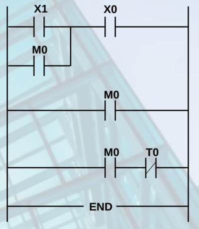


Logic and PLC Ladder diagrams

Dr Diarmuid Ó Briain

Learning objectives

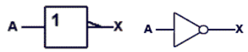
- Logic Circuits
- PLC Operating Principles
- PLC Control Features
- Ladder Diagrams.

Logic Circuits

IEC	ANSI	Description	Boolean
&	AND	The AND gate output is at logic 1 when, and only when all its inputs are at logic 1, otherwise the output is at logic 0.	$X = A \cdot B$
≥1	OR	The OR gate output is at logic 1 when one or more of its inputs are at logic 1. If all the inputs are at logic 0, the output is at logic 0.	$X = A + B$
&	NAND	The NAND Gate output is at logic 0 when, and only when all its inputs are at logic 1, otherwise the output is at logic 1.	$X = \overline{A \cdot B}$
≥1	NOR	The NOR gate output is at logic 0 when one or more of its inputs are at logic 1. If all the inputs are at logic 0, the output is at logic 1.	$X = \overline{A + B}$
=1	XOR	The XOR gate output is at logic 1 when one and ONLY ONE of its inputs is at logic 1. Otherwise the output is logic 0.	$X = A \oplus B$
=1	XNOR	The XNOR gate output is at logic 0 when one and ONLY ONE of its inputs is at logic 1. Otherwise the output is logic 1. (It is similar to the XOR gate, but its output is inverted).	$X = \overline{A \oplus B}$
1	NOT	The NOT gate output is at logic 0 when its only input is at logic 1, and at logic 1 when its only input is at logic 0. For this reason it is often called an INVERTER.	$X = \overline{A}$

- International Electrotechnical Commission (IEC)
- American National Standards Institute (ANSI)

Logic Circuits – N/OT (Inverter)



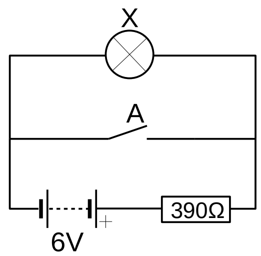
Truth table

A	X | $\overline{A}$
F	T
T	F

Binary Truth table

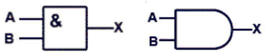
A	X | $\overline{A}$
0	1
1	0

$\overline{A}$ means N/OT A



- Bulb lit when switch is not pressed.
- Bulb is not lit when switch is pressed.
- “X” is the opposite state to “A”.

Logic Circuits - AND



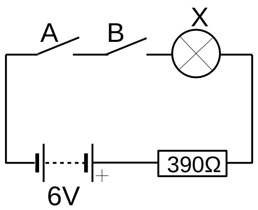
Truth table

A	B	X | A.B
F	F	F
F	T	F
T	F	F
T	T	T

Binary Truth table

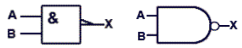
A	B	X | A.B
0	0	0
0	1	0
1	0	0
1	1	1

A.B means "A" AND "B"



- Bulb will only light if "A" AND "B" are depressed together.

Logic Circuits - NAND



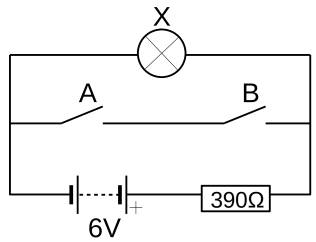
Truth table

A	B	X | $\overline{A.B}$
F	F	T
F	T	T
T	F	T
T	T	F

Binary Truth table

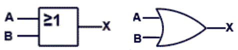
A	B	X | $\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

$\overline{A.B}$ means N/OT "A" AND "B"



- Bulb will always light unless "A" AND "B" are depressed together.

Logic Circuits - OR



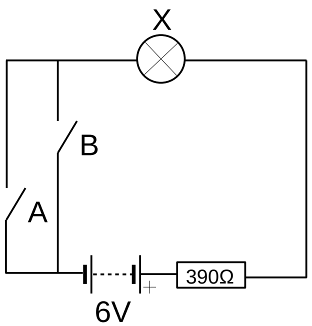
Truth table

A	B	X | A+B
F	F	F
F	T	T
T	F	T
T	T	T

Binary Truth table

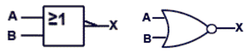
A	B	X | A+B
0	0	0
0	1	1
1	0	1
1	1	1

A+B means "A" OR "B"



- Bulb will light if either "A" OR "B" are depressed.

Logic Circuits - NOR



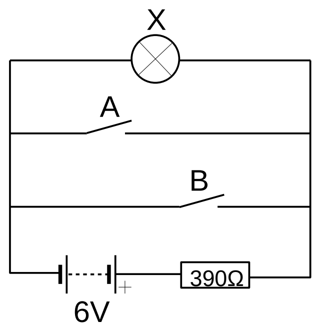
Truth table

A	B	X | $\overline{A+B}$
F	F	T
F	T	F
T	F	F
T	T	F

Binary Truth table

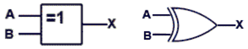
A	B	X | $\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

$\overline{A+B}$ means "A" N/OR "B"



- Bulb will light unless either "A" or "B" or both are depressed.

Logic Circuits - XOR



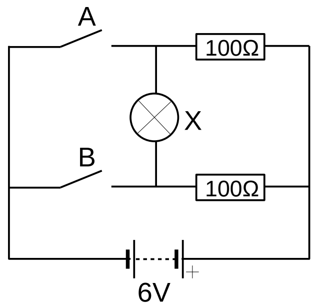
Truth table

A	B	X | A⊕B
F	F	F
F	T	T
T	F	T
T	T	F

Binary Truth table

A	B	X| A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

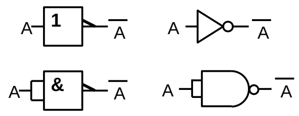
A⊕B means “A” XOR “B”



- Bulb will light if either “A” OR “B” are depressed.

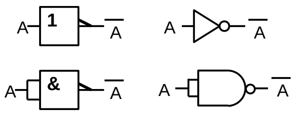
N/OT gates using NAND gates

N/OT

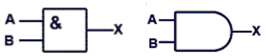


N/OT gates using NAND gates

N/OT



AND

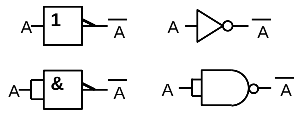


Binary Truth table

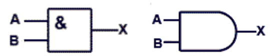
A	B	X | A.B
0	0	0
0	1	0
1	0	0
1	1	1

N/OT gates using NAND gates

N/OT

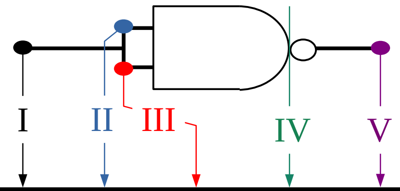


AND



Binary Truth table

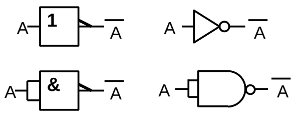
A	B	X | A.B
0	0	0
0	1	0
1	0	0
1	1	1



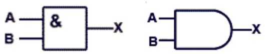
I	II	III	IV	V
0				
0				
1				
1				

N/OT gates using NAND gates

N/OT

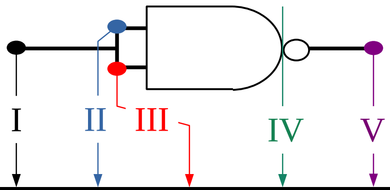


AND



Binary Truth table

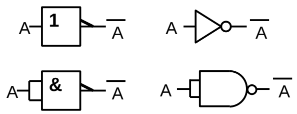
A	B	X | A.B
0	0	0
0	1	0
1	0	0
1	1	1



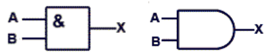
I	II	III	IV	V
0	0			
0	0			
1	1			
1	1			

N/OT gates using NAND gates

N/OT

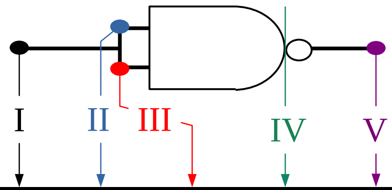


AND



Binary Truth table

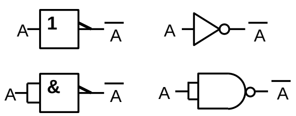
A	B	X | A.B
0	0	0
0	1	0
1	0	0
1	1	1



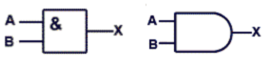
I	II	III	IV	V
0	0	0		
0	0	0		
1	1	1		
1	1	1		

