PLC Concepts

1

This chapter introduces basic and advanced concepts of ladder logic, which is the mostly adopted programming language of PLC. Users familiar with the PLC concepts can move to the next chapter for further programming concepts. However, for users not familiar with the operating principles of PLC, please refer to this chapter to get a full understanding of PLC concepts.

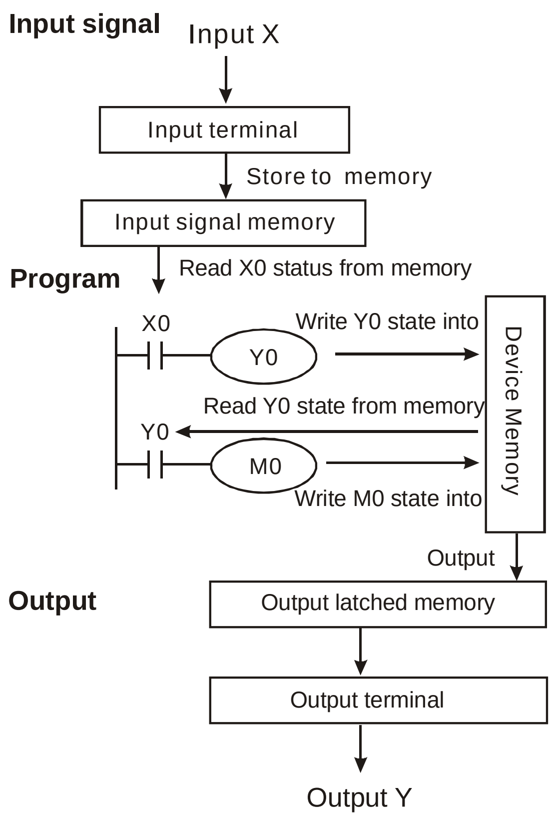
Chapter Contents

1.1 PLC Scan Method

PLC utilizes a standard scan method when evaluating user program.

Scanning process:

Scan input status	Read the physical input status and store the data in internal memory.
Evaluate user program	Evaluate the user program with data stored in internal memory. Program scanning starts from up to down and left to right until reaching the end of the program.
Refresh the outputs	Write the evaluated data to the physical outputs



Input signal:
PLC reads the ON/OFF status of each input and stores the status into memory before evaluating the user program.
Once the external input status is stored into internal memory, any change at the external inputs will not be updated until next scan cycle starts.
Program:
PLC executes instructions in user program from top to down and left to right then stores the evaluated data into internal memory. Some of this memory is latched.
Output:
When END command is reached the program evaluation is complete. The output memory is transferred to the external physical outputs.

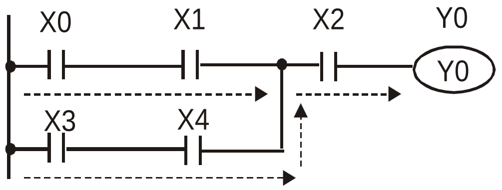
Scan time
The duration of the full scan cycle (read, evaluate, write) is called "scan time." With more I/O or longer program, scan time becomes longer.

Read scan time	PLC measures its own scan time and stores the value (0.1ms) in register D1010, minimum scan time in register D1011, and maximum scan time in register D1012.
Measure scan time	Scan time can also be measured by toggling an output every scan and then measuring the pulse width on the output being toggled.
Calculate scan time	Scan time can be calculated by adding the known time required for each instruction in the user program. For scan time information of individual instruction please refer to Ch3 in this manual.

Scan time exception
PLC can process certain items faster than the scan time. Some of these items interrupts and halt the scan time to process the interrupt subroutine program. A direct I/O refresh instruction REF allows the PLC to access I/O immediately during user program evaluation instead of waiting until the next scan cycle.

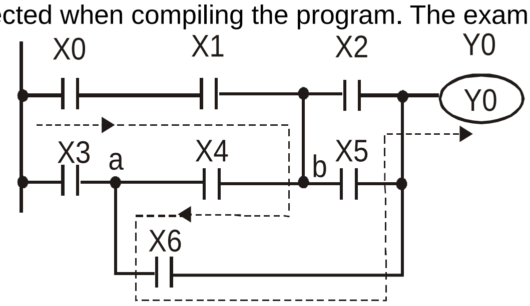
1.2 Current Flow

Ladder logic follows a left to right principle. In the example below, the current flows through paths started from either X0 or X3.

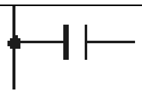
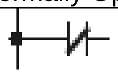


Reverse Current

When a current flows from right to left, which makes a reverse current logic, an error will be detected when compiling the program. The example below shows the reverse current flow.



1.3 NO Contact, NC Contact

NO contact	 Normally Open Contact, A contact
NC Contact	 Normally Closed Contact, B contact

1.4 PLC Registers and Relays

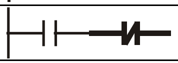
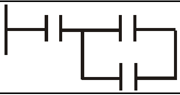
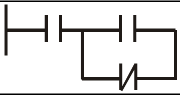
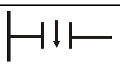
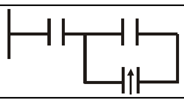
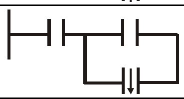
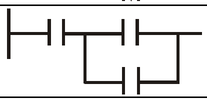
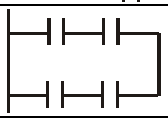
Introduction to the basic internal devices in a PLC

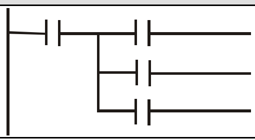
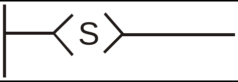
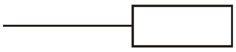
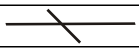
X (Input Relay)	Bit memory represents the physical input points and receives external input signals. ■ Device indication: Indicated as **X** and numbered in octal, e.g. X0~X7, X10~X17…X377
Y (Output Relay)	Bit memory represents the physical output points and saves the status to be refreshed to physical output devices. ■ Device indication: Indicated as **Y** and numbered in octal, e.g. Y0~Y7, Y10~Y17. ..Y377
M (Internal Relay)	Bit memory indicates PLC status. ■ Device indication: Indicated as **M** and numbered in decimal, e.g. M0, M1, M2…M4095
S (Step Relay)	Bit memory indicates PLC status in Step Function Control (SFC) mode. If no STL instruction is applied in program, step point S can be used as an internal relay M as well as an annunciator. ■ Device indication: Indicated as **S** and numbered in decimal, e.g. S0, S1, S2…S1023
T (Relay) (Word) (Dword)	Bit, word or double word memory used for timing and has coil, contact and register in it. When its coil is ON and the set time is reached, the associated contact will be energized. Every timer has its resolution (unit: 1ms/10ms/100ms). ■ Device indication: Indicated as **T** and numbered in decimal, e.g. T0, T1, T2…T255

