

מעבדה בהנדסת חשמל 1א' 044157

ניסוי SV1 תדריך מעבדה ודוח סיכום

גרסה 1.61
קיץ תשפ"ב 2022

מועד	ביצוע עד סעיף	שם המדריך בפועל	תאריך
ביצוע הניסוי			4.8.22
השלמת חלקים חסרים			

סטודנט	שם פרטי	שם משפחה
1	עמיחי	שטרנליכט
2	יקיר	לוגסי

תוכן עניינים

4	1	התאמת ה- mux מדוח ההכנה
5	1.1	סינתזה
6	1.2	הקצאת הדקים וצריבה לכרטיס
7	2	מונה סינכרוני עם קפיצות
8	2.1	צריבה לכרטיס
9	3	מונה מתנפח
9	3.1	הגדרת הבעיה והפתרון שלה
11	3.2	מימוש משווה
11	3.2.1	כתיבת קוד
12	3.2.2	סימולציה ל-Control-Path
13	3.2.3	סימולציה ל-Data-Path
14	3.3	מימוש מונה (slow/fast)
15	3.3.1	סינתזה
16	3.3.2	סימולציה
17	3.4	מימוש המונה המתנפח כתכן הירארכי בקוד SystemVerilog
18	3.4.1	שרטוט "גרפי" של המימוש ב-SV
19	3.4.2	סימולציה למונה המתנפח
20	4	המונה המתנפח בתכן הירארכי גרפי
20	4.1	אכטקטורה
21	4.2	הוספת מודולים שלך לשרטוט
21	4.3	שעון של קצב 1 Hz – סעיף הסבר, לא לביצוע
22	4.4	מונה מתנפח הירארכי עליונה
23	4.4.1	סימולציה למכלול המלא
25	4.4.2	הקצאת הדקים
26	4.4.3	צריבה לכרטיס והדגמה
26	5	גבוי העבודה

הערות לפני תחילת העבודה:

מטרות ניסוי זה היא לתרגל:

- כתיבת קוד בשפת SV
- כתיבת מונים סינכרוניים
- תכן הירארכי בתוכנה
- תכן הירארכי גרפי

<https://youtu.be/hlaMRqzOvPo58> :ראו את הסרטון שמתאר את תכני המעבדה:

ראו גם את הסרטון בו השתמשתם במעבדה הקודמת על טיפול בקבצים

רשום את השעה בה התחלת את המעבדה: 14:55

1 התאמת ה- mux מדוח ההכנה

מטרה: להוסיף מודול לפרויקט ע"י שינוי במודול קיים.

1. הורד מהמודל ופתח את קובץ הארכיב ששמרת בסיום עבודת ההכנה למעבדת SV1.

שים לב לא לשמור אוטומטית בשם שמציעים לך אלא:

- שמור את הקובץ ב desktop שלך בתיקייה שתייצר עבור מעבדה זו ופתח אותו לפרויקט בתיקייה הנ"ל בשם עם התאריך של יום המעבדה. (הקפד לא להשאירו בתיקיית downloads).
- שים לב להשתמש בתיקייה קרובה ל- DESKTOP ושאינה עמוק בעץ.

2. פתח את הקובץ של מימוש ה- MUX בעזרת משפט IF קומבינטורי **mux_4to1_if.sv**.

3. שנה את שם המודול ל- **mux_2to1_if** וצמצם אותו לשתי כניסות מידע וכניסת select של סיבית אחת.

4. שמור את הקובץ המצומצם בשם זהה לשם המודול **mux_2to1_if.sv** (ע"י פקודת השמירה הרגילה
(File -> Save as) תחת אותו פרויקט והגדר אותו כ- TOP.

הוסף את הקוד המעודכן לדוח:

```
module mux_2to1_if
(
    input logic [1:0] datain,
    input logic select,
    output logic outd
);

always_comb
begin
    outd = 0;
    if(select == 1'b0) begin
        outd = datain[0];
    end
    else if(select == 1'b1) begin
        outd = datain[1];
    end
end

endmodule
```

1.1 סינתזה

1. בצע אנליזה סינתזה Analysis & Synthesis.
2. בדוק ששמות הכניסות/יציאות שלך מתאימים לקובץ ההדקים הנתון לך. אם לא, התאם לפי הצורך.

שים לב: אם עשית שינויים, ובכלל אחרי כל שינוי בקובץ ההדקים, יש לבצע:

- הסרת כל הדקים (Assignmemnts -> Remove assignments -> All)
- והרצת קובץ הדקים (Tools -> TCL Scripts -> Run)

1.2 הקצאת הדקים וצריבה לכרטיס

1. הרץ את קובץ ה- *.tcl.
2. בצע קומפילציה מלאה Compilation לתכן.

שים לב: תמיד לוודא בסיכום הקומפילציה המלאה שהוקצו יותר מ- 0 מודולים (ALMS) לפרויקט.

3. הצג את ה- PIN PLANNER הכולל את הקצאת ההדקים. בדוק שהיא נכונה ושכל הכניסות והיציאות הוגדרו כהלכה. כמו כן, וודא שלכל הפינים הוגדר 3.3V אם לא, בצע Remove Assignment והקצאת הדקים מחדש.

הוסף את השורות הרלוונטיות של ה- PIN PLANNER לדוח.

Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard
datain[1]	Input	PIN_AA15	3B	B3B_N0	3.3-V LVTTTL
datain[0]	Input	PIN_AA14	3B	B3B_N0	3.3-V LVTTTL
outd	Output	PIN_AA24	5A	B5A_N0	3.3-V LVTTTL
select	Input	PIN_AB30	5B	B5B_N0	3.3-V LVTTTL

4. הורד את התכן לכרטיס.

הגדר מה תרצה לבדוק על הכרטיס, כדי להראות פעולה תקינה של המערכת.

תשובה: כאשר ה-select פעיל (sw0 למעלה), נרצה לבדוק האם היציאה (ledr0) מקבלת את ערך הכניסה datain[1] (המשוּיך ל-key3), ובאותו אופן כאשר select כבוי (sw0 למטה), נרצה לבדוק האם היציאה (ledr0) מקבלת את ערך הכניסה datain[0] (המשוּיך ל-key2).

5. בדוק שהמערכת פועלת כנדרש על הכרטיס.

קרא למדריך, רשום את השערה בה הוא ראה את המעגל: 15:17

2 מונה סינכרוני עם קפיצות

מטרה: לבדוק מעגל שנכתב בעבודת הכנה, על הכרטיס.

1. קבע את קובץ המונה עם קפיצות `jmp_counter.sv` מעבודת ההכנה כ- TOP של הפרויקט.
2. בדוק ששמות הכניסות/יציאות שלך מתאימים לקובץ ההדקים הנתון לך. אם לא, התאם לפי הצורך והרץ קובץ הדקים.
3. הרץ קומפילציה מלאה לתכן והצג את ה- PIN PLANNER.

הוסף את סיכום הקומפילציה המלאה וטבלת ההדקים הרלוונטים מה- PIN PLANNER לדו"ח.

Flow Status	Successful - Thu Aug 04 15:22:51 2022
Quartus Prime Version	17.0.0 Build 595 04/25/2017 SJ Standard Edition
Revision Name	SV1Exs
Top-level Entity Name	jmp_counter
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	3 / 41,910 (< 1 %)
Total registers	8
Total pins	6 / 499 (1 %)
Total virtual pins	0
Total block memory bits	0 / 5,662,720 (0 %)
Total DSP Blocks	0 / 112 (0 %)
Total HSSI RX PCSs	0 / 9 (0 %)
Total HSSI PMA RX Deserializers	0 / 9 (0 %)
Total HSSI TX PCSs	0 / 9 (0 %)
Total HSSI PMA TX Serializers	0 / 9 (0 %)
Total PLLs	0 / 15 (0 %)
Total DLLs	0 / 4 (0 %)

Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standar
in clk	Input	PIN_AK4	3B	B3B_NO	PIN_AK4	3.3-V LVTTTL
out count[3]	Output	PIN_AG25	4A	B4A_NO	PIN_AG25	3.3-V LVTTTL
out count[2]	Output	PIN_AD24	4A	B4A_NO	PIN_AD24	3.3-V LVTTTL
out count[1]	Output	PIN_AC23	4A	B4A_NO	PIN_AC23	3.3-V LVTTTL
out count[0]	Output	PIN_AB23	5A	B5A_NO	PIN_AB23	3.3-V LVTTTL
in resetN	Input	PIN_AJ4	3B	B3B_NO	PIN_AJ4	3.3-V LVTTTL

2.1 צריבה לכרטיס

1. צרוב את תכן המונה עם הקפיצות לכרטיס.

שים לב: ה- resetN והשעון clk המופעל ידנית, נקבעו לשני לחצנים, זהה אותם!

2. בדוק שהמונה עובד נכון על הכרטיס.

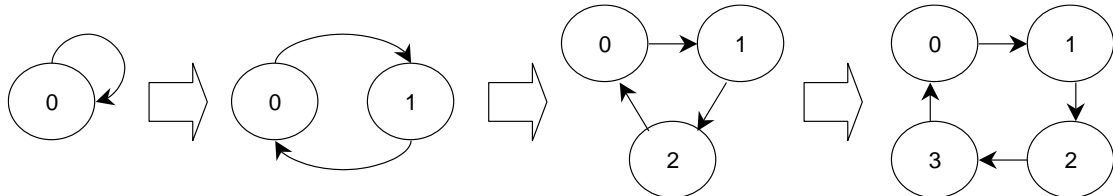
קרא למדריך, רשום את השעה בה הוא ראה את המעגל: 15:26

3 מונה מתנפח

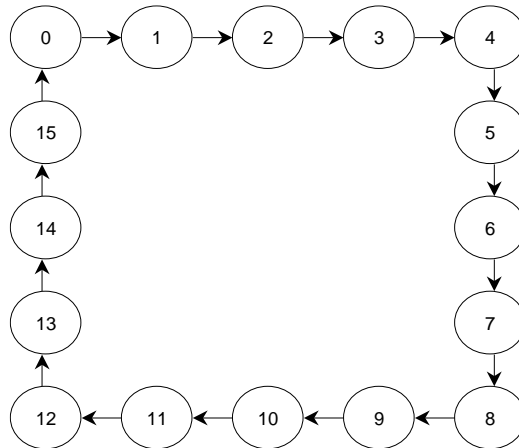
מטרה: בתרגיל זה יש לממש מונה מתנפח בקוד, מונה זהה לזה שנבנה במעבדת תכן סכמתי.

3.1 הגדרת הבעיה והפתרון שלה

נחזור על ההגדרה של מונה מתנפח: שיא הספירה של המונה המתנפח הולך וגדל עם כל מחזור ספירה. בתחילת הספירה (מיד לאחר איפוס המונה) הספירה המרבית היא 0:



במחזור הספירה הבא, הספירה המרבית מגיעה ל-1, במחזור הספירה הבא היא מגיעה ל-2, אחר כך ל-3 וכן הלאה. בסופו של דבר מחזור הספירה עולה ומגיע לספירה מרבית עד 15. לאחר מכן, המונה מתאפס והספירות המרבית שלו שוב עולה 0, 1, 2, 3 וכו'.



<https://youtu.be/fJsrXKCBbj8> ראו את הסרטון שמתאר את המשימה

הגדרת הדרישות:

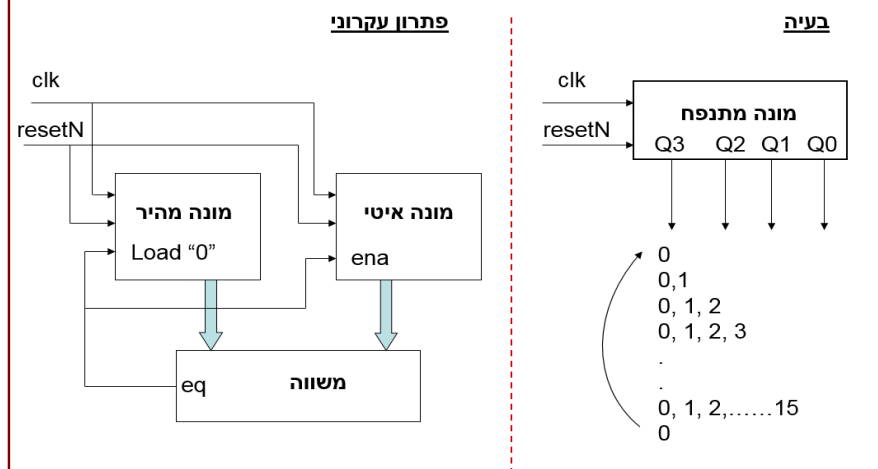
בשלב הראשון נבנה מונה מתנפח בעל שתי יציאות בינאריות. לשם כך יש להשתמש בשני מונים בינאריים עולים וברכיב משווה.

בשלב השני נרחיב את המונה המתנפח בהירארכיה עליונה לעבודה אוטומטית עם שעון הכרטיס ולהצוגת הספירה על תצוגת 7Seg.

