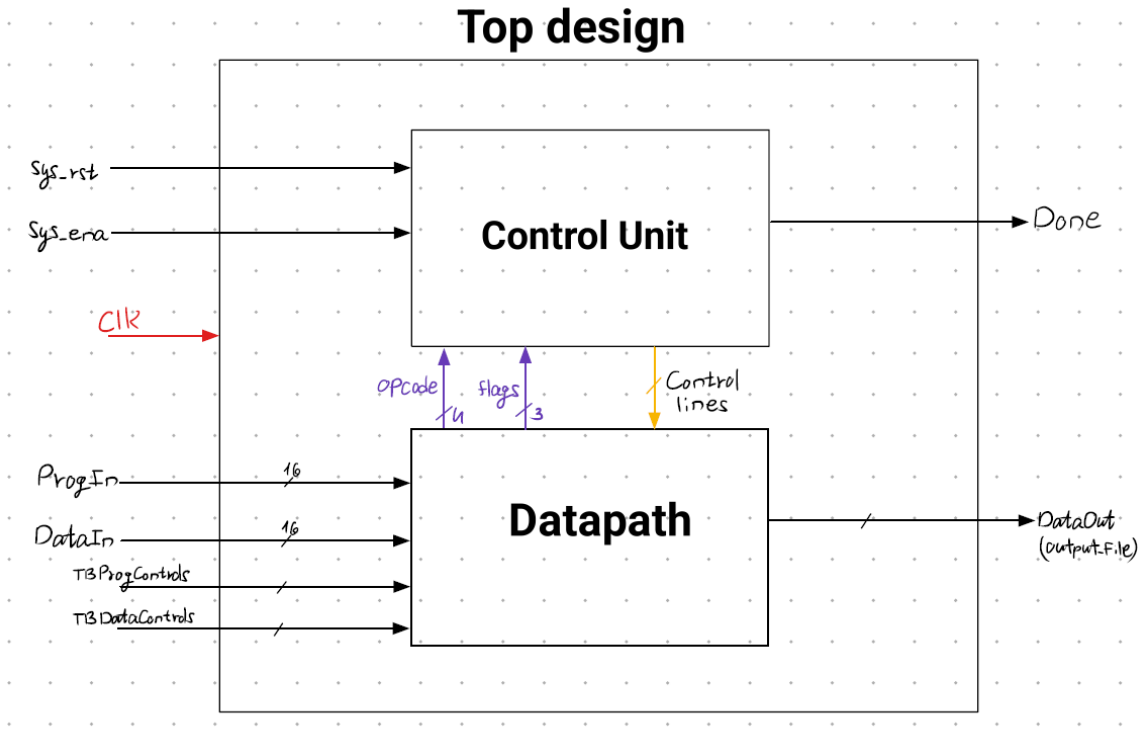
Preparation Report – Lab 3

DIGITAL SYSTEM DESIGN WITH VHDL (MULTI-CYCLE CPU DESIGN)

Dolev Eisenberg (208212845), Amir Aboutboul (318931797) | 24.6.2024

Top Design Layout



מצורף שרטוט של כלל המעבד ממבט על.