C (Counter) (Relay) (Word) (Dword)	Bit, word or double word memory used for counting and has coil, contact and register in it. The counter count once (1 pulse) when the coil goes from OFF to ON. When the predefined counter value is reached, the associated contact will be energized. There are 16-bit and 32-bit high-speed counters available for users. ■ Device indication: Indicated as **C** and numbered in decimal, e.g. C0, C1, C2…C255
D (Data register) (Word)	Word memory stores values and parameters for data operations. Every register is able to store a word (16-bit binary value). A double word will occupy 2 consecutive data registers. ■ Device indication: Indicated as **D** and numbered in decimal, e.g. D0, D1, D2…D4999
E, F (Index register) (Word)	Word memory used as a modifier to indicate a specified device (word and double word) by defining an offset. Index registers not used as a modifier can be used as general purpose register. ■ Device indication: indicated as E0 ~ E7 and F0 ~ F7.

1.5 Ladder Logic Symbols

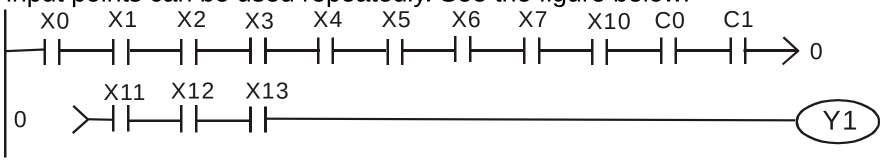
The following table displays list of WPLSoft symbols their description, command, and memory registers that are able to use the symbol.

Ladder Diagram Structure	Explanation	Instruction	Available Devices
	NO (Normally Open) contact / A contact	LD	X, Y, M, S, T, C
	NC (Normally Closed) contact / B contact	LDI	X, Y, M, S, T, C
	NO contact in series	AND	X, Y, M, S, T, C
	NC contact in series	ANI	X, Y, M, S, T, C
	NO contact in parallel	OR	X, Y, M, S, T, C
	NC contact in parallel	ORI	X, Y, M, S, T, C
	Rising-edge trigger switch	LDP	X, Y, M, S, T, C
	Falling-edge trigger switch	LDF	X, Y, M, S, T, C
	Rising-edge trigger in series	ANDP	X, Y, M, S, T, C
	Falling-edge trigger in series	ANDF	X, Y, M, S, T, C
	Rising-edge trigger in parallel	ORP	X, Y, M, S, T, C
	Falling-edge trigger in parallel	ORF	X, Y, M, S, T, C
	Block in series	ANB	None
	Block in parallel	ORB	None

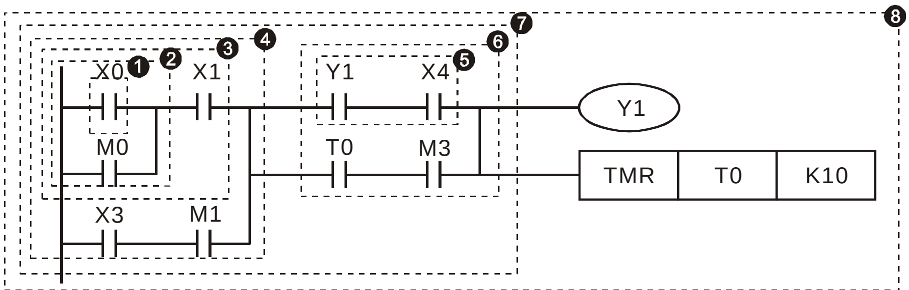
Ladder Diagram Structure	Explanation	Instruction	Available Devices
	Multiple output branches	MPS MRD MPP	None
	Output coil	OUT	Y, M, S
	Step ladder	STL	S
	Basic / Application instruction	-	Basic instructions and API instructions. Please refer to chapter 3 Instruction Set
	Inverse logic	INV	None

1.5.1 Creating a PLC Ladder Program

The editing of the program should start from the left side bus line to the right side bus line, and from up to down. However, the right side bus line is omitted when editing in WPLSoft. A single row can have maximum 11 contacts on it. If more than 11 contacts are connected, a continuous symbol “0” will be generated automatically and the 12th contact will be placed at the start of next row. The same input points can be used repeatedly. See the figure below:



When evaluating the user program, PLC scan starts from left to right and proceeds to next row down until the PLC reaches END instruction. Output coils and basic / application instructions belong to the output process and are placed at the right of ladder diagram. The sample program below explains the execution order of a ladder diagram. The numbers in the black circles indicate the execution order.

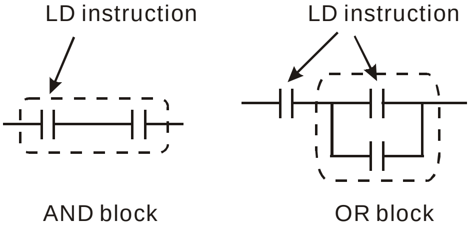


Execution order of the sample program:

1	LD	X0
2	OR	M0
3	AND	X1
4	LD	X3
	AND	M1
	ORB	
5	LD	Y1
	AND	X4
6	LD	T0
	AND	M3
	ORB	
7	ANB	
8	OUT	Y1
	TMR	T0 K10

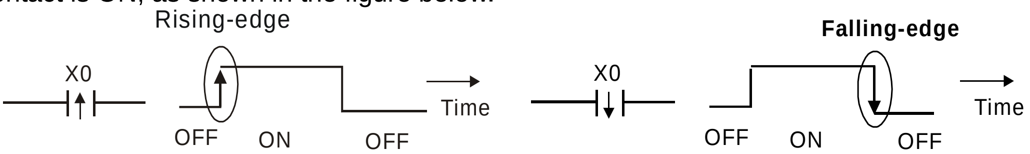
1.5.2 LD / LDI (Load NO contact / Load NC contact)

LD or LDI starts a row or block



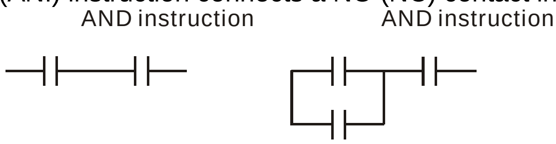
1.5.3 LDP / LDF (Load Rising edge trigger/ Load Falling edge trigger)

Similar to LD instruction, LDP and LDF instructions only act at the rising edge or falling edge when the contact is ON, as shown in the figure below.



1.5.4 AND / ANI (Connect NO contact in series / Connect NC contact in series)

AND (ANI) instruction connects a NO (NC) contact in series with another device or block.

