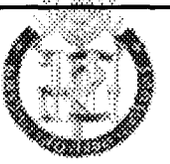
มหาวิทยาลัยสงขลานครินทร์
คณะวิศวกรรมศาสตร์



สอบปลายภาค: ภาคการศึกษาที่ 2 ปีการศึกษา: 2547
วันที่สอบ: 27 กุมภาพันธ์ 2548 เวลาสอบ: 9.00-12.00 น.
รหัสวิชา: 240-342 ห้องสอบ: A201
ชื่อวิชา: Logic Circuits Design

อ่านรายละเอียดของข้อสอบ และคำแนะนำให้เข้าใจก่อนเริ่มทำข้อสอบ

เวลา: 3 ชั่วโมง (180 นาที)

รายละเอียดของข้อสอบ: ข้อสอบมีทั้งหมด 15 หน้า

- เป็นกระดาษคำถามจำนวน 11 หน้า
- เป็น Datasheet จำนวน 4 หน้า

อนุญาต: เครื่องเขียนต่าง ๆ เช่น ปากกา หรือดินสอ

ไม่อนุญาต: เครื่องคิดเลข หนังสือและสมุดโน๊ตใด ๆ เข้าห้องสอบ

คำสั่ง:

- ให้ทำทุกข้อ
- คำตอบทั้งหมดจะต้องเขียนลงในข้อสอบ
- เขียนชื่อและรหัสให้ชัดเจนในข้อสอบ<u>ทุกแผ่น</u> **แผ่นใดไม่เขียนหรือเขียนไม่ครบจะถูกหักคะแนน 1 คะแนน**
- คำตอบส่วนใดอ่านไม่ออก จะถือว่าคำตอบนั้นผิด
- อ่านคำสั่งเพิ่มเติมในแต่ละข้อให้ชัดเจน
- **ทุจริตในการสอบมีโทษขั้นต่ำปรับตกในรายวิชานั้นและพักการเรียน 1 ภาคการศึกษา โทษสูงสุดให้ออก**

1. จงออกแบบวงจรคูณเลขขนาด 4 บิตของตัวเลขแบบ 2's complement signed number โดยมีเอาต์พุตของผลคูณขนาด 8 บิต

- หากป้อนอินพุต เข้าที่วงจรค่า "1001" และ "0110" จงคำนวณหาเอาต์พุตของวงจรที่ควรจะเป็น (3 คะแนน)

- บล๊อกไดอะแกรมของวงจร (5 คะแนน)

- จงหาย่านของค่าที่เป็นไปได้ทั้งหมดของอินพุตและเอาต์พุตของวงจร(3 คะแนน)

2. จงตอบคำถามต่อไปนี้

- Carry-lookahead adder มีความแตกต่างจากวงจร ripple adder อย่างไรในแง่ของจำนวนเกทและความเร็วของวงจร (3 คะแนน)

- Booth algorithm คืออะไร มีประโยชน์อย่างไร และ กรณีใดที่ Booth algorithm จะมีประโยชน์ในการทำงานสูงสุด (5 คะแนน)

- Carry-Save addition คืออะไรมีประโยชน์อย่างไรจงอธิบายหลักการทำงานมาพอสังเขป (5 คะแนน)

- Multiplicand คืออะไร(0.25 คะแนน)

- Multiplier คืออะไร(0.25 คะแนน)

- Dividerคืออะไร(0.25 คะแนน)

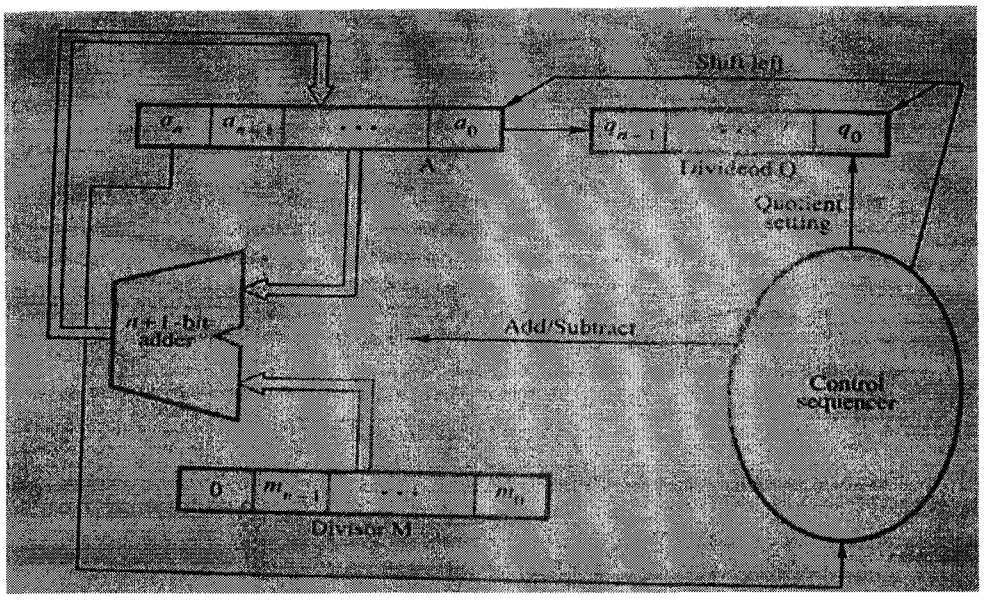
- Dividend คืออะไร(0.25 คะแนน)