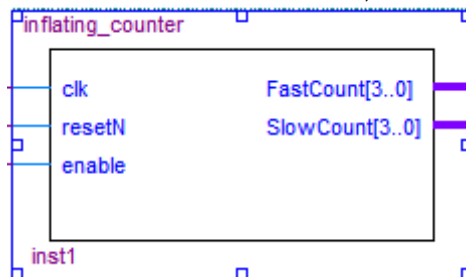
נתחיל עם השלב הראשון ונבנה מונה מתנפח בתוכנה.

הבעיה (בנית מונה מתנפח) והפתרון העקרוני שלה מוצגים בדיאגרמה הבאה.

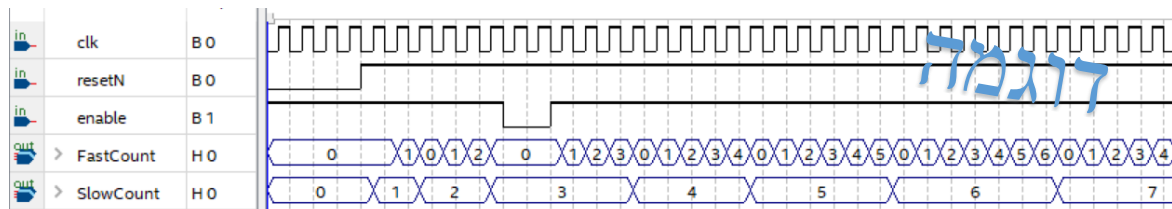
מונה מתנפח



הסימבול של מונה מתנפח צריך להיראות כך:



להלן דוגמה לתוצאות סימולציה של מונה מתנפח:



הגדרת Interface של המונה המתנפח בשלב הראשון:

SIGNAL		
<code>clk</code>	Manual	Input from key
<code>resetN</code>	Active low – asynchronous	Key
<code>enable</code>	Active high - synchronic	Key
<code>SlowCount[3:0]</code>	Output vector	Leds
<code>FastCount[3:0]</code>	Output vector	Leds

נתחיל עם הבניה של כל אחד מהרכיבים הדרושים למונה המתנפח, תחילה המשווה ואז המונה. אחר כך נחבר את שלושת הרכיבים למימוש המונה המתנפח.

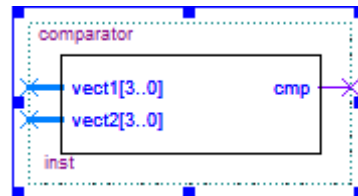
3.2 מימוש משווה

3.2.1 כתיבת קוד

נתון קובץ שהוא שלד של משווה. `comparator.sv`.

1. השלם את הקוד החסר לפי הלוגיקה הנתונה להלן.

	cmp
<code>Vect1[3..0] == Vect2[3..0]</code>	1
<code>Vect1[3..0] != Vect2[3..0]</code>	0



2. הריץ סינתזה מוצלחת לקראת סימולציה.

הוסף את הקוד השלם לדו"ח.

```
module comparator
(
    // Input, Output Ports
    input logic [3:0] vect1,
    input logic [3:0] vect2,
    output logic cmp
);

// Combinatorial logic

assign cmp = (vect1==vect2) ? 1 : 0;

endmodule
```

הנחיה: כיון שיש 256 קומבינציות אפשריות, נחלק את הסימולציה לשני שלבים, כמתואר בסעיפים הבאים.

3.2.2 סימולציה ל- Control-Path

הנחיה: בשלב הראשון נבדוק את הלוגיקה בעלת שני מצבים: כניסות שוות וכניסות שונות.

1. לכן בדוק בסימולציה זו רק **שנים-שלושה מקרים** (מדגם) מתוך כל האפשרויות.

יש לפרט את המצבים שתוצאה לבדוק במקרה זה:

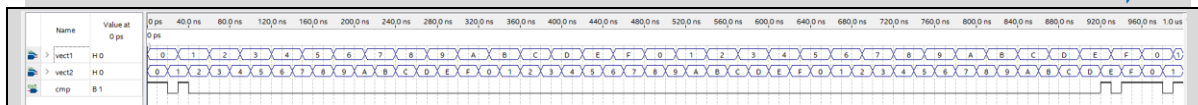
תשובה:

שמנו בכל אחת מהכניסות אות כניסה מסוג counter בתדרים **שונים וזרים** (כלומר לא כפולות אחד של השני) נדגיש את תחילת הסימולציה בא בדקנו מה יהיה הערך של המשווה כאשר ערכי שני הווקטורים שווים ל0, לאחר מכן כיצד הוא משתנה כאשר ערך vect2 משתנה ל1, וכן מה ערך המשווה כאשר vect1 גם כן נהיה 1.

2. בצע סימולציה. **שים לב** להציג את התוצאות כ UNSIGNED או HEX ולא כמספר בינארי.

שים לב: באופן כללי הקפד בכל הסימולציות להציג את התוצאות כ- UNSIGNED או HEX ולא כמספר בינארי, כשרלוונטי. כמו כן, ניתן להוסיף לדוח יותר מתמונה אחת כדי להציג מצבים שונים בסימולציה ומקרי קצה (כגון סיום ספירה והתחלת ספירה מחדש).

הוסף את תוצאות הסימולציה לדוח.



3.2.3 סימולציה ל- Data-Path

הנחיה: בשלב השני נבדוק את כל צירופי הכניסות: יש לכסות את כל $16 \times 16 = 256$ הקומבינציות האפשריות. כדי להיות יעילים מומלץ להשתמש בשני מונים בשני קצבים שונים.

