Ladder Logic / Diagrams

CPE200, Fall 2023

Hank Dietz

http://aggregate.org/hankd/
Why are we doing this?

• Because career fair folks asked for it… ;-)  

• Ladder logic is *still* commonly used
  – PLCs (Programmable Logic Controllers)
  – Makes programming look like circuits;
    originally switches and relays
  – Each “rung” is a rule relating inputs & output;
    sort-of like an if-then statement
Ladder Basics

- [ ] - or - ] [ - Normally open contact
- [\] - or - ] \ [ - Normally closed contact
- [blocktype] - Special block
- [ ] -- [ ] - Series means AND
- ++ [ ] +-+ Parallel means OR
- -- [ ] --
- ( ) - Normally inactive coil (output)
- (\) - Normally active coil (output)
LED is on
LED is on while Switch is pressed

[Diagram of a switch and LED connected in series]
LED is on while Switch0 is pressed AND Switch1 isn’t
LED is on while Switch0 is pressed OR Switch1 isn’t
LED0 is as LED before, but LED1 is on while LED0 is off
An example: **Alarm** sounds when either sensor is triggered
But we also want **Alarm** if a wire to a sensor breaks...
Implementations?

- There is a standard: IEC 61131-3, however, Allen Bradley, Siemens, etc., PLCs differ a bit.
- **LDmicro** is an old free editor+compiler
  [https://cq.cx/ladder.pl](https://cq.cx/ladder.pl)
- **OpenPLC** is open-source PLC software compliant with IEC 61131-3
  [https://autonomylogic.com/docs/openplc-overview/](https://autonomylogic.com/docs/openplc-overview/)
LDmicro export text
for 'ANSI C Code', 4.000000 MHz crystal, 10.0 ms cycle time

LADDER DIAGRAM:

```
XSensor0     XSensor1     RAOK
1                     ( )
1                     ( )
2                     ( )
[END]
```

I/O ASSIGNMENT:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>XSensor0</td>
<td>digital in</td>
<td>(not assigned)</td>
</tr>
<tr>
<td>XSensor1</td>
<td>digital in</td>
<td>(not assigned)</td>
</tr>
<tr>
<td>YAlarm</td>
<td>digital out</td>
<td>(not assigned)</td>
</tr>
<tr>
<td>RAOK</td>
<td>int. relay</td>
<td></td>
</tr>
</tbody>
</table>
LDmicro structured text
for Alarm

LDmicro0.1
MICRO=ANSI C Code
CYCLE=10000
CRYSTAL=4000000
BAUD=2400
compiled=Z:\Big\Courses\CPE200\LADDER\alarm.c

IO LIST
  XSensor0 at 0
  XSensor1 at 0
  YAlarm at 0
END

PROGRAM
RUNG
  CONTACTS XSensor0 1
  CONTACTS XSensor1 1
  COIL RAOK 0 0 0
END
RUNG
  CONTACTS RAOK 1
  COIL YAlarm 0 0 0
END
/* U_xxx symbols correspond to user-defined names. There is such a symbol for every internal relay, variable, timer, and so on in the ladder program. I_xxx symbols are internally generated. */

STATIC BOOL I_b_mcr = 0;
#define Read_I_b_mcr() I_b_mcr
#define Write_I_b_mcr(x) I_b_mcr = x

STATIC BOOL I_b_rung_top = 0;
#define Read_I_b_rung_top() I_b_rung_top
#define Write_I_b_rung_top(x) I_b_rung_top = x

/* You provide this function. */
PROTO(extern BOOL Read_U_b_XSensor0(void));

/* You provide this function. */
PROTO(extern BOOL Read_U_b_XSensor1(void));

STATIC BOOL U_b_RAOK = 0;
#define Read_U_b_RAOK() U_b_RAOK
#define Write_U_b_RAOK(x) U_b_RAOK = x

/* You provide these functions. */
PROTO(BOOL Read_U_b_YAlarm(void));
PROTO(void Write_U_b_YAlarm(BOOL v));
/* Call this function once per PLC cycle. You are responsible for calling it at the interval that you specified in the MCU configuration when you generated this code. */

void PlcCycle(void)
{
    Write_I_b_mcr(1);

