Final Project - REPORT

ADVANCED CPU ARCHITECTURE AND HARDWARE

MIPS Based MCU Architecture

Tal Adoni – 319087300

Omri Aviram – 312192669

במעבדה זו היה עלינו לממש *MIPS based MCU architecture* המשתמשת במעבד ה-*MIPS* שיצרנו במעבדה קודמת ומחברת אותו עם רכיבים פריפריאליים באופן הבא:

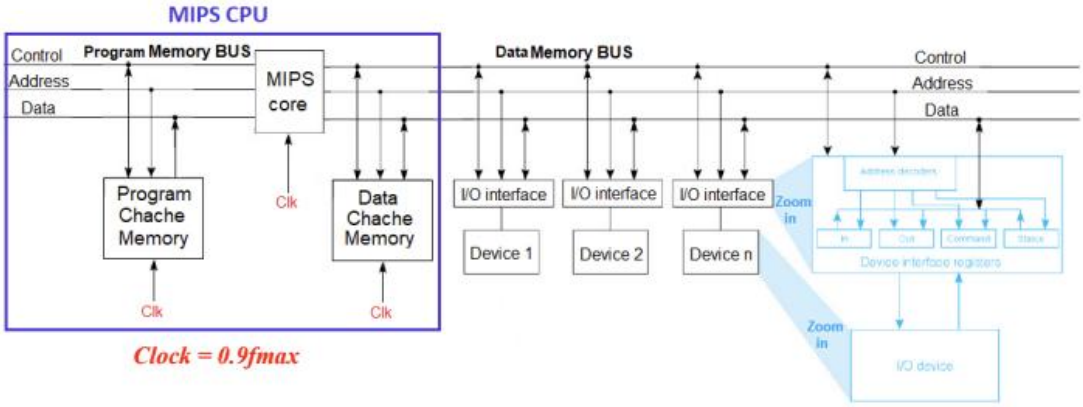
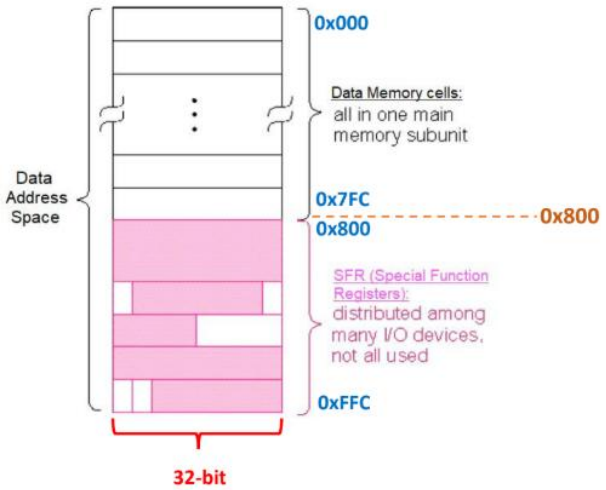


Figure 1 : MCU System architecture

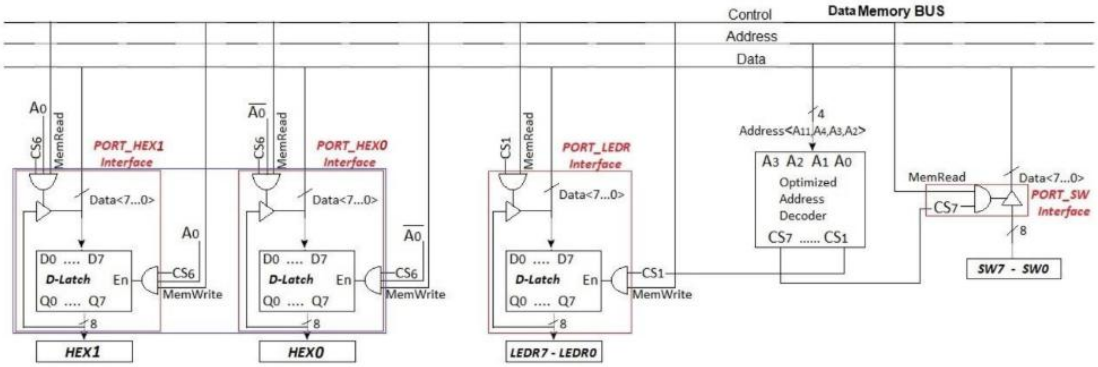
איור 1: מעבד ה- MCU

כחלק מהפרויקט היה עלינו לבנות ולחבר בין המעבד לבין רכיבים פריפריאליים כאשר דרך התקשורת בינם לבין ה- CPUנעשית על ידי שלושה קווים: Control, Address and Data וכל אחד מהקווים הללו נכנסו למודולים כך שקווי הבקרה היו כתיבה/קריאה מהזיכרון (שהגיעו מתוך השלב של הזיכרון) כיוון שאופן העבודה במערכת נעשה באמצעות מיפוי רכיבי ה-IO והרגיסטרים שלהם באמצעות הזיכרון לפי התמונה הבאה – כאשר החלק הלבן הוא הזיכרון שלנו והחלק הורוד הוא הזיכרון "הוירטואלי" זיכרון שאינו חלק ממשי בזיכרון אלא הוא מגיע מתוך רגיסטרי הרכיב ואף חלקם משתנים באופן דינאמי כמו למשל רגיסטר המנייה של Basic Timer.



איור 2: מבנה זיכרון המערכת

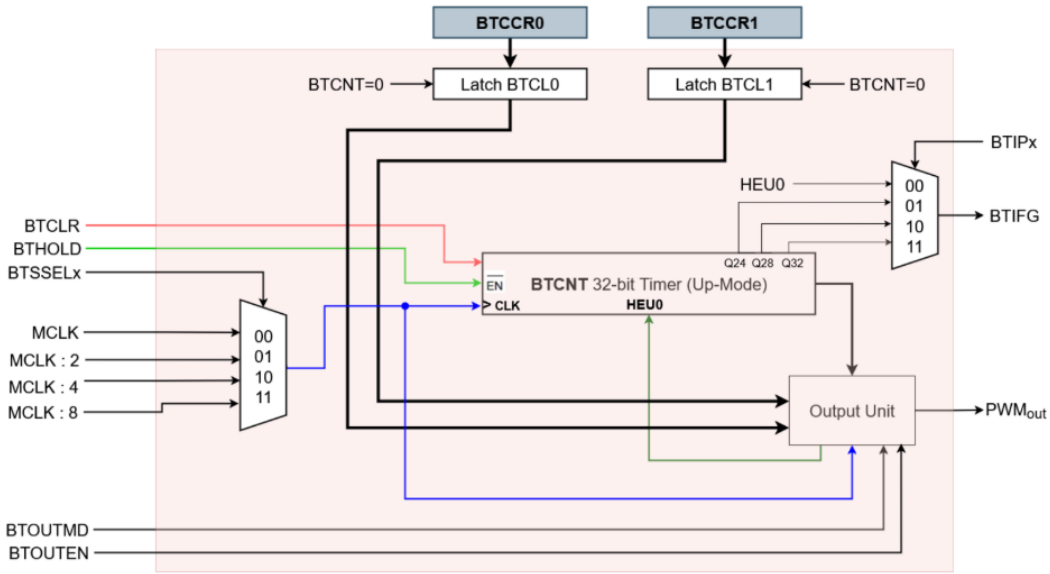
:IO_decoder



איור 3: רכיב ה-GPIO

ברכיב זה נעשית כתיבה ואף קריאה של ערכי הGPI וה-GPO של המערכת או באופן יותר מדויק של הרגיסטרים שלהם, המידע שמוכנס לרכיבי ה-GPO לבסוף הולך אל הרכיב שנמצא על גבי ה-FPGA.