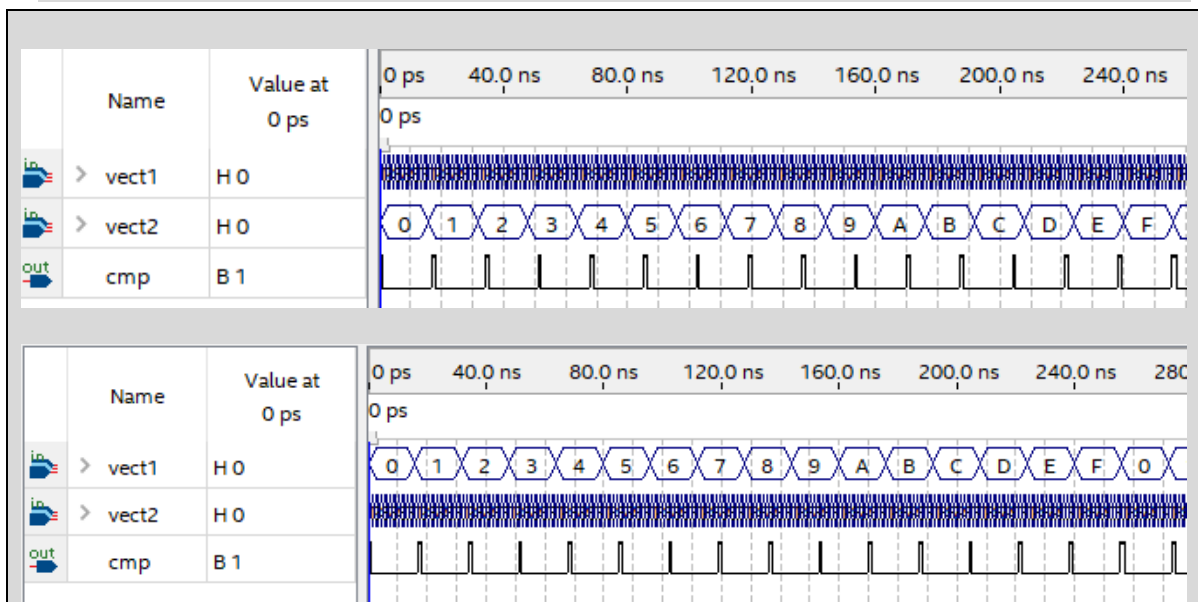
הסבר כיצד תגדיר מונים אלה.

תשובה:

כדי לבדוק את כל הקומבינציות בצורה יעילה נגדיר מונה אחד בתדר 1, ומונה שני בתדר פי 16 ממנו, כך לכל ערך של המונה האיטי נבדוק את כל ערכי המונה השני, לאחר מכן החלפנו בין תפקידי המונים בווקטורים כך שנעבור על כל האפשרויות (משום שלכל ערך של המונה האיטי, נבדקים 16 ערכים של המונה המהיר, סה"כ $16 \times 16 = 256$ צירופים).

1. בצע סימולציה. שים לב להציג את התוצאות כ UNSIGNED או HEX ולא כמספר בינארי.

הוסף תוצאות סימולציה לדוח. ניתן להוסיף יותר מתמונה אחת כדי להציג מצבים שונים.



3.3 מימוש מונה (slow/fast)

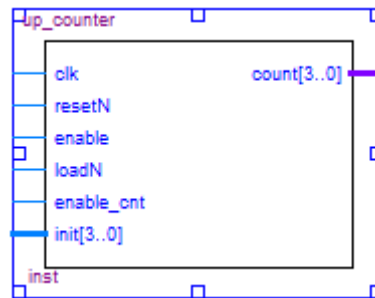
מטרה: לבנות מונה עולה בתוכנה, כולל פונקציות של טעינת נתונים ואפשרות ספירה.

נתונה טבלת האמת שמגדירה את המונה:

resetN	CLOCK_50	enable	loadn	enable_cnt	init[3..0]	count[3..0]	
0	x	x	x	x	x	0000	Reset
1	0, 1, ↓	x	x	x	x	previous count	
1	↑	0	x	x	x	previous count	
1	↑	1	0	x	Init[3..0]	Init[3..0]	Load
1	↑	1	1	0	x	previous count	
1	↑	1	1	1	x	count+1	Increment

שים לב לממש את פקודות ה-IF לפי סדר ההירארכיה של הטבלה ולבצע הזחות INDENT כדי שהקוד יהיה קריא. כמו כן **הקפד** להשתמש בלוגיקה סינכרונית בלבד.

1. **פתח** את קובץ המונה `up_counter`. הוסף לו לוגיקה על פי הדרישות בטבלת הנ"ל. מומלץ למחזור חלקים מהקוד של מונה שכבר כתבתם.



3.3.1 סינתזה

1. הרץ סינתזה Analysis & Synthesis היות והשלב הבא הוא סימולציה.

והוסף את הקוד שלך לדו"ח.

```
module up_counter
(
  // Input, Output Ports
  input logic clk,
  input logic resetN,
  input logic enable,
  input logic loadN,
  input logic enable_cnt,
  input logic [3:0] init,
  output logic [3:0] count
);

always_ff @( posedge clk or negedge resetN )
begin
  if ( !resetN ) begin // Asynchronous reset
    count <= 4'b0;
  end
  else if ( !enable ) begin
    count <= count;
  end
  else if ( !loadN ) begin
    count <= init;
  end
  else if ( !enable_cnt ) begin
    count <= count;
  end
  else begin
    count <= count + 4'b1;
  end
end // always
endmodule
```

3.3.2 סימולציה

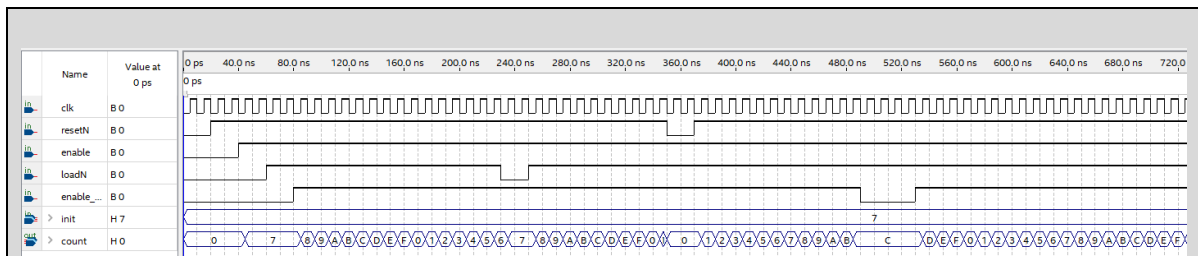
הגדר מה המצבים שתוצאה לסמלך. (יש לפרט את המצבים ולכסות את כל האפשרויות)

תשובה:

נרצה לבדוק את המצבים המתוארים בטבלת האמת, באופן היררכי לפי הדומיננטיות של הרגליים. בהמשך הסימולציה הוספנו אות ריסט / טעינה / אפשרור כדי לבדוק שהלוגיקה עובדת באופן תקין גם לאחר תחילת הריצה.

1. בצע סימולציה. שים לב להציג את התוצאות כ UNSIGNED או HEX ולא כמספר בינארי.

הוסף תוצאות סימולציה לדוח.



קרא למדריך, רשום את השעה בה הוא ראה את המעגל: 16:15

3.4 מימוש המונה המתנפח כתכנ הירארכי בקוד SystemVerilog

מטרה: לממש את המונה המתנפח על ידי חיבור שני מונים ומשווה. תבנה מימוש זה כתכנ הירארכי בתוכנה, ב-SV, על ידי instantiation (הפעלה) של המודולים הנ"ל.