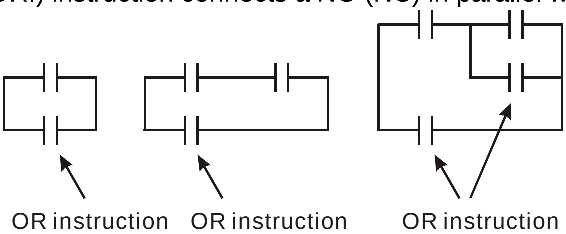


1.5.5 ANDP / ANDF (Connect Rising edge in series/ Connect Falling edge in series)

Similar to AND instruction, ANDP (ANDF) instruction connects rising (falling) edge triggers in series with another device or block.

1.5.6 OR / ORI (Connect NO contact in parallel / Connect NC contact in parallel)

OR (ORI) instruction connects a NO (NC) in parallel with another device or block.

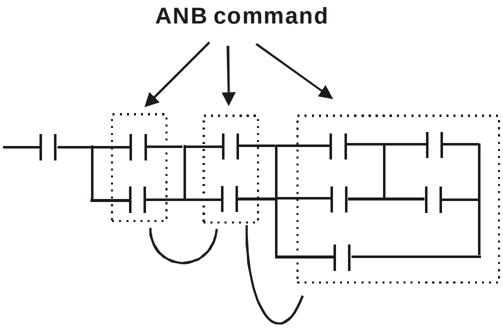


1.5.7 ORP / ORF (Connect Rising edge in parallel/ Connect Falling edge in parallel)

Similar to OR instruction, ORP (ORF) instruction connects rising (falling) edge triggers in parallel with another device or block

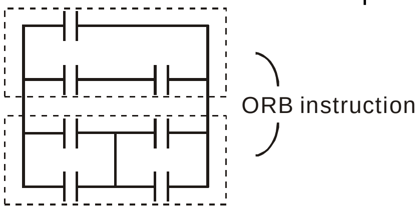
1.5.8 ANB (Connect block in series)

ANB instruction connects a block in series with another block



1.5.9 ORB (Connect block in parallel)

ORB instruction connects a block in parallel with another block



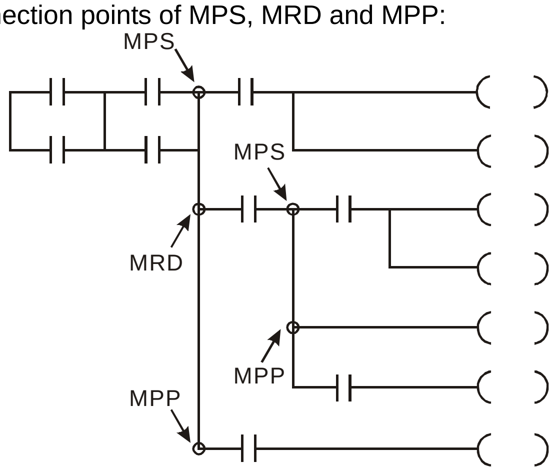
1.5.10 MPS / MRD / MPP (Branch instructions)

These instructions provide a method to create multiplexed output branches based on current result stored by MPS instruction.

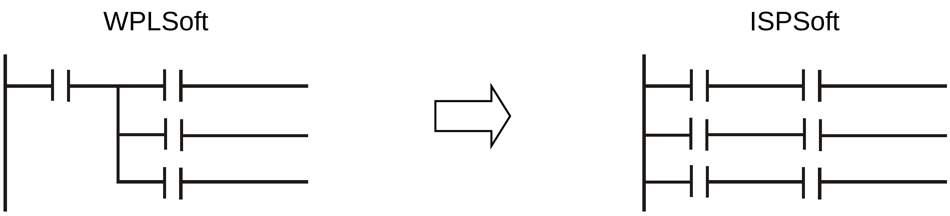
Branch instruction	Branch Symbol	Description
MPS	┬	Start of branches. Stores current result of program evaluation. Max. 8 MPS-MPP pairs can be applied
MRD	├	Reads the stored current result from previous MPS
MPP	└	End of branches. Pops (reads then resets) the stored result in previous MPS

Note: When compiling ladder diagram with WPLSoft, MPS, MRD and MPP could be automatically added to the compiled results in instruction format. However, sometimes the branch instructions are ignored by WPLSoft if not necessary. Users programming in instruction format can enter branch instructions as required.

Connection points of MPS, MRD and MPP:

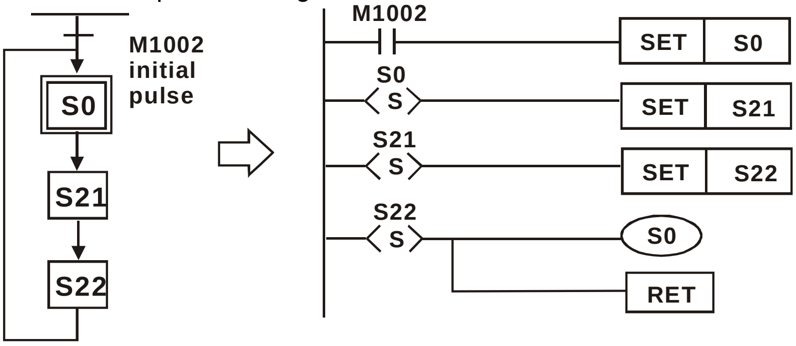


Note: Ladder diagram editor in ISPSoft does not support MPS, MRD and MPP instructions. To achieve the same results as branch instructions, users have to connect all branches to the left hand bus bar.



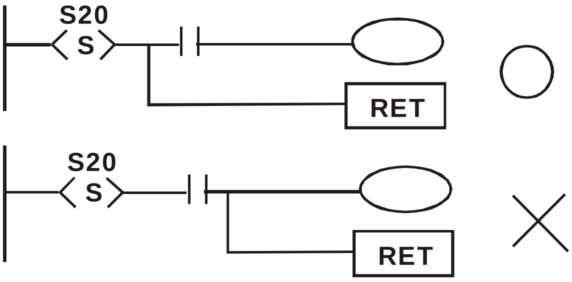
1.5.11 STL (Step Ladder Programming)

STL programming uses step points, e.g. S0 S21, S22, which allow users to program in a clearer and understandable way as drawing a flow chart. The program will proceed to next step only if the previous step is completed, therefore it forms a sequential control process similar to SFC (Sequential Function Chart) mode. The STL sequence can be converted into a PLC ladder diagram which is called “step ladder diagram” as below.