N/OT gates using NAND gates

N/OT

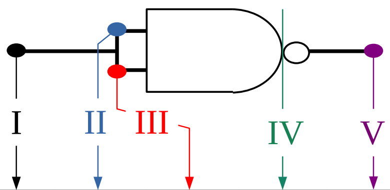


AND



Binary Truth table

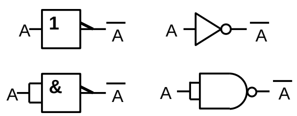
A	B	X | A.B
0	0	0
0	1	0
1	0	0
1	1	1



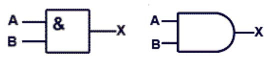
I	II	III	IV	V
0	0	0	0	
0	0	0	0	
1	1	1	1	
1	1	1	1	

N/OT gates using NAND gates

N/OT

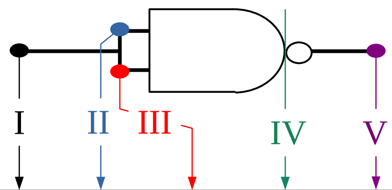


AND



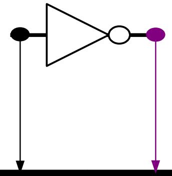
Binary Truth table

A	B	X | A.B
0	0	0
0	1	0
1	0	0
1	1	1

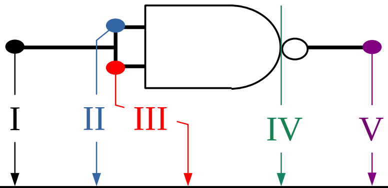


I	II	III	IV	V
0	0	0	0	1
0	0	0	0	1
1	1	1	1	0
1	1	1	1	0

N/OT gates using NAND gates



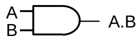
A	X | $\overline{A}$
0	1
1	0



I	II	III	IV	V
0	0	0	0	1
0	0	0	0	1
1	1	1	1	0
1	1	1	1	0

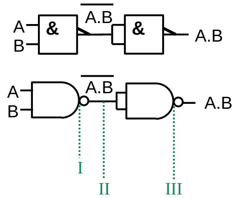
AND gates using NAND gates

AND



A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

AND from NAND gates



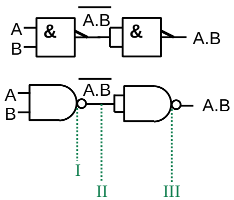
A	B	I	II	III	A.B
0	0				
0	1				
1	0				
1	1				

AND gates using NAND gates

AND

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

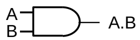
AND from NAND gates



A	B	I	II	III	A.B
0	0	0			
0	1	0			
1	0	0			
1	1	1			

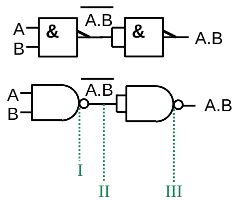
AND gates using NAND gates

AND



A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

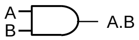
AND from NAND gates



A	B	I	II	III	A.B
0	0	0	1		
0	1	0	1		
1	0	0	1		
1	1	1	0		

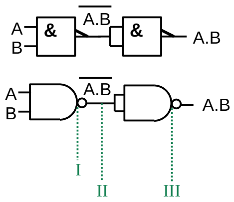
AND gates using NAND gates

AND



A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

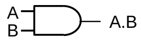
AND from NAND gates



A	B	I	II	III	A.B
0	0	0	1	1	
0	1	0	1	1	
1	0	0	1	1	
1	1	1	0	0	

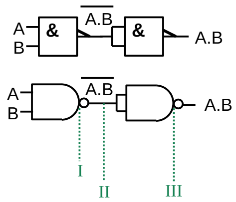
AND gates using NAND gates

AND



A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

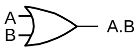
AND from NAND gates



A	B	I	II	III	A.B
0	0	0	1	1	0
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	0	0	1

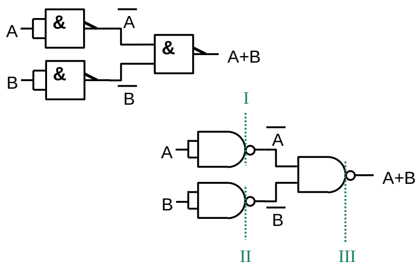
OR gates using NAND gates

OR



A	B	A.B
0	0	0
0	1	1
1	0	1
1	1	1

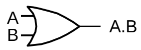
OR from NAND gates



A	I	$\overline{A}$	B	II	$\overline{B}$	III	A+B
0			0				
0			1				
1			0				
1			1				

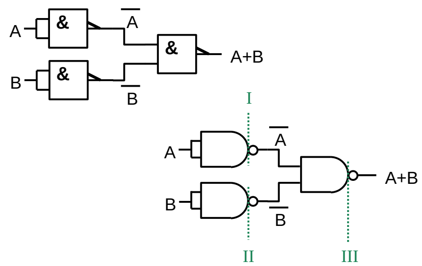
OR gates using NAND gates

OR



A	B	A.B
0	0	0
0	1	1
1	0	1
1	1	1

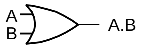
OR from NAND gates



A	I	$\overline{A}$	B	II	$\overline{B}$	III	A+B
0	0	1	0				
0	0	1	1				
1	1	0	0				
1	1	0	1				

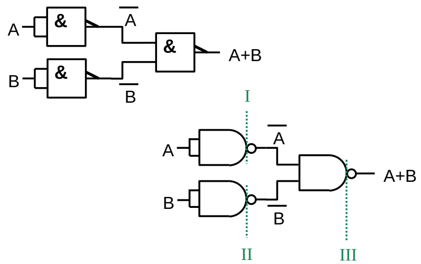
OR gates using NAND gates

OR



A	B	A.B
0	0	0
0	1	1
1	0	1
1	1	1

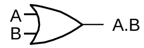
OR from NAND gates



A	I	$\overline{A}$	B	II	$\overline{B}$	III	A+B
0	0	1	0	0	1		
0	0	1	1	1	0		
1	1	0	0	0	1		
1	1	0	1	1	0		

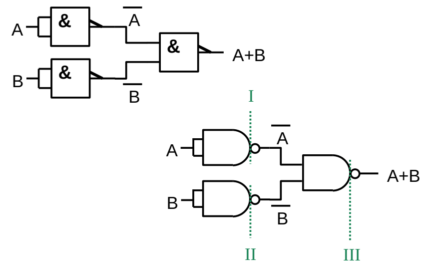
OR gates using NAND gates

OR



A	B	A.B
0	0	0
0	1	1
1	0	1
1	1	1

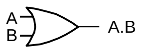
OR from NAND gates



A	I	$\overline{A}$	B	II	$\overline{B}$	III	A+B
0	0	1	0	0	1	1	
0	0	1	1	1	0	0	
1	1	0	0	0	1	0	
1	1	0	1	1	0	0	

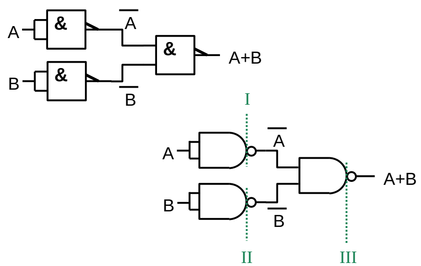
OR gates using NAND gates

OR



A	B	A.B
0	0	0
0	1	1
1	0	1
1	1	1

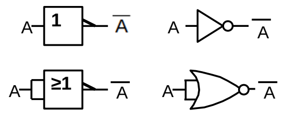
OR from NAND gates



A	I	$\overline{A}$	B	II	$\overline{B}$	III	A+B
0	0	1	0	0	1	1	0
0	0	1	1	1	0	0	1
1	1	0	0	0	1	0	1
1	1	0	1	1	0	0	1

All gates using NOR gates

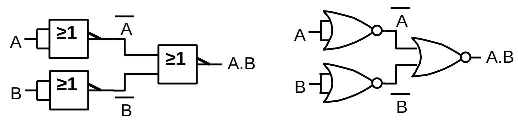
N/OT



Binary Truth table

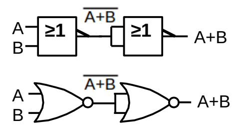
A	B	$\overline{A+B}$
0	0	0
0	1	1
1	0	1
1	1	1

AND



A	B	$\overline{A}$	$\overline{B}$	A.B
0	0	1	1	0
0	1	1	0	0
1	0	0	1	0
1	1	0	0	1

OR

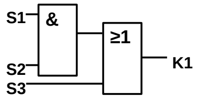


A	B	$\overline{A+B}$	A+B
0	0	1	0
0	1	0	1
1	0	0	1
1	1	0	1

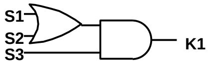
Exercise #1

7

1. Draw the equivalent circuit for the following logic gates

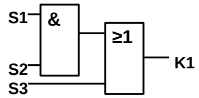


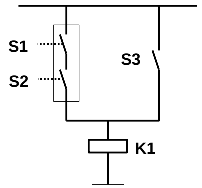
2. Draw the equivalent circuit for the following logic gates

