1. CM-900 IDE software

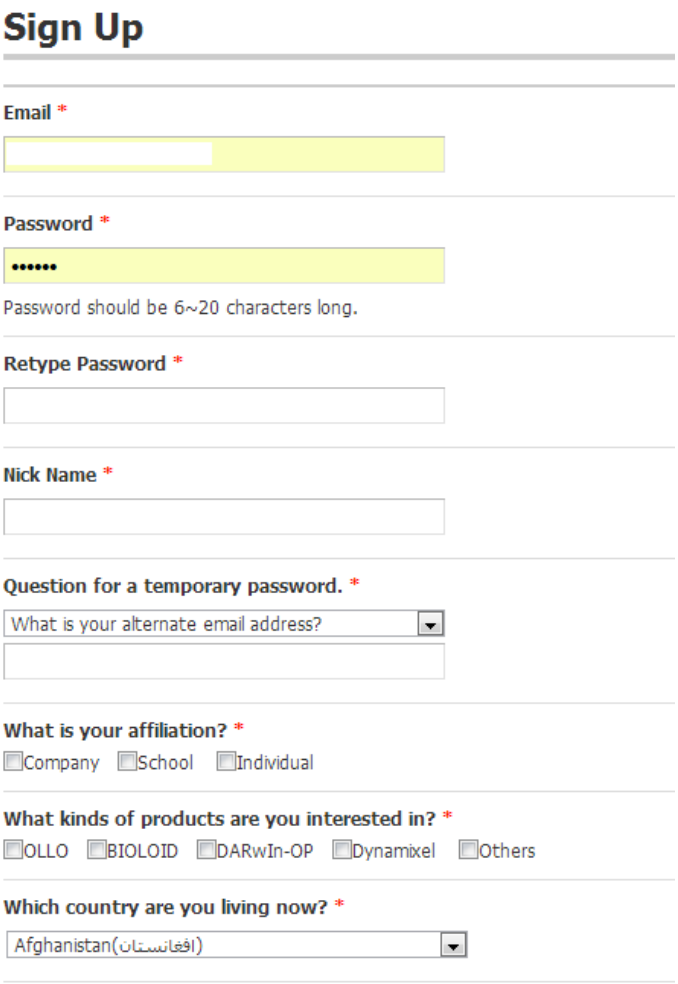
① Download ROBOTIS CM-9

To program the CM-900 you need ROBOTIS CM-9; get it from CM9 Developer's World Circle in BOTSOURCE.

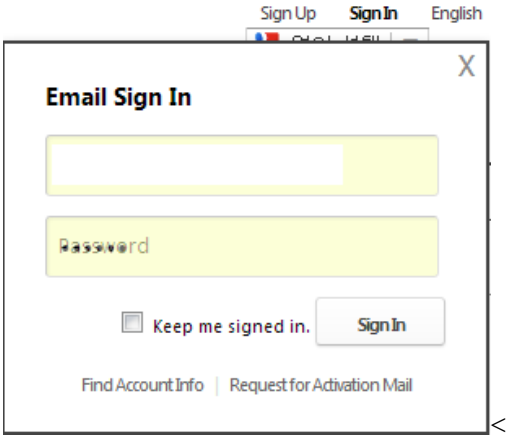
www.robotsource.org

If you have not yet signed up with ROBOTSOURCE we strongly recommend you do so.

Sign Up is a very simple process.



After signing up log in you can proceed to download ROBOTIS CM-9

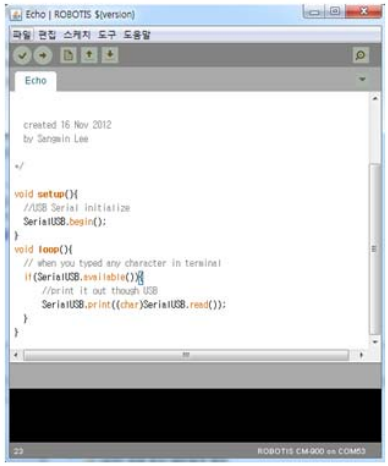


< simply by signing in with your registered email address>

Go to CM-9 Developer's World Circle's Notice and download ROBOTIS CM-9 SW.

No.	Subject	Author	Date	Views
Notice	[New Circle Leader] Prof. Martin Mason [2]	Admin	2013.02.23	147
Notice	Getting Started with CM900 workshop posted [3]	profmason	2013.02.02	270
Notice	[S/W Release]CM9 IDE beta version v0.9.8 release (Windows/Linux/Mac) [2]	Pandora	2013.01.04	508
Notice	CM-900 QuickStart Guide	Pandora	2012.10.23	601
Notice	[Registration] Post your project and get a Free CM-900 for evaluation ***** CLOSED [15]	Jinux	2012.10.20	1191

Click on the link according to your computer OS.



Windows

Linux 32/64bit

Mac OS X

CM9 IDE Beta version 0.9.8 Release

[Windows XP,Vista, 7, 8]
https://www.dropbox.com/s/cygnyh3g7975k0t/ROBOTIS_v0.9.8_win.zip

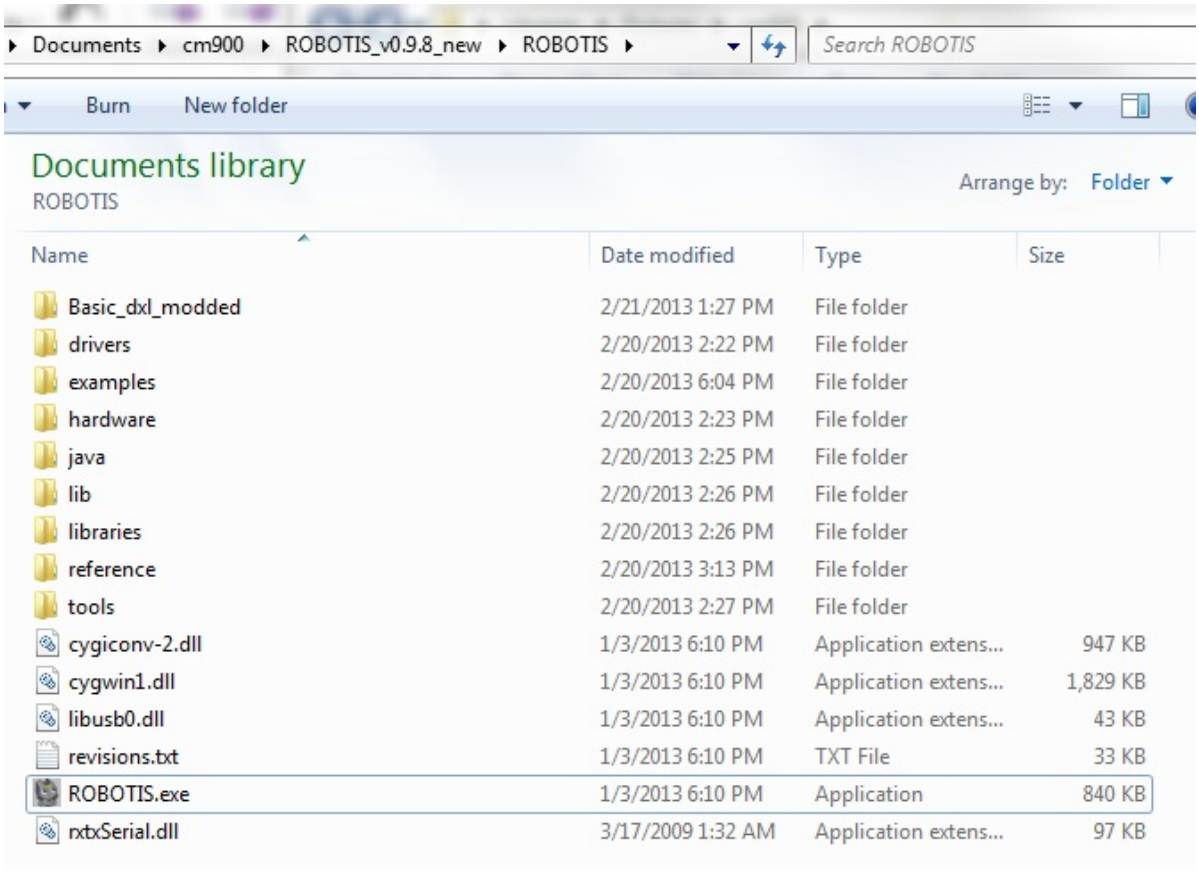
[Mac OS X] Tested in OS X 10.6.8
https://www.dropbox.com/s/3up2cq9gq5x2il7/ROBOTIS_v0.9.8_osx.dmg

[Linux 64bit] Tested in Ubuntu 12.04
https://www.dropbox.com/s/u07wp21yedm1egj/ROBOTIS_v0.9.8_linux64.tar.gz

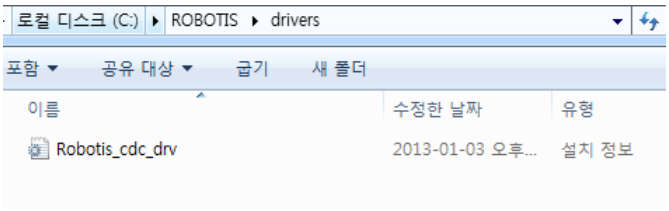
[Linux 32bit] Tested in Ubuntu 10.10
https://www.dropbox.com/s/y11chy26hlc886n/ROBOTIS_v0.9.8_linux32.tar.gz

② ROBOTIS CM-9 structure

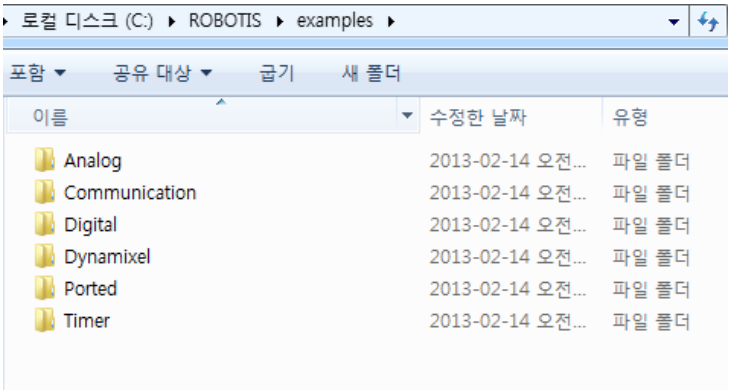
2. After decompressing the downloaded file the structure will appear as shown below.



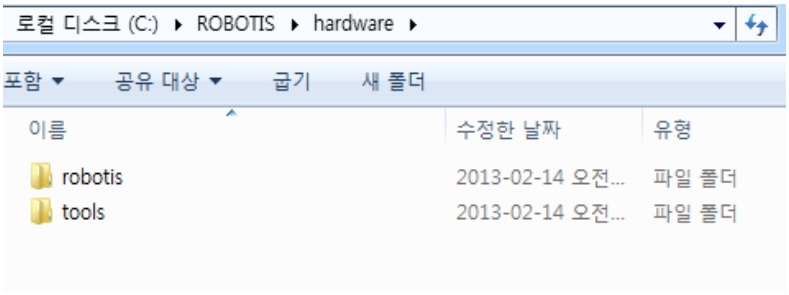
A. Drivers : contains the Windows .inf USB drivers



B. Examples : Contains the files for examples for ROBOTIS CM-9.

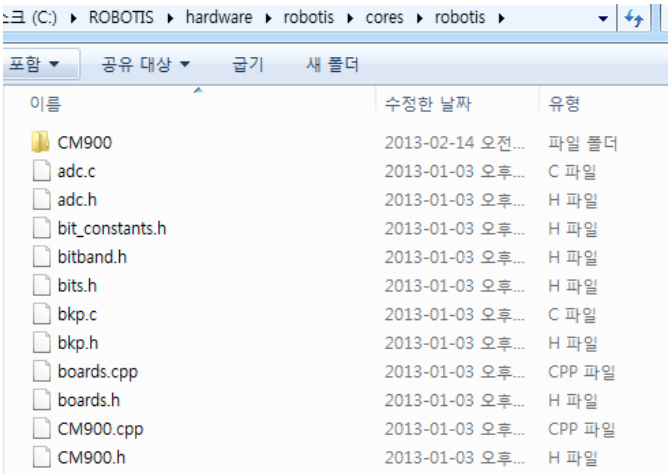


C. Hardware : contains the CM-9-series C/C++ sources + ARM-based compiler



Robotis folder contains the CM-900's API core library

ROBOTIS\hardware\robotis\cores\robotis

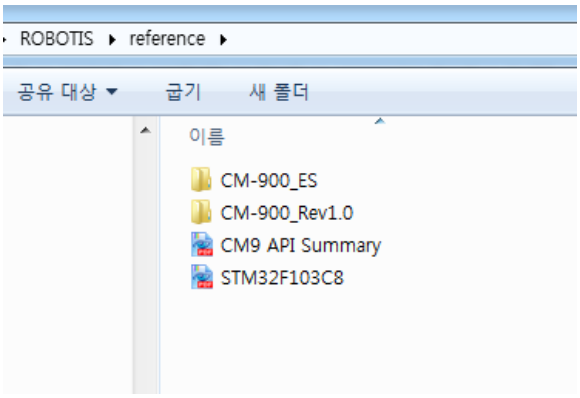


D. Java : contains JRE (Java Runtime Environment).

E. Lib : ROBOTIS CM-9 resources

F. Libraries : sketch libraries

G. Reference : CM-9-series data suite and API documentation



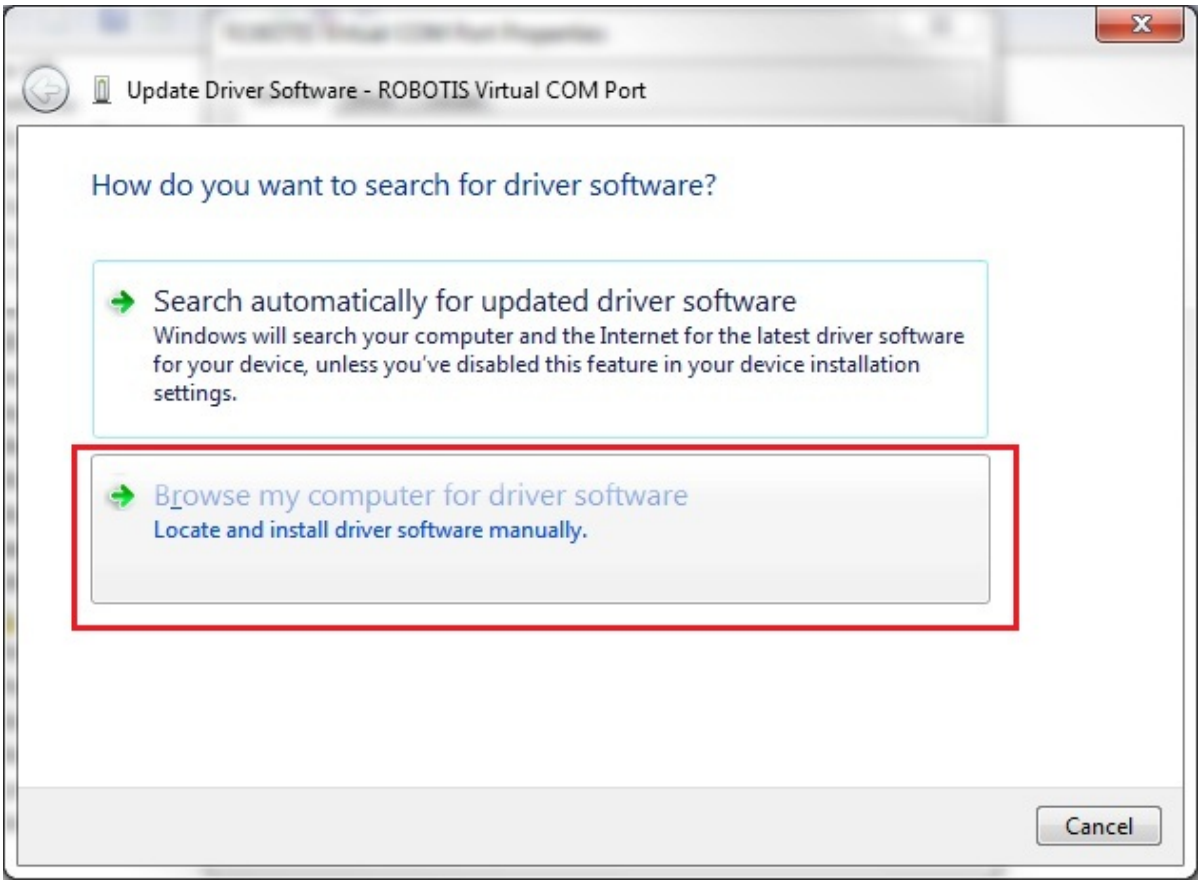
H. Tools : ROBOTIS CM-9's processing-related tools

I. ROBOTIS CM-9.exe : ROBOTIS CM-9's executable

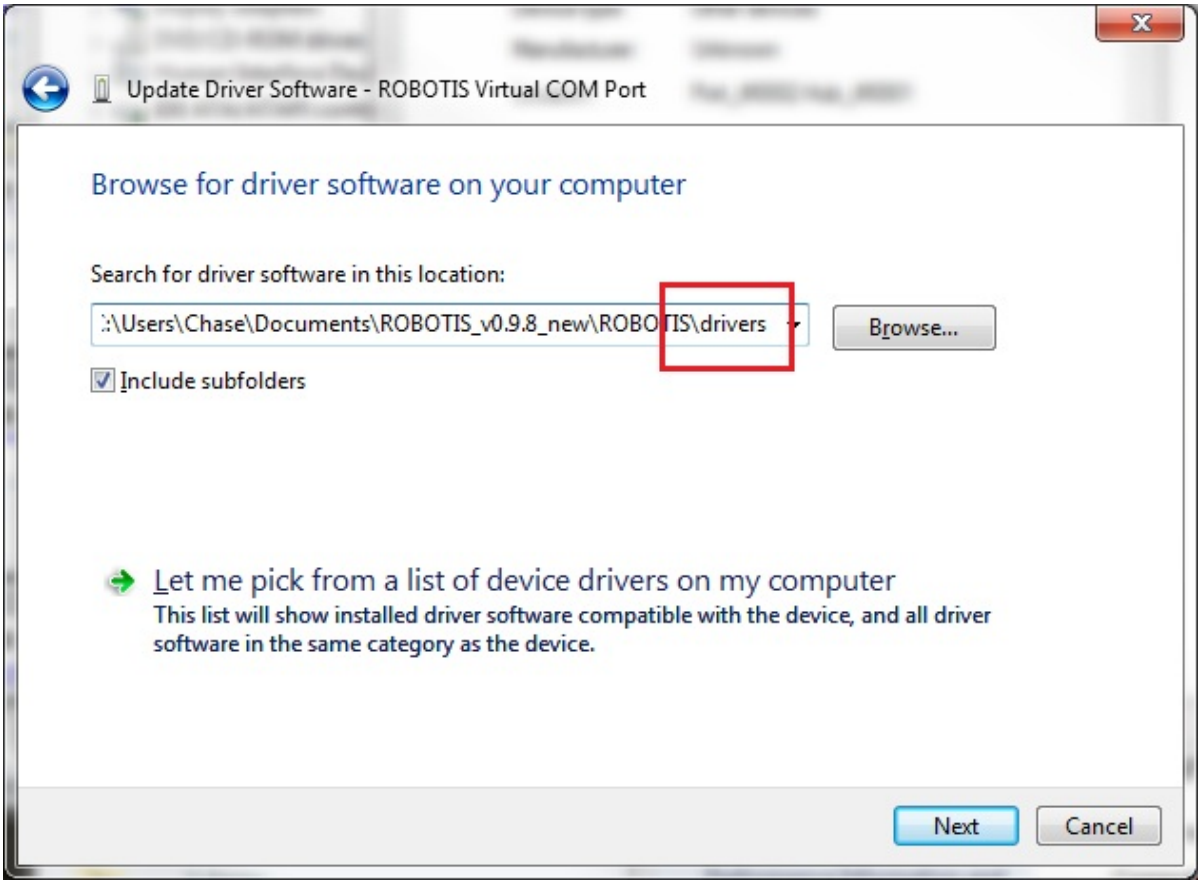
② USB drivers installation

The CM-900 USB driver installation is an essential requirement. The following procedure is Windows-specific. Mac OSX and Linux users do not need the following procedure as drivers are already included with the OS.

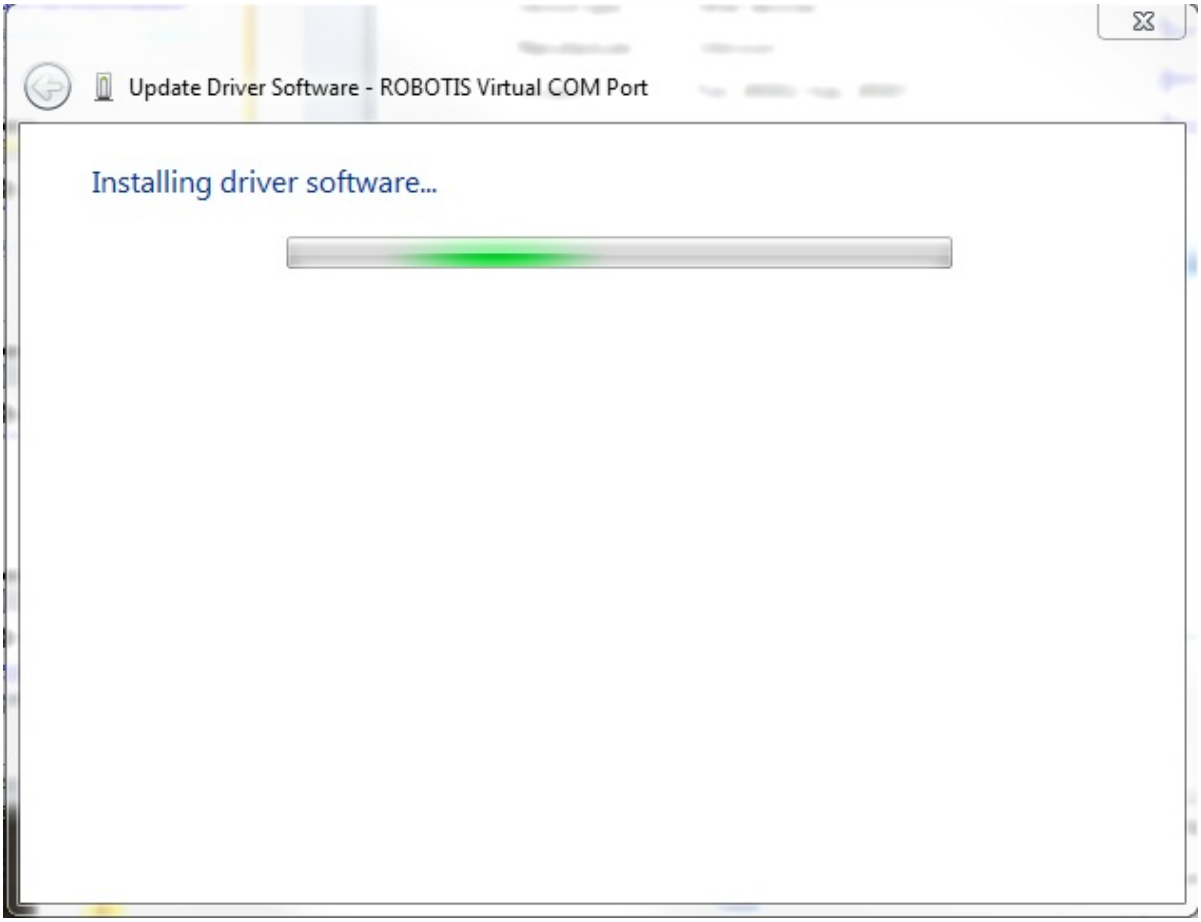
When the CM-900 is connected to the PC it will appear as ROBOTIS Virtual COM Port in Windows Device Manager. With the right mouse click select update driver software.



Pick “browse my computer for driver software”

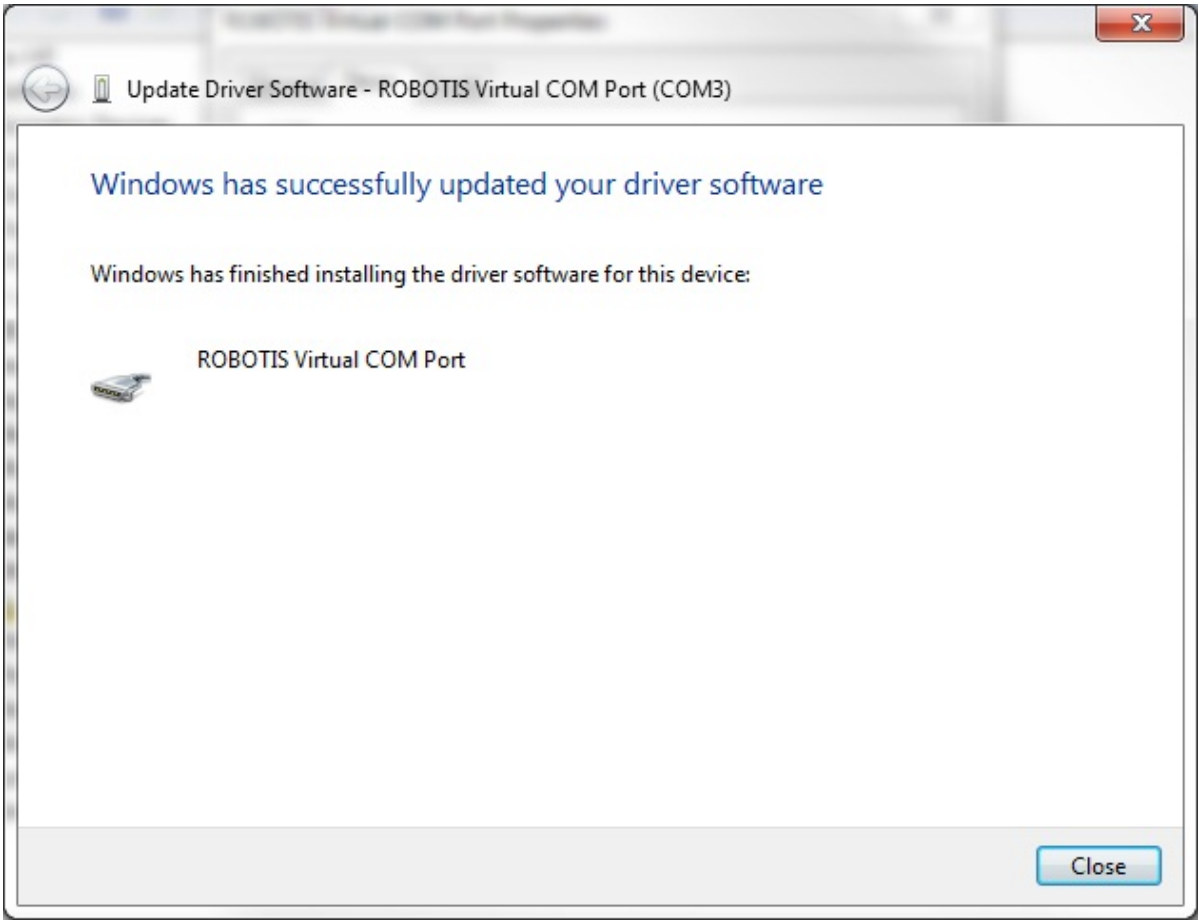


Click on “browse” and select ‘drivers’ folder (from ROBOTIS\drivers).



Click on “install this driver software anyway”

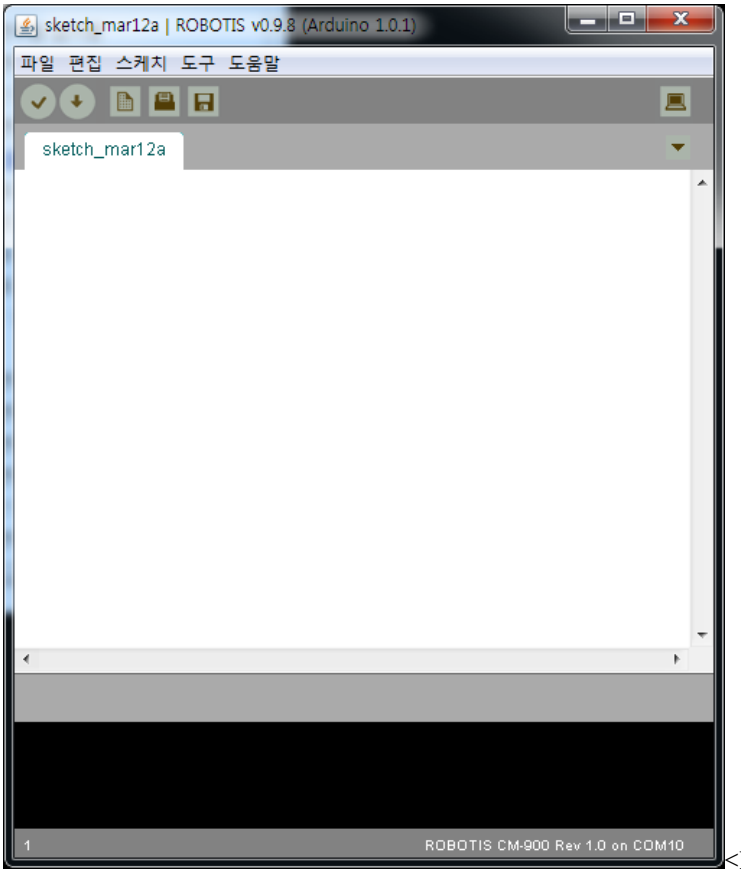
Once install is successful a window will appear as illustrated below



Look for the COM port number under ROBOTIS Virtual COM Port.

③ Software environment setup

After USB driver setup double-click on ROBOTIS CM-9.exe.

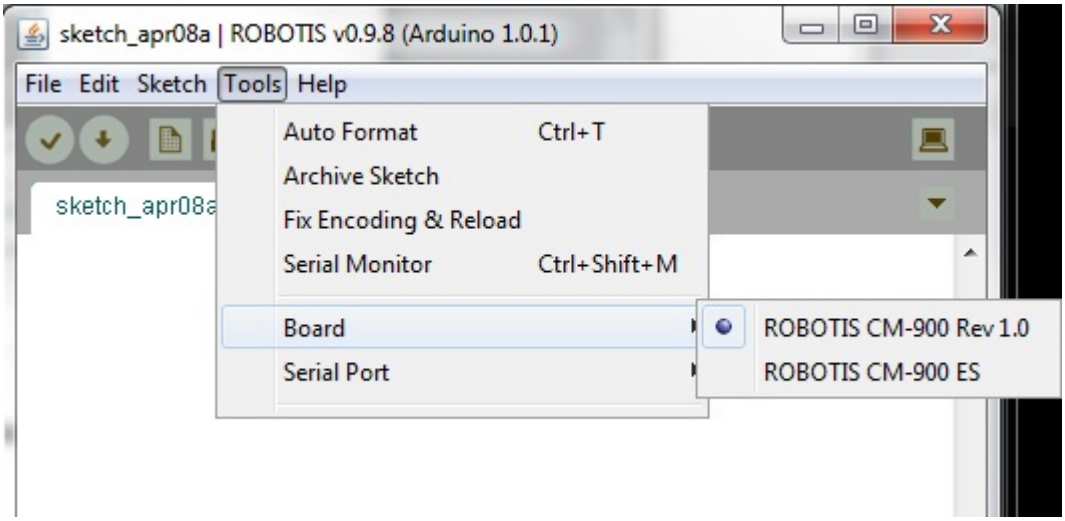


<ROBOTIS CM-9 window>

From ROBOTIS CM-9 window you must select a board type and COM number.

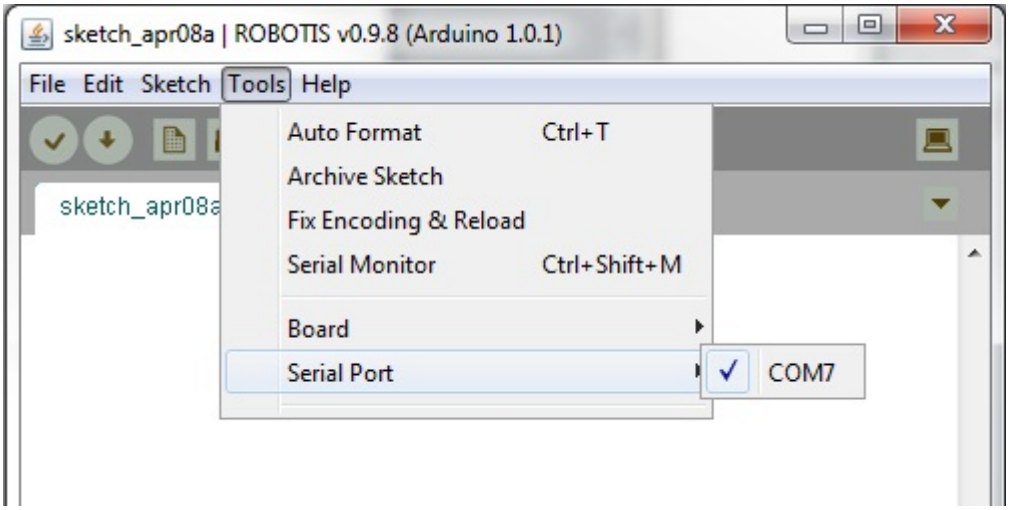
A. Select a board

Select the matching version of your CM-900 board. In this case select CM-900 REV 1.0 (ROBOTIS CM-900 ES is for previous test versions).

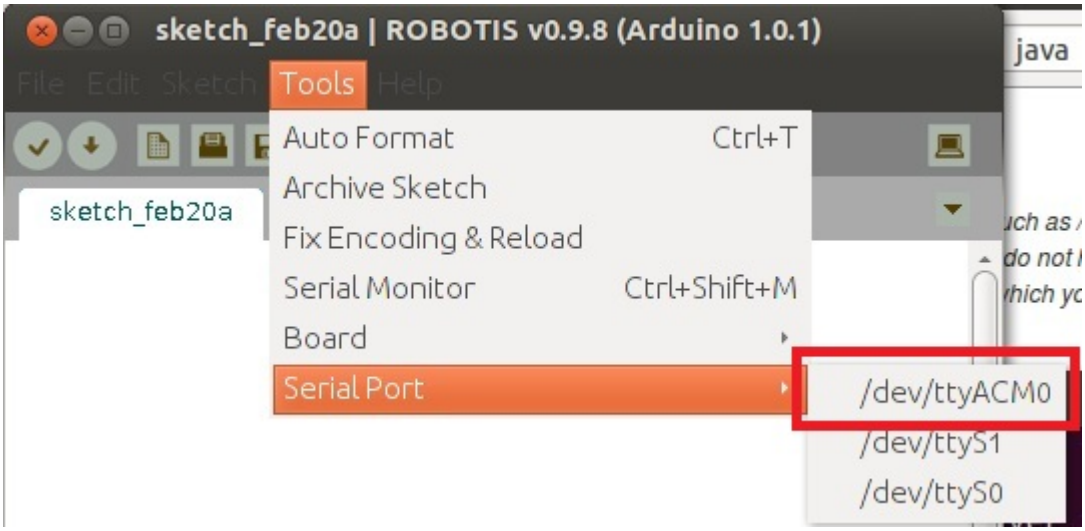


B. Select serial port

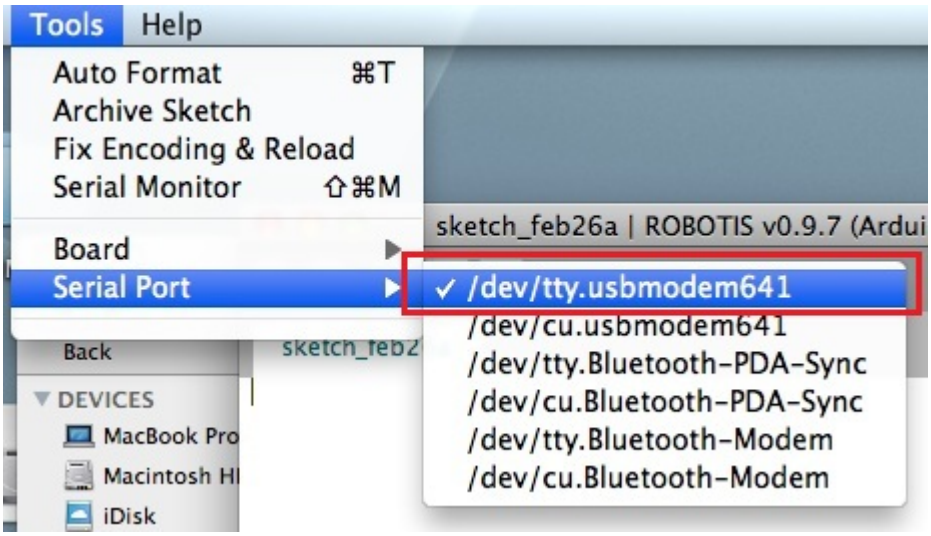
Select the COM port number.



Linux users select /dev/ttyACMX.

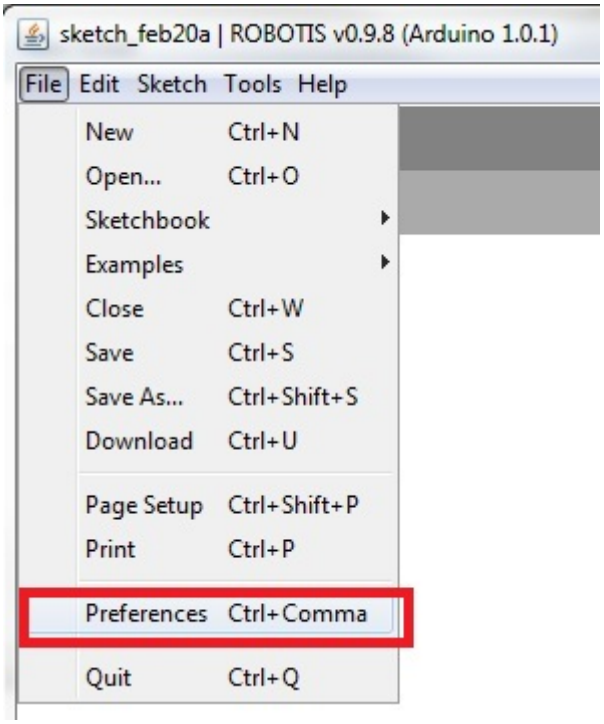


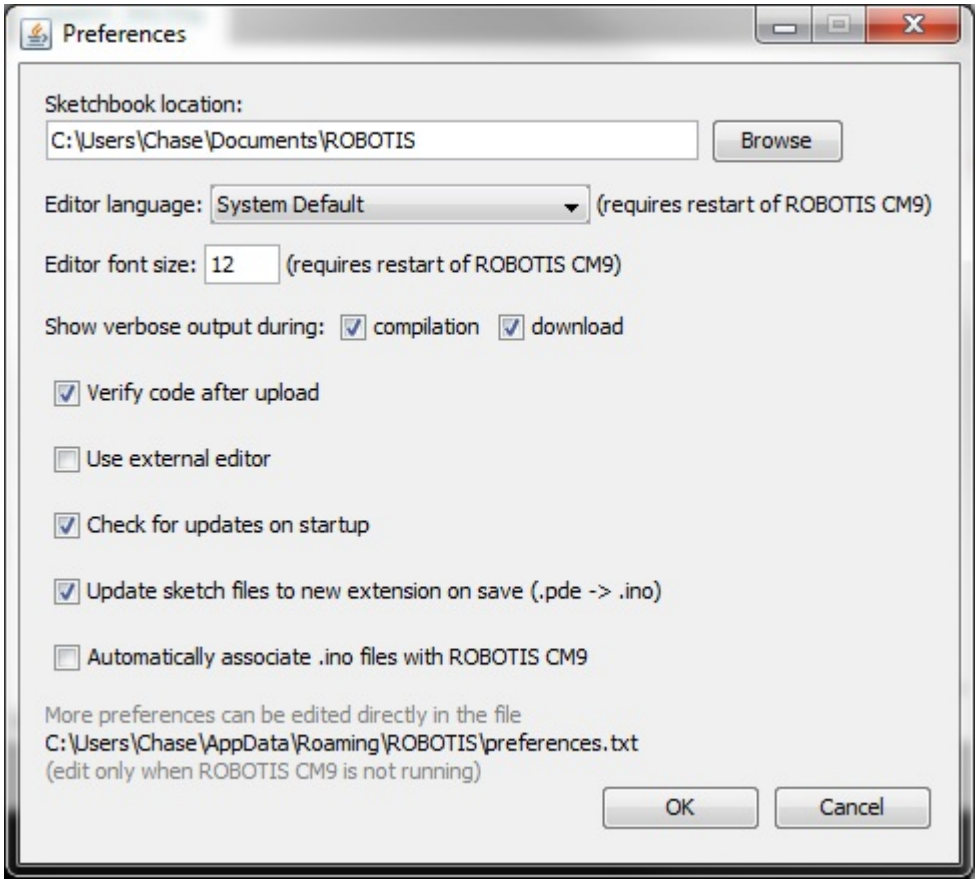
Mac OS X users select tty.usbmodemX11.



C. Environment setup

Go to File -> Preferences environment to make changes.





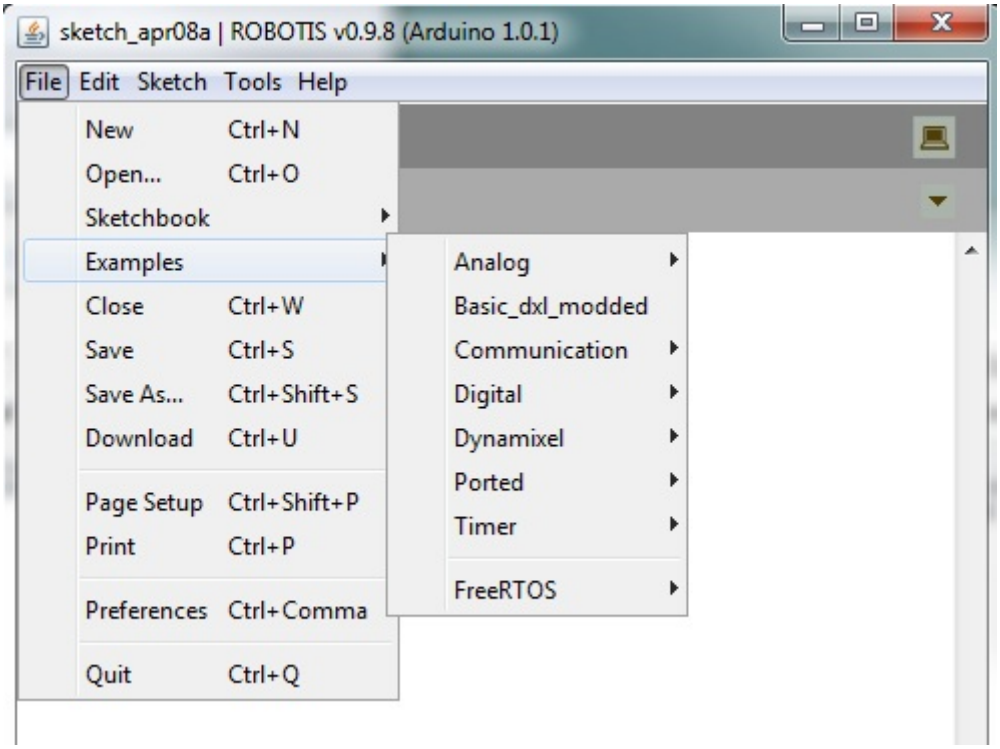
i. Sketchbook location: directory for sketch-based projects including examples.

ii. Editor language: change font type.

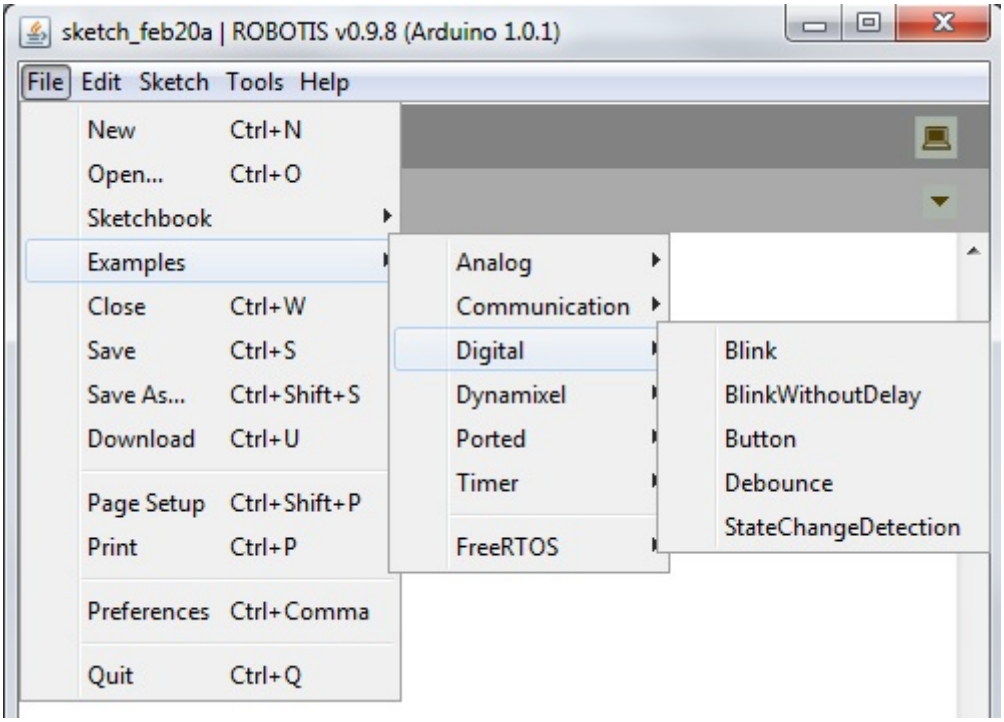
iii. Console window : view the compilation's output. Check download to download code after compilation.

④ Download examples

Get ROBOTIS CM-9 example programs from file -> examples.



For example: with Digital I/O open the Blink example, analyze the code then download it to the CM-900. This should help make development easier.



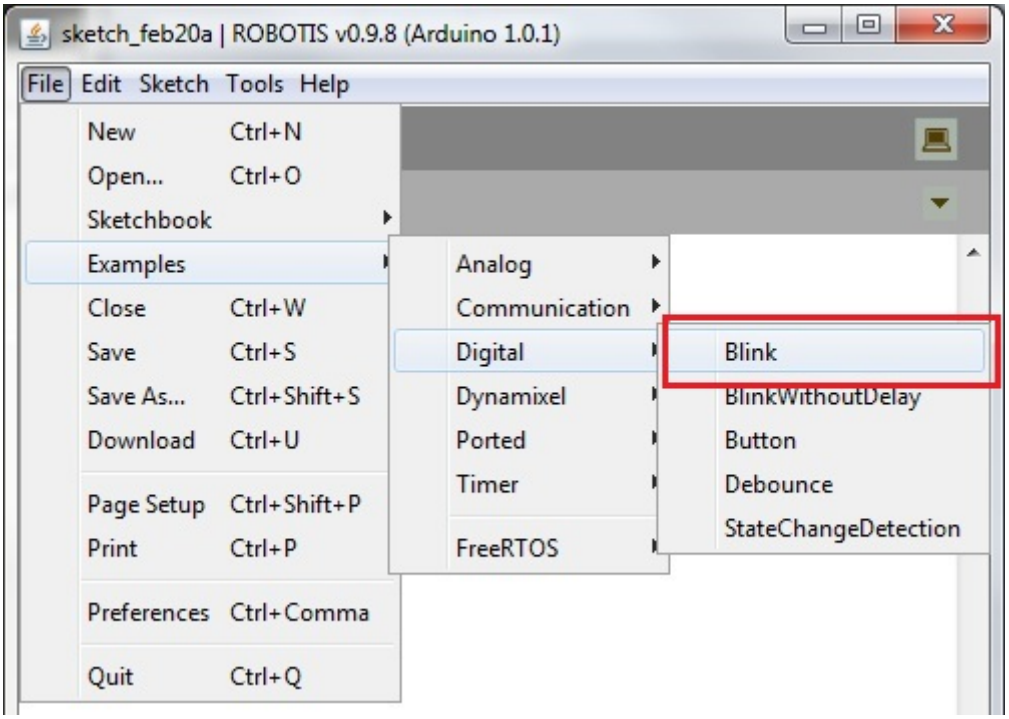
In this Blink example shown simply click on the downwards arrow to download the code to the CM-900.

The following examples are useful for API reference. Please refer to these when developing the CM-900.

⑤ Blink example

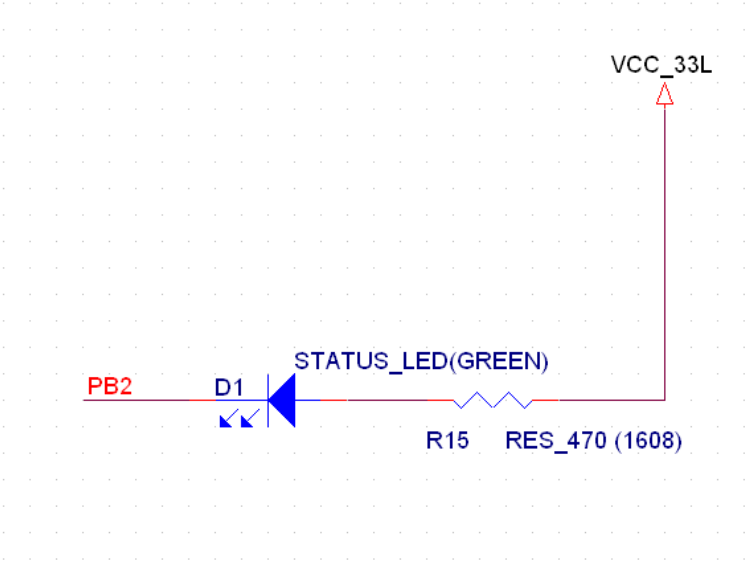
The CM-900 Blink example is a port of Arduino's Blink example.

Go to File -> Examples -> Digital -> Blink



A. Schematic

The CM-900's status LED connects to the CPU via D16(PB2).



When D16(PB2) is high the LED is off; when low the LED is on.

B. Sketch code

```
void setup() {
    // Set up the built-in LED pin as an output:
    pinMode(BOARD_LED_PIN, OUTPUT);
}

void loop() {
    digitalWrite(BOARD_LED_PIN, HIGH);
    delay(100);          // Wait for 1 second (1000 milliseconds)
    digitalWrite(BOARD_LED_PIN, LOW);
    delay(100);          // Wait for 1 second (1000 milliseconds)
}
```

The function **pinMode(pin_number, pin_mode)** function is used to unitialize.

Refer to the CM-900 I/O port silk screen; BOARD_LED_PIN is defined for pin D16. This is illustrated in the header file CM-900.h.

ROBOTIS\hardware\robotis\cores\robotis\CM-900.h

```
#ifndef CM_900_H_
#define CM_900_H_

#include "gpio.h"


#define CYCLES_PER_MICROSECOND  72
#define SYSTICK_RELOAD_VAL      71999 /* takes a cycle to reload */

#define BOARD_BUTTON_PIN        38
#define BOARD_LED_PIN           16
```

The Blink example is a simple high/low signal manipulator with OUTPUT being the output fuction.

Once setup() function has been set you can control the LED with **digitalWrite(pin_number, HIGH/LOW)**를 in the loop() via time with delay(millisecond).

C. Verify data

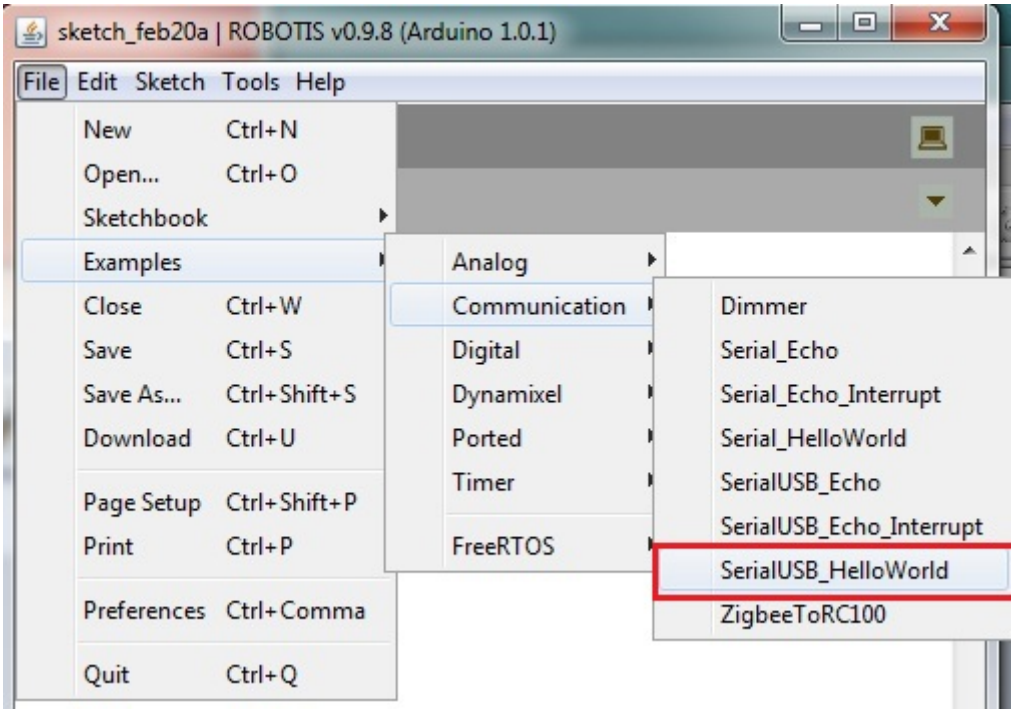
Verify the STATUS LED (on or off)

⑥ SerialUSB_HelloWorld example

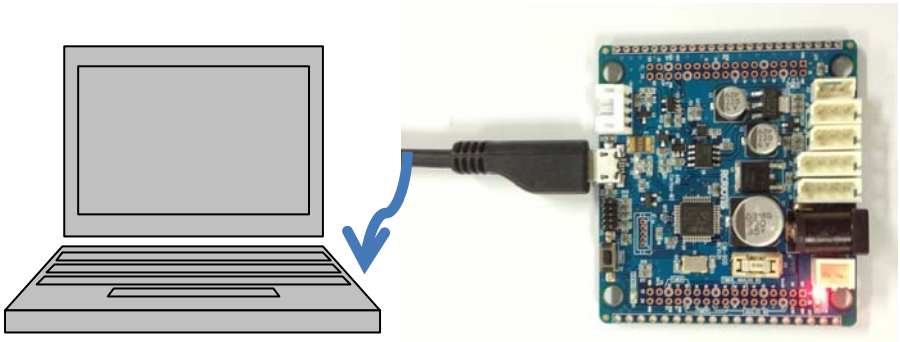
This is an example to communicate between the CM-900 and external device (i.e. PC) via USB. Declare SerialUSB instance to enable USB communications.

This example show how SerialUSB_HelloWorld communicated with a terminal window (PC).

Go to File -> Examples -> Communication -> SerialUSB_HelloWorld.



A. Connect the CM-900 to the PC



B. Sketch code

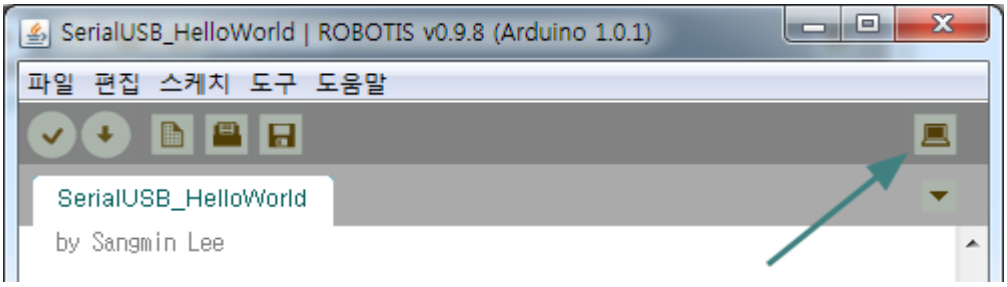
```
void setup() {
  //Initialize USB Serial
  SerialUSB.begin();
}
int nCount=0;
void loop() {
    //print "Hello World!!" to PC though USB Virtual COM port
    SerialUSB.println("Hello World!!");
    SerialUSB.print("nCount : "); // display nCount variable and increase
    SerialUSB.println(nCount++);
    delay(1000);
}
```

Initialize SerialUSB instance in Setup() with begin() method. The void() type returns nothing. Regardless of other serial devices with SerialUSB.begin() method setting the baud rate is not necessary.

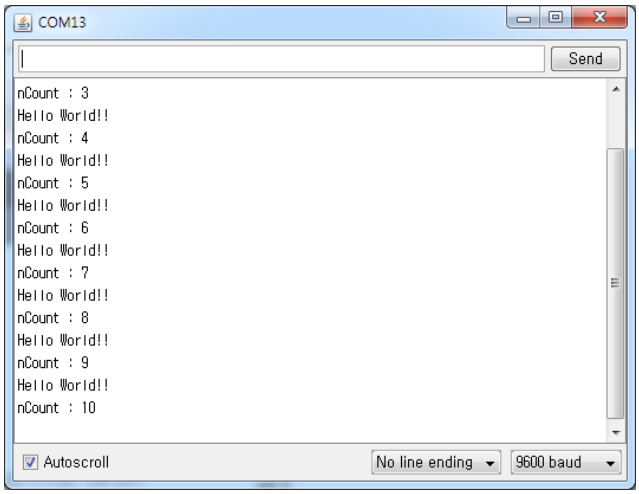
In Loop() with SerialUSB.print() or SerialUSB.println() its possible to get output.

C. Verify data

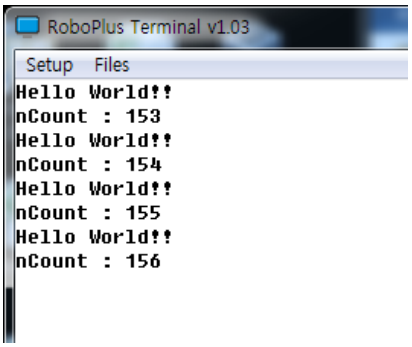
Click on the serial monitor to see output. This is also possible with RoboPlus Terminal.



The serial monitor window can be activated by clicking on the laptop icon located on the upper right side.



The same is possible with RoboPlus Terminal (no need to set baud rate).

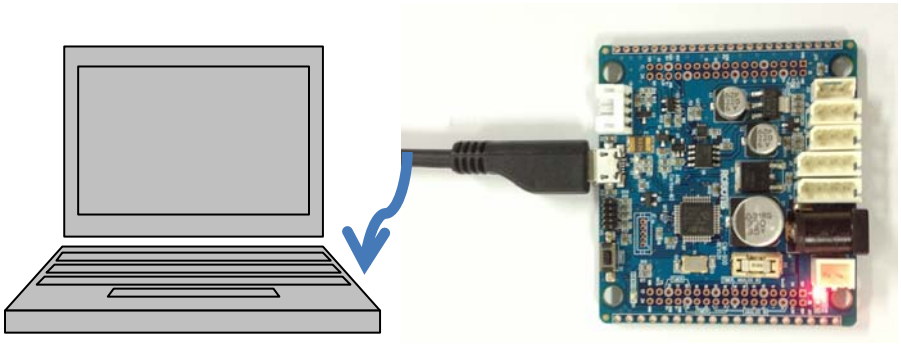


Other terminal window applications are not yet supported.

⑦ SerialUSB_Echo example

SerialUSB_HelloWorld example only showed output SerialUSB_Echo example allows for both input and output.

A. Connect the CM-900 to the PC



B. Sketch code

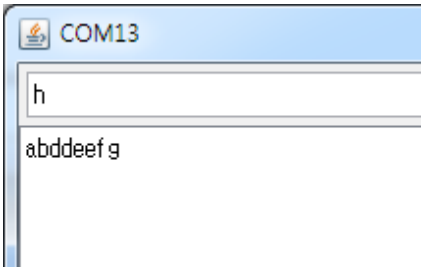
```
void setup(){
  //USB Serial initialize
  SerialUSB.begin();
}
void loop(){
  // when you typed any character in terminal
  if(SerialUSB.available()){
      //print it out though USB
      SerialUSB.print((char)SerialUSB.read());
  }
}
```

Like SerialUSB_HelloWorld there is no need to set baud rate in SerialUSB.begin().

In Loop() the CPU checks for input repeatedly. In the if clause SerialUSB.avaliable() outputs 0 until the condition is met. Once condition is met SerialUSB.read() sends 1 byte SerialUSB.print().

C. Verify data

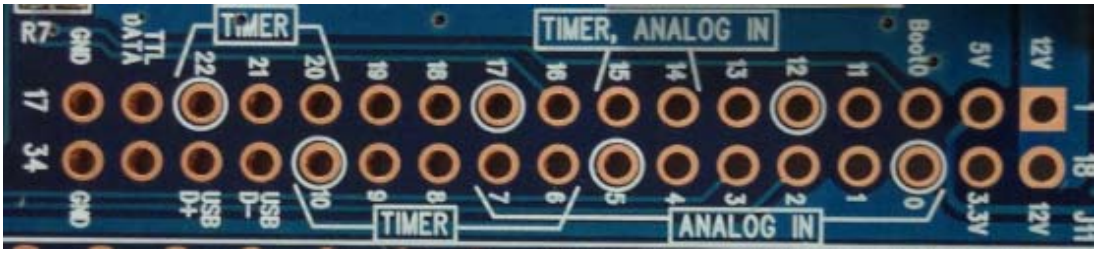
Use the serial monitor or RoboPlus Termincal to view data. Use the keyboard to input data and the CM-900 returns the same input as output, therefore is an echo.



Any input is returned exactly as output.

⑧ AnalogInSerial example

The CM-900 has a 12-bit resolution ADC with 10 ports. This makes possible to connect multiple devices. The silk screen below shows the available ports.



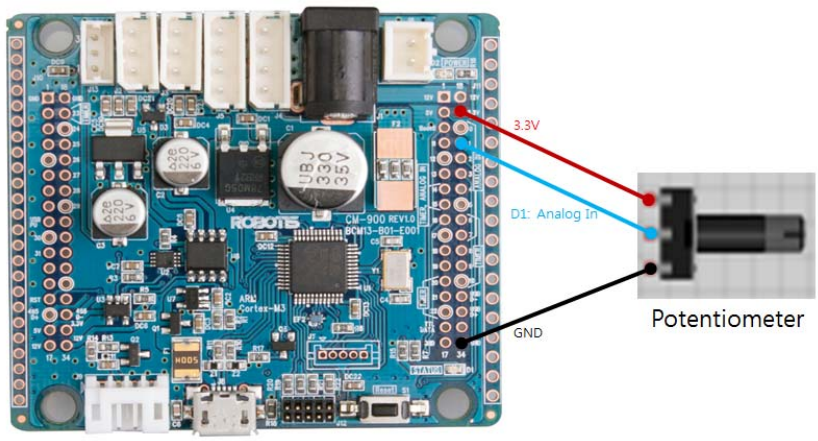
Pins 14 and 15 are for TIMER, ANALOG IN respectively and are duplicated features; with analog input via pinMode() function it is possible to set in analog mode. Input pins 0 through 7, 14 and 15 are available for analog input.

With AnalogInSerial example an analog input received then transmitted via SerialUSB.

This example is accredited to Tom Igoe for Arduino's board therefore this is an Arduino example.

A. Schematics

The CM-900 is connected to a variable resistor (potentiometer). The important point is that the maximum allowed input voltage of the CM-900 for analog inputs is 3.3V. The schematic below the variable resistor is implemented to limit the voltage to 3.3V.



B. Sketch code

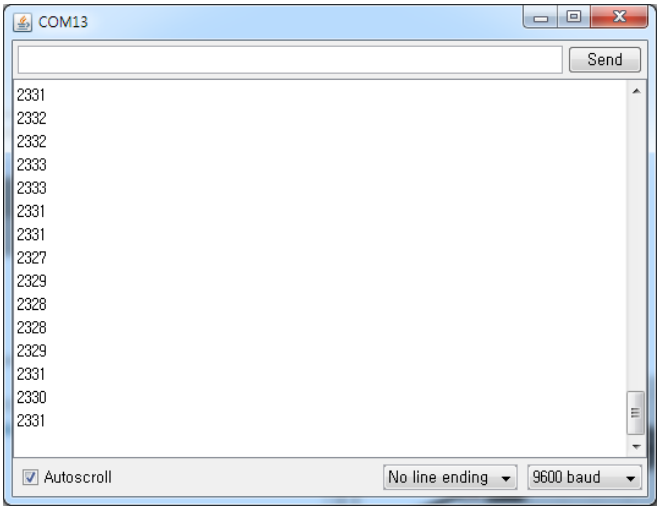
```
const int analogInputPin = 1;

void setup() {
  //USB Virtual COM port init(no need baud rate argument)
  SerialUSB.begin();
  // Declare analogInputPin as INPUT_ANALOG:
  pinMode(analogInputPin, INPUT_ANALOG);
}

void loop() {
  // Read the analog input into a variable:
  int analogValue = analogRead(analogInputPin);

  // print the result:
  SerialUSB.println(analogValue);
  //need some delay because coming out too fast from USB COM port
  delay(100);
}
```

This example shows 2 declarations in setup(). In loop() int analogValue repeatedly looks for analogRead(pin_number) for analog input. The input value is of integer type with 12 bits in range (0-4095).

SerialUSB.println() outputs value(s) from analogValue. If a hexadecimal value output is desired then set SerialUSB.println(analogValue,16), where 16 denotes hexadecimal; for Binary then 2; octal then 8. The default value is in decimal.

C. Verify data

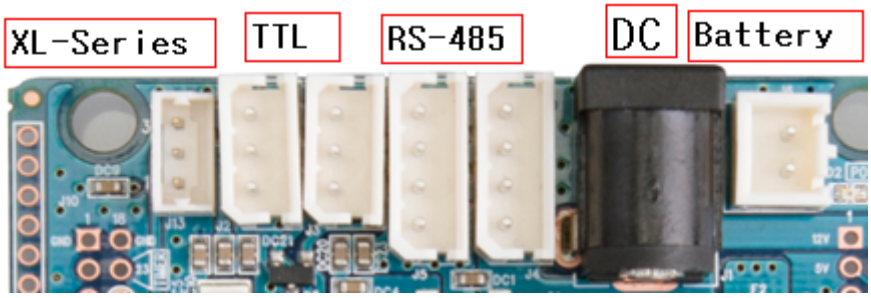
Open the serial monitor to see output.



⑨ Dynamixel Basic example

The CM-900 includes Dynamixel connectors to facilitate robot development. A pair of 3-pin TTL, a pair of 4-pin RS-485, and a XL-series connector are embedded onto the board. Also, a DC jack and battery connector are also embedded so power can be properly supplied to any connected Dynamixel(s).

Dynamixel Basic example is analogous to the Blink example as it switches Dynamixel between one position to another.

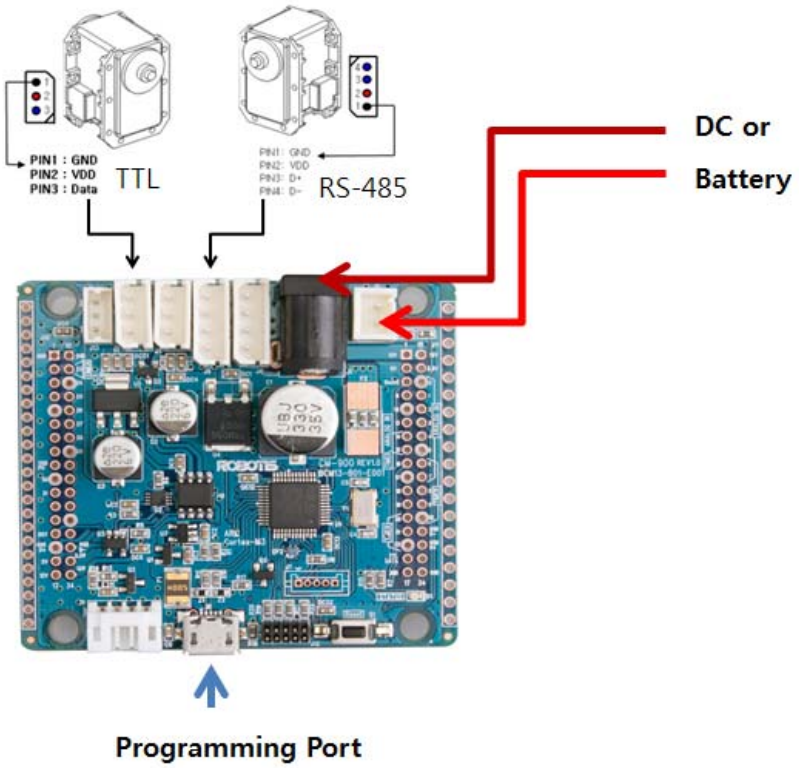


A. Connecting a Dynamixel

Connect a 3-pin or 4-pin DYnamixel. Connect the SMPS or battery to the CM-900 then run ROBOTIS CM9.

The default values for Dynamixel are 1 for ID and 1 for baud rate (1Mbps). If not, then set said values with Dynamixel Wizard.

The CM-900 communicates with Dynamixel serially.



B. Sketch code

```
void setup() {
    // Initialize the dynamixel SDK:
    Dxl.begin(1);
}

void loop() {
    delay(1000);              // Wait for 1 second (1000 milliseconds)
    Dxl.writeWord(1, 30, 100); //Turn dynamixel ID 1 to position 100
    delay(1000);              // Wait for 1 second (1000 milliseconds)
    Dxl.writeWord(1, 30, 1000);//Turn dynamixel ID 1 to position 1000
}
```

Dynamixel bus must be initialized. From setup() Dxl.begin(baud_rate) is also initialized. From here any 3-pin or 4-pin Dynamixel device connected to the CM-900 gets initialized. Baud_rate value of 1 means communications speed is set to 1Mbps. For further information on Dynamixel API please consult the e-manuals.

From loop() with Dxl.writeWord(ID, Address, Value) function set the value for goal position(L) in Address; this corresponds to position portion of Dynamixel, and Value being the value of the position. In this example the position switches between 100 to 1000 in intervals of 1000ms.

Note that the actual position varies with different models of Dynamixel. For Dynamixelw with 12-bit resolution (0~4095, 0xFFF) will have a smaller range of motion and reach goal position quicker. For more information on goal position please consult the e-manuals.

C. Verify data

The only way to verify data is to check motor movement visually.

⑩ Dynamixel ReadWrite example

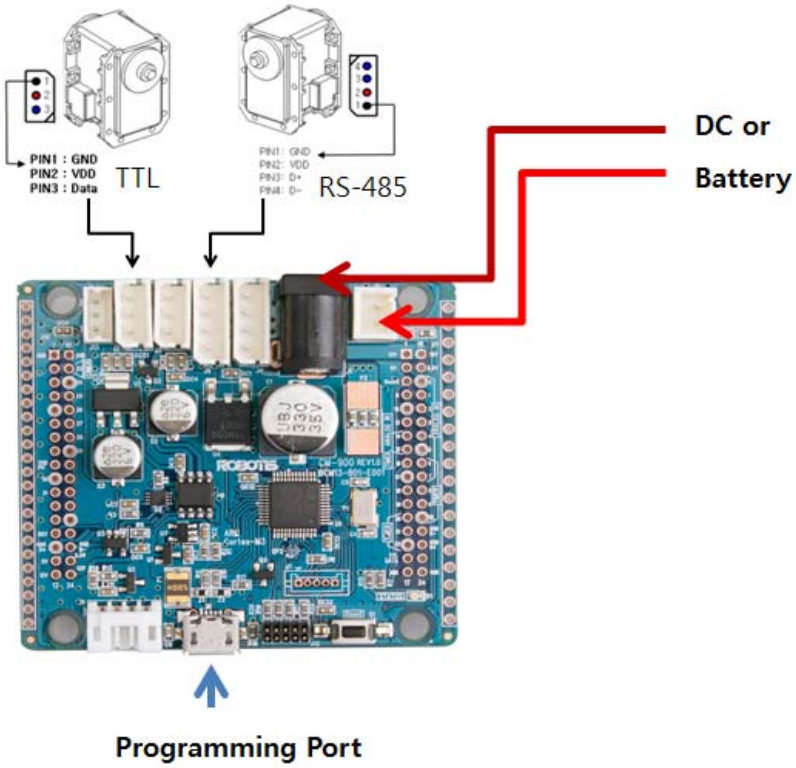
This example shows Dynamixel read/write features. This example checks Dynamixel movement and change of moving (rotating) direction. Once moving is complete position data is then outputted and Dynamixel moves to the next position.

A. Connect Dynamixel

Connect a 3-pin or 4-pin DYnamixel. Connect the SMPS or battery to the CM-900 then run ROBOTIS CM9.

The default values for Dynamixel are 1 for ID and 1 for baud rate (1Mbps). If not, then set said values with Dynamixel Wizard.

The CM-900 communicates with Dynamixel serially.



B. Sketch code

Some of the parameters from Dynamixel control table have been dfined in the preprocessor for simplicity.

```
#define P_GOAL_POSITION_L      30
#define P_PRESENT_POSITION_L   36
#define P_MOVING               46

word  Position;
word  wPresentPos;
byte  INDEX = 0;
byte  bMoving, CommStatus;
byte  id = 1;
word  GoalPos[2] = {0, 1023};

void setup() {
  Dxl.begin(1);
  //print to USB port
  SerialUSB.begin();
}
```

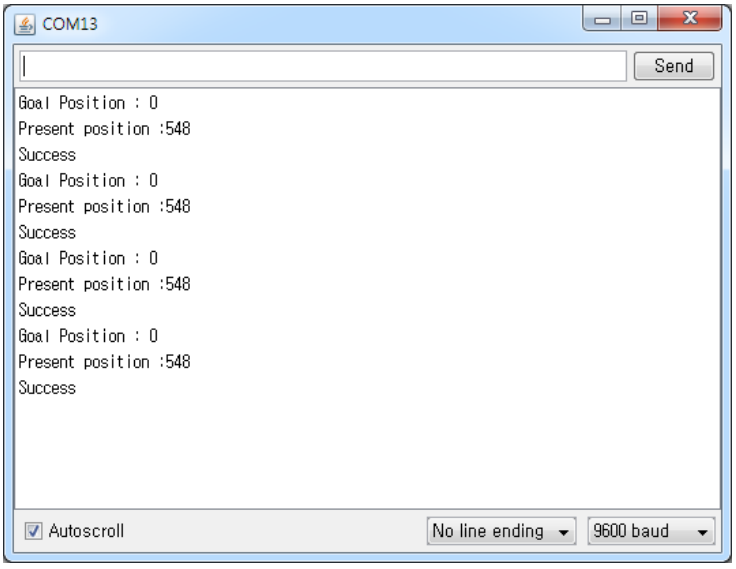
```
void loop() {
  bMoving = Dxl.readByte( id, P_MOVING);
  CommStatus = Dxl.getResult();
  if( CommStatus == COMM_RXSUCCESS ){
    if( bMoving == 0 ){
      // Change goal position
      if( INDEX == 0 )
        INDEX = 1;
      else
        INDEX = 0;
      // Write goal position
      Dxl.writeWord( id, P_GOAL_POSITION_L, GoalPos[INDEX] );
    }
    // Read present position
    wPresentPos = Dxl.readWord( id, P_PRESENT_POSITION_L );
    SerialUSB.print("Goal Position : ");
    SerialUSB.println(GoalPos[INDEX]);
    SerialUSB.print("Present position :");
    SerialUSB.println(wPresentPos);
    SerialUSB.println("Success");
  }else {
    SerialUSB.println("Fail");
  }
  delay(1000);
}
```

bMoving = Dxl.readByte(id, P_MOVING) returns a 1 when Dynamixel is moving and 0 when not. If transmission via Dxl.getResult() is successful and bMoving = 0 the Goal Position's index changes; Dxl.writeWord(id, P_GOAL_POSITION, GoalPos[INDEX]) transmits new data. Value from GoalPos[INDEX] is outputted via USB via the following command

wPresentPos = Dxl.readWord(id, P_PRESENT_POSITION_L);

C. Verify data

Open up serial monitor to see output from GoalPos[INDEX] and see position of Dynamixel visually.



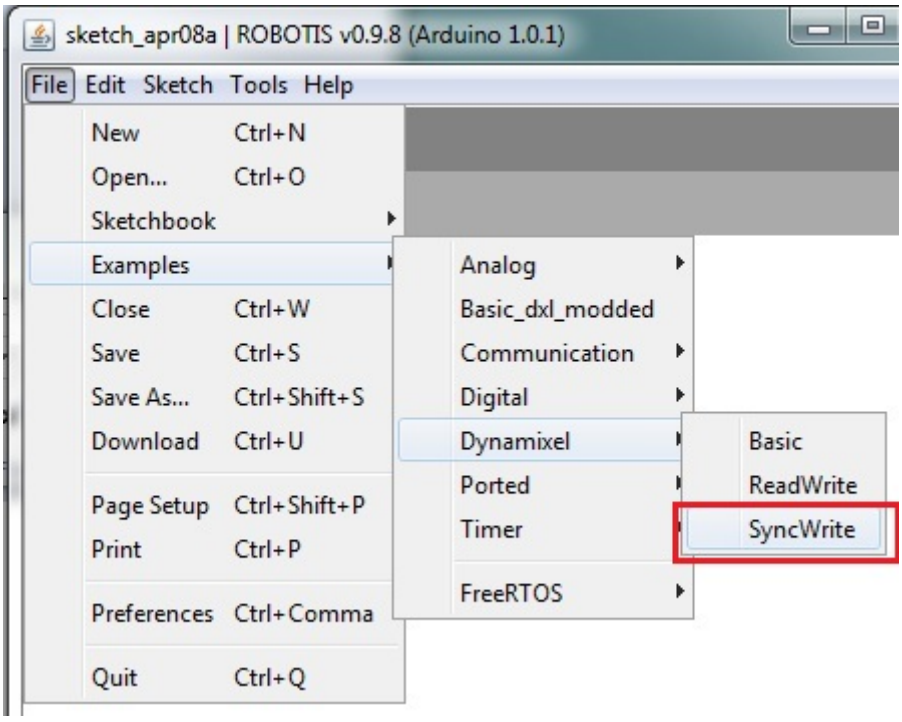
⑪ Dynamixel SyncWrite example

With Dynamixel Broadcast ID its possible to control multiple Dynamixels simultaneously.

This example shows how to control 5 Dynamixels via Syncwrite packet. For more information on Syncwrite please consult the e-manuals.

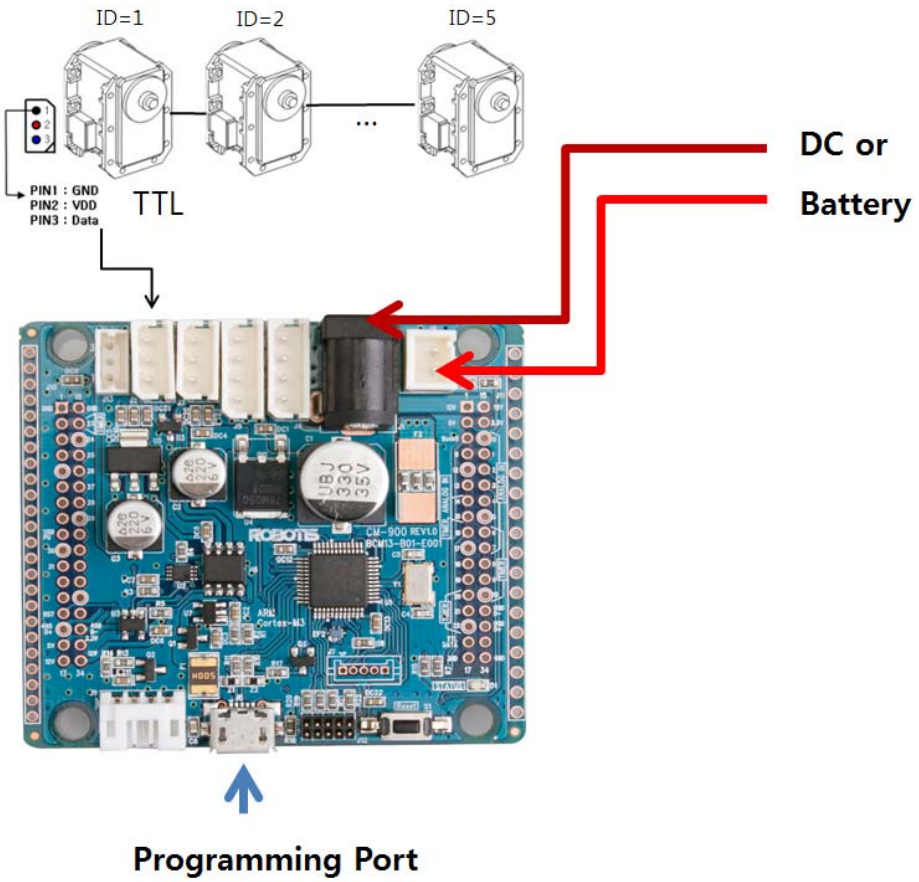
http://support.robotis.com/ko/e-manual_kor.htm#product/dynamixel/communication/dxl_instruction.htm

Go to FIle -> Examples -> Dynamixel -> SyncWrite



A. Connect 5 Dynamixels

Set ID from 1 to 5 use either 3-pin or 4-pin Dynamixel, or a combination of 5 using both pin types; connect them in any order. Set baud rate to 1Mbps to all 5 Dynamixels.



The CM-900 communicates with the Dynamixels serially.

B. Sketch code

Some of the parameters from Dynamixel control table have been defined in the preprocessor. For more information on Dynamixel control table please consult the e-manuals.

Note that 1-byte Word LOW (LSBs) is enough for control.

```
#define P_GOAL_POSITION_L   30
#define P_GOAL_SPEED_L      32

#define NUM_ACTUATOR        5 // Number of actuator
#define MAX_POSITION        1023
```

AmpPos is the initial position of all 5 Dynamixels.

```
word  AmpPos = 512;
word  wPresentPos;
word  GoalPos = 0;
byte  id[NUM_ACTUATOR];
byte  CommStatus;
byte  i;
```

Initial position

Dynamixel bus initialized in setup() with Dxl.begin(1) along with SerialUSB.begin().

```
void setup() {
  Dxl.begin(1);
  SerialUSB.begin();
  //Insert dynamixel ID number to array id[]
  for(i=0; i<NUM_ACTUATOR; i++ ){
    id[i] = i+1;
  }
  // Set goal speed
  Dxl.writeWord( BROADCAST_ID, P_GOAL_SPEED_L, 0 );
  // Set goal position
  Dxl.writeWord( BROADCAST_ID, P_GOAL_POSITION_L, AmpPos );
  delay(1000);
}
```

In loop() a Syncwrite packet can be divided for Dynamixel communications and output. For packet creation instructions please consult the e-manuals.

ID 0XFE
Length (L+1) X N + 4 (L: Data Length per RX-64, N: the number of RX-64s)
Instruction 0X83
Parameter1 Start address to write Data
Parameter2 Length of Data to write
Parameter3 First ID of RX-64
Parameter4 First data of the first RX-64
Parameter5 Second data of the first RX-64
...
Parameter L+3 Lth Data of the first RX-64
Parameter L+4 ID of the second RX-64
Parameter L+5 First data of the second RX-64
Parameter L+6 Second data of the second RX-64
...
Parameter 2L+4 Lth data of the second RX-64

Generally, in the event 1 command packet is 4 byte, 26 Dynamixel can be controlled simultaneously, Make sure that the length of packet does not to exceed 143 bytes since the volume of receiving buffer of RX-64 is 143 bytes.

Please note a word (2 bytes) in a Dynamixel packet includes both High byte (MSBs) and Low byte word (LSBs).

```
void loop() {
// Make syncwrite packet
  Dxl.setTxPacketId(BROADCAST_ID);                                    1
  Dxl.setTxPacketInstruction(INST_SYNC_WRITE);                        2
  Dxl.setTxPacketParameter(0, P_GOAL_POSITION_L);                     3
  Dxl.setTxPacketParameter(1, 2);                                     4

  for( i=0; i<NUM_ACTUATOR; i++ ){
    Dxl.setTxPacketParameter(2+3*i, id[i]);                           5
    Dxl.setTxPacketParameter(2+3*i+1, Dxl.getLowByte(GoalPos));       6
    Dxl.setTxPacketParameter(2+3*i+2, Dxl.getHighByte(GoalPos));

    SerialUSB.println(GoalPos);                                       7
  }
  Dxl.setTxPacketLength((2+1)*NUM_ACTUATOR+4);                        8
  Dxl.txrxPacket();                                                   9

  CommStatus = Dxl.getResult();
  //SerialUSB.print("CommSatus = ");SerialUSB.println(CommStatus);
  if( CommStatus == COMM_RXSUCCESS ){
    PrintCommStatus(CommStatus);
  }
  else{
    PrintErrorCode();
  }
  GoalPos += 100;                    Report result of CommStatus

  if( GoalPos > MAX_POSITION )
    GoalPos -= MAX_POSITION;
  delay(CONTROL_PERIOD);

}
```

#1: Syncwrite Packet set to Broadcast ID.

#2: set Instruction Sync Write (0x83)

#3: Goal Position parameter with value 0.

#4: assign a word (2 bytes) to Goal Position.

#5: Assignment for IDs and Parameters

(data length +1)*(index value i=0,1,2,…) + 2(BROADCAST_ID, INST_SYNC_WRITE)

#6: set word (2 bytes) for Goal Position.

#7: output goal position via USB

#8: calculates Packet length (see below)

Length (L+1) X N + 4 (L:RX-64별 Data Length, N:RX-64의 개수)

#9: the created Packet is transmitted via Dxl.txrxPacket() method

i. Verify data

Open the serial monitor to see GoalPos[INDEX] of all 5 Dynamixels.