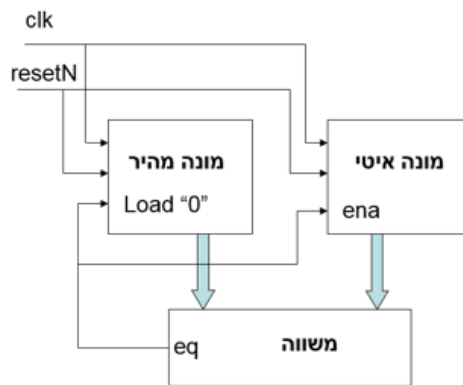
הנחיות:

יש להשתמש בקובץ השלד הנתון לך `inflating_counter.sv` ולהוסיף את החלקים החסרים תוך קביעת החיבורים המתאימים בין שני המונים והמשווה.

ובפרט:

- יש לחבר את כניסת ה-ENABLE החיצונית לשני המונים. (כי בשלב הבא, בהירארכיה העליונה, נשתמש בה להפעלת המערכת רק פעם אחת בשניה.)
- יש לתת ערכים לכל הכניסות של המודולים
- יש לוודא שלכל הסיגנלים שמות מובנים ומשמעותיים (coding convention).

פתרון עקרוני



1. השלם את הקוד בקובץ הנתון.
2. בצע סינתזה מוצלחת לקוד שלך.

הוסף את הקוד לדו"ח.

```
module inflating_counter
// Input, Output Ports
input logic clk,
input logic resetN,
input logic enable,

output logic [3:0] FastCount,
output logic [3:0] SlowCount
);

logic enable_cnt; // internal variable - output of the comparator

// Fast counter instantiation
up_counter fastC(
    .clk(clk),
    .resetN(resetN),
    .enable(enable),
    .loadV(enable_cnt),
    .enable_cnt(enable_cnt),
    .init(4'b0000),
    .count(FastCount)
);

// Slow counter instantiation
up_counter slowC(
    .clk(clk),
    .resetN(resetN),
    .enable(enable),
    .loadV(1'b0),
    .enable_cnt(enable_cnt),
    .init(4'b0000),
    .count(SlowCount)
);

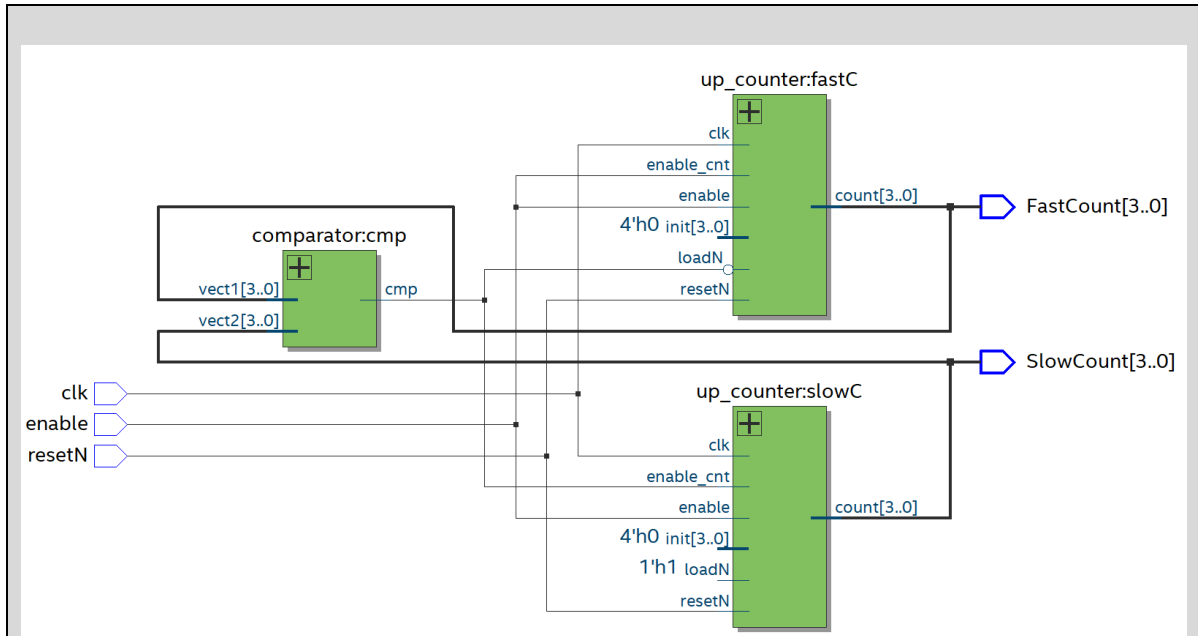
// Comparator instantiation
comparator cmp(
    .vect1(FastCount),
    .vect2(SlowCount),
    .cmp(enable_cnt)
);

endmodule
```

3.4.1 שרטוט "גרפי" של המימוש ב-SV

1. בדומה לדו"ח ההכנה, **הצג בצורה גרפית** את מימוש המונה בעזרת ה- RTL Viewer **ובדוק** שהחיבורים שלך נכונים. אם לא, תקן בהתאם.

הוסף RTL VIEW לדו"ח.



3.4.2 סימולציה למונה המתנפח

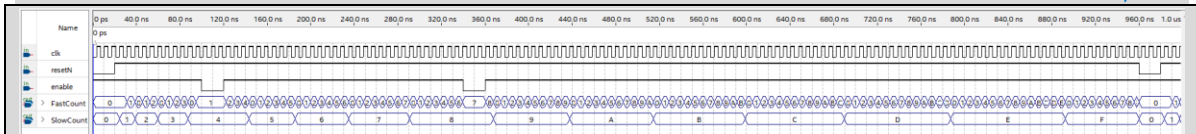
הגדר מה תרצה לבדוק בסימולציה.

תשובה:

נבדוק את הפעולה התקינה של המעגל, תוך שליחת אות ריסט / איפשור.

1. בצע סימולציה. שים לב להציג את התוצאות כ- UNSIGNED או HEX ולא כמספר בינארי.

הוסף את תוצאות הסימולציה לדו"ח.



2. וודא שבדקת את כניסת ה- ENABLE, אם לא חזור על הסימולציה.

שים לב: אין צורך להוריד לכרטיס בשלב זה!