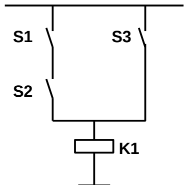


Exercise #1

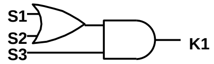
1. Draw the equivalent circuit for the following logic gates

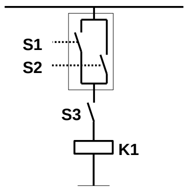


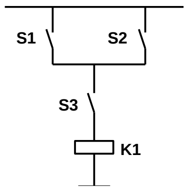




2. Draw the equivalent circuit for the following logic gates



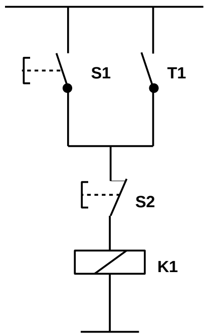




Exercise #2

7

Draw the logic gate combination to represent the following circuit diagram



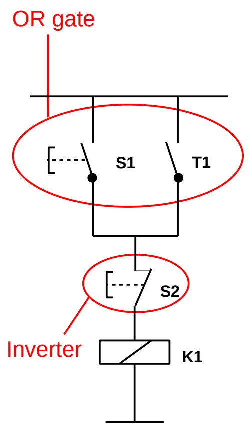
Truth table

2^3	S1	T1	S2			K1
0						0
1						0
2						1
3						0
4						1
5						0
6						1
7						0

S2 button not pressed conducts, button pressed short-circuit so it acts like an inverter.

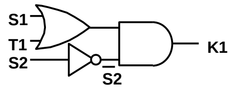
Exercise #2

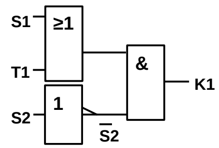
Draw the logic gate combination to represent the following circuit diagram



Truth table

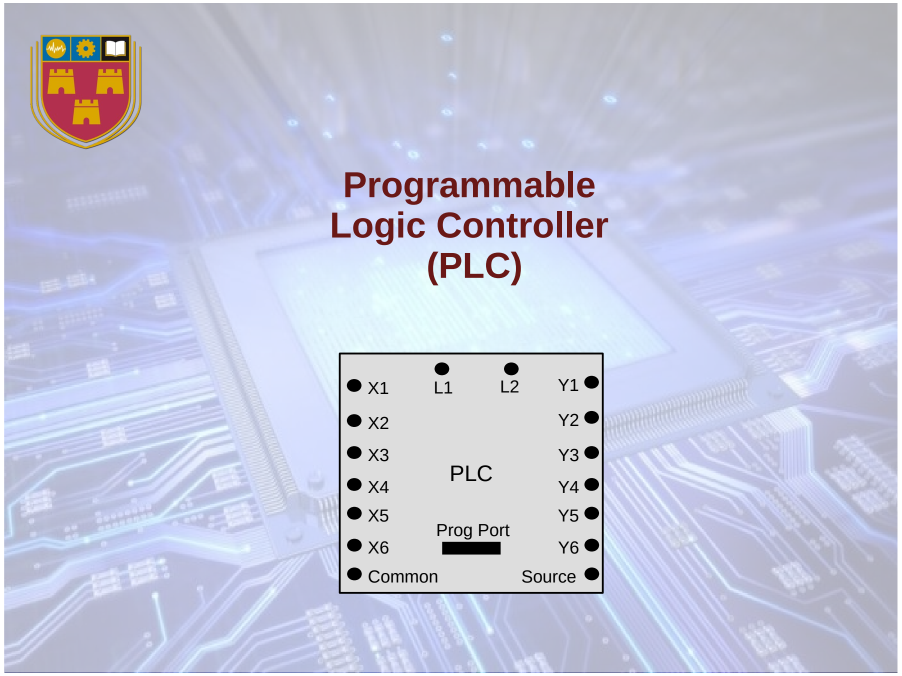
2^3	S1	T1	S2	X|OR	$\overline{S2}$	K1
0	0	0	0	0	1	0
1	0	0	1	0	0	0
2	0	1	0	1	1	1
3	0	1	1	1	0	0
4	1	0	0	1	1	1
5	1	0	1	1	0	0
6	1	1	0	1	1	1
7	1	1	1	1	0	0



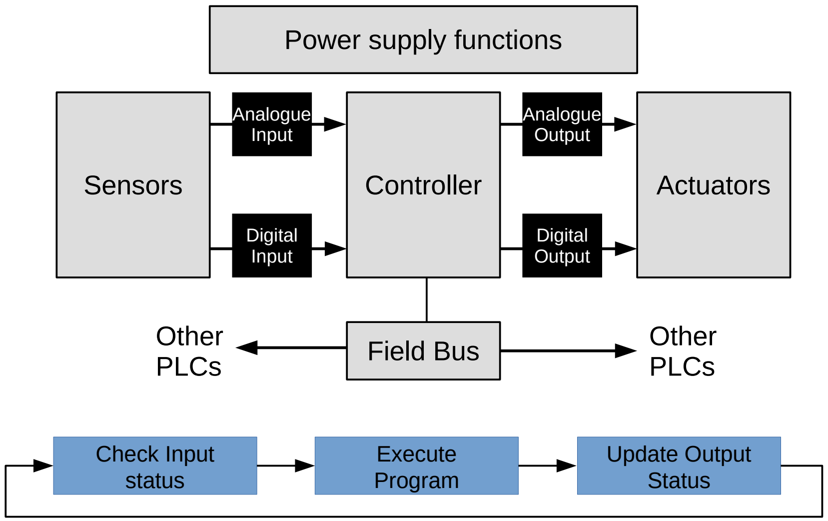


S2 button not pressed conducts, button pressed short-circuit so it acts like an inverter.

Programmable Logic Controller (PLC)

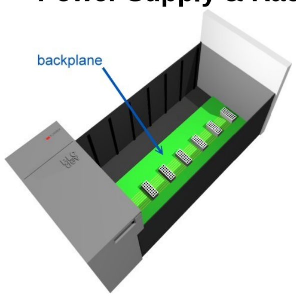


Programmable Logic Controller (PLC)

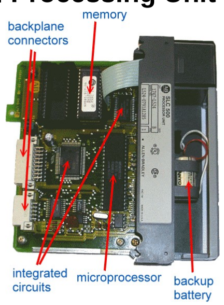


PLC

Power Supply & Rack

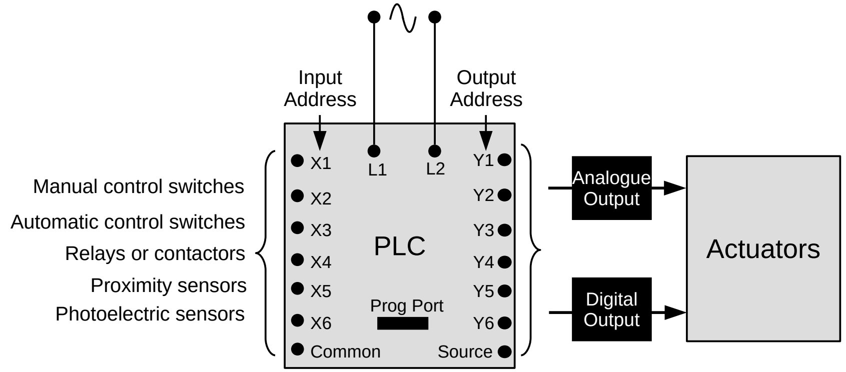


Controller / Central Processing Unit (CPU)



Input/Output(I/O) Sub-System

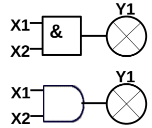
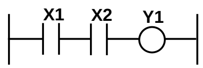
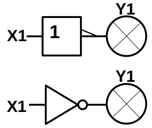
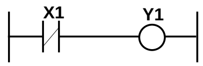
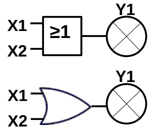
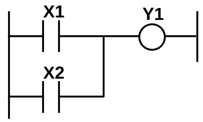
PLC inputs



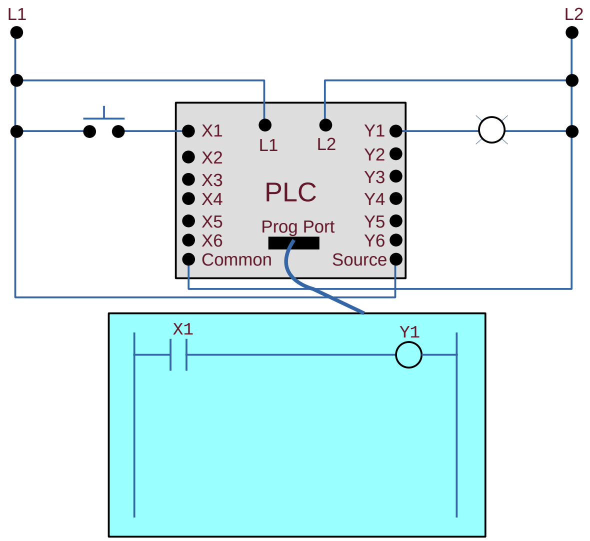
Ladder diagram features

- Power flows from left to right.
- Output on right side can not be connected directly with left side.
- Contact can not be placed on the right of output.
- Each rung contains one output at least.
- Each output can be used only once in the program.
- A particular input and/or output can appear in more than one rung of a ladder.
- The inputs and/or outputs are all identified by their addresses, the notation used depending on the PLC manufacturer.

