



אוניברסיטת בן גוריון בנגב

בית הספר להנדסת חשמל ומחשבים

קורס: מעבדת ארכיטקטורת מעבדים מתקדמת ומאיצי חומרה 361.1.4693

Final Project – MIPS based MCU Architecture Preparation Report

208849042

איל טראב

מגישים:

315150987

נעם מגדל

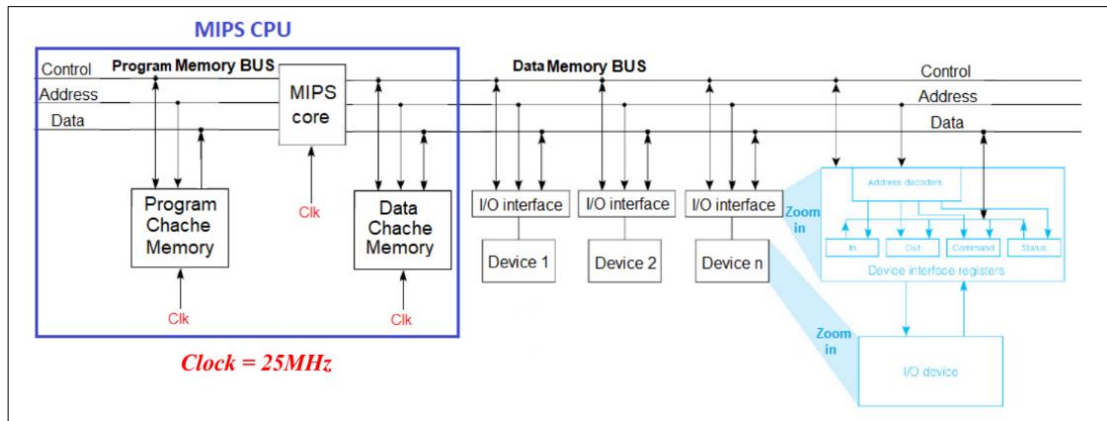
תאריך הגשה: 04.09.24

תוכן עניינים

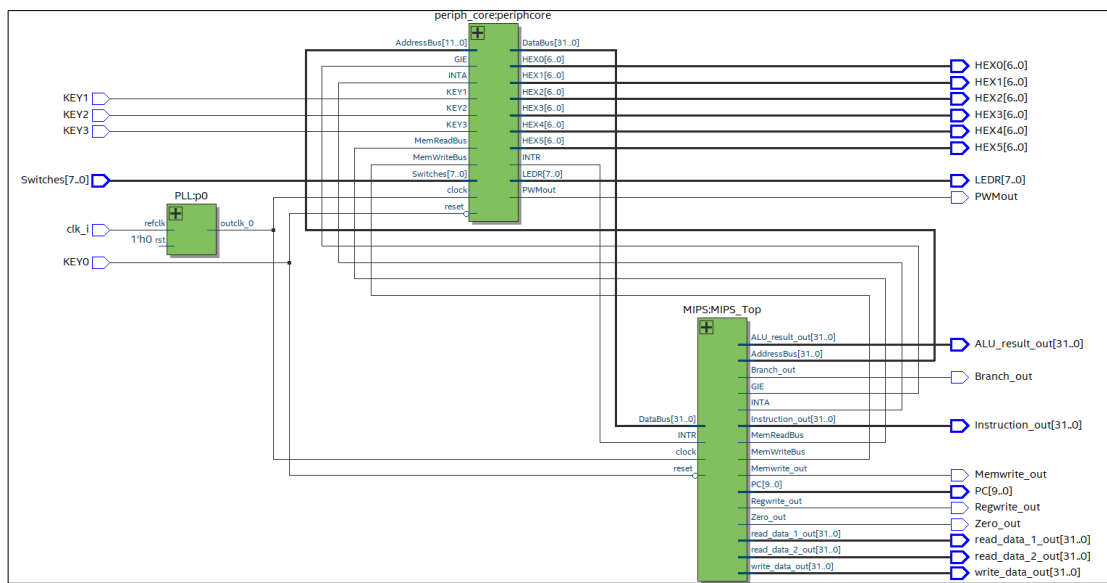
3	תיאור המעבדה	.1
5	MIPS	.2
7	GPIO	.3
8	Basic Timer	.4
10	Division accelerator	.5
12	Interrupt Controller	.6
14	נתיב קריטי	.7
16	ניתוח תוצאות	.8

תיאור המעבדה -

מטרת הפרויקט היא לעצב מיקרו-בקר (MCU) המבוסס על מעבד MIPS. המעבד ישתמש בארכיטקטורת Single Cycle MIPS ויבצע את כל קבוצת ההוראות של MIPS. ארכיטקטורת MIPS היא ארכיטקטורה מסוג הארוורד (Harvard). יחד עם המעבד נרצה להוסיף פריפריות חומרה נוספות שיעבדו בשיתוף פעולה בניהם בהתאם לצורך. התקשורת בין רכיבי החומרה השונים תהיה באמצעות 3 קווי BUS שיעבירו מידע רלוונטי בין הצרכים לצרכנים.



איור 1: שרטוט מערכת הMCU



איור 2: שרטוט RTL של מערכת הMCU

Analysis & Synthesis Resource Usage Summary		
<<Filter>>		
	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	1875
2		
3	▼ Combinational ALUT usage for logic	2259
1	-- 7 input functions	68
2	-- 6 input functions	1096
3	-- 5 input functions	330
4	-- 4 input functions	393
5	-- <=3 input functions	372
4		
5	Dedicated logic registers	1841
6		
7	I/O pins	238
8	Total MLAB memory bits	0
9	Total block memory bits	65536
10		
11	Total DSP Blocks	2
12		
13	▼ Total PLLs	1

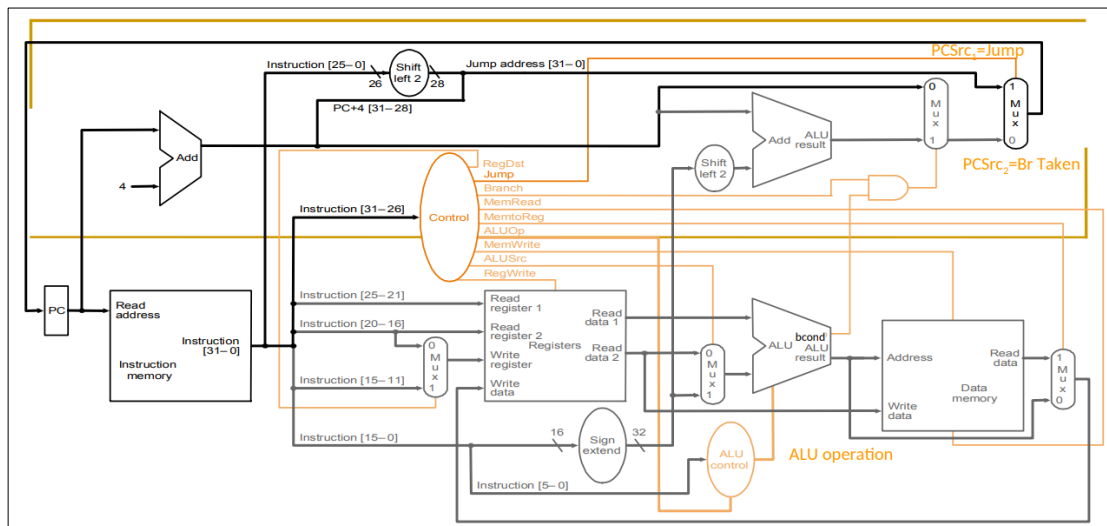
איור 3: שימוש בקומבינטוריקה לוגית עבור מודל זה

