דוח מכין מעבדה 1

מעבדת ארכיטקטורת מעבדים מתקדמת ומאיצי חומרה

Liav Ben Or 315909390 Guy Cohen 207881004

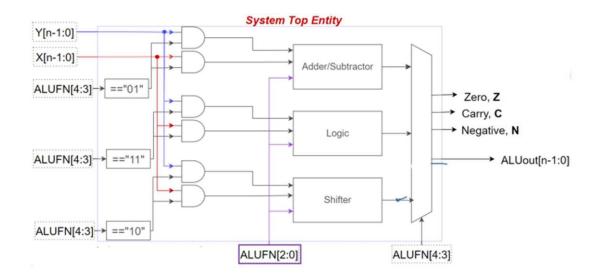
תוכן עניינים

2-3	
3-4	רכיב AdderSub
3-4	הסבר תיאורטי
4	תוצאות הסימולציה
5-6	רכיב Shifter
5-6	הסבר תיאורטי
6	תוצאות הסימולציה
7	Logic רכיב
7	הסבר תיאורטי
7	תוצאות סימולציה
8	רכיב Top
8	הסבר תיאורטי
8_9	חוטאות חימולטיה

מבוא

במעבדה זו נלמד יכולות בסיסיות בעולם החומרה המקבילית בשפת VHDL. במעבדה זו, נממש מערכת המכילה מספר רכיבים שונים, כאשר כל פעם מודול אחר מתבצע בנפרד לפי בחירת המשתמש.

:המערכת שנממש נמצאת באיור הבא



חלקי המערכת:

- אות כניסה X.
- אות כניסה Y.
- יקו בקרה ALUFN כאשר ביטים 3,4 קובעים את הרכיב הנבחר באופן הבא:
 - .AdderSub הוא רכיב 01 ס
 - .Shifter הוא רכיב 10 ס
 - .Logic הוא רכיב 11 ס

המודולים יכולים לבצע מספר פעולות שונות כתלות בשלושת הביטים לבצע מספר פעולות שונות כתלות בשלושת לבצע מספר פעולות בקוח (ביטים $0,\,1,\,2$).

מוצאי המערכת:

C (Carry), N (Negative) דגלי בקרה	•
(Overflow)V, Z(Zero),

ולח הרכיר יוצאח רALUOUT	מוצאה פעו
-------------------------	-----------

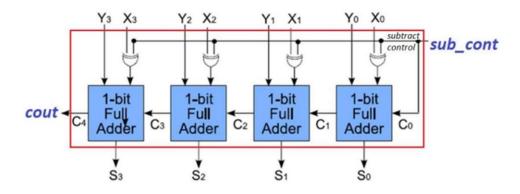
Function Decimal ALUFN Kind value										
Arithmetic	8	01000	Res=Y+X							
	9	01001	Res=Y-X	Used also for compare operation						
	10	01010	Res=neg(X)							
Shift	16	10000	Res=SHL Y,X(k-1 to 0)	Shift Left Y of $q \triangleq X(k-10)$ times Res=Y(n-1-q0)#(q@0) When $k = log_2 n$						
	17	10001	Res=SHR Y,X(k-1 to 0)	Shift Right Y of $q \triangleq X(k-10)$ times Res=(q@0)#Y(n-1q) When $k = log_2 n$						
Boolean	24	11000	Res=not(Y)							
	25	11001	Res=Y or X							
	26	11010	Res=Y and X							
	27	11011	Res=Y xor X							
	28	11100	Res=Y nor X							
	29	11101	Res=Y nand X							
	30	11111	Res=Y xnor X							

AdderSub מודול

תיאור

רכיב זה יופעל כאשר שני הביטים האחרונים ב ALUFN יהיו "01". ויבצע את אחת משלושת הכיב זה יופעל כאשר שני הביטים האחרונים ב ALUFN באופן הבא:

- Y ל X לבור בין אל "000" •
- X ל Y היסור בין 1001"
 - NEG(X) -"010" •

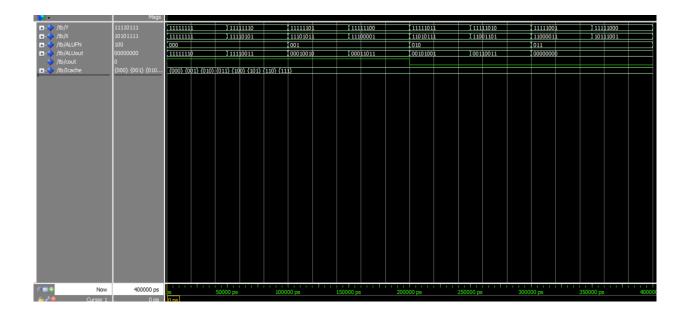


הרכיב. אחראי על בחירת אחת מתת פעולות SUB_CONT

מימוש בVHDL

- 1. הגדרנו את FA כרכיב
- .2 יצרנו סיגנלים עבור משתני עזר, למשל וקטור אפסים ואחדות
 - בסיס ביטי הבקרה SUB CONT אל מייצרים את 3.
 - את הוקטור X ע"פ ביטי הבקרה .4
 - הבקרה Y ע"פ ביטי הבקרה 5.
 - SUB_CONT בין X לבין XOR סיגנל המכיל תוצאת 6.
 - ראשון FA חיבור.7
 - FA חיבור יתר.8
 - 29. הוצאת COUT

תוצאות הסימולציה



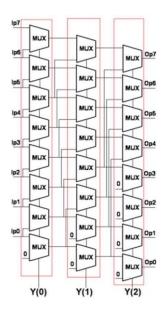
ps−. delt	a⊸v	/tb/Y-	/tb/X-y/tb/ALU		/ALUout-	₹
0	+5	11111111	11111111	000	11111110	1
50000	+5	11111110	11110101	000	11110011	1
100000	+4	11111101	11101011	001	00010010	1
150000	+4	11111100	11100001	001	00011011	1
200000	+6	11111011	11010111	010	00101001	0
250000	+4	11111010	11001101	010	00110011	0
300000	+4	11111001	11000011	011	00000000	0
350000	+1	11111000	10111001	011	00000000	0
400000	+1	11110111	10101111	100	00000000	0

Shifter מודול תיאור

רכיב זה מופעל בהתאם לכניבת ALUFN ומבצע פעולת הזזה המבוססת על ALUFN רכיב זה מופעל בהתאם לכניבת (200 ההזזה תתבצע ימינה או שמאלה, "001", "000" בהתאמה. ניכנס לרכיב זה כאשר שני הביטים השמאליים שך ALUFNיהיו "10".

מימוש

- 1. הגדרנו סיגנלים שימושיים כגון MATהשומרת את ההזזות במהלך תהליך זרימת המידע.
 - .2 איתחלנו בשורה הראשונה של המטריצה את המספר המקורי Y לפני ההזזות.
 - 3. בהתאם לX הביטים הראשונים של X ביצענו הזזות Y ושמרנו בשורות המטריצה.
- 4. חישוב הCARRYOUT שצריך לצאת והוצאת השורה האחרונה מהמטריצה כתוצאה הסופית.



הערה: כאשר ההזזה מוגדרת ימינה נבצע את שלב 2 בצורה הפוכה ולבסוף לאחר סיום שלב 5 נהפוך את התוצאה הסופית על מנת לקבל הזזות כרצוי.

תוצאות הסימולציה

- → /tb2/Y	11110111	11111111	(11111110	11111101	(11111100	11111011	(11111010	11111001	(11111000
<u>+</u> -♦ /tb2/X	10101111	11111111	(11110101	11101011	(11100001	11010111	(11001101	11000011	(10111001
+ /tb2/ALU		000		001		010		011	
/tb2/ALU		10000000	(11000000	00011111	(01111110	00000000	 		
/tb2/cou		10 (000) (001) (010)} {011} {100} {101} {110	(111)					
- /W2/1Cat	che {000} {001} {0:	10 {000} {001} {010)} {011} {100} {101} {1110	7 11117					
△ 🐷 💿	Now 400000	ps Committee of the com	50000	100000	150000	200000	250000 2	20000	350000 ps 4000
A 60		JS .	50000 ps	100000 ps	150000 ps	200000 ps	250000 ps 3	00000 ps	350000 ps 4000

ps delta		/tb2/X-/tb2 /tb2/ALUFN-	
0 +6	11111111	11111111 000	10000000 1
50000 +6	11111110	11110101 000	11000000 1
100000 +6	11111101	11101011 001	00011111 1
150000 +6	11111100	11100001 001	01111110 0
200000 +6	11111011	11010111 010	000000000
250000 +1	11111010	11001101 010	000000000
300000 +1	11111001	11000011 011	000000000
350000 +1	11111000	10111001 011	000000000
400000 +1	11110111	10101111 100	0 00000000

