**3HA02**

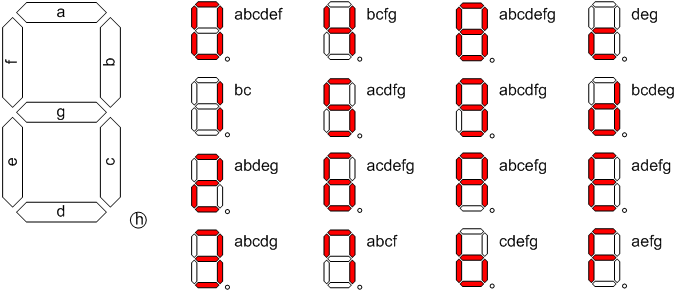
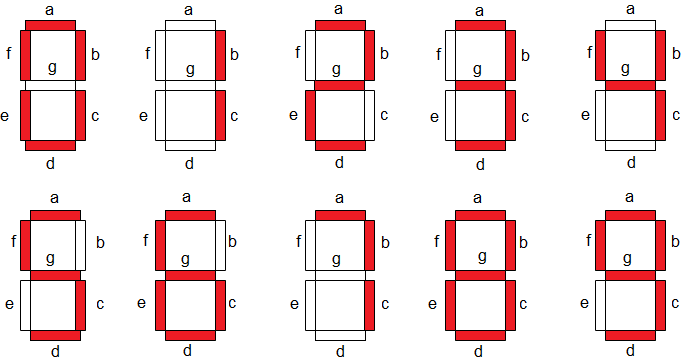
**Stop Watch**

**วัตถุประสงค์**

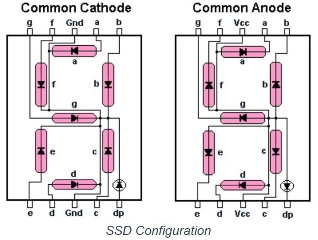
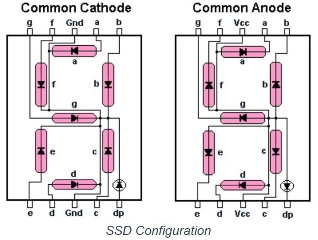
1. นศ. เข้าใจหลักการทำงานในการแสดงผลบน 7-Segment
2. นศ. สามารถออกแบบวงจรจับเวลาได้อย่างแม่นยำ

**7-Segment**

7-Segment คือหน้าจอแสดงผลตัวเลข โดยประกอบไปด้วย Led 8 ดวงเรียงตามตำแหน่งแสดงดังรูป 1 ซึ่งในการแสดงตัวเลขต่าง ๆ บน 7-Segment จะสามารถแสดงได้ตั้งแต่ เลข 0-9 และตัวอักษรบางตัว ซึ่งในการแสดงตัวเลขต่าง ๆ จะเป็นการควบคุมการติดดับของ Led บนตำแหน่ง a,b,c,d,e,f,g และ h (หรือ dp) โดยวงจรภายในของ 7-Segment จะสามารถแบ่งออกเป็น Common Anode หรือ Cathode แสดงดังรูป 2 สำหรับวงจร FPGA ที่ นศ.จะทำการทดลองจะเป็นประเภท Common Cathode

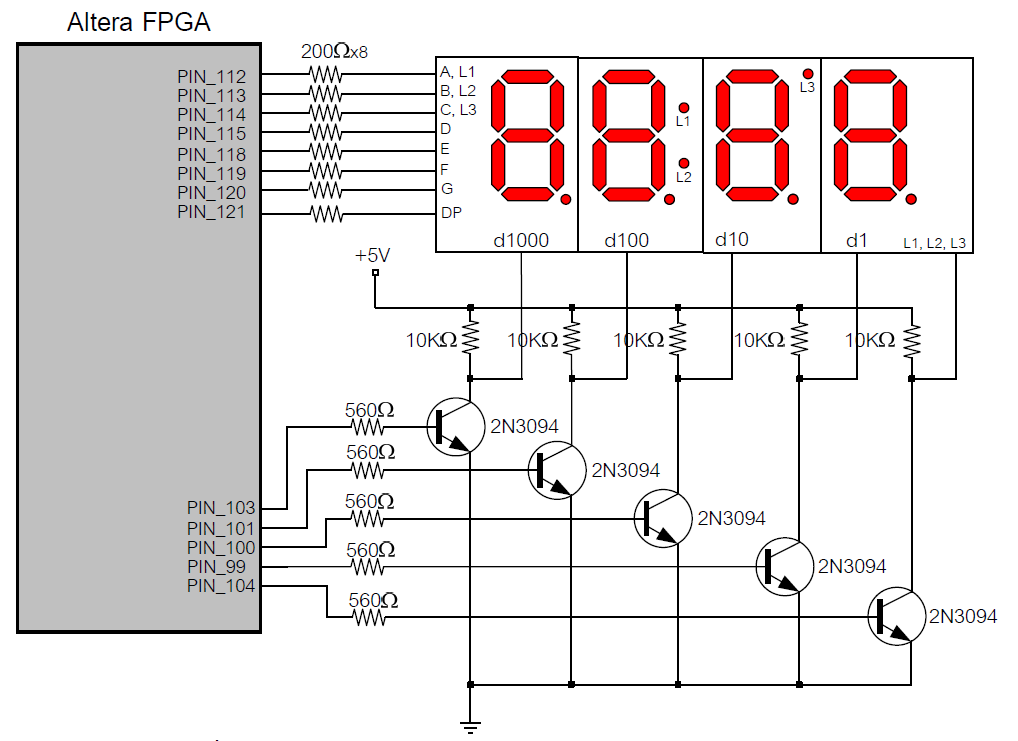
 

รูป 1 ตำแหน่ง Led บน 7-Segment และการแสดงตัวเลข

รูป 2 7- Segment แบบ Common Anode และ Cathode

โดยในวงจรของ Board FPGA ที่จะทำการทดลองจะมีการกำหนดการเชื่อมโยง 7- Segment ดังรูป 3 ในการพัฒนาโปรแกรมโดยภาษา Verilog สำหรับการควบคุมการทำงาน 7-Segment ตัวเลขใด ๆ แสดงดังตัวอย่างโปรแกรมที่1



รูป 3 7-Segment 4 หลักบน FPGA-Shield

ตัวอย่างโปรแกรมที่1 แปลงตัวเลขเพื่อแสดงบน 7-Segment

module digit2segment(clk,segmentShow,dp,segment1,segment2,segment3,segment4);

input clk;

output segment1,segment2,segment3,segment4,dp;

output [6:0] segmentShow;

reg [3:0] num; // temporary number

reg [6:0] abcdefg; // abcdefg on segment

always @(posedge clk) begin

num = 1;

//-------- Interger to segment ----------------------------------

case(num)

1: abcdefg = 7'b0110000;

2: abcdefg = ………………………………….;

3: abcdefg = ………………………………….;

4: abcdefg = ………………………………….;

5: abcdefg = ………………………………….;

6: abcdefg = ………………………………….;

7: abcdefg = ………………………………….;

8: abcdefg = ………………………………….;

9: abcdefg = ………………………………….;

0: abcdefg = ………………………………….;

endcase

end

assign segment1 = 1; assign segment2 = 1; assign segment3 = 1; assign segment4 = 1;

assign segmentShow = abcdefg; //num;

endmodule

**คำสั่งที่ 1** จงทำการแปลงเลขฐานสิบจากตัวแปร num เป็น binary 7 bit เพื่อแสดงบน 7-Segment (กรอกลงในเอกสารนี้ได้)

**คำสั่งที่ 2** ทำการทดสอบการทำงานของโปรแกรมโดยกำหนดให้เลขที่จะแสดงผลบน 7-Segment ทั้ง 4 หลักเป็นเลขตัวสุดท้ายของรหัส นศ.

จากตัวอย่างโปรแกรมที่ 1 แสดงให้เห็นถึงขั้นตอนอย่างง่ายในการแปลตัวเลขฐานสิบเพื่อแสดงบน 7-Segment ทุกตัวพร้อม ๆ กัน แต่อย่างไรก็ดีในการแสดงผลบน 7-Segment นั้นจะมีการแสดงผลโดยตัวเลขที่แตกต่างกันเช่น 123 122 1250 ซึ่งจะต้องมีการกำหนดความเร็วในการแสดงผลของแต่ละ 7-Segment (refresh rate) ให้เหมาะสม โดยปรกติแล้วสายตามนุษย์จะมีความเร็วตอบสนองต่อภาพที่เปลี่ยนแปลงเริ่มต้นที่ 24 ภาพต่อวินาที สำหรับการดูภาพเคลื่อนไหวบนหน้าจอแล้วมีความรู้สึกว่าภาพมีความต่อเนื่องไม่เป็นจังหวะ เป็นที่สังเกตได้ว่าในปัจจุบันจอคอมพิวเตอร์จะมีความเร็วในการ refresh rate เริ่มต้นที่ 60 Hz. เพื่อให้การแสดงผลภาพทั้งแบบภาพนิ่งและภาพเคลื่อนไหวมีความราบลื่น จากรูป 3 แสดงให้เห็นถึงการใช้สัญญาณ a,b,c,d,e,f,g,dp ร่วมกันของ 7-Segment ทั้ง 4 ตัว ซึ่งหมายความว่า 7-Segment แต่ละตัวจะแสดงผลตัวเลขตามคำสั่งควบคุมจากขา 99,100,101,103 และการแสดงจุดผ่านขา 104 รูป 4 ตัวอย่าง State diagram อย่างง่ายการแสดงผลเลข 1234 บน 7-Segment โดยความเร็วในการอัพเดตตัวเลขแต่ละหลักนั้นควรมีความเร็วไม่ต่ำกว่า 60 Hz. (t = 1/60 s.) โดยสามารถใช้ แหล่งกำเนิดสัญญาณนาฬิกาที่มีความเร็ว 50 Mhz. ในช่องสัญญาณ PIN\_17 ในการอ้างอิงการทำงานของระบบ



รูป 4 การแสดงเลข 1234 บน 7-Segment 4 หลัก

ตัวอย่างโปรแกรมที่ 2 refresh บน 7-Segment

parameter inClk=50000000; //reference clock

parameter refreshT = inClk/(60\*4); //60Hz 4 digit

reg [6:0] abcdefg; //รองรับตำแหน่งของ led

reg s4,s3,s2,s1; //ควบคุมการทำงานของ led แต่ละหลัก

reg [1:0] segmentID; // control 7-Segment id

reg [3:0] digit,ten,hundred,thousand; //รองรับตัวเลข 0-16 ของแต่ละตำแหน่ง

reg [15:0] num; // interger 16 bit

reg [26:0] clkRefresh; // count every clk (max= 67 million)

always @(posedge clk) //วงจร refresh

begin

clkRefresh = clkRefresh+1;

if (clkRefresh == refreshT)

begin

segmentID = segmentID + 1;

clkRefresh = 0; //reset to zero

end

case(segmentID) // shift 7-segment

2'b00 : begin s4 = 0; s3 = 0; s2 = 0; s1 = 1; num = thousand; end

2'b01 : begin s4 = 0; s3 = 0; s2 = 1; s1 = 0; num = hundred; end

2'b10 : begin s4 = 0; s3 = 1; s2 = 0; s1 = 0; num = ten; end

2'b11 : begin s4 = 1; s3 = 0; s2 = 0; s1 = 0; num = digit; end

endcase

case(num)

1: abcdefg = 7'b0110000;

2: abcdefg = ………………………………….;

.

.

.

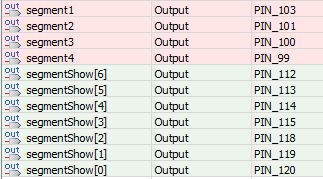
endcase

end

assign segment1 = s1; assign segment2 = s2; assign segment3 = s3; assign segment4 = s4;

assign segmentShow = abcdefg; //num;

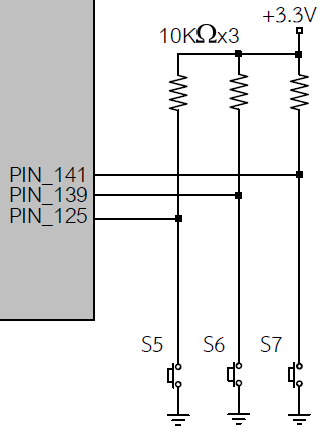
**คำสั่งที่ 3** จงพัฒนาโปรแกรมวงจรแสดงเลขรหัสนักศึกษา 3 ตัวสุดท้ายบน 7-Segment 3 หลัก บันทึก Code และผลการทำงาน



**คำสั่งที่ 4** จงพัฒนาโปรแกรมวงจร Stop watch มีความแม่นยำ 0.1 วินาที โดยกำหนดให้ปุ่ม บันทึก Code และผลการทำงาน

* S5 ทำหน้าที่เริ่มจับเวลา
* S6 ทำหน้าที่หยุดการจับเวลา
* S7 ทำหน้าที่รีเซ็ตเวลา

ตัวอย่างโปรแกรมฐานเวลา 0.1 วินาที



parameter pointOneT = inClk/10; // count every 0.1 second

reg [26:0] clkPointOne; //นับสัญญาณนาฬิกา

always @(posedge clk) begin // count every 0.1 ms

clkPointOne = clkPointOne +1;

if(clkPointOne == pointOneT) begin

countPointOne = (countPointOne<9999 ? countPointOne+1 : 0);

clkPointOne = 0;

end

digit = countPointOne%10;

ten = …………………………………..;

hundred = …………………………………..;

thousand = …………………………………..;

end



ตัวอย่างผลการทำงานของวงจรจับเวลา

