Contents

תרגיל 1 תכנון MUX.....2
תרגיל 2 תכנון מונה 5
תרגיל 3 ניטור כור גרעיני 8
תרגיל 4 חניון אוטומטי 9
תרגיל 5 זכרונות 11
תרגיל 6 מסנן מעביר נמוכים.....18
תרגיל 7 מיני פרוייקט היסטוגרמה.....19
תרגיל 8 מכונת מצבים.....21
תרגיל 9 PWM.....23
תרגיל 10 הכרות עם הכרטיס.....24
תרגיל 11 CHIPSCOPE 30
תרגיל 12 PLL 38
תרגיל 13 -מימוש היסטוגרמה על הכרטיס 39
תרגיל 14 יצירת שעונים והפעלתם.....41
תרגיל 15 מחשבון 44
תרגיל 16 שבירת לוגיקה 45
תרגיל 17 FIFO 47
תרגיל 18 IP INTEGRATOR &I/O PLANNING 49
תרגיל 19 הכרות ותפעול עם ממשק AXI בשילוב CHIPESCOPE ו IPINETGARTOR.....61
תרגיל 20 הכרות ותפעול עם ממשק AXI 71
תרגיל 21 פתרון תקלות וקצבים משתנים.....79
תרגיל 22 AXI PERFORMANCE MONITOR 83
תרגיל 23 TIMING REPORTS.....92

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 1 תכנון MUX

<u>מטרות התרגיל :</u>

✓ התנסות בעבודה מעשית עם הVIVADO

<u>שלב א –פתיחת פרויקט והגדרת כניסות ויציאות:</u>

1. פתח פרויקט חדש שיקרא Demo_ Multiplexer
2. **הגדר את הפרויקט בדומה למה שעשיתם במעבדה מס 1 בכיתה Using the tools (באם לא זוכרים את השלבים ניתן להיעזר במעבדה הנ"ל לצורך ביצוע השלבים הראשונים של :הגדרת פרויקט –בחירת רכיב-כניסות יציאות וכו)**
3. הגדירו את כניסות ויציאות התכנון של הMUX ע"פ ההנחיות הבאות :

כניסות התכנון :(מרבב 1 ל4)

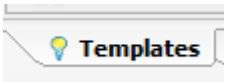
1. IN1
2. IN2
3. IN3
4. IN4
5. **Selector(1 downto 0)**

מוצא התכנון :

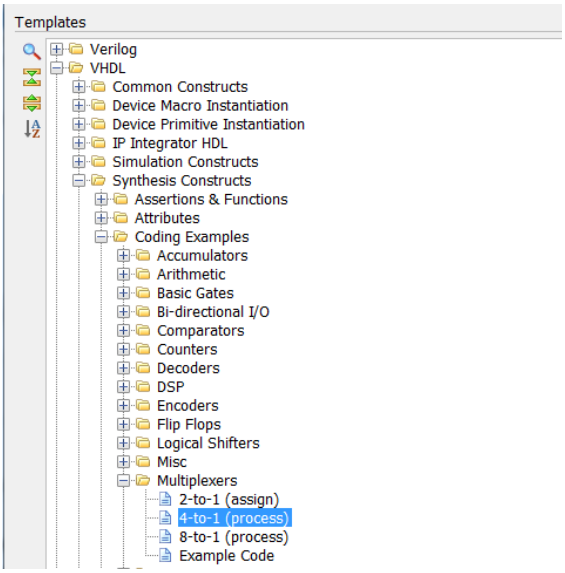
1. RES

<u>שלב ב-קידוד ה mux בעזרת דוגמא</u>

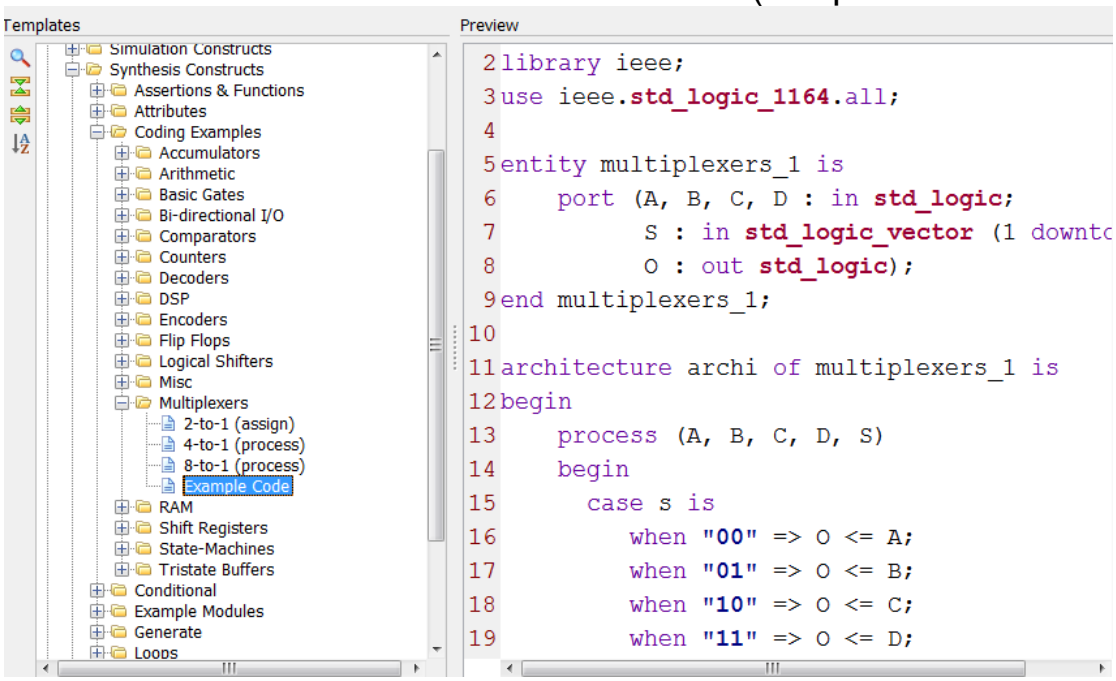
1. **פתחו את**



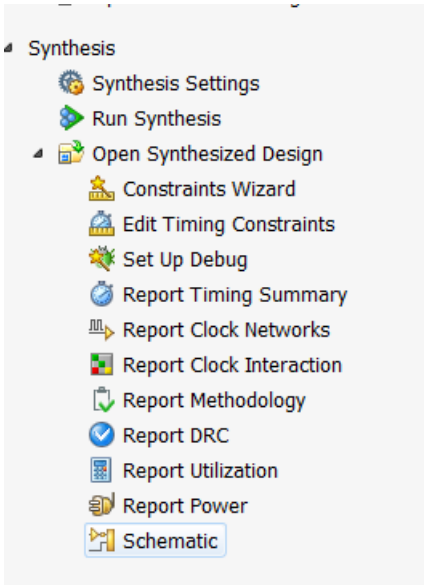
2. הוסיפו לתכן את דוגמת התכנון הבאה:



3. בדקו שאין לכם שגיאות קוד
(תוכלו להשוות את התכן שקיבלתם ל example code שמופיע שתי שורות מתחת לדוגמא שלקחתם)

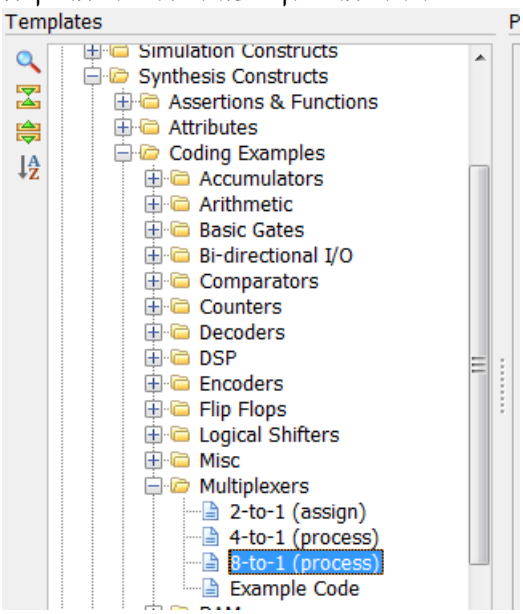


4. הריצו סינתזה וכעת בחנו את התוצאה בעזרת השרטוט שנוצר בעזרת הschematic



5. כמה LUT צורך התכנון שלכם ?
6. כעת שנו את התכנון שלכם לMUX 1 ל8 ?

7. שנו את התכנון בעזרת דוגמת התכן הבאה :



דגשים :

1. לא לשכוח לשנות את ה entity בהתאם(עדכנו את מספר הכניסות לשמונה וכן את Selector ל3 ביט)

2. לא לשכוח להוסיף לאחר כל כניסה שהגדרתם נקודה פסיק
3. שימו לב בשורה האחרונה בENTITY הנקודה פסיק מופיעה לאחר הסוגריים
8. הריצו סינתזה וכעת בחנו את התוצאה בעזרת השרטוט שנוצר בעזרת הschematic(בדומה לדוגמא שביצעתם בשלב 4)

ניר בלולו
החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 2 תכנון מונה

מטרות התרגיל :

- ✓ התנסות בעבודה מעשית עם הVIVADO
- ✓ מימוש טכניקות של בנייה היררכית וסימולציה בעזרת העקרונות שביצעתם במעבדות 1 ו2

שלב א –פתיחת פרוייקט והגדרת כניסות ויציאות:

4 פתח פרויקט חדש שיקרא DEMO_COUNER
5 הגדירו את כניסות ויציאות התכנון של המונה ע"פ ההנחיות הבאות :

כניסות התכנון :

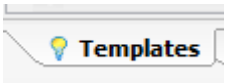
6 CLOCK
7 RESET
8 CE(אפשור המנייה)

מוצא התכנון :

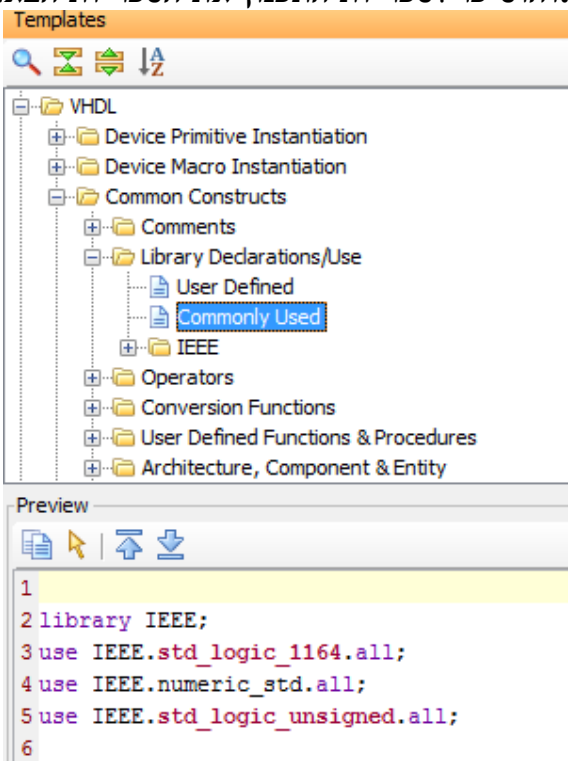
1. COUNTER (ברוחב 10 ביט)

שלב ב-קידוד המונה בעזרת דוגמא

9. פתחו את

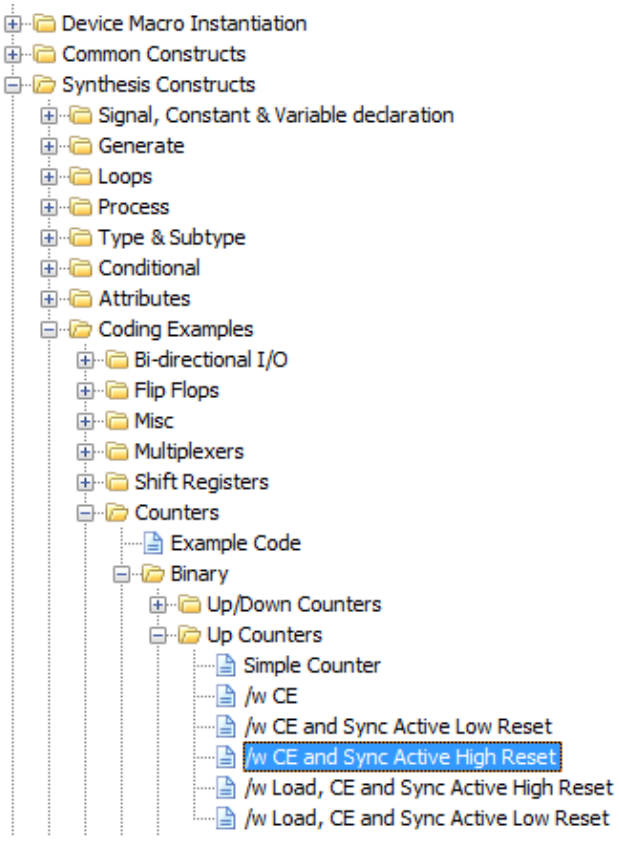


10.והוסיפו לספריות התכנון את הספריות הבאות:



נקודה למחשבה למה נדרש להוסיף את הספריות הנ"ל ?

11.הוסיפו לתכן את דוגמת התכנון הבאה



12.הוסיפו בן architecture לבין begin SIGANL ברוחב 10 ביט בשם COUNT

13.חברו את הסיגנל הנ"ל למוצא COUNTER(דגש הנ"ל יעשה בשורה נפרדת מחוץ למקטע קוד הדוגמא שהבאתם למה ??)

14.בדקו תקינות התכן והריצו את הסינתזה

שלב ג-סימולציה של התכן

השתמשו בהנחיות שקיבלתם במעבדה הבאה :

Lab 2: Using Concurrent Statements

בחלק הבא:

Simulating the Design STEP 5

בצעו סימולציה של התכן ובדקו את השתנות המונה

דגשים :

לא לשכוח לאתחל את סיגנל CEלערך' 1'

נא לדאוג לזה RESETיהיה לאחר האתחול בערך '0'

בהצלחה

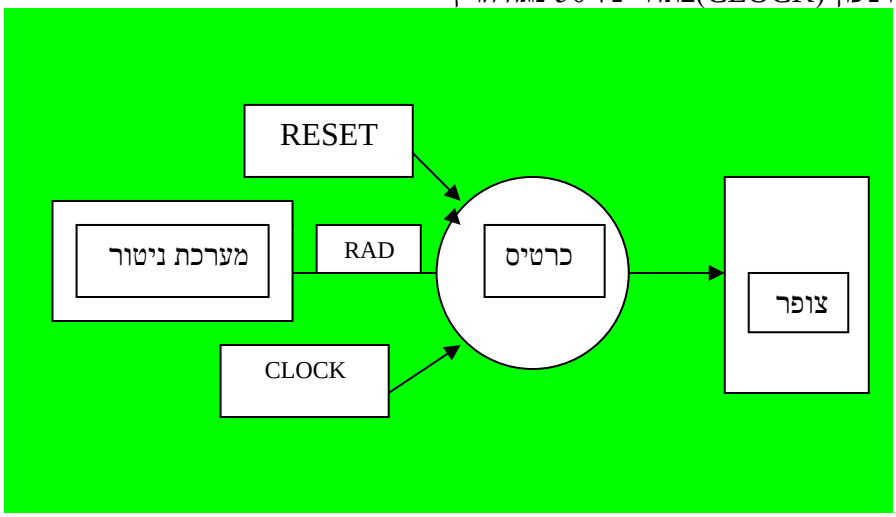
ניר בלולו

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 3 ניטור כור גרעיני

במכון מחקר גרעיני (למטרות שלום כמובן....)מצוי מכשיר לביטחונם של העובדים
מכשיר זה מנטר את רמת הקרינה הקיימת באותו זמן בתוך המכון.
המכשיר מוציא אות מחזורי אחת ל 100 מיקרו שנייה האות שיוצא הינו בעל רוחב משתנה(בהתאם לרמת הקרינה שקיימת באותו זמן).
ידוע שהמכשיר כויל לכך שאם רוחב האות עולה על 10 מיקרו שנייה השהייה במקום מסוכנת לעובדים.
עליכם לתכנן מערכת שתנטר את האות הנ"ל וברגע שתתגלה רמת קרינה מעל המותר תשלחו אות שיפעיל את האזעקה שתורה לעובדים להתפנות.
לצורך כך עומדת לרשותם החומרה הבאה כרטיס עם רכיב FPGA
בכרטיס קיים אות RESET שמופעל עם הדלקת המתחים לכרטיס
ושעון (CLOCK)בתדר של 50 מגה הרץ



<u>כניסות התכנון :</u>

1. CLOCK
2. RESET
3. RAD-סיגנל שמגיע ממכשיר ניטור הקרינה

<u>מוצא התכנון :</u>

1. ALRAM(מחוברת לצופר ,הצופר ב'1' לוגי)

<u>דגשים לתכנון :</u>

עליכם לתכנן הprocess הבאים:

1. מנייה של אורך האות
2. קבלת החלטה באם להפעיל הצופר

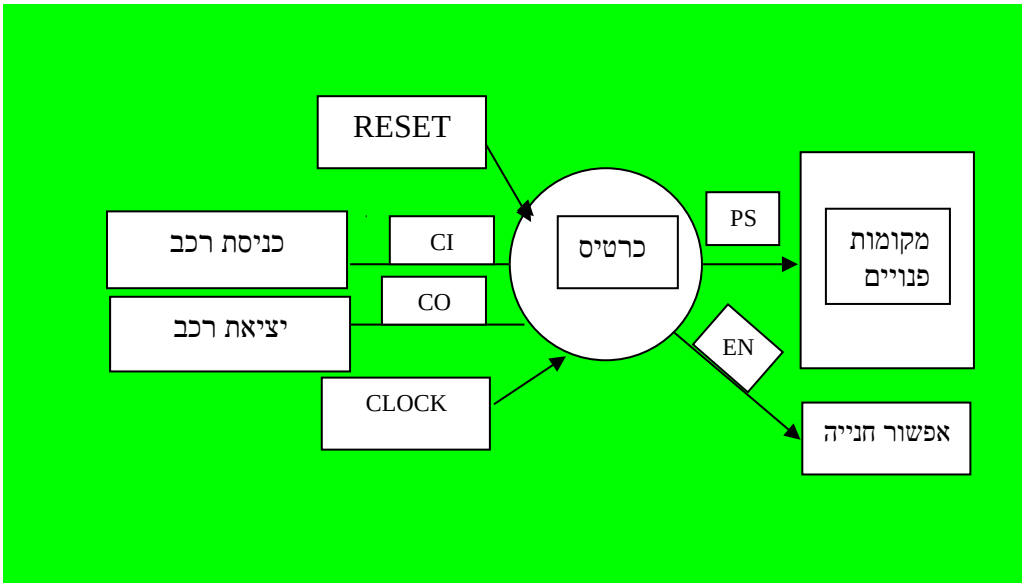
בהצלחה
ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 4 חניון אוטומטי

נדרשתם לתכנן מערכת ניהול מקומות החנייה לטובת חניון אוטומטי -המערכת נותנת חיווי על מספר מקומות פנויים בחניון .
בחניון יש כ1000 מקומות חנייה ועליכם לנהל את החניון כך שברגע שמגיעה מכונית הנהג רואה למול עיניו את מספר המקומות הפנויים בחניון ,באם אין מקומות פנויים לא תתאפשר כניסה לחניון.
לצורך כך עומדת לרשותם החומרה הבאה כרטיס עם רכיב FPGA



כל הרמת שער לטובת כניסת רכב משנה את אות הכניסה **CI** ל'1' לוגי בצורה הבאה :(למשך הזמן שהשער פתוח)

ובאופן דומה כל הרמת שער לטובת יציאת רכב משנה את אות הכניסה **CO** ל'1' לוגי בצורה הבאה:(למשך הזמן שהשער פתוח)

כניסות התכנון :

1. **CLOCK**-שעון בתדר של 100 מגה הרץ
2. **RESET** אות RESET שמופעל עם הדלקת המתחים לכרטיס
3. **CI**–סיגנל חיווי לטובת כניסת רכב
4. **CO**-סיגנל חיווי לטובת יציאת רכב

מוצאי התכנון :

1. **PS(parking space)**-מקומות חנייה פנויים בחניון-10 ביט שבהם אתם שולחים את הנתון לתצוגה של מספר המקומות הפנויים בחניון
2. **Enable-** באם יש מקומות חנייה האות הנ"ל יהיה ב'1' לוגי באם החניון מלא האות הנ"ל יהיה ב'0' לוגי

דגשים לתכנון :

<u>סנכרון אות הכניסה</u>

1. שים לב: אות הכניסה איננו מסונכרן לשעון המערכת עליכם לדאוג לסנכרון האות
2. הסבירו מהו תכנון סינכרוני? ולמה הוא כל כך חשוב ?
3. תארו את מעגל הסנכרון שעליכם לבנות ?
4. באם לא יהיה מעגל סנכרון מה עלול לקרות באיזה תופעה נתקל ?

<u>גילוי שינויים ומנייה</u>

1. שים לב: אות הכניסה רחב הרבה יותר מתדר השעון שלם
2. בכדי שתוכלו למנות עליכם לתכנן "מעגל גזירה " (לזהות את עליית הסיגנל)לרוחב שעון אחד בלבד!
3. כל כניסת רכב תעלה את המונה-וכמובן כל יציאה תוריד את המנייה

בהצלחה

ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 5 זכרונות

מטרות התרגיל

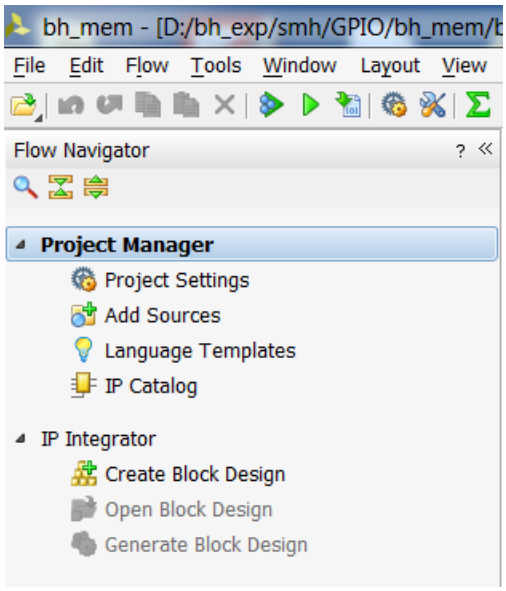
1. בניית זיכרון והתנסות בעבודה אתו-
2. הכרת סביבת עבודה IP CATALOG(הסביבה אתה אתם יוצרים הזיכרונות ושאר הIP)
3. התנסות בשיקולי תכן דיגיטלי (בניית קונספט)ראו נקודות למחשבה
4. הכנת אבן בניין חשובה למיני פרויקט שתבצעו **בעתיד הקרוב**

שלב א

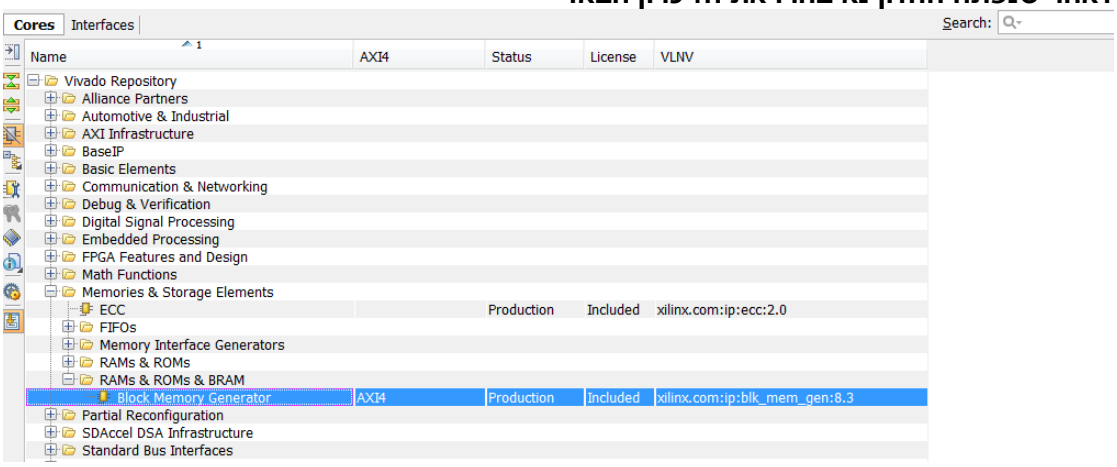
יצירת זיכרון מסוג ROM בעזרת IP CATLOG:

1. צור זיכרון מסוג ROM בגודל של 1024 תאים כל תא ברוחב של שמונה סיביות נא למלא הזיכרון בקובץ COE (לפי הקובץ ששלחתי במייל)

פתחו את IP CATLOG:



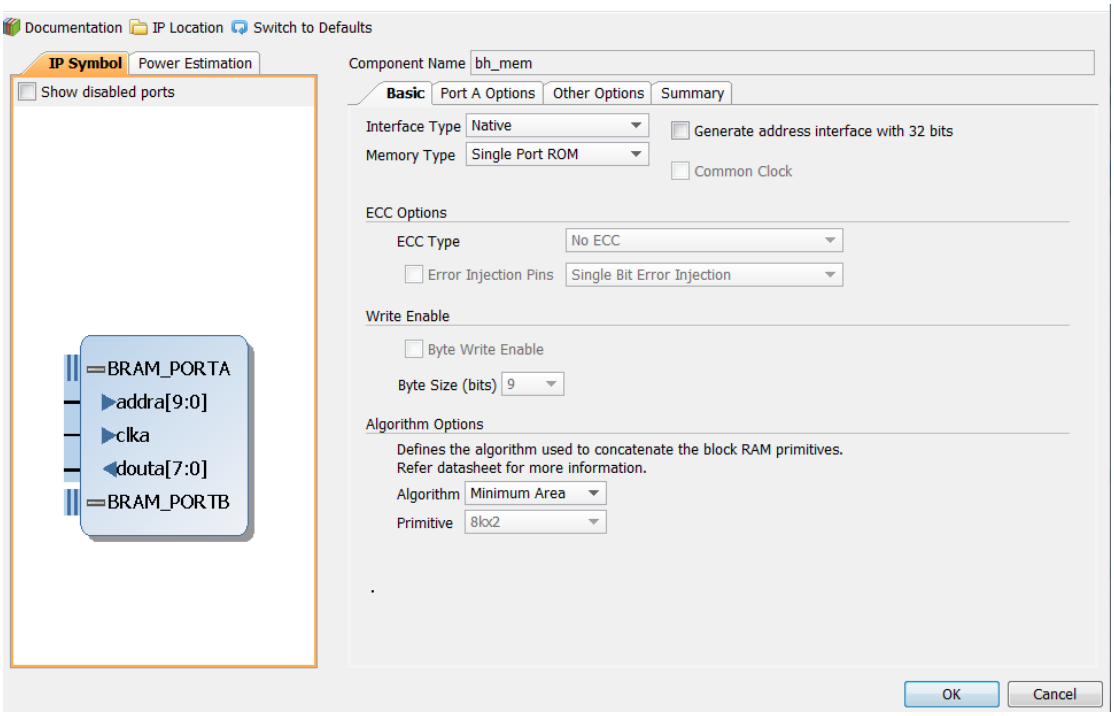
לאחר שנפתח החלון נא בחרו את הזיכרון הבא:



שאלות לחזרה

שימו לב בחרנו BRAM מה ההבדל בינם לבין distributed memory?

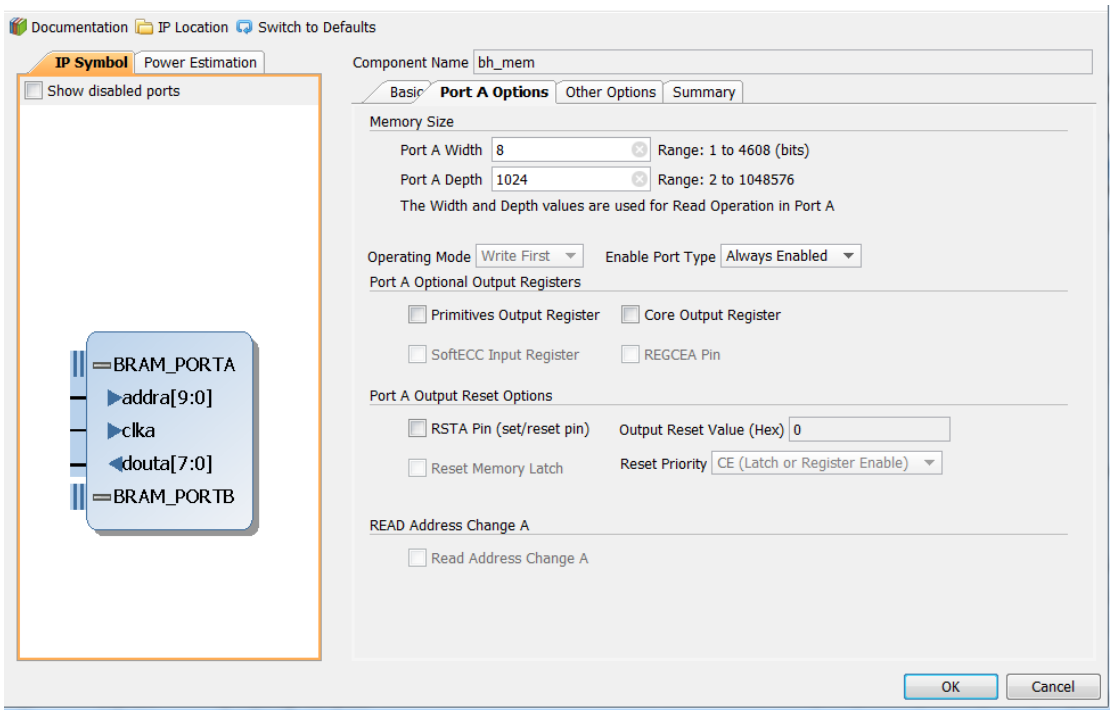
בלשונית BASIC נא הגדירו את הפרמטרים הבאים :



דגשים

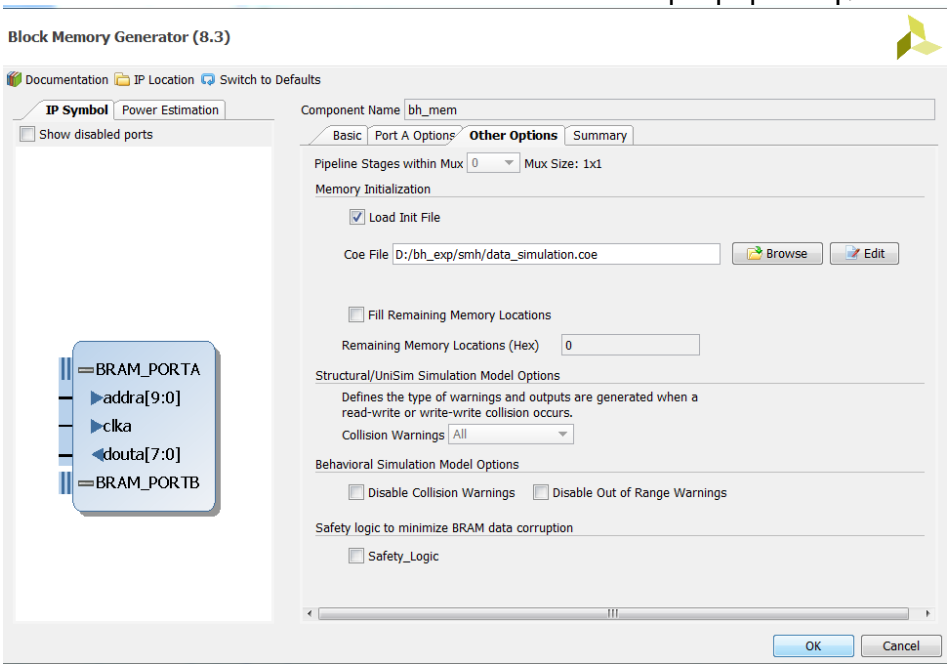
שימו לב לתמונה של הזיכרון התמונה הנ"ל משתנת בהתאם להגדרות שלכם

בלשונית PORT A options נא הגדירו את הפרמטרים הבאים :

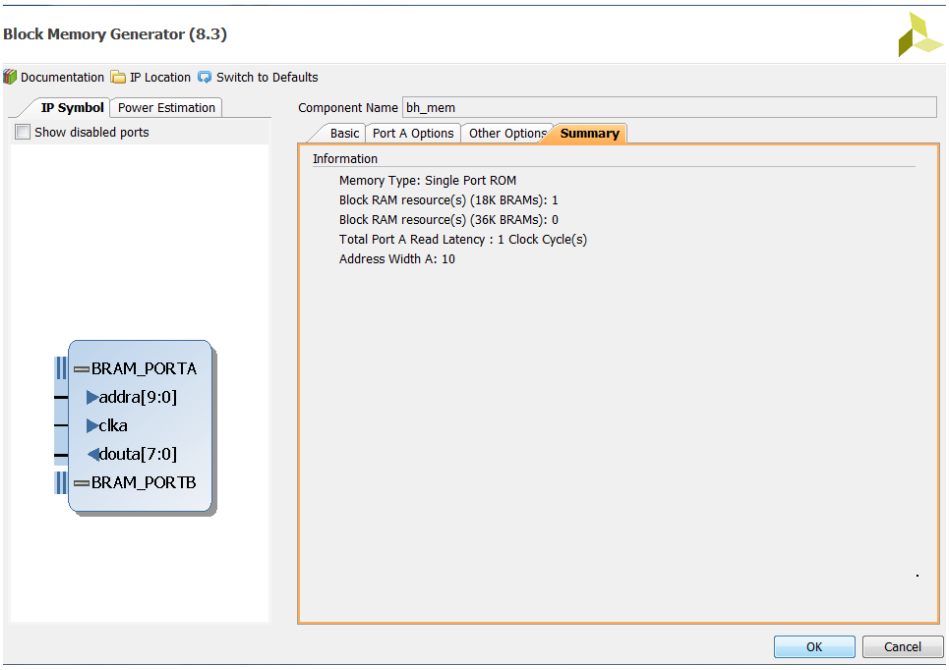


דגשים
שימו לב אתם נדרשים להוריד את V ב primitives output registers
מה משמעות החלופה הנ"ל ?