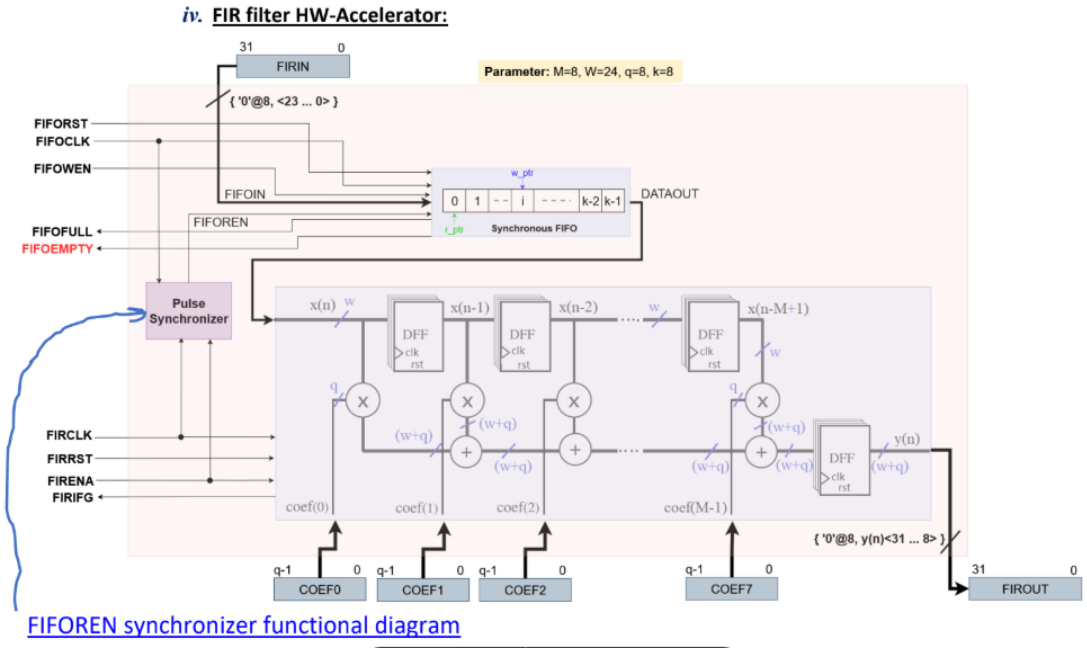
:Basic Timer



איור 4: רכיב ה-Basic Timer

רכיב דומה לרכיב שבנינו במעבדה 4 אשר מוציא אות PWM מאחת מרגלי ה-GPIO של ה-FPGA אך כעת הכנסת המידע מתבצעת ע"י רגיסטרים וכמו כן יש לנו גם פסיקות ממס' מקורות הנשלטים באמצעות BTIPx.

:FIR filter



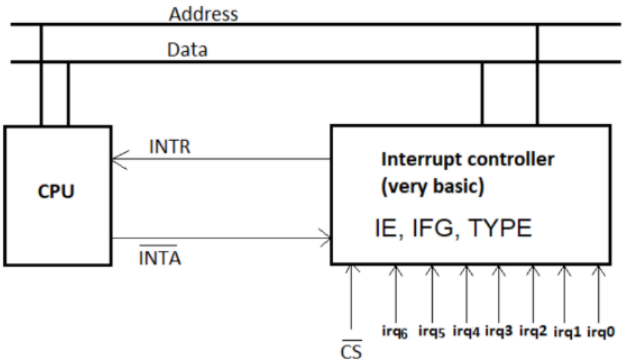
איור 5: רכיב ה-FIR

פילטר FIR הוא מסנן ספרתי דיסקרטי שעובד בעזרת שני שעונים, איטי ומהיר במקרה שלנו ובו היציאה בכל רגע מחושבת כסכום משוקלל של מספר דגימות קלט אחרונות. היתרון המרכזי שלו הוא יציבות מוחלטת (כי אין משוב פנימי) ושליטה מדויקת בצורת התגובה של התדר.

המודול מקבל רצף דגימות מתוך ה-synchronous FIFO, מכפיל כל דגימה בקבוע משקל (Coefficient), ושולח את התוצאה לסכימה. בצורה זו מתקבלת יציאה שהיא קונבולוציה של האות עם מקדם הפילטר.

בנוסף לרכיבים הפריפריאליים יש לנו את יחידת ה-INTR אשר נותנת לנו פסיקות בהתאם להגדרות העבודה:

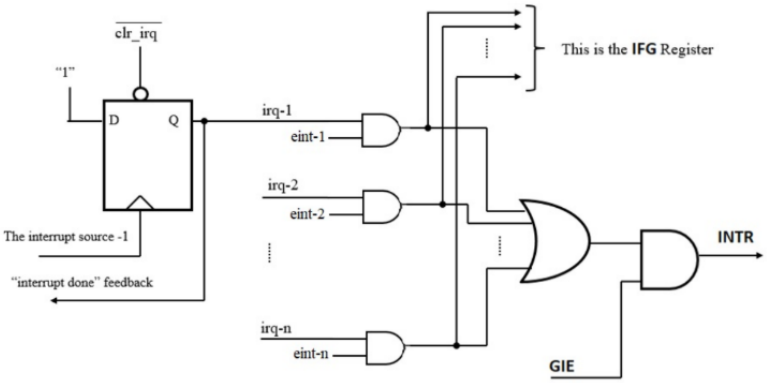
vi. **Interrupt controller:**



איור 6: יחידת ה-INTR

הפסיקות שמתקבלות נשלחות חזרה אל ה-CPU ע"י תיעדוף ולוגיקה פנימית שנעשית באופן של מיסוך פסיקות וקבלת פסיקות לפי הגדרות המשתמש באופן הבא:

Handling interrupts from several sources:



איור 7: לוגיקה פנימית של מטפל הפסיקות

חלוקת הפסיקות לכתובת ועדיפות:

TYPE Contents	Interrupt Source	Interrupt Flag	Interrupt Priority
00h	RESET	NMI	Highest
04h	UART status error	RXIFG	
08h	UART RX		
0Ch	UART TX	TXIFG	
10h	Basic Timer	BTIFG	
14h	KEY1	KEY1IFG	
18h	KEY2	KEY2IFG	
1Ch	KEY3	KEY3IFG	
20h	FIFOEMPTY	FIRIFG	Lowest
24h	FIROUT		

(Non)-Maskable Interrupt

Maskable Interrupt

איור8 : עדיפות וכתובות לפסיקות

בעמוד הבא מצורף האיור של המערכת אותה יצרנו.

