

# ĐIỀU KHIỂN LOGIC VÀ PLC

# Nội dung

1. Cơ sở cho Điều khiển logic
2. Tổng hợp và tối thiểu hóa mạch logic tổ hợp
3. Tổng hợp mạch logic tuần tự
4. Tổng quan về PLC
- 5. Kỹ thuật lập trình PLC**

# 5. Kỹ thuật lập trình PLC

5.1. Chu trình thiết kế chương trình PLC

5.2. Các ngôn ngữ lập trình theo chuẩn IEC61131-3

5.3. Thiết kế chương trình sử dụng hàm logic

5.4. Thiết kế chương trình sử dụng SFC

# 5. Kỹ thuật lập trình PLC

## **5.1. Chu trình thiết kế chương trình PLC**

5.2. Các ngôn ngữ lập trình theo chuẩn IEC61131-3

5.3. Thiết kế chương trình sử dụng hàm logic

5.4. Thiết kế chương trình sử dụng SFC

## 5.1. Chu trình thiết kế chương trình PLC

**Bước 1 Phân tích**

**Bước 2 Thiết kế**

**Bước 3 Lập trình**

**Bước 4 Kiểm tra**

**Bước 5 Viết tài liệu**

**Bước 6 Vận hành**

# 5.1. Chu trình thiết kế chương trình PLC

## **Bước 1: Phân tích**

- Tìm hiểu công nghệ
  - ✓ Thảo luận trực tiếp với khách hàng
  - ✓ Sơ đồ công nghệ P&ID, hồ sơ nâng cấp cải tạo (nếu có)
- Trả lời các câu hỏi
  - ✓ Cần điều khiển những gì?
  - ✓ Các hành động điều khiển được thực hiện như thế nào
  - ✓ Người vận hành tác động được những gì?
  - ✓ Xử lý thế nào khi có sự cố?

# 5.1. Chu trình thiết kế chương trình PLC

## **Bước 1: Phân tích (tiếp)**

- Kết quả
  - ✓ Mô tả được hoạt động của quá trình
  - ✓ Tác động từ trạm vận hành và cách thức tác động
  - ✓ Danh sách các tín hiệu vào/ra
  - ✓ Chế độ vận hành khi có lỗi

# 5.1. Chu trình thiết kế chương trình PLC

## Bước 2: Thiết kế

- Lựa chọn phần cứng
  - ✓ Số lượng đầu vào/ra logic hoặc tương tự
  - ✓ Số lượng đầu vào/ra đặc biệt: high speed counter, PWM
  - ✓ Truyền thông: RS232, Modbus, Ethernet...

STT	Tên tín hiệu	Đầu vào		Đầu ra		Truyền thông hoặc tín hiệu đặc biệt		Ghi chú
		Logic	Tương tự	Logic	Tương tự	Loại truyền thông	Loại tín hiệu	
1								
...								



## 5.1. Chu trình thiết kế chương trình PLC

- **Bước 2: Thiết kế (tiếp)**
  - Lựa chọn phần cứng
    - ✓ Tốc độ xử lý, dung lượng bộ nhớ

STT	Loại câu lệnh	Thời gian max ( $\mu$ s)	Thời gian min ( $\mu$ s)	Bộ nhớ chương trình (Words)	Bộ nhớ dữ liệu (Words)	Số lượng câu lệnh (number)	Tổng bộ nhớ chương trình (Words)
1	Đọc đầu vào						

# 5.1. Chu trình thiết kế chương trình PLC

## Bước 2: Thiết kế (tiếp)

- Phân địa chỉ tín hiệu
  - ✓ Theo chức năng: đầu vào/ra tương tự, tốc độ cao...
  - ✓ Các biến có cùng một đối tượng hoặc các biến trong cùng trình tự tác động nên có địa chỉ gần nhau

STT	Tên tín hiệu	Đầu vào		Đầu ra		Địa chỉ	Ghi chú
		Logic	Tương tự	Logic	Tương tự		
1							
2							
...							

## 5.1. Chu trình thiết kế chương trình PLC

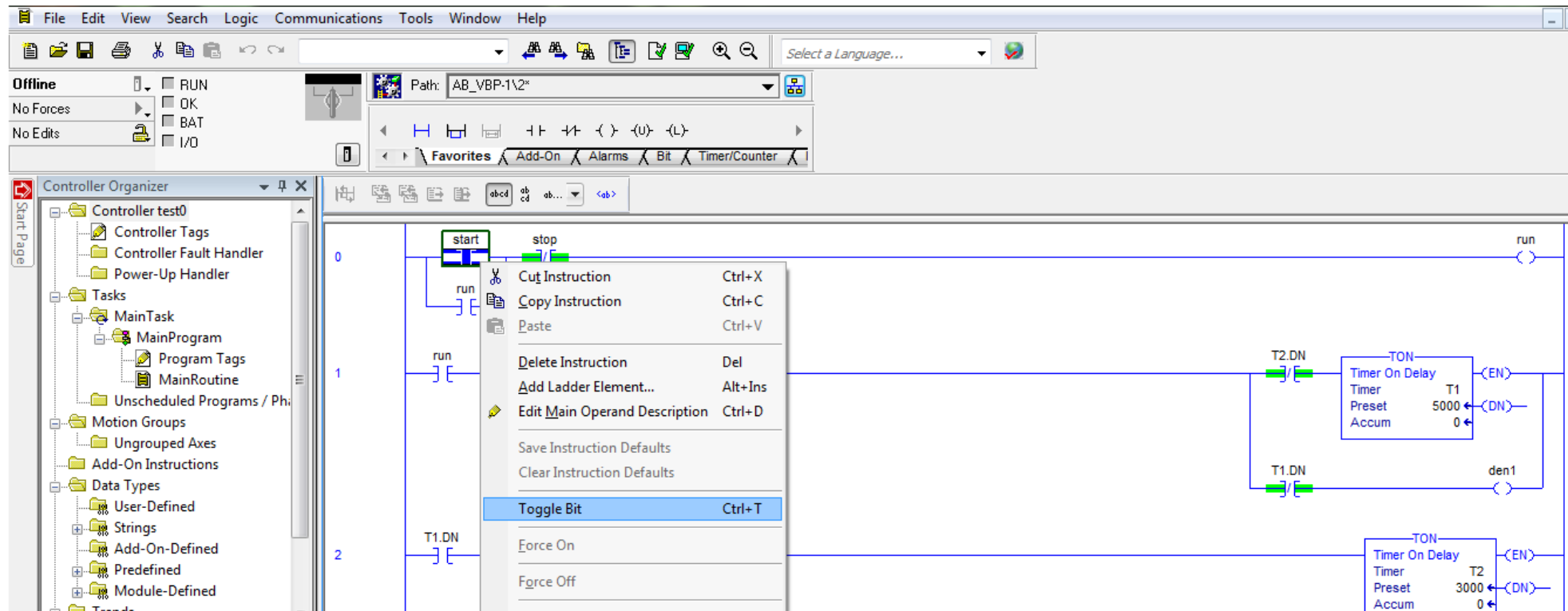
### **Bước 3: Lập trình**

- Tổng hợp hàm logic --> chương trình LD, IL, FBD
- Grafcet --> chương trình SFC
- Lưu đồ thuật toán --> chương trình ST

# 5.1. Chu trình thiết kế chương trình PLC

## Bước 4: Kiểm tra

- Mô phỏng bằng phần mềm, giả lập các tín hiệu vào



# 5.1. Chu trình thiết kế chương trình PLC

## Bước 5: Viết tài liệu

- Hướng dẫn vận hành
- Phục vụ bảo trì, bảo dưỡng
- Viết song song và được cập nhật trong quá trình phát triển dự án

### 2.3.2. PLC Communication

The PLC communicates with

- HMI by Profinet
- Each VFD of conveyor by Profinet
- UHT PLC by Profinet
- Each Stopper, Sensor by Hardwire (Inputs/Outputs on the ASI Network)

For the Communication table between PLC and machines, refer to [802ST3 Conveyor 012 A](#)

### 2.3.3. PLC Inputs/Outputs list

For the inputs and outputs list, refer to the file [802ST3 Conveyor 011 A](#)

### 2.3.4. Defaults list

For the defaults list, refer to the file [802ST3 Conveyor 010 A](#)

## 3. DETAILED PROCESS SYSTEM DESCRIPTION

All the start-up, running and stop requirements for each function are listed in the different function descriptions.

This chapter summarizes:

- The aim of each function or each operation,
- The document number of the function description.