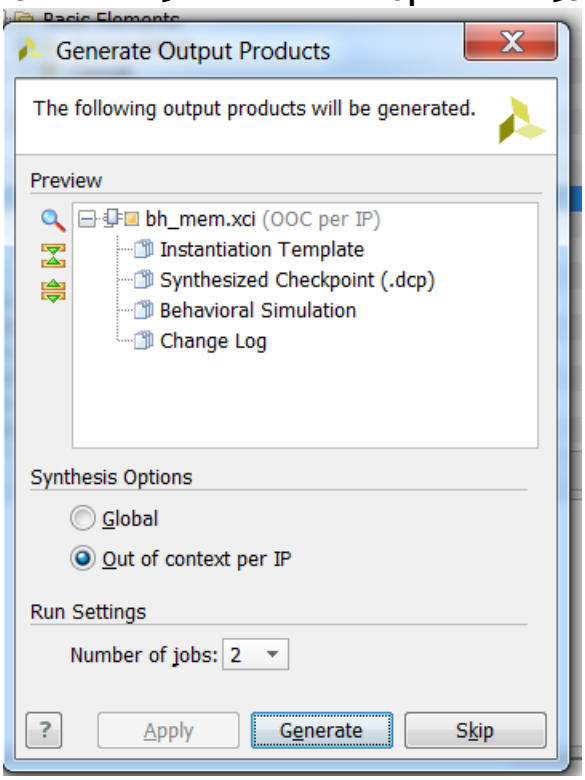
בלשונית other options נא הגדירו את הפרמטרים הבאים :
נא להטעין את הקובץ שקיבלתם ממני במייל



בלשונית summary נא בחנו את הפרמטרים הבאים :

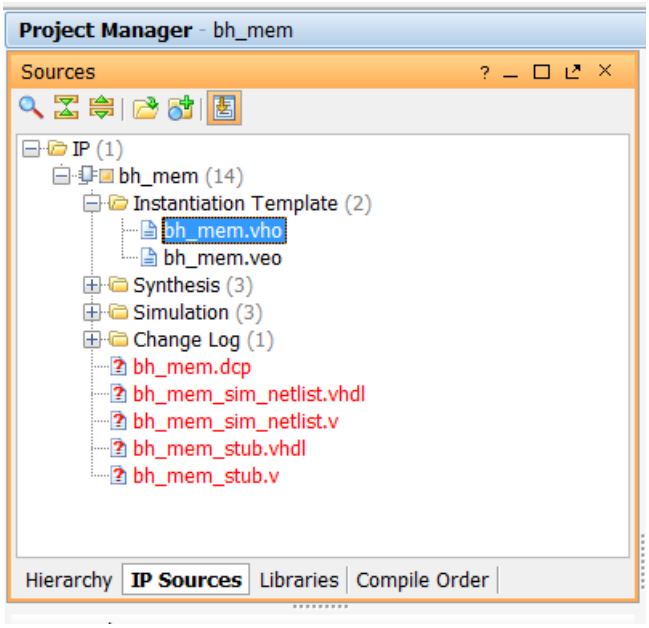


כעת תצרו בבקשה את ה IP לחיצה על GENERATE:

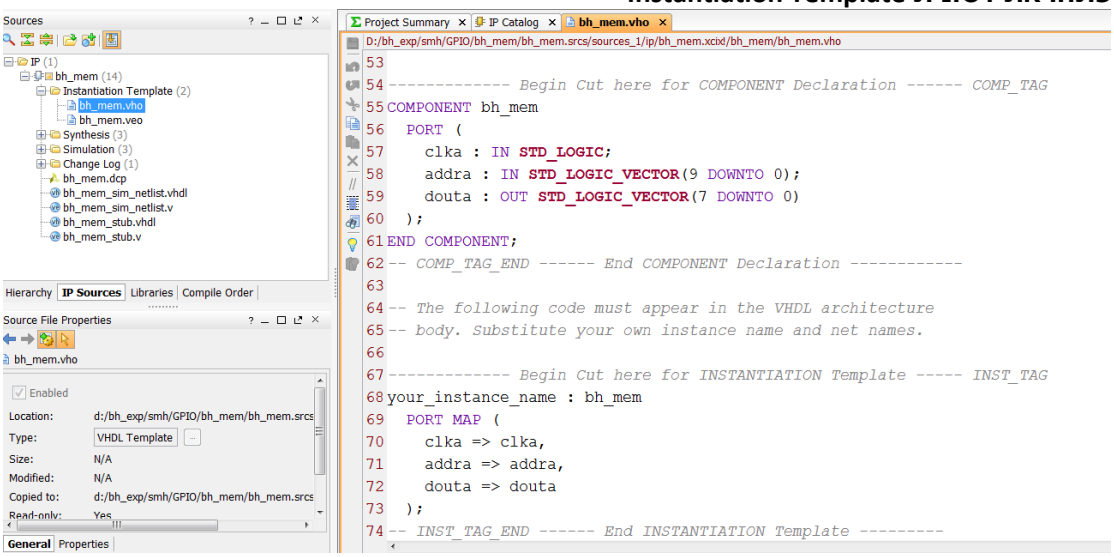


<u>שלב ב בניית סימולציה לזיכרון שיצרנו</u>

פתחו את הלשונית השנייה שנקראת IP SOURCES
שימו לב בד"כ סביבת העבודה תהיה על HIERARCHY



פתחו את לשונית Instantiation Template



צרו קובץ סימולציה חדש וסמלצו (אתם מצפים לראות את הערכים שיש בקובץ שטענתם data simulation)

דגשים :

1. **entity ריקה**
2. **component בן ה architecture לbegin**
3. **לא לשכוח להגדיר ספריות NUMERIC ו UNSIGNED**
4. **לא לשכוח להגדיר את signals ולאתחל אותם דגש על CLKA ועל ADDRA**
5. **Port map אחרי begin**
6. **צרו שעון לדוגמא :**

   clka <= not (clka) after 5 ns ;
7. **בצעו קידום של סיגנל ADDRA בצורה הבאה**

   Process

   begin

```
addra <=  addra +1;
wait for 10 ns;

end process;
```

<u>שלב ג יצירת SIMPLE DUAL PORT RAM וביצוע סימולציה של פעולות קריאה כתיבה</u>

1. צרו זיכרון מסוג ***<u>SIMPLE DUAL PORT RAM</u>*** בגודל של 256 תאים כל תא ברוחב של עשרה סיביות בדומה למה שעשיתם בסעיף א

<u>אותות הכניסה</u> :

1. **ADRESS A**-כתובת לכתיבה לזיכרון ברוחב של 8 סיביות
2. **CLK A**-שעון לPORT הכתיבה בתדר של 100 מגה
3. **DATA IN A**-המידע שנכתב לזיכרון ברוחב של 10 סיביות
4. **WEA**-אפשור כתיבה לזיכרון
5. **ADRESS B**-כתובת לקריאה מן הזיכרון ברוחב של 8 סיביות
6. **CLK B**-*שעון לPORT הקריאה בתדר של 100 מגה*

<u>אותות היציאה:</u>

- **DATA OUT B**-המידע שנקרא מן זיכרון ברוחב עשרה סיביות

2. צרו קובץ סימולציה שיבצע קריאה וכתיבת מידע מתוך הזיכרון בדומה לסעיף ב
3. הבלוק יכתוב לזיכרון לכל התאים מ0 עד 255 מספר כפול מערך הכתובת לדוגמא:
   1. לתא שהכתובת שלו מס 1 הערך 2
   2. לתא מספר 2 ערך 4
   3. לתא מספר 3 ערך 6 וכו
4. לאחר סיום הכתיבה אתם תבצעו קריאה של כל התאים ותוודאו שהערכים שכתבתם אכן נכתבו
5. **דגשים**
   1. PORTA משמש לכתיבה
   2. PORTB משמש לקריאה
   3. את הכתיבה מבצעים ע"י WEA כאשר שמים את ערך הסיגנל '1' להנ"ל שימו לב WEA מוגדר בתור BUS את ההשמה תבצעו בצורה הבאה :

```
WEA(0)<='1';
```

שלב ד

<u>התממשקות ל זיכרון מסוג ROM דרך כתיבת תכן</u>

1. השתמשו בזיכרון מסוג ROM בגודל של 1024 תאים כל תא ברוחב של שמונה סיביות שיצרת בסעיף א
2. קודד בלוק שישלוט על קריאת המידע מתוך הROM (כעת הכוונה היא לא לקובץ סימולציה אלא לקוד שעובר סינתזה ז"א תצטרכו לקודד COUNTER שעובר סינתזה)
3. הבלוק יקרא מן הROM בקצב של 100 מגה

<u>אותות הכניסה</u> :

1. **CLK** -שעון המערכת בתדר של 100 מגה
2. **RST**-RESET של המערכת

<u>אותות היציאה:</u>

2. **DATA**-המידע היוצא מROM ברוחב שמונה סיביות (סה"כ 1024 נתונים בקצב של 100 מגה)
   - **לאחר סיום הקידוד כמובן יש לסמלץ ולראות שהכול תקין!**

שלב ה

<u>התממשקות SIMPLE DUAL PORT RAM וביצוע פעולות קריאה כתיבה</u>

1. השתמש בזיכרון מסוג ***<u>SIMPLE DUAL PORT RAM</u>*** בגודל של 256 תאים כל תא ברוחב של עשרה סיביות שיצרת בסעיף ג
2. קודד בלוק שישלוט על קריאת וכתיבת מידע מתוך הזיכרון (כעת הכוונה היא לא לקובץ סימולציה אלא לקוד שעובר סינתזה ז"א תצטרכו לקודד COUNTER שעובר סינתזה)
3. הבלוק יקרא ויכתוב לזיכרון בקצב של100 מגה (ז"א שעון יקרא בלבד שעון יכתוב בלבד וכן חוזר חלילה ....)
4. לדוגמא באם נכתב לתא שכתובתו 5 את הערך 10
   עליכם במחזור שעון הבא לקרוא מהתא שכתובתו 5 את הערך 10
5. דגשים:
   - שימו לב הCOUNTER יהיה איטי פי 2 מתדר השעון ז"א 50 מגה למה ???
   - שימו לב WEA יתנדנד בן 1 ל0 בתדר השעון ז"א 100 מגה למה ??

<u>אותות הכניסה</u> :

- **CLK** -שעון המערכת בתדר של 100 מגה
- **RST**-RESET של המערכת

<u>אותות היציאה:</u>

- **DATA**-המידע היוצא מ***<u>SIMPLE DUAL PORT RAM</u>*** ברוחב עשרה סיביות (סה"כ 256 נתונים בקצב של 100 מגה)

6. **לאחר סיום הקידוד כמובן יש לסמלץ ולראות שהכול תקין!**

-

בהצלחה

ניר בלולו

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 6 מסנן מעביר נמוכים

שלב א

יצירת מודול

אותות הכניסה :

1. **CLK** -שעון המערכת בתדר של 50 מגה
2. **RESET**-RST של המערכת
3. **DATA_IN**-מידע ברוחב סיבית אחת

אות מוצא:

1. **DATA_OUT**-מידע ברוחב סיבית אחת

שלב ב

בניית מעגל סנכרון

האות שמגיע מבחוץ אינו מסונכרן צרו מעגל סנכרון שימנע בעיית מטסבליות

שלב ג

בניית מעגל הסינון

יש לבנות מעגל שיסנן את אות הכניסה

אות הכניסה מורכב מפולס מחזורי של 100 שעונים ב'1' לוגי ופולס מחזורי של 100 שעונים ב'0' לוגי

כאשר תופעת המעבר הינה 10 שעונים בן מעברים בן '0' לוגי ל'1'לוגי

בהצלחה

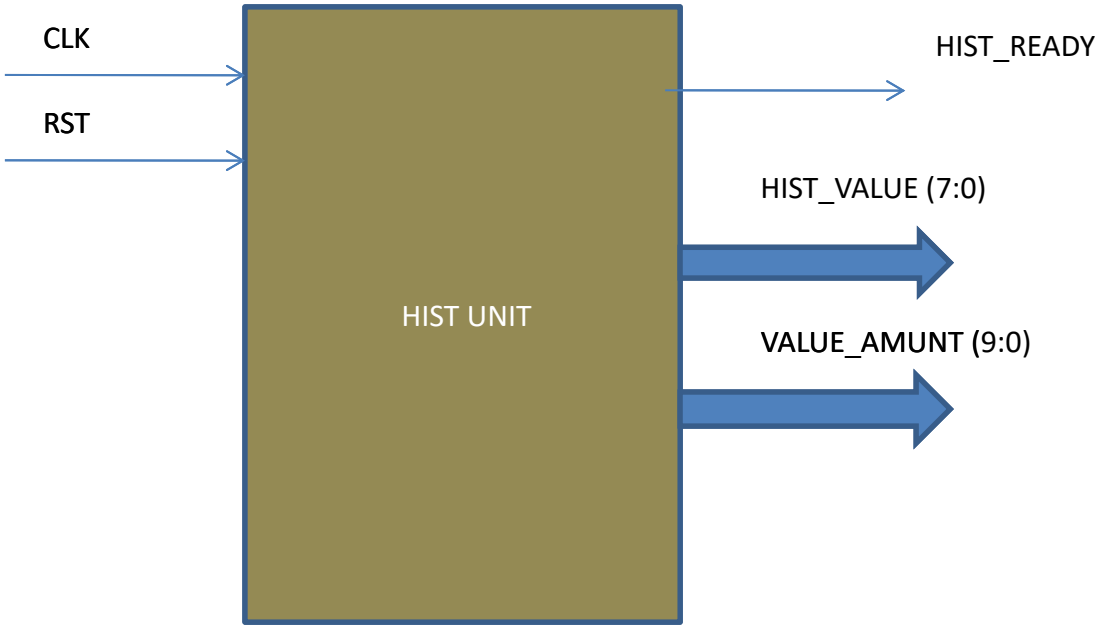
ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

תרגיל 7 מיני פרוייקט היסטוגרמה

<u>תרגיל מיני פרויקט בניית היסטוגרמה</u>
בניית דיאגראמת בלוקים שמתארת את התכנון לתרגיל כולל פרוט הבלוקים והאותות וציון תפקידם

תיאור התרגיל :
נדרשתם ע"י הלשכה המרכזית לסטטיסטיקה בסין לייצר מכשיר המקבל נתונים מנתונים שונים ובונה בעזרתן היסטוגרמה המתארת את התפלגות הנתונים שנבדקים
לדוגמא אם אנחנו בוחנים את שכיחות המשקלים השונים באוכלוסייה אנחנו מקבלים את הנתונים ומסדרים אותם כך שעל כל משקל באוכלוסייה נקבל את מספר הפרטים שיש עבור אותו משקל
לדוגמא אם ישנם 19 אנשים ששוקלים 5 ק"ג אזי תחת הנתון 5 שמציין משקל יופיע הערך 19 וכן שאר הנתונים .
דיאגראמת הבלוקים של המכשיר נראית כך :



אותות הכניסה :
1. **CLK** -שעון המערכת בתדר של 100 מגה
2. **RST**-RESET של המערכת
3. **DATA**-המידע הנכנס שמנותח בעזרת המערכת סה"כ 1023 נתונים בקצב של 50 מגה(שימו לב כניסת הDATA היא למעשה הDATA שיוצאת ממה שהכנתם בחלק א! מבלוק ROM)

<u>יציאות התכנון :</u>
<u>HIST READY</u> – אות המציין מתי המידע ביציאה תקף (פירוש הדבר גמרנו איסוף היסטוגרמה כעת אנחנו מוכנים להצגת היסטוגרמה)
<u>HIST VALUE</u>-האובייקט שעליו מדובר לדוגמא משקל 5 ק"ג
<u>HIST AMUNT-</u> ערכי האובייקט עליו אנחנו רוצים להציג הנתונים לדוגמא מתוך 1023 נתוני המשקל שאספנו במערכת במשקל 5 ק"ג הינם 19 פריטים

<u>נקודות למחשבה</u>

מה צריך להיות תדר העבודה של הזיכרון ?

(פי שניים מתדר כניסת המידע זאת אומרת אם הנתונים הינם בתדר 50 מגה עלינו לכתוב לזיכרון בתדר 100 מגה.הואיל ועל נתון שמגיע אנו מבצעים שתי פעולות פעולה ראשונה פנייה לכתובת שעליה מצביע הנתון קריאת מספר הנתונים שנצטברו באותה כתובת והוספת 1 –פעולה ראשונה פעולה שנייה כתיבת התוצאה(לאחר הוספת 1 לתוך הזיכרון)

מה מימדי הזיכרון הנדרשים?

(גודל כל תא ומה מספר התאים הכולל-גודל של 256 תאים כל גודל תא של 10 ביט)

מה מכניזם הפעולה?

(נתון נכנס- פונים לכתובת שעליו הוא מצביע קוראים את ערך התא ומוסיפים אחד לערך שנמצא בתא)

איזה סוג זיכרון נדרש?

(במה כדאי להשתמש זיכרון עם PORT אחד ?שניים וכו ?

(זיכרון עם שני PORT אחד להכנסת הנתונים בלבד

השני להוצאת הנתונים של הזיכרון בלבד)

מתי מאפסים את הזיכרון?

(בסוף כל הוצאת נתוני היסטו גרמה)

ואיך מאפסים זכרון?

(ע"י כתיבת אפסים לתוך התאים)

בהצלחה

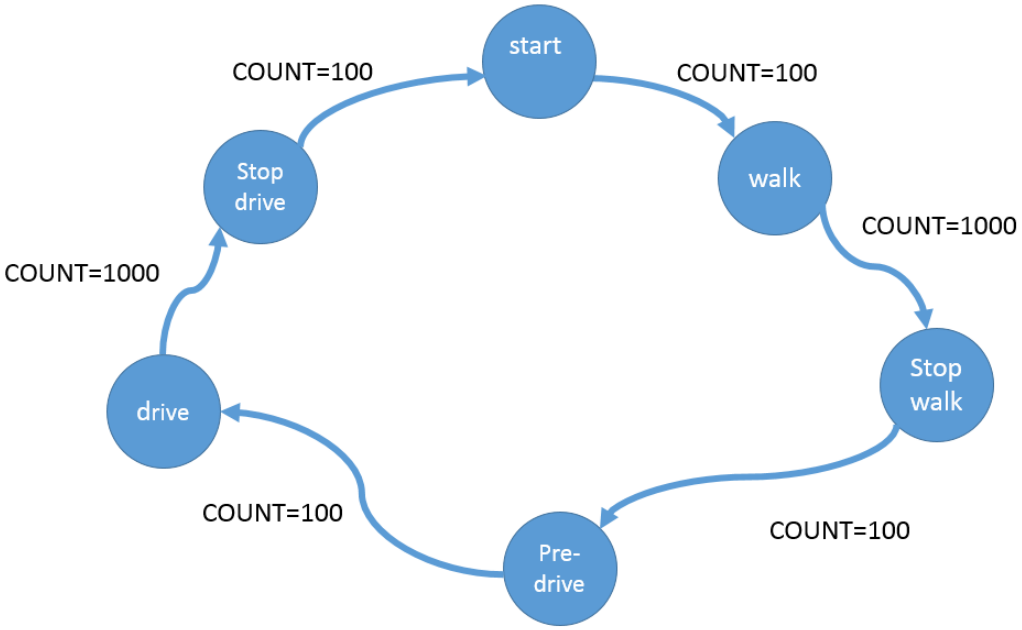
ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 8 מכונת מצבים

מכונת מצבים רמזורים



עליכם לתכנן מערכת בקרה שתשלוט בצומת הן ברמזור להולכי רגל והן ברמזור למכוניות

התכנון יהיה בעזרת מכונת מצבים מסוג MOORE (עדיף ONE PROCESS)

תנאי המצב למעבר יהיה בעזרת מניית COUNTER שיספור זמן בו מצב למצב

כניסות התכנון :

- **RST**- אתחול המערכת
- **CLK**-שעון המערכת

יציאות התכנון :

- **RED_W**- אדום להולכי רגל
- **GREEN_W**-ירוק להולכי רגל
- **RED_C**-אדום למכוניות
- **ORNAGE _C**-כתום למכוניות
- **GREEN_C**-ירוק למכוניות

מצבי המערכת :

- **START**-אתחול המערכת -תנועה אסורה הן להולכי רגל והן למכוניות
- **WALK**-ירוק להולכי רגל אדום למכוניות
- **STOP WALK**-אדום להולכי רגל עדיין אדום למכוניות
- **PRE DRIVE**- היכון לנסיעה כתום למכוניות אדום להולכי רגל
- **DRIVE**-ירוק למכוניות אדום להולכי רגל
- **STOP DRIVE -**עדיין אדום להולכי רגל -אדום למכוניות

דגשים לתכנון :

עליכם לתכנן הprocess הבאים:

3. מונה שימנה את הזמן בן המצבים השונים של מכונת המצבים
4. מכונת מצבים לפי התיאור לעיל-מכונת המצבים

בסוף כמובן לסמלץ ולראות שעושה העבודה

בהצלחה

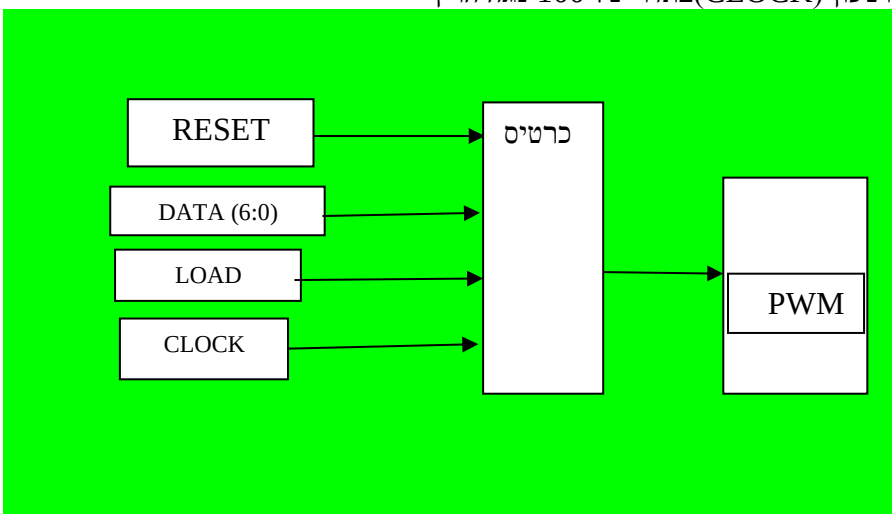
ניר בלולו

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 9 PWM

אתם נדרשים לתכנן אות מחזורי ברוחב של 1 מיקרו שנייה עם duty cycle משתנה
הערך ההתחלתי של האות עומד של על 50 אחוז (0.5 מיקרו שנייה ב'1' לוגי ו0.5 מיקרו שנייה ב'0')
אחת לסוף מחזור (כל עשרה מיקרו)אתם נדרשים לבדוק האם השתנה הרוחב
שינוי רוחב הסיגנל נעשה ע"י כניסות DATA(הערכים יכולים לנוע בן 1 ל99 אחוז)
לדוגמא 60 פירושו 60 אחוז מערך האות המחזורי יהיה ב'1' סה"כ: 0.6 מיקרו שנייה
הערך יתעדכן כאשר סיגנל LOADעולה ל1 (סיגנל LOAD עולה למשך שעון אחד לערך לוגי '1')
יש לציין שLOAD ו DATA מסונכרנים לשעון
לצורך כך עומדת לרשותם החומרה הבאה כרטיס עם רכיב FPGA
בכרטיס קיים אות RESET שמופעל עם הדלקת המתחים לכרטיס
ושעון (CLOCK)בתדר של 100 מגה הרץ



<u>כניסות התכנון :</u>

4. CLOCK
5. RESET
6. LOAD-סיגנל שנותן הפקודה לעדכן את רוחב האות המחזורי
7. DATA- נותן את הערך באחוזים של הסיגנל

<u>מוצא התכנון :</u>

2. PWM(אות מחזורי ברוחב של 1 מיקרו שנייה עם duty cycle משתנה בהתאם לערך)

בהצלחה
ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 10 הכרות עם הכרטיס

מטרת התרגיל :

הכרות עם חלק מהתקני כרטיס NEXYS4DDR(מתגים ותצוגת 7 ספרות)

- התנסות בעבודה מעשית עם הכרטיס

1 **NEXYS 4 DDR GPIO Demo- היכנס לקישור הנ"ל**

https://reference.digilentinc.com/learn/programmable-logic/tutorials/nexys-4-ddr-gpio-demo/start

בקישור בצעו הורדה לקובץ ZIP שמצוין לעיל

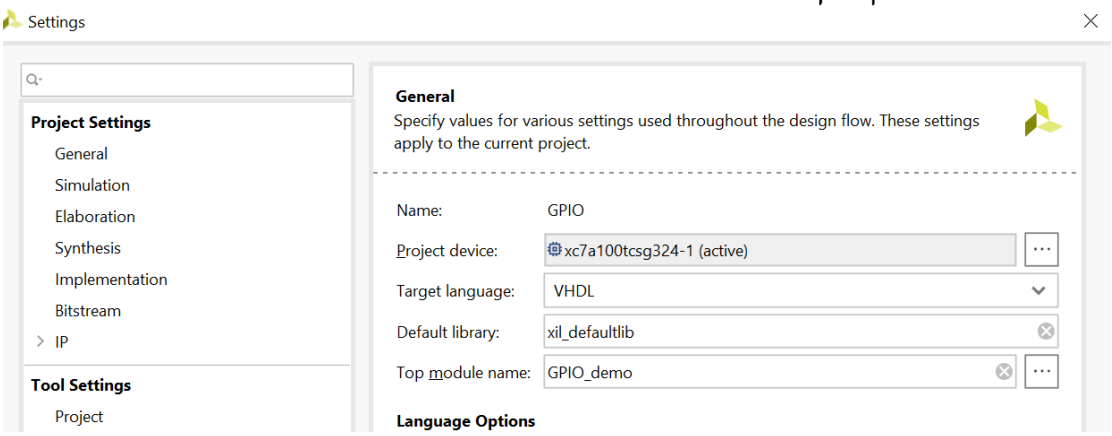
—

Downloads

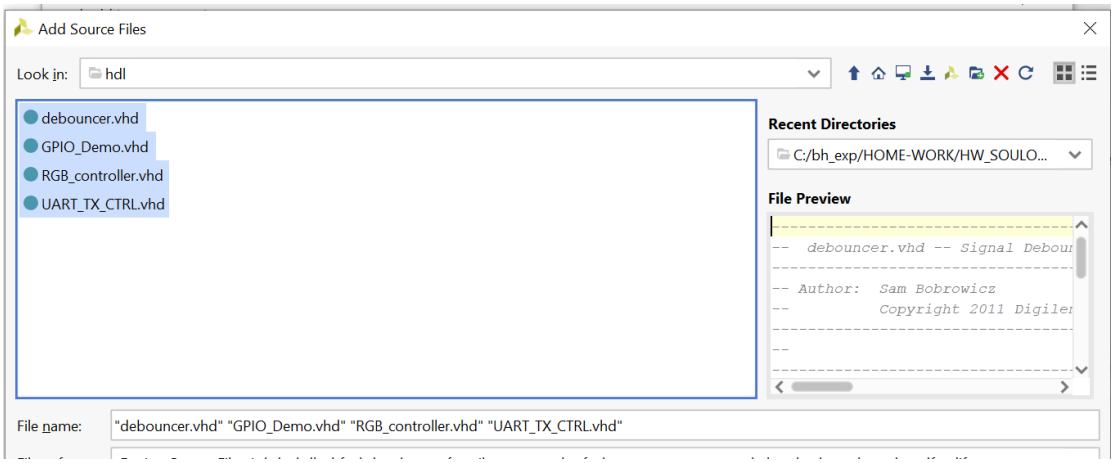
Nexys 4 DDR GPIO Demo Project Repository – ZIP GIT Repo

פתחו פרויקט חדש שיקרא **GPIO**

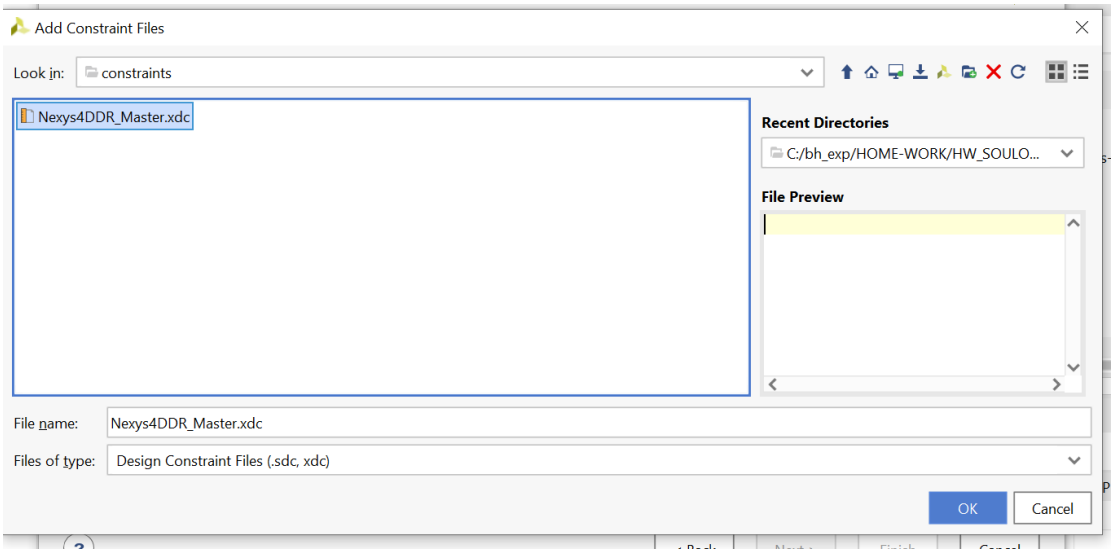
הגדירו את הפרויקט כך :



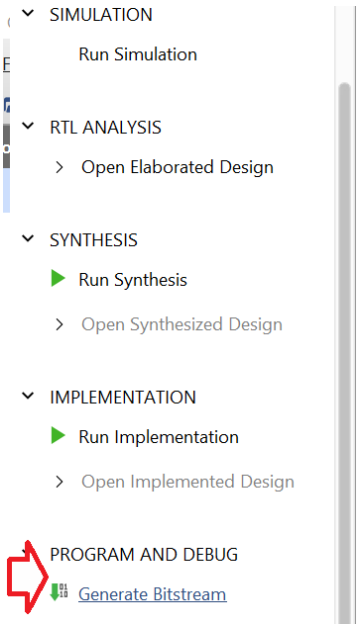
תחת תיקייה שנקראת hdl שנמצאת תחת src הוסף לפרויקט את הקבצים הבאים



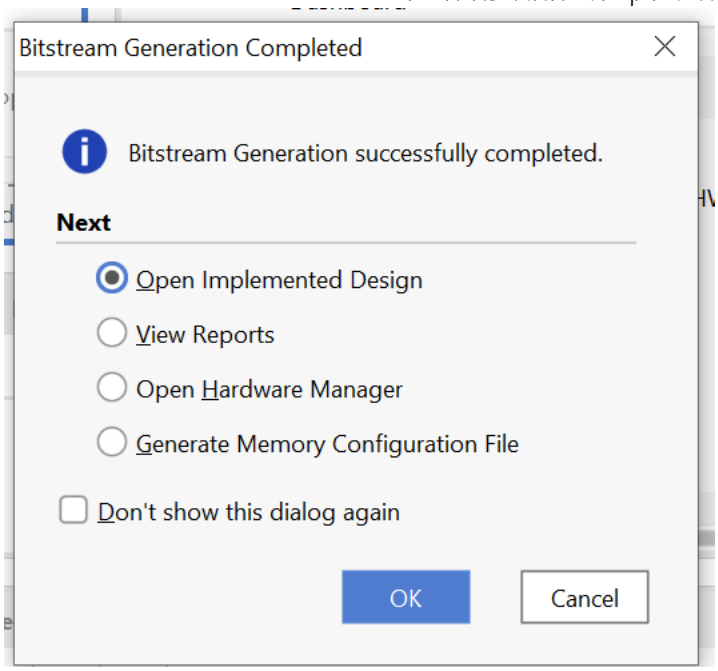
תחת תיקייה שנקראת constraints שנמצאת תחת src הוסף לפרויקט את הקובץ הבא



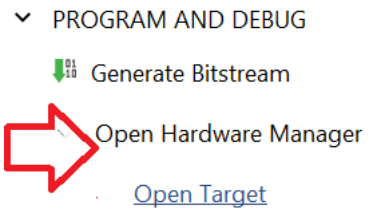
הרץ התכן עד לקבלת קובץ עם סיומת BIT כמתואר לעיל



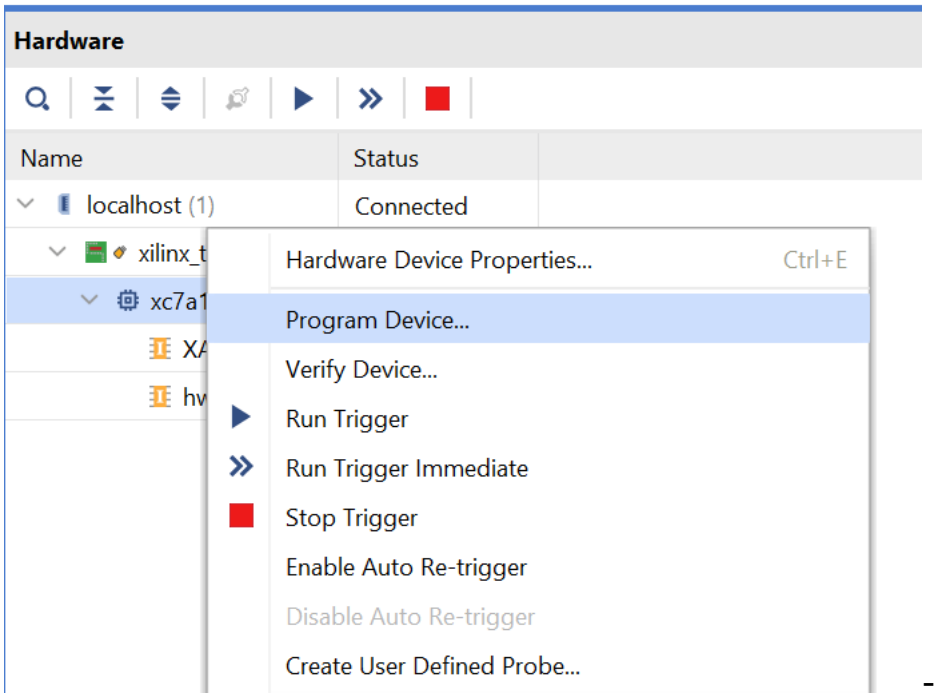
חכו לקבלת ההודעה הנ"ל :



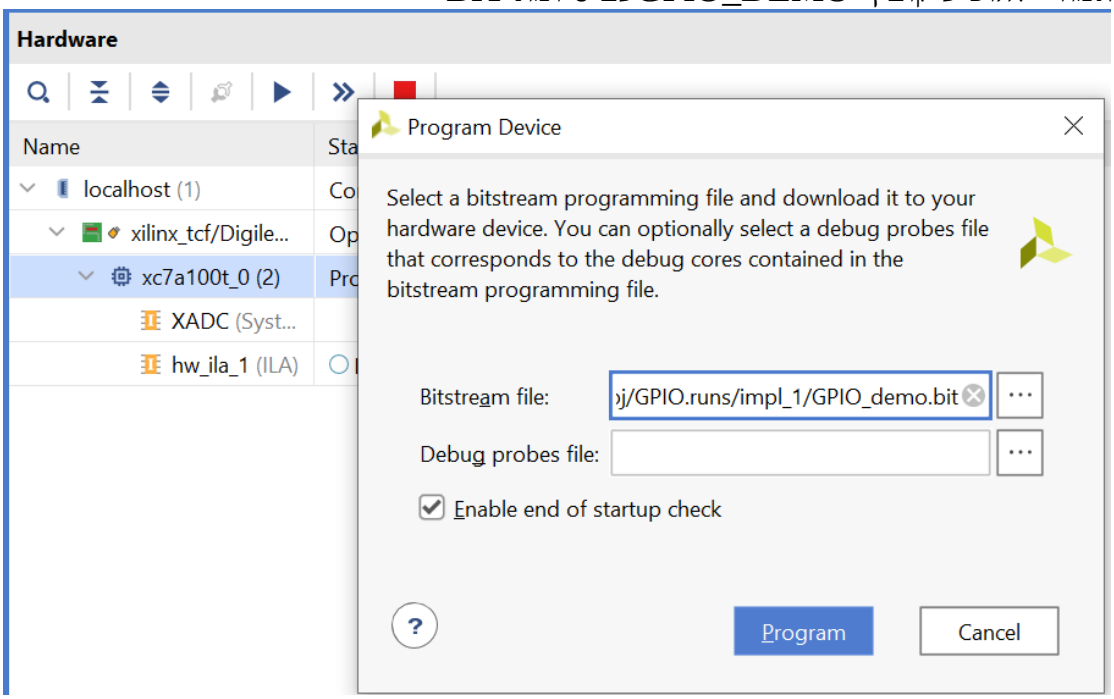
חברו הכרטיס והדליקו אותו
פתחו תוכנת HARDWARE MANGER כמתואר לעיל
לחץ על open target



ודאו שהקובץ נצרב לתוך הכרטיס כמתואר לעיל



אמור להופיע קובץ GPIO_DEMO עם סיומת BIT



צרבו את הקובץ
שמרו עותק MASTER בצד של הפרויקט בצד
ובצע הפעולות הבאות:
2. ממשו COUNTER בעל שישה עשר סיביות(ללא CE)
תכנן את תדר המניה כך שתוכל לראות התוצאות ע"ג הלדים(הנורות הצהובות בכרטיס)LED
בשביל זאת כנסו לתכן של GPIO DEMO קודדו את COUNTER
והיכנסו למקטע קוד הבא :

כל שעליכם לעשות הוא להחליף את האפסים שמסומנים עם החץ האדום ב counter שיצרתם

Project Summary | GPIO_Demo.vhd

E-WORK/HW_SOULOTION/Nexys-4-DDR-GPIO-2016.4-1/Nexys-4-DDR-GPIO/proj/GPIO.srcs/sources_1/imports

```
247  begin
248
249  ------------------------------------------------------------
250  ------                LED Control                  -------
251  ------------------------------------------------------------
252
253  with BTN(4) select
254      LED <= SW            when '0',
255          "0000000000000000" when others;
256
257  ------------------------------------------------------------
258  ------            7-Seg Display Control            -------
```

כמו כן שנו את את התנאי ל'0' when

בשלב שני חזור על הנ"ל באמצעות **DISPLAY 7 SEG**

לשם כך כנסו לקישור הבא

https://reference.digilentinc.com/reference/programmable-logic/nexys-4/reference-manual

וקראו איך עובדת תצוגת 7 סיביות :

מבקש שתתכננו את הנ"ל בשני שלבים

1. תצוגה קבועה של ספרות 1 עד 8 בהתאמה על אחת מהתצוגות
2. מנייה דינמית של מונה כל הזמן מתקדם

תכנן את תדר המניה כך שתוכל לראות התוצאות ע"ג

DISPLAY 7 SEG (שימו לב צריך לבצע המרה מבינארי לעשרוני)

3.במודול שהורדתם ישנו מודל של תקשורת UART TX מבקש שתפתחו טרמינל במחשב ותקלטו את ההודעה שמתקבלת ע"פ לחיצה על הכפתורים מבקש שתשנו את זה לשמכם הפרטי ותוודאו שאכן זה מה שמופיע בטרמינל

4השתמש בשישה עשר המפסקים בתור אמצעי להזנת מספר בינארי שיוצג בתור מספר עשרוני DISPLAY 7 SEG

הצג תוצאות באמצעות **DISPLAY 7 SEG**

5.משימת רשות בשלב זה –הצגה דרך ערוץ VGA (חברו זאת למסך) לקבלת תמונה סינטטית שתצרו כרצונכם

בהצלחה
ניר

כל החומרים לשימוש פנימי בלבד אין להשתמש ללא רשות מניר בלולו

בס"ד

תרגיל CHIPSCOPE 11

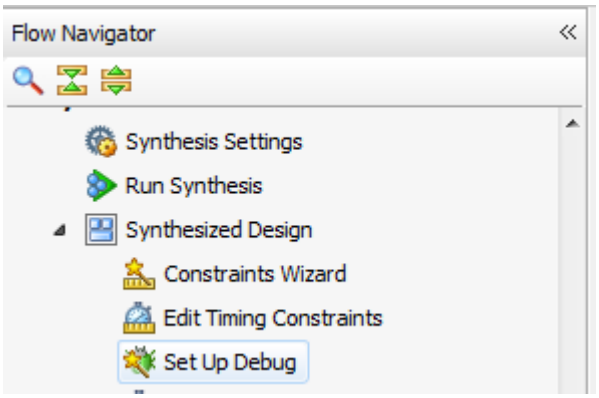
מטרת התרגיל :

✓ התנסות בעבודה מעשית עם הILA במצבי העבודה השונים

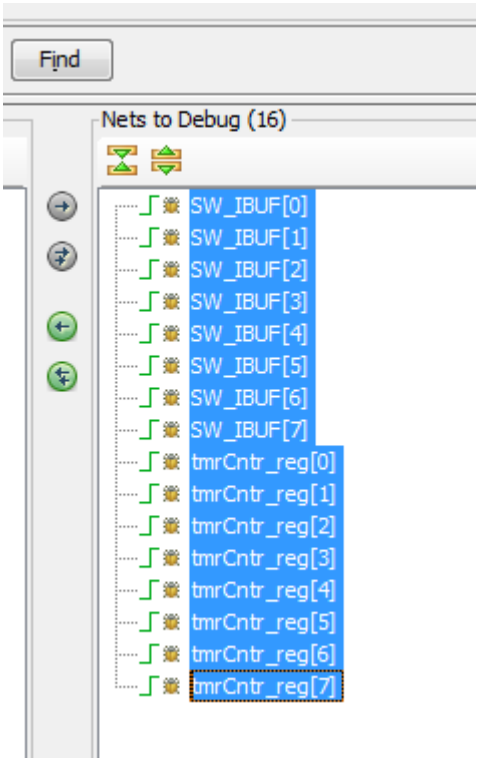
שלב א –יצירת SETUP:

6 פתח הפרויקט שנקרא **GPIO_DEMO**

7 **פתח את :**



8 **בחר את הסיגנלים הבאים :**



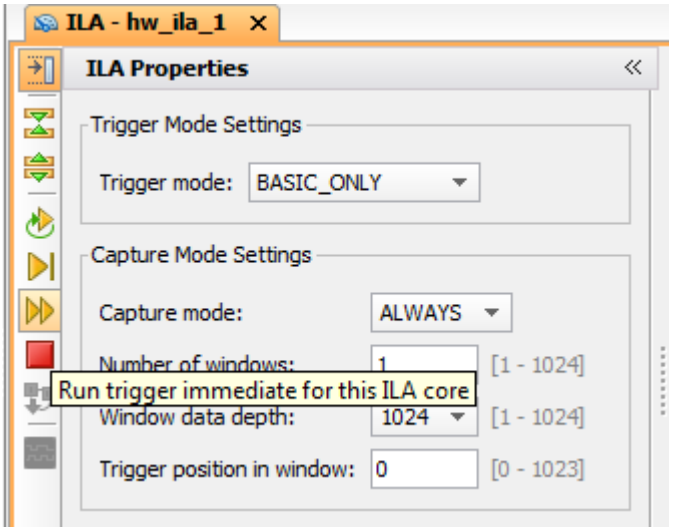
9 הרץ התכן עד לקבלת קובץ עם סיומת BIT

10 פתח תוכנת HARDWARE MANGER ודא שהקובץ נצרב לתוך הכרטיס

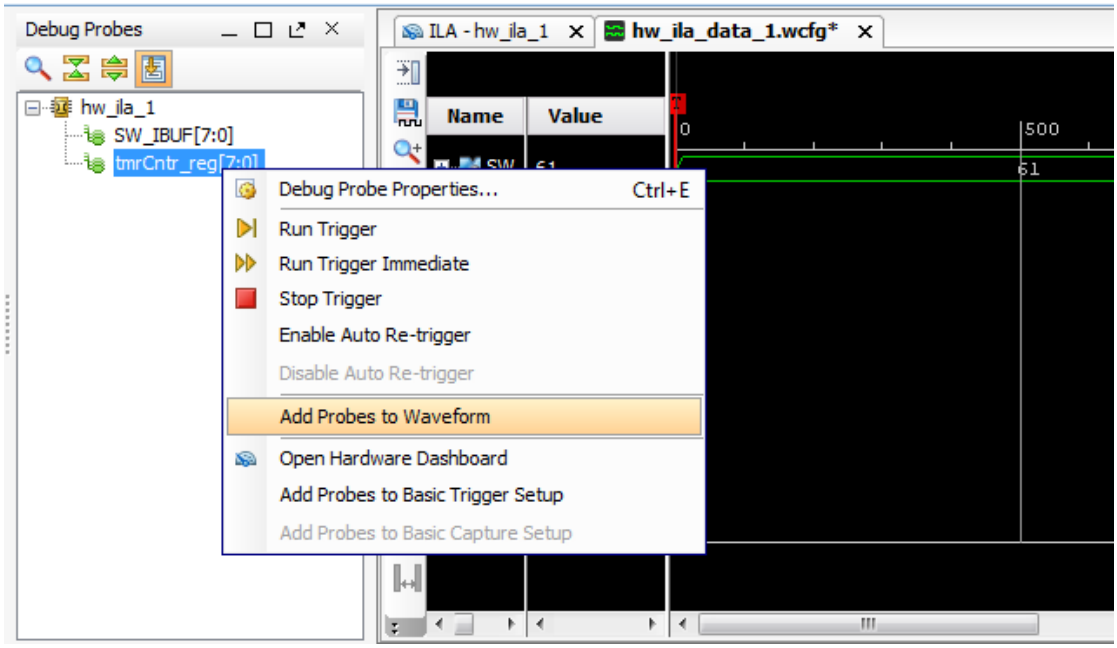
שלב ב-התנסות ב ILA במצבי העבודה השונים

15.הרצה חופשית

הריצו כמסומן לעיל(הרצה ללא תנאי)

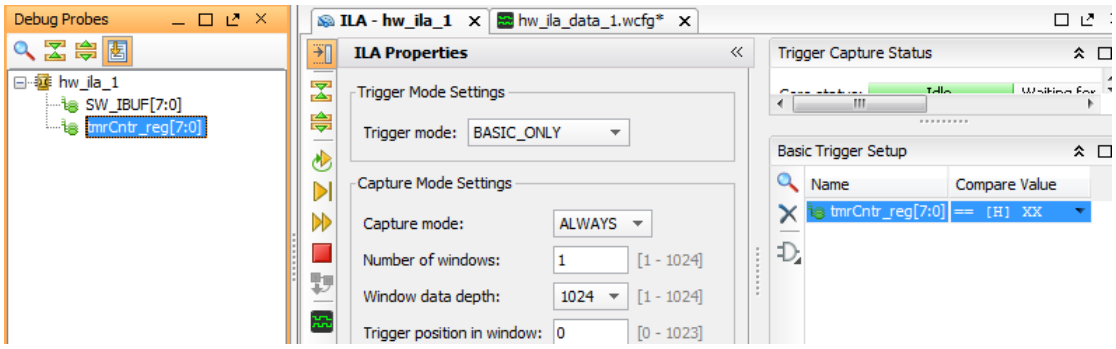


16.הוסף אותות לwave from view

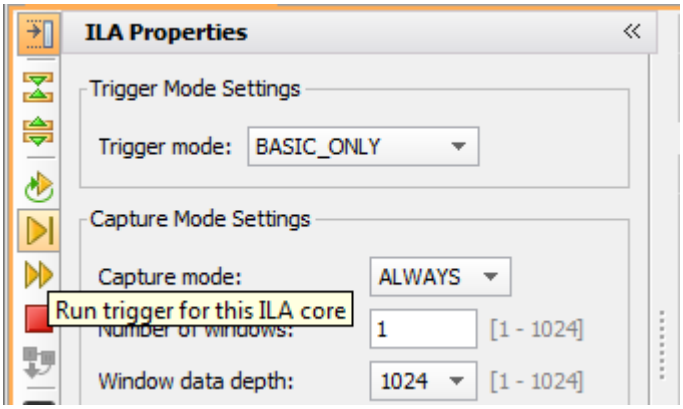


17.עבודה עם trigger mode setting במצב basic

a. גרור מחלון debug probes את האות המבוקש לחלון basic trigger setup כמתואר לעיל :



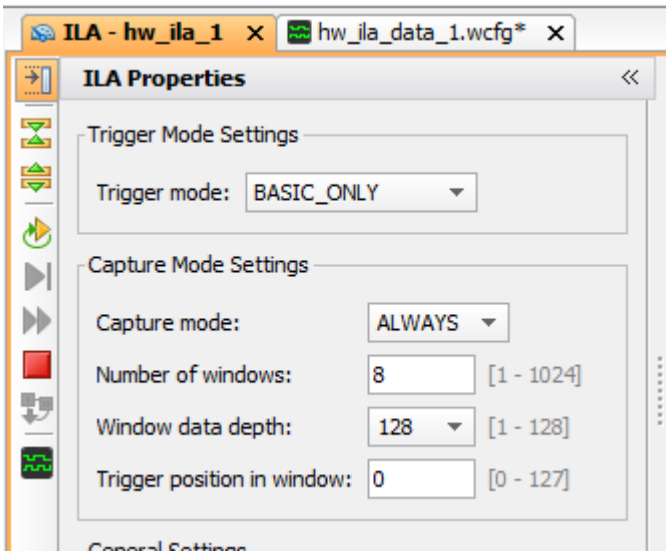
b. כעת קבע את ערך tmrCntr_reg ל96 HEX והרץ כעת כמתואר לעיל



c. פתח wave from view מה ערך המספר הראשון שמופיע ב reg_tmrCntr ?

18.עבודה ב"חלונות " עם capture mode setting במצב always

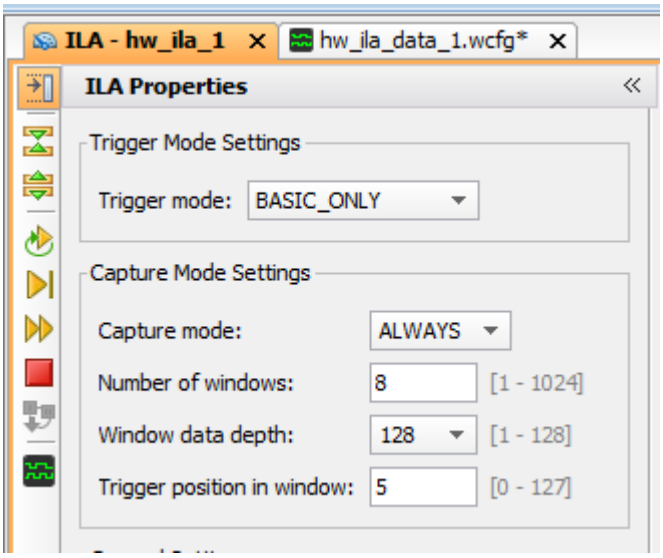
a. הגדר 8 חלונות שכל אחד מהם בעל גודל של128 דגימות והרץ



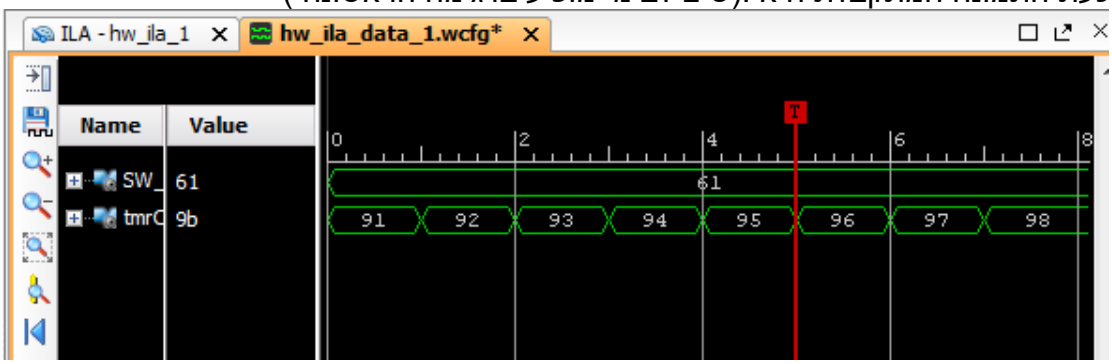
התמונה המתקבלת היא :

a) בדוק באיזה מספר מתחילה כל דגימה(HEX 96)

b) כעת שנה את מיקום הדגימה ל5 (פירוש הדבר שכעת ה96 HEXיופיע בדגימה החמישית במקום הדגימה הראשונה)



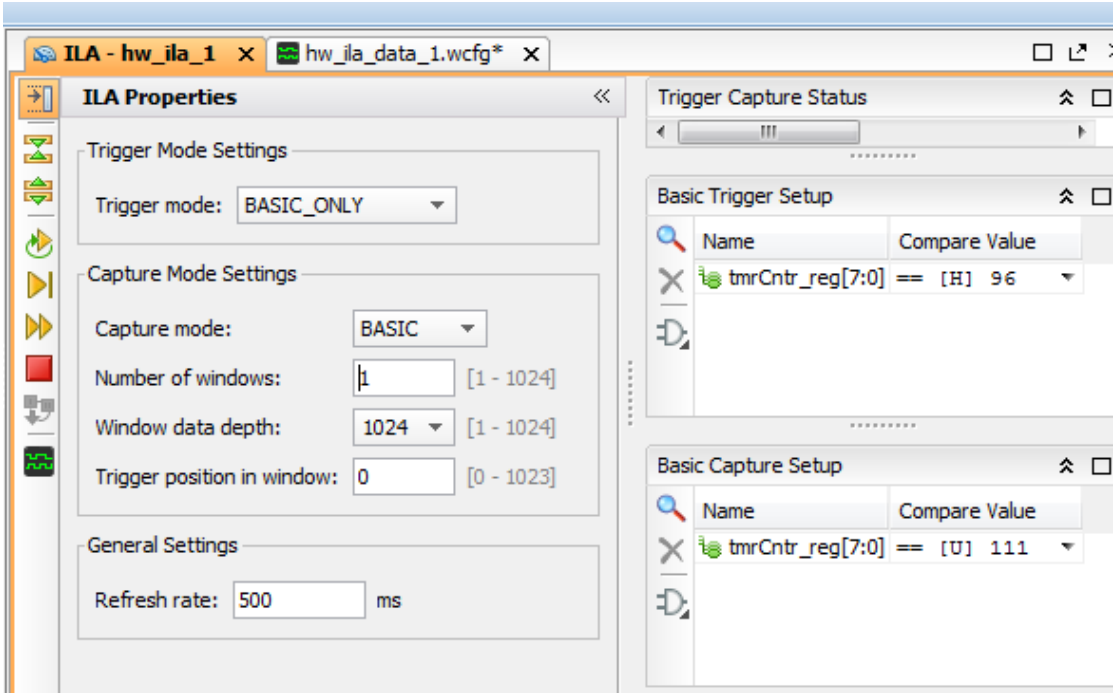
כעת התמונה המתקבלת היא :(שים לב מי מופיע בדגימה הראשונה)



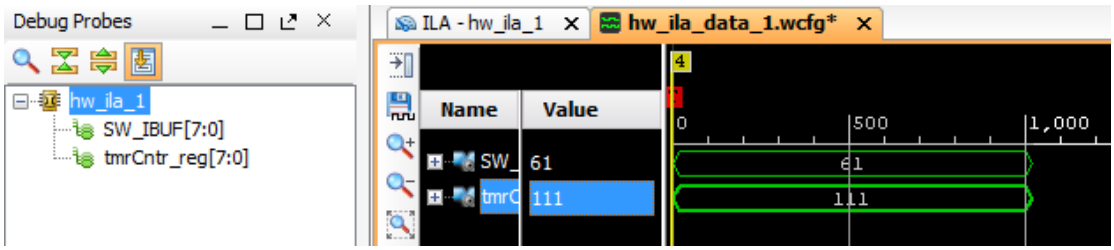
19.עבודה עם capture mode setting במצב basic

a. החזר מיקום הדגימה ל0 ושנה capture mode setting למצב basic

כעת יפתח חלון נוסף שנקרא basic capture setup



b. הרץ התכן התמונה המתקבלת היא :

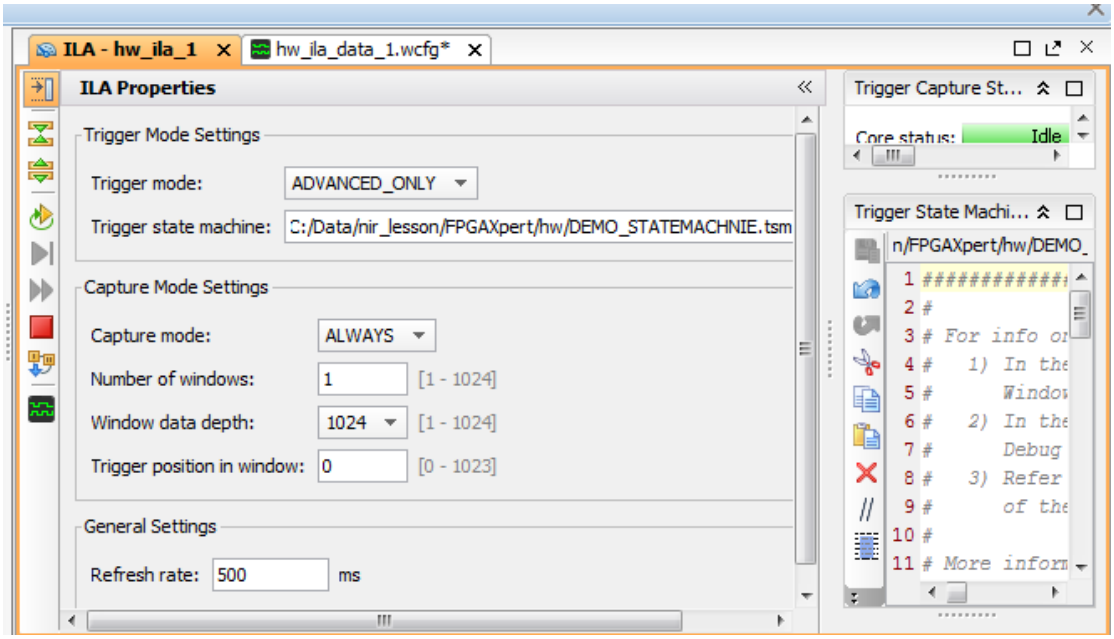


נקודה למחשבה מדוע מתקבל מספר בדיד ?

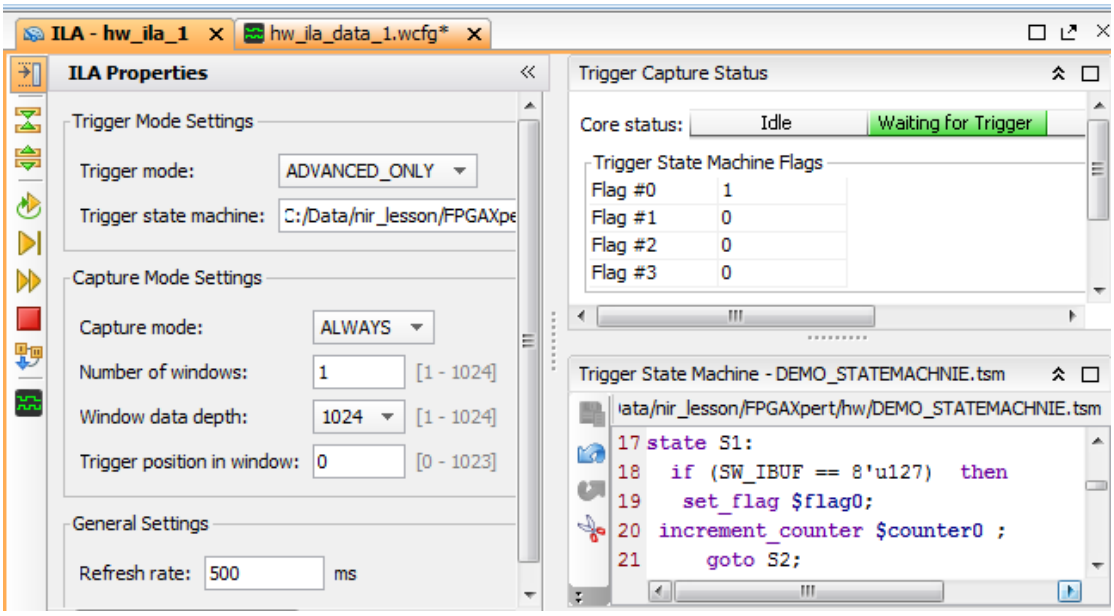
20.עבודה עם trigger mode setting במצב advanced

a. עבור בtrigger mode setting למצב advanced וסמן את הקובץ שקיבלתם TSM.STATEMACHINE_DEMO

b. חזור ל capture mode setting למצב always כמתואר לעיל :



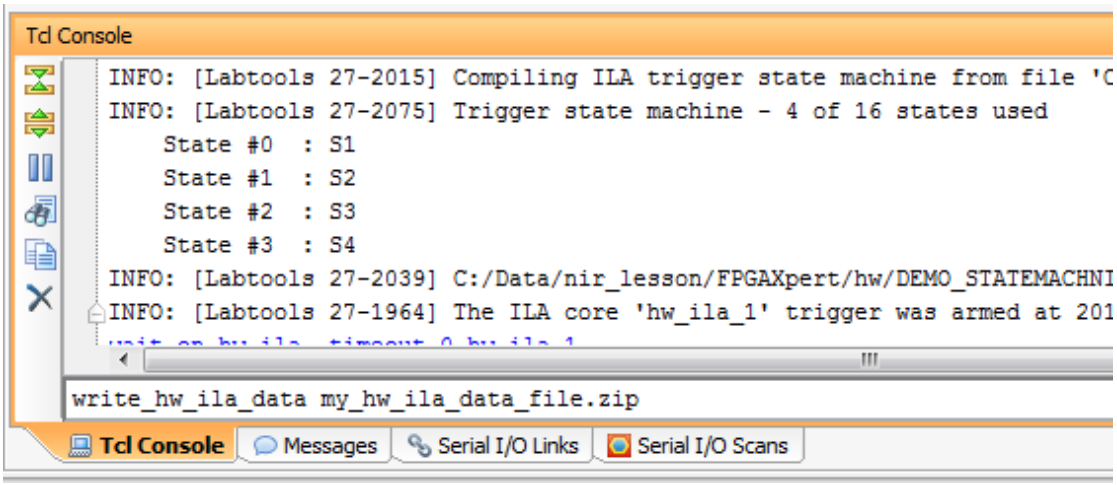
c. הרץ ושנה המתגים(SW)בהתאם לערך הכתוב בתוכנית שים לב ש FLAG משתנים בהתאם ועקוב אחרי חלון TRIGGER CAPTURE STATUS



21.שמירה של תוצאות המידע כקובץ במחשב

a. הרץ את שורת הקוד הבאה בחלון TCL

write_hw_ila_data my_hw_ila_data_file.zip



b. כעת יתקבל קובץ ששמו my_hw_ila_data_file.zip שמכיל את כל הנתונים שדגמתם
c. פתחו והסתכלו בקבצים שהתקבלו

בהצלחה
ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 12 PLL

מטרת התרגיל :

- בנית שעונים שונים ושימוש בהם

1 פתח את IP CATLOG ובחר משם את CORE יצירת השעונים
נא לבחור MMCM ולא PLL !

2.בחרו את השעונים הבאים :

Output Clock	Output Freq (MHz)		Phase (degrees)		Duty Cycle (
	Requested	Actual	Requested	Actual	Requested
clk_out1	100.000	100.000	0.000	0.000	50.000
clk_out2	300.000	300.000	0.000	0.000	50.000
clk_out3	50	50.000	0.000	0.000	50.000
clk_out4	150.000	150.000	0.000	0.000	50.000

3 צרו 4 מונים כל אחד של 10 ביט לכל אחד מהם יהיה שעון אחר (מ4 השעונים שיצרתם)

4.בצעו סימולציה של התכן הנ"ל
שימו לב לסיגנל RESET(פעיל בעליה)
וכן לסיגנל LOCK (שאחראי לציין מתי יש נעילה ותוכלו להשתמש בשעונים)

בהצלחה
ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

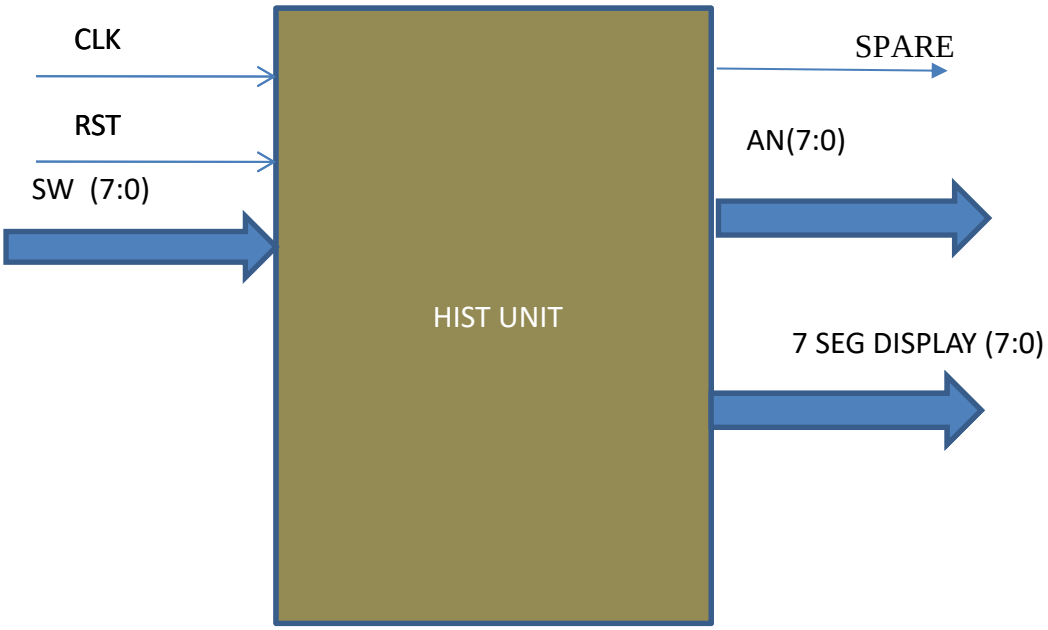
בס"ד

תרגיל 13 -מימוש היסטוגרמה על הכרטיס

לאחר שמימשתם את תרגיל בניית היסטוגרמה בקידוד וסימולציה
עליכם לממש את זאת בכרטיס
עליכם לממש את היכולות הבאות :

1. **הצגת תכולה נבחרת של תא מתוך ההיסוטגרמה** למשל אם נבחר תא מספר 5 והתכולה שלו היא 16 על ארבע הספרות הראשונות של תצוגת 7 סיביות בכרטיס יופיעו הספרות 0016
2. **מציאת ערך חציוני** קרי האיבר במקום ה512 וזה יופיע על ארבע הספרות האחרונות של תצוגת 7 סיביות למשל אם ערך האיבר החציוני יצא 400 על ארבע הספרות האחרונות של תצוגת 7 סיביות בכרטיס יופיעו הספרות 0400
3. **יש להראות את התוצאות גם בעזרת ILA(CHIP SCOPE) בדיוק כפי שראיתם בסימולציה**

דיאגראמת הבלוקים של המכשיר נראית כך :



אותות הכניסה :(את מיקום הפינים יש לקחת מXDC הקודם)

4. **CLK** -שעון המערכת בתדר של 100 מגה

5. **RST**-RESET של המערכת
6. **(7:0) SW-בחירת התכולה של התא באיסוף הסיטוגרמה להצגה**

<u>יציאות התכנון :</u>

(7:0) 7 SEG DISPLAY -תצוגת שבע סיביות

(7:0) AN -בחירה איזו ערך יוצג באיזה סיפרה

1. **(3:0) AN– הספרות המשמשות לצורך תצוגת התא**
2. **(7:4) AN– הספרות המשמשות לצורך תצוגת החציון**
3. **SPARE(למה מיועד?)**

בברכה

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 14 יצירת שעונים והפעלתם

מטרת התרגיל :

- תפעול logic analyzer
- בנית שעונים שונים ושימוש בהם

1 נא צרו ENTITY שבה :

הכניסות הבאות :

a. (E3)CLK

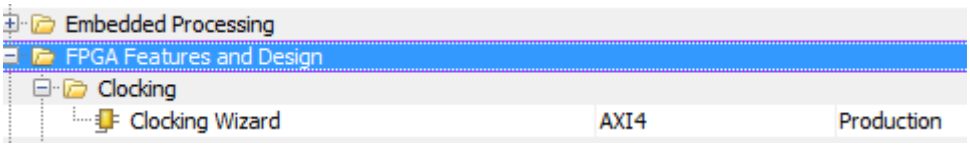
b. (J15)SW

היציאות הבאות

(H17-N14)LED(4:0)

2פתח את IP CATLOG ובחר משם את CORE יצירת השעונים

נא לבחור MMCM ולא PLL !



2.בחרו את השעונים הבאים :

Output Clock	Output Freq (MHz)		Phase (degrees)		Duty Cycle (
	Requested	Actual	Requested	Actual	Requested
clk_out1	100.000	100.000	0.000	0.000	50.000
clk_out2	300.000	300.000	0.000	0.000	50.000
clk_out3	50	50.000	0.000	0.000	50.000
clk_out4	150.000	150.000	0.000	0.000	50.000

3הוסיפו לתכן

נא צרו ארבעה מונים של 9 סיביות (כל מונה ירוץ עם השעון שיצרתם)

4 בנוסף לזה נא כתבו PROCESS סנכרוני (מותנה שעון) עבור כל מונה שיצרתם בהתאמה שבו אתם כותבים תנאי שנראה כך :

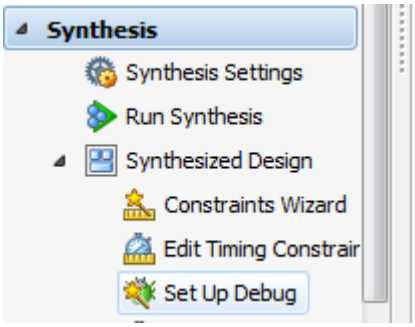
```
process(clk1)
…….
 if count1=203 then
led (0)<='1';
else
led(0)<='0';
end if ;
……..
end process;
```

כמו כן נא חברו את SW(0) כמתג ה RESET של כניסת CORE של יצירת השעונים

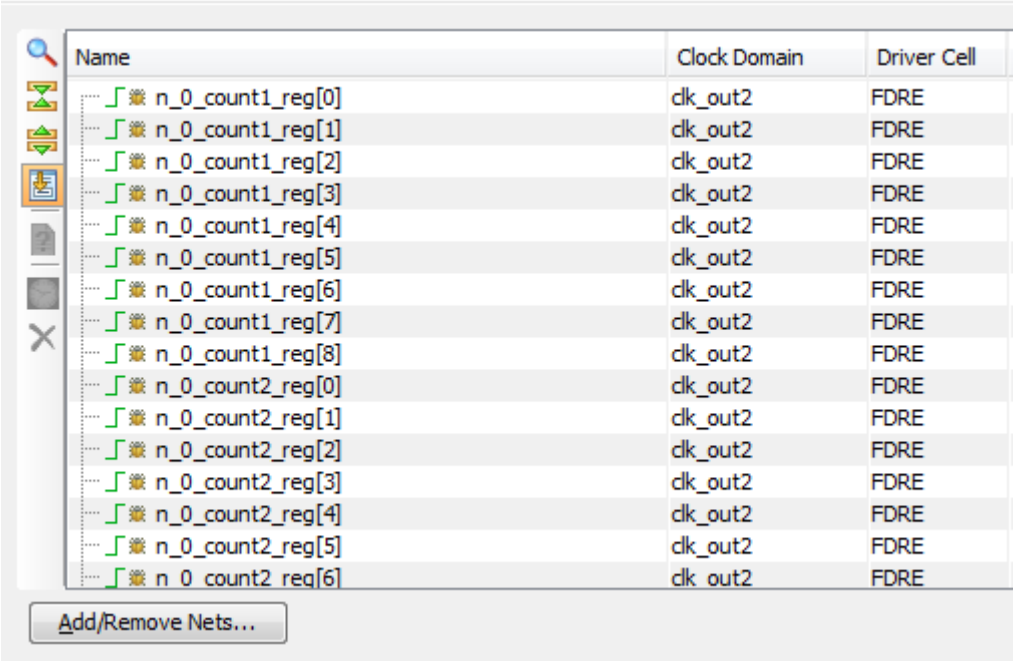
5.בצעו סימולציה של התכן הנ"ל
שימו לב לסיגנל RESET(פעיל בעליה)
וכן לסיגנל LOCK (שאחראי לציין מתי יש נעילה ותוכלו להשתמש בשעונים)

6. הוספת LOGIC ANALYZER

נא פתחו את SET UP DEBUG



נא הוסיפו את כל ארבעת המונים לאותות שאתם מעוניינים לראות בlogic analyzer
נא שנו את CLOCK DOMAIN של **כל המונים** כך שיתאימו לתדר של השעון הגבוה בצורה הבאה :



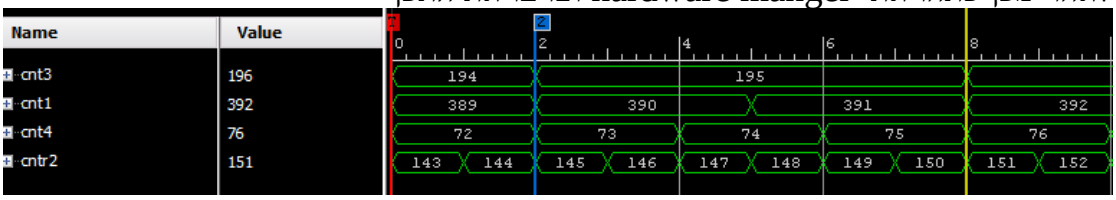
Name	Clock Domain	Driver Cell
n_0_count1_reg[0]	clk_out2	FDRE
n_0_count1_reg[1]	clk_out2	FDRE
n_0_count1_reg[2]	clk_out2	FDRE
n_0_count1_reg[3]	clk_out2	FDRE
n_0_count1_reg[4]	clk_out2	FDRE
n_0_count1_reg[5]	clk_out2	FDRE
n_0_count1_reg[6]	clk_out2	FDRE
n_0_count1_reg[7]	clk_out2	FDRE
n_0_count1_reg[8]	clk_out2	FDRE
n_0_count2_reg[0]	clk_out2	FDRE
n_0_count2_reg[1]	clk_out2	FDRE
n_0_count2_reg[2]	clk_out2	FDRE
n_0_count2_reg[3]	clk_out2	FDRE
n_0_count2_reg[4]	clk_out2	FDRE
n_0_count2_reg[5]	clk_out2	FDRE
n_0_count2_reg[6]	clk_out2	FDRE

Add/Remove Nets...

7 בחינת התוצאות

נא הריצו התכן עד לקבלת קובץ ביט

לאחר מכן פתחו את hardware manger וצרבו את התכן



שימו לב ליחס המניה בן השעונים השונים האם תואם למה שתכננתם ?

בהצלחה

ניר

בס"ד

תרגיל 15 מחשבון

שלב א

יצירת מודול

אותות הכניסה :

4. **CLK** -שעון המערכת בתדר של 100 מגה
5. **RESET**-RST של המערכת
6. **DATA_IN_A**-מידע ברוחב 16 סיביות
7. **DATA_IN_B**-מידע ברוחב 16 סיביות
8. **SEL**-מידע ברוחב 3 סיביות

אות מוצא:

DATA_OUT-התוצאה מידע ברוחב 20 סיביות

שלב ב

בניית מחשבון

עליכם לבנות מחשבון שיבצע את הפעולות הבאות:

1. חיבור
2. חיסור
3. כפל
4. חילוק(אלגוריתם "אריה במדבר")
5. שורש(אלגוריתם "אריה במדבר")
6. חישוב ממוצע (של 100 מספרים)
7. חישוב סטיית תקן של 64 מספרים
8. פתרון משוואת השורשים(כנ"ל)

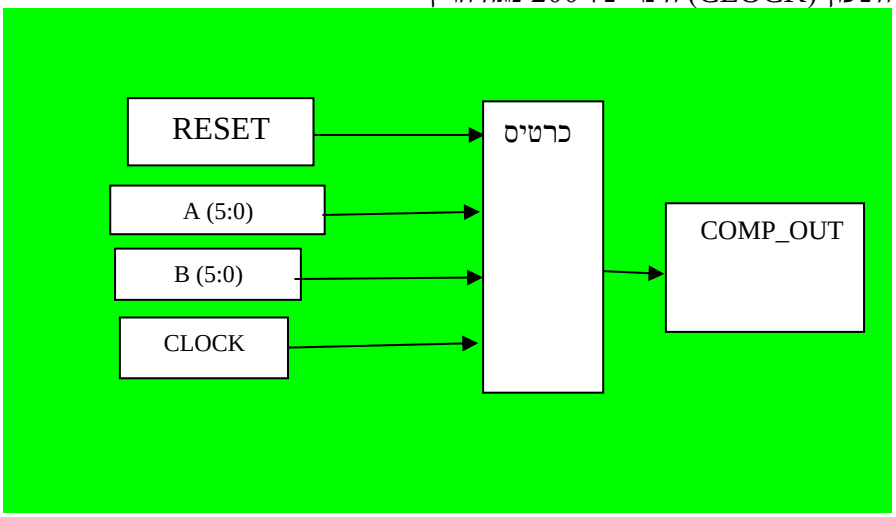
בהצלחה ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 16 שבירת לוגיקה

שלב א
עליכם לתכנן משווה שישווה בין שני מספרים של 6 ביט
השעון (CLOCK) הינו של 200 מגה הרץ



כניסות התכנון :

8. CLOCK
9. RESET
10. A(5:0)-המספר הראשון
11. B(5:0)- המספר השני

מוצא התכנון :

3. COMP_OUT-מוצא המשווה באם התוצאה שווה מוציא 1 לוגי באם לא 0 לוגי

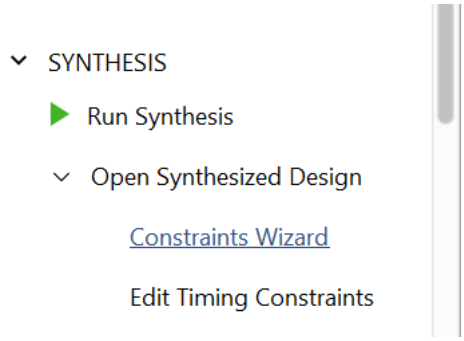
שאלה למחשבה/דגש :
יש לדגום את אותות הכניסה בעזרת שעון A וB בהתאמה בעזרת שעון
למה ??

