**הטכניון - מכון טכנולוגי לישראל**

**הפקולטה להנדסת חשמל**



ניסויVHDL2 - שאלות ודוח הכנה

גרסה 1.02

קיץ 2018

מחבר: אברהם קפלן, דודי בר-און

על פי חוברת של עמוס זסלבסקי מ 2009

|  |  |
| --- | --- |
| תאריך הגשת דו"ח ההכנה |  |
| שם המדריך |  |

|  |  |  |
| --- | --- | --- |
| סטודנט | שם פרטי | שם משפחה |
| 1 | ברק | זן |
| 2 | בועז | טייטלר |

**שימו לב: הפרויקט שתפתחו ותממשו בסעיפים 2 , 3, 4 להלן, ישמש אתכם במעבדה.**

**בסיומו, יש לבצע עליו פעולת ARCHIVE ב QUARTUS (כמתואר בפרק 16 של [quartus 17 cook book](https://moodle.technion.ac.il/mod/resource/view.php?id=616928) במודל).**

**את הקובץ המכווץ שתקבלו מפעולה זו יש להעלות במודל ל**



Contents

[1 מונה ציקלי 2](#_Toc520627852)

[1.1 מימוש סינכרני של היציאה Q 3](#_Toc520627853)

[1.2 מימוש אסינכרוני של היציאה - TC 3](#_Toc520627854)

[1.3 טעינה סינכרונית 4](#_Toc520627855)

[2 מונהBCD יורד עד 00 5](#_Toc520627856)

[2.1 קוד VHDL 6](#_Toc520627857)

[2.2 סימולציה 7](#_Toc520627858)

[3 מכונת מצבים – רמזור מבוקר 9](#_Toc520627859)

[3.1 קוד VHDL 10](#_Toc520627860)

[3.2 סימולציה 10](#_Toc520627861)

[4 פצצה – פרוייקטון 11](#_Toc520627862)

[4.1 ארכיטקטורה 12](#_Toc520627863)

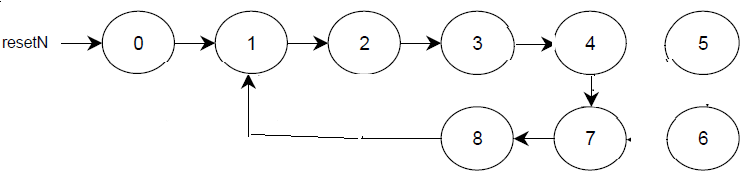
[4.2 מימוש מכונת המצבים 13](#_Toc520627864)

[4.3 סימולציה למכונת המצבים 14](#_Toc520627865)

[4.4 השלמת דיאגרמת התהליכים. 14](#_Toc520627866)

# מונה ציקלי

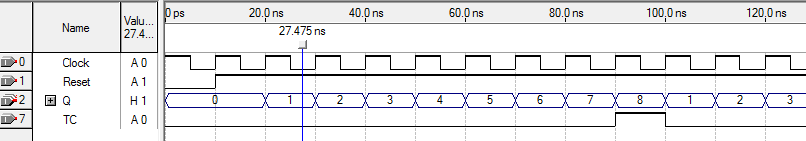
בצע תכן של מונה סינכרוני עולה, שהוא בעל מצבים מ-1 ועד 8 ומדלג על המצבים 5,6



5 ו 6.

יש לממש באמצעות מונה ולא במכונת מצבים.

דוגמה לסימולציה של מונה מ- 1 עד 8 ( **שונה** מהתרגיל שלנו):

המונה מוציא TC אסינכרוני ויציאה סינכרונית של 4 BIT



הקפד על תכן סינכרוני נקי למונה לפי הכללים שהוגדרו בחומר רקע לניסוי תוך שימוש בקוד יעיל.

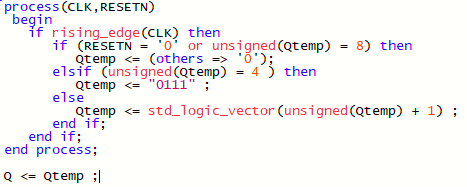
הערה: עליך לזהות את מצב 8 ואז:

* להוציא TC - מיידית (CARRY לספרה הבאה)
* לטעון למונה את המספר -1 בשעון הבא

## מימוש סינכרני של היציאה Q

כתוב תהליך סינכרוני המייצר את היציאות Q ומחזיק משתנה פנימי Qtmp

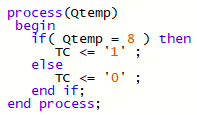
קוד VHDL



## מימוש אסינכרוני של היציאה - TC

הוסף תהליך **אסינכרוני** המייצר את TC מהמשתנה הפנימי Qtmp

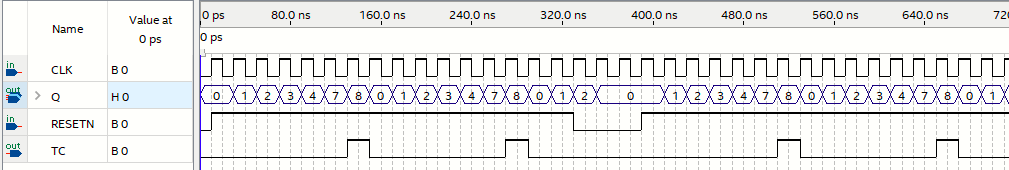
קוד VHDL של התוספת בלבד



לאחר שהקומפילציה עברה בהצלחה בצע לקובץ זה סימולצית **functional** מלאה בה תבדוק את כל הכניסות והיציאות ואת כל מקרי הקצה, כולל בדיקת הכניסה ResetN באמצע פעולת המונה ולא רק בתחילת הסימולציה (בנגוד לדוגמת הסימולציה למעלה).