### 3.1. Transfer empty boxes from Box Dispenser Room to Corazza 1

## 5.1. Chu trình thiết kế chương trình PLC

### **Bước 6: Vận hành**

- Chạy thử bộ từng phần riêng lẻ, không tải
- Chạy thử từng bộ phần riêng lẻ, có tải
- Chạy thử toàn hệ thống không tải
- Chạy thử toàn hệ thống có tải

# 5. Kỹ thuật lập trình PLC

5.1. Chu trình thiết kế chương trình PLC

## **5.2. Các ngôn ngữ lập trình**

5.3. Thiết kế chương trình sử dụng hàm logic

5.4. Thiết kế chương trình sử dụng SFC

## 5.2. Các ngôn ngữ lập trình

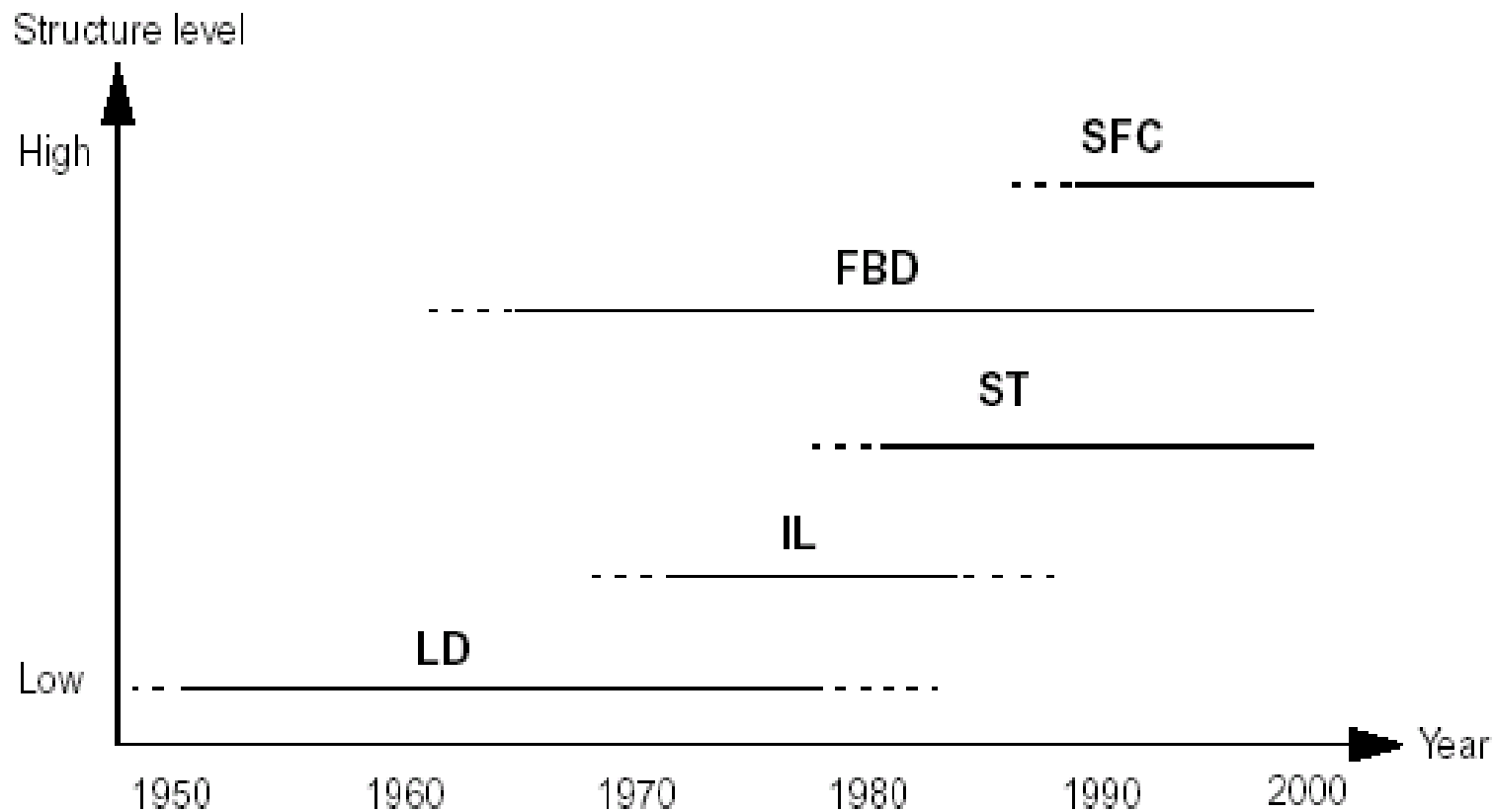
Ladder Diagrams (LD)

Instruction List (IL)

Function Block Diagram (FBD)

Structured Text (ST)

Sequential Function Chart (SFC)





## 5.2. Các ngôn ngữ lập trình

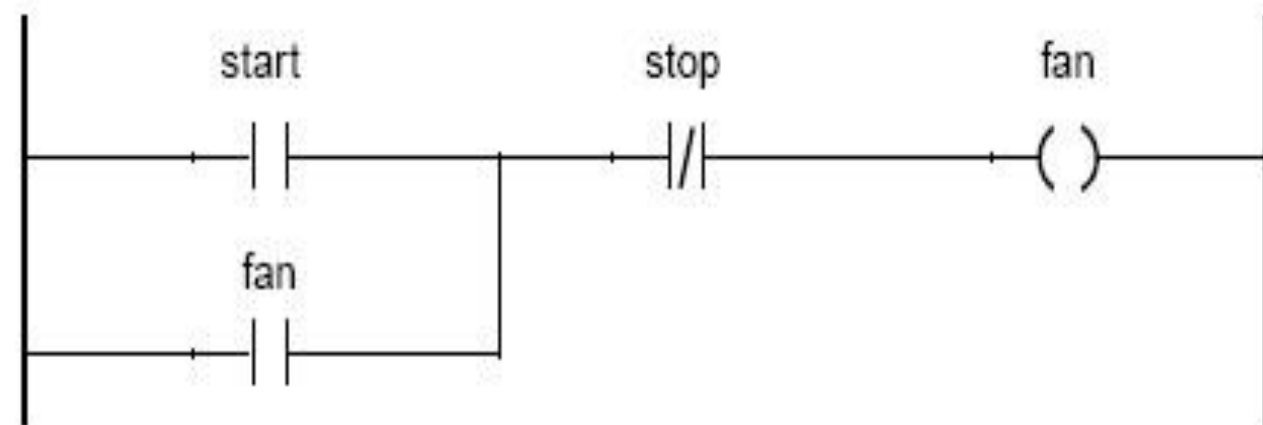
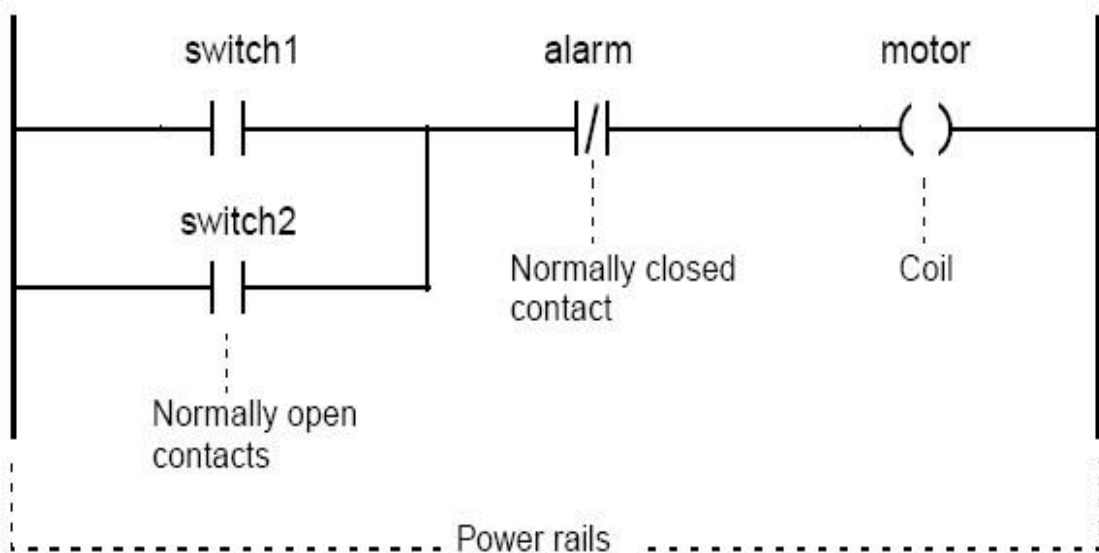
### Ngôn ngữ LD

- Hai thanh nguồn (Power rails).
- Tiếp điểm (Contacts) đại diện cho biến logic
  - ✓ Thường mở (Normally Open)
  - ✓ Thường đóng (Normally Closed)
  - ✓ Tiếp điểm nối tiếp --> logic AND
  - ✓ Nhánh song song --> logic OR
- Cuộn dây (Coils) đại diện cho đầu ra
- Tạo mạch phản hồi: tên tiếp điểm trùng tên cuộn dây

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ LD



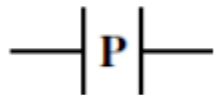
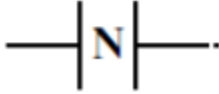
*Motor := (switch1 OR switch 2) AND (NOT alarm)*



*fan := (start OR fan) AND (NOT stop)*

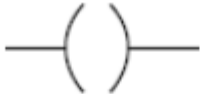

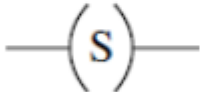
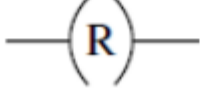
## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ LD

STT	Tên	Ký hiệu	Ý nghĩa
1	Tiếp điểm thường hở		Khi biến có giá trị bằng 1 thì tiếp điểm này sẽ đóng lại.
2	Tiếp điểm thường kín		Khi biến có giá trị bằng 0 thì tiếp điểm này sẽ đóng lại
3	Tiếp điểm sườn lên		Phát hiện trạng thái của biến thay đổi từ 0 lên 1 và sẽ cho giá trị 1 tại thời điểm đó. Còn các thời điểm khác là 0.
4	Tiếp điểm sườn xuống		Phát hiện trạng thái của biến thay đổi từ 1 xuống 0 và sẽ cho giá trị 1 tại thời điểm đó. Còn các thời điểm khác là 0.

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ LD

5	Cuộn dây		Trạng thái của biến tương ứng với trạng thái bên trái cuộn dây.
6	Cuộn dây đảo		Trạng thái của biến tương ứng với nghịch đảo của trạng thái bên trái cuộn dây.
7	Cuộn dây SET		Trạng thái của biến tương ứng được set lên ON khi trạng thái ON ở phía trước và duy trì trạng thái này cho đến khi được reset bằng cuộn RESET.
8	Cuộn dây RESET		Trạng thái của biến tương ứng được reset xuống OFF khi có trạng thái ON phía trước.