שלב ב
סמלצו התכן ובדקו שהעבודה נעשית

שלב ג
סנתזו את התכן
הגדירו פיני כניסה ויציאה לפי הכרטיס שברשותכם (שעון E3 המספר הראשון SW 6 הראשונים
המספר השני SW 6 בהתאמה והמוצא נא לחבר לled0)

שלב ד

כעת הגדירו את השעון כ200 מגה בעזרת CONSTRAINTS WIZARD

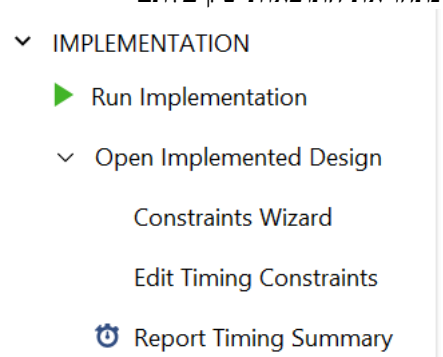


שלב ה

הריצו אמלפנטציה

כעת בעזרת report timing summary

נתחו את התוצאות שקיבלתם



- מה במסלול הארוך ביותר ??
- הסתכלו על הסכמה של המסלול הארוך ביותר ??
- כמה רמות לוגיות יש בן FF לFF??
- מה ניתן לעשות ??(שבירת לוגיקה)??

שלב ו

פתחו פרוייקט נוסף קחו את קבצי XDC VHD שיצרתם בשלב הקודם

בצעו שבירת לוגיקה כפי שהדגמתי בכיתה חזרו על שלבים ג ד וה בהתאמה

מה המסלול הארוך ביותר ??

כמה רמות לוגיות ישנה בן FFלFF ??

שלב ז

פתחו פרוייקט נוסף

כעת שנו את רוחב אותות הכניסה ל60 ביט לA וB בהתאמה

נא חזרו על שלבים ב עד ה בהתאמה

בהצלחה

ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

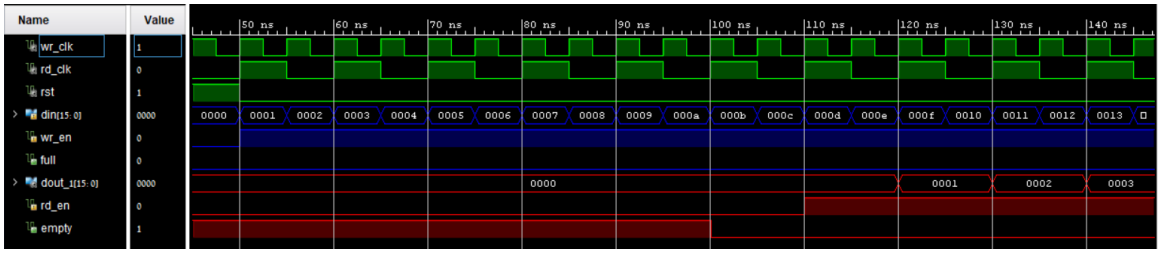
תרגיל 17 FIFO

צור FIFO. (FIF0 מבוסס על-BRAM) ומשני שעונים בלתי תלויים בתצורה הבאה :

1. צור מונה 16 ביט ודוחף את הערך לFIFO (הערך נשלף מFIFOהחוצה מהצד של הקריאה)
2. צור סימולציה בתצורה הבאה:

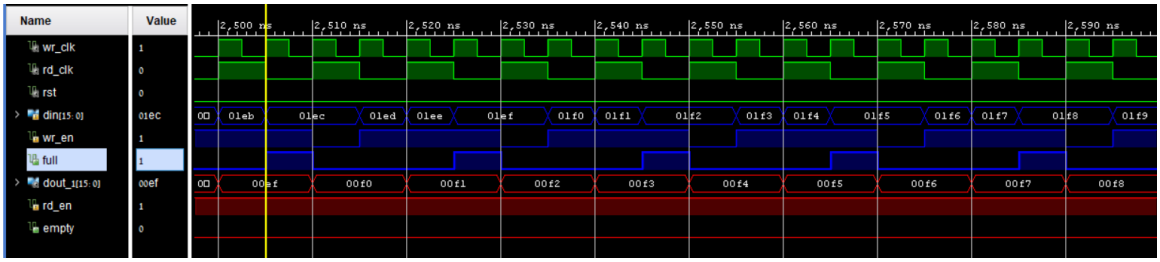
שעון כתיבה: 200MHz

שעון קריאה: 100MHz



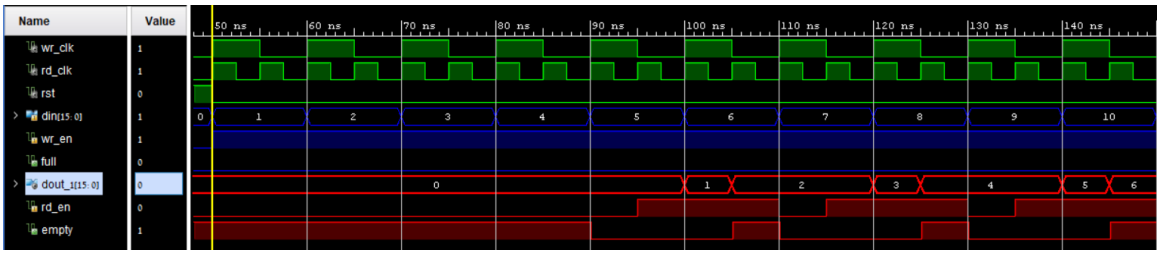
תחילת העבודה לאחר יציאה מ-RST.

ניתן לראות כתיבה בקצב של 200MHz ומחזור שעון קריאה אחד לאחר ירידת ה-empty מתחילות קריאות בקצב של 100MHz.



כאשר הFIFO מלא נא הורידו את אות ה-wr_en (והמונה יעצר) בירידת ה-FULL החזירו את מניית המונה והכתיבה לFIFO
כמובן בזמן זה תמשיך פעולת הקריאה ברצף ללא הפרעה למה ??

3. צור סימולציה בתצורה הבאה:
שעון כתיבה: 100MHz
שעון קריאה: 200MHz



שימו לב במקרה זה, מלכתחילה, אות ה-empty ירד ויעלה לסירוגין למה ?

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 18 IP INTEGRATOR & I/O PLANNING

מטרת התרגיל :
הכרות עם התוכנות שלמדנו עליהם היום :

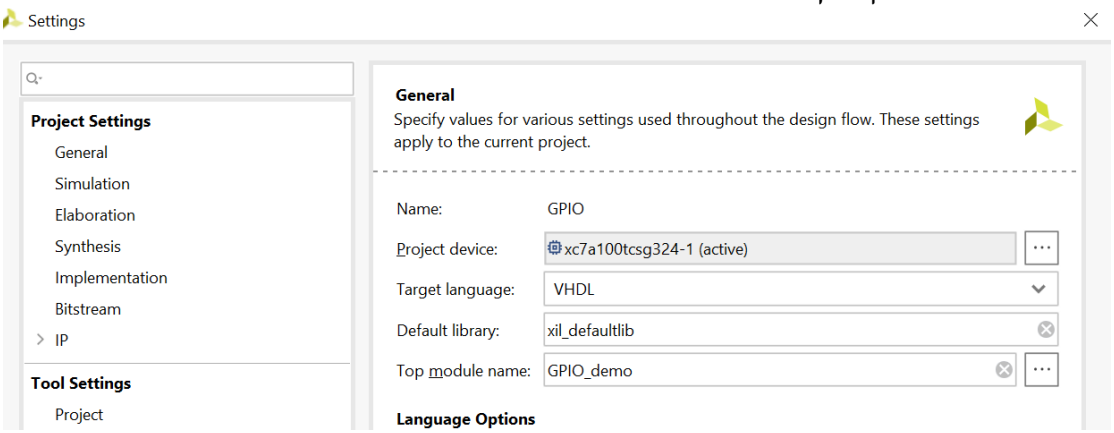
1. ***ip integrator***
2. ***i/o planning***

1 **NEXYS 4 DDR GPIO Demo- היכנס לקישור הנ"ל**
https://reference.digilentinc.com/learn/programmable-logic/tutorials/nexys-4-ddr-gpio-demo/start
בקישור בצעו הורדה לקובץ ZIP שמצוין לעיל

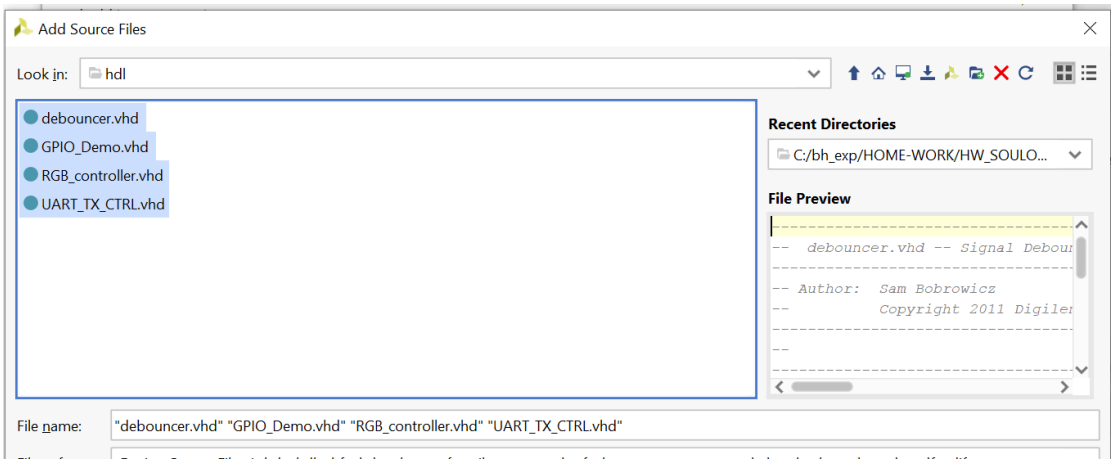
Downloads

Nexys 4 DDR GPIO Demo Project Repository – ZIP GIT Repo

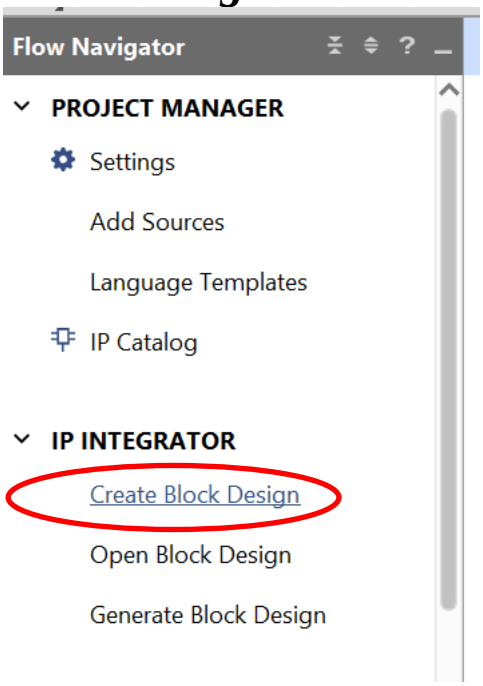
פתחו פרויקט חדש שיקרא **GPIO**
הגדירו את הפרויקט כך :



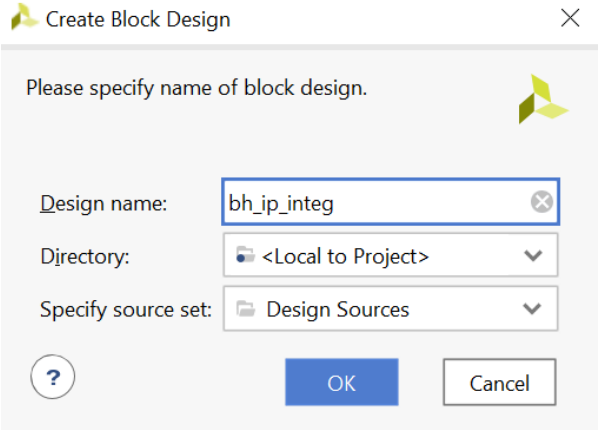
תחת תיקייה שנקראת hdl שנמצאת תחת src הוסף לפרויקט את הקבצים הבאים



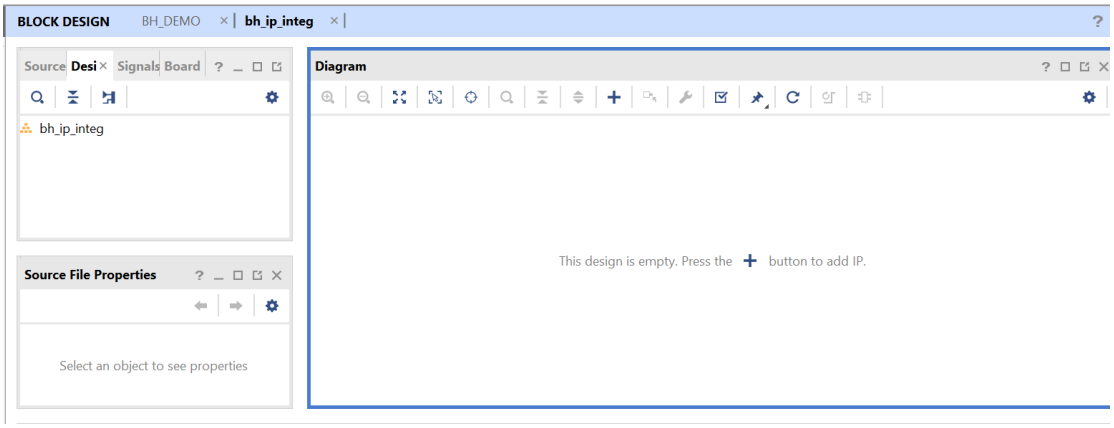
כעת לחצו על ***create block design***



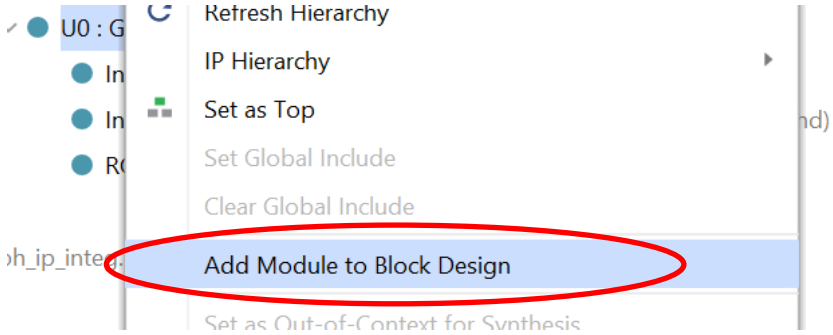
הגדירו את שם ה block design



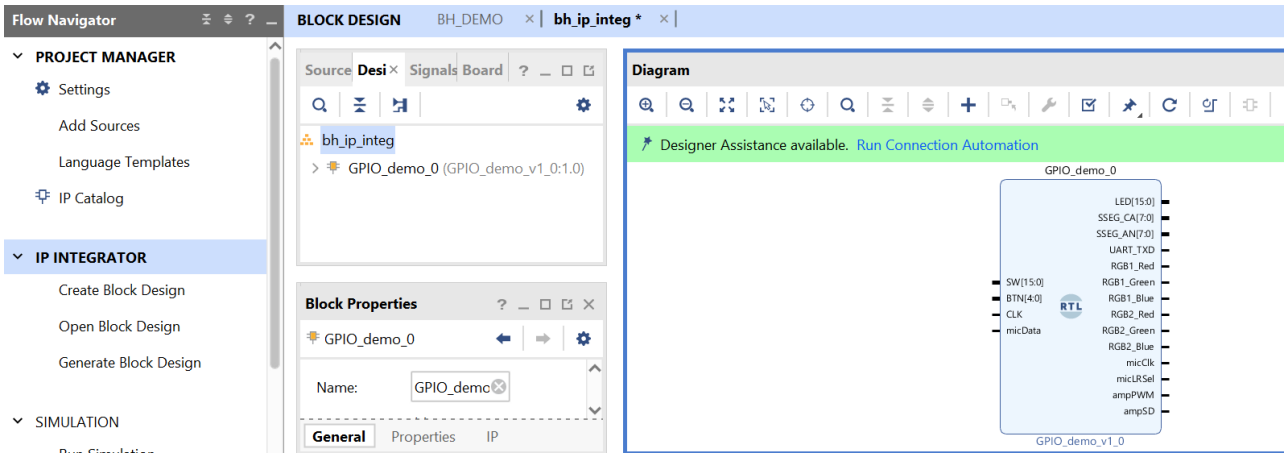
כעת יפתח לכם החלון הבא :



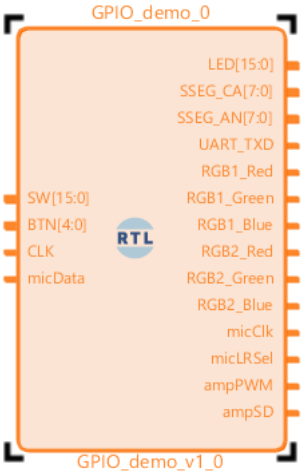
עמדו על קובץ GPIO DEMO וסמנו אותו בצורה הבאה :



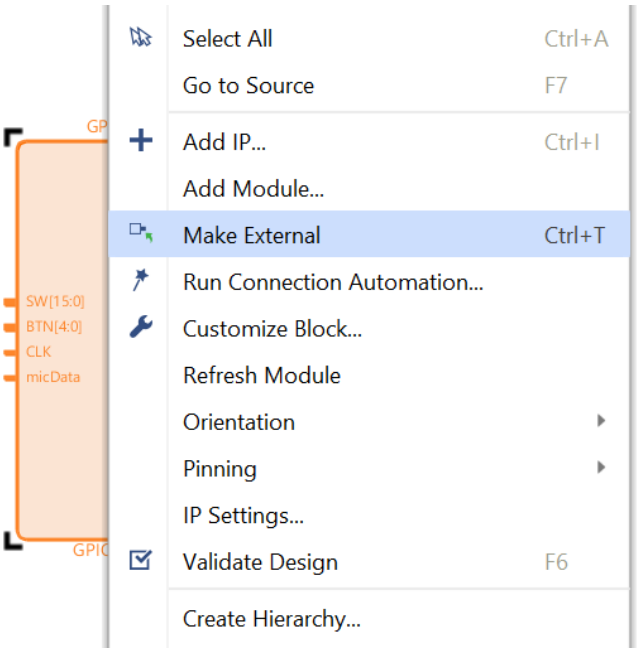
התוצאה שתתקבל תופיע על הdiagram בצורה הבאה :



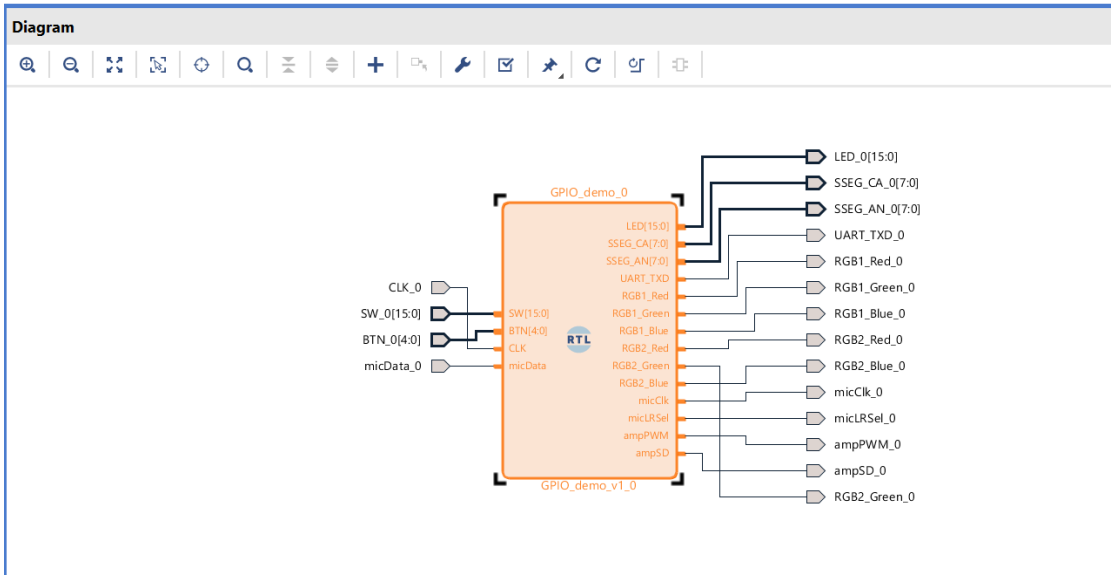
סמנו את התכנון בצורה הבאה :



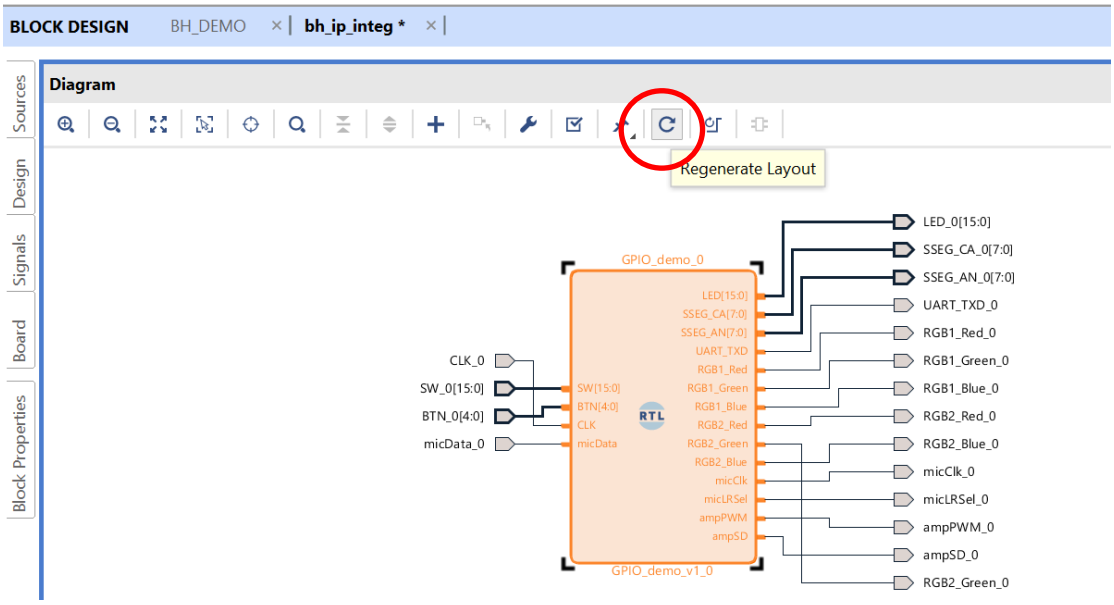
כעת הוסיפו כניסות ויציאות לתכנון בתצורה הבאה :



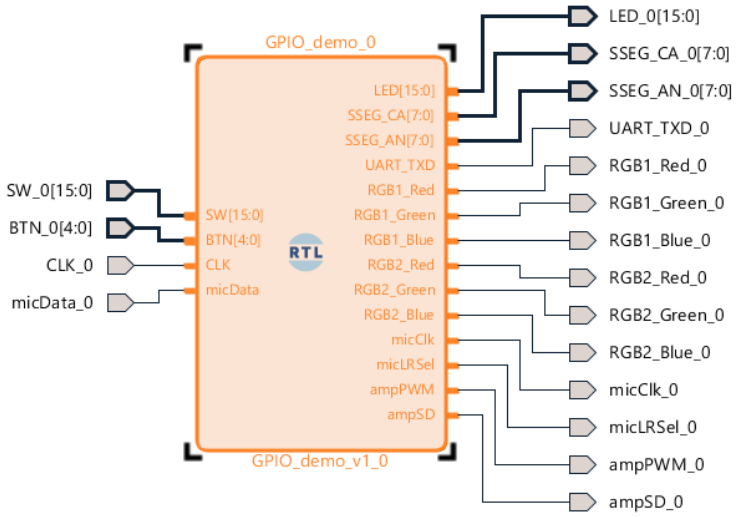
התוצאה שתתקבל תראה כך :
נקודה למחשבה :מדוע חובה להגדיר את הפינים החיצוניים ??



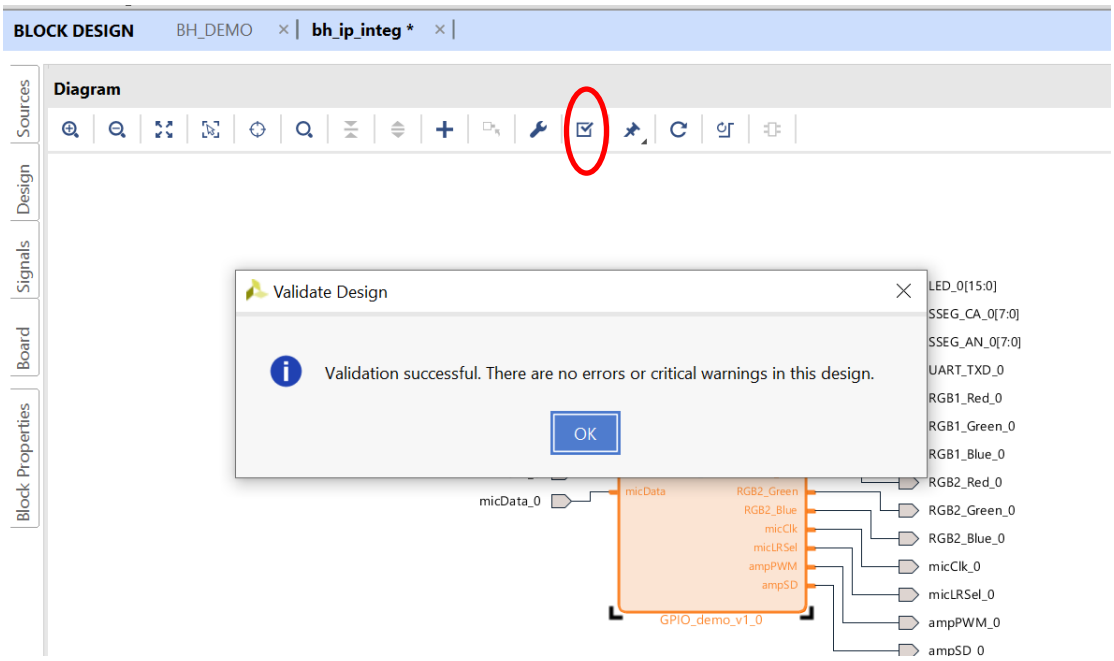
לחצו כעת על הכפתור הבא :



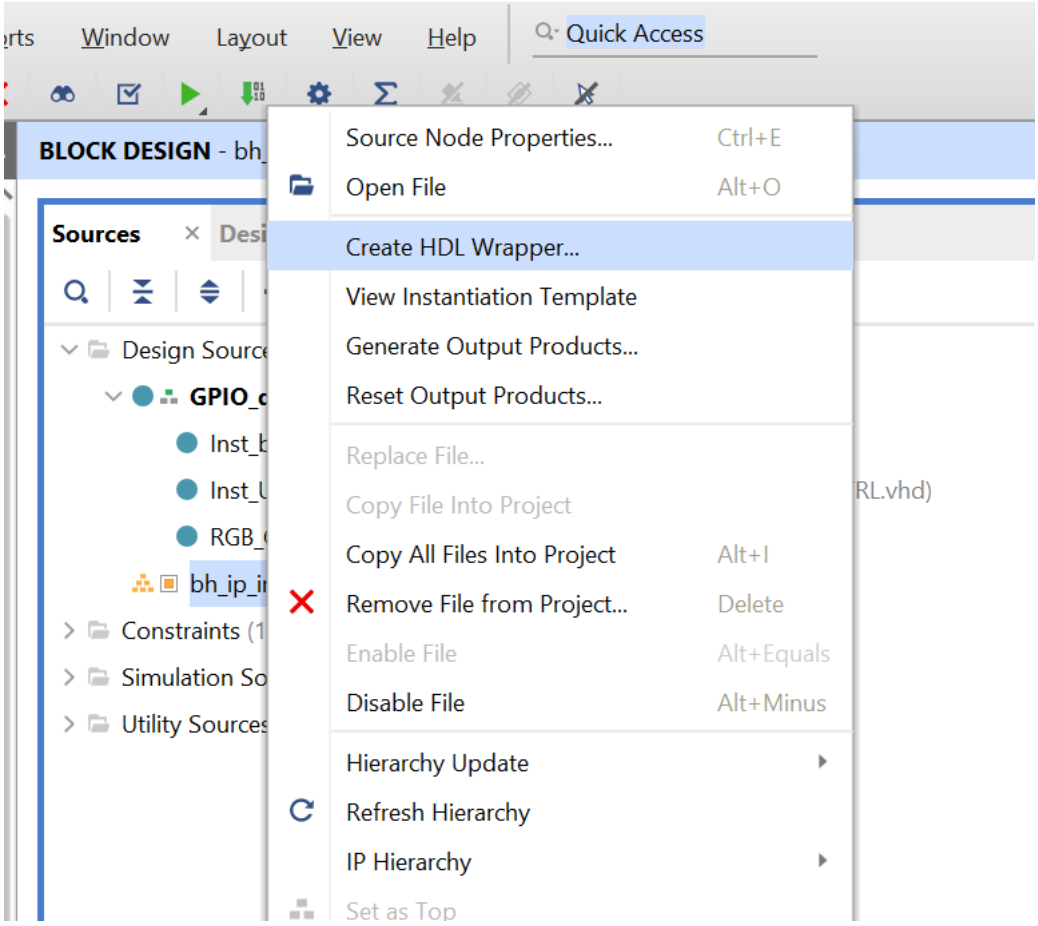
שימו לב לתוצאה המתקבלת, מה מהות הפעולה?



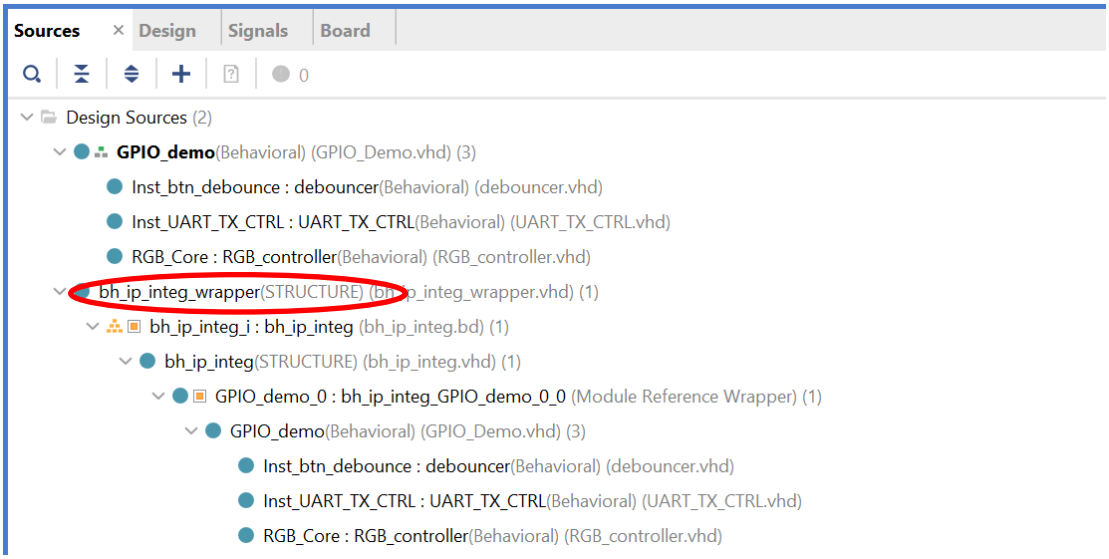
כעת לחצו על כפתור ה validate design בכדי לוודא שהתכן תקין :



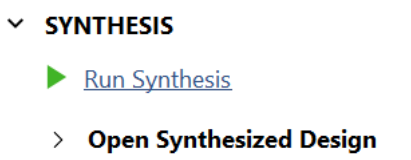
כעת עמדו על תכן block diagram ובצע המרה לקוד HDL בצורה הבאה :



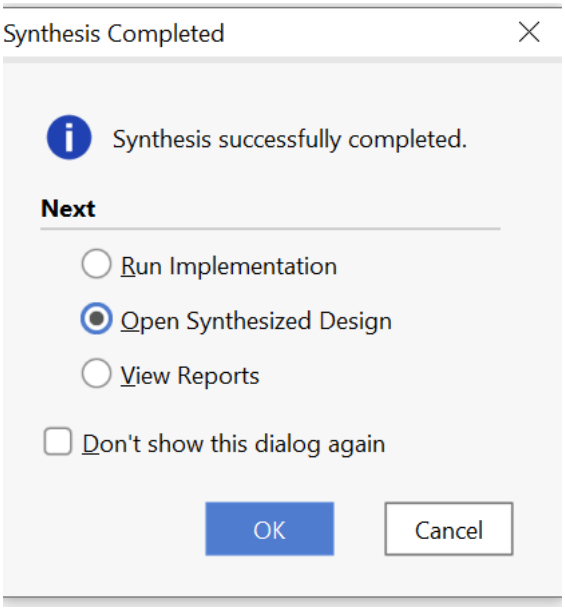
שימו לב ל"שכבה " שנוספה מעל block diagram:



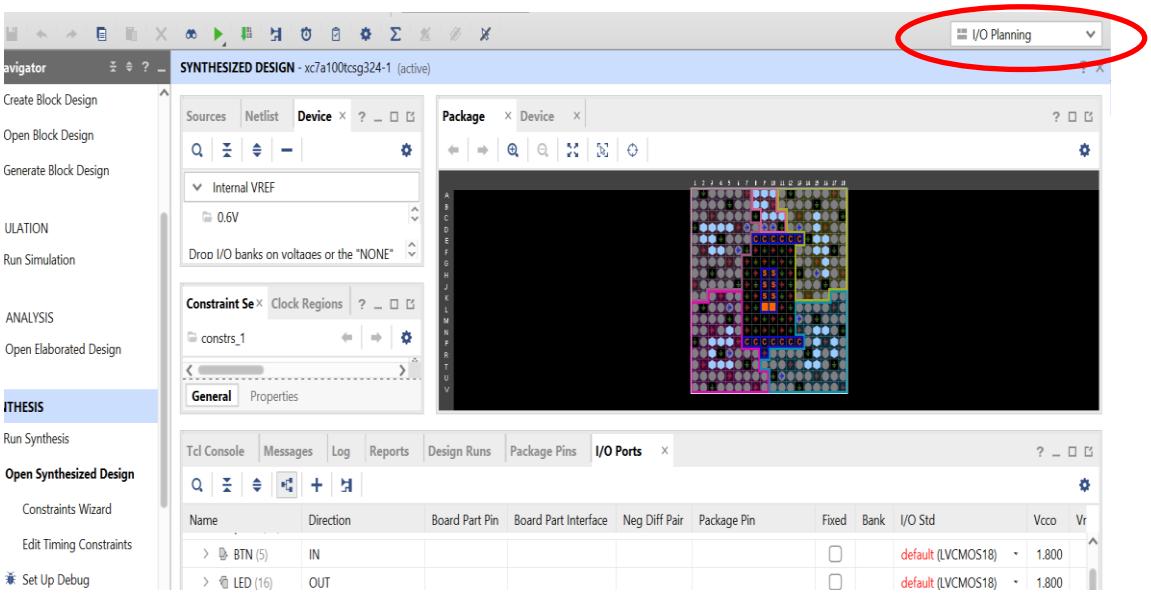
הריצו סינתזה :



