05/11/2013

Tuesday, November 05, 2013 8:12 AM

3.4. Mô tả máy trạng thái hữu hạn tường minh (Định nghĩa trạng thái và mã hóa trạng thái; Khai báo biến trạng thái hiện tại và biến trạng thái kế tiếp; Mô tả hàm chuyển trạng thái và hàm đầu ra bằng câu lệnh if/case; Mô tả phần tử nhớ biến trạng thái; Mô tả sự khởi tạo FSM; Một số chú ý về sự đầy đủ các nhánh trong câu lệnh if/case) – 2 LT

3.4.1. Khái niệm về máy trạng thái hữu hạn (nhắc lại)
3.4.2. Mô tả FSM bằng Verilog HDL

3.5. Thiết kế mô tả hệ thống số dùng phương pháp ASMD (Khái niệm nút trạng thái, nút đầu ra có điều kiện, nút hoạt động thanh ghi và nút quyết định trong đồ thị ASMD; Chuyển đổi đồ thị ASMD thành mô tả Verilog; Ví dụ minh họa:) – 2 LT.
3.6. Thiết kế, mô tả hệ thống số dùng phương pháp FSMD (Khái niệm về đường dữ liệu và khối điều khiển; Giao tiếp giữa đường dữ liệu và khối điều khiển; Mô tả đường dữ liệu; Mô tả khối điều khiển; Khái niệm về đường dữ liệu pipeline; Ví dụ minh họa: Mạch đếm tăng giảm) – 2 LT

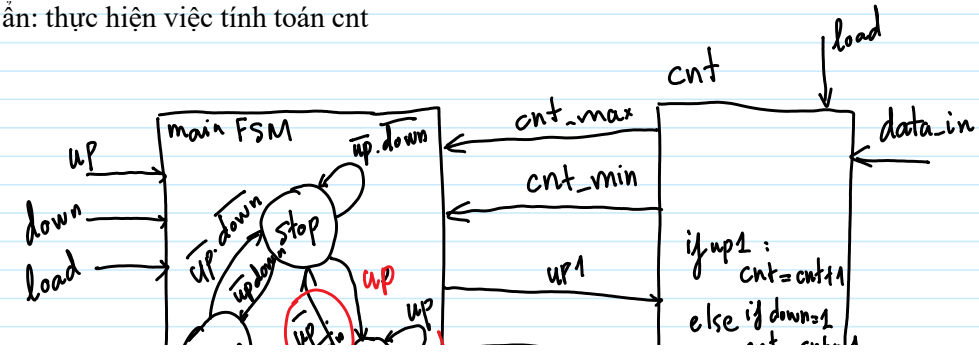
3.5.1. Máy trạng thái tương tác - Interactive FSM

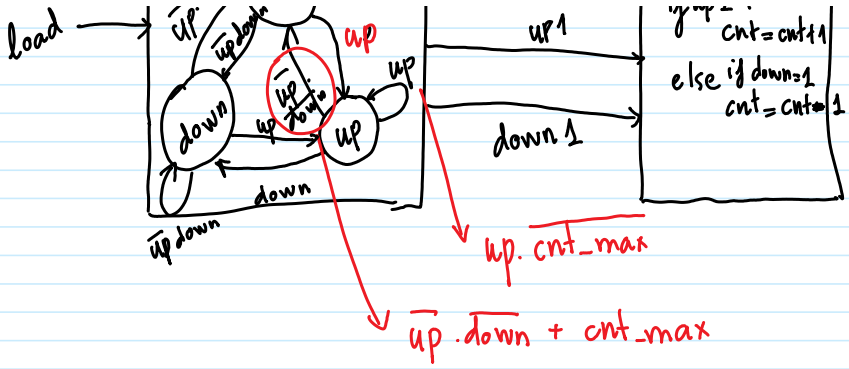
- Các hệ thống số phức tạp được thiết kế gồm nhiều máy trạng thái FSM tương tác với nhau
  - Đầu vào của một FSM có thể là đầu vào của toàn bộ hệ thống hoặc là đầu ra trạng thái của một FSM khác
  - Thông thường, khi 1 FSM ở một trạng thái nhất định, thì các FSM còn lại sẽ thực hiện một chuỗi chuyển trạng thái và khi chúng kết thúc sự chuyển trạng thái, thì FSM ban đầu sẽ thực hiện chuyển trạng thái.
  - Trong các hệ thống phức tạp, sẽ có một FSM chính (main-FSM) làm nhiệm vụ điều khiển hoạt động của toàn bộ hệ thống
- Ví dụ: Mạch đếm tăng, giảm, dừng
  - module counter(

```
input clk, rst_n,
input load, up, down,
input [3:0] data_in,
output reg [3:0] cnt);

// khi load = 1 thì cnt = data_in
// khi up = 1 thì cnt = cnt+1 cho đến khi cnt đạt giá trị max = 15, thì dừng
// khi down = 1 thì cnt = cnt-1 cho đến khi cnt đạt min = 0
// ngoài ra cnt = cnt
```

