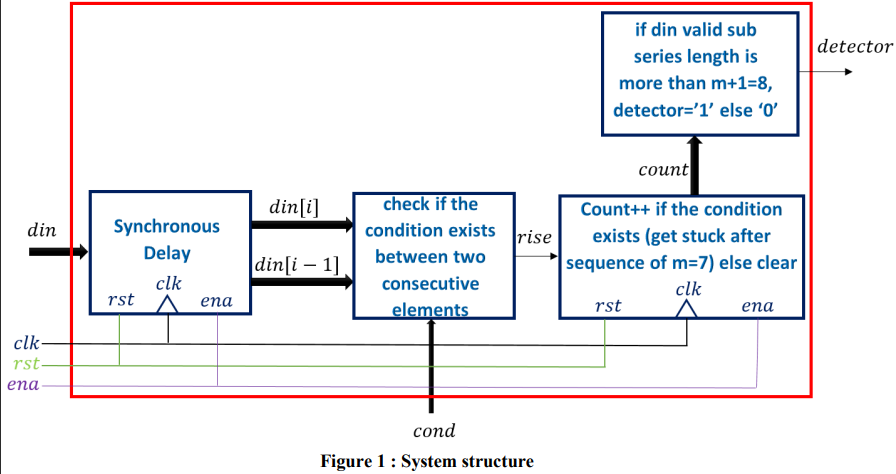
**top.vhd:**

בקובץ הזה נמצא Entity בשם top בעל הכניסות והמוצאים הבאים:

|  |  |  |
| --- | --- | --- |
| הסבר | סוג | שם הפורט |
| ביט שתפקידו לאפס את המערכת | Input bit | rst |
| סיגנל מחזורי שמסנכרן את כל המערכת | Input bit | clk |
| סיגנל אשר מפעיל את פעולת המערכת | Input bit | ena |
| ווקטור הכניסה, נדגם בכל עליית שעון (clk) | Input vector (n bits) | din |
| מספר אשר פוקד איזה סוג תת-סדרה אנו בודקים | Input integer (range 0-3) | cond |
| ביט שמאשר שתת-הסדרה תקינה | Output bit | detector |

מכיל את ה-processes הבאים: updateCondProcess, counterProc, delayProc,.

בעצם היצוג של תת מודולים הבאים: : Adder.vhd, Cond.vhd,Counter.vhd, detector\_val.vhd, SynchronousDelay.vhd הנמנצאים בתיקיה **MODELING כלל TB.**

איור המודול:

updateCondProcess:  
process זה אחראי לעדכון ביט הכניסה (cinSIG) ואחד משני וקטורי הכניסה למודול Adder (adderInSIG). למעשה, יש לעדכן את ערכים אלו כל פעם כאשר הערך של cond משתנה – כי אנחנו מתאימים אותם רק ל-cond.בנוסף הוא אחרי על בדיקת שינוי בערך של המוצא של מודול Adder (adderS). כל פעם שיש שינוי בערך הסיגנל adderS, אנחנו בודקים אם adderSהעדכני שווה ל-D\_next (שהוא בעצם din[i]). כאשר הם שווים, זה אומר ששני סיגנלים עוקבים מקיימים את תנאי תת-הסדרה הנתונה, ונעלה את הביט riseSig להיות '1'.

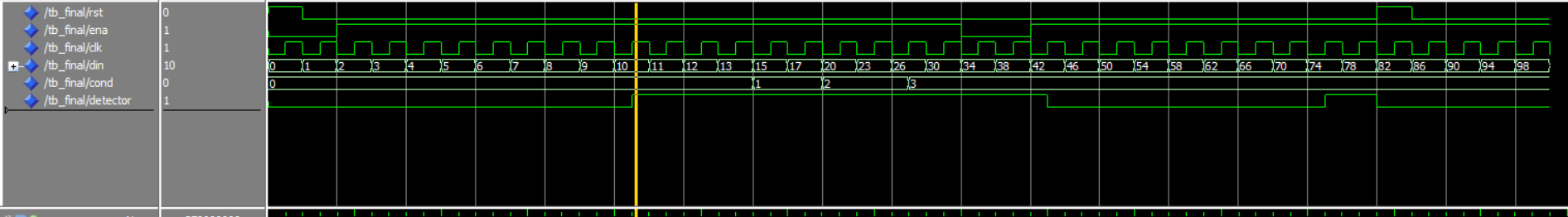
counterProc:

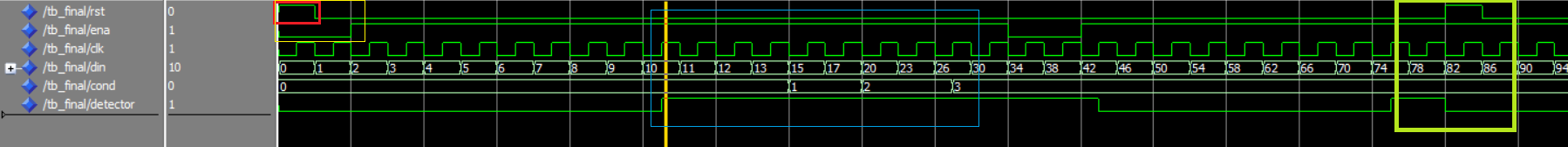
process שאחראי על ספרות כמות הפעמים שרצף תקין הופיע.

delayProc:

מייצר דילאי של מחזור ומוציא את הקלט הנוכחי וקלט שהיה מחזור 1 קודם.

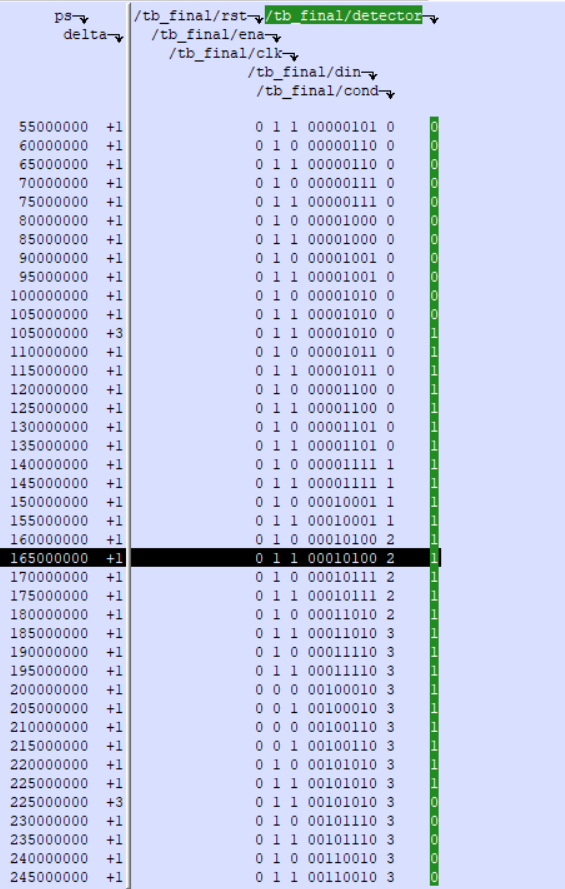
דיאגרמה נקייה:



דיאגרמה עם הסבר:

ניתן לראות מה שמסומן באדום וצהוב בהתחלה זה מצב של RST של המעגל והדלת FLAG ENABLE.אחרי זה ניתן לראות שקיים רצף של 7 תנאי שהתקיימו לכן detector עולה ל1.בTB הנוכחי ניתן לראות מצב מעניין שמסומן בצבע כחול. COND השתנה וdetector נשאר על 1 למרות שלא הופיע רצף של 7 באותו COND אף ניתן לראות שקלט השתנה גם כך ש RISE עדיין 1 ולכן detector נשאר על 1.בריבוע האחרון בצבע ירוק ניתן לראות שכאשר הדליקו דגל RST אזי פלט של מעגל התאפס.

דיאגרמת ה-LIST כדי שנוכל לראות את טבלת האמת של גרף ה-wave:



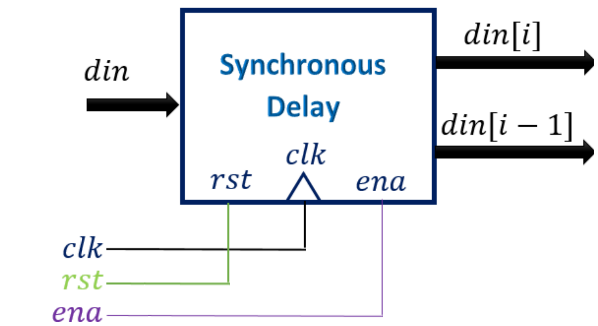
**Adder.vhd:**

מודול זה בעצם מקבל 2 וקטורים באורך n וביט בשם cin ותפקידו פשוט לחבר ביניהם. אנו משתמשים בו לשם ווידוא ששני וקטורי כניסה עוקבים מקיימים את תנאי התת-הסדרה שאותה אנחנו רוצים לבדוק (לפי הערך של cond).  
בפועל, אנחנו מחברים אליו לווקטור אחד את din[i-1], ובכניסה לווקטור השני ולביט הכניסה אנחנו מחברים אות שמתאים לתת-הסדרה שאנחנו בוחנים. לבסוף, אנחנו משווים את המוצא ל- din[i]וכך נדע אם התנאי מתקיים.