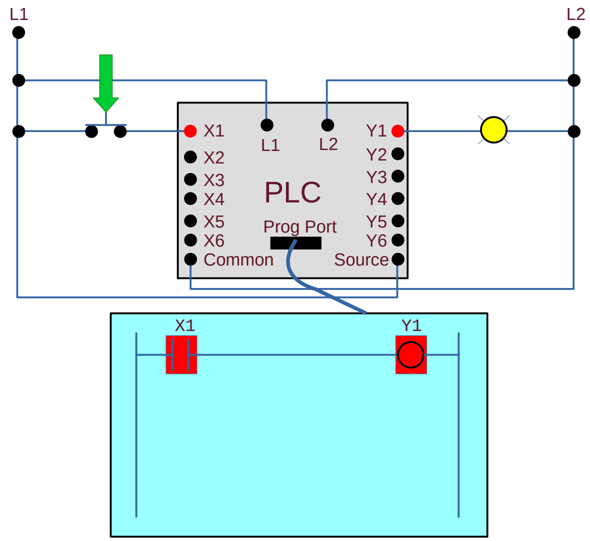
Ladder Logic Function Block Diagrams

	Functional Block Diagram	Ladder Diagram	Program List
AND			LD X1 AND X2 OUT Y1 END
NOT			LDI X1 OUT Y1 END
OR			LD X1 OR X2 OUT Y1 END

Ladder Logic



Ladder Logic



Ladder Logic

- Logical **AND**
  - Switch 1 AND Switch 2 must be pressed for current to flow to the motor

```
------[ ]--------------[ ]----------------( )
   Switch 1          Switch 2             Motor
```

- Logical **AND** with N/O**T**
  - Door switch 1 AND N/OT Obstruction to run the motor (to close door)

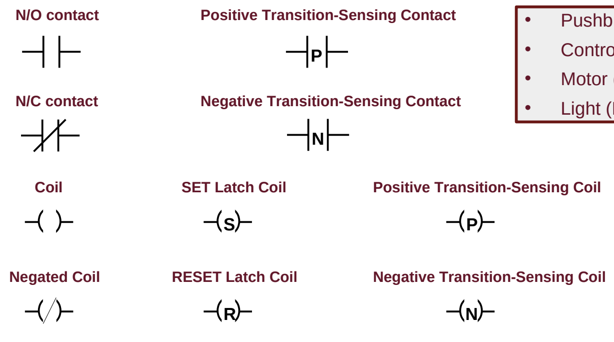
```
------[ ]--------------[\]----------------( )
  Door switch       Obstruction           Motor
```

- Logical **OR**
  - Either the Inside OR the Outside switch must be depressed to activate motor

```
--+-------[ ]-------+-----------------( )
  |  Inside switch  |                 Motor
  |                 |
  +-------[ ]-------+
     Outside switch
```

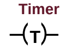

Ladder Logic

- A programming language that represents a program via a graphical diagram based on the circuit diagrams of relay logic hardware.
- Ladder logic is used to develop software for PLCs used in industrial control applications.



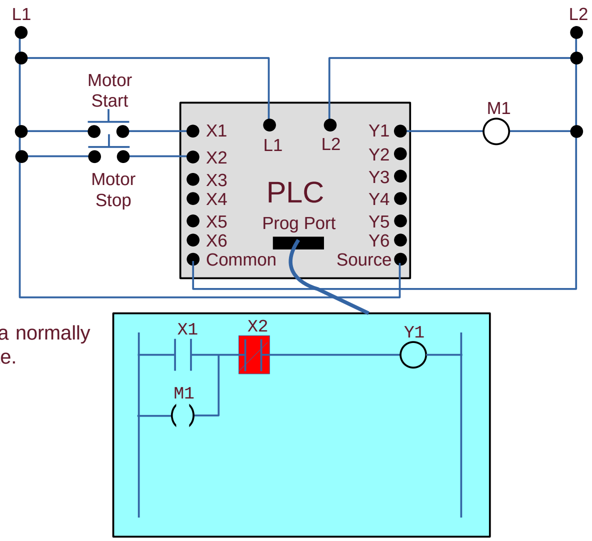
- Pushbuttons (PB)
- Control Relay (CR)
- Motor (M)
- Light (L)

Timer



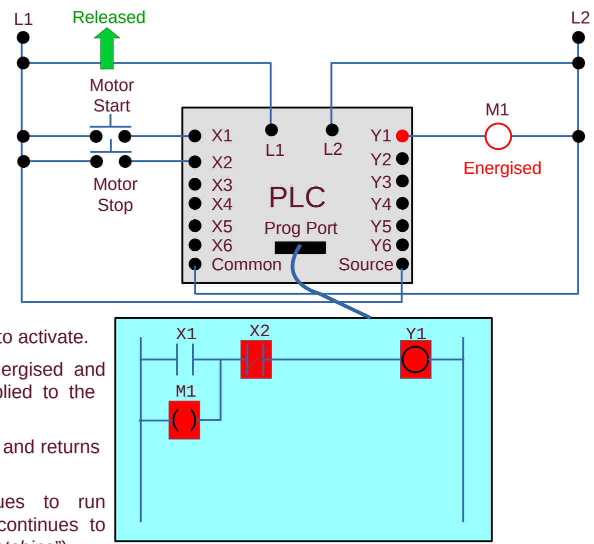
Normally with number of 'ticks' beside it. A tick is typically 100ms. i.e. K50 ~ 100ms x 50 = 5,000ms = 5 sec

Ladder Logic



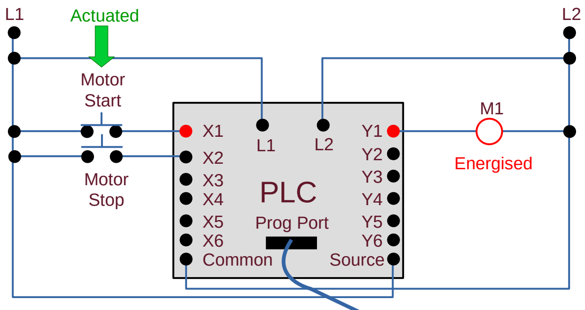
- X2 is in in a normally Closed state.

Ladder Logic

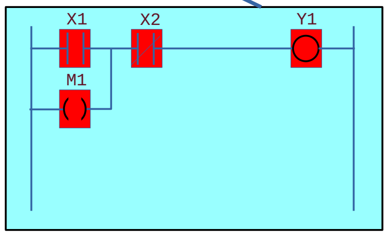


- X1 is pressed to activate.
- Coil M1 is energised and voltage is applied to the motor Y1.
- X1 is released and returns to open state.
- Motor continues to run because M1 continues to power rung ("*latching*").

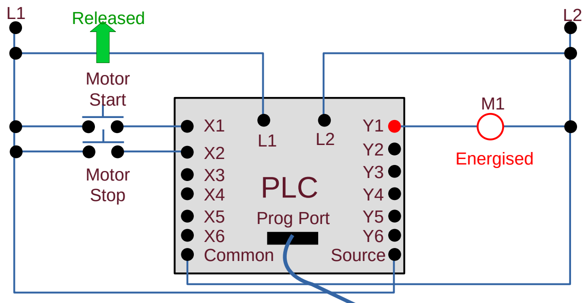
Ladder Logic



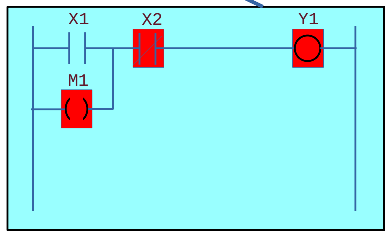
- X1 is pressed to activate.
- Coil M1 is energised and voltage is applied to the motor Y1.



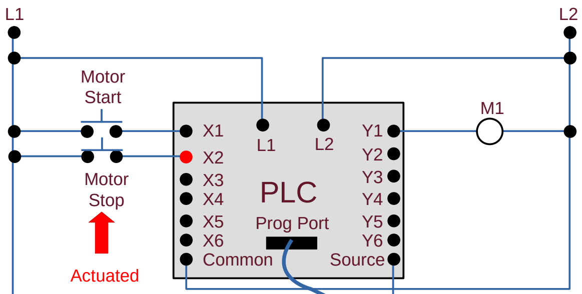
Ladder Logic



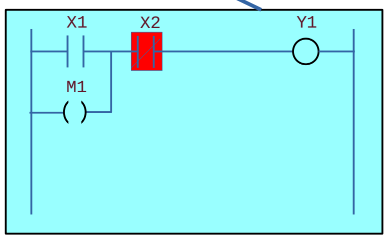
- X1 is pressed to activate.
- Coil M1 is energised and voltage is applied to the motor Y1.
- X1 is released and returns to open state.
- Motor continues to run because ‘seal-in’ contact continues to power Y1.



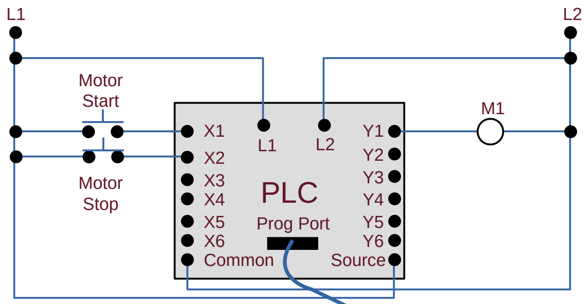
Ladder Logic



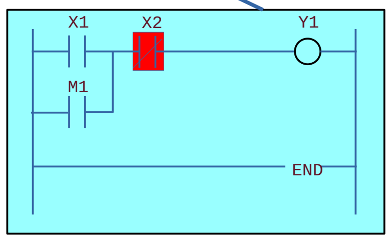
- X2 is actuated and then released.
- X2 changes to an open state breaking continuity to M1.
- X2 returns to Normally Closed state but motor does not start until X1 is actuated again.



Ladder Logic



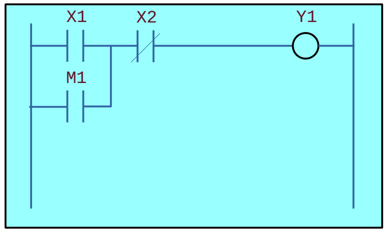
- It is typical also that Control Relays (CR) are illustrated as N/O or N/C contact.
- Diagram is normally completed with a rung with END added as shown.



PLC Programming, Instruction Set

Instruction	Abbreviation
Load, Normally Open (N/O) contact	LD
Load, Normally Closed (NC) contact	LDI
AND; N/O contact in series	AND
AND; N/C contact in series	ANI
OR; N/O contact in parallel	OR
OR; N/C contact in parallel	ORI
Connect block in series	ANB
Connect block in parallel	ORB
Timer On Delay	TON
Timer Off Delay	TOF
Retentive Timer	RTO
Output	OUT

Program the PLC




```
0   LD  X1   Load N/O contact on input X1
1   OR  M1   Define M1 Control Relay in OR block with X1
2   ANB -    Connect block to next block in series
3   LDI X2   Load N/C contact on input X2
4   OUT Y1   Current is supplied to motor on Y1
9   END
```


Exercise #1

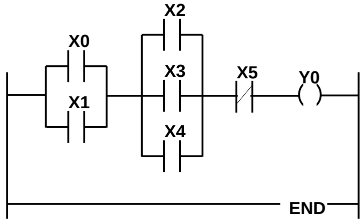
1. Draw a ladder diagram represented by the following instruction list


```
0   LD  X0
1   OR  X1
2   ANB -
3   LD  X2
4   OR  X3
5   OR  X4
6   ANB -
7   ANI X5
8   OUT Y0
9   END
```

Exercise #1

1. Draw a ladder diagram represented by the following instruction list

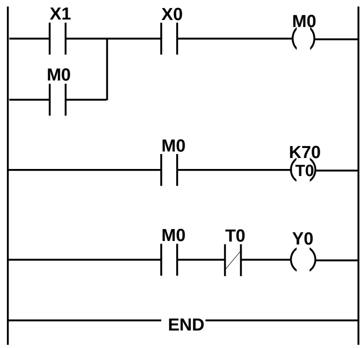
```
0   LD  X0
1   OR  X1
2   ANB -
3   LD  X2
4   OR  X3
5   OR  X4
6   ANB -
7   ANI X5
8   OUT Y0
9   END
```



Ladder diagram example

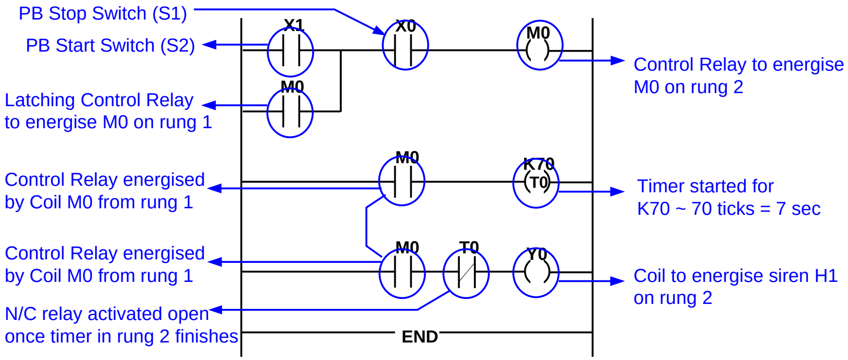
The figure and table represent a ladder diagram and the input/output schedule for a process.

Draw a circuit diagram of the process and describe its operation.



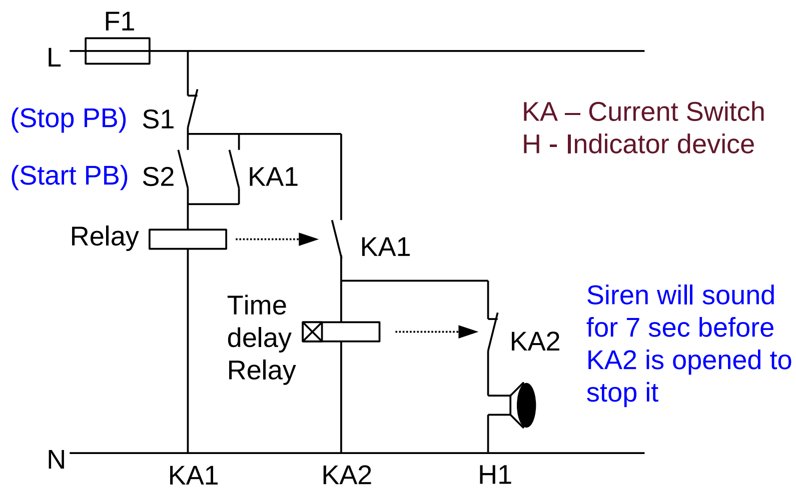
Input / Output Schedule					
Inputs			Outputs		
Device	Item desc	PLC term no	Device	Item desc	PLC term no
Stop PB	S1	X0	Siren	H1	Y0
Start PB	S2	X1			

Ladder diagram example

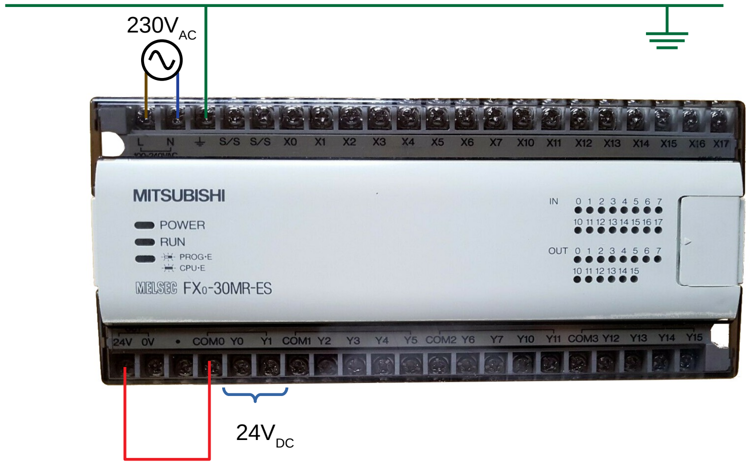


Input / Output Schedule					
Inputs			Outputs		
Device	Item desc	PLC term no	Device	Item desc	PLC term no
Stop PB	S1	X0	Siren	H1	Y0
Start PB	S2	X1			

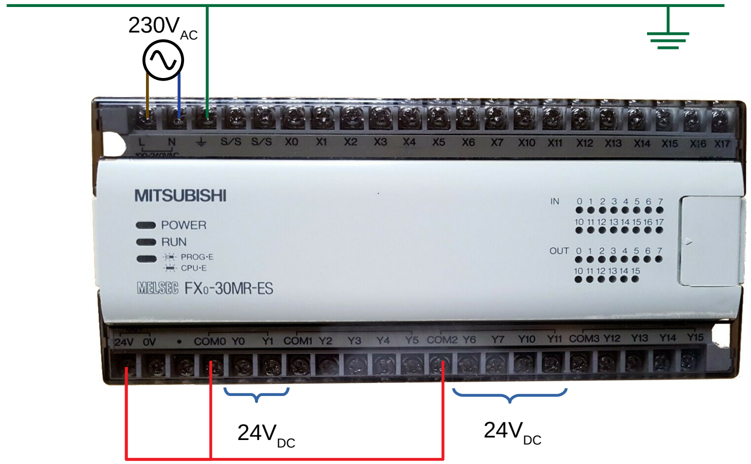
Ladder diagram example



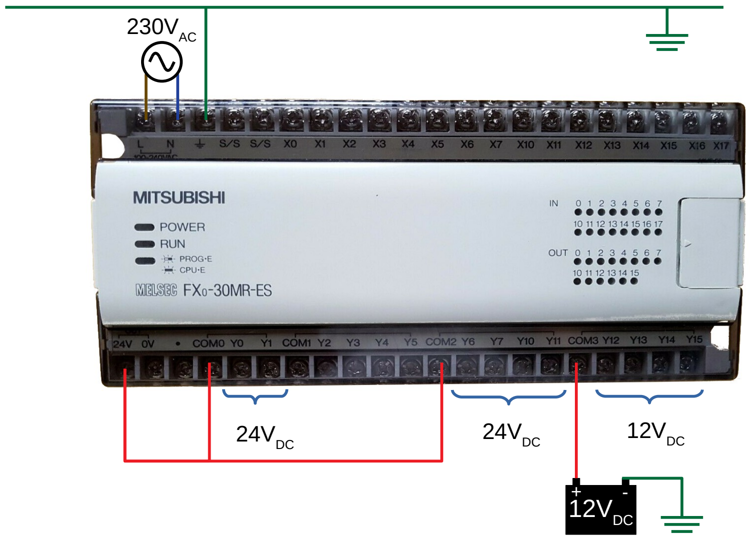
PLC Power



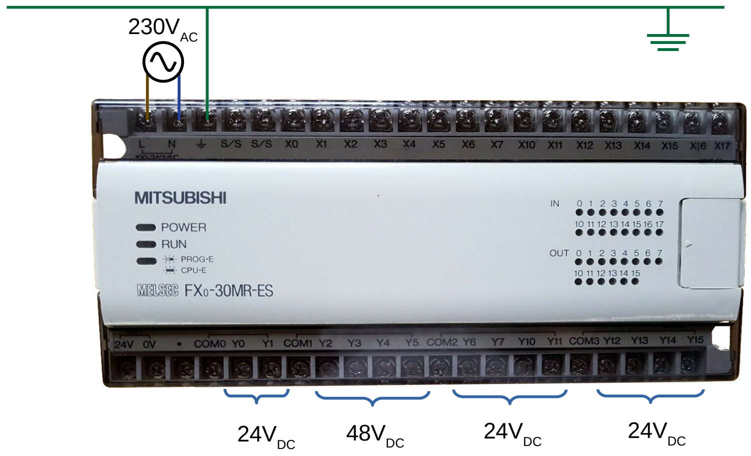
PLC Power



PLC Power

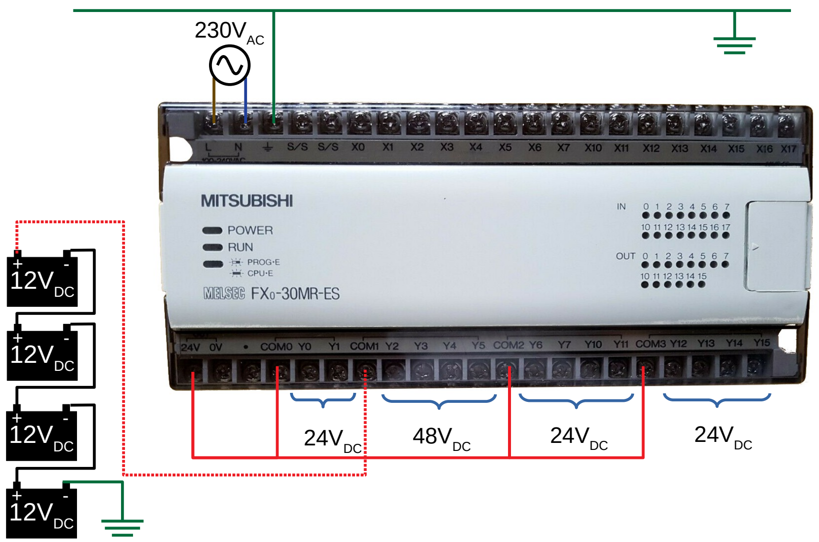


PLC Power

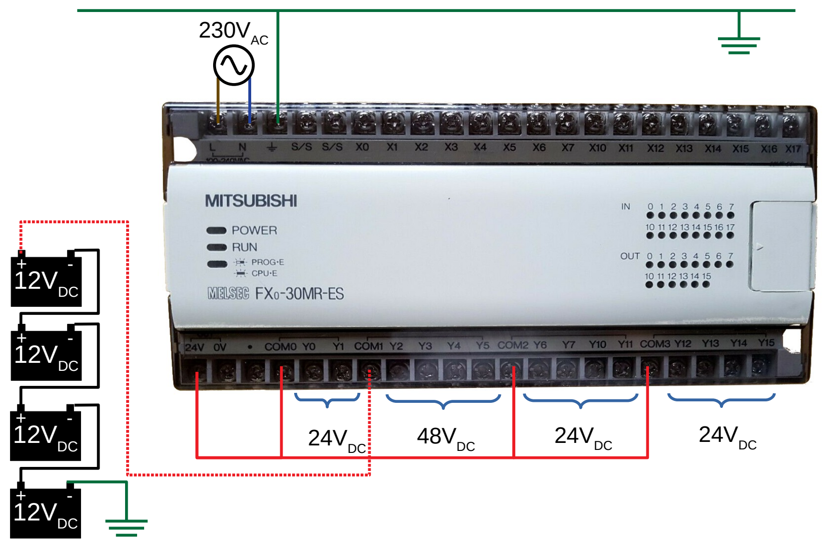


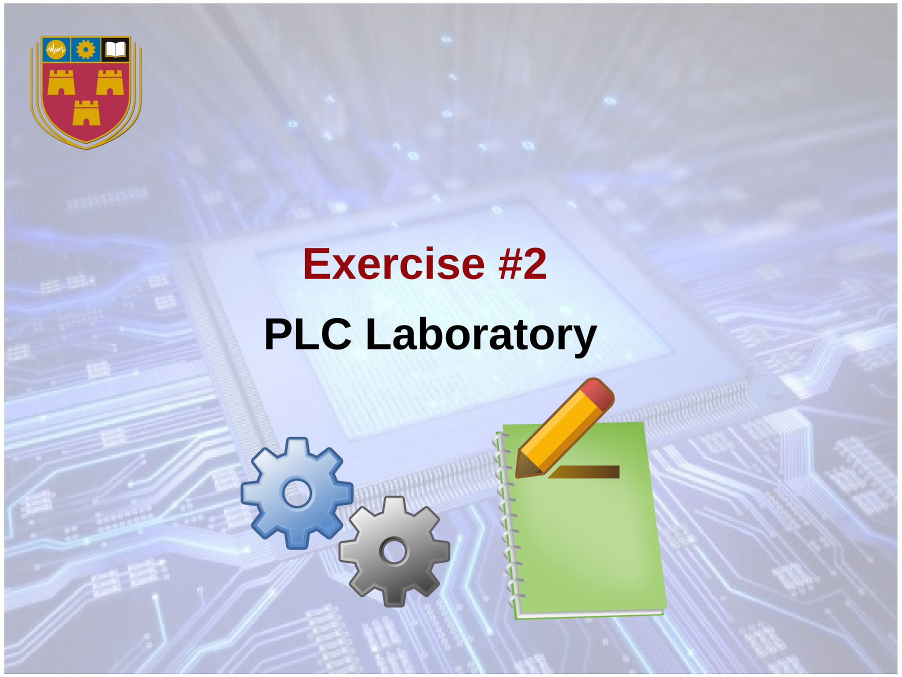
How would I get $48V_{DC}$ outputs here and $24V_{DC}$ everywhere else ?