1.5.12 RET (Return)

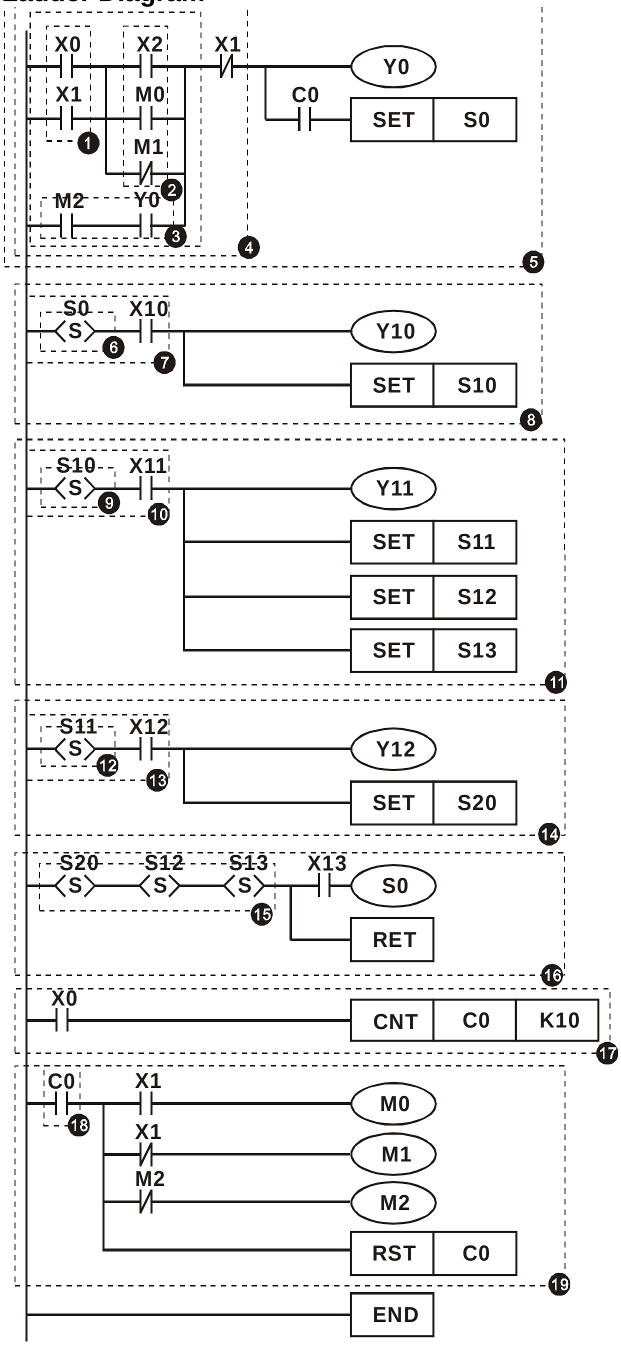
RET instruction has to be placed at the end of sequential control process to indicate the completion of STL flow.



Note: Always connect RET instruction immediately after the last step point indicated as the above diagram otherwise program error may occur.

1.6 Conversion between Ladder Diagram and Instruction List Mode

Ladder Diagram



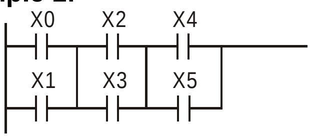
Instruction

LD	X0	❶ OR block
OR	X1	
LD	X2	❷ OR block
OR	M0	
ORI	M1	
ANB		Block in series
LD	M2	❸ AND block
AND	Y0	
ORB		Block in parallel
ANI	X1	❹ ANI
OUT	Y0	Multiple outputs; ❺ The output continues based on status of ❹
AND	C0	
SET	S0	
STL	S0	Start of step ladder ❻; ❼; ❽
LD	X10	S0 status operates with X10
OUT	Y10	Output Y10 and transfer of step point
SET	S10	
STL	S10	❾ Read S10 status; ⓫
LD	X11	❿ S10 operates with X11
OUT	Y11	Output Y11 and transfer of step points
SET	S11	
SET	S12	
SET	S13	
STL	S11	⓬ Read S11 status; ⓮
LD	X12	⓭ S11 operates with X12
OUT	Y12	Output Y12 and transfer of step points
SET	S20	
STL	S20	⓯ Convergence of multiple status; ⓰ End of step ladder
STL	S12	
STL	S13	
LD	X13	Read X13 status and transfer of step point
OUT	S0	
RET		Return
LD	X0	⓱
CNT	C0 K10	
LD	C0	⓲ Read C0; ⓳
MPS		Multiple outputs
AND	X1	
OUT	M0	
MRD		
ANI	X1	
OUT	M1	
MPP		
ANI	M2	
OUT	M2	
RST	C0	
END		End of program

1.7 Fuzzy Syntax

Generally, the ladder diagram programming is conducted according to the “up to down and left to right” principle. However, some programming methods not following this principle still perform the same control results. Here are some examples explaining this kind of “fuzzy syntax.”

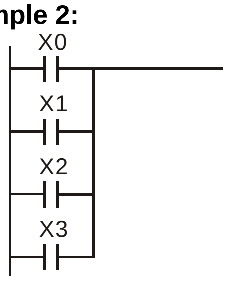
Example 1:



Better method		OK method	
LD	X0	LD	X0
OR	X1	OR	X1
LD	X2	LD	X2
OR	X3	OR	X3
ANB		LD	X4
LD	X4	OR	X5
OR	X5	ANB	
ANB		ANB	

The two instruction programs can be converted into the same ladder diagram. The difference between Better and OK method is the ANB operation conducted by MPU. ANB instruction cannot be used continuously for more than 8 times. If more than 8 ANB instructions are used continuously, program error will occur. Therefore, apply ANB instruction after a block is made is the better method to prevent the possible errors. In addition, it’s also the more logical and clearer programming method for general users.

Example 2:

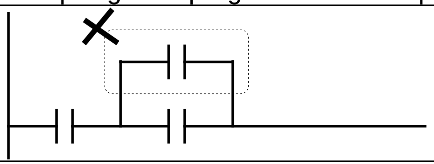
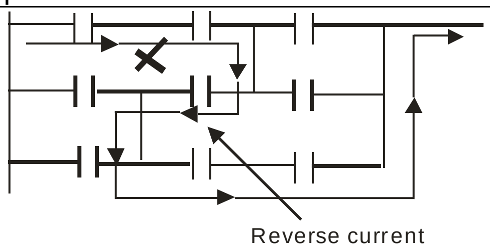
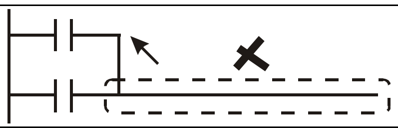