Analysis & Synthesis Resource Utilization by Entity		
<<Filter>>		
Compilation Hierarchy Node	Combinational ALUTs	Dedicated
project_Top	2259 (245)	1841 (0)
▼ MIPS:MIPS_Top	1509 (54)	1153 (13)
Execute:EXE	464 (464)	0 (0)
Idecode:ID	851 (851)	1000 (1000)
▼ Ifetch:IFE	79 (29)	74 (8)
▼ altsyncram:inst_memory	50 (0)	66 (0)
▼ altsyncram_t264:auto_generated	50 (0)	66 (0)
altsyncram_pc43:altsyncram1	0 (0)	0 (0)
▼ sld_mod_ram_rom:mgl_prim2	50 (32)	66 (57)
sld_rom_sr:\ram...gen:info_rom_sr	18 (18)	9 (9)
control:CTL	12 (12)	0 (0)
▼ dmemory:MEM	49 (0)	66 (0)
▼ altsyncram:data_memory	49 (0)	66 (0)
▼ altsyncram_1684:auto_generated	49 (0)	66 (0)
altsyncram_al53:altsyncram1	0 (0)	0 (0)

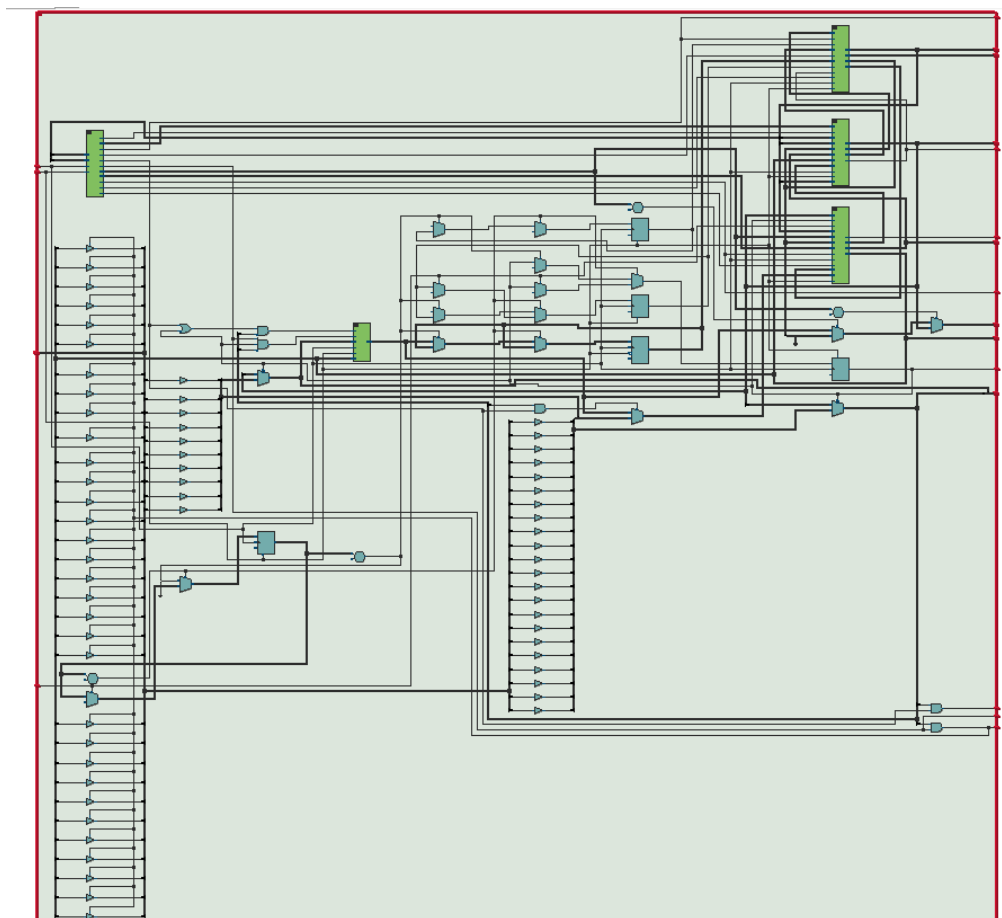
איור 4: שימוש בקומבינטוריקה עבור יתר המודלים

- MIPS

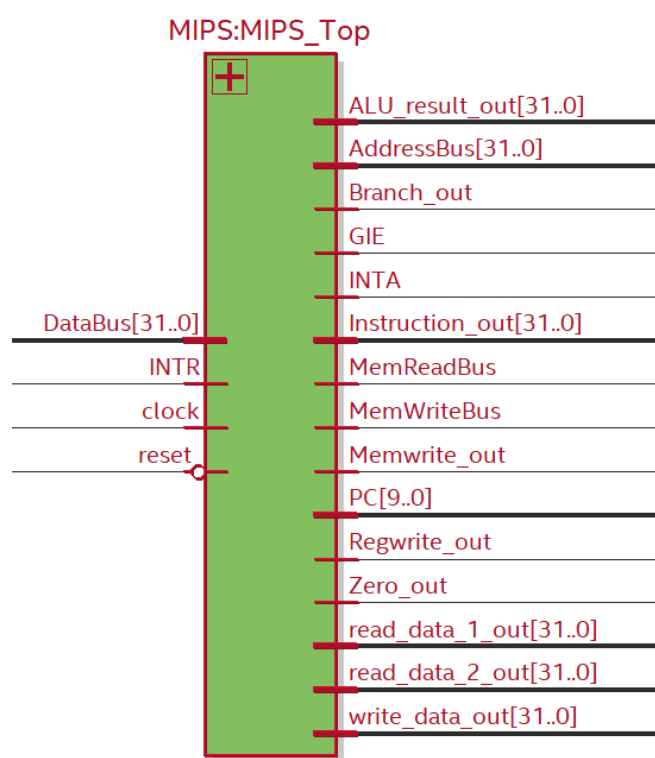
רכיב זה הוא CPU של ה-MCU, הפועל בארכיטקטורת Single Cycle MIPS. המשמעות היא שהמעבד מבצע כל הוראה במחזור שעות אחד בלבד. המעבד כולל זיכרון נתונים (DTCM) וזיכרון תוכנה (ITCM) עבור אחסון נתוני התוכנית ומקטעי הקוד. הארכיטקטורה מבוססת על עקרונות של ארכיטקטורת Harvard, המאפשרת גישה נפרדת ומהירה לזיכרונות התוכנית והנתונים. המעבד תומך בהוראות המצוינות בנספח, ומסוגל לבצע תוכניות המשתמשות אך ורק בהוראות אלו. להלן שרטוט המעבד –



איור 5: שרטוט מעבד ה-CPU



איור 6: שרטוט RTL של CPU



איור 7: שרטוט גרפי של CPU

Fetch

שלב ה-Fetch במעבד MIPS הוא השלב הראשון בתהליך ביצוע ההוראות. במהלך שלב זה, המעבד קורא את ההוראה מהזיכרון לפי הכתובת הנוכחית של המצביע להוראה (Program Counter, PC). ההוראה שנקראה מאוחסנת ברישום הוראות (Instruction Register), וה-PC מעודכן לכתובת ההוראה הבאה, כך שהמעבד יוכל לקרוא את ההוראה הבאה במחזור השעון הבא.

Control

שלב ה-Control במעבד MIPS הוא השלב שבו נקבעים האותות הנדרשים כדי לשלוט על פעולות המעבד בהתאם להוראה שנקראה ופורשה. במהלך שלב זה, המעבד בודק את הקוד של ההוראה (opcode) ומנפיק את האותות המתאימים לשאר החלקים של המעבד, כגון יחידת החישוב (ALU), זיכרון הנתונים, והרגיסטרים. אותות אלו קובעים את סוג הפעולה שתבוצע.

Decode

שלב ה-Decode במעבד MIPS הוא השלב השני בתהליך ביצוע ההוראות. במהלך שלב זה, המעבד מפרש את ההוראה שנקראה בשלב ה-Fetch. ההוראה מתפרקת לחלקים המרכיבים אותה, כמו קוד ההוראה (opcode), רגיסטר המקור, ורגיסטר היעד, במטרה להבין את הפעולה שיש לבצע ואת הפרמטרים שלה. בשלב זה, המעבד גם מזהה את הסוג של ההוראה (למשל, R-type, I-type או J-type) ומבצע את ההתאמות הנדרשות, כמו הבאת הערכים הנדרשים מהרגיסטרים. בנוסף, אם ההוראה דורשת עדכון של הרגיסטרים, שלב זה מבצע את הפעולה הזו בהתאם לפרמטרים שנמסרו מהשלב הקודם.