- Quotient คืออะไร(0.25 คะแนน)

- Remainder คืออะไร(0.25 คะแนน)

- Summand คืออะไร(0.25 คะแนน)

3. จากบล๊อกไดอะแกรมของวงจร 6-bit unsigned binary division ต่อไปนี้ หาก implement วงจรดังกล่าวด้วยเทคนิค Restoring division และ เทคนิค Non-Restoring division แล้ว จงหาว่าการทำงานในแต่ละ สเตปของวงจรทั้งสองแตกต่างกันอย่างไร (12 คะแนน)

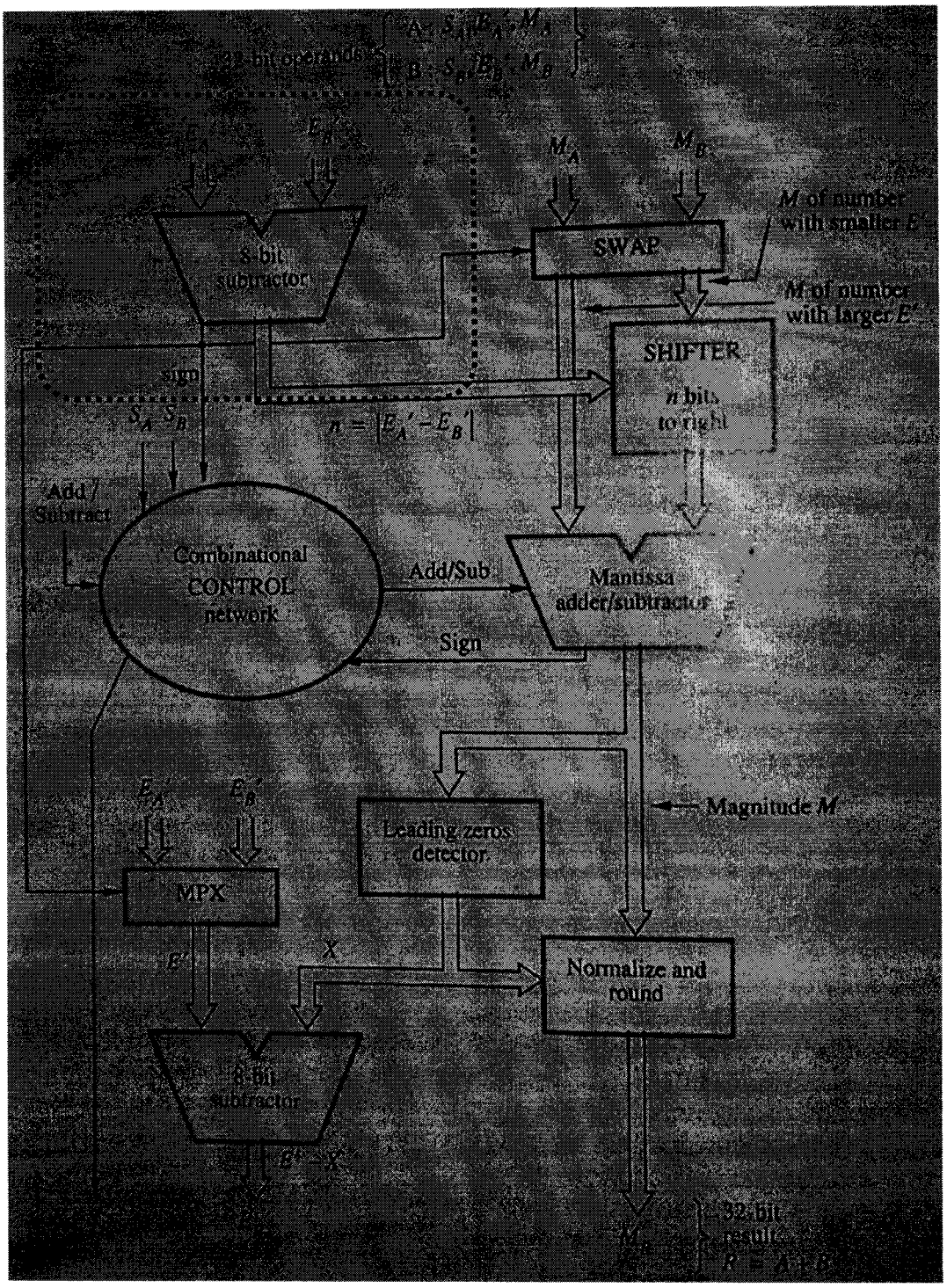


- จงแสดงลำดับขั้นตอนการคำนวณเลขของวงจรในแต่ละคล๊อกไซเคิลของแต่ละวงจรเมื่อป้อนตัวตั้งเท่ากับ "101000" และตัวหารเท่ากับ "000101" (9 คะแนน)

Restoring Division	Non-restoring division

- จำนวนคล๊อกไซเคิลที่ใช้ในการทำงานของวงจร Non-Restoring division(1 คะแนน).....
- จำนวนคล๊อกไซเคิลที่ใช้ในการทำงานของวงจร Non-Restoring division(1 คะแนน).....

4. จากบล๊อกไดอะแกรมของวงจร Floating-point addition-subtraction unit ต่อไปนี้ จงออกแบบวงจร SWAP และ วงจร 8-bit subtractor(วงจรที่ล้อมรอบด้วยเส้นประ) ด้วยภาษาวีเอชดีแอล (15 คะแนน)



- วงจร SWAP เขียนด้วยภาษาวีเอชดีแอล(7 คะแนน)

- วงจร 8-bit subtractor(วงจรที่ล้อมรอบด้วยเส้นประ) ด้วยภาษาวีเอชดีแอล (8 คะแนน)

5. จากซีพียู COE1 ที่นักศึกษาได้เรียนมา หากต้องการสร้าง control unit แบบ Hardwired แล้ว จงออกแบบวงจรสร้างสัญญาณควบคุม INC_PC (12 คะแนน)

คำแนะนำ ให้ดู Datasheet ของซีพียูท้ายข้อสอบประกอบ

