Reporte de tarea 1.

Dinámica de Robots



“Control”

Salguero Hernández Juan Pablo

\_\_\_\_\_\_\_

**Objetivo:**

Utilizar un sensor mediante un nodo publicador en Ros .

**Materiales:**

* Computadora
* Sistema operativo Ubuntu 16.04
* Potenciómetro
* Ros kinetic previamente instalado.
* Placa de programación Arduino
* Placa de programación Cypress CY8CKIT-42xx

Lo primero a realizar es cargar el código en la placa de Arduino.

**Es muy importante notar la forma en la que se crean los nodos de tipo publicador en Ros como se muestra en el código de color azul.**

#include <ros.h>

#include <std\_msgs/Float32.h>

ros::NodeHandle nh;

std\_msgs::Float32 temp\_msg;

ros::Publisher pub\_temp("pot", &temp\_msg);

int analogPin=A0;

int valor;

int posicion;

void setup()

{

nh.initNode();

nh.advertise(pub\_temp);

}

void loop()

{

valor=analogRead(analogPin);

temp\_msg.data = valor;

pub\_temp.publish(&temp\_msg);

nh.spinOnce();

delay(1000);

}

**En el caso de la tarjeta cypress se usa el siguiente código:**

#include <ros.h>

#include <std\_msgs/Int32.h>

#include <std\_msgs/Int16.h>

#include "ADC\_psoc4.h"

#include "Encoder.h"

extern ros::NodeHandle nh;

namespace Encoder {

std\_msgs::Int32 encoder\_msg;

ros::Publisher p("encoder", &encoder\_msg);

std\_msgs::Int16 counter\_msg;

ros::Publisher pc("counter", &counter\_msg);

uint32\_t next\_report\_time;

const uint32\_t kReportIntervalMs = 200;

uint32\_t next\_read\_time;

const uint32\_t kReadIntervalMs = 10; // fast enough to avoid 0x4000 counts in one interval

int16\_t prev\_counter\_val;

int16\_t adc;

int16\_t conversion;

void setup()

{

nh.advertise(p);

nh.advertise(pc);

next\_report\_time = SysTimer::millis();

next\_read\_time = SysTimer::millis();

ADC\_SAR\_SEQ\_Start();

ADC\_SAR\_SEQ\_StartConvert();

ADC\_SAR\_SEQ\_IsEndConversion(ADC\_SAR\_SEQ\_WAIT\_FOR\_RESULT);

prev\_counter\_val = 0; // reload sets value to 0x8000

encoder\_msg.data = 0;

}

void read\_counter() {

// PSoC4 16-bit hardware counter rolls over in a special way; ignore that and use lower 15 bits

int16\_t counter\_val = 0x7FFF & QuadDec\_1\_ReadCounter();

counter\_msg.data = counter\_val;

int32\_t delta = counter\_val - prev\_counter\_val;

if (delta > 0x4000) { delta -= 0x8000; }

else if (delta < -0x4000) { delta += 0x8000; }

encoder\_msg.data += delta;

prev\_counter\_val = counter\_val;

conversion = ADC\_SAR\_SEQ\_GetResult16(0);

adc=ADC\_SAR\_SEQ\_CountsTo\_Volts(0,conversion);

adc=adc\*237.9;

encoder\_msg.data = adc;

counter\_msg.data = adc;

}

void loop()

{

if ((int32\_t)(SysTimer::millis()-next\_read\_time) > 0) {

next\_read\_time += kReadIntervalMs;

read\_counter();

pc.publish(&counter\_msg);

}

if ((int32\_t)(SysTimer::millis()-next\_report\_time) > 0) {

next\_report\_time += kReportIntervalMs;

read\_counter();

p.publish(&encoder\_msg);