יש להקפיד להציג את המספרים בצורה יפה RADIX = HEX ולא בבינארי

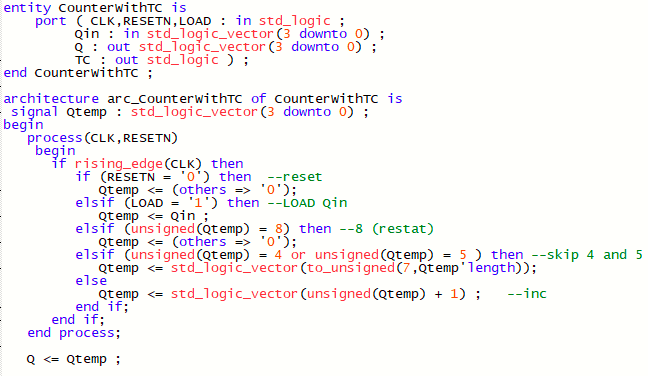
חלון סימולציה



## טעינה סינכרונית

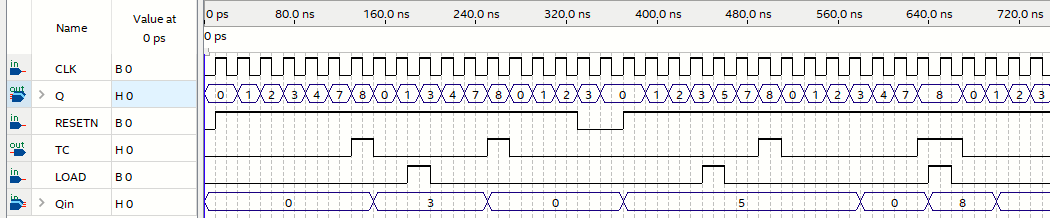
הוסף וקטור כניסה של 4 סיביות וכניסה סינכרונית LOAD ) פעילה בגבוה) , המאפשרת לטעון כל מספר חוקי (קרי קטן מ 8) שנבחר למונה

קוד VHDL



לאחר שהקומפילציה עברה בהצלחה בצע לקובץ זה סימולציה **FUNCTIONAL** בה תבדוק כמה מקרי טעינה כולל טעינת 5

חלון סימולציה



# מונהBCD יורד עד 00

תכנן מונה ציקלי בעל 2 ספרות BCD שסופר בקוד בינארי כלפי **מטה** . ונטען סינכרונית. ניתן להרחיב את המונה מהתרגיל הקודם, כל ספרה היא כמובן ברוחב 4 BIT

הכניסה resetN היא כניסת איפוס א-סינכרונית שפעילה בנמוך. כניסת השעון נקראת clk. הכניסה ena\_cnt היא כניסת אפשור סינכרונית לספירה.

למונה יש 2 יציאות וקטוריות שנקראת countL, countH ויציאה בודדת Terminal Count)TC ) שמפיקה '1' לוגי כאשר המונה נמצא במצב האחרון שלו (00) בירידה.

הכניסה LOAD היא כניסת טעינה סינכרונית שפעילה בנמוך. כאשר כניסה זו ב- 0, והשעון בעליה, המספר בכניסות DATA שמיוצגת ב BCD (כל ספרה בנפרד) נטען למונה , ניתן להניח שהDATA חוקית וכל NIBBLE (ארבעה BIT) אינו מכיל מספרים הגדולים מ- 9.

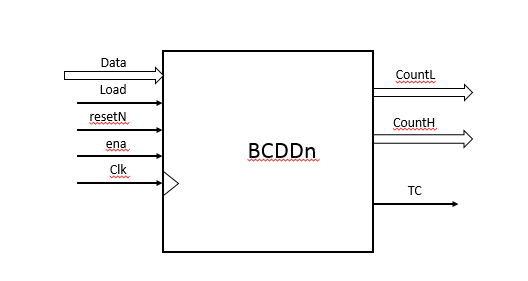
כאשר ספרת האחדות, CountL, מגיעה למצב 0 עשרוני ("0000" בינארי) היא מאותחלת ל- 9 ("1001"), וספרת העשרות, CountH, יורדת באחד באות השעון הבא. כאשר המונה מגיע ל- 00 (עשרוני), הוא :

* מוציא TC

יש להתנות יציאת TC גם ב ena =1.

* מאותחל ל- 99? באות השעון הבא (**נא למלא**).

הציור שלהלן מתאר את כניסות והיציאות של הרכיב



**הדרכה:**

למעט עבור הTC עליך לרשום בארכיטקטורה תהליך סינכרוני בלבד . הקפד על כתיבת קוד חוקי שיעבור קומפילציה נקייה.

בנוסף לכך הקפד לבצע תכן סינכרוני שמתאים לכללי התכן הסינכרוני (אל תשתמש למשל ב - Ripple Clock או בתכן א-סינכרוני כל שהוא). קרא לישות ולקובץ שמכיל אותו BCDDN.VHD.

הוצא את ה-TC על ידי תהליך **אסינכרוני**

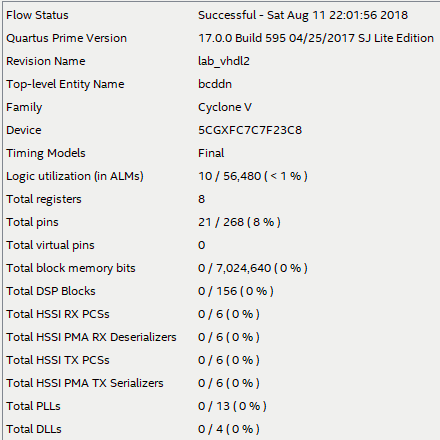
להזכירכם : אין צורך לאפס תהליך אסינכרוני (אין לו רכיבי זכרון שיש לאתחל).