Execute

שלב ה-Execute במעבד MIPS הוא השלב שבו מתבצע החישוב או הפעולה שהוראה מבקשת לבצע. במהלך שלב זה, יחידת החישוב (ALU) או רכיבים אחרים במעבד מבצעים את הפעולה המוגדרת בהוראה, כמו חיבור, חיסור, או חישובים לוגיים. עבור הוראות שמבצעות גישה לזיכרון, בשלב זה מחושבת הכתובת הנדרשת לגישה. ועבור הוראות מסוג J-type מתבצע החישוב של כתובת ההסתעפות.

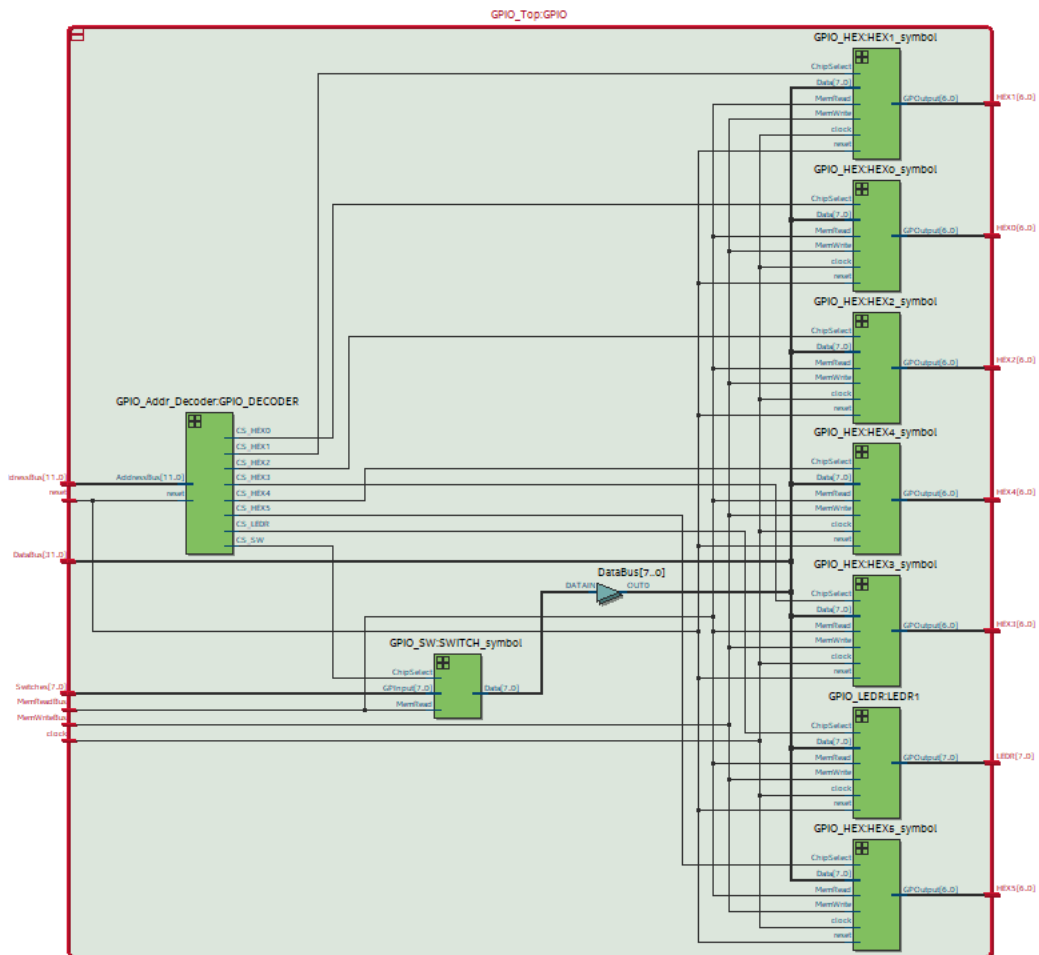
Data Memory

שלב ה-Data Memory במעבד MIPS הוא השלב שבו מתבצעת גישה לזיכרון הנתונים, בהתאם לתוצאה שנשארה משלב ה-Execute. בשלב זה, אם ההוראה דורשת קריאה או כתיבה של נתונים לזיכרון, המעבד מבצע את הפעולות הנדרשות על פי הכתובת והנתונים שנקבעו:

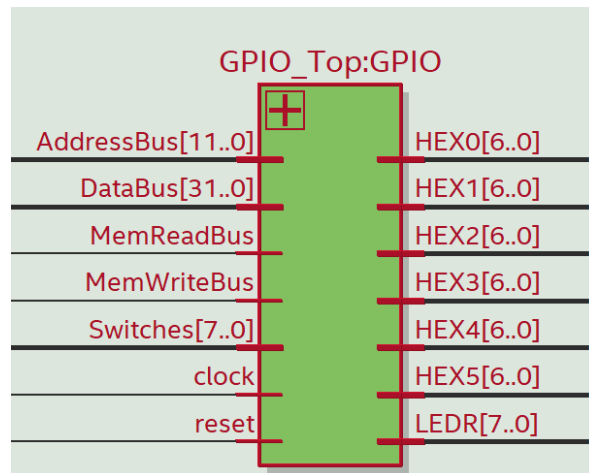
- קריאה מזיכרון: אם ההוראה דורשת לקרוא נתונים מהזיכרון, הכתובת הנכונה נשלחת לזיכרון (Data Memory) והתוצאה נקראת מהכתובת הזו ומשוויכת לרגיסטר המיועד.
- כתיבה לזיכרון: אם ההוראה דורשת כתיבה של נתונים לזיכרון, הנתונים שצריך לכתוב והכתובת שאליה יש לכתוב נשלחים לזיכרון, והנתונים נכתבים למיקום המתאים בזיכרון.

:GPIO

כדי להוסיף ממשק משתמש, הוספנו למיקרו-בקר ממשק של כניסות ויציאות (GPIO) שיכול לכלול רכיבים כמו לדים, כפתורים ועוד. הכתיבה והקריאה מרכיבי ה-GPIO מתבצעות דרך קווי ה-BUS המיועדים לכך במערכת, בהתאם לצורך. בנוסף, ממשק זה יכול לספק פסיקות, גבון לחיצה על כפתור, שתוביל להפעלת ISR שיבצע רוטינה מסוימת.