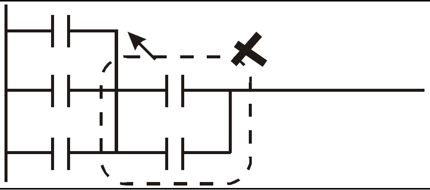
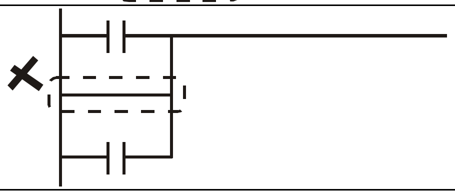
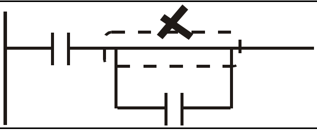
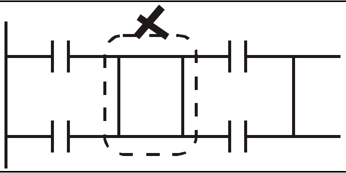
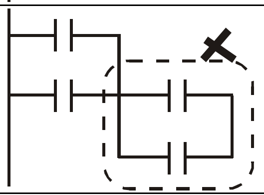
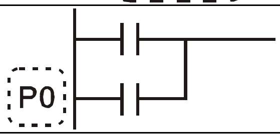
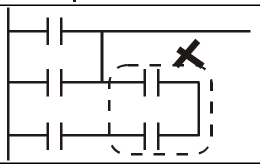
Good method		Bad method	
LD	X0	LD	X0
OR	X1	LD	X1
OR	X2	LD	X2
OR	X3	LD	X3
		ORB	
		ORB	
		ORB	

The difference between Good and Bad method is very clear. With longer program code, the required MPU operation memory increases in the Bad method. To sum up, following the general principle and applying good / better method when editing programs prevents possible errors and improves program execution speed as well.

Common Programming Errors

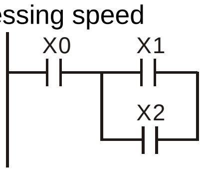
PLC processes the diagram program from up to down and left to right. When editing ladder diagram users should adopt this principle as well otherwise an error would be detected by WPLSoft when compiling user program. Common program errors are listed below:

	OR operation upward is not allowed.
	“Reverse current” exists.
	Output should be connected on top of the circuit.

	Block combination should be made on top of the circuit.
	Parallel connection with empty device is not allowed..
	Parallel connection with empty device is not allowed.
	No device in the middle block.
	Devices and blocks in series should be horizontally aligned
	Label P0 should be at the first row of the complete network.
	“Reverse current” exists

1.8 Correcting Ladder Diagram

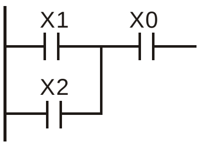
Example 1:
Connect the block to the front for omitting ANB instruction because simplified program improves processing speed



Instruction List

LD	X0
LD	X1
OR	X2
ANB	

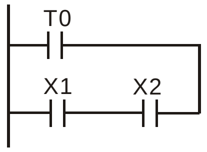
⇩



Instruction List

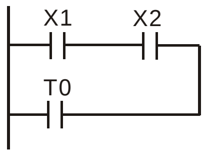
LD	X1
OR	X2
AND	X0

Example 2:
When a device is to be connected to a block, connect the device to upper row for omitting ORB instruction



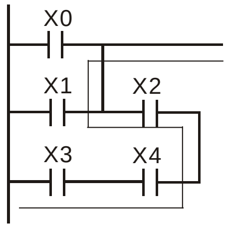
Instruction List
LD T0
LD X1
AND X2
ORB

⇩



Instruction List
LD X1
AND X2
OR T0

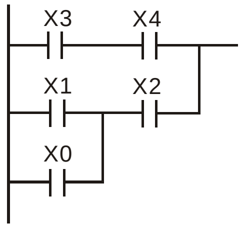
Example 3:
“Reverse current” existed in diagram (a) is not allowed for PLC processing principle.



(a)

Instruction List
LD X0
OR X1
AND X2
LD X3
AND X4
ORB

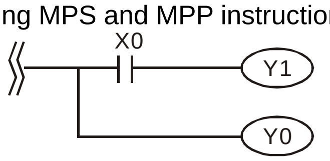
⇩



(b)

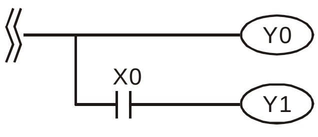
Instruction List
LD X3
AND X4
LD X1
OR X0
AND X2
ORB

Example 4:
For multiple outputs, connect the output without additional input devices to the top of the circuit for omitting MPS and MPP instructions.



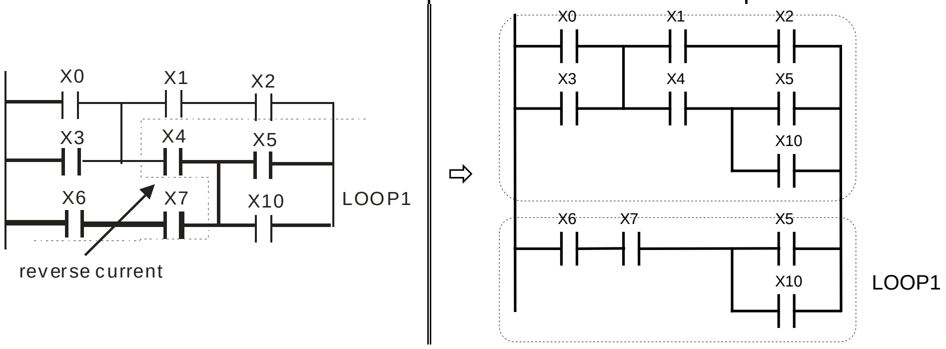
Instruction List
MPS
AND X0
OUT Y1
MPP
OUT Y0

⇩

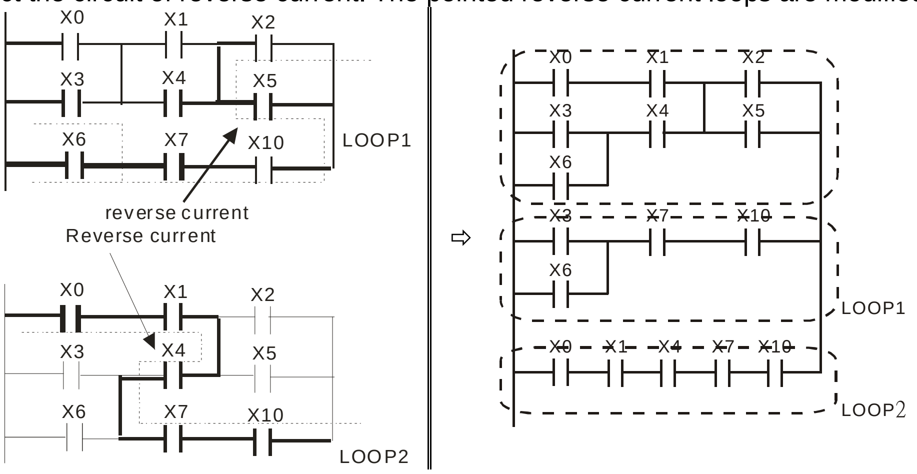


Instruction List
OUT Y0
AND X0
OUT Y1

Example 5:
Correct the circuit of reverse current. The pointed reverse current loops are modified on the right.

