

MFS605/EE605

Systems for Factory Information and Control

Lecture 9 – PLCs (half lecture)

Fall 2005

Larry Holloway

Dept. of Electrical Engineering and
Center for Robotics and Manufacturing Systems

1

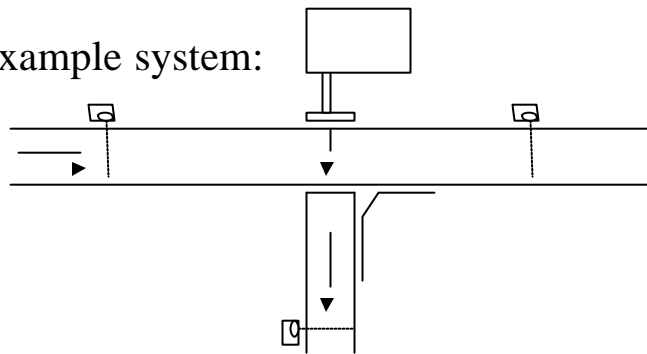
- So far in class:
 - Modeling of manufacturing systems
 - simple
 - queueing
 - simulation
 - Control of manufacturing systems
 - MRP
 - Kanban
 - Now...
 - more focus on low-level control and information

2

- PLC: Programmable Logic Controller
 - Microcomputer-like instrument for controlling industrial equipment
 - “Programmable”: control can be modified
 - “Logic” operate primarily with on-off voltages as inputs and outputs
 - fans, motors, pumps, valves, warning horns/lights, etc.

3

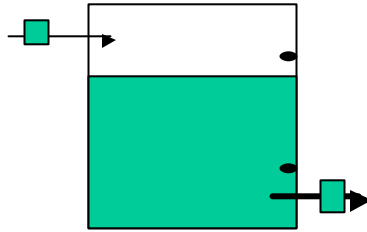
- Example system:



- Discrete actuation and sensing

4

Another Example: Fluid Control



5

Discrete Sensors

- Limit Switches
- Proximity Sensors:
 - Inductive
 - Capacitive
 - Magnetic/Hall effect
- Photoelectric Sensors

6

Photoelectric Sensors

Through beam

Reflection from target

RetroReflection

Infrared vs. visible, pulsed vs. constant, use of fiber

7

Other sensing applications

(Variations on basic sensors described)

- Liquid level
- Temperature
- Pressure

8

Actuation

- Discrete Actuation:
 - Motors off and on
 - Lamps
 - Buzzers
 - Heating Coils
 - Etc.

9

Logic implementation

- Electronic logic gates
- Programmable logic devices (PALs, FPGA's)
- Relays
- Hydraulic/Pneumatic logic
- Computer based

10

Interfacing logic

- Relays
 - Basic terminology: NC, NO, Coil, SPST, DPST

11

12

Interfacing logic

- Optoisolators

13

- PLC: Programmable Logic Controller
 - Microcomputer-like instrument for controlling industrial equipment
 - “Programmable”: control can be modified
 - “Logic” operate primarily with on-off voltages as inputs and outputs
 - fans, motors, pumps, valves, warning horns/lights, etc.

14

Why PLCs?

- rugged for reliable operation in industrial environments
- operating systems won't crash or hang-up
- designed for extensive I/O -- easily interfaced
 - examples: 220VAC input and output
 - current I/O
 - thermocouple I/O
 - etc.
- Designed for industrial programming:

15

Disadvantages of PLCs:

- Often limited arithmetic/computational capability
- Typically missing the ability for extensive data-logging, user-interface, and other "PC" computer tasks.
- Ladder logic programming languages primitive -- hard to debug or maintain large programs

16

PLC Background

- Sequential control used to be accomplished by hard-wired banks of relays:
 - Simple logic of relays:
 - AND
 - OR
 - NOTs

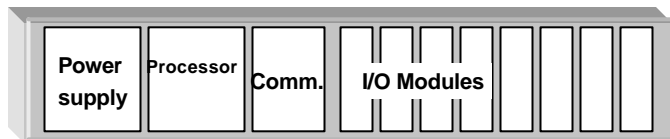
17

- Hardwired relays still used in some applications, but several problems:
 - Hard-wiring is time consuming, doesn't allow rapid change of logic.
 - Hardware is failure prone, noisy.
- In the 1960's, push (by GM and others) to replace Relays with a software equivalent
- More recently: communications, networking, more advanced functions, faster processors, better software languages

18

Structure of PLC

- Modular System: pieces easily interchanged



Example I/O Modules:

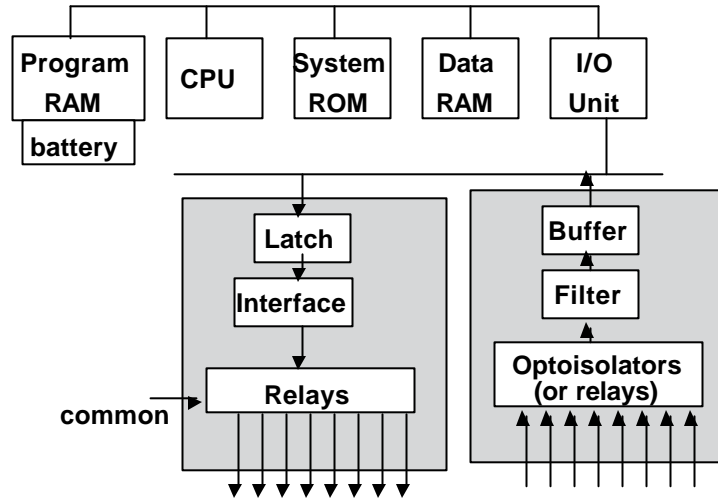
- 5v DC
- 24VDC
- 110VAC
- 220VAC
- 3-phase AC
- current
- PID controllers
- “Fuzzy” control modules
- Thermocouple interfaces
- etc.

19

- Where is the keyboard and monitor?
 - Unnecessary in normal use: only used when programming
 - either program on PC and download, or have auxiliary programming station
- Where is the disk?
 - Disks often vulnerable in harsh environments, so not typically used
 - Programs typically small and stored in memory (RAM, EPROMS, or Flash)

20

Typical PLC Architecture



21

PLC Programming -- Ladder Logic

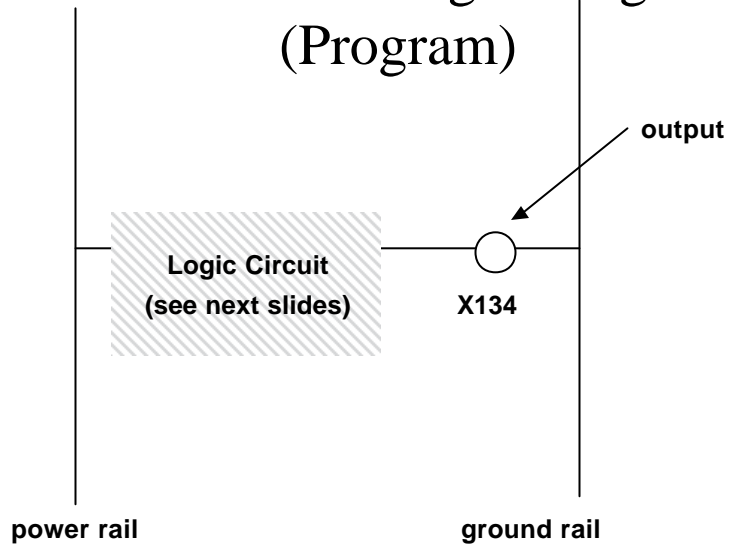
Ladder Logic is software version of original relay control circuitry.

- Advantages:
 - direct extension of the control methods previously used
 - simple to do simple logic
- Disadvantage:
 - seems primitive
 - difficult to do complex logic and sequencing

–

22

Basic Ladder Logic Diagram (Program)



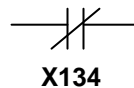
23

- Standard Symbols

- Normally Open relay:



- Normally Closed relay:



- Other Symbols (less common):

- Positive transition sensing
 - Negative transition sensing

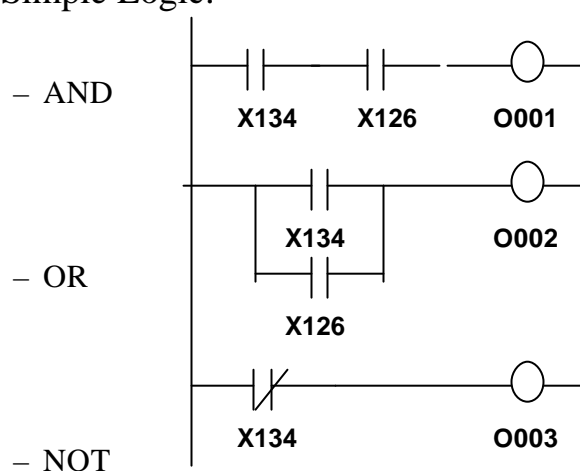
24

Coils

- Output coils vs. internal coils (memory coils)
- Standard types:
 - Coil
 - Negated Coil
 - Set Coil
 - Reset Coil
 - Retentive Memory Coil
 - Positive Transition Sensing Coil
 - Negative Transition Sensing Coil

25

- Simple Logic:



26

- Assume power high rung and power low rung.
- Outputs of system (and intermediate logic variables) are on when “current” flows to them through logic.
- A ladder program is then a sequence of these ladder rungs
- EXAMPLE: (safety system): Machine is on if pushbutton 1 is on and safety sensor is low, or if pushbutton 2 is on, but never when the emergency stop is pushed.