איור 8: שרטוט RTL של GPIO TOP



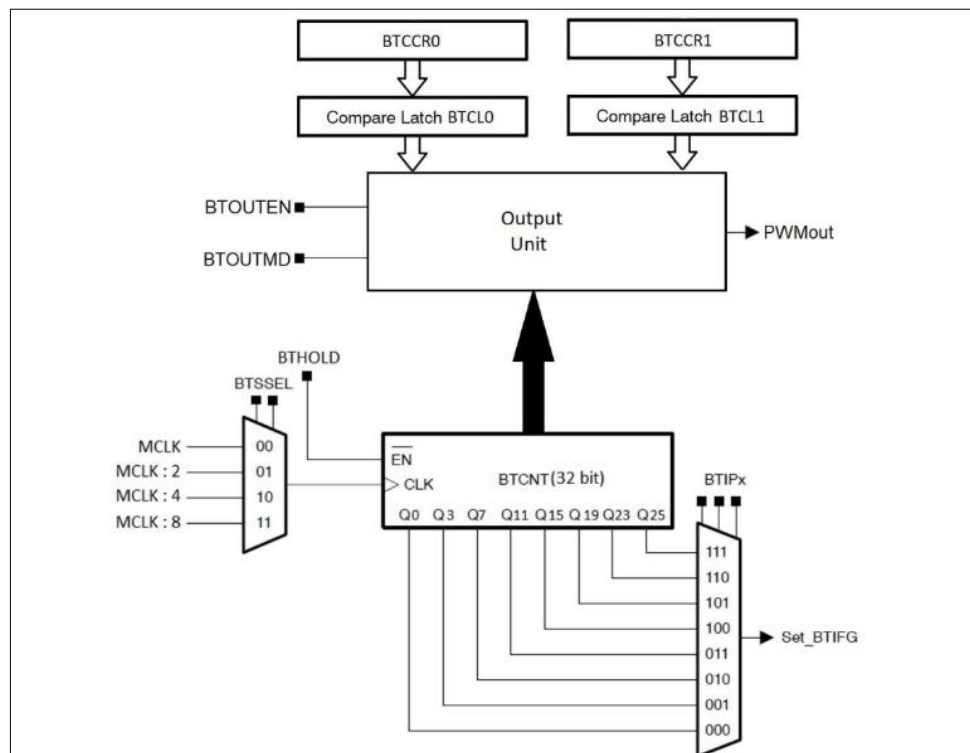
איור 9: שרטוט גרפי של GPIO TOP

Basic Timer

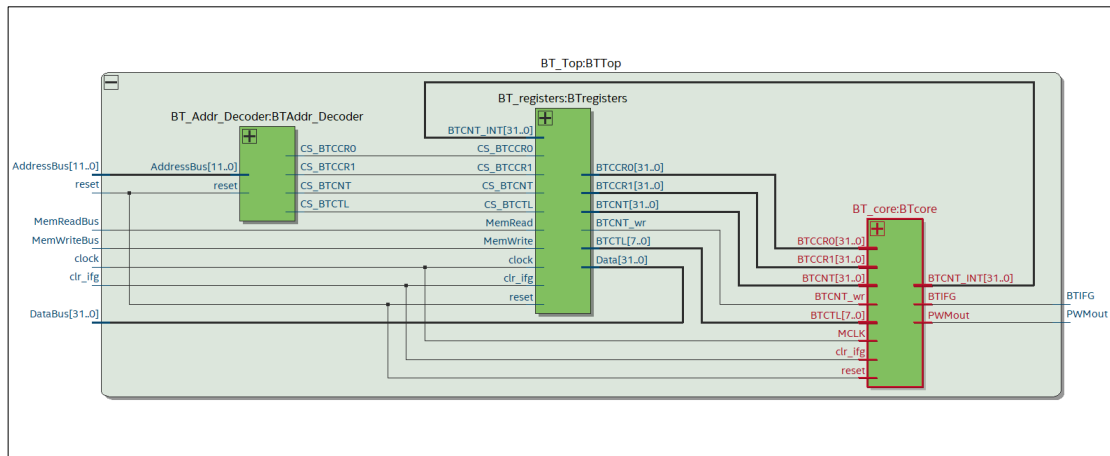
רכיב ה Basic Timer הוא רכיב פריפריאלי הממלא שתי מטרות עיקריות:

הוא מהווה רכיב חומרה פשוט שמבצע ספירה על בסיס עלויות שעון. הרכיב כולל רגיסטר בקרה שמחזיק הגדרות שונות בהתאם למשימה הנדרשת, ורגיסטר BTCNT שמייצג את ערך הספירה הנוכחי של הטיימר. ברגיסטר הבקרה נמצא גם ערך BTIP שלפיו הרכיב מעלה דגל כאשר ערך רגיסטר הספירה מגיע לערך מסוים. דגל זה יכול ליצור Interrupt ולעצור את ביצוע התוכנית הראשית של המיקרו-בקר ולהפעיל ISR מתאים.

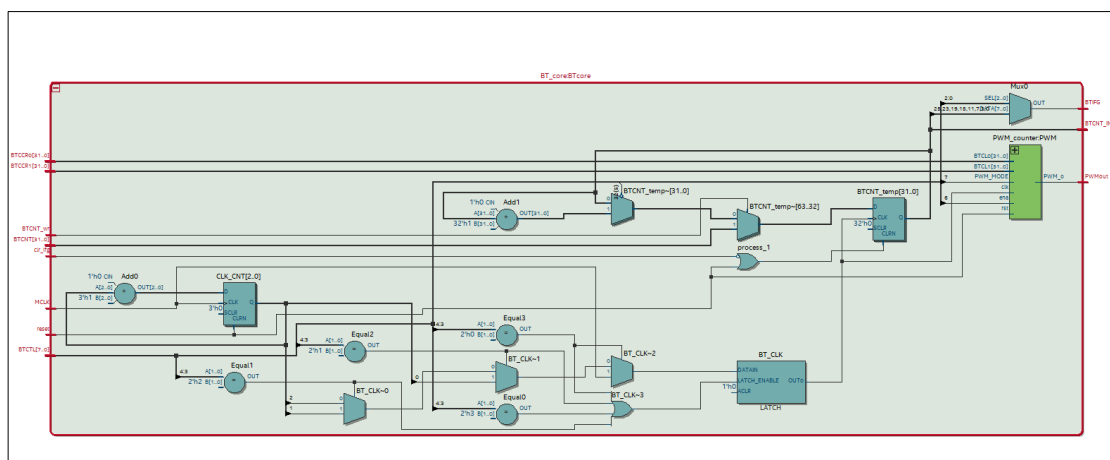
בנוסף, הרכיב יוצר אות PWM בעזרת הטיימר הבסיסי. הטיימר סופר כמקודם וניתן לשלוט על זמן המחזור של האות ועל ה-Cycle Duty שלו באמצעות הרגיסטרים CCR0 ו-CCR1 בהתאמה.



