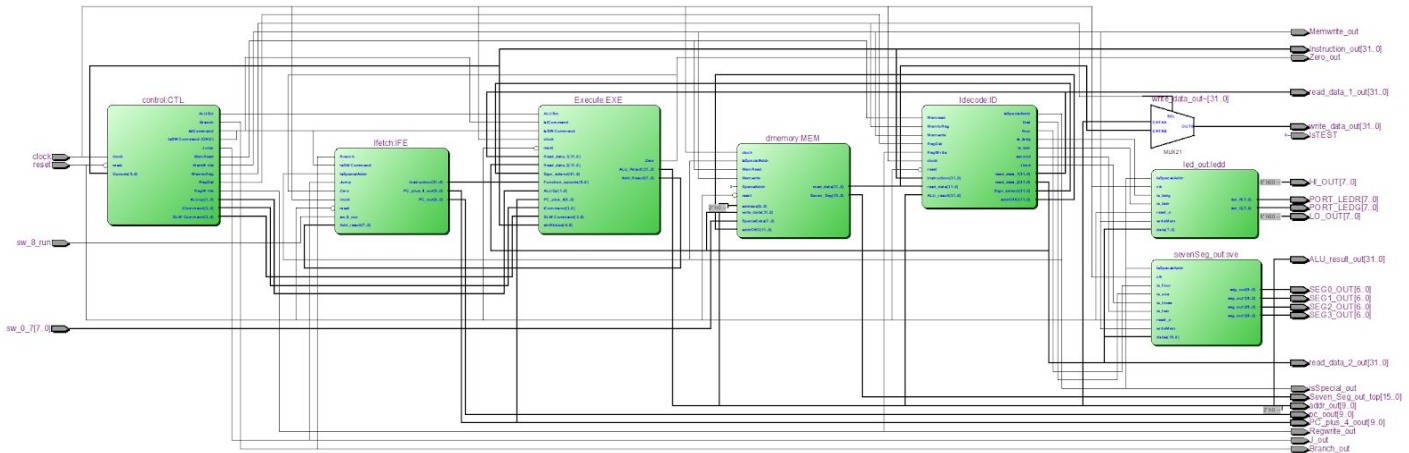


מטלה 3

ראשית נציג את דיאגרמה של כל המערכת, בהמשך נציג ונסביר על כל מודול בנפרד. נראה איך הוא בנוי ומה מטרתו בתוך המערכת הכוללת. כפי שניתן לראות מהדיאגרמה הכללית שמערכת עומד בכל הדרישות:



איור 1

פירוט ה-PORTS:

- Reset - אות כניסה בגודל ביט. על מנת לאתחל את המערכת.
- clock - אות כניסה בגודל של ביט. כניסה של השעון למערכת.
- IsTEST - אות מוצא בגודל של ביט על מנת לבצע בדיקות על המערכת
- sw_8_run - אות כניסה למערכת בגודל של 1. מטרתו לבצע בדיקות על המערכת
- Sw_0_7 - אות כניסה בגודל 8 ביט, כניסה של SWITCHES לתוך המערכת.
- HI_OUT - אות מוצא בגודל 8 ביט בשביל לוודא את ההמרות לפני הצגה על SEGMENTS.
- LO_OUT - אות מוצא בגודל 8 ביט בשביל לוודא את ההמרות לפני הצגה על SEGMENTS.
- addr_out - אות מוצא בגודל 9 ביט בשביל לבדוק כתובות בזמן הריצה.
- SEG2_OUT, SEG3_OUT, SEG0_OUT, SEG1_OUT - אות מוצא בגודל 7 ביט שמחובר ל7SEGMENT המתאים.
- PORT_LEDG, PORT_LEDR - אות מוצא בגודל 8 ביט שמחובר לLEDs המתאים.
- PC_plus_4_oout - אות מוצא בגודל 10 ביט, מטרתו לעזור בזמן הבדיקה של המערכת.
- isSpecial_out - אות מוצא בגודל של ביט, מטרתו לבדוק שמערכת מצליחה לזהות כתובות של IO.
- J_out - אות מוצא בגודל של ביט, מטרתו לזהות שהיתה קפיצה במערכת. עוזר בזמן הבדיקות של המערכת.
- Seven_Seg_out_top - אות מוצא בגודל של 16 ביט עלמנת לוודא תצוגה של 7 SEGMENT.
- pc_oout - אות מוצא בגודל 10 ביט, מטרתו לעזור בזמן הבדיקה של המערכת.
- ALU_result_out - אות מוצא בגודל 32 ביט, מטרתו לעזור בזמן הבדיקה שמוציא ערך של ALU
- read_data_1_out - אות מוצא בגודל 32 ביט, מטרתו לעזור בזמן הבדיקה.
- read_data_2_out - אות מוצא בגודל 32 ביט, מטרתו לעזור בזמן הבדיקה.

write_data_out-אות מוצא בגודל 32 ביט, מטרתו לעזור בזמן הבדיקה שמוציא ערך שנכתב לתוך הרגיסטר המתאים.

Branch_out- אות מוצא בגודל ביט שמסמנת האם היתה קפיצה או לא.

Zero_out- אות מוצא בגודל ביט שמסמנת האם היתה תוצאה של 0 בALU.

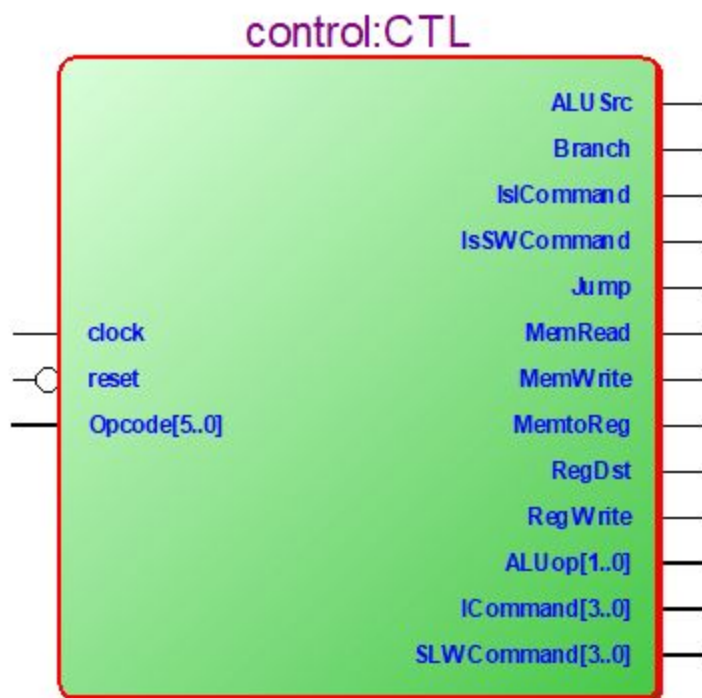
Memwrite_out-אות מוצא בגודל ביט שמסמנת האם כתבו לתוך הזיכרון או לא.

Regwrite_out-אות מוצא בגודל ביט שמסמנת האם כתבו לתוך רגיסטר או לא.

מודול CONTROL:

מודול שאחראי על סיגנלים של CONTROL בתוך MIPS בעצם מגדיר איזה חלק בתוך המעגל יעבוד או לא יעבוד וגם באיזה צורה.