  - mạch đếm có thể thiết kế gồm 2 FSM:
    - FSM chính: điều khiển hoạt động
    - FSM ẩn: thực hiện việc tính toán cnt





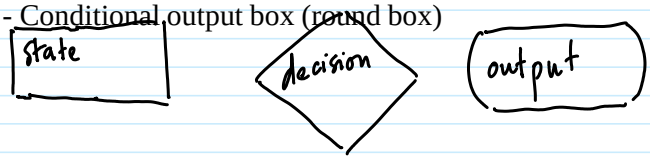
- Khi khối điều khiển ở trạng thái UP, bộ đếm sẽ thay đổi trạng thái: 0->1->2->3..., Khi khối điều khiển ở trạng thái DOWN, bộ đếm sẽ thay đổi trạng thái 4->3->2->...

Ví dụ: Mạch điều khiển đèn giao thông

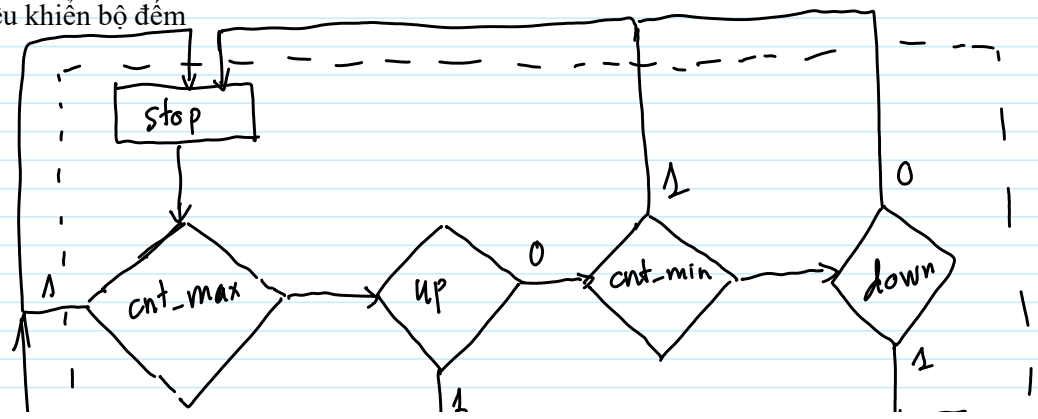
- Để thiết kế hệ thống gồm nhiều FSM tương tác nhau người ta sử dụng mô hình ASMD (Agorithmic State Machine Datapath) hoặc FSMD (Finite State Machine Datapath)