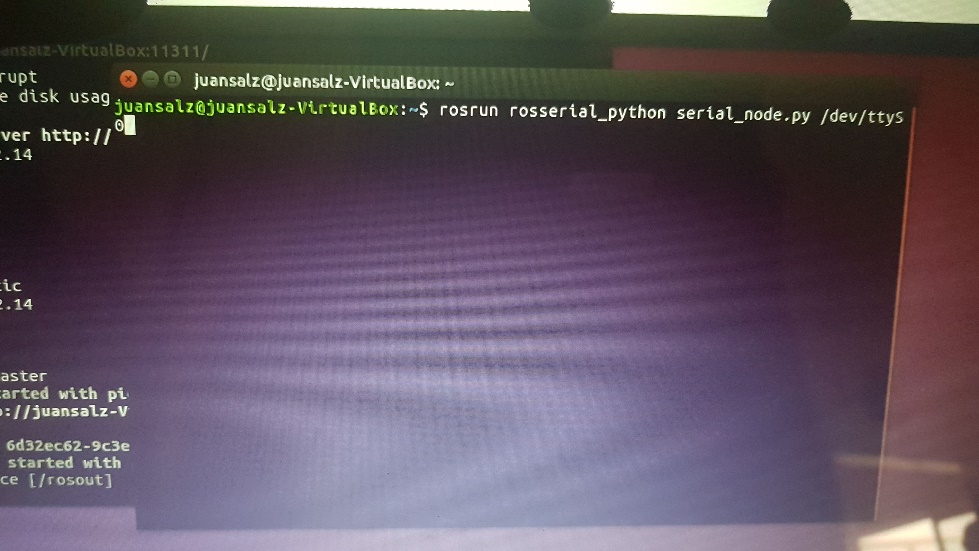
}

}

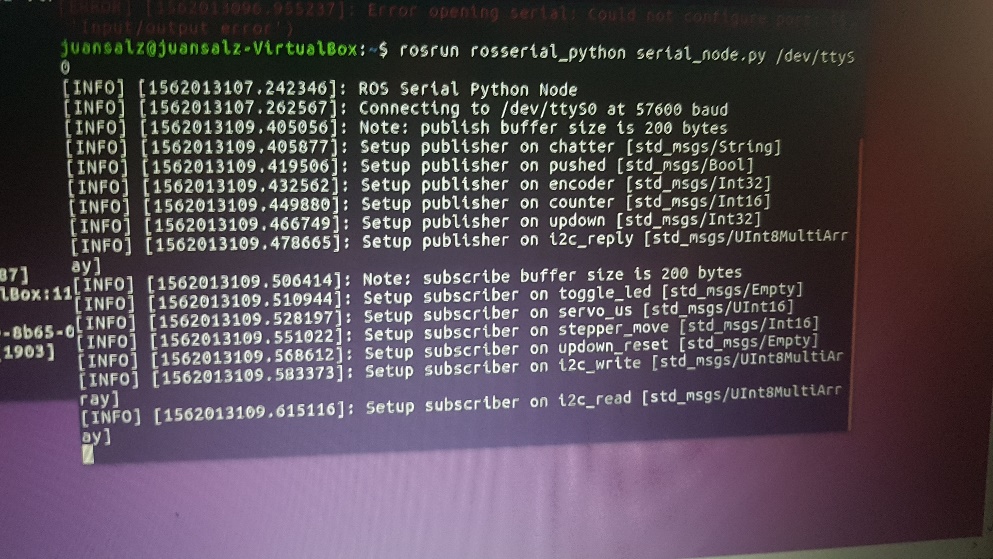
} // namespace Encoder

Para probar el funcionamiento de los códigos tanto de Arduino como de la tarjeta cyprees es necesario iniciar el nodo de roscore desde la terminal y una vez abierto en nodo maestro (roscore) se debe usar la siguiente línea en la terminal para comunicar ros con nuestro dispositivo.