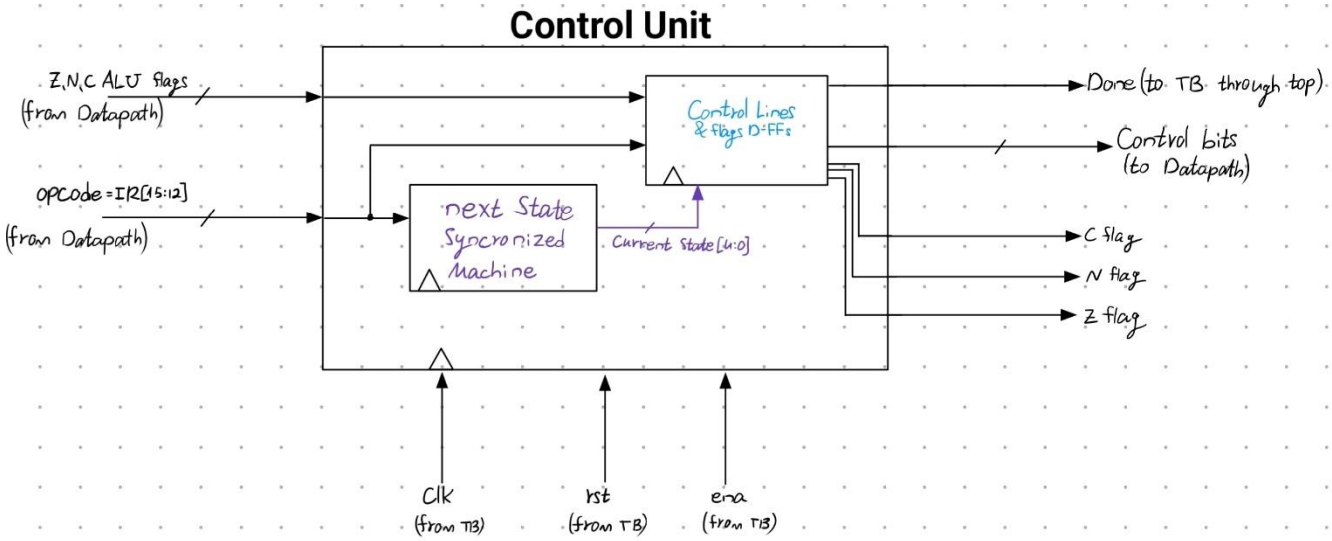
<u>למעבד הכניסות הבאות:</u>

- **Sys_rst, Sys_ena** – משמש לביצוע פעולת reset למעבד (לטובת אתחול לפני הפעלה), ולהפעלת המעבד (enable) לאחר סיום טעינת הזיכרון (Prog Memory, Data Memory) על ידי ה-TB.
- **ProgIn, DataIn** – קווים המשמשים לטעינת המידע לזיכרון ה-ProgMem וה-DataMem, בהתאמה, לתוך המעבד.
- קווי בקרה השולטים על הכנסת המידע אל הזיכרונות הנ"ל – כוללים בתוכם קו המשמש בתור הכתובות אליהן מכניסים בתוך יח' הזיכרון וביטי בקרה לטובת ביצוע פעולות אלו.
- כניסת שעון (**clk**) המשמש את היחידות הסינכרוניות בתוך המעבד.

<u>והמוצאים הבאים:</u>

- **Done** – ביט המשמש בתור אינדיקציה ל-TB לכך שהמעבד סיים את עבודתו (סיים לבצע את התוכנה שהוטענה אל הזיכרון שלו).
- **Data_Out** – משמש להוצאת המידע מתוך ה-Data Memory אל קובץ output. ההעברה מתוך המעבד אל הקובץ החיצוני מתבצעת על ידי ה-TB.

CONTROL.VHD



<u>יחידת הבקרה מכילה בתוכה 2 sub-modules – ControlLines, StateLogic.</u>

- יחידת הבקרה המרכזית מקבלת בתור כניסות את 3 הדגלים הנוכחיים מהיחידה האריתמטית בתוך ה-Datapath, ואת 4 ביטי ה-opcode מהפקודה שמאוחסנת באותו זמן ב-IR (בתוך ה-Datapath), ומקבלת מה-TB דרך ה-Top את קווי השליטה sys_rst, sys_ena ושעון, כפי שפירטנו למעלה.

