**תרגיל בית 2 - יסודות המחשוב**

**תרגיל ראשון**

לקובץ זה מצורפת תיקיה מקווצת בשם "mul8" שאותה עליכם להעתיק לתיקיית project. התיקייה כוללת את קבצי ה-hdl הריקים אותם עליכם לממש )שכוללים את הגדרת הממשק של הצ'יפ(, וקבצי בדיקה והשוואה לחלק מהצ'יפים. במידת הצורך תוכלו לממש קבצים נוספים ולכתוב קבצי בדיקה נוספים או שתוכלו לבדוק את הצ'יפים גם על ידי בדיקה ידנית בסימולטור (ע"י הזנת קלטים ובדיקת הפלטים).  
הקפידו לפני מימוש הצ'יפים לתכנן אותם בעזרת שירטוט, אך אין חובה בתרגיל זה להגיש את השרטוט.

בתרגיל זה תוכלו להשתמש בכל הצ'יפים שמימשנו בתירגולים הראשון עד לרביעי חוץ מה-ALU.

בתרגיל זה תממשו צ'יפ ALU חדש שעובד על קלטים חיוביים בגודל 8 ביט וגם יודע להכפיל אותם. ההכפלה תהיה בעזרת כפל ארוך. כלומר, על מנת להכפיל a\*b נדרש כל פעם להכפיל את a בביט אחד מ-b. ובהתאם למיקום, נדרש לבצע Shift Left.

**א.** ממשו ב-HDL צ'יפ בשםMul16x1.hdl המוגדר באופן הבא:

CHIP Mul16x1 {

IN a[16], b;

OUT out[16];

…

}

פעולת הצ'יפ היא: הצ'יפ מקבל מספר בן 16 ביטים ומספר בן ביט אחד ומכפיל ביניהם (אם b==1 אז out=a. אחרת (b==0) אז out=0).

לדוגמא:

A = 45 ( 0000 00000010 1101), b=1 -> out = 45 ( 0000 00000010 1101)

a = 45 ( 0000 00000010 1101), b=0 -> out = 0 ( 0000 00000000 0000)

**ב**. ממשו ב-HDL צ'יפ בשם ShiftLeft16.hdl המוגדר באופן הבא:

CHIP ShiftLeft16{

IN in[16];

OUT out[16];

…

}

פעולת הצ'יפ היא: הצ'יפ מקבל מספר בן 16 ביטים ומבצע הזזה שמאלה. בביט הימני ביותר נוסף 0. התוצאה מוחזרת כמספר בן 16 ביטים. הביט השמאלי ביותר ב-in נזרק.  
לדוגמא:

in=45 ( 0000 00000010 1101) -> out = 90 ( 0000 00000101 1010)

ממשו את הצ'יפ בצורה יעילה (הזזה שמאלה ניתן לממש ע"י כפל ב-2 או חיבור של המספר עם עצמו אך ניתן לעשות זאת עם פחות רכיבים לוגיים).

**ג.** ממשו ב-HDL צ'יפ בשם Mul8.hdl המוגדר באופן הבא:

CHIP Mul8 {

IN a[8], b[8];

OUT out[16];

…

}

פעולת הצ'יפ היא: הצ'יפ מקבל שני מספרים **ללא סימן** בגודל 8 ביט כל אחד.

הפלט הוא מכפלת המספרים.

הכפלה לדוגמא :

a=00000101 = 5  
 \*

b=**00101001** = 41

----------

00000101 = (**1**\* 0 shiftleft of 00000101)

00000000 = (**0**\* 1 shiftleft of 00000101)

00000000 = (**0**\* 2 shiftleft of 00000101)

00000101 = (**1**\* 3 shiftleft of 00000101)

00000000 = (**0**\* 4 shiftleft of 00000101)

00000101 = (**1**\* 5 shiftleft of 00000101)

00000000 = (**0**\* 6 shiftleft of 00000101)

00000000 = (**0**\* 7 shiftleft of 00000101)

-----------------

11001101 = 205

למימוש צ'יפ זה חובה להשתמש בצ'יפים שמומשו בסעיפים הקודמים.

רמז למימוש: שימו לב שלצורך המימוש מספיק להשתמש בשבע רכיבי ShiftLeft16 בלבד. הסיבה היא שלדוגמא, כדי לבצע shift של חמישה ביטים מספיק לקחת את התוצאה של shift של ארבעה ביטים ולבצע עליה shift נוסף.