## 5.2. Các ngôn ngữ lập trình

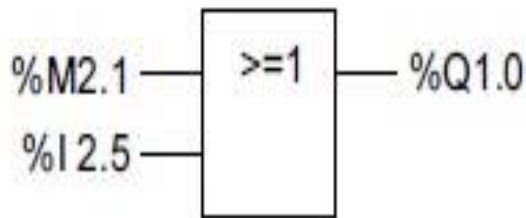
### **Ngôn ngữ LD**

- Ưu điểm:
  - ✓ Dễ lập trình, dễ hiểu: giống sơ đồ mạch điện.
  - ✓ Dễ bảo dưỡng: có khả năng chẩn đoán lỗi online, từ đó định vị lỗi logic hoặc lỗi thiết bị.
- Nhược điểm:
  - ✓ Khó module hóa.
  - ✓ Hạn chế với kiểu dữ liệu có cấu trúc.

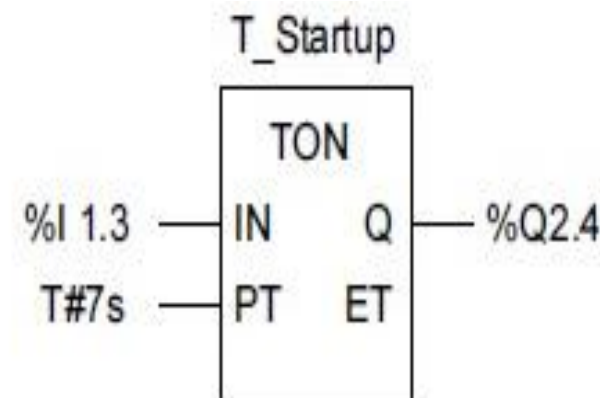
## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ FBD

- Gồm nhiều Function Block (FB).
- Tín hiệu chạy từ đầu ra của FB này đến đầu vào của FB khác.
- Đầu ra FB được cập nhận kết quả từ tính toán của FB dựa trên các tham số vào.
- Dòng tín hiệu chạy từ trái qua phải.

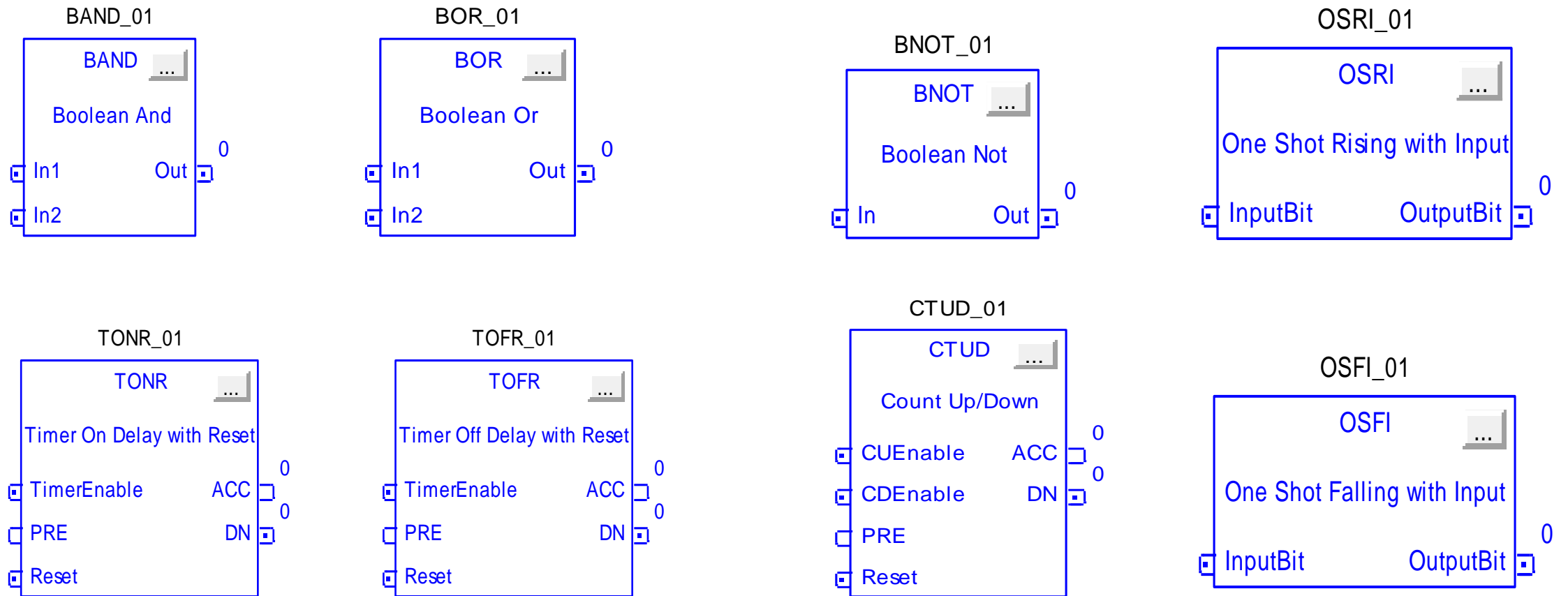


TDH-VD-BK



## 5.2. Các ngôn ngữ lập trình

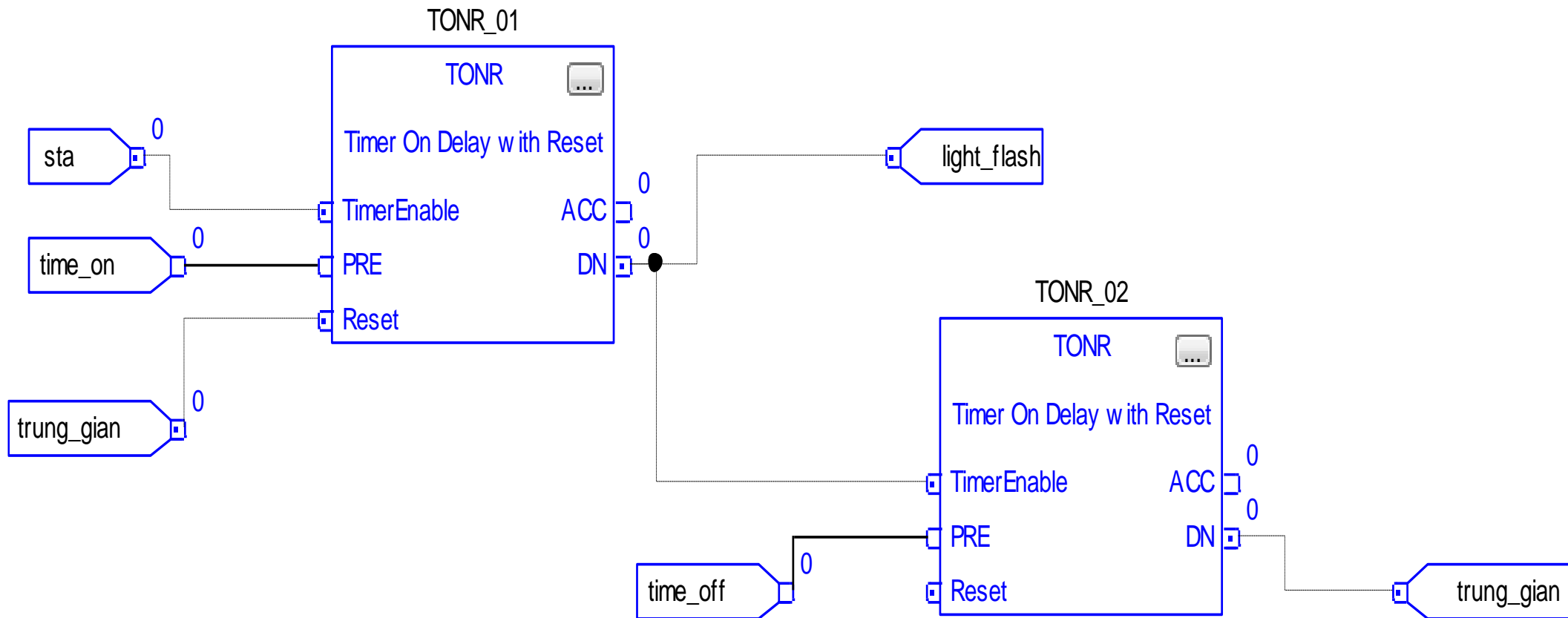
### Ngôn ngữ FBD



*Một số khối chức năng trong FBD cho PLC CompactLogix*

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ FBD





## 5.2. Các ngôn ngữ lập trình

### **Ngôn ngữ FBD**

- Ưu điểm:
  - ✓ Tương tự các mạch IC trong điện tử.
  - ✓ Ứng dụng kế hợp chức năng điều khiển logic và điều khiển phản hồi.
- Nhược điểm:
  - ✓ Hỗ trợ kém khi có một hay nhiều hành động lặp lại trong một khoảng thời gian định trước.
  - ✓ Khá cồng kềnh.

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ IL

- Ngôn ngữ lập trình bậc thấp, gồm các chuỗi câu lệnh, mỗi lệnh một dòng.
- Mỗi câu lệnh gồm một toán tử và một hay nhiều toán hạng.

Label	Operator	Operand	Comment
	LD	temp1	(*Load temp1 and*)
	GT	temp2	(*Test if temp1 > temp2*)
	JMPCN	Greater	(*Jump if not true to Greater*)
	LD	speed1	(*Load speed1*)
	ADD	200	(*Add constant 200*)
	JMP	End	(*Jump unconditional to End*)
Greater:	LD	speed2	(*Load speed2*)

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ IL

- Ưu điểm:
  - ✓ Thích hợp với ứng dụng nhỏ.
  - ✓ Tối ưu hóa bộ nhớ và tốc độ thực thi.
  - ✓ Có thể module hóa và tái sử dụng.
- Nhược điểm:
  - ✓ Ngôn ngữ bậc thấp, khó theo dõi.
  - ✓ Thanh ghi chỉ có một giá trị tại một thời điểm, khó làm việc với các dữ liệu có cấu trúc.

