

Manual



FlexiBowl®

ABB Plug-In

INDICE

1. Installazione Plug-In
2. Lista comandi FlexiBowl
3. Script

Questo Plugin è nato con l'idea di comunicare in maniera **rapida e sicura con il FlexiBowl®** tramite i robot **ABB**, mediante l' utilizzo di istruzioni in linguaggio **RAPID**.

Il Plugin necessita di una licenza aggiuntiva per la gestione dei socket:

-Pc interface.

Può essere utilizzata anche una licenza opzionale:

-Multi tasking

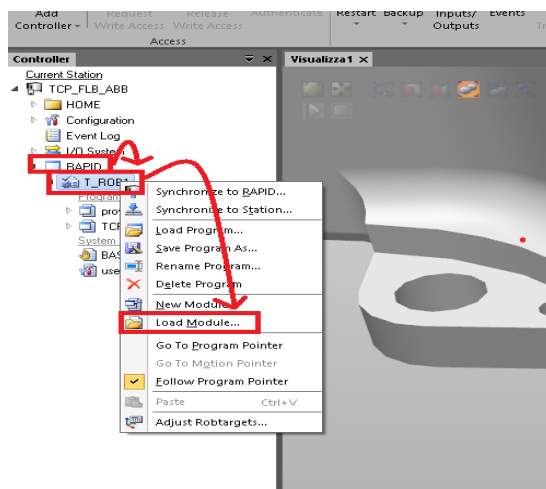
Per poter effettuare il controllo e l' attivazione della tramoggia in un task parallelo senza dover bloccare il ciclo principale del robot.

FlexiBowl® Plug-In



Installazione Plug-In

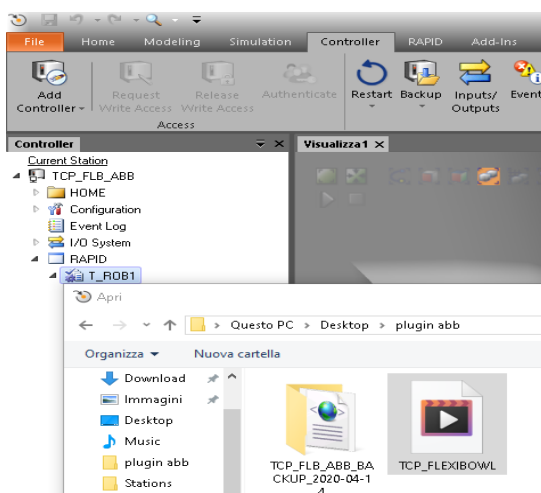
Step 1.



Nel menù **Controller** selezionare:
Rapid → **T_ROB1**

Premere con il tasto destro su
T_ROB1 e selezionare **Load Module**

Step 2.



Nella finestra di dialogo
che apparirà ,
selezionare il PlugIn
TCP_FLEXIBOWL fornito
da ARS

Step 3.

```
4
5 var string returnflb;
6 returnflb:= TCP_FLB("192.168.0.161","QX2
7
```

Dopo aver importato il PlugIn
per il FlexiBowl,
semplicemente
aggiungendo queste due
righe di codice nel nostro
main saremo in grado di
controllare tutte le funzioni.

Installazione Plug-In

Step 4.

```
5 | var string returnflb;  
6 | returnflb:= TCP_FLB("192.168.0.161", "QX2")
```

Ora bisogna semplicemente assegnare alla funzione **TCP_FLB** "IP" e "**COMANDO**"

Step 5.

```
returnflb:= TCP_FLB
```

La funzione **TCP_FLB**, fornirà in output una stringa contenente :

- "**done**" se il comando inviato era un comando di movimento.
- "**|00" "|07" "risposta dal FlexiBowl" "|0D"**" se viene inviato un comando di interrogazione al driver

Es.

Comando inviato= "\ 00"
"\ 07" "SC" "\ 0D"

Risposta dal FlexiBowl= -"\ 00"
"\ 07" "SC=0001" "\ 0D"

Lista Comandi FlexiBowl

Lista dei comandi e descrizioni da inviare al FlexiBowl:

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

Action	Description
MOVE	Moves the feeder the current parameters.
MOVE-FLIP	Moves the feeder and activates Flip simultaneously
MOVE-BLOW- FLIP	Moves the feeder and activates Flip and blow simultaneously
MOVE-BLOW	Moves the feeder and activates Flip simultaneously
SHAKE	Shakes the feeder with the current parameters
LIGHT ON	Light on
LIGHT OFF	Light off
FLIP	Flip
BLOW	Blow
QUICK_EMPTYING	Quick Emptying Option
RESET_ALARM	Reset Alarm and enable the motor

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

Script

Modulo "TCP_FLEXIBOWL".

```

MODULE TCP_FLEXIBOWL
!Dichiarazione variabili globali
!Global variables declaration VAR string
Return_From_FLB; VAR socketdev
client_socket; VAR byte rispByteArray{50};

func string TCP_FLB(string ip,string command)
!Dichiarazione variabili locali VAR string
TCP_IP;
VAR num TCP_port;

!creazione socket SocketCreate
client_socket;
!define ip and port (standard tcp port= 7776 /standard udp port=7775) TCP_IP:=ip;
TCP_port:=7776;
Return_From_FLB:="";
!Convert all Lower Char in Upper Char
command := StrMap(command,STR_LOWER, STR_UPPER);
!call connection proc Connection_FLB
TCP_IP,TCP_port;
!call sending formatted command proc Send_FLB
command;
!If was a movement command (QX...) will wait untill the end of the movement IF (
StrFind(Return_From_FLB,1,"%")<= strlen(Return_From_FLB)) THEN
!aspetto che il movimento sia finito e inserisco "done"come valore di ritorno IF (
StrFind(command,1,"Q")<= strlen(Return_From_FLB)) THEN
Wait_No_Move;
ENDIF ENDIF
!Return "done" if the command was a movement or return the
!answare from FlexiBowl if was an instruction
RETURN Return_From_FLB; endfunc

|*****
*****
*****
!handle the connection and catch rising exceptions PROC
Connection_FLB(string ip,num port)
SocketConnect client_socket, ip, port;
ERROR
!If an error occurs during the connection close the socket and write on the tp SocketClose
client_socket;
TPWrite("Connection Problems");

ENDPROC

```

Script

```

!format and send the command string to flb  PROC Send_FLB(string command)
    Return_From_FLB:="";
    !format the message adding the header (chr0 chr7) and the end of line character (chr13)
    !and read the response from Flexibowl
    SocketSend client_socket \Str := "\00\07"+command+"\0D";
    SocketReceive client_socket\Data:=rispByteArray;
    Return_From_FLB:=Get_String_Answer(rispByteArray);
ERROR
    !If an error occurs during the connection close the socket and write on the tp
    SocketClose client_socket;
    TPWrite("Connection Problems"); ENDPROC

FUNC string Get_String_Answer(byte byteArray{*})  VAR string sChar;
    VAR bool bLoop:=TRUE;  VAR num ii;
    ii:=3;
    WHILE bLoop DO  sChar:=ByteToStr(rispByteArray{ii}\Char);

        IF sChar<>"\0D" THEN
            Incr ii; Return_From_FLB:=Return_From_FLB+sChar;
        ELSE
            bLoop:=FALSE;  ENDIF
    ENDWHILE

    RETURN Return_From_FLB; ENDFUNC

!aspetto la fine del movimento
!wait untill the movement finish
PROC Wait_No_Move()
    VAR string bit_s;
    VAR string bit_i;
    VAR bool moving:=TRUE;  VAR num len;

    WHILE moving DO
        !send the "status controll" command SC  Send_FLB "SC";

        !la risposta sarà del tipo : (chr0)(chr7)SC=abcd(chr13) andremo quindi ad controllare
        !il bit "c" per sapere se il movimento è terminato
        !the standard answare will be like : (chr0)(chr7)SC=abcd(chr13) we will controll
        !the "c" bit to know if the movement is terminated
        !Return_From_FLB:="";
        bit_s:=StrPart(Return_From_FLB,6,1);

        Send_FLB "IO"; len:=StrLen(Return_From_FLB);
        bit_i:=StrPart(Return_From_FLB,len,1);

        IF (bit_i="1" and bit_s="0") THEN moving:=FALSE;
        ENDIF
    ENDWHILE
    SocketClose client_socket;
    Return_From_FLB:="done";
    RETURN;
ENDPROC

ENDMODULE

```

Script

Modulo "F_PLUGIN".

Di seguito viene riportato un semplice esempio di utilizzo del modulo di comunicazione con il Flexibowl, che è contenuto all'interno del modulo "F_plugin".

```
MODULE F_plugin
  PERS robtarget place:=[[324.27,-107.97,225.31],[0.000552446,0.00305038,-0.999985,-
0.00448437],[-1,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,0.0300813]];
  PERS robtarget place1:=[[324.27,-107.97,320.31],[0.000552446,0.00305038,-0.999985,-
0.00448437],[-1,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,0.0300813]];
  VAR string ritorno;
  PROC main()
    pick_place;
  ENDPROC

  PROC pick_place()
    WHILE TRUE DO
      movel place,vmax,fine,tool0 \WObj:=wobj0;
      ritorno:=TCP_FLB ("192.168.125.23","QX2");
      movel place1,vmax,fine,tool0 \WObj:=wobj0;
      ritorno:=TCP_FLB ("192.168.125.23","QX6");
    ENDWHILE
  Endproc
ENDMODULE
```