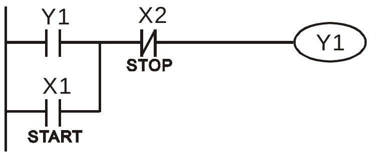


Example 6:
Correct the circuit of reverse current. The pointed reverse current loops are modified on the right.

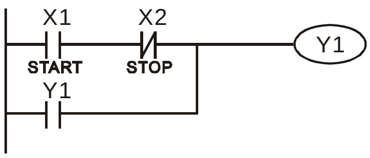


1.9 Basic Program Design Examples

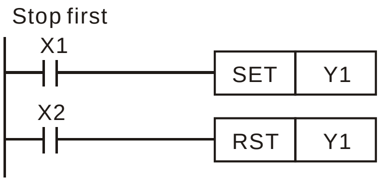
Example 1 - Stop First latched circuit
When X1 (START) = ON and X2 (STOP) = OFF, Y1 will be ON. If X2 is turned on, Y1 will be OFF. This is a Stop First circuit because STOP button has the control priority than START



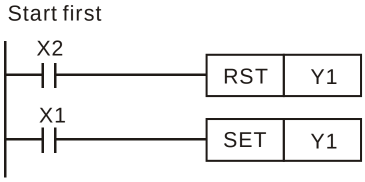
Example 2 - Start First latched circuit
When X1 (START) = ON and X2 (STOP) = OFF, Y1 will be ON and latched. If X2 is turned ON, Y1 remains ON. This is a Start First circuit because START button has the control priority than STOP



Example 3 - Latched circuit of SET and RST
The diagram opposite are latched circuits consist of RST and SET instructions.
In PLC processing principle, the instruction close to the end of the program determines the final output status of Y1. Therefore, if both X1 and X2 are ON, RST which is lower than SET forms a

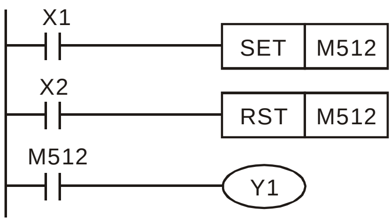


Stop First circuit while SET which is lower than RST forms a Start First circuit.

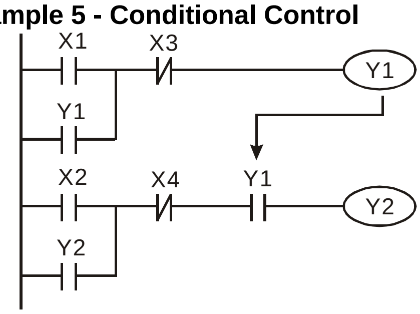


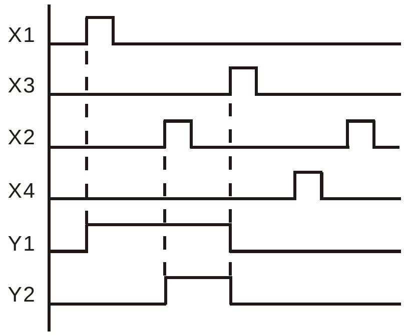
Example 4 - Power down latched circuit

The auxiliary relay M512 is a latched relay. Once X1 is ON, Y1 retains its status before power down and resumes after power up.



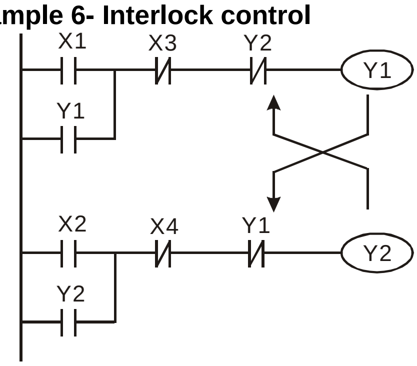
Example 5 - Conditional Control

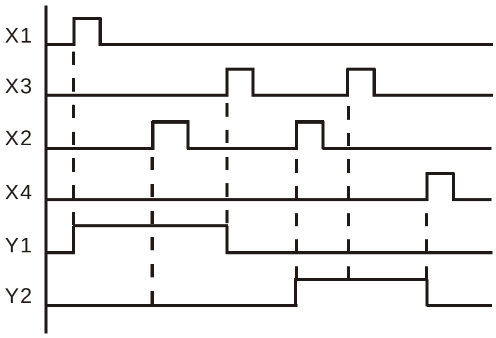




Because NO contact Y1 is connected to the circuit of Y2 output, Y1 becomes one of the conditions for enabling Y2, i.e. for turning on Y2, Y1 has to be ON

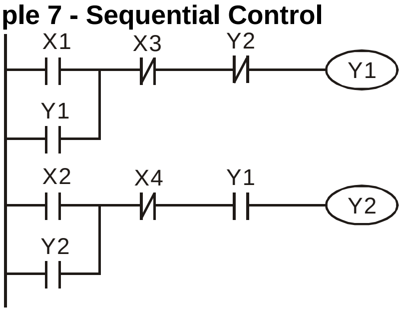
Example 6- Interlock control





NC contact Y1 is connected to Y2 output circuit and NC contact Y2 is connected Y1 output circuit. If Y1 is ON, Y2 will definitely be OFF and vice versa. This forms an Interlock circuit which prevents both outputs to be ON at the same time. Even if both X1 and X2 are ON, in this case only Y1 will be enabled.

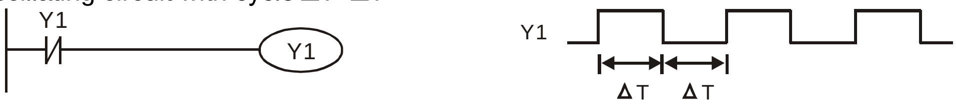
Example 7 - Sequential Control



Connect NC contact Y2 to Y1 output circuit and NO contact Y1 to Y2 output circuit. Y1 becomes one of the conditions to turn on Y2. In addition, Y1 will be OFF when Y2 is ON, which forms an sequential control process.

Example 8 - Oscillating Circuit

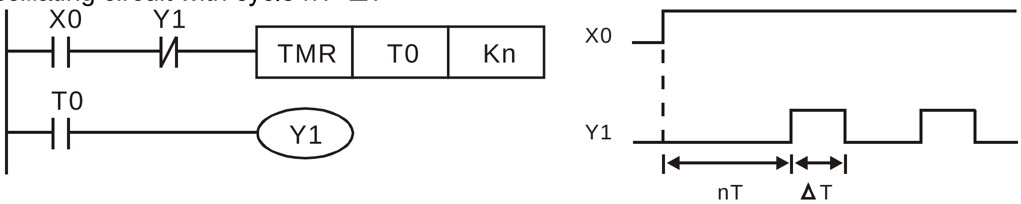
An oscillating circuit with cycle ΔT+ΔT



In the first scan, Y1 turns on. In the second scan, Y1 turns off due to the reversed state of contact Y1. Y1 output status changes in every scan and forms an oscillating circuit with output cycleΔT(ON)+ΔT(OFF)

Example 9 – Oscillating Circuit with Timer

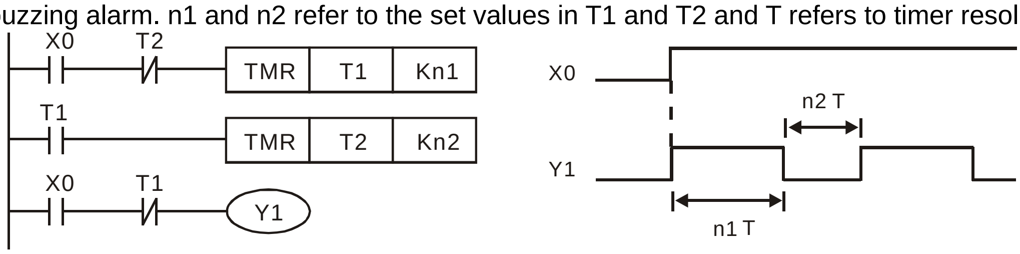
An oscillating circuit with cycle nT+ΔT



When X0 = ON, T0 starts timing (nT). Once the set time is reached, contact T0 = ON to enable Y1(ΔT). In next scan, Timer T0 is reset due to the reversed status of contact Y1. Therefore contact T0 is reset and Y1 = OFF. In next scan, T0 starts timing again. The process forms an oscillating circuit with output cycle nT+ΔT.

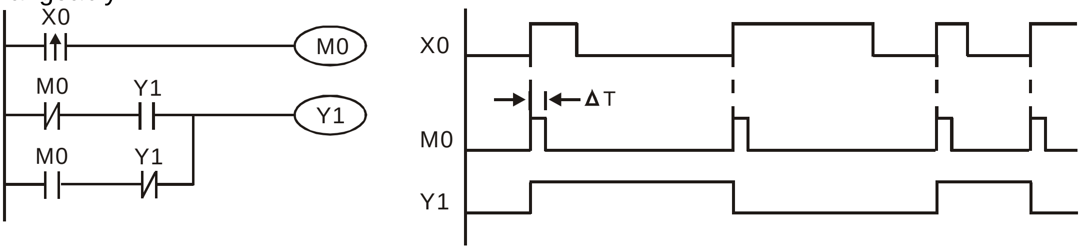
Example 10 - Flashing Circuit

The ladder diagram uses two timers to form an oscillating circuit which enables a flashing indicator or a buzzing alarm. n1 and n2 refer to the set values in T1 and T2 and T refers to timer resolution.



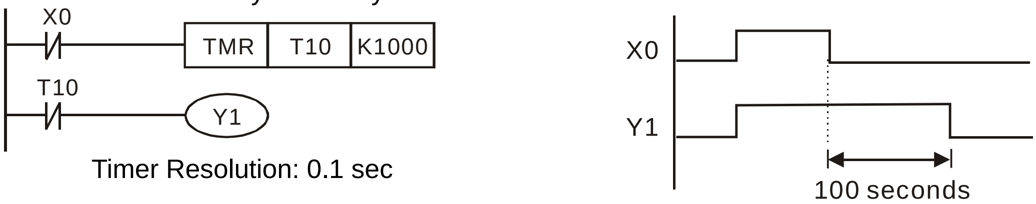
Example 11 - Trigger Circuit

In this diagram, rising-edge contact X0 generates trigger pulses to control two actions executing interchangeably.



Example 12 - Delay OFF Circuit

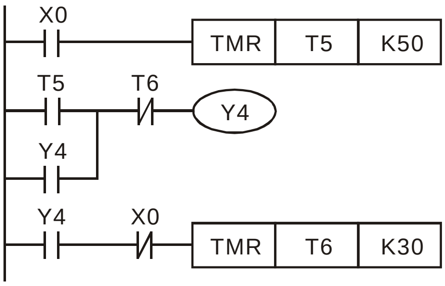
If X0 = ON, timer T10 is not energized but coil Y1 is ON. When X0 is OFF, T10 is activated. After 100 seconds (K1000 × 0.1 sec = 100 sec), NC contact T10 is ON to turn off Y1. Turn-off action is delayed for 100 seconds by this delay OFF circuit.

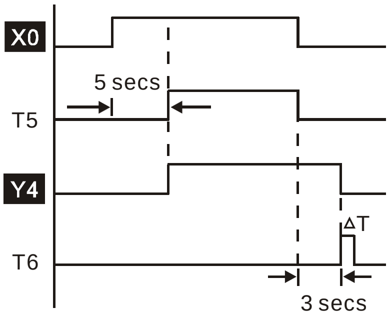


Timer Resolution: 0.1 sec

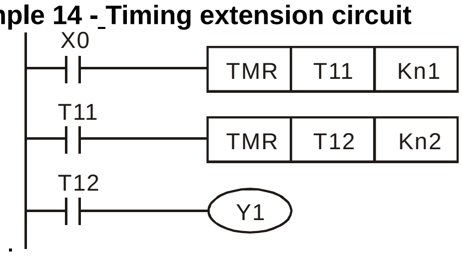
Example 13 - Output delay circuit

The output delay circuit is composed of two timers executing delay actions. No matter input X0 is ON or OFF, output Y4 will be delayed.



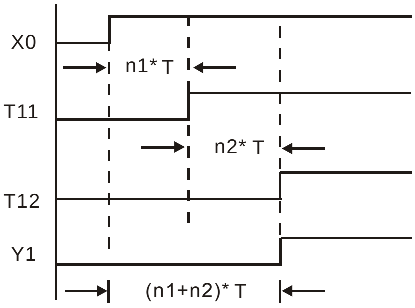


Example 14 - Timing extension circuit

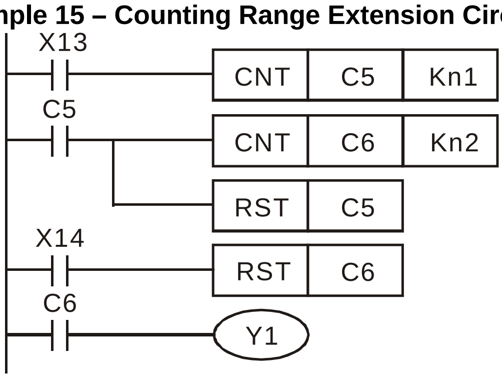


Timer = T11, T12
Timer resolution: T

The total delay time: (n1+n2)* T. T refers to the timer resolution.



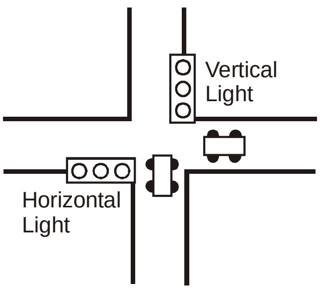
Example 15 – Counting Range Extension Circuit



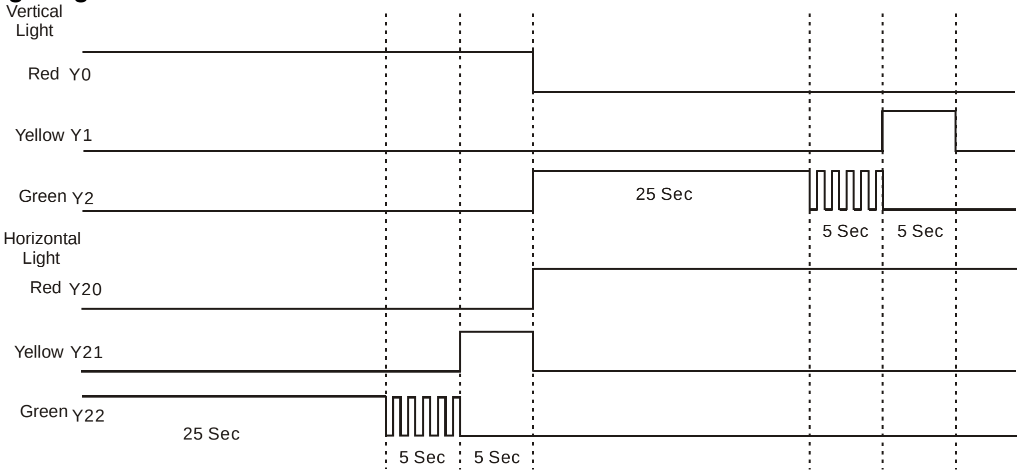
The counting range of a 16-bit counter is 0 ~ 32,767. The opposite circuit uses two counters to increase the counting range as n1*n2. When value in counter C6 reaches n2, The pulses counted from X13 will be n1*n2.

Example 16 - Traffic light control (Step Ladder Logic)
Traffic light control

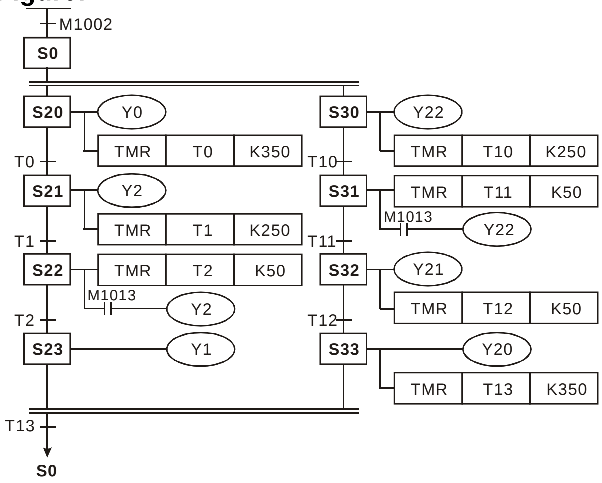
	Red light	Yellow light	Green light	Green light blinking
Vertical light	Y0	Y1	Y2	Y2
Horizontal light	Y20	Y21	Y22	Y22
Light Time	35 Sec	5 Sec	25 Sec	5 Sec



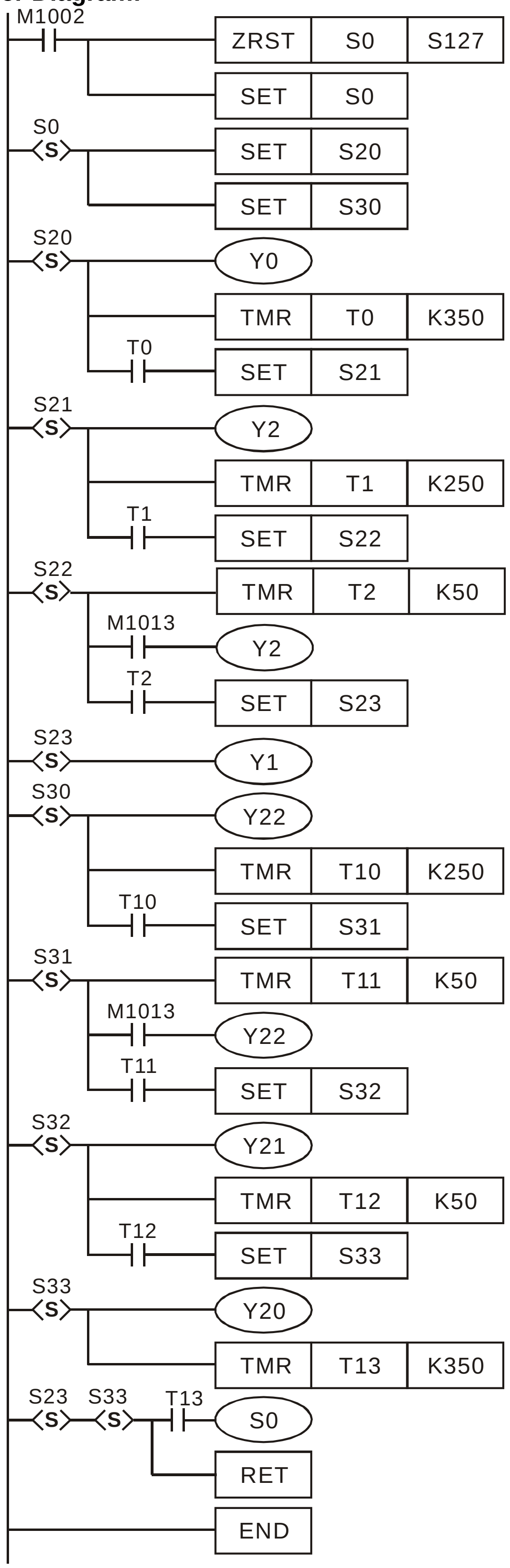
Timing Diagram:



SFC Figure:



Ladder Diagram:



WPLSoft programming (SFC mode)

SFC logic	Internal Ladder Logic