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ IL

Toán tử	Ý nghĩa
LD	Đặt giá trị hiện tại cho toán hạng, nghịch đảo là LDN
ST	Đưa giá trị hiện tại tới địa chỉ toán hạng
S	Đặt toán hạng loại logic lên 1
R	Đặt lại logic 0 cho toán hạng
AND	Logic AND, nghịch đảo là ANDN
OR	Logic OR, nghịch đảo là ORN
XOR	Hoặc loại trừ
NOT	Logic nghịch đảo
ADD	Cộng
SUB	Trừ
MUL	Nhân

TĐH-VĐ-BT

Toán tử	Ý nghĩa
DIV	Chia
MOD	Phép chia lấy dư
GT	So sánh lớn hơn
GE	So sánh lớn hơn hoặc bằng
EQ	So sánh bằng
NE	So sánh khác nhau
LE	So sánh nhỏ hơn hoặc bằng
LT	So sánh nhỏ hơn
JMP	Nhảy tới nhãn
CAL	Gọi khối chức năng
RET	Trở về từ gọi hàm, khối chức năng hay chương trình

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ ST

- Ngôn ngữ lập trình bậc cao (tương tự Pascal, C).
- 5 loại câu lệnh chính
  - ✓ Lệnh gán: *biến := giá trị*
  - ✓ Lệnh lựa chọn: IF... THEN
  - ✓ Lệnh vòng lặp: WHILE, REPEAT, FOR
  - ✓ Function và function block
  - ✓ Lệnh điều khiển: RETURN, EXIT

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ ST

- Lệnh gán
  - Gán giá trị cho biến hoặc biểu thức
  - Cấu trúc:

$X := Y;$                       (*\* X và Y có cùng kiểu dữ liệu \**)

- Ví dụ:

$Rate := 13.1;$                       (*\*Gán giá trị hằng số \**)

$Count := Count + 1;$  (*\*Gán giá trị bằng một biểu thức\**)

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ ST

- Lệnh điều kiện IF... THEN:

Dạng 1: *IF <Biểu thức điều kiện> THEN*  
*<Các câu lệnh>*  
*END\_IF;*

Dạng 2: *IF <Biểu thức điều kiện> THEN*  
*<Các câu lệnh>*  
*ELSE*  
*<Các câu lệnh>*  
*END\_IF;*

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ ST

- Lệnh điều kiện liệt kê CASE ... OF

*Case speed of:*

*Stop: rate: = 0.0; (\*Nếu speed bằng stop, gán rate bằng 0\*)*

*Slow: rate: = 20.4; (\*Nếu speed bằng slow, gán rate bằng 20.4\*)*

*Else*

*rate: = 0; (\*Các trường hợp còn lại rate bằng 0\*)*

*End\_case;*



## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ ST

- Lệnh vòng lặp FOR
  - Số vòng lặp là hữu hạn
  - Cấu trúc:  
*FOR* <giá trị bắt đầu>  
*TO* <giá trị kết thúc>  
*BY* <bước nhảy> *DO*  
    <Các câu lệnh ...>  
*END\_FOR*;

Ví dụ:

```
count := 0;  
FOR i:=1 TO 10 DO  
count := count + i;  
END_FOR;
```

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ ST

- Lệnh vòng lặp WHILE
  - Số vòng lặp không xác định trước nhưng điều kiện kết thúc xác định
  - Cấu trúc:  
*WHILE* <điều kiện> *DO*  
    <Các câu lệnh>  
*END\_WHILE*;

Ví dụ:

```
WHILE switch1 OR switch3  
DO  
    pump := FALSE;  
    alarm := TRUE;  
END_WHILE;
```

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ ST

- Lệnh vòng lặp REPEAT
  - Kiểm tra điều kiện sau khi thực hiện lệnh
  - Cấu trúc:  
*REPEAT*  
    *<statements...>*  
*UNTIL <boolean expression>*  
*END\_REPEAT;*

Ví dụ:

```
B := 0  
REPEAT  
    B := B + 1;  
UNTIL B > 10  
END_REPEAT;
```

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ ST

- Ưu điểm:
  - ✓ Phù hợp với tính toán phức tạp và vòng lặp
  - ✓ Dùng nhiều trong điều khiển tương tự
- Nhược điểm:
  - ✓ Đòi hỏi kiến thức nhất định về lập trình.

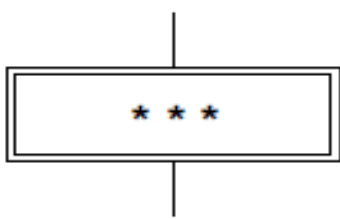
## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ SFC

- Phát triển từ Grafcet
- Có thể mạnh trong mô tả hệ thống điều khiển tuần tự.
- Sử dụng kết hợp với 4 ngôn ngữ còn lại

## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ SFC



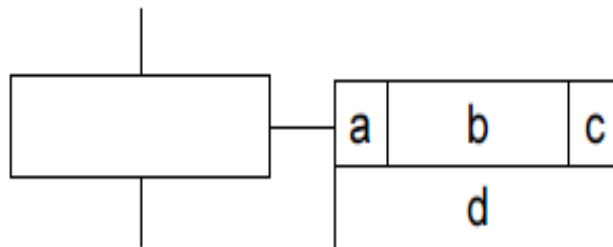
Trạng thái ban đầu



Trạng thái

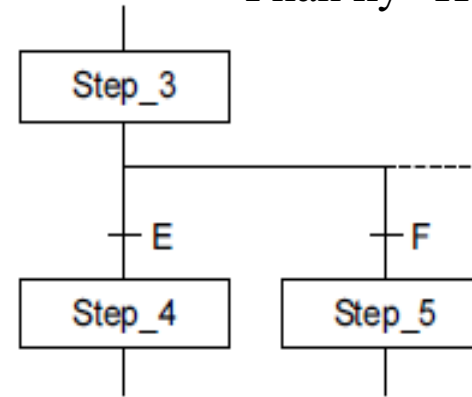


Chuyển tiếp

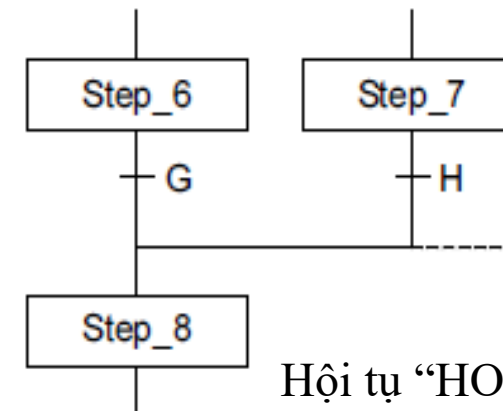
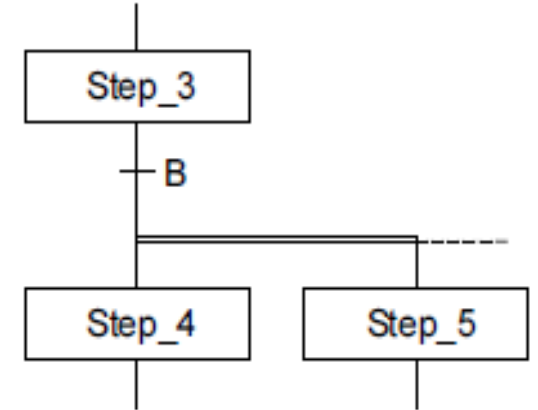


Hành động

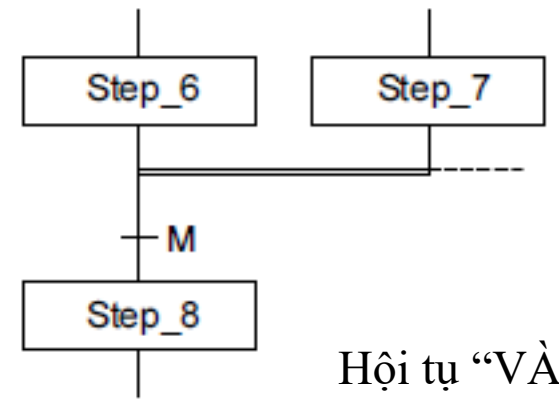
Phân kỳ “HOẶC”



Phân kỳ “VÀ”



Hội tụ “HOẶC”



Hội tụ “VÀ”