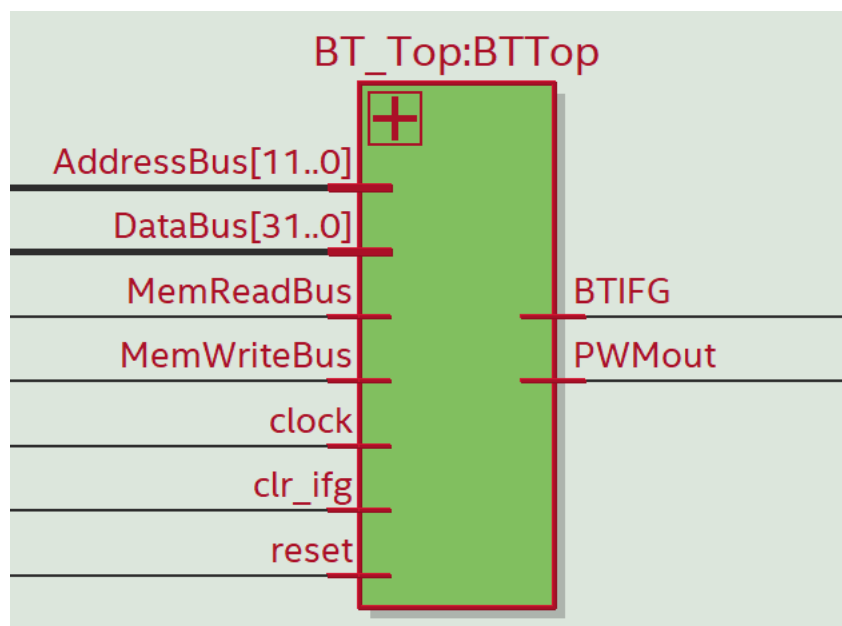
איור 10: שרטוט של ליבת הBT



איור 11: שרטוט RTL של מעטפת הBT



איור 12: שרטוט RTL של ליבת הBT

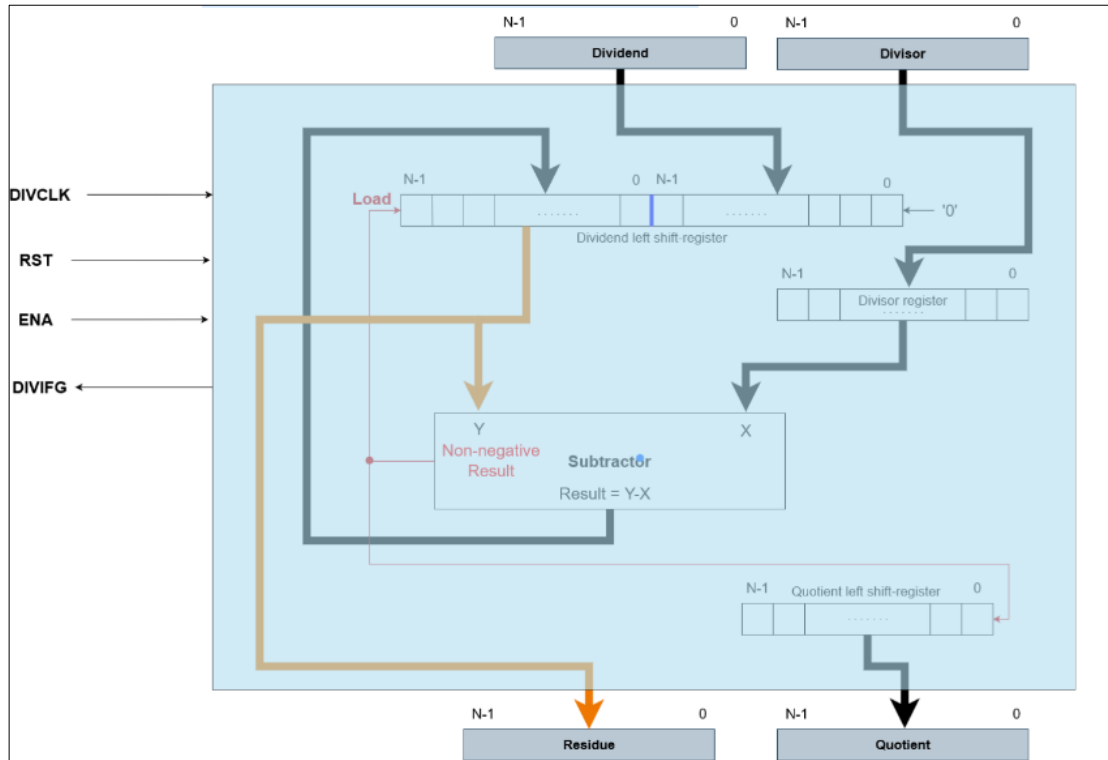


איור 13: שרטוט גרפי של ליבת הBT

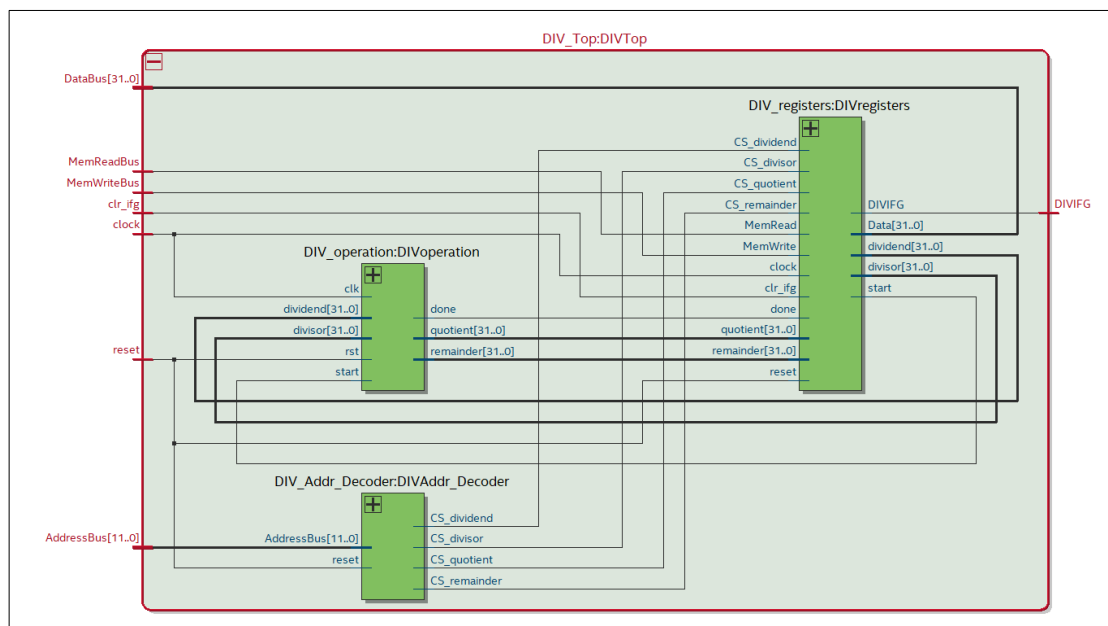
Division accelerator

ה- Unsigned Binary Division Multicycle Accelerator הוא רכיב חומרה שנועד לבצע חלוקות בינאריות ללא סימן (unsigned binary division) ב-32 מחזורי שעון. רכיב זה מבצע את תהליך החלוקה תוך במספר שלבים נפרדים, כל אחד מהם עוסק בחלק אחר של החישוב, מה שמאפשר לבצע חלוקות בצורה מדויקת ויעילה.

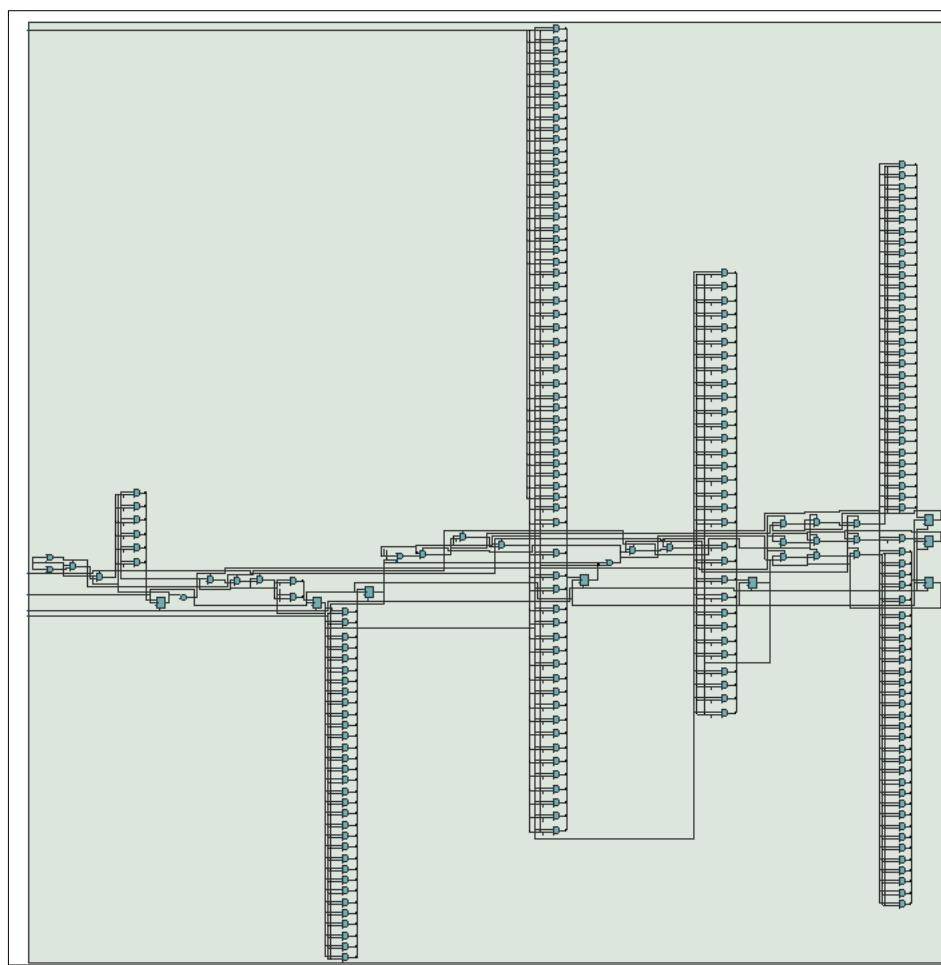
בנוסף, ה-Accelerator מעלה דגל פסיקה שנותן interrupt במקרה של סיום החלוקה.