רכיב Logic תיאור

ניכנס לרכיב זה כאשר שני הביטים השמאליים שך ALUFN יהיו "11". רכיב זה מבצע פעולות לוגיות על וקטורים X ו Y ע"פ כניסת ALUFN:

- NOT(Y) -"000" •
- X OR Y -"001" •
- Y AND X -"010" •
- Y XOR X -"011" •
- Y NOR X -"100" •
- Y NAND X -"101" •
- Y XNOR X -"111" •

תוצאות סימולציה



ps-↓ delt		/tbl/Y¥	/tbl/X-y/		/ALUout⊸
0	+2	11111111	11111111	000	00000000
50000	+2	111111110	11110101	000	00000001
100000	+2	11111101	11101011	001	11111111
150000	+2	11111100	11100001	001	11111101
200000	+2	11111011	11010111	010	11010011
250000	+2	11111010	11001101	010	11001000
300000	+2	11111001	11000011	011	00111010
350000	+2	11111000	10111001	011	01000001
400000	+2	11110111	10101111	100	00000000

מודול Top

תיאור

רכיב זה הוא הרכיב המרכזי של המערכת המכיל את כל הרכיבים המתוארים מעלה כתת רכיבים המרכיבים אותו "השלם גדול מסך חלקיו!". הוא מקבל את הכניסות ALUFN ,Y ,X ויודע לנתב אותם לרכיב המתאים בהתאם. מימשנו כך שרכיבים שאינם בשימוש יהיו בנתק ורק הרכיב הדרוש יבצע את פעולתו, פלט המערכת ALUOUT יהיה בהתאם. כמו כן למערכת יש דגלי בקרה הנדלקים בהתאם לצורך ע"פ הגדרתם במשימה.

אופן המימוש

- .1 הגדרת כל תתי רכיבי המערכת כCOMPONENTS.
 - .2 יצירת סיגנלי עזר לחיבור בין רכיבי המערכת.
- 3. חילוץ שלושת הביטים הראשונים מו ALUFN וניתובם לרכיב הרלוונטי.
 - 4. חישוב דגלי הבקרה ע"פ הגדרתם במשימה.
 - לרכיב שהופעל. ALU_OUT בהתאם לרכיב שהופעל.

תוצאות הסימולציה

ps⊸ deli	ta-y	/tb3/Y¬	/tb3/ALUFN-, /tb3/Cflag-, /tb3/Nflag-, /tb3/Zflag-, /tb3/Vflag-,												
0	+8	111111111	11111111	01000	11111110	1	1	0	0						
50000	+7	111111110	11110101	01000	11110011	1	1	0	0						
100000	+9	111111101	11101011	01001	00010010	0	1	0	0						
150000	+6	11111100	11100001	01001	00011011	0	1	0	0						
200000	+8	11111011	11010111	01010	00101001	0	0	0	0						
250000	+6	11111010	11001101	01010		0	0	0	0						
300000	+10	111111001	11000011	01000	10111100	1	1	0	0						
350000	+8	11111000	10111001	01000	10110001	1	1	0	0						
400000	+9	11110111	10101111	01001	01001000	0	1	0	0						
450000	+7	11110110	10100101	01001	01010001	0	1	0	0						
500000	+9	11110101	10011011	01010	01100101	0	0	0	0						
550000	+6	11110100	10010001	01010	01101111	0	0	0	0						
600000	+10	11110011	10000111	01000	01111010	0	1	0	1						
650000	+9	11110010	01111101	01000	01101111	0	1	0	0						
700000	+13	11110001	01110011	01001	01111110	0	1	0	1						
750000	+9	11110000	01101001	01001	10000111	1	1	0	0						
800000	+6	11101111	01011111	10000	10000000	1	1	0	0						
850000	+8	11101110	01010101	10000	11000000	1	1	0	0						
900000	+8	11101101	01001011	10001	00011101	0	1	0	0						
950000	+7	11101100	01000001	10001	01110110	0	0	0	0						
1000000	+7	11101011	00110111	10010	00000000	0	0	1	0						
1050000	+1	11101010	00101101	10010	00000000	0	0	1	0						
1100000	+8	11101001	00100011	10000	01001000	0	1	0	0						
1150000	+8	11101000	00011001	10000	11010000	1	1	0	0						
1200000	+8	11100111	00001111	10001	00000001	0	1	0	0						
1250000	+6	11100110	00000101	10001	00000111	0	0	0	0						
1300000	+7	11100101	11111011	10010	00000000	0	0	1	0						
1350000	+1	11100100	11110001	10010	00000000	0	0	1	0						
1400000	+4	11100011	11100111	11001	11100111	1	0	0	0						
1450000	+4	11100010	11011101	11001	11111111	1	0	0	0						
1500000	+4	11100001	11010011	11010	11000001	1	0	0	0						
1550000	+4	11100000	11001001	11010	11000000	1	0	0	0						
1600000	+4	11011111	10111111	11101	01100000	0	0	0	0						
1650000	+4	11011110	10110101	11101	01101011	0	0	0	0						
1700000	+4	11011101	10101011	11111	10001001	1	0	0	0						
1750000	+4	11011100	10100001	11111	10000010	1	0	0	0						
1800000	+4	11011011	10010111	11011	01001100	0	0	0	0						
1850000	+4	11011010	10001101	11011	01010111	0	0	0	0						