<u>מוצאי יחידת הבקרה הם:</u>

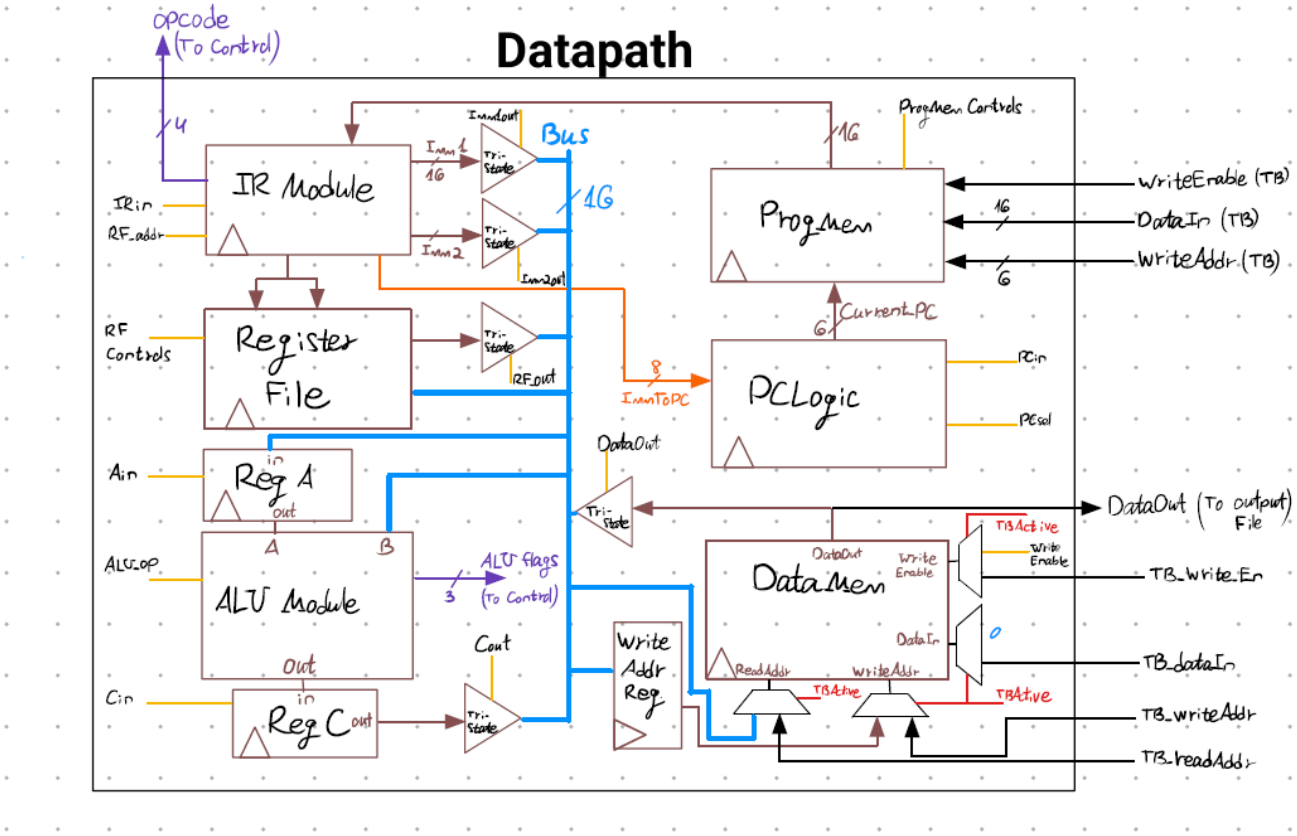
- אל ה-Datapath (דרך ה-Top)- Control bits/lines (משמש לשליטה על ה-Datapath)
- אל ה-TB (דרך ה-Top) – Done bit (כפי שהוסבר למעלה).

<u>המודולים הפנימיים, בקצרה, מבצעים את הפעולות הבאות:</u>

<u>StateLogic</u> – מערכת סינכרונית שקובעת את המצב הבאה של המערכת (על פי ה-FSM המצורף) ומאחסנת את המצב הנוכחי.

<u>ControlLines</u> – מערכת צירופית טהורה ברובה (מלבד שלושה רכיבי זיכרון המשמשים לשמירת ערכי הדגלים הרלוונטיים), שעל פי המצב הנוכחי שמוזן אליה מה-StateLogic ונתונים נוספים המגיעים מה-Datapath, מדליקה את קווי הבקרה הרלוונטיים לביצוע הפעולות המבוקשות ב-Datapath.
רכיבי הזיכרון משמשים לשמירת ערכי הדגלים Z,N,C רק כאשר הם רלוונטיים (לדוגמה, כאשר מתבצעת פקודת add, ב-state הספציפי בו הפעולה בעלת משמעות).

DATAPATH.VHD



מודול ה-Datapath מממש חומרתית את פעילות המעבד הרצויה, ובהתאם לקווי הבקרה (מסומנים בצהוב בשרטוט) המוזנים כל מחזור שעון מבצע את הפעולות הנדרשות בהתאם ל-state בו המערכת נמצאת.

כלל הקופסאות המסומנים בצבע חום בשרטוט ממומשות ע"י sub-modules, ומחוברים בינן לבין עצמן במודול ה-Datapath. המודולים העובדים על בסיס שעון מכילים בתוכם סימון המייצג זאת (משולש בפינה השמאלית של המודול). הממשק עם שאר המערכת:

אל מול יחידת הבקרה (Control):

- **כניסות:** ביטי הבקרה. מסומנים בצהוב בשרטוט.
- **יציאות:** Opcode (4 ביטי הפקודה הנוכחית המוחזקת ב-IR Module), 3 דגלי הסטטוס המגיעים מה-ALU Module (יחידה לוגית טהורה, Z Flag, N Flag, C Flag). מסומנים בסגול בשרטוט.

אל מול ה-TB (דרך ה-Top):

- **כניסות:** קווי שליטה על כתיבת ProgMem, DataMem המשמשים לטעינת התוכנה והנתונים הראשונית לפני הפעלת המעבד.
- **יציאות:** קו יציאה של תוכן ה-Data Memory, שמשמש להוצאת תוכן ה-Data Memory אל קובץ txt בסיום פעולת המעבד (מבוצע על ידי ה-TB).