PLC Power



PLC Power



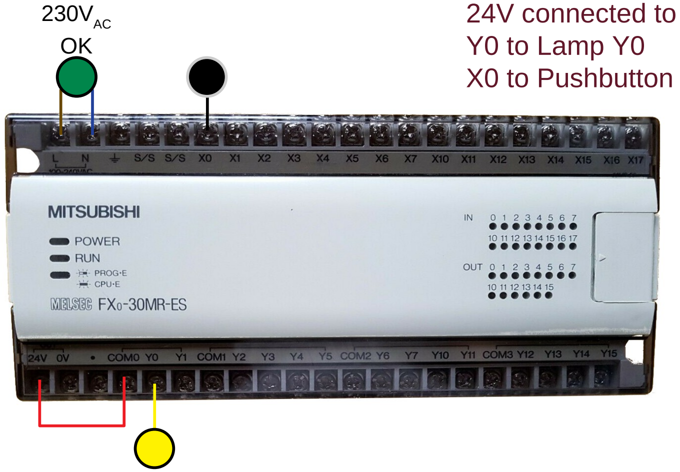


Exercise #2

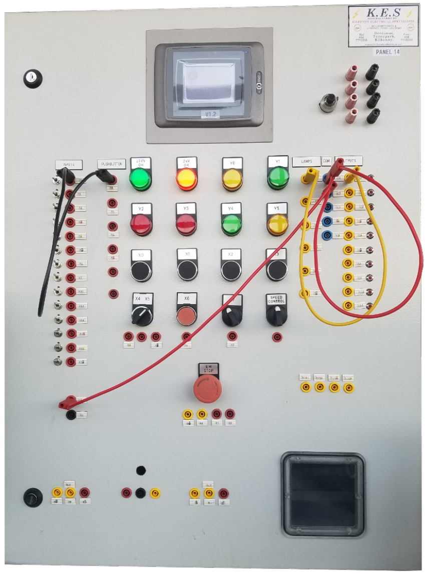
PLC Laboratory

PLC Laboratory

230V mains light OK
All switches in UP position
24V connected to COM /0 /1
Y0 to Lamp Y0
X0 to Pushbutton X0



PLC Laboratory

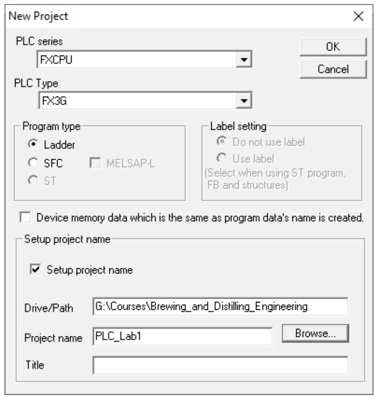


230V mains light OK
All switches in UP position
24V connected to COM /0 /1
Y0 to Lamp Y0
X0 to Pushbutton X0

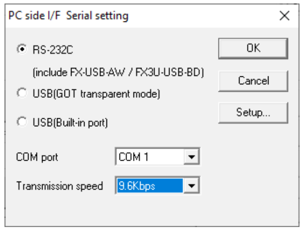
GX Developer

GX Developer

New Project



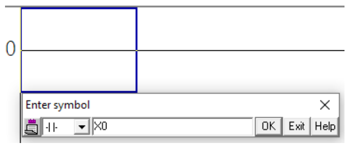
Online → Transfer Setup



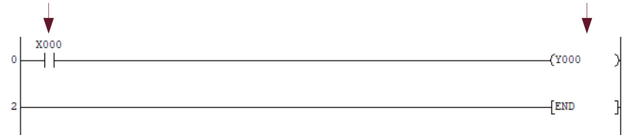
Baud rate: 9,600 b/s

Program #1

New Project



N.O. Switch (F5) Coil (lamp) (F7)



- Convert → Convert (Compile program)
- Online → Write to PLC

Program #1

- Press button X0, what happens ?

Program #1

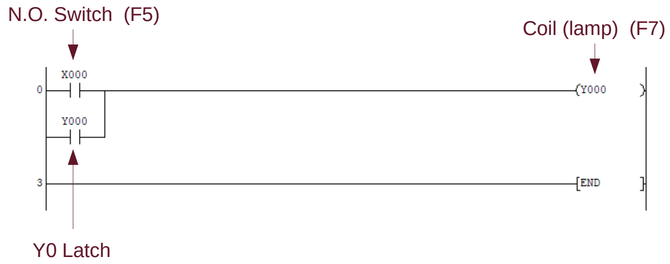
- Press button X0, what happens ?

- What happens when you release ?

Program #1

- Press button X0, what happens ?
- What happens when you release ?

Program #2



- Convert → Convert (Compile program)
- Online → Write to PLC

- What do you think the latch does ?

Program #2

- Press button X0, what happens ?

Program #2

- Press button X0, what happens ?

Y0

- What happens when you release ?

Program #2

- Press button X0, what happens ?
- What happens when you release ?

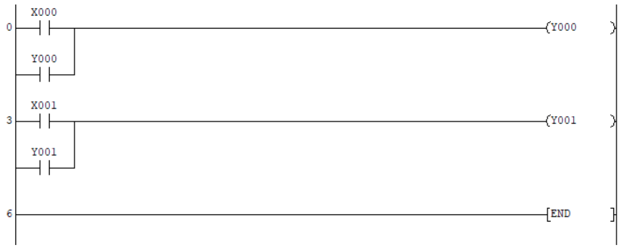
Y0

- Why ?

Adding an additional button

- Connect X1 to X1 pushbutton

Program #3



- Convert → Convert (Compile program)
- Online → Write to PLC

Program #3

- Press button X0, what happens ?

Program #3

- Press button X0, what happens ?

- What happens when you release ?

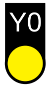
Program #3

- Press button X0, what happens ?
- What happens when you release ?

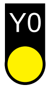
- Why ?
- Press button X0 again, what happens ?
- Why ?

Program #3

- Press button X0, what happens ?
- What happens when you release ?



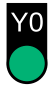
- Why ?
- Press button X0 again, what happens ?



- Why ?

Program #3

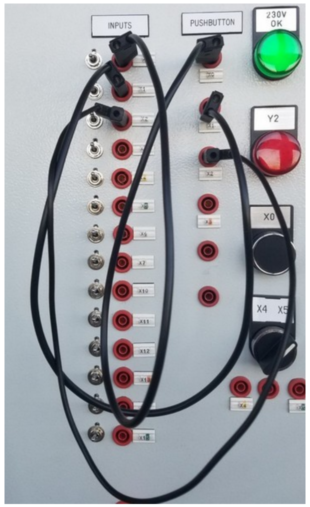
- Press button X1, what happens ?
- What happens when you release ?



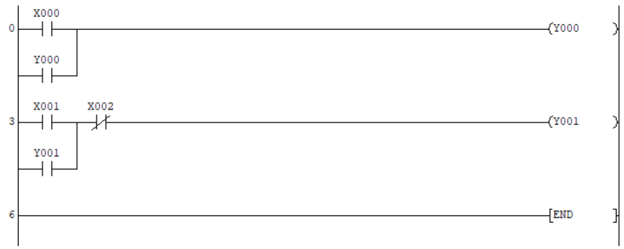
- Why ?
- Press button X1 again, what happens ?
- Why ?

Adding an additional button

- Connect X2 to button X2 with a black wire.



Program #4



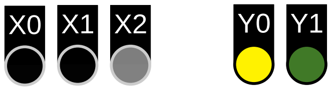
- Convert → Convert (Compile program)
- Online → Write to PLC

Program #4

- Press button X0, what happens ?

- Press button X1, what happens ?

- Press button X2, what happens ?



Program #5

- Convert → Convert (Compile program)
- Online → Write to PLC

Program #5

Press button X0

Press button X1

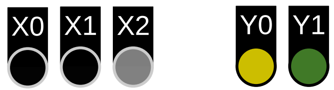
Why doesn’t Y1 light ?

Program #5

Press button X0

Press button X1

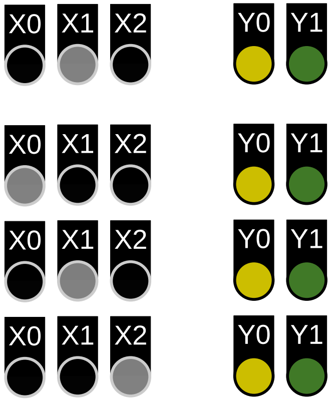
Press X2



Why did Y0 extinguish ?