RTL:



איור 2

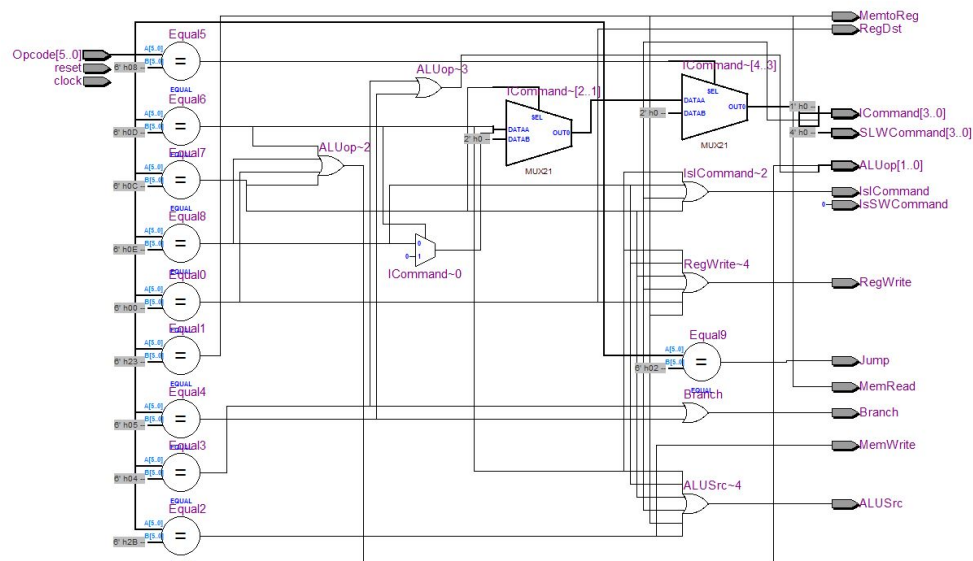
טבלת הסבר על הפורטים :

שם	גודל(ביט)	כיוון IN\OUT	מטרה
Opcode	6	IN	אות כניסה של OPCODE של הפקודה
RegDst	1	OUT	אות בקרה לתוך MUX המתאים שבוחר רגיסטר היעד.
ALUSrc	1	OUT	אות בקרה לתוך MUX המתאים, בחירה בין SIGNI READ DATA2 .EXTEND
MemtoReg	1	OUT	בחירה בין תוצאה של זכרון ותוצאה של ALU .RESULT
RegWrite	1	OUT	אות בקרה לתוך ה REGISTERS
MemRead	1	OUT	אות בקרה לקריאה מהזיכרון
MemWrite	1	OUT	אות בקרה בשביל כתיבה לתוך הזיכרון
Branch	1	OUT	אות בקרה לתוך הAND על מנת לבצע קפיצה
ALUOp	2	OUT	אותות בקרה לתוך ALU בעזרתן ניתן לקבוע מצב פעולה של ALU
Jump	1	OUT	אות בקרה עבור קפיצה
ICommand	4	OUT	סוג של פקודת I TYPE
SLWCommand	4	OUT	סוג של פקודת שמירה לזכרון או לIO
IsICommand	1	OUT	דגל האם I TYPE
IsSWCommand	1	OUT	דגל האם כתיבה או קריאה
clock	1	IN	שעון

איפוס ואיתחול	IN	1	reset
---------------	----	---	-------

טבלה 1

Logical Usage:



איור 3

בנוסף לטבלה 1, צירפנו תמונות מתוך ה-QUARTUS איורים 4,5,6. ניתן לראות את הפורטים וחייבורים שלהם למודולים האחרים:

Ports:

Port Name	
1	Opcode[5..0]
2	clock
3	reset

איור 4

Fan-In:

Input Port	Fan-in Node
clock	clock
reset	reset
Opcode[5..0]	Ifetch:IFE

איור 5

Fan-Out:

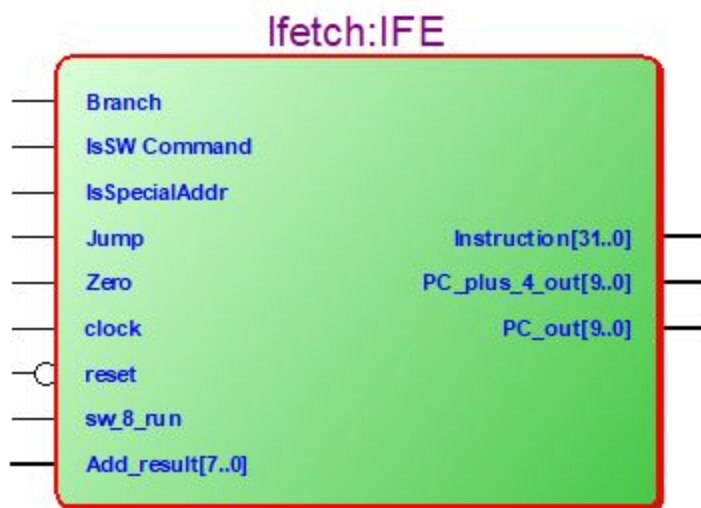
Output Port	Fan-out Node
ALUSrc	<input type="checkbox"/> Execute:EXE
Branch	<input checked="" type="checkbox"/> Branch_out
	<input type="checkbox"/> Ifetch:IFE
IsICommand	<input type="checkbox"/> Execute:EXE
IsSWCommand	<input type="checkbox"/> Ifetch:IFE
	<input type="checkbox"/> Execute:EXE
Jump	<input checked="" type="checkbox"/> J_out
	<input type="checkbox"/> Ifetch:IFE
MemRead	<input type="checkbox"/> Idecode:ID
	<input type="checkbox"/> dmemory:MEM
MemWrite	<input checked="" type="checkbox"/> Memwrite_out
	<input type="checkbox"/> Idecode:ID
	<input type="checkbox"/> dmemory:MEM
	<input type="checkbox"/> sevenSeg_out:sve
	<input type="checkbox"/> led_out:ledd
MemtoReg	<input checked="" type="checkbox"/> write_data_out~[31..0]
	<input type="checkbox"/> Idecode:ID
RegDst	<input type="checkbox"/> Idecode:ID
RegWrite	<input checked="" type="checkbox"/> Regwrite_out
	<input type="checkbox"/> Idecode:ID
ALUop[1..0]	<input type="checkbox"/> Execute:EXE
ICommand[3..0]	<input type="checkbox"/> Execute:EXE
SLWCommand[3..0]	<input type="checkbox"/> Execute:EXE

איור 6

מודול **IFETCH**:

מודול שאחראי על שליפת פקודה הבא לפי כתובת בNEXT_PC שתלוי בהתנהגות של תוכנית.

RTL:



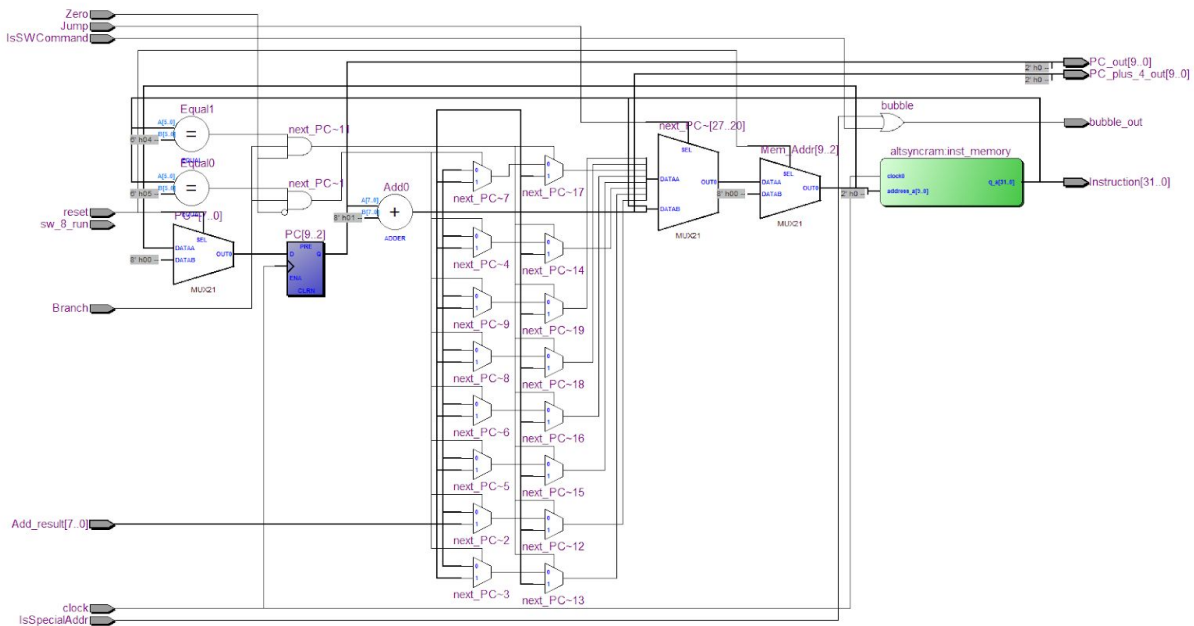
איור 7

טבלת הסבר על הפורטים :

שם	גודל(ביט)	כיוון IN\OUT	מטרה
Instruction	32	OUT	מוציא פקודה שצריך לבצע
PC_plus_4_out	10	OUT	נוציא אות בשביל לוודא שמערכת עובדת בצורה תקינה
Add_result	8	IN	כניסה של כתובת של קפיצה
IsSpecialAddr	1	IN	במידה ונצטרך להאט את PC
Branch	1	IN	כניסה שמסמן שפקודה הבא תהיה BRANCH
Zero	1	IN	כניסה שמסמנת שבALU היה ZERO
Jump	1	IN	כניסה -פקודה הבא קפיצה
bubble_out	1	OUT	להוציא NOP למערכת
IsSWCommand	1	IN	אם פקודה נוכחית כתיבה לקריאה מהזיכרון
sw_8_run	1	IN	בדיקה של המערכת
PC_out	10	OUT	בדיקה של המערכת
clock	1	IN	שעון
reset	1	IN	איפוס ואיתחול

טבלה 2

Logical Usage:



איור 8

בנוסף לטבלה 2, צירפנו תמונות מתוך ה-QUARTUS איורים 9,10,11. ניתן לראות את הפורטים וחיבורים שלהם למודולים אחרים:

Ports:

	Port Name
1	Add_result[7..0]
2	Branch
3	IsSWCommand
4	IsSpecialAddr
5	Jump
6	Zero
7	clock
8	reset
9	sw_8_run

איור 9

Fan-In:

Input Port	Fan-in Node
Branch	<input type="checkbox"/> control:CTL
IsSWCommand	<input type="checkbox"/> control:CTL
IsSpecialAddr	<input type="checkbox"/> Idecode:ID
Jump	<input type="checkbox"/> control:CTL
Zero	<input type="checkbox"/> Execute:EXE
clock	<input type="checkbox"/> clock
reset	<input type="checkbox"/> reset
sw_8_run	<input type="checkbox"/> sw_8_run
Add_result[7..0]	<input type="checkbox"/> Execute:EXE

איור 10

Fan-Out:

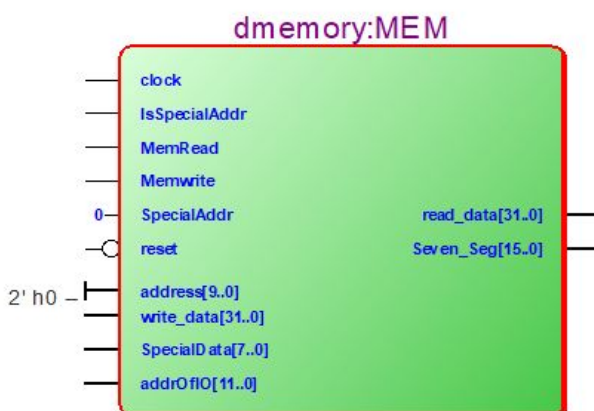
Output Port	Fan-out Node
Instruction[31..0]	<input type="checkbox"/> Execute:EXE
	<input type="checkbox"/> Idecode:ID
	<input type="checkbox"/> control:CTL
	<input type="checkbox"/> Instruction_out[31..0]
PC_plus_4_out[9..0]	<input type="checkbox"/> PC_plus_4_oout[9..0]
	<input type="checkbox"/> Execute:EXE
PC_out[9..0]	<input type="checkbox"/> pc_oout[9..0]

איור 11

מודול **DMEMORY**:

מודול שאחראי על ממשק עם הזיכרון. נותן יכולת כתיבה וקריאה מהזיכרון. ברגע שמגיע דגל כתובת IO או לא משתמשים בזכרון.

RTL:



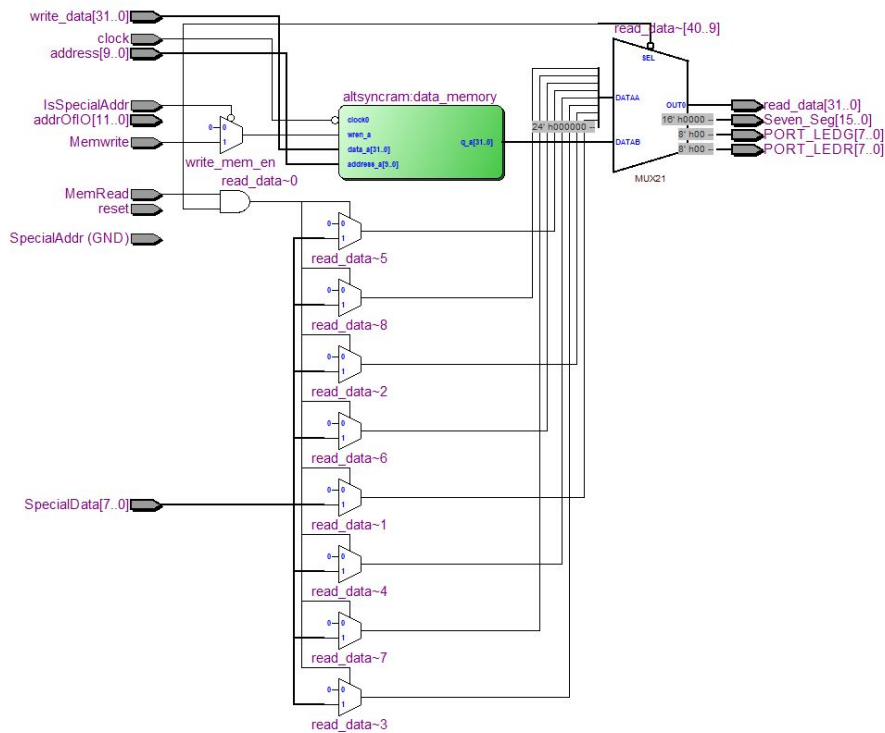
איור 12

טבלת הסבר על הפורטים :

שם	גודל(ביט)	כיוון IN\OUT	מטרה
read_data	32	OUT	מוציא דאטא הנראה מהזיכרון
address	10	IN	כניסת כתובת עבור קריאה מהזיכרון
write_data	32	IN	כניסה של דאטא לכתובת לזכרון
MemRead	1	IN	כניסת אות בקרה
Memwrite	1	IN	כניסת אות בקרה
SpecialAddr	1	IN	כניסת אות בקרה
SpecialData	8	IN	מה לכתוב לתוך מתוך SWITCHES
IsSpecialAddr	1	IN	כניסת אות בקרה
Seven_Seg	16	OUT	מוצא של אות עבור בדיקת המערכת
addrOfIO	12	IN	כניסה עבור כתובת מיוחדת
PORT_LEDG	8	OUT	מוצא עבור LEDS בשביל לבדוק תקינות המערכת
PORT_LEDR	8	OUT	מוצא עבור LEDS בשביל לבדוק תקינות המערכת
clock	1	IN	שעון
reset	1	IN	איפוס ואתחול

טבלה 3

Logical Usage:



איור 13

בנוסף לטבלה 3, צירפנו תמונות מתוך ה-QUARTUS איורים 14, 15, 16. ניתן לראות את הפורטים וחייבורים שלהם למודולים אחרים:

Ports:

	Port Name
1	IsSpecialAddr
2	MemRead
3	Memwrite
4	SpecialAddr
5	SpecialData[7..0]
6	addrOfIO[11..0]
7	address[9..0]
8	clock
9	reset
10	write_data[31..0]

איור 14

Fan-In:

Input Port	Fan-in Node
clock	clock
IsSpecialAddr	Idecode:ID
MemRead	control:CTL
Memwrite	control:CTL
SpecialAddr	0
reset	reset
address[9..0]	
[9..2]	Execute:EXE
[1..0]	2' h0
write_data[31..0]	Idecode:ID
SpecialData[7..0]	sw_0_7[7..0]
addrOfIO[11..0]	Idecode:ID

איור 15

Fan-Out:

Output Port	Fan-out Node
read_data[31..0]	write_data_out~[31..0]
	Idecode:ID
Seven_Seg[15..0]	Seven_Seg_out_top[15..0]

איור 16

מודול **IDCODE**:

מודול שמטרתו לפענח את הפקודה, להבין האם מדובר בפניה לIO או לא, מצבע קריאה מתוך REGISTERS ומבצע SIGN EXTEND ומכין את דאטא לפני שלב EXE.

RTL:



איור 17

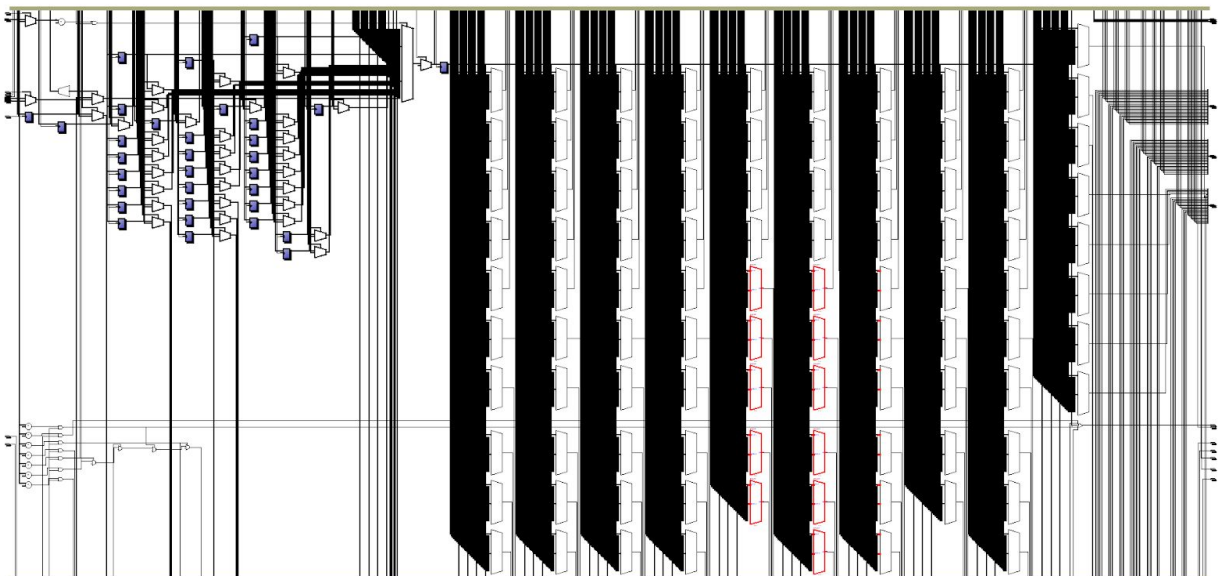
טבלת הסבר על הפורטים :

שם	גודל(ביט)	כיוון IN\OUT	מטרה
read_data_1	32	OUT	מוצא של REGISTERS, ערך של רגיסטר המתאים
read_data_2	32	OUT	מוצא של REGISTERS, ערך של רגיסטר המתאים
wrReg_out	32	OUT	מוצא של אות בקרה עבור בדיקות
Instruction	32	IN	כניסה של פקודה נוכחית
read_data	32	IN	כניסה עבור דאטא שתיכתב לתוך הרגיסטר המתאים
ALU_result	32	IN	כניסה עבור דאטא שתיכתב לתוך הרגיסטר המתאים
RegWrite	1	IN	כניסה עבור סיגנל בקרה על מנת לכתוב בחזרה לתוך הרגיסטר
MemtoReg	1	IN	כניסה לאות בקרה על מנת להחליט מה לכתוב לתוך הרגיסטר
RegDst	1	IN	אות בקרה על מנת לכתוב בחזרה לרגיסטר
Memwrite	1	IN	כניסה עבור אות בקרה על מנת לזהות כתיבה לכתובת IO
Memread	1	IN	כניסה עבור אות בקרה על מנת לזהות קריאה מ IO
Sign_extend	32	OUT	יציאה עבור תוצאה של הרכבת סימן
IsSpecialAddr	1	OUT	יציאה עבור אות זיהוי של כתובת IO
first	1	OUT	יציאה עבור אות בקרה

של 7SEG הראשון			
יציאה עבור אות בקרה של 7SEG השני	OUT	1	second
יציאה עבור אות בקרה של 7SEG השלישי	OUT	1	third
יציאה עבור אות בקרה של 7SEG הרביעי	OUT	1	four
יציאה עבור אות בקרה של LED הראשון	OUT	1	is_ledr
יציאה עבור אות בקרה של LED השני	OUT	1	is_ledg
יציאה עבור כתובת מיוחדת	OUT	32	addrOfIO
שעון	IN	1	clock
איפוס ואיתחול	IN	1	reset

טבלה 4

Logical Usage:



איור 18

בנוסף לטבלה 4, צירפנו תמונות מתוך ה-QUARTUS איורים 19, 20, 21. ניתן לראות את הפורטים וחיבורים שלהם למודולים אחרים:

Ports:

	Port Name /
1	ALU_result[31..0]
2	Instruction[31..0]
3	Memread
4	MemtoReg
5	Memwrite
6	RegDst
7	RegWrite
8	clock
9	read_data[31..0]
10	reset

איור 19

Fan-In:

Input Port	Fan-in Node
Memread	<input checked="" type="checkbox"/> control:CTL
MemtoReg	<input checked="" type="checkbox"/> control:CTL
Memwrite	<input checked="" type="checkbox"/> control:CTL
RegDst	<input checked="" type="checkbox"/> control:CTL
RegWrite	<input checked="" type="checkbox"/> control:CTL
clock	<input checked="" type="checkbox"/> clock
reset	<input checked="" type="checkbox"/> reset
Instruction[31..0]	<input checked="" type="checkbox"/> Ifetch:IFE
read_data[31..0]	<input checked="" type="checkbox"/> dmemory:MEM
ALU_result[31..0]	<input checked="" type="checkbox"/> Execute:EXE

איור 20

Fan-Out:

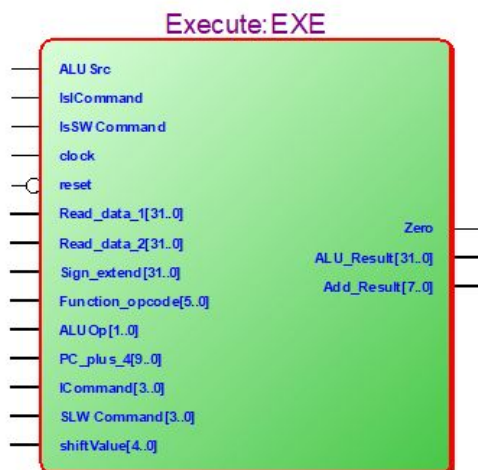
Output Port	Fan-out Node
IsSpecialAddr	<input checked="" type="checkbox"/> isSpecial_out
	<input type="checkbox"/> Ifetch:IFE
	<input type="checkbox"/> dmemory:MEM
	<input type="checkbox"/> sevenSeg_out:sve
	<input type="checkbox"/> led_out:ledd
first	<input type="checkbox"/> sevenSeg_out:sve
four	<input type="checkbox"/> sevenSeg_out:sve
is_ledg	<input type="checkbox"/> led_out:ledd
is_ledr	<input type="checkbox"/> led_out:ledd
second	<input type="checkbox"/> sevenSeg_out:sve
third	<input type="checkbox"/> sevenSeg_out:sve
read_data_1[31..0]	<input checked="" type="checkbox"/> read_data_1_out[31..0]
	<input type="checkbox"/> Execute:EXE
read_data_2[31..0]	<input checked="" type="checkbox"/> read_data_2_out[31..0]
	<input type="checkbox"/> Execute:EXE
	<input type="checkbox"/> dmemory:MEM
	<input type="checkbox"/> led_out:ledd
	<input type="checkbox"/> sevenSeg_out:sve
Sign_extend[31..0]	<input type="checkbox"/> Execute:EXE
addrOfIO[11..0]	<input type="checkbox"/> dmemory:MEM

איור 21

מודול **EXECUTE**:

יחידת חישוב של המערכת. ALU נמצא בתוך המודול הנוכחי.

RTL:



איור 22

טבלת הסבר על הפורטים :

שם	גודל(ביט)	כיוון IN\OUT	מטרה
Read_data_1	32	IN	כניסה עבור כניסת A לתוך הALU
Read_data_2	32	IN	כניסה עבור כניסת B לתוך הALU
Sign_extend	32	IN	כניסה עבור דאטא שעבר SIGN EXTENTION
Function_opcode	6	IN	כניסה עבור OPCODE של הפקודה
ALUOp	2	IN	כניסה עבור ALUOP
ALUSrc	1	IN	כניסה עבור אות בקרה שבוחר מה2
Zero	1	OUT	יציאה עבור אות שמשווא בין 2 כניסות של ALU
ALU_Result	32	OUT	יציאה עבור ALURESULT
Add_Result	8	OUT	יציאה עבור ADD_RESULT
PC_plus_4	10	IN	כניסה עבור אות של P ועוד 4
ICommand	4	IN	כניסה עבור אות שמביא OPCODE של ITYPE פקודות
SLWCommand	4	IN	כניסה עבור פקודות של שמירה
IsICommand	1	IN	כניסה עבור אות בקרה שאכן התקבלה פקודה מסוג I
IsSWCommand	1	IN	כניסה עבור אות בקרה שאכן התקבלה פקודה מסוג SW
shiftValue	5	IN	כניסה עבור ערך של הזזה

שעון	IN	1	clock
איפוס ואיתחול	IN	1	reset

טבלה 5


Logical Usage:



איור 23















בנוסף לטבלה 5, צירפנו תמונות מתוך ה-QUARTUS איורים 24,25,26. ניתן לראות את הפורטים וחיבורים שלהם למודולים אחרים:

Ports:

	Port Name 
1	ALUOp[1..0]
2	ALUSrc
3	Function_opcode[5..0]
4	ICommand[3..0]
5	IsICommand
6	IsSWCommand
7	PC_plus_4[9..0]
8	Read_data_1[31..0]
9	Read_data_2[31..0]
10	SLWCommand[3..0]
11	Sign_extend[31..0]
12	clock
13	reset
14	shiftValue[4..0]

איור 24

Fan-in:

Input Port	Fan-in Node
ALUSrc	 control:CTL
IsICommand	 control:CTL
IsSWCommand	 control:CTL
clock	 clock
reset	 reset
Read_data_1[31..0]	 Idecode:ID
Read_data_2[31..0]	 Idecode:ID
Sign_extend[31..0]	 Idecode:ID
Function_opcode[5..0]	 Ifetch:IFE
ALUOp[1..0]	 control:CTL
PC_plus_4[9..0]	 Ifetch:IFE
ICommand[3..0]	 control:CTL
SLWCommand[3..0]	 control:CTL
shiftValue[4..0]	 Ifetch:IFE

איור 25

Fan-out:

Output Port	Fan-out Node
Zero	Zero_out
	Ifetch:IFE
ALU_Result[31..0]	write_data_out~[31..0]
	ALU_result_out[31..0]
	addr_out[9..0]
	dmemory:MEM
	Idecode:ID
Add_Result[7..0]	Ifetch:IFE

איור 26

מודול sevenSeg_out:

מודול שממיר מבינארי לHEX ומציג את המספר על 7SEGMENT המתאים.