**SynchronousDelay process:**

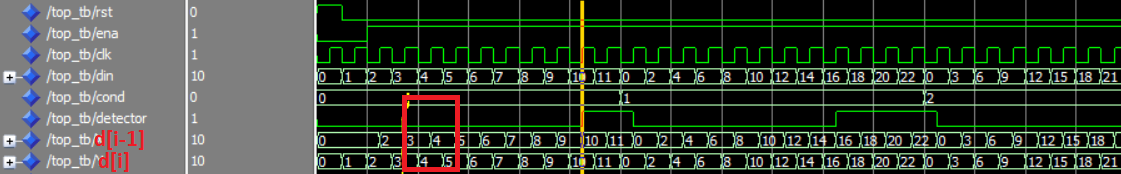
מודול זה אחראי על דגימת ערכי ווקטור הכניסה din ולאחר מכן שמירת הערך הנוכחי והערך הקודם. הדגימה נעשית בכל עליית שעון וכל עוד ena=’1’. כאשר rst=’1’ ערכי din[i] ו din[i-1]מתאפסים.  
פירוט הכניסות והמוצאים של המודול:

|  |  |  |
| --- | --- | --- |
| הסבר | סוג | שם הפורט |
| ביט שתפקידו לאפס את המערכת | Input bit | rst |
| סיגנל מחזורי שמסנכרן את כל המערכת | Input bit | clk |
| סיגנל אשר מפעיל את פעולת המערכת | Input bit | ena |
| ווקטור הכניסה, נדגם בכל עליית שעון (clk) | Input vector (n bits) | din |
| ווקטור הכניסה ברגע הנתון din[i] | Output vector (n bits) | din\_i |
| ווקטור הכניסה ברגע הקודם din[i-1] | Output vector (n bits) | din\_iMinus |



איור המודול:

דיאגרמה עם הסבר:



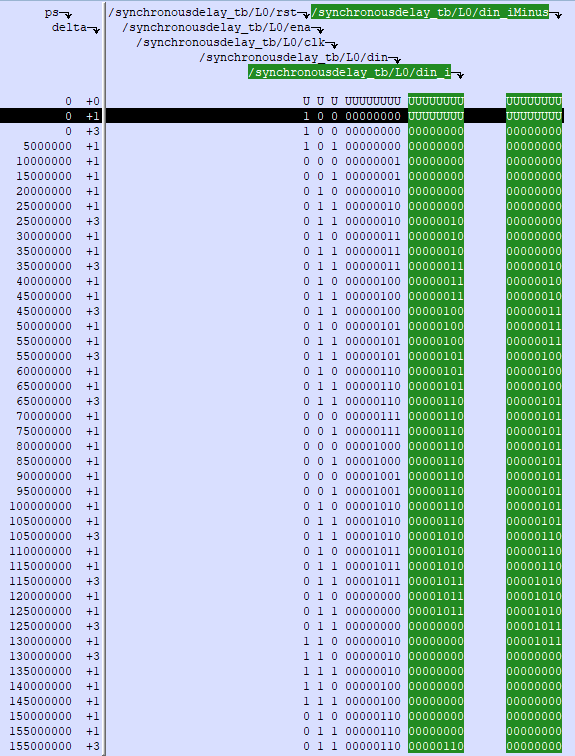
ניתן לראות במלבן האדום בתמונה למעלה שחלק של דילאי עובד כמו שהוגדר בתרגיל.הוא סינכרוני לCLK ואסינכרוני לRST.בנוסף ניתן לראות שכאשר קיימת עליית שעון אזי מודול הזה מוציא את הקלט בדילאי .

דיאגרמה של מודול מחוץ למערכת:



ניתן לראות את הדילאי כפי שהוגדר בתרגיל.