## קוד VHDL



קמפל את הקובץ bcddn.vhd

העתק דוח קומפילציה



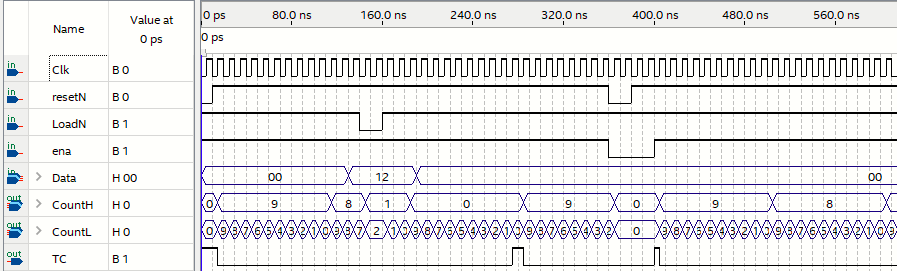
## סימולציה

לאחר שהקומפילציה עברה בהצלחה בצע לקובץ זה סימולציה מלאה בה תבדוק את כל הכניסות והיציאות **המעניינות** ואת כל מקרי הקצה **המעניינים**

הגדר מה תרצה לבדוק בסימולציה – איזה מצבים מעניינים (המשך למלא את הטבלה)

|  |  |
| --- | --- |
| **מצב** | **תוצאות צפויות** |
| יציאה מRESET | כל היציאות מאותחלות |
| COUNT עם ENABLE | לא משתנה |
| COUNT בלי ENABLE | יורד ב1 |
| CARRY | נדלק ב00 ותלוי בena |
| Load | עובד |
|  |  |

הקפד להציג את המשתנים בפורמט רלוונטי BIT HEX DECIMAL

חלון סימולציה

# מכונת מצבים – רמזור מבוקר

בפרוייקט הקודם פתח TOP חדש, וכתוב מכונת מצבים של רמזור, מצבי הרמזור הם אדום, אדום\_צהוב, ירוק, צהוב .

הרמזור יהיה 3.7 שניות באדום .2.9 בירוק ו 1.5 שניות במצבי המעבר (צהוב, אדום-צהוב)

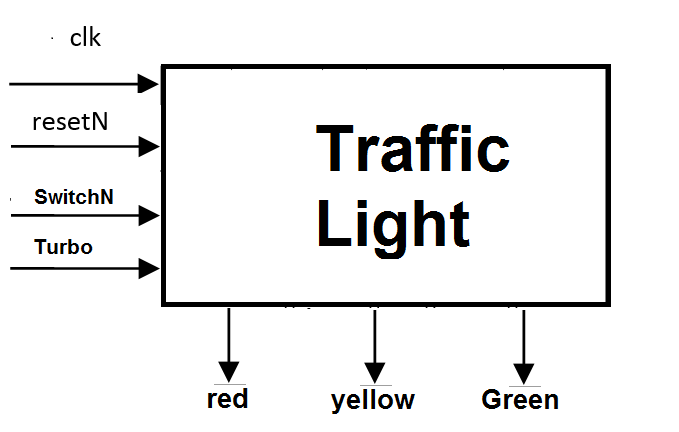
למכונה ינתן שעון מהיר 50MHz

במכונה יוגדר תהליך פנימי שייצר ENABLE של 10Hz ובו תשתמש מכונת המצבים לספירת הזמן

מכונת המצבים תמנה את הזמן על ידי מונה **פנימי**, ותתקדם כשצריך.

לחיצה על לחצן הכניסה SwitchN:

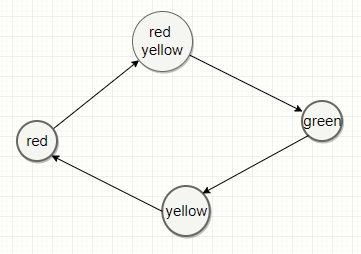
* אם הוא היה במצב אדום תעביר את הרמזור ישירות (בשעון הבא) מאדום לאדום צהוב
* אחרת לא תעשה כלום



הנחייה: יש לנהל מונה סינכרוני **בתוך** מכונת המצבים, המונה יספור מטה וכל פעם שהוא מגיע לאפס עוברים מצב במכונה ומאתחלים אותו לערך חדש.

* יש להשתמש בהגדרת **type** count\_state **is(red,yellow…)**
* יש להשתמש בCONSTANTS ולא במספרים בתוך הקוד
* יש לכתוב נכון את הקבועים למונה הפנימי שיתאימו גם לסימולציה וגם לעבודה בשטח בשינוי יחיד
* כל המכונה תעבוד ב 50MHz
* לרמזור כניסת TURBO באמצעותה אפשר לזרז את פעולת הרמזור פי 10. הערה – הכניסה תחובר למחולל ה- 10Hz ותהפוך אותו למונה 100Hz
* כניסת הלחצן בלוגיקה שלילית, 0 כשהוא לחוץ

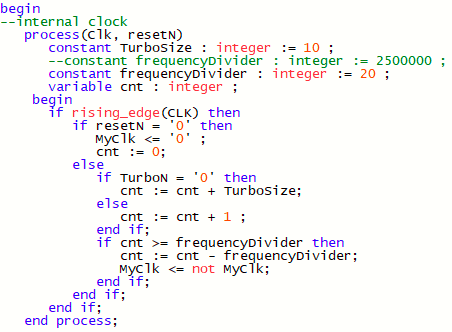
ממש עם מכונה בעלת **תהליכים סינכרונים בלבד**, יציאות המערכת הם שלוש נורות צבעוניות



