Preparation Report LAB4

Guy Cohen 207881004

Liav Ben Or 315909390

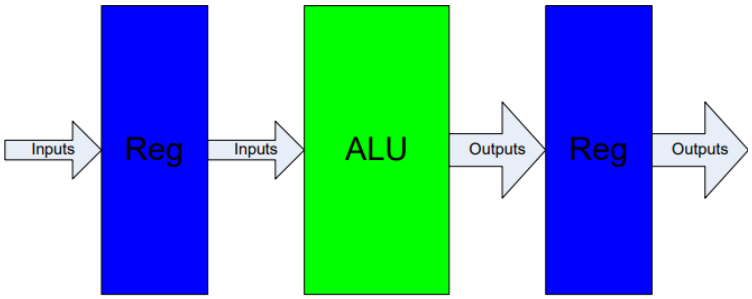
תוכן עניינים:

מטרת המעבדה

במעבדה זו למדנו להשתמש ביכולות של תוכנת Quartus ובפרט לבצע סינתזה עבור מודלים שפיתחנו בעבר במעבדה 1. את הסינתזה ביצענו על גבי FGPA.

בדיקת ביצועים

בדיקה זו נעשתה על מנת לבדוק ביצועים ראשוניים של הALU שפיתחנו במעבדה 1, ובנוסף בדיקה של התוספת של הוצאת גל ריבועי עבור קוד מסויים.
אך הפעם בשילוב של סינתזה על גבי כרטיס FPGA אמיתי. מכיוון שהמערכת שפיתחנו (ALU) הינה אסינכרונית, כלומר ללא שעון, נצטרך לחבר למערכת רגיסטרים סינכרוניים לכניסה ולמוצא המערכת על מנת לבצע אנליזה בזמן.



כאשר אל רגיסטרים אלו נחבר את שעון מהכרטיס בעל תדר של 50מגה הרץ.

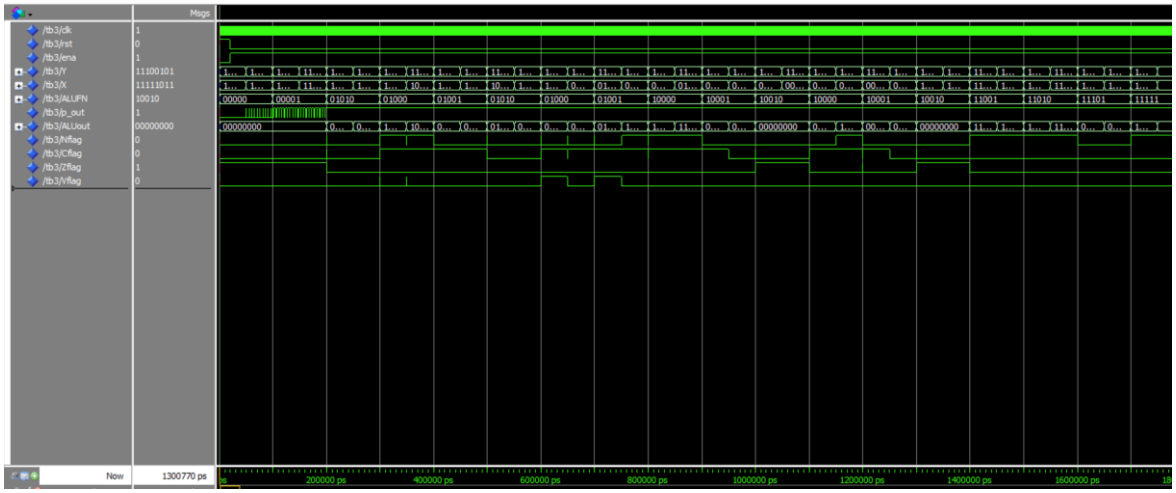
סימולצית ModelSim

בסימולציה זו תחילה נבצע TB ונראה את הגלים על גבי המערכת כולה.

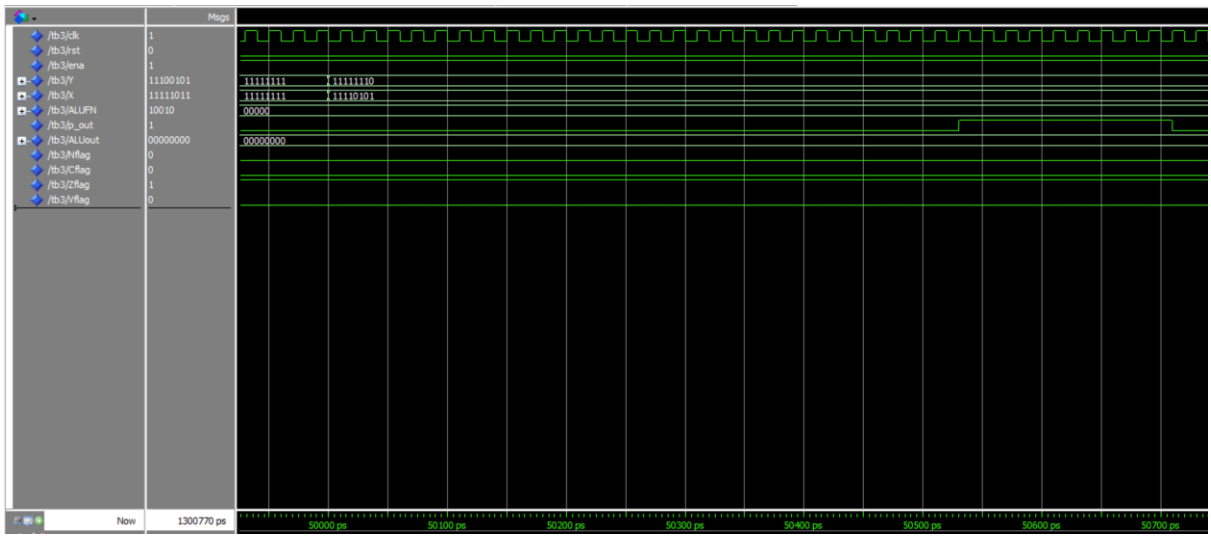
להלן כל הגלים עבור המערכת כולה, כאשר הTB שלנו מתחיל מפקודות של הוצאת גל ריבועי ולאחר מכן מריץ את שאר הפקודות המבוקשות. ערכי Y ו X משתנים במהלך הבדיקה.

על מנת לבדוק את המערכת יש לצרף את הקובץ top_sim.

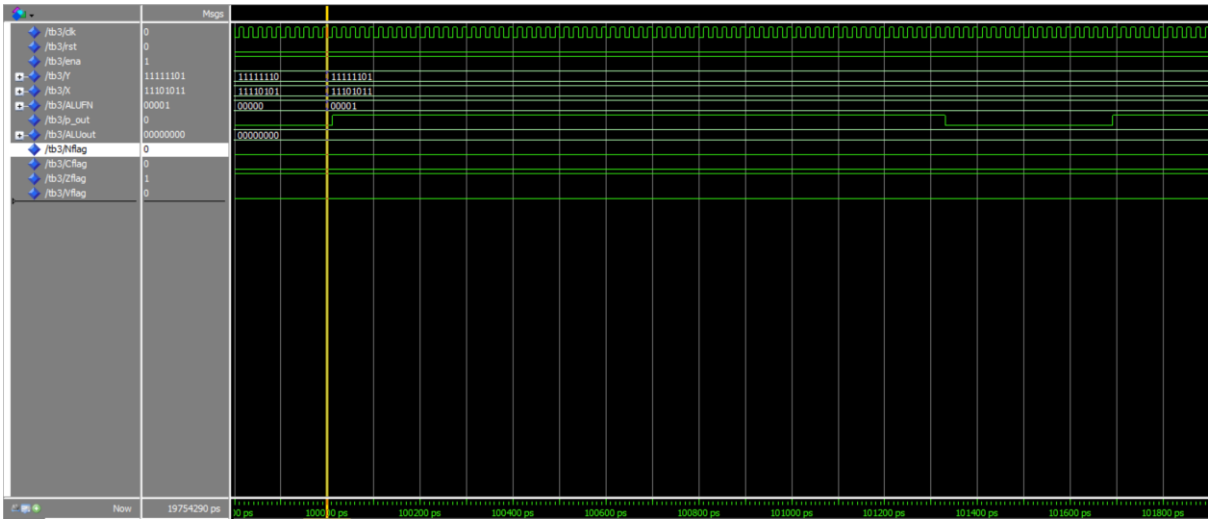
נריץ את הבדיקה במשך 1800000ps:



בתמונה מטה ניתן לראות את פעולת הגל הריבועי עבור ALUFN כשערכו הוא "00000" כאשר ערכו של X הוא "11110101" וערכו של Y הינו "11111110", נשים לב שהחיסור ביניהם יוצא 9 ולכן הגל הריבועי עולה לערך 1 לוגי למשך 9 מחזורי שעון:



בתמונה מטה ניתן לראות את פעולת הגל הריבועי עבור ALUFN בתמונה מטה ניתן לראות את פעולת הגל הריבועי עבור ALUFN כשערכו הוא "00001" כאשר ערכו של X הוא "11101011" וערכו של Y הינו "11111101", נשים לב שהחיסור ביניהם יוצא 18 ולכן הגל הריבועי יורד לערך 0 לוגי למשך 18 מחזורי שעון:



מציאת תדר מקסימלי

בכדי למצוא תדר מקסימלי של המערכת שלנו, כאמור נצטרך להוסיף רגיסטרים בכניסה ובמוצא הALU, לכן יצרנו קובץ VHDL חדש שעוטף את מערכת הALU עם הרגיסטרים שמוזנים מאותו השעון. נוסיף את הקובץ הבא:

```
# Constrain clock port clk with a 20-ns requirement

create_clock -period 20 [get_ports clk]

# Automatically apply a generate clock on the output of phase-locked loops (PLLs)
# This command can be safely left in the SDC even if no PLLs exist in the design

derive_pll_clocks
```

התוכנה יודעת לחשב את הערך המקסימלי, במערכת שלנו התדר המקסימלי הוא:

	Fmax	Restricted Fmax	Clock Name	Note
1	371.75 MHz	371.75 MHz	clk_di...divclk	

פירוט המערכת

בסעיף זה נסביר על המערכת שלנו ככלל ועל תתי המודולים שלה בפרט. עבור כל אחד ניתן סקירה קצרה על אופן פעילותו, נציג את הRTL שלו לאחר ביצוע הסינתזה, נפרט את הלוגיקה בה הוא משתמש ונמצא נתיב קריטי למערכת.

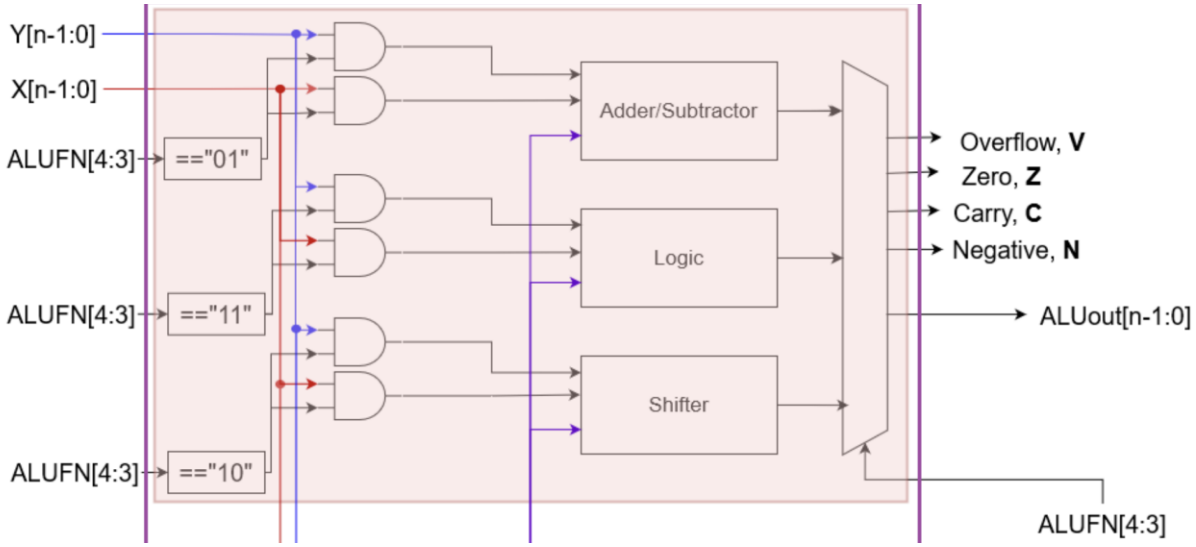
מודול ALU

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 X, Y, ALUFN 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 מ ALUFN_i וניתובם לרכיב הרלוונטי.
4. חישוב דגלי הבקרה ע"פ הגדרתם במשימה.
5. הוצאת ALU_OUT בהתאם לרכיב שהופעל.

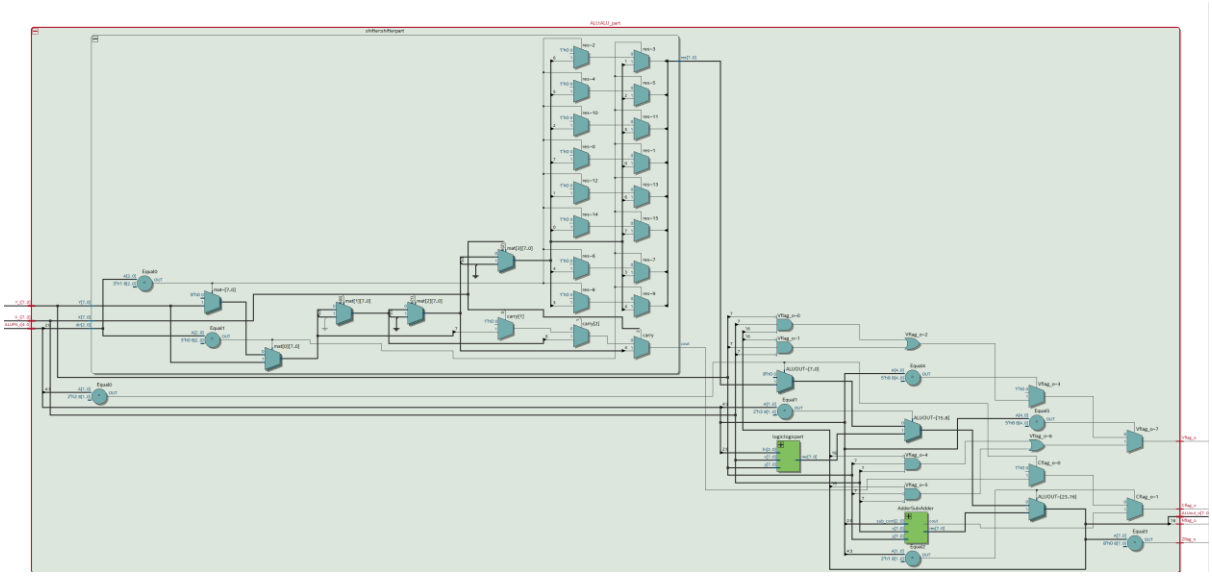


מוצאי המערכת:

- דגלי בקרה C (Carry), N (Negative), Z (Zero), V (Overflow)
- תוצאת פעולת הרכיב יוצאת בALUOUT