- หาสมการลอจิกของสัญญาณควบคุม INC_PC (7 คะแนน)

- วาดวงจรสร้างสัญญาณควบคุม INC_PC (5 คะแนน)

6. จากอัลกอริทึมการคูณเลขแบบทั่วไปและแบบ Booth algorithm จงแสดงวิธีการคูณเลขแบบทั่วไป เปรียบเทียบกับวิธีของ Booth กับตัวเลขที่กำหนดให้(15 คะแนน)

0101101 ตัวตั้ง

0011110 ตัวคูณ

-จงหาค่า Booth recoded multiplier (3 คะแนน)

- จงแสดงวิธีการคูณเลขแบบทั่วไปพร้อมหาจำนวนครั้งของการบวกของ Summand(3 คะแนน)

- จงแสดงวิธีการคูณเลขแบบ Booth algorithm พร้อมหาจำนวนครั้งของการบวกของ Summand(5 คะแนน)

.....อ.ปัญญยศ ไชยกาฬ ผู้ออกข้อสอบ

Control signal activation of COE1 CPU

Fetching phase for all instructions

Step1 PC_OUT, MAR_IN, RD
Step2 INC_PC, MDR_IN_F_DATABUS, RD
Step3 IR_IN, MDR_OUT

1. LD Rx,Ry instruction execution phase
Step4 PC_OUT, MAR_IN, RD
Step5 INC_PC, MDR_IN_F_DATABUS, RD
Step6 MDR_OUT, IR_High_in
Step7 Rx_out, Ry_in, End

2. LD Rx,[Reg16] instruction execution phase
Step4 MAR_IN, RD, Reg16_out
Step5 MDR_IN_F_DATABUS, RD
Step6 Rx_in, MDR_out, End

3. LD Ri,#data8 instruction execution phase
Step4 PC_out, MAR_IN, RD
Step5 INC_PC, MDR_IN_F_DATABUS, RD
Step6 Ri_in, MDR_out, End

4. Store [Reg16], Rx instruction execution phase
Step4 Reg16_out, MAR_IN, Rx_out, MDR_in
Step5 MDR_OUT_to_DATABUS, WR
Step6 MDR_OUT_to_DATABUS, End