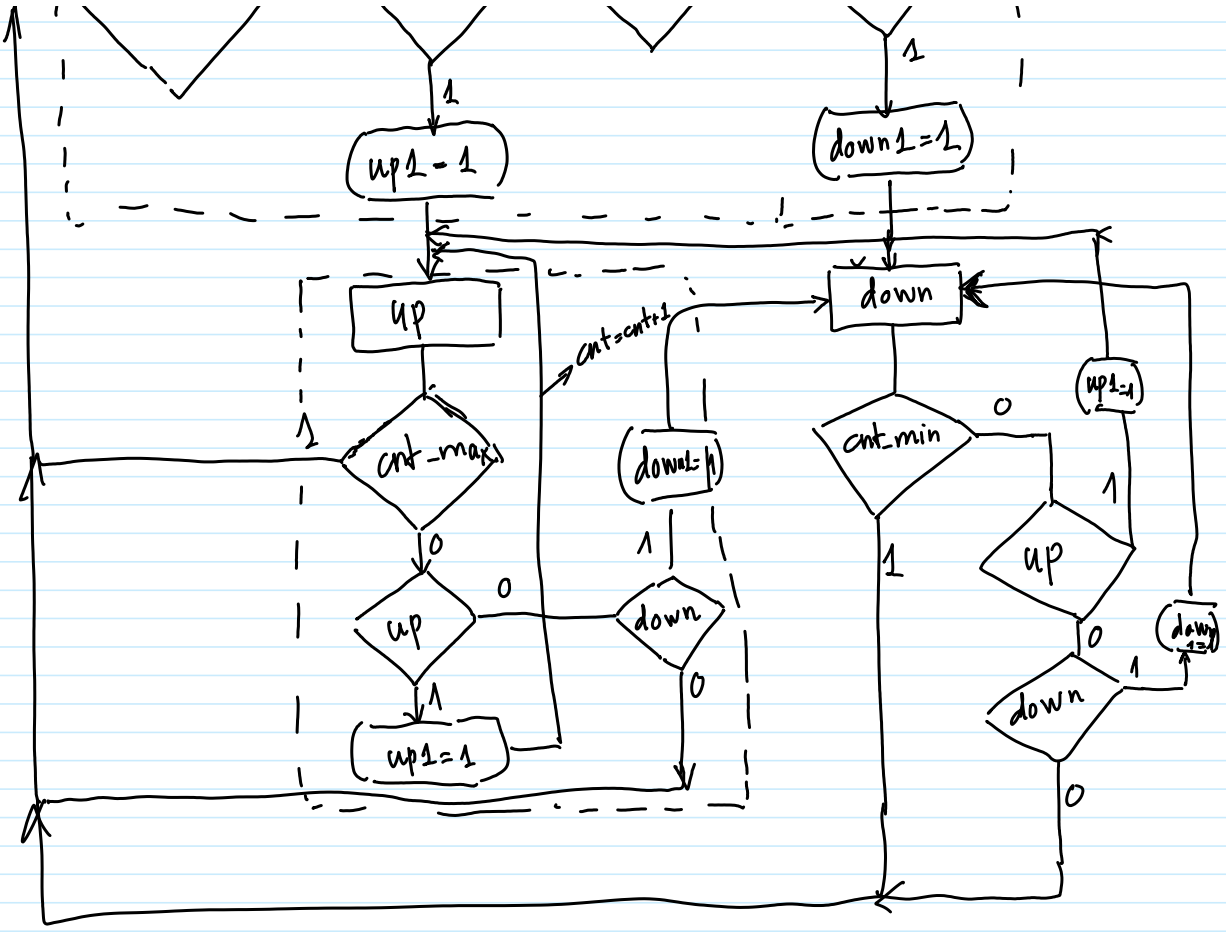
3.5.2. Mô hình thiết kế ASMD

- ASM: Là một đồ thị biểu diễn thuật toán sử dụng trong phần cứng gồm các nốt thuộc 3 loại
  - Node trạng thái - state box (hình chữ nhật)
  - Node quyết định - decision box (hình thoi)
  - Đầu ra điều kiện - Conditional output box (round box)



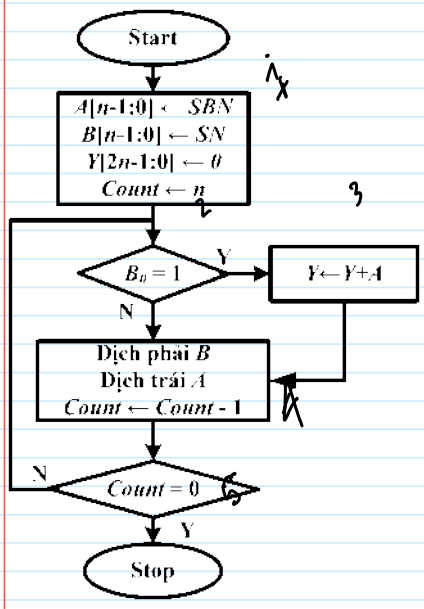
  - Các nốt trong ASM sẽ được nhóm thành các khối. Mỗi khối gồm
    - 1 nốt trạng thái
    - Nhiều nốt quyết định, nhiều nốt điều kiện
    - Một khối có nhiều cạnh đi vào nốt trạng thái của khối
    - Một khối có thể có nhiều cạnh đi ra nối với các khối khác
- ASM hoạt động như sau:
  - Đầu vào của ASM là các biến được kiểm tra trong nốt quyết định
  - Đầu ra của ASM là các biến được gán giá trị trong nốt đầu ra có điều kiện
  - Trạng thái của ASM là tập hợp các nốt trạng thái
  - Mỗi khối trong ASM tương ứng với 1 trạng thái của máy FSM và hoạt động trong 1 chu kỳ xung nhịp
  - Các cạnh giữa các khối tương ứng với 1 sự chuyển trạng thái của máy FSM và được thực hiện khi có sườn xung nhịp
- Ví dụ: ASM điều khiển bộ đếm

