EPSON FLEXIBOWL PLUGIN

This Plugin was developed with the idea of communicating quickly and safely with FlexiBowl® through Epson robots by using instructions in RC+.
The Plugin does not require any additional licence

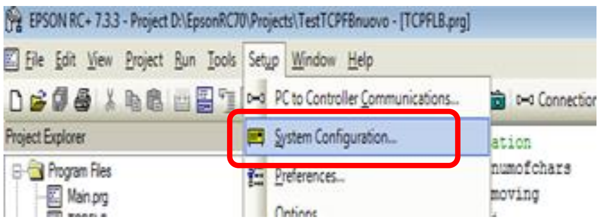
FlexiBowl®

EPSON SEIKO EPSON CORPORATION

EPSON RC+ 7.0

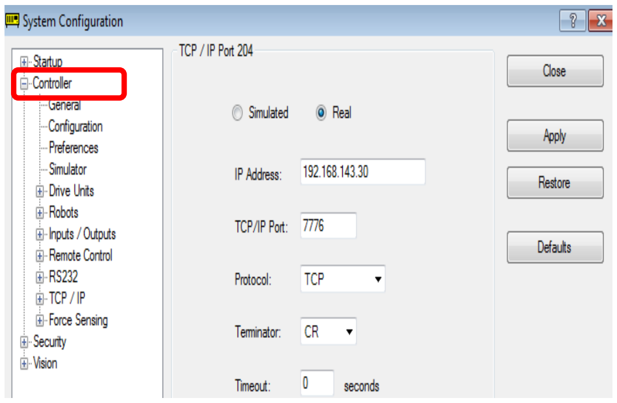
Version: 7.4.2

STEP 1:


EPSON RC+ 7.3.3 - Project D:\EpsonRC70\Projects\TestTCPFBnuovo - [TCPFLB.prg]
File Edit View Project Run Tools Setup Window Help
PC to Controller Communications...
System Configuration...
Preferences...
Options
Project Explorer
Program Files
Main.prg
Connection

In the Rc+ menu select:
Setup→ System Configuration

STEP 2:



TCP / IP Port 204

Simulated [✓ Real]

IP Address: 192.168.143.30

TCP/IP Port: 7776

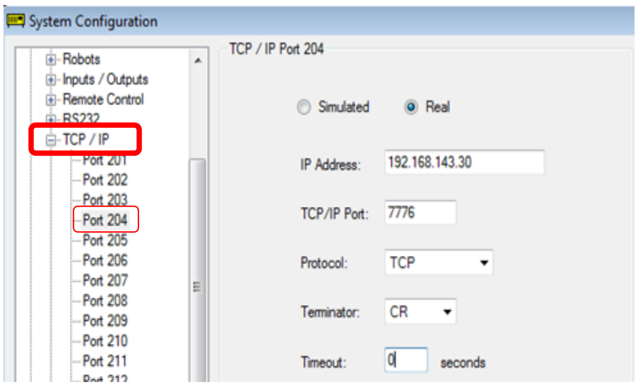
Protocol: TCP

Terminator: CR

Timeout: 0 seconds

In the dialogue window that will appear, select ***Controller***

STEP 3:



TCP / IP Port 204

Simulated [✓ Real]

IP Address: 192.168.143.30

TCP/IP Port: 7776

Protocol: TCP

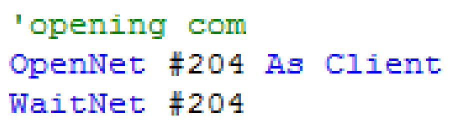
Terminator: CR

Timeout: 0 seconds

Then select the menu relating to ***TCP/IP*** communication.
Once the communication port has been selected, it just needs to be configured by following the standard below

- ***IP:*** IP of the flexibowl
- ***Tcp/ip port:*** 7776(standard)
- ***Protocol:*** TCP
- ***Terminator:*** CR

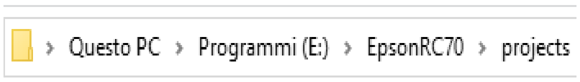
STEP 4:



```
'opening com
OpenNet #204 As Client
WaitNet #204
```

By default the Epson Plug-in uses port ***TCP/IP*** 204. Should you wish to change it, edit the part of code where the port to be used is selected.

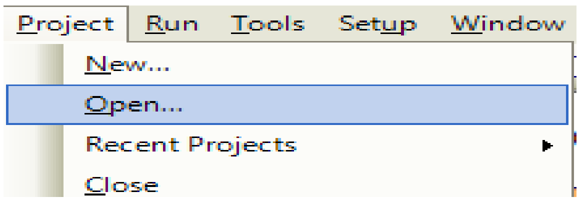
STEP 5:



Questo PC › Programmi (E:) › EpsonRC70 › projects

Copy the Plug-in provided by ARS to the ***projects*** folder of the ***RC+*** program

STEP 6:



Open the Project of the Plug-in previously imported.

STEP 7:

```
Call TCPFLB(Command$)
```

Just enter this line of code into our main program to be able to control all the functions of the **FlexiBowl®**.
All we need to do is enter the command to run as argument.

COMMAND STRING FORMAT:

Correct syntax for each packet			
Header		**Command**	**Footer**
Chr(0)	Chr(7)	Command	Chr(13)

COMMAND LIST:

Commands	Description
QX2	Move
QX3	Move-Flip
QX4	Move-Flip-Blow
QX5	Move-Blow
QX6	Shake
QX7	Light on
QX8	Light off
QX9	Blow
QX10	Flip
QX11	Quick Emptying Option
QX12	Reset Alarm

SCRIPT:

```
'varible declaration
Global Integer numofchars
Global Integer moving
Global Integer i
Global Integer pos
Global Double timeblow
Global String appoggio$
Global String appoggiomoving$
Global String data_tmp_Str$(0)
Global Byte inputbin(7)
Global Byte commandarray(5)
Global Byte movingbit(0)
Global Byte inputbyte
Global String messagearray$(0)

Function TCPFLB(message$ As String)
' Uncomment next line for debug mode
' #define DEBUG 1

'manual test command  ONLY IN DEBUG MODE
#ifdef DEBUG
Print "input command"
Line Input message$ 'Read one line input data into message
#endif

'clear vars
Redim commandarray(5)

'header
commandarray(0) = 0
commandarray(1) = 7

'command BODY
commandarray(2) = Asc("Q")
commandarray(3) = Asc("X")

'close communication char
commandarray(5) = 13

'opening com
OpenNet #204 As Client
WaitNet #204

message$ = LCase$(message$)
 'select the correct type of movement according to the input
    Select message$
```

SCRIPT:

```
'*******************************************MOVE***************************
    Case "move"

    commandarray(4) = Asc("2")

    #ifdef DEBUG
    Print "message =", message$
    Print "case = move"
    For i = 0 To UBound(commandarray)
    Print "pos", i, "=", commandarray(i)
    Next
    i = 0
    #endif

    WriteBin #204, commandarray(), 6
    Call waitmoving

'*******************************************MOVE FLIP***********************

    Case "move flip"

    commandarray(4) = Asc("3")

    #ifdef DEBUG
    Print "message =", message$
    Print "case = move flip"
    For i = 0 To UBound(commandarray)
    Print "pos", i, "=", commandarray(i)
    Next
    i = 0
    #endif

    WriteBin #204, commandarray(), 6
    Call waitmoving
'******************************************MOVE FLIP BLOW*****************

    Case "move blow flip"

    commandarray(4) = Asc("4")

    #ifdef DEBUG
    Print "message =", message$
    Print "case = move blow flip"
    For i = 0 To UBound(commandarray)
    Print "pos", i, "=", commandarray(i)
    Next
    i = 0
    #endif

    WriteBin #204, commandarray(), 6
    Call waitmoving

```

```
'*******************************************MOVE BLOW**************************
      Case "move blow"

          commandarray(4) = Asc("5")

          #ifdef DEBUG
          Print "message =", message$
          Print "case = move blow"
          For i = 0 To UBound(commandarray)
          Print "pos", i, "=", commandarray(i)
          Next
          i = 0
          #endif

          WriteBin #204, commandarray(), 6
          Call waitmoving
    '********************************************SHAKE*************************

          Case "shake"

          commandarray(4) = Asc("6")

          #ifdef DEBUG
          Print "message =", message$
          Print "case = shake"
          For i = 0 To UBound(commandarray)
          Print "pos", i, "=", commandarray(i)
          Next
          i = 0
          #endif

          WriteBin #204, commandarray(), 6
          Call waitmoving
    '********************************************LIGHTON***********************
          Case "lighton"
          commandarray(4) = Asc("7")

          #ifdef DEBUG
          Print "message =", message$
          Print "case = shake"
          For i = 0 To UBound(commandarray)
          Print "pos", i, "=", commandarray(i)
          Next
          i = 0
          #endif

          WriteBin #204, commandarray(), 6


```

SCRIPT:

```
'********************************************LIGHTOFF***********************
*
      Case "lightoff"

      commandarray(4) = Asc("8")

      #ifdef DEBUG
      Print "message =", message$
      Print "case = shake"
      For i = 0 To UBound(commandarray)
      Print "pos", i, "=", commandarray(i)
      Next
      i = 0
      #endif

      WriteBin #204, commandarray(), 6

'*************************************************************************

      Default
'********************************************BLOW**************************

         If Not InStr(message$, "blow") Then
        'reallocate the command array
         Redim commandarray(7)
        'prepare the blowing request
         commandarray(0) = 0
         commandarray(1) = 7
         commandarray(2) = Asc("S")
         commandarray(3) = Asc("O")
         commandarray(4) = Asc("3")
         commandarray(6) = 13
         'extract the time for the blow
         pos = InStr(message$, "_")
         appoggio$ = Right$(message$, Len(message$) - pos)
         timeblow = Val(appoggio$)


         Print "blowing"
         commandarray(5) = Asc("L") 'SO3L turn on blow
         WriteBin #204, commandarray(), 6

         Wait timeblow

         commandarray(5) = Asc("H") 'SO3H turn off blow
         WriteBin #204, commandarray(), 6

         Print "stop blowing"
```

Epson RC+

SCRIPT:

```
'*******************************************GENERIC COMMAND***************

        Else
       ' to upper message
        message$ = UCase$(message$)
' re allocate the dimension of the array according to the input string
        Redim commandarray(Len(message$) + 2)
        Redim messagearray$(Len(message$) - 1)
        commandarray(0) = 0
        commandarray(1) = 7
        appoggio$ = message$
       'split the input message in a chr array
        For i = 0 To (Len(message$) - 1)
           messagearray$(i) = Left$(appoggio$, 1)
           appoggio$ = Right$(appoggio$, Len(appoggio$) - 1)
        Next
       'format the commandarray
        For i = 0 To UBound(messagearray$)

           commandarray(i + 2) = (Asc(messagearray$(i)))

        Next

        commandarray(Len(message$) + 2) = 13


        'send the command array
        WriteBin #204, commandarray(), UBound(commandarray) + 1


        Call waitmoving


        EndIf
'************************************************************************
```

SCRIPT:

```
Send

CloseNet #204

Fend
Function waitmoving
Print "Start..."
'clear the commandarray
Redim commandarray(4)

'prepare the moving? request
commandarray(0) = 0
commandarray(1) = 7
commandarray(2) = Asc("S")
commandarray(3) = Asc("C")
commandarray(4) = 13

'wait the start of the movement
moving = 0
i = 0
numofchars = ChkNet(204)
If (numofchars >= 0) Then
ReadBin #204, inputbyte
EndIf
'loop until the moving bit turn at 0 (chr48=0)
Do While (moving <> 48)

    WriteBin #204, commandarray(), 5
    numofchars = ChkNet(204)
    If (numofchars >= 5) Then
    Redim inputbin(0)
    Redim inputbin(numofchars)
    ReadBin #204, inputbyte
    ReadBin #204, inputbin(), numofchars
    Redim movingbit(UBound(inputbin))
    For i = 0 To UBound(inputbin)
    If (inputbin(i) = 13 Or inputbin(i) = 37) Then
    i = UBound(inputbin)
    Else

    movingbit(i) = inputbin(i)

    EndIf
    Next
    moving = movingbit(5)
    Wait 0.01

    EndIf

Loop

Print "Done..."

Fend
```