

Robot Motion Control

(aka Embedded Systems Design and Implementation)

<http://ssl.kaist.ac.kr/class/18fall/socrobotdesign>

Hoi-Jun Yoo
KAIST

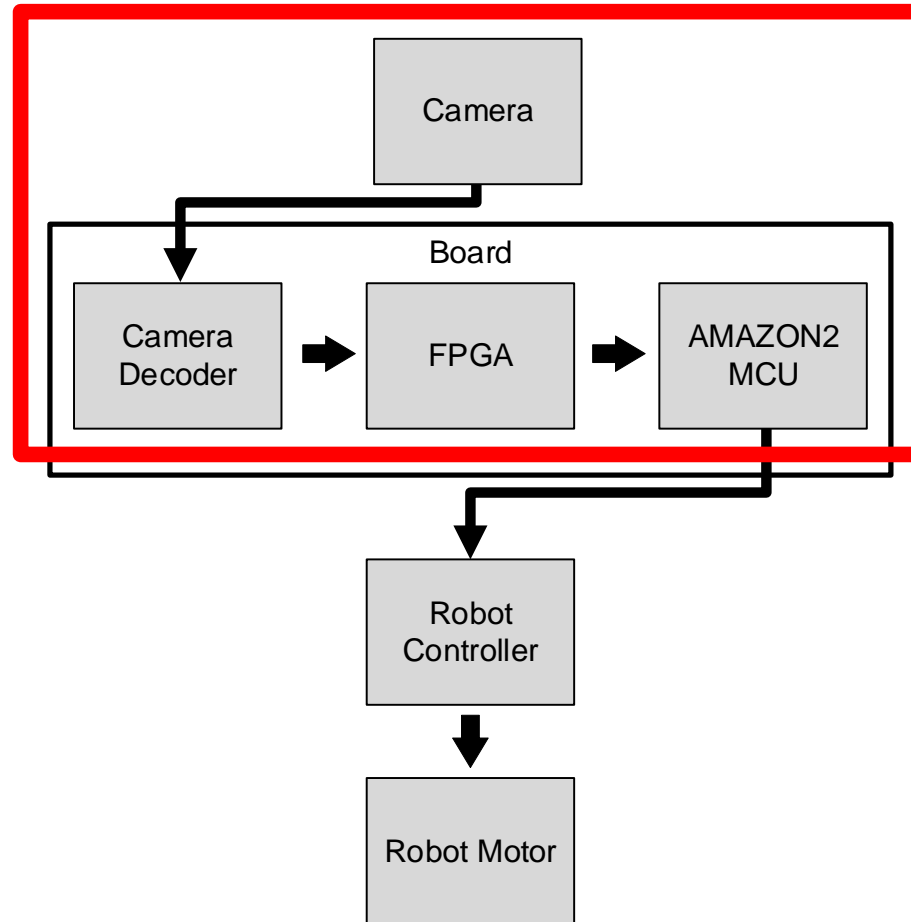


Introduction



- There are 18 motors in our robots.
- We can control each motor to have desired angle and speed.
- You can control robot motion by controlling these motors.

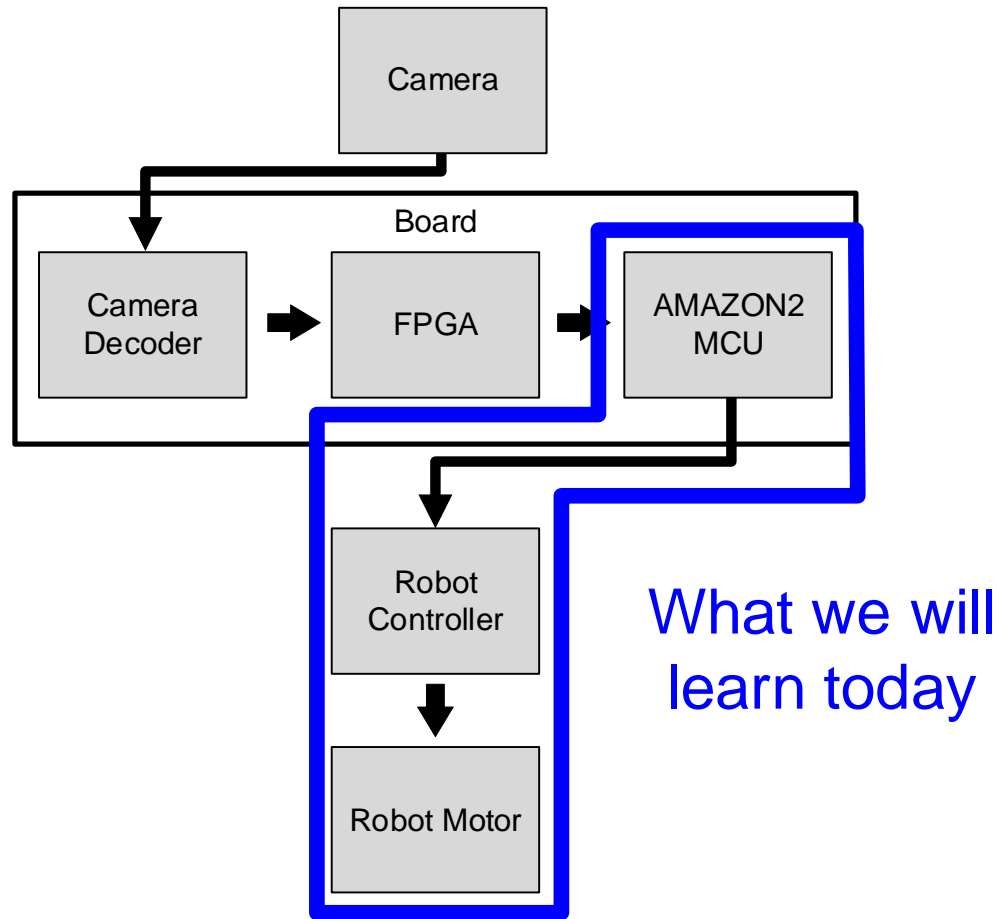
Overall Flow



What we
have learned

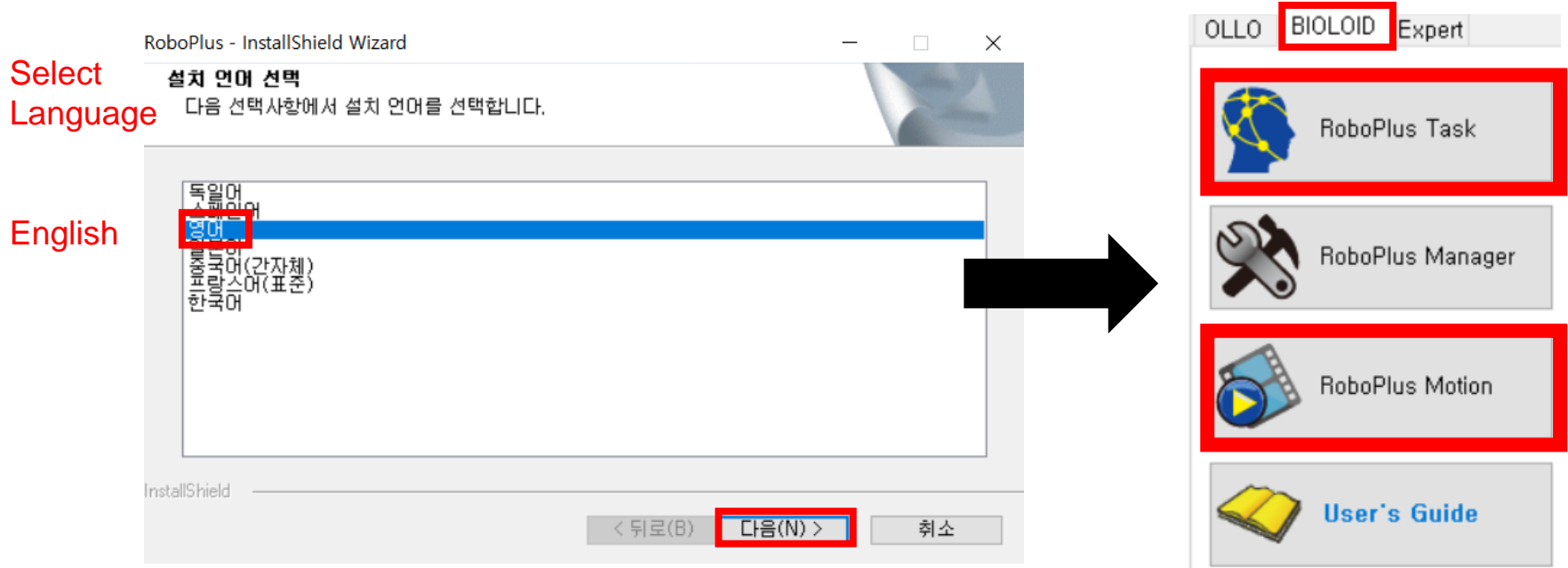
- So far, we have learned how to do image processing with FPGA and AMAZON2 MCU

Overall Flow



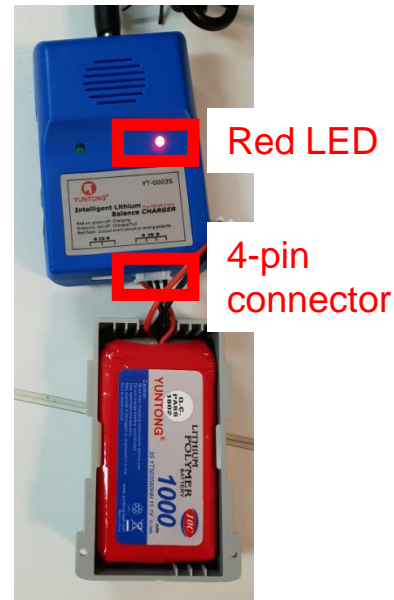
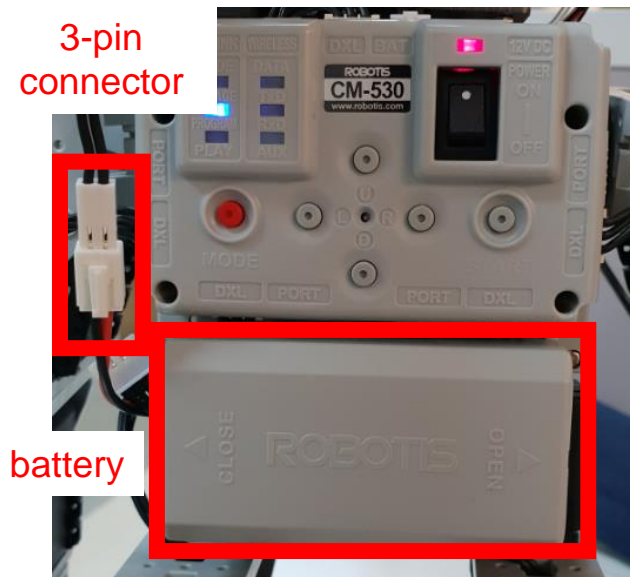
- This week we will learn how to control the **robot motor** with the **robot controller**, then learn how to control **robot controller** using **AMAZON2 MCU**.

Install Software



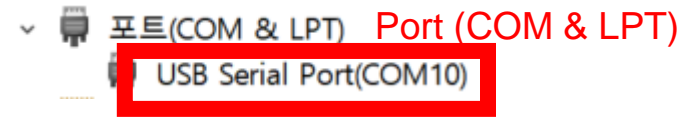
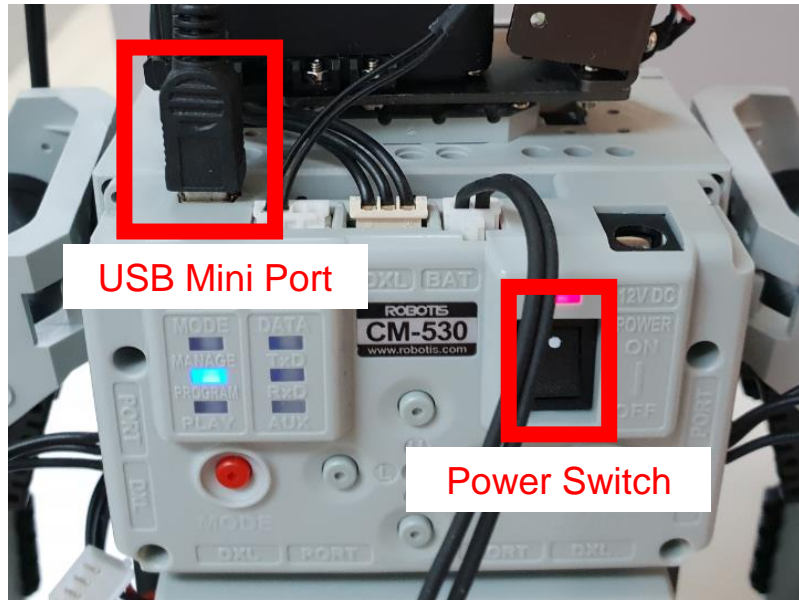
- Execute *RoboPlusWeb(v1.1.3.0).exe*
- Select **English (영어)**
- **After Installation**
 - Click **Bioloid** Tab
 - We will use **RoboPlus Task** and **RoboPlus Motion** program
 - **RoboPlus Task**: Robot task editor.
 - **RoboPlus Motion**: Robot motion editor. You can define simple robot motion.

Charge Robot Battery



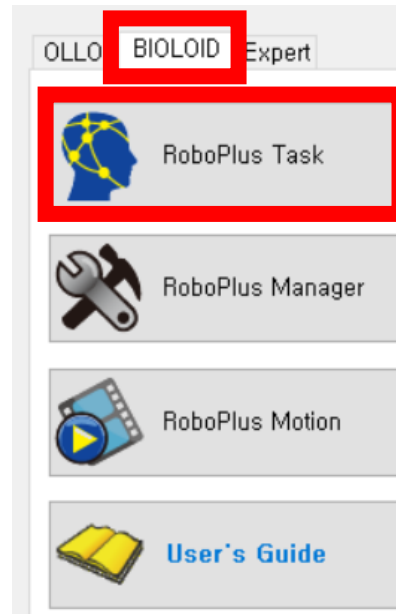
- Disconnect the 3-pin port which the robot and the battery are connected.
- Slide the battery case to the right to release the battery case.
- Connect the battery charger and the battery case via 4-pin port. (Be careful not to charge via the 3-pin port)
- The red LED of battery charger turned on and charging starts. When charging is completed, the green LED is on.
- Mount the battery case on the robot and connect a battery with a robot via the 3-pin port.

Connecting Robot-PC



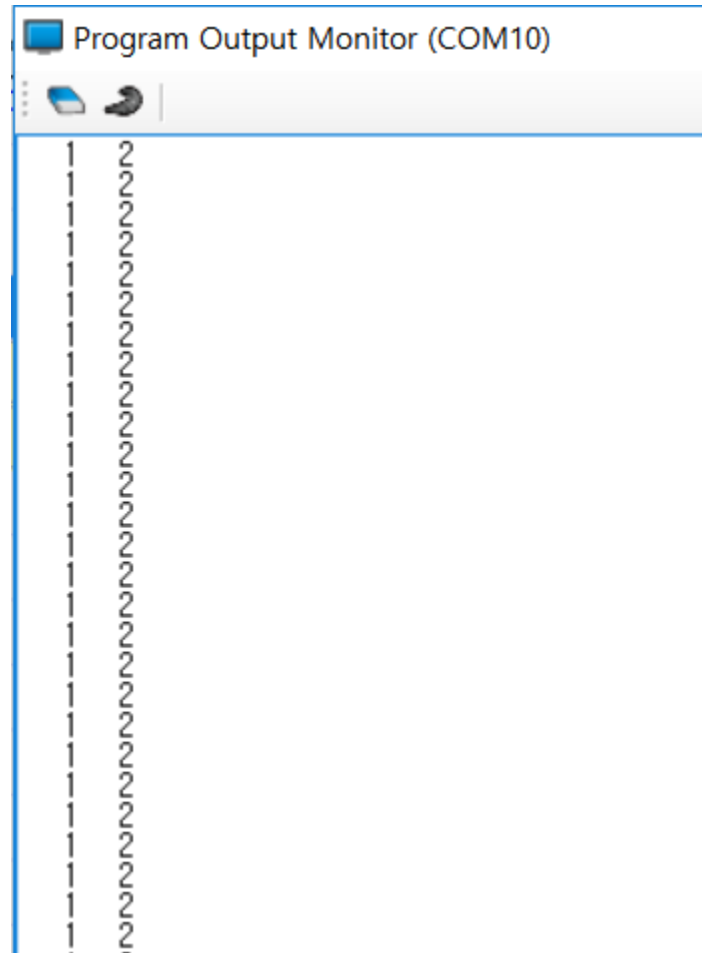
- First turn on the CM-530 controller.
- As shown in the figure above, there is a USB Mini type port on the top of the CM-530 controller. Insert the cable to this port. Then connect the cable to your PC via USB.
- If you are able to see “**USB Serial Port(COMxx)**” in the Device Manager, the robot and the PC are successfully connected.

Connecting Robot-PC



- Execute **RoboPlus**
- Execute **RoboPlus Task**
- 'Task code' refers to the source code that specifies tasks to be executed by the robot. The robot moves according to your 'task codes'. You can easily write 'task code' with **RoboPlus Task** software.
- The **RoboPlus Task** program is very similar to C. If you are familiar with C, you can write 'task code' very easily.

RoboPlus Task Tutorial



- **Objective:** print 1 and 2 on the output screen as above.

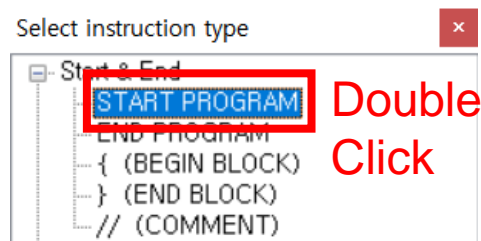
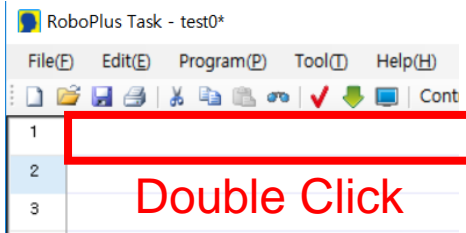
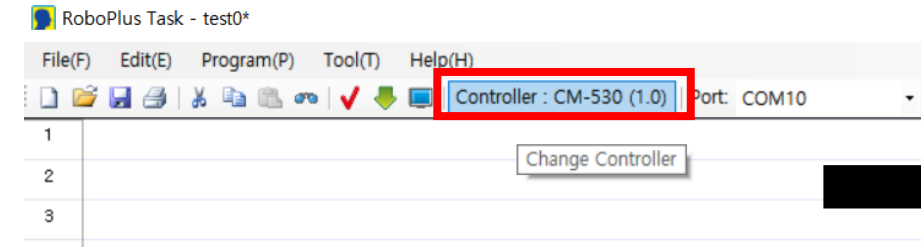
RoboPlus Task Tutorial

Please select the appropriate controller

Firmware Version
☒ 1.0 ☐ 2.0

Controller	Product
CM-100	OLLO
CM-5	BIOLOID
CM-510	BIOLOID Premium/GP
CM-530	BIOLOID Premium/GP/S..
CM-700	Platform

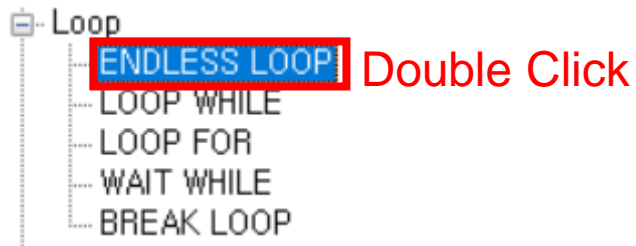
OK



1	START PROGRAM
2	{
3	
4	}

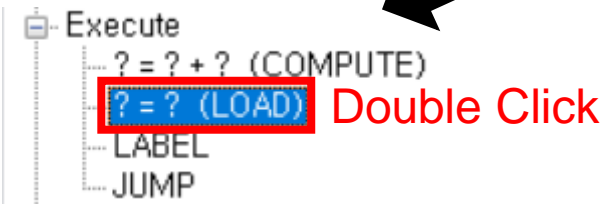
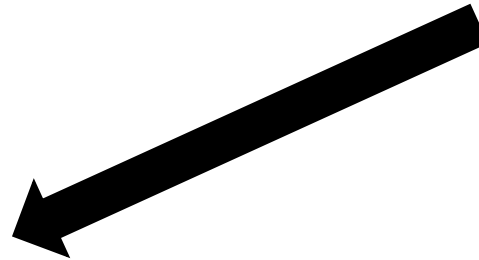
- Click **Controller** → Select **Firmware Version 1.0** → Select **CM-530** → Click **OK**
- Double click an empty line or press 'Enter'.
- Select **START PROGRAM** and double click it.

RoboPlus Task Tutorial



1	START PROGRAM
2	{
3	ENDLESS LOOP
4	{
5	
6	}
7	}

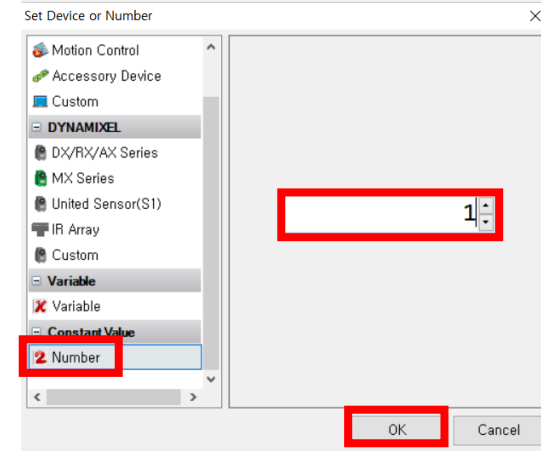
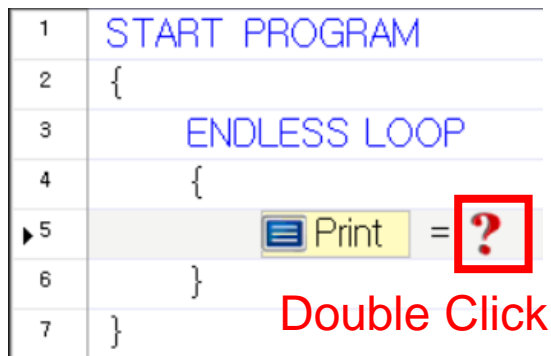
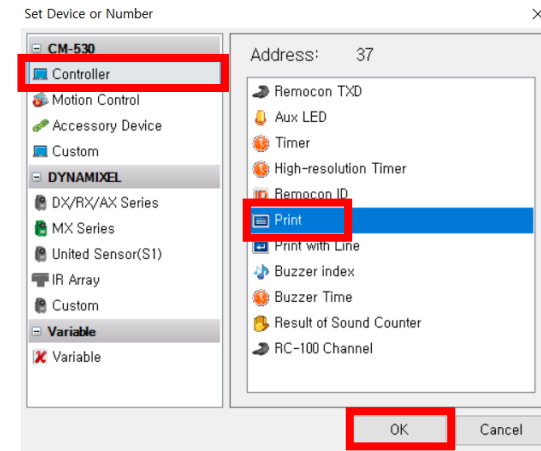
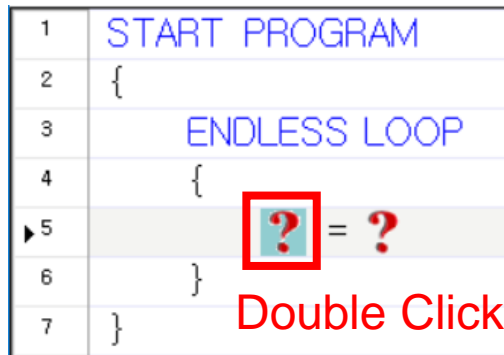
Double Click



1	START PROGRAM
2	{
3	ENDLESS LOOP
4	{
5	? = ?
6	}
7	}



- Select **ENDLESS LOOP** and double click it.
- Select **LOAD** and double click it.

RoboPlus Task Tutorial





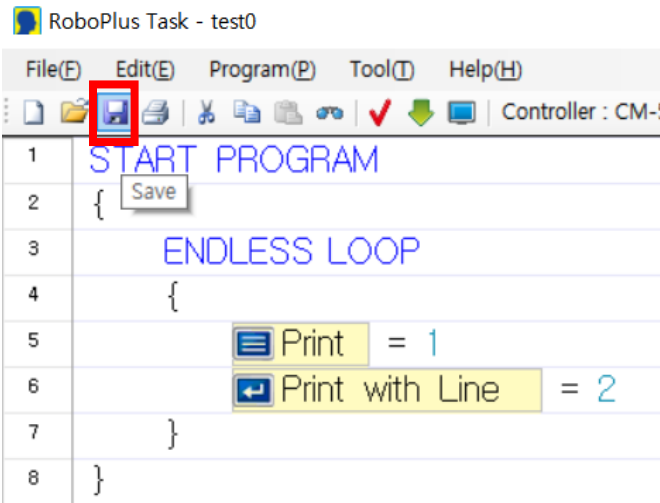
- Double click the **left parameter (?)**
- Click **Controller** → Click **Print** → Click **OK**
- Double click the **right parameter(?)**
- Click **Number**→ set the value to **1** → Click **OK**

RoboPlus Task Tutorial

1	START PROGRAM
2	{
3	ENDLESS LOOP
4	{
5	 Print = 1
6	 = ?
7	}
8	}

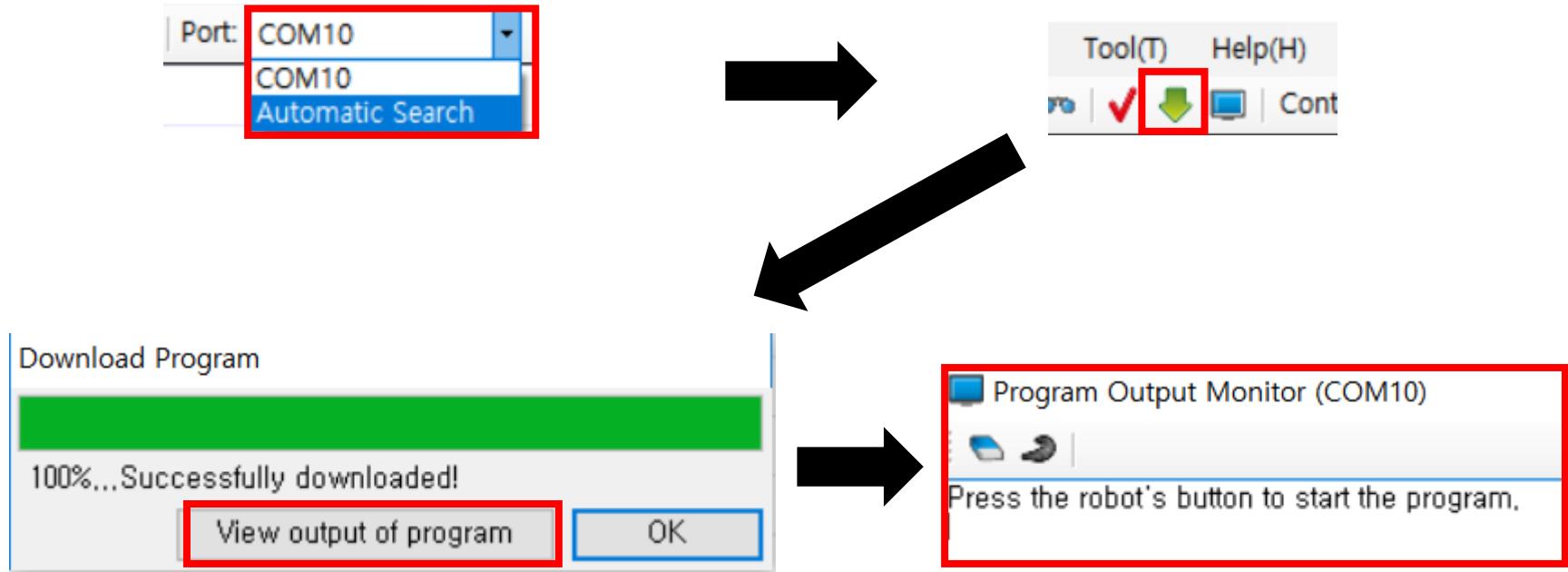
Double Click

1	START PROGRAM
2	{
3	ENDLESS LOOP
4	{
5	 Print = 1
6	 Print with Line = 2
7	}
8	}



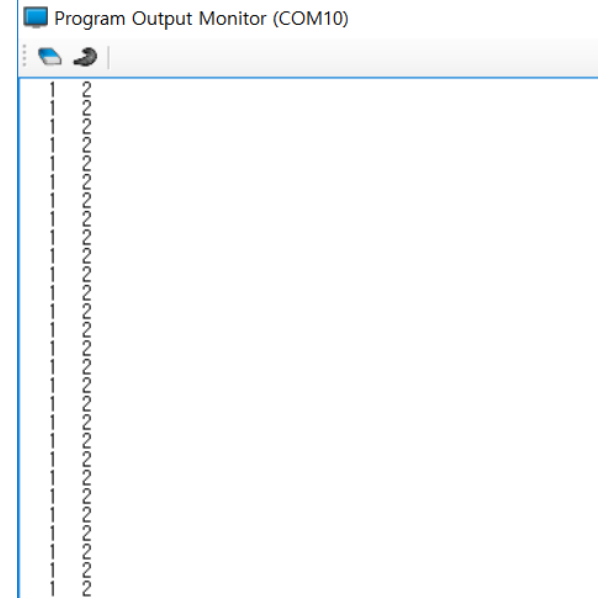
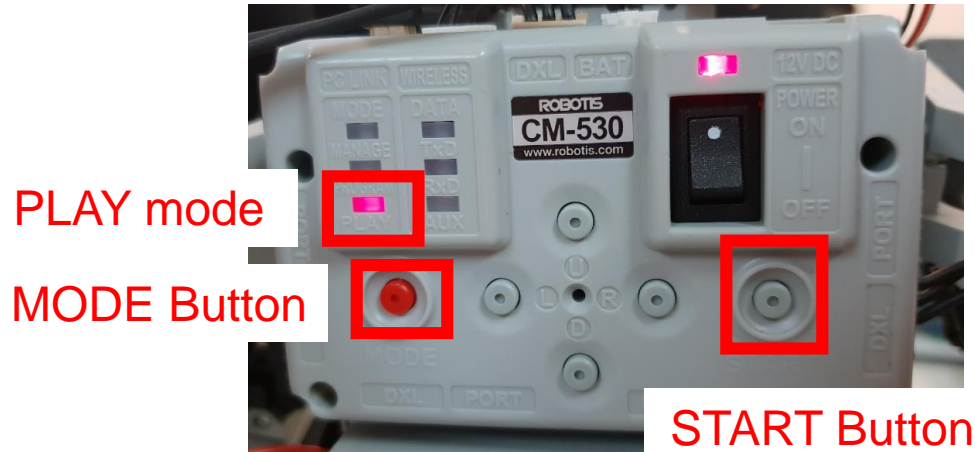
- Select } under **Print** command. Add new lines by pressing 'spacebar'. Repeat previous steps (**Print** is changed to **Print Line**)
- Press Ctrl+S or save icon to save the file

RoboPlus Task Tutorial



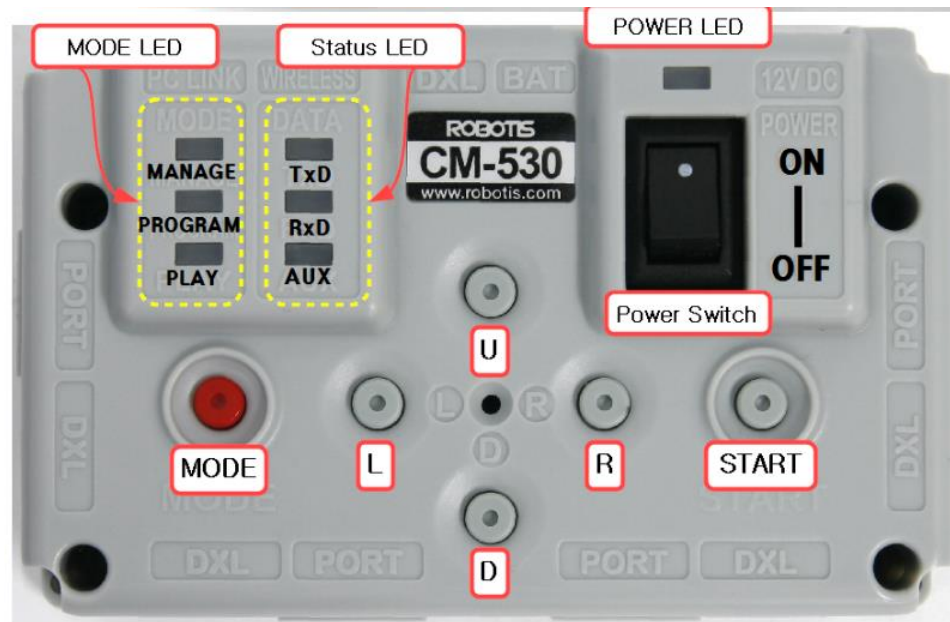
- Now, we need to load the 'task code' to the robot. Select the correct COM port or use '**Automatic Search**' function.
- Download the program by clicking the green arrow.
- After downloading, click on the **View Print of Program** in the Download Program window.

RoboPlus Task Tutorial



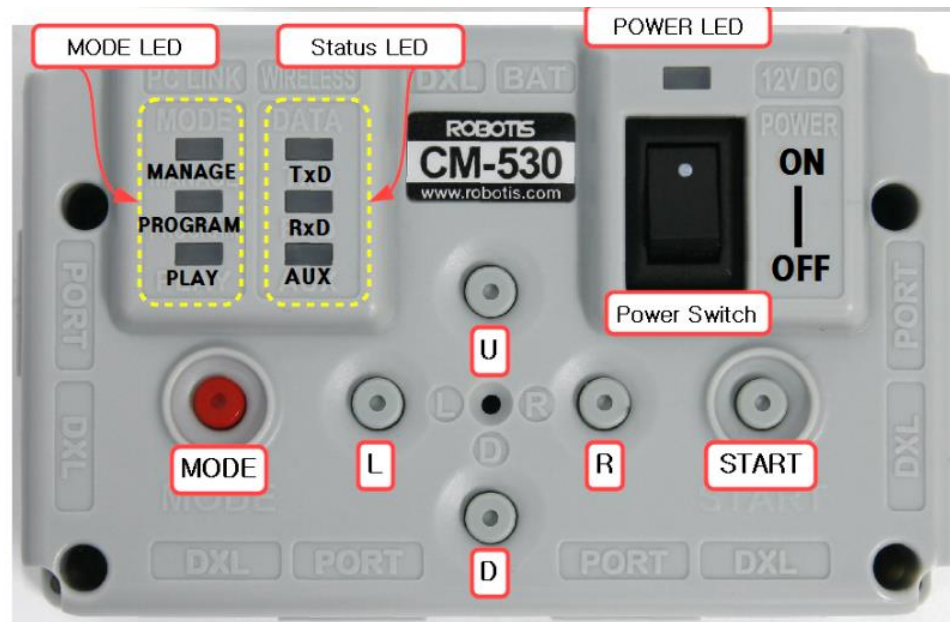
- Press **MODE button** on the CM-530 controller to change the mode of the controller.
- Press **MODE button** to change the robot to **PLAY mode**.
- Press **START button** to start the program. You can see '1' and '2' being printed on the **Program Output Monitor**.

Controller



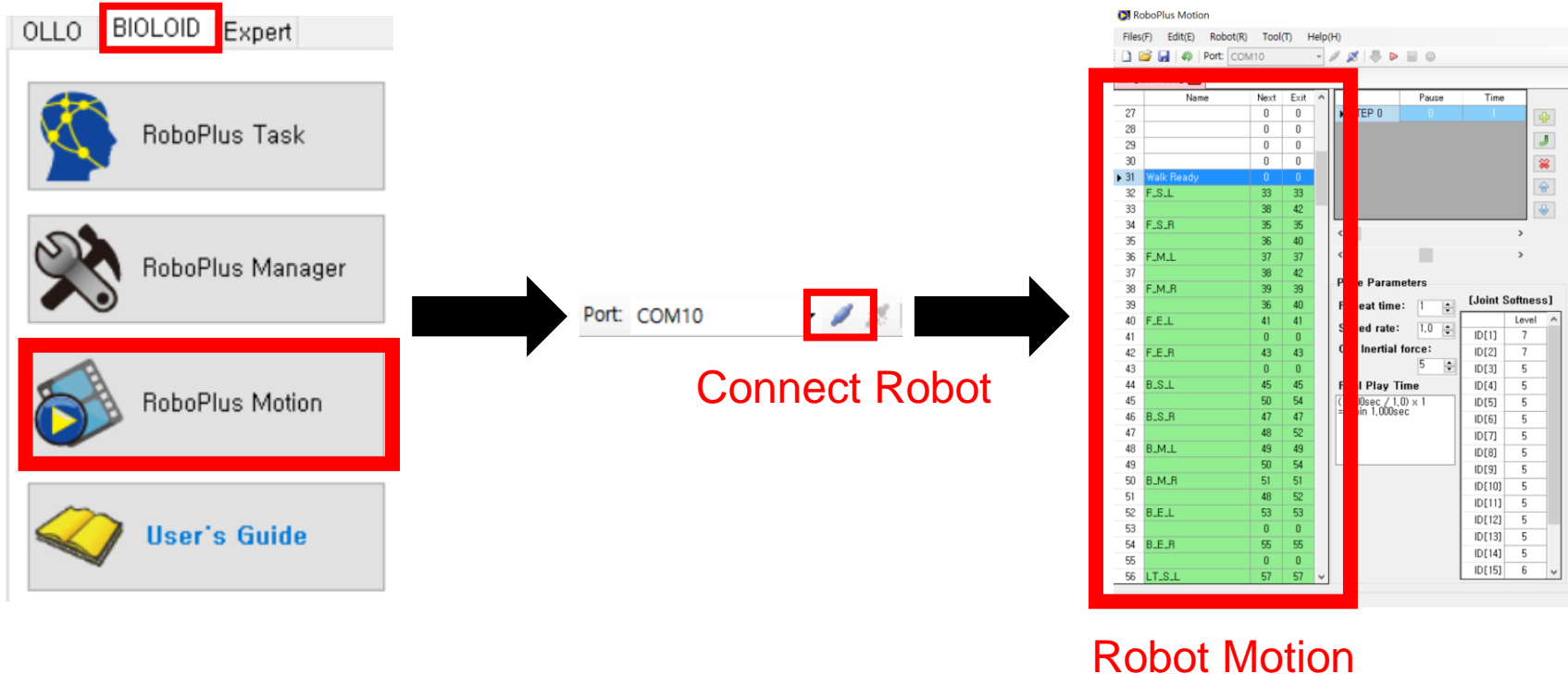
- **Power LED** : ON and OFF LED status for the power
- **Power Switch** : Used to turn the robot ON / OFF.
- **MODE Button** : Used to change the operation mode of CM-530.
- **MODE LED** : LED that displays current operation mode of CM-530;
 - **PROGRAM**
 - Motion edit mode is in progress.
 - **PLAY**
 - Task code mode is in progress.
 - The Start button must be pressed by the user to start robot motion when PLAY LED flickers.

Controller



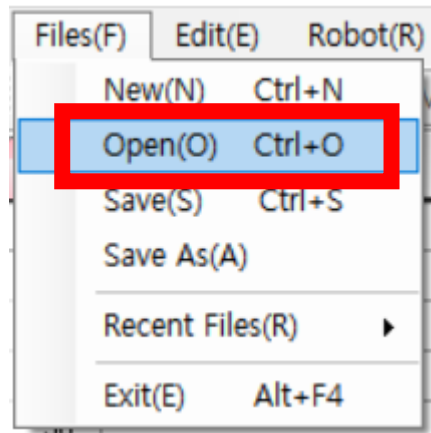
- **Status Display LED** : The LED represents the current status of CM-530.
 - **TxD** : Turned on while CM-530 is transmitting the data to the outside.
 - **RxD** : Turned on while CM-530 is receiving the data from the outside.
 - **AUX** : Assigned LED to be used by the user in the program. It can be turned on or off using 'task code'.
- **START Button** : Used to START selected mode.
- **U / L / D / R Button** : These buttons can be used to send commands to the robot.
- **Mode Display LED** : LED to display current operation mode of CM-530

Robot Motion



- You can define robot motion using **RoboPlus Motion**.
- “Robot Motion” refers to the motion data in the controller.
- Open **RoboPlus Motion** program.
- Click **Connect Robot** icon
- After a while, the motions stored in the robot controller will appear.
- If error occurs, close and restart **RoboPlus Motion**

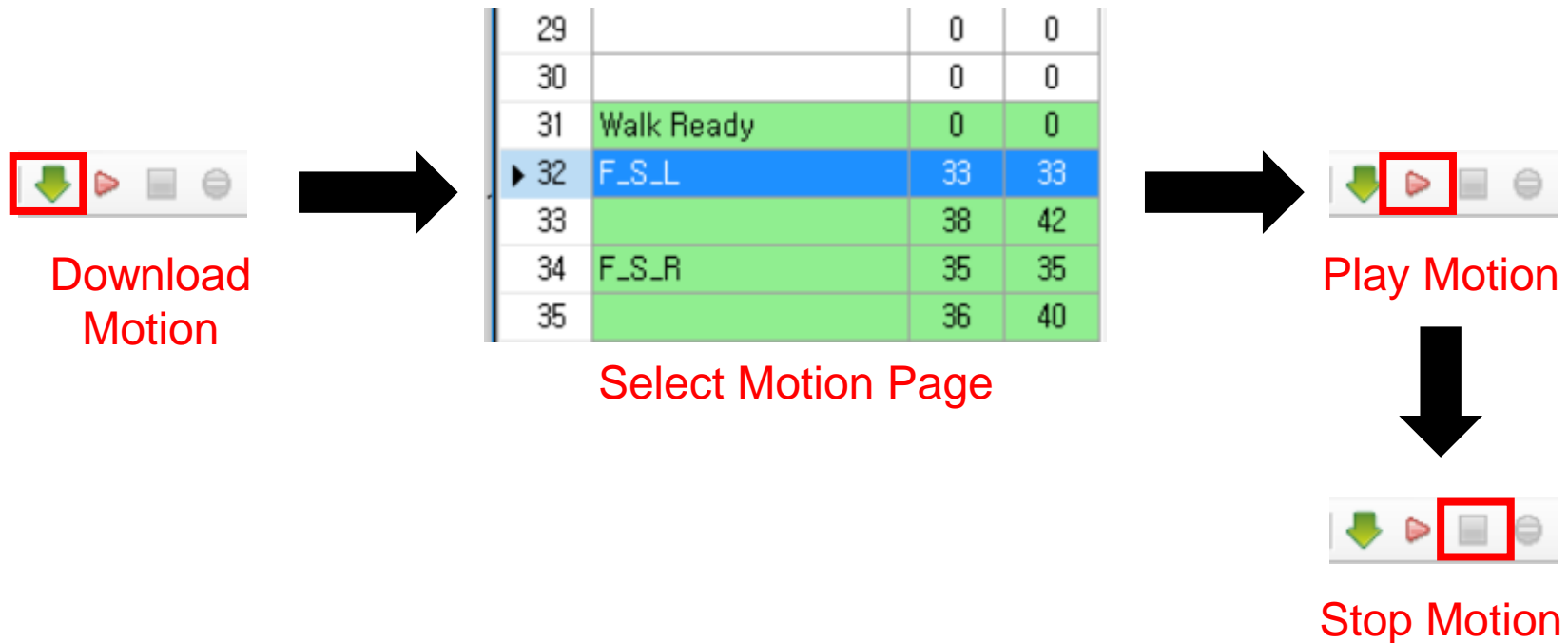
File Motion



	Name	Next	Exit
18		0	0
19		0	0
20		0	0
21		0	0
22		0	0
23		0	0
24		0	0
25		0	0
26		0	0
27		0	0
28		0	0
29		0	0
30		0	0
31	Walk Ready	0	0
32	F_S_L	33	33
33		38	42
34	F_S_R	35	35
35		36	40

- Open *Walking_Robot.mtn*. This is called “File Motion”, and it refers to the motion data in the form of files in the PC.
- The file contains combination of a walking motion file made up of specific patterns.

Download Motion



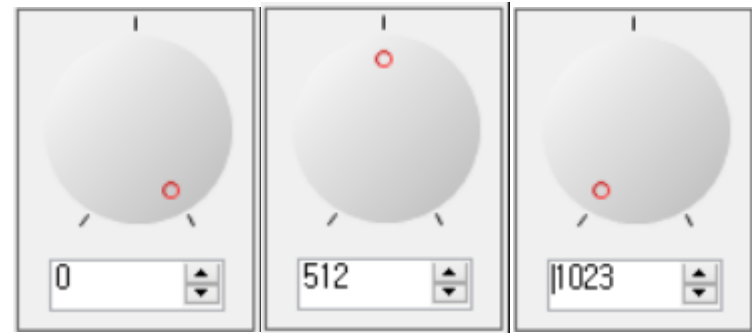
- “File Motions” can be converted into “Robot Motions”.
- Click on the **Download Motion** menu and wait for the download to be completed.
- You can play the created motions. Search the page to play and click **Play Motion**.
- To stop the motion that is being carried out, click **Stop Motion**

Motion Editing

Basic Pose Editor Pose Utility Edit All page

<Pose of Step>			<Pose of Robot>	
	Value		ID	Value
ID[1]	235	<input checked="" type="checkbox"/>	ID[2]	788
ID[2]	788	<input checked="" type="checkbox"/>	ID[3]	279
ID[3]	279	<input checked="" type="checkbox"/>	ID[4]	744
ID[4]	744	<input checked="" type="checkbox"/>	ID[5]	462
ID[5]	462	<input checked="" type="checkbox"/>	ID[6]	561
ID[6]	561	<input checked="" type="checkbox"/>	ID[7]	358
ID[7]	358	<input checked="" type="checkbox"/>	ID[8]	666
ID[8]	666	<input checked="" type="checkbox"/>	ID[9]	507
ID[9]	513	<input checked="" type="checkbox"/>	ID[10]	516
ID[10]	522	<input checked="" type="checkbox"/>	ID[11]	341
ID[11]	342	<input checked="" type="checkbox"/>	ID[12]	682
ID[12]	681	<input checked="" type="checkbox"/>	ID[13]	240
ID[13]	241	<input checked="" type="checkbox"/>	ID[14]	783
ID[14]	781	<input checked="" type="checkbox"/>	ID[15]	647
ID[15]	646	<input checked="" type="checkbox"/>	ID[16]	376
ID[16]	377	<input checked="" type="checkbox"/>	ID[17]	507
ID[17]	513	<input checked="" type="checkbox"/>	ID[18]	516
ID[18]	522	<input checked="" type="checkbox"/>		

Navigation controls: Right arrow, Left arrow, and a circular dial with a red dot and a numeric input field (0).



- Each motion is combination of many poses.
- A pose is the robot's position at a point in time. It is a collection of motor position values required for the posture.
- **<Pose of Step>** refers to the direction of the pose. This data is stored in the PC or robot controller.
- **<Pose of Robot>** refers to the current direction of the connected robot motors.
- Value of motor indicates direction of the motor.

Motion Editing

Basic Pose Editor			Pose Utility	Edit All page
<Pose of Step>			<Pose of Robot>	
	Value			Value
ID[1]	235	<input checked="" type="checkbox"/>	ID[2]	788
ID[2]	788	<input checked="" type="checkbox"/>	ID[3]	279
ID[3]	279	<input checked="" type="checkbox"/>	ID[4]	744
ID[4]	744	<input checked="" type="checkbox"/>	ID[5]	462
ID[5]	462	<input checked="" type="checkbox"/>	ID[6]	561
ID[6]	561	<input checked="" type="checkbox"/>	ID[7]	358
ID[7]	358	<input checked="" type="checkbox"/>	ID[8]	666
ID[8]	666	<input checked="" type="checkbox"/>	ID[9]	507
ID[9]	513	<input checked="" type="checkbox"/>	ID[10]	516
ID[10]	522	<input checked="" type="checkbox"/>	ID[11]	341
ID[11]	342	<input checked="" type="checkbox"/>	ID[12]	682
ID[12]	681	<input checked="" type="checkbox"/>	ID[13]	240
ID[13]	241	<input checked="" type="checkbox"/>	ID[14]	783
ID[14]	781	<input checked="" type="checkbox"/>	ID[15]	647
ID[15]	646	<input checked="" type="checkbox"/>	ID[16]	376
ID[16]	377	<input checked="" type="checkbox"/>	ID[17]	507
ID[17]	513	<input checked="" type="checkbox"/>	ID[18]	516
ID[18]	522	<input checked="" type="checkbox"/>		

Control
motion








Capture
motion



Torque on Torque off

- You can control robot motion or capture robot motion.
- First, write the direction of each motor in the **<Pose of Step>** then, click the right arrow. The robot moves according to the input motor direction.
- Second, click dark bulb then move the robot motor by hand. Then press the left arrow. Then you can see the current direction of each motor.
- Clicking the light bulb turns on the torque, and clicking the dark bulb turns off the torque. To manipulate the robot by hand, you must turn off the torque by clicking the dark bulb.

Motion Editing

	Pause	Time		
► STEP 0	0	0,08		Add step
STEP 1	0	0,08		Insert step
STEP 2	0	0,08		Stop step
STEP 3	0	0,08		
STEP 4	0	0,08		
STEP 5	0	0,08		
STEP 6	0	0,08		

- Motion is composed of bunch of steps of poses.
- The speed of a motion is determined by the time of each step. (If time of each step is short, motor will move fast)
- You can set the time from the start of the current step to the end of the current step.
- You can pause between the end of the current step and the start of the next step.

Motion Editing

Each column
is a page

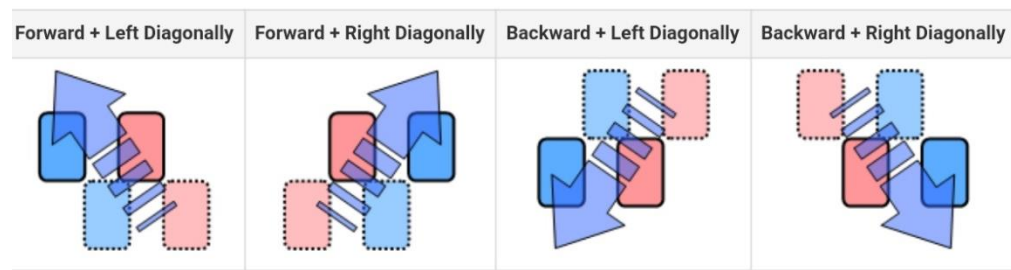
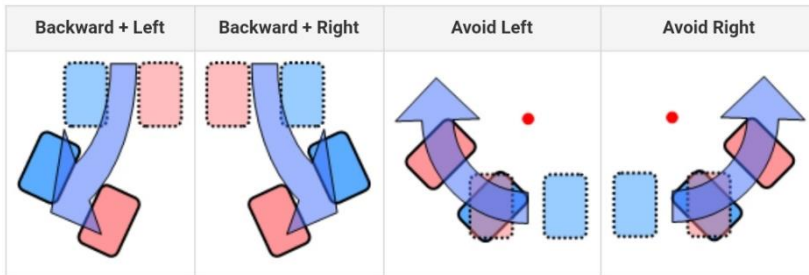
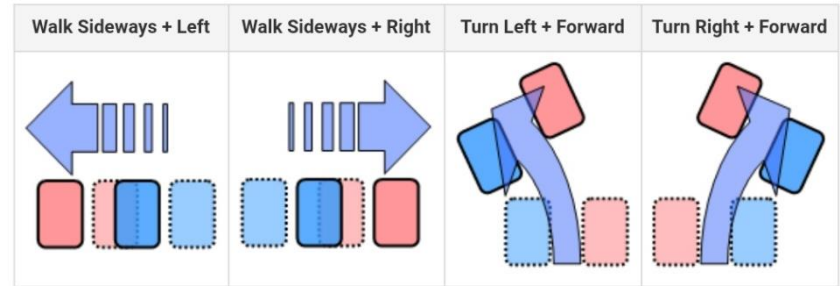
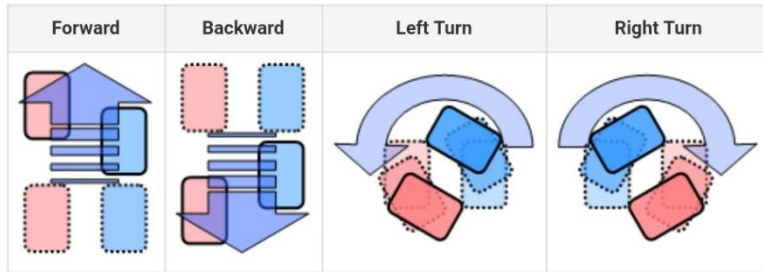
	Name	Next	Exit
31	Walk Ready	0	0
▶ 32	F_S_L	33	33
33		38	42
34	F_S_R	35	35
35		36	40
36	F_M_L	37	37
37		38	42
38	F_M_R	39	39
39		36	40
40	F_E_L	41	41

Next page

Exit page

- “Motion page” is the unit used to distinguish between saved motions.
- Imported motions are read in terms of pages.
Motion data consists of 255 pages.
- A single page can have a maximum of 7 steps. Therefore, some motions may not fit in one page.
- You need to use multiple pages for motion whose step size bigger than 7.
- When commands are made to stop a motion, the robot will usually be in a highly unstable state due to the motion being executed. To stop a motion in a stable state, designate an exit page.

Walk Execution



- You can control robot with predefined motion.
- Template task code is already given, so you can easily control the robot with coding.
- Template task code defined 16 basic movements.
- The figure above explains how the robot will move under certain commands.

Walk Execution

Start walking forward

Wait for 4 seconds

Start walking backward

Wait for 4 seconds

Stop moving

```
START PROGRAM
{
  CALL InitializationWalk
  WalkCommand = 1
  CALL WalkExecute
  Timer = 4.096sec
  WAIT WHILE ( Timer > 0 )
  WalkCommand = 2
  CALL WalkExecute
  Timer = 4.096sec
  WAIT WHILE ( Timer > 0 )
  WalkCommand = 0
  CALL WalkExecute
}
```

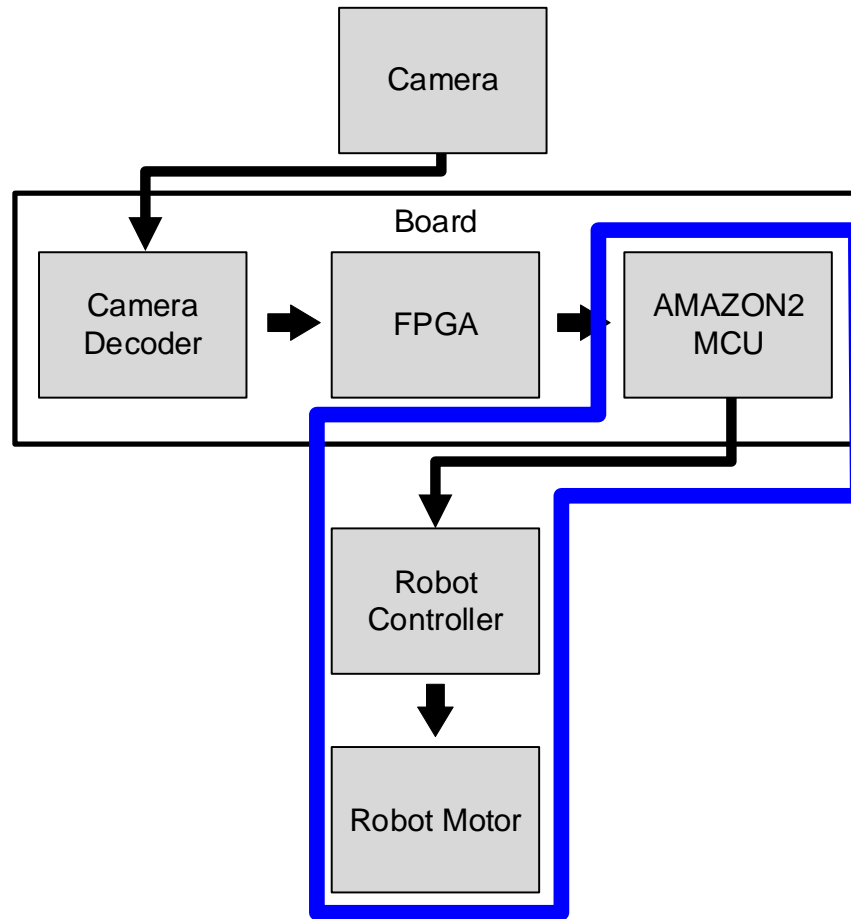
- Open *Walking_Robot.tsk* file using **RoboPlus Task**
- This code let the robot to move forward and backward.
- You can use timer variable to control time.
- Robot move forward (**WalkCommand 1**) for 4 seconds, then move backwards (**WalkCommand 2**) for 4 seconds, then stops movement (**WalkCommand 0**)

WalkCommand

WalkCommand	Action	WalkCommand	Action
0	Stop	9	2 + 5
1	Forward	10	2 + 6
2	Backward	11	Avoid Left
3	Left Turn	12	Avoid Right
4	Right Turn	13	1 + 3
5	Left Side	14	1 + 4
6	Right Side	15	2 + 3
7	1 + 5	16	2 + 4
8	1 + 6		

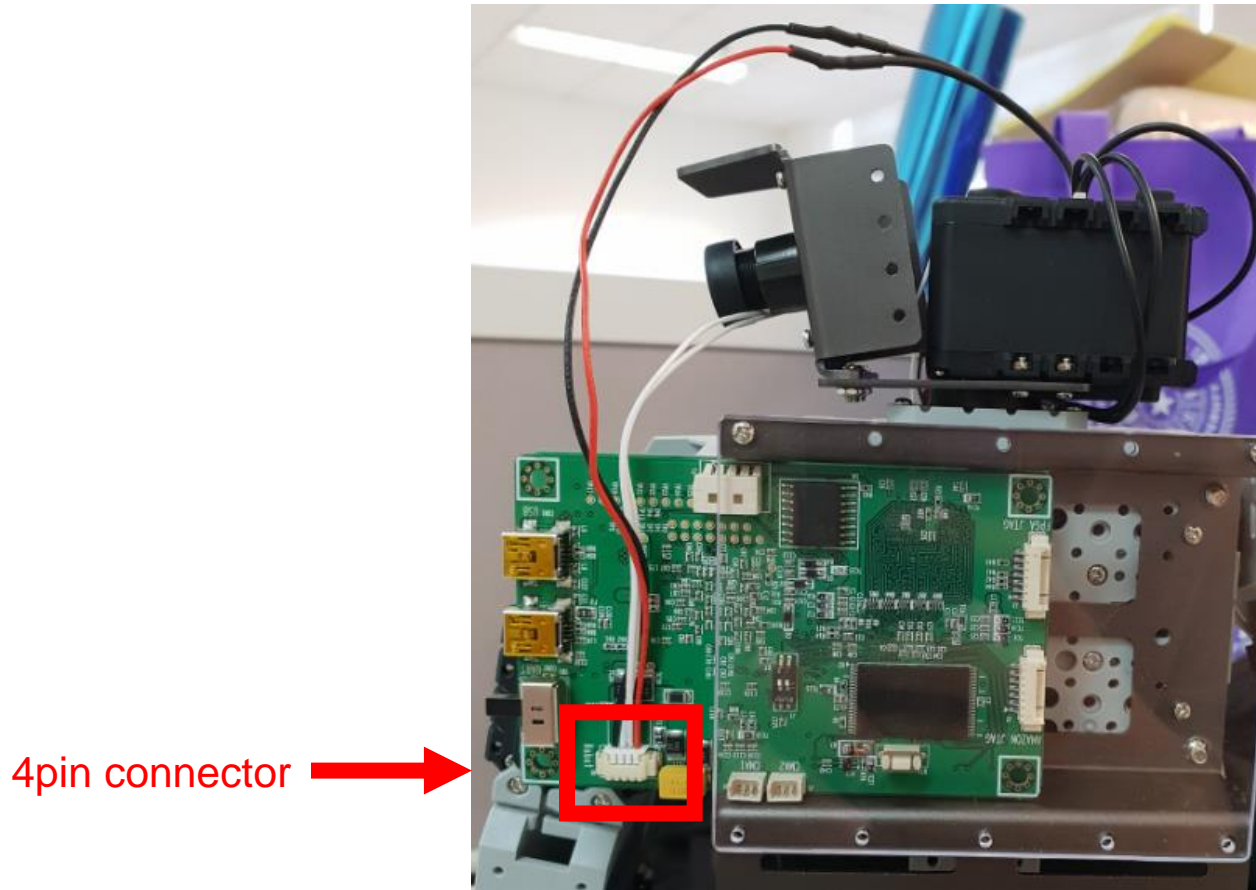
- Table above indicates the value of **WalkCommand** variable and it's corresponding action

Introduction



- Second, let's learn how to control the **robot motor** with the **AMAZON2 MCU**.

Introduction



- Connect the board and the robot.
- As shown in the figure, 4pin connector is used to communicate between the robot and the board and also supply power to the board.

MCU – Robot Controller Communication

- The MCU and robot controller communicate via UART.
- One packet consists of six bytes - 2-bytes of packet header, 2-bytes of data, and 2-bytes of 1's complementary of the data.
- **Packet Format : 0xFF, 0x55, Data_L, ~Data_L, Data_H, ~Data_H**
 - 0xFF, 0x55: packet header
 - Data_L: Data's low 1byte
 - ~Data_L: Data_L's 1's complement
 - Data_H: Data's low 1byte
 - ~Data_H: Data_H's 1's complement
- Ex) if Data=0x0001
 - Packet data : 0xFF, 0x55, 0x01, 0xFE, 0x00, 0xFF

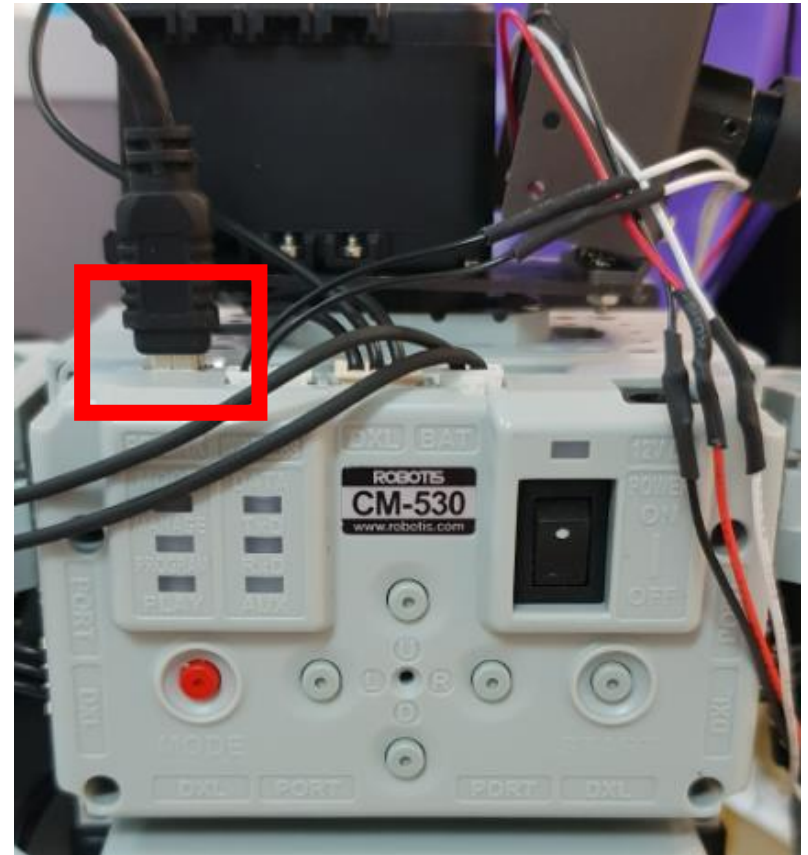
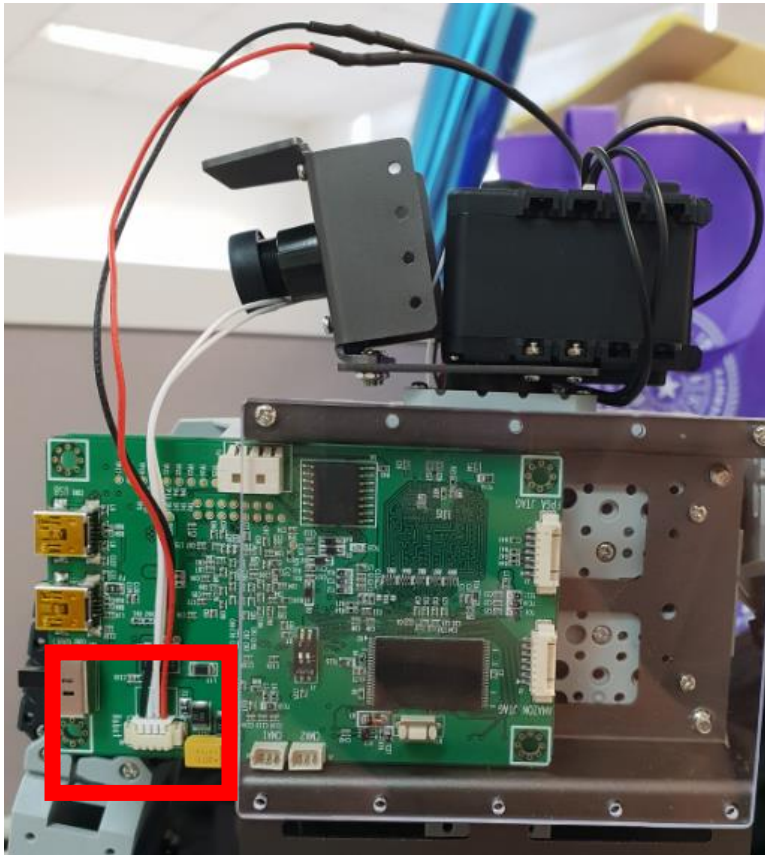
Initiating Communication

- Here is the program that let the robot perform the first motion if 'a' is typed on the keyboard, perform the second motion if 'b' is typed on the keyboard
- Unzip Bioloid_RobotBody.zip and place the file at *AMAZON2_robot_v1.13.4\user_app*
- Look at the *AMAZON2_robot_v1.13.4\user_app\Bioload_RobotBody\RobotBody_Test.c*
- The code first initialize UART communication.
 - Call **init_console** function.
 - Call **uart_open** function. (opens UART device of MCU.)
 - Call **uart_config** function. (baudrate of 57600, no parity bits, use UART1 channel)
- Second, send command to robot_controller.
 - Call **Send_Command** function in *robot_protocol.c*
- At last, close the UART.
 - Call **uart_close** function

MCU – Robot Controller Communication

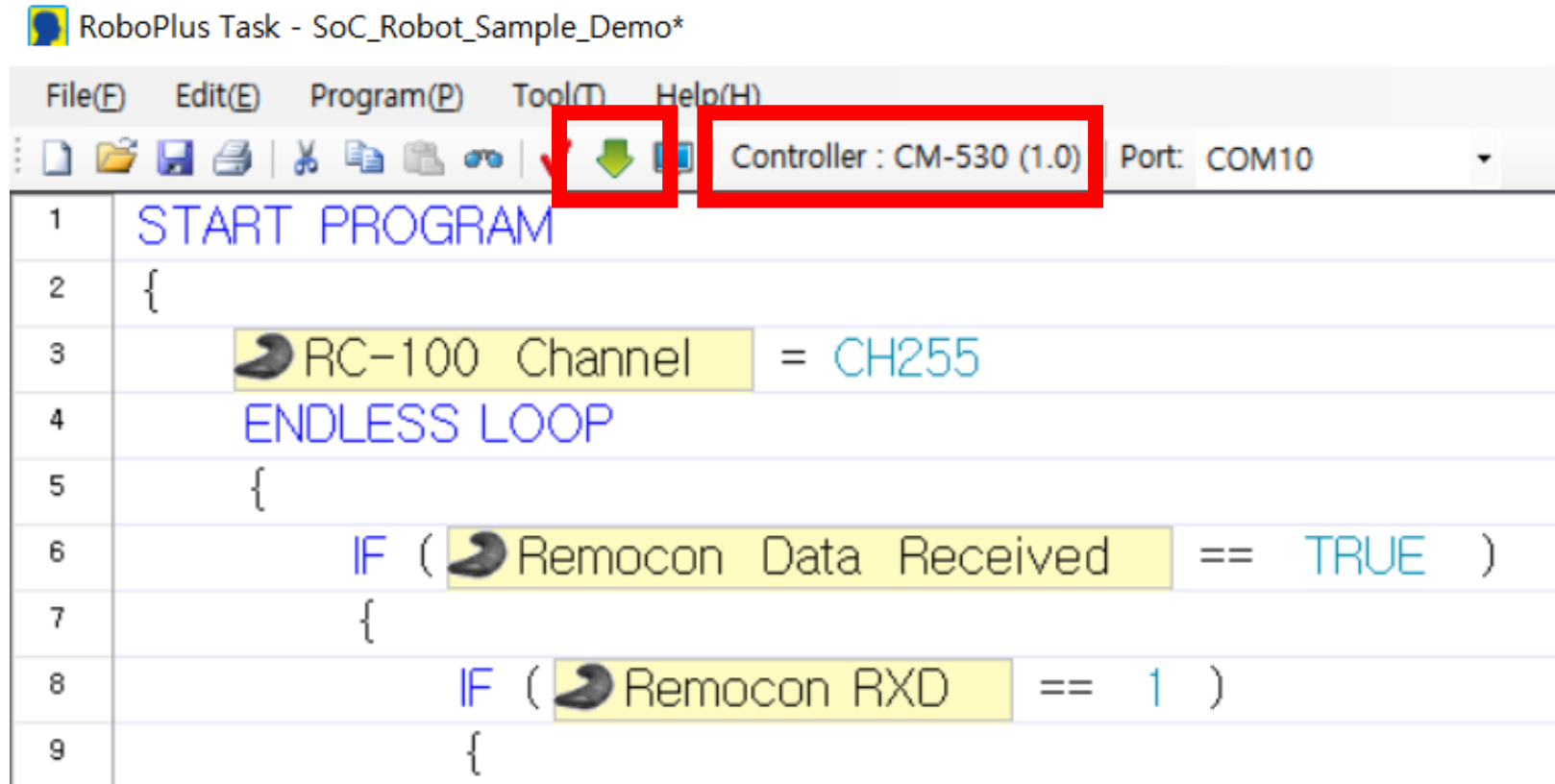
- See the *SoC_Robot_Sample_Demo.tsk* file.
- The robot controller must detect the packet sent from the MCU.
- Robot controller recognizes packets from the MCU as **remocon data**.
- Therefore, when the MCU sends packet, the variable **Remocon Data Received** changes to true.
- Value of received data is stored in variable **remocon RXD**. Then, the robot controller can take the appropriate motion according to the data.
- Let's program the MCU and robot controller.

MCU – Robot Controller Communication



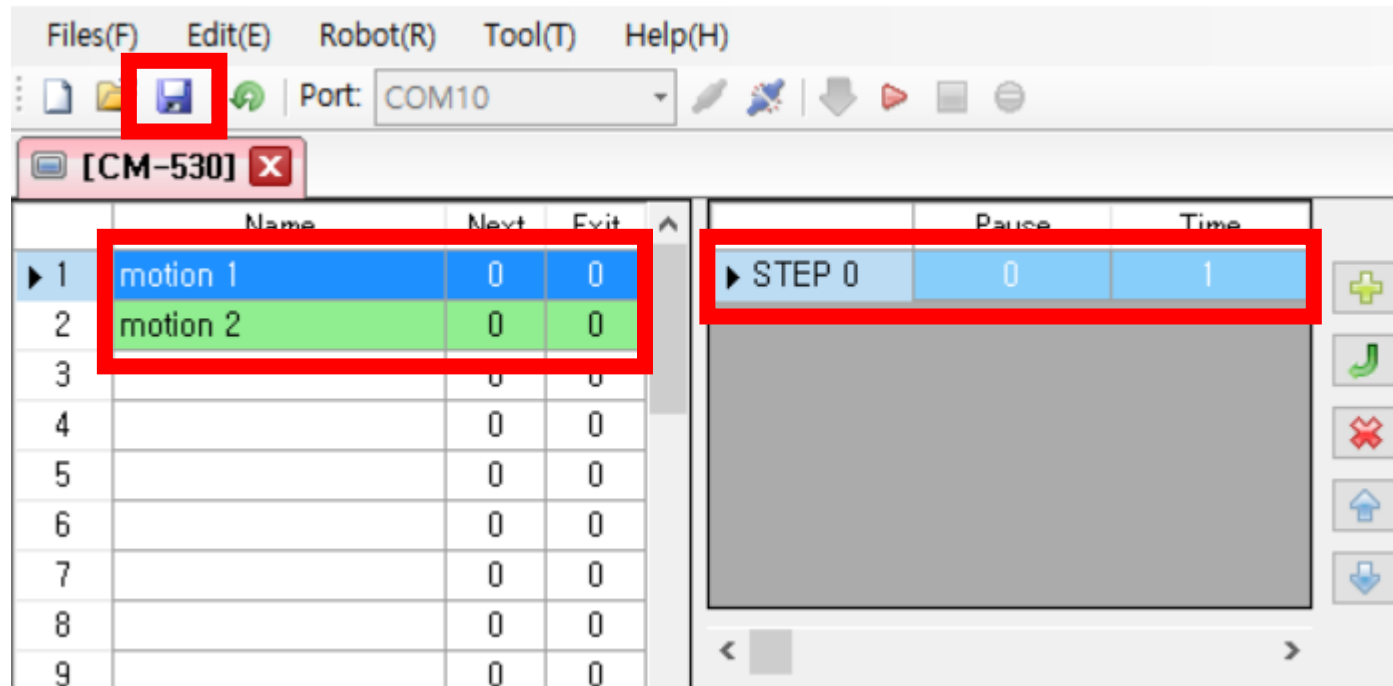
- Connect the robot controller and the board
- Connect the PC and robot controller
- Turn on the controller and the board

MCU – Robot Controller Communication



- Download the *SoC_Robot_Sample_Demo.tsk* to robot controller

MCU – Robot Controller Communication



- Write any robot motion of page 1 and 2 using **RoboPlus Motion**
- Download the motion to robot controller

MCU – Robot Controller Communication

```
# ./RobotBody_Test

*****
Welcome to Eagle Robot Platform Board
*****

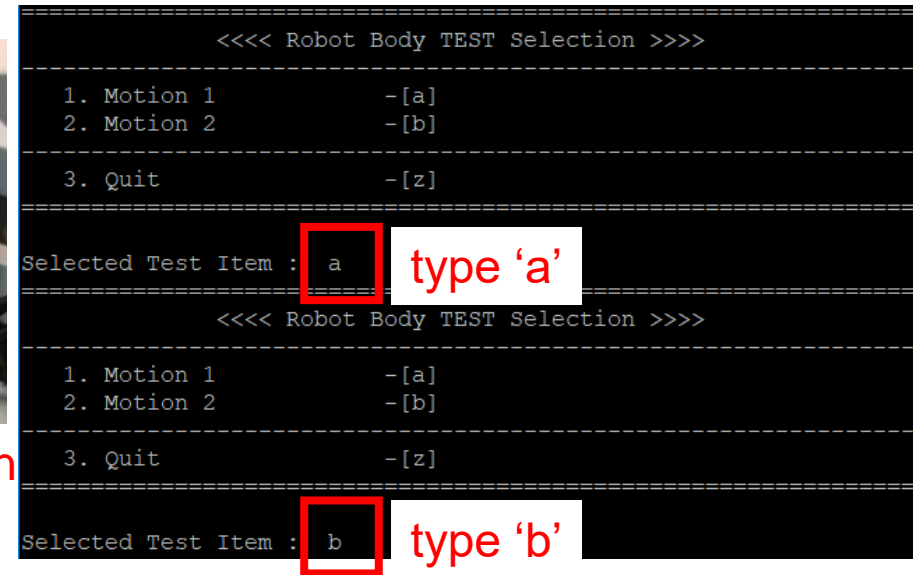
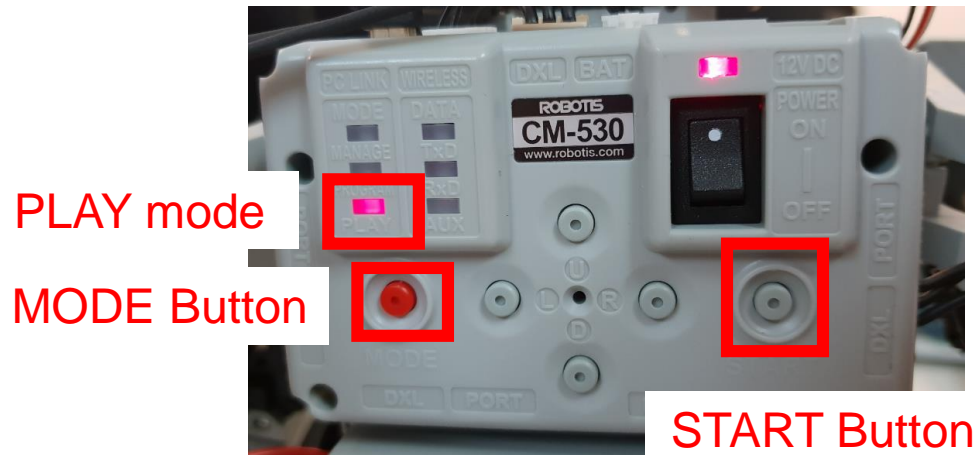
=====
<<<< Robot Body TEST Selection >>>>
=====

1. Motion 1          -[a]
2. Motion 2          -[b]
=====

3. Quit              -[z]
=====
```

- Disconnect the PC and robot controller
- Connect PC and the board
- Download the
AMAZON2_robot_v1.13.4\user_app\Bioloid_RobotBody\RobotBody_Test to the board (Xuecheng distributed the file)
- Type **chmod 777 RobotBody_Test**
- execute *RobotBody_Test*.

MCU – Robot Controller Communication



- Press **Mode Button** to set the robot in **PLAY mode**
- Press **START Button**
- Type '**a**' on the putty to execute motion 1
- Type '**b**' on the putty to execute motion 2