DOCUMENTS RELATED TO HARDWARE

Circuit Diagram:

This is the image of general purpose board that we made.

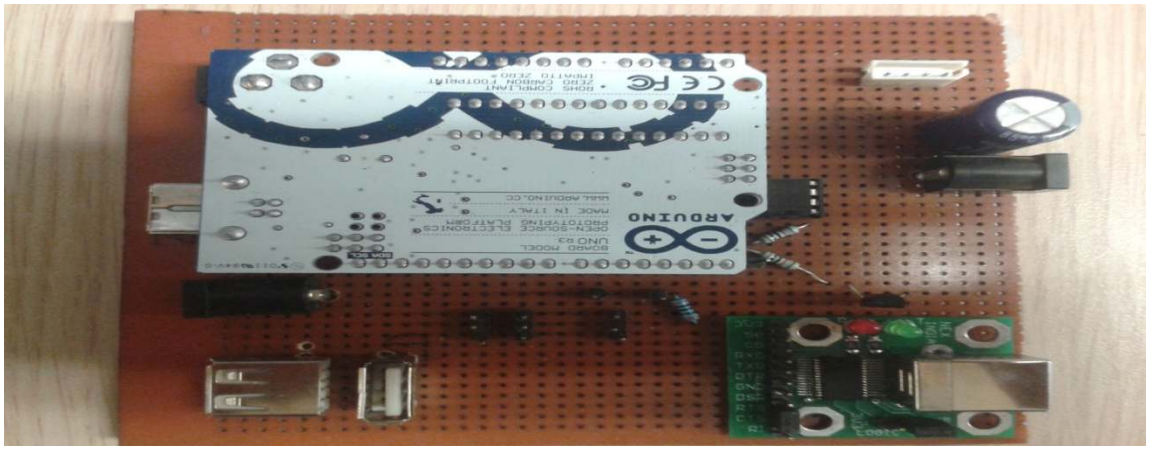


Figure 1

The naming of the pins is as follows:

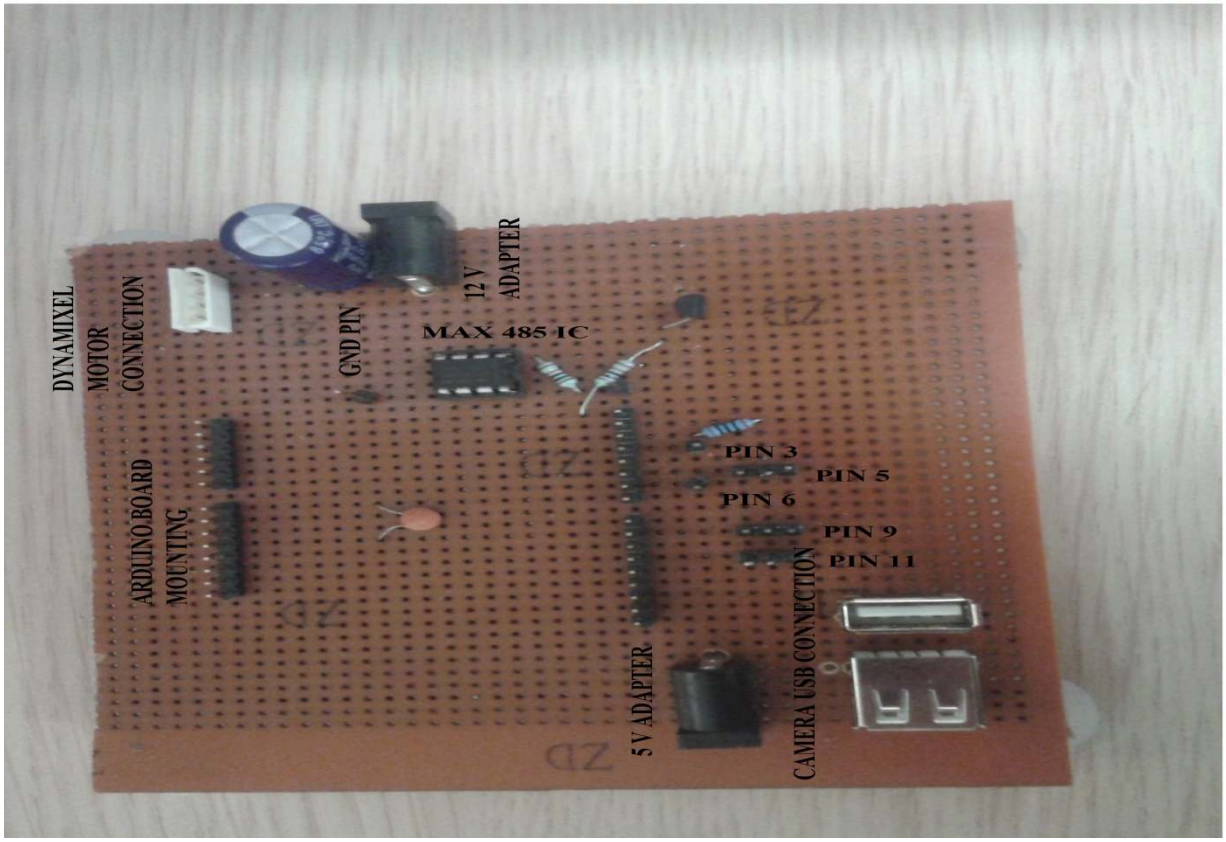


Figure 2

In the above picture, there are pins soldered on which arduino is mounted. The supply voltage is indicated accordingly. The connections for the servo motors and dynamixel motors are given. The above image shows PIN 3 and PIN6 written. These pins are used for serial communication with the python.

PIN 3 behaves as the Rx of arduino connected to Tx of USB to serial whose image will be found below.

PIN 6 behaves as the Tx of the arduino connected to Rx of USB to serial.

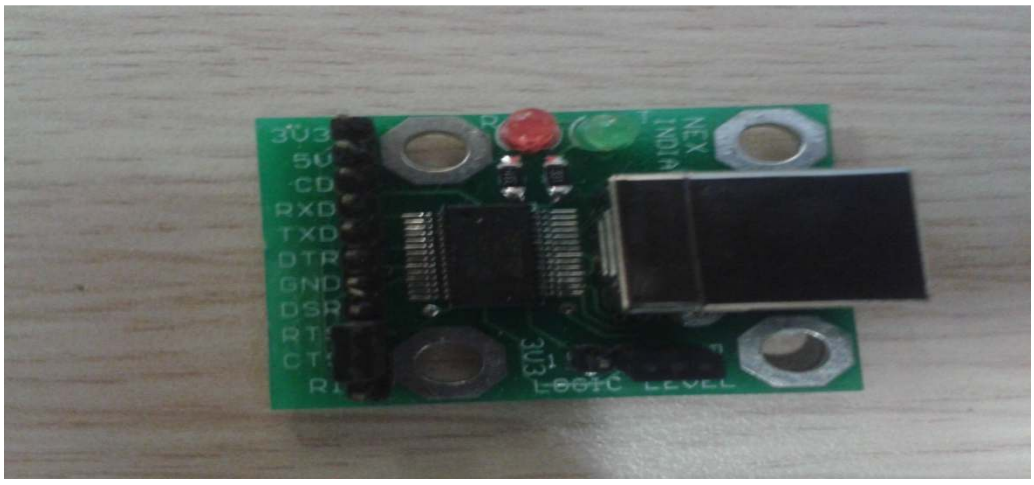


Figure 3

The labeling of motor wires is done in the following image.

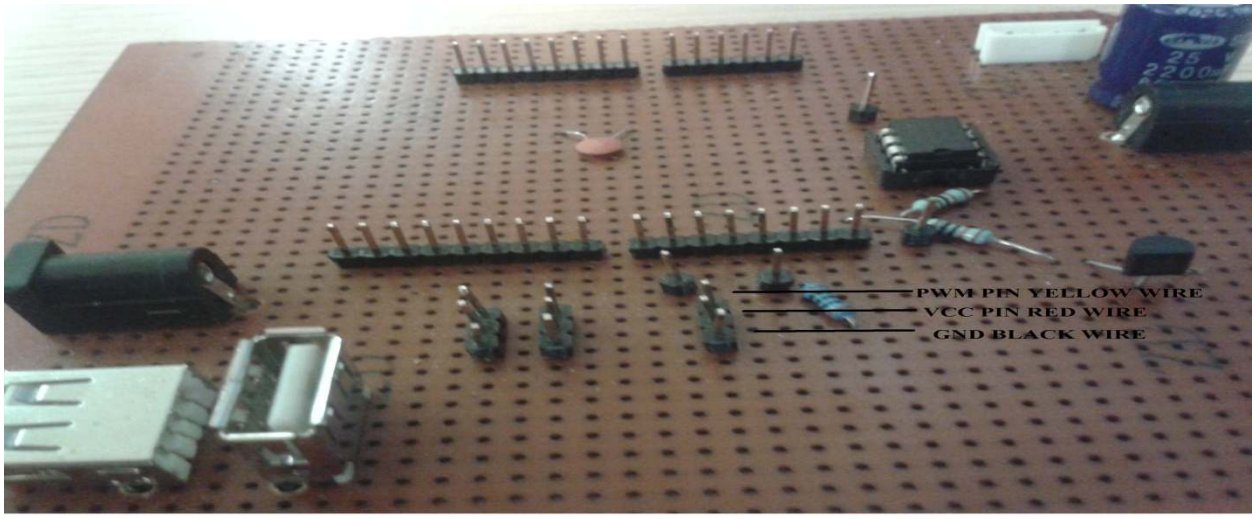


Figure 4

Above all the ground lines are short. That is Pin 5, pin 9, pin 11 have common ground. Similarly for Vcc line. It is connected to 5V supply as shown in the figure 2.

By convention PWM line is of yellow color wire, Vcc red color wire and Gnd black color wire.

There are 3 servo motors used

1) Crankshaft servo motor. The name to this motor is given as 5 and the label '5' is attached to the motor wires. This motor is to be connected to PIN 5 as labeled in the image 2.
2) Gripper motor. The name to this motor is given as 11 and the label '11' is attached to the motor wires. This motor is to be connected to PIN 9 as labeled in the image 2.
3) Roll motor. The name to this motor is given as 9 and the label '9' is attached to the motor wires. This motor is to be connected to PIN 11 as labeled in the image 2.

Connection of Dynamixel motors:

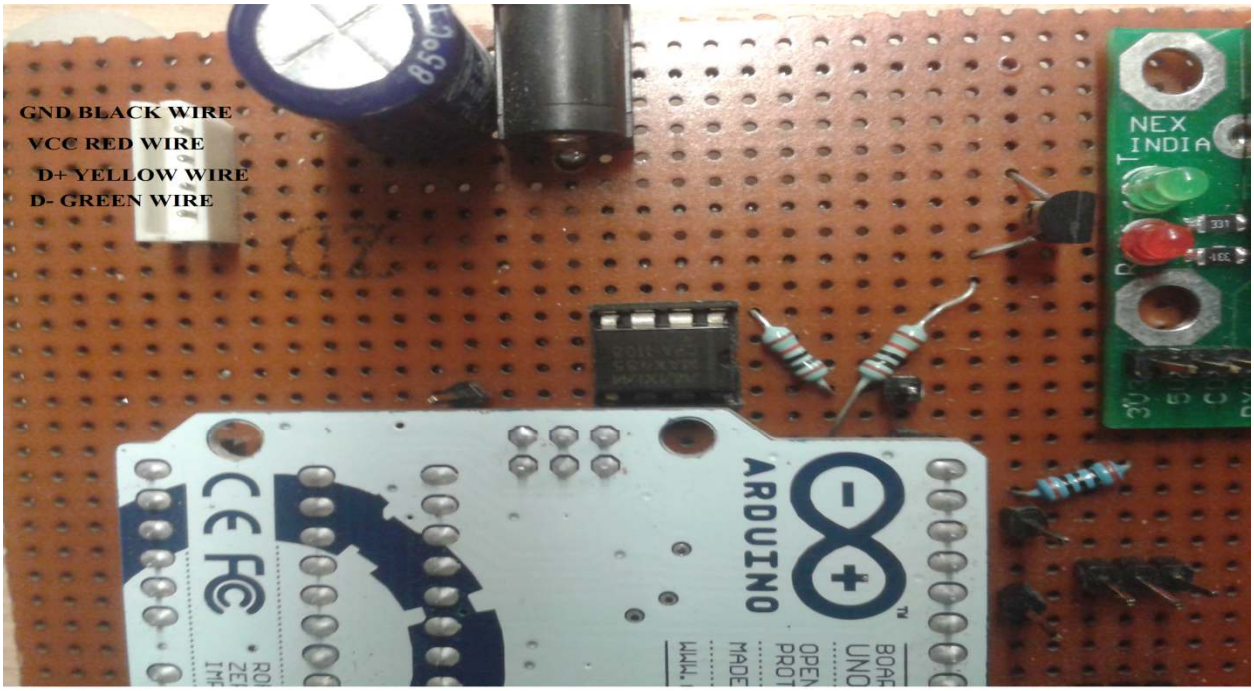


Figure 5

As can be seen in the above image, a white color header box is used for connection of motors. The corresponding color wire can be connected to the header box as mentioned in the above image.

The dynamixel motor connection and circuit diagram is critical and needs to be done very carefully. All the documents regarding the motors needs to be studied carefully which is provided in the references section.

First of all, the motors work on RS485 protocol. These motors support daisy chain method i.e. the connection of the second motor is done to the first motor. The connection in daisy chain fashion is as shown below:

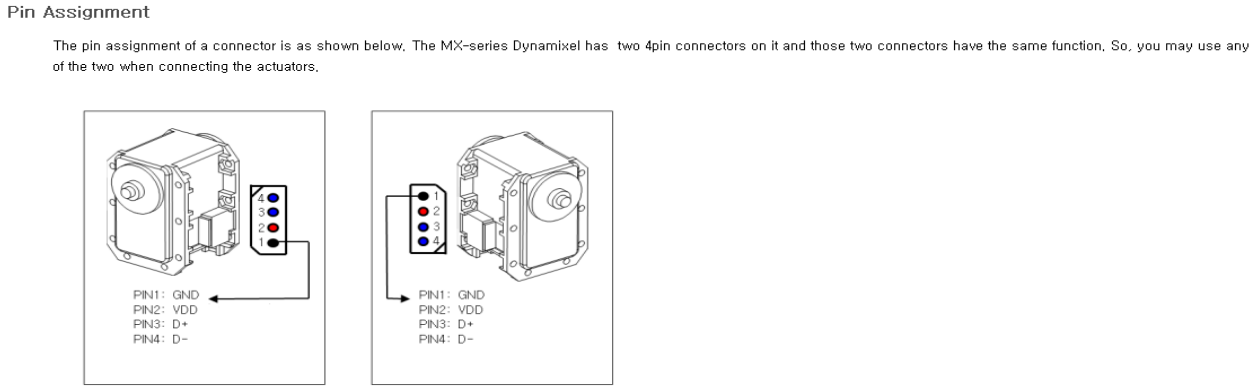


Figure 6

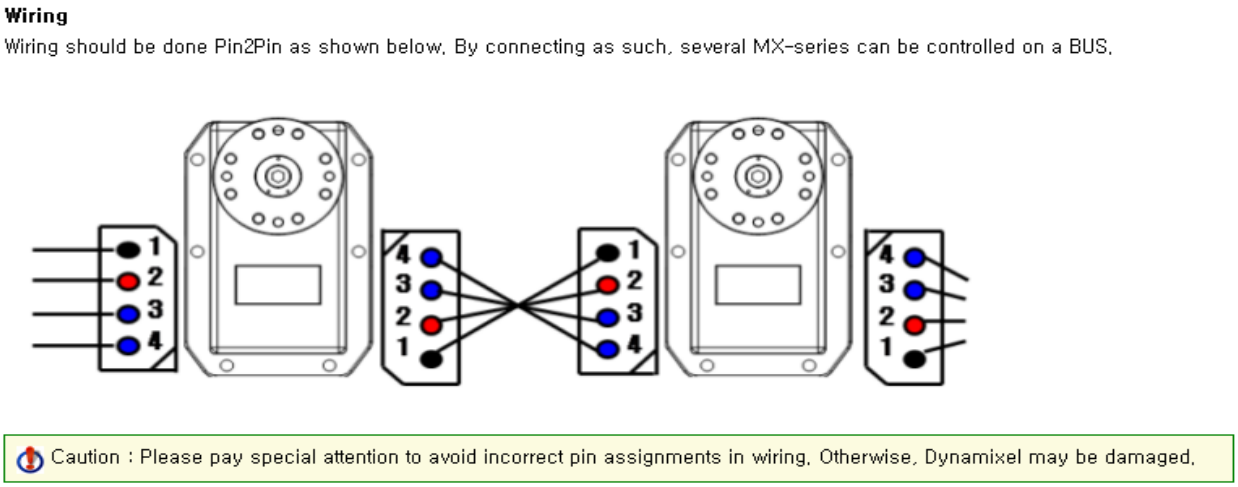


Figure 7

As can be seen in the figure 7, keep the motor shaft in front , the left header box of the ID1 motor (i.e. the shoulder joint motor) is directly connected to the general purpose board with the connections of wire shown in the figure 5. The right header box of ID1 motor is connected to the left header box of ID2 motor as shown in the figure 7. It requires 12 V power supply . The same can be found on the general purpose board. To check the connection is done properly, a led at the bottom of the motor will glow for 1 sec. If it does not glow immediately shut the power source and check for connections.

These motors follow a certain protocol to run. The instructions to these motors are sent in packets. This is further explained in this document.

Operating the Dynamixel motors:

These motors have IDs to identify which motor needs to be given instruction when in daisy chain fashion. The packets which are sent to the motors can be understood with the help of code and the image as follows.

Figure 8

The above Instruction packet can be found at http://support.robotis.com/en/

To support with the code:

```
void activateServos (byte servoID, byte posl, byte posh, byte movspdl, byte movspdh){
        digitalWrite(2,LOW);                    //explained in figure 10
        byte checksum_ACK;
        byte notchecksum;
        checksum_ACK =  servoID + 0x07 + 0x03 + 0x1E + posl + posh + movspdl + movspdh;
        notchecksum = ~checksum_ACK;
        delay(5);
        Serial.write(byte(0xFF));           //first start bit of the code
        Serial.write(byte(0xFF));           //second start bit of the code
        Serial.write(byte(servoID));        //to specify the servo ID either 0x01 or 0x02.
        Serial.write(byte(0x07));           //to specify the length of data to send. Length =No. of parameters +2
        Serial.write(byte(0x03));           //to specify that we want to write data to the motor.
        Serial.write(byte(0x1E));           // to specify the starting address of what type of data to be written. 0x1E specifies the starting
address of goal position
        Serial.write(byte(posl));           //lower byte of position
        Serial.write(byte(posh));           //higher byte of position
        Serial.write(byte(movspdl));        //lower byte of speed
        Serial.write(byte(movspdh));        //higher byte of speed
        Serial.write(byte(notchecksum));    //final checksum which is negation of (+ 0x07 + 0x03 + 0x1E + posl + posh + movspdl +
movspdh)
        delayMicroseconds(1500);
}
```

30 (0X1E)	Goal Position(L)	Lowest byte of Goal Position	RW	-
31 (0X1F)	Goal Position(H)	Highest byte of Goal Position	RW	-
32 (0X20)	Moving Speed(L)	Lowest byte of Moving Speed	RW	-
33 (0X21)	Moving Speed(H)	Highest byte of Moving Speed	RW	-

To call this function

activateServos(0x02, 0xFD, 0x07,0x40,0x00)
0x02 is the ID2 ;0xFD is the lower byte of goal position; 0x07 is the higher byte of goal position; 0x40 is the lower byte of desired speed; 0x00 is the higher byte of desired speed.

activateServos(0x01, 0x50, 0x0F,0x40,0x00);
0x01 is the ID1 ;0x50 is the lower byte of goal position; 0x0F is the higher byte of goal position; 0x40 is the lower byte of desired speed; 0x00 is the higher byte of desired speed.

Reading the values from motors: Firstly, we need to tell the motor that we want to read the values and what parameters are to be read so we write to the motors and after writing we receive the following format of the packet from the motor shown in figure 9.

Status Packet (Return Packet)

Dynamixel executes command received from the Main controller and returns the result to the Main Controller. The returned data is called Status Packet. The structure of Status Packet is as follows:

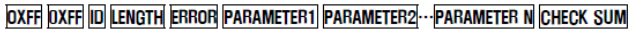


Figure 9

To support with the code:

```
void readServos (byte servoID, byte newValue){
digitalWrite(2,LOW);                                    //explained in figure 10

 byte checksum_ACK;
  byte notchecksum;
  startAddress = 0X1E;     // Turning on led

  checksum_ACK =  servoID + 0x04 + 0x02 + 0x24 + 0x04;
  notchecksum = ~checksum_ACK;
  delay(5);                // Allow this to take effect

  Serial.write(byte(0xFF));  // 1.These 2 bytes are 'start message'
  Serial.write(byte(0xFF));  // 2.These 2 bytes are 'start message'
  Serial.write(byte(servoID));  // 3.Address 1 is target servo or 0xfe which is broadcast mode
  Serial.write(byte(0x04));  // 4.Length of string .  length=No. of parameters + 2
  Serial.write(byte(0x02));  // 0x02 is to tell motors that we want to read
  Serial.write(byte(0x24)); // 0x24is the address of present position
  Serial.write(byte(0x04));// Number of parameters to be read

  Serial.write(byte(notchecksum)); //8. the notchecksum
  delayMicroseconds(1500);
   // allow last byte to go through
}

Storing the values in the buffer

digitalWrite(2,HIGH);                          //explained in figure 10

while(Serial.available()==0)
{}
j=0;
while(Serial.available())
  {
  Val[j]=Serial.read();
  Serial.print("    ");
  Serial.print(val,HEX);
  delay(1);
j++;
  }
```

Thus in val array all the returned packets are stored.

Circuit diagram for direction control of the motor:

Connection to UART

To control MX Series with a personally made Main Controller, the signal of Main Controller UART should be converted into RS485 type signal. The following is a recommended circuit diagram.

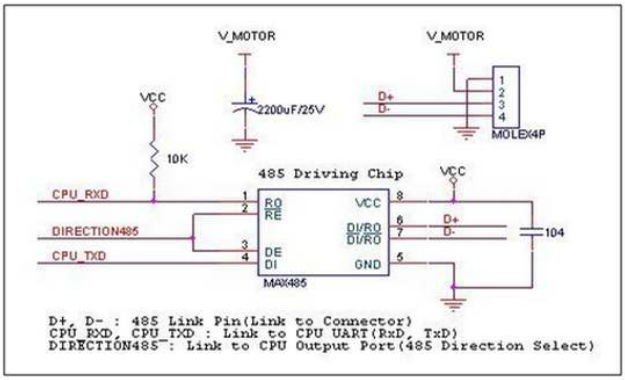


Figure 10

Pin 0 of arduino is connected to PIN 1 of MAX485.
Pin 1 of arduino is connected to PIN 4 of MAX485.
Pin 2 of arduino is connected to PIN 2 and 3 of MAX485. This is used for direction control. When we want to write data to the motor we send HIGH signal and when we want to read data from motor we send LOW signal but because this signal needs to 5V and arduino is not capable of doing that we have used BJT as a switch and thus the signals are inverted in the above codes.
PIN 6 and 7 are the output given to the motor.
Pin 5 and 8 are Gnd and Vcc lines.

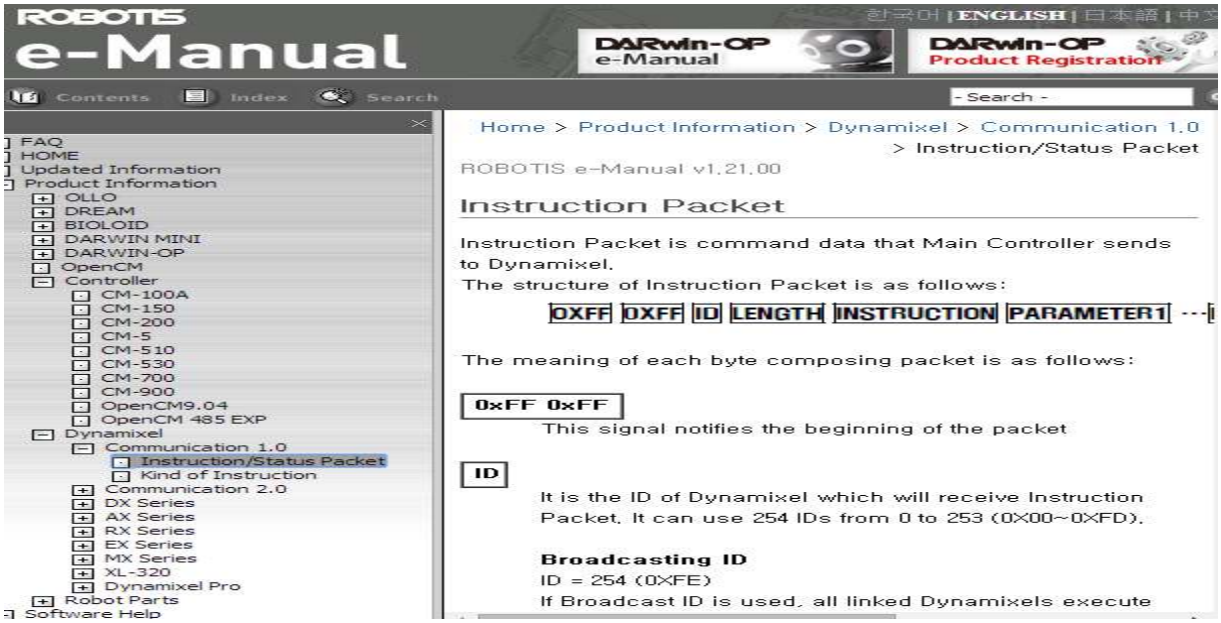
Refrences:

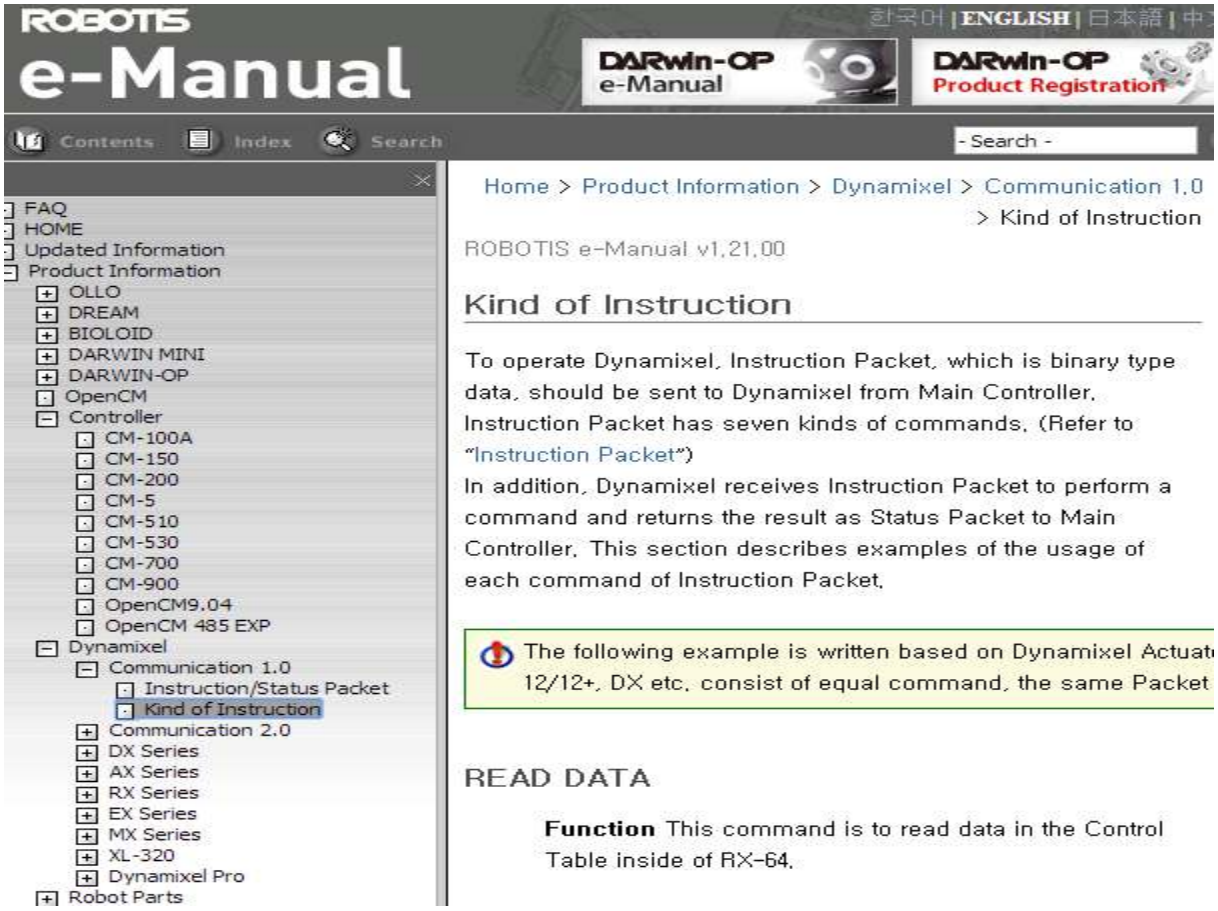
Arduino download: http://arduino.cc/en/main/software

USB to serial driver download: http://www.nex-robotics.com/index.php?page=shop.product_details&flypage=flypage.tpl&product_id=267&category_id=59&option=com_virtuemart&Itemid=45

Dynamixel motor: Read all the pages then only start work on this.

http://support.robotis.com/en/





ROBOTIS e-Manual

한국어 | ENGLISH | 日本語 | 中文

DARwin-OP e-Manual | DARwin-OP Product Registration

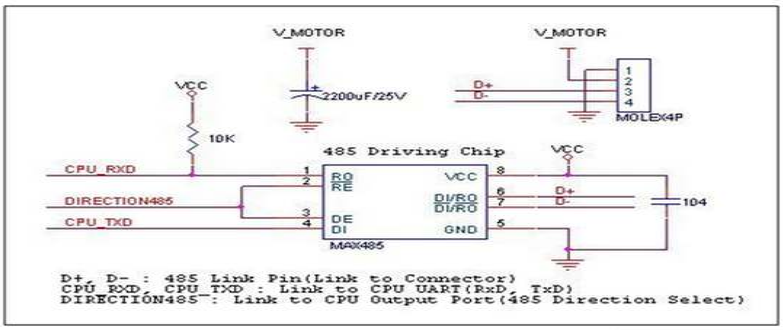
Contents | Index | Search | - Search - | Go

- FAQ
- HOME
- Updated Information
- Product Information
  - OLLO
  - DREAM
  - BIOLOID
  - DARWIN MINI
  - DARWIN-OP
  - OpenCM
  - Controller
    - CM-100A
    - CM-150
    - CM-200
    - CM-5
    - CM-510
    - CM-530
    - CM-700
    - CM-900
    - OpenCM9.04
    - OpenCM 485 EXP
  - Dynamixel
    - Communication 1.0
      - Instruction/Status Packet
      - Kind of Instruction
    - Communication 2.0
    - DX Series
    - AX Series
    - RX Series
    - EX Series
    - MX Series
    - XL-320
    - Dynamixel Pro
  - Robot Parts
- Software Help

MX-28R, MX-64R, MX-106R

Connection to UART

To control MX Series with a personally made Main Controller, the signal of Main Controller UART should be converted into RS485 type signal. The following is a recommended circuit diagram.



ROBOTIS e-Manual

한국어 | ENGLISH | 日本語 | 中文

DARwin-OP e-Manual | DARwin-OP Product Registration

Contents | Index | Search | - Search - | Go

- Product Information
  - OLLO
  - DREAM
  - BIOLOID
  - DARWIN MINI
  - DARWIN-OP
  - OpenCM
  - Controller
    - CM-100A
    - CM-150
    - CM-200
    - CM-5
    - CM-510
    - CM-530
    - CM-700
    - CM-900
    - OpenCM9.04
    - OpenCM 485 EXP
  - Dynamixel
    - Communication 1.0
      - Instruction/Status Packet
      - Kind of Instruction
    - Communication 2.0
    - DX Series
    - AX Series
    - RX Series
    - EX Series
    - MX Series
      - MX-12W
      - MX-28
      - MX-64
      - MX-106
    - XL-320
    - Dynamixel Pro
  - Robot Parts
- Software Help

Parts Photo

[MX-106T] [MX-106R]

※ Control Table's Compliance replaced by PID.

※ The control table's order for PID has changed to DIP from this version onwards. Please make reference of this change.

※ Although the MX-106T (TTL) and MX-106R (RS-485) differ in communications protocols both have the same features and perform equally. (TTL uses 3-pin connectors while RS-485 uses 4)