5. ADDC Rx,Ry instruction execution phase
Step4 PC_OUT, MAR_IN, RD
Step5 INC_PC, MDR_IN_F_DATABUS, RD
Step6 MDR_OUT, IR_High_in
Step7 Rx_out, TMP1_in
Step8 ALU_ADD, RY_out
Step9 Y_out, Rx_in, End

5. SUBB Rx,Ry instruction execution phase
Step4 PC_OUT, MAR_IN, RD
Step5 INC_PC, MDR_IN_F_DATABUS, RD
Step6 MDR_OUT, IR_High_in
Step7 Rx_out, TMP1_in
Step8 ALU_SUB, RY_out
Step9 Y_out, Rx_in, End

7. AND Rx,Ry instruction execution phase
Step4 PC_OUT, MAR_IN, RD
Step5 INC_PC, MDR_IN_F_DATABUS, RD
Step6 MDR_OUT, IR_High_in
Step7 Rx_out, TMP1_in
Step8 ALU_AND, RY_out
Step9 Y_out, Rx_in, End

8. OR Rx,Ry instruction execution phase
Step4 PC_OUT, MAR_IN, RD
Step5 INC_PC, MDR_IN_F_DATABUS, RD
Step6 MDR_OUT, IR_High_in
Step7 Rx_out, TMP1_in
Step8 ALU_OR, RY_out
Step9 Y_out, Rx_in, End

9. IN Ri, port instruction execution phase
Step4 in_port_pin_to_internal_bus, Ri_in, End

10. OUT port,Ri instruction execution phase
Step4 Port_in_f_internal_bus, Ri_out, End

11. NOT Rx instruction execution phase
Step4 Rx_out, ALU_NOT
Step5 Y_out, Rx_in,End

12. SHR Rx instruction execution phase
Step4 Rx_out, ALU_SHR
Step5 Y_out, Rx_in, End

13. RL Rx instruction execution phase
Step4 Rx_out, ALU_RL
Step5 Y_out, Rx_in, End

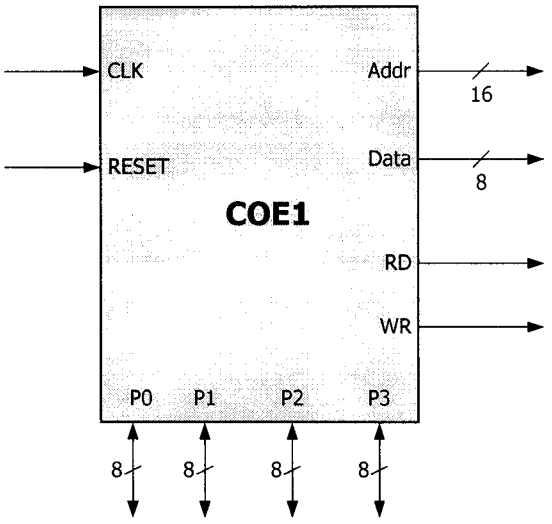
14. JMP nn instruction execution phase
Step4 PC_OUT, MAR_IN, RD
Step5 INC_PC, MDR_IN_F_DATABUS, RD
Step6 MDR_OUT, TMP3_in
Step7 PC_OUT, MAR_IN, RD
Step8 INC_PC, MDR_IN_F_DATABUS, RD
Step9 MDR_OUT, TMP2_in
Step10 PC_in, TMP2_out, TMP3_out, End

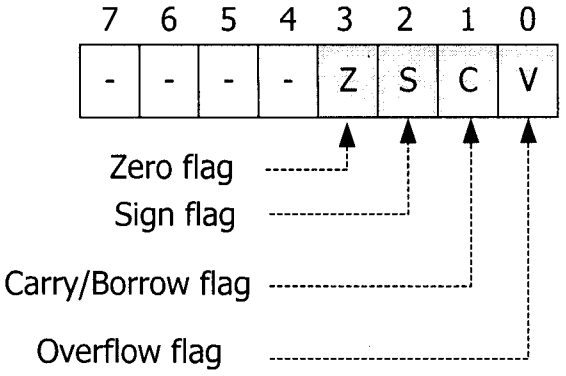
15. JZ nn instruction execution phase

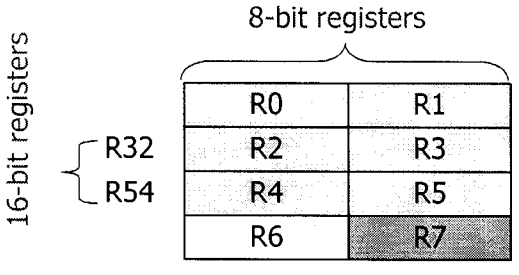
Step4 PC_OUT, MAR_IN, RD
Step5 INC_PC, MDR_IN_F_DATABUS, RD
Step6 MDR_OUT, TMP3_in
Step7 PC_OUT, MAR_IN, RD
Step8 INC_PC, MDR_IN_F_DATABUS, RD
Step9 MDR_OUT, TMP2_in
Step10 TMP2_out, TMP3_out, if Z=1 then PC_in = 1 else PC_in = 0, End