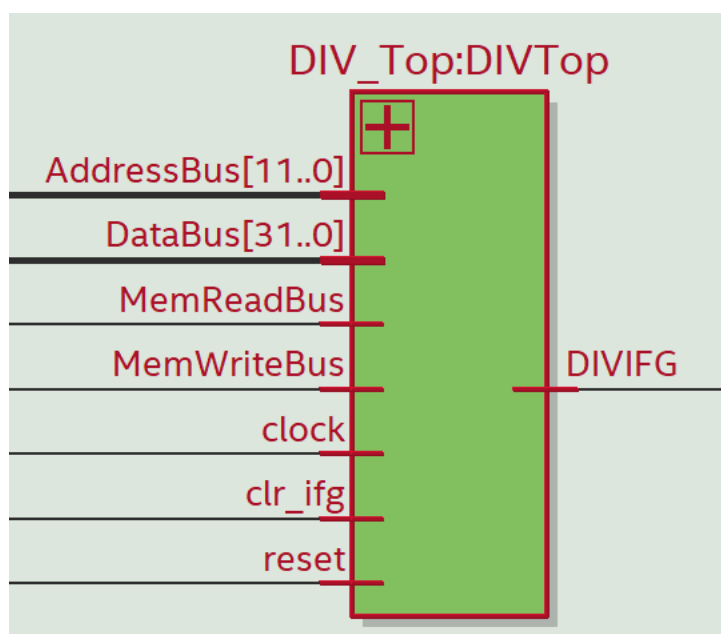
איור 14: שרטוט של ליבת ה- Division accelerator



איור 15: שרטוט RTL של מעטפת ה- Division accelerator



איור 16: שרטוט RTL של ליבת ה-Division accelerator



איור 17: שרטוט גרפי של מעטפת ה-Division accelerator

Interrupt Controller

רכיב זה מנהל את כל הInterrupts של המערכת. בחלק מתפקידו הוא מתעדף בין פסיקות לפי סדר עדיפות שהוגדר מראש (מצורף באיור), ומנהל את הפסיקה מול MIPS. פסיקה מתבצעת כאשר רכיבי החומרה מבקשים לבצע פסיקה על ידי שליחת סיגנל בשם IRQ. פסיקה תתבצע רק אם הפסיקה הרלוונטית מאפשרת ואין פסיקה אחרת בתהליך.

בפרוטוקול הבניסה לפסיקה, הרכיב יוצא דגל INTR שיודיע על כך שרכיב פריפריה ביקש לבצע פסיקה, בנוסף הרכיב יוציא את כתובת ה-ISR (Interrupt Service Routine) אליה נרצה לקפוץ על ה-BUS (בהתאם למי שביצע את הפסיקה), וכך המעבד ידע לקפוץ לכתובת של הISR הרלוונטית. בקבלת, INTA, חזרה מהמעבד נדע שהפסיקה התחילה להתבצע.

TYPE	Contents	Interrupt Source	Interrupt Flag	Interrupt Priority
00h		RESET	NMI	Highest
04h		UART status error	-	
08h		UART RX	RXIFG	
0Ch		UART TX	TXIFG	
10h		Basic Timer	BTIFG	
14h		KEY1	KEY1IFG	
18h		KEY2	KEY2IFG	
1Ch		KEY3	KEY3IFG	
20h		DIVIDER	DIVIFG	Lowest

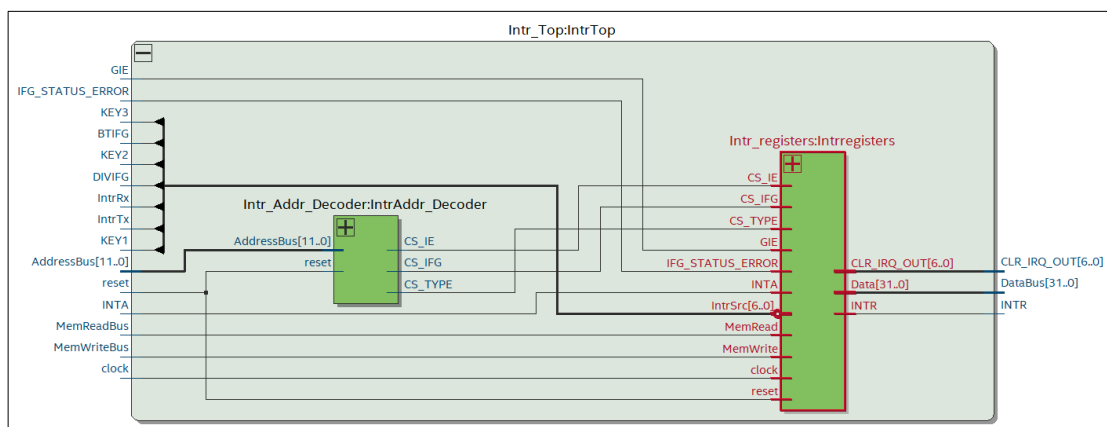
(Non)-Maskable Interrupt

Maskable Interrupt

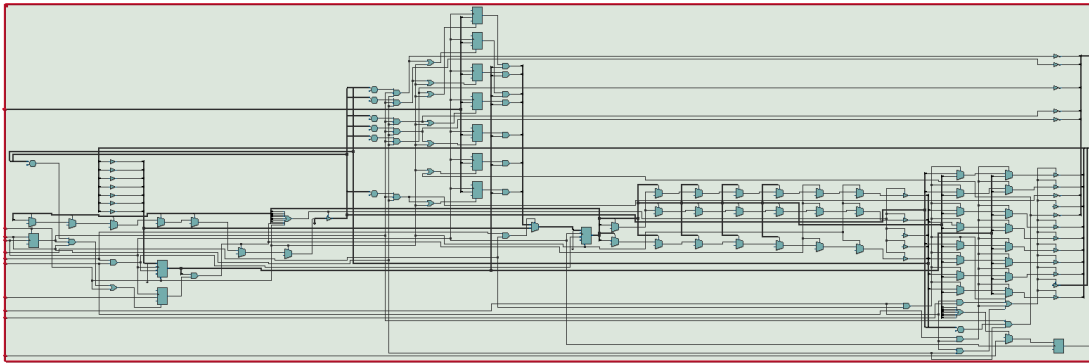
איור 18: תיעוד בקשת הפסיקות

<i>Peripherals with interrupt capability</i>	#define	PORT_KEY[3-1]	0x814	- LSB nibble (3 push-buttons - Input Mode)
	#define	UCTL	0x818	- Byte
	#define	RXBF	0x819	- Byte
	#define	TXBF	0x81A	- Byte
	#define	BTCTL	0x81C	- LSB byte
	#define	BTCNT	0x820	- Word
	#define	BTCCR0	0x824	- Word
	#define	BTCCR1	0x828	- Word
	#define	DIVIDEND	0x82C	- Word
	#define	DIVISOR	0x830	- Word
	#define	QUOTIENT	0x834	- Word
	#define	RESIDUE	0x838	- Word
	#define	IE	0x83C	- LSB byte
	#define	IFG	0x83D	- LSB byte
	#define	TYPE	0x83E	- LSB byte

