IAI SEMINAR BASIC

POSITIONING CONTROLLER

Technical support Team
SUS BKK

Agenda

-Positioning controller operation

- How to link and using "RC Software"?
- Position data table and How to Teaching method.

-Testing move and checking position

- Incremental movement option set up.
- Push force movement option set up.
- PLC control to PCON (Simulate by Switch box)
- I/O description.

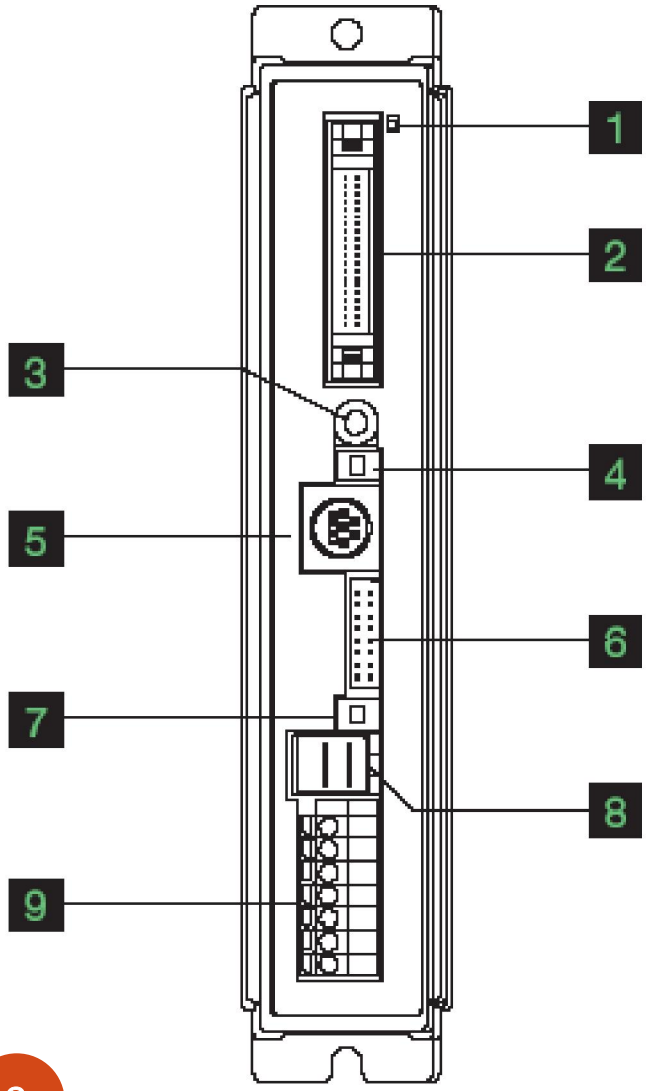
-Parameter detail.

- Programming controller operation (SEL language) .
  - How to use "*IA* Software" ?
  - Position data table and How to Teaching method.

  -Testing move and checking position

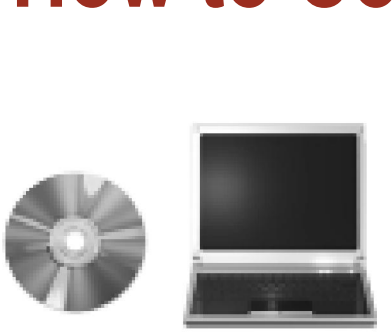
  - Basic Command SEL language.
    - Movement command.
    - I/O and Condition control (Selection process).
    - Program control Command.
    - Data position & Calculate Command.
    - Palletizing Command.
    - Parameter optional.

PCON Part..

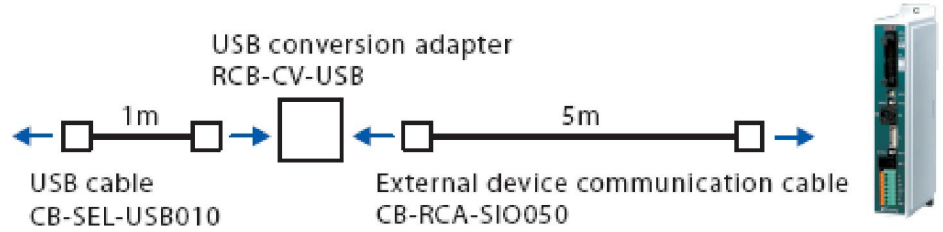


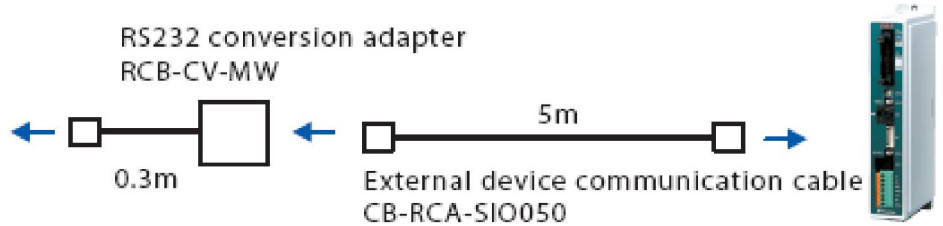
1. LED display
2. PIO connector
3. Rotary switch that sets axis numbers
4. Mode switch
5. PC cable connector
6. Encoder connector
7. Brake release switch
8. Motor connector
9. Power terminal block

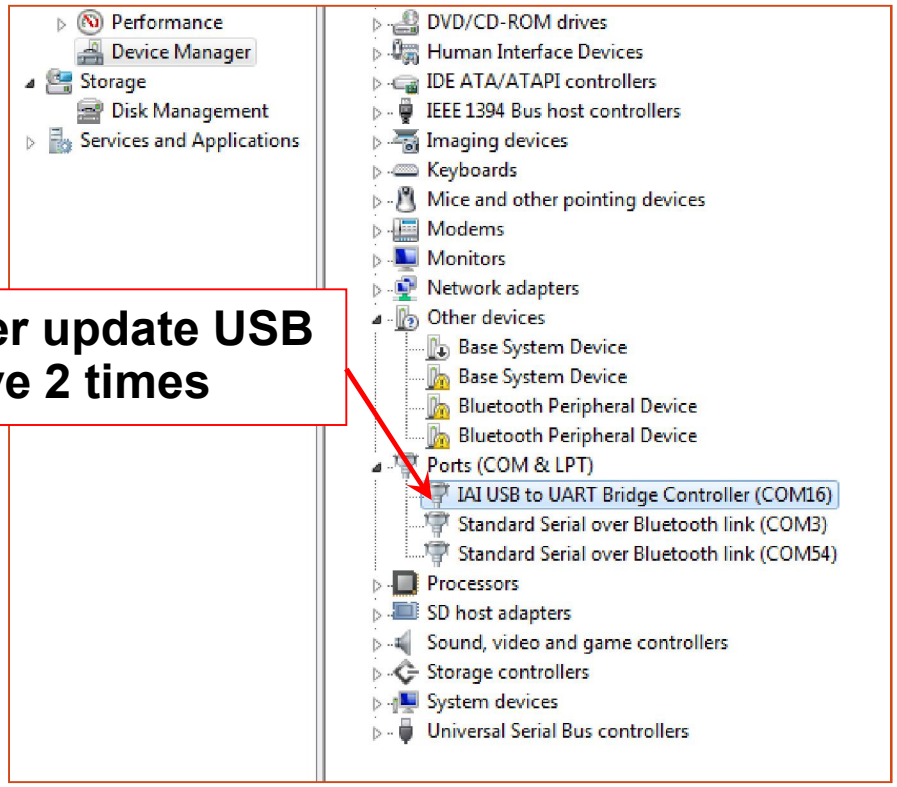
How to Connect PCON...?



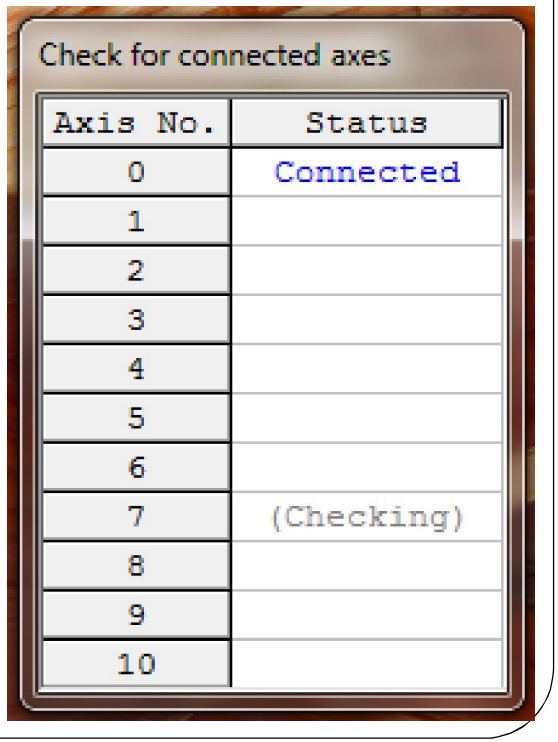
PC software (CD)



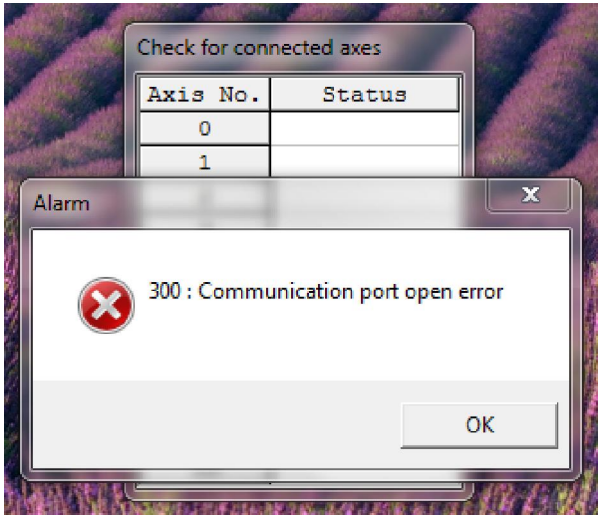




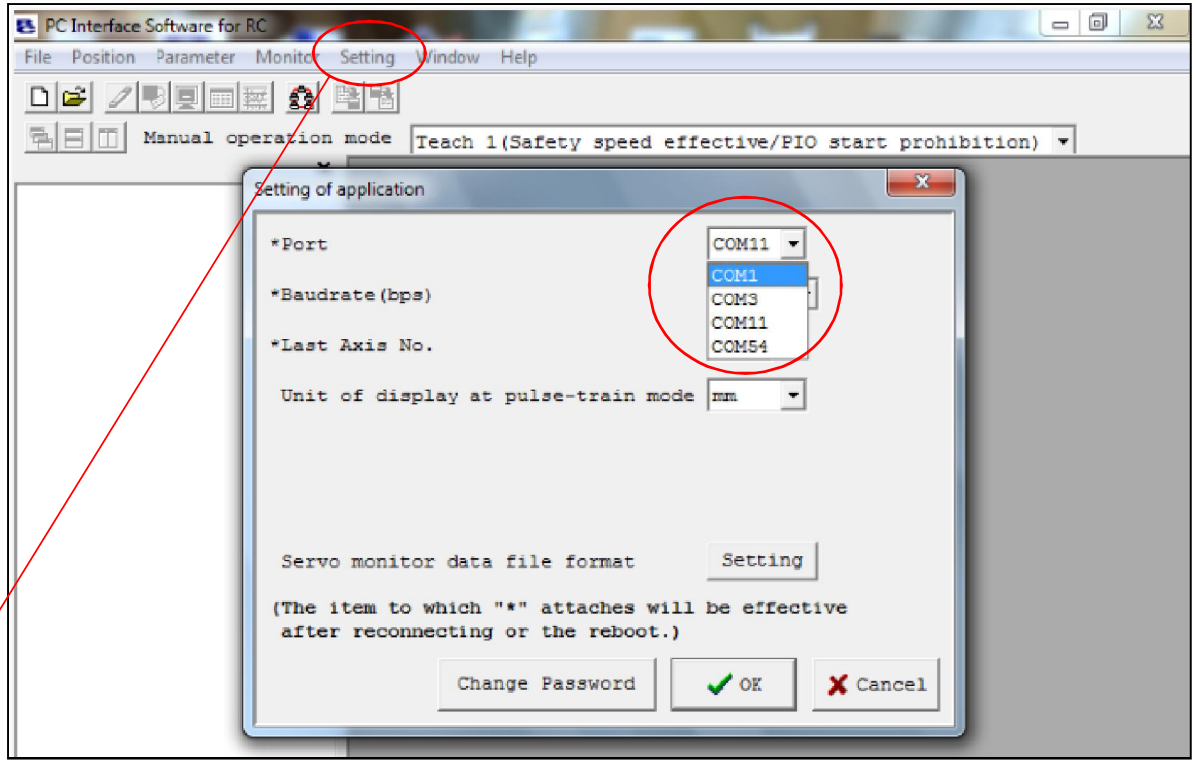
After update USB drive 2 times

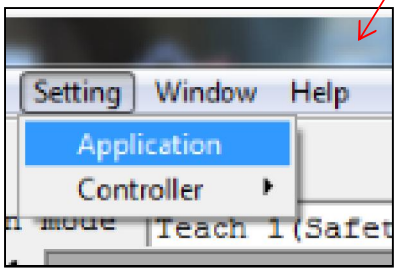


Re-Connect PCON…



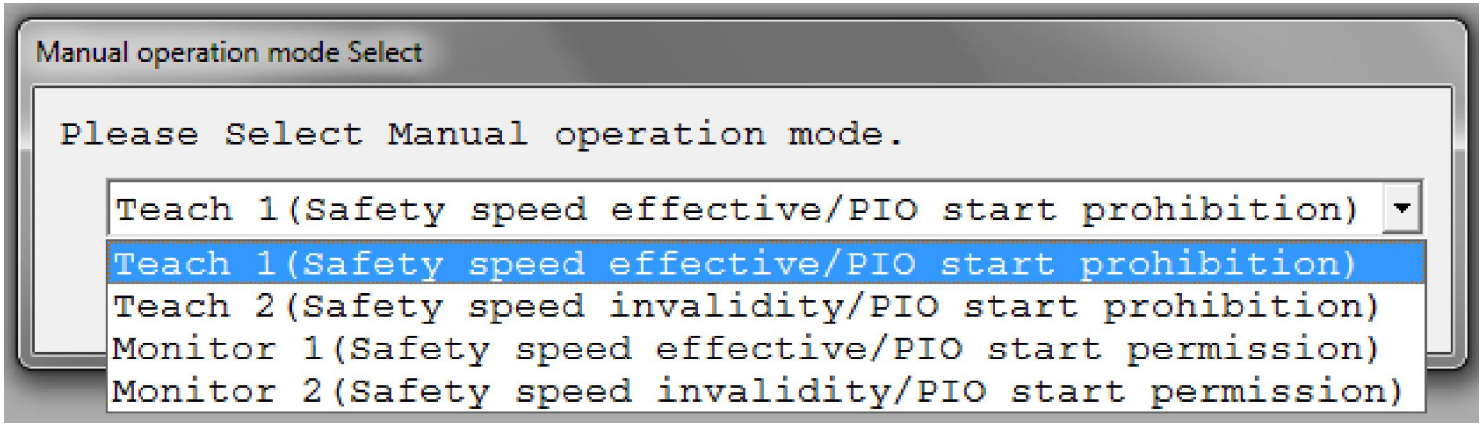
Comport mismatch





Click on “Setting” then choose “Application” and next change comport as Device Manager and then adjust Baudrate finally click “OK”

Operation Mode of RC. software



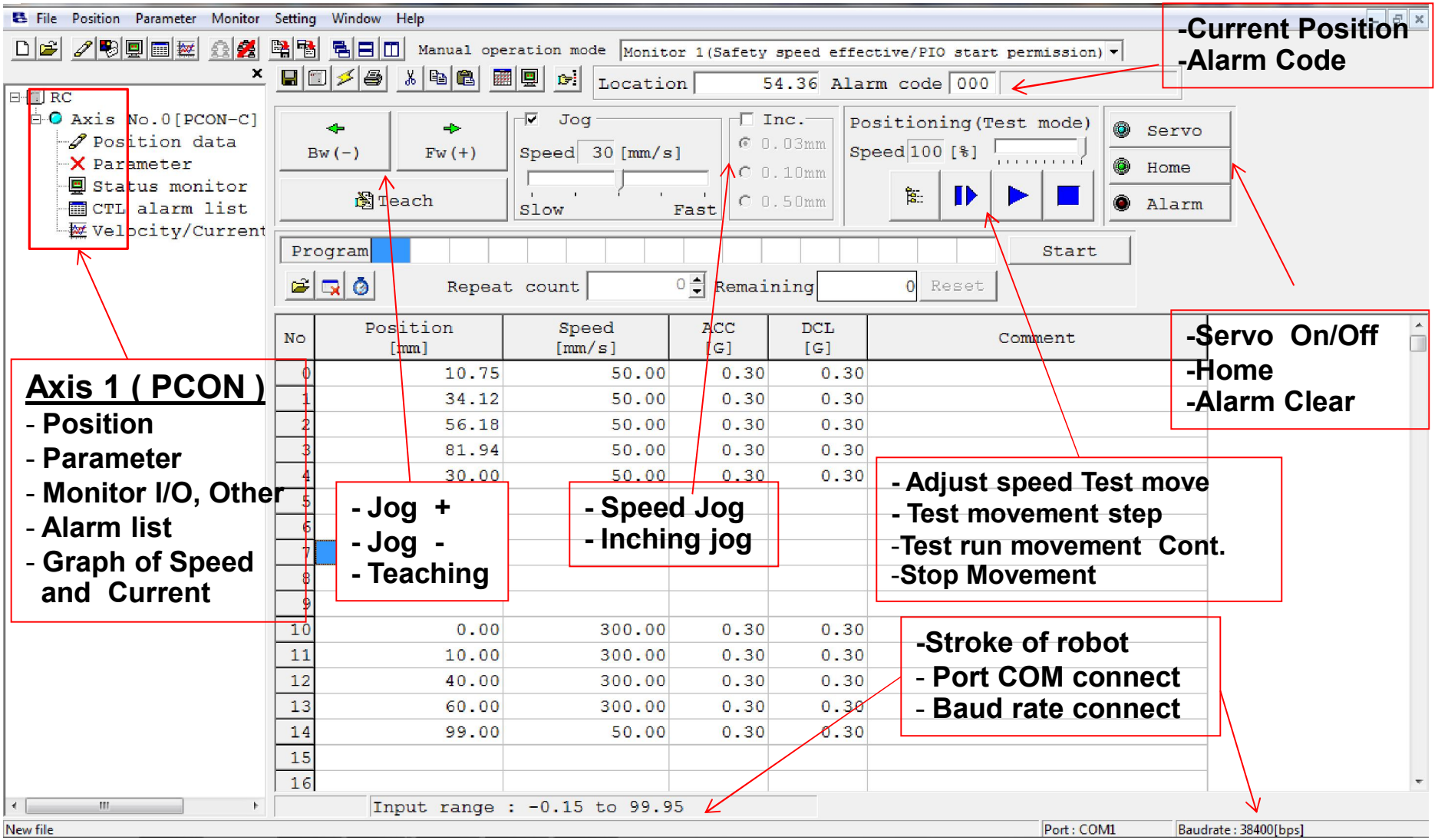
Teach 1: Can edit data and testing movement but block Speed limit is 100mm/s in Manual Mode.