27

- EXAMPLE: (safety system): Machine is on if pushbutton 1 is on and safety sensor is low, or if pushbutton 2 is on, but never when the emergency stop is pushed.

28

- Typical operation:
 - scan inputs (to *image register*)
 - execute program
 - output status (from *image register*)
 - housekeeping, diagnostics, communications
 - Repeat

- The time to repeat this is called the “scan time”
 - Depends on
 - # of program rungs
 - Speed of processor
 - (# of inputs, # of outputs)

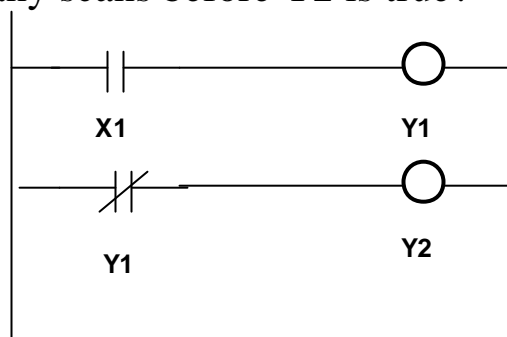
 - 1-25msec is common

- Scan time issues...(immediate I/O updates)

29

Scan issues

- How many scans before Y2 is true?



30

Memory Logic: Feedback within rungs

EXAMPLE:

X7 is start switch

X8 is stop switch

X9 is over-temperature switch

Have motor Y3 on if start switch pressed in the past, as long as stop switch and over-temperature switch have not been true.

31

Have motor Y3 on if start switch pressed in the past, as long as stop switch and over-temperature switch have not been true

32

PLC's lecture 3

33

- Modern PLCs have special function blocks including timers, counters, etc.
- **EXAMPLE (continued):**Start switch X7 must be held down for at least 2 seconds before Y3 is started.

34

Timer Example

35

- PLC Programming (Lecture 18)

36

PLC Review

- Standard Symbols
 - Normally Open relay:
 - Normally Closed relay:
- Simple Logic:
 - AND
 - OR
 - NOT

37

PLC Programming

- Simple Combinatorial Logic
- State logic -- latching
- Sequential logic

38

Simple Combinatorial Logic

- Key steps:
 - Identify inputs and outputs
 - For each output, determine expression for what makes it true
 - Construct ladder logic to reflect expression

39

Combinatorial Example

If “Oven On” switch is activated ($S=1$)
and oven door closed ($DLS=1$)
and Temperature below threshold ($TLS = 0$)
then heat is on ($H=1$)

If oven is on ($H=1$)
or (temp > thresh ($TLS = 1$)
and door is closed ($DLS=1$))
then fan is on ($F=1$)

If oven is on ($H=1$)
or light switch is on ($LS=1$)
then light is on ($L=1$)

40

State Logic -- Latching

- Key steps:
 - Identify inputs and outputs. Consider outputs to be state variables
 - For each state variable X ,
 - determine expression $Reset_X$ of what makes state go false
 - determine expression Set_X of what makes state go true
 - Construct ladder logic to reflect the following expression:

$$\overline{(Reset_X)} (Set_X + X) \text{ ® } X$$

(If you have difficulty converting $Reset_X$ into $Reset_X$, then you may use a different rung to calculate $Reset_X$, then invert it in the rung for X)

41

Example

Tank System:

- S1 is turned on whenever level is below LLS switch (LLS=0)
- S1 is turned off when level above ULS switch (ULS =1)

42

Example

- Motor Control
 - Turn on motor when On-switch is pressed
 - Turn off motor when Off-switch is pressed, *or* when temperature sensor is high.

43

Sequential Logic -- Cascade method

- Sequential Logic has steps.
 - The functions that drive the outputs differ depending upon which step is being done
- Use extra state variables to represent which step is active
 - Draw state diagram representing steps, indicating what causes changes in the states
 - Establish “Flip-Flop” rung for each step

 - Define Output rungs that depend on the steps
 - Initialization rung may be required

44

Sequential Logic -- Example

- An industrial oven has four states, Off, Preheat, Superheat, and CoolDown. The oven starts in the Off state. If *start* is pushed,
- then it enters the Preheat state and turns the heater on. It remains in the preheat state until the temperature sensor *T1* becomes true, at which time it then enters the Superheat state. In this state, it continues to heat, but locks the oven door. Once temperature sensor *T2* becomes true, then it moves to state CoolDown, where the heater turns off. The door remains locked, until the sensor *T1* becomes false, at which point the OFF state is entered and the door is unlocked.
- There is a stop switch also, but it only works during the preheat state, and returns the system to off.

45

Industrial Oven -- state diagram

46

Industrial Oven -- Ladder Program

47

- Auto-ID

48