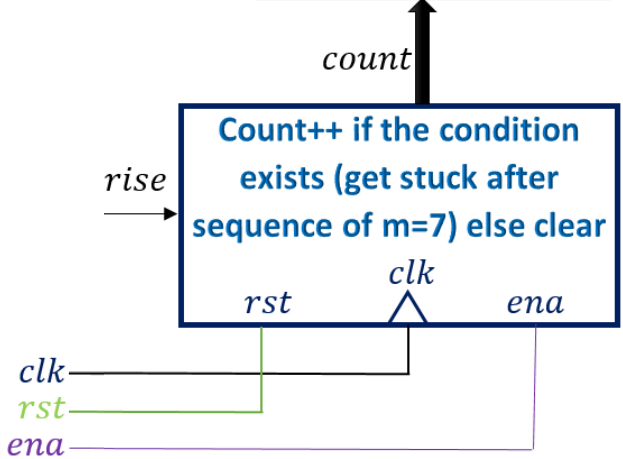
דיאגרמת ה-LIST כדי שנוכל לראות את טבלת האמת של גרף ה-wave:

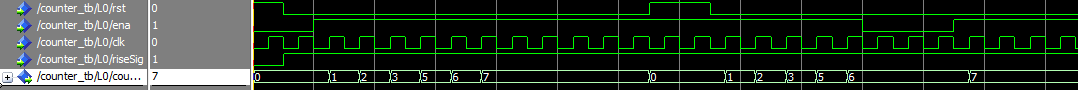


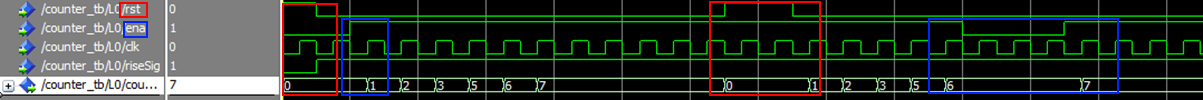
**Counter process:**

מודול זה אחראי על ספירת כמות הפעמים שהביט rise היה מורם. הספירה היא בכל עליית שעון, כל עוד ena=’1’ וכל עוד riseSig=’1’. פירוט הכניסות והמוצאים של המודול:

|  |  |  |
| --- | --- | --- |
| הסבר | סוג | שם הפורט |
| ביט שתפקידו לאפס את המערכת | Input bit | rst |
| סיגנל מחזורי שמסנכרן את כל המערכת | Input bit | clk |
| סיגנל אשר מפעיל את פעולת המערכת | Input bit | ena |
| סיגנל ביט כניסה, מורה על כך שתת סדרה מתקיימת ברגע הנתון | Input bit | riseSig |
| סיגנל שמסמן שהיה רצף באורך רצוי | input bit | counterMax |
| ווקטור מוצא שסופר כמה פעמים תת-הסדרה התקיימה | Output vector (k bits) | counterResult |

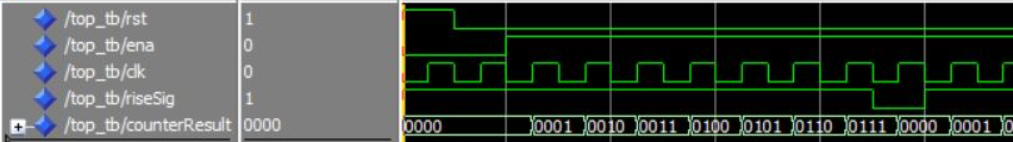
איור המודול:

דיאגרמה נקייה עבור :

דיאגרמה עם הסבר:

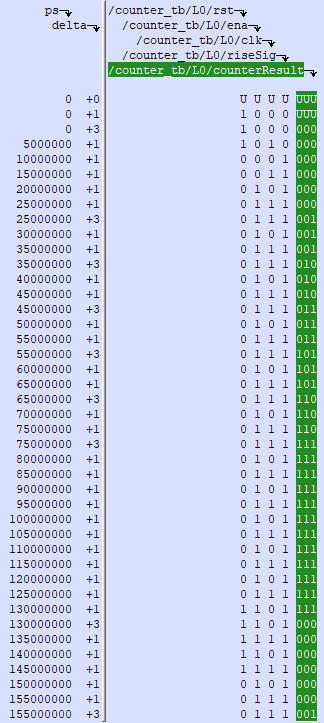
המסומן באדום: ניתן לראות שברגע ש rst=’1’המוצא ישר מתאפס, ללא תלות ב-clk.  
המסומן בכחול: ניתן לראות שברגע ש ena=’0’המוצא שומר על הערך שהיה בו לפני השינוי של ביט ena. אחרי שהוא חוזר להיות ‘1’ המערכת ממשיכה לפעול כרגיל.  
בכללי, כל עוד riseSig=’1’ ויש עלייה של clk מוצא הcounter עולה ב-1.