Program #5

Try this sequence and explain

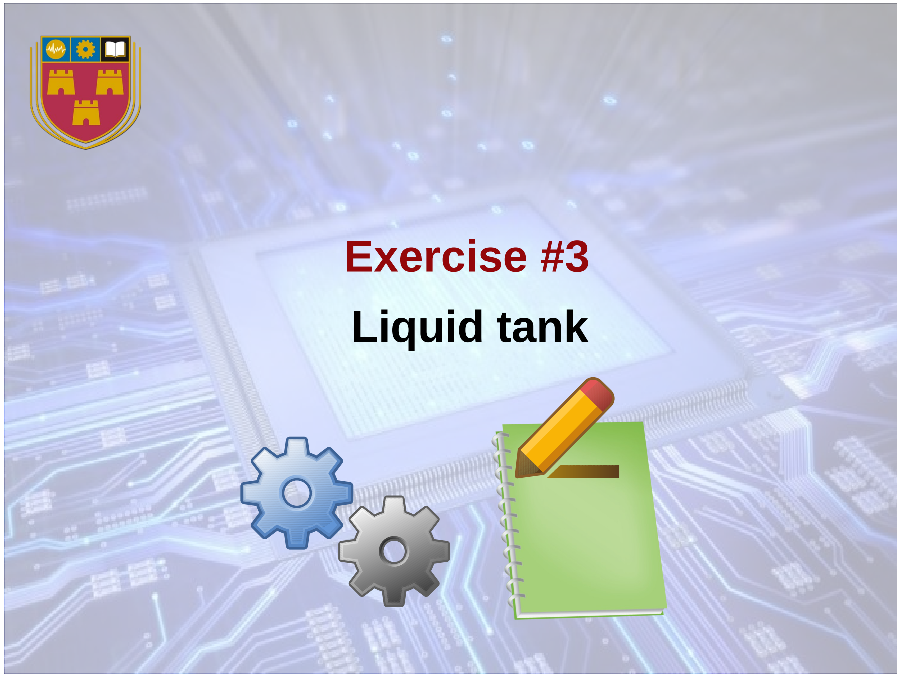


Instruction Set

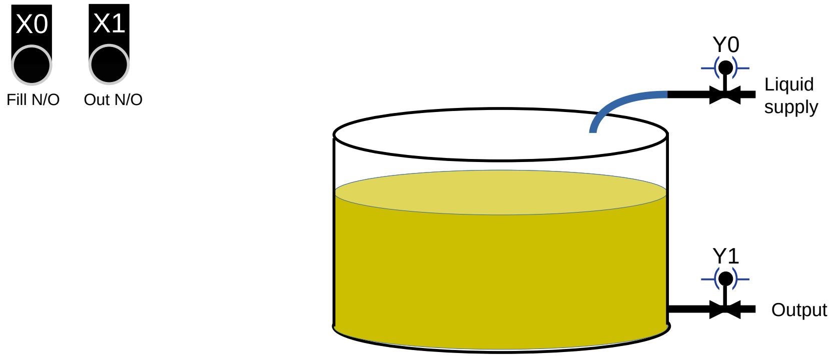
- View → Instruction List (Alt+F1)

```
0   LD    X000
1   OR    Y000
2   ANI   Y001
3   OUT   Y000
4   LD    X001
5   OR    Y001
6   ANI   Y000
7   OUT   Y001
8   END
```

Instruction	Abbreviation
Load, Normally Open (N/O) contact	LD
AND; N/C contact in series	ANI
OR; N/O contact in parallel	OR
Output	OUT



Exercise #3.1



- Build a PLC circuit to control the filling of the liquid tank.
- Draw a ladder diagram to control the supply and output of the liquid.

Exercise #3.1

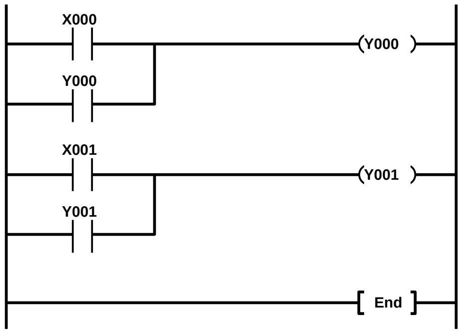
- Implement program on PLC.

Exercise #3.2

- Adjust the program so the button can be pressed once and the fill will continue or the output will continue.

Exercise #3.2

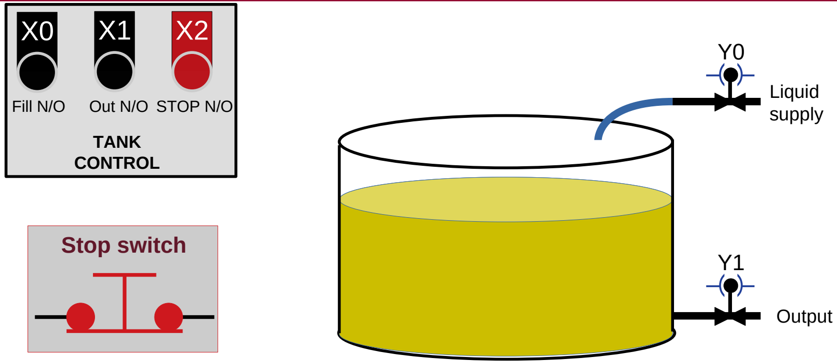
- Adjust the program so the button can be pressed once and the fill will continue or the output will continue.



Exercise #3.2

- Adjust the program so the button can be pressed once and the fill will continue or the output will continue
- What is the flaw with this ?

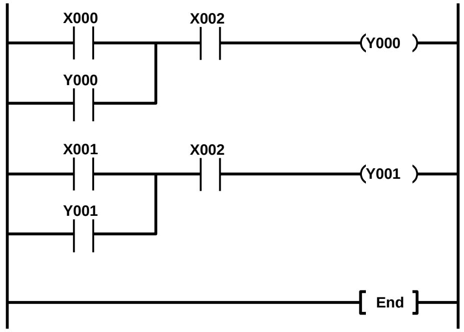
Exercise #3.3



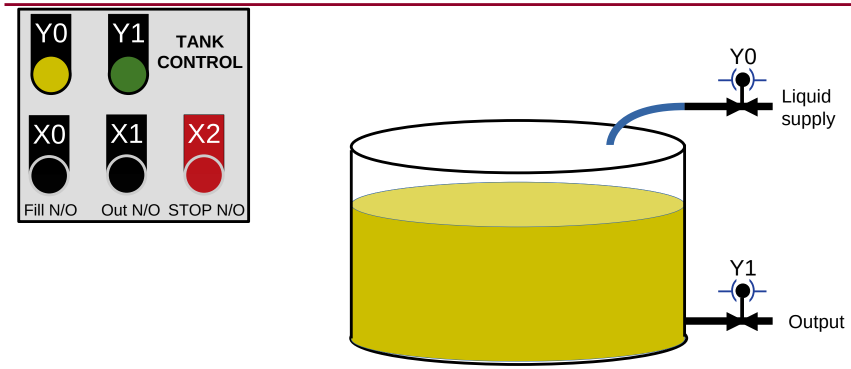
- Add a Stop button
  (Stop buttons are physically N/C push-buttons by default and therefore need to be Examine-ON so in the ladder diagram it appears NO).

Exercise #3.3

- Adjust the program so the button can be pressed once and the fill will continue or the output will continue.

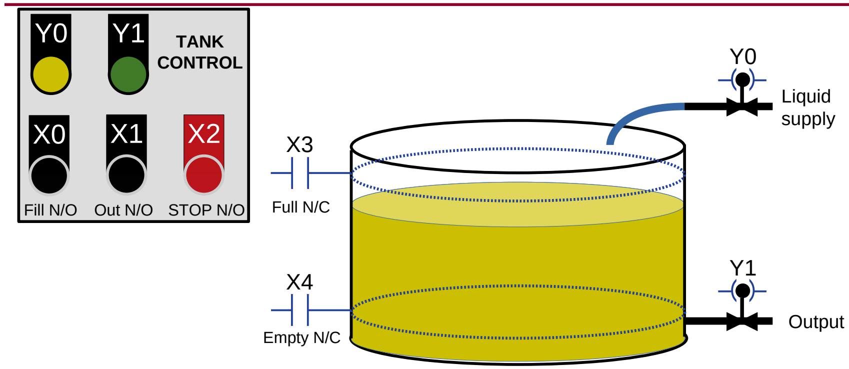


Exercise #3.4



- Add visual indicators for the fill and output states.

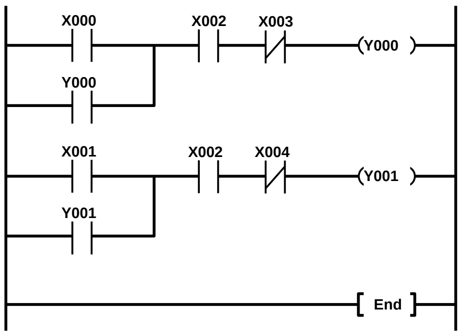
Exercise #3.5



- The liquid in the tank cannot empty below the output pipe
- The liquid in the tank cannot over fill either, two sensors are added
- Include these in the program to prevent overfill or the tank emptying.

Exercise #3.5

- Adjust the program so the button can be pressed once and the fill will continue or the output will continue.



Exercise #3.5

- Show the associated Instruction List.

0	LD	X000	(Fill switch)
1	OR	Y000	(Tap latch)
2	AND	X002	(Stop switch)
3	ANI	X003	(High level sensor)
4	OUT	Y000	(Tap actuator)
5	LD	X001	(Drain switch)
6	OR	Y001	(Drain latch)
7	AND	X002	(Stop switch)
8	ANI	X004	(Low level sensor NC)
9	OUT	Y001	(Drain actuator)
10	END		(End)

Learning outcomes

- Logic Circuits ✓
- PLC Operating Principles ✓
- PLC Control Features ✓
- Ladder Diagrams ✓