Simulation results

TOP TESTBENCH

יצרנו תוכנית Assembly פשוטה שמבצעת כפל בין 2 מספרים (אחד מהם אי-שלילי), ותרגמנו את הנתונים האלו לקבצי txt כנדרש כפי שמוגדר ב-task:

```
// c <= a * b
// f <= c xor 0xFFFF

data segment
0 | a = 5
1 | b = 7
2 | c = 0 (result for later)
3 | d = 0xFFFF
4 | e = 1
5 | f = 0 (result for xor)

code segment
        ld r1, 0(r0)     # load r1, a
        ld r2, 1(r0)     # load r2, b
        mov r3, 0        # r3 = 0 (res)
        ld r6, 4(r0)     # load r6, e (e=1)
        sub r2,r2,r6
Loop:   add r3,r1,r3     # r3 = r3+r1
        sub r2,r2,r6     # r2 = r2 - 1
        jnc end
        jmp Loop
end:    st r3,2(r0)      # c = r3
        ld r7,3(r0)      # load r7, d = 0xFFFF
        xor r7,r7,r3     # r7 = XOR(r7,r3)
        st r7, 5(r0)     # f = r7
        done
```

קבצי ה-ITCMinit, DTCMinit (בהתאמה):

1	0005
2	0007
3	0023
4	FFFF
5	0001
6	FFDC

1	D100
2	D201
3	C300
4	D604
5	1226
6	0313
7	1226
8	9001
9	70FC
10	E302
11	D703
12	4773
13	E705
14	F000

לאחר הרצת ה-tb, קיבלנו את קובץ הטקטס הבא (ה-TB מוציא את תוכן ה-DataProg אל קובץ טקסט בשם DTCMcontent):

```
0005
0007
0023
FFFF
0001
FFDC
```

$0x5 * 0x7 = 0x23$ כנדרש

צילום מסך של הזכרונות במערכת בסוף ההרצה:

Program Memory

```
00000000 D100 D201 C300 D604 1226 0313 1226 9001
00000008 70FC E302 D703 4773 E705 F000 XXXX XXXX
00000010 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000018 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000020 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000028 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000030 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000038 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

Data Memory

```
00000000 0005 0007 0023 FFFF 0001 FFDC XXXX XXXX
00000008 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000010 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000018 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000020 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000028 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000030 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
00000038 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

Register File

```
00000000 0000 0005 FFFF 0023
00000004 XXXX XXXX 0001 FFDC
00000008 XXXX XXXX XXXX XXXX
0000000c XXXX XXXX XXXX XXXX
```

DATAPATH TESTBENCH

עבור ה-Datapath בנינו קובץ tb שטוען את הקוד למערכת בדומה ל-tb עבור ה-top, ומתוך ה-tb עצמו אנחנו בכל מחזור שעון מפעילים קווי בקרה המדמים את פעולת ה-control unit.

תוכנית האסמבלי המסומלצת:

```
data segment
arr dc16    5,7,-1,1,2,3,4

code segment
ld r1,0(r0)
ld r2,1(r0)
mov r3,4
add r2,r2,r1
sub r1,r1,r3
st r1,2(r0)
st r2,4(r0)
done
```

הקבצים המוזנים למערכת הם: (משמאל – ITCMinit, מימין – DTCMinit)

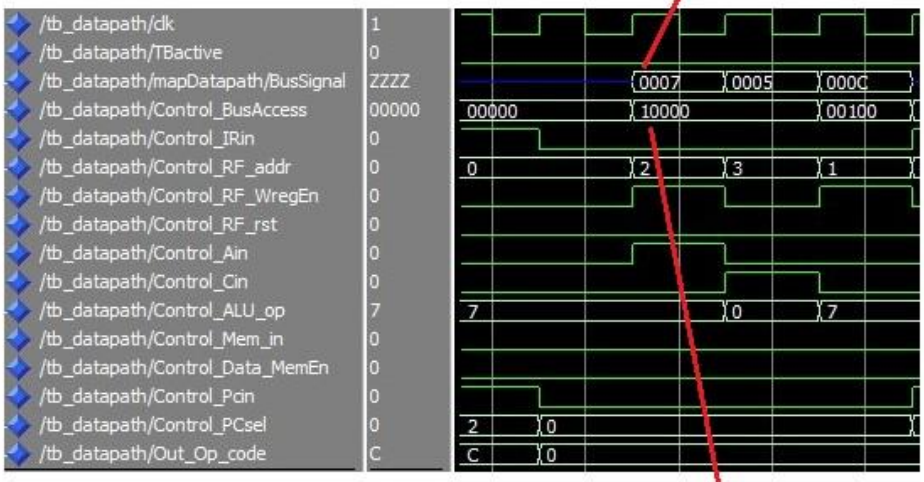
D100	
D201	
C304	
0221	
1113	
E102	
E204	
F000	