דיאגרמה של תת מודול מחוץ למערכת:



ניתן לראות שCOUNTER עובד כפי שמוגדר

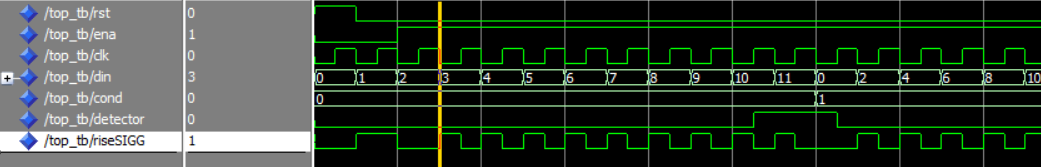
דיאגרמת ה-LIST כדי שנוכל לראות את טבלת האמת של גרף ה-wave:



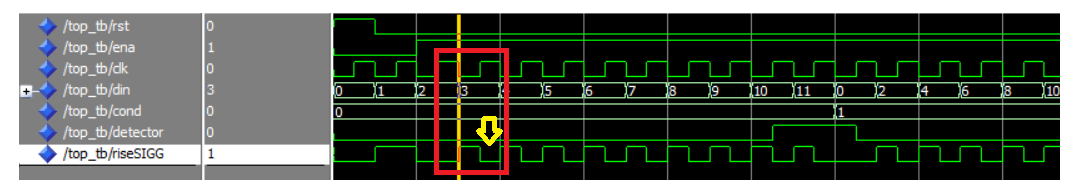
**Condition process:**

מודול זה בעצם לוקח שתי סיגנלים d[i] וגם d[i-1] ובודק האם הם מקיימים את cond . אם כן אזי RISE עולה ל1 אחרת 0.

דיאגרמה נקייה עבור :

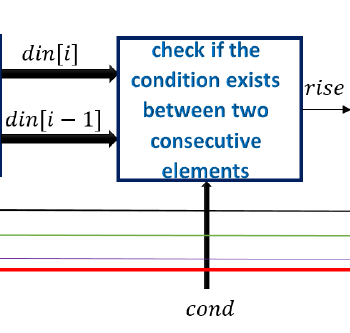


דיאגרמה עם הסבר:

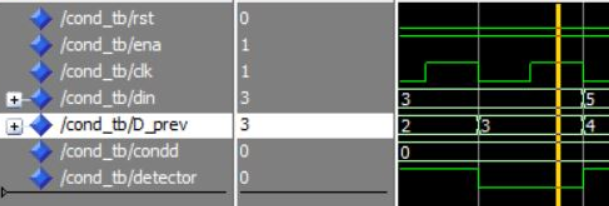


אנו עובדים בלוגיקה של עליית שעון לכן אנו דאגנו שסיגנל של RISE יהיה תקין לקראת דגימה של חלק הcounter.במלבן האדום ניתן לראות שRISE שווה לקראת עליית השעון מפני ש d\_i=3 וd\_i-1=2 וCOND בזמן הזה שווה ל0 וזה בדיוק מקיים את התנאי לכן RISE שווה ל1, לאחר עליית שעון חלק של דילאי מוציא 3 ואז כבר תנאי לא מתקיים לכן RISE יורד ל0.

איור המודול:

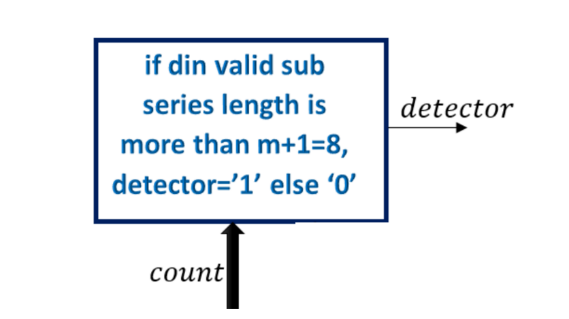


דיאגרמה של תת מודול מחוץ למערכת:



ניתן לראות שכאשר תנאי מתקיים DETECTOR עולה ל1 אחרת0

**Detector:**

מודול זה בעצם לוקח את כמות הפעמים שתתי-הסדרות קיימו את התנאי הרצוי, ומוציא '1' אם כל הסדרה תקינה – כלומר אם אורך הסדרה הוא m+1=8אחרת 0.החלטנו לממש את זה לא כPROCESS. איור המודל: 

דיאגרמה של תת מודול בנפרד מכל המערכת:

