הטכניון - מכון טכנולוגי לישראל
הפקולטה להנדסת חשמל

מעבדה 1

פרויקט סיום
תבנית דוח מסכם

גרסה 1.0

חורף 2018-19

מחברים: אברהם קפלן, דודי בר-און

סטודנט	שם פרטי	שם משפחה
1	אוהד	ואנו
2	עידו	צ'רנינסקי

שם הפרויקט	Bubble Trouble
שם המדריך הקבוע	קובי

תוכן עניינים – פרויקט

Contents

1 נספח מנהלתי

תיאור	תאריך	שם המדריך	הערות ומסקנות
דיון בהגדרת הפרויקט		קובי	
סכמת מלבנים סיפתח		קובי	
סכמת מלבנים PIPE		קובי	
מכונת מצבים של כל הפרויקט		קובי	
הגדרת שני המכלולים העיקריים		קובי	
CODE REVIEW		קובי	
דיונים על בעיות		קובי	

2 הקדמה

2.1 צילום של הפרויקט

2.2 הנחיות כלליות

- מטרת הדוח לתעד בצורה מלאה את פרויקט הסיום שבצעתם.
- יש לכתוב בצורה מלאה וברורה, כך שנתן יהיה בעתיד על סמך קריאת הדוח, להבין את הפרויקט.
- יש לוודא שכל השרטוטים, הסכמות, האיורים, הגרפים, התמונות וכו' ברורים ומובנים. שרטוט מ QUARTUS ע"י : סימון השרטוט, העתק, הדבק, ולא Print-Screen.
- בכל אחד מפרקי הדוח, יש לציין את החלק השייך לתוספת היצירתית.

3 אפיון הפרויקט

3.1 הדרישות המקוריות מהפרויקט -(כמו במצגת)

2.1 הגדרת הדרישות– מינימום לציון 60

כדור אחד, שחקן אחד, טיל יחיד
הכדור יכול להתפצל מקסימום לשניים.

2.2 הגדרת הדרישות– מינימום לציון 80

הכדור יכול להתפצל עד 3 פעמים (ל-8 כדורים)

במידה וחסרו פרטים בהגדרת בפרויקט, הוסף את ההנחות שלך לפיהם פעלת.

מטרת השחקן היא לצבור כמה שיותר נקודות. צבירת הנקודות נעשית על ידי פגיעה בכדורים שמופיעים על המסך. הפגיעה בכדורים מתבצעת באמצעות חבל אשר נורה מראש השחקן ומגיע עד לתקרת המסך במהירות קבועה.

במשחק קיימים 12 שלבים. המעבר בין השלבים יתרחש כאשר השחקן עבר רף מסוים (נקבע מראש) של ניקוד. רמת הקושי במשחק עולה ביחד עם השלבים, כלומר ככל שהשחקן מגיע לשלב מתקדם יותר, כך רמת המשחק תהיה קשה יותר. קושי ברמת המשחק מתבטא בגודל הכדורים וכמותם. (מתחילים בכדור אחד קטן ומסיימים בשלושה כדורים גדולים)

ככל שהכדורים גדולים יותר, כך הם מעניקים יותר נקודות.

כאשר כדור פוגע בשחקן הוא מאבד נקודת חיים.
כאשר מתאפסות לו נקודות החיים הוא מסיים את המשחק והניקוד שהוא צבר מופיע על המסך.

כאשר פוגעים בכדור, יש הסתברות מסוימת (שנקבעה מראש) לקבל מתנה אשר נועדה לעזור לשחקן במהלך המשחק. ישנן מתנות מ4 סוגים שונים אשר מתוארים מטה.

3.2 החלק היצירתי

הדרישות הנוספות מהפרויקט כתוצאה מהחלק היצירתי שהוספת.

מערכת תגמול לשחקן :
בכל פגיעה בכדור, בהסתברות חצי ייפלו מתנות ממיקומים אקראיים אשר יעניקו לשחקן יתרון זמני.