16. JC nn instruction execution phase

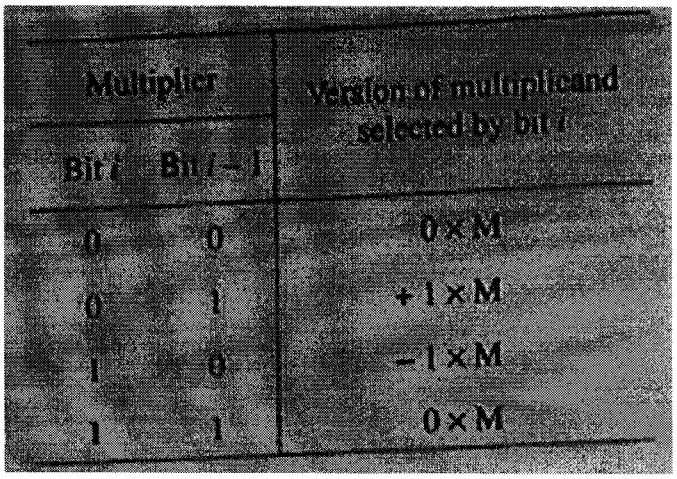
Step4 PC_OUT, MAR_IN, RD
Step5 INC_PC, MDR_IN_F_DATABUS, RD
Step6 MDR_OUT, TMP3_in
Step7 PC_OUT, MAR_IN, RD
Step8 INC_PC, MDR_IN_F_DATABUS, RD
Step9 MDR_OUT, TMP2_in
Step10 TMP2_out, TMP3_out, if Cy=1 then PC_in = 1 else PC_in = 0, End

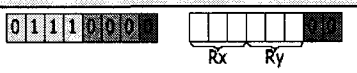
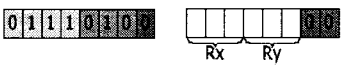
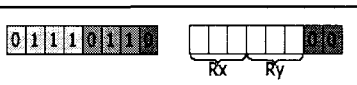
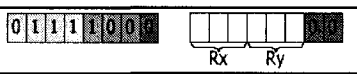
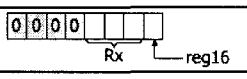
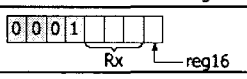
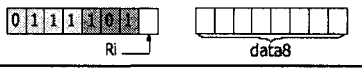
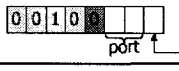
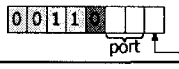
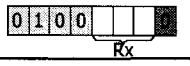
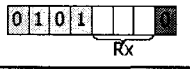
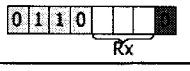
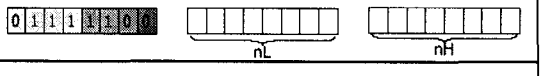
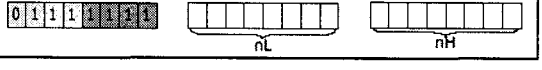






Booth multiplier recoding table



Instruction		Effected flag			Instruction length	Instruction format (1st-byte.....3rd-byte)
		C	Z	V		
ADDC	Rx, Ry	Y	Y	Y	2 bytes	
SUBB	Rx, Ry	Y	Y	Y	2 bytes	
AND	Rx, Ry	N	Y	N	2 bytes	
OR	Rx, Ry	N	Y	N	2 bytes	
LOAD	Rx, Ry	N	N	N	2 bytes	
LOAD	Rx, [reg16]	N	N	N	1 byte	
STORE	[reg16],Rx	N	N	N	1 byte	
LOAD	Ri, #data8	N	N	N	2 bytes	
IN	Ri, port	N	N	N	1 byte	
OUT	port, Ri	N	N	N	1 byte	
NOT	Rx	N	N	N	1 byte	
SHR	Rx	N	N	N	1 byte	
RL	Rx	N	N	N	1 byte	
JMP	nn	N	N	N	3 bytes	
JZ	nn	N	N	N	3 bytes	
JC	nn	N	N	N	3 bytes	

Note:

Rx, Ry = R0-R7
Ri = R0, R1
Port = P0, P1, P2, P3
Reg16 = R32, R54
Y = Yes
N = No