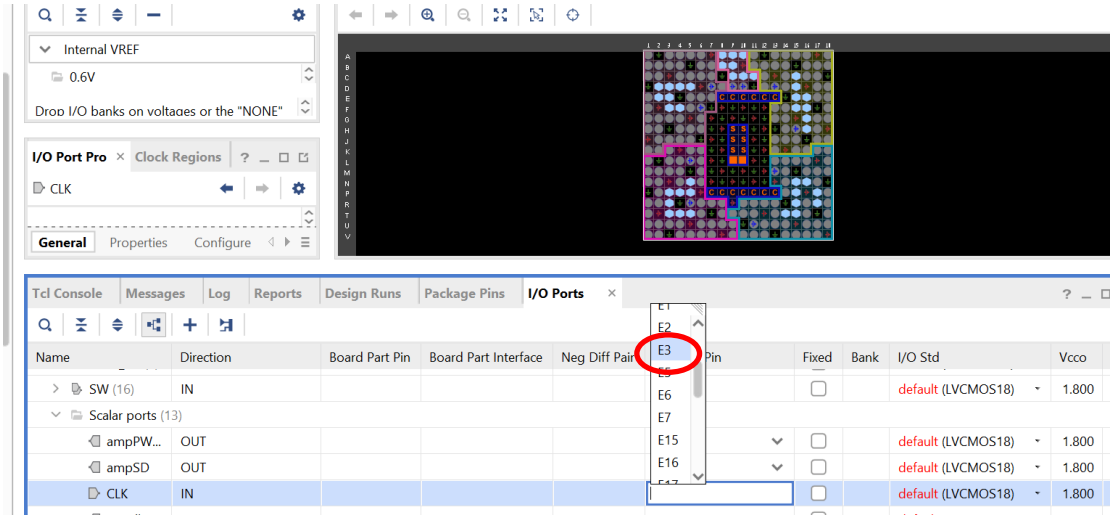
עם סיומה פתחו את תכן הסינתזה :



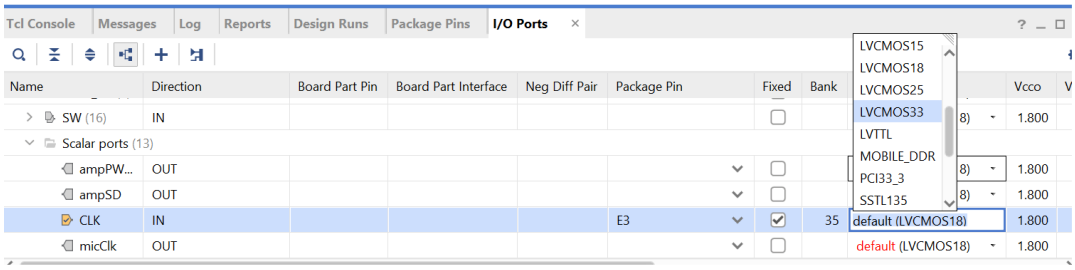
כעת התצוגה תראה כך :



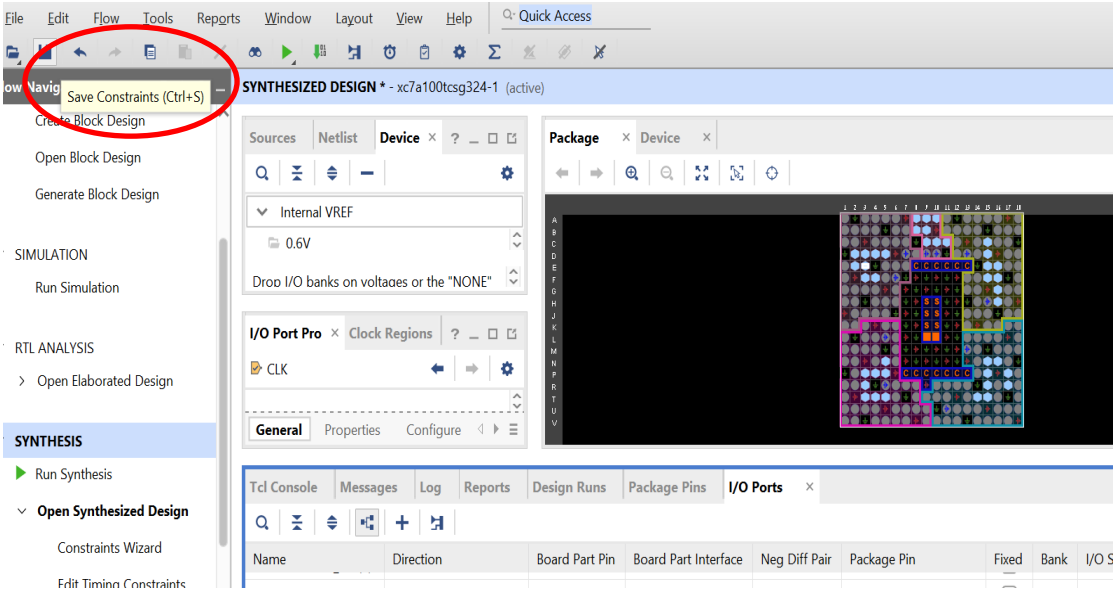
פתחו כעת את התוכנה שנמצאת בחלק התחתון של המסך I/O PORTS :



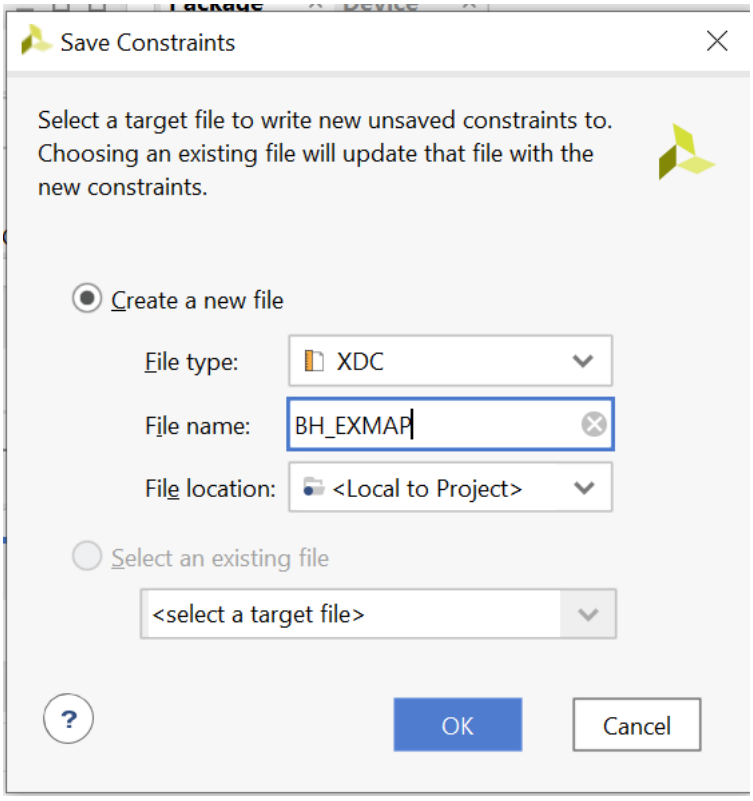
נא סמנו את רגל CLK ל E3 ובהתאמה שנו את פורמט המתח ל LVCMOS33



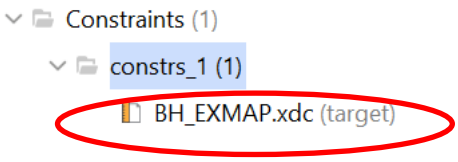
שמרו את התוצאה בצורה הבאה :



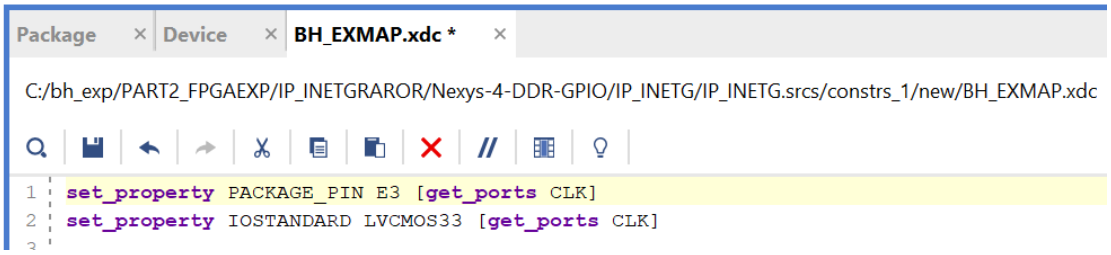
יפתח חלון הבא נא תנו את השם הבא :



כעת תחת ה constrains יופיע הקובץ :

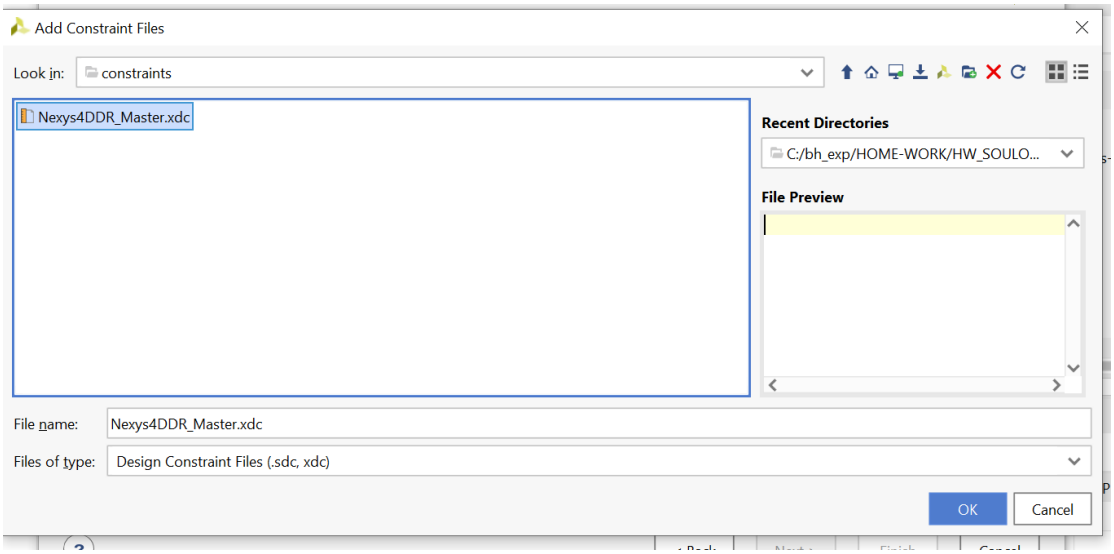


פתחו אותו מה שיופיע זה למעשה זה שהגדרתם :

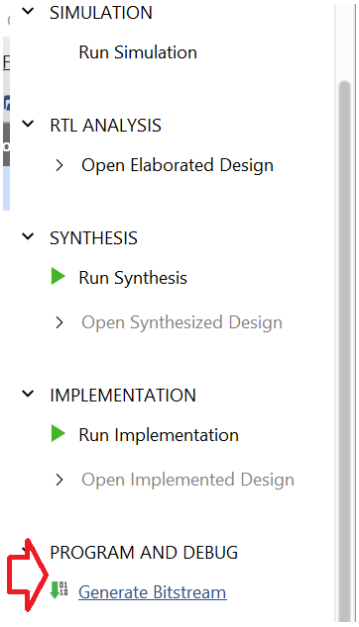


חזרו על התהליך עם כל הכניסות והיציאות וכמובן שמרו את הקובץ

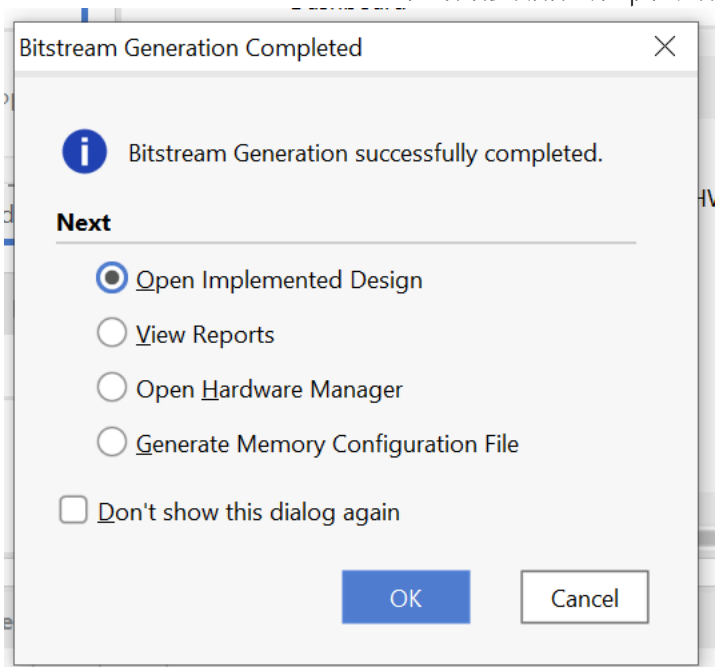
הערה רשימת המיקומים לכניסות ויציאות נמצא תחת תחת תיקייה שנקראת constraints שנמצאת תחת src תוכלו לקחת משם את כל המיקומים של הכניסות והיציאות וכמובן בהתאמה שנו את פורמט המתח ל LVCMOS33 :



הרץ התכן עד לקבלת קובץ עם סיומת BITכמתואר לעיל



חכו לקבלת ההודעה הנ"ל :



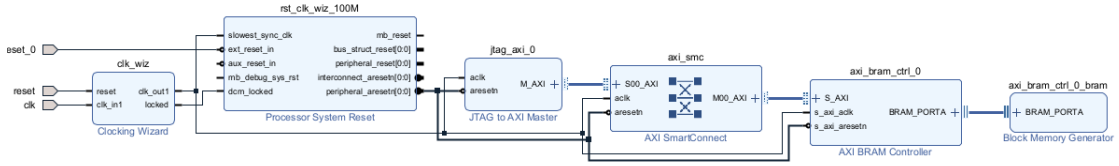
בהצלחה
ניר

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

תרגיל 19 הכרות ותפעול עם ממשק AXI בשילוב CHIPESCOPE ו IPINETGARTOR

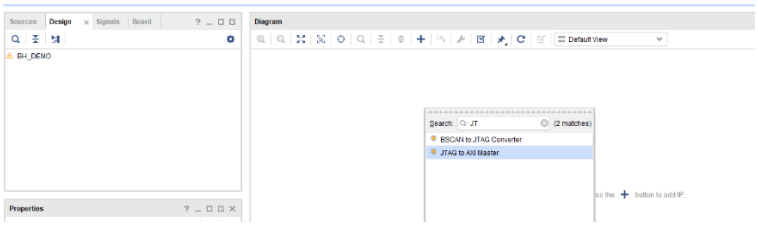
1. בנה בעזרת ה-IP INETGARTOR את התכנון הבא



כעת נפרוט את זה לשלבים:

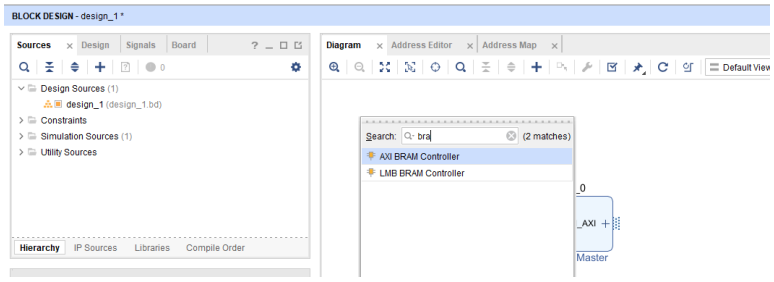
שלב א

בחר את JTAG TO AXI MASTER

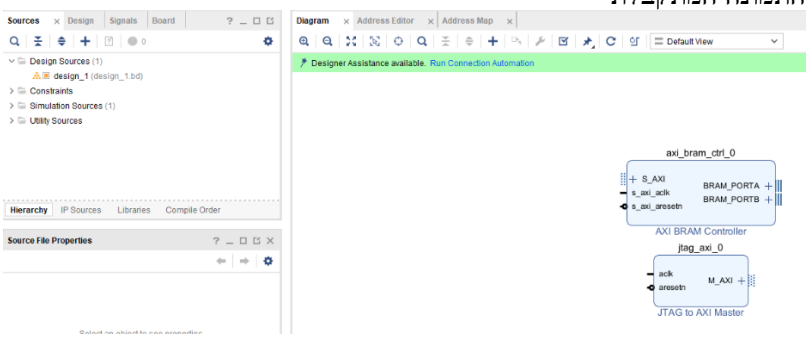


שלב ב

בחר את AXI BRAM CONTROLLER



התמונה המתקבלת

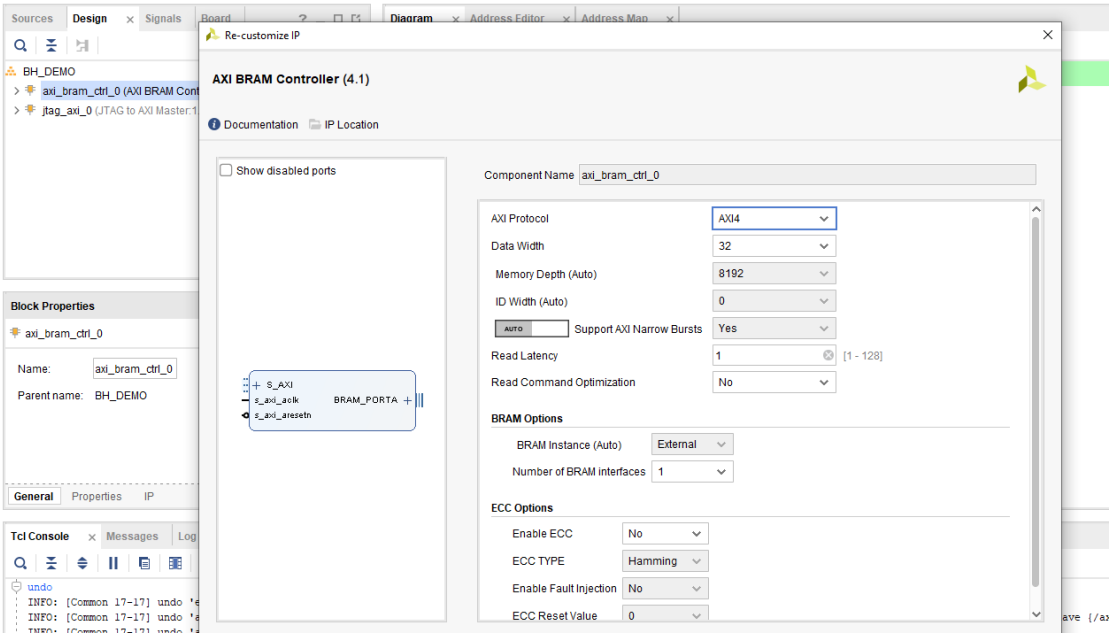


שלב ג

הקליקו פעמיים על axi bram controller

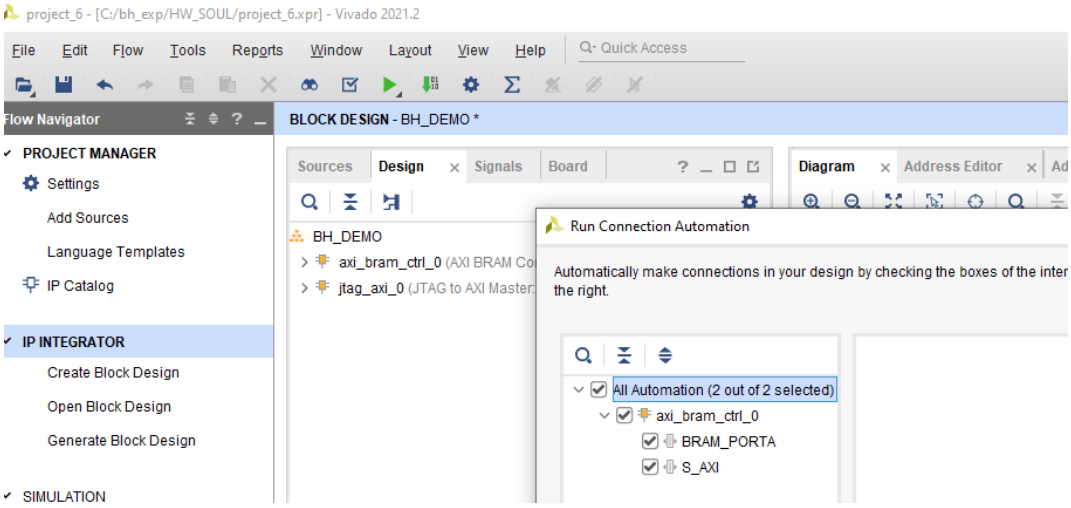
יפתח לפניכם החלון הבא :

נא לשנות ל1 בלשונית number of brams interfaces

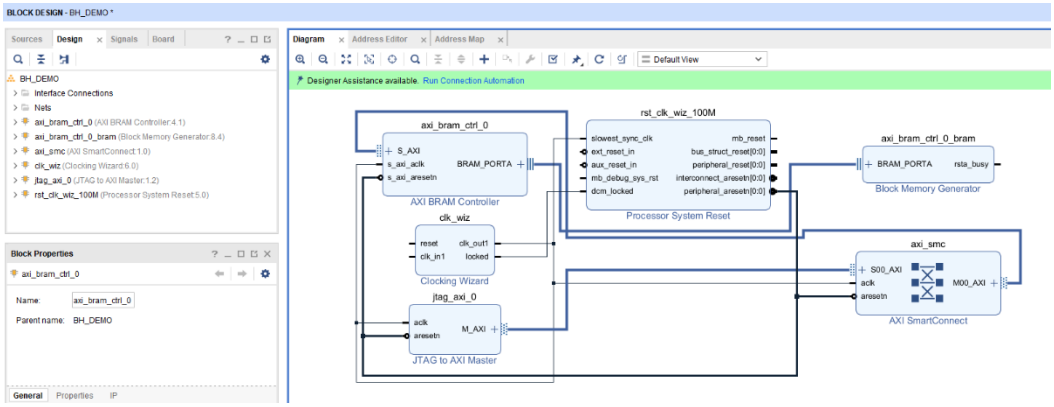


כעת נבצע run connection automation

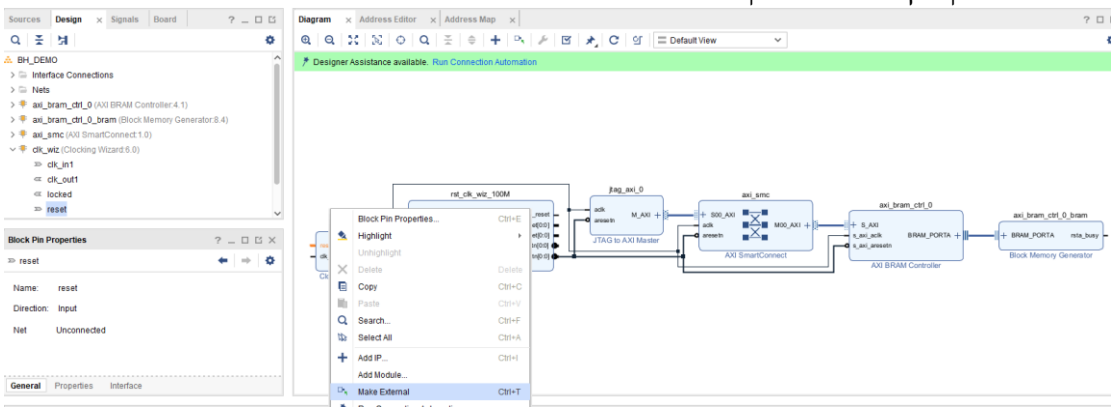
נא לסמן את כל הממשקים ב√



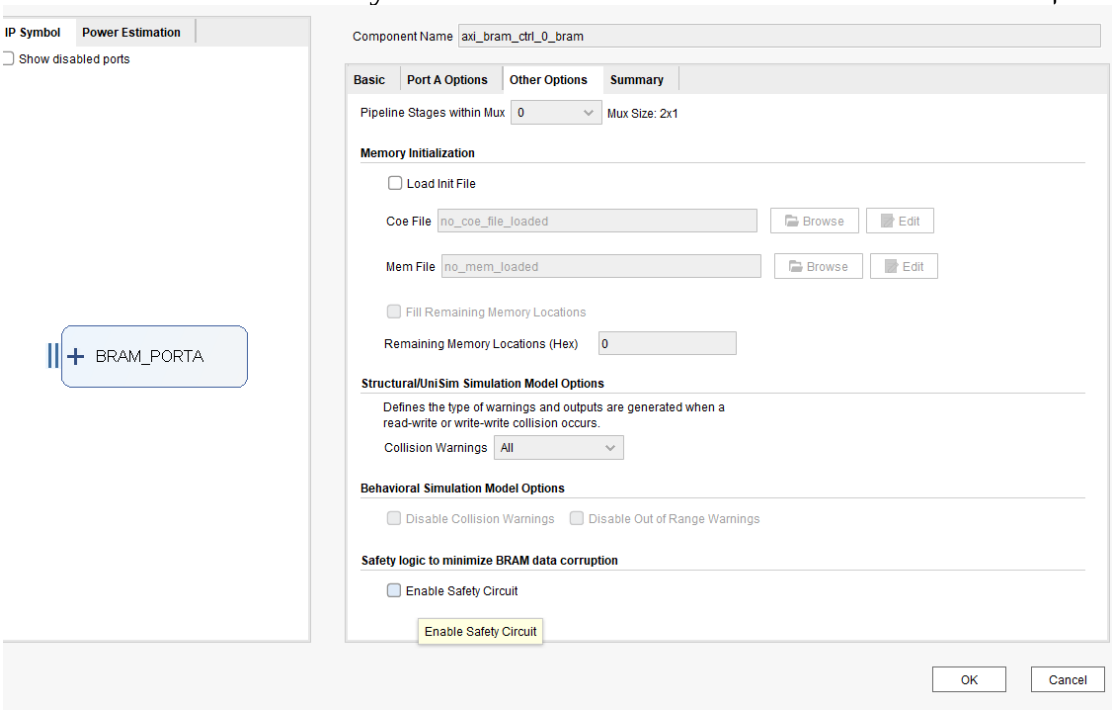
כעת התמונה שתתקבל הינה:



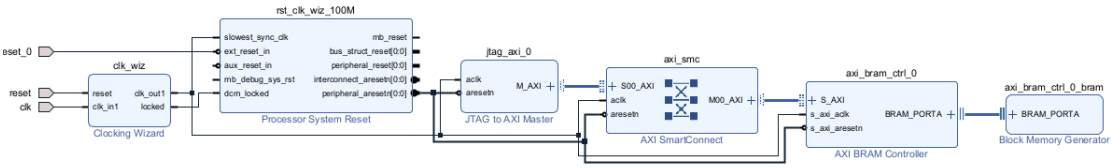
ניצור ממשקים חיצוניים כמצויין לעיל שימו לב לשמות האותות החיצוניים נדרשים להיות זהים לשמות שמופיעים כאן בהתאמה (RESET,reset_0 ,CLK)באם ברצונכם להשתמש בקובץ XDC שתקבלו



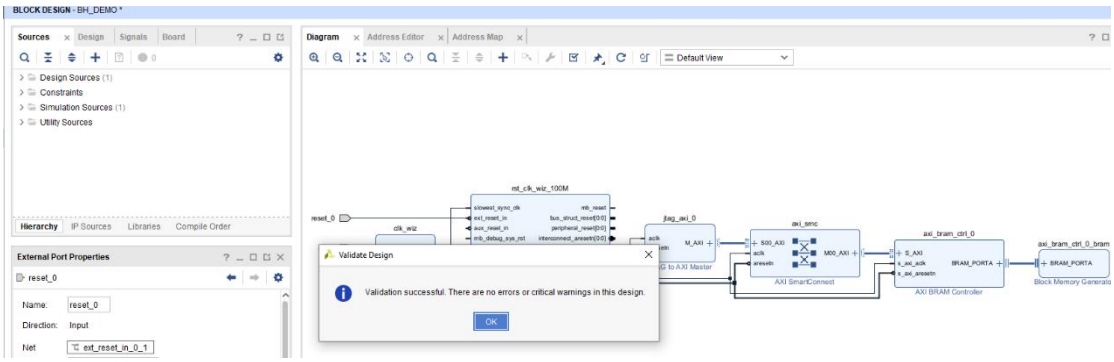
נלחץ פעמיים על רכיב הBRAM ונוריד את √ מ enable safety circuit



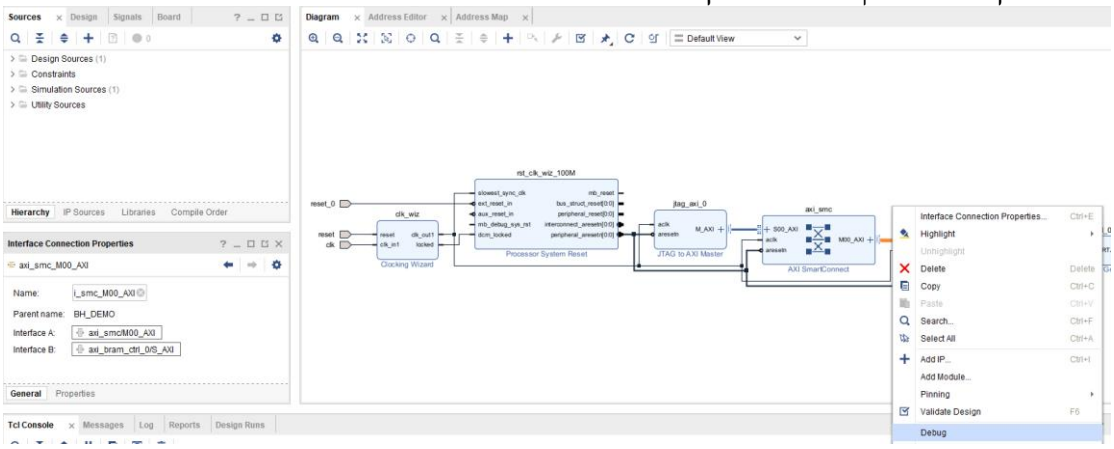
כעת התמונה שתתקבל היא



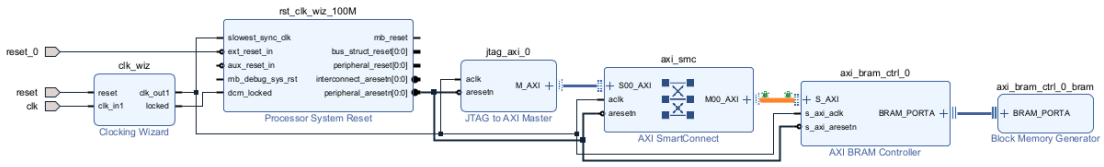
נבצע ולדיציה לתכן ונוודא שהכול מחובר שימו לב לשמות האותות החיצוניים נדרשים להיות זהים לשמות שמופיעים כאן בהתאמה (RESET,reset_0 ,CLK)באם ברצונכם להשתמש בקובץ XDC שתקבלו



כעת נסמן את ממשק AXI ונלחץ על DEBUG

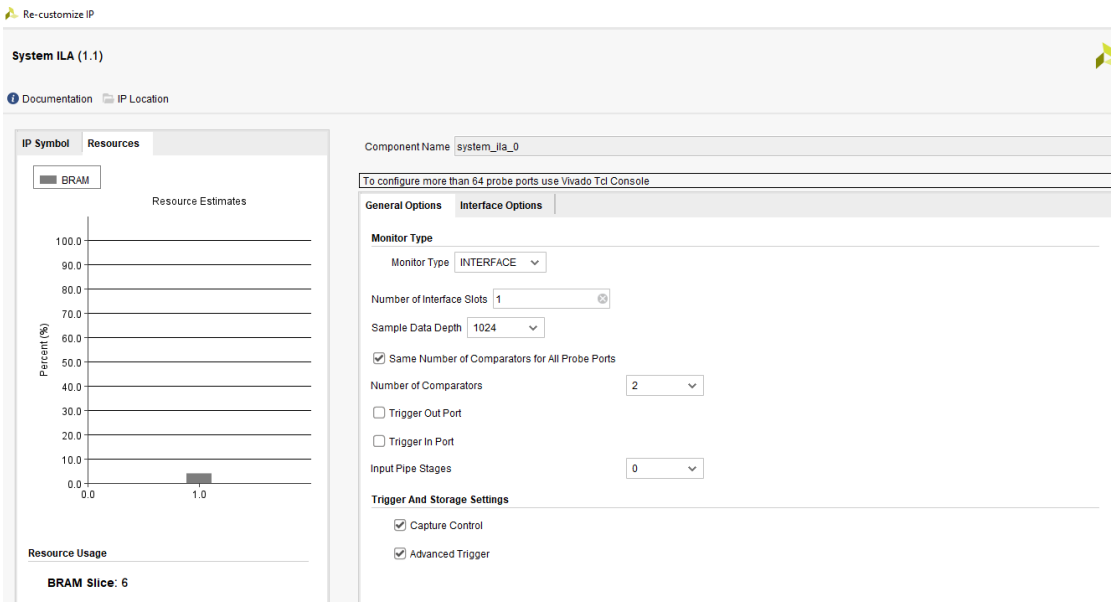


התמונה שתתקבל היא כלדקמן שימו לציור "החיפושית " על קווי הדיבג אותם אנחנו רוצים לבדוק

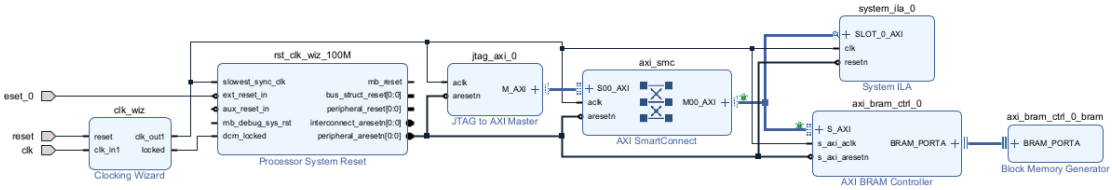


כעת נצטרך שוב לבצע פעולת run connection automation מדוע ?

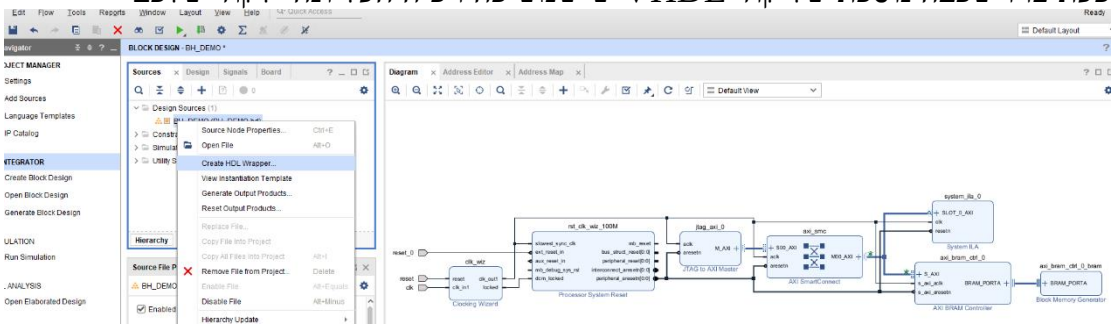
כעת תפתח חלונית שבו יוגדר chip scope ה core
נא להוסיף √ ל capture control והן לadvanced triger



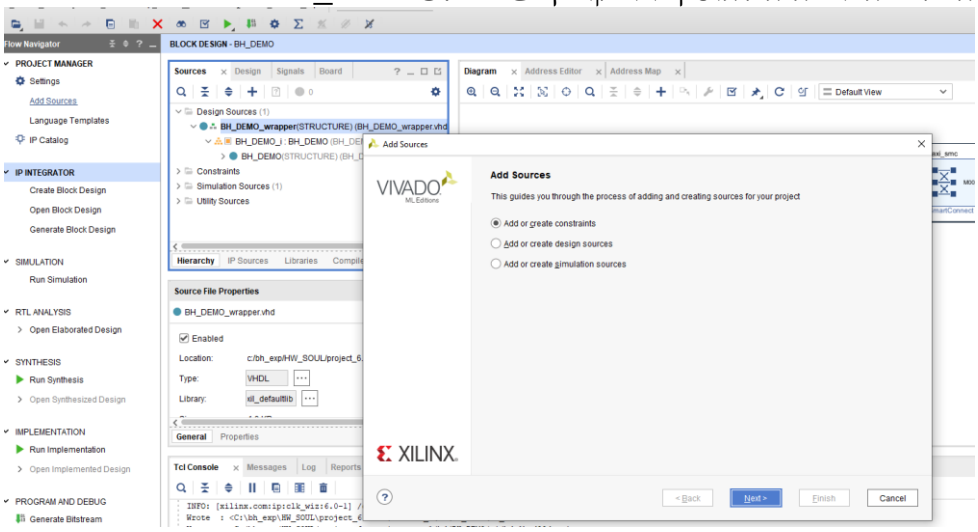
לאחר ריצת run connection automation תתקבל התמונה הבאה
שימו לב איזה בלוק התווסף ? למה ?