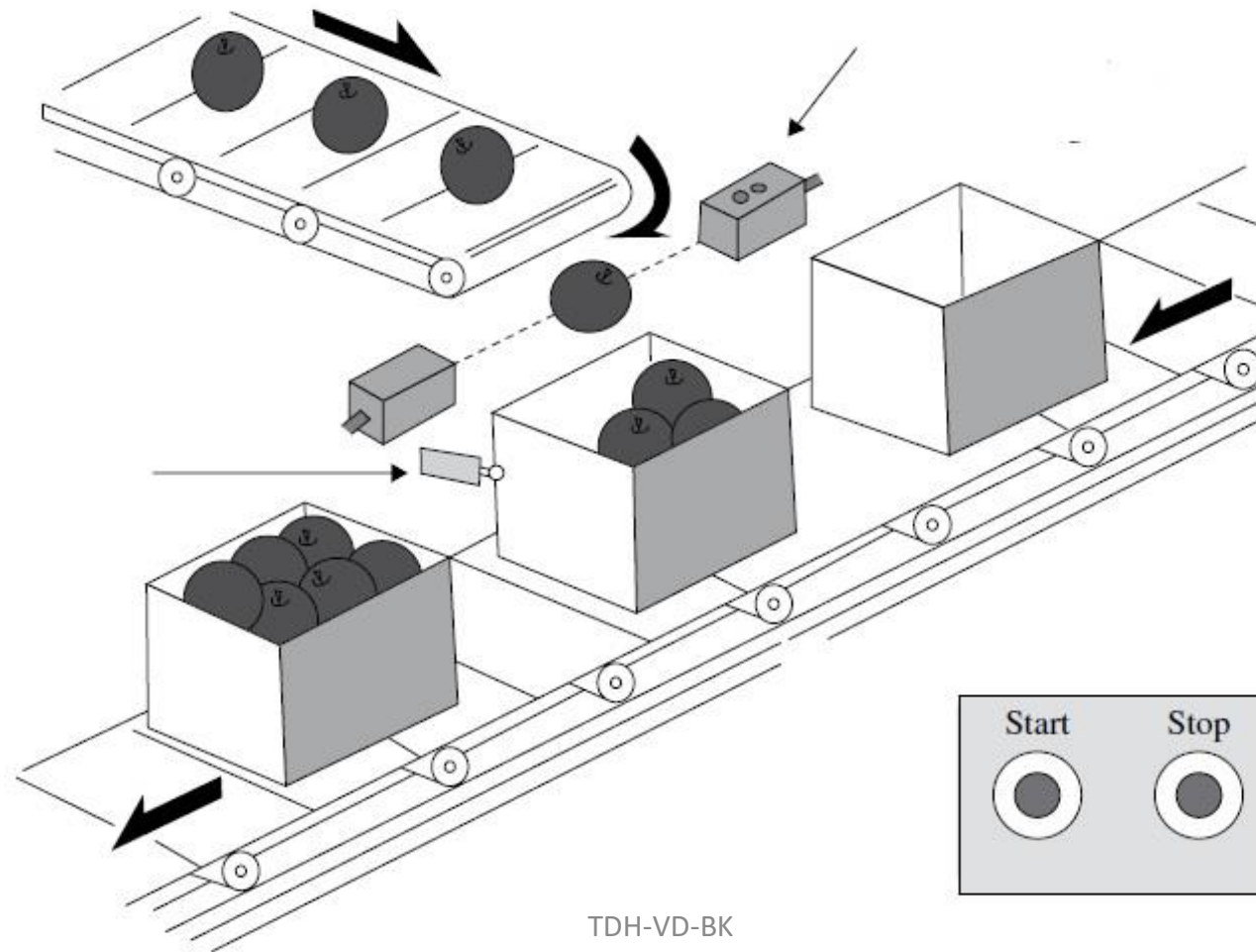
## 5.2. Các ngôn ngữ lập trình

### Ngôn ngữ SFC

- Ưu điểm:
  - ✓ Phù hợp với hệ thống tuần tự
- Nhược điểm:
  - ✓ Không phải ngôn ngữ hoàn chỉnh.
  - ✓ Thời gian thực hiện các phép điều kiện đơn giản lâu hơn các ngôn ngữ khác.

## 5.2. Các ngôn ngữ lập trình

### Ví dụ: Điều khiển dây chuyền đóng hộp sản phẩm





## 5.2. Các ngôn ngữ lập trình

### Mô tả công nghệ

- Khi nút Start được ấn, băng tải hộp chạy. Băng tải hộp sẽ dừng lại khi có tín hiệu từ cảm biến báo về rằng đã có hộp ở vị trí đóng gói. Lúc này, băng tải tảo bắt đầu hoạt động.
- Băng tải tảo làm nhiệm vụ cung cấp tảo đồ vào hộp. Một cảm biến được sử dụng để đếm từng quả tảo được đổ vào hộp.
- Khi số tảo đồ vào hộp bằng 10, băng tải tảo dừng, băng tải hộp lại chạy.
- Quá trình tiếp tục lặp lại cho đến khi nút Stop được ấn thì quy trình dừng ngay lập tức.

## 5.2. Các ngôn ngữ lập trình

### Các bước lập trình:

- Liệt kê tín hiệu vào/ra: 4 đầu vào số, 3 đầu ra số

STT	Tên tín hiệu	Loại tín hiệu	Đầu vào/Ra	
			Vào	Ra
1	Nút ấn khởi động - Start	Số	1	
2	Nút ấn dừng - Stop	Số	1	
3	Cảm biến hộp - CB_hop	Số	1	
4	Cảm biến tảo – CB_tao	Số	1	
5	Báo trạng thái làm việc (Lam_viec)	Số		1
6	Điều khiển bang tải hộp (Bang_hop)	Số		1
7	Điều khiển băng tải tảo (Bang_tao)	Số		1

## 5.2. Các ngôn ngữ lập trình

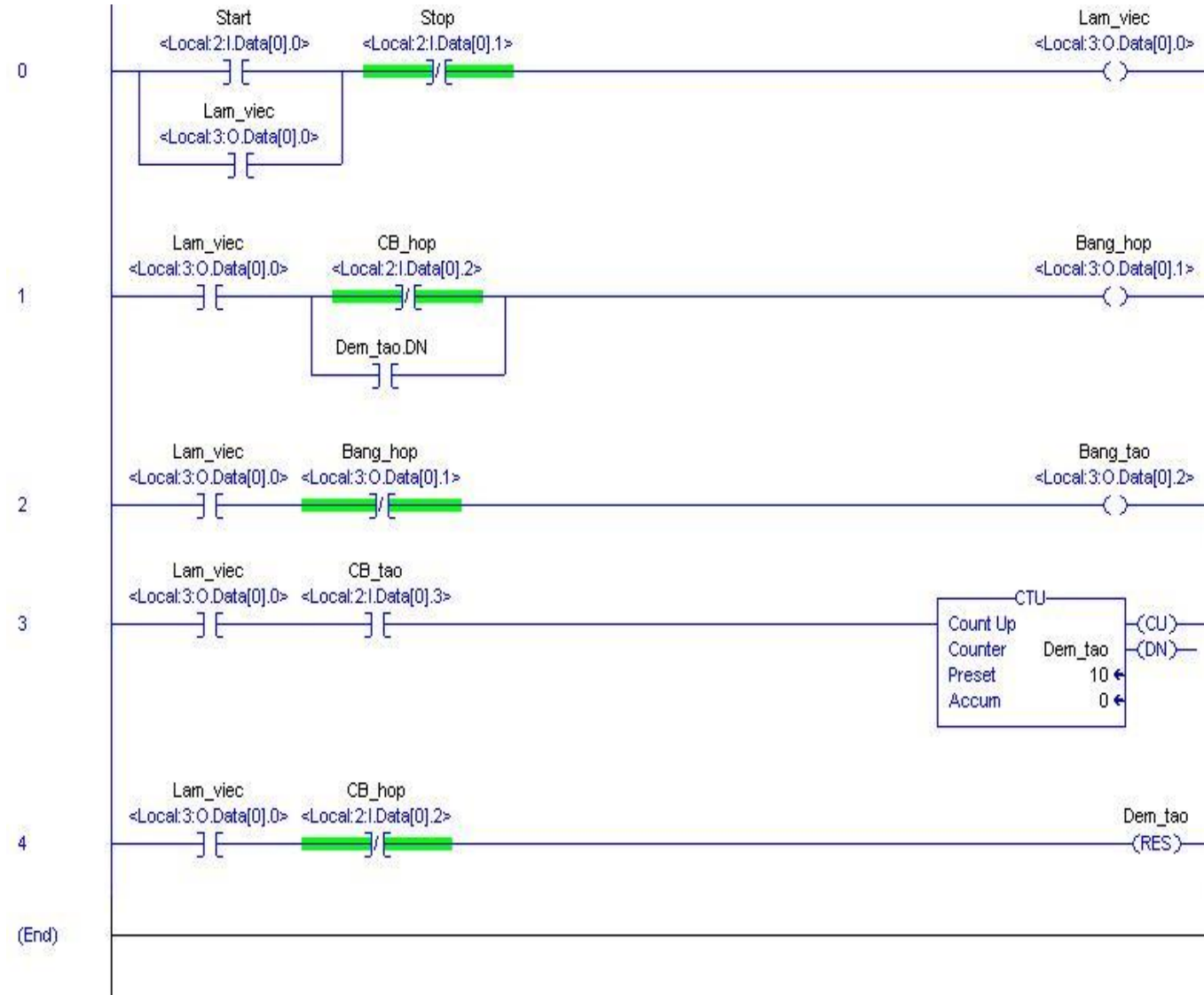
### Các bước lập trình:

- Lựa chọn PLC: CPU CompactLogix L32E, module IQ16 và OB16
- Phân địa chỉ vào/ra

STT	Tên tín hiệu	Loại tín hiệu	Đầu vào/Ra		Địa chỉ
			Vào	Ra	
1	Nút ấn khởi động - Start	Số	1		Local:2:I.Data.0
2	Nút ấn dừng - Stop	Số	1		Local:2:I.Data.1
3	Cảm biến hộp - CB_hop	Số	1		Local:2:I.Data.2
4	Cảm biến tảo – CB_tao	Số	1		Local:2:I.Data.3
5	Báo trạng thái làm việc (Lam_viec)	Số		1	Local:3:O.Data.0
6	Điều khiển bang tải hộp (Bang_hop)	Số		1	Local:3:O.Data.1
7	Điều khiển băng tải tảo (Bang_tao)	Số		1	Local:3:O.Data.2