קרא למדריך, רשום את השעה בה הוא ראה את המעגל: 16:48

4 המונה המתנפח בתכן הירארכי גרפי

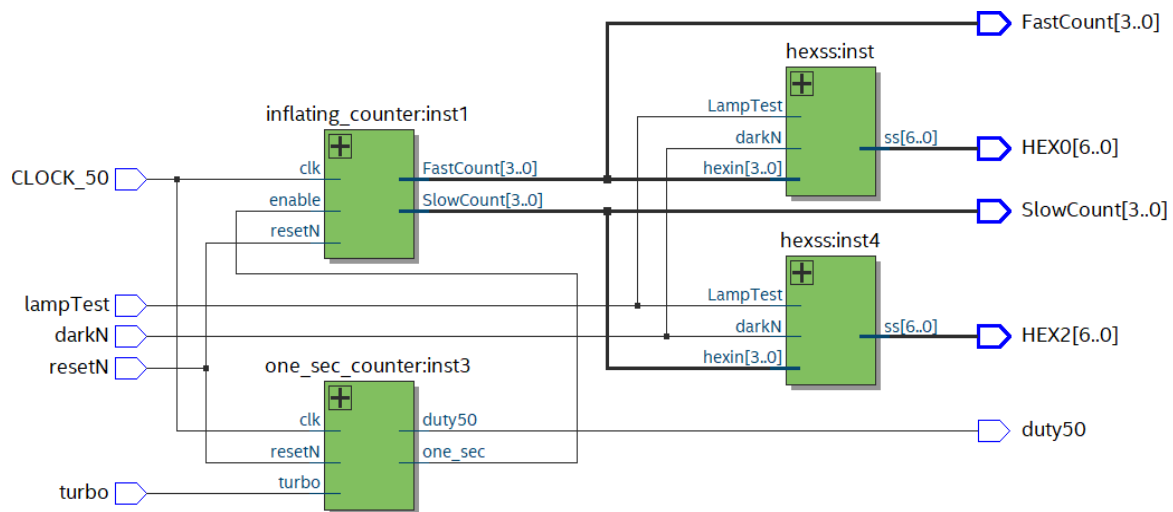
מטרה: שילוב המונה המתנפח בתכן הירארכי עליון שיאפשר הפעלה קלה שלו.

הנחיה: מכיוון שקשה ללחוץ על לחצן השעון עשרות פעמים, וקשה לקרוא תצוגת נוריות שמשתנה קצב של 20nsec, נרחיב את מימוש המונה המתנפח בהירארכיה עליונה, על ידי:

- שימוש בשעון פנימי של הכרטיס בקצב 50MHz – להפעלה אוטומטית של המעגל במקום הלחצן
- הוספת שעון בקצב איטי יותר של 1 Hz – להאטת קצב הספירה
- הוספת תצוגות של 7Seg להצגת ספרות – הרכיב מעבודת ההכנה

◆ פתח את הקובץ הגרפי **inflating_cnt_top.bdf** וקבע אותו כ- TOP.

נתון: המערכת אותה תממש לצורך הבדיקה, **דומה** למערכת הבאה, אבל תוך שימוש ברכיבים שלך.



4.1 איטקטורה

לאילו מתגים/לחצנים תחבר את הכניסות **darkN** ו- **LampTest** בכרטיס ה-DE-10 ומה צריך להיות מצבם (0 או 1) כדי שהמעגל יתפקד כמונה רץ?

תשובה:

לפי הדוח הכנה, נרצה ש- $LampTest = 0$, $darkN = 1$ וזאת בשביל שהם לא יהיו "דומיננטיים" ויקבעו את המוצא, כלומר יאפשרו להציג את המספרים שנרצה. לפי קובץ ה-tcl נחבר את $darkN$ ל-SW8, $LampTest$ ל-SW9.

לאיזה רכיב שעל הכרטיס תחבר את היציאות של המודול **HEXSS**?

תשובה:

לפי קובץ ה-tcl נשתמש בשתי תצוגות 7-מקטעים שהן $hex0$, $hex2$.

4.2 הוספת מודולים שלך לשרטוט

מטרה: להוסיף מודולים יעודיים לשלד הירארכי עליון קיים.

נתונים:

נתון לך שלד של המערכת בהירארכיה עליונה, בקובץ **inflating_cnt_top.bdf**. יש להשלימו עם הרכיבים והחיבורים המתאימים, כפי שיוסבר להלן. להצגת הספירה של המונה המתנפח נשתמש הן בנוריות והן בתצוגת **Seven Segments (7Seg)**.

לצורך השלמת השרטוט בהירארכיה עליונה תחילה יש ליצור סימבול גרפי לכל אחד מהמודולים שכתבת, המודול **inflating_counter.sv** ו- **hexss.sv**, ולהוסיף אותם למערכת. לשם כך יש לבצע את הפעולות הבאות:

1. צור סימבולים למודולים שכתבת. רצוי להעזר ב- Quartus Cook Book.
2. הוסף אותם לשרטוט במקומות המסומנים.

4.3 שעון של קצב 1 Hz – סעיף הסבר, לא לביצוע

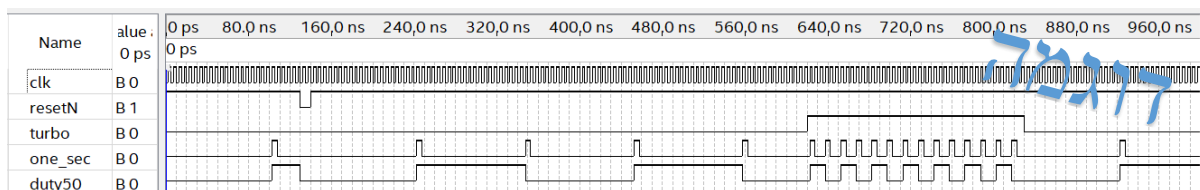
נתון: המודול של מונה מחלק תדר, בשם **one_sec_counter.sv**, סופר פולסי שעון וממיר את קצב השעון המאד מהיר של הכרטיס DE10 (50 MHz), לפולס בקצב איטי יותר של 1 Hz. מודול זה בעל הכניסות:

- clk – שעון (50 MHz של הכרטיס)
- resetN – כניסת איפוס אסינכרונית
- TURBO – כניסה של ביט אחד. ב- 1 לוגי תדר הפולס עולה ל-10 Hz, ב- 0 לוגי התדר נשאר ללא שינוי (1 Hz)

והיציאות:

- **one_sec** – פולס צר, ברוחב של 20 nsec (לפי 50 MHz), בתדר של 1 Hz, מעין "שעון איטי".
- הערה: כדי לא לייצר תופעה של clock skew (הטיית שעון) יש לחבר יציאה זו לכניסת enable של הרכיב (המונה המתנפח), ולחבר את שעון המערכת (50 MHz) לכניסת השעון clk של הרכיב.
- **duty50** – פולס רחב, ברוחב של שניה, בתדר של 0.5 Hz.