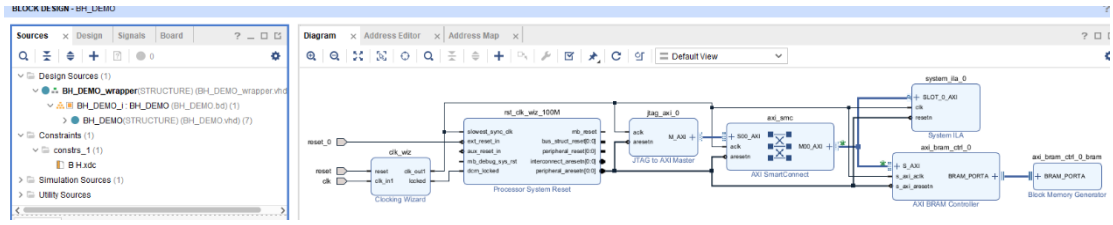
כעת צרו שכבה נוספת של קוד VHDL שישמש כהירכיה העליונה לקוד שלכם



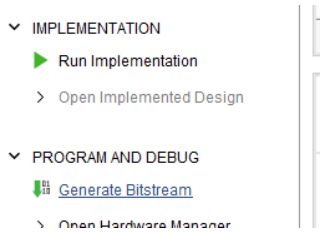
השלב הבא יהיה להוסיף את קובץ BH_DEMO.XDC



שימו לב להירכיה העליונה שנוצרה עם הסיומת WRAPPER

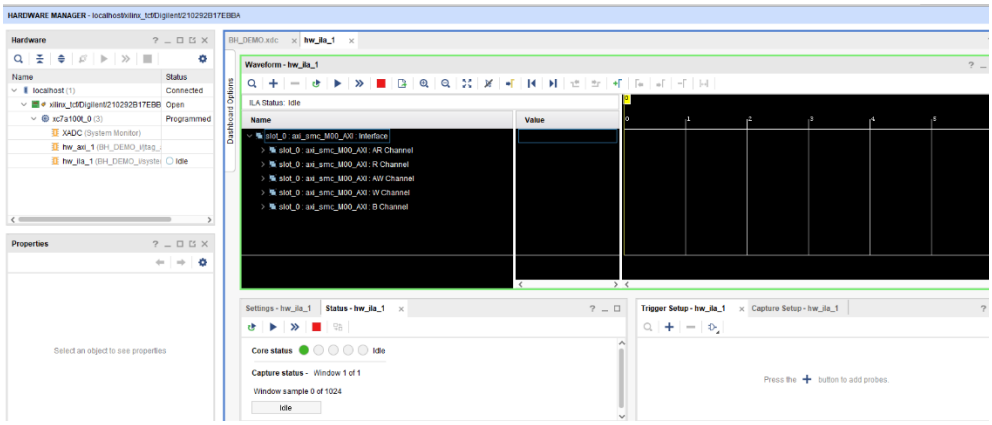


בשלב זה ניצור קובץ צריבה שיכלול את התכן וכן את הקובץ לטובת הchip scope



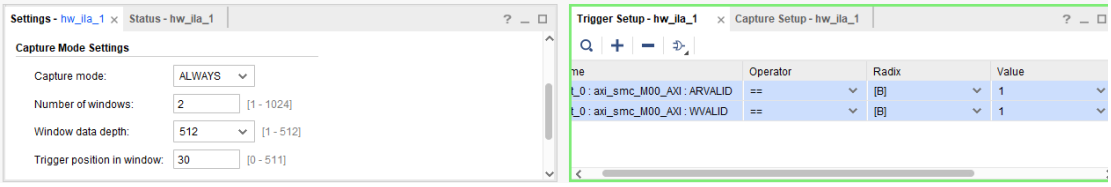
בצעו חיבור דרך הJTAG וצרבו את התכן -החלון שיפתח בפניכם יראה כך -שימו לב לחמשת ערוצי ה AXI שיופיעו בפניכם הם מכילים מאות אותות !

שימו לב לזה שאתם ממקמים את J15 המתג הראשון (0) SW למעלה וכן למטה את את L16 המתג השני(1) SW הסיבה היא הואיל ו RESET מחובר למתג השני עליו להיות לא פעיל ב'0' לוגי וכן RESET_0 מחובר למתג הראשון עליו להיות לא פעיל ב'1' לוגי -אחרת זה לא יעבוד !

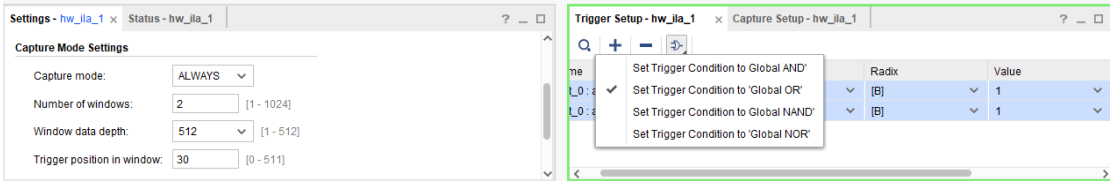


בשלב זה ניצור את התנאים שברצוננו לדגום -מה שנרצה לראות זה שתי פעולות של AXI פעולה אחת של כתיבה ל BRAM ולאחר מכן פעולה של קריאה -ולכן נגדיר שתי חלונות חלון של כתיבה וחלון של קריאה -לצורך כך נבחר את האותות של ARVAID לטובת הקריאה וכן WVALID לטובת הכתיבה

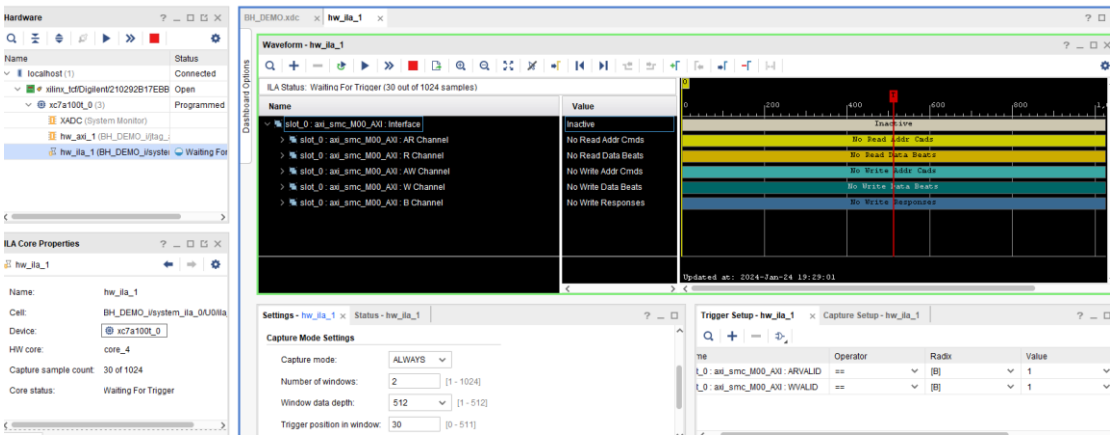
מדוע מיקמנו את החלון במקום 30 ?



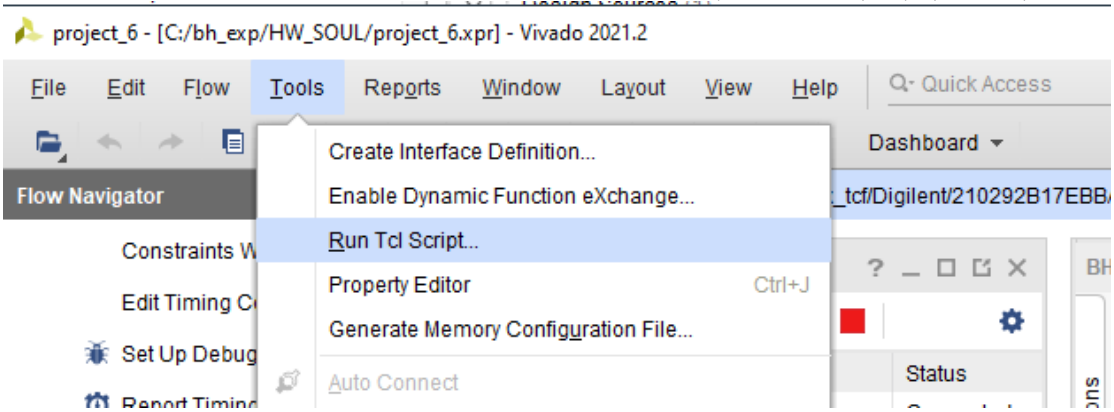
שימו לב שבן התנאים נדרש להיות שער OR מדוע ?



בשלב זה נלחץ על כפתור הPLAY שימו בצד שמאל שעון החול יחכה לדגימה



זה הזמן להריץ קובץ TCL שנקרא BH_DEMO.TCL



נעיין בפקודות שיש בקובץ BH_DEMO.TCL

פקודה ראשונה לבצע RESET להתקן JTAG TO AXI

reset_hw_axi [get_hw_axis hw_axi_1]

פקודה שנייה הכנת התשדורת **לכתיבה** להתקן JTAG TO AXI לא שידור!הפקודה כוללת : סוג הפקודה במקרה הזה כתיבה -הכתובת אליה מיועד המידע – תכולת המידע שיכתב בפועל(תזכרו שהמידע הוא קבוע 32 ביט !)-וכן את גודל הBURST במקרה זה 4

create_hw_axi_txn write_txn [get_hw_axis hw_axi_1] -address c0000000 -data {44444444_33333333_22222222_11111111} -len 4 -type write

פקודה שלישית זה שידור המידע שהכנו בפקודה שנייה להתקן JTAG TO AXI

run_hw_axi [get_hw_axi_txns write_txn]

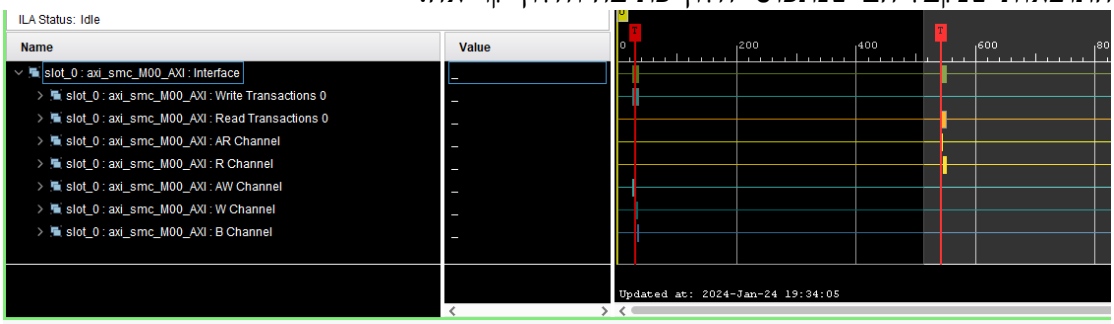
פקודה רביעית הכנת התשדורת **לקריאה** מהתקן JTAG TO AXI לא שידור!הפקודה כוללת : סוג הפקודה במקרה הזה כתיבה -הכתובת אליה מיועד המידע –וכן את גודל הBURST במקרה זה 4

create_hw_axi_txn read_txn [get_hw_axis hw_axi_1] -address c0000000 -len 4 -type read

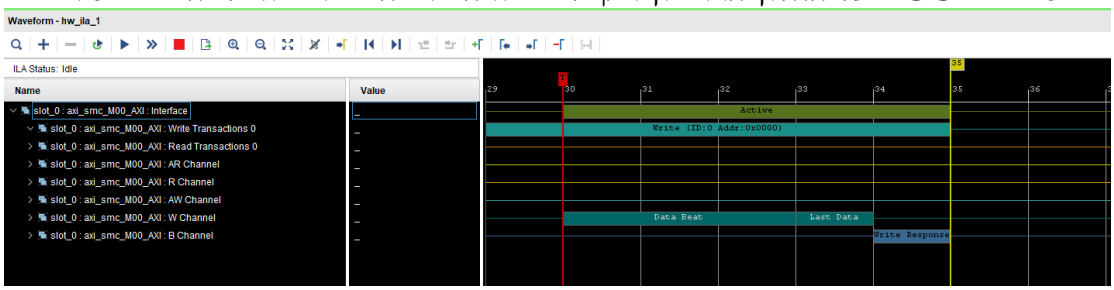
פקודה חמישית זה קריאת המידע שהכנו בפקודה שנייה דרך התקן JTAG TO AXI מתוך BRAM

run_hw_axi [get_hw_axi_txns read_txn]

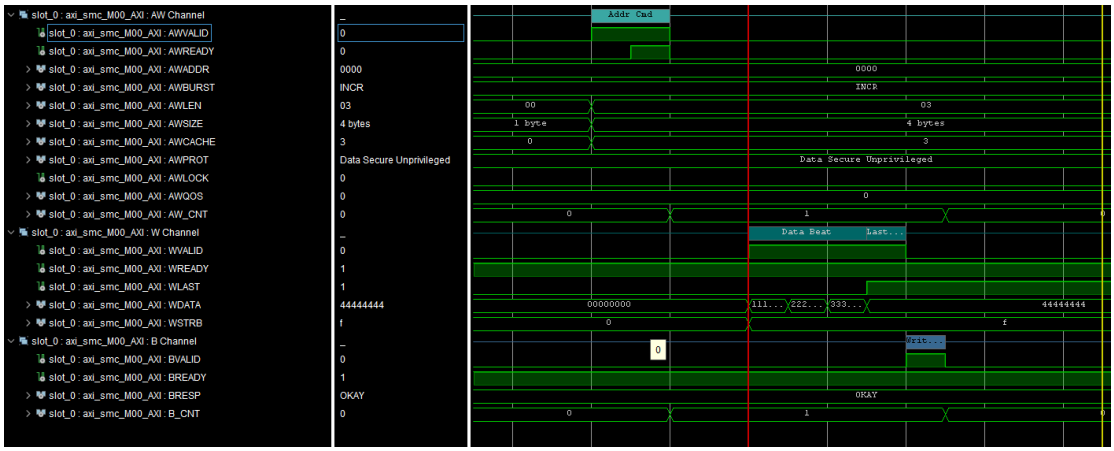
התוצאות שנקבל הם שנתפוס חלון כתיבה וחלון קריאה:



נבצע ZOOM IN על החלון הראשון ונקבל ב"צורה מובנה " פרשנות למה שביצענו



כמובן שנוכל לפתוח כל BUS ולראות את פירוט האותות כמו שלמדנו בכיתה
במקרה הזה ישנו פירוט של אותות של ערוצי הכתיבה.



בהצלחה

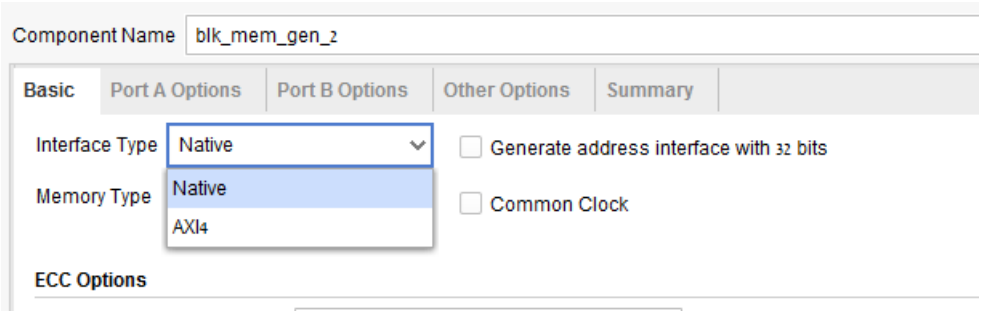
החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו

בס"ד

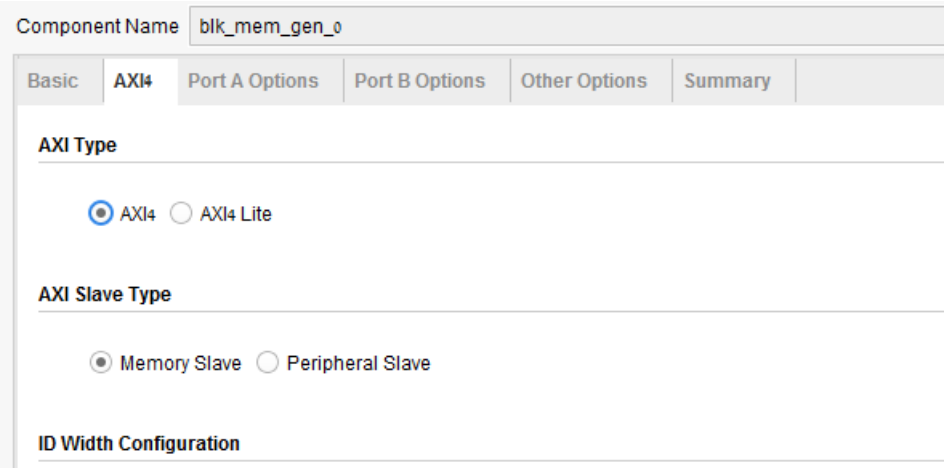
תרגיל 20 הכרות ותפעול עם ממשק AXI

<u>שלב א יצירת CORE וביצוע סימולציה:</u>

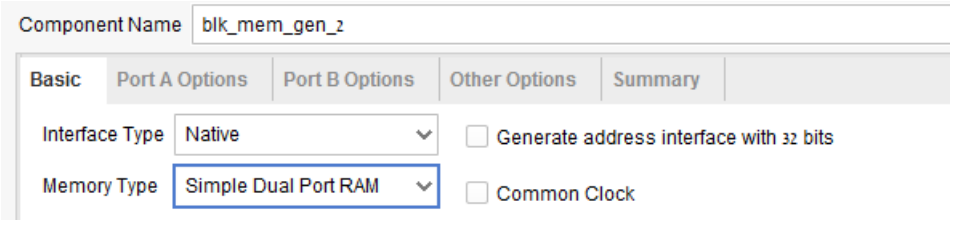
1. פתח פרויקט חדש
2. פתח את IP CATLOG
3. הגדר את זיכרון עם תצורת ממשק AXI לפי הפרמטרים הבאים:
4. בחר את הממשק לזיכרון כ- AXI



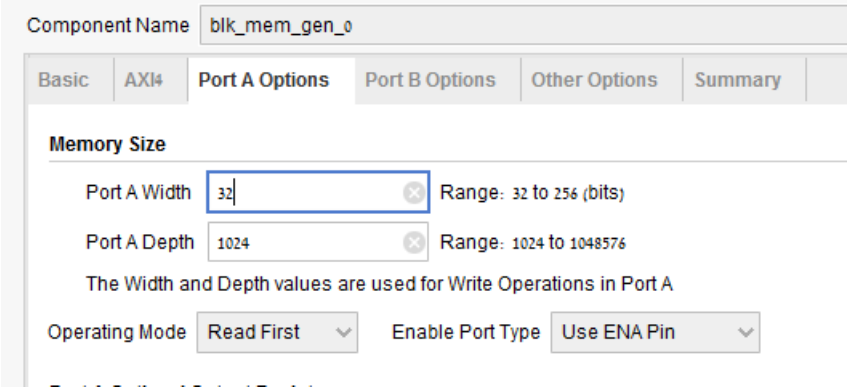
5. לטובת הסימולציה בחר את הממשק כ- AXI (שימו לב בתרגיל בפועל בשלב ב נעבוד AXI4LITE)



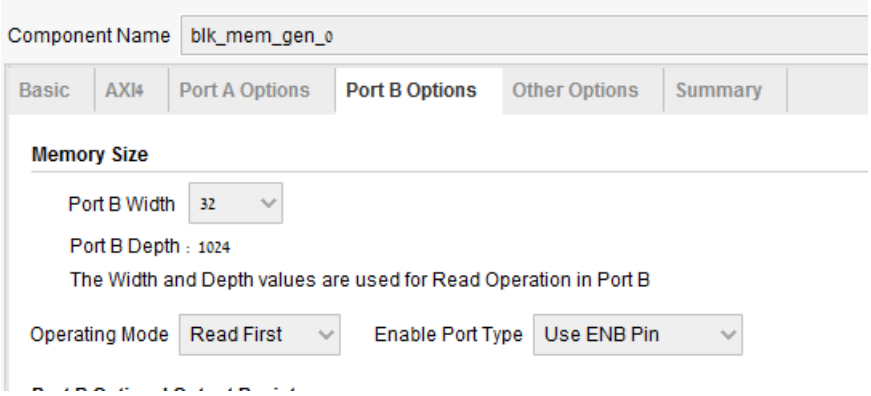
1. הגדר את הזיכרון כ-simple dual ram



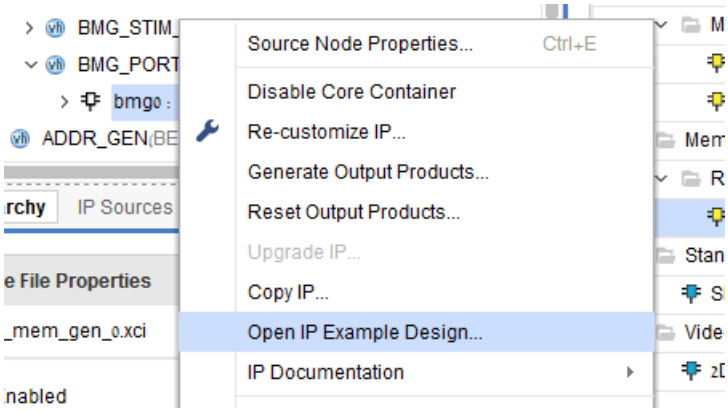
2. הגדר אתPORTA כרוחב 32 ביט ל1024 תאים



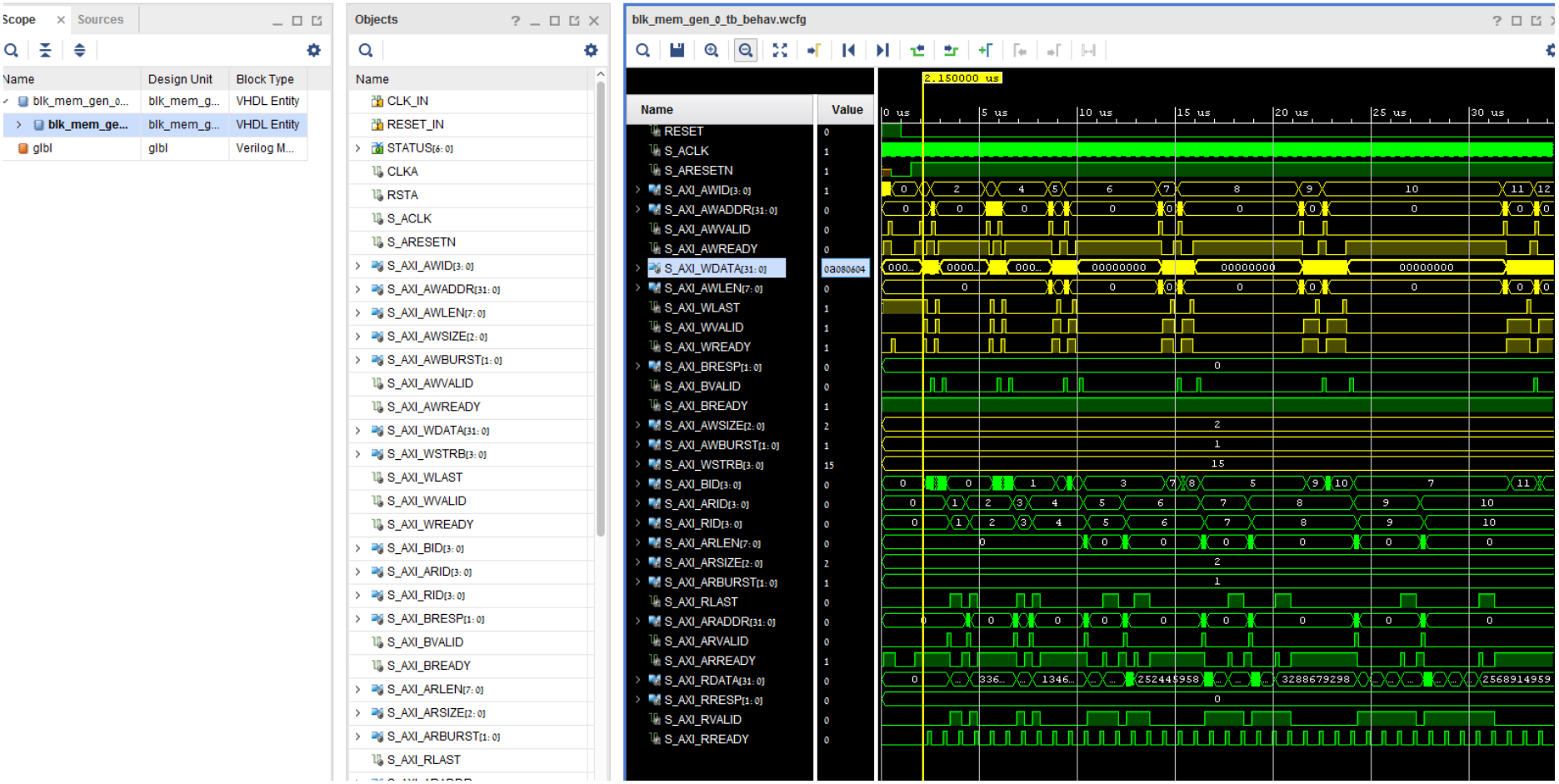
3. הגדר אתPORTB כרוחב 32 ביט ל1024 תאים



4. עמוד על CORE שיצרת והרץ את OPEN IP EXMAPLE(הגדל זמן ריצה בצורה ניכרת ל- 1MS)

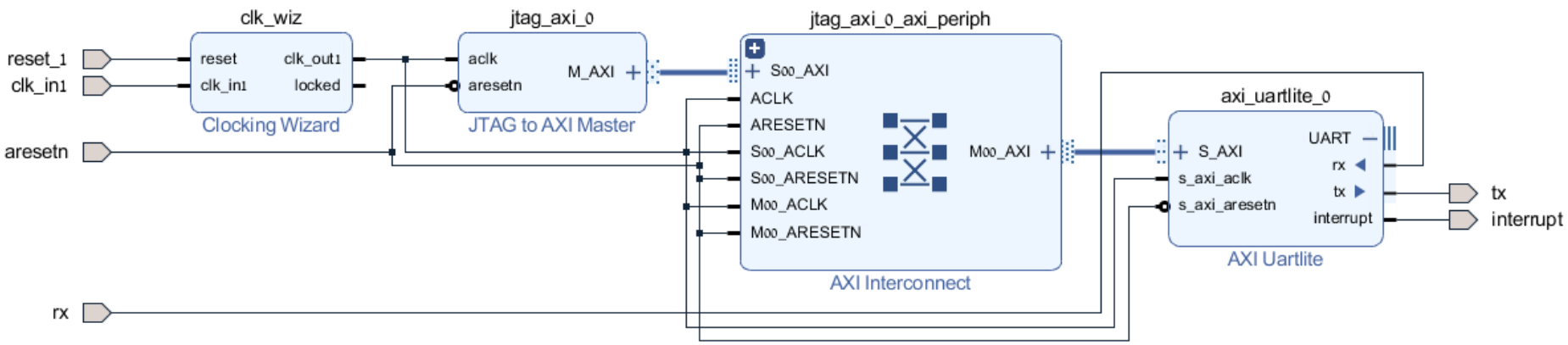


בחר צורות גלים שמתארות את ממשק ה AXI לפי הרשימה לעיל והרץ שוב


Scope
Sources
Name
Design Unit
Block Type
blk_mem_gen_0...
blk_mem_g...
VHDL Entity
blk_mem_ge...
blk_mem_g...
VHDL Entity
glbl
glbl
Verilog M...
Objects
Name
CLK_IN
RESET_IN
STATUS[6:0]
CLKA
RSTA
S_ACLK
S_ARESETN
S_AXI_AWID[3:0]
S_AXI_AWADDR[31:0]
S_AXI_AWLEN[7:0]
S_AXI_AWSIZE[2:0]
S_AXI_AWBURST[1:0]
S_AXI_AWVALID
S_AXI_AWREADY
S_AXI_WDATA[31:0]
S_AXI_WSTRB[3:0]
S_AXI_WLAST
S_AXI_WVALID
S_AXI_WREADY
S_AXI_BID[3:0]
S_AXI_ARID[3:0]
S_AXI_RID[3:0]
S_AXI_BRESP[1:0]
S_AXI_BVALID
S_AXI_BREADY
S_AXI_ARLEN[7:0]
S_AXI_ARSIZE[2:0]
S_AXI_ARBURST[1:0]
S_AXI_RLAST
blk_mem_gen_0_tb_behav.wcfg
2.150000 us
Name
Value
0 us
5 us
10 us
15 us
20 us
25 us
30 us
RESET
S_ACLK
S_ARESETN
S_AXI_AWID[3:0]
S_AXI_AWADDR[31:0]
S_AXI_AWVALID
S_AXI_AWREADY
S_AXI_WDATA[31:0]
S_AXI_AWLEN[7:0]
S_AXI_WLAST
S_AXI_WVALID
S_AXI_WREADY
S_AXI_BRESP[1:0]
S_AXI_BVALID
S_AXI_BREADY
S_AXI_AWSIZE[2:0]
S_AXI_AWBURST[1:0]
S_AXI_WSTRB[3:0]
S_AXI_BID[3:0]
S_AXI_ARID[3:0]
S_AXI_RID[3:0]
S_AXI_ARLEN[7:0]
S_AXI_ARSIZE[2:0]
S_AXI_ARBURST[1:0]
S_AXI_RLAST
S_AXI_ARADDR[31:0]
S_AXI_ARVALID
S_AXI_ARREADY
S_AXI_RDATA[31:0]
S_AXI_RRESP[1:0]
S_AXI_RVALID
S_AXI_RREADY
0a080604
252445958
3288679298
2568914959

שלב ב

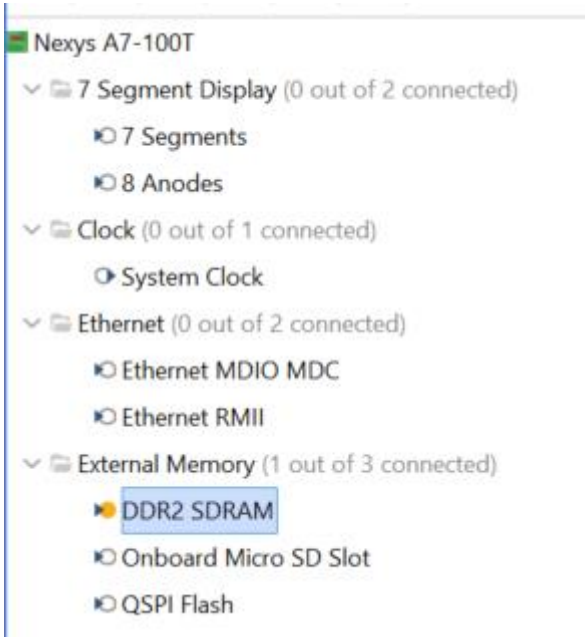
1. בנה בעזרת ה-IP INETGARTOR את התכנון הבא

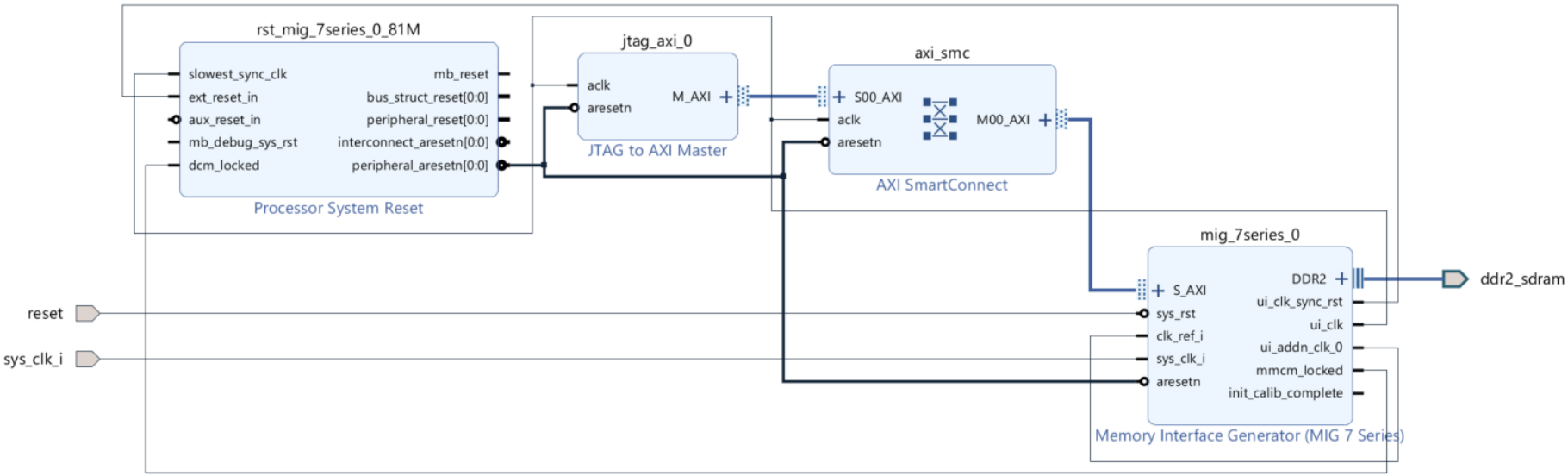


2. נא וודא שאין טעויות תכן בעזרת VALIDATE
3. נא הוסף את קובץ XDC בהתאם (bh_llb)
4. הרץ את SCRIPT הבא:(bh_demo_uart_transsmit)
5. נא בדוק בטרנימל שאכן מתקבל השידור ששלחת (ABCDE)

שלב ג

1. בנה בעזרת ה-IP INETGARTOR את התכנון הבא
2. נא הוסף את ממשק DDR2 בעזרת הTAB הבא




rst_mig_7series_0_81M
slowest_sync_clk
ext_reset_in
aux_reset_in
mb_debug_sys_rst
dcm_locked
mb_reset
bus_struct_reset[0:0]
peripheral_reset[0:0]
interconnect_aresetn[0:0]
peripheral_aresetn[0:0]
Processor System Reset
jtag_axi_0
aclk
aresetn
M_AXI
JTAG to AXI Master
axi_smc
S00_AXI
aclk
aresetn
M00_AXI
AXI SmartConnect
mig_7series_0
S_AXI
sys_rst
clk_ref_i
sys_clk_i
aresetn
DDR2
ui_clk_sync_rst
ui_clk
ui_addn_clk_0
mmcm_locked
init_calib_complete
Memory Interface Generator (MIG 7 Series)
reset
sys_clk_i
ddr2_sdram

1. נא וודא שאין טעויות תכן בעזרת VALIDATE
2. נא בדוק למול הקובץ XDC בהתאם (bh_ddr_demo_warpper)
3. הרץ את SCRIPT הבא:(bh_demo_DDR2)
4. נא בדוק שמתקבלות התוצאות הבאות

```
#reset_hw_axi [get_hw_axis hw_axi_1]
#create_hw_axi_txn write_txn [get_hw_axis hw_axi_1] -address 80000000 -data {44444444_33333333_22222222_11111111} -len 4 -type write
#run_hw_axi [get_hw_axi_txns write_txn]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 44444444_33333333_22222222_11111111
#create_hw_axi_txn read_txn [get_hw_axis hw_axi_1] -address 80000004 -len 4 -type read
#run_hw_axi [get_hw_axi_txns read_txn]
INFO: [Labtoolstcl 44-481] READ DATA is: 1111bbbb44444444333333332222222 2
#reset_hw_axi [get_hw_axis hw_axi_1]
#create_hw_axi_txn write_txn1 [get_hw_axis hw_axi_1] -address 800000A0 -data {12345678_12345678_87654321_87654321_12345678_12345678_87654321_87654321} -len 8 -type write
#run_hw_axi [get_hw_axi_txns write_txn1]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 12345678_12345678_87654321_87654321_12345678_12345678_87654321_87654321
#create_hw_axi_txn read_txn1 [get_hw_axis hw_axi_1] -address 800000A0 -len 8 -type read
#run_hw_axi [get_hw_axi_txns read_txn1]
INFO: [Labtoolstcl 44-481] READ DATA is: 1234567812345678876543218765432112345678123456788765432187654321
```