**ד**. ממשו ב-HDL צ'יפ בשם NewALU8bit.hdl המוגדר באופן הבא:

CHIP NewALU8bit {

IN a[8], b[8], opcode[2];

OUT out[16];

…

}

הקלטים לצ'יפ הם 2 מספרים בינאריים בגודל 8 ביטים וקוד בינארי opcode[2] בגודל 2 ביטים .

הפלט של הצ'יפ הוא מספר בינארי out[16] בן 16 ביטים.

הרכיב עובד בצורה הבאה :

אם code=00 אז out=a+b (חיבור אריתמטי בין 2 הקלטים(

אם code=01 אז out=a and b (bitwise and בין 2 הקלטים(

אם code=10 אז out=a\*b (כפל אריתמטי בין 2 הקלטים(

אם code=11 אז out = a XOR b

למימוש צ'יפ זה אין לממש צ'יפי עזר נוספים. חישבו איך לממש בצורה קצרה ויעילה תוך שימוש בצ'יפים שמימשתם בסעיפים הקודמים ובצ'יפים שמימשנו בתירגולים 1-4. אין להשתמש בצ'יפ ALU שמימשנו בתירגול.

**תרגיל שני**

כפי שראינו בהרצאה יש 18 צירופים של ביטי בקרה שנמצאים בשימוש. בפועל ישנם צירופים נוספים אך רק 18 הצירופים האלו נמצאים בשימוש.  
בטבלה הבאה ישנם כמה צירופים שאינם בשימוש. עבור כל צירוף ענה על השאלות הבאות:

א. איזו פעולה יבצע ה-ALU אם יוזן לו צירוף זה כשמוזנים לו הקלטים x ו-y?  
ב. הוכח את הטענה מ-א.  
ג. הפעל את ה-ALU בסימולטור עם צירוף ביטי הבקרה וקלט מייצג והראה את התוצאה כולל חישוב ידני שמראה שהתוצאה מתאימה לציפיות שלך. צרף תמונה של ההפעלה בסימולטור.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| pre-setting  the x input | | pre-setting  the y input | | selecting between computing + or & | post-setting the output | Resulting  ALU output |
| **Zx** | **Nx** | **zy** | **ny** | **F** | **No** | **out** |
| if zx  then  x=0 | if nx  then  x=!x | if zy  then  y=0 | if ny  then  y=!y | if f  then out=x+y  else out=x&y | if no  then  out=!out | out(x,y)= |
| 1 | 1 | 0 | 1 | 0 | 0 | ? |
| 1 | 1 | 0 | 1 | 0 | 1 | ? |
| 1 | 1 | 1 | 1 | 0 | 1 | ? |
| 1 | 1 | 1 | 1 | 1 | 0 | ? |
| 0 | 1 | 0 | 1 | 1 | 1 | ? |

**הוראות הגשה:**

ההגשה **בזוגות בלבד**. ההגשה במודל לפי התאריך המעודכן בתיבת ההגשה. יש להגיש קובץ hw2.zip שמכיל את התיקיה "mul8" שאותה קיבלתם עם הקבצים המעודכנים.

|  |  |  |
| --- | --- | --- |
|  | קבצים שנדרשתם לעדכן או לכתוב | קובץ word |
|  |  | שמות + ת.ז. של המגישים |
| תרגיל ראשון | Mul16x1.hdl | צילום מסך של הרצה ידנית של הצ'יפ עם שני קלטים תיפוסיים עבור כל צ'יפ. |
| ShiftLeft16.hdl |
| Mul8.hdl |
| NewALU8bit.hdl |
| קבצי בדיקה והשוואה נוספים שכתבתם וקבצי בדיקה והשוואה ששיניתם |
| תרגיל שני |  | תשובה לשאלה, כולל צילומי מסך של הרצת ביטי הקלט עבור המקרים שבשאלה עבור קלט טיפוסי. |
| מהתירגולים | קבצי הצ'יפים  מתירגולים 3-4  ALU, Bit, Register, PC, RAM8, RAM64, RAM512, RAM4K, RAM16K |  |

**ההגשה עד לתאריך המעודכן במודל בזוגות בלבד.**

**שאלות נא להפנות למרצה האחראי על התרגיל רן במייל** [**rand@braude.ac.il**](mailto:rand@braude.ac.il)**.**

**בהצלחה מכל צוות הקורס!**