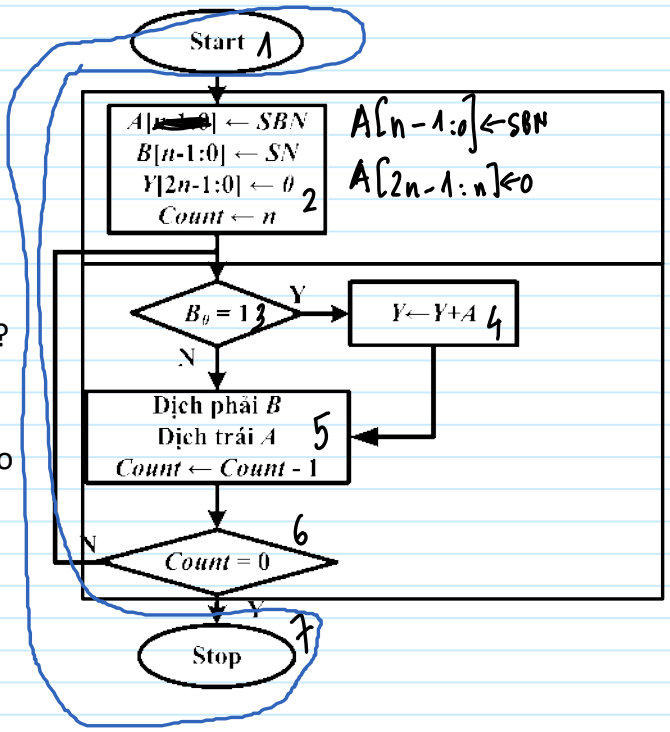
08/11/2013

Friday, November 08, 2013 3:11 PM

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

3.5.2. Mô hình thiết kế ASMD



B1: Tìm hiểu vẽ lưu đồ thuật toán
B2: Chia lưu đồ thuật toán thành các khối
Khối 1: Box 2 trong lưu đồ thuật toán
Khối 2: Box 3, 4, 5, 6 trong lưu đồ thuật toán
Khối 3: Box 1 và 7 trong lưu đồ thuật toán

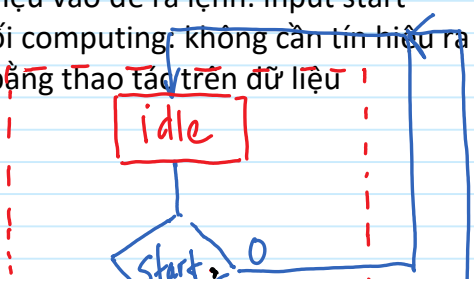
Q: Tại sao không ghép box 2 vào trong cùng khối 2?
Q: Lúc nào thì nên chia các box 3, 4, 5, 6 ra thành 2 khối?