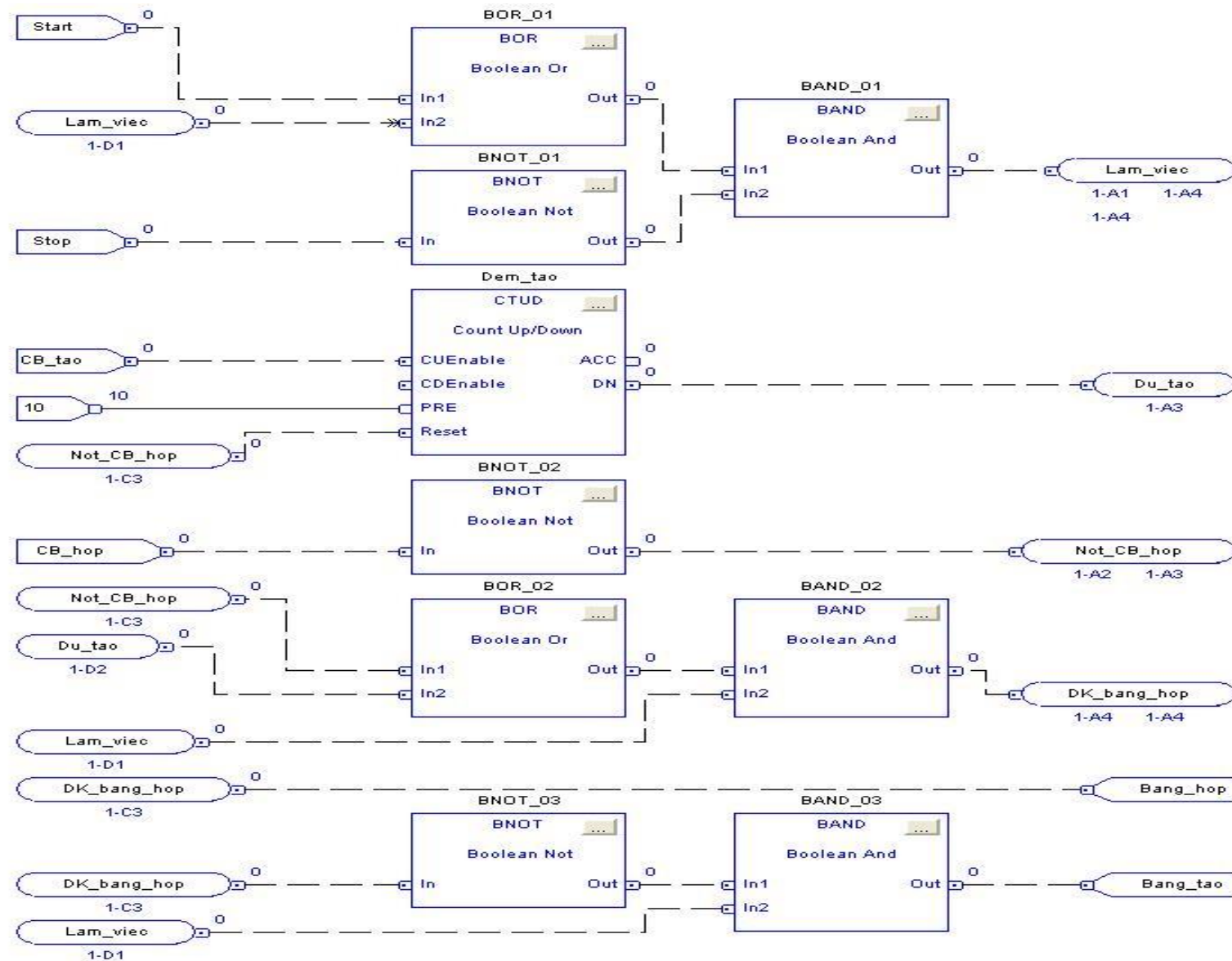
## 5.2. Các ngôn ngữ lập trình

### Dùng LD



## 5.2. Các ngôn ngữ lập trình

### Dùng FBD



## 5.2. Các ngôn ngữ lập trình

### Dùng ST

*Lam\_viec [:=] (Start or Lam\_viec) and not Stop;*

*CTUD(Dem\_tao);*

*Dem\_tao.CUEnable := CB\_tao;*

*Dem\_tao.pre := 10;*

*Dem\_tao.Reset:= Lam\_viec and not CB\_hop;*

*Bang\_hop [:=] Lam\_viec and (( not CB\_hop ) or  
Dem\_tao.DN);*

*Bang\_tao [:=] Lam\_viec and not Bang\_hop;*

# 5. Kỹ thuật lập trình PLC

5.1. Chu trình thiết kế chương trình PLC

5.2. Các ngôn ngữ lập trình

**5.3. Thiết kế chương trình sử dụng hàm logic**

5.4. Thiết kế chương trình sử dụng SFC

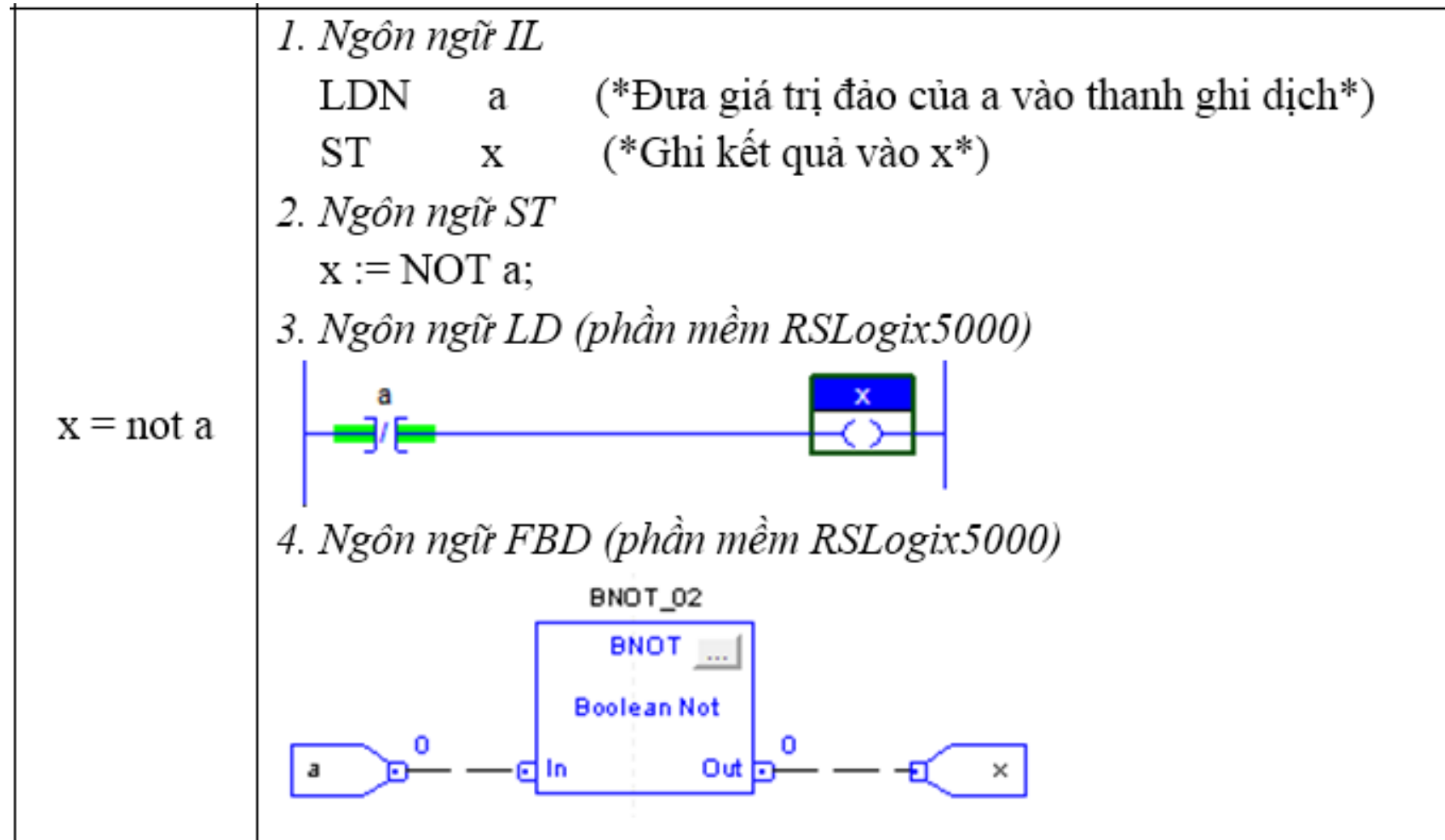
## 5.3. Thiết kế chương trình sử dụng hàm logic