תאור המצבים – השלם את המצבים החסרים

|  |  |  |
| --- | --- | --- |
| **שם המצב** | **פעילות עיקרית** | **לאיזה מצב עוברים מהמצב הנוכחי ובאילו תנאים – למלא את התאים הריקים** |
| RED | מאתחלים את המונה | לאחר זמן- עוברים לאדום צהוב |
| RED\_YELLOW | מאתחלים את המונה | לאחר זמן – עוברים לירוק |
| GREEN | מאתחלים את המונה | לאחר זמן – עוברים לצהוב |
| ORANGE | מאתחלים את המונה | לאחר זמן – עוברים לאדום |
|  |  |  |

## קוד VHDL





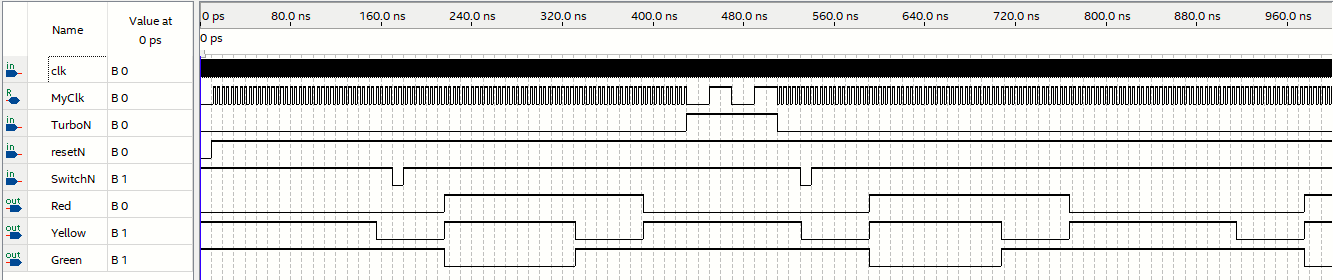
הגדר מה תרצה לבדוק בסימולציה – איזה מצבים מעניינים (המשך למלא את הטבלה)

|  |  |
| --- | --- |
| **מצב** | **תוצאות צפויות** |
| יציאה מRESET | כל היציאות מאותחלות |
| Switch באדום | עובר לכתום אדום |
| Switch בצבע אחר | לא עושה כלום |
| כיבוי טורבו | מאט את התדר פי עשר |
| לא עושים כלום | כל המצבים עובדים ועוברים לבד |
|  |  |

## סימולציה

כתוב והרץ סימולציה של המעגל, שים לב שהשעון צריך להיות מהיר מ-1Hz (לפחות 5Hz )

חלון(נות) סימולציה



# C:\Users\dudyb\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Outlook\HWZC6PZ7\20151224_083020.jpg פצצה – פרוייקטון

**יש לבנות מנגנון לפצצה** על פי ההגדרות להלן:

**כניסות הפצצה**:

* כניסת clk שבכרטיס DE10 בתדר 50MHz.
* לחצן START יטען לשעון את זמן הפיצוץ שנקבע (קבוע)
* לחצן WAIT כל לחיצה תעצור את המניה כל זמן הלחיצה.

**שימו לב לחצן לחוץ =0 משוחרר = 1**

* לחצן resetN יאתחל את התצוגה ל **00** . עדיין לא יקרה כלום עד ל START הראשון
* טורבו- מאיץ את פעולת השעון פי 10 לצרכי (DEBUG)

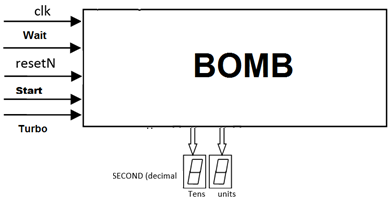
**יציאות הפצצה**:

* תצוגות 7Segments יציגו את הזמן שנותר עד לפיצוץ: בשניות.(עשרות ויחידות)

כשנגמר הזמן, התצוגות ( (7 segments יהבהבו. 00, Blink

**פעולת הפצצה**

1. בRESET הפצצה תאפס את התצוגה והמונים ותמתין ל START .
2. בלחיצה על - START הפצצה רק תטען ל זמן הפצצה שהוא -**17 שניות**
3. בעזיבת ה-START הפצצה תמנה אחורה עד לאפס (פיצוץ)
4. האותות RESET, START פעילים בנמוך (ACTIVE LOW).
5. כששעון הפצצה יגיע לאפס יהיה הבהוב בכל התצוגה בתדר של כ-1/2 Hz . עד ל - RESET חדש
6. בכל לחיצה על לחצן ,WAIT המניה תעצור כל זמן הלחיצה. בשחרור הלחצן הפצצה תמשיך לעבוד
7. הלחיצה יכולה להיות קצרה ולכן יש לדגום אותה במכונת מצבים שרצה ב50 MHz
8. יש לממש את הפצצה באמצעות ארבעה תהליכים:
   1. מכונת המצבים (תמומש בעבודת הכנה זו)
   2. מונה שתי ספרות בקוד VHDL יחיד (מסעיף קודם)
   3. מחלק תדר 1Hz (ממעבדה קודמת)
   4. תצוגה (HEXSS ממעבדה קודמת)