חומרים לשימוש פנימי בלבד אין להעביר ללא רשות מניר בלולו

בס"ד

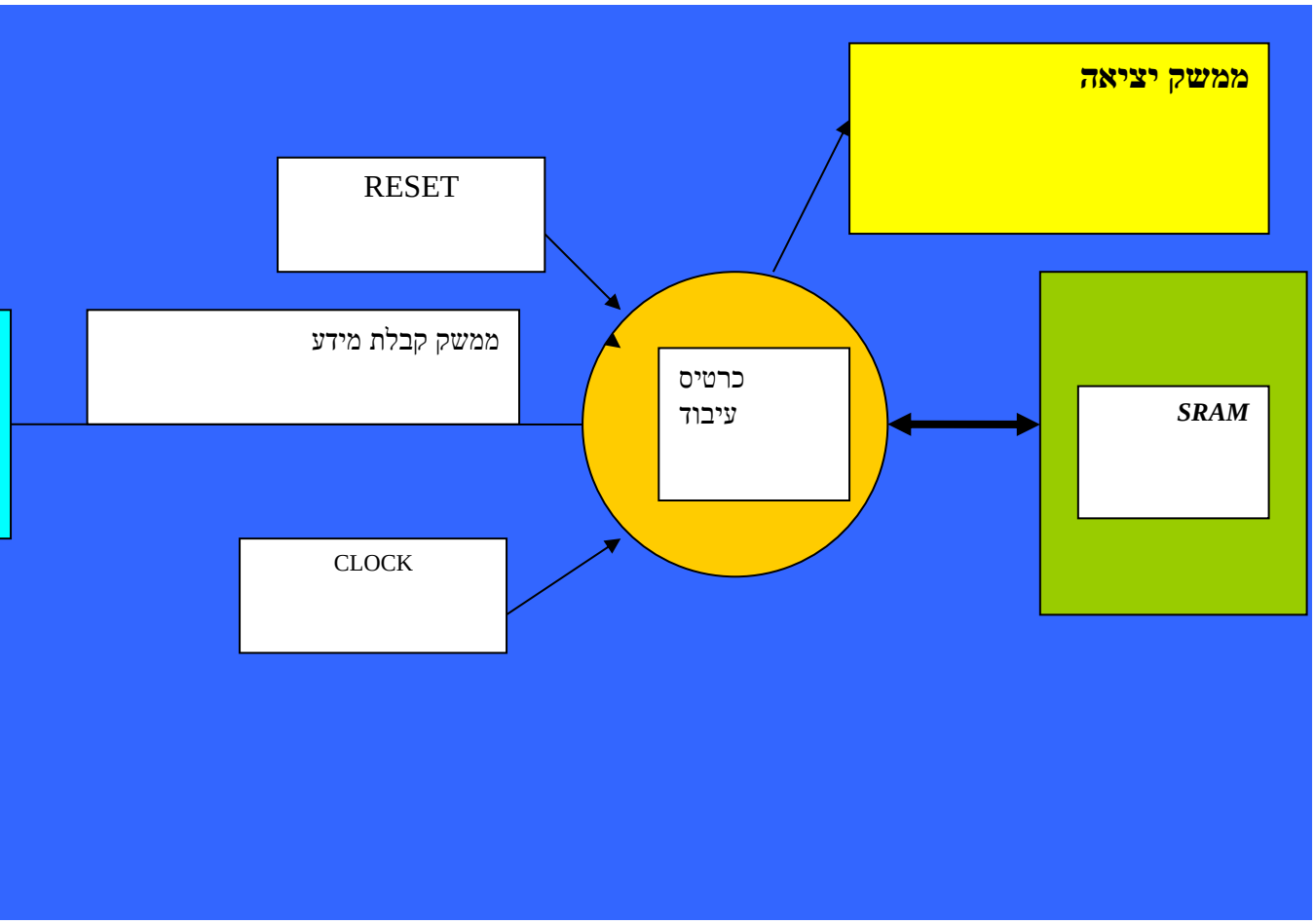
תרגיל 21 פתרון תקלות וקצבים משתנים

חלק א

בלשכה המרכזית לסטטיסטיקה בסין הוחלט לבנות מכשיר המקבל נתונים ממחוזות השונים במדינה המכשיר קולט את הנתונים ומציג את התפלגות הנתונים לפי הפרמטרים השונים
לדוגמא :אם נדרש לבחון את משקל האוכלוסייה המכשיר יציג עבור כל משקל נתון(מעוגל ליחידות שלמות) את כמות האנשים שיש באותו משקל
אם יש 1000000 אנשים במשקל 1 ק"ג עליכם לשדר את ליחידת האגירה שבמשקל 1ק "ג יש מיליון אנשים וכן על זה הדרך
כמות האנשים בסין עומד על כמיליארד איש

לצורך עליכם לתכנן החומרה הבאה :

1. כרטיס רכיב FPGA לצורך עיבוד המידע
2. ממשק קבלת מידע ממחשבי הלשכה המרכזית לסטטיסטיקה
   הממשק כולל את האותות הבאים :
   i. מידע מקבילי ברוחב 10 סיביות קצב 75 MHz
   ii. סיגנל DATA VALID שבא לציין את המידע רלוונטי או לא
   iii. שעון מלווה מידע בתדר 75MHz
3. זיכרון SRAM חיצוני לצורך בניית ההיסטוגרמה שעליכם לבחור מתוך דפי נתונים של יצרנים בשוק.
   כמובן לגודל הזיכרון ולמהירות שעובד הזיכרון יש השפעה על המחיר (ואנחנו רוצים כמה שיותר זול אבל עומד בבצועים)
4. ממשק העברת נתונים מ FPGA ליחידת אגירה שתשמור את הנתונים
   (עדיין לא תוכננה יחידת האגירה- שמורה לכם הזכות והחובה להציע את הממשק היעיל ביותר)
5. בכרטיס קיים אות RESET שמופעל עם הדלקת המתחים לכרטיס
6. נבחר שעון (CLOCK)בתדר של 50 מגה הרץ

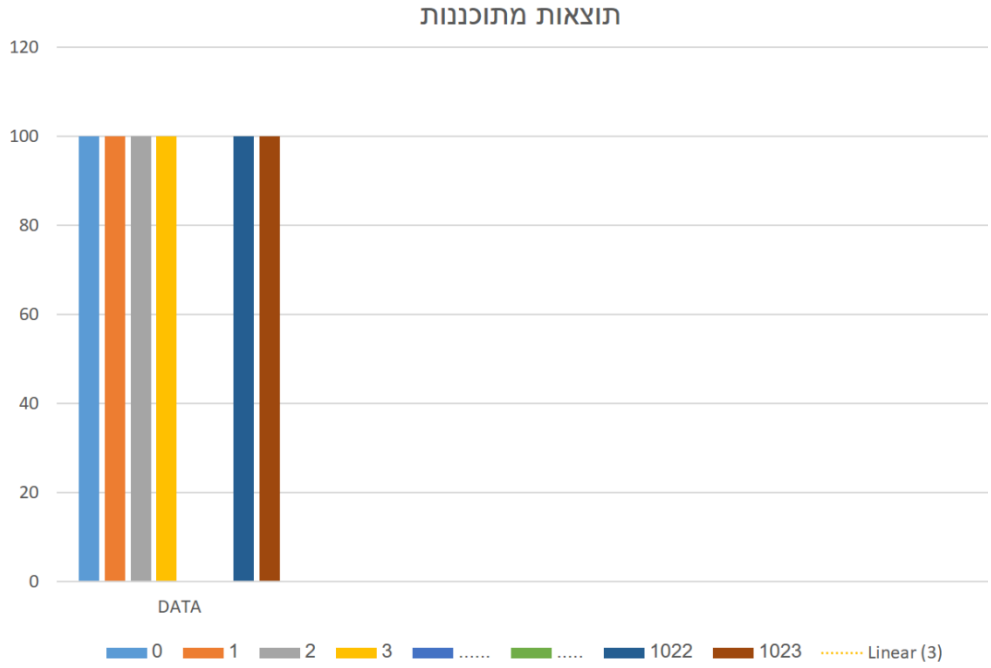


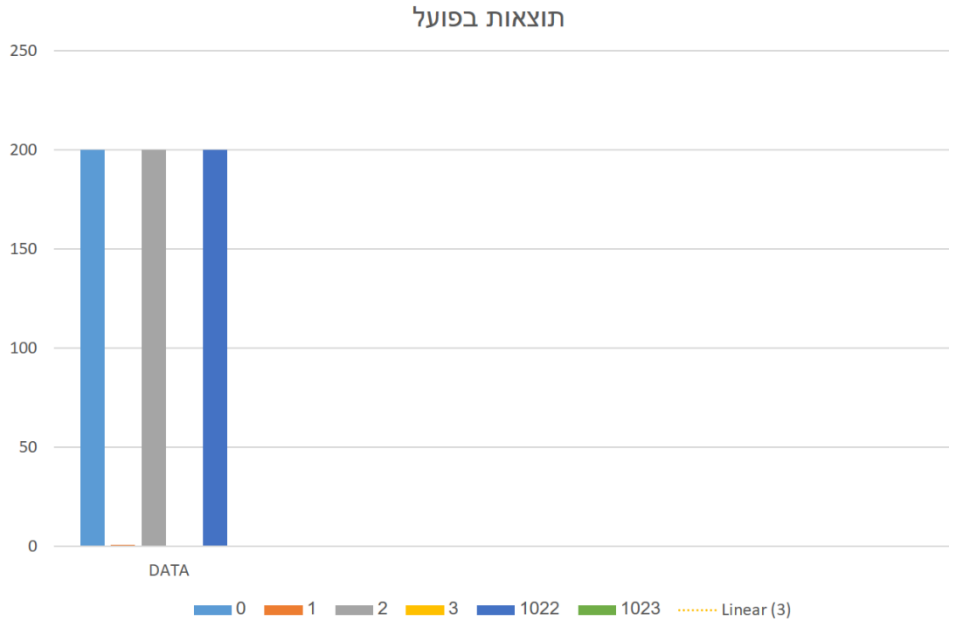
<u>שאלות הבנה בסיסית:</u>

12. הסבירו את " מכניזם "העבודה של המערכת ?
13. מה האתגר בממשק הכניסה איך תטפלו בזאת?
14. מה גודל SRAM שדרוש?
15. מה קצב העבודה מול SRAM?
16. מהו תדר השעון שתדרשו לעבודה ברכיב?
17. מה האפשריות המקובלות ליצירת שעון ברכיבי FPGA ?

חלק ב-פתרון תקלות :

בשעה טובה הגעתם למעבדה לבדוק התכנון
הכנסתם מידע סינטטי בעל ערך שווה לכל התאים
(פירוש הדבר שבן תא אחד למשנהו ההבדל הינו קטן בערך מספרי לדוגמא אם יש בתא 200 הערך 100 בתא 201 יהיה 100)
כאשר קיבלתם את תוצאות העיבוד שלכם קיבלתם את התוצאות הבאות
בתא מספר 0 היה פי 2 ממה שהיה צריך להיות שם
בתא מספר 1 לא היה כלום
בתא מספר 2 היה פי 2 ממה שהיה צריך להיות
בתא מספר 3 לא היה כלום
וכן על זה הדרך -לכל אורך הדרך
ממה יכולה לנבוע התקלה ?





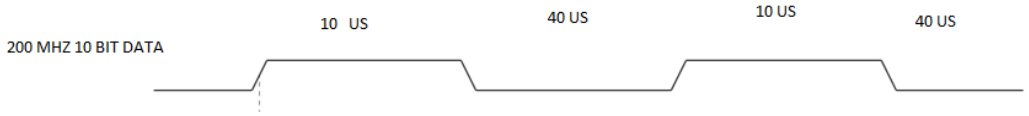
מה ההסבר הלוגי להנ"ל ?

חלק ג-עבודה עם קצבים משתנים :

בלשכה המרכזית לאור ההצלחה הוחלט לשדרג המכשיר
ולהוסיף לו מספר ערוצי שידור בו זמנית

הערוצים שמשדרים בצורה **מחזורית** הינם :

1. ערוץ 1 משדר למשך 10 מיקרו שניות בקצב של 200 מגה 10 ביט(מקבילי)ונח למשך 40 מיקרו



2. ערוץ 2 משדר למשך 10 מיקרו שניות בקצב של 100 מגה 10 ביט(מקבילי)ונח למשך 90 מיקרו
3. ערוץ 3 משדר למשך 10 מיקרו שניות בקצב של 150 מגה 10 ביט(מקבילי)ונח למשך 90 מיקרו

שאלות

1. האם התכנון שביצענו בסעיף 1 קביל לגבי הקצבים הנ"ל ?
2. מה עלינו לשנות
3. מדוע שאר התכנון קביל ?
4.

- בלשכה המרכזית ביקשו את השינוי כדלקמן ,מהו המשמעות לגבי התכן ?
- ערוץ 1 משדר למשך 100 מיקרו שניות בקצב של 200 מגה 10 ביט(מקבילי)ונח למשך 400 מיקרו
- ערוץ 2 משדר למשך 100 מיקרו שניות בקצב של 100 מגה 10 ביט(מקבילי)ונח למשך 900 מיקרו
- ערוץ 3 משדר למשך 100 מיקרו שניות בקצב של 150 מגה 10 ביט(מקבילי)ונח למשך 900 מיקרו

בהצלחה

החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלו

תרגיל 22 AXI PERFORMANCE MONITOR

הקדמה AXI PERFORMANCE MONITOR הינו כלי רב עוצמה לניתוח איכות תעבורת הAXI בטרם ניגש לתרגיל אצרף שני קישורים

1. מסמך רשמי של XILINX המתאר את יכולות הנ"ל
https://docs.amd.com/v/u/en-US/pg037_axi_perf_mon

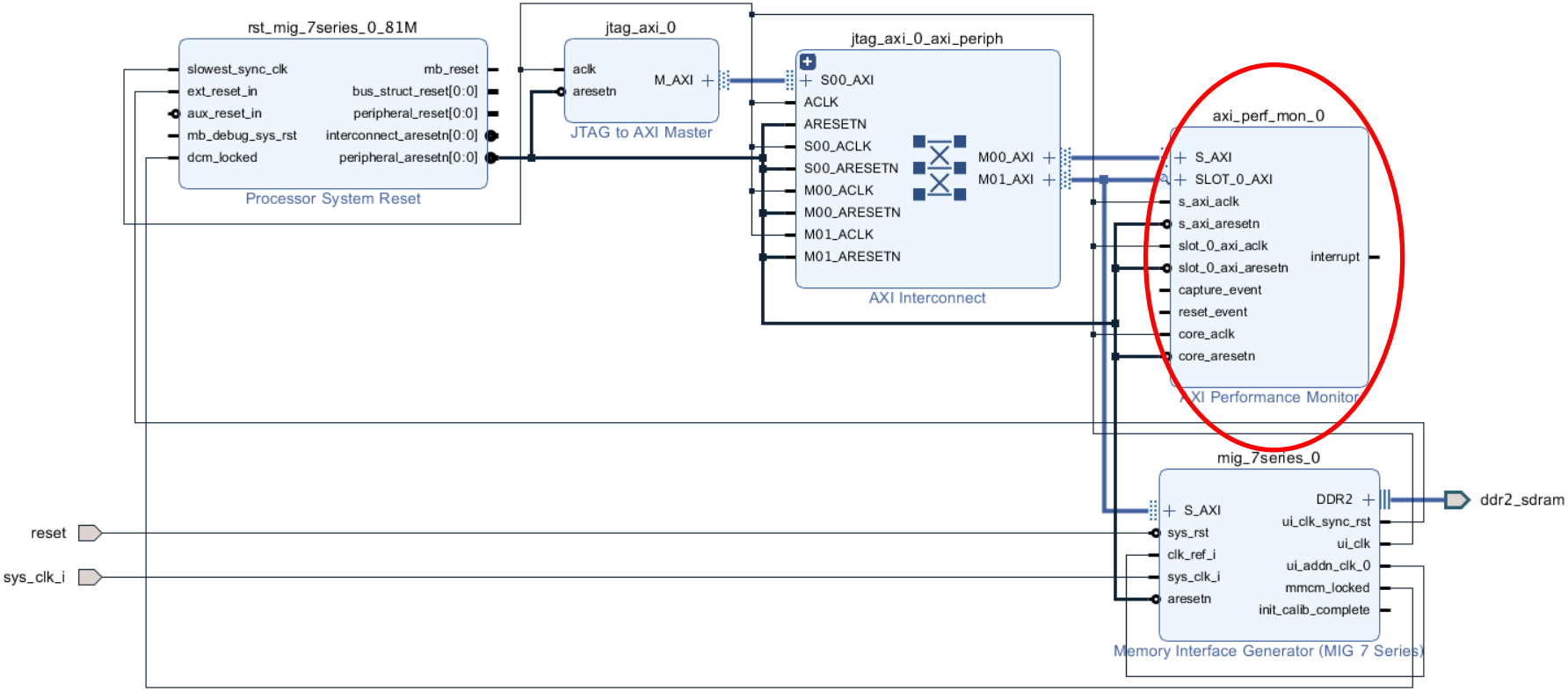
2. Measuring AXI latency and throughput performance
https://zipcpu.com/blog/2021/08/14/axiperf.html

במהלך מעבדה זו נבצע את השלבים הבאים :

1. בניית בעזרת ה-IP INETGARTOR תכנון הכולל JTAG TO AXI MIG AXI PERFORMANCE MONITOR
2. התכנון הנ"ל יקומפל עד לקבלת קובץ צריבה
3. הרצת TCL שבהם נפעיל בצורה בסיסית את AXI PERFORMANCE MONITOR ונעקוב אחר
   a. מספר מחזורי הכתיבה שנכתבו עד כה בAXI
   b. מונה של השעון הגלובלי

שלב א

בנה בעזרת ה-IP INETGARTOR את התכנון הבא



6. נא וודא שאין טעויות תכן בעזרת VALIDATE
7. אין צורך להוסיף את קובץ XDC למה ?

8. הרץ עד לקבלת קובץ צריבה
9. הרץ את SCRIPT הבא:(axi_sm)
10. נא בדקו בחלון ההודעות מתקבלות ההודעות הבאות סימנתי בעיגול אדום ובעיגול ירוק את ההודעות בהן נעסוק :

```
source C:/bh_exp/PHILIPS_K/DAY2/axi_sm.tcl
# reset_hw_axi [get_hw_axis hw_axi_1]
# create_hw_axi_txn write_txn [get_hw_axis hw_axi_1] -address 80000000 -data
{44444444_33333333_22222222_11111111} -len 4 -type write
# run_hw_axi [get_hw_axi_txns write_txn]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 44444444_33333333_22222222_11111111
# create_hw_axi_txn read_txn [get_hw_axis hw_axi_1] -address 80000004 -len 4 -type read
# run_hw_axi [get_hw_axi_txns read_txn]
INFO: [Labtoolstcl 44-481] READ DATA is: 1111bbbb44444444333333332222222 2
# reset_hw_axi [get_hw_axis hw_axi_1]
# create_hw_axi_txn write_txn1 [get_hw_axis hw_axi_1] -address 800000A0 -data
{12345678_12345678_87654321_87654321_12345678_12345678_87654321_87654321} -len 8 -type
write
# run_hw_axi [get_hw_axi_txns write_txn1]
INFO: [Labtoolstcl 44-481] WRITE DATA is:
12345678_12345678_87654321_87654321_12345678_12345678_87654321_87654321
# create_hw_axi_txn read_txn1 [get_hw_axis hw_axi_1] -address 800000A0 -len 8 -type read
# run_hw_axi [get_hw_axi_txns read_txn1]
INFO: [Labtoolstcl 44-481] READ DATA is:
12345678123456788765432187654321123456781234567887654321876543 21
```

```
# reset_hw_axi [get_hw_axis hw_axi_1]
# create_hw_axi_txn write_txn3 [get_hw_axis hw_axi_1] -address 44A00300 -data {00020002} -len 1 -type
write
# run_hw_axi [get_hw_axi_txns write_txn3]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 00020002
# create_hw_axi_txn write_txn4 [get_hw_axis hw_axi_1] -address 44A00044 -data {06010200} -len 1 -type
write
# run_hw_axi [get_hw_axi_txns write_txn4]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 06010200
# create_hw_axi_txn write_txn5 [get_hw_axis hw_axi_1] -address 44A00048 -data {06020100} -len 1 -type
write
# run_hw_axi [get_hw_axi_txns write_txn5]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 06020100
# create_hw_axi_txn write_txn6 [get_hw_axis hw_axi_1] -address 44A0004C -data {00005E1E} -len 1 -
type write
# run_hw_axi [get_hw_axi_txns write_txn6]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 00005E1E
# create_hw_axi_txn write_txn7 [get_hw_axis hw_axi_1] -address 44A00030 -data {00000001} -len 1 -type
write
# run_hw_axi [get_hw_axi_txns write_txn7]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 00000001
# create_hw_axi_txn write_txn8 [get_hw_axis hw_axi_1] -address 44A00034 -data {00000002} -len 1 -type
write
```

```
# run_hw_axi [get_hw_axi_txns write_txn8]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 00000002
# create_hw_axi_txn write_txn9 [get_hw_axis hw_axi_1] -address 44A00024 -data {00001000} -len 1 -type
write
# run_hw_axi [get_hw_axi_txns write_txn9]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 00001000
# create_hw_axi_txn write_txn10 [get_hw_axis hw_axi_1] -address 44A00028 -data {00000001} -len 1 -
type write
# run_hw_axi [get_hw_axi_txns write_txn10]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 00000001
# create_hw_axi_txn write_txn11 [get_hw_axis hw_axi_1] -address 44A00300 -data {00010001} -len 1 -
type write
# run_hw_axi [get_hw_axi_txns write_txn11]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 00010001
# reset_hw_axi [get_hw_axis hw_axi_1]
# create_hw_axi_txn write_txn13 [get_hw_axis hw_axi_1] -address 80000000 -data
{44444444_33333333_22222222_11111111} -len 4 -type write
# run_hw_axi [get_hw_axi_txns write_txn13]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 44444444_33333333_22222222_11111111
# create_hw_axi_txn read_txn13 [get_hw_axis hw_axi_1] -address 80000004 -len 4 -type read
# run_hw_axi [get_hw_axi_txns read_txn13]
INFO: [Labtoolstcl 44-481] READ DATA is: 1111bbbb44444444333333332222222 2
# reset_hw_axi [get_hw_axis hw_axi_1]
```

```
# create_hw_axi_txn write_txn14 [get_hw_axis hw_axi_1] -address 800000A0 -data
{12345678_12345678_87654321_87654321_12345678_12345678_87654321_87654321} -len 8 -type
write
# run_hw_axi [get_hw_axi_txns write_txn14]
INFO: [Labtoolstcl 44-481] WRITE DATA is:
12345678_12345678_87654321_87654321_12345678_12345678_87654321_87654321
# create_hw_axi_txn read_txn14 [get_hw_axis hw_axi_1] -address 800000A0 -len 8 -type read
# run_hw_axi [get_hw_axi_txns read_txn14]
INFO: [Labtoolstcl 44-481] READ DATA is:
12345678123456788765432187654321123456781234567887654321876543 21
# create_hw_axi_txn read_txn15 [get_hw_axis hw_axi_1] -address 44A00200  -len 1 -type read
# run_hw_axi [get_hw_axi_txns read_txn15]
INFO: [Labtoolstcl 44-481] READ DATA is: 00000002
# create_hw_axi_txn write_txn15 [get_hw_axis hw_axi_1] -address 80000000 -data
{44444444_33333333_22222222_11111111} -len 4 -type write
# run_hw_axi [get_hw_axi_txns write_txn15]
INFO: [Labtoolstcl 44-481] WRITE DATA is: 44444444_33333333_22222222_11111111
# create_hw_axi_txn read_txn16 [get_hw_axis hw_axi_1] -address 44A00200  -len 1 -type read
# run_hw_axi [get_hw_axi_txns read_txn16]
INFO: [Labtoolstcl 44-481] READ DATA is: 00000003
# create_hw_axi_txn write_txn16 [get_hw_axis hw_axi_1] -address 80000000 -data
{44444444_33333333_22222222_11111111} -len 4 -type write
# run_hw_axi [get_hw_axi_txns write_txn16]
```

```
INFO: [Labtoolstcl 44-481] WRITE DATA is: 44444444_33333333_22222222_11111111
# create_hw_axi_txn read_txn17 [get_hw_axis hw_axi_1] -address 44A00200  -len 1 -type read
# run_hw_axi [get_hw_axi_txns read_txn17]
INFO: [Labtoolstcl 44-481] READ DATA is: 00000004


# create_hw_axi_txn read_txn18 [get_hw_axis hw_axi_1] -address 44A00004  -len 1 -type read
# run_hw_axi [get_hw_axi_txns read_txn18]
INFO: [Labtoolstcl 44-481] READ DATA is: 006eba0e
# create_hw_axi_txn read_txn19 [get_hw_axis hw_axi_1] -address 44A00004  -len 1 -type read
# run_hw_axi [get_hw_axi_txns read_txn19]
INFO: [Labtoolstcl 44-481] READ DATA is: 0088896a
# create_hw_axi_txn read_txn20 [get_hw_axis hw_axi_1] -address 44A00004  -len 1 -type read
# run_hw_axi [get_hw_axi_txns read_txn20]
INFO: [Labtoolstcl 44-481] READ DATA is: 008ffce7
```

<u>שאלות להבנה:</u>

מדוע המידע מתקדם בכתובת 44A00200 לאחר כל טרנזקציה ?

Sampled Metric Counter Registers		
0x0200	Sampled Metric Counter 0	Sampled Metric Counter 0 Register
0x0204	Sampled Incrementer 0	Sampled Incrementer 0 Register
0x0210	Sampled Metric Counter 1	Sampled Metric Counter 1 Register
0x0214	Sampled Incrementer 1	Sampled Incrementer 1 Register

שימו לב לכתובת בסיס איך שמופיעה בaddress editor

Diagram | Address Editor

Cell	Slave Interface	Base Name	Offset Address	Range	High Address
jtag_axi_0					
Data (32 address bits : 4G)					
axi_perf_mon_0	S_AXI	Reg	0x44A0_0000	64K	0x44A0_FFFF
mig_7series_0	S_AXI	memaddr	0x8000_0000	128M	0x87FF_FFFF

למה הערך של של המידע בכתובת address 44A00004 עולה ? הסתכלו בדפי נתונים מה הוא בא לציין ?

Register Space

This section details the registers and reset values of the AXI Performance Monitor core. Table 2-3 shows all the AXI Performance Monitor registers and addresses in advanced mode. Table 2-4 shows the registers and addresses in profile and trace modes.

Table 2-3: **Core Registers in Advanced Mode**

Address Offset	Register Name	Description
Global Clock Counter Registers		
0x0000	Global Clock Counter	Higher 32-bit data of the Global Clock Counter Register
0x0004	Global Clock Counter	Lower 32-bit data of the Global Clock Counter Register
Sample Interval Registers		
0x0024	Sample Interval	Sample Interval Time Configuration Register
0x0028	Sample Interval Control Register	Sample Interval Control Register

בס"ד

תרגיל 23 TIMING REPORTS

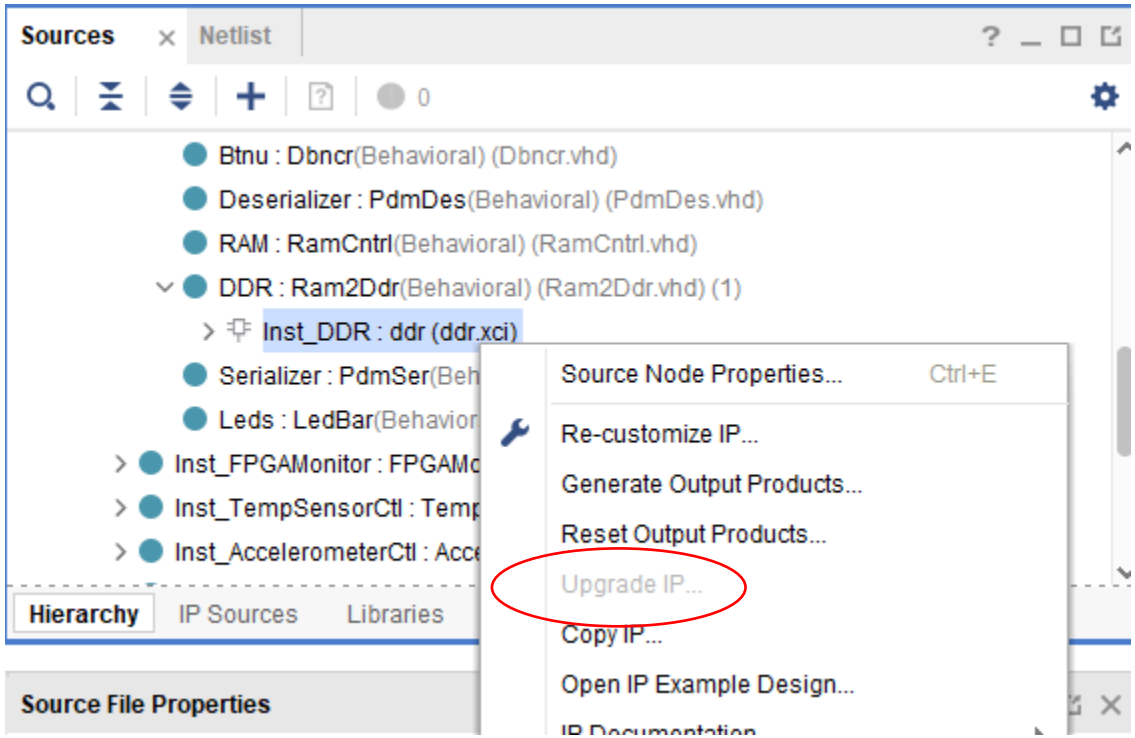
הקדמה :

מטרות המעבדה היא הרכשת הידע לבחינה של דוחו"ת TIMING של התכנים השונים איתם תעבדו במהלך הפרוייקטים השונים
יכולות הבחינה שעומדות לרשותנו היום הם מגוונות ובד"כ נותנות אפשרות בחינה של הדוחות מכמה "זוויות ראייה"
זוויות ראייה השונות נותנות לנו היכולת לעיתים קרובות להבין היכן נמצאת הבעיה ומה ניתן לעשות

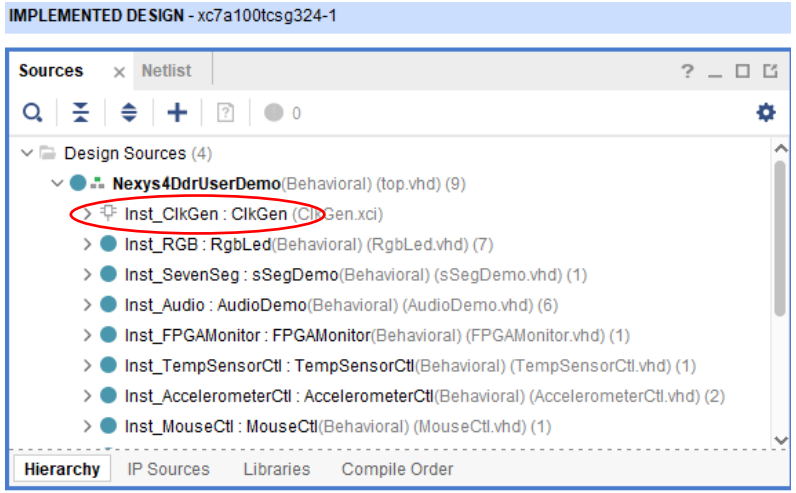
לצורך כך אבקש שתורידו את התכנים הבאים :(תכן שצרוב על הכרטיס NEXYS a7 -100T)
https://digilent.com/reference/programmable-logic/nexys-a7/demos/oob
שימו לב לגרסא שאתם מורידים למול גרסאת VIVADO שלכם (אפשר גרסא מוקדמת יותר לא מאוחרת יותר) לדוגמא אם עובדים עם גרסא 2019 אפשר להוריד תכנים של 2018 לא מוקדם יותר !

Board Variant	Release Tag	Release Downloads	Setup Instructions
Nexys A7-100T	100T/OOB/2022.1-1	Release ZIP Downloads	See *Using the Latest Release*, below
Nexys A7-50T	50T/OOB/2022.1-1	Release ZIP Downloads	See *Using the Latest Release*, below
Nexys A7-100T	100T/OOB/2021.1-1	Release ZIP Downloads	See *Using the Latest Release*, below
Nexys A7-50T	50T/OOB/2021.1-1	Release ZIP Downloads	See *Using the Latest Release*, below
Nexys A7-100T	100T/OOB/2020.1-1	Release ZIP Downloads	See *Using the Latest Release*, below
Nexys A7-50T	50T/OOB/2020.1-1	Release ZIP Downloads	See *Using the Latest Release*, below
Nexys A7-100T	v2018.2-1	Release ZIP downloads	Github README
Nexys A7-50T	v2018.2-1	Release ZIP downloads	Github README

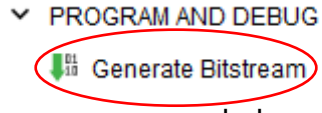
לאחר ההורדה באם הורדתם תכנים מוקדמים יותר נא לבצע UPGARDE IP



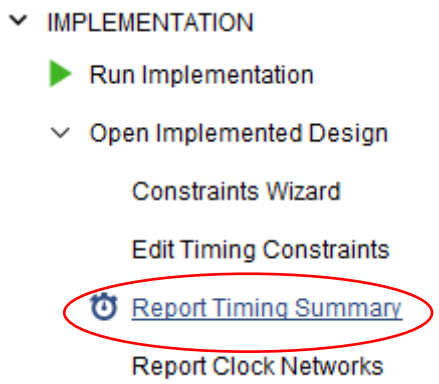
לאחר מכן כל IP אמורים להיות ללא ציור מנעול עליהם



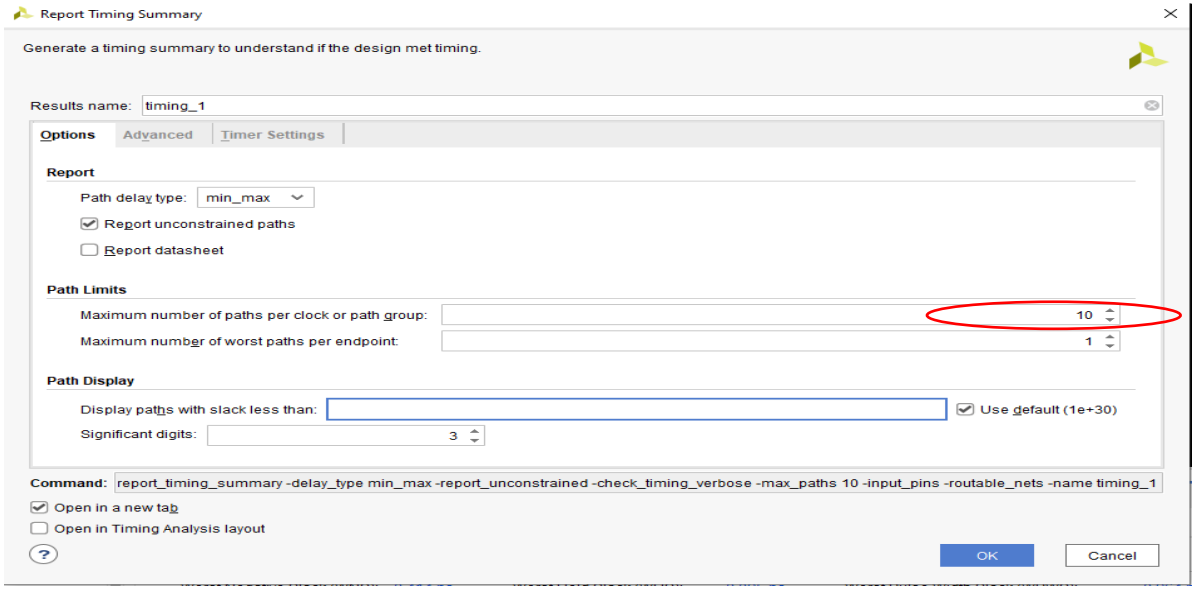
4. לאחר ההקדמות להנ"ל נא בצעו הרצה של התכנים שוב עד לרמת BITSTREAM



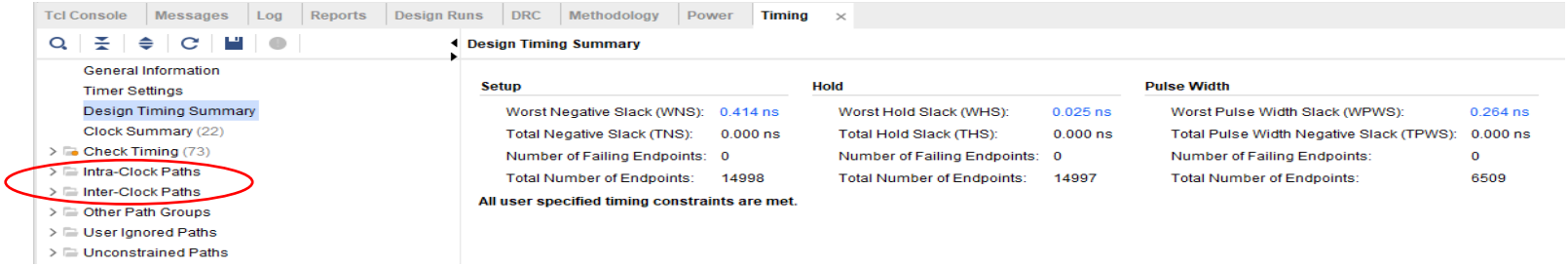
5. כעת הגענו לשלב שבו אנו בוחנים את הדוחו"ת לחצו בבקשה על מחולל הדוחו"ת



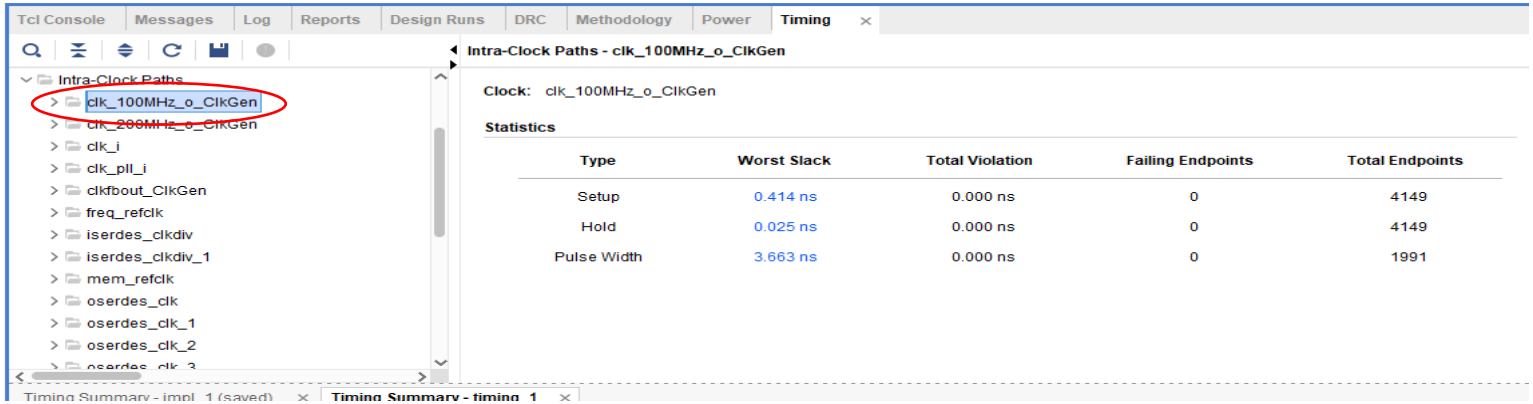
6. לתוכנת הVIVADO יש את כל הנתונים על מסלולי הTIMING בד"כ נציג את 10 המסלולים הגרועים ביותר ? למה ?
כמובן ניתן להרחיב את כמות המסוללים אותם אתם מעוייינים לראות