0005	
0007	
FFFF	
0001	
0002	
0003	
0004	

קובץ המוצא – DTCMcontect:

0005
0007
0001
0001
000C
0003
0004

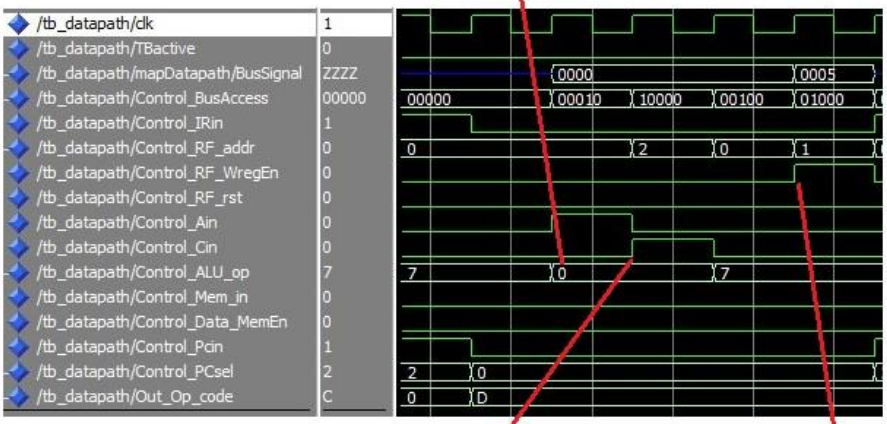
בעליית השעון נשפך ל-bus
תוכן רגיסטר r2, שערכו 0x7



קווי הבקרה ששולטים על
הכניסה ל-bus

ביצוע פקודת ld r1,0(r0)

קו הבקרה ל-ALU הוא
000 - מבצע פעולת ADD

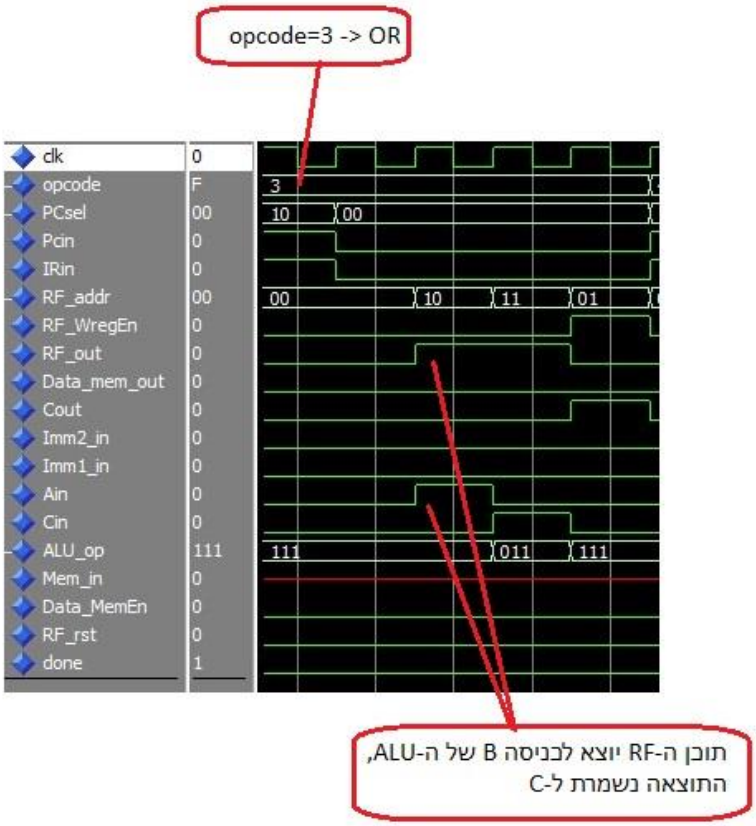


ALU מבצע פעולת חיבור, שאת
התוצאה שלה נשמור ל-C על ידי
קו בקרה Cin=1

קו הבקרה לכתיבה ל-RF
נדלק

CONTROL TESTBENCH

כתבנו testbench עבור ה-control unit, נראה שמתבצעת הפעולה הרצויה בפקודת Or:



כל התוצאות התקבלו כמצופה.