Function Kind	Decimal value	ALUFN	Operation	Note
Arithmetic	8	**01**000	Res=Y+X	
	9	**01**001	Res=Y-X	Used also for compare operation
	10	**01**010	Res=neg(X)	
Shift	16	**10**000	Res=SHL Y,X(k-1 to 0)	Shift Left Y of q≜X(k-1 ..0) times Res=Y(n-1-q…0)#(q@0) ***When*** $k = log_2 n$
	17	**10**001	Res=SHR Y,X(k-1 to 0)	Shift Right Y of q≜X(k-1 ..0) times Res=(q@0)#Y(n-1...q) ***When*** $k = log_2 n$
Boolean	24	**11**000	Res=not(Y)	
	25	**11**001	Res=Y or X	
	26	**11**010	Res=Y and X	
	27	**11**011	Res=Y xor X	
	28	**11**100	Res=Y nor X	
	29	**11**101	Res=Y nand X	
	30	**11**111	Res=Y xnor X	

שרטוט הRTL של מודול זה –

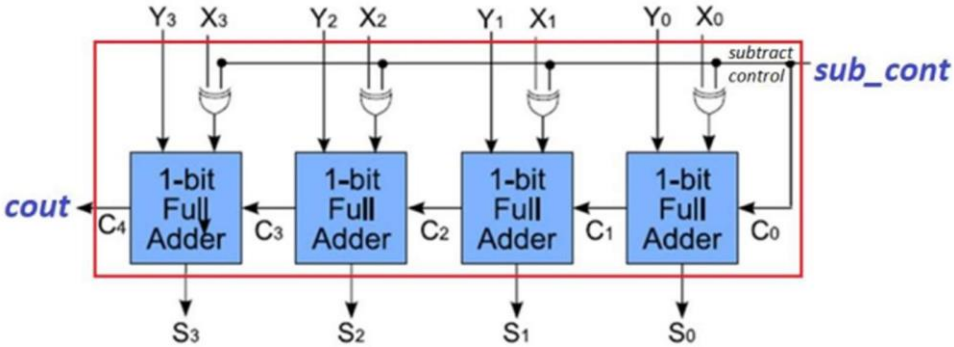


מודול AdderSub

תיאור

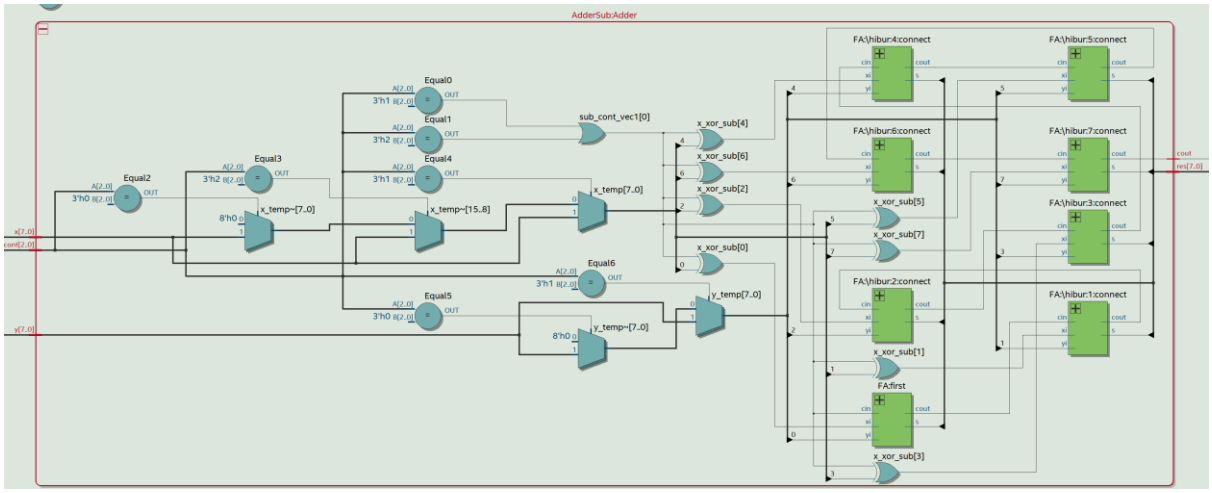
רכיב זה יופעל כאשר שני הביטים האחרונים ב ALUFN יהיו "01". ויבצע את אחת משלושת הפעולות הבאות בהתאם לכניסת ALUFN באופן הבא:

- "000"- חיבור בין X ל Y
- "001"- חיסור בין Y ל X
- "010"- NEG(X)



SUB_CONT אחראי על בחירת אחת מתת פעולות הרכיב.

RTL:



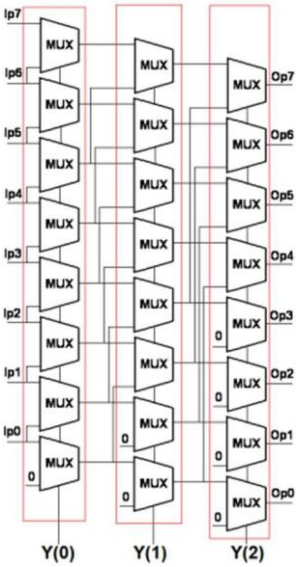
מודול Shifter

תיאור

רכיב זה מופעל בהתאם לכניבת ALUFN ומבצע פעולת הזזה המבוססת על BARREL SHIFTER כאשר ביטים 0,1,2 מחליטים אם ההזזה תתבצע ימינה או שמאלה, "001", "000" בהתאמה. ניכנס לרכיב זה כאשר שני הביטים השמאליים שך ALUFNיהיו "10".

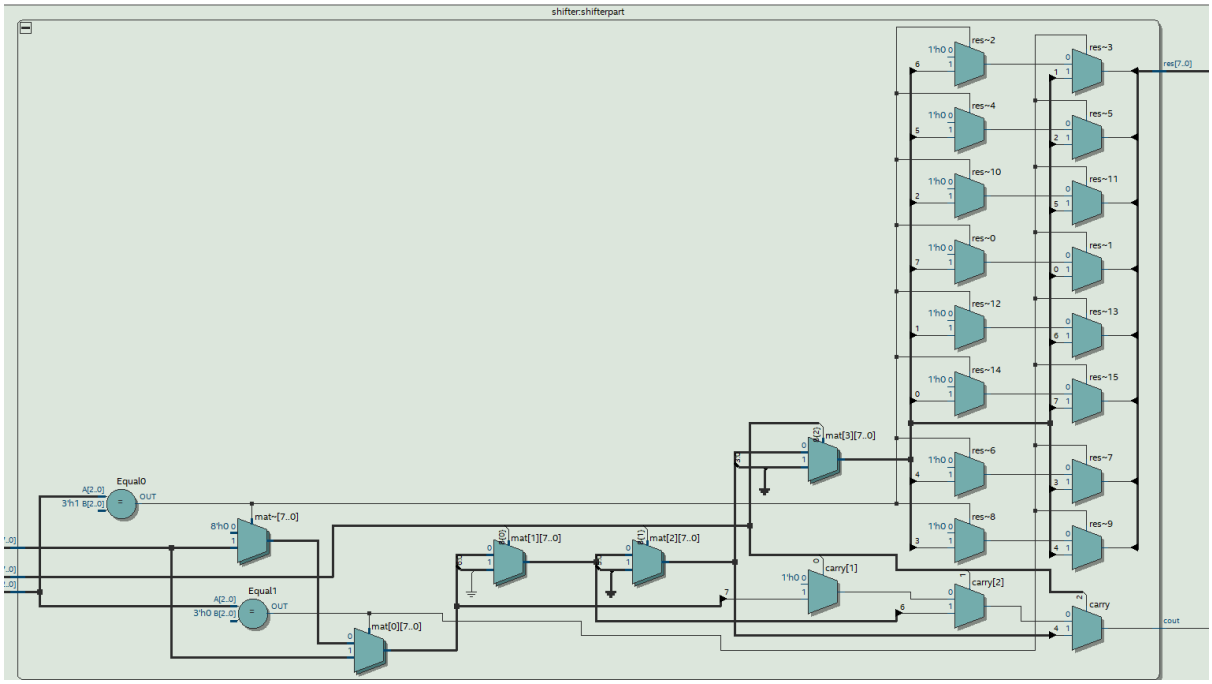
מימוש

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 לK 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.