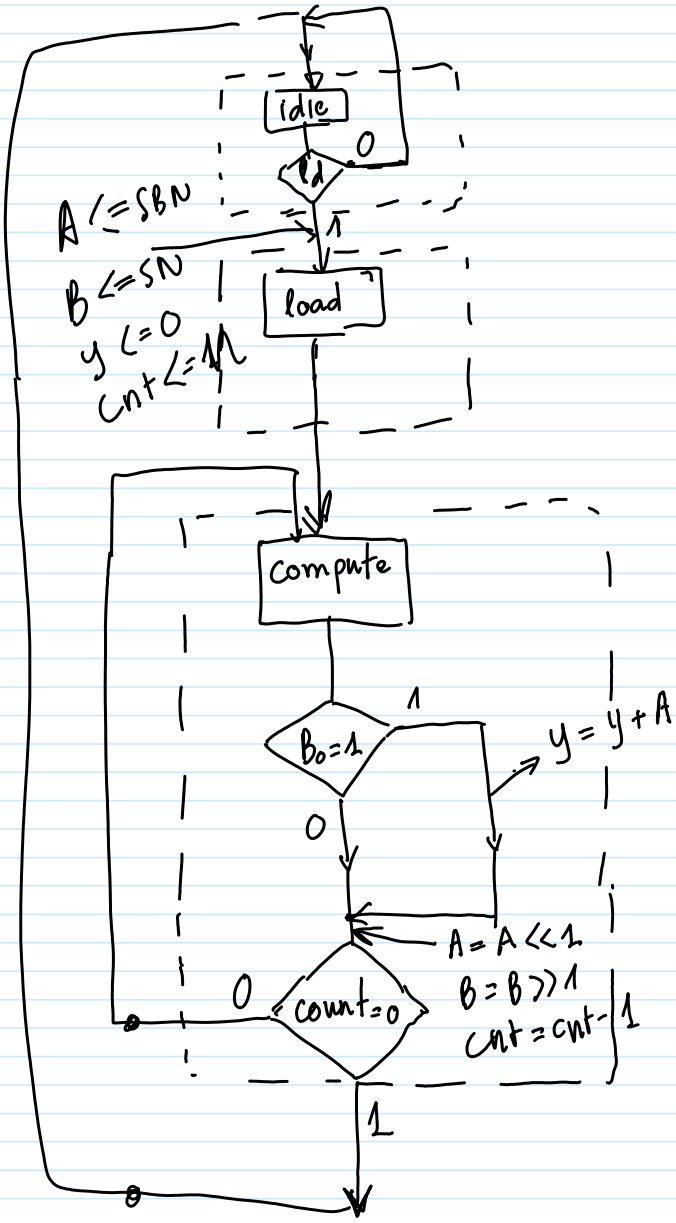




- ASMD: Sử dụng khi mô tả phần cứng gồm khối điều khiển và đường tính toán dữ liệu gồm
  - Đồ thị ASM
  - Các cạnh của đồ thị ASM sẽ được đánh dấu (annotate) bằng các phép toán thao tác trên dữ liệu.
  - Hoạt động của ASMD: Khi hệ thống chuyển từ khối này sang khối khác thông qua 1 cạnh (khi có sườn xung nhịp) thì phép toán tương ứng trên cạnh đó sẽ được thực hiện
- Các bước thiết kế dùng ASMD
  - B1: Tìm hiểu thuật toán và vẽ lưu đồ thuật toán
  - B2: Chia lưu đồ thuật toán thành các khối. Chú ý là độ phức tạp tính toán (số phép toán) trong mỗi khối nên tương đương nhau
  - B3: Chuyển đổi lưu đồ thuật toán thành ASM.
    - Thêm nốt trạng thái cho mỗi khối
    - Tách riêng phép toán thao tác trên dữ liệu ra đánh dấu các cạnh chuyển giữa các khối
  - B4: Xác định đầu vào của phần điều khiển ASM là các biến đầu vào trong nốt điều kiện và kết quả các phép so sánh dữ liệu (các biến đầu ra trạng thái của phần tính toán)
  - B5: Xác định đầu ra của phần điều khiển là các biến điều khiển thực hiện tính toán tại các cạnh chuyển giữa các khối. Thêm nốt đầu ra vào các khối trong ASMD
  - B6: Thực hiện mô tả bằng Verilog

- Ví dụ: Thực hiện bộ nhân nối tiếp n bít.