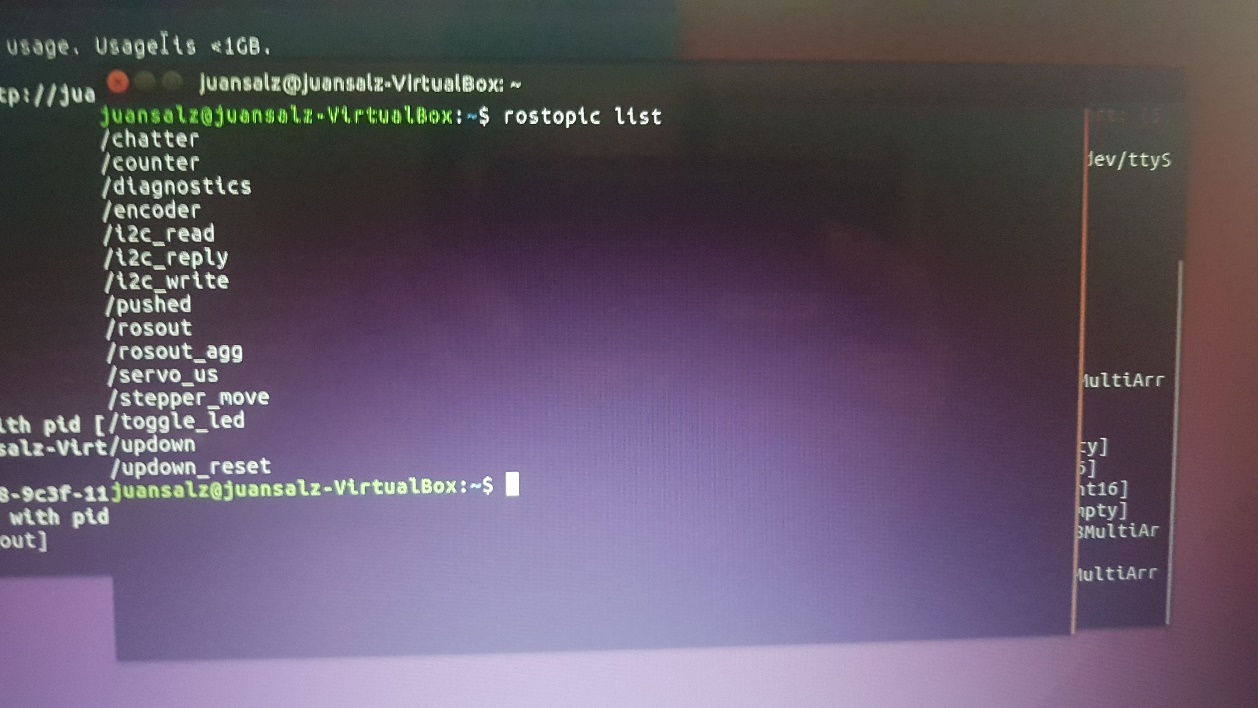
La terminación dev/ttyS0 la determina el puerto donde se encuentra colocada la placa programada.



Si se logra una comunicación exitosa entonces se mostrará información de los nodos programados en la tarjeta en nuestro este caso en particular el nodo utilizado es “encoder” y como se puede observar en la imagen siguiente es un nodo de tipo publicador.