המתנות:

- מהירות כפולה למשך 5 שניות - נעל
- נקודת חיים נוספת - לב
- החבל יישאר על המסך לאחר שיגיע לתקרתו למשך 5 שניות או עד לפגיעה של כדור (חיובי) –חבל אדום
- חסינות לפגיעות הכדורים למשך חמש שניות – מלפפון חמוץ

במקום תמונות של כדורים יופיע תמונות מהסדרה האהובה: Rick & Morty

הטיל הינו חבל רציף. מאפשר אסטרטגית משחק ייחודית – החבל משמיד את הכדורים גם על ידי פגיעה מן הצד.

מערכת שלבים מתוחכמת:

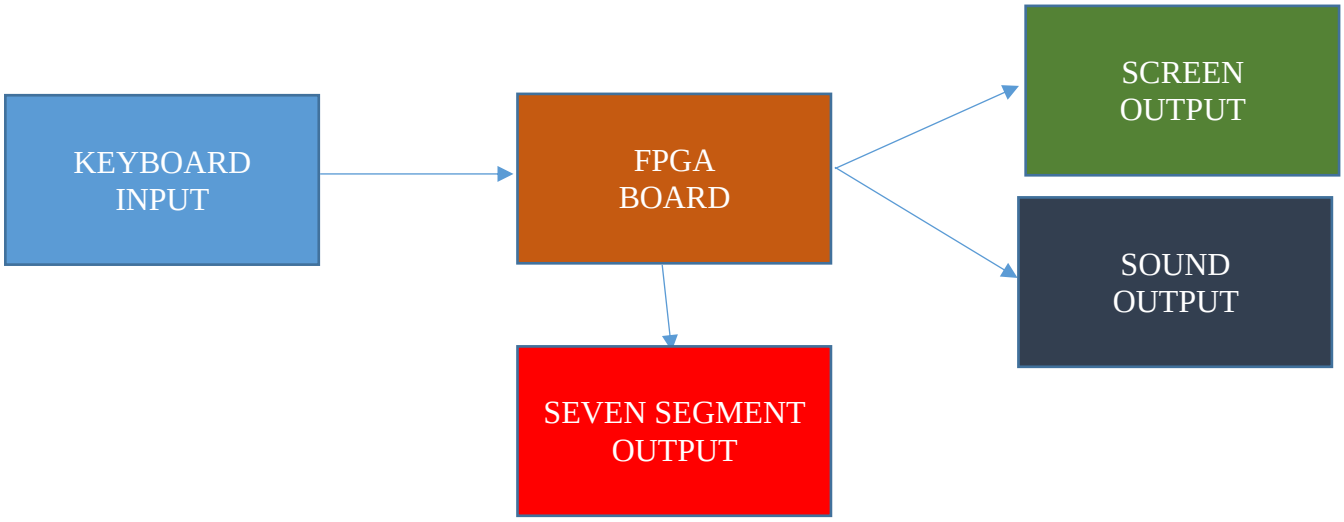
למשחק 12 שלבים אשר מציינים רמות קושי שונות. כפי שצוין קודם המעבר בין השלבים מתרחש כאשר מגיעים לרף נקודות מתאים.

הקושי ממוש על ידי עלייה בכמות הכדורים ובהגדלתם.

4 ארכיטקטורה

היחידות מהן בנוי הפרויקט (כרטיסים, אמצעי קלט/פלט וכו') וזרימת הנתונים דרכן.

שרטוט המבנה והסבר תפקידה של כל יחידה. – *העזר ברכיבים מהמצגת ואל תגיש שרטוט בעפרון*



4.1 תפקיד היחידות:

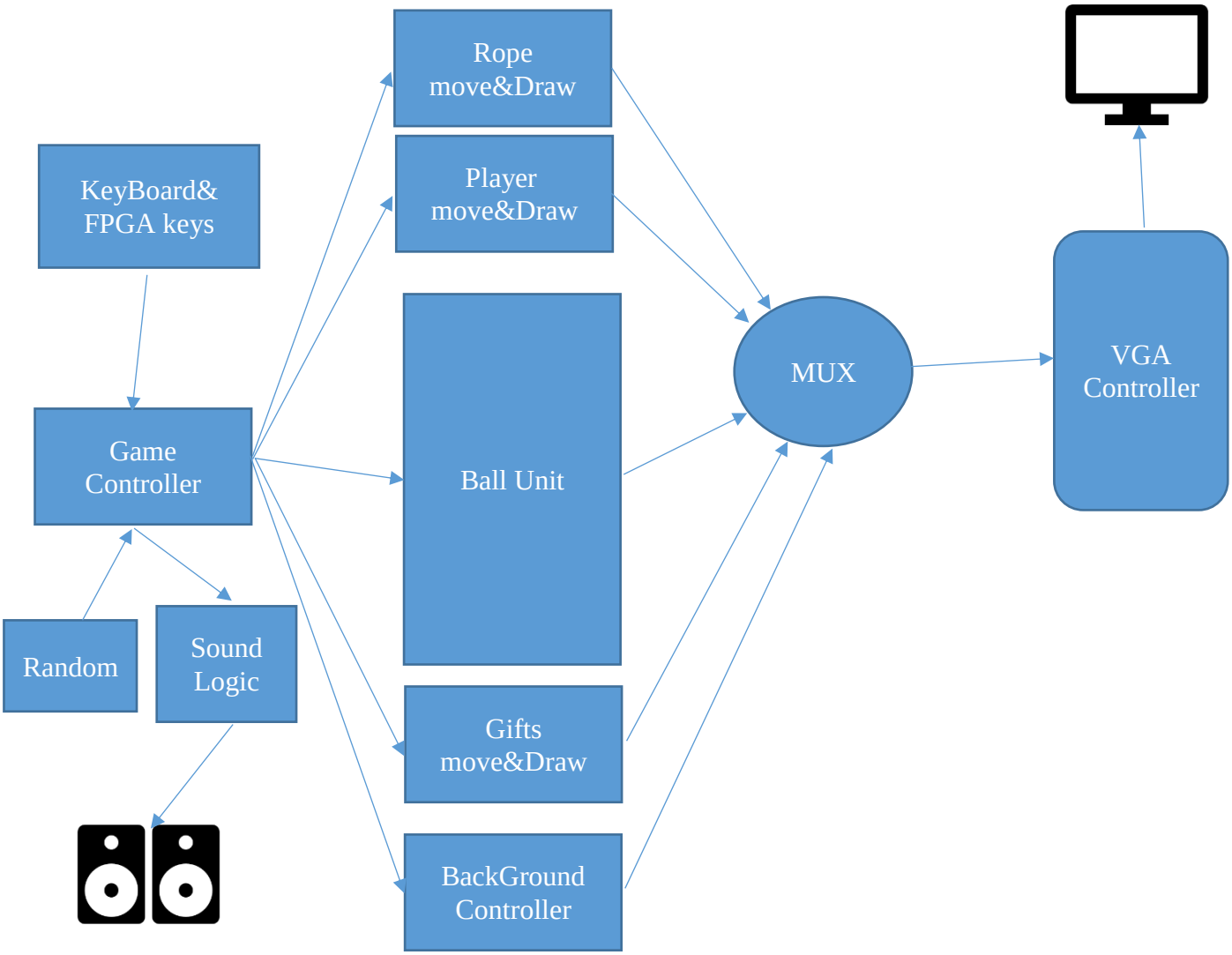
שם	תקציר פעולתה
כרטיס DE10	רכיב מתוכנת אשר מריץ את הקוד המתוכנן. מקבל אות מן המקלדת ומוציא אות אל המסך.
מסך מחשב	מציג את המשחק. קיים בו רכיב הסורק את שורת הפיקסלים במסך וצובע אותן. תפקיד ה-fpga הינו לתקשר

	עם רכיב זה ולהכתיב לו כיצד נדרש לצבוע את הפיקסלים.
מקלדת	אמצעי הקלט במשחק. מוציא שני אותות אשר מחוברים לfpga : KBD_DAT – מוציא את קוד הלחצן שנלחץ. KBD_CLK – מוציא את שעון המקלדת.
לחצנים של הfpga	נועדו לשמש כתחלופה למקלדת בתור קלט למשחק.
רכיב שמע	צ׳יפ הצרוב בfpga. מקבל אות סינוסי ומוציא צליל מתאים.
Seven Segment	נורות אשר יציגו כפלט את הניקוד של המשחק הנוכחי

5 סכמת מלבנים פנימית

חלוקת הפרויקט למודולים פונקציונליים והקשרים ביניהם.