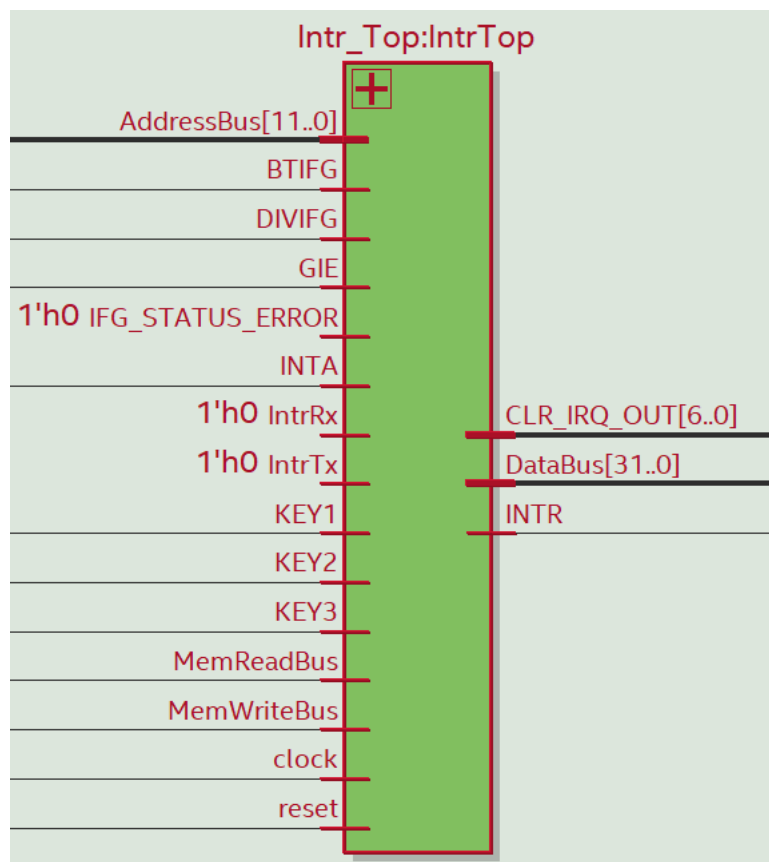
איור 19: כתובות הרכיבים הפריפריאליים בזיכרון



איור 20: שרטוט RTL של מעטפת הInterrupt controller



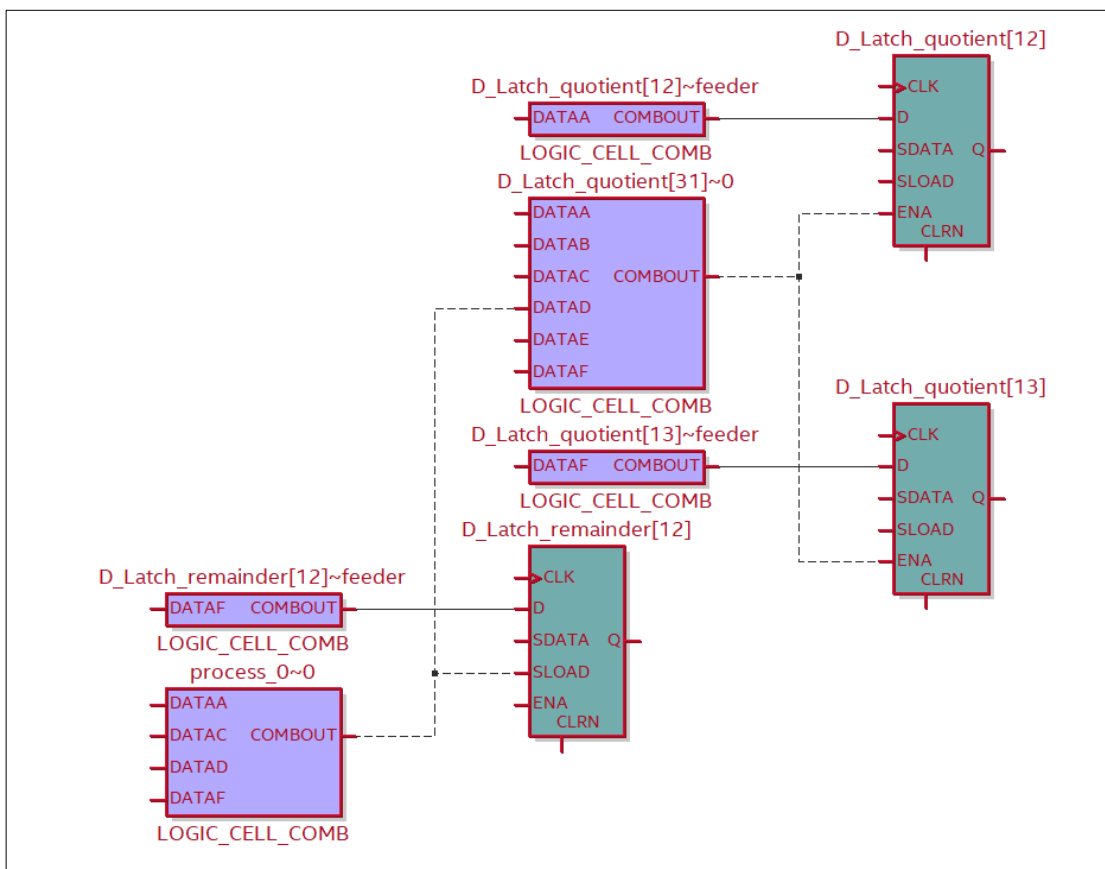
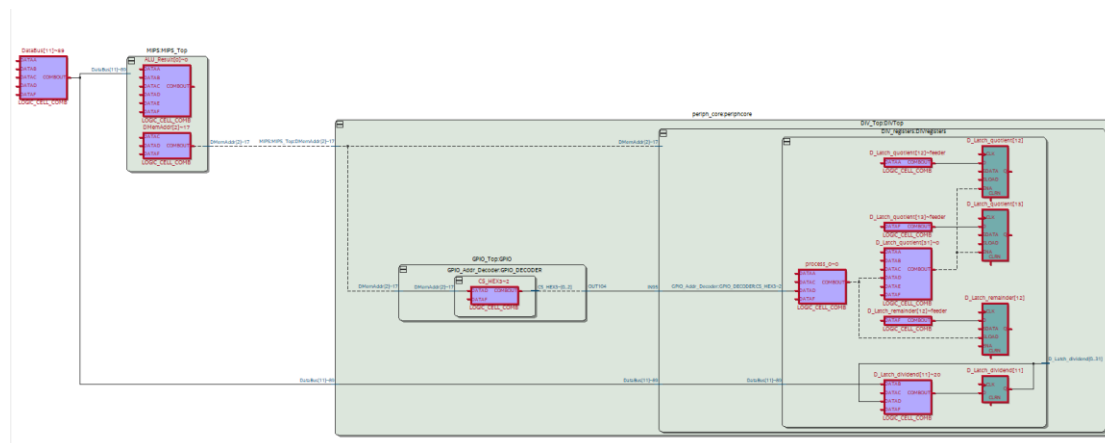
איור 21: שרטוט RTL של ליבת הInterrupt controller



איור 22: שרטוט גרפי של מעטפת הInterrupt controller

הנתיב הקריטי:

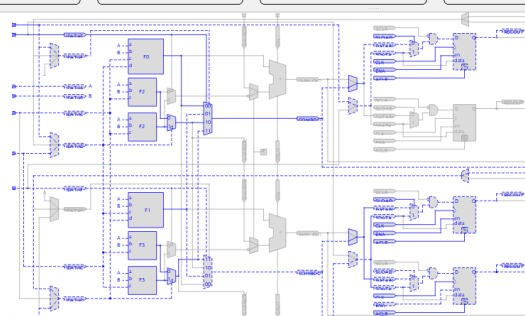
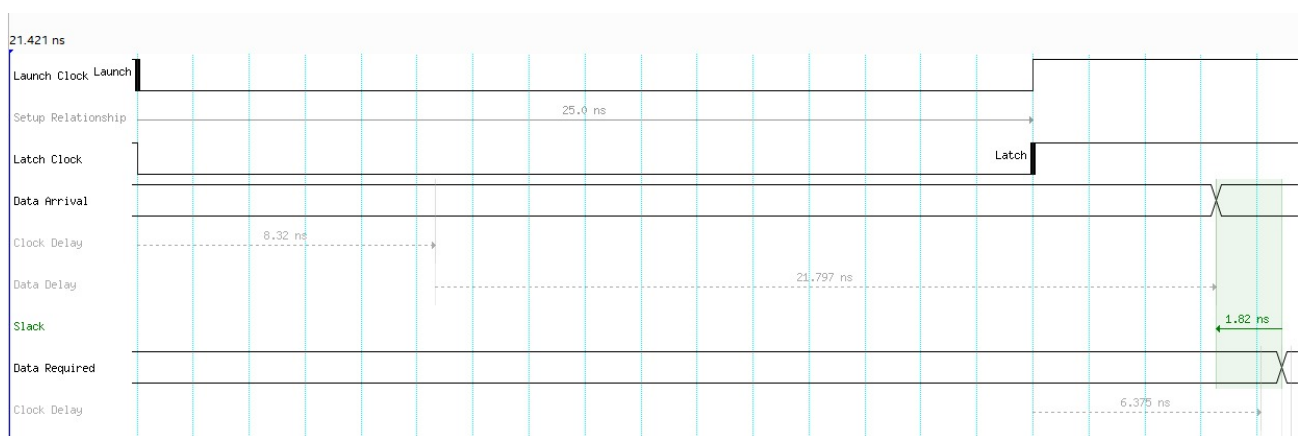
הנתיב הקריטי הוא הנתיב העובר דרך המחלק. בנתיב זה המידע עובר מהMIPS לפרופריה ולאחר 32 מחזורי שעון חוזר חזרה למIPS. לכן, הוא הארוך ביותר.