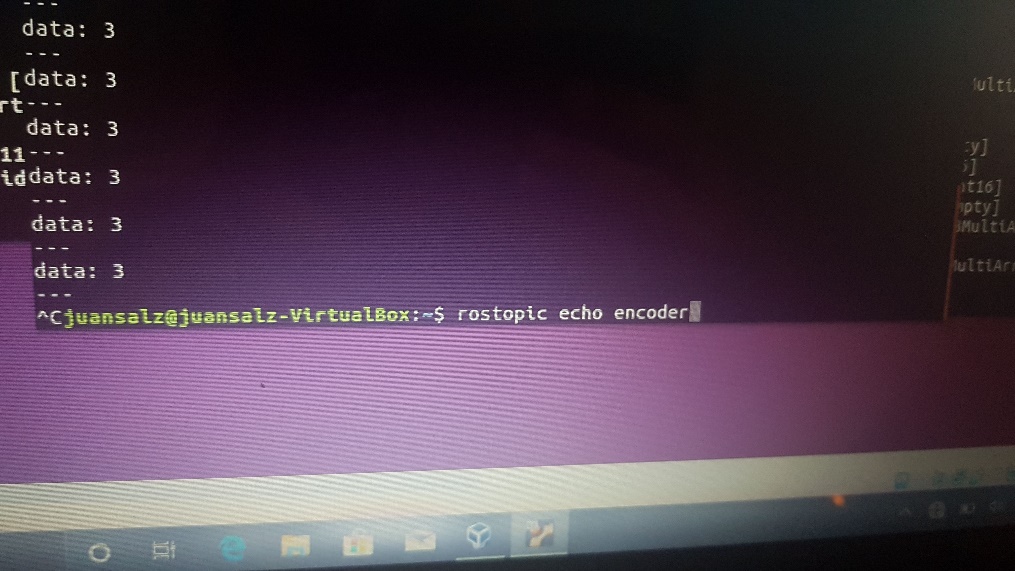


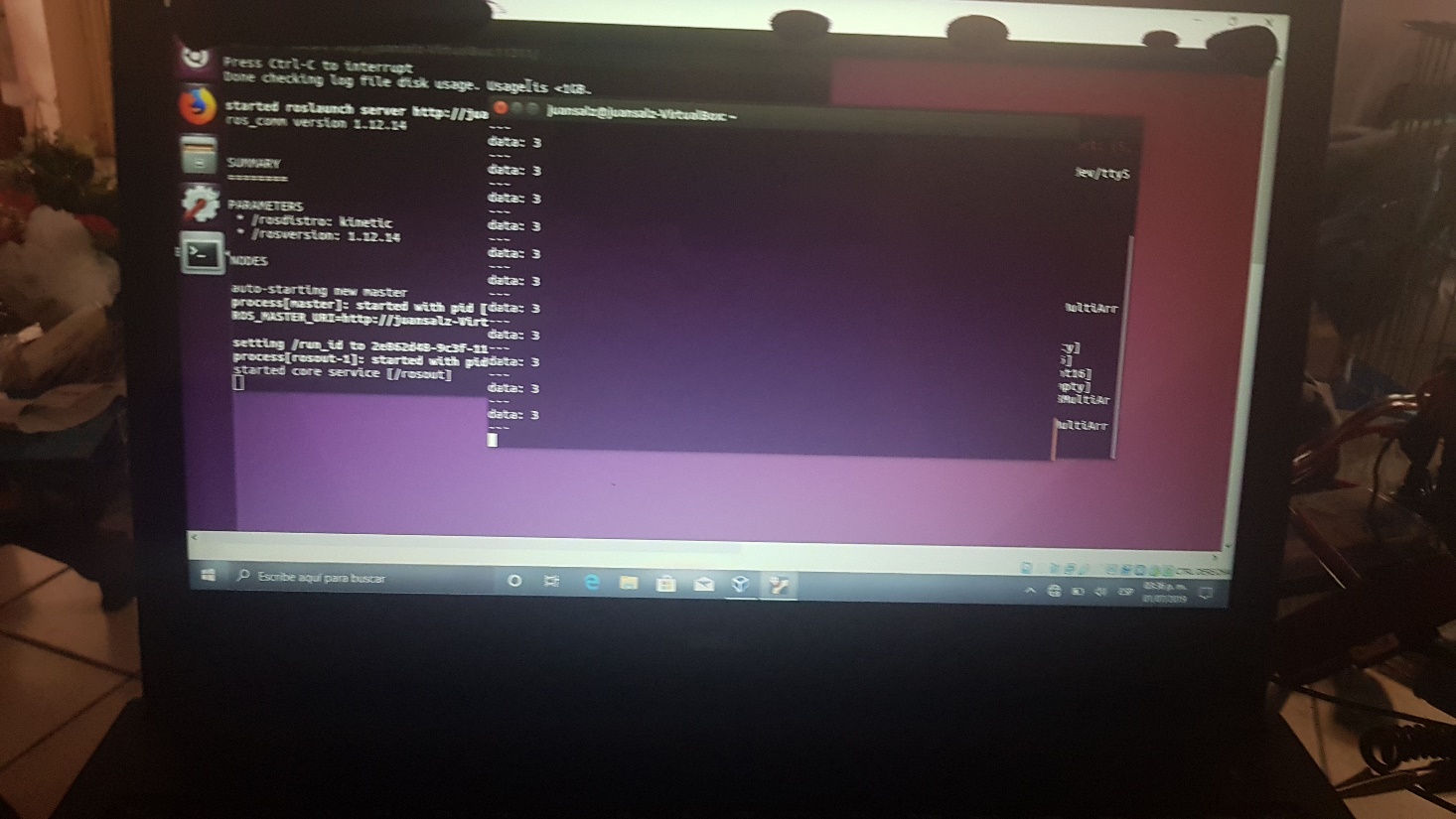
También Podemos usar la línea de comando “rostopic list” para que se muestre en pantalla todos los nodos programados en nuestra placa.



Para poder observar la lectura de nuestro sensor el cual es un potenciómetro se utiliza la línea de código” rostopic echo encoder”. Se usa esa línea de código porque el nodo encoder es el que se encarga de la lectura del sensor.

Los resultados son:





**RESULTADO:**

Se logró la correcta lectura de un sensor, en este caso se opto por el uso del potenciómetro pues puede ser usado como sensor de posición en un control PID de posición de un motor.