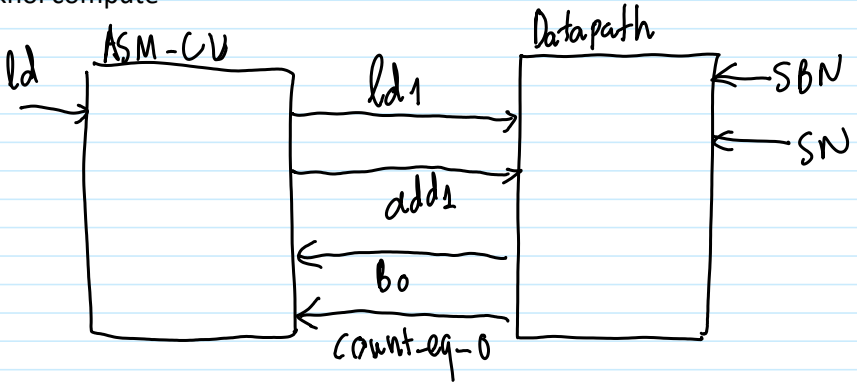


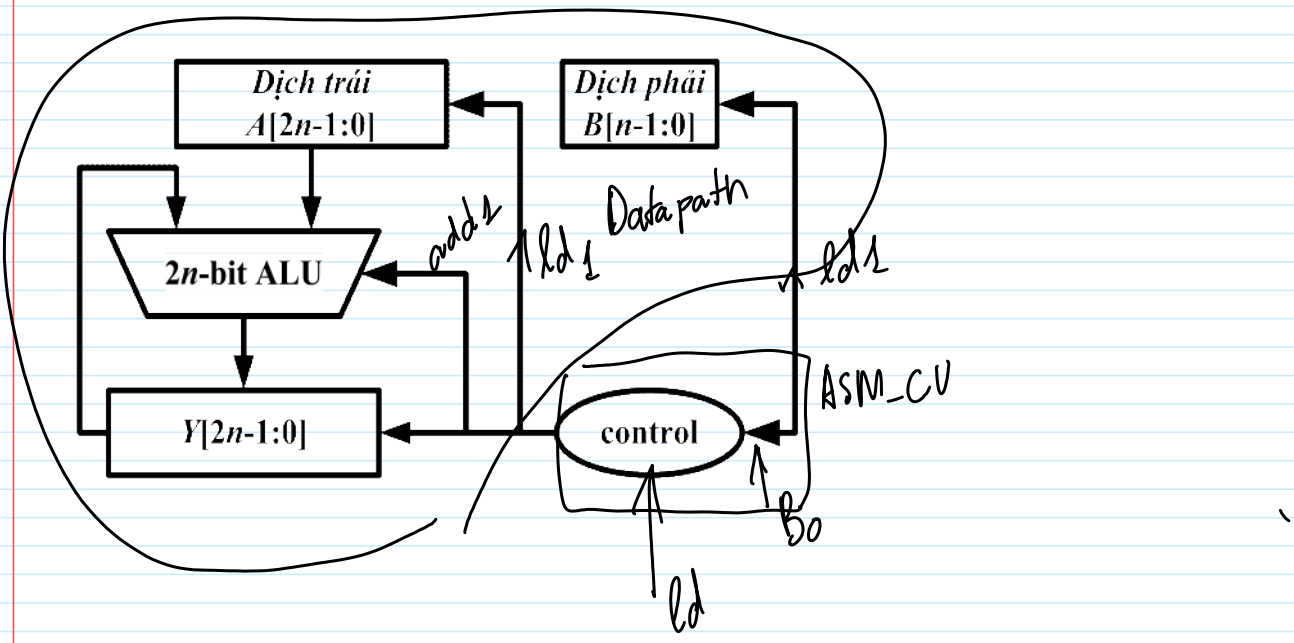


B1: Tìm hiểu vẽ lưu đồ thuật toán
B2: Chia lưu đồ thuật toán thành các khối
B3: Chuyển đổi lưu đồ thuật toán thành ASMD
    B3-1: Thêm nốt trạng thái vào mỗi khối
    B3-2: Xác định các cạnh nối các khối
    B3-3: Đánh dấu các cạnh bằng thao tác trên dữ liệu
B4: Xác định đầu vào, đầu ra của ASM
    B4-1: Đầu vào
    - Đầu vào ld
    - Đầu vào là trạng thái của đường dữ liệu: B0, count_eq_0