**Tìm hàm logic mô tả mối quan hệ giữa các tín hiệu vào ra**

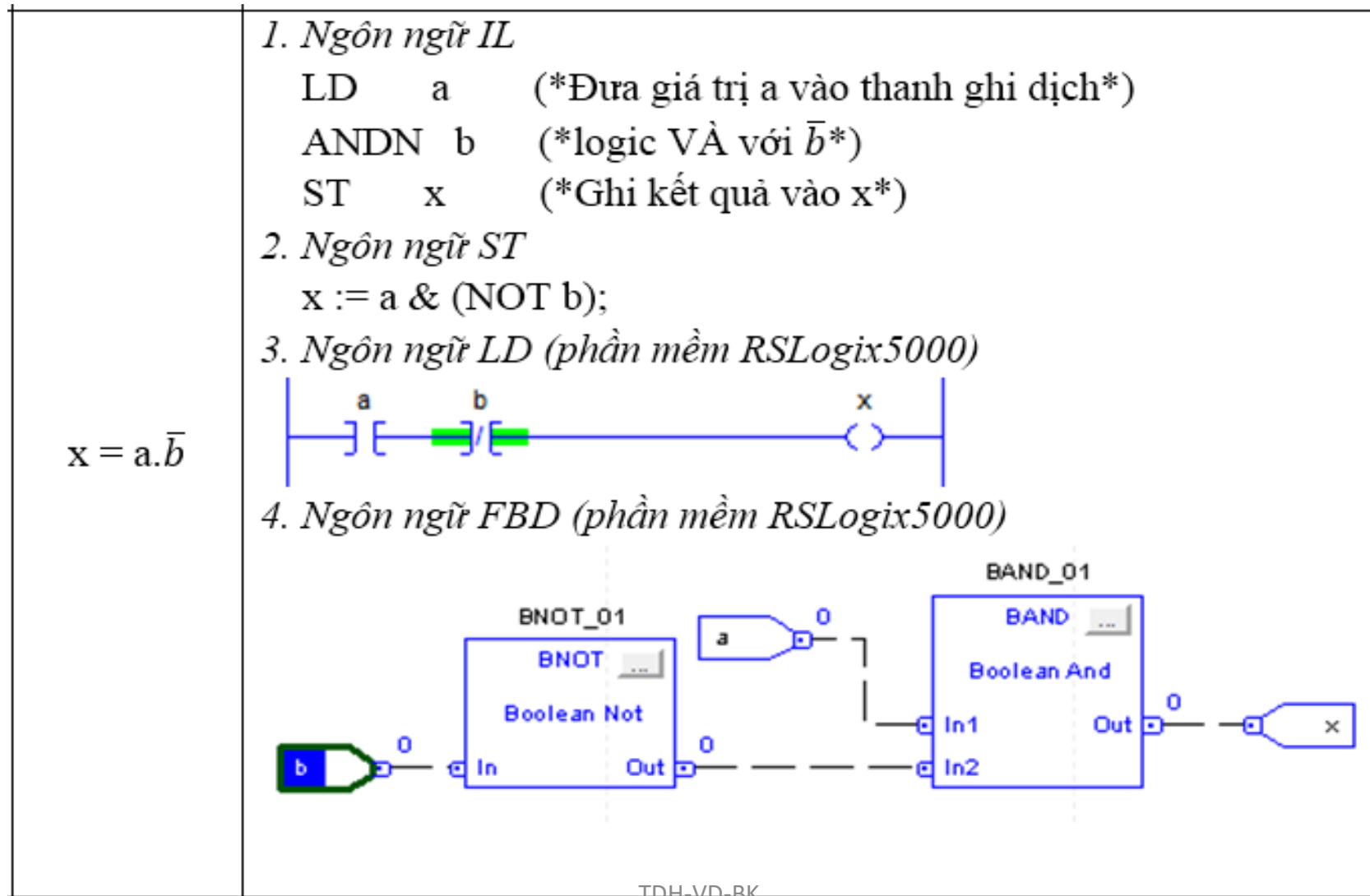
**Chuyển hàm logic sang ngôn ngữ PLC tương ứng**



## 5.3. Thiết kế chương trình sử dụng hàm logic

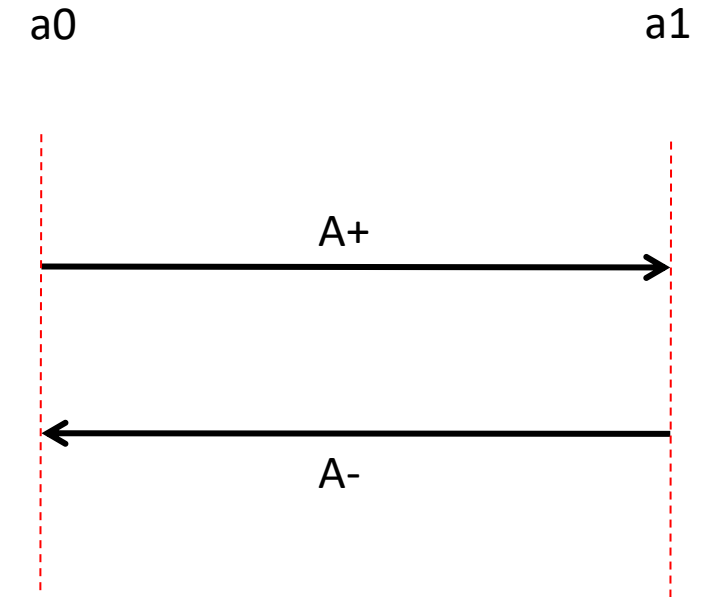


## 5.3. Thiết kế chương trình sử dụng hàm logic



## 5.3. Thiết kế chương trình sử dụng hàm logic

- Ban đầu thiết bị chạm ở vị trí bên trái và tác động vào công tắc hành trình a0. Khi nút m được ấn, thiết bị di chuyển sang phải. Khi chạm vào a1 và thiết bị di chuyển về bên trái. Tiếp theo thiết bị lại chạm vào a0, nếu nút m được ấn thì chu trình được lặp lại.
- Chú ý rằng khi thiết bị rời khỏi vị trí của công tắc hành trình thì công tắc hành trình lại trở về trạng thái không tác động.



## 5.3. Thiết kế chương trình sử dụng hàm logic

Sử dụng phương pháp ma trận trạng thái ta có thể tổng hợp các hàm logic mô tả công nghệ như sau:

$$X = a1 + \overline{a0}.X$$

$$A += \overline{X}$$

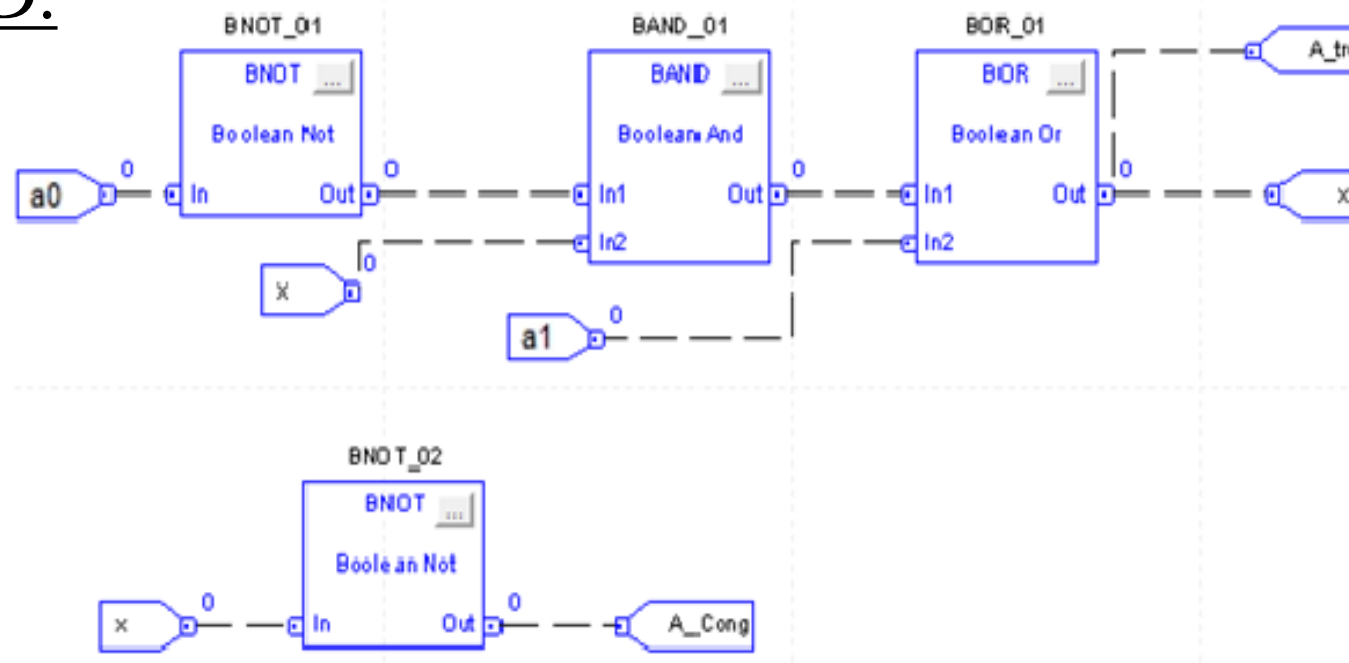
$$A -= X$$

LD:



## 5.3. Thiết kế chương trình sử dụng hàm logic

FBD:



IL:

**LDN a0**

**AND X**

**OR a1**

**ST X**

**LDN X**

**ST A\_cong**

**LD X**

**ST A\_tru**

ST:

$X := a1 \text{ OR } (\text{NOT } a0 \text{ and } X);$

$A\_cong := \text{NOT } X;$

$A\_tru := X;$

# 5. Kỹ thuật lập trình PLC

5.1. Chu trình thiết kế chương trình PLC

5.2. Các ngôn ngữ lập trình

5.3. Thiết kế chương trình sử dụng hàm logic

**5.4. Thiết kế chương trình sử dụng SFC**

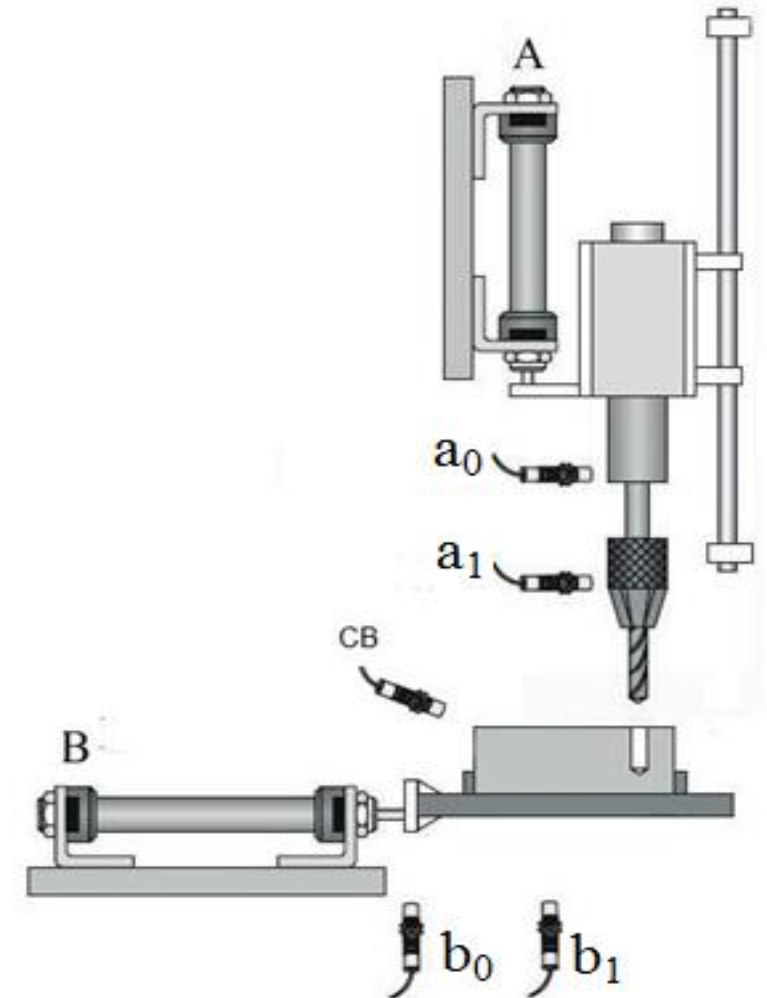
## 5.4. Thiết kế chương trình sử dụng SFC