    /* start rung 1 */
    Write_I_b_rung_top(Read_I_b_mcr());

    /* start series [ */
    if(Read_U_b_XSensor0()) {
        Write_I_b_rung_top(0);
    }
    if(Read_U_b_XSensor1()) {
        Write_I_b_rung_top(0);
    }
    Write_U_b_RAOK(Read_I_b_rung_top());
    /* ] finish series */

    /* start rung 2 */
    Write_I_b_rung_top(Read_I_b_mcr());

    /* start series [ */
    if(Read_U_b_RAOK()) {
        Write_I_b_rung_top(0);
    }
    Write_U_b_YAlarm(Read_I_b_rung_top());
    /* ] finish series */
}
Special Blocks for LDmicro...

- **[OSR]** – One shot rising; **OSF** for falling
- **[TON]** – Turn On with delay; **TOF** for off
- **[CTU]** – Count Up; **CTD** Down; **CTC** Circular
- **[EQU]** – == comparison; also **NEQ** != and **GRT** >; **GEQ** >=; **LES** <; **LEQ** <=
- **[MOV]** – Move; also **ADD**; **SUB**; **MUL**; **DIV**

Others include ADC read, PWM output, shift register, look-up table, and many more...
OpenPLC Editor: Blink a LED

This example cascades two timers (TON and TOF) to generate a square wave. The width of the wave is determined by the size of the PT variable on both timers.
OpenPLC Editor: Traffic Light

Program to control a traffic light with an Arduino Uno as a stand-alone device. This program is aimed at beginners who want to learn PLC code on their own; it can be improved upon.

Date: 12/05/2022  Programmer: Michael Flores (updated)

Activation of the red lamp:

Activation of the orange lamp:

Activation of the green lamp:

Turn off first cycle:
IEC 61131-3 Standard: PLC Programming Languages

- We’ll stick to LD for CPE200, but...
- IEC 61131-3 lists 5 programming languages:
  - IL: Instruction List (text)
  - ST: Structured Text (text)
  - LD: Ladder Diagram (graphical)
  - FBD: Function Block Diagram (graphical)
  - SFC: Sequential Function Chart (mixed)
**IL Reference Card (Siemens)**

### Siemens S7 Statement List (STL)

**Bit Logic**

<table>
<thead>
<tr>
<th><strong>A</strong></th>
<th>AND</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AN</strong></td>
<td>AND Not</td>
</tr>
<tr>
<td><strong>O</strong></td>
<td>OR</td>
</tr>
<tr>
<td><strong>ON</strong></td>
<td>OR Not</td>
</tr>
<tr>
<td><strong>X</strong></td>
<td>Exclusive Or</td>
</tr>
<tr>
<td><strong>XX</strong></td>
<td>Exclusive Or Not</td>
</tr>
<tr>
<td><strong>FB</strong></td>
<td>Edge Negative</td>
</tr>
<tr>
<td><strong>FP</strong></td>
<td>Edge Positive</td>
</tr>
</tbody>
</table>

**Assign**

- **=**
- **Reset**
- **Set**
- **S**
- **NOT**
- **OR**
- **AND**
- **SET**
- **CLR**
- **Save RLO in BR**
- **Register**

### Convert

- **BCD** to Integer
- **Integer** to BCD
- **Integer** to Double Integer
- **Double Integer** to BCD
- **BCD** to Integer
- **Double Integer** to BCD
- **Floating Point**

### Accumulator

- **ACC**
- **ADD**
- **NEG**
- **SUB**
- **MUL**
- **DIV**
- **MOD**

### Floating Point Math

- **ACOS**
- **ASIN**
- **ATAN**
- **COS**
- **EXP**
- **LOG**
- **SIN**
- **SQRT**
- **TAN**

### Word Logic

- **AND**
- **AND Word**
- **OR**
- **OR Word**
- **XOR**
- **XOR Exclusive Or**
- **XOR Exclusive Or Word**