איור 23: הגדלה של המחלק בתוך הנתיב הקריטי

Node Name	Location
<input type="checkbox"/> project_Top MIPS:MIPS_Top decode:ID Mux47~4	LABCELL_X29_Y12_N57
<input checked="" type="checkbox"/> project_Top periph_core:periphcore DIV_Top:DIVTop DIV_registers:DIVRegisters D_Latch_quotient[12]	FF_X19_Y21_N23
<input checked="" type="checkbox"/> project_Top periph_core:periphcore DIV_Top:DIVTop DIV_registers:DIVRegisters D_Latch_quotient[31]~0	LABCELL_X18_Y19_N30
<input checked="" type="checkbox"/> project_Top periph_core:periphcore DIV_Top:DIVTop DIV_registers:DIVRegisters process_0~0	LABCELL_X18_Y18_N48
<input checked="" type="checkbox"/> project_Top periph_core:periphcore GPIO_Top:GPIO GPIO_Addr_Decoder:GPIO_DECODER CS_HEX3~2	LABCELL_X17_Y18_N6
<input type="checkbox"/> project_Top periph_core:periphcore Intr_Top:IntrTop Intr_Addr_Decoder:IntrAddr_Decoder Equal0~5	LABCELL_X18_Y14_N42

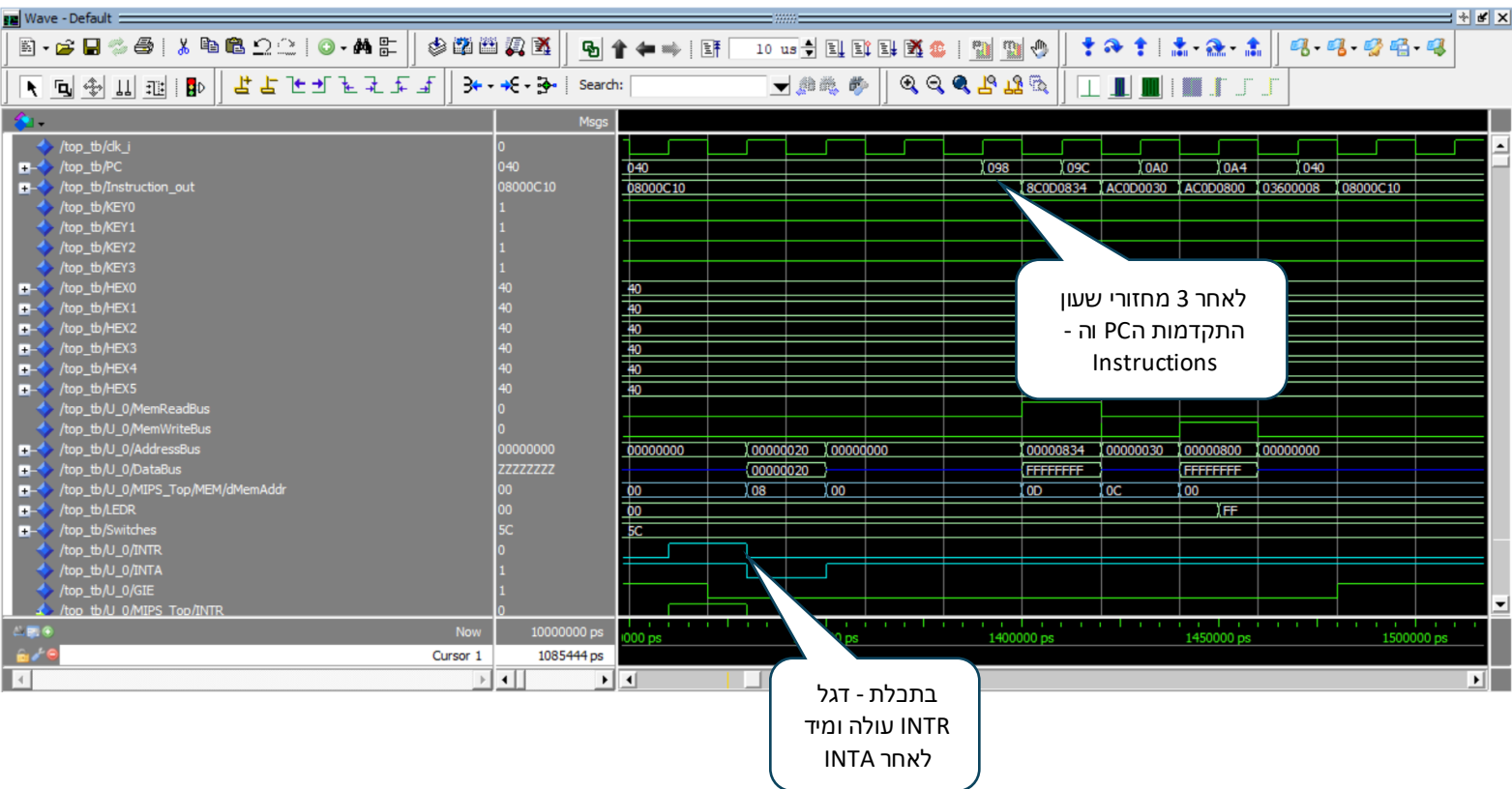
Logic Resource(s) / I/O(s) / RAM(s) / DSP Element(s) / PLL_REFCLK_SELECT(s) / PLL_OUTPUT_COUNTER(s) / FRACTIONAL_PLL(s) / CLKCTRL(s)

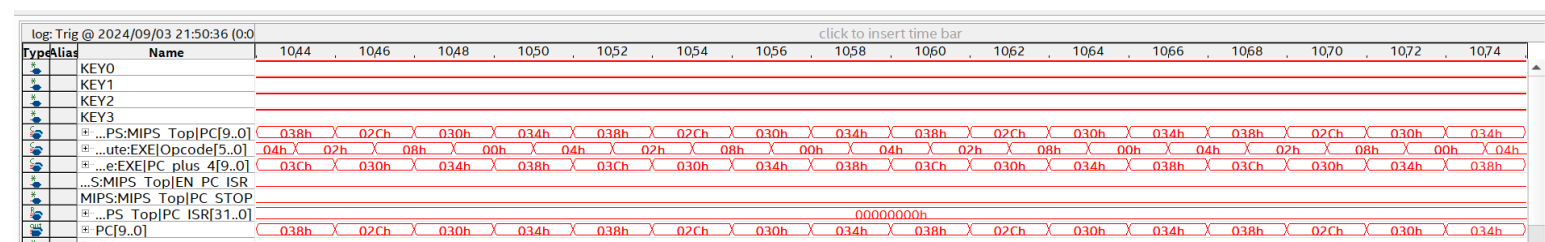
התדר המקסימלי של המערכת כפי שהתקבל משעון המחלק -

Slow 1100mV 85C Model Fmax Summary				
<<Filter>>				
	Fmax	Restricted Fmax	Clock Name	Note
1	20.85 MHz	20.85 MHz	p0 pll_in...ER divclk	

ניתוח תוצאות –



בחרנו לייצר גל ריבועי עם Duty cycle של 75%



טסט 0 – של ה-GPIO