שרטוט ***סכמת המלבנים הכללית (VISIO או PPT)***



רשימת מכלולים (מלבנים) עיקריים, תפקידם וסדר ביצועם

פרט בטבלה את כל המכלולים העיקריים. פחות מעשרה

רצוי להתחיל עם ליבת הפרויקט (החלק הקשה/הארוך/המסובך של הפרויקט)

- בתפקיד מנוון רשום מה תעשה לפתיחת ה- PIPE
- לכל יחידה פרט את הסיבוכיות שתידרש לדעתך למימושה (קל בינוני כבד) \
- החלט מהו סדר המימוש שבחרת

מודול מס	שם	תפקיד	תפקיד מנוון PIPE	סיבוכיות התכן	סדר ביצוע
1	Player Unit	אחראי על קבלת נתונים מהבקר של המשחק (Game controller) ומוציא פלט של תנועת השחקן	קולט מידע מהבקר כמו מקשי מקלדת שנלחצו ומידע מרכיב ה VGA ומוציא כפלט את מיקום השחקן.	קל	שלב ראשוני
2	VGA Unit	אחראי להדפיס למסך את הצבעים המתאימים בכל נתון אשר מקבל מן הMUX	מוציא כקלט את מיקום הפיקסל שצריך להיצבע ברגע נתון ומקבל מן הMUX צבע מתאים לצביעה עבור פיקסל זה.	קל	שלב ראשוני
3	Keyboard Unit	אחראי על קבלת קלט מהמקלדת הפיזית והוצאת סיגנלים המשמשים לעיבוד המידע	קולטת קלט מהמקלדת ע"י אותות שעון ו DATA ומוציאה כפלט סיגנלים לגבי המקש שנלחץ	קל	שלב ראשוני
4	Presents Unit	אחראי על יצירת המתנות, על הצגתן ועל תנועתן.	מקבל כקלט סיגנל ממנהל המשחק אשר מציין כי יש ליצור מתנה חדשה ומוציא כפלט בקשות ציור וצבע מתאים.	קשה	שלב סופי
5	Sound Unit	אחראי על יצירת סאונד מתאים כתלוי באירוע המשחק שהתרחש.	מקבל כקלט אותות שמע על סמך פעולות במהלך המשחק ומוציא כפלט אות אשר מתורגם לאות אנלוגי אשר מומר לשמע	קל	שלב סופי
6	Game Controller	הבקר של המשחק, מכונת המצבים. אחראי על קבלת מידע מכל הרכיבים במשחק	מקבל כקלט מידע מהאובייקטים השונים, ומהמקלדת ומוציא כפלט סיגנלים אשר מיועדים לגרום	קשה	שלב ראשוני

		ולנהל את התקדמות המשחק תוך שמירה על החוקיות כפי שהוגדרה במטרת המשחק ובאופן המימוש.	לאובייקטים השונים לתפקד בסינכרוניזציה על פי הגדרות המשחק.		
7	Rope Unit	אחראי על תצוגת החבל אשר השחקן יורה במהלך המשחק	מקבל כקלט סיגנל מהבקר אשר מורה על שליחת החבל ומוציא כפלט סיגנל צבע ובקשת ציור לרכיב ה MUX	בינוני	שלב ביניים
8	Balls Unit	אחראי על יצירת כדור חדש עם תנאי התחלה מתאימים (מיקום מהירות וגודל) על פי היציאות המתאימות ממנהל המשחק. בעת פגיעת החבל בכדור, הרכיב אחרי על פיצולו אלא אם הוא כבר בגודל המינימלי. אז הכדור ייעלם מהמסך.	מקבל כקלט נתונים מהבקר לגבי מהלך המשחק והתקדמותו ומוציא כפלט את בקשת הצביעה והצבע המתאים.	קשה	שלב ביניים

5.1 פרוט ארבעת המודולים העיקריים

רשום תת פרק לכל מודול אותו תתכננו (לא לבחור מודול שולי כמו ה MUX

5.1.1 [Player Unit]

