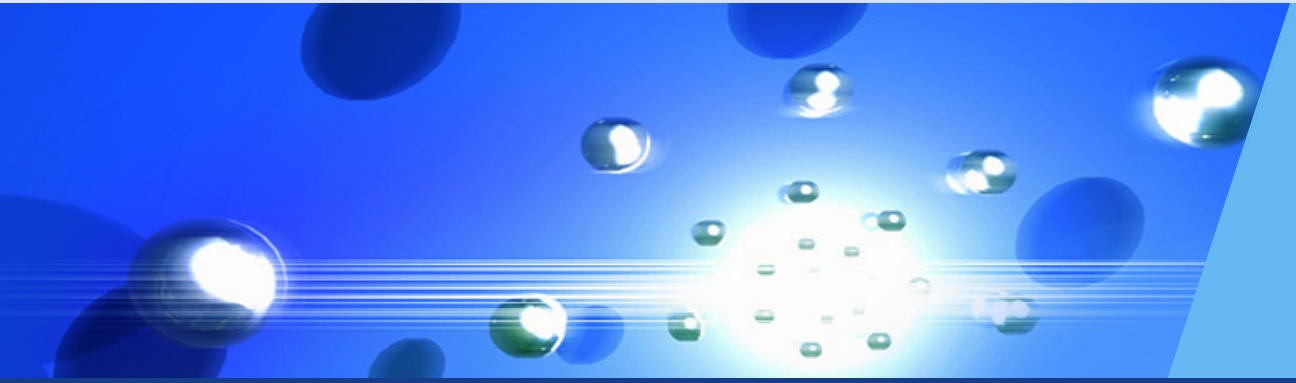
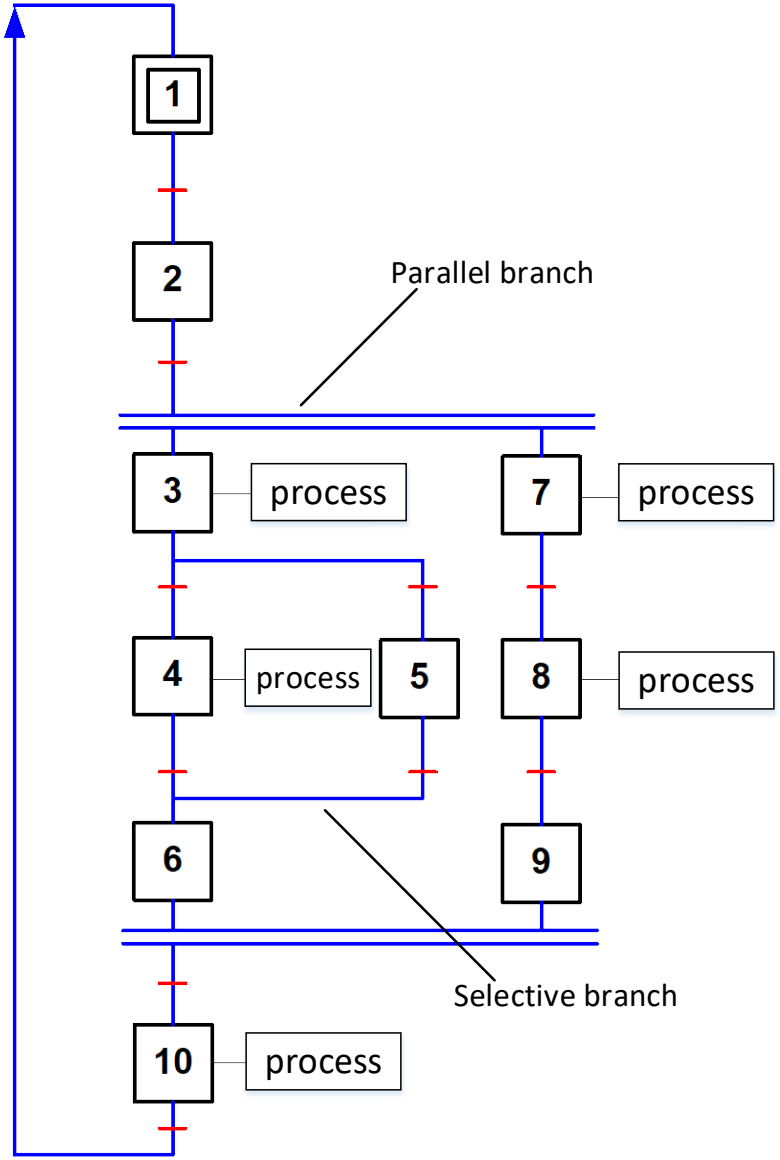


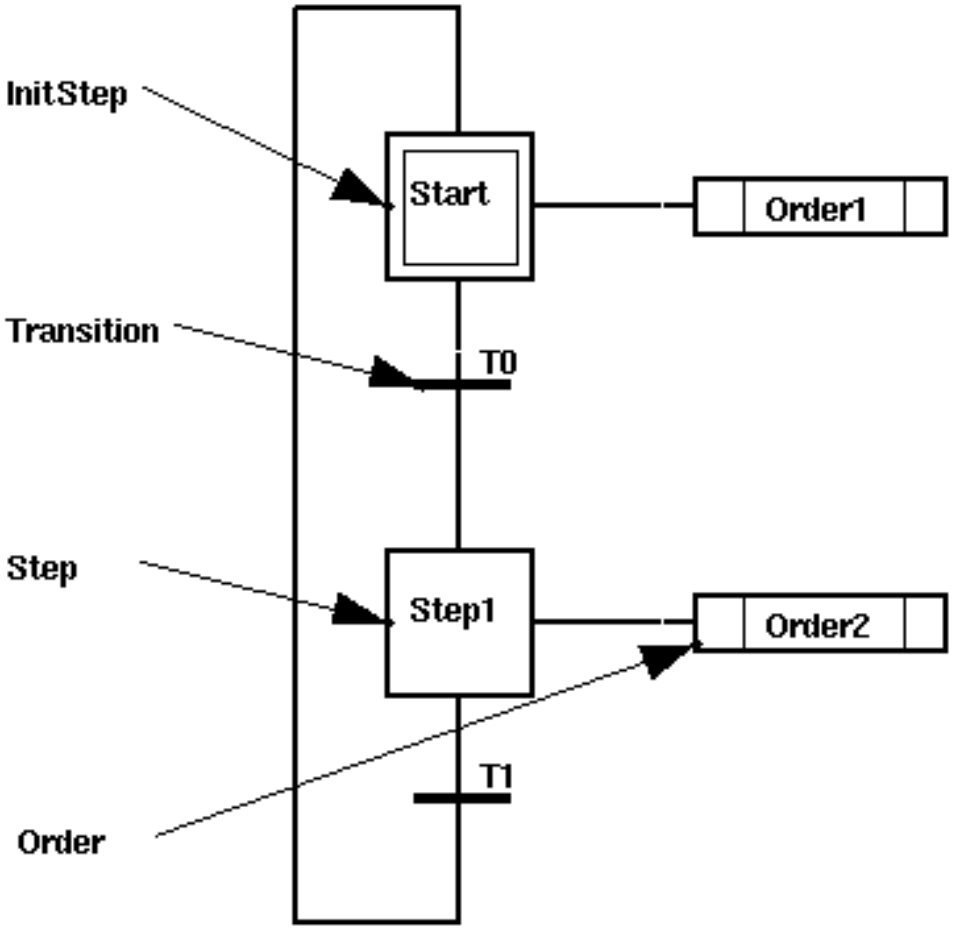
Chapter 9: PLC Programming language – STL/SFC Instructions

Introduction

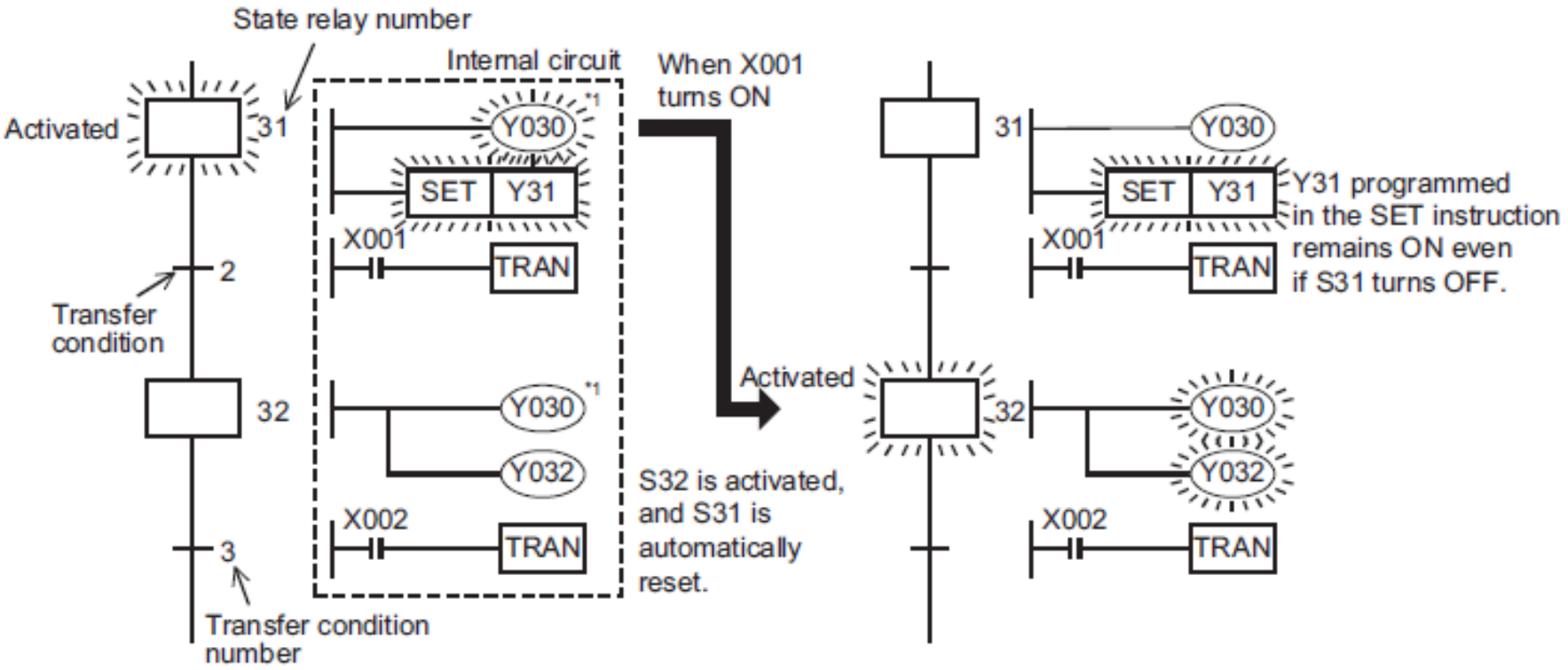
- Sequence control using the SFC (Sequential Function Chart) or STL (STep Ladder) is available in FX PLCs.
- In SFC programs, the role of each process and the overall control flow can be expressed easily based on machine operations → design a sequence easily.
- As a result, the same contents can be handled in relay ladder charts which are familiar and easy to understand.

Introduction



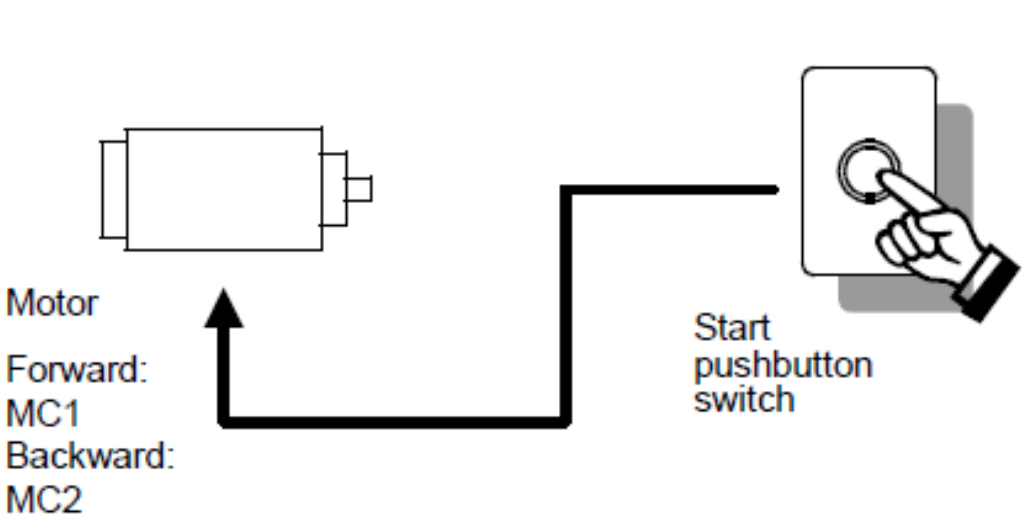


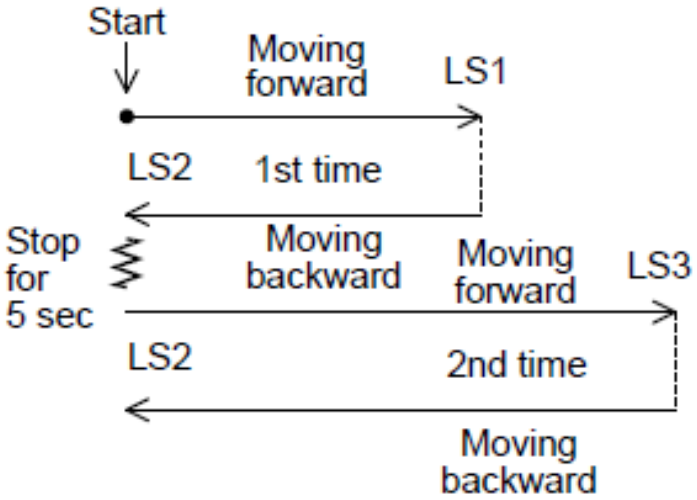
Operation



- **Output coils** can be used again in different state relays.