**Lập Grafcet cho chu trình công nghệ**

**Chuyển Grafcet sang ngôn ngữ SFC**

## 5.4. Thiết kế chương trình sử dụng SFC

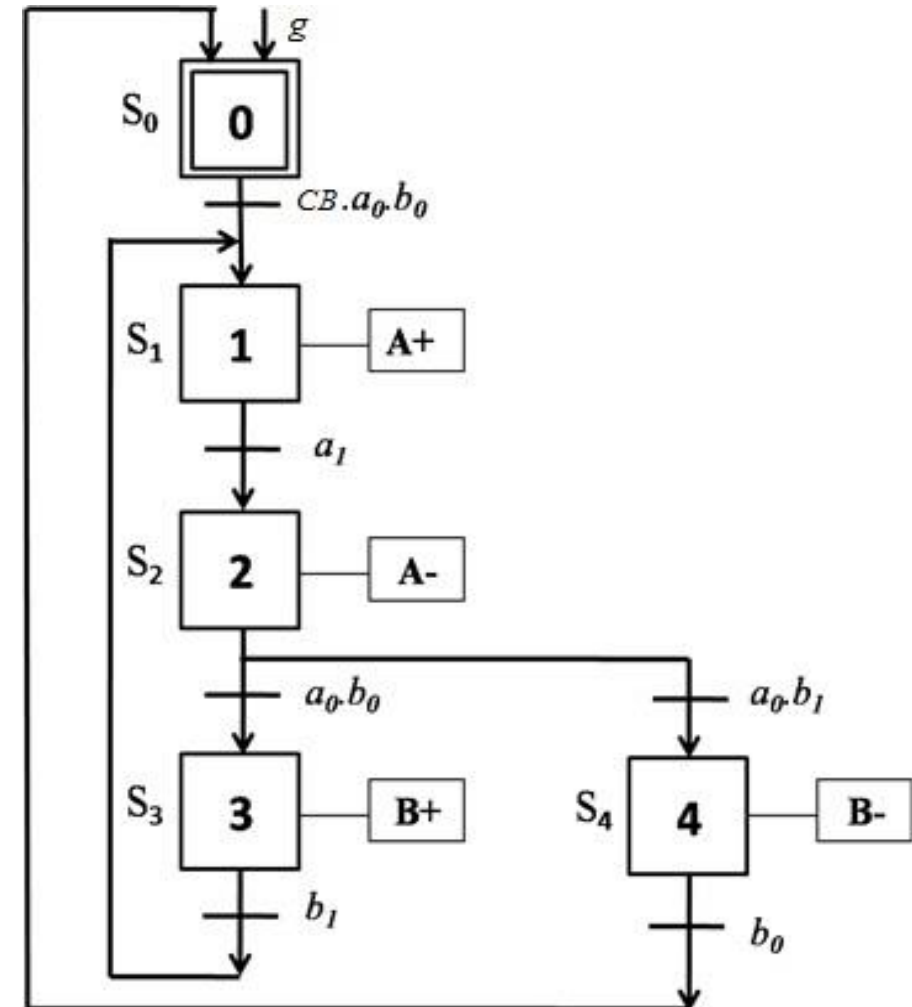
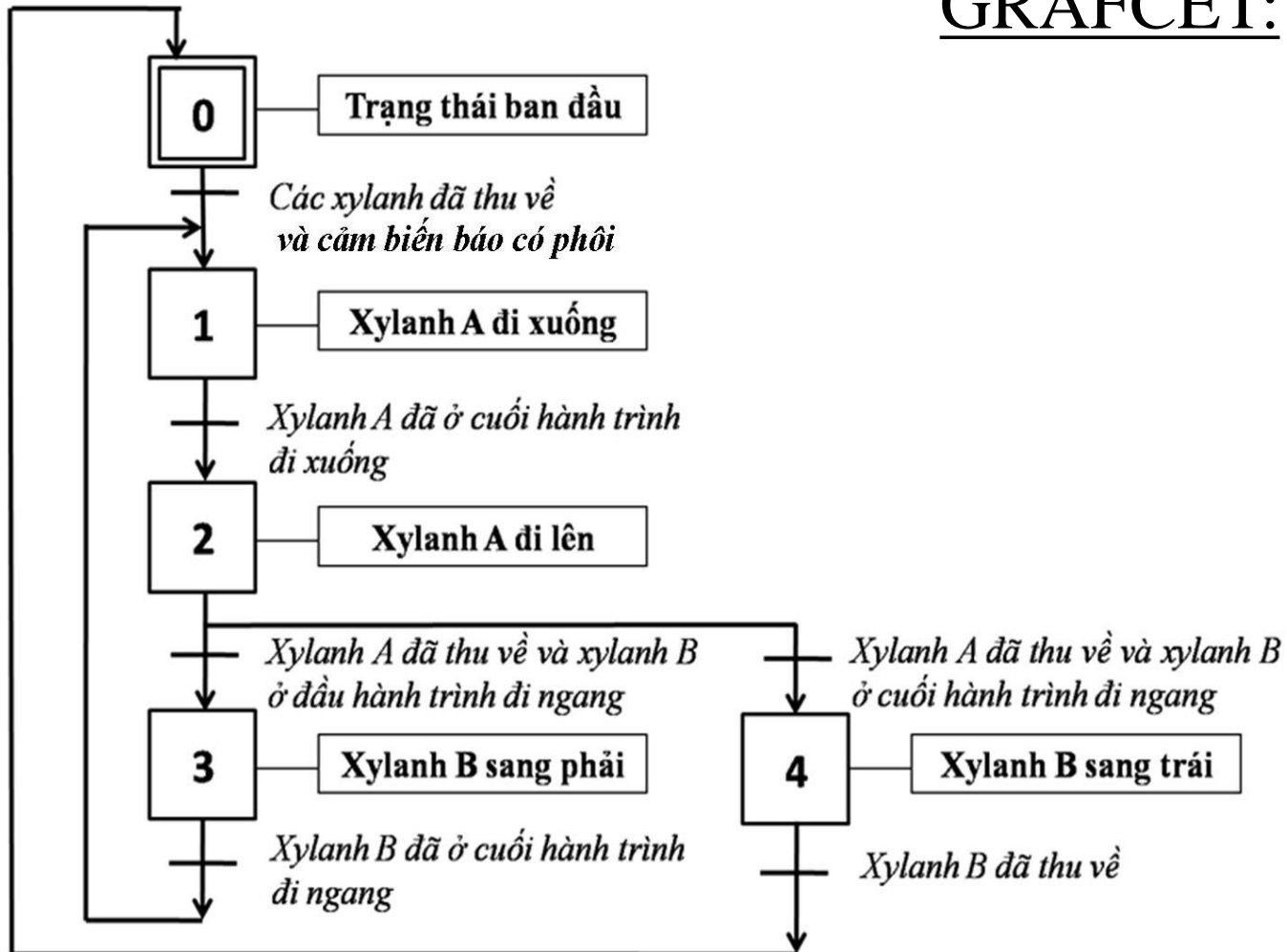
- Xylanh A chuyển động lên-xuống (di chuyển mũi khoan) và xylanh B chuyển động phải-trái (đưa phôi vào vị trí cần khoan). Ban đầu các xylanh đều thu về.
- Khi có tín hiệu từ cảm biến CB báo hiệu có phôi, xylanh A sẽ thực hiện chuyển động đưa mũi khoan đi xuống A+ (xylanh B đứng im) và đến khi có tín hiệu từ cảm biến  $a_1$  thì sẽ thực hiện chuyển động thu về A- (xylanh B vẫn đứng im). Khi cảm biến  $a_0$  có tín hiệu thì xylanh A dừng, xylanh B thực hiện chuyển động sang phải (B+). Khi cảm biến  $b_1$  có tín hiệu thì xylanh B sẽ dừng và xylanh A lại thực hiện chuyển động đi xuống, khi gặp cảm biến  $a_1$  thì sẽ đi lên và khi gặp cảm biến  $a_0$  thì xylanh A dừng. Lúc này xylanh B thực hiện chuyển động sang trái (B-), đến khi cảm biến  $b_0$  có tín hiệu thì dừng. Sản phẩm được lấy ra bởi một cơ cấu khác và chu trình sẽ được lặp lại nếu tiếp tục có phôi.





## 5.4. Thiết kế chương trình sử dụng SFC

### GRAFCET:



# SFC:

