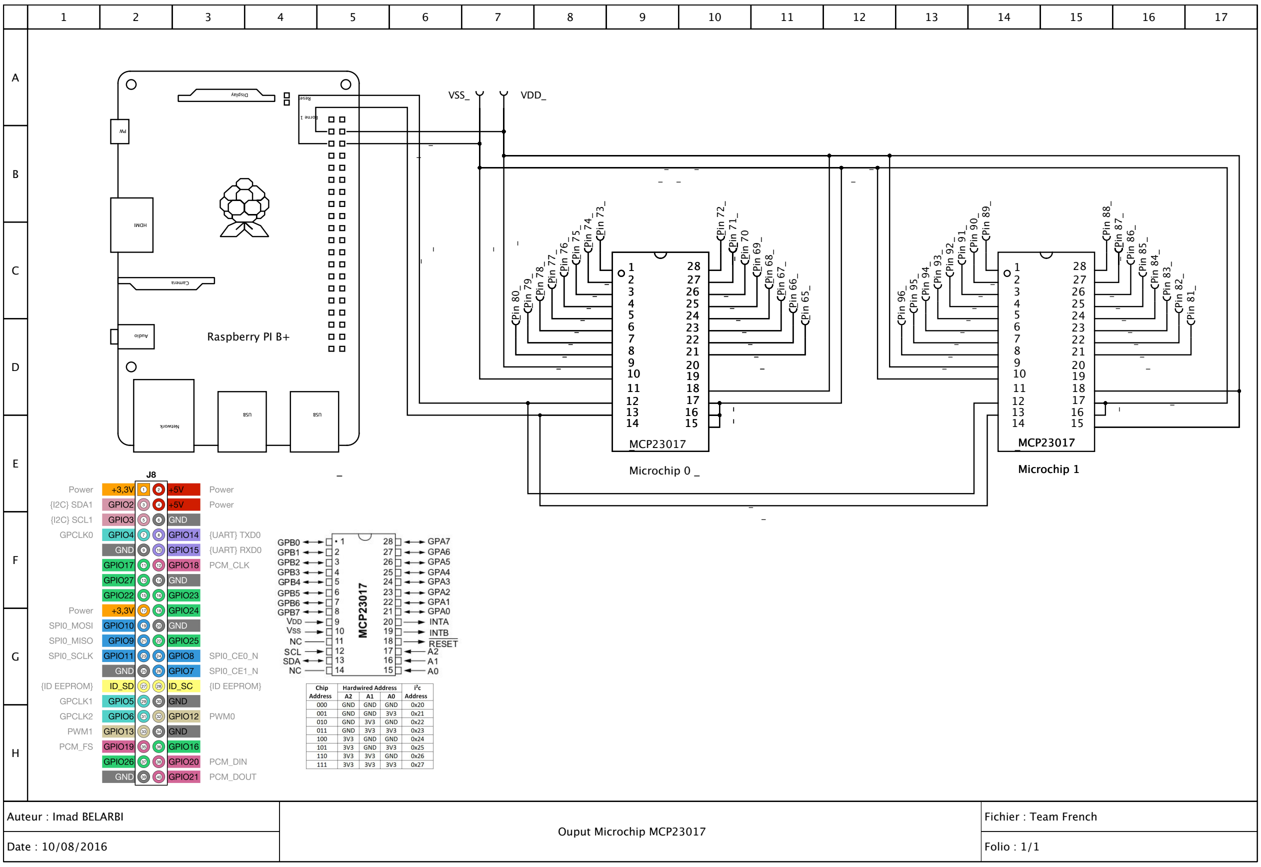
Problematic

Automation of existing prototype, take into account all the existing system, For this we offer two system :

A. An interface, programmable logic controller ZELIO SR3B261BD, Schneider Electric.
B. An interface, Raspberry Pi B.

the selection is focused on Raspberry Pi interface, for financial reasons.

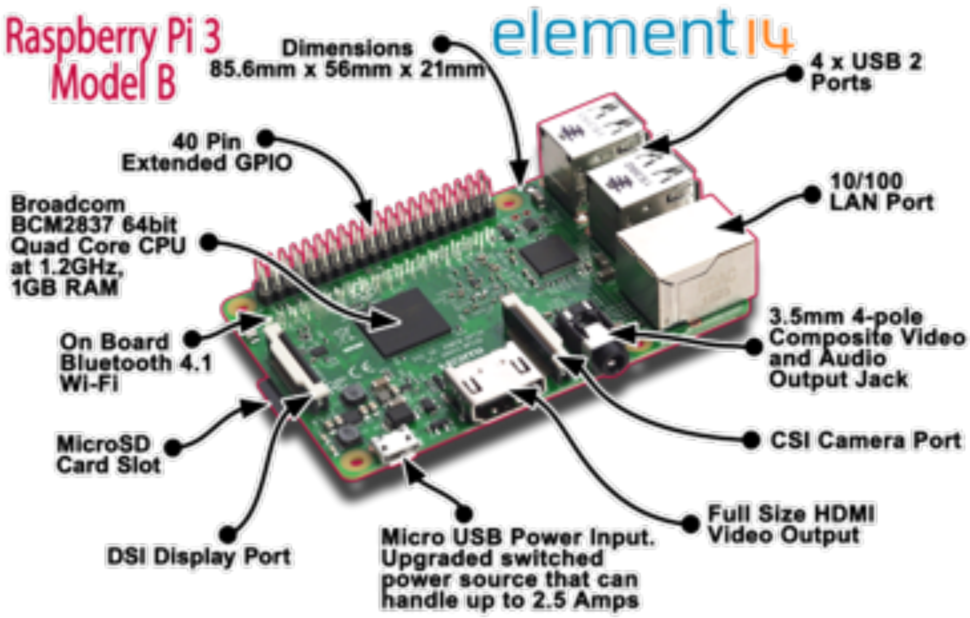
the scheme is as follows :



International Institute of Sustainable Transportation

San jose CA , 2016 SUMMER

Why choose a Raspberry Pi?



Raspberry Pi is primarily a computer, allowing the majority of uses thereof. But it also has a very low cost (about $35).
Many features results in an incredible increase the possible uses of the Raspberry Pi, in our case we will use :

- Controller for motor (With PWM).
- Controlling the opening and closures cabin doors.
- Controller for break cabin
- Detection of the position of the car on way
- Speed for cabin.

the list is long…

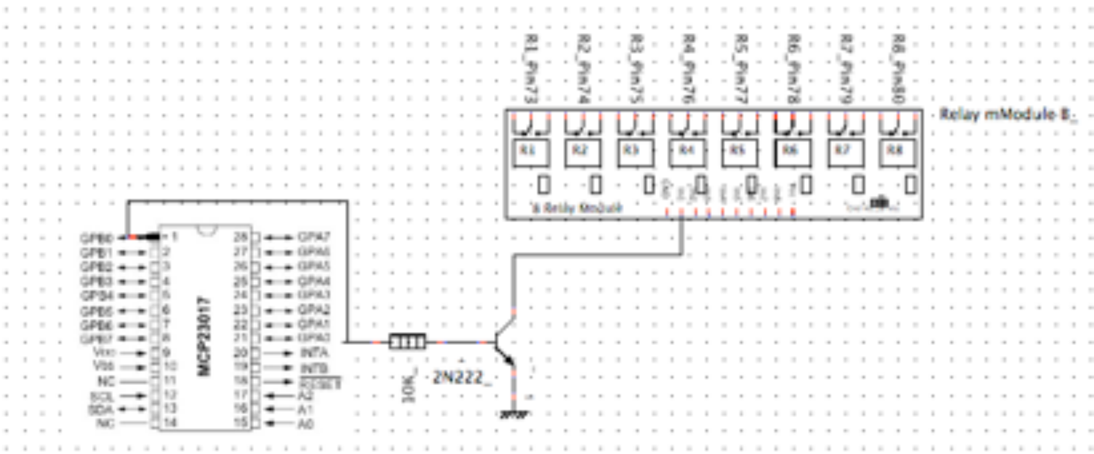
We were decided to set up two microchip MC23017, not to use pin raspberry. Each microchip have 16 pin, we choose one for output (addressing pin : 65 to 80) and other for input (addressing pin : 81 to 96).

The Microchip MCP23017 for Output:

MCP23017 provides 16-bit, general purpose parallel I/O expansion for I2C bus or SPI applications. The data for each input or output is kept in the corresponding input or output register. The polarity of the Input Port register can be inverted with the Polarity Inversion register. All registers can be read by the system master.
The 16-bit I/O port functionally consists of two 8-bit ports (PORTA and PORTB).

It is connected to two relay module Each one consists of 8 relay, characteristic of the latter:

- 5V 8-Channel Relay interface board, and each one needs 15-20mA Driver Current, to limit the current one add 10 kilos to play a ohm resistor and a transistor the



program controller for output follow :

```
import wiringpi
i2c_addr = 0X20 # Defened chip 0
wiringpi.wiringPiSetup()
wiringpi.mcp23017Setup(65,i2c_addr)
wiringpi.pinMode(73,1) # Defened OUTPUT
wiringpi.digitalWrite(73,0) # OUTPUT OFF
wiringpi.digitalWrite(73,1) # OUTPUT ON
```

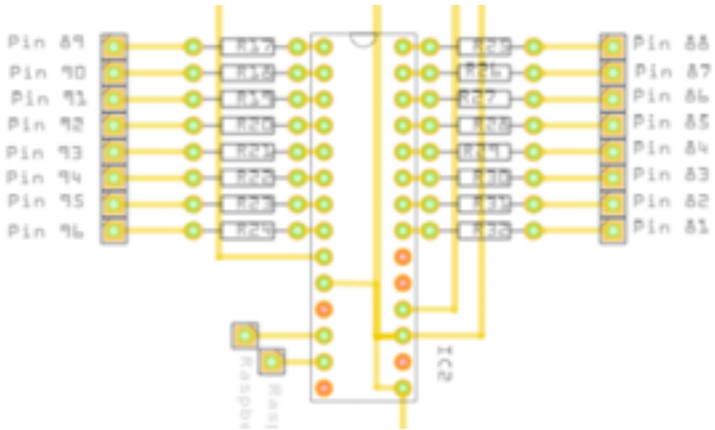
- Equipped with high-current relay, AC250V 10A ; DC30V 11A.
  the relay module function as an insulator between exteriere a system that can work up to 10 amperes and Raspberry

- Standard interface that can be controlled directly by microcontroller (Arduino , 8051, AVR, PIC, DSP, ARM, ARM, MSP432, TTL logic)

- Indication LED's for Relay output status

The Microchip MCP23017 for Input:

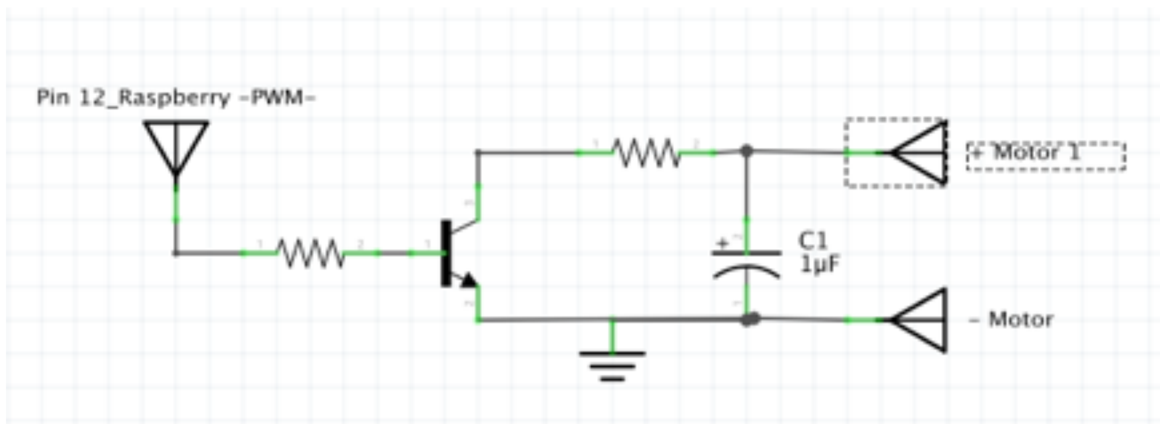
the second microchip is used for have 16 entries. The resistors used, they are nicknamed the pull-up, which determines the default state of the pin (logic high).



The Motor controller :

For this application, we used the Pulse Width Modulation (PWM) function. Pulse Width Modulation (PWM) modules, which produce basically digital waveforms, can be used as cheap Digital-to-Analog (D/A) converters only a few external components. A wide variety of micro controller applications exist that need analog output but do not require high resolution D/A converters. Some speech applications (talk back units, speech synthesis systems in toys, etc.) also do not require high resolution D/A converters. For these applications, Pulse Width Modulated outputs may be converted to analog outputs.
ou will find more detail in Appendices.

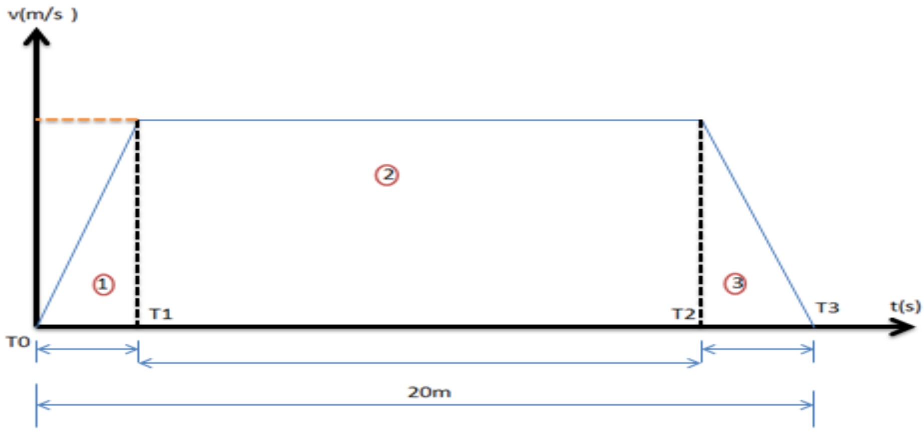
Electrical Schema :



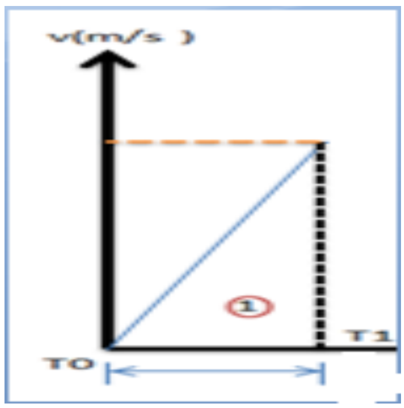
MODELING OF THE CABIN IN ADVANCE SYSTEM

HORIZONTAL TRACK

In this part , we have three stages : acceleration, deceleration constant and modeled as below :



Step 1 : Acceleration



In this step, we set of basic values such as the acceleration (m /s²) , the deceleration (m /s²) and the speed (m / s) (these values can be changed as desired uses)

Equations for calculating hourly

In this step, the movement is defined as : a uniformly accelerated motion and the initial conditions are the following:

$X_o = V_o = T_o = 0$

The equations are defined movement type are:

$X1(t) = 1/2 * a * T1^2 + Vo(T0-T1) + X0$

V1(t)= a*T1
a(t)=constant

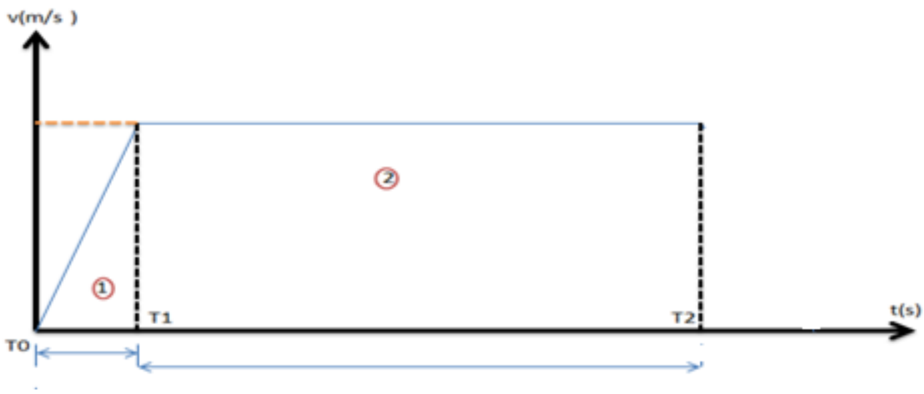
T1=V1/a	ΔT 1(s)	1,60

By making the digital application with the values that are available , we obtain:

		Basic parametre	Value
		Mass(Kg)	270
		Acceleration(m/s²)	1,39
		Deceleration(m/s²)	-1,39
		frotemment(N)	75
Step	Formula	Speed(m.s)	2,22
Step 1	x1= 1/2*a*T1² V1= a*T1		
		X0(m)	0
		X1(m)	1,77

T1=V1/a	ΔT 1(s)	1,60

Step 2 : Constant



In this step, we have a constant speed and a zero acceleration.

Equations for calculating hourly

In this phase the movement is defined as : a Uniform Movement.
The equations are defined movement type are:

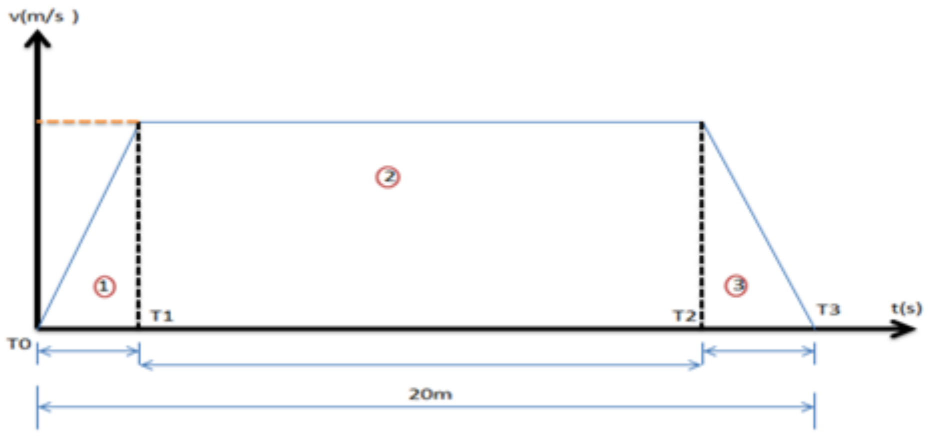
X2(t)= 1/2* a*T2² + V1(T2-T1)+ X1
V2(t)= V1

a=0

By making the digital application with the values that are available , we obtain:

		Basic parametre	Value
		Mass(Kg)	270
		Acceleration(m/s²)	1,39
		Deceleration(m/s²)	-1,39
		frotemment(N)	75
Step	Formula	Speed(m.s)	2,22
Step 1	x1= 1/2*a*T1² V1= a*T1		
		X0(m)	0
		X1(m)	1,77
Step 2	x2= V2*(T2-T1)+X1 V2=V1 a=0	X2(m)	18,23
T2-T1= ((X2-X1)/V2)		ΔT 2(s)	7,41

Step 3: deceleration



In this step, there is a deceleration.

Equations for calculating hourly

In this phase the movement is defined as : a decelerated Uniform Movement.
The equations are defined movement type are:

$X3(t)= 1/2* a*T3^2 + V2(T3-T2)+ X2$

$V3(t)= -a*T3+ V2$

-a=constant

By making the digital application with the values that are available, we obtain:

Step	Formula	Basic parametre	Value
		Mass(Kg)	270
		Acceleration(m/s^2)	1,39
		Deceleration(m/s^2)	-1,39
		frotemment(N)	75
		Speed(m.s)	2,22
Step 1	$x1= 1/2*a*T1^2$ V1= a*T1		
		X0(m)	0
		X1(m)	1,77
Step 2	x2= V2*(T2-T1)+X1 V2=V1 a=0	X2(m)	18,23
Step3	$X3= 1/2*a*(T3-T2)^2+ V2(T3-T2)+X2$ V3=-a*(T3-T2)+V2	X3(m)	20
	T1=V1/a	ΔT 1(s)	1,60
	T2-T1= ((X2-X1)/V2)	ΔT 2(s)	7,41
	T3-T2= V2/a	ΔT 3(s)	1,60

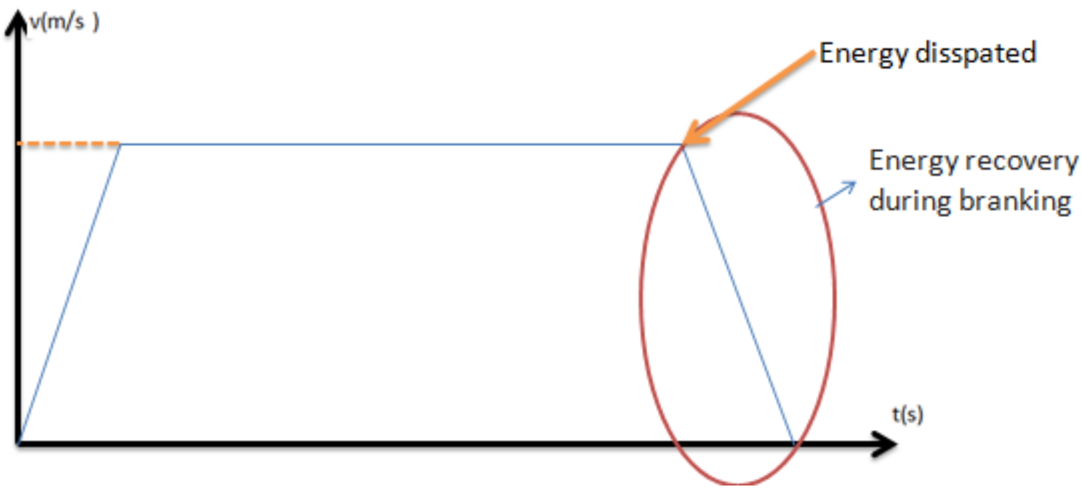
Calculation of energy on each step

In this calculation step, we calculate the total energy required by the cabin in each step.

	Calculation of energy on each step			
	Formula	Energy	Value	Consommation(w.h)
Step 1	F1= M*a + f	F1(N)	450,3	
	P1=F1*V	P1(W)	999,666	
	E1= F1(X1-X0)	E1(J)	798,29	0,22
Step 2	F2= M*a + f	F2(N)	75	
	P2=F2*V	P2(W)	166,5	
	E2= F2(X2-X1)	E2(J)	1234,08	0,34
Step 3 (braking)	F3= M*a + f	F3(N)	-300,3	
	P3=F3*V	P3(W)	-666,666	
	E3= F3(X3-X2)	E3(J)	-532,37	0,00

Energy recovery during braking

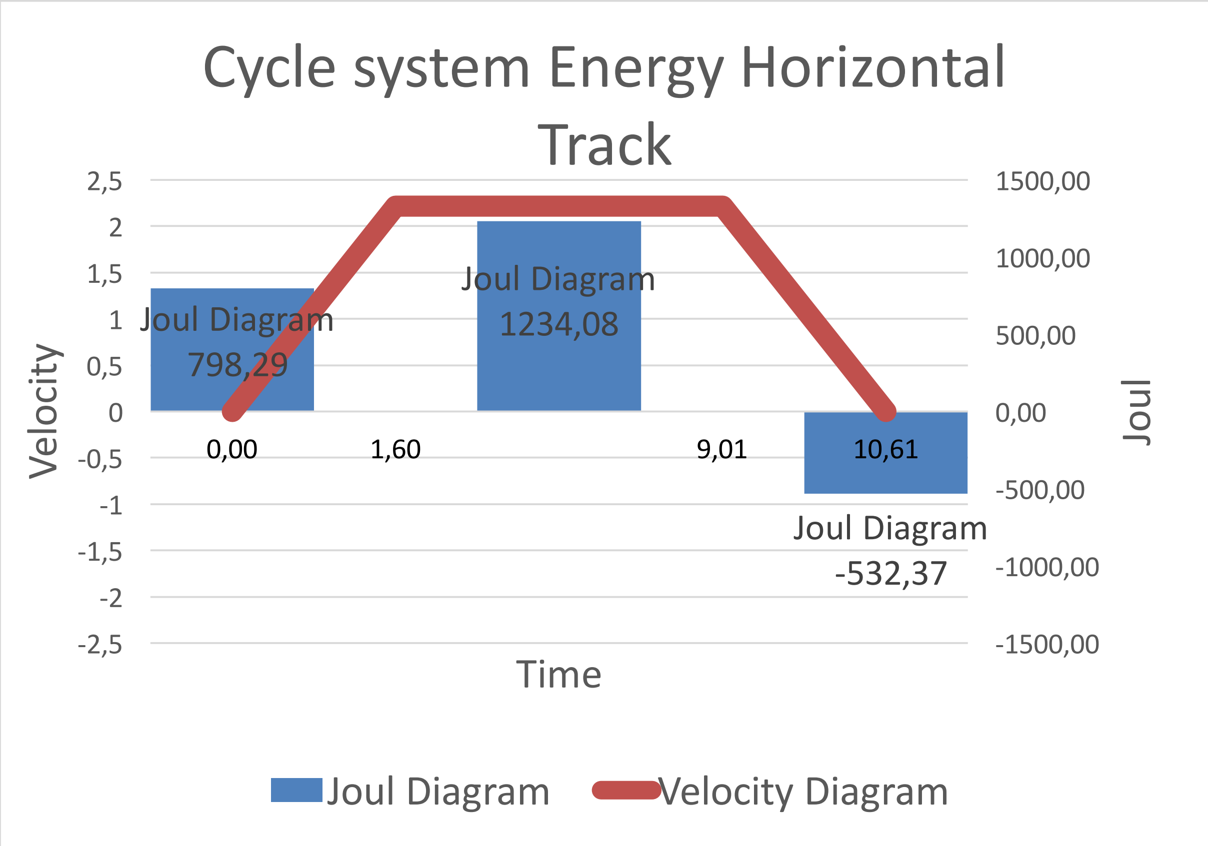
In this part, we will rely only on step 3 for the calculation of energy recovery , on the assumption the fact that an energy dissipation occurs at the beginning of braking as shown in the diagram below :



By making the digital application with the values that are available , we obtain:

		Energy recovery during braking			
		Formula	Energy	Value	Consommation(w.h)
Step 3 (braking)	Energy dissipated	Ediss= 1/2*M*V²	Ediss(J)	665,334	
	Coefficient of recovery	CoefR= (1-(E3/Ediss))*100	Coef(%)	19,98%	
	Total energy	E=E1+E2+E3 (E3=0)	E(J)	2032,37	1
	Percentage or energy branking	%= (Coef*Ediss)/E	**6,54%**	**132,96 j**	**0,04 W.h**

Schema of the cycle system Energy Horizontal Track

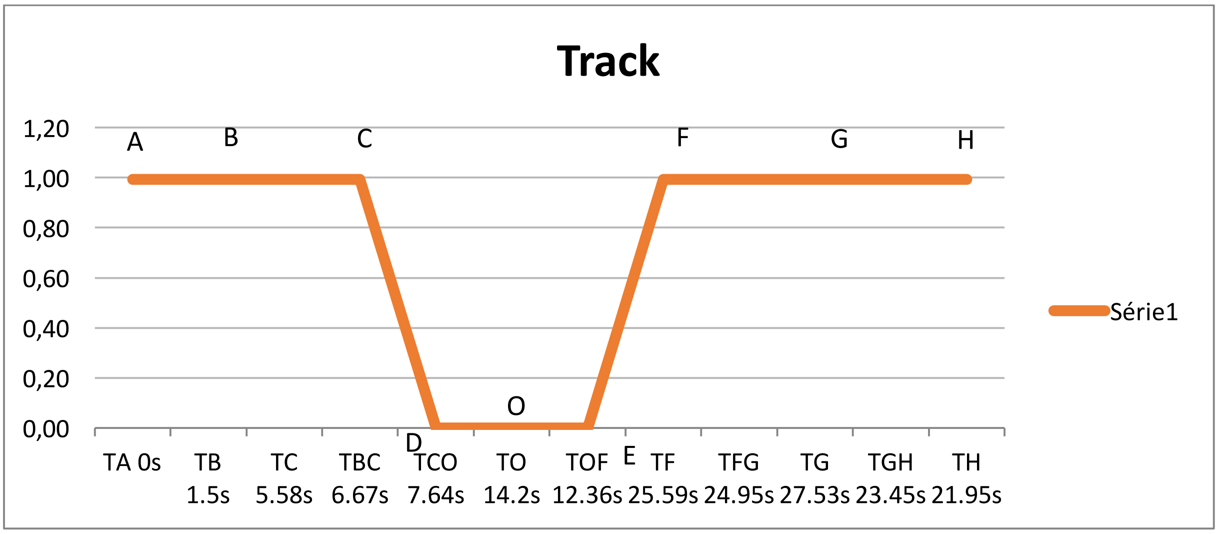


Get down track

This work deals with energy recovery. To calculate this energy recovery, we used:

- ➔ Work breaking
- ➔ Breaking time
- ➔ Breaking energy
- ➔ Breaking force
- ➔ Deceleration
- ➔ Acceleration

Here you have a diagram of the get down truck, with the different times at each point.



The times are calculated with schedule equation (cinetique mechanical). All equations are explain in the work from the horizontal track.

Formula:

➔ **Breaking energy [J] = working breaking= breaking force [N]*breaking distance[m]**

- **Breaking force = (deceleration *masse + friction)/(breaking distance)** [N.m^{-1}]→ because the breaking is gradually (Second low of Newton)

So, **breaking energy = Breaking force [N.m^{-1}]*breaking distance [m]**

➔ **Dissipate energy [J] = mechanical energy = (cinetique energy + potential energy)** [J]

- Cinetique energy = ½*mass*speed^2.
- Potential energy = mass*gravity*different high.

➔ **Energy recover [J] = (1- Absolut(Breaking energy/Dissipate energy))*Dissipate energy [J]**

➔ **Coefficient of recovery [J] = Energy recover / total energy supplied by the motor**

We compile all calculated in an excel file. From the excel file, changing only the value of acceleration and deceleration, calculated are done again.

French team 2016

This scheme is a component of a control and a low pass filter to vary the engine speed, to obtain a stable variation engine, I defined resistance to 1kilos ohm, then I calculated the band décibale me to calculate the cutoff frequency, the procedure is as follows for determining an optimal value of the capacitor :

(1)

$$A_{dB} = 20 \times \log\left(\frac{V_{RIPPLE}}{V_{PWM}}\right)$$

WITH tests we have : V_{RIPPLE} = 0.86 Volts
V_{PWM} = 5 volts

A_{db} = 12.62 db

(2)

$$f_{3dB} = f_{PWM} \times 10^{-\frac{A_{3dB}}{Slope}}$$

f_{3db} = 1.72 Hz

(3)

$$f_{3dB} = \frac{1}{2\,\pi \times R_F \times C_F}$$

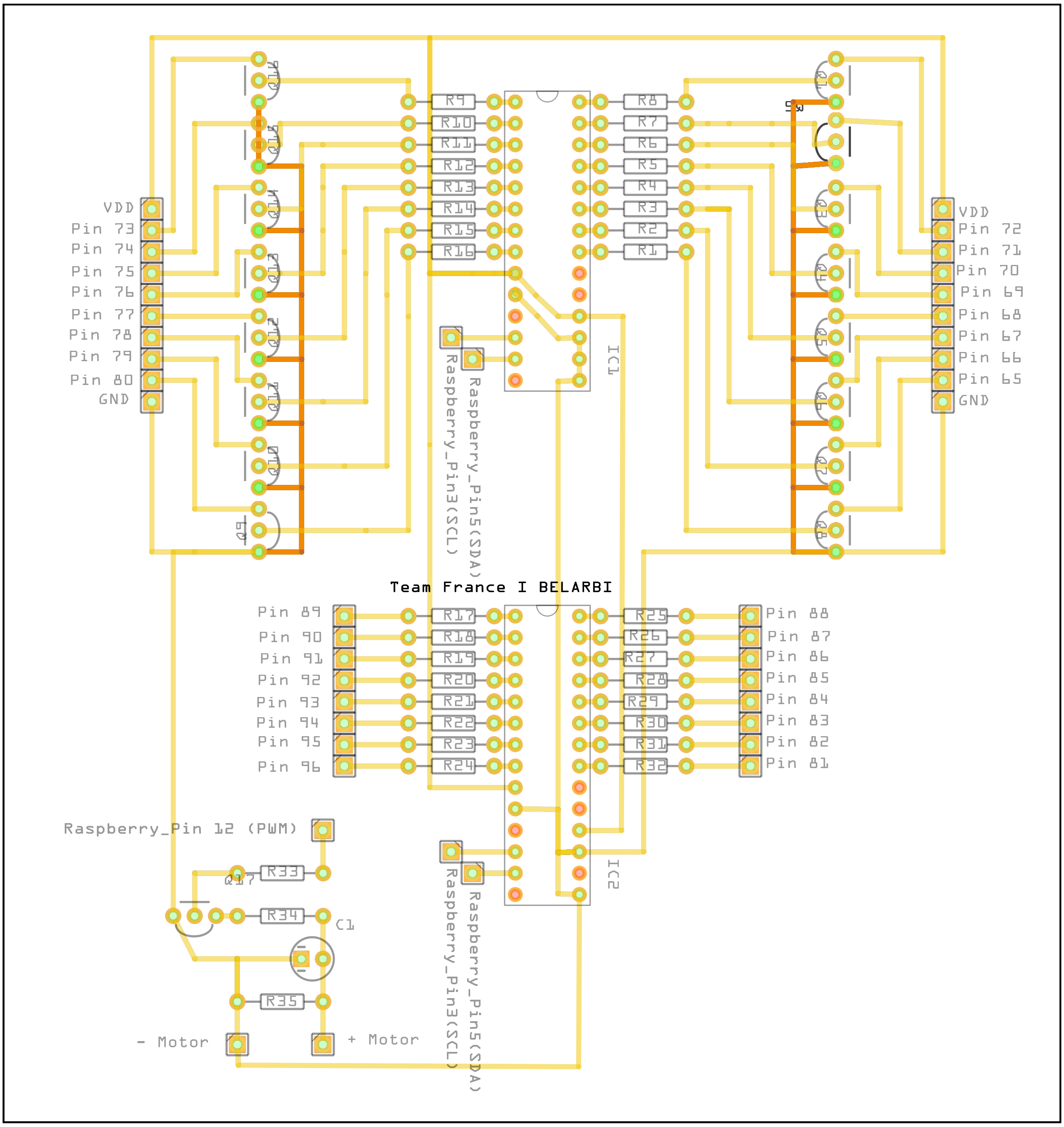
$$C_F = \frac{1}{2\,\pi \times R_F \times f_{3dB}}$$

Fixe resistor 1 k ohm
C_f = 92.5 uF —> normalization C_f = 100 uF

International Institute of Sustainable Transportation

San jose CA , 2016 SUMMER

The printed circuit board



International Institute of Sustainable Transportation

Connecting :

Pin 65 to pin 72, they are outputs connected to relay modules A.
Pin 73 to pin 80, they are outputs connected to relay modules B.
Pin 81 to pin 96, they are inputs connected to system.

The component list :

	Quantity	value
resistors : R1 to R16 R33	17	10 K ohm
resistors : R34	1	1K ohm
resistors : R35	1	30 K ohm
Transistor 2N2222 : Q1 to Q17	17	NPN
Capacitor	1	100 uF
resistors : R17 to R32	16	22 ohm

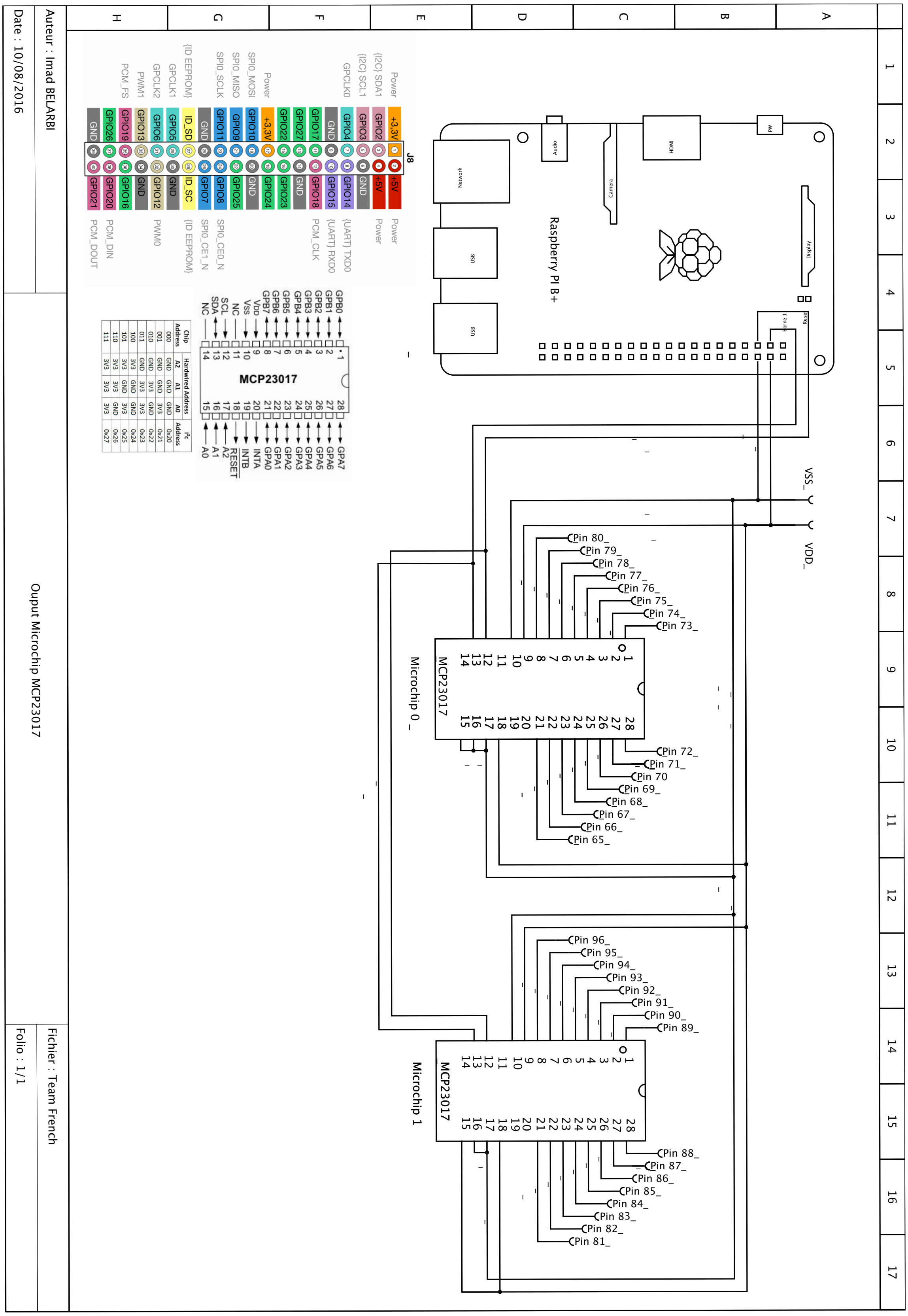
Connecting

Raspberry pi				Micro Chip 0	Micro Chip 1
	N°PIN	Name	Connexion	N°PIN	N°PIN
	1	+3,3V	Free	-	-
	2	+5V	Free	-	-
	3	SDA	↔	13	13
	4	+5V	→	09 / 18	09 / 15 / 18
	5	SCL	→	12	12
	6	GND	←	10 / 15 / 16 / 17	10 / 16 /17
	7	GPIO_04	Free	-	-
	8	GPIO_14	Free	-	-
	9	GND	Free	-	-
	10	GPIO_15	Free	-	-
	11	GPIO_17	Free	-	-
	12	GPIO_18	→	-	-
	13	GPIO_27	Free	-	-
	14	GND	Free	-	-
	15	GPIO_22	Free	-	-
	16	GPIO_23	Free	-	-
	17	+3,3V	Free	-	-
	18	GPIO_24	Free	-	-
	19	GPIO_10	Free	-	-
	20	GND	Free	-	-
	21	GPIO_09	Free	-	-
	22	GPIO_25	Free	-	-
	23	GPIO_11	Free	-	-
	24	GPIO_08	Free	-	-
	25	GND	Free	-	-
	26	GPIO_07	Free	-	-
	27	ID_SD	Free	-	-
	28	ID_SD	Free	-	-
	29	GPIO_05	Free	-	-
	30	GND	Free	-	-
	31	GPIO_06	Free	-	-
	32	GPIO_12	Free	-	-
	33	GPIO_13	Free	-	-
	34	GND	Free	-	-
	35	GPIO_19	Free	-	-
	36	GPIO_16	Free	-	-
	37	GPIO_26	Free	-	-
	38	GPIO_20	Free	-	-
	39	GND	←	-	-
	40	GPIO_21	Free	-	-

Micro Chip 0				Raspberry pi	Module de Relay A	Module de Relay B
	N°PIN Chip	Name	Connexion	N°PIN	N°PIN	N°PIN
	1	GPB0	↔	-	-	Microchip_0_PIN_73
	2	GPB1	↔	-	-	Microchip_0_PIN_74
	3	GPB2	↔	-	-	Microchip_0_PIN_75
	4	GPB3	↔	-	-	Microchip_0_PIN_76
	5	GPB4	↔	-	-	Microchip_0_PIN_77
	6	GPB5	↔	-	-	Microchip_0_PIN_78
	7	GPB6	↔	-	-	Microchip_0_PIN_79
	8	GPB7	↔	-	-	Microchip_0_PIN_80
	9	Vdd	←	4	-	-
	10	Vss	←	6	-	-
	11	NC	Free	-	-	-
	12	SCL	←	5	-	-
	13	SDA	↔	3	-	-
	14	NC	Free	-	-	-
	15	A0	←	6	-	-
	16	A1	←	6	-	-
	17	A2	←	6	-	-
	18	/Reset	→	4	-	-
	19	INTB	Free	-	-	-
	20	INTA	Free	-	-	-
	21	GPA0	↔	-	Microchip_0_PIN_65	-
	22	GPA1	↔	-	Microchip_0_PIN_66	-
	23	GPA2	↔	-	Microchip_0_PIN_67	-
	24	GPA3	↔	-	Microchip_0_PIN_68	-
	25	GPA4	↔	-	Microchip_0_PIN_69	-
	26	GPA5	↔	-	Microchip_0_PIN_70	-
	27	GPA6	↔	-	Microchip_0_PIN_71	-
	28	GPA7	↔	-	Microchip_0_PIN_72	-

Micro Chip 1				Raspberry pi	Input
	N°PIN Chip	Name	Connexion	N°PIN	N°PIN
	1	GPB0	↔	-	Microchip_1_PIN_81
	2	GPB1	↔	-	Microchip_1_PIN_82
	3	GPB2	↔	-	Microchip_1_PIN_83
	4	GPB3	↔	-	Microchip_1_PIN_84
	5	GPB4	↔	-	Microchip_1_PIN_85
	6	GPB5	↔	-	Microchip_1_PIN_86
	7	GPB6	↔	-	Microchip_1_PIN_87
	8	GPB7	↔	-	Microchip_1_PIN_88
	9	Vdd	←	4	-
	10	Vss	←	6	-
	11	NC	Free	-	-
	12	SCL	←	5	-
	13	SDA	↔	3	-
	14	NC	Free	-	-
	15	A0	←	4	-
	16	A1	←	6	-
	17	A2	←	6	-
	18	/Reset	→	4	-
	19	INTB	Free	-	-
	20	INTA	Free	-	-
	21	GPA0	↔	-	Microchip_1_PIN_88
	22	GPA1	↔	-	Microchip_1_PIN_89
	23	GPA2	↔	-	Microchip_1_PIN_90
	24	GPA3	↔	-	Microchip_1_PIN_91
	25	GPA4	↔	-	Microchip_1_PIN_92
	26	GPA5	↔	-	Microchip_1_PIN_93
	27	GPA6	↔	-	Microchip_1_PIN_94
	28	GPA7	↔	-	Microchip_1_PIN_95

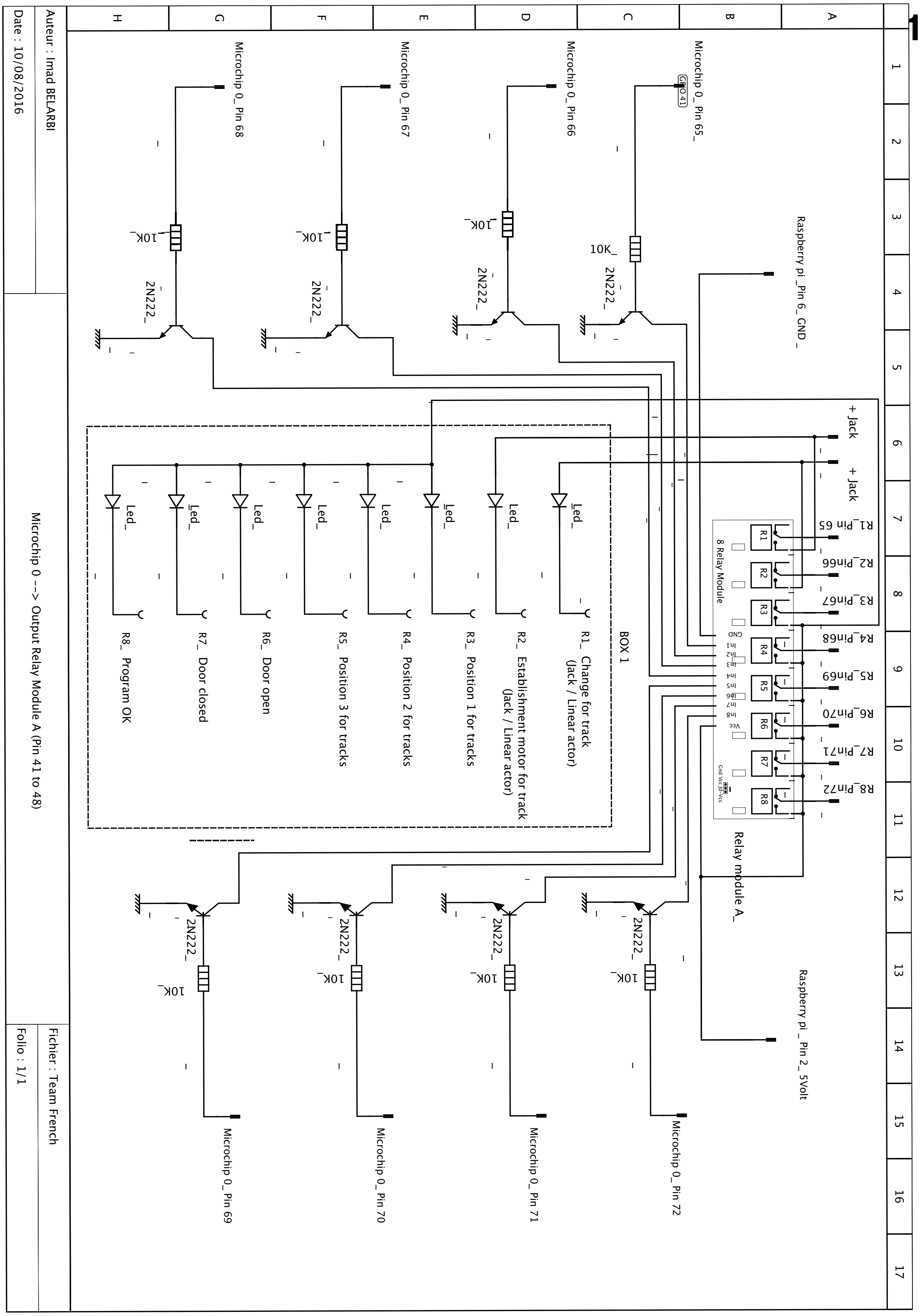
French team 2016



International Institute of Sustainable Transportation

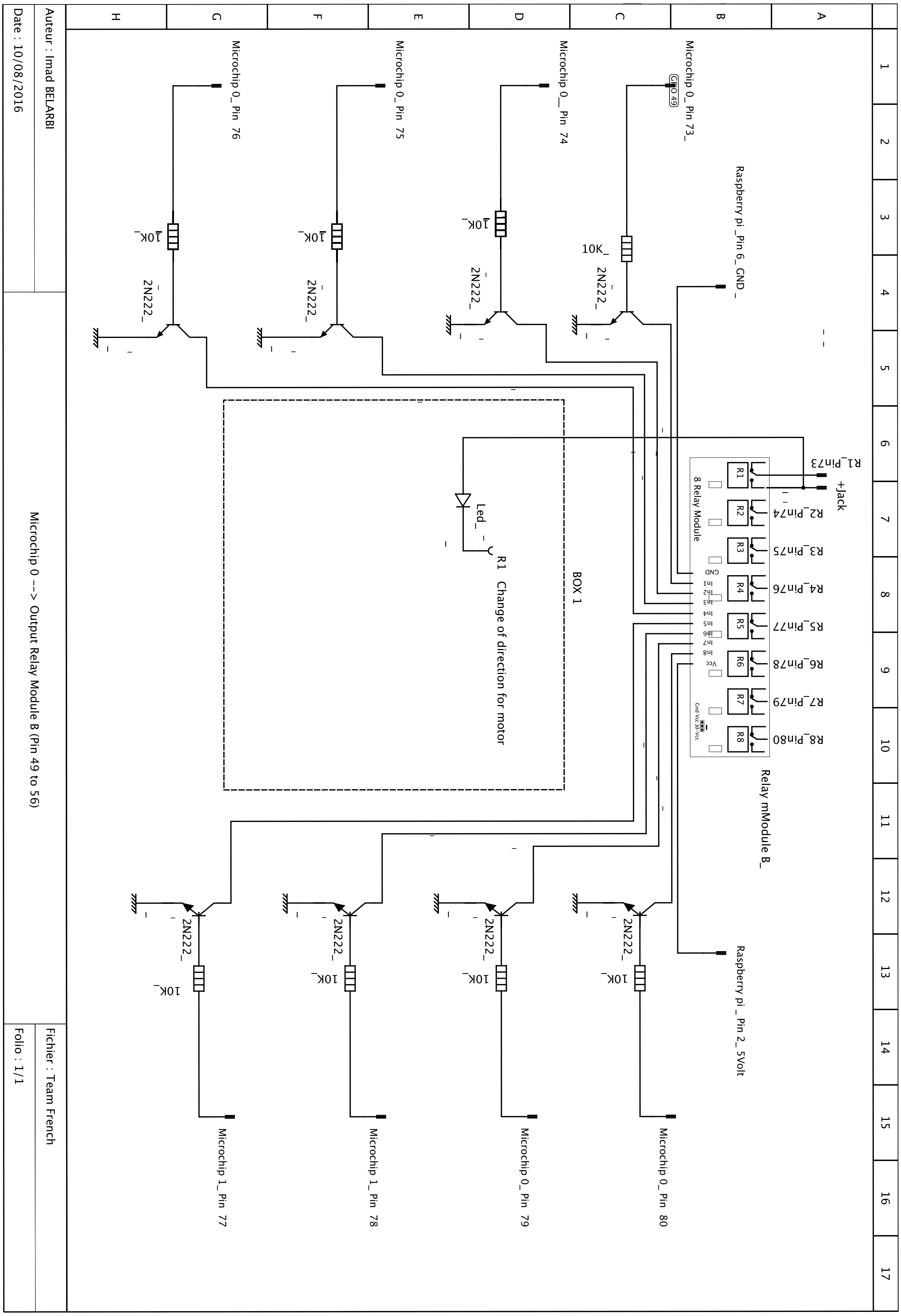
San jose CA , 2016 SUMMER

inGHenia CAMPUS ALTERNANCE MINES Douai



International Institute of Sustainable Transportation

San jose CA , 2016 SUMMER



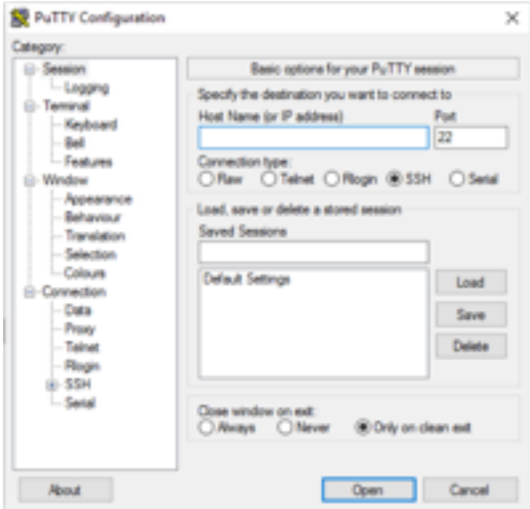
International Institute of Sustainable Transportation

San jose CA , 2016 SUMMER

Take hand of the Raspberry

Windows:

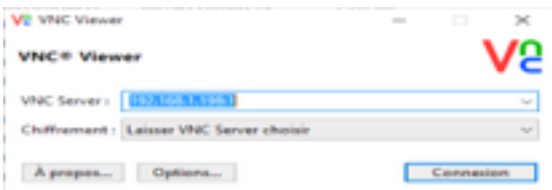
- Download Putty
- Download Fing application to find the Raspberry IP address on the network.
- Download VNC viewer
- Start Putty and enter the IP address of Raspberry



- Click on Open
- An interface opens
- Login: Pi
- Password: raspberry

- Then at the pi@raspberrpi: ~ to read vncserver and open

- Start VNC :

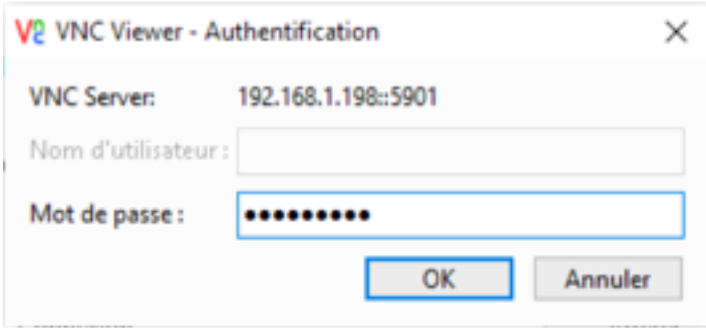


International Institute of Sustainable Transportation

San jose CA , 2016 SUMMER

CAMPUS ALTERNANCE
MINES Douai

- Click on Connection
- Password: raspberry

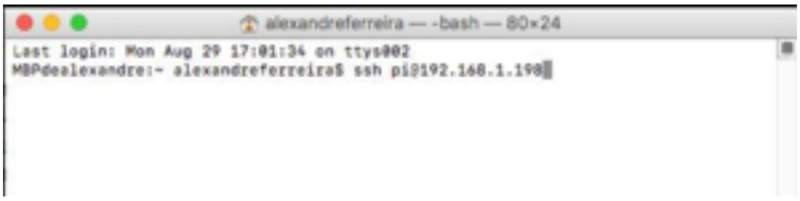


Welcome in the interface of the Raspberry

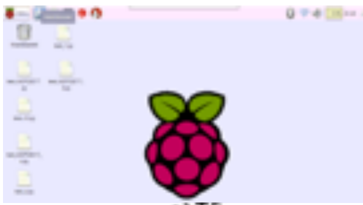
For Mac:

Go to the terminal

- To make an ssh connection
- ssh pi@adresse IP
- See below :



Welcome in the interface of the Raspberry



To read of the program:

This program trains the engine by acting on the push buttons.