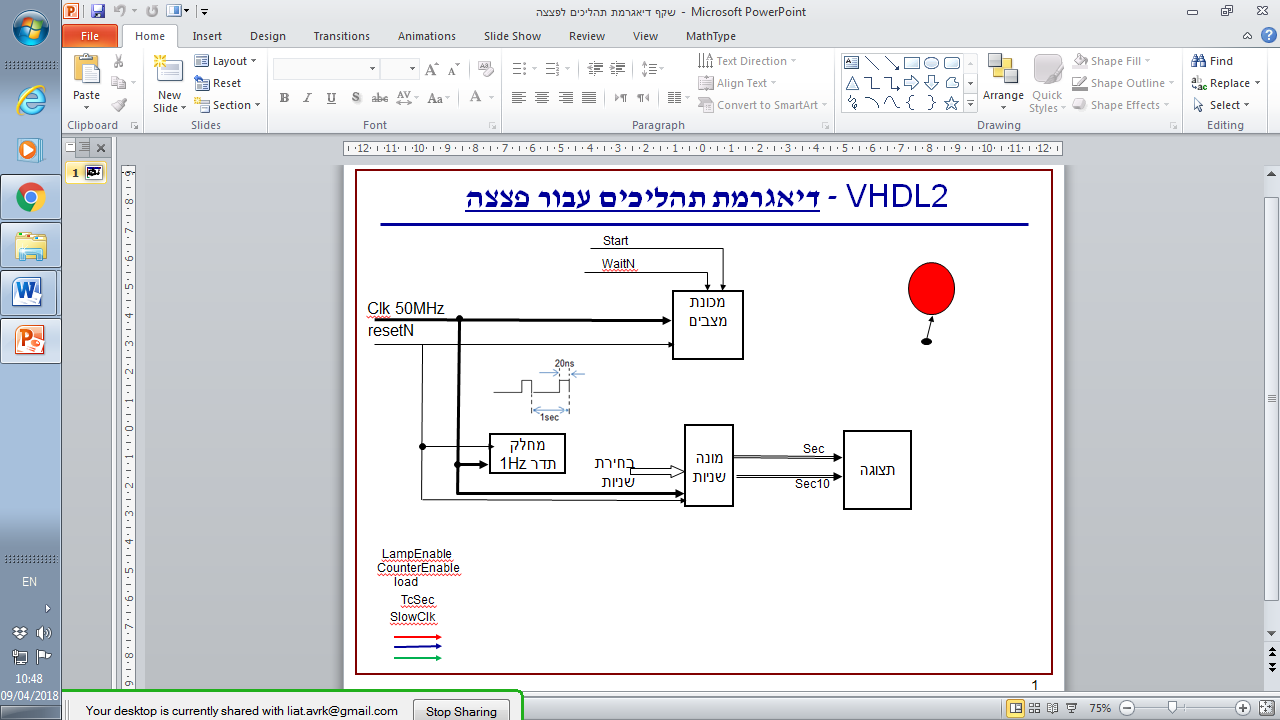
יש להקפיד להוסיףN כסיומת לסיגנלים שהם active Low (לחצנים)

**הערה:** לצורך ספירת הזמן ניתן להשתמש ביחידה שהכנתם בתרגיל קודם ולשנותה ל-BCDDN\_XX. על ידי תיקון הקוד בתוך היחידה

## ארכיטקטורה

לאחר שהבנת את דרישות התכן עליך לתכנן כל אחד ואחד מהתהליכים, חלקם ימוחזרו מפרויקטים קודמים ( בשינויים קלים) ואת חלקם תאלצו לכתוב מבראשית

הסכימה שלהלן מתארת את המודולים המרכיבים את המערכת וחלק מהקשרים ביניהם.



להגדרת המכלולים מלאו את הטבלאות הבאות :

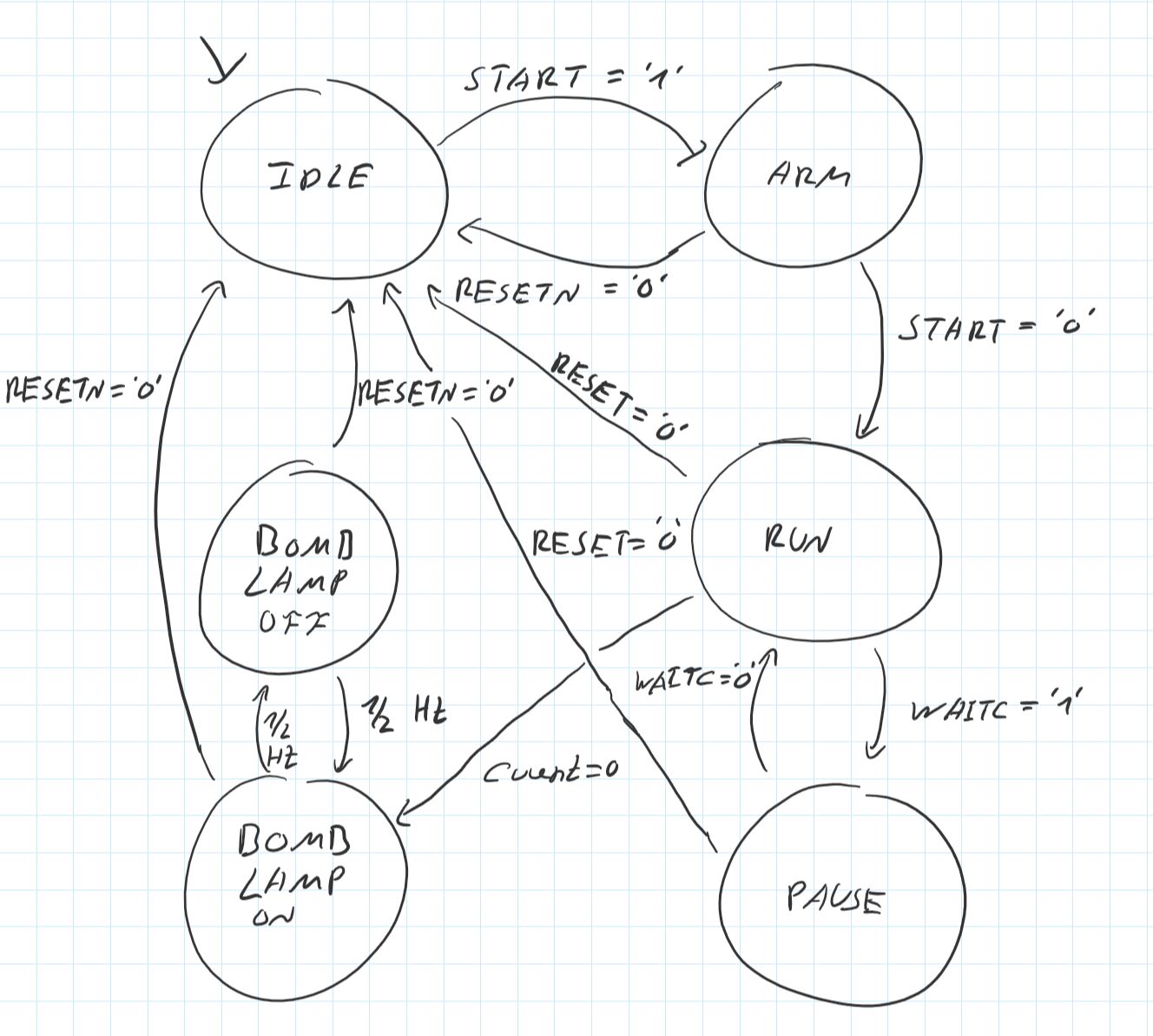
**רשום ופרט כל אחד מהתהליכים (מודולים) שתרצה למחזר מפרויקטים קודמים**