    B4-2: Đầu ra
    - Đầu ra điều khiển quá trình nạp dữ liệu: ld1
    - Đầu ra điều khiển quá trình cộng Y = Y+A: add1
    - Đầu ra điều khiển quá trình dịch A = A << 1; B = B>>1; count=count-1 có thể bỏ qua, vì hành động tính toán này luôn được thực hiện ở cạnh ra khỏi khối compute

thể bỏ qua, vì hành động tính toán này luôn được thực hiện ở cạnh ra khỏi khối compute





B5: Mô tả bằng Verilog

CHƯƠNG 4: Tổng hợp IC số, hệ thống số (12LT+3BT)
4.1. Khái niệm về tổng hợp logic (Lược đồ Y các biểu diễn mô hình biểu diễn mạch số; Đầu vào, đầu ra, các thành phần và các bước thực hiện cơ bản của phần mềm tổng hợp logic; Thư viện đích và thư viện liên kết trong tổng hợp; Giới thiệu sơ lược về một số kỹ thuật, thuật toán tổng hợp, tối ưu logic) – 1,5 LT
4.2. Tổng hợp mạch logic tổ hợp (Tổng hợp phép gán liên tục; Tổng hợp cấu trúc always và phép gán blocking; Tổng hợp cấu trúc if/case-Khái niệm cấu trúc ưu tiên priority structure; Tổng hợp sử dụng don't care; Chia sẻ tài nguyên trong tổng hợp và cấu trúc Verilog tương ứng; Tổng hợp mạch 3 trạng thái và bus) - 2,5 LT
4.3. Tổng hợp mạch dãy (Tổng hợp phần tử chốt: các lỗi thường gặp với phần tử chốt; Tổng hợp flip-flop; Tổng hợp FSM và các mạch dãy có hoạt động được mô tả bằng khối always được kích hoạt bằng 1 sườn đồng hồ; Tổng hợp mạch dãy có hoạt động được mô tả trong nhiều chu kỳ đồng hồ; Tổng hợp thanh ghi; Tín hiệu reset; Điều khiển tín hiệu đồng hồ) – 4LT
4.4. Mô tả và tổng hợp mạch bằng cấu trúc lặp trong Verilog (Vòng lặp tĩnh không có điều khiển thời gian – logic tổ hợp; Vòng lặp tĩnh có điều khiển thời gian – mạch dãy một chu kỳ, đa chu kỳ; Vòng lặp động có điều

khiển thời gian – mạch dãy một chu kỳ, đa chu kỳ; Vòng lặp động không có điều khiển thời gian – không tổng hợp; Cấu trúc generate) – 3 LT
4.5. Thiết kế và tổng hợp mạch phức tạp (Kỹ thuật chia để trị - thiết kế sơ đồ khối của hệ thống; Tái sử dụng thiết kế có sẵn và yêu cầu cần thiết; Khái niệm về nhân sở hữu trí tuệ; Tham số hóa module Verilog; Hàm và thủ tục trong Verilog) – 1 LT

Bài tập thực hành số 1: Thiết kế mạch điều khiển thang máy

- Specification
  - Input
    - Nút tại các tầng
      - input [5:1] up, down
    - Nút trong buồng thang
      - input keep_close, keep_open;
      - input [5:1] input_floor;
  - Output
    - output door; // = 1 mở, = 0 đóng
    - output go_up, go_down; // = 10 đi lên, = 01 đi xuống, 11, 00 dừng - điều khiển đèn led hiển thị hướng đi ở từng tầng
    - output [5:1] led_up, led_down; // đèn led nút bấm ngoài từng tầng
    - output [2:0] current_floor; // điều khiển đèn led hiện thị số tầng
    - output [5:1] led_floor; // đèn led nút bấm floor trong buồng thang
- Bài tập về nhà: Vẽ lưu đồ thuật toán mô tả hoạt động của thang máy