Teach 2 : Can edit data and testing movement by unblock Speed in Manual Mode.

Monitor 1 : Can't edit data and testing movement by PIO but block Speed in Auto Mode.

Monitor 2 : Can't edit data and testing movement by PIO but unblock Speed in Auto Mode.

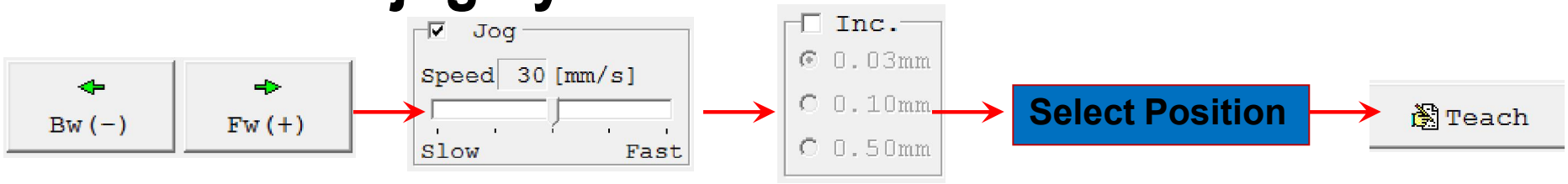
Position data table & Tool icon.



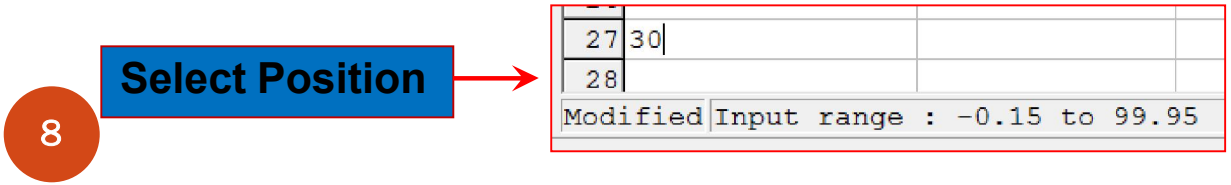
Teaching Method

1. Off servo and move actuator to position by hand

2. Jog finding to position Adjust speed jog and distance jog by software



3. Typing into position table as we know real distance working operate



After we get position
that perfect click
Load to CTL icon

Incremental Movement

Location -0.01 Alarm code 000

Bw (-) | Fw (+) | Teach

Jog — Speed 30 [mm/s] — Slow … Fast

Inc. — 0.03mm | 0.10mm | 0.50mm

Positioning (Test mode) — Speed 100 [%]

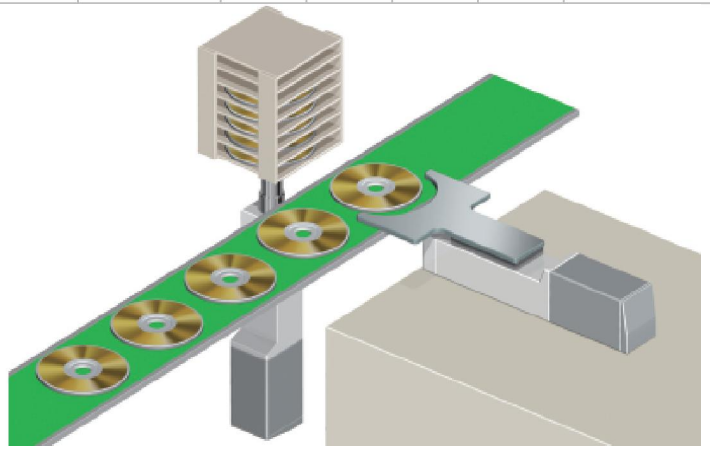
Servo | Home | Alarm

Program | Start

Repeat count 0 | Remaining 0 | Reset

No	Position [mm]	Speed [mm/s]	ACC [G]	DCL [G]	Push [%]	LoTh [%]	Pos.band [mm]	Zone + [mm]	Zone - [mm]	ACC/DCL mode	ABS INC	Gain Set	Stop Mode	VibSup No.
7	80.00	50.00	0.30	0.30	0	0	0.10	0.00	0.00	0	0	0	0	0
8	90.00	50.00	0.30	0.30	0	0	0.10	0.00	0.00	0	0	0	0	0
9														
10	5.00	50.00	0.30	0.30	0	0	0.10	0.00	0.00	0	1	0	0	0
11	-5.00	50.00	0.30	0.30	0	0	0.10	0.00	0.00	0	1	0	0	0
12														

ABS/INC 0 → 1



Advantage point …

Step movement between position same distance value so we can using Incremental movement function for reduce position to once data only

Push force Movement

Location 89.99 Alarm code 000

Bw(-) Fw(+) Teach

Jog Speed 30 [mm/s] Slow Fast

Inc. 0.03mm 0.10mm 0.50mm

Positioning(Test mode) Speed 100 [%]

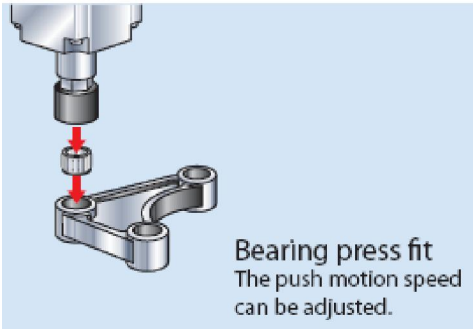
Servo Home Alarm

Program Start

Repeat count 0 Remaining 0 Reset

No	Position [mm]	Speed [mm/s]	ACC [G]	DCL [G]	Push [%]	LoTh [%]	Pos.band [mm]	Zone + [mm]	Zone - [mm]	ACC/DCL mode	ABS INC	Gain Set	Stop Mode	VibSup No.
13	0.00	50.00	0.30	0.30	0	0	0.10	0.00	0.00	0	0	0	0	0
14	50.00	50.00	0.30	0.30	20	0	40.00	0.00	0.00	0	0	0	0	0
15														
16														
17														
18														
19														
20														
21														

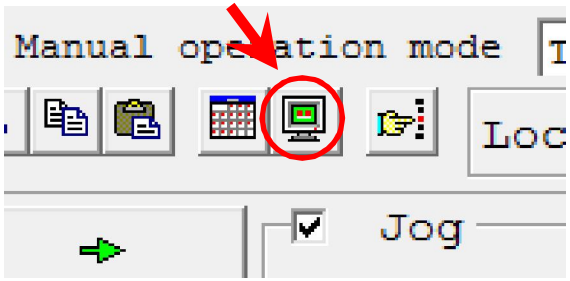
Push Mode set up
Exp : Push = 20 %
Pos.band = 40 mm



Bearing press fit
The push motion speed can be adjusted.

Advantage point …
Robot can adjust the force value for Push force application by checking current during action without “Load cell “

Status Monitor



When we want to check the status overall of robot should be click “Status Monitor “ to see

<u>Example</u>

Status of I/O connecting between PLC and IAI controller, we can check.

Status[Axis No.0]

I/O Test

Axis status

Location(mm)	89.99
Speed(mm/s)	0.00
Alarm code	000

Internal flags

Power	Servo
Home Cmplt	RUN
HMCK sensor	MANU

Current

Cur. cmnd(mA)	561
Cur. ratio(%)	46.8

Input(PIO Pattern=0)

Name	Status
PC1	OFF
PC2	OFF
PC4	OFF
PC8	OFF
PC16	OFF
PC32	OFF
-	OFF
-	OFF
-	OFF
BKRL	OFF
RMOD	OFF
HOME	OFF
*STP	OFF
CSTR	OFF
RES	OFF
SON	OFF

Output(PIO Pattern=0)

Name	Status
PM1	OFF
PM2	ON
PM4	ON
PM8	ON
PM16	OFF
PM32	OFF
MOVE	OFF
ZONE1	ON
PZONE	OFF
RMDS	ON
HEND	ON
PEND	OFF
SV	ON
*EMGS	ON
*ALM	ON
LOAD	OFF

Special input

Name	Status
(Reserved)	OFF
(Reserved)	OFF
(Reserved)	OFF
HMCK sensor	OFF
(Reserved)	OFF
(Reserved)	OFF
(Reserved)	OFF
Enable SW	OFF
Mode SW	ON
(Reserved)	ON
(Reserved)	OFF
(Reserved)	OFF
(Reserved)	ON
(Reserved)	OFF
(Reserved)	ON
(Reserved)	OFF

Selecting Position for moving.(Input)

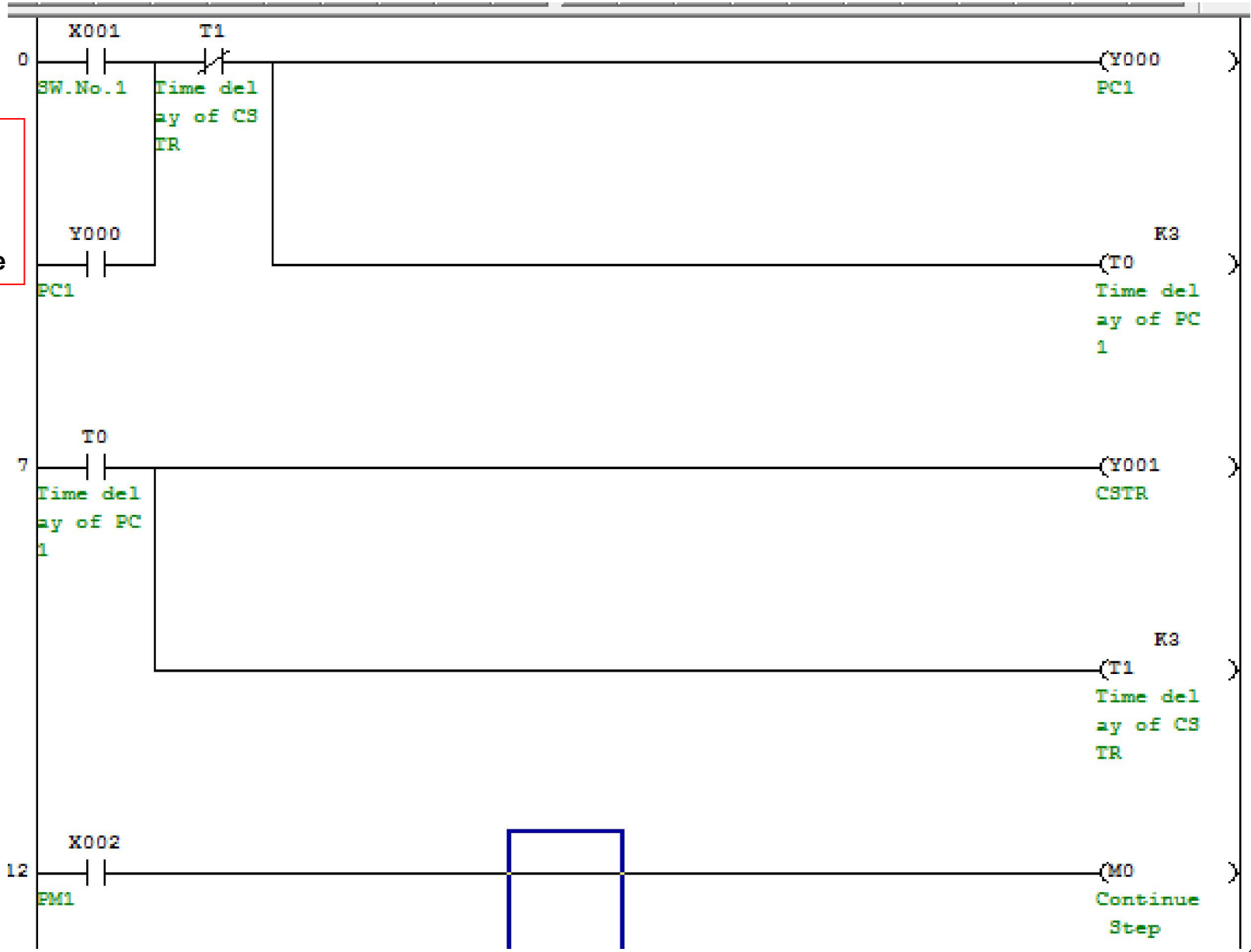
Pin No.	Classification		Parameters (select PIO pattern)					
			0	1	2	3	4	5
			Positioning mode	Teaching mode	256-point mode	512-point mode	Solenoid Valve Mode 1	Solenoid Valve Mode 2
		Positioning Points	64 points	64 points	256 points	512 points	7 points	3 points
		Zone signal	○	×	×	×	○	○
		P-zone signal	○	○	○	×	○	○
1A	24V		P24					
2A	24V		P24					
3A	–		NC					
4A	–		NC					
5A	Input	IN0	PC1	PC1	PC1	PC1	ST0	ST0
6A		IN1	PC2	PC2	PC2	PC2	ST1	ST1 (JOG+)
7A		IN2	PC4	PC4	PC4	PC4	ST2	ST2 (-)
8A		IN3	PC8	PC8	PC8	PC8	ST3	–
9A		IN4	PC16	PC16	PC16	PC16	ST4	–
10A		IN5	PC32	PC32	PC32	PC32	ST5	–
11A		IN6	–	MODE	PC64	PC64	ST6	–
12A		IN7	–	JISL	PC128	PC128	–	–
13A		IN8	–	JOG+	–	PC256	–	–
14A		IN9	BKRL	JOG-	BKRL	BKRL	BKRL	BKRL
15A		IN10	RMOD	RMOD	RMOD	RMOD	RMOD	RMOD
16A		IN11	HOME	HOME	HOME	HOME	HOME	–
17A		IN12	*STP	*STP	*STP	*STP	*STP	–
18A		IN13	CSTR	CSTR/PWRT	CSTR	CSTR	–	–
19A		IN14	RES	RES	RES	RES	RES	RES
20A		IN15	SON	SON	SON	SON	SON	SON

Moving complete signal feed back (Output)

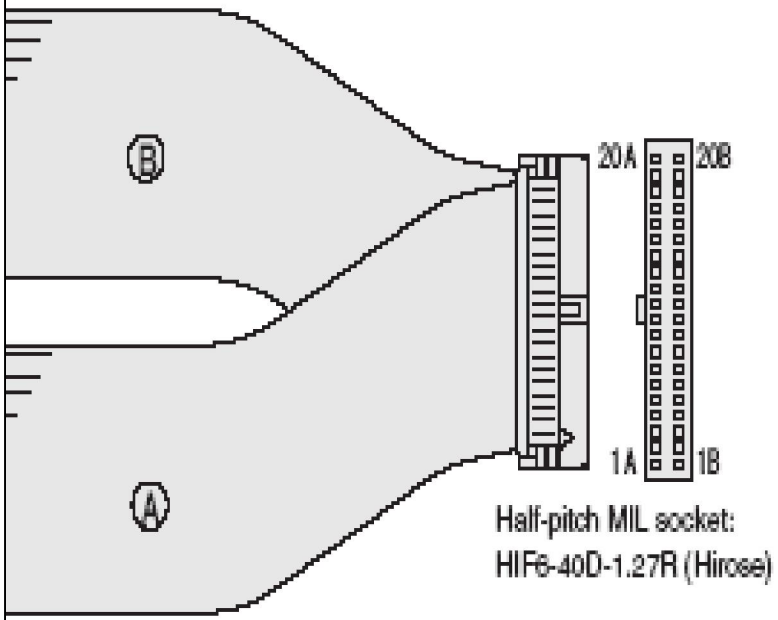
1B	Output	OUT0	PM1		PM1	PM1	PM1	PE0	LSO
2B		OUT1	PM2		PM2	PM2	PM2	PE1	LS1 (TRQS)
3B		OUT2	PM4		PM4	PM4	PM4	PE2	LS2 (–)
4B		OUT3	PM8		PM8	PM8	PM8	PE3	–
5B		OUT4	PM16		PM16	PM16	PM16	PE4	–
6B		OUT5	PM32		PM32	PM32	PM32	PE5	–
7B		OUT6	MOVE		MOVE	PM64	PM64	PE6	–
8B		OUT7	ZONE1		MODES	PM128	PM128	ZONE1	ZONE1
9B		OUT8	PZONE		PZONE	PZONE	PM256	PZONE	PZONE
10B		OUT9	RMDS		RMDS	RMDS	RMDS	RMDS	RMDS
11B		OUT10	HEND		HEND	HEND	HEND	HEND	HEND
12B		OUT11	PEND		PEND/WEND	PEND	PEND	PEND	–
13B		OUT12	SV		SV	SV	SV	SV	SV
14B		OUT13	*EMGS		*EMGS	*EMGS	*EMGS	*EMGS	*EMGS
15B		OUT14	*ALM		*ALM	*ALM	*ALM	*ALM	*ALM
16B		OUT15	LOAD/TRQS		–	LOAD/TRQS	LOAD/TRQS	LOAD/TRQS	–
17B	–					NC			
18B	–					NC			
19B	0V					N			
20B	0V					N			

Simple Ladder of PLC control to PCON by IO.

X001 = Sw.start
Y000 = PC1
Y001 = CSTR
X002 = PM1
M0 = Continue



Name and Color mark of PIN (PIO Cable).

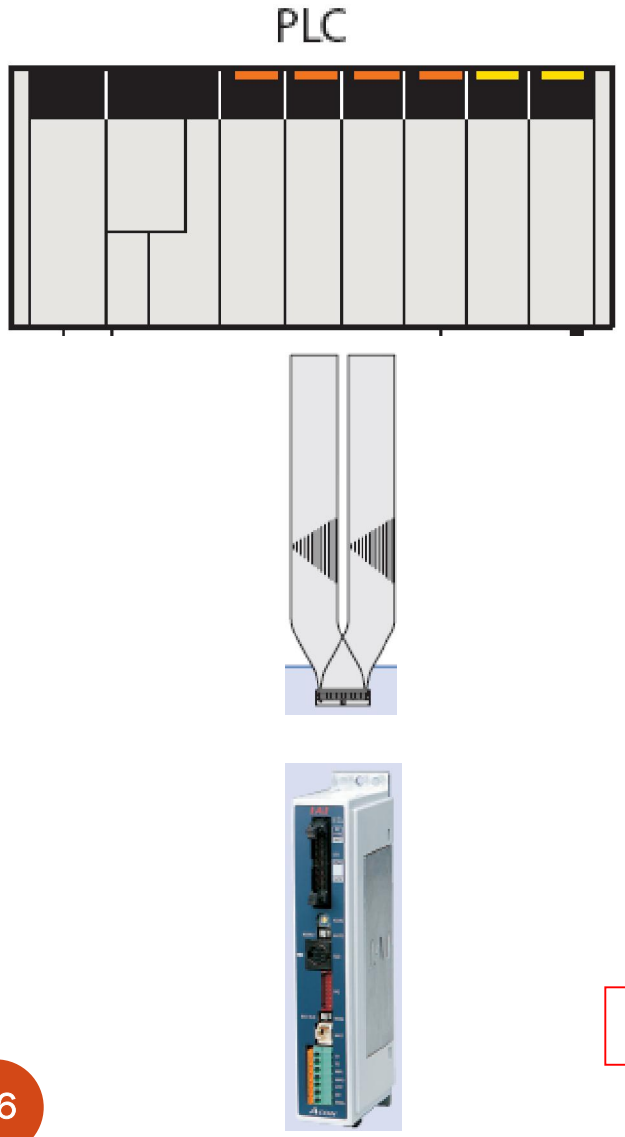


Half-pitch MIL socket:
HIF6-40D-1.27R (Hirose)

No.	Signal	Cable color	Wiring	No.	Signal	Cable color	Wiring
1A	24V	Brown-1	A flat cable (crimped)	1B	OUT0	Brown-3	B flat cable (crimped) AWG28
2A	24V	Red-1		2B	OUT1	Red-3	
3A	—	Orange-1		3B	OUT2	Orange-3	
4A	—	Yellow-1		4B	OUT3	Yellow-3	
5A	IN0	Green-1		5B	OUT4	Green-3	
6A	IN1	Blue-1		6B	OUT5	Blue-3	
7A	IN2	Purple-1		7B	OUT6	Purple-3	
8A	IN3	Gray-1		8B	OUT7	Gray-3	
9A	IN4	White-1		9B	OUT8	White-3	
10A	IN5	Black-1		10B	OUT9	Black-3	
11A	IN6	Brown-2		11B	OUT10	Brown-4	
12A	IN7	Red-2		12B	OUT11	Red-4	
13A	IN8	Orange-2		13B	OUT12	Orange-4	
14A	IN9	Yellow-2		14B	OUT13	Yellow-4	
15A	IN10	Green-2		15B	OUT14	Green-4	
16A	IN11	Blue-2		16B	OUT15	Blue-4	
17A	IN12	Purple-2		17B	—	Purple-4	
18A	IN13	Gray-2		18B	—	Gray-4	
19A	IN14	White-2		19B	0V	White-4	
20A	IN15	Black-2		20B	0V	Black-4	

Remark : ดูในเอกสารประกอบ ภาพผนวกเพิ่มเติม

PLC VS. IAI (Position CT. Series)



No.	Signal	Cable color	Wiring
1A	24V	Brown-1	A flat cable (crimped)
2A	24V	Red-1	
3A	—	Orange-1	
4A	—	Yellow-1	
5A	IN0	Green-1	
6A	IN1	Blue-1	
7A	IN2	Purple-1	
8A	IN3	Gray-1	
9A	IN4	White-1	
10A	IN5	Black-1	
11A	IN6	Brown-2	
12A	IN7	Red-2	
13A	IN8	Orange-2	
14A	IN9	Yellow-2	
15A	IN10	Green-2	
16A	IN11	Blue-2	
17A	IN12	Purple-2	
18A	IN13	Gray-2	
19A	IN14	White-2	
20A	IN15	Black-2	

No.	Signal	Cable color	Wiring
1B	OUT0	Brown-3	B flat cable (crimped) AWG28
2B	OUT1	Red-3	
3B	OUT2	Orange-3	
4B	OUT3	Yellow-3	
5B	OUT4	Green-3	
6B	OUT5	Blue-3	
7B	OUT6	Purple-3	
8B	OUT7	Gray-3	
9B	OUT8	White-3	
10B	OUT9	Black-3	
11B	OUT10	Brown-4	
12B	OUT11	Red-4	
13B	OUT12	Orange-4	
14B	OUT13	Yellow-4	
15B	OUT14	Green-4	
16B	OUT15	Blue-4	
17B	—	Purple-4	
18B	—	Gray-4	
19B	0V	White-4	
20B	0V	Black-4	

Out put from PLC

Go to Input of PLC

Parameter option. (Positioning CT.)

Parameter No.1 & 2	= Zone +,Zone -
Parameter No.3 & 4	= Soft limit =+,-
Parameter No.5	= Home direction
Parameter No.15	= Stop movement momentary
Parameter No.21	= Servo on
Parameter No.22	= Home offset
Parameter No.25	= Change I/O pattern(0-5)
Parameter No.26	= Jog speed

Question…

BASIC PROGRAMMING

Technical support Team
SUS BKK

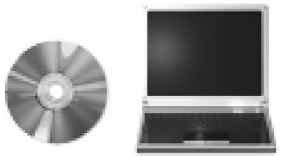
SEL Language (Programmable CT.)

ภาษา SEL (Shimizu Electric Language) คือภาษาคอมพิวเตอร์ที่ใช้โปรแกรมการทำงานคอนโทรลเลอร์ IAI (SEL Series) เพื่อที่จะควบคุมการเคลื่อนที่ของแกน Actuator และยังสามารถเชื่อมต่อกับอุปกรณ์ภายนอกโดยผ่าน port I/O และ RS232 โดยสามารถทำให้เกิดเป็นระบบ Automation ได้เหมือนโปรแกรมภาษาอื่น ๆ

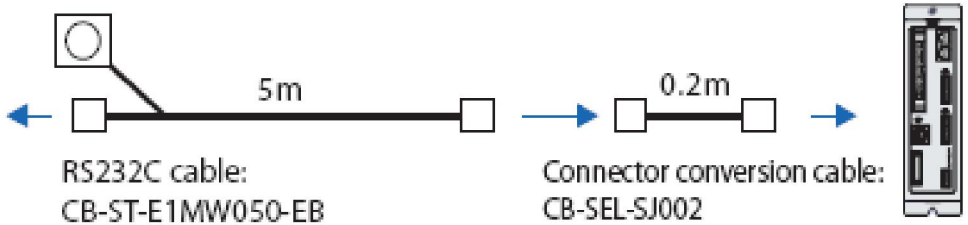
ภาษา SEL เป็นภาษาที่ง่ายต่อการใช้งาน แต่มีชุดคำสั่งให้ใช้งานที่หลากหลายครอบคลุมการใช้งานในระบบ Automation ได้ทั้งหมด

ลักษณะชุดคำสั่งของภาษา SEL จะมีลักษณะที่คล้ายกับภาษา Basic โดย จะ scan คำสั่งจากด้านซ้ายไปหาด้านขวา และ จากด้านบนลงด้านล่างของตารางโปรแกม และ ทำงานทีละคำสั่งตามลำดับ

PC Interface for SEL software.

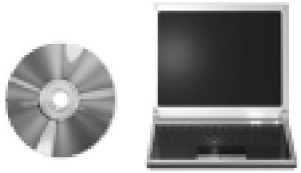


PC software (CD)

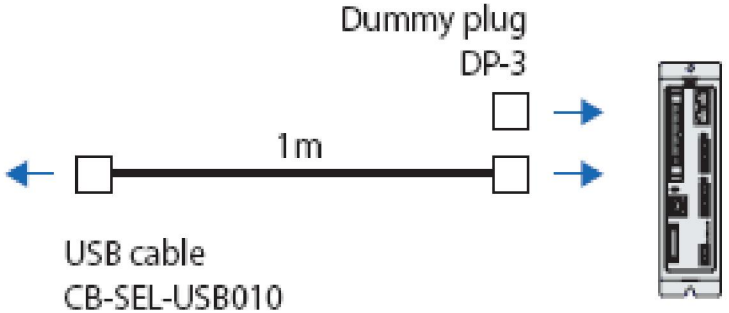


RS232C cable:
CB-ST-E1MW050-EB

Connector conversion cable:
CB-SEL-SJ002



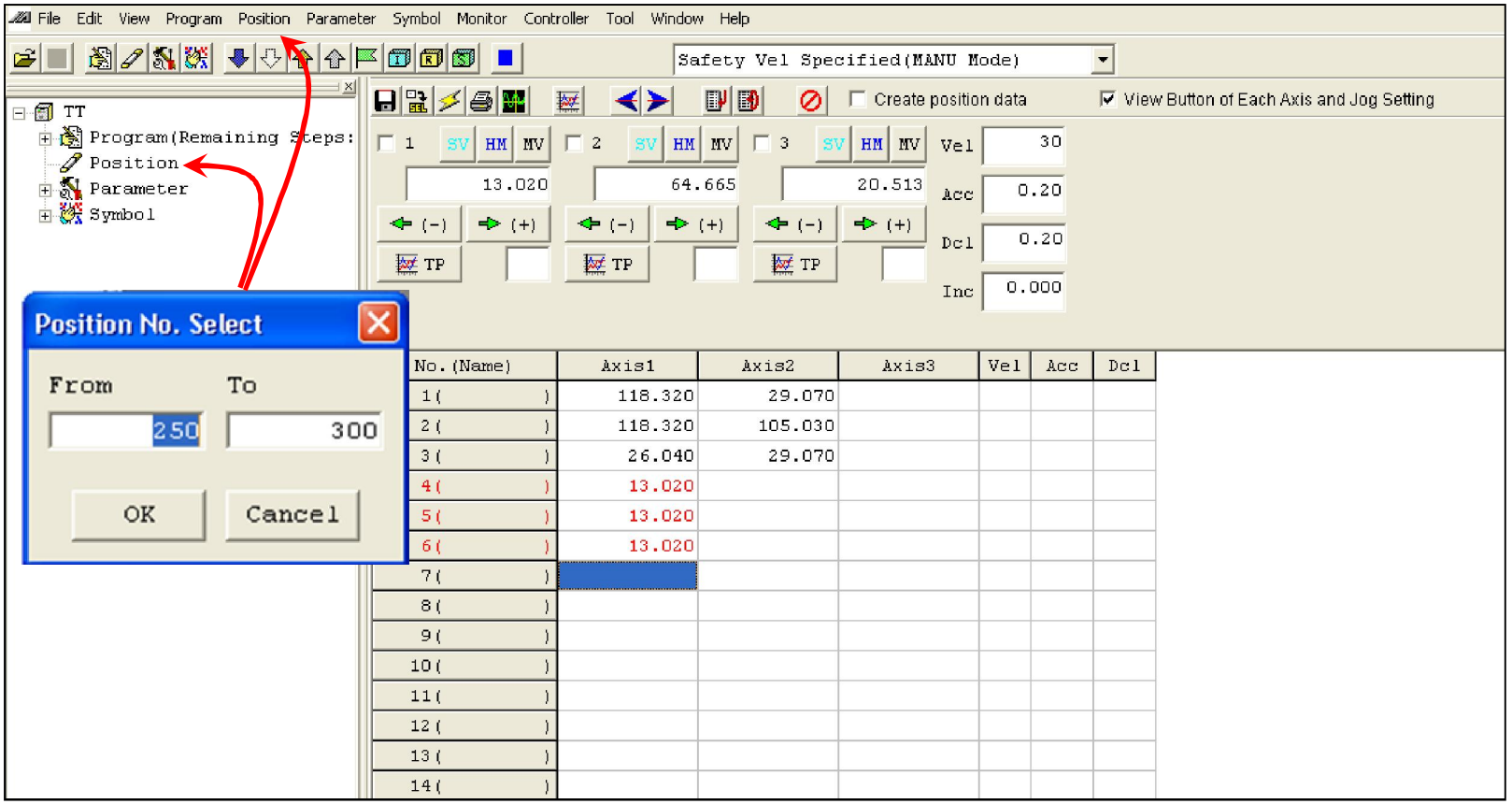
PC software (CD)



USB cable
CB-SEL-USB010

- **"AUTO" mode always when plug for connecting or unplug.*****
- **Check comport as Device Manager and then select on**
- **If comport can't found IAI cable please update USB driver 2 times. First and then go to check for Device Manager**

Position data edit.



"Position" → Edit (1-1500) → HOME Z,X,Y

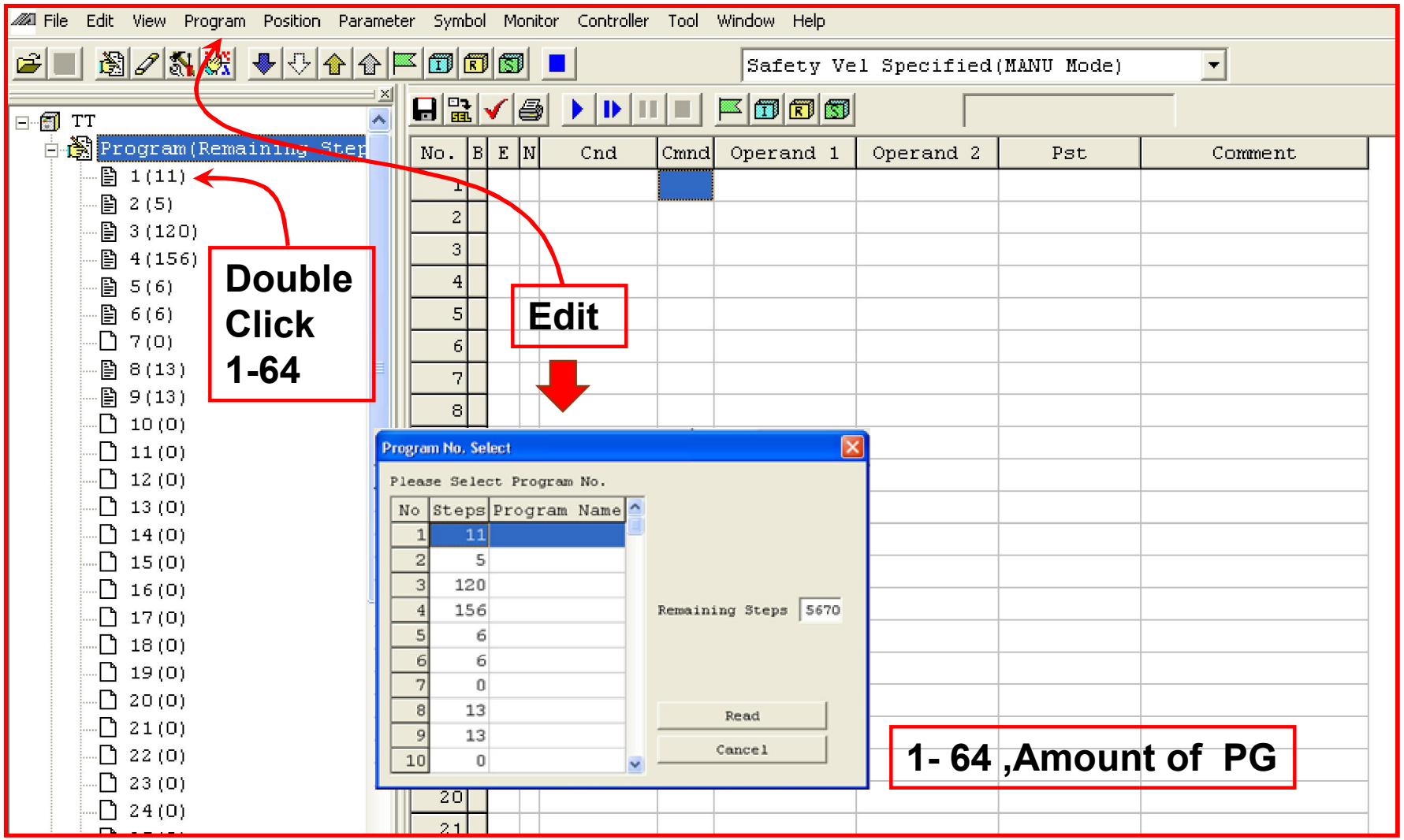
→ Teaching Position → testing Movement

PSEL,ASEL : Position up to 1500

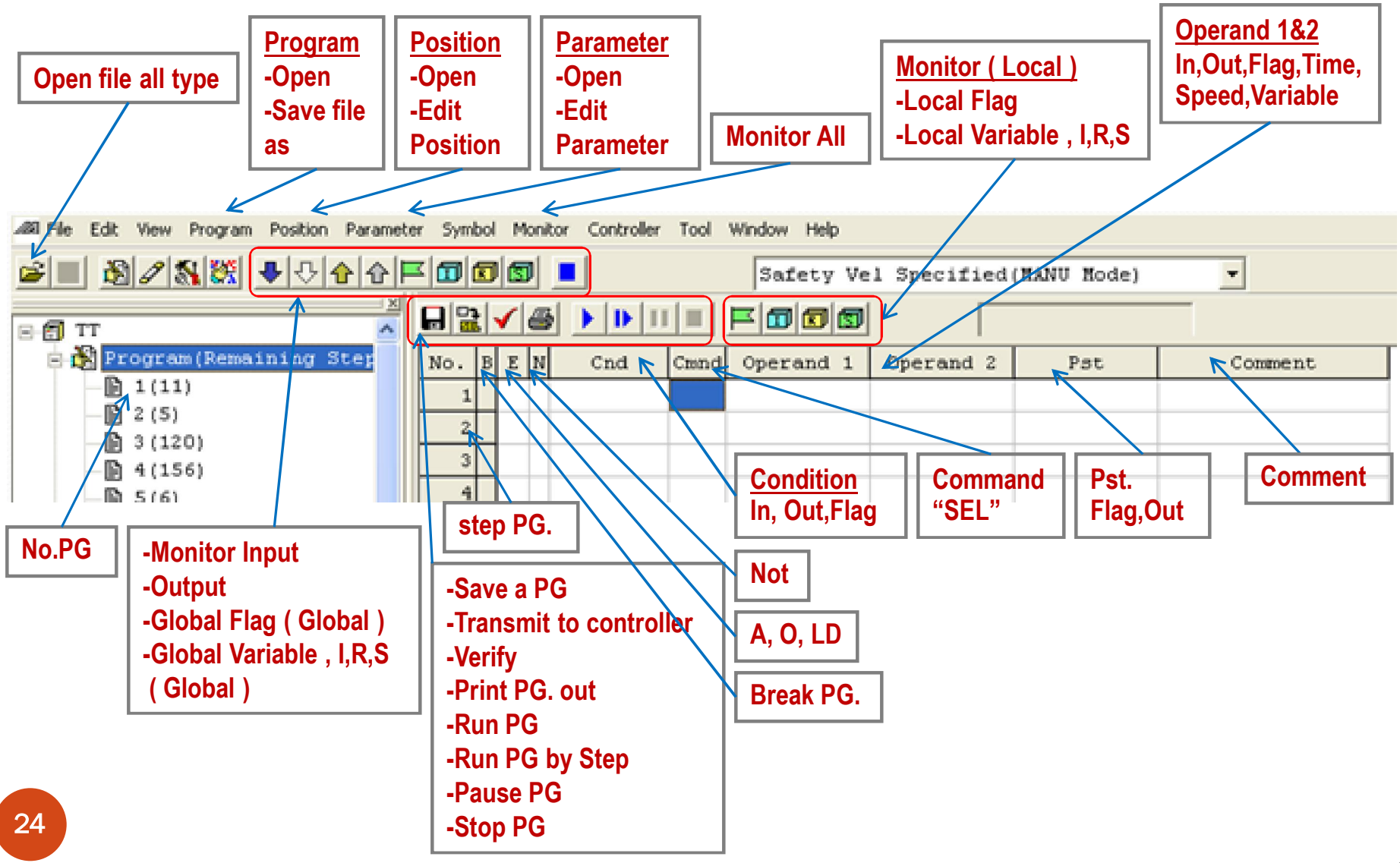
3 Method for Teaching Position

1. Off servo and move by hand
2. Typing value position(mm)
3. Jog +,- for finding teaching

Program Set up.



Tools of PC Interface SW. for SEL



Major Command meaning.

HOME : Sending actuator to home position (Set home)

VEL : Set the velocity in program for movement function

MOVP : Move each axis involved at a designate velocity to the chosen point. The axes are not relate in a movement (Point to Point movement)

MOVL : Move the actuator to the specified position number by adjusting acceleration, velocity and axes to arrive in the shortest path both axes complete move same time (Linear interpolation)

ARC : Move along an arc from the current position to the position specified

CIR : Move along an circle from the current position to the position specified

PATH : Move Continue from the position specified (Move linear) position specifies (Moving together 2 Axis)

ARC2 : Move along arc originating the current position continuing movement

CIR2 : Move along arc originating the current position continuing movement

WTON : Wait for input / flag to turn “on”

WTOF : Wait for input / flag to turn “off”

Major Command meaning.

BTOF : Turn output / flag “off”

TIMW : Sets a timer that pause program for specific time

TAG : Set a place holder for GOTO loop.

GOTO : Return (Jump) the program control to the TAG number specified, thus creating a simple loop

EXSR : Execute Subroutine program

BGSR : Begin Subroutine program

EDSR : End of Subroutine program

EXPG : Sub-program start

ABPG : Finish sub-program

TIMC : Time cancel

LET : Set variables (announce variables)

CPEQ : Compare equal ?

EXIT : Finish program

Actuator Control Designation Command (Move to Position)

● **VEL(Velocity)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output port · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	VEL	Velocity		

Remark : VEL is command for assign speed movement of robot

● **MOVP (Point to-Point Position Data)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output port · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	MOVP	Position no.		Optional

[Function] This command moves the actuator to the designated position number from point to point without interpolation.

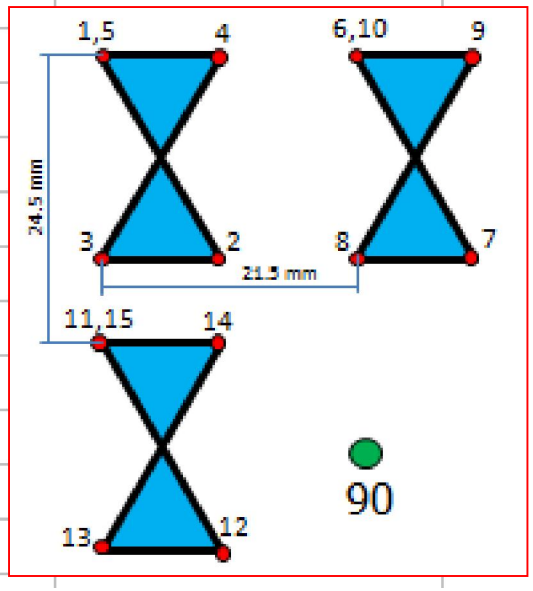
● **MOVL(Position Data with Interpolation)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output port · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	MOVL	Position No. Variable		Optional

[Function] Moves the actuator to the designated point while using interpolation (not point to point).

Example1 : Movement by Position (Non continuesly.)

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2					VEL	50			
3					ACC	0.3			
4					DCL	0.3			
5									
6					MOVP	1			
7					MOVP	2			
8					MOVP	3			
9					MOVP	4			
10					MOVP	5			
11									
12					MOVL	6			
13					MOVL	7			
14					MOVL	8			
15					MOVL	9			
16					MOVL	10			
17									
18					MOVL	90			
19									
20					EXIT				



Actuator Control Designation Command (Arc & CIR Movement)

● ARC2 (Move along arc 2 (CP operation))

Extension condition (LD, A, O, AB, OB)	Input condition (I/O, flag)	Command, declaration			Output (Output, flag)
		Command, declaration	Operand 1	Operand 2	
Optional	Optional	ARC2	Passing position number	End position number	PE

[Function] Move along an arc originating from the current position, passing the specified position and terminating at the end position, via arc interpolation.

● CIR2 (Move along circle 2 (CP operation))

Extension condition (LD, A, O, AB, OB)	Input condition (I/O, flag)	Command, declaration			Output (Output, flag)
		Command, declaration	Operand 1	Operand 2	
Optional	Optional	CIR2	Passing position 1 number	Passing position 2 number	PE

[Function] Move along a circle originating from the current position and passing positions 1 and 2, via arc interpolation.
The rotating direction of the circle is determined by the given position data.
The diagram below describes a CW (clockwise) movement. Reversing passing positions 1 and 2 will change the direction of movement to CCW (counterclockwise).

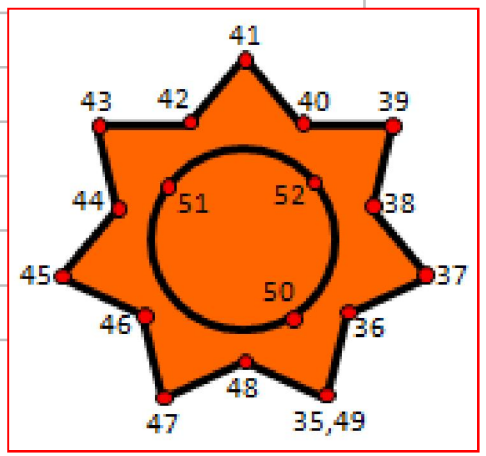
● **PATH (Path Movement)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output port · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	PATH	Starting position No.	Ending position No.	PE

[Function] Actuator moves continuously between the designated starting point and the finishing point. The locus is a B-spline-type, free-form curve which passes through the inside of the designated coordinate. It is possible for the actuator to move close to the designated coordinate by increasing the acceleration. However, when it exceeds the maximum acceleration, an error will occur.

Example2 : Movement by Position & Curve

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2					VEL	50			
3					ACC	0.3			
4					DCL	0.3			
5									
6					MOVL	35			
7					MOVL	36			
8					MOVL	37			
9					MOVL	38			
10					MOVL	39			
11					MOVL	40			
12					MOVL	41			
13									
14					PATH	42	49		
15									
16					MOVL	50			
17					CIR2	51	52		
18					EXIT				



I/O Designation Command

● BTXX (Output Port • Flag Operation)

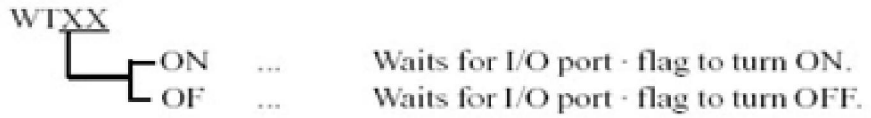
Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output port · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	BTXX	Output · Flag	Optional	

[Function] Turns ON, OFF, or inverts from the output · flag designated in operand 1 to the output · flag designated in operand 2.

● WTXX (I/O Port • Flag Wait)

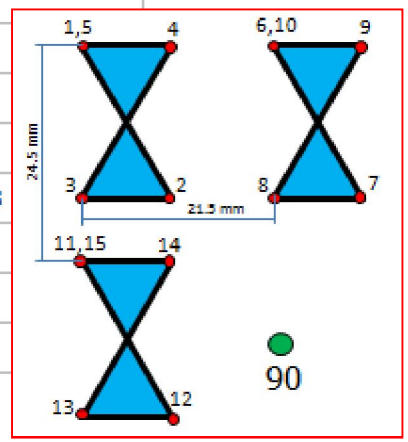
Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	WTXX	I/O · Flag	Optional Time out	TU

[Function] Program waits until designated in operand 2 turns ON/OFF.
Can abort the wait after a set time by designating a time in operand 2.
Setting range is 0.01~99 seconds. After a set time has elapsed, the output turns ON (Only where is an operand 2).

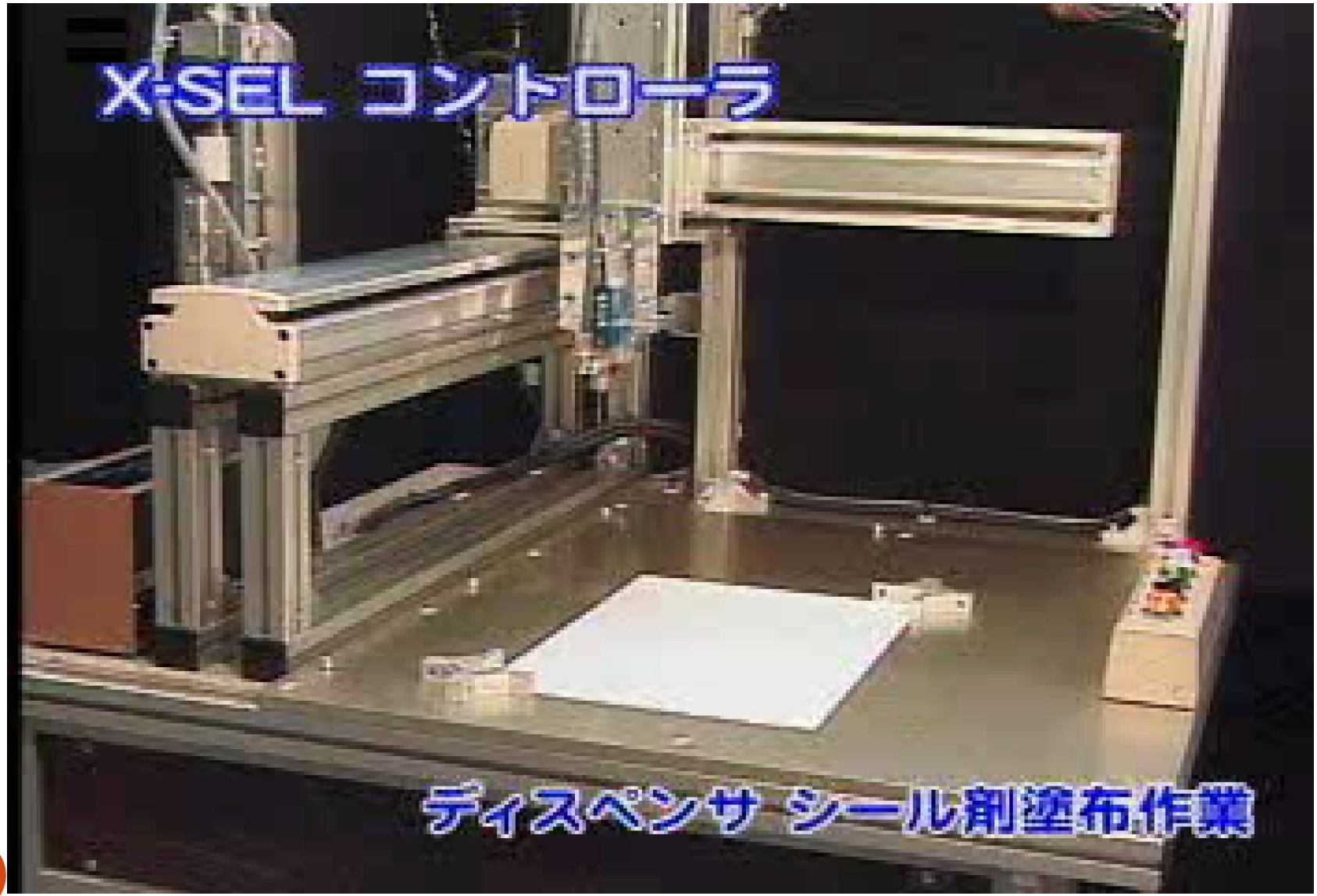


Example3 : I/O Command

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1					VEL	50			
2					ACC	0.3			
3					DCL	0.3			
4									
5					WTON	1			SW. start MC
6									
7					MOVL	1			
8					BTON	303			on bit 303
9					TIMW	0.5			
10					PATH	2	5		
11					BTOF	303			off bit 303
12					TIMW	0.5			
13									
14					MOVL	11			
15					BTON	303			on bit 303
16					TIMW	0.5			
17					PATH	12	15		
18					BTOF	303			off bit 303
19					TIMW	0.5			
20									
21					EXIT				



I/O from IAI Control Dispensing Unit (2D)



Loop and Jump command

● **TAG (Tag Declaration)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	TAG	Tag No.		

[Function] Sets the tag number designated in operand 1.

● **GOTO (Jump)**

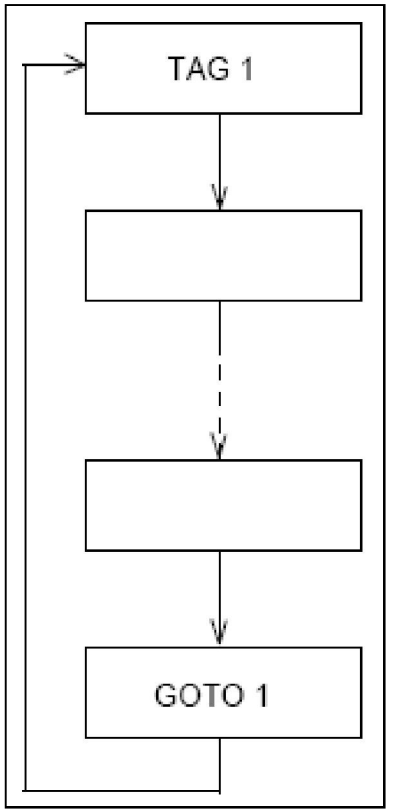
Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	GOTO	Tag No.		

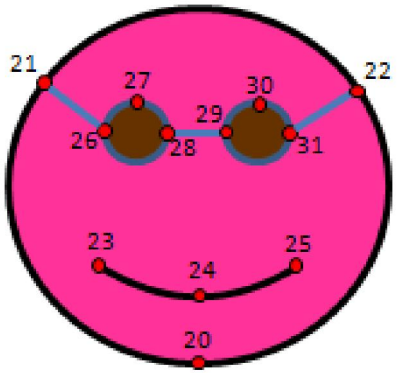
[Function] Jumps to the position of the tag number designated in operand 1.

Note: The GOTO command is valid only within the same program.

Example4 : TAG & GOTO

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1					VEL	50			
2					ACC	0.3			
3					DCL	0.3			
4									
5					TAG	1			
6					MOVL	90			
7					WTON	1			SW. start MC
8									
9					MOVL	20			
10					BTON	303			on bit 303
11					TIMW	0.5			
12					CIR2	21	22		
13					BTOF	303			off bit 303
14					TIMW	0.5			
15									
16					MOVL	23			
17					BTON	303			on bit 303
18					TIMW	0.5			
19					ARC2	24	25		
20					BTOF	303			off bit 303
21					TIMW	0.5			
22									
23					GOTO	1			





Sub program.

● **EXSR (Execute Subroutine)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output · Flag)
		Command	Operand 1	Operand 2	
		EXSR	Subroutine No.		

[Function] Executes the subroutine number designated in operand 1.

● **BGSR (Begin Subroutine)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output · Flag)
		Command	Operand 1	Operand 2	
		BGSR	Subroutine No.		

[Function] Commands the start of the subroutine number designated in operand 1.

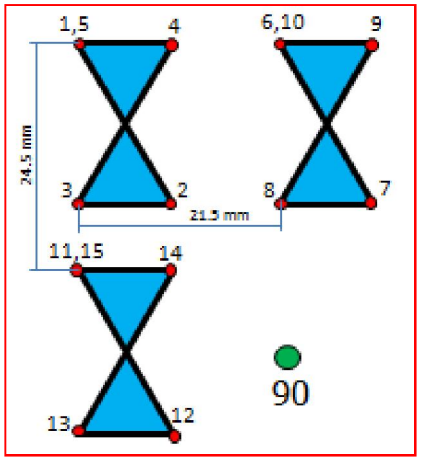
● **EDSR (End Subroutine)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output port · Flag)
		Command	Operand 1	Operand 2	
		EDSR			

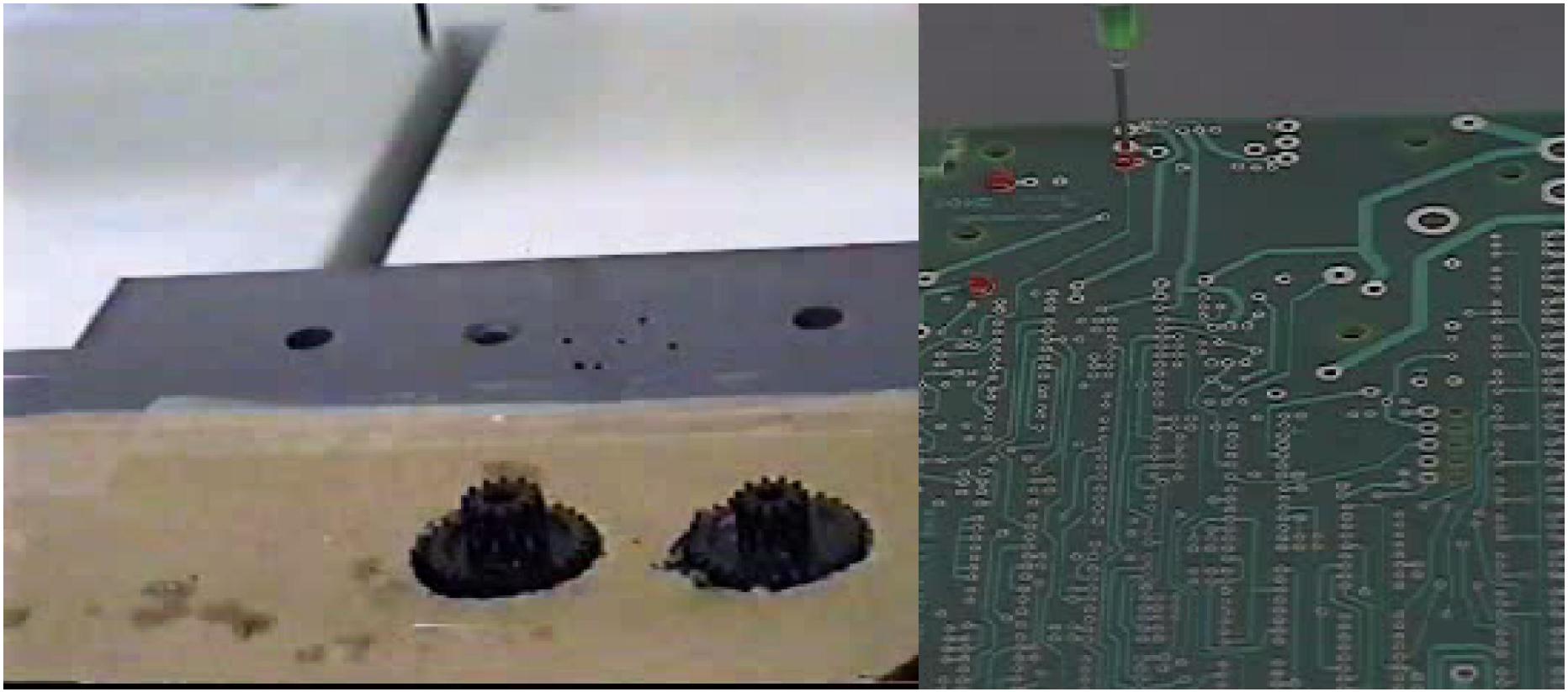
[Function] Commands end of subroutine. This is always required at the end of a subroutine. After this, the program moves to the step after the EXSR called out.

Example5 : Subroutine concept PG.

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1					VEL	50			
2					ACC	0.3			
3					DCL	0.3			
4									
5					MOVL	90			
6					WTON	1			SW. start MC
7									
8					MOVL	6			
9					BTON	303			on bit 303
10					TIMW	0.5			
11					BTOF	303			off bit 303
12									
13					MOVL	7			
14					EXSR	1			
15					MOVL	8			
16					EXSR	1			
17					MOVL	9			
18					EXSR	1			
19									
20					EXIT				
21									
22					BGSR	1			
23					BTON	303			on bit 303
24					TIMW	0.5			
25					BTOF	303			off bit 303
26					EDSR				



Application Subroutine Command

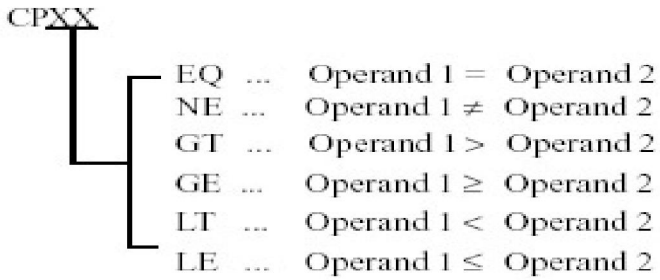


Compare and Counter

● **CPXX (Compare)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output port · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	CPXX	Variable No.	Data	EQ NE GT GE LT LE

[Function] Compares the contents of the variable in operand 1 and the value in operand 2 and if the condition is satisfied, the output turns ON. When the condition is not satisfied, the output turns OFF.



[Example 1]

```
        LET   1   10
        CPEQ  1   10   600
600     ADD   2   1
```

Assign 10 to variable 1.
If the content of variable 1 is 10, flag 600 turns ON,
If flag 600 is ON, 1 is added to variable 2.

[Example 2]

```
LET   1   2
LET   2   10
LET   3   10
CPNE  *1  *3   310
```

Assign 2 to variable 1.
Assign 10 to variable 2.
Assign 10 to variable 3.
If the variable in 2 (the content of variable 1) does not equal the content of variable 3, then output 310 turns ON. Therefore, in this example, output 310 is OFF.

Compare and Counter

-LET : Set variable and value
-CLR : Clear value in variable
-ADD (SUB) : Increase up for value
-CPEQ : Check condition and change status of FLAGS.
-TAG : Loop of process
-GOTO : Jump to Loop assign

Flags เป็น relay ช่วยภายใน สำหรับช่วยในการเขียนโปรแกรม โดย Flag จะมีสถานะแค่ “0” และ “1” เท่านั้น ซึ่งจะมีประโยชน์ในการเช็คค่าสถานะของ Flag ตามเงื่อนไข และเชื่อมโยงเงื่อนไขดังกล่าวไปยังโปรแกรมอื่นๆ Flag มีทั้ง แบบ Global & Local

Variable

What are variables?

Variables คือ register ที่ใช้ในการเก็บค่าตัวเลขต่างๆ โดยแบ่งเป็นสามประเภท

1. Integer variables (I) เก็บ ตัวเลขที่เป็นจำนวนเต็ม (1,2,0,-1)
2. Real variables (R) เก็บ ตัวเลขที่เป็นจำนวน ทศนิยม (0.265,-0.128)
3. String variables (S) เก็บ ตัวเลขและตัวอักษร(AB13F,!10024FCB,#105CBE)

การเลือกใช้ Variables นั้นผู้ใช้จะต้องทราบว่าตัวเลขที่จะเก็บค่าเป็นค่าแบบใด จากนั้นเลือกใช้ชนิดของ Variable ได้ถูกต้องตามชนิดของข้อมูล รวมไปถึงการใช้งานต้องการให้มีการเชื่อมโยงข้อมูลกันของแต่ละโปรแกรมด้วยก็ต้องใช้ชนิดและกลุ่มของข้อมูลที่เป็นแบบ Global

Global and Local

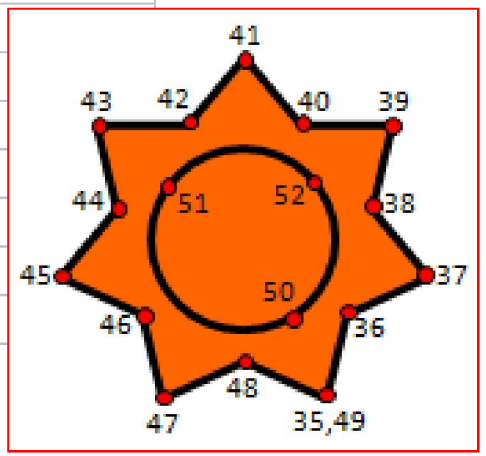
คุณสมบัติที่สำคัญข้อหนึ่งของภาษา SEL คือสามารถที่จะทำงานหลายๆ โปรแกรมพร้อมกันได้ (Multitask Program), Global จะเป็นค่าที่ทุกโปรแกรมรู้จัก แต่ Local จะรู้จักเฉพาะโปรแกรมนั้นๆยกตัวอย่างเช่นที่โปรแกรมที่หนึ่ง มีการประกาศตัวแปรInteger number200 และตัวแปร Integer number1 จะเห็นว่า Integer ที่ 200 จะเป็นตัวแปรชนิด Global ซึ่งเป็นตัวแปรที่โปรแกรมทุกโปรแกรมรู้จัก เมื่อมีการเปลี่ยนแปลงค่าที่ตัวแปรInteger200 ค่าของตัวแปร Integer200 ในโปรแกรมอื่นๆ จะเปลี่ยนตามไปด้วย

สำหรับ Integer1 เป็นตัวแปรแบบ Local เมื่อมีการเปลี่ยนแปลงค่าที่ตัวแปร Integer1จะไม่ส่งผลไปยังโปรแกรมอื่น จะถือว่า Integer1ในโปรแกรมที่หนึ่งและโปรแกรมอื่นๆ เป็นตัวแปรคนละตัวกัน ทุกโปรแกรมสามารถที่จะประกาศ ตัวแปร Integer1ได้ รายละเอียดว่า Global และ Local แต่ละ controller มีอะไรบ้างให้ดูในNumeral.

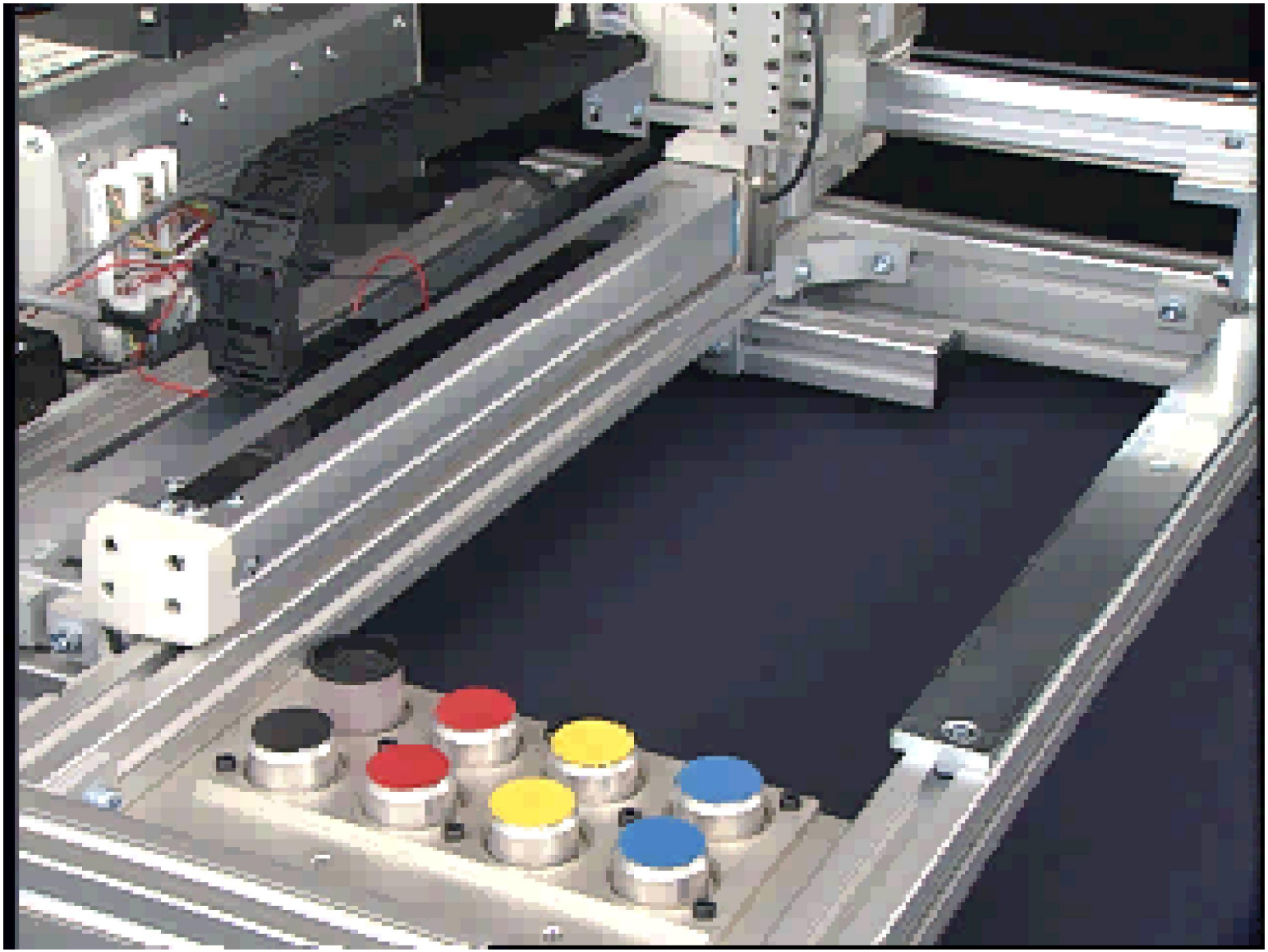
Example6 Compare & Counter

Command : LET + CLR + ADD + CPEQ + TAG + GOTO

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1					VEL	50			
2					ACC	0.3			
3					DCL	0.3			
4									
5					LET	200	0		
6					CLR	200	200		
7									
8					MOVL	90			
9					WTON	1			SW. start MC
10									
11					TAG	1			
12					PATH	35	49		
13					MOVL	50			
14					CIR2	52	52		
15					ADD	200	1		
16					CPEQ	200	3	900	
17			N	900	GOTO	1			
18					MOVL	90			
19					EXIT				



Application Compare and CNT



Program Control and Multi tasking

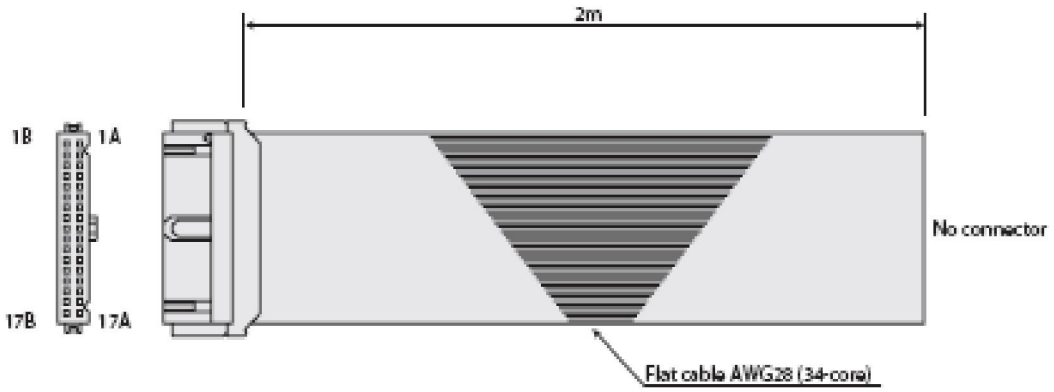
1. By I/O Selecting (Look at an appendix)
2. By command (EXPG,ABPG,TIMC,HOLD,SSPG,RSPG)
3. By Auto start PG. (Set up in Parameter)

Multi tasking : Method for execution multi program on the same time. Can be apply to some command example "EXPG and ABPG and TIMC "

Multi tasking : 1.Safety Concept
2. Transfer Condition
3. Transfer

Multi tasking

Program Control By I/O Cable



No.	Color	Wiring	No.	Color	Wiring
1A	Brown 1		9B	Gray 2	
1B	Red 1		10A	White2	
2A	Orange 1		10B	Black 2	
2B	Yellow 1		11A	Brown-3	
3A	Green 1		11B	Red 3	
3B	Blue 1		12A	Orange 3	
4A	Purple 1		12B	Yellow 3	
4B	Gray 1	Flat	13A	Green 3	Flat
5A	White 1	cable	13B	Blue 3	cable
5B	Black 1	crimped	14A	Purple 3	crimped
6A	Brown-2		14B	Gray 3	
6B	Red 2		15A	White 3	
7A	Orange 2		15B	Black 3	
7B	Yellow 2		16A	Brown-4	
8A	Green 2		16B	Red 4	
8B	Blue 2		17A	Orange 4	
9A	Purple 2		17B	Yellow 4	

Pin number	Classification	Port No.	Program mode
1A	P24		24V input
1B		016	Program No. Select 1
2A		017	Program No. Select 2
2B		018	Program No. Select 4
3A		019	Program No. Select 8
3B		020	Program No. Select 10
4A		021	Program No. Select 20
4B		022	Program No. Select 40
5A		023	CPU reset
5B		000	Start

PSEL

Program Control (By Command)

Start other Program

● **EXPG (Start Another Program)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	EXPG	Program No.		CP

[Function] Starts another program and processes it in parallel.
When that program (task) has been started, the port and flag in the post section is output.

[Example 1]
```
EXPG   10          Start program number 10.
```

[Example 2]
```
LET  1     10      Assign 10 to variable 1.
EXPG *1            Start variable 1 (content 10)  program.
```

Program Control (By Command)

Stop other Program

● **ABPG (Stop Other Program)**

Expansion condition (AND · OR)	Input condition (I/O · Flag)	Command			Post (Output · Flag)
		Command	Operand 1	Operand 2	
Optional	Optional	ABPG	Program No.		CP

[Function] Forces the other program being executed in operand 1 to end.
When that progran (task) is forced to end, the port and flag in the post section is output.

Note: The ABPG command stops the program once the command being executed is completed.

```
[Example 1]   ABPG   10          Stop program No. 10.

[Example 2]   LET    1    10     Assign 10 to variable 1.
              ABPG   *1          Stop variable 1 (content 10) program.
```

ABPG
TIMC
Remark : 2 command should using together

Program Control By command (HOLD)

HOLD : Pause Program at current Step

(Put Command on the top of Program table)

SSPG : Pause Program at current Step

(Have to use Multi tasking concept)

RSPG : Resume Program from current Step

(Have to use Multi tasking concept)

SSPG & RSPG : Should be use together and in Multi tasking concept

Control Program by Command.

Program No.1 (Main)

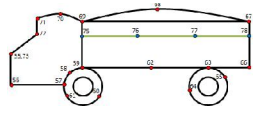
No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2					VEL	50			
3					ACC	0.3			
4					DCL	0.3			
5									
6					EXPG	2			
7									
8					TAG	1			
9					PATH	55	73		
10					GOTO	1			
11									

Program No.2 (Safety)

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst
1								
2					WTON	15		
3								
4					ABPG	1	1	
5					TIMC	1		
6								
7					EXIT			
8								

Remark :

-Input 15 connect to safety SW. after it turn on will stop to Main PG.



Example Program Pause PG.

Program No.1 (Main)

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2					VEL	50			
3					ACC	0.3			
4					DCL	0.3			
5									
6					EXPG	2			
7									
8					TAG	1			
9					PATH	55	73		
10					GOTO	1			
11									

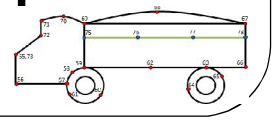
Program No.2 (Pause)

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2
1							
2							
3					TAG	1	
4				1	SSPG	1	1
5				2	RSPG	1	1
6					GOTO	1	
7							
8							

Remark :

-Input 1 and 2 use for Pause PG and Release PG.

-Output port (302 - 307) can't not hold or pause step



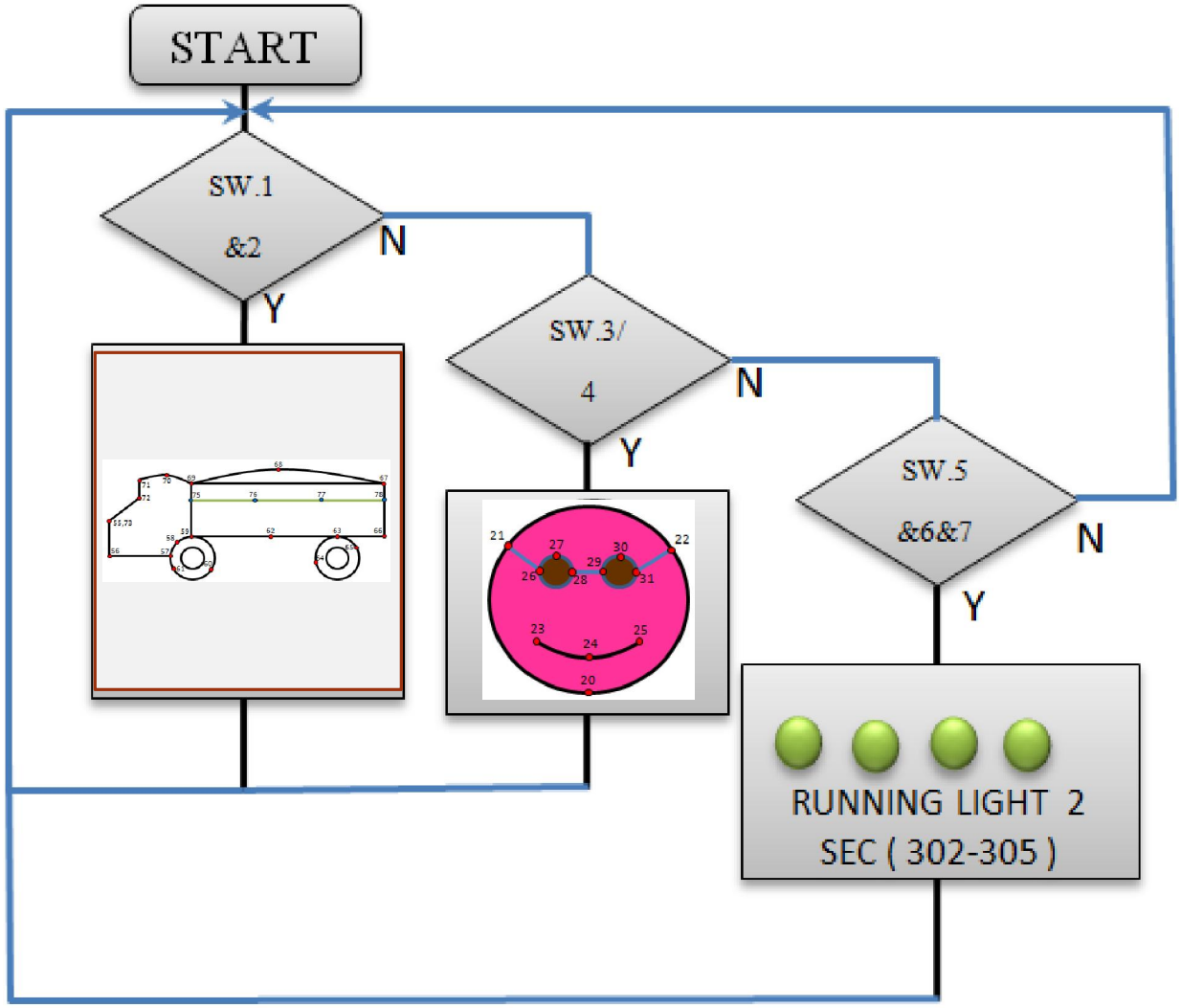
Example7 Program control (Salty concept)

PROGRAMMING & APPLICATION

Technical support Team
SUS BKK

Model Selection.

How to program ?

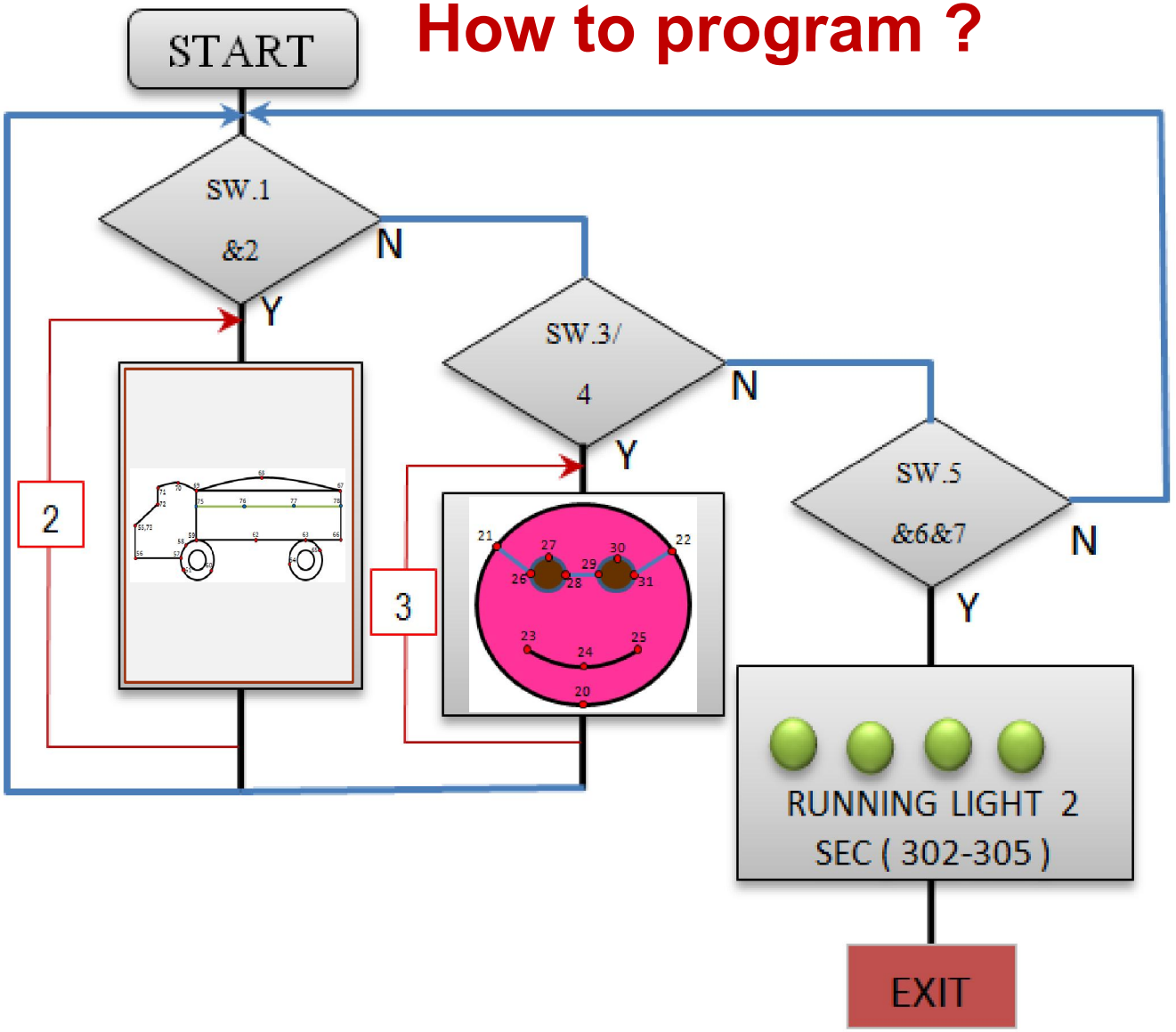


No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									

Model Selection and Compare and CNT.

How to program ?



No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									

Offset distance

OFST : Robot will movement to position but offset distance as value setting

- OFST (Set offset)

Extension condition (LD, A, O, AB, OB)	Input condition (I/O, flag)	Command, declaration			Output (Output, flag)
		Command, declaration	Operand 1	Operand 2	
Optional	Optional	OFST	Axis pattern	Offset value	CP

[Function] Reset the target value by adding the offset value specified in operand 2 to the original target value when performing the actuator movement specified in operand 1.
The offset is set in mm, and the effective resolution is 0.001 mm.
A negative offset may be specified as long as the operation range is not exceeded.
An OFST command is processed with respect to soft axes before a BASE shift.

(Note) An OFST command cannot be used outside the applicable program. To use OFST in multiple programs, the command must be executed in each program.
An OFST command cannot be used with MVPI, MVLI and MVDI commands.

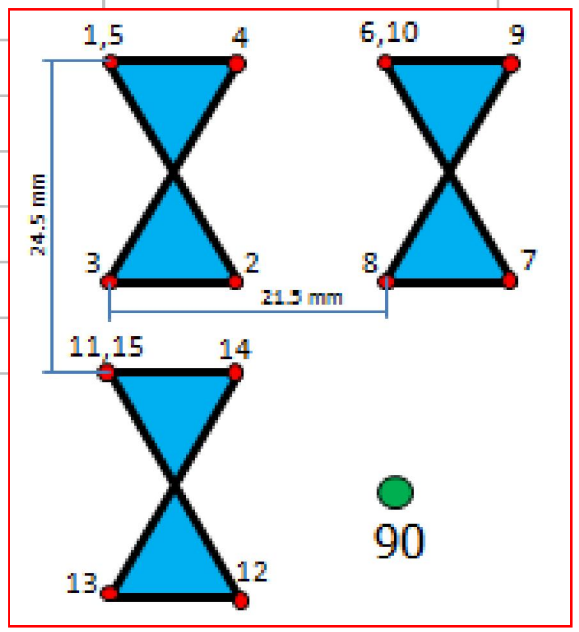
[Example 1]

```
OFST   10   50   Add 50 mm to the specified position of axis 2.
  :
OFST   10   0    Return the offset of axis 2 to 0.
```

*** Don't for get clear OFST each axis to normal condition by "OFST 111 0" ***

Example8 Offset distance concept

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2					VEL	50			
3					ACC	0.3			
4					DCL	0.3			
5									
6					PATH	1	5		
7									
8					OFST	01	21.5		
9									
10					PATH	1	5		
11									
12					MOVL	90			
13									
14					EXIT				
15									



Out put on during movement.

POTP : Set PATH output type

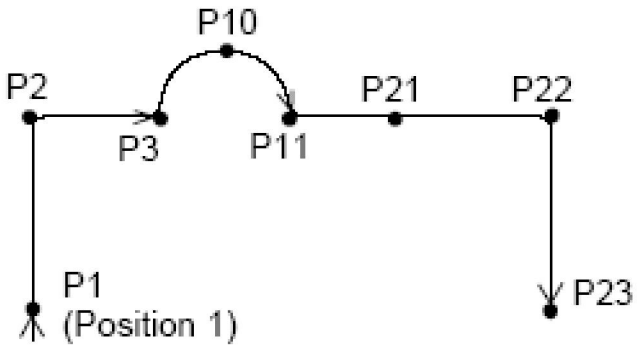
เป็นการสั่งให้ output ออกขณะที่ robot กำลังเคลื่อนที่

(On the fly out put)

- POTP = 0 , output จะ ON ก็ต่อเมื่อ robot ทำงาน move ครบทุก position.
- POTP = 1, Out put จะ ON ในขณะที่ robot กำลังเคลื่อนที่ผ่านตำแหน่งต่าง ๆ และ number out put ที่ ON ก็จะเปลี่ยนแปลงแบบเพิ่มขึ้นใน ทุก ๆ Position ที่ robot เคลื่อนที่ผ่าน

** คำสั่ง POTP ใช้กับกลุ่มคำสั่งแบบ continues เท่านั้น**

Move and On output



[Example 1]

```
(POTP = 1)
POTP     1
 |
 |
 |
PATH     1      3      308
ARC2     10     11     311
PATH     21     23     312
 |
 |
 |
 |
```

Output field	Timing
308	Turn ON as P1 approaches.
309	Turn ON as P2 approaches.
310	Turn ON as P3 approaches.
311	Turn ON as P11 approaches.
312	Turn ON as P21 approaches.
313	Turn ON as P22 approaches.
314	Turn ON when P23 operation is complete.

[Example 2]

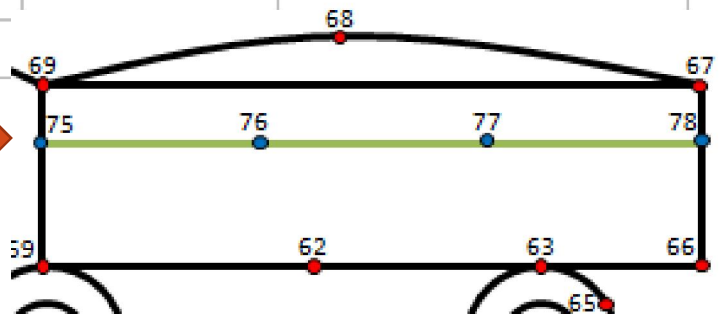
```
(POTP = 0)
PATH     1      3      308
ARC2     10     11     311
```

Output field	Timing
308	Turn ON as P3 approaches.
311	Turn ON as P11 approaches.
312	Turn ON when P23 operation is complete.

Example 9 POTP

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2					VEL	50			
3					ACC	0.3			
4					DCL	0.3			
5									
6					MOVL	90			
7					POTP	1			
8									
9					PATH	75	78	302	
10					MOVL	90			
11									
12					BTOF	302	307		
13					EXIT				
14									

Start Point



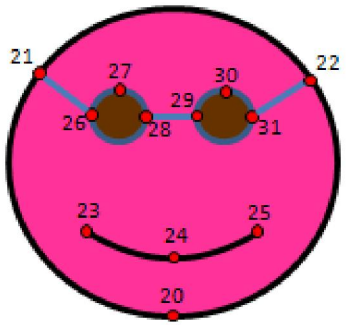
Pointer to data

To made the programming easy, we can use pointer to entry the variable data

Example

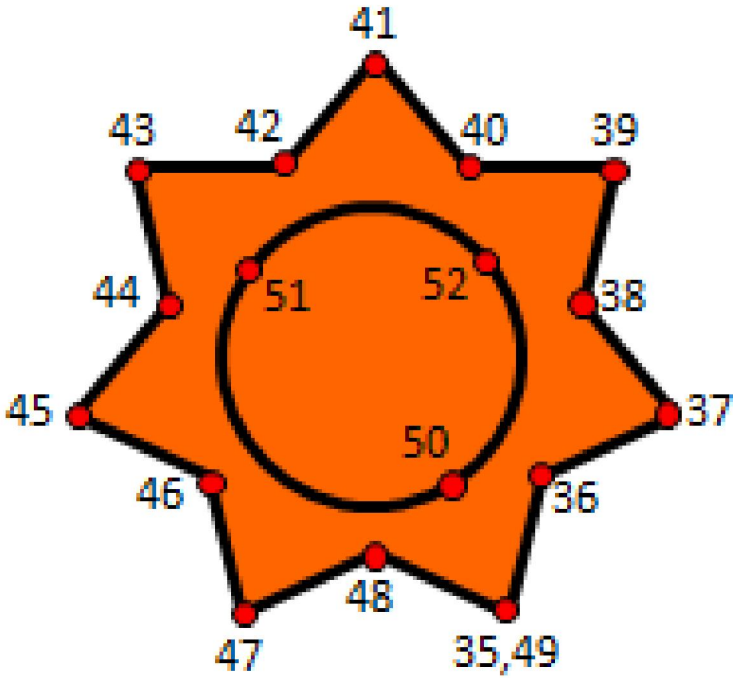
```
LET     201   23    ( Assign value 23 into 201)
MOVL    *201
```

It's mean, move the actuator to position number 23 by using * (Pointer)



Try to change robot move to other point

Pointer Application



Try to make a Program move the robot form Position No.35 – No.49 by apply Pointer and Compare

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									

Teaching by I/O Cable and Command

Useful Command

PRED : อ่านค่า ณ. ตำแหน่งปัจจุบัน (Axis pattern) ที่ robot หยุดอยู่ แล้วนำไปเก็บไว้ที่ position ที่กำหนด

Exp : PRED 11 4 , อ่านค่าปัจจุบัน ทั้ง 2 Axis แล้วนำค่าที่อ่านได้ ไปเก็บไว้ที่ Position No.4 ในตาราง Position.

PGET : อ่านค่าจาก axis และ position ที่กำหนด (Axis No.) ไปเก็บไว้ที่ตัวแปรพิเศษ 199 โดยอัตโนมัติ

Exp : PGET 1 5 , อ่านค่าจาก axis1 position 5 มาเก็บไว้ที่ตัวแปร 199

PPUT : อ่านค่าจากตัวแปรพิเศษ 199 แล้วนำไปเก็บไว้ที่ Axis (Axis No.) และ position ที่กำหนด

Exp : PPUT 1 2 , อ่านค่าจากตัวแปร 199 แล้วไปเก็บไว้ที่ axis No.1 position 3 .ในตาราง Position.

ตัวแปร 199 เป็นตัวแปรพิเศษ แบบ Real ใช้คู่กับคำสั่ง PPUT & PGET

Make a new Postion data.

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									

1. สร้างตำแหน่งที่ 100 (ทั้ง 2 axis) จากตำแหน่งปัจจุบัน ที่ Off servo เลื่อน
2. สร้างตำแน่งที่ 101 โดยให้ เพิ่มค่าจากตำแหน่งที่ 100 ขึ้น 5 mm / X , 10mm / Y (ทั้ง 2 axis) ตามลำดับ Command : PGET, PPUT, ADD,SUB

Make a new Postion data.

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									

1. สร้างตำแหน่งที่ 100 (ทั้ง 2 axis) จากตำแหน่งปัจจุบัน ที่ Off servo เลื่อน
2. สร้างตำแน่งที่ 101 โดยให้ เพิ่มค่าจากตำแหน่งที่ 100 ขึ้น 5 mm / X , 10mm / Y (ทั้ง 2 axis) ตามลำดับ Command : PGET, PPUT, ADD,SUB

Teaching by I/O Cable and Command

● PRDQ (Read current axis position (1 axis direct))

Extension condition (LD, A, O, AB, OB)	Input condition (I/O, flag)	Command, declaration			Output (Output, flag)
		Command, declaration	Operand 1	Operand 2	
Optional	Optional	PRDQ	Axis number	Variable number	CP

[Function] Read the current position of the axis number specified in operand 1 to the variable specified in operand 2.
The current position can be obtained more quickly than when a PRED command is used.
The current position of a synchronized slave axis can also be read.

[Example] PRDQ 2 100 Read the current position of axis 2 to variable 100.

● PCPY (Copy position data)

Extension condition (LD, A, O, AB, OB)	Input condition (I/O, flag)	Command, declaration			Output (Output, flag)
		Command, declaration	Operand 1	Operand 2	
Optional	Optional	PCPY	Position number	Position number	CP

[Function] Copy the position data specified in operand 2 to the position number specified in operand 1.

[Example 1] PCPY 20 10 Copy the data of position No. 10 to position No. 20.

[Example 2]
LET 1 20 Assign 20 to variable 1.
LET 2 10 Assign 10 to variable 2.
PCPY *1 *2 Copy the data of the content of variable 2 (position 10) to the content of variable 1 (position 20).

Palletizing Function

Useful Command

BGPA : PAPI : PAPS : PAPN : EDPA : PSET : PMVL : PINC

BGPA : ประกาศชื่อ Pallet โดยชื่อ จะเป็น ตัวเลข

PAPI : ขนาดของ Pallet เป็น Row x Colum.

PAPS : ตำแหน่ง แรกของ Pallet นั้น ๆเพียงตำแหน่งเดียว เช่น 80,81,82 เรียงลำดับกัน

PAPN : Pallet pattern มีสองแบบในการ movement

EDPA : จบการประกาศ Pallet.

PSET : กำหนดจุดเริ่มต้นในการเคลื่อนที่ไปยัง Pallet นั้น ๆ

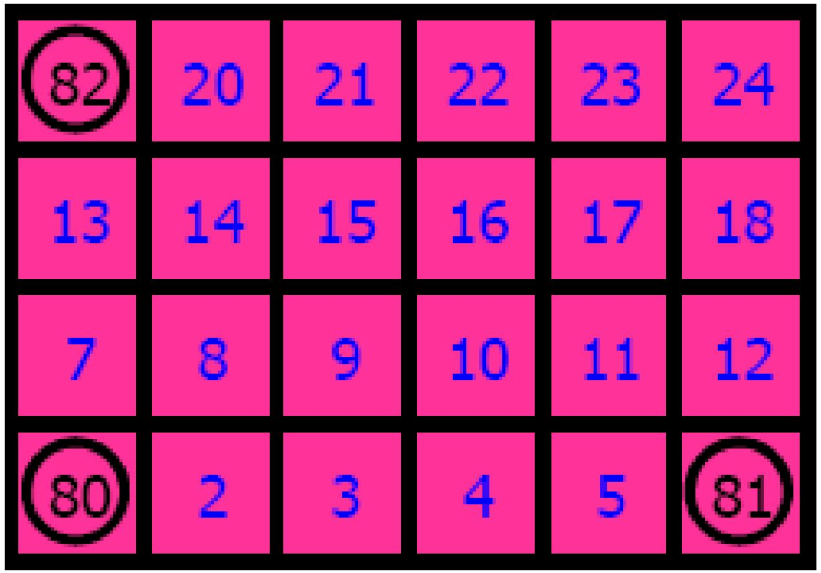
PMVL : MOVL of Pallet.

PINC : เพิ่มค่า Counter ของ Pallet พร้อมทั้ง check Pallet ตำแหน่งสุดท้าย แล้ว off Flag ที่กำหนดไว้

Remark : โดยในหนึ่งโปรแกรมสามารถประกาศได้ 10 pallets.

Three Point Teaching

In put data only Position No.80 but teaching all 3 Point.



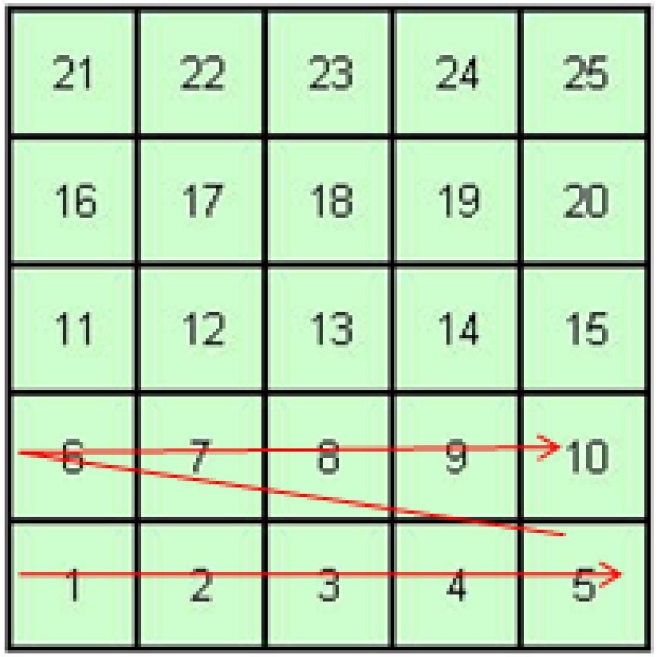
Teaching Position

No.80

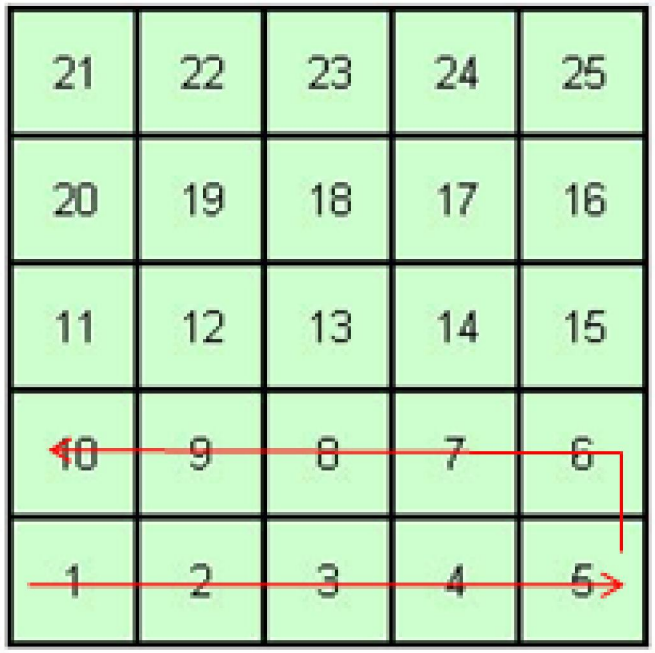
No.81

No.82

Pattern Movement



Pattern 1



Pattern2

Set up Pallet

ตัวอย่างการประกาศ Pallet

BGPA	1		ประกาศ Pallet ชื่อว่า 1
PAPI	4	6	ขนาด 4x6
PAPS	80		ใช้ Three point teaching เริ่ม จาก Position80
PAPN	1		Pattern 1
EDPA			จบการประกาศ Pallet ที่ 1

ถ้ามี Pallet มากกว่าหนึ่ง สามารถประกาศ Pallet ถัดไปในบรรทัดต่อไปได้เลย

Movement Command Pallet

```
PMVL    1
PINC    1                    900
```

คำสั่ง **PMVL** เป็นคำสั่งที่ใช้ในการ **movement robot** ไปยัง **Pallet**
จากนั้นตามด้วย **PINC** คือคำสั่งในการเพิ่มค่า **CNT** ของ **Pallet** พร้อม **compare** ค่า **CNT** ภายใน **Pallet** เมื่อถึงตำแหน่งสุดท้ายแล้วจะเปลี่ยนสถานะของ **Flag** ในช่อง **Post** จาก **“ON”** เป็น **“OFF”**

Example10 Point to Pallet

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2					VEL	300			
3					ACC	0.9			
4					DCL	0.9			
5									
6					BGPA	1			
7					PAPI	4	6		
8					PAPS	80			
9					PAPN	1			
10					EDPA				
11									
12					PSET	1	1		
13									
14					TAG	1			
15					MOVL	90			
16					PMVL	1			
17					PINC	1		900	
18				900	GOTO	1			
19									
20					EXIT				

Example11 Pallet to Pallet

No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2					VEL	300			
3					ACC	0.9			
4					DCL	0.9			
5									
6					BGPA	2			
7					PAPI	4	3		
8					PAPS	83			
9					PAPN	2			
10					EDPA				
11									
12					BGPA	3			
13					PAPI	3	4		
14					PAPS	86			
15					PAPN	1			
16					EDPA				
17									
22					TAG	1			
23					PMVL	2			
24					PINC	2			
25					PMVL	3			
26					PINC	3		900	
27				900	GOTO	1			
28									
29					EXIT				
30									

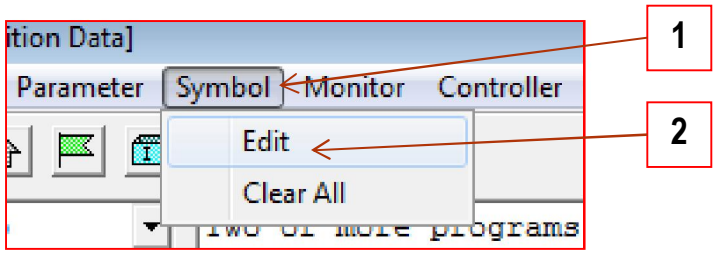
No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									

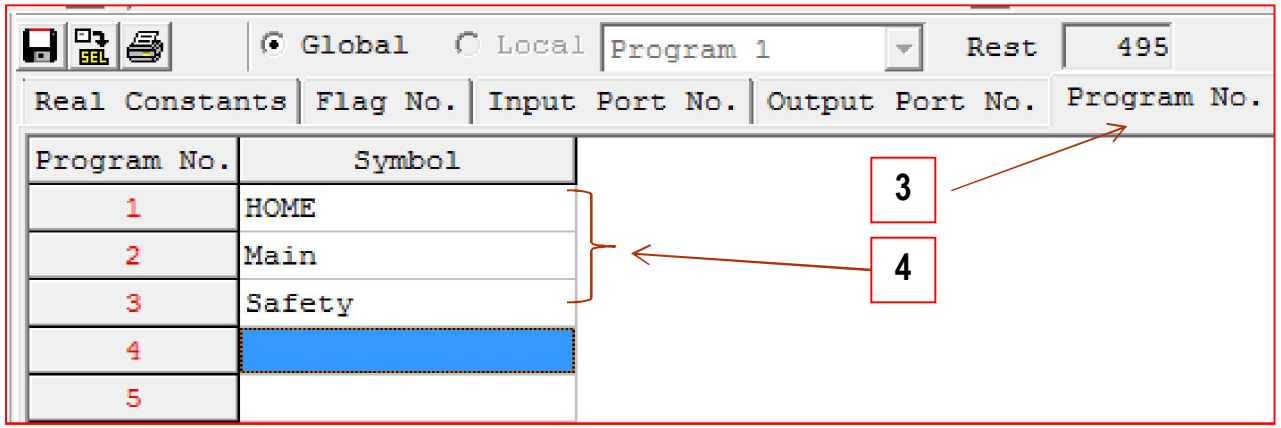
No.	B	E	N	Cnd	Cmnd	Operand 1	Operand 2	Pst	Comment
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									

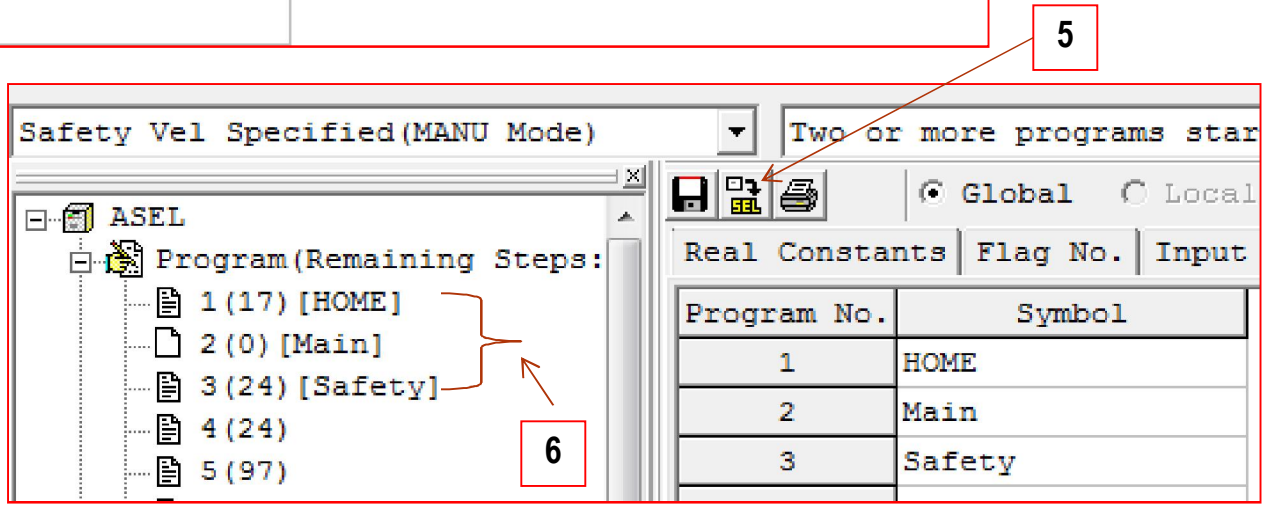
PSEL Parameter option

IO > No.10 > IO sprvs	= 0,1 (Disable & Enable IO port)
IO > No.90 > Useage SIOch0	= 0,1,2 (Monitor & Protocol select)
Common to all axes > No.1	= 00,01,10,11 (Axis pattern Enable)
Specific Axis > No.6	= 0,1 (Home Position Locate)
Specific Axis > No.7&8	= Software Limit +,-
Specific Axis > No.47	= Screw lead size (check only)
Other No.1	= Auto start Pro. No.

Set up Symbol Method.

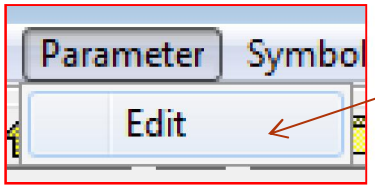






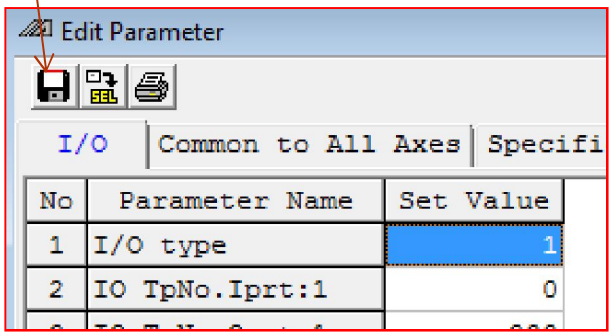
Back up All data

Parameter data Back up

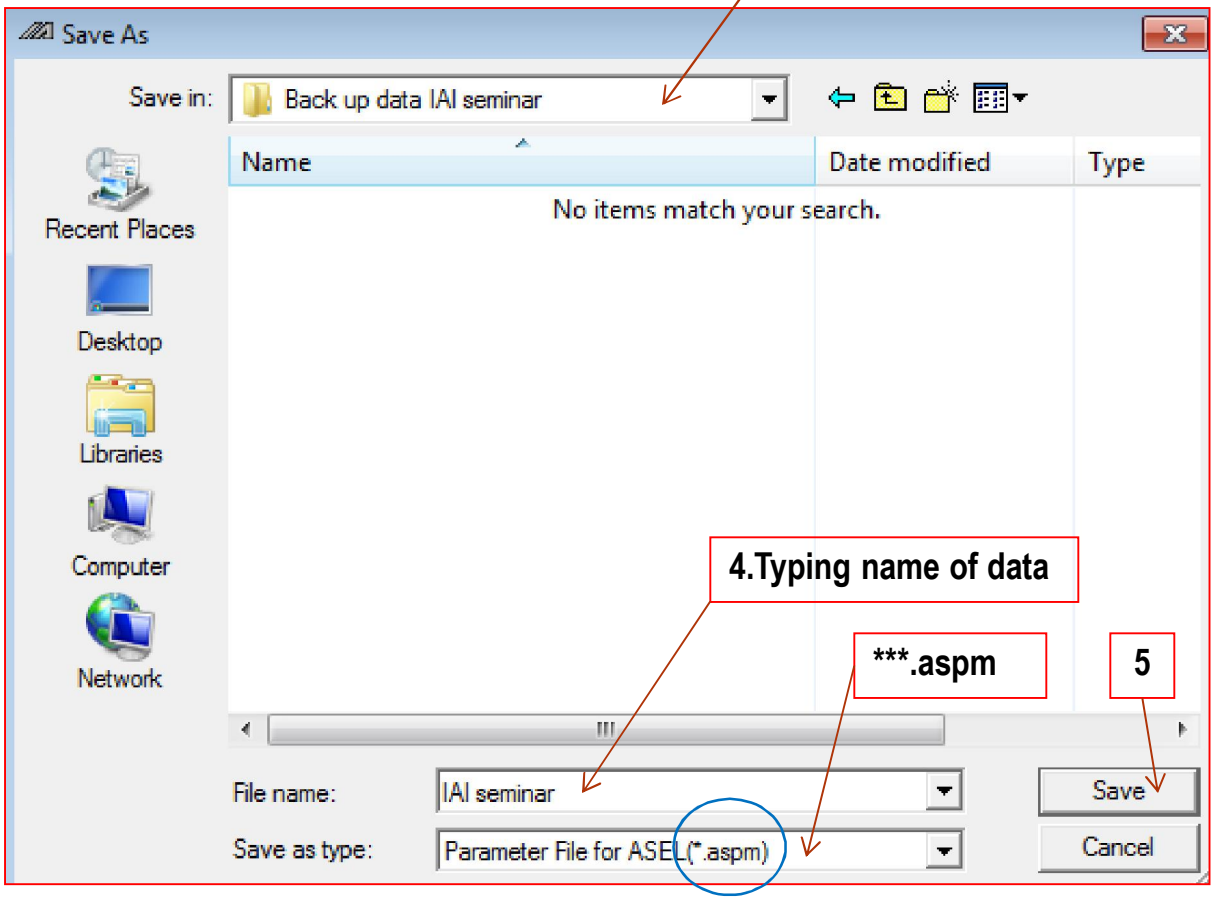


1. Edit Click

2



3. Location keeping data



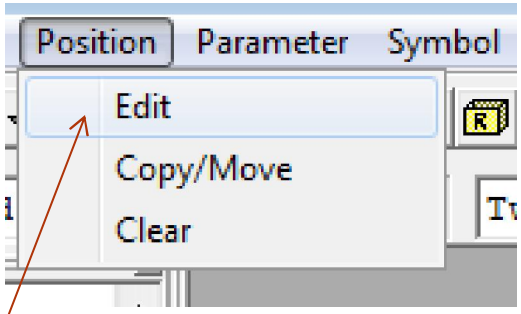
4.Typing name of data

***.aspm

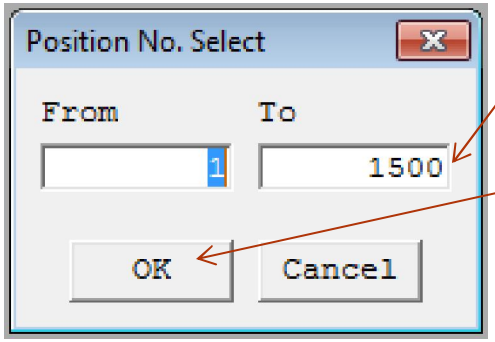
5

Back up All data

Position data Back up

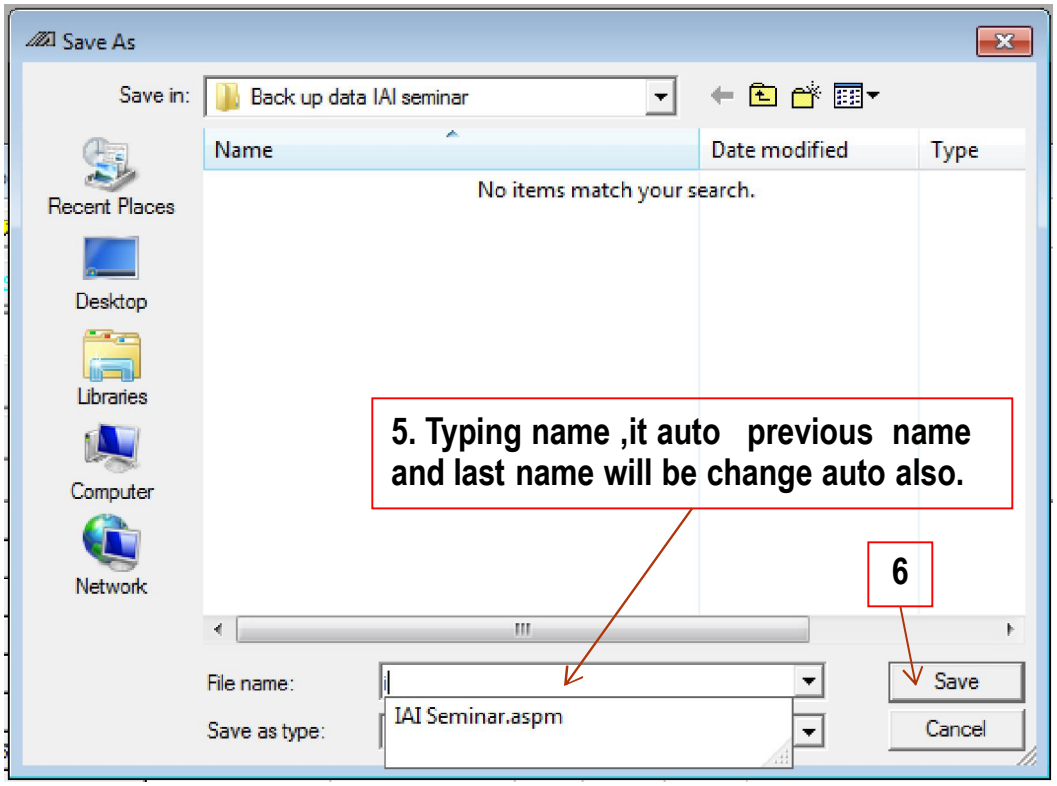


1. Edit Click



2. All of data 's type controller

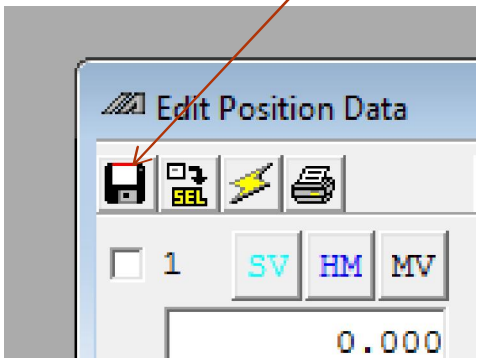
3. OK



5. Typing name ,it auto previous name and last name will be change auto also.

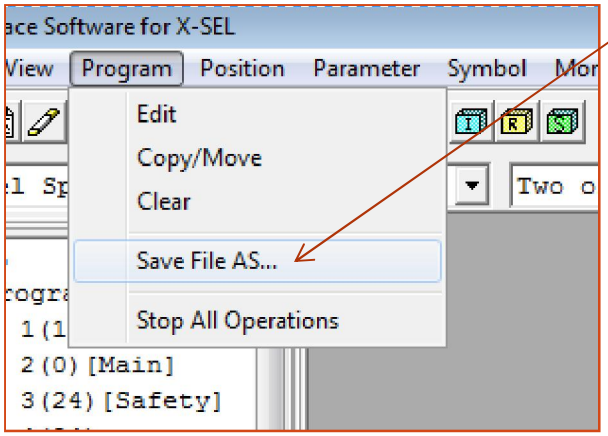
6

4. OK



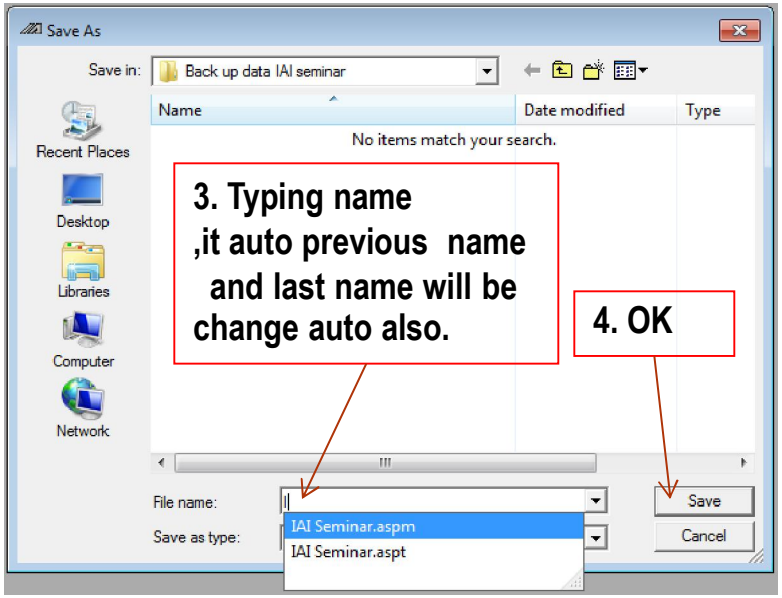
Back up All data

Program data Back up



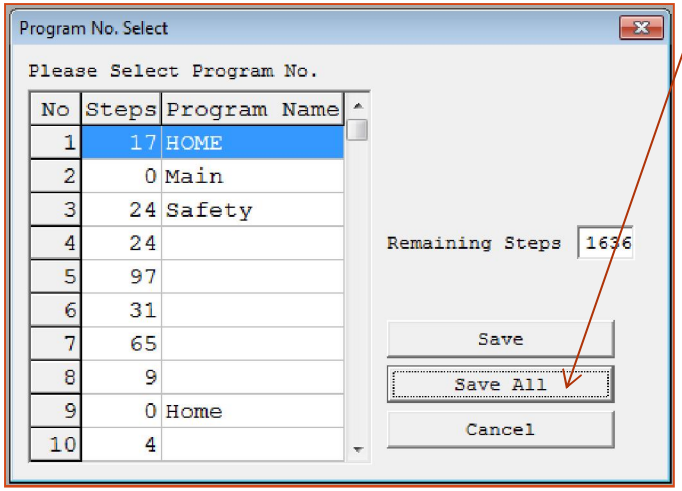
1. Edit Click

2.Save All

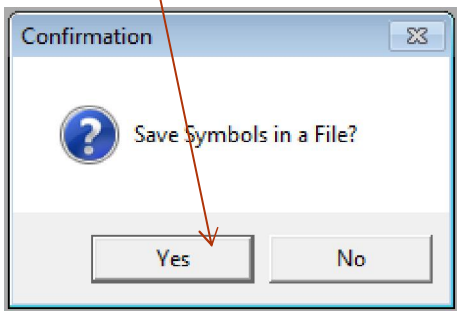


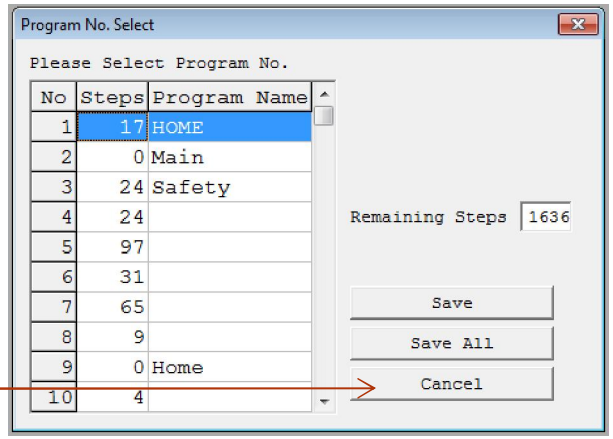
3. Typing name ,it auto previous name and last name will be change auto also.

4. OK



5. Save Symbols and do as No.3-4

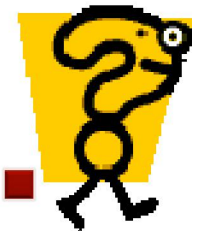




6. After finite save symbols click “Cancel”

THE END

Question...



peerawat_w@susbkk.co.th
084-3629437
pharkphoorm_k@susbkk.co.th
081-7817130