|  |  |  |
| --- | --- | --- |
| שינויים דרושים | הסבר פעולה | שם |
| מונה שניות קדימה | מונה אחורה שניות | מונה שניות |
| אין שינויים | תצוגה ספרתית | HEX SS |
| אין שינויים | ממיר אות שעון מ50MHZ ל1HZ | מחלק תדר |

**רשום ופרט כל אחד מהתהליכים שתרצה לממש מ- Scratch**

|  |  |  |  |
| --- | --- | --- | --- |
| יציאות עיקריות | כניסות עיקריות | הסבר פעולה | שם |
|  | Start – כניסה המעידה על התחלת הפעולה  WAITC – כניסה המעידה על כך שהמכונה תפסיק להתקדם  resetN – כניסה המעידה על איפוס מכונת המצבים  Clk – שעון המערכת | ניהול התהליך | מכונת מצבים |

**הגדר את מכונת המצבים ושרטט את דיאגרמת מצבים** - העזר בשבלונה המוכנה וכתוב את פירוט המעברים .



**הוסף בטבלה שלהלן את כל מצבי מכונת המצבים חסרים** , כתוב מה עושים בהם ומתי עוברים למצב הבא

|  |  |  |
| --- | --- | --- |
| **שם המצב** | **פעילות עיקרית** | **לאיזה מצב עוברים מהמצב הנוכחי ובאילו תנאים –** |
| Idle | מאפסים את המונה וממתינים | לחיצה על START מעבירה למצב ARM |
| RUN | סופר לאחור. בזמן 0 שניות מפוצץ את הפצצה. | Reset נלחץ עוברים ל-IDLE  WAIT עוברים ל-WAIT |
| ARM | טוען את הזמן 17 שניות למונה | Start משתחרר עוברים ל-RUN.  Reset נלחץ עוברים ל-IDLE |
| WAIT | ממתינים עד כניסת ה-wait חוזרת ל-0 | עבור שחרור ה-wait חוזרים ל-RUN  Reset נלחץ עוברים ל-IDLE |
| BOMB LAMP ON | מדליק את הנורות | Reset נלחץ עוברים ל-IDLE  אחרי 1/2HZ עוברים לBOMB LAMP OFF |
| BOMB LAMP OFF | מכבה את הנורות | Reset נלחץ עוברים ל-IDLE  אחרי 1/2HZ עוברים לBOMB LAMP ON |

## מימוש מכונת המצבים

השתמש בפרויקט של המונה – פתח קובץ TOP חדש בשם BOMB\_SM

כתוב וסמלט את מכונת המצבים - ראה שלד במודל (BombSkeleton.vhd)

**שים לב שהמצבים בשלד אינם זהים לחלוטין למצבים שעליך לממש**

**שם המכלול: מכונת מצבים:**

תאור פעולה מקוצר :