### Timers/Counters (0 to 255)

- **Enable Timer/Counter**
- **Load Current**
- **Timer/Counter Value into ACC1 as Integer (i.e., L T [2])**
- **Load Current**
- **Timer/Counter Value into ACC1 as BCD (i.e., LC T [2])**
- **Reset Timer/Counter**
- **Set Counter Present Value (i.e., C 0.15)**
- **On-Delay Timer**
- **Retentive On-Delay Timer**
- **SP Pulse Timer**
- **SP Off Delay Timer**
- **SE Extended Pulse Timer**
- **Counter Down**
- **Counter Up**

### DBs

1. **Main Program Scan**
2. **Time of Day**
3. **Time Delay**
4. **Cyclic (Periodic)**
5. **40-47 Hardware**
6. **Power Supply Error**
7. **Diagnostic Interrupt**
8. **Interf/Remove Module Interrupt**
9. **CPU Hardware Fault**
10. **Program Cycle Error**
11. **Real Error**
12. **Missing Profibus**
13. **Communication Error**
14. **Wart Restart**
15. **Hot Restart**
16. **Cold Restart**
17. **Programmable Error**
18. **I/O Access Error**

**www.plcdev.com**

v2.0

[Image of the Reference Card]

---

### IL IL Reference Card (Siemens)

**Bit Logic**

- **A** = AND
- **AN** = AND Not
- **O** = OR
- **ON** = OR Not
- **X** = Exclusive Or
- **XX** = Exclusive Or Not
- **FB** = Edge Negative
- **FP** = Edge Positive

**Assign**

- **=**
- **Reset**
- **Set**
- **S**
- **NOT**
- **OR**
- **AND**
- **OR**
- **AND**
- **SET**
- **CLR**
- **Save RLO in BR**
- **Register**

### Convert

- **BCD to Integer**
- **Integer to BCD**
- **Integer to Double Integer**
- **Double Integer to BCD**
- **BCD to Integer**
- **Double Integer to BCD**
- **Floating Point**

### Accumulator

- **ACC**
- **ADD**
- **NEG**
- **SUB**
- **MUL**
- **DIV**
- **MOD**

### Floating Point Math

- **ACOS**
- **ASIN**
- **ATAN**
- **COS**
- **EXP**
- **LOG**
- **SIN**
- **SQRT**
- **TAN**

### Word Logic

- **AND**
- **AND Word**
- **OR**
- **OR Word**
- **XOR**
- **XOR Exclusive Or**
- **XOR Exclusive Or Word**

### Timers/Counters (0 to 255)

- **Enable Timer/Counter**
- **Load Current**
- **Timer/Counter Value into ACC1 as Integer (i.e., L T [2])**
- **Load Current**
- **Timer/Counter Value into ACC1 as BCD (i.e., LC T [2])**
- **Reset Timer/Counter**
- **Set Counter Present Value (i.e., C 0.15)**
- **On-Delay Timer**
- **Retentive On-Delay Timer**
- **SP Pulse Timer**
- **SP Off Delay Timer**
- **SE Extended Pulse Timer**
- **Counter Down**
- **Counter Up**

### DBs

1. **Main Program Scan**
2. **Time of Day**
3. **Time Delay**
4. **Cyclic (Periodic)**
5. **40-47 Hardware**
6. **Power Supply Error**
7. **Diagnostic Interrupt**
8. **Interf/Remove Module Interrupt**
9. **CPU Hardware Fault**
10. **Program Cycle Error**
11. **Real Error**
12. **Missing Profibus**
13. **Communication Error**
14. **Wart Restart**
15. **Hot Restart**
16. **Cold Restart**
17. **Programmable Error**
18. **I/O Access Error**

**www.plcdev.com**

v2.0

[Image of the Reference Card]
ST Example from SolisPLC

```c
//Mathematical / Built-In Operators
LocDINT[0] := LocDINT[1] + 5;

//Examples
TempC := (TempF - 32) * 5/9;
AScaled := AnalogInput[0] / 160 * 3;
```
LD Example from SolisPLC

StartProgram

Program_Enabled

Timer On Delay
Timer StartRoutine
Preset 1000+
Accum 0+

StartRoutine.TT

Program_Starting

StartRoutine.DN

Program_Running

Timer On Delay
Timer RunRoutine
Preset 500+
Accum 0+

RunRoutine.DN

Motor_1_Start  Motor_2_Start
FBD Example from SolisPLC
SFC Example from SolisPLC