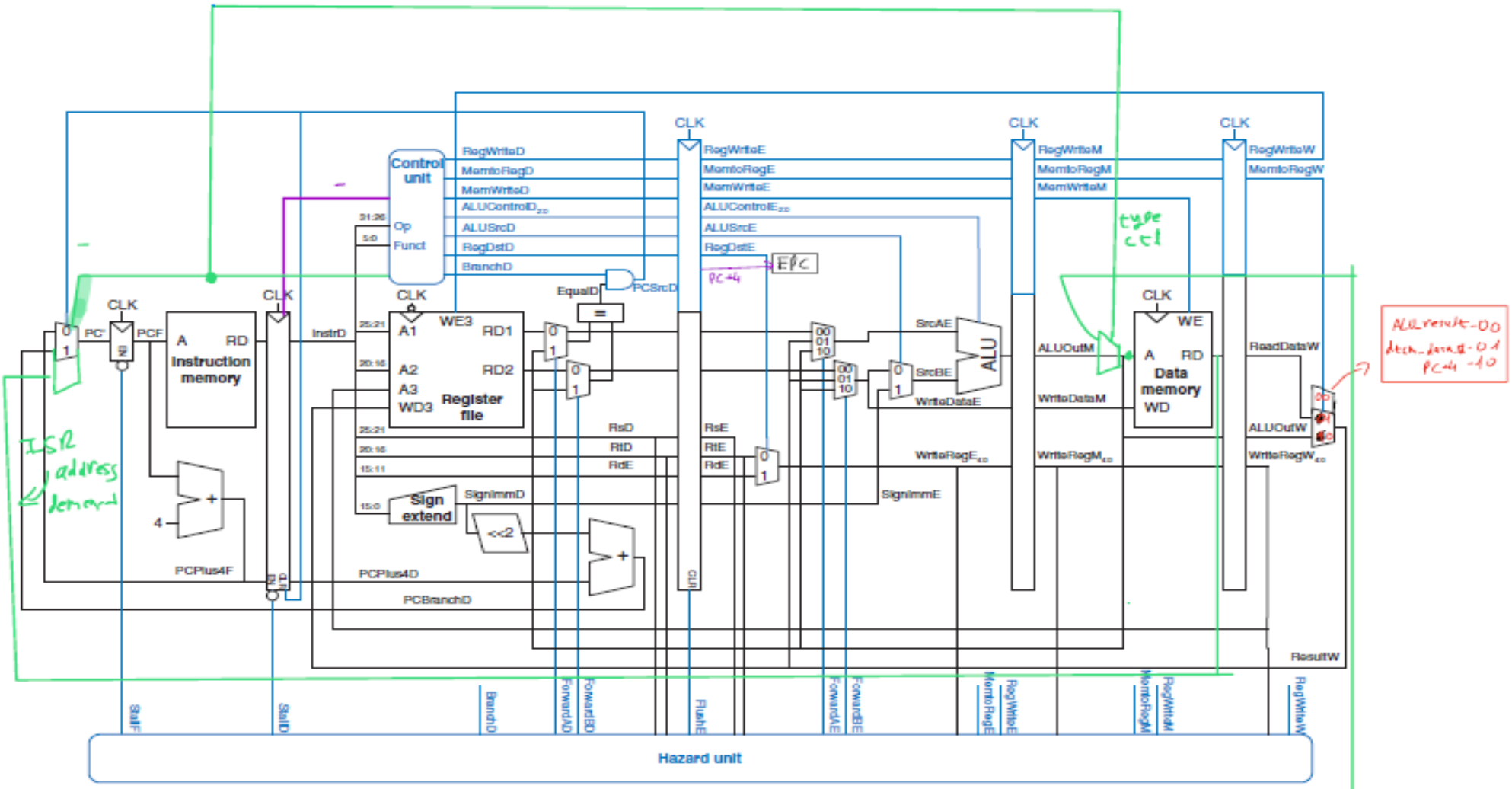
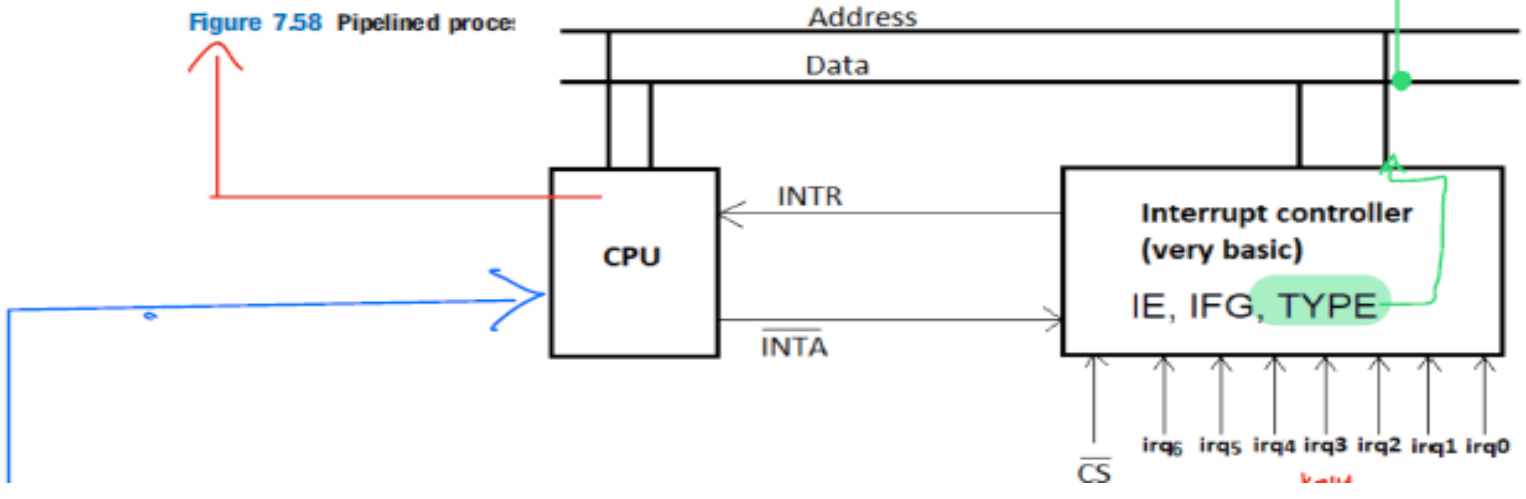
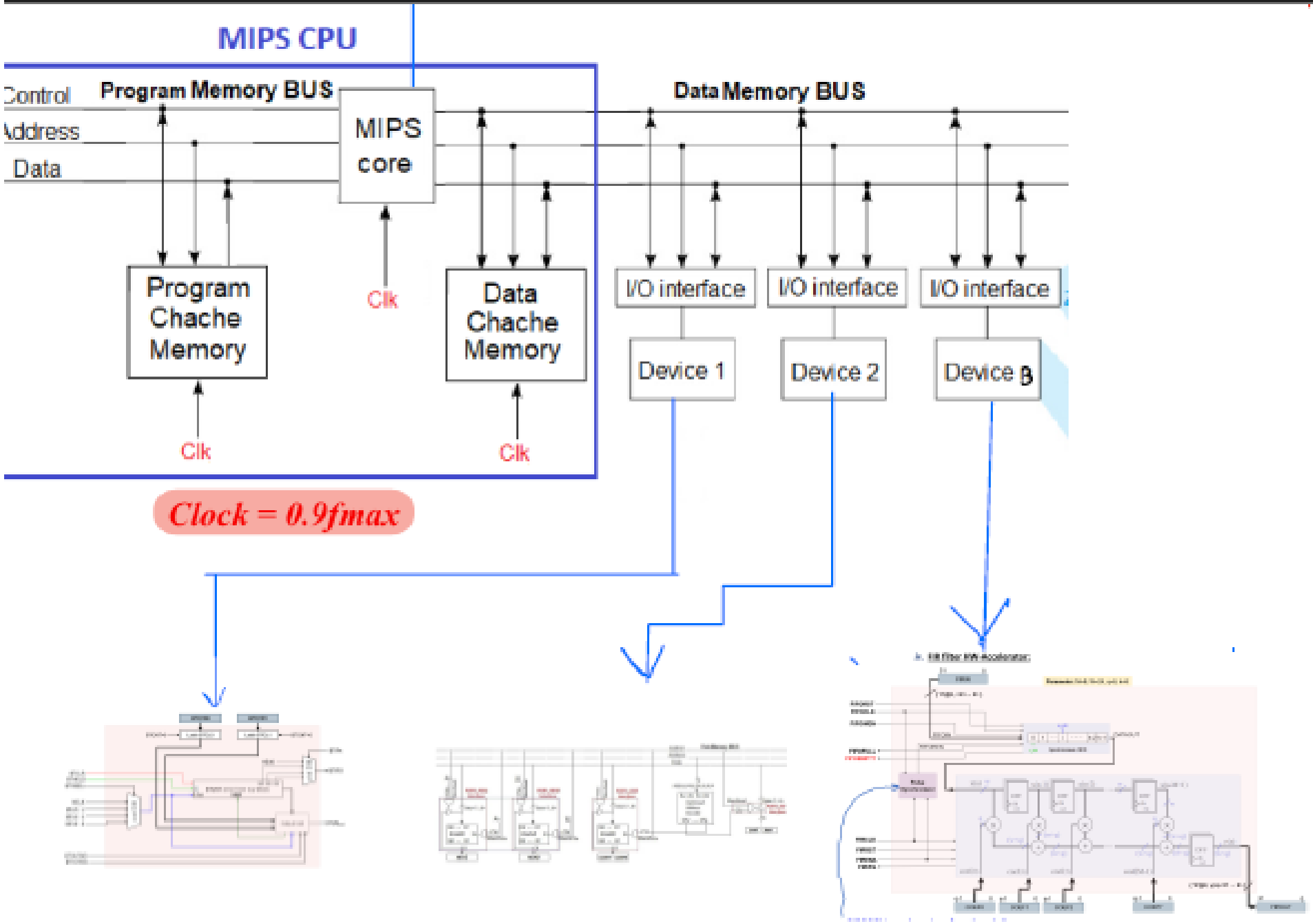
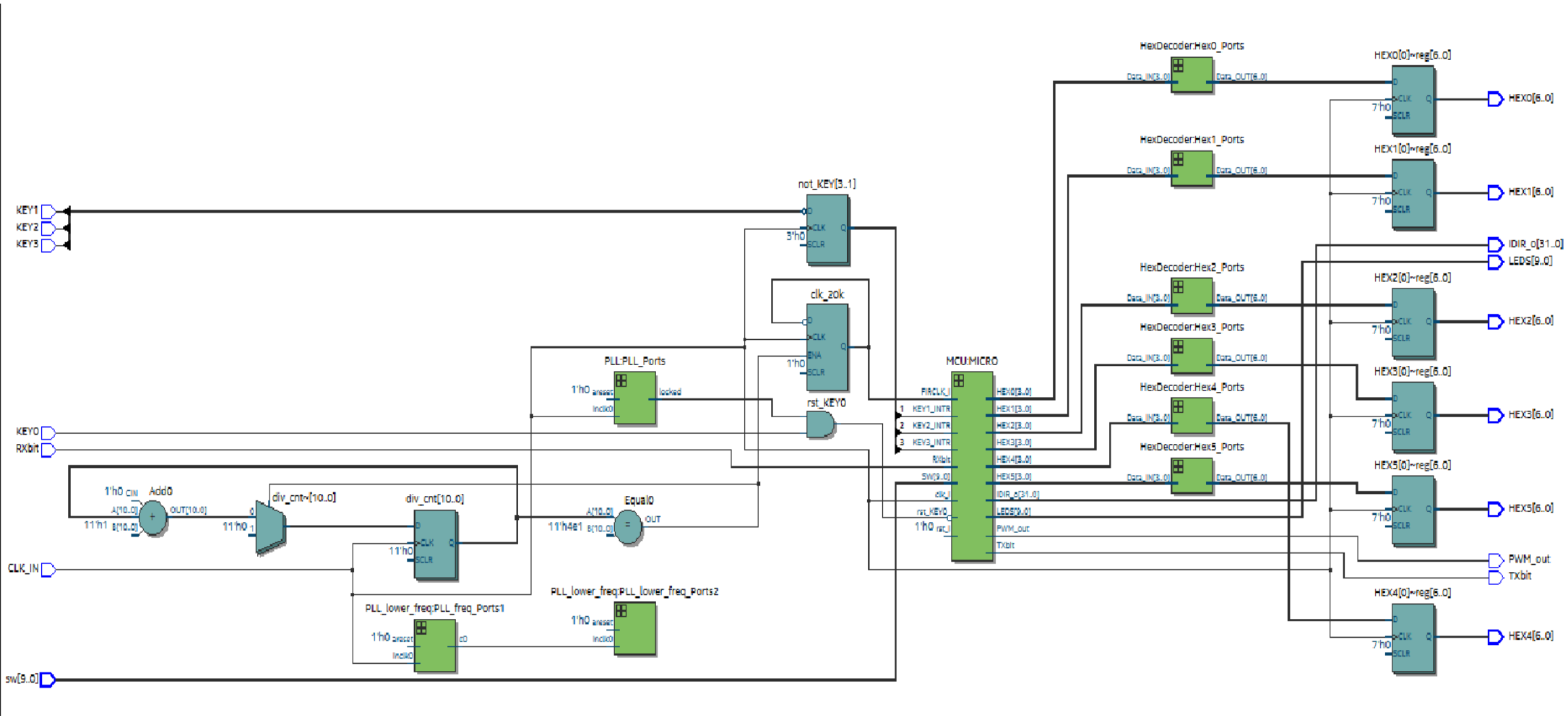