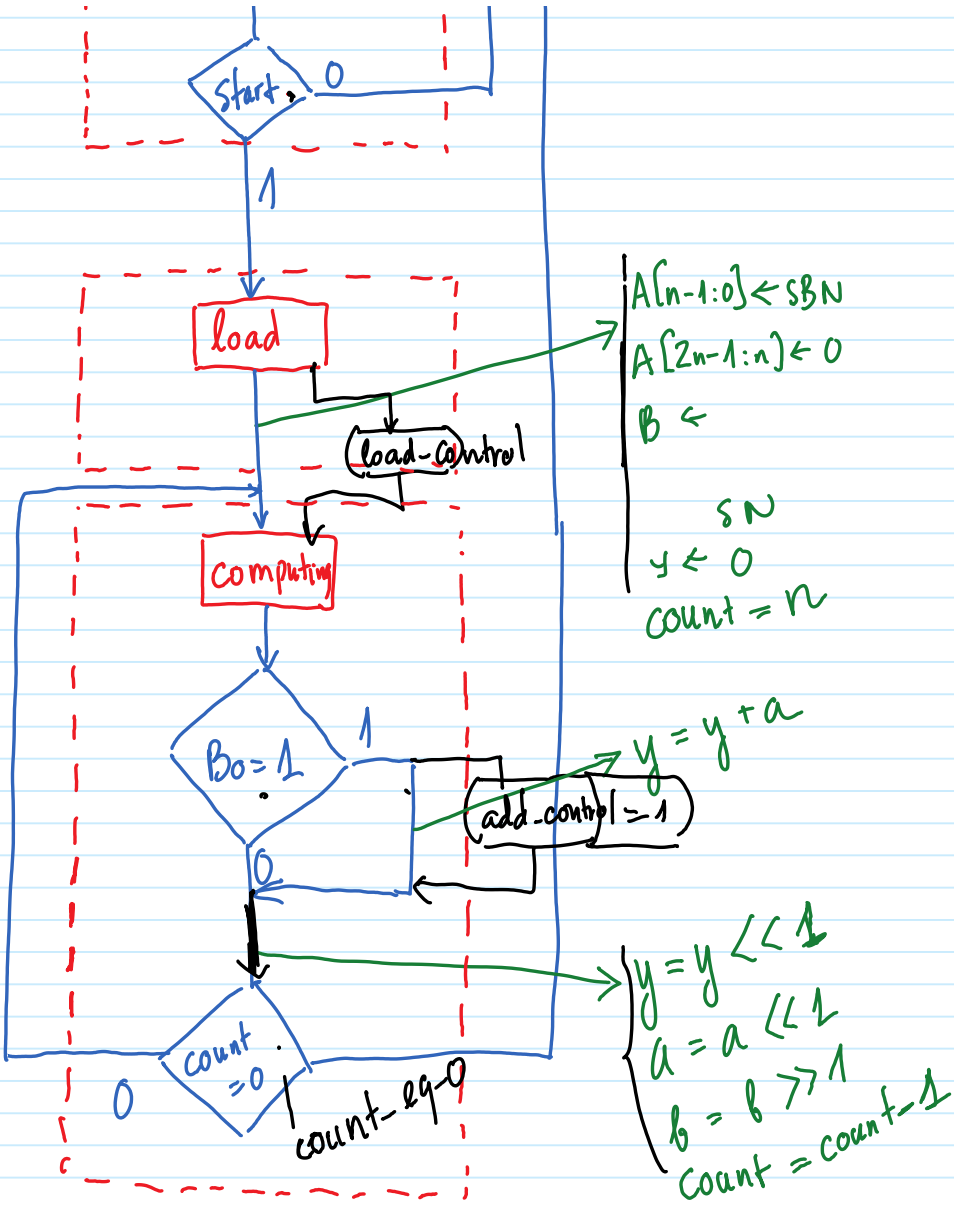
Nguyên tắc để ghép các box trong lưu đồ thuật toán vào cùng 1 khối:
- Các thao tác ở các box có thể cùng được thực hiện song song (bằng phần cứng) trong 1 chu kỳ xung nhịp.
- Các box có số lượng tính toán tương đương

B3: Chuyển đổi lưu đồ thuật toán thành ASMD
B3-1: Thêm nốt trạng thái vào mỗi khối
- Khối 1: Trạng thái load
- Khối 2: Trạng thái compute
- Khối 3: Trạng thái idle

B3-2: Xác định các cạnh nối các khối
- Chú ý: Khi nối khối idle và khối load: Khác với phần mềm tự động chạy, phần cứng cần 1 tín hiệu vào để ra lệnh: input start
- Từ khối load sang khối computing: không cần tín hiệu ra lệnh

B3-3: Đánh dấu các cạnh bằng thao tác trên dữ liệu





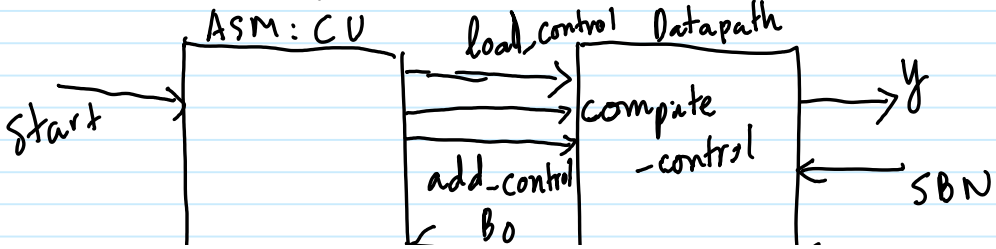
B4: Xác định đầu vào, đầu ra của ASM

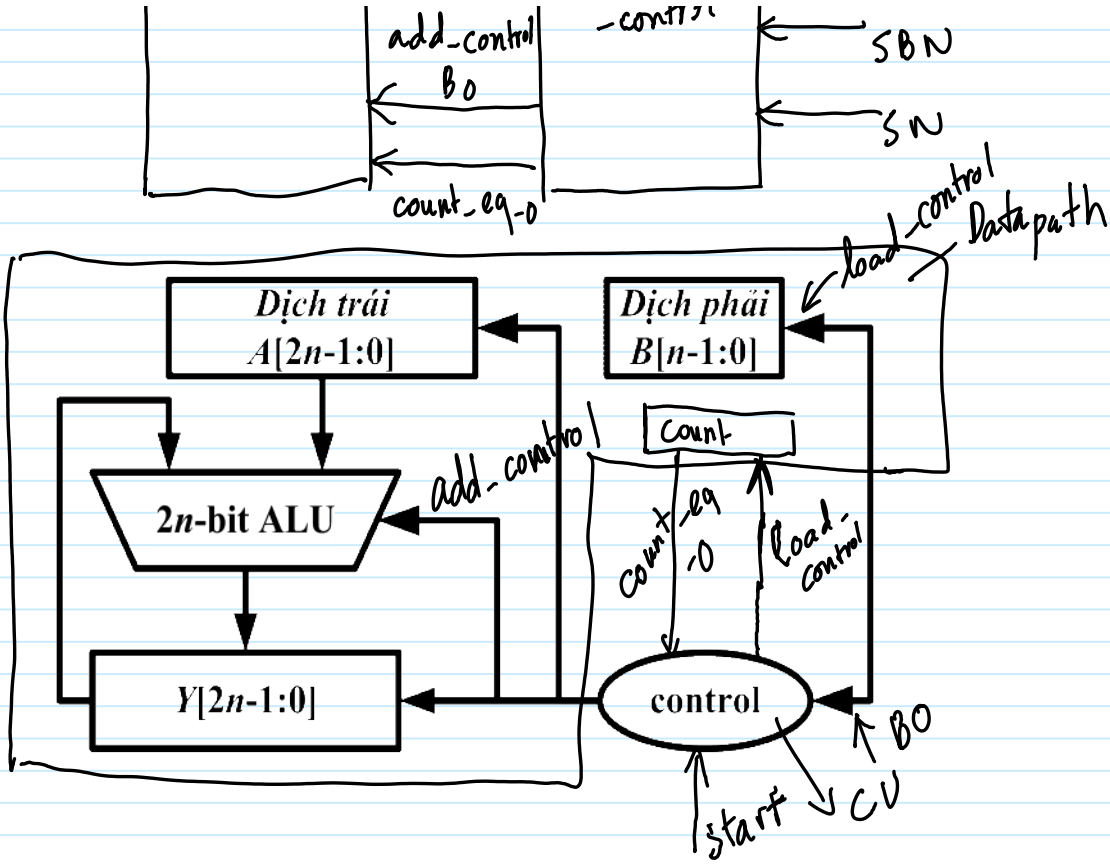
B4-1: Đầu vào

- Đầu vào từ môi trường: start
- 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: load_control
- Đầu ra điều khiển quá trình cộng Y = Y+A: add_control
- Đầu ra điều khiển quá trình dịch A = A << 1; B = B>>1; count=count-1 : compute_control





B5. Mô tả mạch bằng Verilog

```
module serial_mult #(parameter n=8)
      (input clk, rst_n,
            input                         start,
            input [n-1:0]         SBN,
            input [n-1:0]         SN,
            output [2*n-1:0]      y,
            output done);

      wire adder_control, load_control, compute_control;
      wire b0, count_eq_0;


      data_unit #(.n(n)) datapath
            (.clk(clk),
                  .rst_n(rst_n),
                  .SBN(SBN), .SN(SN),
                  .y(y),
                  .load_control(load_control),
                  .add_control(adder_control),
                  .shift_control(shift_control),
                  .count_eq_0(count_eq_0),
                  .b0(b0)
                  );
```

```
        control_unit #(.n(n)) control
                (.clk(clk), .rst_n(rst_n),
                        .start(start),
                        .b0(b0),
                        .count_eq_0(count_eq_0), // status signal from datapath
                        .load_control(load_control), // control signal to datapath
                        .add_control(adder_control),
                        .shift_control(compute_control),
                        .done(done)
                        );
endmodule

module data_unit #(parameter n = 8)
(
        input clk, rst_n,
        input [n-1:0] SBN, SN,
        output reg [2*n-1:0] y,
        input load_control, add_control, shift_control,
        output count_eq_0, b0
);
        reg [n-1:0] b;
        reg [2*n-1:0] a;
        reg [n-1:0] count;

        assign b0 = b[0];
        assign count_eq_0 = (count==0);

        always @(posedge clk or negedge rst_n)
        begin
                if (~rst_n)
                begin
                        a <= 0;
                        b <= 0;
                        count <= n;
                        y <= 0;
                end
                else
                begin
                        if (load_control)
                        begin
                                a(n-1:0) <= SBN;
                                b <= SN;
                                count <= n;
                                y <= 0;
                        end
                        if (add_control)
```

```
                y <= y+a;

            if (shift_control)
                begin
                    b <= b >> 1;
                    a <= a << 1;
                    count = count -1;
                end

        end
    end
endmodule
```

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: Vẽ lưu đồ thuật toán mô tả hoạt động của thang máy. Chuyển đổi lưu đồ thuật toán thành ASMD. Xây dựng kiến trúc mạch gồm Control_Unit và Datapath

Bài tập tự chọn (+10): Tối ưu bộ nhân nối tiếp sao cho kích thước là nhỏ nhất.