► ♦ /tb3/Y	-No Data-	(1 Y1)	1 /1	1 11	1 11	1 Y1	1 Y1	1 11	V1 V1	1 11	1 Y1	1 11	1 1	11 11	1 11	1 /1	1 Y 1	11 Y1	11 Y1	11 Y1	1 Y1
- → /tb3/X	-No Data-	1 / 1	1 / 1 · · ·	1 / 1	1 /1	1 V 1			10 10						1 \ 1	1 / 1 · · ·	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	11 \ 1	1 / 1	1 \ 1
√ /tb3/ALUFN	-No Data-			01010	01000	101001	01010	01000			10001	10010	10000								00100
★ /tb3/ALUout	-No Data-	1 (1			01000	10 10			10 11												
/tb3/Nflag	-No Data-	1 / 1	U (U	(υ (υ	1 / 1	Ιυ χυ	Ιυ χυ	,	V (I	11 <u>/ 1</u>	(υ (υ	00000000	, U , I	(υ χυ	00000000	1 / ₁ 1	4 I / I	10 10	1 / 1	ν χυ	0000000
↓ /tb3/Cflag ↓ /tb3/Cfla	-No Data-				-						_								1		
↓ /tb3/Zflag ↓ /tb3/Zfla	-No Data-																				
	-No Data-							\vdash	=												
/b3/Vilag /tb3/Icache	-No Data-	(0.1000) (0.	(001) (0101	0) (01000)	01001) (01	10) (01000	(01001) (1	0000) (1000	1} {10010} {	(10000) (100	01) (10010)	(11001) (1	1010) (1110	0 (11110 (11011) (001	20)					
√ /b3/Y	-No Data-	{01000} {0	10015 (0101	D} {01000} (010015 (01	110} (01000	1 {U 1UU 1} {1	0000} {1000	113 { 100 10 } {	10000} {100	01) {10010}	{11001} {1	1010} {1110	15 (11111) (11011} {001	JU}	· · ·	V. V.	V. V.	V. V.	
√ /tb3/X	-No Data-	4	1 / 1	1 \ 1	1 1	1 \ 1	1 1	1 Yo	1 / 1	1 \ 1	1 1	10 10	1 1	1 (1	1 1	1 / 1	1 1	1 / 1	1 / 1	1 / 1	1 \ 1.
√ /tb3/ALUFN ✓ /tb3/ALUF	-No Data-	(01000	01001	01010	01000	101001			1					10001	10010	11001	11010	111101		11 <u>/</u> 1	
tb3/ALUout	-No Data-	1 (1				01001	01010														00100
	-No Data-	1 / 1	U (U	.0 (0	1 / 1	Ιυ χυ	, 0		0 1	1 (1	0 (0	00000000	0 1	, υ , υ	,00000000	1 /, 1	1 / 1	, 0	1 / 1	, U , U	,0000000
♦ /tb3/Nflag ♦ /tb3/Cflag	-No Data-										_		===						1		
	-No Data-																				
	-No Data-				-			\vdash	\vdash									_	_		1
	-No Data-	(21222) (2		2 (21222)	24224) (24	10) (01000	(0.1001) (1	1000) (1000	(10010)	(10000) (100	n.) (10010)	(11001) (1	0.10) (1.110	2 (11112) (11011) (001						
- /> /tb3/Icache	TVO Data-	{01000} {0	10015 (0101	n	010015 (01	010} {01000	5 (01001) (1	0000} {1000	115 { 100 105 }	10000} {100	015 (10010)	{11001} {1	1010} {1110	15 (111111) (110115 (001	JU}		-	-		
()	Now 2000000 ps			00 ps		00 ps		100 ps		00 ps		000 ps		000 ps	14000			000 ps		000 ps	