Figure 7.58 Pipelined proce



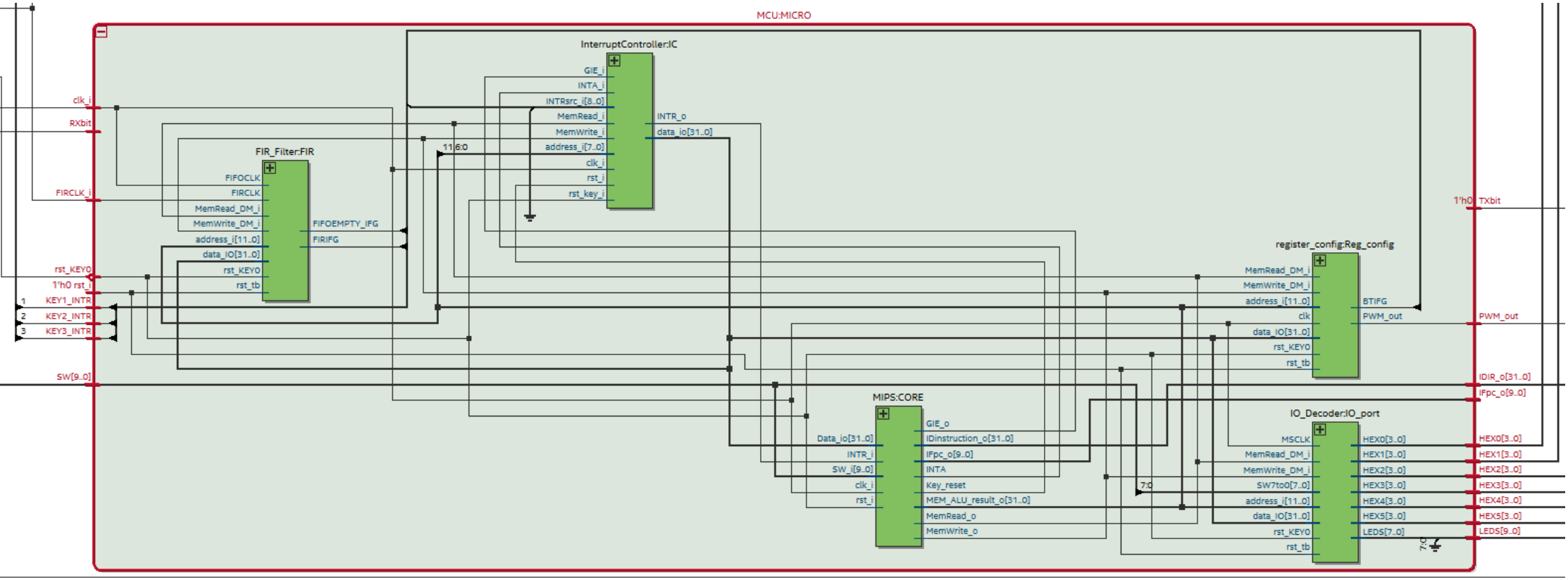

MIPS CPU
Control
Address
Data
Program Memory BUS
MIPS core
Data Memory BUS
Program Chache Memory
Clk
Data Chache Memory
I/O interface
I/O interface
I/O interface
Device 1
Device 2
Device β
Clk
Clk
Clock = 0.9fmax

שכבה עליונה ביותר – Quartus Top:



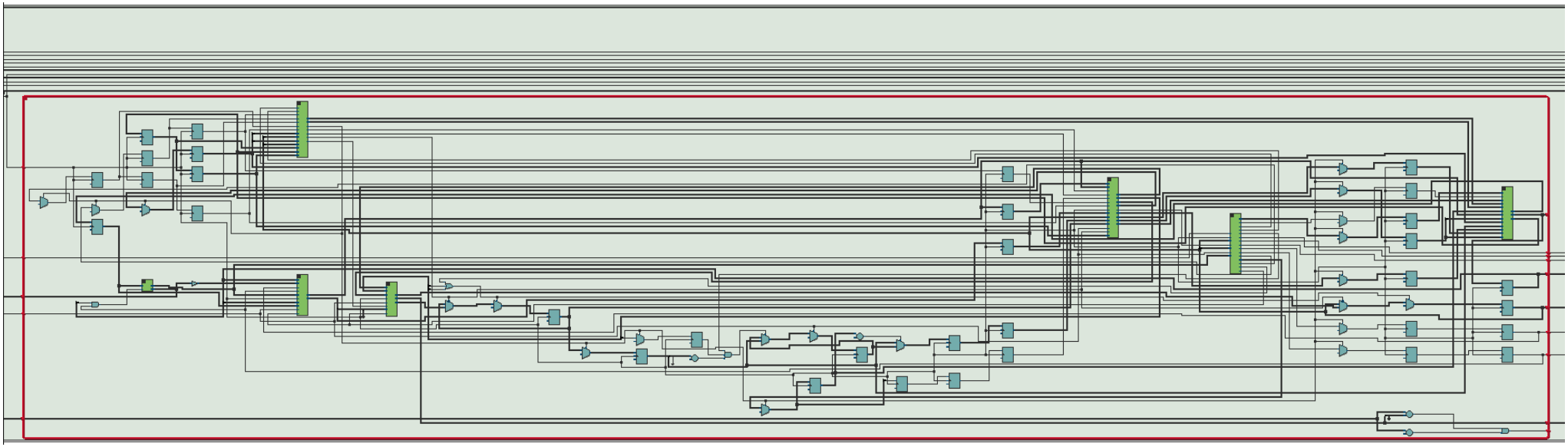
איור 9 : RTL view – Quartus Top

שכבת ה-MCU המכילה את הרכיבים הפריפריאליים ואת המעבד:



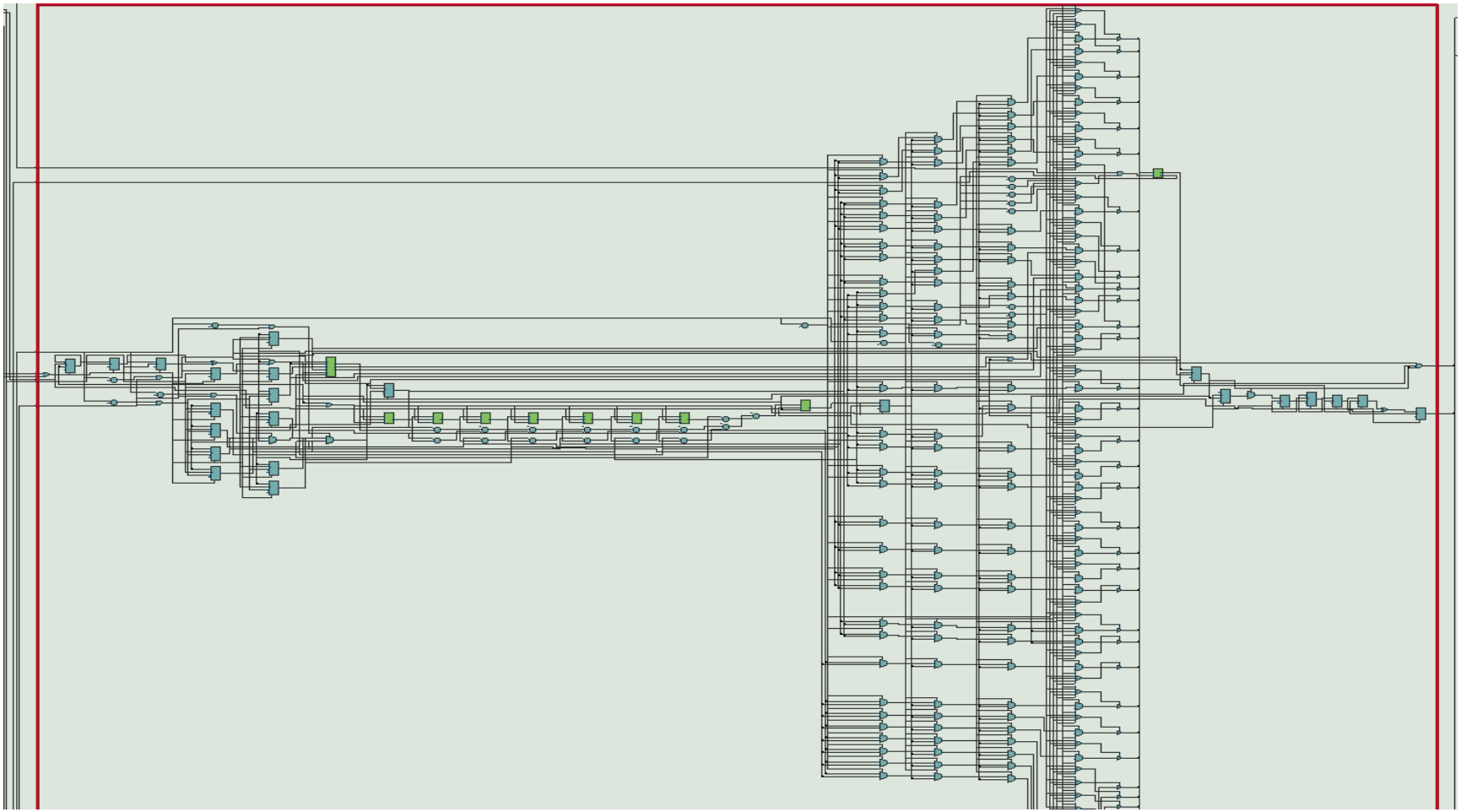
איור10 : RTL view - MCU

מעבד המערכת – Pipelined MIPS:



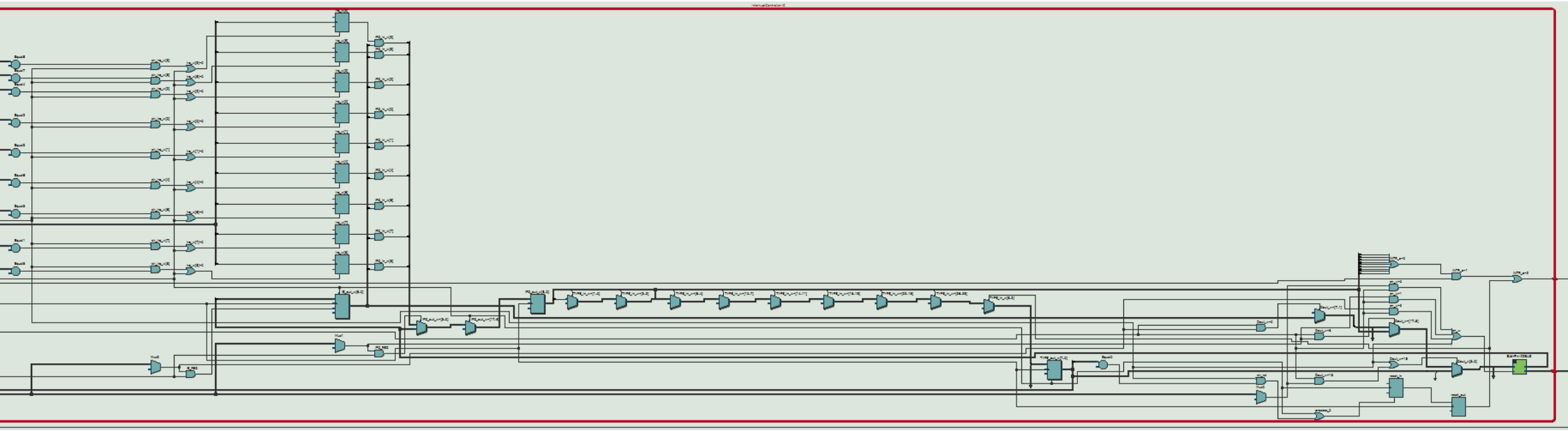
איור 11: RTL view – Pipeline

רכיב ה-FIR:



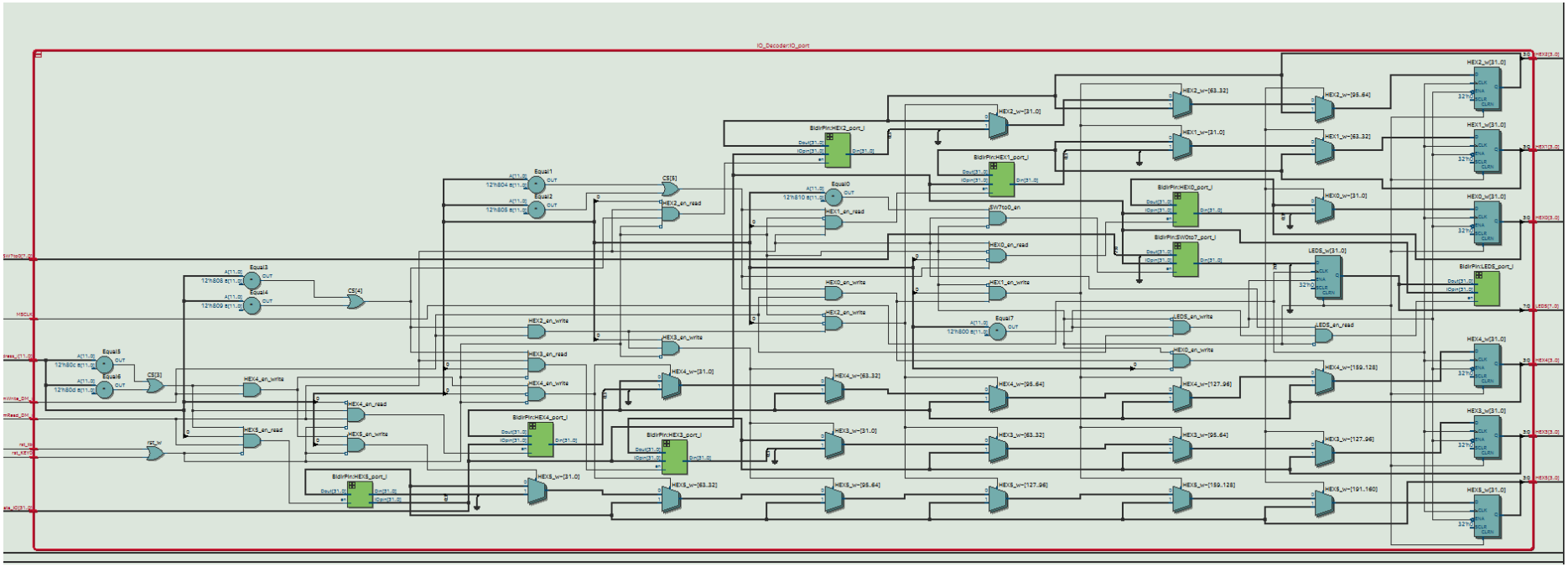
RTL view – FIR : 12איור

רכיב ה-Interrupt Controller:



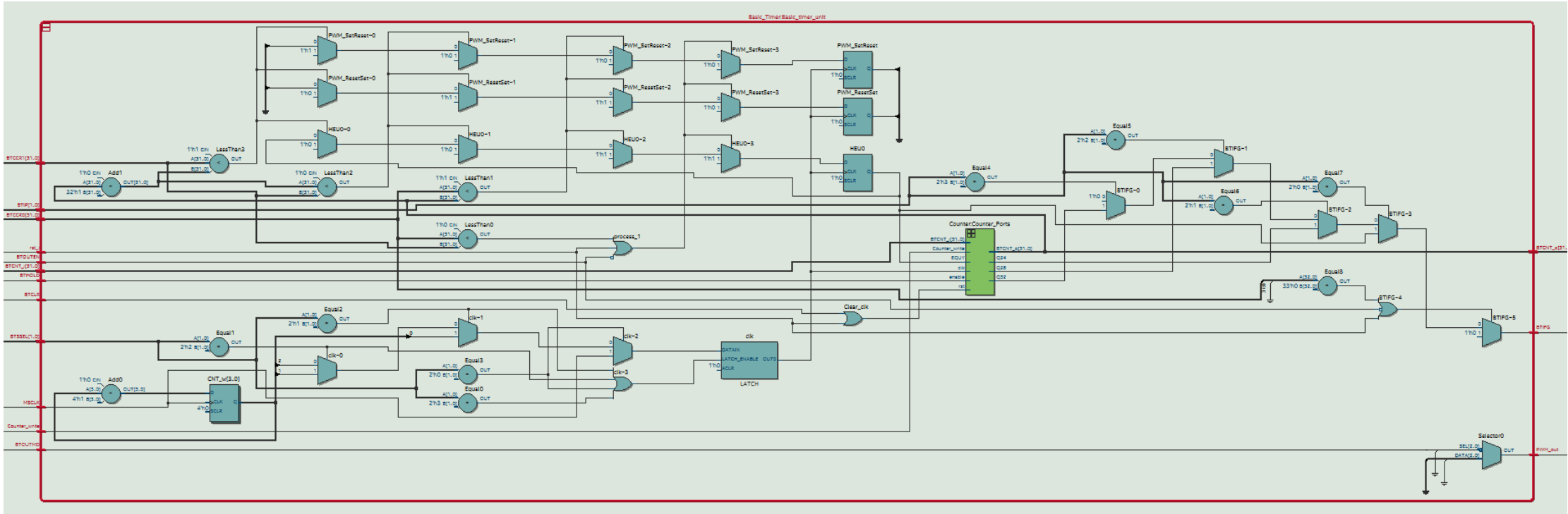
RTL view – FIR : 13איור

רכיב ה-IO decode:



איור 14: RTL view – IO decoder

רכיב ה-Basic Timer:



RTL view – Basic Timer : 15איור

כעת נבדוק את ה-Logic Usage של המערכת:

<u>של המערכת כולה:</u>

Analysis & Synthesis Resource Usage Summary

<<Filter>>

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	3221
2		
3	Combinational ALUT usage for logic	3057
1	-- 7 input functions	26
2	-- 6 input functions	1580
3	-- 5 input functions	503
4	-- 4 input functions	392
5	-- <=3 input functions	556
4		
5	Dedicated logic registers	4432
6		
7	I/O pins	134
8	Total MLAB memory bits	0
9	Total block memory bits	573640
10		
11	Total DSP Blocks	9
12		
13	Total PLLs	1
1	-- PLLs	1
14		
15	Maximum fan-out node	CLK_IN~input
16	Maximum fan-out	3179
17	Total fan-out	36116
18	Average fan-out	4.51

איור 16: בדיקת Logic Usage

<u>:FIR</u>

|FIR_Filter:FIR|	145 (24)	453 (134)	200
|BidirPin:data_IO_tristate_read|	73 (73)	0 (0)	0
|DFF_lab:\coef_DFF:0:DFFs_port|	0 (0)	25 (25)	0
|DFF_lab:\coef_DFF:1:DFFs_port|	0 (0)	25 (25)	0
|DFF_lab:\coef_DFF:2:DFFs_port|	0 (0)	25 (25)	0
|DFF_lab:\coef_DFF:3:DFFs_port|	0 (0)	25 (25)	0
|DFF_lab:\coef_DFF:4:DFFs_port|	0 (0)	25 (25)	0
|DFF_lab:\coef_DFF:5:DFFs_port|	0 (0)	25 (25)	0
|DFF_lab:\coef_DFF:6:DFFs_port|	0 (0)	25 (25)	0
|DFF_lab:yn_DFF|	0 (0)	24 (24)	0
|sync_fifo:synchronus_FIFO|	48 (48)	120 (120)	200
|altsyncram:mem_rtl_0|	0 (0)	0 (0)	200
|altsyncram_3lt1:auto_generated|	0 (0)	0 (0)	200

איור 17: FIR – logic usage

<u>:IO decode</u>

Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits
|HexDecoder:Hex2_Ports|	7 (7)	0 (0)	0
|HexDecoder:Hex3_Ports|	7 (7)	0 (0)	0
|HexDecoder:Hex4_Ports|	7 (7)	0 (0)	0
|HexDecoder:Hex5_Ports|	7 (7)	0 (0)	0
|MCU:MICRO|	2352 (0)	2184 (0)	16584
|FIR_Filter:FIR|	145 (24)	453 (134)	200
|IO_Decoder:IO_port|	132 (18)	32 (32)	0
|BidirPin:HEX0_port_i|	23 (23)	0 (0)	0
|BidirPin:HEX1_port_i|	2 (2)	0 (0)	0
|BidirPin:HEX2_port_i|	2 (2)	0 (0)	0
|BidirPin:HEX3_port_i|	2 (2)	0 (0)	0

איור 18: Logic Usage IO decoder

:Interrupt Controller

Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits
|IO_Decoder:IO_port|	132 (18)	32 (32)	0
|InterruptController:IC|	52 (43)	28 (28)	0
|BidirPin:IC2BUS|	9 (9)	0 (0)	0

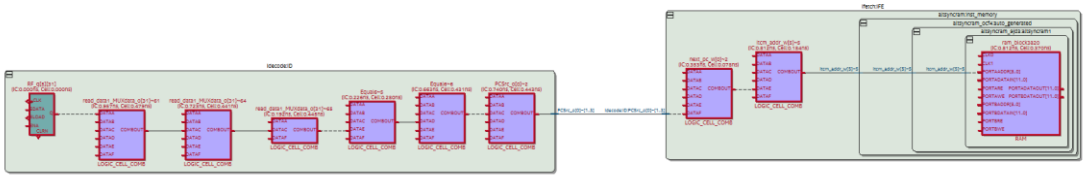
איור 19: Logic usage Interrupt

:MIPS

Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits
|IO_Decoder:IO_port|	132 (18)	32 (32)	0
|InterruptController:IC|	52 (43)	28 (28)	0
|MIPS:CORE|	1803 (35)	1455 (317)	16384
|BidirPin:CORE2BUS|	66 (66)	0 (0)	0
|Execute:EXE|	518 (424)	0 (0)	0
|HazardAndForwarding:HAZ|	23 (23)	0 (0)	0
|Idecode:ID|	894 (894)	992 (992)	0
|Ifetch:IFE|	107 (60)	73 (9)	8192
|control:CTL|	32 (32)	9 (9)	0
|dmemory:\G1:MEM|	128 (81)	64 (0)	8192
|register_config:Reg_config|	220 (19)	216 (178)	0

איור 20: Logic usage MIPS

כעת נבדוק מהו הנתיב הקריטי של המערכת משעון המוצא מה-PLL לעצמו (לצורך מניעה של קבלת ה-PLL כנתיב קריטי):



איור 21: נתיב קריטי של מערכת

כצפוי, ובדומה למערכת קודמת קיבלנו שהנתיב הקריטי עובד מה-Fetch ל-Deocode. הדבר מסתדר עם העובדה שהרכיבים האלה אומנם עטופים ברגיסטרים אך החישוב של הכתובות לקפיצה גורם לכך שיש העברת מידע בין שני המודולים.

תדר מקסימלי:

Slow 1100mV 85C Model Fmax Summary

<<Filter>>

	Fmax	Restricted Fmax	Clock Name	Note
1	55.1 MHz	55.1 MHz	CLK_IN	
2	64.93 MHz	64.93 MHz	altera_reserved_tck	

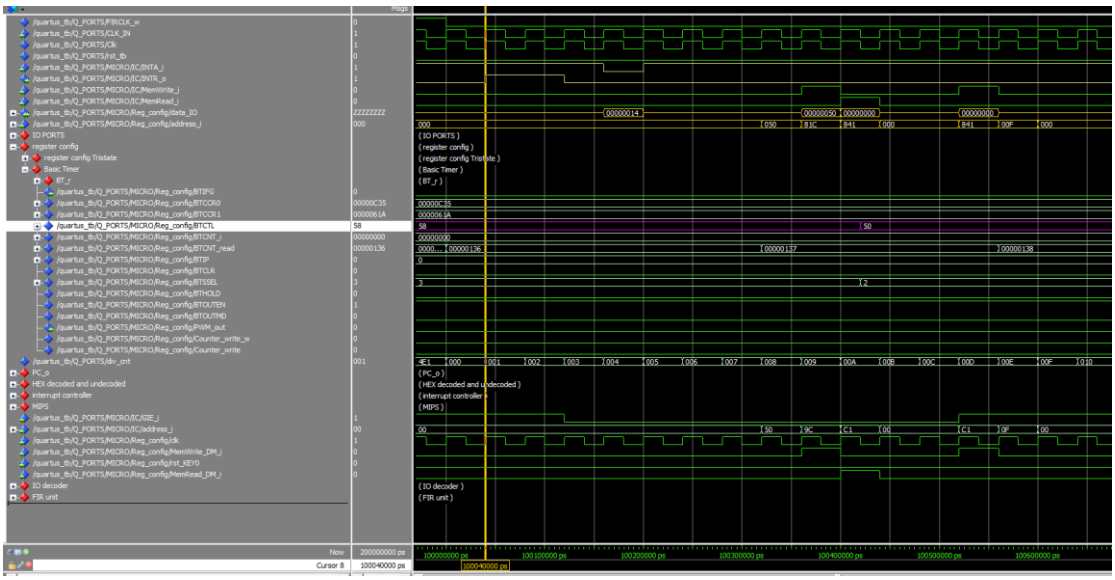
איור 22: תדר מקסימלי של המערכת

הדבר מסתדר עם תדרי העבודה שלנו כדרוש, כדי לקבל תדר יותר גבוה נוכל לפצל את הנתיב הקריטי שלנו לשני חלקים באמצעות רגיסטר שיפריד בין המודולים וכך נשיג תדר יותר מהיר לכל חלק.

Code Verification:

לצורך וריפיקציה והוכחת עבודה של הקוד נציג ModelSim ו-SignalTap של test2 מתוך קובץ בדיקות עם פסיקה שניתנו לנו בעבודה זו וגם את test2 מ-GPIO.

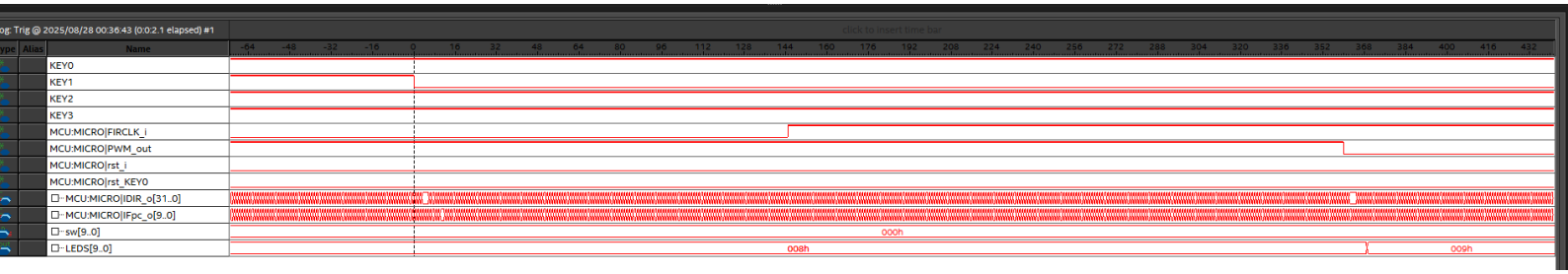
בבדיקת ה-ModelSim הכנסנו לחיצה על KEY1 לאחר 100us לכן נקפוץ לנקודה זו ונראה כיצד הדבר משפיע:



איור 23: תמונה מ-modelsim test2

ניתן לראות כי אכן מתקבלת בקשת פסיקה עבור כתובת 0x14 ברגע שINTA יורד – כלומר רוטינת השירות שתתקבל תהיה של KEY1 בהתאם למה שרצינו.
כמו כן ניתן לראות כי לאחר זמן מה משתנה ערך BTCTL מ-50 ל-58 כדרוש בטסט.

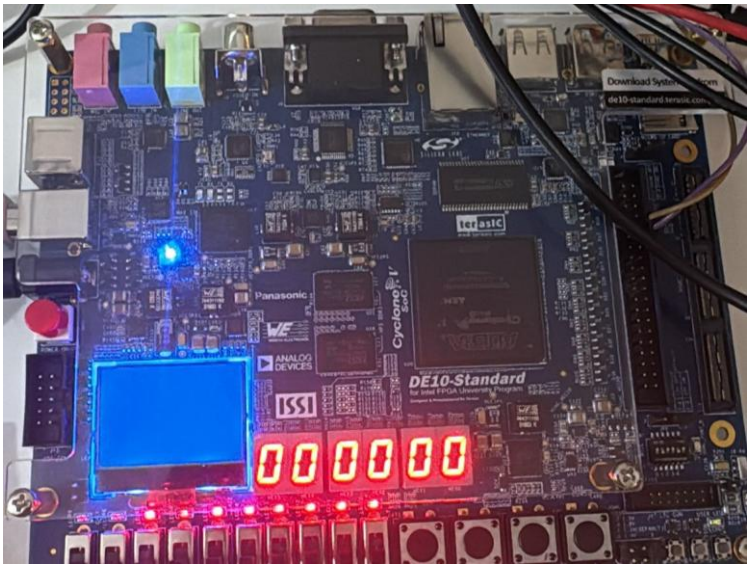
כעת נרצה להשוות את התוצאות לאלו שיתקבלו ב-SignalTap:



איור24 : טסט 2 interrupts-מSignalTap

בחרנו את KEY1 בתור טריגר ואכן ניתן לראות, באופן דומה ל-ModelSim כצפוי, כי קיבלנו INTR ובעת ירידת INTA אנו מעבירים דרך data_IO את ערך הפסיקה 14 ולאחריו מתחיל הטיפול בפסיקה ואנחנו רואים את הטיפול בפסיקה על ידי שינוי תדר ה-PWM.

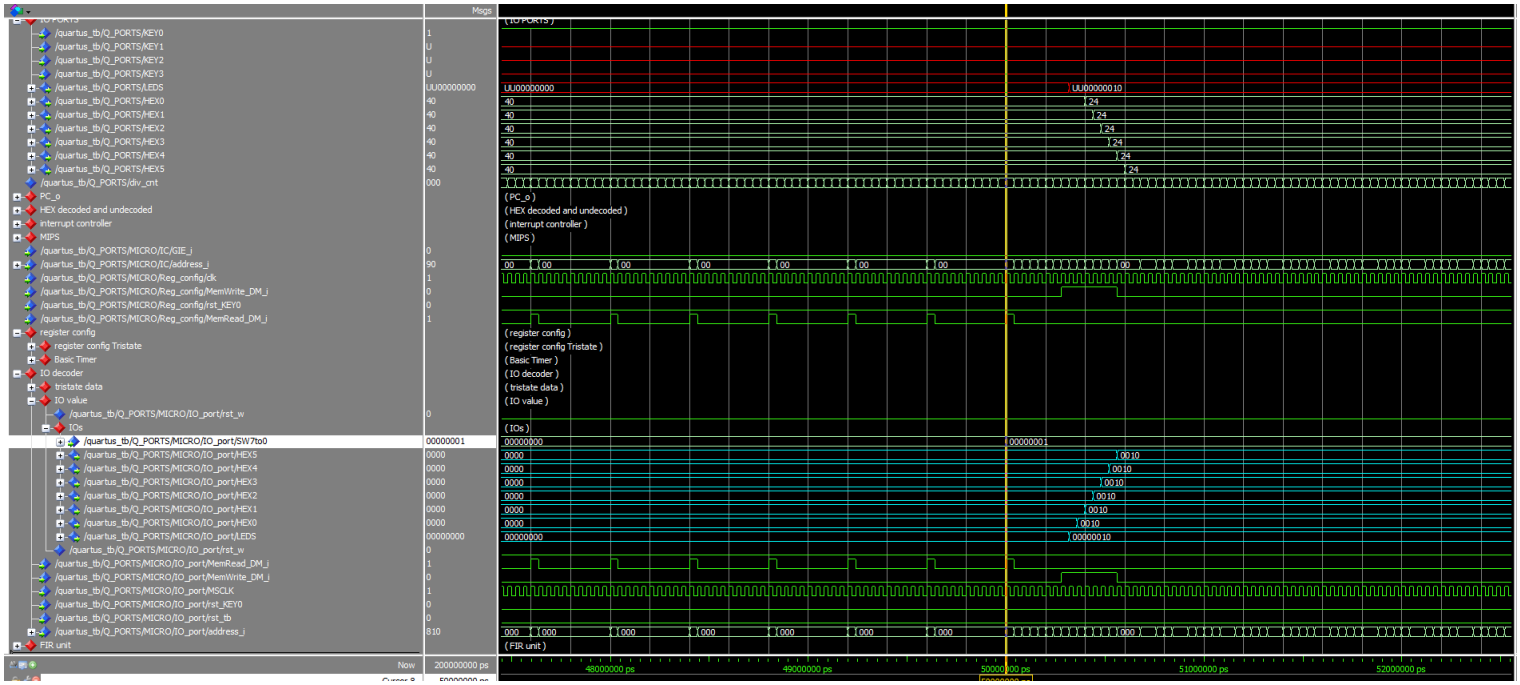
תמונה של ה-FPGA:



איור 25: תמונה של FPGA test2 Interrupt

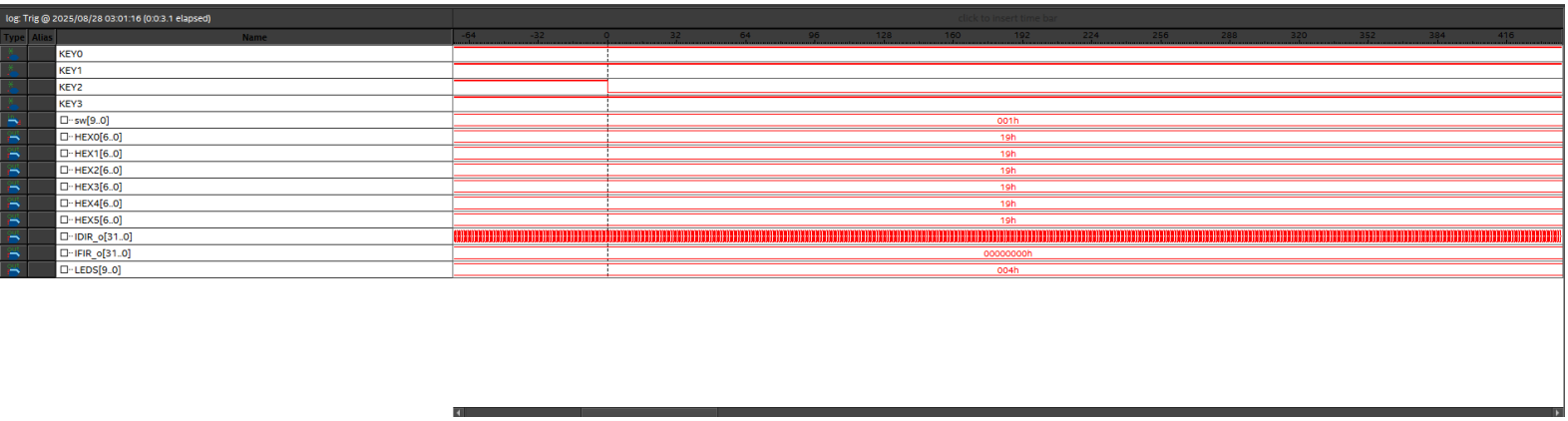
כעת לבדיקת קוד GPIO הרצנו את הקוד לדוגמה של test2.

בבדיקת ה-ModelSim הכנסנו לאחר 50us עלייה של SW1 ואכן קיבלנו כמבוקש כי אנחנו מתחילים בספירה למעלה והצגתו על ה-HEX ועל ה-LEDS:



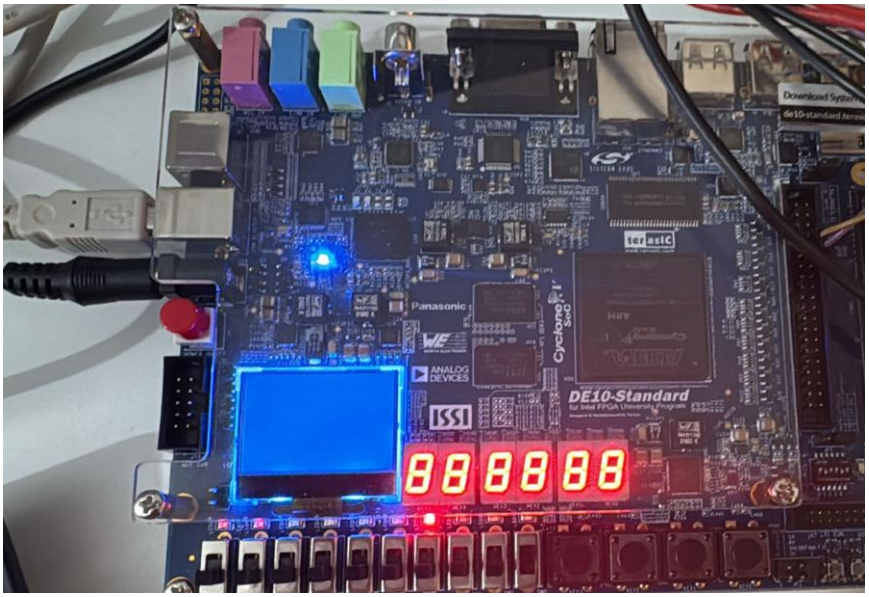
איור : 26 טסט 2 עבור GPIO על Modelsim

ובאותו אופן גם על גביי ה- FPGA אנו מקבלים את אותו התהליך:



איור:27 : טסט 2 GPIO

תמונה של ה-FPGA:



איור28 : תמונה של FPGA test2 GPIO