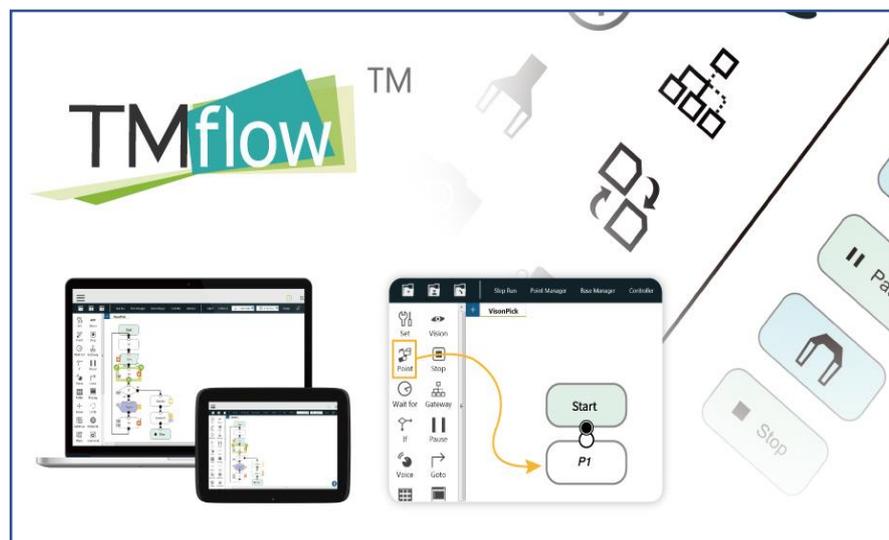


# TM FLEXIBOWL PLUGIN



**This Plugin was developed with the idea of communicating quickly and safely with the flexibowl through TM robots, using TMFLOW 1.76 or later software**  
**The Plugin does not require any license to operate**

# FlexiBowl®



The Plugin is in the USB\TMROBOT folder, formed by two global variables and a Flexibowl Plugin program for Flexibowl movement. From here, the Plugin can be imported or created from scratch with the documents we will provide.

We will describe the operation of the Plugin and the creation process below.

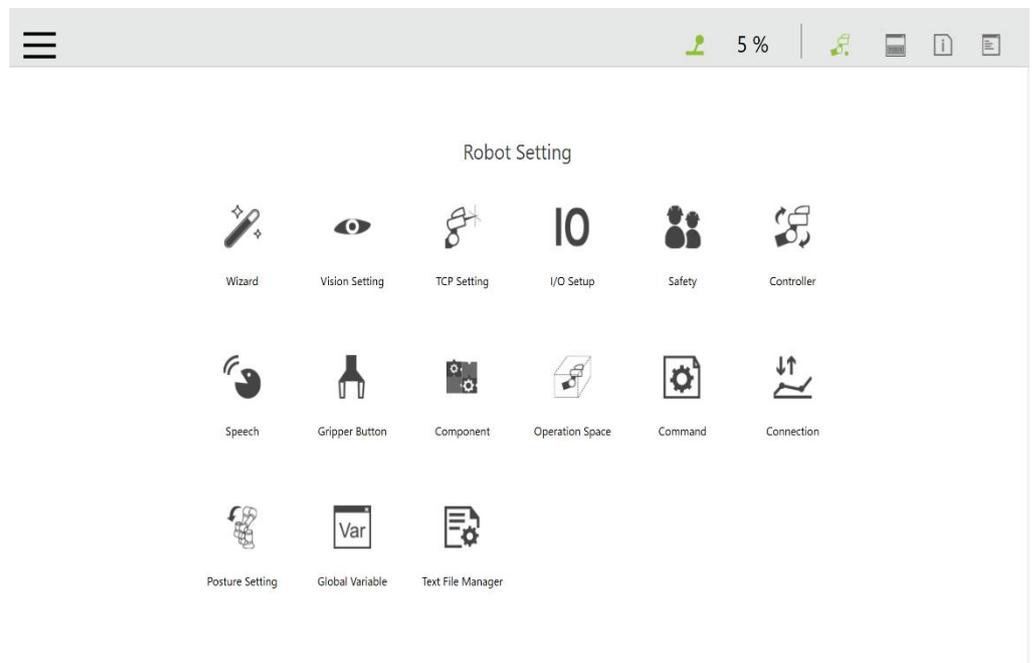
STEP 1:

Creation of two global variables for operation.

We will create two global variables

- 1) *Send\_Command* to send the command to the Flexibowl
- 2) *Return\_Flb* to receive the return string from the Flexibowl

Go to the Robot Setting → Global Variable by means of the TmFlow menu  
And create two string variables as shown in the image



**Global Variable Setting**

Variable	Initial Value
string g_SendCommand = "QX2"	Type: int
string g_Return_Flb = "Done"	Name: <input type="text"/>
	Value: <input type="text"/>

Buttons: Save, Add

STEP 2:

Creation of a simple program to handle the robot and activate the Flexibowl.  
Two local support variables will be created here

- 1) BYTE [] ARRAY *Byte\_To\_Send* from 15 positions, to send the command
- 2)String *Local\_Return\_Flb*

The screenshot shows a subflow named 'Flexibowl\_Plugin' with the following steps:

- Start
- MODE F1
- SET COMMAND TO SEND FLB
- SEND THE COMMAND
- DISPLAY THE RETURN\_FILE STRING
- MODE F2
- SET THE COMMAND TO SEND FLB
- SEND THE COMMAND

**Variables**

Buttons: Create Variable, Create Array

byte[]	var_Byte_To_Send	=	{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}
string	var_Local_Return_Flb	=	empty

Let's see how to set the Flexibowl command variable for movement

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

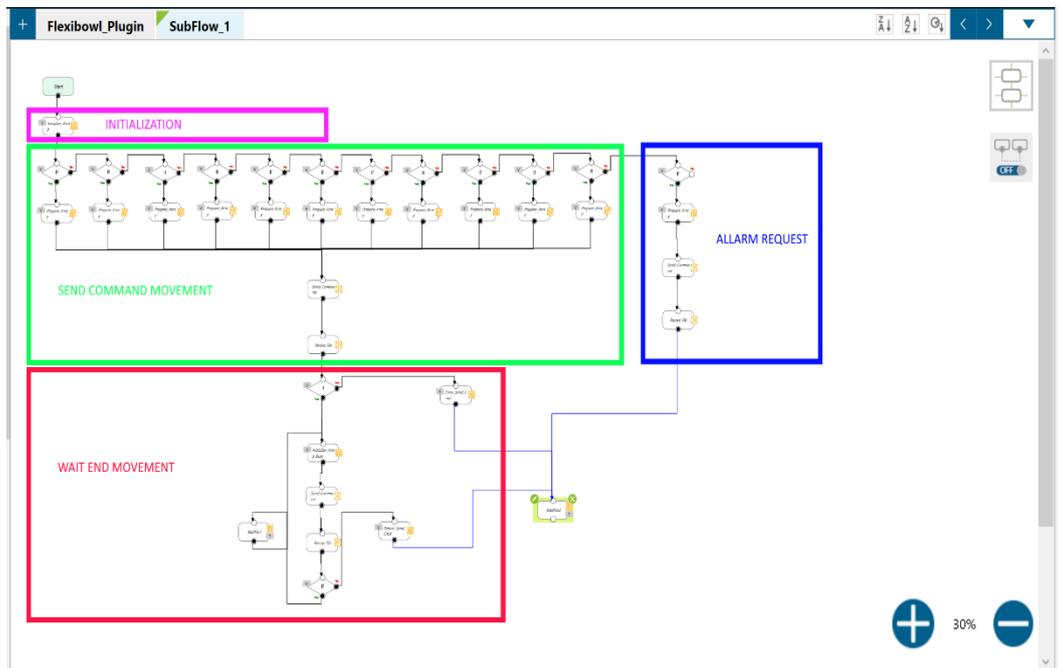
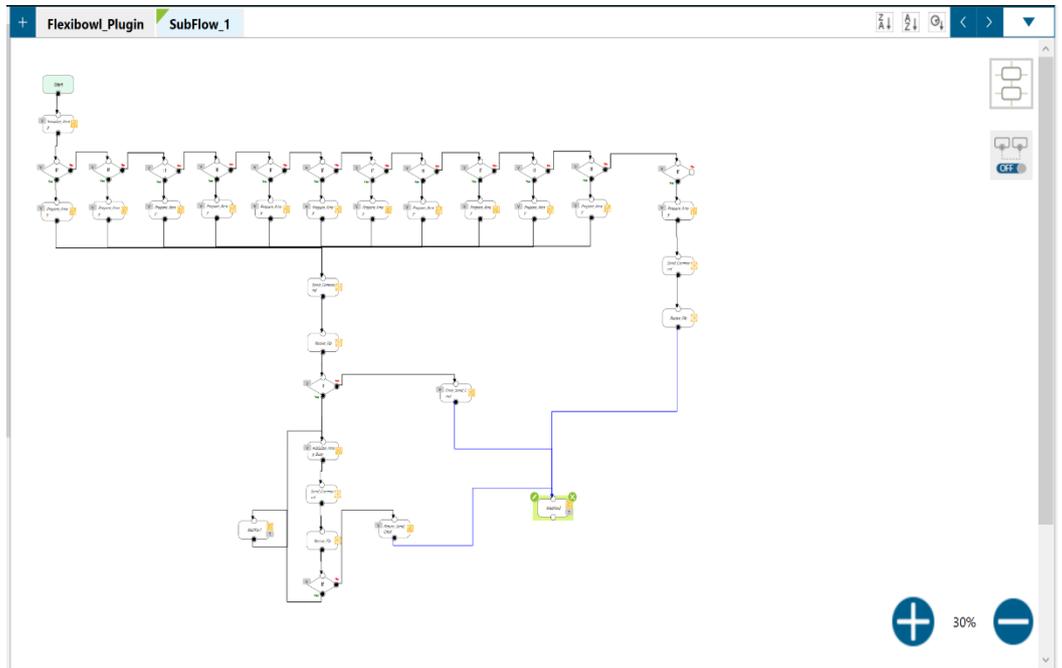
Action	Description
MOVE	Moves the feeder the current parameters.
MOVE-FLIP	Moves the feeder and activates Flip simultaneously
MOVE-BLOW-FLIP	Moves the feeder and activates Flip and blow simultaneously
MOVE-BLOW	Moves the feeder and activates Flip simultaneously
SHAKE	Shakes the feeder with the current 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

# TM FLOW



## STEP 3:

Now we will analyse the SUBFLOW for Flexibowl movement



STEP 4:

**Initialisation:**

The two local variables, the byte array and the Flb response will always be initialised here



**IMPORTANT!**  
The default IP addresses is 192.168.1.10 in class B (Subnet Mask 255.255.0.0). The TCP/IP port is 7776 and the UDP port is 7775.

The correct syntax for each pack is:

Header	Description	Footer
Chr(0)	Chr(7) Command (ASCII character vector)	Chr(13)

For each string sent to the Flexibowl®, an ECHO of the command received will be returned. If the string is interpreted correctly, the ECHO will be:

Header	ECHO	Footer
Chr(0)	Chr(7) %	Chr(13)

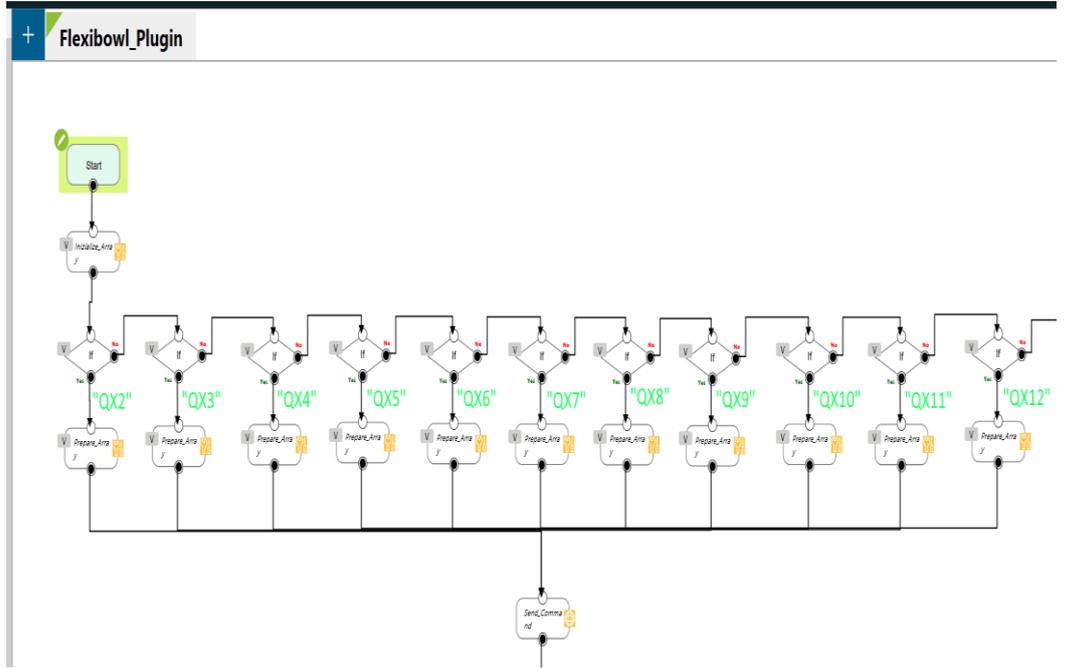
Otherwise if the string is not interpreted correctly, the ECHO will be:

Header	ECHO	Footer
Chr(0)	Chr(7) ?	Chr(13)

STEP 5:

**Send command movement:**

The array to be sent to the FIB is prepared here according to the global variable where movement is set.



**IF** ✕

**Node Name**

---

**Digital I/O**  >

---

**Variables**  >

---

**Analog I/O**  >

---

**Stop Criteria**  >

---

**Variables Setting**

---

**Variables Judge Rule**

All
  One

---

>

---

⬇ ⬆

g\_SendCommand == "QX2"

---

**Set** ✕

**Node Name**

---

**Digital I/O**

---

**Variables**

---

**Analog I/O**

---

← **Expression Editor Setting**

=

---

**Network** ✕

**Node Name**

---

**Choose Device**

---

Receive to Variable  Send

Typing

---

Variable

---

Wait Time  ms

Send Status

---

← **Add/Modify Device**

Device Name

IP

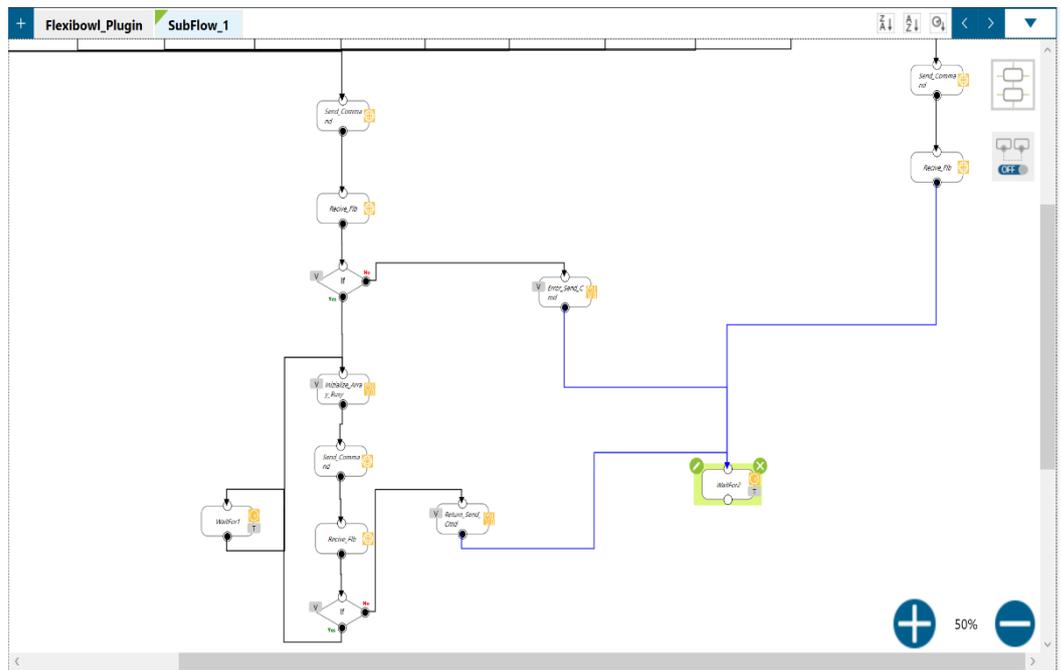
Port

---

STEP 4:

**Wait End Movement:**

Once the Flb movement command is sent, we will wait for the movement to stop in this small sub-program. To do so the "IO" request must be sent and the Loop response must be analysed



**Network**

**Node Name** Recive\_Flb

**Choose Device** Flexibowl

Receive to Variable  Send

Variable var\_Local\_Return\_Flb

Maximum received data time 50 ms

Wait Time ms

Connection Status(bool)

OK Delete this node

**IF**

**Node Name** IF

**Digital I/O** IO(0)

**Variables** Variables(1)

**Analog I/O** AIO(0)

**Stop Criteria** Stop Criteria(0)

OK Delete this node

**Variables Setting**

**Variables Judge Rule**

All  One

>

**Add**

---

String\_Substring  
(var\_Local\_Return\_Fl  
b\_Length == "%"  
(var\_Local\_Return\_Fl  
b)-2,1)

**OK**

**Set**

**Node Name** Initialize\_Array\_Busy

**Digital I/O** IO(0)

**Variables** Variables(16)

**Analog I/O** AIO(0)

**OK** **Delete this node**

If the % symbol is not returned in the previous If, the FLB has not interpreted the correct command, this means we have encountered an error and will write Return\_Flb = "Error Send Command" on the global variable.

Send the IO command to understand if the Flb is moving.

**Expression Editor Setting**

=

**Add**

---

byte[] var\_Byte\_To\_Send[0] = 0

byte[] var\_Byte\_To\_Send[1] = 7

byte[] var\_Byte\_To\_Send[2] = 73

byte[] var\_Byte\_To\_Send[3] = 79

byte[] var\_Byte\_To\_Send[4] = 13

byte[] var\_Byte\_To\_Send[5] = 0

**OK**

**Network**

**Node Name** Send\_Command

**Choose Device** Flexibowl

**Add Device** **Edit Device**

Receive to Variable  Send

Typing

Variable **var\_Byte\_To\_Send**

**Wait Time**  ms **Text**

**Send Status**

**OK** **Delete this node**

✕
Network

**Node Name**

**Choose Device** Flexibowl >
 

Add Device

Edit Device

Receive to Variable     Send
 

Variable

var\_Local\_Return\_Flb >

Maximum received data time  ms

Wait Time  ms Text

Connection Status(bool)  >

OK
Delete this node

✕
IF

**Node Name**

**Digital I/O** IO(0) >

**Variables** Variables(1) >

**Analog I/O** AIO(0) >

**Stop Criteria** Stop Criteria(0) >

OK
Delete this node

←
Variables Setting

**Variables Judge Rule**
 All     One

> ▼ 

Add

▼ ▲

String\_Substring  
 (var\_Local\_Return\_Fl  
 b.Length != "1"  
 (var\_Local\_Return\_Fl  
 b)-2,1)

OK

←
Variables Setting

**Variables Judge Rule**
 All     One

> ▼ 

Add

▼ ▲

String\_Substring  
 (var\_Local\_Return\_Fl  
 b.Length != "1"  
 (var\_Local\_Return\_Fl  
 b)-2,1)

OK

If SubString == 1 the movement is finished, if <> gives 1 the flexibowl is still moving, therefore we will take a small pause (50ms) and interrogate the Flb once again.

**WaitFor** [Close]

**Node Name** WaitFor1

All  One

**Digital I/O** DIO(0) [>]

**Time** Time(50 ms) [>]

**Variables** Variables(0) [>]

**Analog I/O** AIO(0) [>]

**Stop Criteria** Stop Criteria(0) [>]

OK Delete this node

Once the movement is finished we will write "Done" on the Return\_Flb global variable

**Set** [Close]

**Node Name** Return\_Send\_Cmd

**Digital I/O** IO(0) [>]

**Variables** Variables(1) [>]

**Analog I/O** AIO(0) [>]

OK Delete this node

**Expression Editor Setting** [Back]

[ ] = [ ]

Add

string g\_Return\_Flb = "Done"

OK

STEP 5:

**Alarm Request:**

If we send the “AL” command to the Flb it will tell us if there are active alarms, refer to the Flb manual for details.

**IF** [Close]

**Node Name**

**Digital I/O**

**Variables**

**Analog I/O**

**Stop Criteria**

**Variables Setting**

**Variables Judge Rule**

All  One

>

g\_SendCommand == "AL"

**Set** [Close]

**Node Name**

**Digital I/O**

**Variables**

**Analog I/O**

**Expression Editor Setting**

=

byte[] var\_Byte\_To\_Send[2] = 65

byte[] var\_Byte\_To\_Send[3] = 76

byte[] var\_Byte\_To\_Send[4] = 13

**Network** ✕

**Node Name**

**Choose Device**  >

Receive to Variable  Send

Typing ✎

Variable  >

Wait Time  ms

Send Status  >

**Network** ✕

**Node Name**

**Choose Device**  >

Receive to Variable  Send

Variable  >

Maximum received data time  ms

Wait Time  ms

Connection Status(bool)  >