RTL:



איור 27

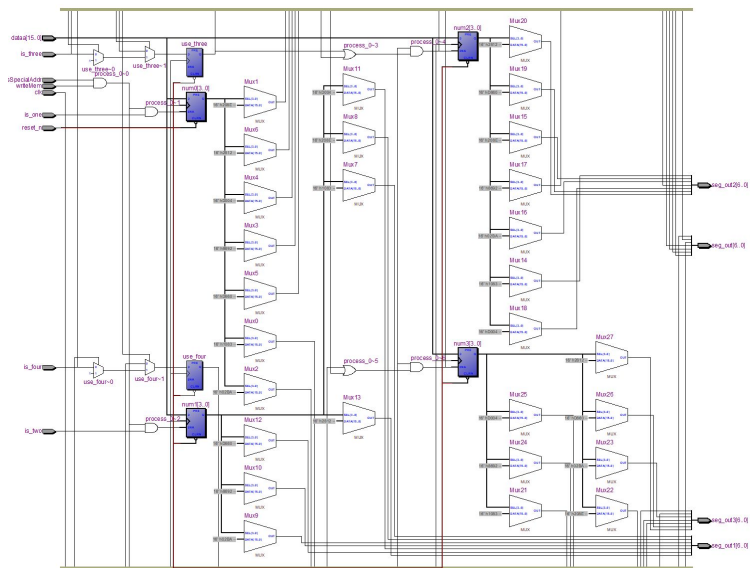
טבלת הסבר על הפורטים :

שם	גודל(ביט)	כיוון IN\OUT	מטרה
clk	1	IN	שעון
reset_n	1	IN	איפוס ואיתחול
is_one	1	IN	כניסה עבור אות בקרה שמטרתו לסמן האם לכתוב ל7SEG הראשון

כניסה עבור אות בקרה שמטרתו לסמן האם לכתוב ל7SEG השני	IN	1	is_two
כניסה עבור אות בקרה שמטרתו לסמן האם לכתוב ל7SEG השלישי	IN	1	is_three
כניסה עבור אות בקרה שמטרתו לסמן האם לכתוב ל7SEG הרביעי	IN	1	is_four
כניסה עבור אות בקרה שמסמן האם IO	IN	1	IsSpecialAddr
כניסה עבור אות בקרה שמסמן האם כתיבה לזכרון	IN	1	writeMem
כניסה עבור מידע שצריך לכתוב	IN	16	dataa
יציאה שמחוברת ל7seg	OUT	7	seg_out
יציאה שמחוברת ל7seg	OUT	7	seg_out1
יציאה שמחוברת ל7seg	OUT	7	seg_out2
יציאה שמחוברת ל7seg	OUT	7	seg_out3

טבלה 6

Logical Usage:



איור 28

בנוסף לטבלה 6, צירפנו תמונות מתוך ה-QUARTUS איורים 29,30,31. ניתן לראות את הפורטים וחיבורים שלהם למודולים אחרים:

Ports:

	Port Name /
1	IsSpecialAddr
2	clk
3	dataa[15..0]
4	is_four
5	is_one
6	is_three
7	is_two
8	reset_n
9	writeMem

איור 29

Fan-In:

Input Port	Fan-in Node
IsSpecialAddr	<input type="checkbox"/> Idecode:ID
clk	<input checked="" type="checkbox"/> clock
is_four	<input type="checkbox"/> Idecode:ID
is_one	<input type="checkbox"/> Idecode:ID
is_three	<input type="checkbox"/> Idecode:ID
is_two	<input type="checkbox"/> Idecode:ID
reset_n	<input checked="" type="checkbox"/> reset
writeMem	<input type="checkbox"/> control:CTL
dataa[15..0]	<input type="checkbox"/> Idecode:ID

איור 30

Fan-Out:

Output Port	Fan-out Node
seg_out[6..0]	<input checked="" type="checkbox"/> SEG0_OUT[6..0]
seg_out1[6..0]	<input checked="" type="checkbox"/> SEG1_OUT[6..0]
seg_out2[6..0]	<input checked="" type="checkbox"/> SEG2_OUT[6..0]
seg_out3[6..0]	<input checked="" type="checkbox"/> SEG3_OUT[6..0]

איור 31

מודול led_out

מודול שאחראי על להציג על LEDS

RTL:



איור 32

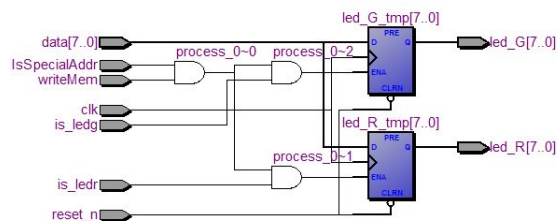
טבלת הסבר על הפורטים :

שם	גודל(ביט)	כיוון IN\OUT	מטרה
clk	1	IN	שעון

איפוס ואיתחול	IN	1	reset_n
כניסה עבור אות בקרה שמסמן את LEDs האדומים	IN	1	is_ledr
כניסה עבור אות בקרה שמסמן את LED הירוקים	IN	1	is_ledg
כניסה עבור אות בקרה שמסמן את כתובת IO	IN	1	IsSpecialAddr
כניסה עבור אות בקרה שמסמן את כתיבה	IN	1	writeMem
כניסה עבור מידע	IN	16	data
יציאה שמחוברת ישירות לLEDs	OUT	8	led_R
יציאה שמחוברת ישירות לLEDs	OUT	8	led_G

טבלה 7

Logical Usage:



איור 33

בנוסף לטבלה 7, צירפנו תמונות מתוך ה-QUARTUS איורים 34, 35, 36. ניתן לראות את הפורטים וחיבורים שלהם למודולים אחרים:

Ports:

	Port Name /
1	IsSpecialAddr
2	clk
3	data[7..0]
4	is_ledg
5	is_ledr
6	reset_n
7	writeMem

איור 34

Fan-In:

Input Port	Fan-in Node
IsSpecialAddr	<input type="checkbox"/> Idecode:ID
clk	<input checked="" type="checkbox"/> clock
is_ledg	<input type="checkbox"/> Idecode:ID
is_ledr	<input type="checkbox"/> Idecode:ID
reset_n	<input checked="" type="checkbox"/> reset
writeMem	<input type="checkbox"/> control:CTL
data[7..0]	<input type="checkbox"/> Idecode:ID

איור 35

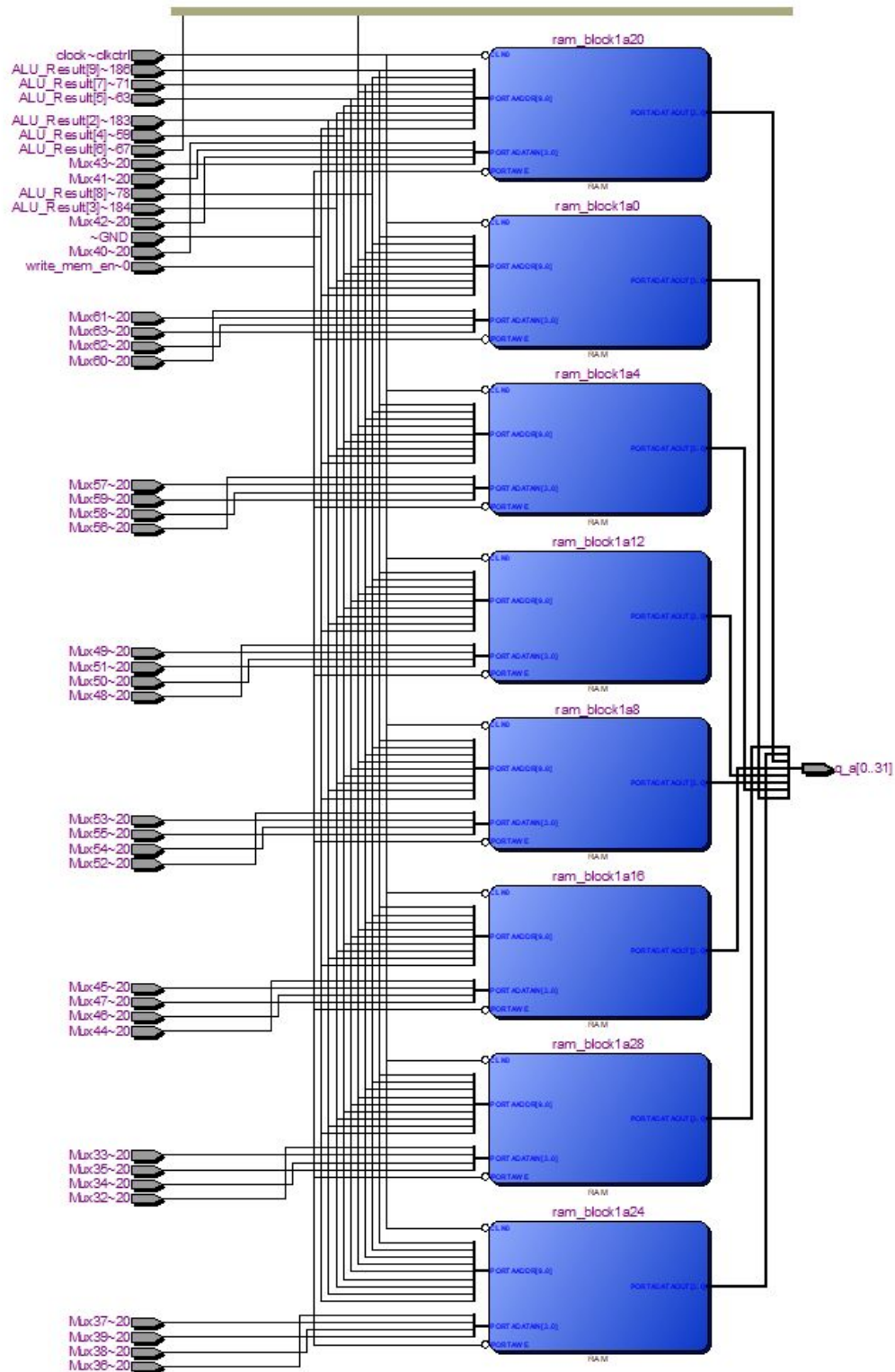
Fan-Out:

Output Port	Fan-out Node
led_R[7..0]	<input checked="" type="checkbox"/> PORT_LEDR[7..0]
led_G[7..0]	<input checked="" type="checkbox"/> PORT_LEDG[7..0]

איור 36

מסלול הכי ארוך:

כפי שנלמד בכיתה וגם מה שראינו בזמן התרגיל הנוכחי, מסלול הכי ארוך מתקבל כאשר מבצעים פקודה של קריאה מהזכרון וכתיבת ערך לתוך הרגיסטר. BOTTLENECK במסלול הנוכחי זה גישה לזכרון RAM:



איור 37

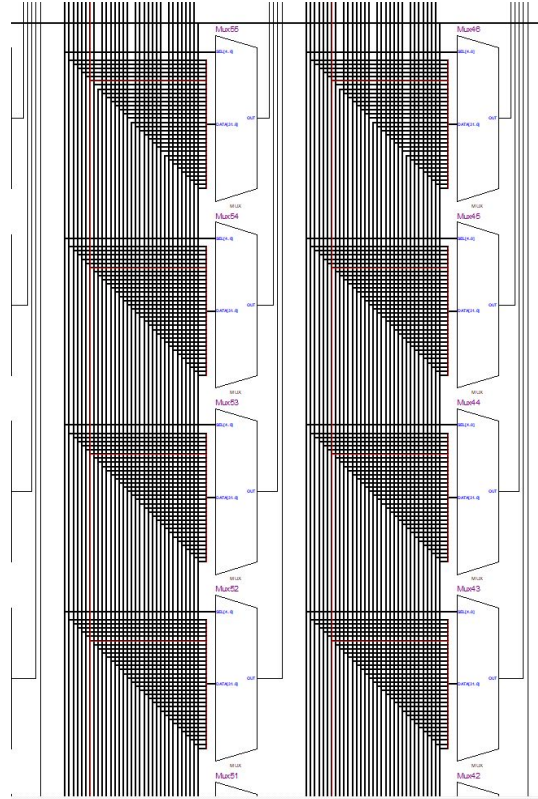
```

data_memory : altsyncram
  GENERIC MAP (
    operation_mode      => "SINGLE_PORT",
    width_a              => 32,
    widthad_a           => 10,
    lpm_type             => "altsyncram",
    outdata_reg_a        => "UNREGISTERED",
    init_file            => "dmemory.hex",
    intended_device_family => "Cyclone"
  )
  PORT MAP (
    wren_a      => write_mem_en,
    clock0      => write_clock,
    address_a   => address,
    data_a      => write_data,
    q_a         => mem_read_data
  );

```

איור 38

חלק הזה קיבלנו לכן לא ביכולתינו לשנות אותו. מה שכן בתכנון שלנו היינו משפרים את השלב של DECODE. בתוך בשלב של DECODE אנו בודקים האם כתובת היא של IO ומשווים ממש ביט ביט על מנת לוודא. דבר כזה מייצר הרבה לוגיקה מיותר וגם תופס מקום על הציפ:



איור 39

אם היה לנו יותר זמן אז כנראה היינו חושבים על דרך הרבה יותר יעילה.

```
first_t <= '1' WHEN Instruction(11 DOWNTO 0) = "100000001000" AND
Memwrite = '1' ELSE '0'; --PORT_HEX0[7-0] 0x808
second_t <= '1' WHEN Instruction(11 DOWNTO 0) = "100000001100" AND
Memwrite = '1' ELSE '0'; --PORT_HEX1[7-0] 0x80c
third_t <= '1' WHEN Instruction(11 DOWNTO 0) = "100000010000" AND
Memwrite = '1' ELSE '0'; --PORT_HEX2[7-0] 0x810
four_t <= '1' WHEN Instruction(11 DOWNTO 0) = "100000010100" AND
Memwrite = '1' ELSE '0'; --PORT_HEX3[7-0] 0x814
is_ledg_t <= '1' WHEN Instruction(11 DOWNTO 0) = "100000000000" AND
Memwrite = '1' ELSE '0'; --PORT_ledg 0x800
is_ledr_t <= '1' WHEN Instruction(11 DOWNTO 0) = "100000000100" AND
Memwrite = '1' ELSE '0'; --PORT_ledr 0x804
switch_input <= '1' WHEN Instruction(11 DOWNTO 0) = "100000011000" AND
Memread = '1' ELSE '0'; --PORT_SW[7-0] 0x818
```

איור 39

תדר השעון :