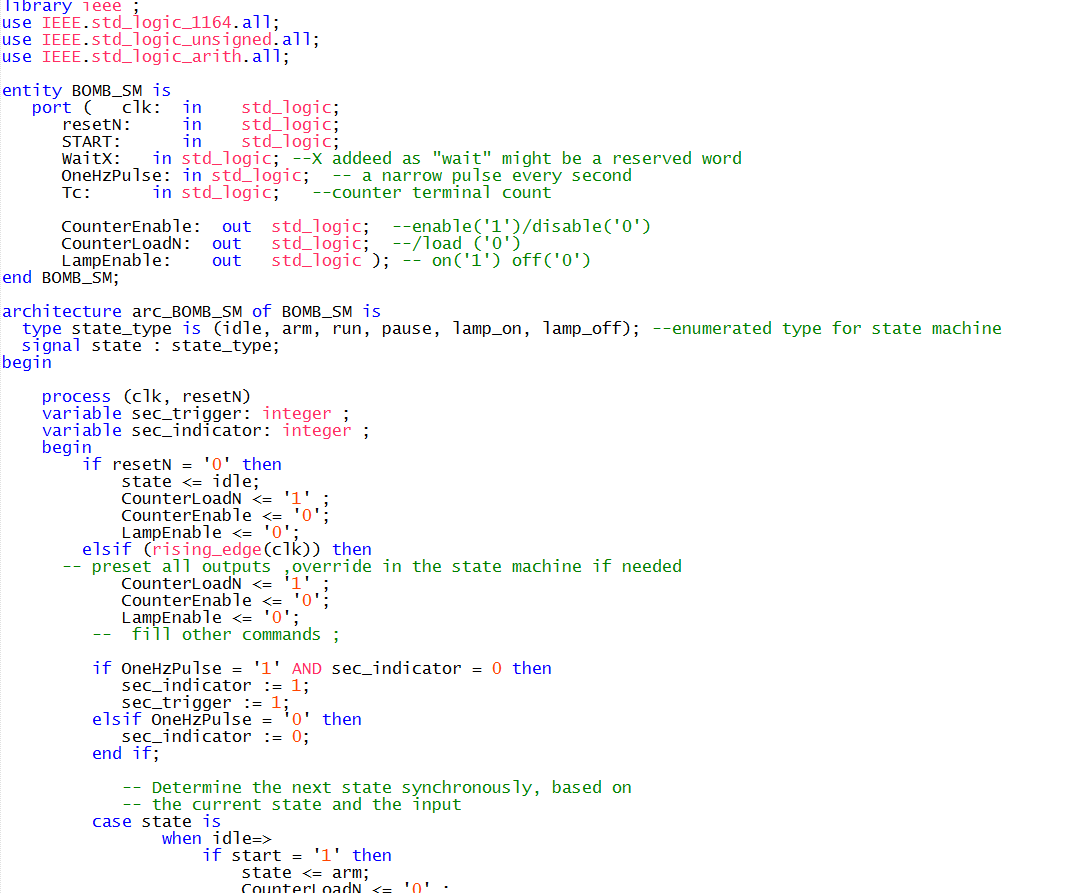
|  |
| --- |
| מודול אחראי על מעבר בין המצבים ומייצג קומפוננט של ניהול עבור המערכת. המצבים הם אלו המעידים מתי לבצע את הפעולות השונות כגון טעינת הזמן, תחילת ספירת הזמן וכו. |
|  |

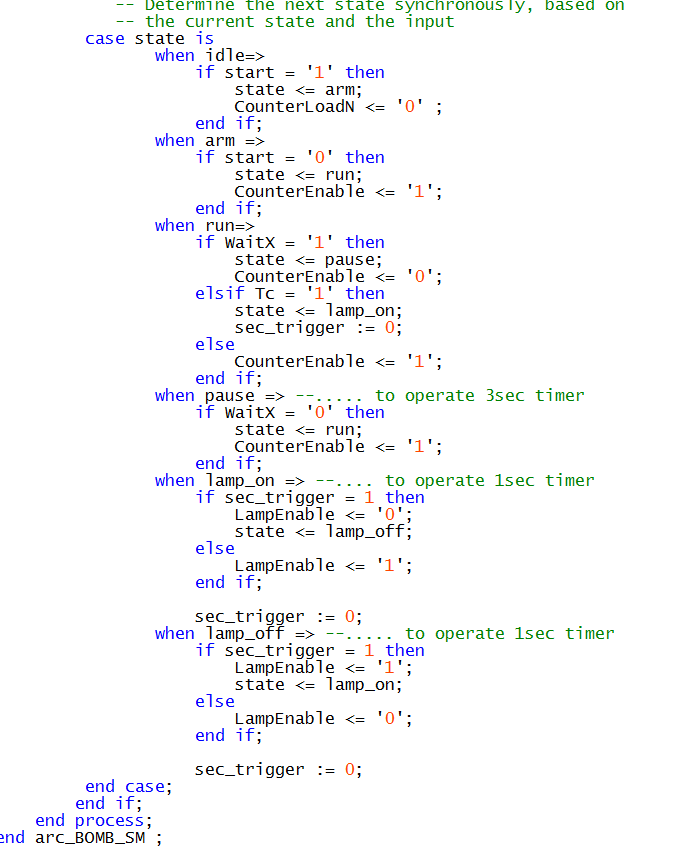
פירוט כניסות:

|  |
| --- |
| resetN – מאפס את מכונת המצבים ומחזיר למצב IDLE |
| START – בלחיצה מהווה אינדיקטור לטעינת הזמן למונה שניות. בשחרור מהווה אינדיקטור לתחילת הספירה לאחור |
| WaitX – אינדיקטור להכניס את מכונת המצבים למצב השהייה |
| OneHzPulse – שעון שעולה כל שנייה |
| TC – מעיד מתי המונה שניות סיים את פעולתו |

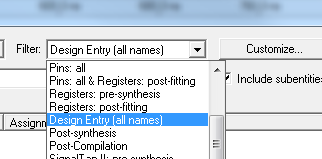
פירוט יציאות:

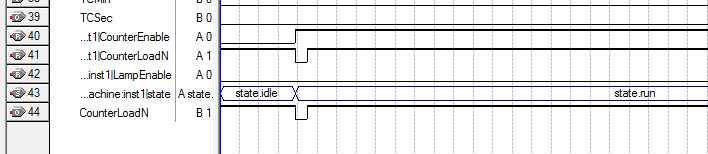
|  |
| --- |
| CounterEnable – אות המעיד על תחילת הספירה לאחור |
| CounterLoadN – מעיד על טעינת ערך חדש למונה שניות |
| LampEnable – אות המעיד על הדלקת הלדים |





ניתן לראות בסימולציה את מצב מכונת המצבים על ידי בחירת: all\_names





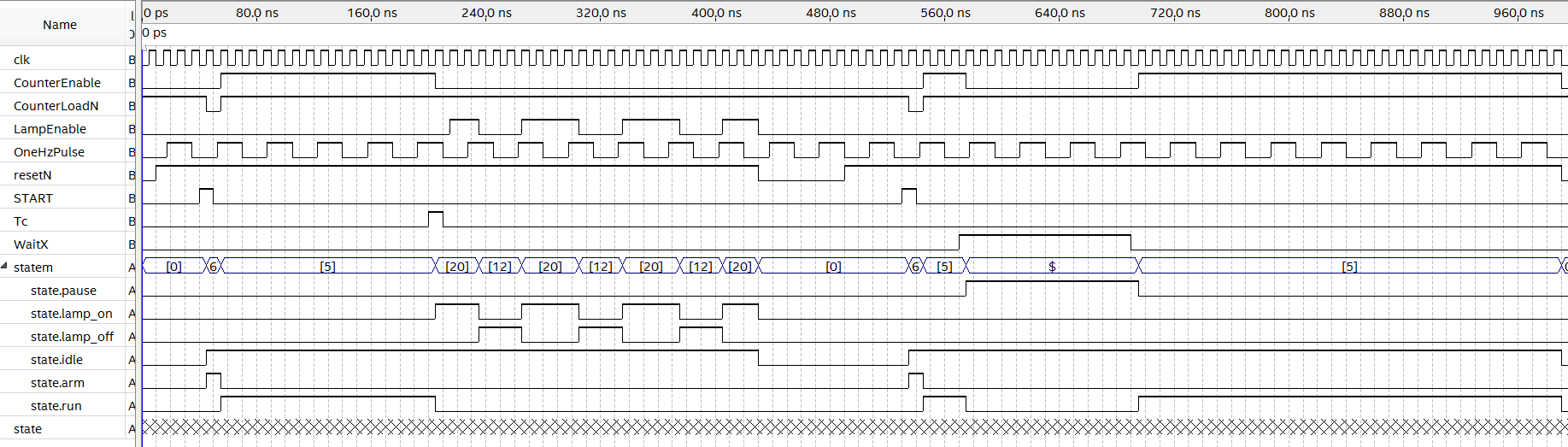
שימו לב שיש לבקש להציג בסימולטור את כל המצבים כל אחד בנפרד, מצב Idle מופיע הפוך ( בגלל BUG ופעיל בנמוך. כל השאר פעילים בגבוה)

## סימולציה למכונת המצבים

הגדר מה תרצה לבדוק בסימולציה – איזה מצבים מעניינים (המשך למלא את הטבלה)

|  |  |
| --- | --- |
| **מצב** | **תוצאות צפויות** |
| יציאה מRESET | כל היציאות מאותחלות |
| לחיצת START | מעבר למצב ARM וטעינת המונה שניות |
| עזיבת START | מעבר למצב RUN והתחלת פעולת מונה שניות |
| לחיצה על waitx | מעבר למצב pause |
| מצב led\_on | עוברים אליו לאחר TC, וממנו עוברים ל-led\_off לאחר 1/2HZ |
| מצב led\_off | ממנו עוברים ל-led\_on לאחר 1/2HZ |

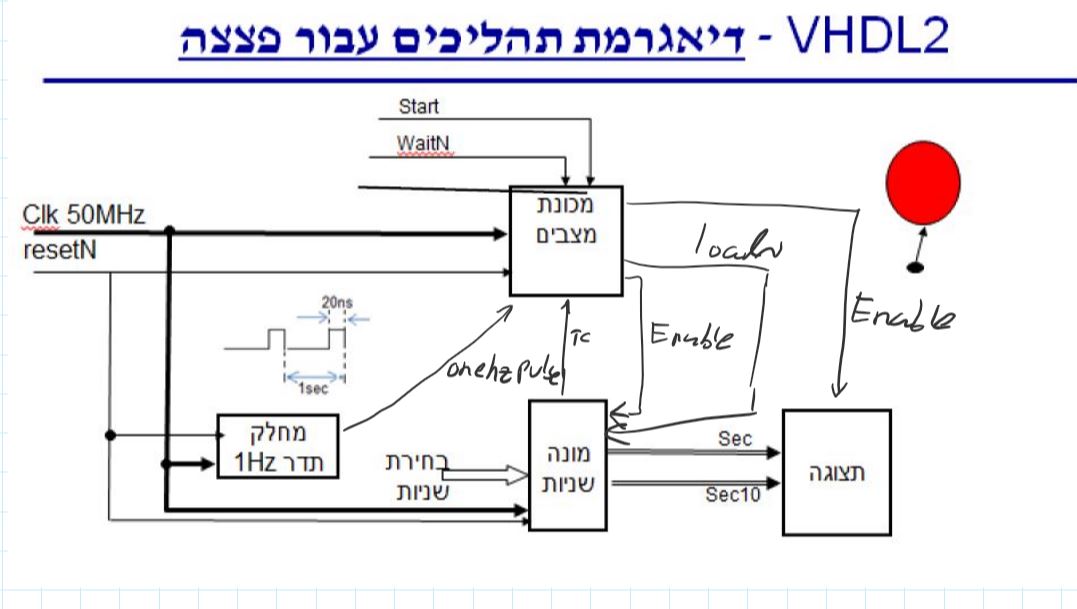
שיב לב לבחור RADIX מתאים לכל תצוגה למש ל HEX ולא בינארי



**להזכירכם- יש להביא למעבדה את כל הקבצים – כי תשתמשו בהם**

## השלמת דיאגרמת התהליכים.

לאחר שממשת את מכונת המצבים, השלם את דיאגרמת התהליכים (סכימת המלבנים) ע"י הוספת כל הקשרים בין התהליכים.

**תזכורת:**

**לבצע פעולת ARCHIVE במודל על הפרויקט שכתבתם ולהעלות במודל ל**

****

***לאחר שסיימת - לחץ על ה LINK ומלא בבקשה את השאלון המצורף***

|  |
| --- |
|  |
| |  | | --- | | [**מלא את הטופס**](https://docs.google.com/forms/d/1tO1v_J1GNnuBFqGURbuZZzX8uuGXzKp8RdkdBOaakfY/viewform?c=0&w=1&usp=mail_form_link) | |
|  |