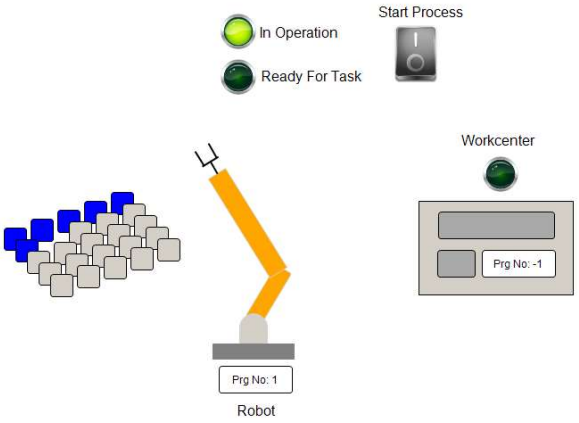
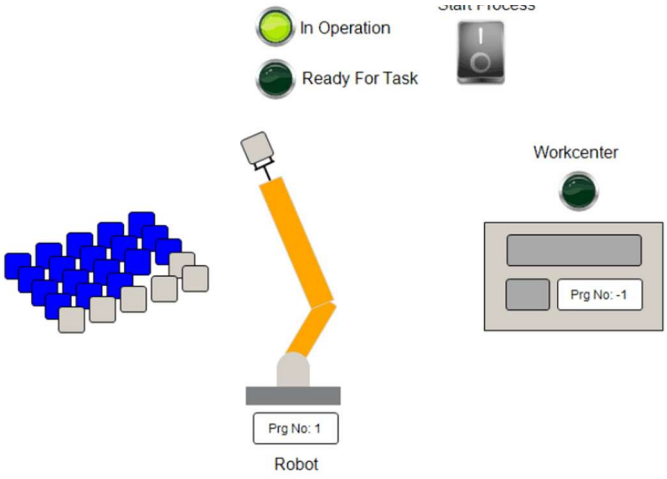
Robot Workcenter Control



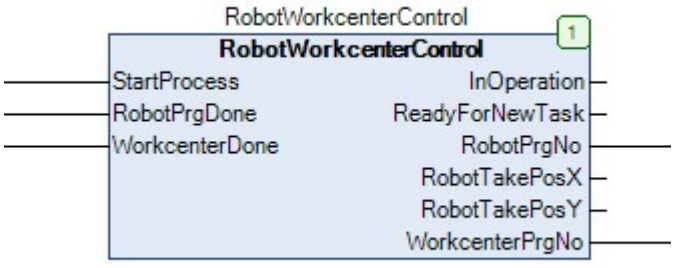
Process machine consists of a table with 25 items, a robot and workcenter.

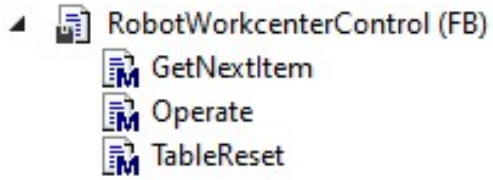
Operating mode:

- The robot moves an item from the table to the workcenter
- Workcenter processes the item
- The robot moves the item back to the table
- When all 25 items were processed, the job is done.




```
TYPE ItemState :
STRUCT
      InProcess    :     BOOL; // item picked-up by robot & processing by workcenter
      Done         :     BOOL; // item processing completed
END_STRUCT
END_TYPE
```





```
FUNCTION_BLOCK RobotWorkcenterControl

VAR_INPUT
	StartProcess		:	BOOL;		// machine start
	RobotPrgDone		:	BOOL;		// input from robot  - program (1 or 2) done
	WorkcenterDone		:	BOOL;		// input from workcenter - program 1 done
END_VAR
VAR_OUTPUT
	InOperation		:	BOOL;		// indication that machine in operation mode
	ReadyForNewTask		:	BOOL;		// indication that machine ready for 25 new items

	RobotPrgNo		:	INT:=-1;	// call robot program number
	RobotTakePosX :	INT;			// pick up position for robot
	RobotTakePosY :	INT;			// pick up position for robot

	WorkcenterPrgNo	: INT:=-1;	// call workcenter program number (process the item)
END_VAR
VAR CONSTANT
	TableX			:	INT:=5;		// number of items in X direction
	TableY			:	INT:=5;		// number of items in Y direction
END_VAR
VAR
	// table with items
	ItemsTable		:	ARRAY[1..TableX, 1..TableY] OF ItemState;
	RunAll			:	BOOL;		// run for all items
	RunOne			:	BOOL;		// run for one item
END_VAR

IF StartProcess AND ReadyForNewTask THEN
	TableReset();
	ReadyForNewTask := FALSE;			// turn off the task done
	InOperation := TRUE;				// turn on operating mode
	RunAll:=TRUE;
	RunOne:=FALSE;
END_IF

// take next item from the table
IF RunAll AND NOT RunOne THEN
	GetNextItem();
END_IF

// If no more items, then stop
IF RobotTakePosX=0 AND RobotTakePosY=0 THEN
	RunAll:=FALSE;
	ReadyForNewTask:=TRUE;		// operation completed
	InOperation:=FALSE;		// no operating mode
END_IF

// one item is in operating mode
IF RunAll THEN
	Operate();
END_IF
```

```
METHOD GetNextItem : BOOL
VAR_INPUT
END_VAR
VAR
	X, Y		: INT;			// index for loop
	Found		: BOOL;			// item is taken
END_VAR

// find the next item to be moved from the table

// reset next item position to zero
RobotTakePosX:=0;
RobotTakePosY:=0;
Found:=FALSE;

// loop all items
// get the last item
FOR X:=1 TO TableX DO
	FOR Y:=1 TO TableY DO
		IF NOT ItemsTable[X,Y].InProcess AND NOT Found THEN
			RobotTakePosX:=X;
			RobotTakePosY:=Y;
			ItemsTable[X,Y].InProcess:=TRUE;	// set as item to pickup
			Found:=TRUE;
		END_IF
	END_FOR
END_FOR



METHOD TableReset : BOOL
VAR_INPUT
END_VAR
VAR
	X, Y		: INT;			// index for loop
END_VAR

// init new table
// reset all elements in the ARRAY
// set all elements to FALSE i.e. its ready to be moved

FOR X:=1 TO TableX DO
	FOR Y:=1 TO TableY DO
		ItemsTable[X,Y].InProcess := FALSE;
		ItemsTable[X,Y].Done := FALSE;
	END_FOR
END_FOR
```

```
METHOD Operate : BOOL
VAR_INPUT
END_VAR

// Send TakeXPos and TakeYPos position to the robot before
// executing the code below

// signal to robot. Move one item
IF NOT RunOne THEN
      RobotPrgNo:=1;                 // start robot program 1
END_IF

RunOne:=TRUE;                        // one item in progress

// robot task done, start workcenter
IF RobotPrgDone AND RobotPrgNo=1 THEN
      RobotPrgNo:=-1;
      WorkcenterPrgNo:=1;
END_IF

// workcenter task done, item back to table
IF WorkcenterDone THEN
      WorkcenterPrgNo:=-1;
      RobotPrgNo:=2;
END_IF

// robot task done
IF RobotPrgDone AND RobotPrgNo=2 THEN
      RobotPrgNo:=-1;
      RunOne:=FALSE;                 // stop item in progress
      ItemsTable[RobotTakePosX,RobotTakePosY].Done := TRUE;
END_IF
```