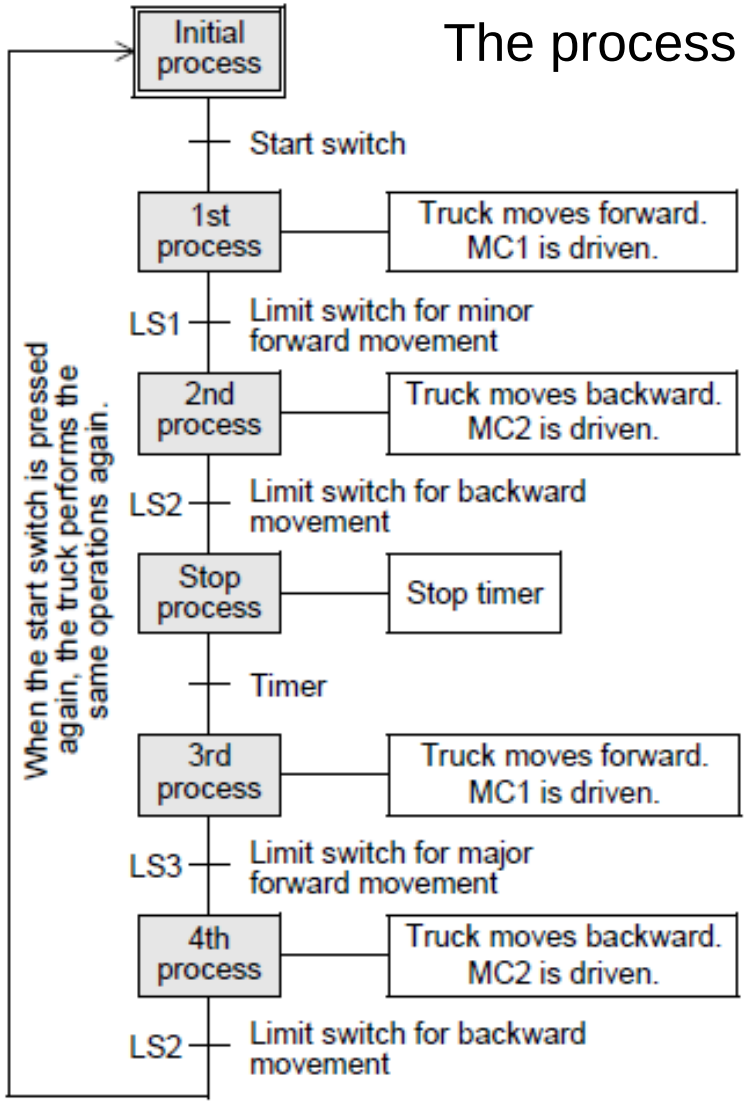
Operation example

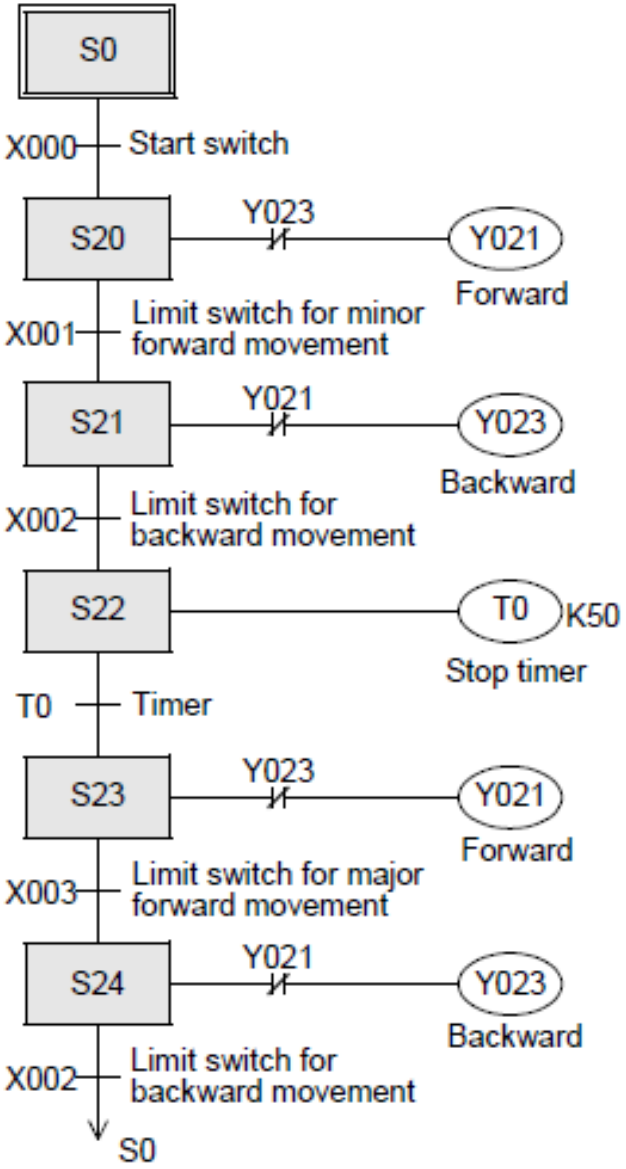




Operation example

The process drawing



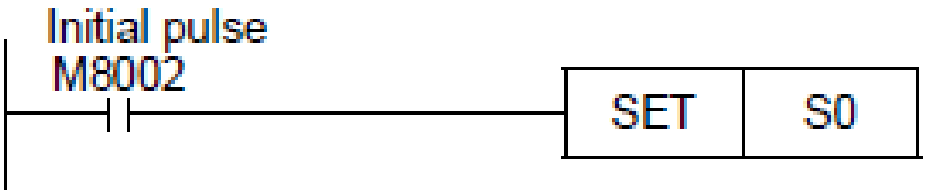


Operation example

- Assign devices of a PLC in the created process drawing.
  - Assign a state relay to a rectangle indicating a process. At this time, assign a state relay (S0 to S9) to the initial process.
  - After the first process, arbitrarily assign state relay numbers (S20 to S899) except the initial state relays. There are latched (battery backed) type state relays(S500 to S899) whose ON/OFF status is stored against power failure.
  - The state relays S10 to S19 are used for special purposes when the IST (Initial State - FNC 60) instruction is used.
  - Assign a device to each transfer condition. NO contact and NC contact are available for a transfer condition. If there are two or more transfer conditions, AND circuit or OR circuit is available.
  - Assign a device (output terminal number connected to external equipment, timer number, etc.) used for an operation performed in each process. Many devices such as timers, counters and auxiliary relays are provided in a PLC, and can be used arbitrarily.
  - If there are two or more loads such as timers and counters which are driven at the same time, two or more circuits can be assigned to one state relay.
  - When performing repeated operations or skipping some processes (jump operation), use "↳" and specify the jump destination state relay number.

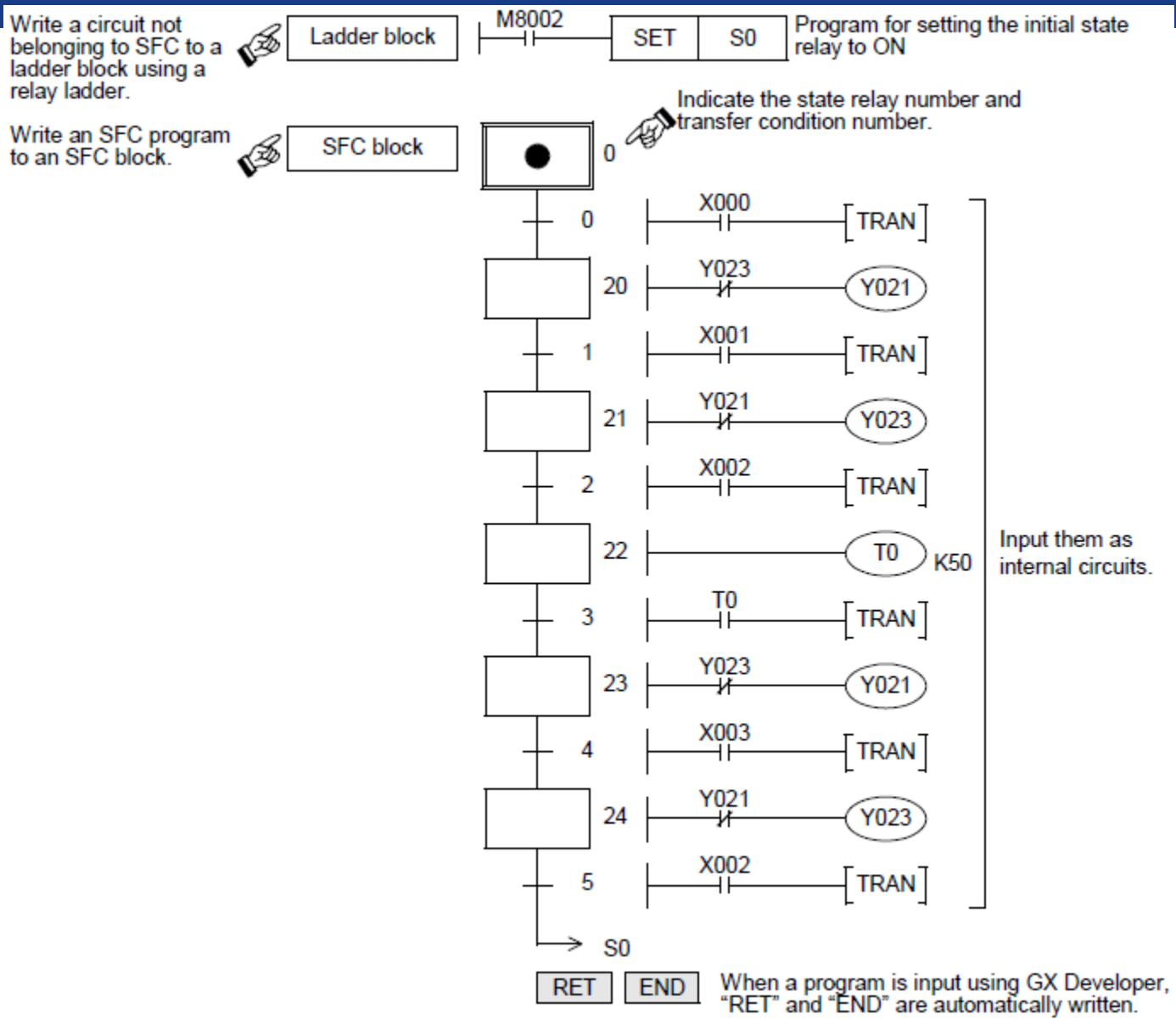
Operation example

- A circuit for setting the initial state relay to ON is required to execute the SFC program

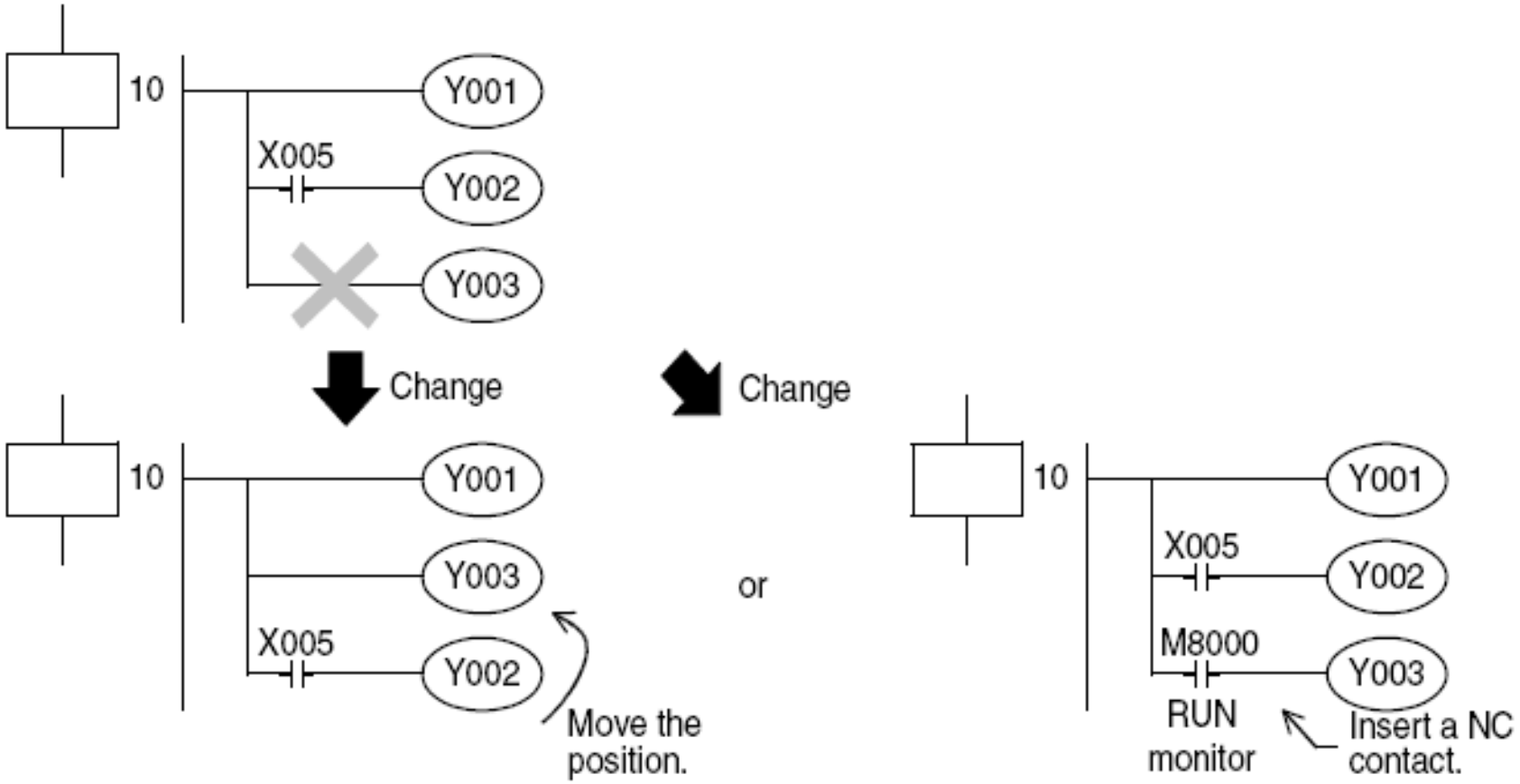


- After that, write the SFC program to an SFC block.

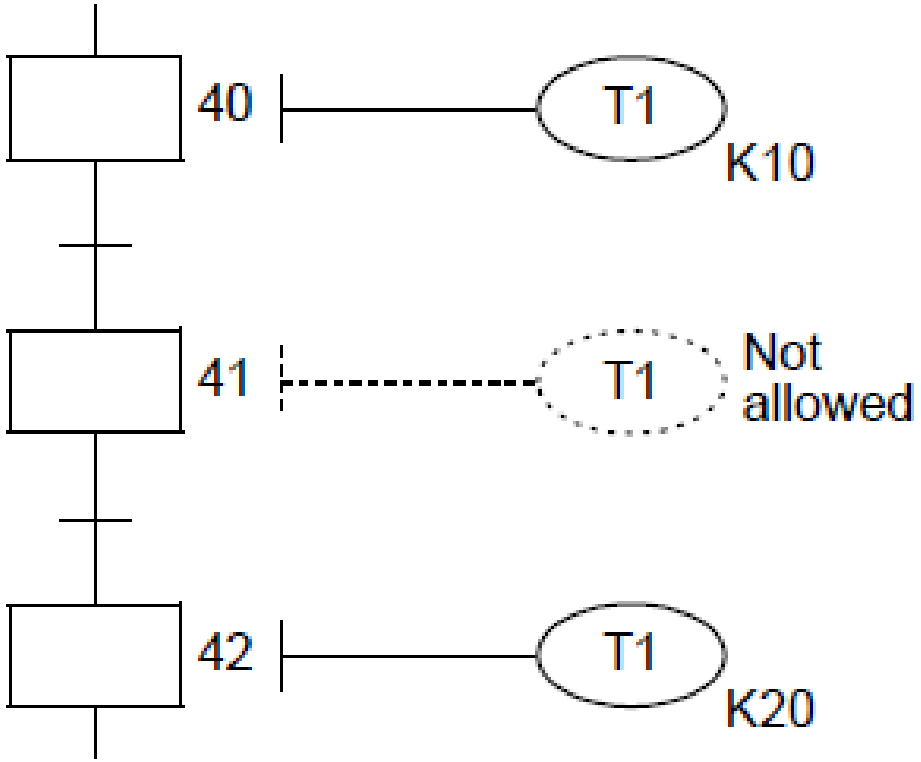
Operation example



Notes


10
Y001
X005
Y002
Y003
Change
Change
10
Y001
Y003
X005
Y002
Move the position.
or
10
Y001
X005
Y002
M8000
Y003
RUN monitor
Insert a NC contact.

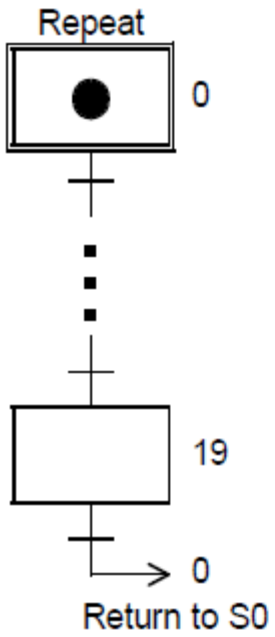
Notes

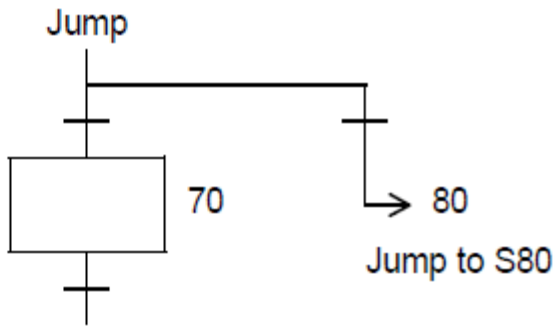


Notes

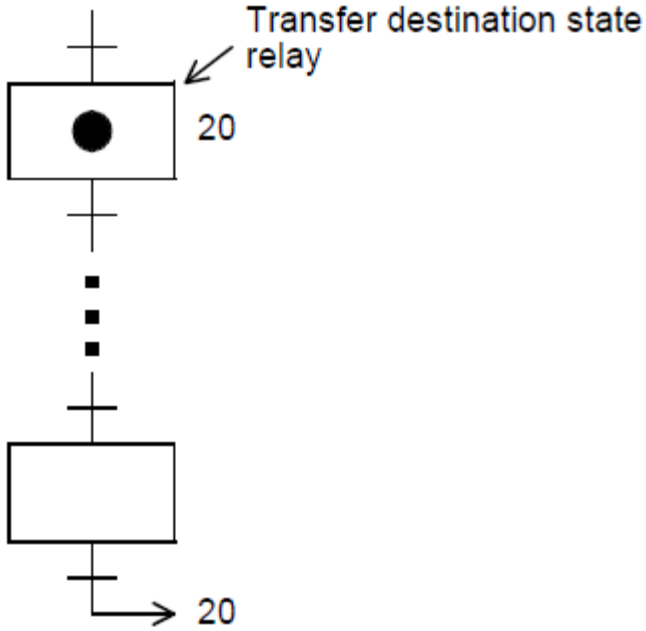
❖ Operations of "↳" and "▽"

- Use "↳" to express **transfer** to a state relay in an **upper position** (repeat), transfer to a state relay in a **lower position** (jump), or transfer to a state relay in another **separate flow**.



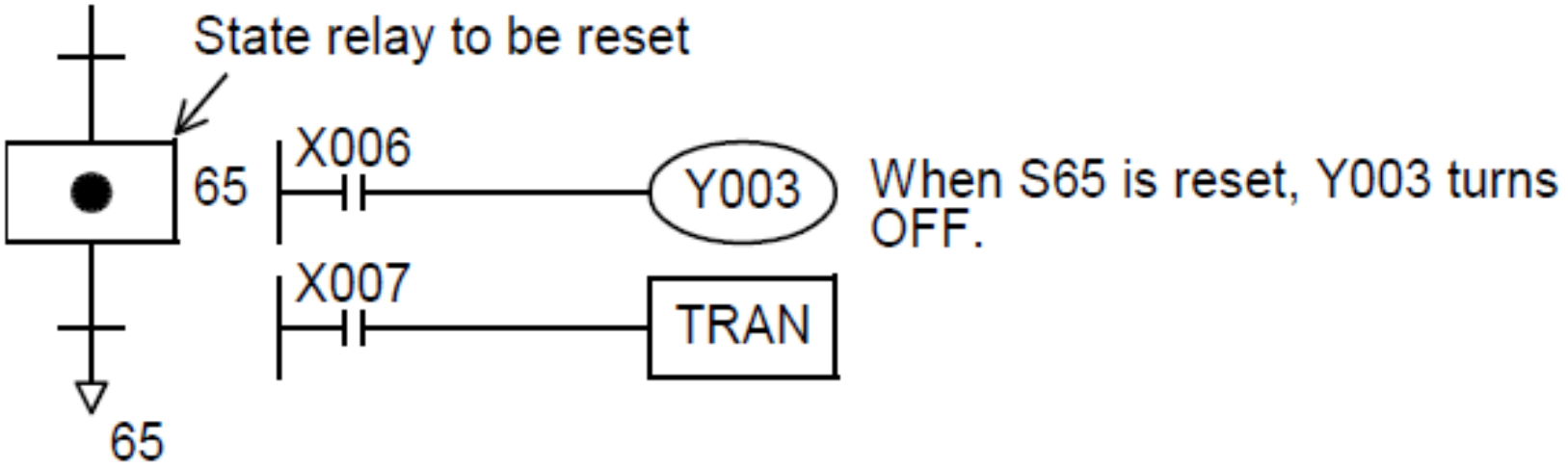


"•" is automatically displayed in the transfer destination state relay.



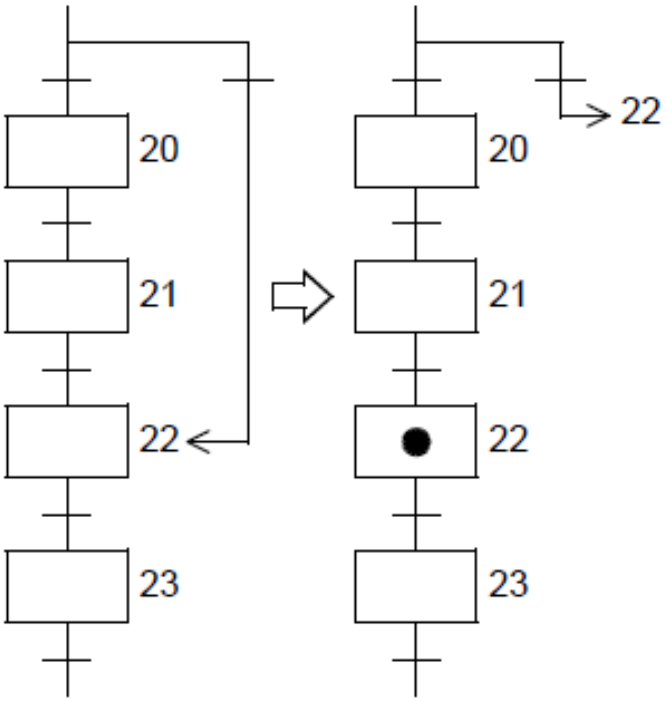
Notes

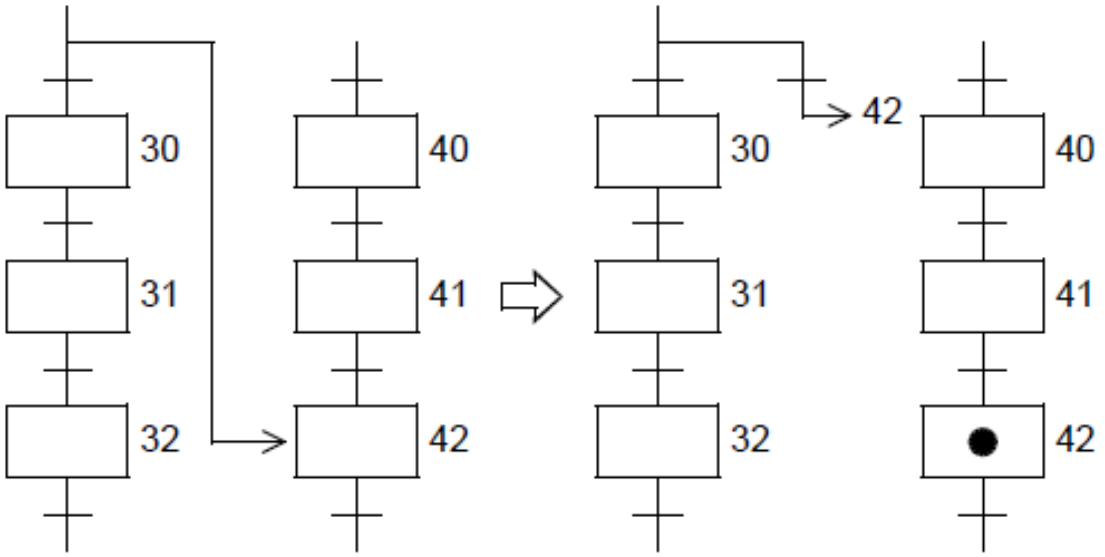
- **Use "▽" to express reset of a state relay.**



Notes

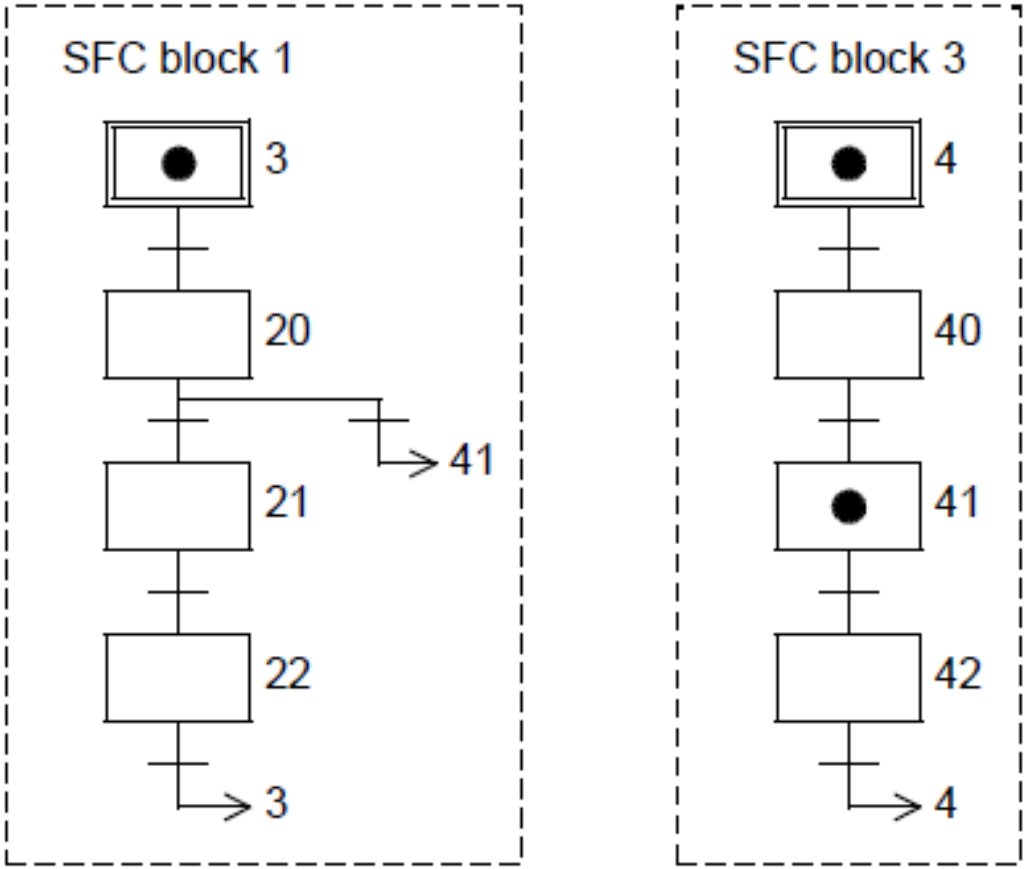
Jump





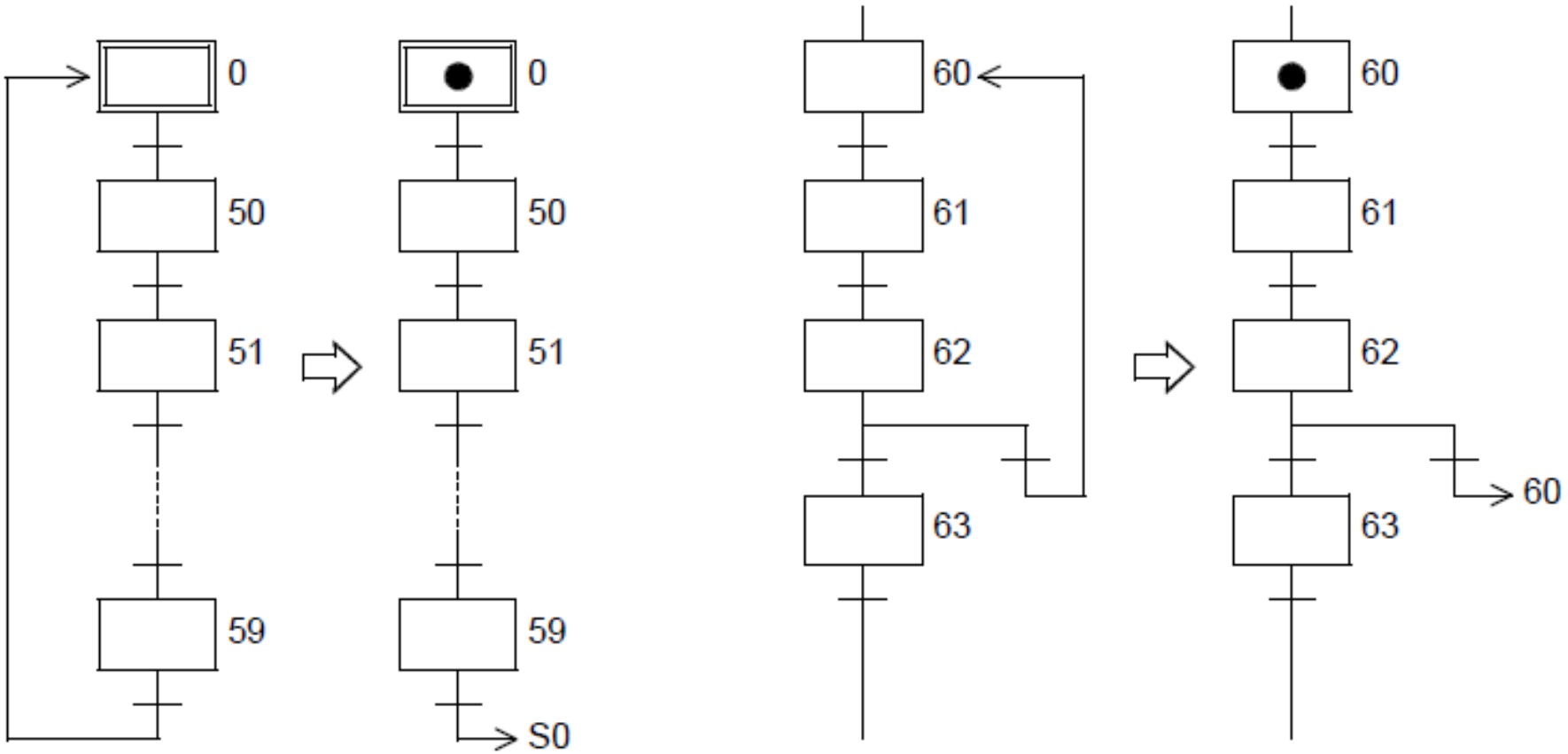
Notes

❖ Jump to another flow

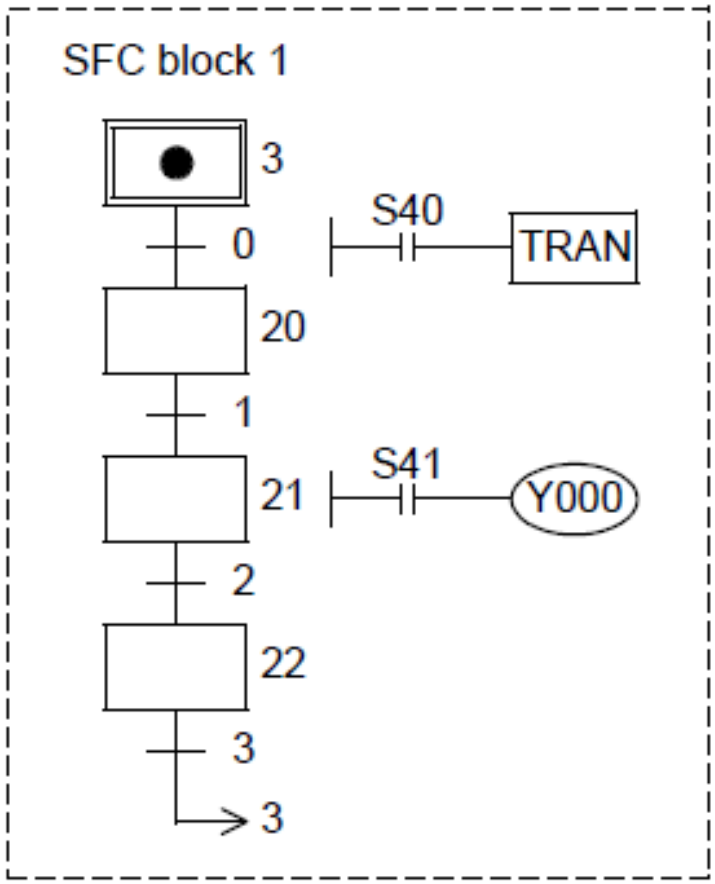


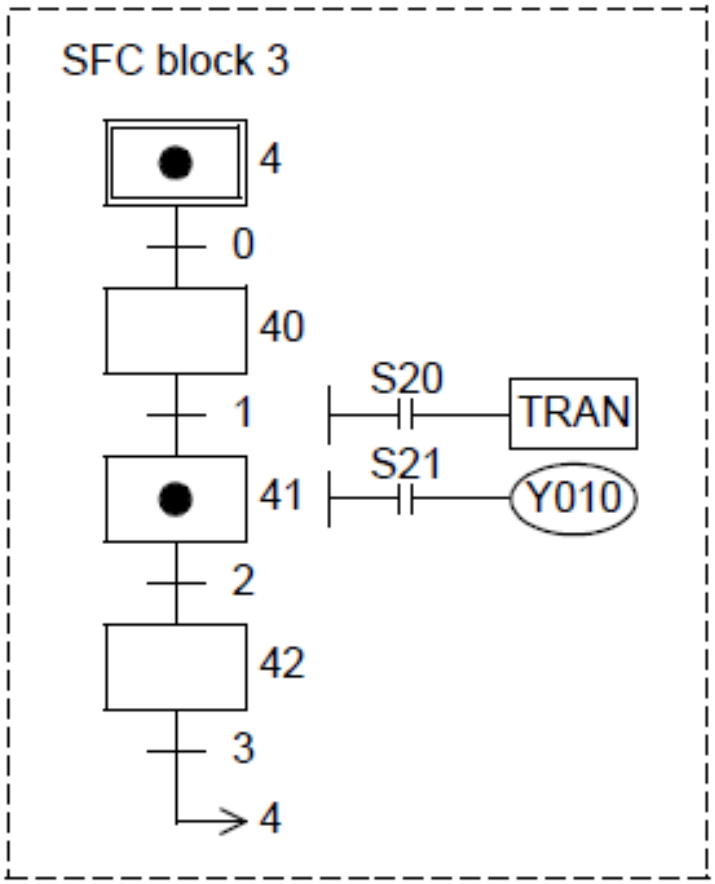
Notes

❖ Repeat



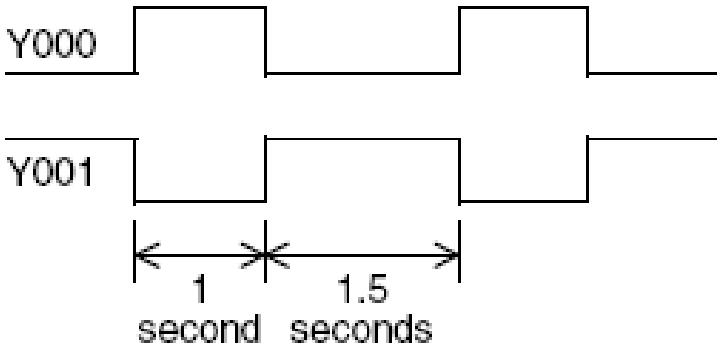
Notes

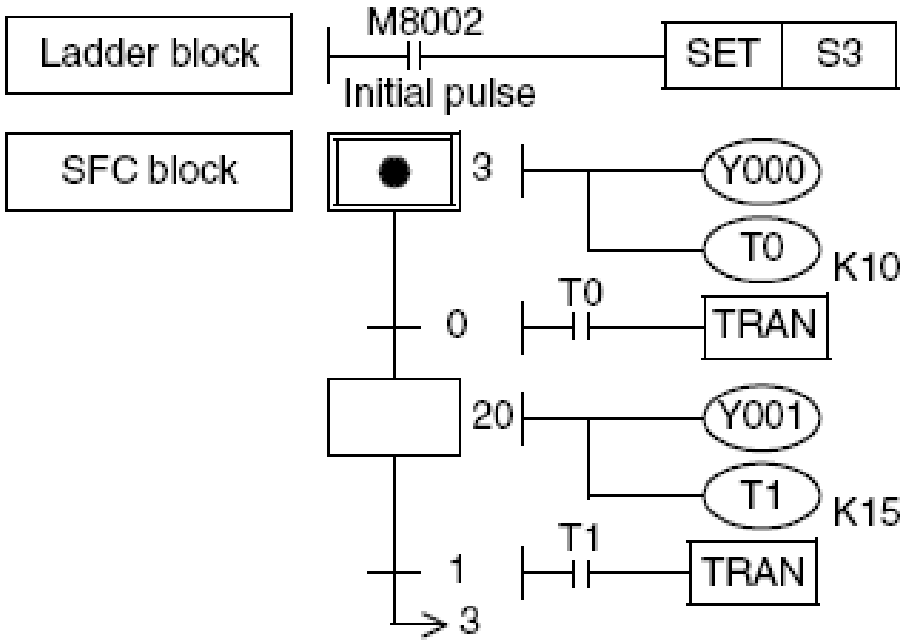

SFC block 1
3
0
S40
TRAN
20
1
21
S41
Y000
2
22
3
3


SFC block 3
4
0
40
1
S20
TRAN
41
S21
Y010
2
42
3
4

Example

❖ Clock signal generator by SFC

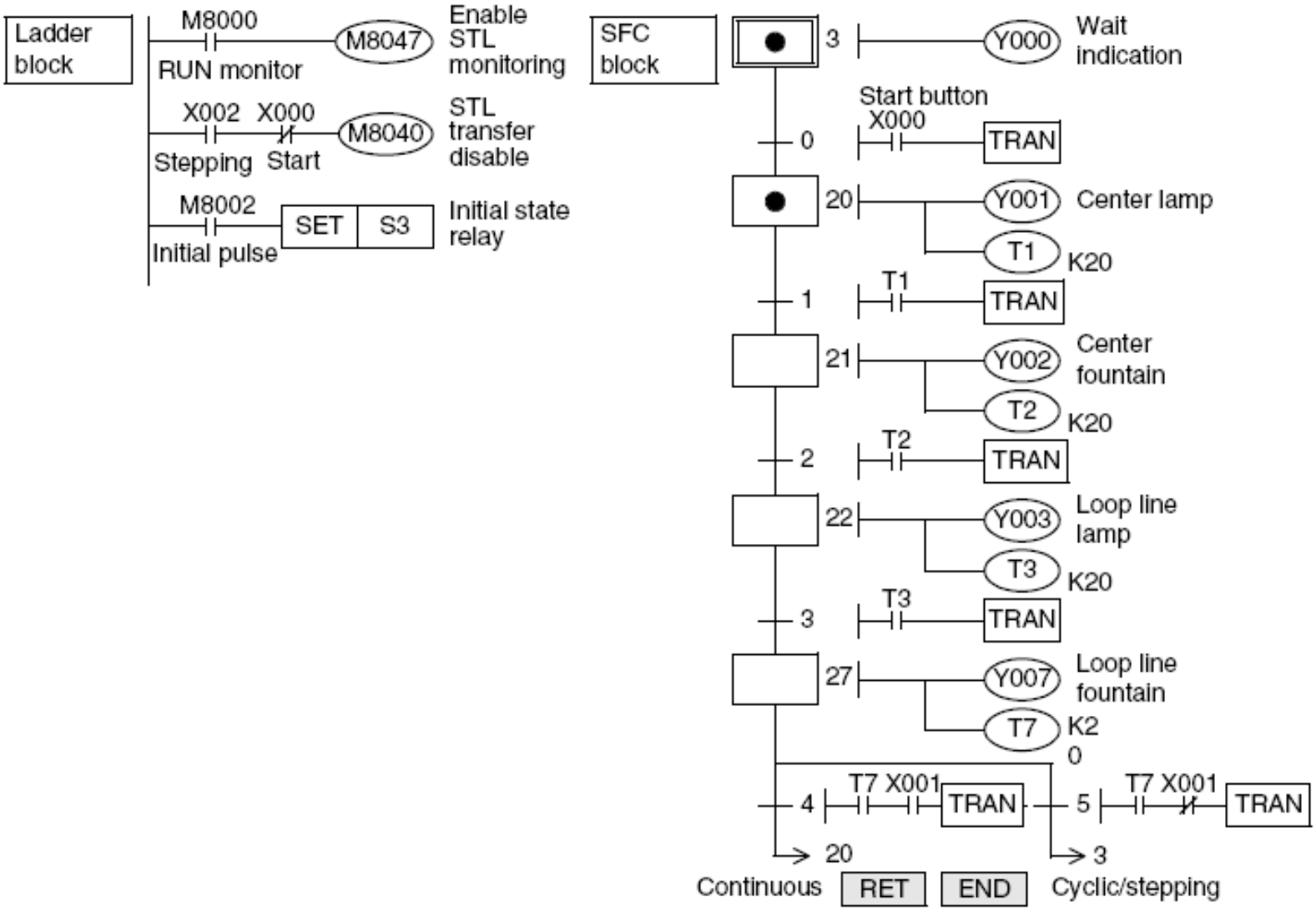




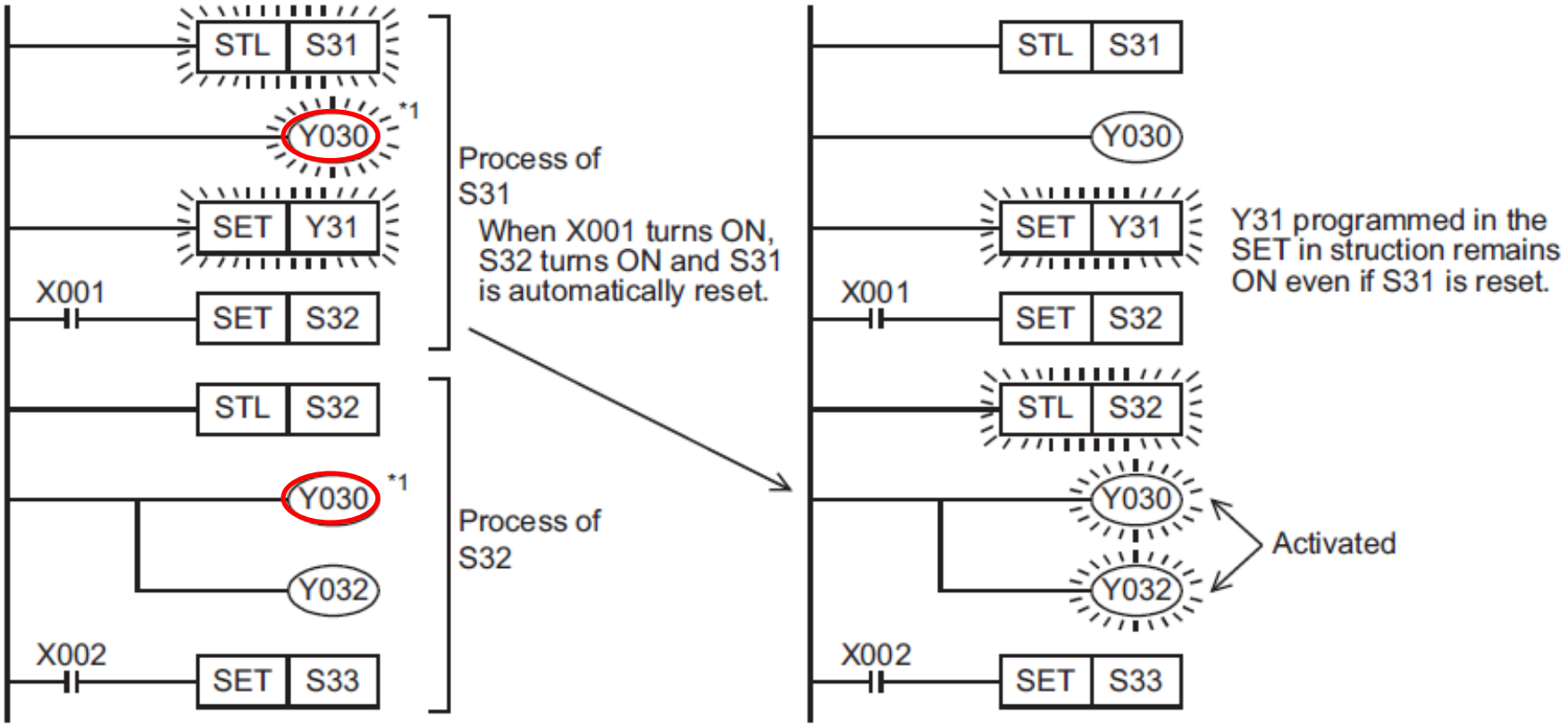
Example

- Fountain control
- 1) Cyclic operation (X001 = OFF, X002 = OFF)
  - When the start button X000 is pressed, the outputs turn ON in the order “Y000 (wait indication) → Y001(center lamp) → Y002 (center fountain) → Y003 (loop line lamp) → Y007 (loop line fountain) → Y000(wait indication)”, and then the outputs return to the wait status. Each output is switched in turn every 2 seconds by a timer.
- 2) Continuous operation (X001 = ON)
  - Y001 to Y007 turn ON in turn repeatedly.
- 3) Stepping operation (X002 = ON)
  - Every time the start button is pressed, each output turns ON in turn.

Example


Ladder block
M8000
M8047
Enable STL monitoring
RUN monitor
X002 X000
M8040
STL transfer disable
Stepping Start
M8002
SET S3
Initial state relay
Initial pulse
SFC block
3
Y000
Wait indication
Start button
X000
0
TRAN
20
Y001
Center lamp
T1
K20
T1
1
TRAN
21
Y002
Center fountain
T2
K20
T2
2
TRAN
22
Y003
Loop line lamp
T3
K20
T3
3
TRAN
27
Y007
Loop line fountain
T7
K2
0
T7 X001
4
TRAN
5
T7 X001
TRAN
20
3
Continuous
RET
END
Cyclic/stepping

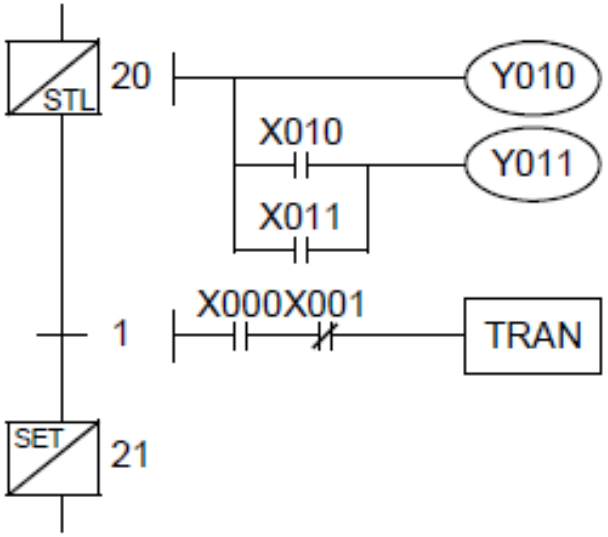
STep Ladder (STL)



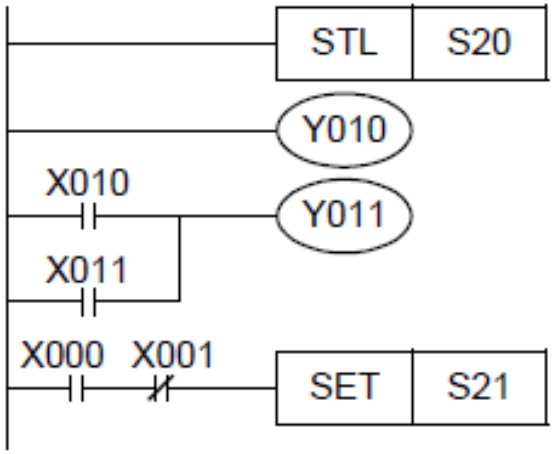
*1. Output coils can be used again in different state relays.

STep Ladder (STL)

<SFC program>



<Step ladder>



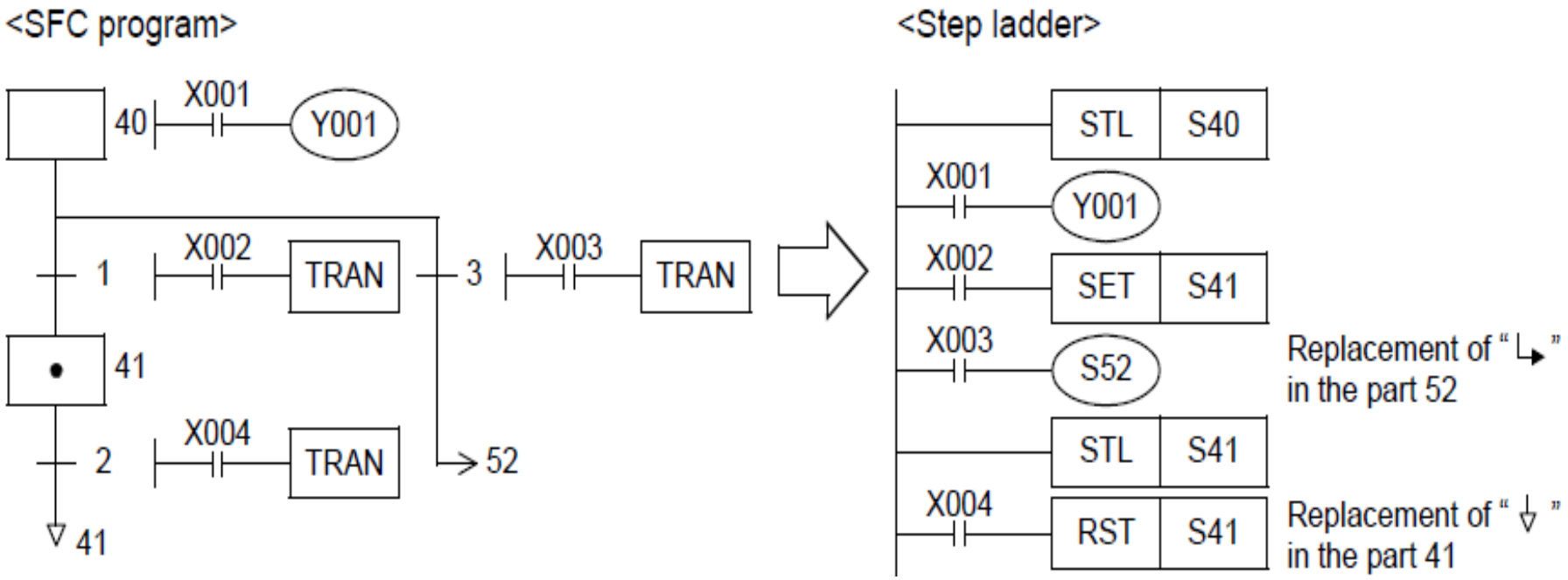
<List program>

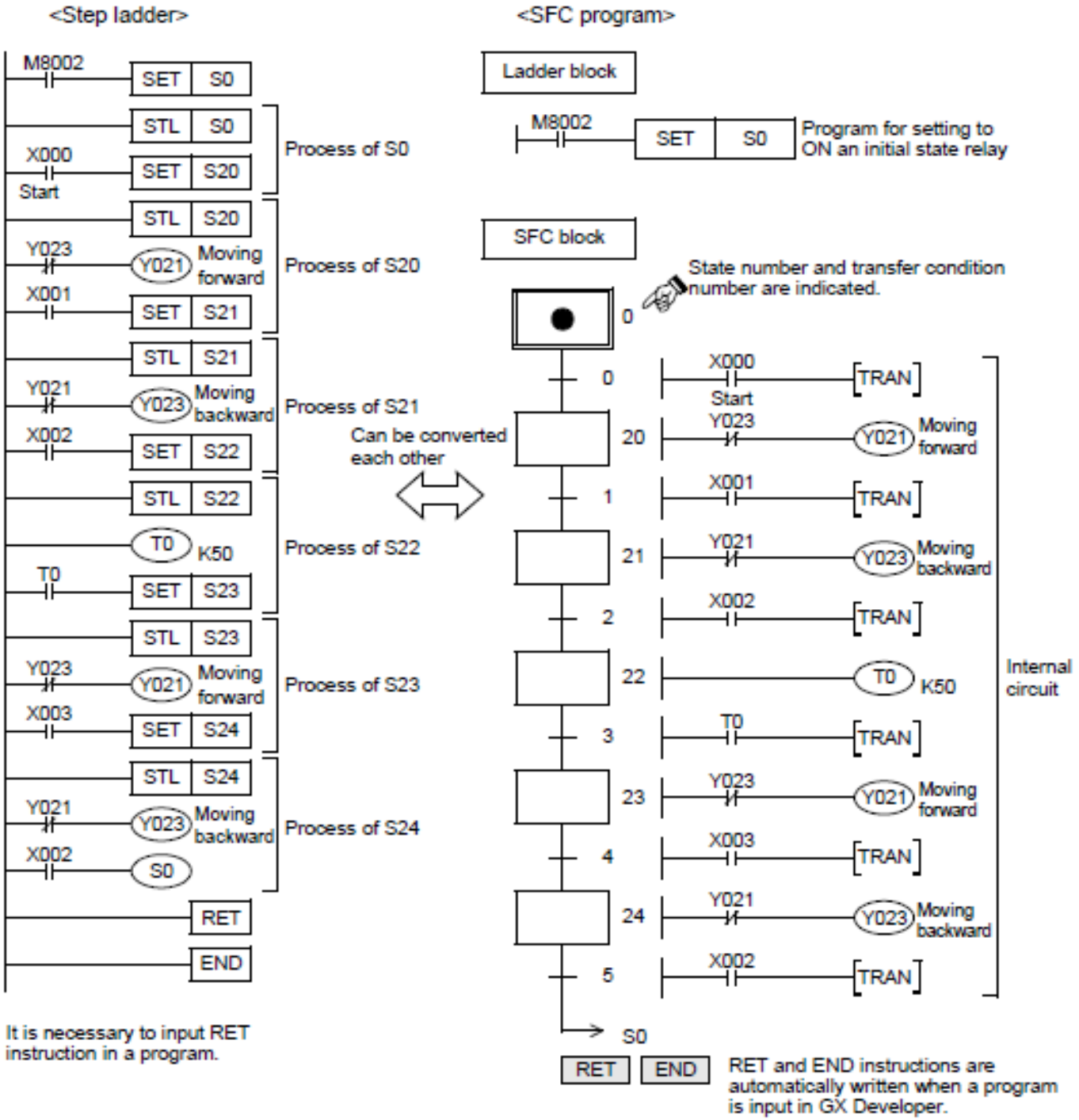
0	STL	S20
1	OUT	Y010
2	LD	X010
3	OR	X011
4	OUT	Y011
5	LD	X000
6	ANI	X001
7	SET	S21 *1

The above program can be expressed in the list format (list program) shown on the left.
The segment from the STL instruction to the RET instruction is handled as a step ladder program.

STep Ladder (STL)

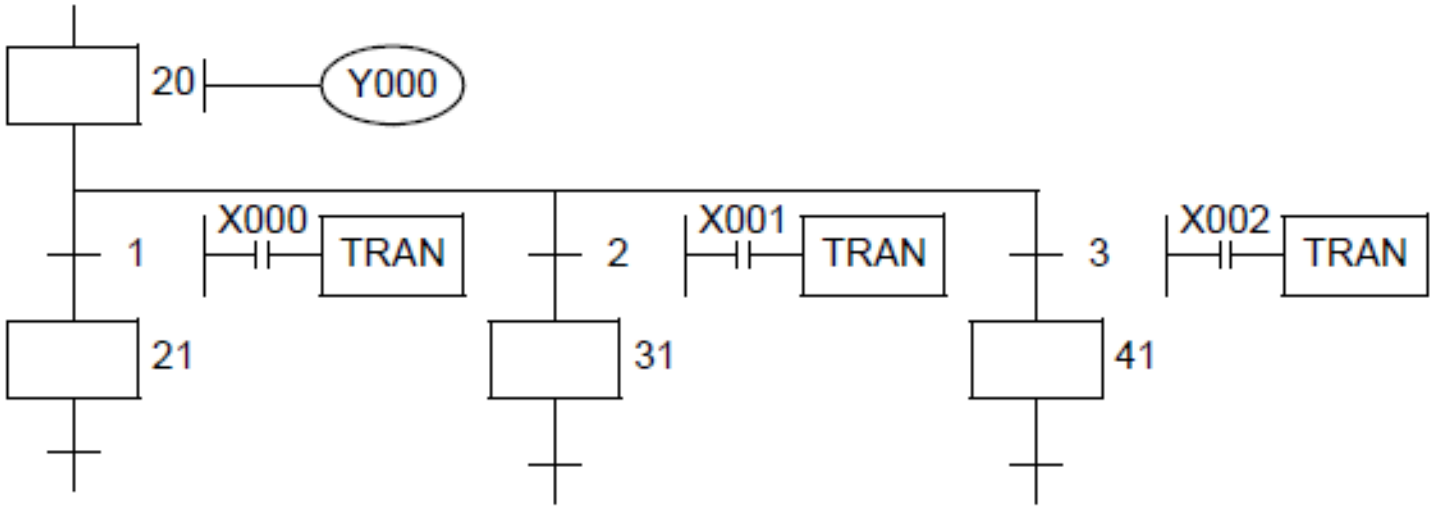
❖ **Replacement "↳" and "▽" in SFC to STL**



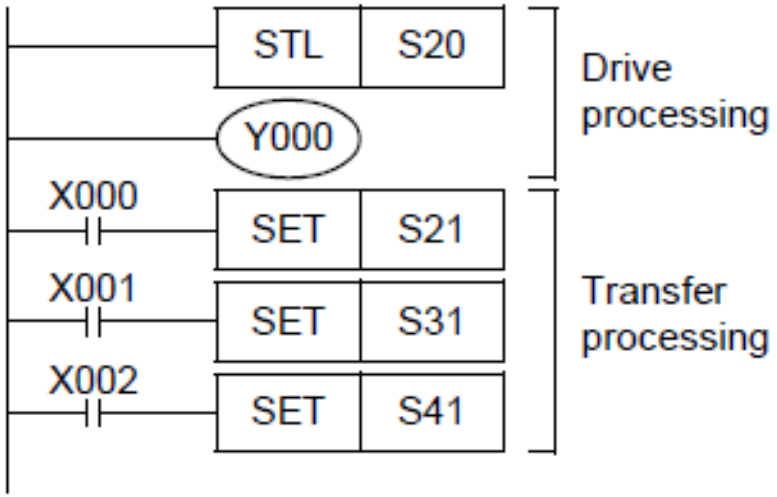

<Step ladder>
M8002
SET S0
STL S0
Process of S0
X000
Start
SET S20
STL S20
Y023
Y021
Moving forward
Process of S20
X001
SET S21
STL S21
Y021
Y023
Moving backward
Process of S21
X002
SET S22
STL S22
T0
K50
Process of S22
T0
SET S23
STL S23
Y023
Y021
Moving forward
Process of S23
X003
SET S24
STL S24
Y021
Y023
Moving backward
Process of S24
X002
S0
RET
END
It is necessary to input RET instruction in a program.
<SFC program>
Ladder block
M8002
SET S0
Program for setting to ON an initial state relay
SFC block
State number and transfer condition number are indicated.
Can be converted each other
0
0
X000
Start
TRAN
20
Y023
Y021
Moving forward
1
X001
TRAN
21
Y021
Y023
Moving backward
2
X002
TRAN
22
T0
K50
3
T0
TRAN
23
Y023
Y021
Moving forward
4
X003
TRAN
24
Y021
Y023
Moving backward
5
X002
TRAN
Internal circuit
S0
RET
END
RET and END instructions are automatically written when a program is input in GX Developer.

Selective branch

<SFC program>



<Step ladder>

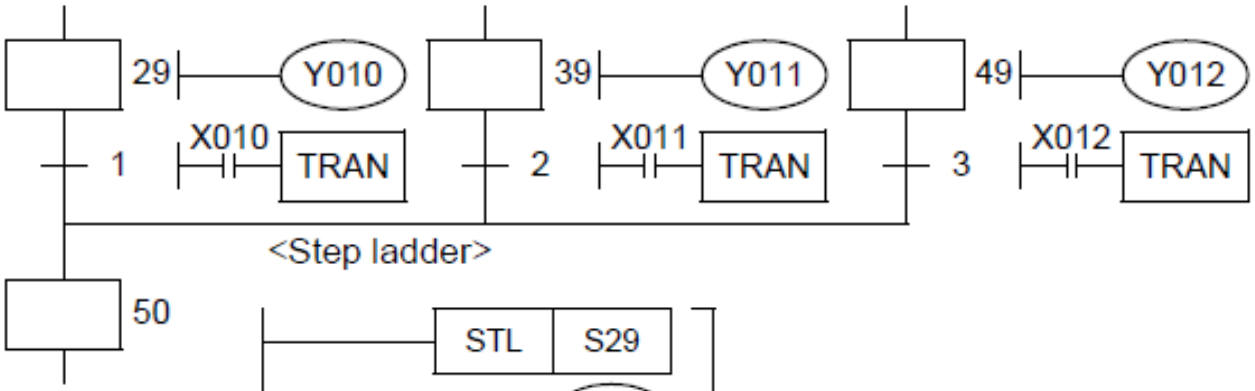


<List program>

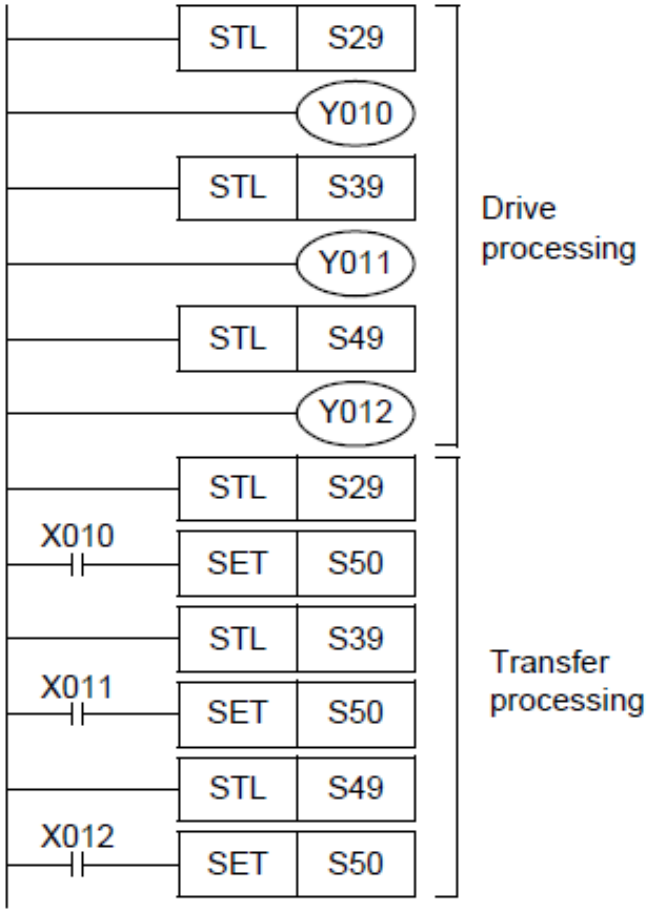
```
STL   S20
OUT   Y000
LD    X000
SET   S21
LD    X001
SET   S31
LD    X002
SET   S41
```

Selective recombination

<SFC program>



<Step ladder>

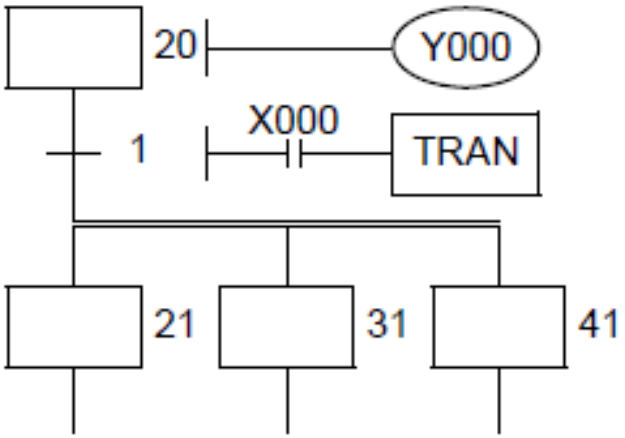


<List program>

```
STL   S29
OUT   Y010
STL   S39
OUT   Y011
STL   S49
OUT   Y012
STL   S29
LD    X010
SET   S50
STL   S39
LD    X011
SET   S50
STL   S49
LD    X012
SET   S50
```

Parallel branch

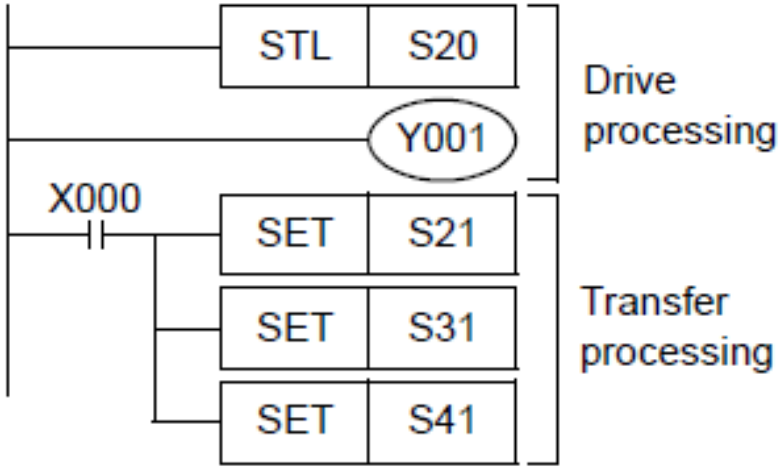
<SFC program>



<List program>

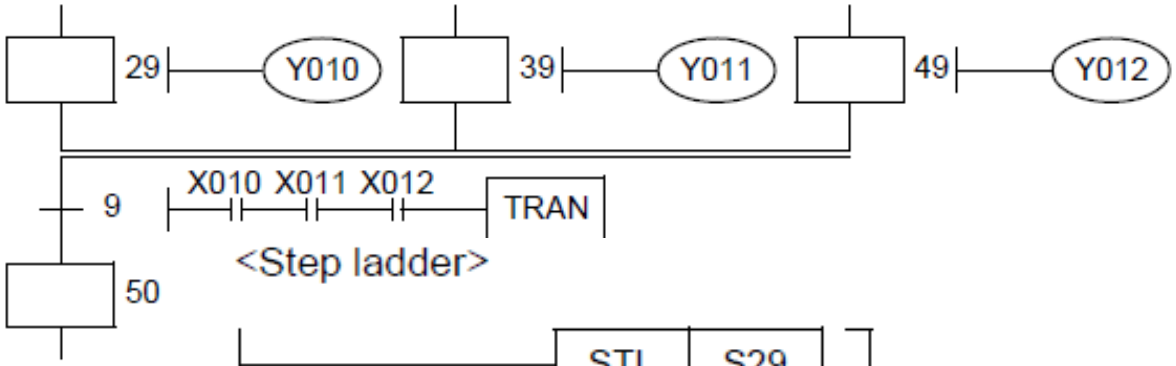
```
STL   S20
OUT   Y000
LD    X000
SET   S21
SET   S31
SET   S41
```

<Step ladder>

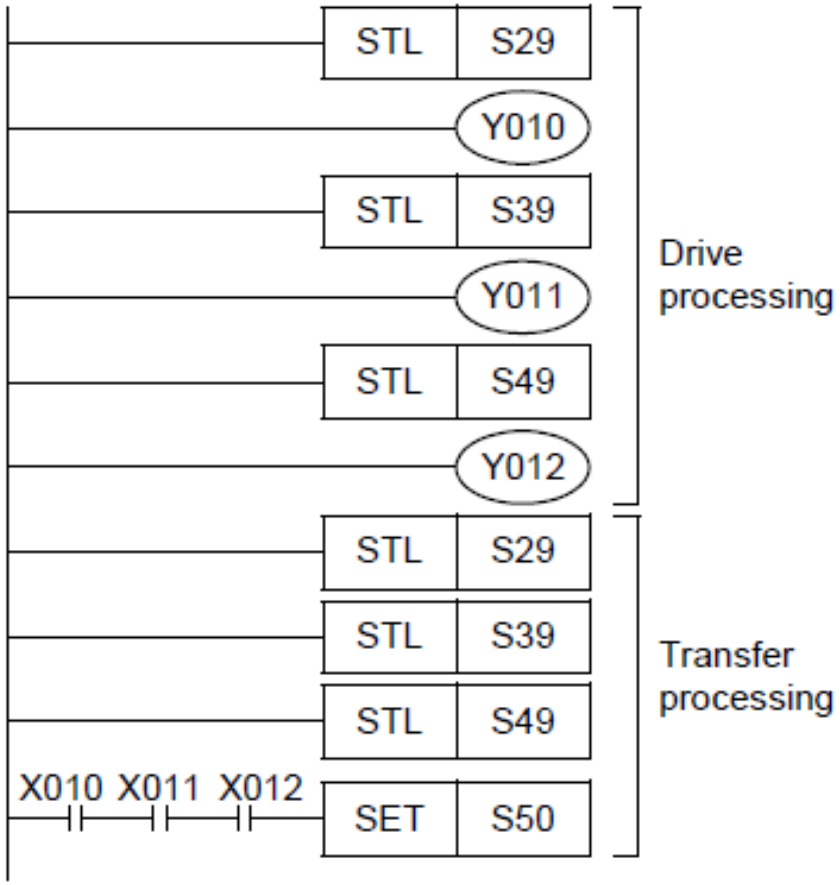


Parallel recombination

<SFC program>



<Step ladder>



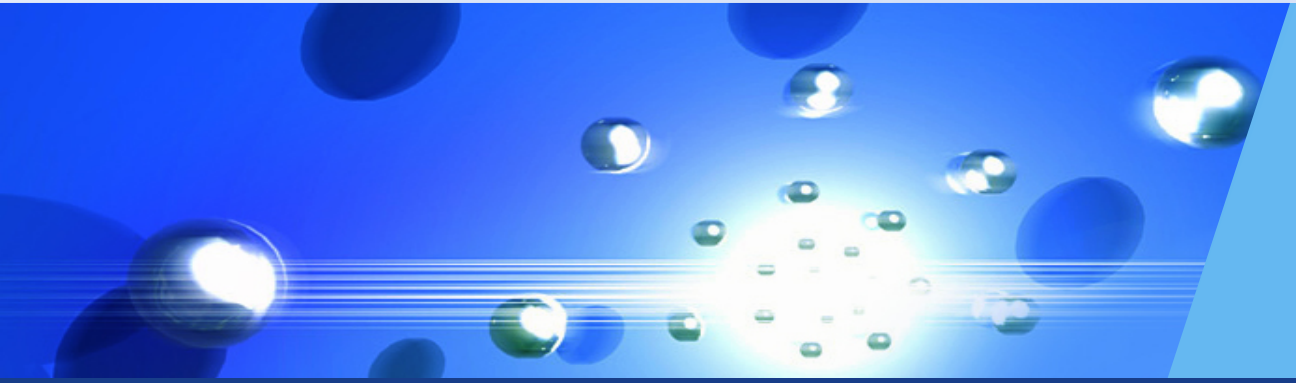
<List program>

```
STL  S29
OUT  Y010
STL  S39
OUT  Y011
STL  S49
OUT  Y012
STL  S29
STL  S39
STL  S49
LD   X010
AND  X011
AND  X012
SET  S50
```

Importance Note

❖ **Limitation in the number of branch circuits:**

In one parallel branch or selective branch, up to 8 circuits can be provided.



Appendix: Advanced Functions

Introduction

- Program Flow functions
- Move and Compare functions
- Arithmetic and Logical Operation functions
- Rotation and Shift Operation functions
- Data Operation functions
- High Speed Processing functions
- Handy functions

Read the reference: *FX3G_FX3U-Programming manual_Basic and applied instruction.pdf*

These instructions call Function (FNC) with number from 00 to 69.

Summary

Program flow functions	CJ	Conditional Jump to a program position
	CALL	Calls (executes) a subroutine
	SRET	Subroutine Return, marks the end of a subroutine
	IRET	Interrupt Return, marks the end of an interrupt routine
	EI	Enable Interrupt, enables processing of interrupt routines
	DI	Disable Interrupt, disables processing of interrupt routines
	FEND	First End, marks end of main program block
	WDT	WatchDog Timer refresh
	FOR	Marks beginning of a program loop
	NEXT	Marks end of a program loop
Move and compare functions	CMP	Compare numerical values
	ZCP	Zone Compare, compares numerical ranges
	MOV	Move data from one storage area to another
	SMOV	Shift Move
	CML	Compliment, copies and inverts
	BMOV	Block Move
	FMOV	Fill Move, copy to a range of devices
	XCH	Exchange data in specified devices
	BCD	BCD conversion
	BIN	Binary conversion

Math and logic instructions	ADD	Add numerical values
	SUB	Subtract numerical values
	MUL	Multiply numerical values
	DIV	Divide numerical values
	INC	Increment
	DEC	Decrement
	WAND	Logical AND
	WOR	Logical OR
	WXOR	Logical exclusive OR
	NEG	Negation, logical inversion of device contents
Rotate and shift functions	ROR	Rotate right
	ROL	Rotate left
	RCR	Rotation right with carry
	RCL	Rotation left with carry
	SFTR	Shift right, bitwise shift to the right
	SFTL	Shift left, bitwise shift to the left
	WSFR	Word shift right, shift word values to the right
	WSFL	Word shift left, shift word values to the left
	SFWR	Shift register write, writes to a FIFO stack
	SFRD	Shift register read, reads from a FIFO stack

Summary

Data operation functions	ZRST	Zone Reset, resets ranges of like devices
	DECO	Decode data
	ENCO	Encode data
	SUM	Sum (number) of active bits
	BON	Bit on, checks status of a bit
	MEAN	Calculates mean values
	ANS	Timed annunciator set, starts a timer interval
	ANR	Annunciator reset
	SQR	Square root
	FLT	Floating point, converts data
High-speed instructions	REF	Refresh inputs and outputs
	REFF	Refresh inputs and filter adjust
	MTR	Input matrix, read a matrix (MTR)
	DHSCS	High-speed counter set
	DHSCR	High-speed counter reset
	DHSZ	High speed zone compare
	SPD	Speed detection
	PLSY	Pulse Y output (frequency)
	PWM	Pulse output with pulse width modulation
	PLSR	Pulse ramp (accelleration/deceleration setup)

Application instructions	IST	Initial state, set up multi-mode STL system
	SER	Search data stack
	ABSD	Absolute counter comparison
	INCD	Incremental counter comparison
	TTMR	Teaching timer
	STMR	Special timer
	ALT	Alternate state, flip-flop function
	RAMP	Ramp function
	ROTC	Rotary table control
	SORT	Sort table data on selected fields
Instructions for external I/O devices	TKY	Ten key input
	HKY	Hexadecimal key input
	DSW	Digital switch
	SEGD	7-segment display decoder
	SEGL	7-segment display with latch
	ARWS	Arrow switch
	ASC	ASCII conversion
	PR	Print, data output via the outputs
	FROM	Read data from a special function module
	TO	Write data to a special function module

Note

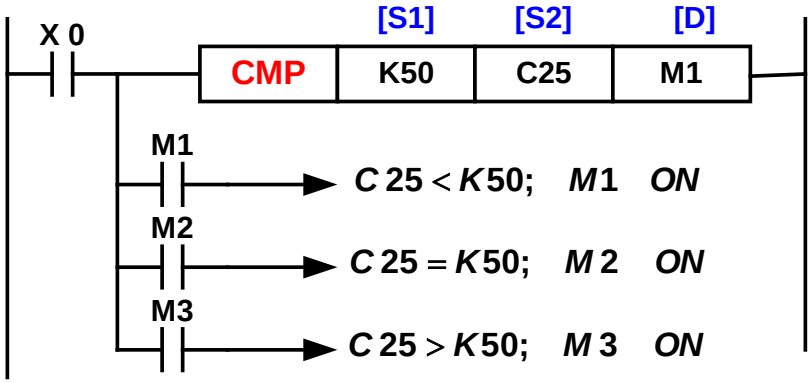
- **The functions with “P” character at last position, such as: CMPP, ZCPP, INCP, ADDP, MULP, … execute only once based on rising edge of input signal.**

16-bit Instruction	Mnemonic	Operation Condition
7 steps	ADD	 Continuous Operation
	ADDP	 Pulse (Single) Operation

CMP (Compare) function

Mnemonic	Description	Operand			Step
		S1	S2	D	
CMP	Compare two values. Results: smaller; equal or larger.	K, H KnX; KnY; KnM; KnS T, C, D, V, Z		Y, M, S Note: Using 3 bits consecutively	CMP, CMPP 7 steps. DCMP; DCMPP 13 steps

CMP: the comparison value (S1) and the comparison source (S2) are compared each other.

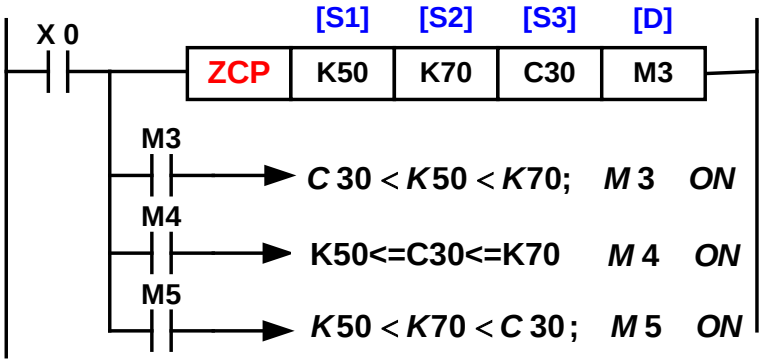


S2 < S1 ((value in counter C25 is smaller than K = 50), bit D (M1) is ON.
S2 = S1 ((value in counter C25 is equal K = 50), bit D+1 (M2) is ON.
S2 > S1 ((value in counter C25 is larger than K = 50), bit D+2 (M3) is ON.

ZCP (Zone compare) function

- **ZCP (Zone Compare): compares two values (S1 & S2) with the comparison source (S3).**

Mnemonic	Description	Operand				Step
		S1	S2	S3	D	
ZCP	Compare a zone with source. Result: smaller; equal or larger.	K, H KnX; KnY; KnM; KnS T, C, D, V, Z Note: S1 is always less than S2.			Y, M, S Note: Using 3 bits consecutively.	ZCP, ZCPP: 9 steps. DZCP, DZCPP: 17 steps.

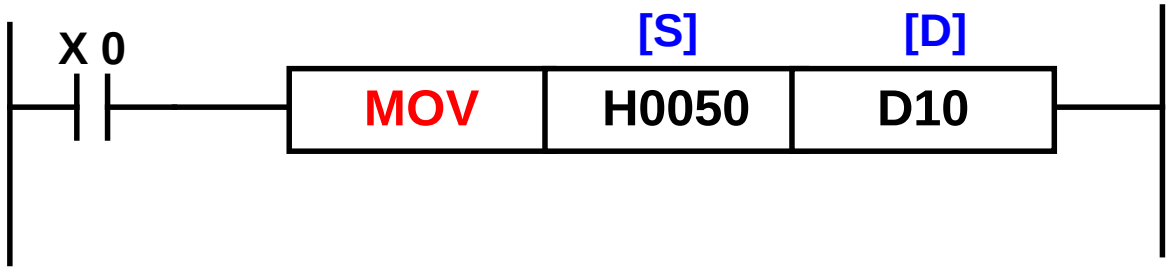


S3 < S1 < S2; (value in counter C30 is smaller than 50), bit D (M3) is ON.
S1 <= S3 <= S2; ((value in counter C30 is in range [50, 70]), bit D+1 (M4) is ON.
S3 > S2 > S1; (value in counter C30 is larger than 70), bit D+2 (M5) is ON.

MOV (Move) function

- **MOV (Move): transfers (copies) the contents of a device to another device.**

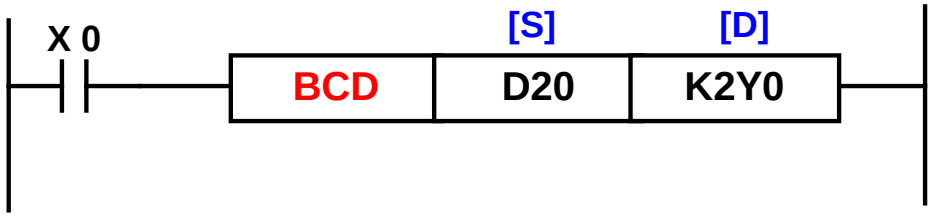
Mnemonic	Description	Operand		Steps
		S	D	
MOV	Transfer the source data to destination device	K, H KnX; KnY; KnM; KnS T, C, D, V, Z	KnY; KnM; KnS T, C, D, V, Z	MOV, MOVP 5 steps. DMOV ; DMOVP 9 steps



BCD (Binary Coded Decimal) function

- **BCD (Binary Coded Decimal) : converts binary data to**
  - **Convert binary data (BIN) to Binary Coded Decimal data.**
  - **Display numeric values on the seven-segment display unit equipped with BCD decoder.**

Mnemonic	Description	Operand		Step
		S	D	
BCD	Convert binary data of [S] to BCD data and transfer the converted BCD data to [D]	KnX; KnY; KnM; KnS T, C, D, V, Z	KnY; KnM; KnS T, C, D, V, Z	BCD, BCDP 5 steps. DBCD ; DBCDP 9 steps

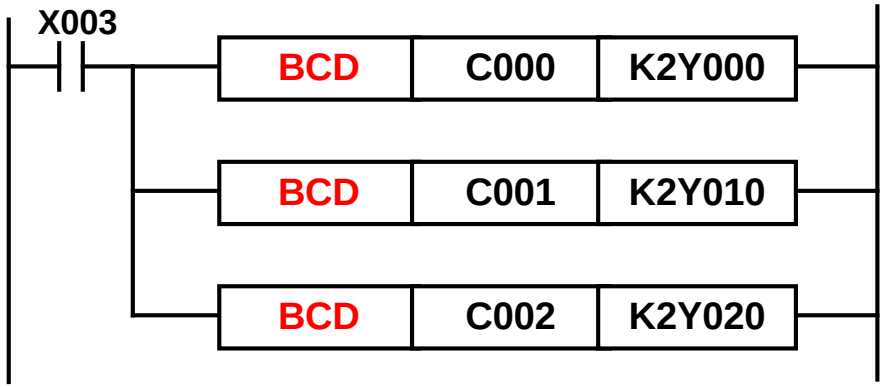


K2Y0: 2x4bit = Y000 ÷ Y007

BCD (Binary Coded Decimal) function

- A program shows the time on the 7-segment LED with C000 for second; C001 for minute; and C002 for hour. I/O of PLC is shown in the following figure.

OUTPUT	Y24-7	Y20-3	Y14-7	Y10-3	Y4-7	Y0-3
BCD CODE	2	0	1	6	1	4

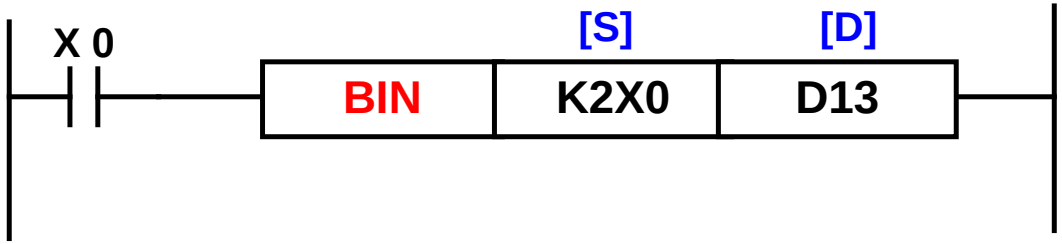


BIN (Binary) function

BIN (Binary):

- **Convert BCD data into binary data.**
- **Use this function to convert a BCD value set by a digital switch (Thumbwheel switch) into binary value and to receive the converted binary data.**

Mnemonic	Description	Operand		Step
		S	D	
BIN	Convert BCD data into binary data	KnX; KnY; KnM; KnS T, C, D, V, Z	KnY; KnM; KnS T, C, D, V, Z	BIN, BINP: 5 steps. DBIN, DBINP: 9 steps

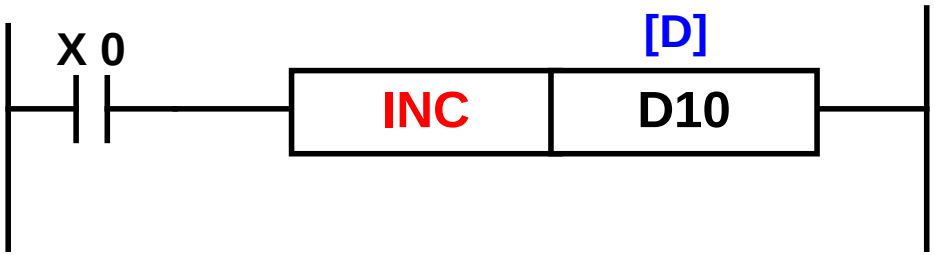


Arithmetic and Logical Operation functions

- ADD (Addition)
- SUB (Subtraction)
- MUL (Multiply)
- DIV (Division)
- INC (Increment)
- DEC (Decrement)

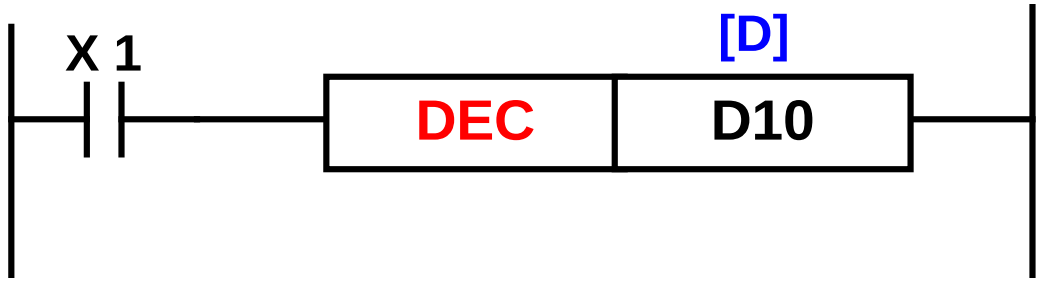
INC (Increment) function

Mnemonic	Description	Operand	Step
		D	
INC	Increment the data of specified device by “1”	KnY; KnM; KnS T, C, D, V, Z	INC, INCP: 3 step. DINC; DINCP: 5 step



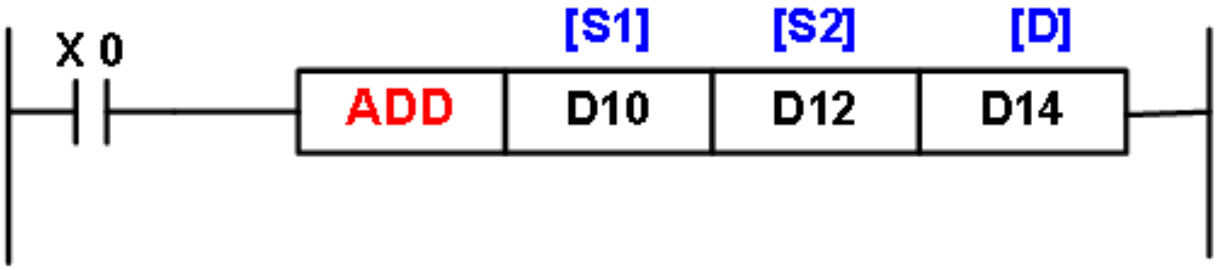
DEC (Decrement) function

Mnemonic	Description	Operand	Step
		D	
DEC	Decrement the data of the specified device by "1"	KnY; KnM; KnS T, C, D, V, Z	DEC, DECP : 3 steps. DDEC; DDECP : 5 steps



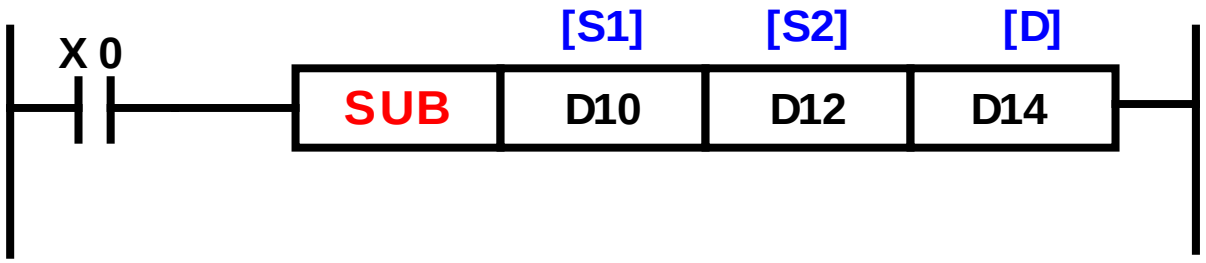
ADD (Addition) function

Mnemonic	Description	Operand			Step
		S1	S2	D	
ADD	Add two values (S1 + S2) and save the result to D	K, H, KnX; KnY; KnM; KnS T, C, D, V, Z		KnY; KnM; KnS T, C, D, V, Z	ADD, ADDP : 7 steps DADD; DADDP : 13 steps



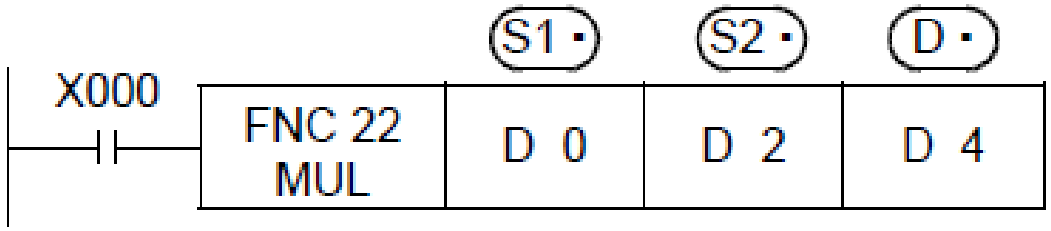
SUB (Subtraction) function

Mnemonic	Description	Operand			Step
		S1	S2	D	
SUB	Subtract two values (S1 – S2) and save the result to D	K, H, KnX; KnY; KnM; KnS T, C, D, V, Z		KnY; KnM; KnS T, C, D, V, Z	SUB, SUBP : 7 steps DSUB; DSUBP : 13 steps



MUL (Multiplication) function

Mnemonic	Description	Operand			Step
		S1	S2	D	
MUL	Multiply two values (S1 * S2) and save the result to D	K, H, KnX; KnY; KnM; KnS T, C, D, V, Z		KnY; KnM; KnS T, C, D, V, Z(V)	MUL, MULP : 7 steps DMUL; DMULP : 13 steps

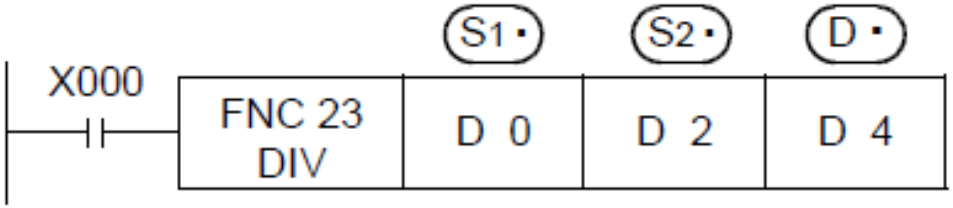


(D 0) × (D 2) → (D 5,D 4)
8 9 72

<u>Note</u>: If S1 and S2 are 16-bit data registers, the multiplication result is transferred to 32-bit data register.

DIV (Division) function

Mnemonic	Description	Operand			Step
		S1	S2	D	
DIV	Divide two values (S1 / S2) and save the result to D	K, H, KnX; KnY; KnM; KnS T, C, D, V, Z		KnY; KnM; KnS T, C, D, V, Z(V)	DIV, DIVP: 7 steps DDIV; DDIVP: 13 steps



Dividend		Divisor		Quotient		Remainder
(D 0)	÷	(D 2)	→	(D 4)	...	(D 5)
100		33		3		1

<u>Note</u>: If S1 and S2 are 16-bit data registers, the quotient is transferred to D (16-bit data register), the reminder is transferred to D+1.