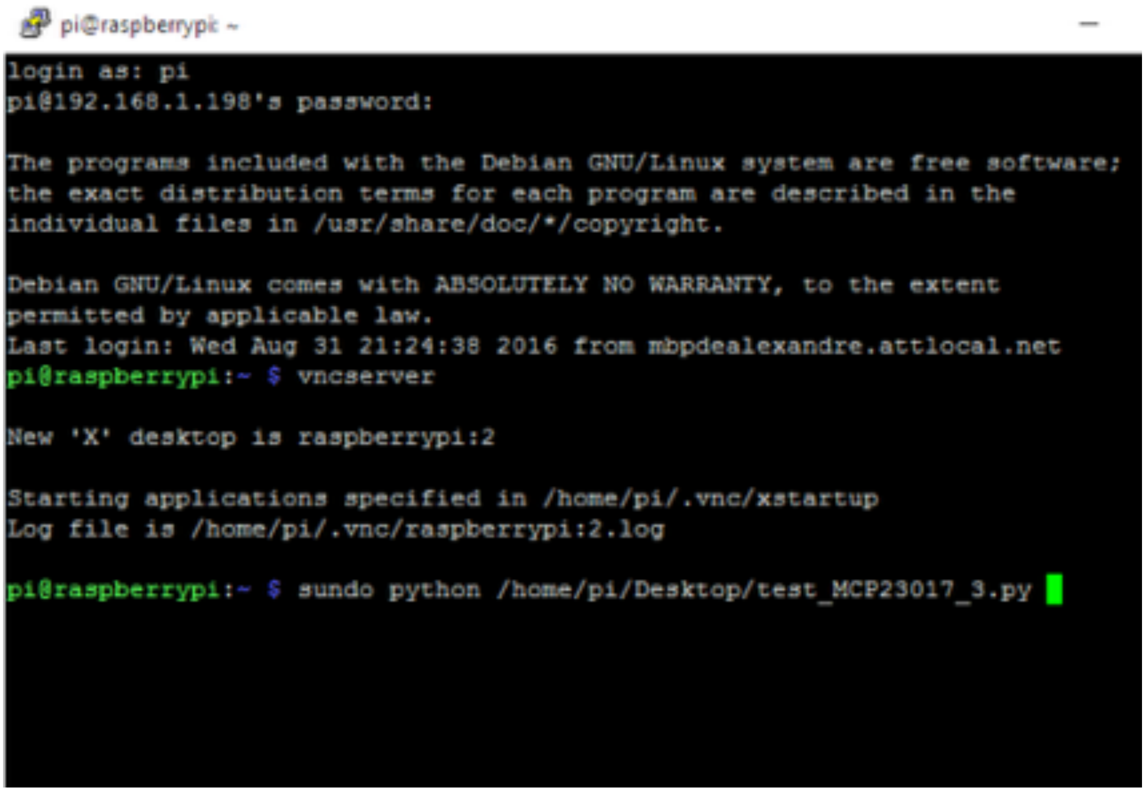
```
import wiringpi #for chips

import time #for time.sleep()

import RPi.GPIO as GPIO #for PWM in raspberry


# PWM motor

GPIO.setmode(GPIO.BOARD)

M1 = 12

GPIO.setup(M1, GPIO.OUT)


# Chip 0 and 1

pin_base = 65

i2c_addr = 0X20 # chip 0

i2c_addr1 = 0X21 # chip 1


wiringpi.wiringPiSetup()

wiringpi.mcp23017Setup(pin_base,i2c_addr) # chip 0

wiringpi.mcp23017Setup(pin_base+16,i2c_addr1) # chip 1


#OUTPUT Chips 0

L = 80

L1 = 75

Q1 = 77

L2 = 76
```

```
F1 = 78

L3 = 79

D1 = 72


#INPUT Chips 1

P1 = 93

P2 = 92

P3 = 91

P = 90

EM = 89

S1 = 81

S2 = 82

S3 = 83


#Initialisation Chips 0 and Chip 1

i= 1

pin=64

try:

    while (i<= 16):

        wiringpi.pinMode(pin + i,1)

        wiringpi.digitalWrite(pin + i,0)

        wiringpi.digitalWrite(pin + i,1)

        wiringpi.digitalWrite(pin + i,0)

        wiringpi.pinMode(pin + i + 16,0)

        i= i+1

finally:
```

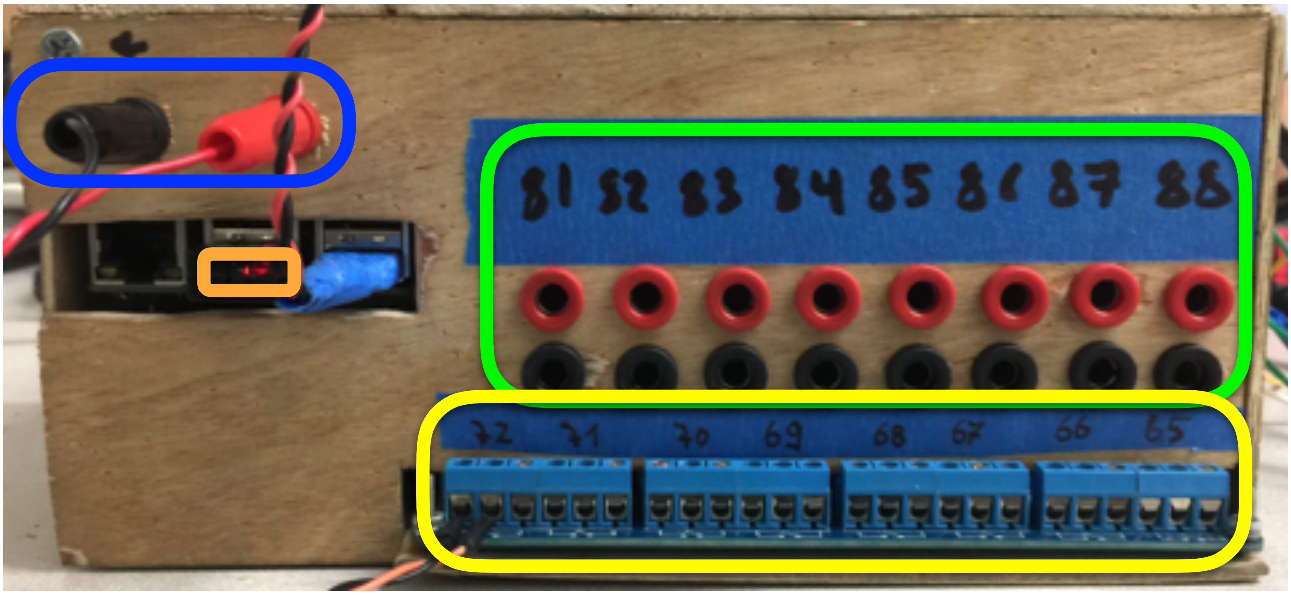
```
    print"OK PIN"

running = True

#Programme
try:
    while running: #Loop for Total Programme.
```

Once the written program , it must send the Raspberry order to start the motor road.

- Open File test_MCP23017_3
- Write in Putty : sudo python /home/pi/Desktop/test_MCP23017_3.py

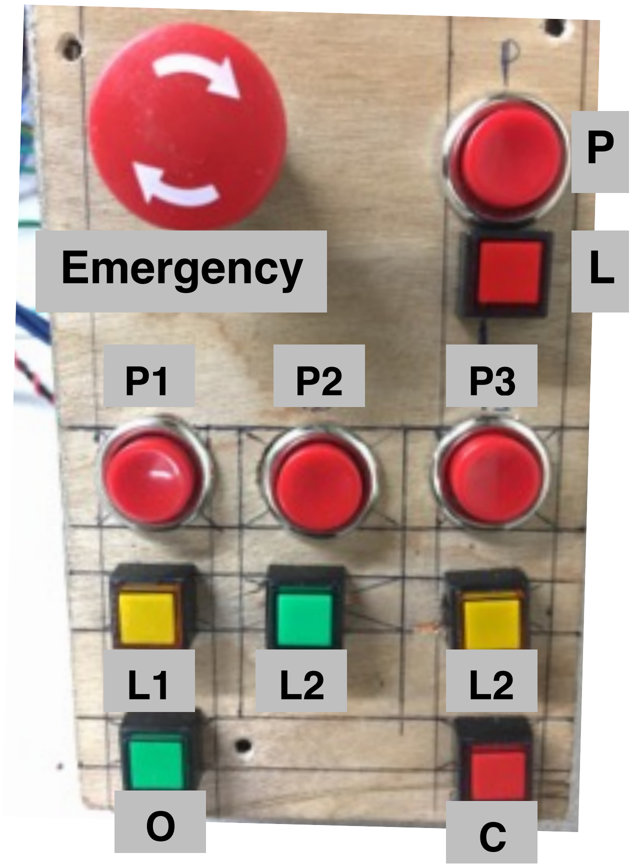


PLC Box :



- Input (Microchip 1)
- Output (Microchip 0)
- Input Motor
- USB Wifi

Controller Box :



Emergency :

P : INIZATION
L : LED inization

P1 : POSITION 1
L1 : LED1 for position1
P2 : POSITION 2
L2 : LED2, for position 2
P3 : POSITION 3
L3 : LED3, for position 3

O : DOOR OPEN
C : DOOR CLOSED

Automated system: