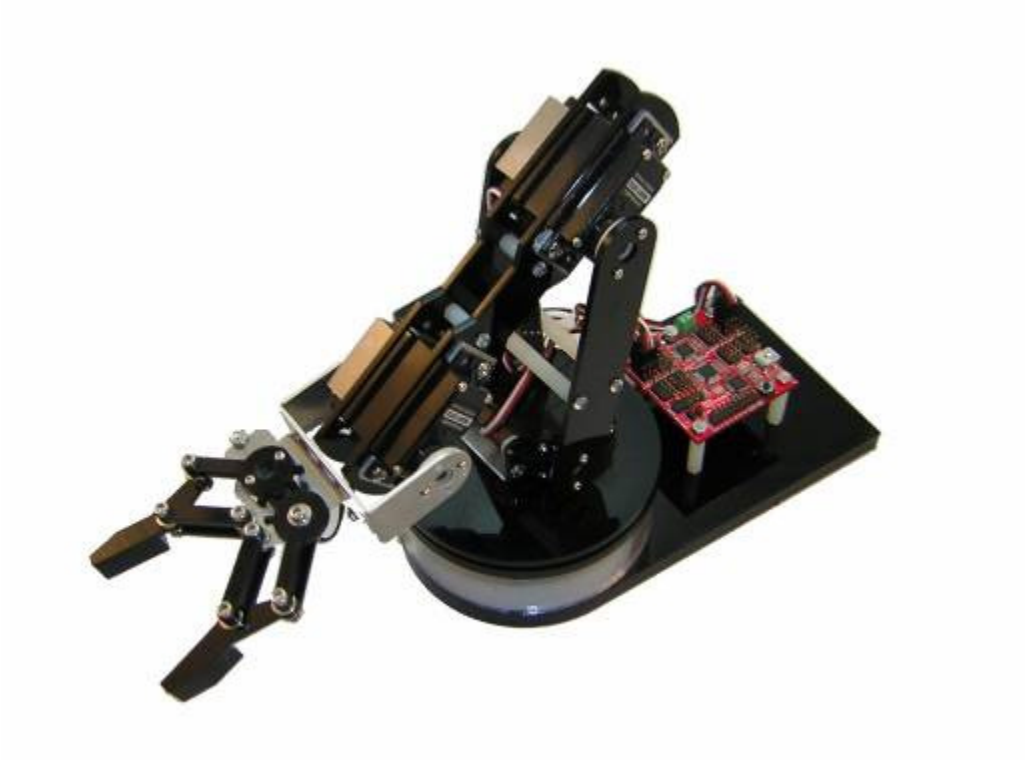
Robotic Arm

User's Manual

Document Rev 1.0



Passion for innovation

Trademark

Disclaimer

Full responsibility for any applications using Innovati products rests firmly with the user and as such Innovati will not be held responsible for any damages that may occur when using Innovati products. This includes damage to equipment or property, personal damage to life or health, damage caused by loss of profits, goodwill or otherwise. Innovati products should not be used for any life saving applications as Innovati's products are designed for experimental or prototyping purposes only. Innovati is not responsible for any safety, communication or other related regulations. It is advised that children under the age of 14 should only conduct experiments under parental or adult supervision.

Errata

We hope that our users will find this user's guide a useful, easy to use and interesting publication, as our efforts to do this have been considerable. Additionally, a substantial amount of effort has been put into this instruction manual to ensure accuracy and complete and error free content, however it is almost inevitable that certain errors may have remained undetected. If you find any errors in the instruction manual, contact us via email service@innovati.com.tw. For the most up-to-date information, please visit our web site at http://www.innovati.com.tw.

Overview

The Robotic Arm is a 5-DOF Robotic Arm especially designed for the robotics education. It is equipped with 6 RC servos for beginners to learn robotic arm basics. Its learning-oriented features make it an excellent curriculum for robotics applications. Students learn how to employ the servo control functions to move the Robotic Arm in the 3-D space. Simple sensing components are introduced to integrate environment sensing into the robotic arm applications.

Innovati's Servo Commander™ 16 (SC16) module is used as the main control board of the Robotic Arm, which incorporates the BASIC Commander® and a ServoRunnerA module with 16 general purpose I/Os controlling up to 16 servos simultaneously. For the Robotic Arm, only 6 servos are deployed, of which two servos are for the shoulders to operate simultaneously. The simple and integrated software functions enable users to directly control the servo movement by given speed or time. The positions and motion configurations including speed or time can be stored up to 250 frames. Thus various motions combinations can be achieved easily from these frames to perform a combination of actions.

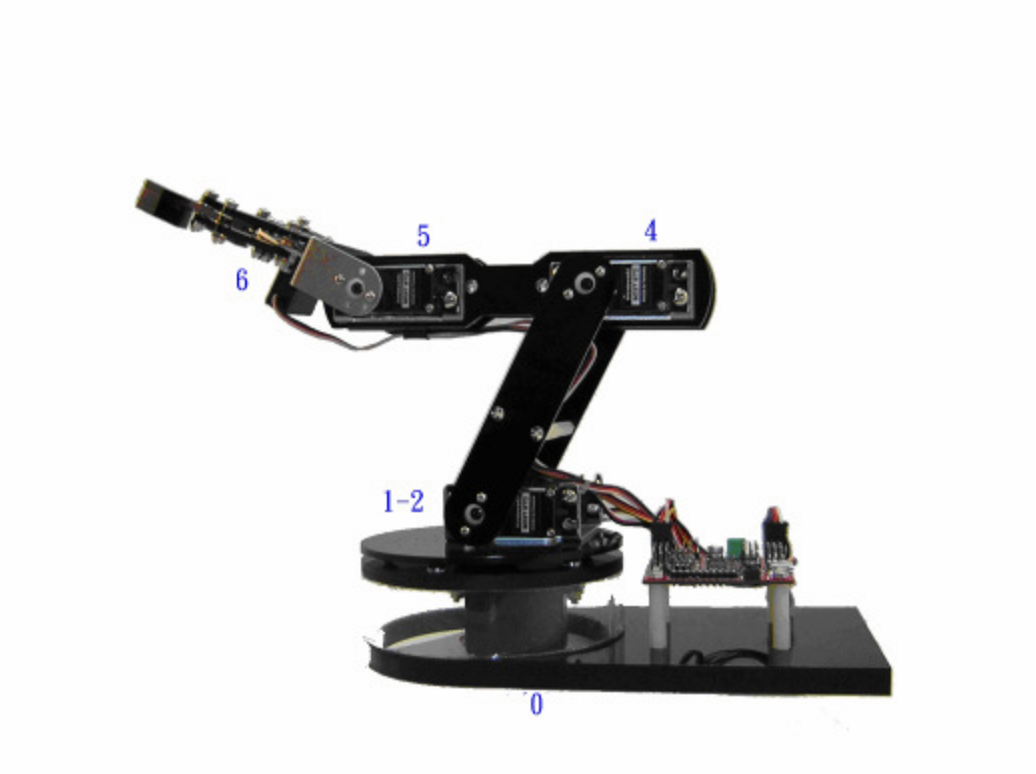
Additionally a variety of advanced Smart modules could be purchased to enhance the robotic arm functions in various projects and to integrate robotic arm with mobile vehicles to make the robotic arm to move objects from one place to another. By adding a video camera onto the vehicle, it can be used as surveillance robot.

Note that this manual mainly describes the functionality of the Robotic Arm unit. For details of the BASIC Commander® system and usage of the innoBASIC™ language, please refer to "BASIC Commander & innoBASIC Workshop User's Manual."

Operating the Robotic Arm

The Robotic Arm is preprogrammed and ready to be used. It can be control with Smart Peripheral such GamePad PS Wireless, Joysticks or Keypad (sold separately).

Servo Assignment



Servo	Description
0	Body
1	Shoulder
2	Shoulder
4	Elbow
5	Wrist
6	Hand

Servos

The servos used in the Robotic Arm can turn close to 180 degree. When the servo turns in its rotatable range, it has a corresponding positioning. It does not need to start with 0. For example, 499~2500, 800~2200, etc. If the servo turns beyond this range, it may not move and it can damage the servo. Therefore, it is important to pay attention to the position range of the servo.

Position values

Servo	Lowest	Reference	Highest
0	800 Right turn 80 deg	1500 Center, Forward	2200 Left turn 80 deg
1	1100 Backward 40 deg	1500 Center, vertical	1900 Forward 40 deg
2	1100 Backward 40 deg	1500 Center, vertical	1900 Forward 40 deg
4	1000 Downward 70 deg	1500 Forearm leveled with vertical arm	2000 Upward 70 deg
5	800 Upward 80 deg	1500 Hand leveled parallel to forearm	2200 Downward 80 deg
6	2200 All open	1500 All close	1500 All close

Notes

- The above table is for reference only. It may vary based on the actual Servo used.
- The Servos might have moved during assembly and transportation. Please use the Motion Editor to adjust the values before using the Arm.
- Different power supply or battery might damage the Servos. Please use DC 6~7V/5A with the provided Servos.
-

Servo Commander 16 control board

The Servo Commander 16 is the control board of the Robotic Arm. Check the figure and description below for their functions. Note that the rating power input to the Servo Commander 16 is 6~12V. However, due to the electrical characteristics of the servos used, power input range should be within 6~7V for this application.

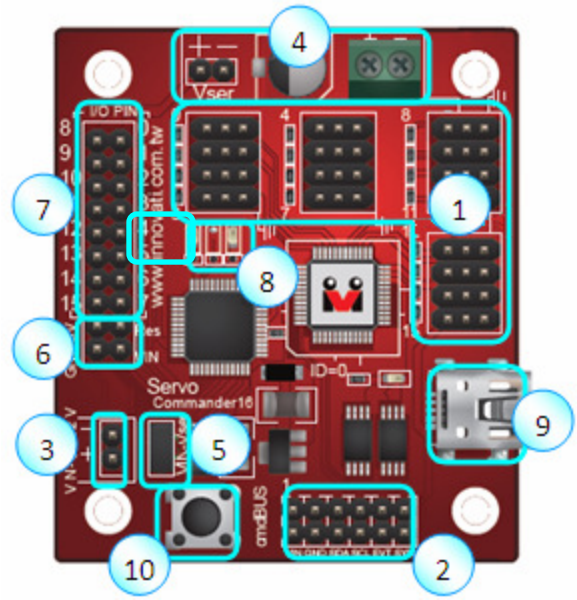


Fig. 1 Servo Commander™ 16

Item	Description
1	Sixteen Servo Connectors numbering from 0 through 15 are controlled by ServoRunnerA module with ID 0. Please check the pin label on the board, incorrect servo pin insertion may cause device damages.
2	Two cmdBUS™ connectors for other Innovati's Smart module connection. Please check the label on board when connecting the cmdBUS cable, incorrect insertion may damage the modules.
3	6~7V Power Input: It will be regulated to 5V for the electronics on the board.
4	Servo Power Input: Either of the two connectors can be used for servo power supply. Make sure the input voltage range is within the servo input voltage rating, otherwise the servos will be easily damaged.
5	These two pins are default shorted with a jumper, the Servo Power supply will be supplied to the 6~7V Power input pins also. In this way, only one power source is required. If you want to separate the electronics power supply from the servo power supply, remove the jumper and connect the electronics power supply to 6~7V power input pins.

6	Power source (VIN), Regulated 5V 300mA output pin (5V), ground pin (GND) and reset pin (RES) for your application circuit use.
7	16 general-purpose digital I/Os with labeled pin numbers on the board. Through the built-in software commands, they can be used as I2C or UART pins.
8	LED Indicators. Red LED will be lit when power is on. Yellow LED will be lit when Master/Slave is in communication. Green LED will be lit when USB is in communication.
9	Mini USB connector: via a USB cable connecting to computer for downloading and debugging programs.
10	RESET Button. To restart the program while the program is in execution. Note that it is prohibited to press this button during downloading, which will result in download failure.

Table 1 Servo Commander™ 16 Description

Servo and Power Connection

The Robotic Arm may be shipped assembled or unassembled upon user's requirement. If your Robotic Arm is not assembled, follow the description here to connect the servo cables to the Servo Commander 16 control board.

The module has 16 servo connectors with 3 pins for each connector. The servo connectors provide power and control signals to the servos, and are labeled 0 through 15. However, in this Robotic Arm application, only servo connectors 0,1,2,4,5 and 6 are used. To control the servos, connect the proper pins from of servo's connector cable to these connectors. Note that the servo cable colors and order may be different. Make sure the differences before you replace the servos. The power supply connection is shown in Figures 2. Connect the power adaptor to the green connector on the board and before connecting the power, check the servo operating voltage and current ratings to avoid damages to the servos.

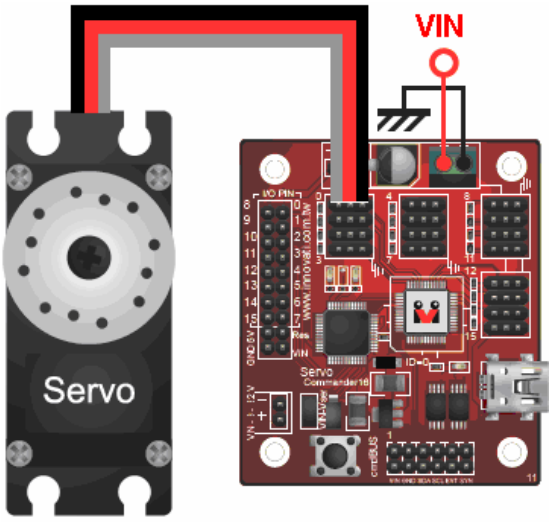


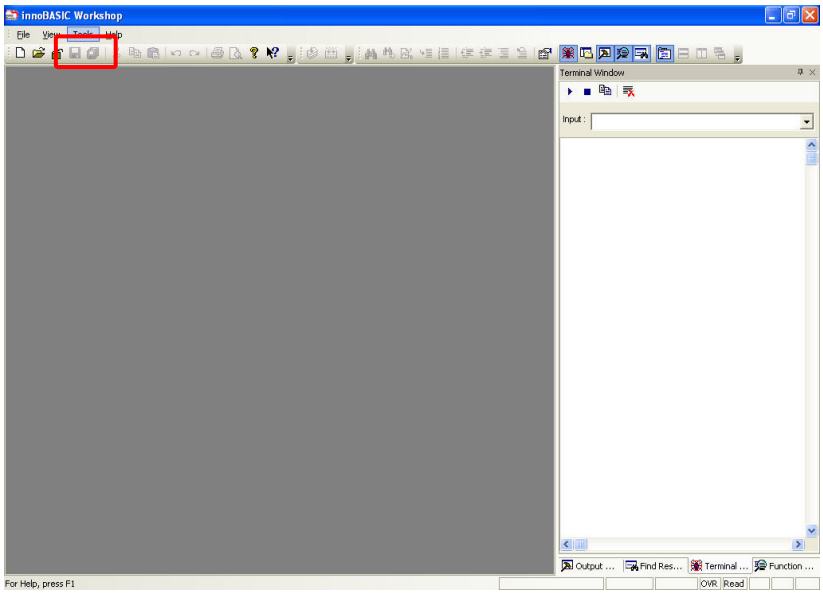
Fig. 2 Servo and Circuit use the same power source

Precautions for Operations

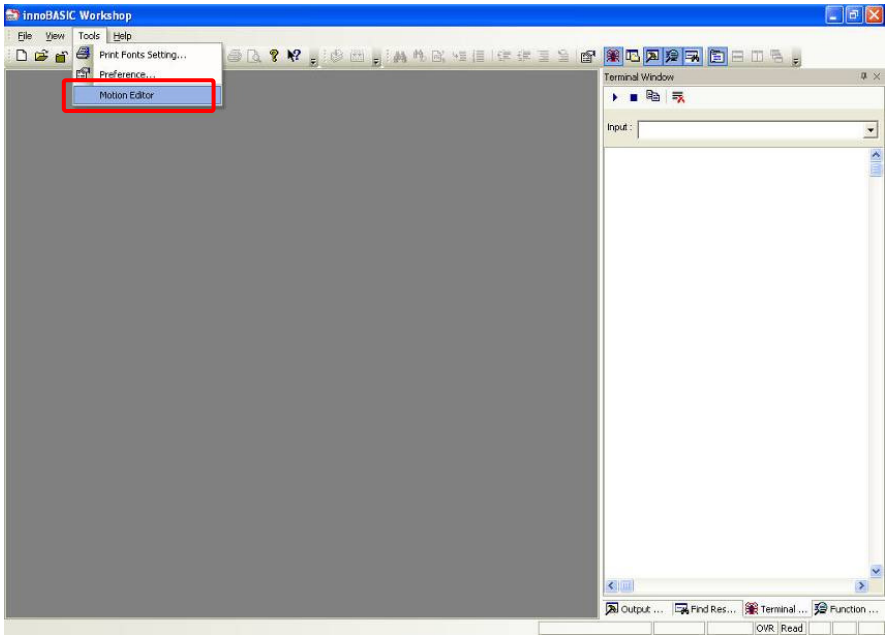
Servo number 1 and 2 are mechanically linked together to provide sufficient lift power to the arm, Remember to control these two servos simultaneously while using the Motion Editor or in the program. Long time confliction may result in permanent damage to the servos. For experienced users, they may wire the signal lines of servo 1 and 2 to the same connector, say 1 or 2, to control two servos with the same servo channel.

Motion Editor

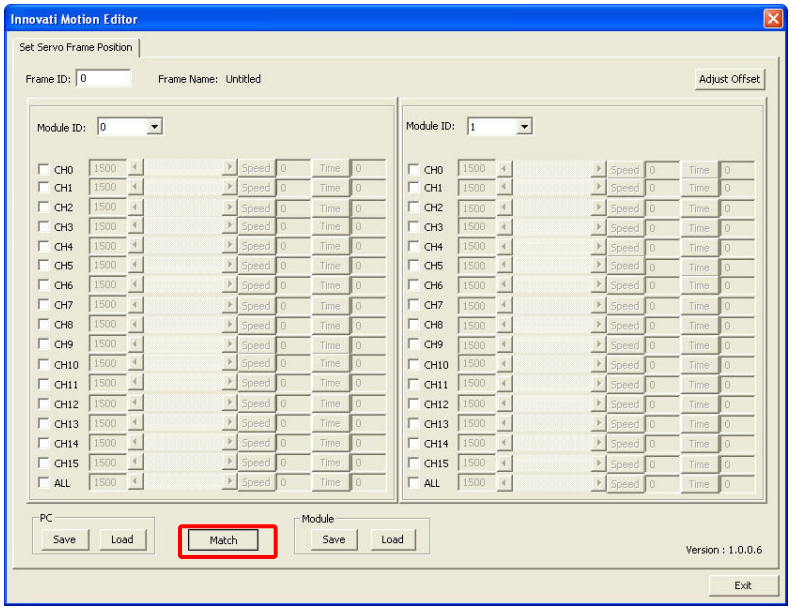
In the InnoBASIC™ Workshop, click “Tool” in the menu bar on the top.



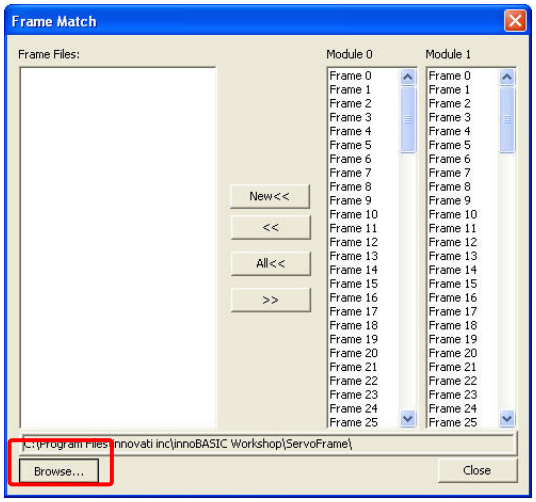
Click “Motion Editor” in the pull-down menu.



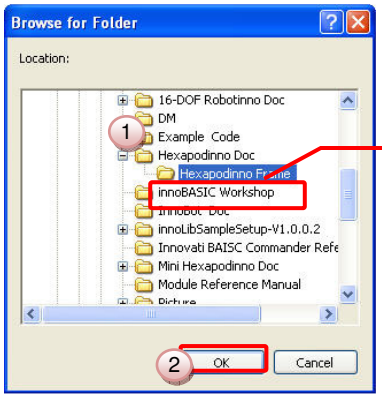
Click the button “Set the Corresponding Motion” at the bottom of the Motion Editor.



Click the “Browse” button at the lower left corner.

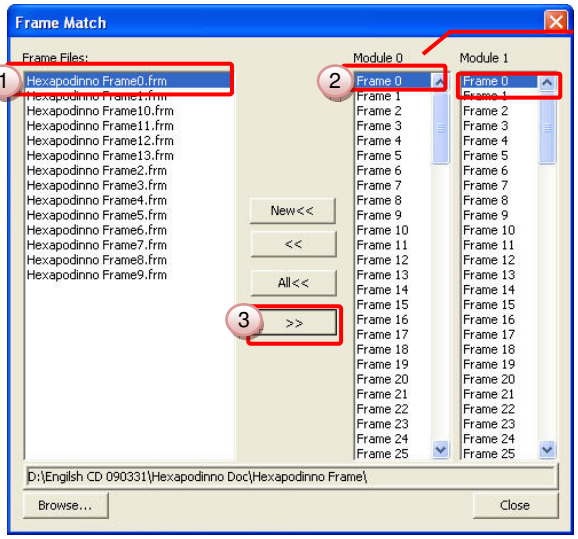


Set the “Browse File” folder to the folder where the frame are stored and then click the “OK” button.



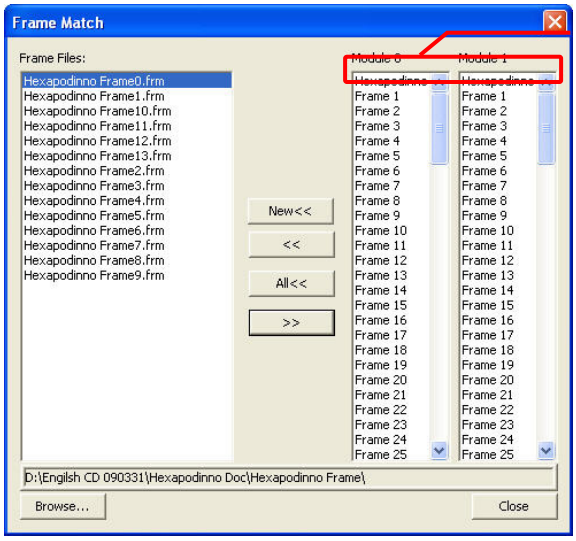
The selected folder will be highlighted.

Click the “Frame0.frm” below the motion files on the left side, click the “Frame 0” under the Module 0 and Module 1 and then click the “>>” button.



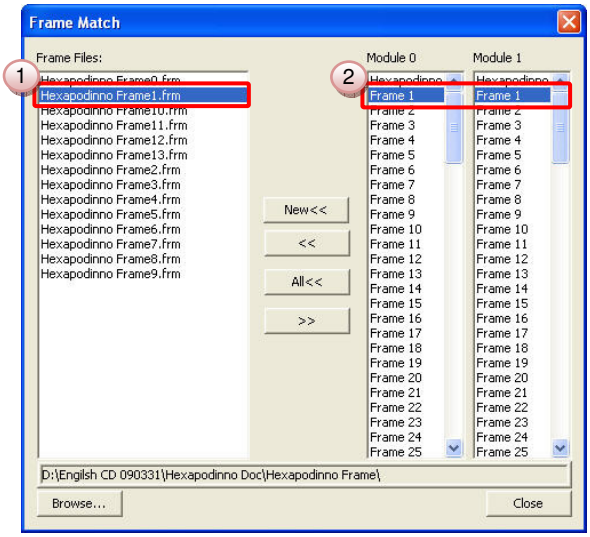
Before clicking the “>>” button to download the motion file into the module, please make sure that the “Frame 0” under the Module 0 and Module 1 has been selected and highlighted.

Make sure that the “Frame 0” under the Module 0 and Module 1 has become “Frame 0”.

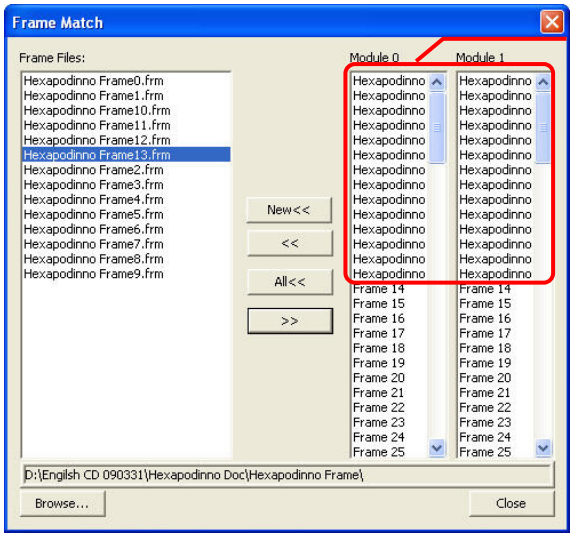


After the download is complete, the original text “frame0” will turn into “Frame0”.

Repeat the operation for all the motions till all the Frames have been downloaded to the corresponding frames.



After all the download operations are complete, it is clear that all the motions above Frame14 under the Module 0 and Module 1 have been changed to the corresponding motions.

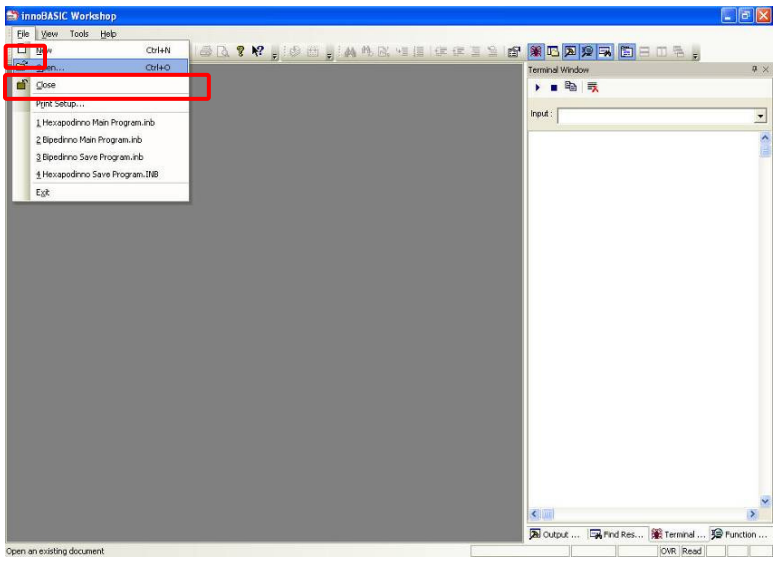


Please make sure that first 14 Frames have been successfully downloaded.

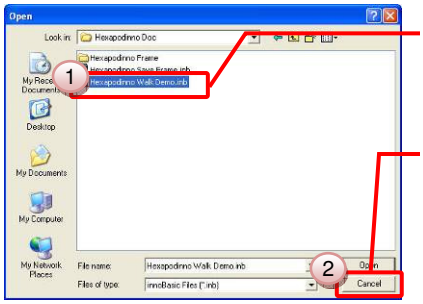
After the verifying the operations, click the “Close” button at the lower right corner to close the window for setting the corresponding motions.

In the Edit Servomotor Motions window, click the “Exit” button at the lower right corner to close the Motion Editor.

Click “File” in the menu bar and click “Open”.



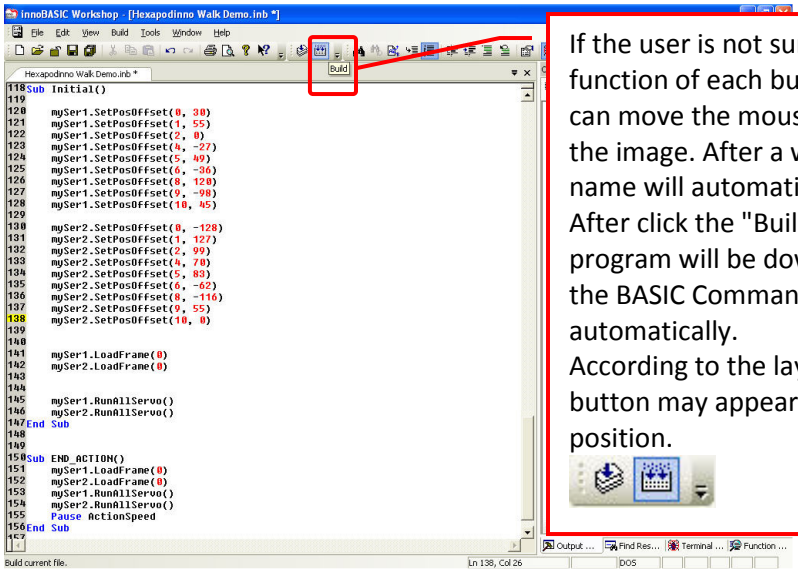
Please select the folder and click “Open”.



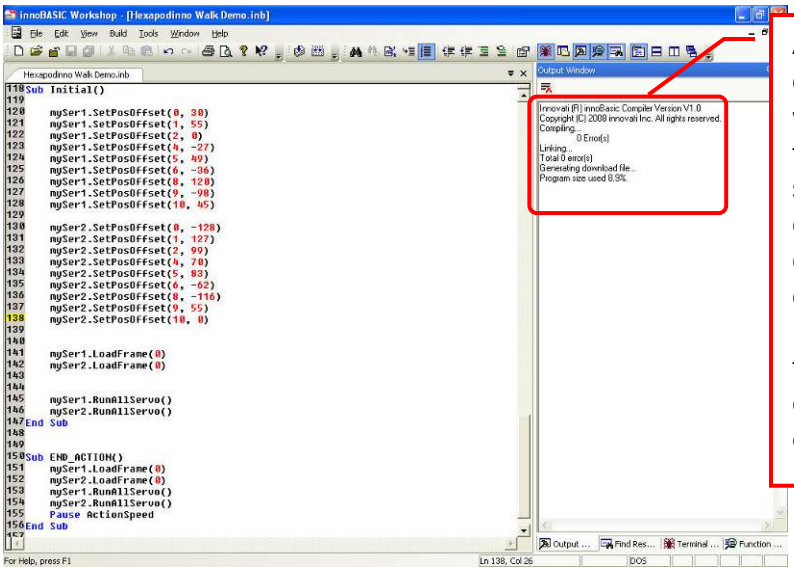
The selected folder will be highlighted.

Click the “Open” button to download the program into the innoBASIC Workshop for editing or creating motions.

Press the “Build” button and wait until the download is complete.



If the user is not sure about the function of each button, the user can move the mouse pointer over the image. After a while, the English name will automatically appear. After click the "Build" button, the program will be downloaded into the BASIC Commander and stored automatically.
According to the layout, the "Build" button may appear at different position.



After the download is complete, the output window will display the used memory space. If there is any error, it will be displayed in the output window. Please make sure that no error is displayed in the output window.

Command Set

The following table lists all the unique commands provided with the ServoRunnerA Module. Note that essential words in the commands will be written in **bold** type and ***italics*** in bold type. The bold type word must be written exactly as shown, whereas the italic bold type words must be replaced with the user values. Note that the innoBASIC™ language is case-insensitive.

Note that the Servo Commander 16 board is composed of a BASIC Commander unit and a ServoRunnerA peripheral module. To execute functions related to ServoRunnerA module, please declare the module ID number as 0 in the program, i.e. **Peripheral *ModuleName* As ServoRunnerA @ 0**.

Command Syntax	Description
Servo Position Commands	
SetPos (*ID*, *Pos*)	Sets the servo with ***ID***, ranging from 0 to 15, for operation. The target position is set by ***Pos*** ranging from 499~2500 in µs unit. If the given value is out of this range, the command will not be executed.
SetPosAndRun(*ID*, *Pos*)	Same as command above. Except after settings are done, the servo will start to move.
SetPosSpd(*ID*, *Pos*, *Spd*)	Sets the servo with ***ID***, ranging from 0 to 15, for

	operation. The target position is set by ***Pos*** ranging from 499~2500 in μs unit and traveling at a speed of ***Spd*** ranging 0~65535 with unit μs/s. The larger the ***Spd*** value is, the faster the servo travels. Note that the ***Spd*** with value 0 will be regarded as full speed.
SetPosSpdAndRun(*ID*, *Pos*, *Spd*)	Same as command above. Except after settings are done, the servo will start to move.
SetPosTime(*ID*, *Pos*, *Time*)	Sets the servo with ***ID***, ranging from 0 to 15, for operation. The target position is set by ***Pos*** ranging from 499~2500 in μs unit and traveling to the target position in ***Time*** ranging from 0~65535 milliseconds. Note that if the value of ***Time*** is too short, including 0, the servo will travel at full speed.
SetPosTimeAndRun(*ID*, *Pos*, *Time*)	Same as command above. Except after settings are done, the servo will start to move.
Servo Start Commands	
Run1Servo(*ID1*) **Run2Servo(*ID1*, *ID2*)** **Run3Servo(*ID1*, …, *ID3*)** **:** **Run15Servo(*ID1*, …, *ID15*)** **RunAllServo()**	According to the set value of servo ***ID***s, ranging from 0 to 15, each corresponding servo will perform the preset operation. If the servo starts without the speed or time settings but only the position setting, the servo will travel at the maximum speed. If any ***ID*** value is out of its range, this command will not be executed.
Run1ServoWithEventA(*ID1*) **Run2ServoWithEventA(*ID1*, *ID2*)** **Run3ServoWithEventA(*D1*, …, *ID3*)** **:** **Run15ServoWithEventA(*ID1*, …, *ID15*)** **RunAllServoWithEventA()**	Same as above, except that the event A will be triggered when all the indicated servos reach their target positions.
Run1ServoWithEventB(*ID1*) **Run2ServoWithEventB(*ID1*, *ID2*)** **Run3ServoWithEventB(*D1*, …, *ID3*)** **:** **Run15ServoWithEventB(*ID1*, …, *ID15*)** **RunAllServoWithEventB()**	Same as above, except that the event B will be triggered when all the indicated servos reach their target positions.
Run1ServoWithEventC(*ID1*) **Run2ServoWithEventC(*ID1*, *ID2*)** **Run3ServoWithEventC(*D1*, …, *ID3*)** **:** **Run15ServoWithEventC(*ID1*, …, *ID15*)** **RunAllServoWithEventC()**	Same as above, except that the event C will be triggered when all the indicated servos reach their target positions.
Run1ServoWithEventD(*ID1*) **Run2ServoWithEventD(*ID1*, *ID2*)** **Run3ServoWithEventD(*D1*, …, *ID3*)** **:** **Run15ServoWithEventD(*ID1*, …, *ID15*)** **RunAllServoWithEventD()**	Same as above, except that the event D will be triggered when all the indicated servos reach their target positions.

Servo Stop Commands	
Pause1Servo(*ID1*) **Pause2Servo(*ID1*, *ID2*)** **Pause3Servo(*ID1*, …, *ID3*)** **:** **Pause15Servo(*ID1*, …, *ID15*)** **PauseAllServo()**	According to the set value of servo ***ID***s, ranging from 0 to 15, each corresponding servo will stop and hold at the present position. If any **ID** value is out of its range, this command will not be executed.
Stop1Servo(*ID1*) **Stop2Servo(*ID1*, *ID2*)** **Stop3Servo(*ID1*, …, *ID3*)** **:** **Stop15Servo(*ID1*, …, *ID15*)** **StopAllServo()**	Same as above, except that the module will cease sending control signal to the servo. As a result, the servo will stop but not hold at the present position. External force might be able to change its position.
Servo Status and Setting Commands	
Get1ServoReadyStatus(*ID1*, *Status*) **Get2ServoReadyStatus(*ID1*, *ID2*, *Status*)** **Get3ServoReadyStatus(*ID1*, …, *ID3*, *Status*)** **:** **Get15ServoReadyStatus(*ID1*, …, *ID15*, *Status*)** **GetAllServoReadyStatus(*Status*)**	Gets the operation status of the servo(s) indicated by ***ID***s, ranging from 0 to 15, and stores the status in ***Status***. When all the servos reach their target positions, the returned status will be 1, otherwise value 0 will be returned.
GetNowPos (*ID*, *Pos*)	Gets the current position of the servo indicated by ***ID***, ranging from 0 to 15, and then stores it in the variable ***Pos*** of type Word. Note that the position returned is an estimated position.
GetPos(*ID*, *Pos*)	Gets the target position of the servo indicated by ***ID***, ranging from 0 to 15, and then stores it in the variable ***Pos*** of type Word.
GetPosOffset(*ID*, *Offset*)	Gets the offset position of the servo indicated by ***ID***, ranging from 0 to 15, and then stores it in the variable ***Offset*** of type Short, ranging form -128 to 127 μs.
GetSpdAndTime(*ID*, *Type*, *Value*)	Gets the motion type of the servo indicated by ***ID***, ranging from 0 to 15, and stores the values in ***Type***. The corresponding setting values are stored in the variable ***Value*** of type Word. If the servo travel type is set as speed, then the returned value for ***Type*** will be 1. If the servo travel type is set as time, then the returned value for ***Type*** will be 0.
LoadFrame(*FrameID*)	Loads the servo operation settings from the frame memory block indicated by ***FrameID***, ranging from 0 to 249, as the current target position and motion type of the servos.
LoadOffset()	Loads the servo offset settings from EEPROM.
SaveFrame(*FrameID*)	Saves the current settings of servo operations into the frame indicated by ***FrameID***, ranging from 0 to 249.
SaveOffset()	Saves the servo offset settings into EEPROM.

SetPosOffset(*ID*, *Offset*)	Sets the offset of the servo indicated by ***ID*** with the value ***Offset***, ranging from -128 to 127.

Events Name	Description
ServoPosReadyEventA	Executes the **Run*N*ServoWithEventA** command, where ***N*** can be literally **1~15** or **All**. When all the indicated servos reach their target positions, event A will be triggered.
ServoPosReadyEventB	Executes the **Run*N*ServoWithEventB** command, where ***N*** can be literally **1~15** or **All**. When all the indicated servos reach their target positions, event B will be triggered.
ServoPosReadyEventC	Executes the **Run*N*ServoWithEventC** command, where ***N*** can be literally **1~15** or **All**. When all the indicated servos reach their target positions, event C will be triggered.
ServoPosReadyEventD	Executes the **Run*N*ServoWithEventD** command, where ***N*** can be literally **1~15** or **All**. When all the indicated servos reach their target positions, event D will be triggered.

Example Program

Wrist Twisting

```
Peripheral mySer As ServoRunnerA @ 0

Dim i,t As Integer

Sub main()
  ' initial positions
  mySer.SetPosTimeAndRun(0,1500,0)
  mySer.SetPosTimeAndRun(1,1500,0)
  mySer.SetPosTimeAndRun(2,1500,0)
  mySer.SetPosTimeAndRun(4,1500,0)
  mySer.SetPosTimeAndRun(5,1500,0)
  mySer.SetPosTimeAndRun(6,1800,0)
  Pause 3000

  Do
    mySer.SetPosSpdAndRun(0,2200,500) ' body turns left at 500ms
    Pause 6000                        ' give time to turn

    mySer.SetPosSpdAndRun(0,800,0)    ' body turns right
    Pause 6000

    mySer.SetPosSpdAndRun(0,1500,1800)  ' body returns to center
fast
    Pause 6000
```

```
    ' the following commands uses 3 sec to turn,
    ' make sure to give enough Pause before executing the next
command
    mySer.SetPosTimeAndRun(0,2000,3000)
    Pause 6000

    mySer.SetPosTimeAndRun(0,1000,3000)
    Pause 6000

    mySer.SetPosTimeAndRun(0,1500,3000)
    Pause 10000

    ' the following commands are run and then saved as frames
    mySer.SetPosTimeAndRun(0,800,2000)
    Pause 3000
    mySer.SaveFrame(1)
    Pause 100

    mySer.SetPosTimeAndRun(0,1100,2000)
    Pause 3000
    mySer.SaveFrame(2)
    Pause 100

    mySer.SetPosTimeAndRun(0,1900,2000)
    Pause 3000
    mySer.SaveFrame(3)
    Pause 100

    mySer.SetPosTimeAndRun(0,2200,2000)
    Pause 3000
    mySer.SaveFrame(4)
    Pause 100

    mySer.SetPosTimeAndRun(0,1500,2000)
    Pause 3000
    mySer.SaveFrame(5)
    Pause 6000

    ' load the frames and execute automatically 3 times
    For i=1 To 3
      mySer.LoadFrame(3)
      mySer.Run1Servo(0)
      Pause 2000
      mySer.LoadFrame(2)
      mySer.Run1Servo(0)
      Pause 2000
```

```
    Next

    For i=1 To 3
      mySer.LoadFrame(4)
      mySer.Run1Servo(0)
      Pause 2000
      mySer.LoadFrame(1)
      mySer.Run1Servo(0)
      Pause 2000
    Next

    mySer.LoadFrame(5)
    mySer.Run1Servo(0)
    Pause 10000

  Loop

End Sub
```

Gripping

Make sure not to grip or lift any object does not have any sharp edge or, is not too big or too heavy to avoid damage to the servo or the brackets.

```
Peripheral mySer As ServoRunnerA @ 0

Sub main()
  mySer.SetPosTimeAndRun(0,1500,0)
  mySer.SetPosTimeAndRun(1,1500, 0)
  mySer.SetPosTimeAndRun(2,1500, 0)
  mySer.SetPosTimeAndRun(4,1500,0)
  mySer.SetPosTimeAndRun(5,1500, 0)
  mySer.SetPosTimeAndRun(6,1800,0)
  Pause 3000

  Do
    mySer.SetPosSpdAndRun(6,2200,500)
    Pause 6000

    mySer.SetPosSpdAndRun(6,1500, 500)
    Pause 6000

    mySer.SetPosSpdAndRun(6,2200,1500)
    Pause 6000

    mySer.SetPosSpdAndRun(6,1800,0)
```

```
    Pause 6000

    mySer.SetPosSpdAndRun(6,2200,0)
    Pause 6000

    mySer.SetPosTimeAndRun(6,1500,2000)
    Pause 6000

    mySer.SetPosTimeAndRun(6,2200,2000)
    Pause 6000

    mySer.SetPosTimeAndRun(6,1800,500)
    Pause 10000

  Loop
End Sub
```

Lifting shoulder and elbow, wrist bending

When lifting the shoulders, make sure both servos 1 and 2 are turning at the same time and at the same speed.

```
Peripheral mySer As ServoRunnerA @ 0
Dim t, i As Integer

Sub main()
  t =1000

  '******************(0)*******************
  mySer.SetPosTime (0,1500, t)
  mySer.SetPosTime (1,1500, t)
  mySer.SetPosTime (2,1500, t)
  mySer.SetPosTime (4,1500, t)
  mySer.SetPosTime (5,1500, t)
  mySer.SetPosTime (6,1800, t)
  mySer.SaveFrame(0)
  Pause 100

  mySer.LoadFrame(0)
  Pause 100
  mySer.RunAllServo()
  Pause 2000

  '******************(1)*******************
  mySer.SetPosTimeAndRun(1,1800, t)
  mySer.SetPosTimeAndRun(2,1800, t)
```

```
Pause 2000
mySer.SetPosTimeAndRun(4,1200, t)
Pause 2000
mySer.SetPosTimeAndRun(5,1800, t)
Pause 2000

mySer.LoadFrame(0)
Pause 100
mySer.RunAllServo()
Pause 2000

mySer.SetPosTime (1,1800, t)
mySer.SetPosTime (2,1800, t)
mySer.SetPosTime (4,1200, t)
mySer.SetPosTime (5,1800, t)
mySer.RunAllServo()
Pause 3000
mySer.SaveFrame(1)
Pause 100

mySer.LoadFrame(0)
Pause 100
mySer.RunAllServo()
Pause 2000

'******************(2)*******************
mySer.SetPosTimeAndRun(1,1800, t)
mySer.SetPosTimeAndRun(2,1800, t)
Pause 2000
mySer.SetPosTimeAndRun(4,1800, t)
Pause 2000
mySer.SetPosTimeAndRun(5,1800, t)
Pause 2000

mySer.LoadFrame(0)
Pause 100
mySer.RunAllServo()
Pause 2000

mySer.SetPosTime (1,1800, t)
mySer.SetPosTime (2,1800, t)
mySer.SetPosTime (4,1800, t)
mySer.SetPosTime (5,1800, t)
mySer.Run4Servo(1, 2, 4, 5)
Pause 3000
mySer.SaveFrame(2)
Pause 100
```

```
mySer.LoadFrame(0)
Pause 100
mySer.RunAllServo()
Pause 2000

'******************(3)*******************
mySer.SetPosTimeAndRun(1,1800, t)
mySer.SetPosTimeAndRun(2,1800, t)
Pause 2000
mySer.SetPosTimeAndRun(4,1800, t)
Pause 2000
mySer.SetPosTimeAndRun(5,1200, t)
Pause 2000

mySer.LoadFrame(0)
Pause 100
mySer.RunAllServo()
Pause 2000

mySer.SetPosTime (1,1800, t)
mySer.SetPosTime (2,1800, t)
mySer.SetPosTime (4,1800, t)
mySer.SetPosTime (5,1200, t)
mySer.Run4Servo(1, 2, 4, 5)
Pause 3000
mySer.SaveFrame(3)
Pause 100

mySer.LoadFrame(0)
Pause 100
mySer.RunAllServo()
Pause 2000

'******************(4)*******************
mySer.SetPosTimeAndRun(1,1200, t)
mySer.SetPosTimeAndRun(2,1200, t)
Pause 2000
mySer.SetPosTimeAndRun(4,1200, t)
Pause 2000
mySer.SetPosTimeAndRun(5,1800, t)
Pause 2000

mySer.LoadFrame(0)
Pause 100
mySer.RunAllServo()
Pause 2000
```

```
mySer.SetPosTime (1,1200, t)
mySer.SetPosTime (2,1200, t)
mySer.SetPosTime (4,1200, t)
mySer.SetPosTime (5,1800, t)
mySer.Run4Servo(1, 2, 4, 5)
Pause 3000
mySer.SaveFrame(4)
Pause 100

mySer.LoadFrame(0)
Pause 100
mySer.RunAllServo()
Pause 2000

'******************(119)*******************
mySer.SetPosTimeAndRun(1,1100, t)
mySer.SetPosTimeAndRun(2,1100, t)
mySer.SetPosTimeAndRun(4,1100, t)
mySer.SetPosTimeAndRun(5,1600, t)
Pause 3000
mySer.SaveFrame(119)
Pause 100

'************************************
mySer.LoadFrame(0)
Pause 100
mySer.RunAllServo()
Pause 2000

For i = 1 To 4
  mySer.LoadFrame(1)
  Pause 100
  mySer.RunAllServo()
  Pause 1000
  mySer.LoadFrame(2)
  Pause 100
  mySer.RunAllServo()
  Pause 1000
  mySer.LoadFrame(3)
  Pause 100
  mySer.RunAllServo()
  Pause 1000
  mySer.LoadFrame(4)
  Pause 100
  mySer.RunAllServo()
  Pause 1000
```

```
  Next

  mySer.LoadFrame(0)
  Pause 100
  mySer.RunAllServo ()
  Pause 1000

  mySer.LoadFrame(119)
  Pause 100
  mySer.RunAllServo()
  Pause 2000

End Sub
```

Arm Coordination

```
Peripheral mySer As ServoRunnerA @ 0
Dim t As Integer

Sub main()
  Do
    t=500
    Call start()
    Call main2()
    Call start()
    Call main3()
    Call start()
    Call main4()
    Call start()
    Call main5()
    Call start()
    Call main6()
    Call start()
    Call main7()
    Call start()
    Call main8()
    Call start()
    Call main9()
    Call start()
    Call over()
  Loop
End Sub

Sub start()
  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1500,t)
```

```
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1500,t)
  mySer.SetPosTimeAndRun(5,1500,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 2000
End Sub

Sub main2()
  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1200,t)
  mySer.SetPosTimeAndRun(5,2100,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700

  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1200,t)
  mySer.SetPosTimeAndRun(5,2100,t)
  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(0,2100,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1500,t)
  mySer.SetPosTimeAndRun(5,2000,t)
  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(0,2100,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1200,t)
  mySer.SetPosTimeAndRun(5,2100,t)
  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(0,2100,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1200,t)
  mySer.SetPosTimeAndRun(5,2100,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
```

```
End Sub

Sub main3()

  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1600,t)
  mySer.SetPosTimeAndRun(2,1600,t)
  mySer.SetPosTimeAndRun(4,1300,t)
  mySer.SetPosTimeAndRun(5,2050,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(0,900,t)
  mySer.SetPosTimeAndRun(1,1600,t)
  mySer.SetPosTimeAndRun(2,1600,t)
  mySer.SetPosTimeAndRun(4,1500,t)
  mySer.SetPosTimeAndRun(5,2050,t)
  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(1,1600,t)
  mySer.SetPosTimeAndRun(2,1600,t)
  mySer.SetPosTimeAndRun(4,1300,t)
  mySer.SetPosTimeAndRun(5,2050,t)
  Pause 700
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
End Sub

Sub main4()

  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1700,t)
  mySer.SetPosTimeAndRun(2,1700,t)
  mySer.SetPosTimeAndRun(4,1400,t)
  mySer.SetPosTimeAndRun(5,2000,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(0,1800,t)
  mySer.SetPosTimeAndRun(1,1650,t)
  mySer.SetPosTimeAndRun(2,1650,t)
  mySer.SetPosTimeAndRun(4,1400,t)
```

```
  mySer.SetPosTimeAndRun(5,2000,t)
  Pause 700
  mySer.SetPosTimeAndRun(1,1700,t)
  mySer.SetPosTimeAndRun(2,1700,t)
  mySer.SetPosTimeAndRun(4,1400,t)
  mySer.SetPosTimeAndRun(5,2000,t)
  Pause 700
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
End Sub

Sub main5()
  mySer.SetPosTimeAndRun(0,900,t)
  Pause 700
  mySer.SetPosTimeAndRun(1,1600,t)
  mySer.SetPosTimeAndRun(2,1600,t)
  mySer.SetPosTimeAndRun(4,1300,t)
  mySer.SetPosTimeAndRun(5,2050,t)
  Pause 700
  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1600,t)
  mySer.SetPosTimeAndRun(2,1600,t)
  mySer.SetPosTimeAndRun(4,1500,t)
  mySer.SetPosTimeAndRun(5,2050,t)
  Pause 700
  mySer.SetPosTimeAndRun(1,1600,t)
  mySer.SetPosTimeAndRun(2,1600,t)
  mySer.SetPosTimeAndRun(4,1300,t)
  mySer.SetPosTimeAndRun(5,2050,t)
  Pause 700
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
End Sub

Sub main6()
  mySer.SetPosTimeAndRun(0,2100,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1500,t)
  mySer.SetPosTimeAndRun(5,2100,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700

  mySer.SetPosTimeAndRun(4,1200,t)
```

```
  mySer.SetPosTimeAndRun(5,2100,t)
  Pause 700

  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(4,1380,t)
  mySer.SetPosTimeAndRun(5,2150,t)
  Pause 700

  mySer.SetPosTimeAndRun(1,1580,t)
  mySer.SetPosTimeAndRun(2,1580,t)
  Pause 700

  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
End Sub

Sub main7()
  mySer.SetPosTimeAndRun(0,1800,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1500,t)
  mySer.SetPosTimeAndRun(5,1500,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700

  mySer.SetPosTimeAndRun(1,1700,t)
  mySer.SetPosTimeAndRun(2,1700,t)
  mySer.SetPosTimeAndRun(4,1400,t)
  mySer.SetPosTimeAndRun(5,2000,t)
  Pause 700

  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1600,t)
  mySer.SetPosTimeAndRun(5,2100,t)
  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
```

```
  mySer.SetPosTimeAndRun(4,1400,t)
  mySer.SetPosTimeAndRun(5,2200,t)
  Pause 700

  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
End Sub

Sub main8()
  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1600,t)
  mySer.SetPosTimeAndRun(5,1500,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700

  mySer.SetPosTimeAndRun(4,1400,t)
  mySer.SetPosTimeAndRun(5,2200,t)
  Pause 700

  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(4,1550,t)
  mySer.SetPosTimeAndRun(5,2100,t)
  Pause 700

  mySer.SetPosTimeAndRun(1,1700,t)
  mySer.SetPosTimeAndRun(2,1700,t)
  mySer.SetPosTimeAndRun(4,1400,t)
  mySer.SetPosTimeAndRun(5,2000,t)
  Pause 700

  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
End Sub

Sub main9()
  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1380,t)
  mySer.SetPosTimeAndRun(5,2150,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
```

```
  mySer.SetPosTimeAndRun(1,1580,t)
  mySer.SetPosTimeAndRun(2,1580,t)
  Pause 700

  mySer.SetPosTimeAndRun(6,1550,t)
  Pause 700

  mySer.SetPosTimeAndRun(1,1400,t)
  mySer.SetPosTimeAndRun(2,1400,t)
  mySer.SetPosTimeAndRun(5,2100,t)
  Pause 700

  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1200,t)
  mySer.SetPosTimeAndRun(5,2100,t)
  Pause 700

  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
End Sub

Sub over()
  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1100,t)
  mySer.SetPosTimeAndRun(2,1100,t)
  mySer.SetPosTimeAndRun(4,1000,t)
  mySer.SetPosTimeAndRun(5,1600,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700
End Sub
```

Smart Peripherals

The Robotic Arm comes with the Servo Runner A to control the servos. You can add the following Smart Peripherals to interact with the Arm.

KeypadA


LCD2x16

For more information on how to use these modules, see the corresponding manual of the Smart peripherals.

Control with Keypad and LCD

Use the Keypad to control the Arm.

```
Peripheral mySer As ServoRunnerA @ 0
Peripheral myKeypad As KeypadA @ 3
Peripheral myLCD As LCD2x16a @ 4

Dim t,KeyID,c,i,j,p1,p2 ,p3 As Integer

Sub main()

  Call start()

  Do
    myKeypad.SetKeypadmode(0)
    Pause 100
    myKeypad.GetKeyID(KeyID)
    myKeypad.EnableKeyPressedEvent ()

    myLCD.Clear()
    myLCD.Display("Press 1 to3 or 0.")

    If (c = 1) Then
```

```
      myLCD.Clear()
      myLCD.Display("X-axle")
      Call main2()
    End If

    If (c = 2) Then
      myLCD.Clear()
      myLCD.Display("Y-axle ")
      Call main3()
    End If

    If (c = 3) Then
      myLCD.Clear()
      myLCD.Display("Z-axle ")
      Call main4()
    End If

    If (c = 13) Then
      myLCD.Clear()
      myLCD.Display("rest")
      Call over()
      Pause 500
    End If
  Loop
End Sub

Sub start()
  t=500
  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1500,t)
  mySer.SetPosTimeAndRun(2,1500,t)
  mySer.SetPosTimeAndRun(4,1500,t)
  mySer.SetPosTimeAndRun(5,1500,t)
  mySer.SetPosTimeAndRun(6,1650,t)
  Pause 2000
  c=0
End Sub

Sub main2()
  t=1000
  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1900,t)
  mySer.SetPosTimeAndRun(2,1900,t)
  mySer.SetPosTimeAndRun(4,1100,t)
  mySer.SetPosTimeAndRun(5,750,t)
  mySer.SetPosTimeAndRun(6,1650,t)
  Pause 1500
```

```
  p1=1150 : j=1650
  p2=750
  t=500

  For i=0 To 7
    mySer.SetPosTimeAndRun(0,j,t)
    mySer.SetPosTimeAndRun(4,p1,t)
    mySer.SetPosTimeAndRun(5,p2,t)
    Pause 500
    j-=25
    p1-=11
    p2-=9
    High 0: High 1: Pause 10
    Low 0: Low 1 : Pause 10
  Next

  j+=25
  For i=0 To 7
   p1+=11
   p2+=9
   mySer.SetPosTimeAndRun(0,j,t)
   mySer.SetPosTimeAndRun(4,p1,t)
   mySer.SetPosTimeAndRun(5,p2,t)
   Pause 500

   High 0: High 1: Pause 10
   Low 0: Low 1 : Pause 10
   j-=25
  Next

  j+=25
  mySer.SetPosTimeAndRun(0,j,t)
  Pause 500
  Call start()
End Sub

Sub main3()
  t=1000
  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1900,t)
  mySer.SetPosTimeAndRun(2,1900,t)
  mySer.SetPosTimeAndRun(4,1100,t)
  mySer.SetPosTimeAndRun(5,750,t)
  mySer.SetPosTimeAndRun(6,1650,t)
  Pause 1500
```

```
  p1=1100
  p2=750
  p3=1900
  t=300

  For i= 0  To 11
    mySer.SetPosTimeAndRun(0,1500,t)
    mySer.SetPosTimeAndRun(1,p3,t)
    mySer.SetPosTimeAndRun(2,p3,t)
    mySer.SetPosTimeAndRun(4,p1,t)
    mySer.SetPosTimeAndRun(5,p2,t)
    Pause 300
    p1+=55
    p2+=25
    p3+=25
    High 0 : High 1 : Pause 10
    Low 0 : Low 1 : Pause 10
  Next
  Call start()
End Sub

Sub main4()
  t=1000
  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1900,t)
  mySer.SetPosTimeAndRun(2,1900,t)
  mySer.SetPosTimeAndRun(4,1100,t)
  mySer.SetPosTimeAndRun(5,750,t)
  mySer.SetPosTimeAndRun(6,1650,t)
  Pause t

  p1=1100
  p2=750
  p3=1900
  t=100

  For i= 0  To  14
    p1+=33
    p2+=47
    p3-=27
    mySer.SetPosTimeAndRun(0,1500,t)
    mySer.SetPosTimeAndRun(1,p3,t)
    mySer.SetPosTimeAndRun(2,p3,t)
    mySer.SetPosTimeAndRun(4,p1,t)
    mySer.SetPosTimeAndRun(5,p2,t)
    Pause t
    High 0 : High 1 : Pause 10
```

```
    Low 0 : Low 1 : Pause 10
  Next
  Call start()
End Sub

Sub over()
  t=500

  mySer.SetPosTimeAndRun(0,1500,t)
  mySer.SetPosTimeAndRun(1,1100,t)
  mySer.SetPosTimeAndRun(2,1100,t)
  mySer.SetPosTimeAndRun(4,1000,t)
  mySer.SetPosTimeAndRun(5,1500,t)
  mySer.SetPosTimeAndRun(6,2000,t)
  Pause 700

End Sub

Event myKeypad.KeyPressedEvent ()
  myKeypad.GetKeyID(KeyID)

  Select Case KeyID
    Case 0 :  c = 1
    Case 1 :  c = 2
    Case 2 :  c = 3
    Case 13 : c = 13
  End Select
End Event
```