:RTL



רכיב Logic

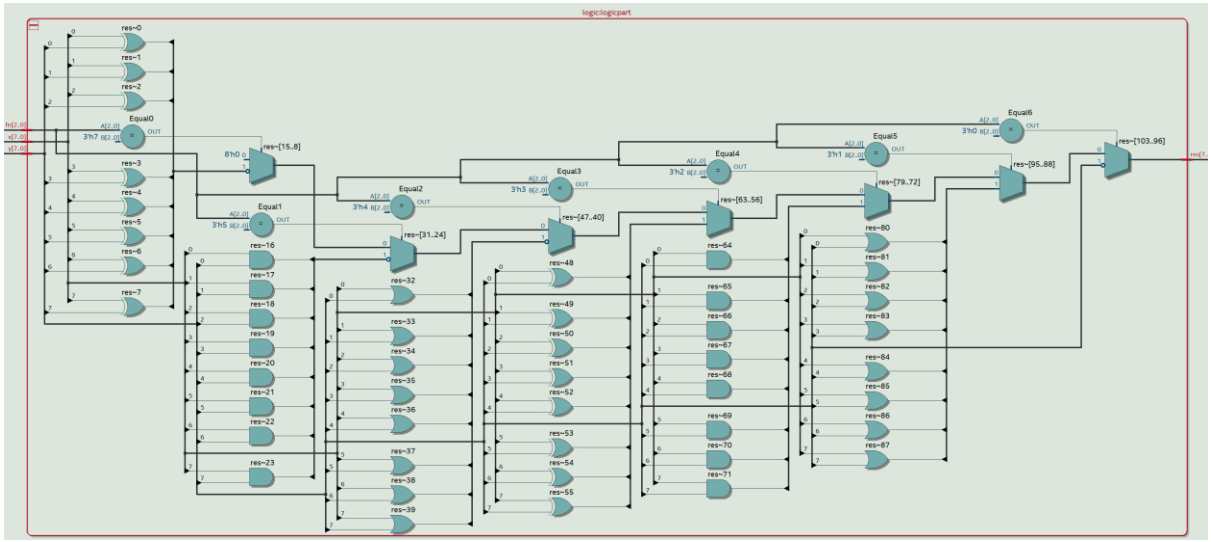
תיאור

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:

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

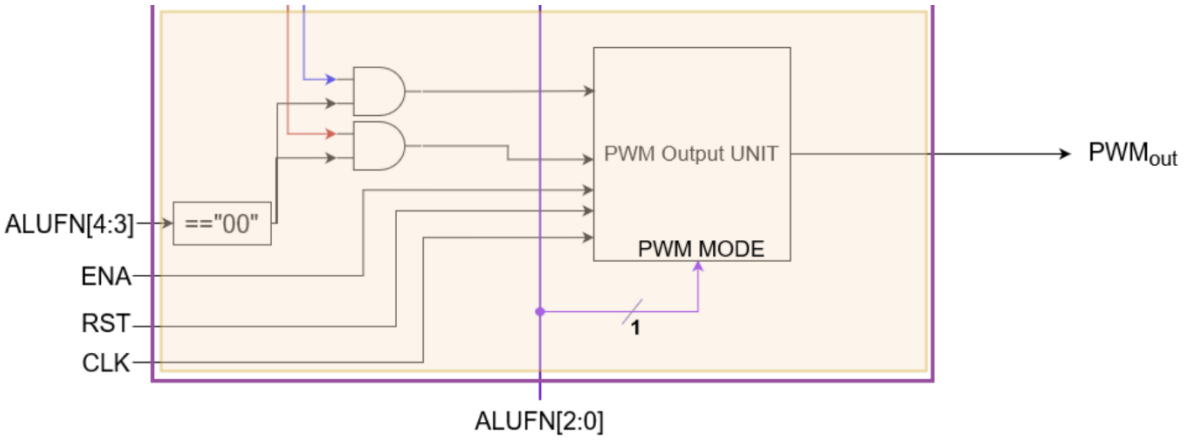
RTL:



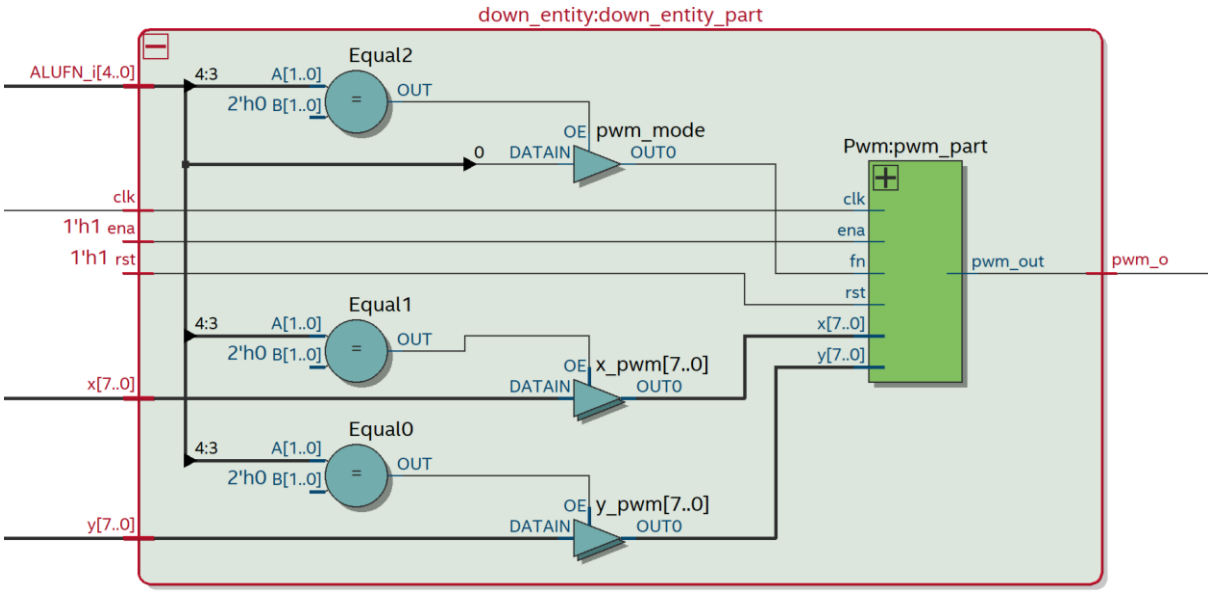
מודול down_entity

תיאור

רכיב זה הוא רכיב המכיל את רכיב ה PWM. הוא מקבל את הכניסות X, Y, ALUFN ויודע לנתב אותם לרכיב המתאים בהתאם.
מימשנו כך שרכיב ה pwm יהיה בנתק כאשר הפקודה בכניסה שונה מהפקודה המפעילה אותו, פלט המערכת pwm_out יהיה בהתאם.



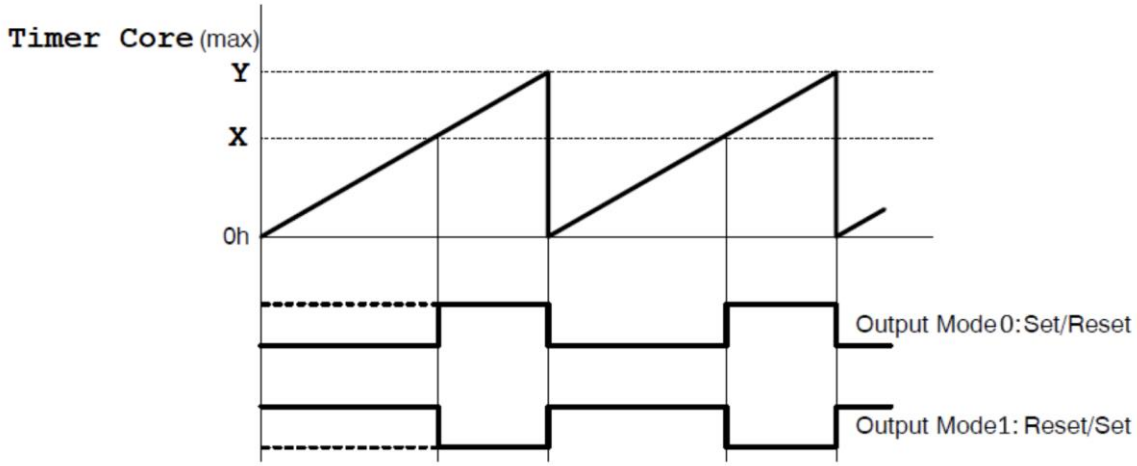
RTL:



מודול PWM

תיאור

רכיב זה מופעל בהתאם לכניסת ALUFN ומבצע פעולת גל ריבועי עולה או יורד לפי הקלט של המערכת באופן הבא:

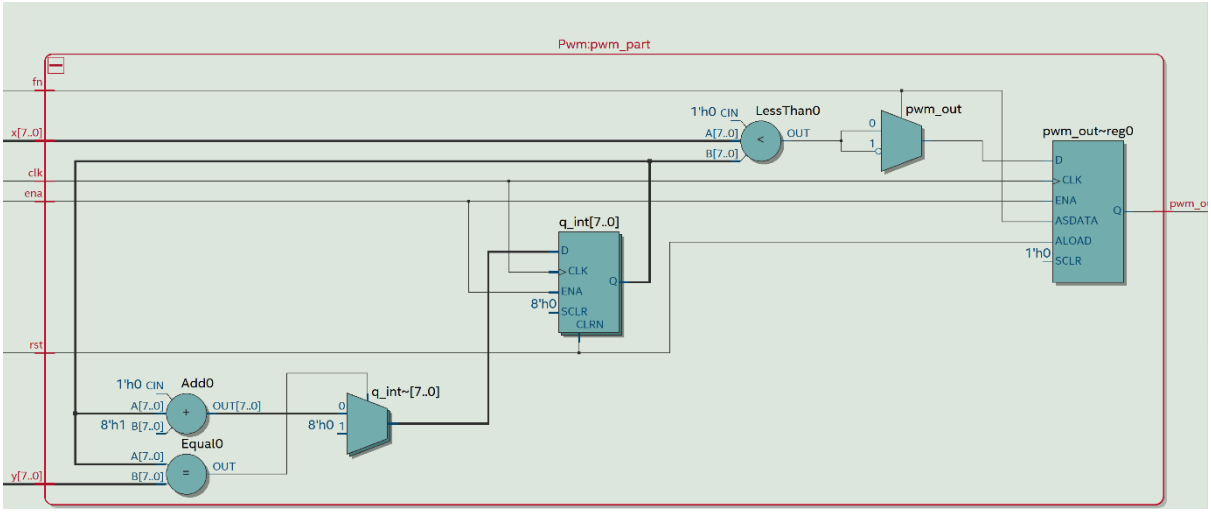


מימוש

1. הגדרנו סיגנל הסופר את מספר עליות השעון בפרוסס.
2. בדקנו בתנאים האם הסיגנל קטן מX או גדול ממנו.
3. במידה וגדול ממנו להוציא 0 או 1 לוגי, תלוי בMODE של המערכת.

PWM Output	0	00000	PWM MODE0	PWM Mode is Set/Reset
	1	00001	PWM MODE1	PWM Mode is Reset/Set

:RTL

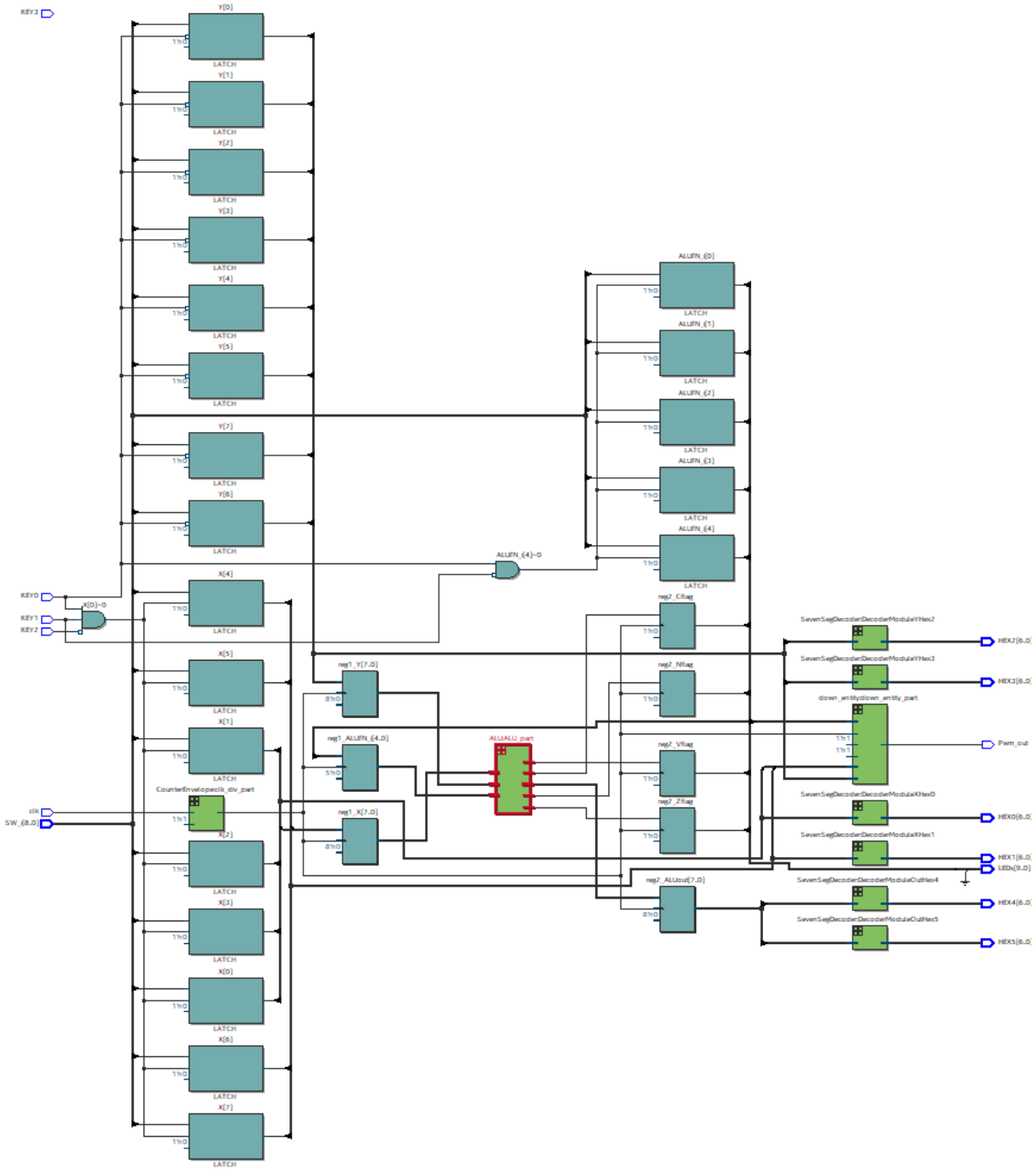


מודול TOP

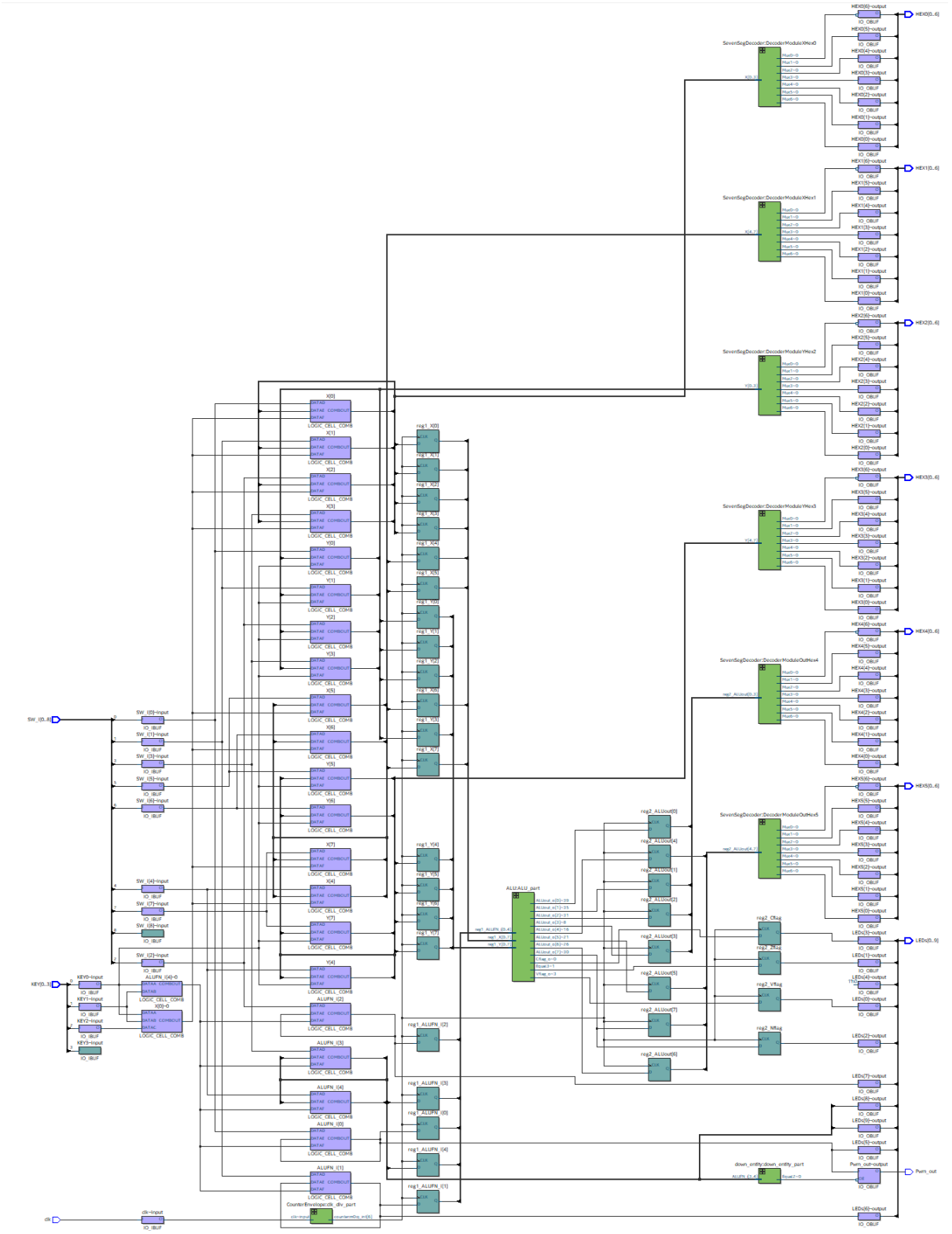
תיאור

המעטפת של המערכת אשר מכילה את ALU ואת ENTITY_DOWN, את המעטפת חיברנו לפינים של שבב הFPGA ע"י ה PLANNER_PIN.

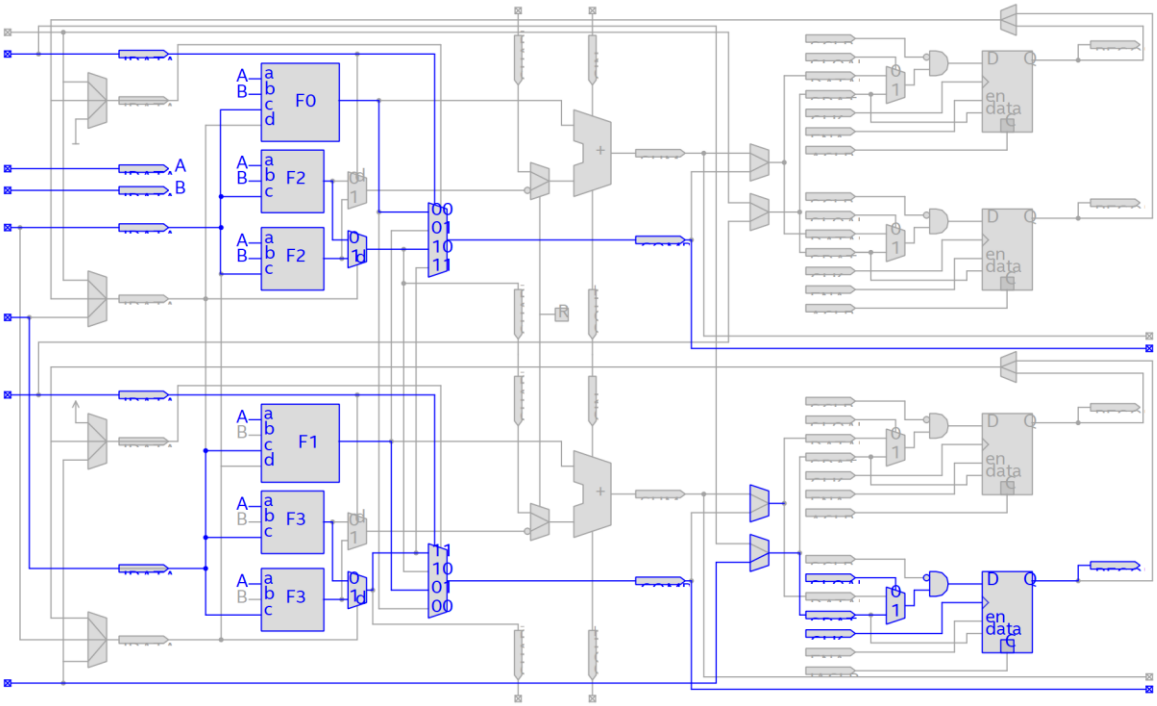
RTL:



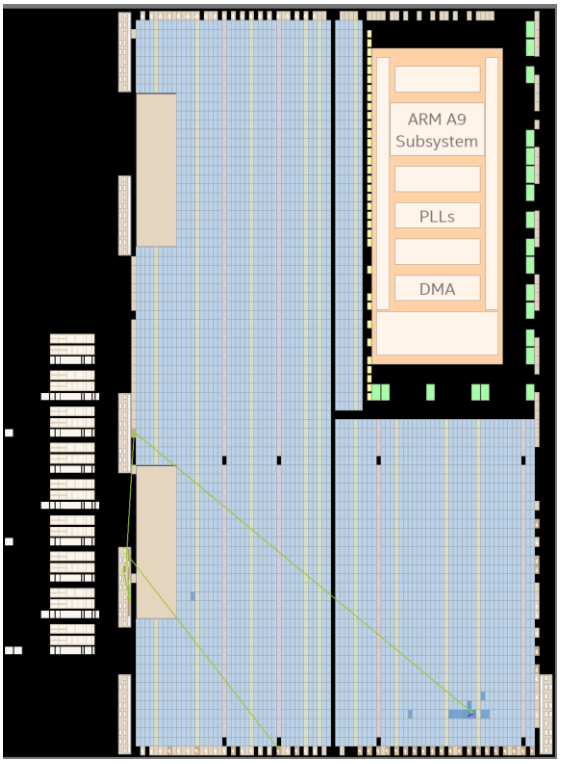
Map planner:



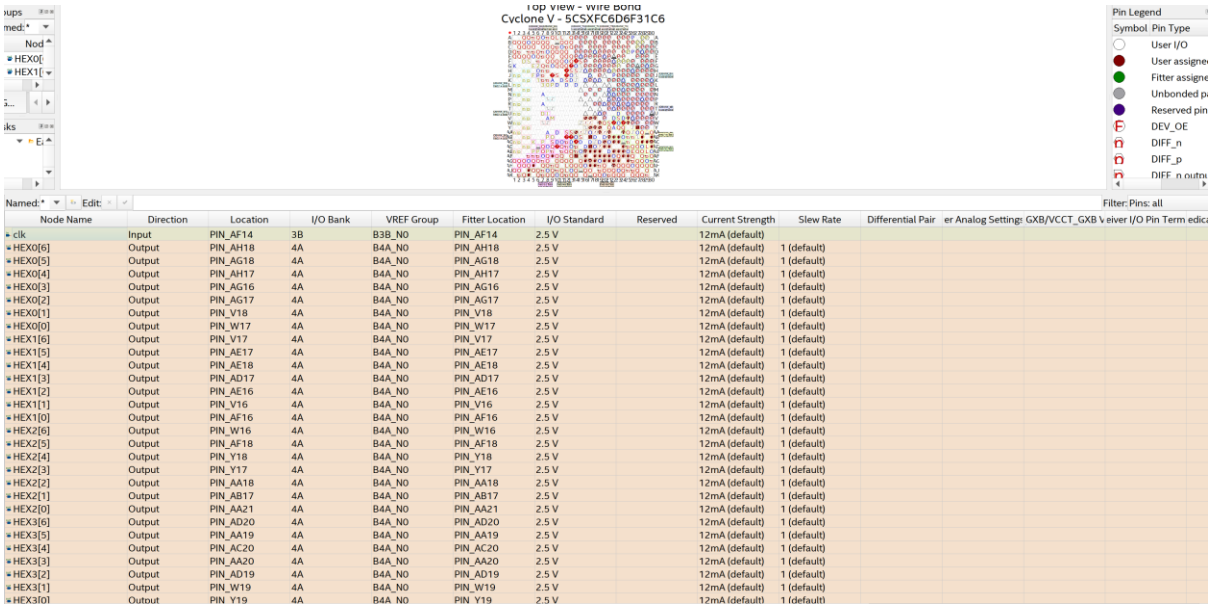
נתיב קריטי:



צריבה על גבי הכרטיס:



חיווטים לממשק הכרטיס:



Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate	Differential Pair	er Analog Settings	GXB/VCCT_GXB	eiver I/O Pin Term	edica
clk	Input	PIN_AF14	3B	B3B_N0	PIN_AF14	2.5 V		12mA (default)						
HEX0[6]	Output	PIN_AH18	4A	B4A_N0	PIN_AH18	2.5 V		12mA (default)	1 (default)					
HEX0[5]	Output	PIN_AG18	4A	B4A_N0	PIN_AG18	2.5 V		12mA (default)	1 (default)					
HEX0[4]	Output	PIN_AH17	4A	B4A_N0	PIN_AH17	2.5 V		12mA (default)	1 (default)					
HEX0[3]	Output	PIN_AG16	4A	B4A_N0	PIN_AG16	2.5 V		12mA (default)	1 (default)					
HEX0[2]	Output	PIN_AG17	4A	B4A_N0	PIN_AG17	2.5 V		12mA (default)	1 (default)					
HEX0[1]	Output	PIN_V18	4A	B4A_N0	PIN_V18	2.5 V		12mA (default)	1 (default)					
HEX0[0]	Output	PIN_W17	4A	B4A_N0	PIN_W17	2.5 V		12mA (default)	1 (default)					
HEX1[6]	Output	PIN_V17	4A	B4A_N0	PIN_V17	2.5 V		12mA (default)	1 (default)					
HEX1[5]	Output	PIN_AE17	4A	B4A_N0	PIN_AE17	2.5 V		12mA (default)	1 (default)					
HEX1[4]	Output	PIN_AE18	4A	B4A_N0	PIN_AE18	2.5 V		12mA (default)	1 (default)					
HEX1[3]	Output	PIN_AD17	4A	B4A_N0	PIN_AD17	2.5 V		12mA (default)	1 (default)					
HEX1[2]	Output	PIN_AE16	4A	B4A_N0	PIN_AE16	2.5 V		12mA (default)	1 (default)					
HEX1[1]	Output	PIN_V16	4A	B4A_N0	PIN_V16	2.5 V		12mA (default)	1 (default)					
HEX1[0]	Output	PIN_AF16	4A	B4A_N0	PIN_AF16	2.5 V		12mA (default)	1 (default)					
HEX2[6]	Output	PIN_W16	4A	B4A_N0	PIN_W16	2.5 V		12mA (default)	1 (default)					
HEX2[5]	Output	PIN_AF18	4A	B4A_N0	PIN_AF18	2.5 V		12mA (default)	1 (default)					
HEX2[4]	Output	PIN_Y18	4A	B4A_N0	PIN_Y18	2.5 V		12mA (default)	1 (default)					
HEX2[3]	Output	PIN_Y17	4A	B4A_N0	PIN_Y17	2.5 V		12mA (default)	1 (default)					
HEX2[2]	Output	PIN_AA18	4A	B4A_N0	PIN_AA18	2.5 V		12mA (default)	1 (default)					
HEX2[1]	Output	PIN_AB17	4A	B4A_N0	PIN_AB17	2.5 V		12mA (default)	1 (default)					
HEX2[0]	Output	PIN_AA21	4A	B4A_N0	PIN_AA21	2.5 V		12mA (default)	1 (default)					
HEX3[6]	Output	PIN_AD20	4A	B4A_N0	PIN_AD20	2.5 V		12mA (default)	1 (default)					
HEX3[5]	Output	PIN_AA19	4A	B4A_N0	PIN_AA19	2.5 V		12mA (default)	1 (default)					
HEX3[4]	Output	PIN_AC20	4A	B4A_N0	PIN_AC20	2.5 V		12mA (default)	1 (default)					
HEX3[3]	Output	PIN_AA20	4A	B4A_N0	PIN_AA20	2.5 V		12mA (default)	1 (default)					
HEX3[2]	Output	PIN_AD19	4A	B4A_N0	PIN_AD19	2.5 V		12mA (default)	1 (default)					
HEX3[1]	Output	PIN_W19	4A	B4A_N0	PIN_W19	2.5 V		12mA (default)	1 (default)					
HEX3[0]	Output	PIN_Y19	4A	B4A_N0	PIN_Y19	2.5 V		12mA (default)	1 (default)					

מודול ENVELOPE_COUNTER

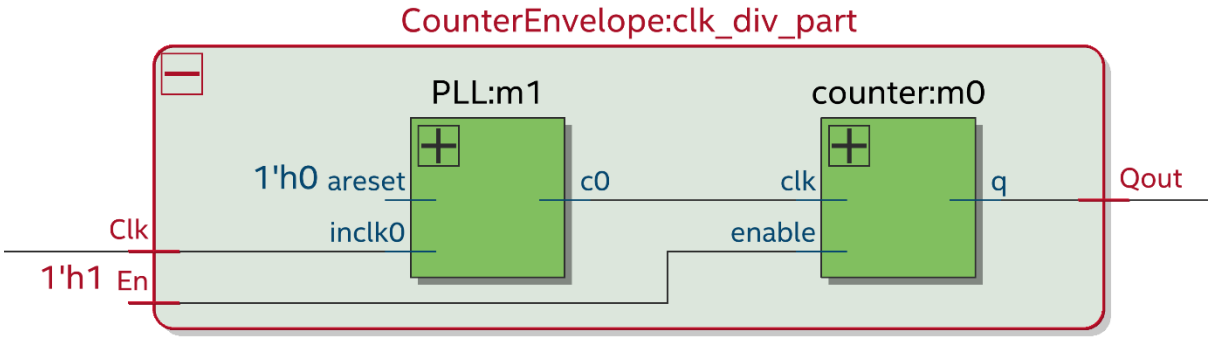
תיאור

רכיב זה מורכב משני תתי רכיבים:

- PLL
- COUNTER

מטרתם לקחת את תדר השעון המובנה של הבקר ולהקטינו באופן הבא:

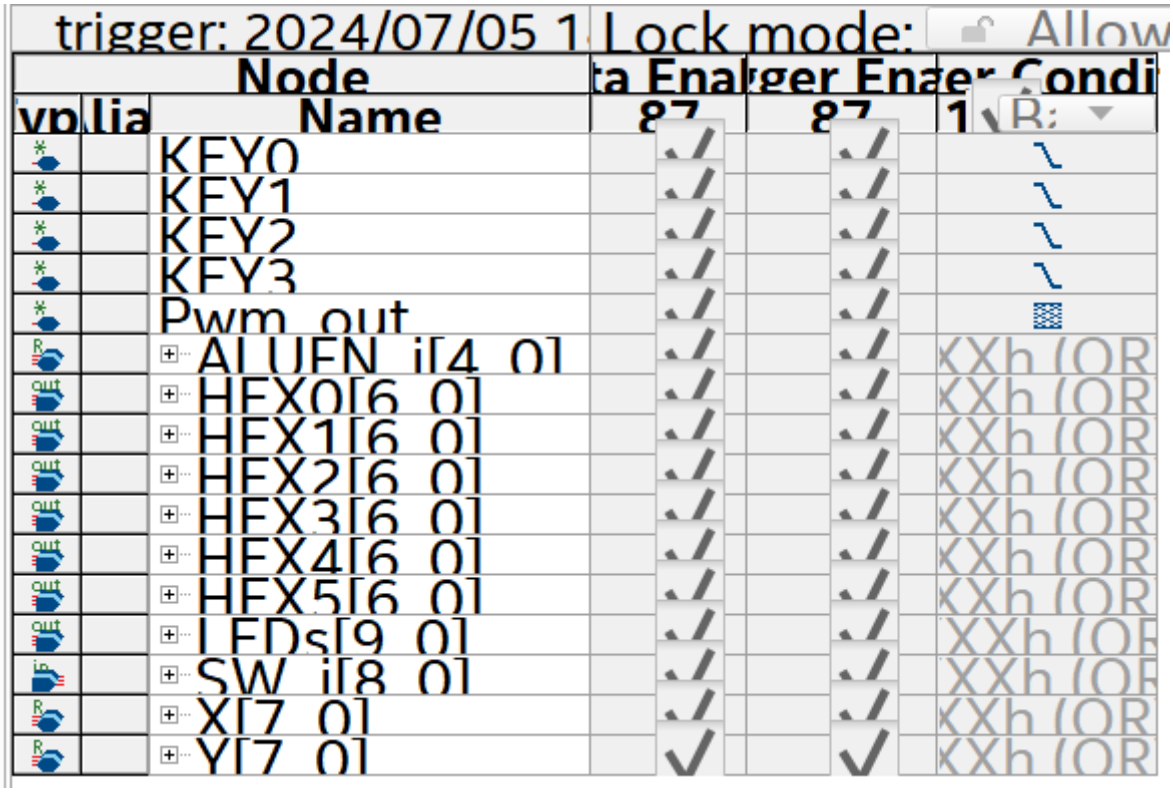
RTL:



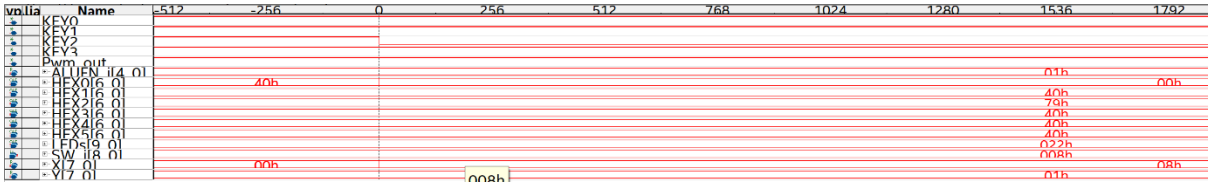
Signal Tap

נרצה לבצע ווריפיקציה של החומרה על ידי פונקציית ה-*signal tap* של ה-*quartus*. נתפוס בזמן אמת את מצב הסיגנלים של הרכיב, ובהתאם לסיגנל שאותו נרצה לתפוס, ברגע שהסיגנל ישתנה למה שאנחנו רוצים נקבל את תוצאות הסיגנלים שנדפיס למסך.

נשים את הסיגנלים הבאים, כאשר הסיגנלים שאנחנו לוכדים הם ה-*Keys* שהם במצב *Pull Down* לכן נתפוס אותם בירידת מתח, ונדפיס את הכניסות והמוצאים של המערכת. בנוסף נשים את תנאי הלכידה כ BASIC OR.

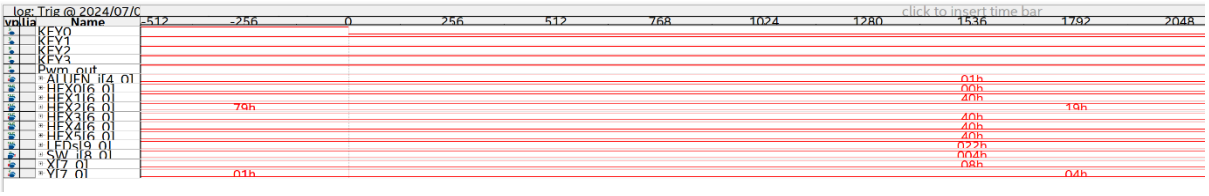


תחילה עבור שינוי הערך של *X*: *(לחיצה על 2KEY)*



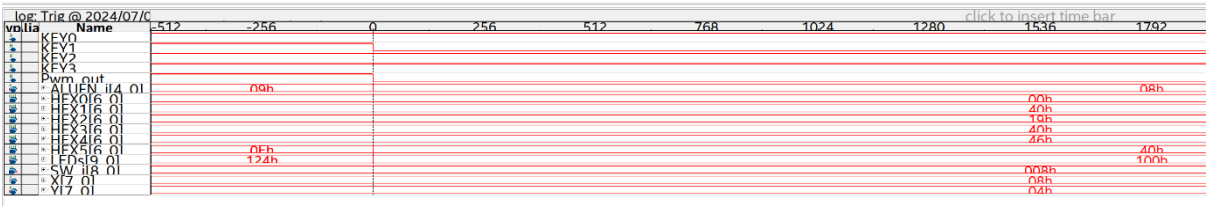
ניתן לראות כי הערך שלו השתנה מ 00 ל08.

עבור שינוי הערך של Y: *(לחיצה על 0KEY)*

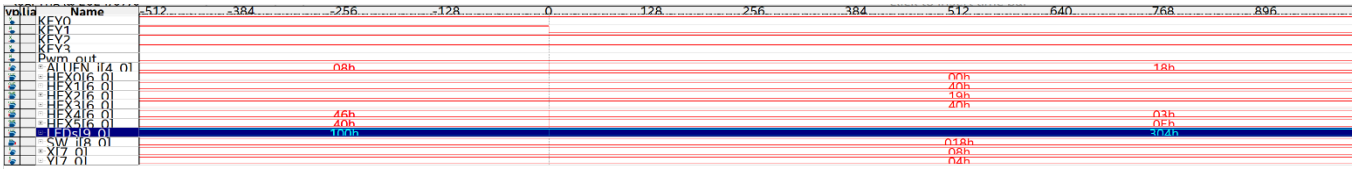


ניתן לראות כי הערך שלו השתנה מ 01 ל04.

עבור חיבור של X ו Y : *(לחיצה על 1KEY)*



עבור NEG Y : *(לחיצה על 1KEY)*



עבור SHIFT OPERATION : *(לחיצה על 1KEY)*