במצב TURBO שתי היציאות מהירות פי 10, כפי שניתן לראות להלן בסימולציה של מודול זה.



4.4 מונה מתנפח הירארכיה עליונה

מטרה: להשלים את ההירארכיה העליונה עם הרכיבים והחיבורים הנדרשים. ניתן להיעזר בסכימת ה-RTL הנתונה.

1. **בדוק חיבורי הכניסות** (חלקם כבר קיימים):

- כניסת ה- resetN ללחצן KEY0
- כניסת השעון למתנד 50MHZ פין

PIN_AF14 -to CLOCK_50

ולא למפסק (או שתלחצו עליו 50 מיליון פעמים ☺)

- כניסת הטורבו למפסק (לא ללחצן)
- היציאה one_sec של שעון השניות מהווה כניסת enable של המונה המתנפח

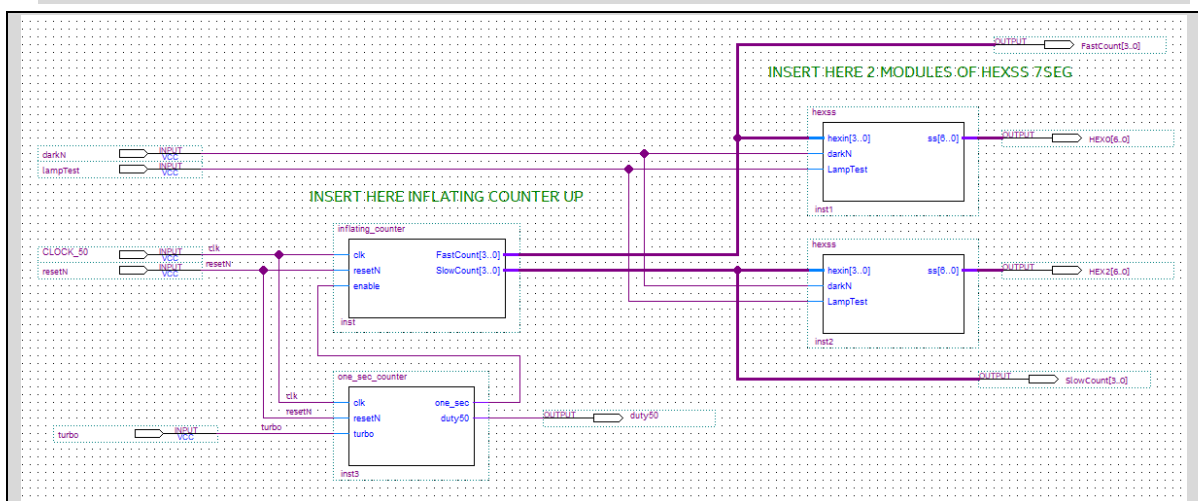
2. **בדוק חיבורי יציאות** (חלקם כבר קיימים):

- היציאה duty50 של השעון one_sec_counter - לנורית אדומה LEDR
- היציאות של המונה המתנפח מחוטים באופן ישיר לנוריות LEDR ולכניסות של HEXSS
- היציאות של HEXSS - ל- 7Seg - דרך קובץ ההדקים

שים לב: חיבורים אלה כבר קיימים בקובץ ההדקים הנתון. אבל

3. יש לעדכן בקובץ ההדקים את שמות הנוריות האדומות לפי שמות היציאות של המונה המתנפח שלך.

הוסף לדו"ח את שרטוט המערכת.




4.4.1 סימולציה למכלול המלא

שים לב! היות והשלב הבא הוא סימולציה, לפני ביצוע הסינתזה יש להקטין את הקבוע במונה של מחלק התדר one_sec_counter.sv מ- 50,000,000 למספר קטן יותר, למשל 20. ראה הוראות בהערות הקובץ.

1. לקראת סימולציה הרץ סינתזה מוצלחת

צורף את סיכום הסינתזה לדו"ח.

Flow Summary	
	
Flow Status	Successful - Thu Aug 04 17:18:31 2022
Quartus Prime Version	17.0.0 Build 595 04/25/2017 SJ Standard Edition
Revision Name	SV1Exs
Top-level Entity Name	inflating_cnt_top
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	42
Total pins	28
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0

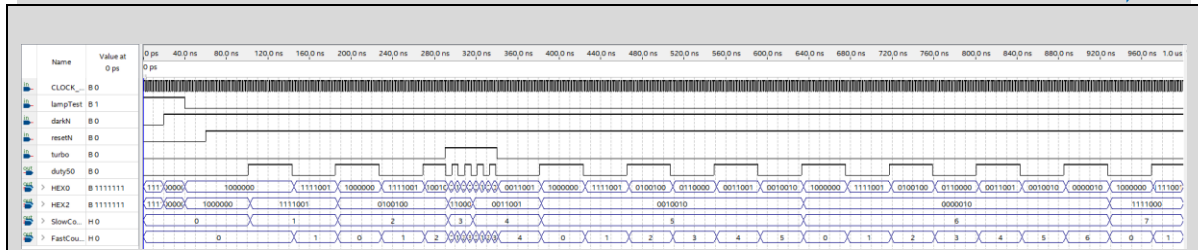
הגדר מה תרצה לבדוק בסימולציה. רשום את כל מצבי הכניסות ויציאות המיוחדים.

תשובה:

Reset, LampTest, DarkN, Turbo, All Outputs

2. בצע סימולציה, שים לב להציג את תוצאות המונים כ UNSIGNED INTEGER או HEX ולא כמספר בינארי.

הוסף את תוצאות הסימולציה לדו"ח.



קרא למדריך, רשום את השעה בה הוא ראה את המעגל: 17:32

4.4.2 הקצאת הדקים

שים לב! לקראת הצריבה חזור את הקבוע במונה מחלק התדר ל- 50,000,000 עבור פעולה עם השעון 50MHz של הכרטיס.

1. בצע הסרת הדקים.
2. הרץ את קובץ ההדקים המעודכן TCL.
3. בצע קומפילציה מלאה לתכן.

רשום להלן כמה זמן ארכה הקומפילציה (זמן הקומפילציה נתון בפינה הימנית בתחתית במסך) 1:21



4. **בדוק** ב- PIN PLANNER שכל ההדקים הרלוונטים מוגדרים נכון וכ- 3.3 V.

הצג את ה- PIN PLANNER הכולל את הקצאת ההדקים הרלוונטים.