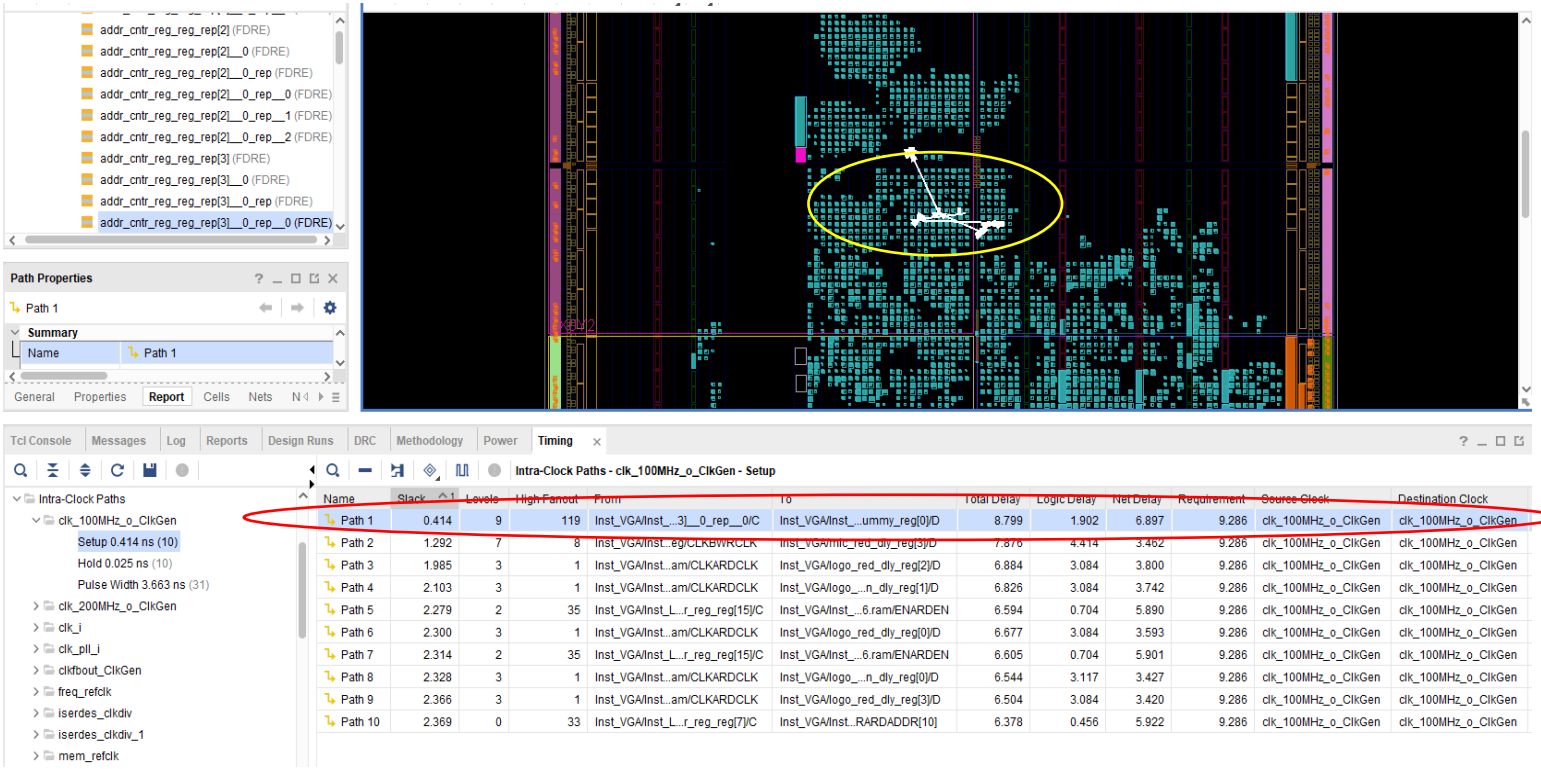
7. כעת יפתח לנו חלון הסיכום של ריכוז התוצאות:



שימו לב להבדל בן intra clock paths שמציין את ניתוח כל המסלולים באותו שעון עצמו למול inter clock path שמציין מעבר בן clock domains כעת נסתכל על אחד השעונים שמסומן לעיל

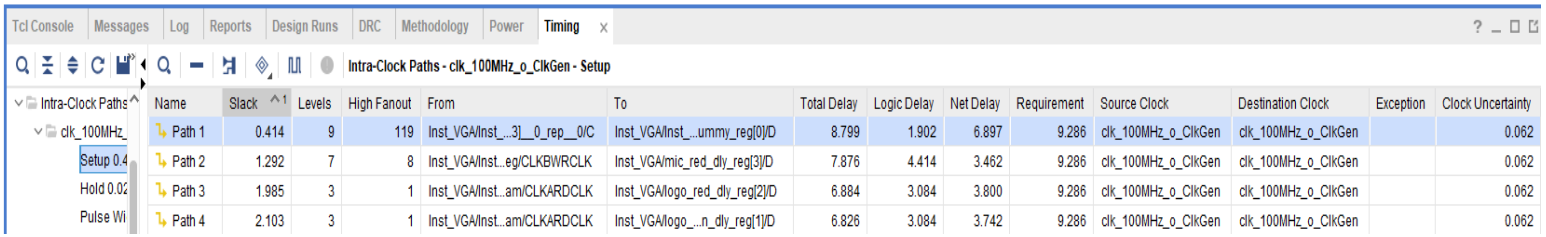


ניתן לראות את DEIVCE VIEW שמראה היכן ממוקם המסלול המנותח בתוך הרכיב


addr_cntr_reg_reg_rep[2] (FDRE)
addr_cntr_reg_reg_rep[2]__0 (FDRE)
addr_cntr_reg_reg_rep[2]__0_rep (FDRE)
addr_cntr_reg_reg_rep[2]__0_rep__0 (FDRE)
addr_cntr_reg_reg_rep[2]__0_rep__1 (FDRE)
addr_cntr_reg_reg_rep[2]__0_rep__2 (FDRE)
addr_cntr_reg_reg_rep[3] (FDRE)
addr_cntr_reg_reg_rep[3]__0 (FDRE)
addr_cntr_reg_reg_rep[3]__0_rep (FDRE)
addr_cntr_reg_reg_rep[3]__0_rep__0 (FDRE)
Path Properties
Path 1
Summary
Name
Path 1
General
Properties
Report
Cells
Nets
Tcl Console
Messages
Log
Reports
Design Runs
DRC
Methodology
Power
Timing
Intra-Clock Paths - clk_100MHz_o_ClkGen - Setup
Intra-Clock Paths
clk_100MHz_o_ClkGen
Setup 0.414 ns (10)
Hold 0.025 ns (10)
Pulse Width 3.663 ns (31)
clk_200MHz_o_ClkGen
clk_i
clk_pll_i
clkfbout_ClkGen
freq_refclk
iserdes_clkdiv
iserdes_clkdiv_1
mem_refclk
Name | Slack | Levels | High Fanout | From | To | Total Delay | Logic Delay | Net Delay | Requirement | Source Clock | Destination Clock
Path 1 | 0.414 | 9 | 119 | Inst_VGA/Inst_...3]__0_rep__0/C | Inst_VGA/Inst_...ummy_reg[0]/D | 8.799 | 1.902 | 6.897 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen
Path 2 | 1.292 | 7 | 8 | Inst_VGA/Inst...eg/CLKBWRCLK | Inst_VGA/mic_red_dly_reg[3]/D | 7.876 | 4.414 | 3.462 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen
Path 3 | 1.985 | 3 | 1 | Inst_VGA/Inst...am/CLKARDCLK | Inst_VGA/logo_red_dly_reg[2]/D | 6.884 | 3.084 | 3.800 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen
Path 4 | 2.103 | 3 | 1 | Inst_VGA/Inst...am/CLKARDCLK | Inst_VGA/logo_...n_dly_reg[1]/D | 6.826 | 3.084 | 3.742 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen
Path 5 | 2.279 | 2 | 35 | Inst_VGA/Inst_L...r_reg_reg[15]/C | Inst_VGA/Inst_...6.ram/ENARDEN | 6.594 | 0.704 | 5.890 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen
Path 6 | 2.300 | 3 | 1 | Inst_VGA/Inst...am/CLKARDCLK | Inst_VGA/logo_red_dly_reg[0]/D | 6.677 | 3.084 | 3.593 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen
Path 7 | 2.314 | 2 | 35 | Inst_VGA/Inst_L...r_reg_reg[15]/C | Inst_VGA/Inst_...6.ram/ENARDEN | 6.605 | 0.704 | 5.901 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen
Path 8 | 2.328 | 3 | 1 | Inst_VGA/Inst...am/CLKARDCLK | Inst_VGA/logo_...n_dly_reg[0]/D | 6.544 | 3.117 | 3.427 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen
Path 9 | 2.366 | 3 | 1 | Inst_VGA/Inst...am/CLKARDCLK | Inst_VGA/logo_red_dly_reg[3]/D | 6.504 | 3.084 | 3.420 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen
Path 10 | 2.369 | 0 | 33 | Inst_VGA/Inst_L...r_reg_reg[7]/C | Inst_VGA/Inst...RARDADDR[10] | 6.378 | 0.456 | 5.922 | 9.286 | clk_100MHz_o_ClkGen | clk_100MHz_o_ClkGen

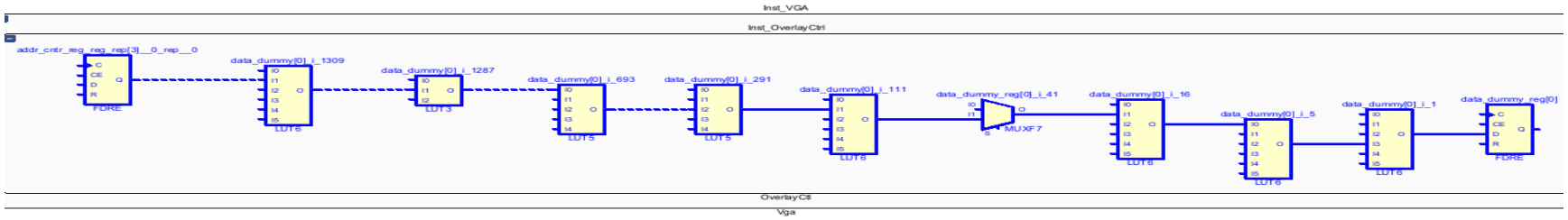
בואו נכיר מספר מושגים

- **SLACK**-הפער שלנו למול הדרישה כל עוד אנחנו מ0 ומעלה מצבנו טוב !
- **LEVELS**-כמות הלוגיקה בן אלמנט סנכרוני אחד למשנהו !
- **TOTAL DELAY-** ההשהייה בפועל
- **NET DELAY-** ההשהייה של ROUTING
- **LOGIC DELAY-** ההשהייה של הלוגיקה

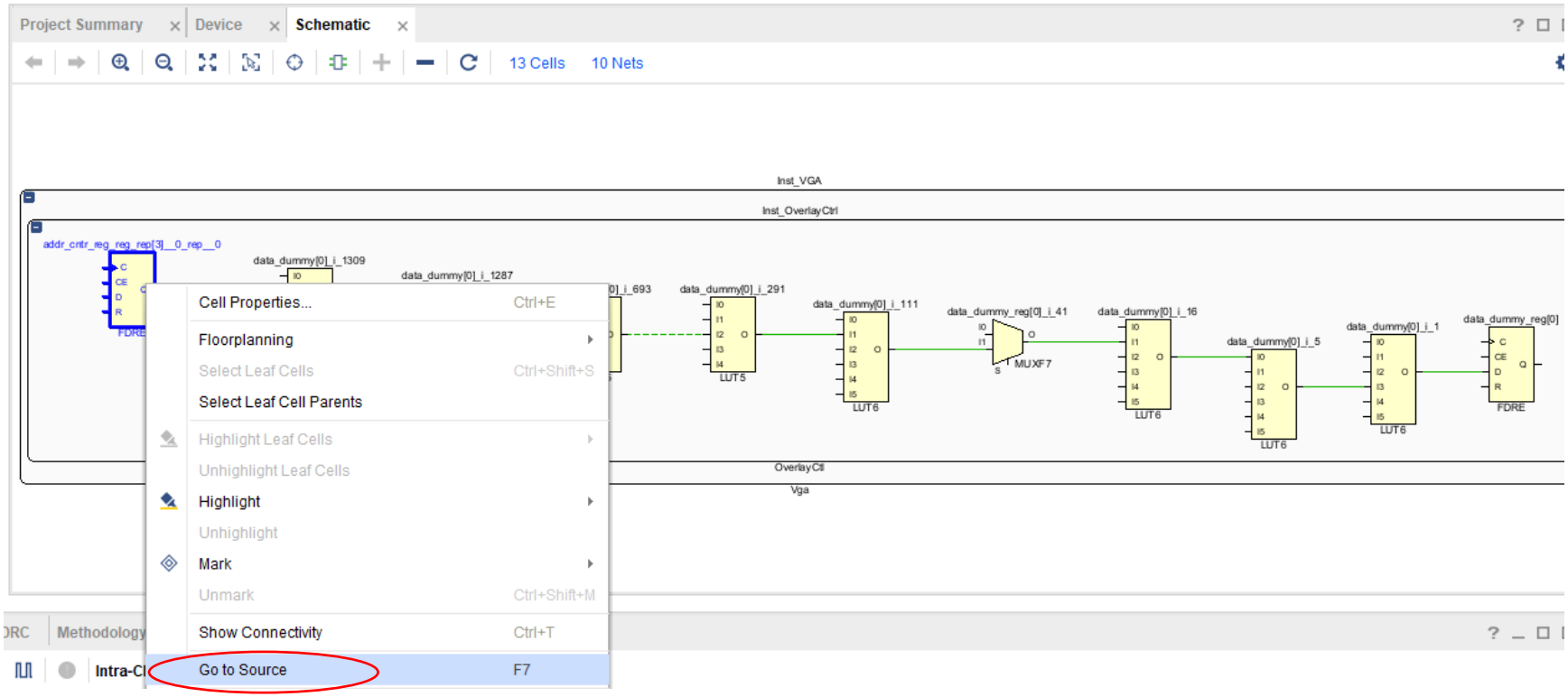


Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception	Clock Uncertainty
Path 1	0.414	9	119	Inst_VGA/Inst_...3]__0_rep__0/C	Inst_VGA/Inst_...ummy_reg[0]/D	8.799	1.902	6.897	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		0.062
Path 2	1.292	7	8	Inst_VGA/Inst..eg/CLKBWRCLK	Inst_VGA/mic_red_dly_reg[3]/D	7.876	4.414	3.462	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		0.062
Path 3	1.985	3	1	Inst_VGA/Inst..am/CLKARDCLK	Inst_VGA/logo_red_dly_reg[2]/D	6.884	3.084	3.800	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		0.062
Path 4	2.103	3	1	Inst_VGA/Inst..am/CLKARDCLK	Inst_VGA/logo_...n_dly_reg[1]/D	6.826	3.084	3.742	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		0.062

8. כעת נסתכל על SCHEMATIC VIEW:



נוכל גם לנווט ולהגיע לקוד עצמו בצורה הבאה



מה החשיבות בן בן חיבור הקוד לשרטוט החשמלי ??

Project Summary | Device | Schematic | **OverlayCtl.vhd**

C:/bh_exp/HW_SOUL/BH_TIMING_NEXSUS_A7/hw/hw.srcs/sources_1/imports/src/hdl/OverlayCtl.vhd

```
34
35  entity OverlayCtl is
36      Port ( CLK_I : in  STD_LOGIC;
37             VSYNC_I : in  STD_LOGIC;
38             ACTIVE_I : in  STD_LOGIC;
39             OVERLAY_O : out  STD_LOGIC
40             );
41  end OverlayCtl;
42
43  architecture Behavioral of OverlayCtl is
44
45  COMPONENT overlay_bram
46    PORT (
47      clka : IN STD_LOGIC;
48      addra : IN STD_LOGIC_VECTOR(20 DOWNTO 0);
49      douta : OUT STD_LOGIC_VECTOR(0 DOWNTO 0)
50    );
51  END COMPONENT;
52
53  type OVERLAY_ARRAY is array (integer range<>) of std_logic_vector(0 downto 0);
54
```

מבט נוסף שניתן להסתכל הוא דרך view path report
שימו לב לשני מושגים שדברנו עליהם במהלך הקורס

a. clock path skew
b. clock uncertainty

מה הם אומרים?

c. כמו שימו לב ל data path delay? מה היחס בן logic לבן route באחוזים? מה המשמעות ?

Project Summary | Device | Schematic | OverlayCtl.vhd | Path 1 - timing_1

Summary

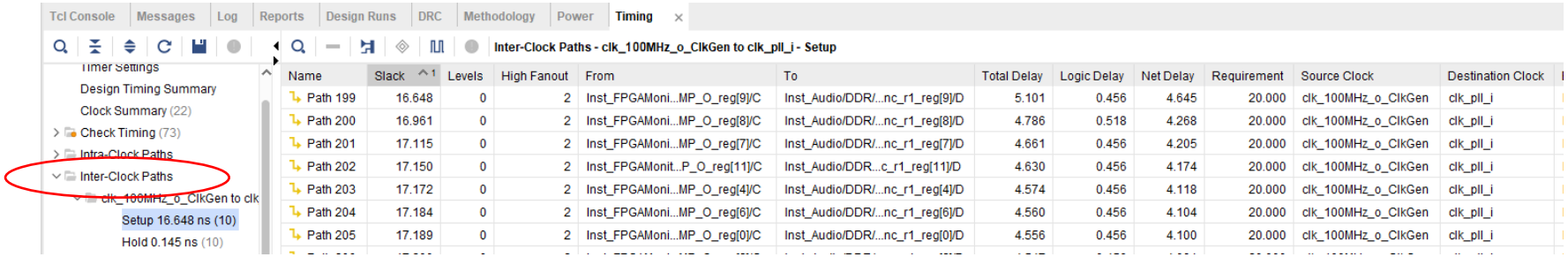
Name	Path 1
Slack	0.414ns
Source	Inst_VGA/Inst_OverlayCtrl/addr_cntr_reg_reg_rep[3]__0_rep__0/C (rising edge-triggered cell FDRE clocked by clk_100MHz_o_ClkGen {rise@0.000ns fall@4.643ns period=9.286ns})
Destination	Inst_VGA/Inst_OverlayCtrl/data_dummy_reg[0]/D (rising edge-triggered cell FDRE clocked by clk_100MHz_o_ClkGen {rise@0.000ns fall@4.643ns period=9.286ns})
Path Group	clk_100MHz_o_ClkGen
Path Type	Setup (Max at Slow Process Corner)
Requirement	9.286ns (clk_100MHz_o_ClkGen rise@9.286ns - clk_100MHz_o_ClkGen rise@0.000ns)
Data Path Delay	8.799ns (logic 1.902ns (21.617%) route 6.897ns (78.383%))
Logic Levels	9 (LUT3=1 LUT5=2 LUT6=5 MUXF7=1)
Clock Path Skew	-0.042ns
Clock Un...rtainty	0.062ns

Source Clock Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock clk_100MHz_o_ClkGen rise edge)	(r) 0.000	0.000		
	(r) 0.000	0.000	Site: E3	clk_i
net (fo=0)	0.000	0.000		Inst_ClkGen/inst/clk_100MHz_i
			Site: E3	Inst_ClkGen/inst/clkin1_ibufg/I
IBUF (Prop_ibuf_I_O)	(r) 1.482	1.482	Site: E3	Inst_ClkGen/inst/clkin1_ibufg/O

כעת נעבור לניתוח המעבר בן אזורי שעון

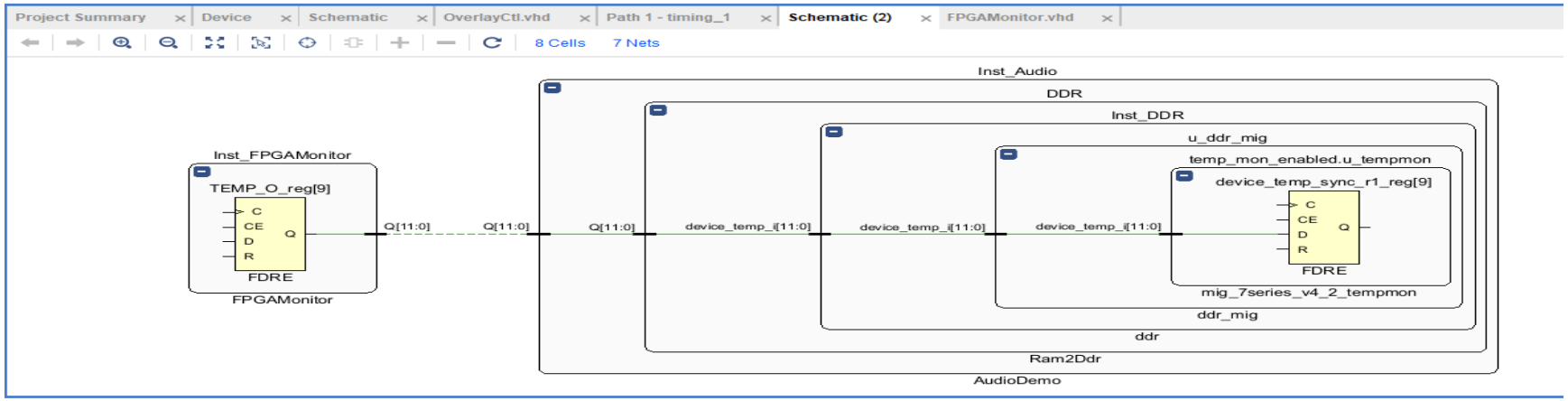
כיצד בכלל ניתן לעבור בן איזורי שעון ?(התאמת פאזה בעזרת PLL)



Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock
Path 199	16.648	0	2	Inst_FPGAMoni...MP_O_reg[9]/C	Inst_Audio/DDR/...nc_r1_reg[9]/D	5.101	0.456	4.645	20.000	clk_100MHz_o_ClkGen	clk_pll_i
Path 200	16.961	0	2	Inst_FPGAMoni...MP_O_reg[8]/C	Inst_Audio/DDR/...nc_r1_reg[8]/D	4.786	0.518	4.268	20.000	clk_100MHz_o_ClkGen	clk_pll_i
Path 201	17.115	0	2	Inst_FPGAMoni...MP_O_reg[7]/C	Inst_Audio/DDR/...nc_r1_reg[7]/D	4.661	0.456	4.205	20.000	clk_100MHz_o_ClkGen	clk_pll_i
Path 202	17.150	0	2	Inst_FPGAMonit...P_O_reg[11]/C	Inst_Audio/DDR...c_r1_reg[11]/D	4.630	0.456	4.174	20.000	clk_100MHz_o_ClkGen	clk_pll_i
Path 203	17.172	0	2	Inst_FPGAMoni...MP_O_reg[4]/C	Inst_Audio/DDR/...nc_r1_reg[4]/D	4.574	0.456	4.118	20.000	clk_100MHz_o_ClkGen	clk_pll_i
Path 204	17.184	0	2	Inst_FPGAMoni...MP_O_reg[6]/C	Inst_Audio/DDR/...nc_r1_reg[6]/D	4.560	0.456	4.104	20.000	clk_100MHz_o_ClkGen	clk_pll_i
Path 205	17.189	0	2	Inst_FPGAMoni...MP_O_reg[0]/C	Inst_Audio/DDR/...nc_r1_reg[0]/D	4.556	0.456	4.100	20.000	clk_100MHz_o_ClkGen	clk_pll_i

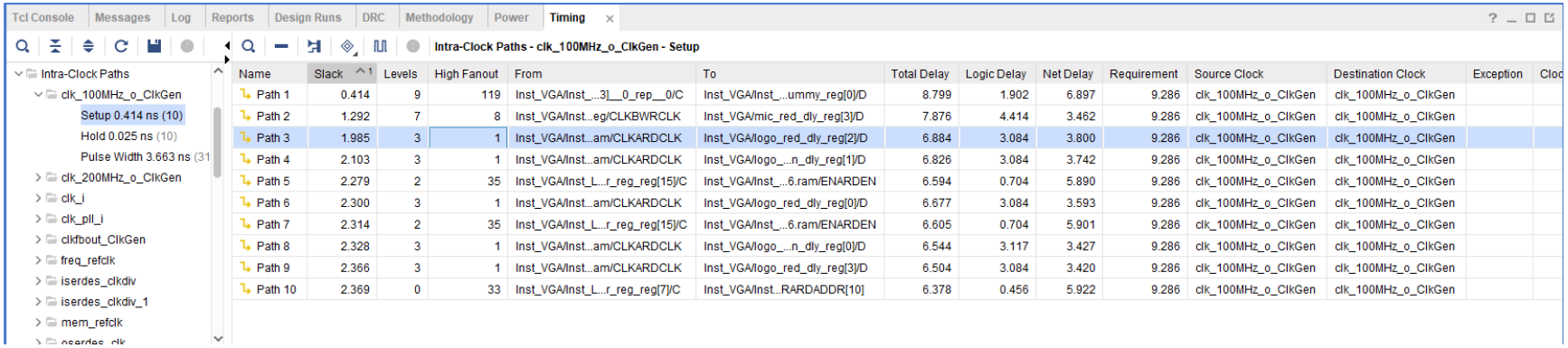
שימו לשרטוט החשמלי

יש אפס רמות לוגיות בן מעבר אחד לשני



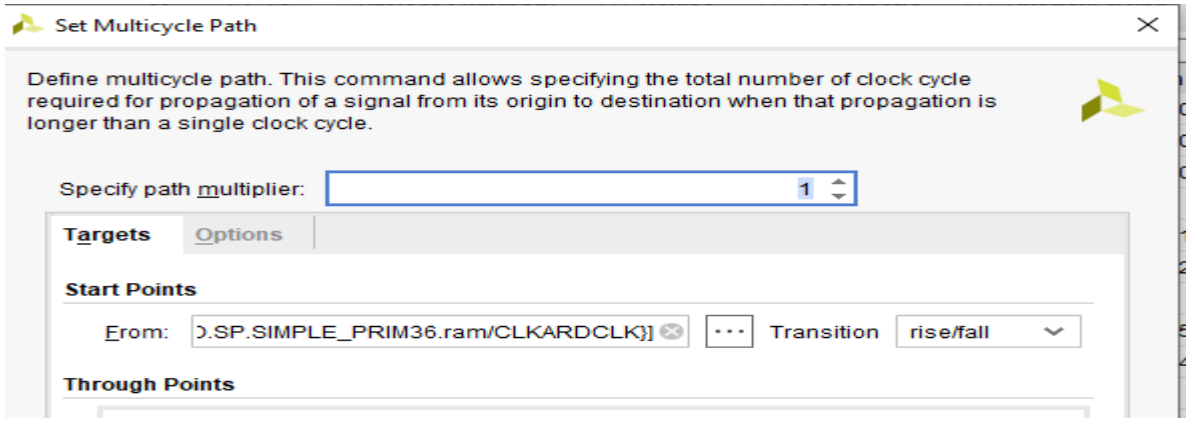
נחזור לאותו מחזור שעון

באם יש לנו CLCOK ENABLE פעיל בתכנון נוכל לעבוד עם multi cycle path

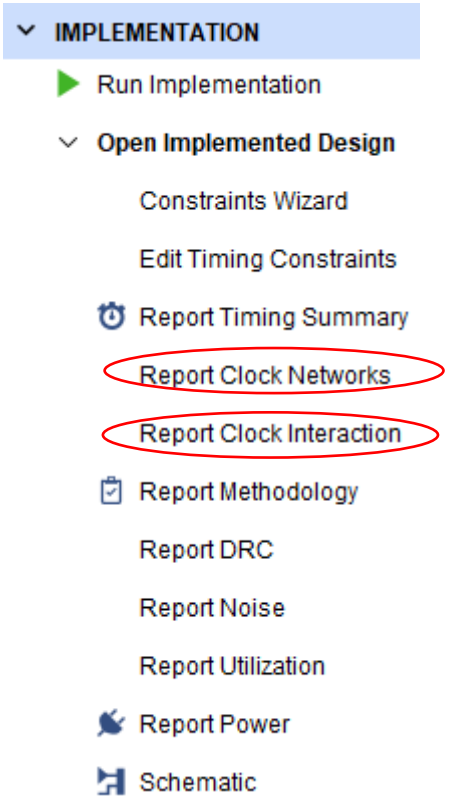


Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception	Cloc
Path 1	0.414	9	119	Inst_VGA/Inst_...3]__0_rep__0/C	Inst_VGA/Inst_...ummy_reg[0]/D	8.799	1.902	6.897	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		
Path 2	1.292	7	8	Inst_VGA/Inst...eg/CLKBWRCLK	Inst_VGA/mic_red_dly_reg[3]/D	7.876	4.414	3.462	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		
Path 3	1.985	3	1	Inst_VGA/Inst...am/CLKARDCLK	Inst_VGA/logo_red_dly_reg[2]/D	6.884	3.084	3.800	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		
Path 4	2.103	3	1	Inst_VGA/Inst...am/CLKARDCLK	Inst_VGA/logo_...n_dly_reg[1]/D	6.826	3.084	3.742	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		
Path 5	2.279	2	35	Inst_VGA/Inst_L...r_reg_reg[15]/C	Inst_VGA/Inst_...6.ram/ENARDEN	6.594	0.704	5.890	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		
Path 6	2.300	3	1	Inst_VGA/Inst...am/CLKARDCLK	Inst_VGA/logo_red_dly_reg[0]/D	6.677	3.084	3.593	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		
Path 7	2.314	2	35	Inst_VGA/Inst_L...r_reg_reg[15]/C	Inst_VGA/Inst_...6.ram/ENARDEN	6.605	0.704	5.901	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		
Path 8	2.328	3	1	Inst_VGA/Inst...am/CLKARDCLK	Inst_VGA/logo_...n_dly_reg[0]/D	6.544	3.117	3.427	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		
Path 9	2.366	3	1	Inst_VGA/Inst...am/CLKARDCLK	Inst_VGA/logo_red_dly_reg[3]/D	6.504	3.084	3.420	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		
Path 10	2.369	0	33	Inst_VGA/Inst_L...r_reg_reg[7]/C	Inst_VGA/Inst..RARDADDR[10]	6.378	0.456	5.922	9.286	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen		

מה משמעות הפעולה הנ"ל ?

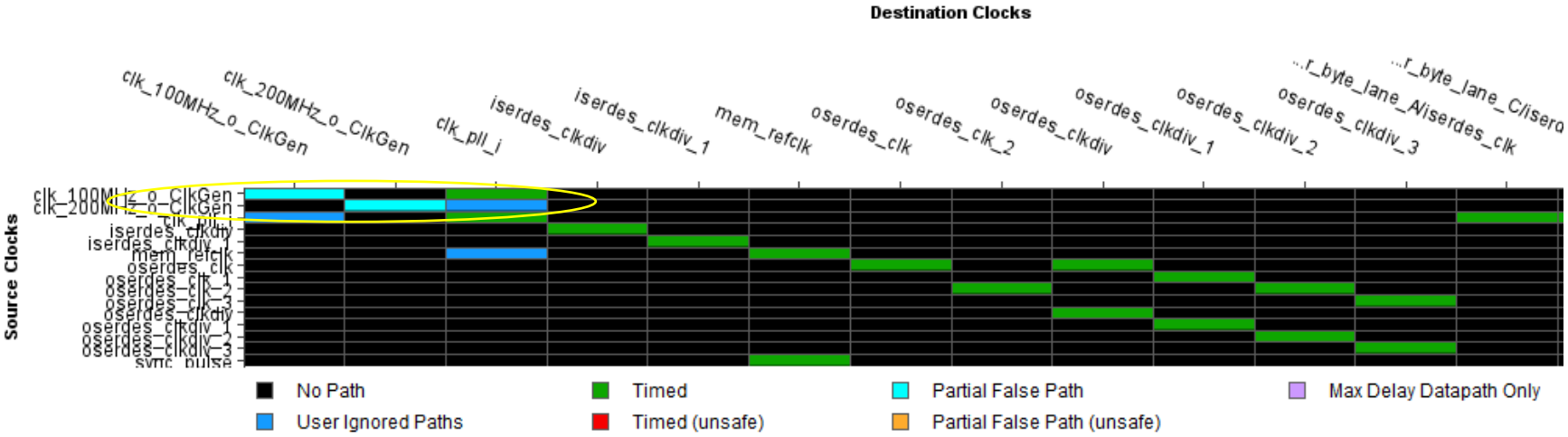


עוד שני מבטים להשלמת התמונה


IMPLEMENTATION
Run Implementation
Open Implemented Design
Constraints Wizard
Edit Timing Constraints
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Noise
Report Utilization
Report Power
Schematic

ראו להלן תוצרי report clock interaction

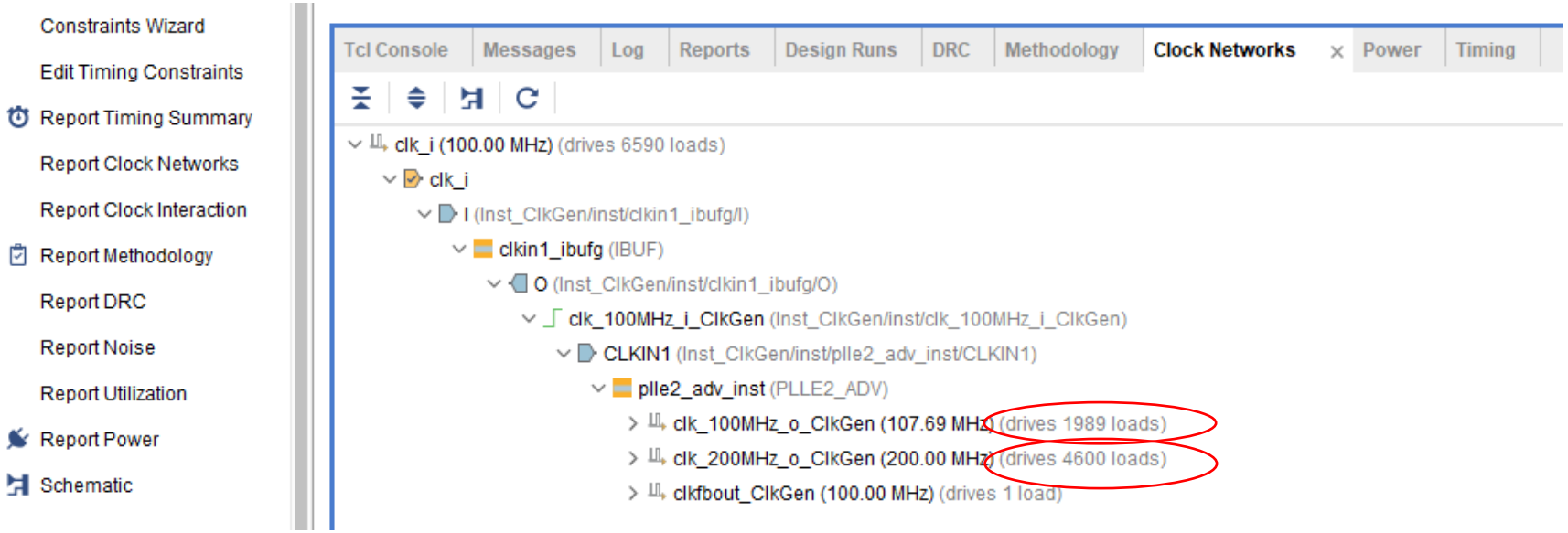
- מה משמעות בציור הנ"ל ?
- האם לכל השעונים ישנם יחסי גומלים אחד עם השני ?



Id	Source Clock	Destination Clock	Edges (WNS)	WNS (ns)	TNS (ns)	Failing Endpoints (TNS)	Total Endpoints (TNS)	Path Req (WNS)	Edges (WHS)	WHS (ns)
1	clk_200MHz_o_ClkGen	clk_200MHz_o_ClkGen	rise - rise	2.988	0.000	0	31	5.000	rise - rise	0.029
2	clk_200MHz_o_ClkGen	clk_pll_i				0	2	Total Endpoints (Total Negative Slack)		
3	clk_100MHz_o_ClkGen	clk_100MHz_o_ClkGen	rise - rise	0.414	0.000	0	4150	9.286	rise - rise	0.025

ראו להלן תוצרי report clock net works

- מי הם השעונים עם מספר LOADS הגדולים ביותר ?
- למה זה חשוב ?



החומרים לשימוש פנימי בלבד אין להשתמש או להעביר ללא רשות מפורשת מניר בלולו