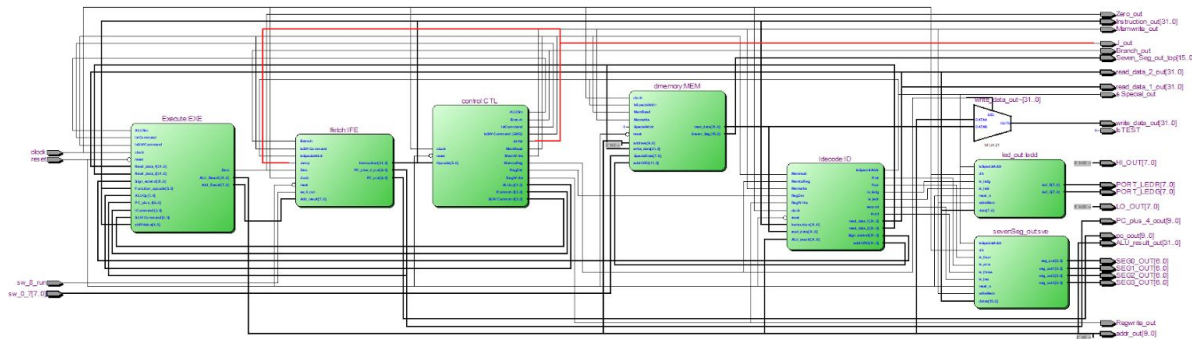
תכננו את המעגל לפי הדרישות של התרגיל כך שתדר השעון יהיה קרוב כמה שיותר ל24MHz:

	Clock Name	Type	Period	Frequency
1	clock	Base	41.000	24.39 MHz

איור 40

מסלול הקצר ביותר:

מסלול הכי קצר בתכנון שלנו קורא כאשר מתבצעת פקודה של J. אנו תכננו כך שכל החישוב מתבצעת בשלב של IFETCH. ניתן לראות את המסלול הזה על איור 41.



איור 41

בנוסף חישוב זה ניתן לראות בקוד VHDL הבא:

```
J_PC(5 DOWNTO 0) <= Inst(5 DOWNTO 0);
J_PC(7 DOWNTO 6) <= PC_plus_4(9 DOWNTO 8);
-- Mux to select Branch Address or PC + 4
Next_PC <= X"00" WHEN Reset = '1' ELSE
  --PC(9 DOWNTO 2) WHEN bubble = '1' ELSE
  J_PC(7 DOWNTO 0) WHEN Jump = '1' ELSE
  Add_result WHEN ((Branch = '1') AND (Zero = '1') AND (Inst(31
DOWNTO 26) = "000100")) ELSE
  Add_result WHEN ((Branch = '1') AND (Zero = '0') AND (Inst(31
DOWNTO 26) = "000101")) ELSE
  PC_plus_4(9 DOWNTO 2);
```

איור 42

הוכחה שמערכת עובדת:

ראשית נראה ב-SINGLE TAP של קוד הבא:

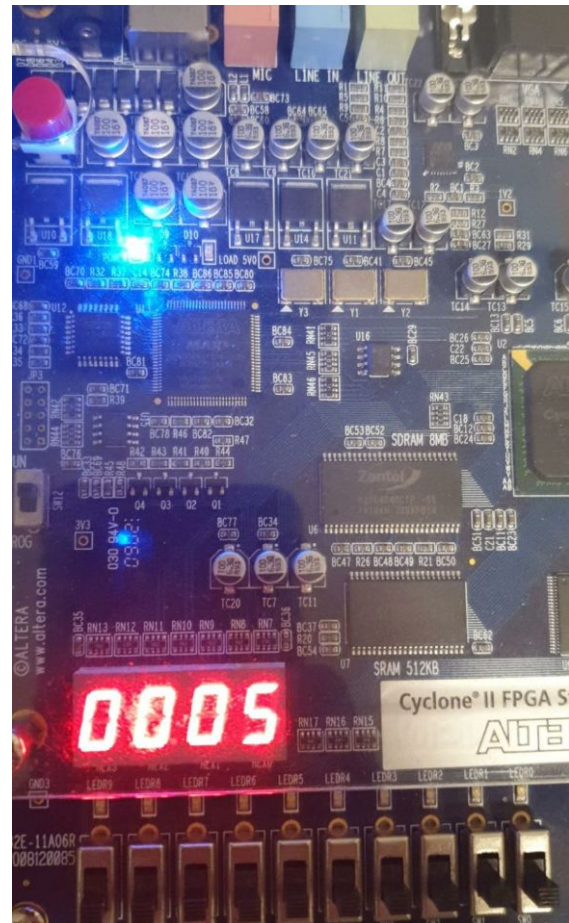
```
data.
N: .word 3000000
a: .word 5
text.
lw $t3, a
sw $t3, 0x808
```

כפי שניתן לראות קוד הזה כולל פקודה שקובעת את התדר השעון. באיור 43 ניתן לראות את WAVES על הצ'יפ. באיור הזה ניתן לראות שטוענים ערך ב-OFFSET של 4 (ALU_RESULT) ב-DATA שזה a ולאחר שקראו את הערך שומרים את מספר 5 בתוך הרגיסטר (WRITE_DATA). לאחר טעינת ערך לתוך הרגיסטר שומרים אותו לתוך IO שבכתובת 0x808. לפי איור 41 מובן שהמערכת עובדת.

out	ALU_result_out	00000000h	00000000h	00000004h	00000808h
out	Memwrite_out	0			
out	read_data_1_out	00000000h			00000000h
out	read_data_2_out	00000000h	00000000h		00000005h
out	Regwrite_out	1			
out	write_data_out	00000000h	00000000h	00000005h	00000808h
out	Zero_out	1			
in	sw_0_7	00h			00h
out	Instruction_out	00000000h	00000000h	8C0B0004h	AC0B0808h
out	addr_out	000h	000h	004h	008h

איור 43

באיור 44 ניתן לראות שאכן מקבלים ערך 5 שנטען לתוך הרגיסטר t3\$:



איור 44

סיכום:

במהלך המטלה למדנו איך לממש את התיאוריה שרכשנו בהרצאות במהלך הסמסטר הנוכחי. אנו למדנו איך לממש מעבד MIPS בשפת VHDL עם סט פקודות נרחב, דבר זה דרש להבין לעומק איך הדברים עובדים בפועל. חלק מהמשימה היתה דרישה להוסיף יכולת כתיבה וקריאה מהתקני IO ובנוסף לזה לעמוד בדרישת התדר של השעון. לקח לנו לא מעט זמן להגיע לתדר המבוקש. על מנת לבדוק את הפרויקט שלנו, אנו הרצו אלגוריתם מיון ודוגמא שקיבלנו ואכן הצלחנו להגיע לתוצאות שציפינו לקבל.

חשוב להגיד שבגלל קוצר הזמן לא מימשנו אופטימיזציות שנלמדו בכיתה על מעבד MIPS. בנוסף חלק מהמימושים שלנו לא הכי יעילים אף כולם עובדים.

אנחנו למדנו המון ממימוש של מעבד MIPS וגם לא פחות חשוב שלמדנו לעומק איך להשתמש בתוכנות כמו MODELSIM וQUARTUS שבטוח יתרום לנו לעתיד.

לסיכום הצלחנו לעמוד בכל הדרישות של המטלה ושמחנו לראות שקוד שלנו עבר כל הבדיקות.