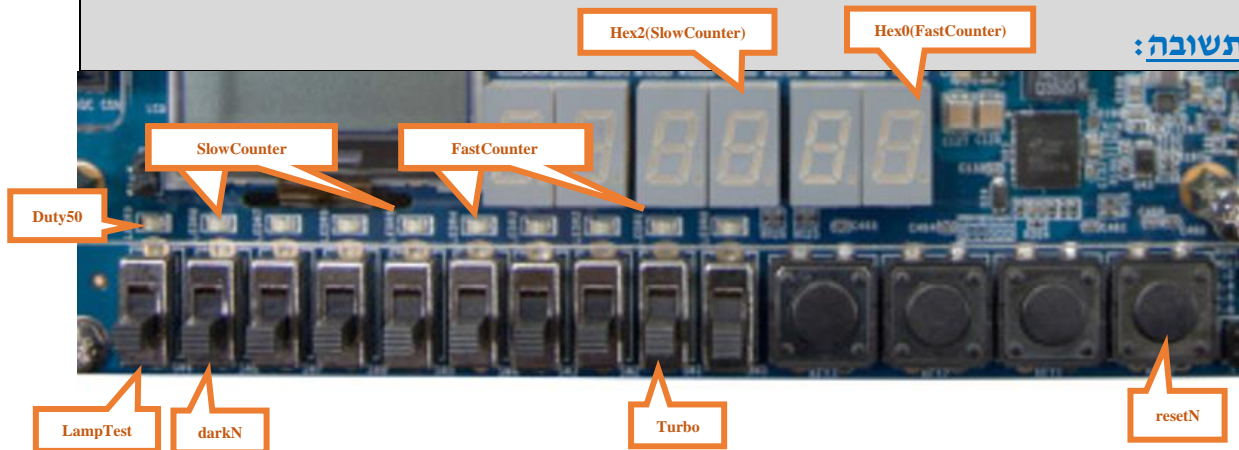
Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate
in CLOCK_50	Input	PIN_AF14	3B	B3B_NO	PIN_AF14	3.3-V LVTTTL		16mA (default)	
in darkN	Input	PIN_AC29	5B	B5B_NO	PIN_AC29	3.3-V LVTTTL		16mA (default)	
out duty50	Output	PIN_AC22	4A	B4A_NO	PIN_AC22	3.3-V LVTTTL		16mA (default)	1 (default)
out FastCount[3]	Output	PIN_AG25	4A	B4A_NO	PIN_AG25	3.3-V LVTTTL		16mA (default)	1 (default)
out FastCount[2]	Output	PIN_AD24	4A	B4A_NO	PIN_AD24	3.3-V LVTTTL		16mA (default)	1 (default)
out FastCount[1]	Output	PIN_AC23	4A	B4A_NO	PIN_AC23	3.3-V LVTTTL		16mA (default)	1 (default)
out FastCount[0]	Output	PIN_AB23	5A	B5A_NO	PIN_AB23	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX0[6]	Output	PIN_AH18	4A	B4A_NO	PIN_AH18	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX0[5]	Output	PIN_AG18	4A	B4A_NO	PIN_AG18	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX0[4]	Output	PIN_AH17	4A	B4A_NO	PIN_AH17	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX0[3]	Output	PIN_AG16	4A	B4A_NO	PIN_AG16	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX0[2]	Output	PIN_AG17	4A	B4A_NO	PIN_AG17	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX0[1]	Output	PIN_V18	4A	B4A_NO	PIN_V18	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX0[0]	Output	PIN_W17	4A	B4A_NO	PIN_W17	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX2[6]	Output	PIN_W16	4A	B4A_NO	PIN_W16	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX2[5]	Output	PIN_AF18	4A	B4A_NO	PIN_AF18	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX2[4]	Output	PIN_Y18	4A	B4A_NO	PIN_Y18	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX2[3]	Output	PIN_Y17	4A	B4A_NO	PIN_Y17	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX2[2]	Output	PIN_AA18	4A	B4A_NO	PIN_AA18	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX2[1]	Output	PIN_AB17	4A	B4A_NO	PIN_AB17	3.3-V LVTTTL		16mA (default)	1 (default)
out HEX2[0]	Output	PIN_AA21	4A	B4A_NO	PIN_AA21	3.3-V LVTTTL		16mA (default)	1 (default)
in lampTest	Input	PIN_AA30	5B	B5B_NO	PIN_AA30	3.3-V LVTTTL		16mA (default)	
in resetN	Input	PIN_AJ4	3B	B3B_NO	PIN_AJ4	3.3-V LVTTTL		16mA (default)	
out SlowCount[3]	Output	PIN_AB22	5A	B5A_NO	PIN_AB22	3.3-V LVTTTL		16mA (default)	1 (default)
out SlowCount[2]	Output	PIN_AF24	4A	B4A_NO	PIN_AF24	3.3-V LVTTTL		16mA (default)	1 (default)
out SlowCount[1]	Output	PIN_AE24	4A	B4A_NO	PIN_AE24	3.3-V LVTTTL		16mA (default)	1 (default)
out SlowCount[0]	Output	PIN_AF25	4A	B4A_NO	PIN_AF25	3.3-V LVTTTL		16mA (default)	1 (default)
in turbo	Input	PIN_Y27	5B	B5B_NO	PIN_Y27	3.3-V LVTTTL		16mA (default)	

4.4.3 צריבה לכרטיס והדגמה

1. הורד את התכן לכרטיס.

סמן על התמונה להלן רק את הנוריות, המפסקים והלחצנים הרלוונטיים להפעלת המונה המתנפח. השתמש בבוטת ה- XX הנתונה לדוגמה, שכפל אותה, עדכן שמות ומקם אותה לפי הצורך.

תשובה:



2. בדוק שהמערכת פועלת כנדרש על הכרטיס:

- המונה מתנפח מ- 0 עד 15, מתאפס ומתחיל להתנפח מחדש.
- בדוק איפוס ב- resetN.
- בדוק שבמצב TURBO המונה מתקדם בקצב מהיר פי 10.
- בדוק את הפעולות של lampTest ו- darkN.

3. אם המערכת אינה פעולת כנדרש מצא את הבעיה, תקן אותה ובדוק שוב.

קרא למדריך, רשום את השעה בה הוא ראה את המעגל: 17:52

5 גבוי העבודה

שמור דוח זה רגיל וכ- PDF והעלה את קובץ ה- PDF למודל.

שמור את הפרויקט רגיל וגם כארכיב (באמצעות Project -> Archive Project).
העלה את קובץ הארכיב למודל, כי תצטרך אותו בהמשך.

גבה את הדוח וארכיב הפרויקט גם באמצעים אחרים.

רשום את השעה בה סיימת את המעבדה: 17:55