תפקיד מפורט	תפקידו להדפיס למסך את מיקום השחקן בכל זמן נתון במהלך המשחק.
מימוש מצומצם (PIPE)	מקבל כקלט כניסות אשר מציינות אינטראקציה עם המשתמש (דרך המקלדת) ומוציא את בקשת הצביעה ואת הצבע המתאים.
אופן המימוש	מורכב משלושה תתי רכיבים עיקריים: • playerMove – אחראי על ההזזה של השחקן במהלך המשחק. מקבל כקלט את הסיגנלים מהמקלדת ומוציא כפלט את מיקום הפינה השמאלית העליונה של אובייקט השחקן לאחר עיבוד המידע מהמקלדת • square_object – מקבל מה-VGA Controller את מיקום הפיקסל אשר הוא צובע ברגע זה. מוציא אות בודד האם פיקסל זה נמצא בגבולות האובייקט. • playerBitMap – מקבל את היציאה של ה-square_object ואחרי להוציא את הצבע הרלוונטי לצביעה וביחד עם זאת בקשת צביעה.
כניסות עיקריות	הכניסות למודול: • אות שעון מחזורי

	• אות reset • Start of frame – מציין בקשת מידע מרכיב ה VGA • playerMoveRight – סיגנל מהבקר אשר מציין שהשחקן מתבקש לזוז ימינה • playerMoveLeft – סיגנל מהבקר אשר מציין שהשחקן מתבקש לזוז שמאלה. • pixelX/pixelY – הפיקסל המבוקש על ידי רכיב ה VGA.
יציאות עיקריות	היציאות מהמודול: • playerDrawRequest – מציין שהמודול מבקש לצבוע את הפיקסל הנוכחי • playerRGBout – קוד הצבע אשר הפיקסל ייצבע בו.

5.1.2 [Game Controller]

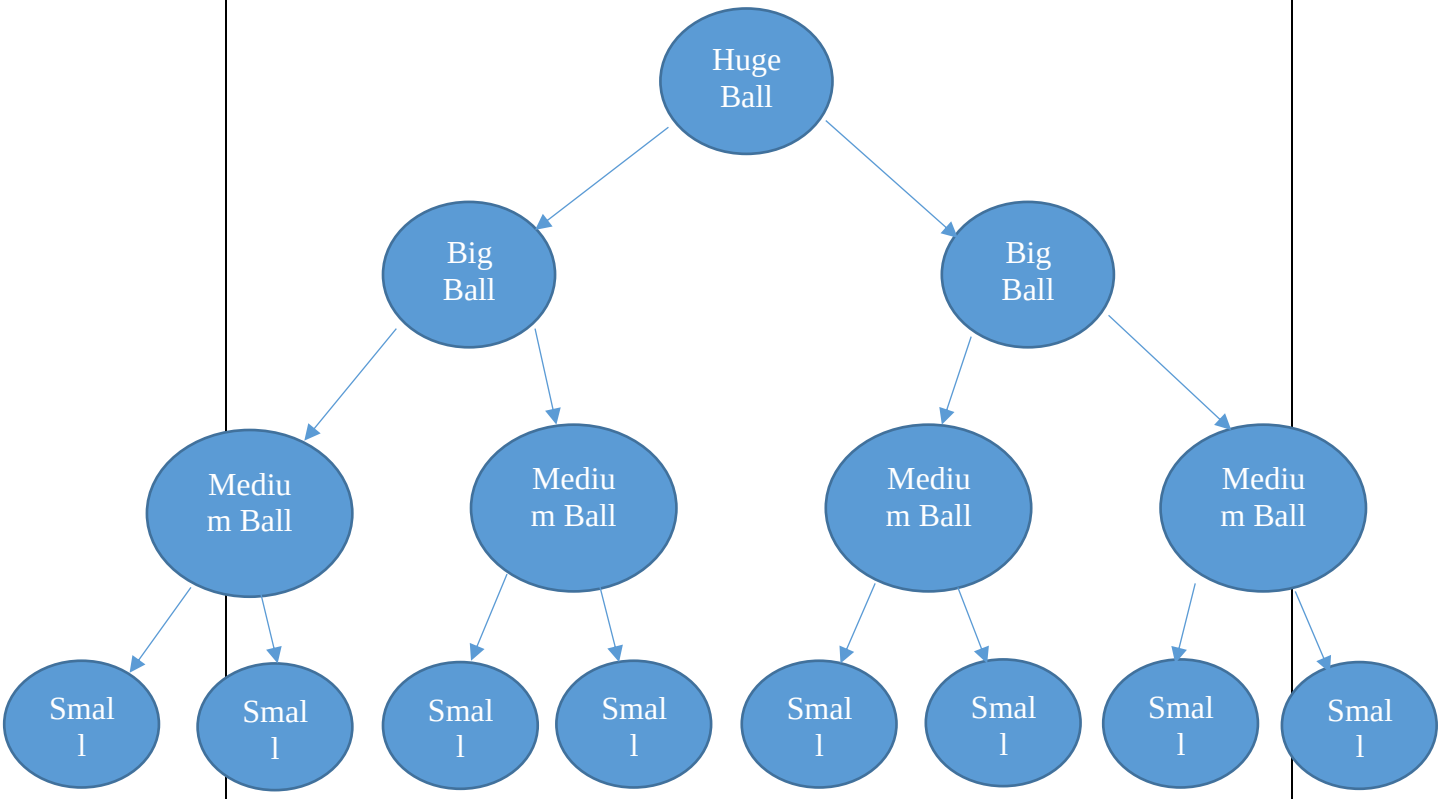
תפקיד מפורט	תפקידו לשלוט באובייקטים של המשחק, מוציא מידע בעזרת סיגנלים אשר מנהלים את מהלך המשחק על פי החוקיות שהוגדרה מראש.
מימוש מצומצם (PIPE)	מקבל כקלט מידע מהאובייקטים השונים כמו מיקום כדורים, מיקום השחקן, לחיצות מקלדת וכו' ומוציא כפלט אותות אשר משמשים כהוראות הפעלה לאובייקטים השונים. ההוראות כוללות בין היתר תנועת אובייקטים, הסתרת אובייקטים, שינוי מצב המשחק וכו'.
אופן המימוש	• Collision detector: אחראי לרכז למנהל המשחק אילו התרחשויות קרו ברגע צביעת הפיקסל. • gameStateMachine: הלב של מנהל המשחק. זה הרכיב שאחראי על הלוגיקה המוציאה את הסיגנלים אשר "שולטים" באובייקטים השונים במשחק בהתאם למצבו (מסך פתיחה, מהלך המשחק וסוף משחק). למשל: בעת פגיעת החבל בכדור רכיב זה אחראי להוציא בקשה ליצירת מתנה ממודול המתנות (בהסתברות המתאימה לפי הגדרת המשחק).
כניסות עיקריות	• כניסות מקשי מקלדת • התנגשויות אשר חושבו במודולים חיצוניים (מתנות וכדורים) • מיקום ראש החבל • אות שעון ואות reset
יציאות עיקריות	• מצב המשחק: [מסך פתיחה/ריצת המשחק/סיום המשחק] • סיגנלים אשר מציינים הוראות תזוזה לשחקן • סיגנל אשר מציין שליחת חבל על ידי השחקן • סיגנל אשר יוצר מתנות • סיגנל אשר יוצר כדור חדש עם תנאי התחלה מתאימים. • סיגנלים המתארים את היכולות אשר נתנו המתנות במשחק • סיגנל עבור החיים שנותרו לשחקן • סיגנל עבור הנקודות שצבר השחקן

5.1.3 [Rope Unit]

תפקיד מפורט	תפקידו להדפיס למסך את החבל שהשחקן יורה ולאותת על מיקום החבל בכל זמן נתון למנהל המשחק
מימוש מצומצם (PIPE)	מקבל כקלט את הפיקסל מרכיב ה VGA וסיגנל ממנהל המשחק אשר מציין את שליחת החבל. מוציא כפלט את הצבע של הפיקסל המבוקש וכך יוצר את תמונת החבל.
אופן המימוש	מורכב משלושה תתי רכיבים עיקריים: • ropeMove – אחראי על הזזת החבל לתקרת המסך • square_object מקבל מה-VGA Controller את מיקום הפיקסל אשר הוא צובע ברגע זה. מוציא אות בודד האם פיקסל זה נמצא בגבולות האובייקט. • ropeBitMap – אחראי על הוצאת בקשת צביעה וסיגנל צבע מתאים.
כניסות עיקריות	• אות שעון מחזורי • אות reset • Start of frame – מציין בקשת מידע מרכיב ה VGA • pixelX/pixelY – הפיקסל המבוקש על ידי רכיב ה VGA. • ropeActive – סיגנל המסמל לחבל כי הוא פעיל.
יציאות עיקריות	• ropeDrawRequest – מציין שפיקסל כלשהו במסך מבקש להיצבע על ידי המודול • ropeRGBout – קוד הצבע אשר הפיקסל ייבצע בו.

5.1.4 [Balls Unit]

תפקיד מפורט	תפקידו להדפיס למסך את מיקומי הכדורים "האויבים". הכדורים נוצרים באופן רנדומלי בהתאם להתקדמות המשחק ונעלמים או קטנים ברגע שהחבל פוגע בהם
מימוש מצומצם (PIPE)	הרכיב מקבל כקלט סיגנלים מהבקר אשר מציינים שיש ליצור כדורים עם תנאי התחלה מתאימים. הרכיב ישדר לבקר האם הכדור ביצע התנגשות עם האובייקטים במשחק והאם הכדור הנוכחי בשימוש. בנוסף, ישדר לרכיב ה VGA את צבעי הפיקסלים המבוקשים להדפסה למסך.
אופן המימוש	היחידה מורכבת מתתי רכיבים : • 15 מודולים של כדורים –כדור אחד ענק, שניים גדולים, ארבעה בינוניים ושמונה קטנים (כל כדור ממומש בדומה לשאר האובייקטים במשחק מלבד למודול ה-move). • Ball controller – אחראי לנהל את היררכיית הכדורים כפי שמתוארת מטה (אחראי לאותות הנראות של כל אחד מן הכדורים). • Collision detector – אחראי לזהות התנגשויות בין כל אחד מ15 הכדורים לבין השחקן והחבל. • Ball mux – משמש כמעין MUX ביניים , תפקידו לרכז את כל בקשות הצביעה של הכדורים לבקשה אחת וצבע אחד אשר ילכו לMUX הראשי.



כניסות עיקריות	• אות שעון מחזורי • אות reset • Start of frame – מציין בקשת מידע מרכיב ה VGA • pixelX/pixelY – הפיקסל המבוקש על ידי רכיב ה VGA. • סיגנל שמציין יצירת כדור חדש • תנאי התחלה כפי שתוארו מעלה
יציאות עיקריות	• ballDrawRequest – מציין שפיקסל כלשהו במסך מבקש להיצבע על ידי המודול. • ballRGBout – קוד הצבע אשר הפיקסל ייבצע בו. • התנגשויות עם האובייקטים במשחק • Col_ball_type – סיגנל המציין את גודל הכדור איתו התנגש החבל.

5.2 בחירת המודולים למצגת סופית

מודול	Game Controller
סטודנט	עידו
למה הוא חשוב	אחראי על בקרת המשחק בכל רבדיה
מה נציג	בקרת המתנות

מודול	Ball duplicate
סטודנט	אוהד
למה הוא חשוב	אחראי על לוגיקת שכפול הכדורים
מה נציג	כיצד מימשנו את דרישה זו

6 שלבים במימוש הפרויקט

בגלל המורכבות של הפרויקט יחסית למה שתכננתם עד היום, וכדי שהפיתוח יעשה בצורה חלקה, ביצוע הפרויקט נעשה בשלושה שלבים, מהקל לכבד.

1. סיפתח – ביצוע פריט אחד או שניים הקשורים לממשקים של הפרויקט: תצוגה על מסך VGA וצליל.
2. PIPE – ביצוע מסלול שלם ומנוון של הפרויקט הדורש שיתוף מכלולים עיקריים שלו.
3. הפרויקט השלם.

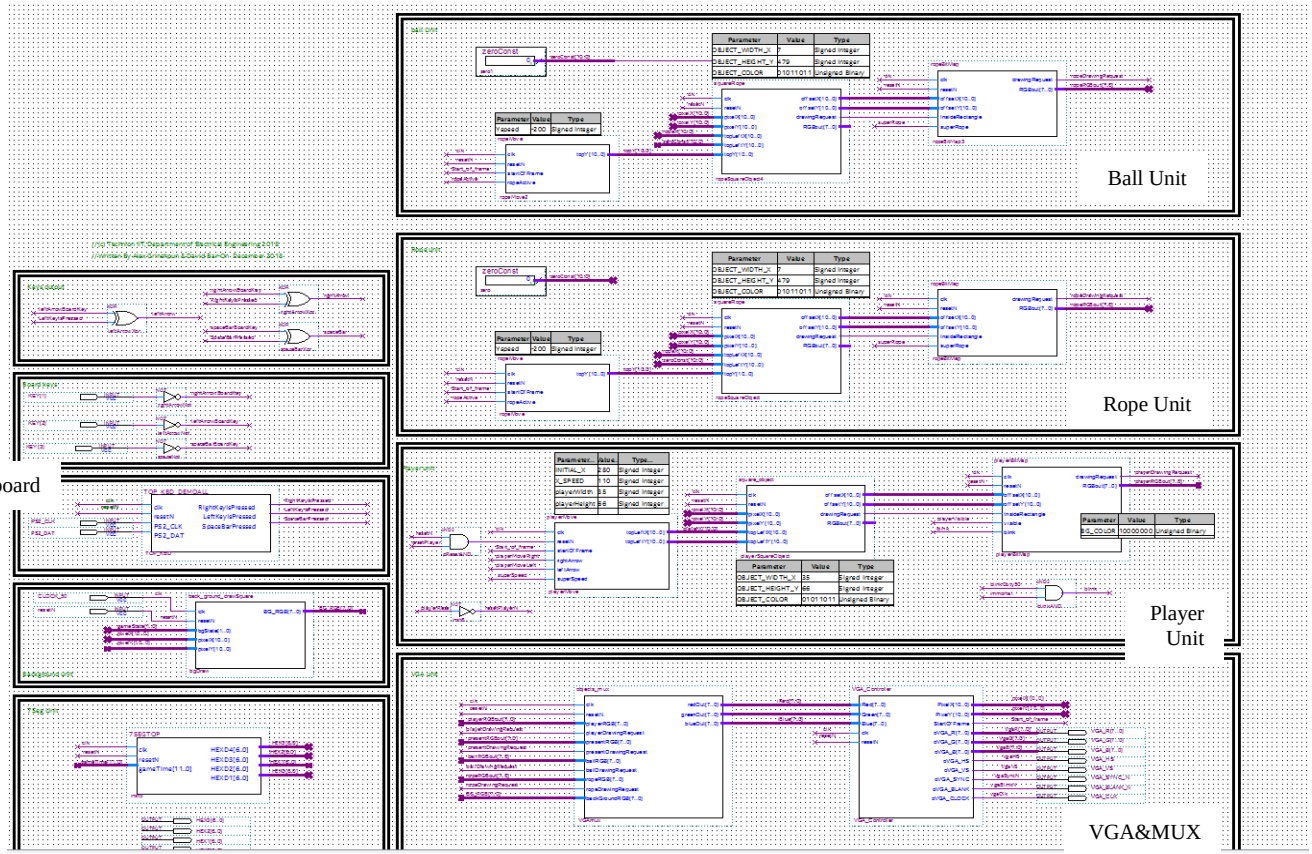
חובה לבצע את כל השלבים בסדר שלמעלה וכל שלב יש לו חלק בציון על הפרויקט.
כל שלב הוא חלק מדוח הכנה בהתאם ללו"ז המופיע במודל.

6.1 סיפתח

לאחר המימוש העתק סכמת ה TOP לכאן

על מנת להתחמם, מימשנו שחקן שיכול לנוע ימינה ושמאלה וכדור שמבצע תנועה פרבולית כנדרש.
בעת לחיצה על אחד המקשים, אות יעבור מן הממשק מקלדת אל רכיב תנועת השחקן ויסמן לו שהוא צריך לשנות את מיקומו בהתאם. רכיב התנועה, אשר אחראי על הזזת הפינה השמאלית העליונה של השחקן, יזיז את השחקן במהירות קבועה כל עוד מקש התנועה לחוץ.

מהירות הכדור תהיה קבועה בציר האופקי ועם תאוצה בציר האנכי.



6.2 פתיחת PIPE

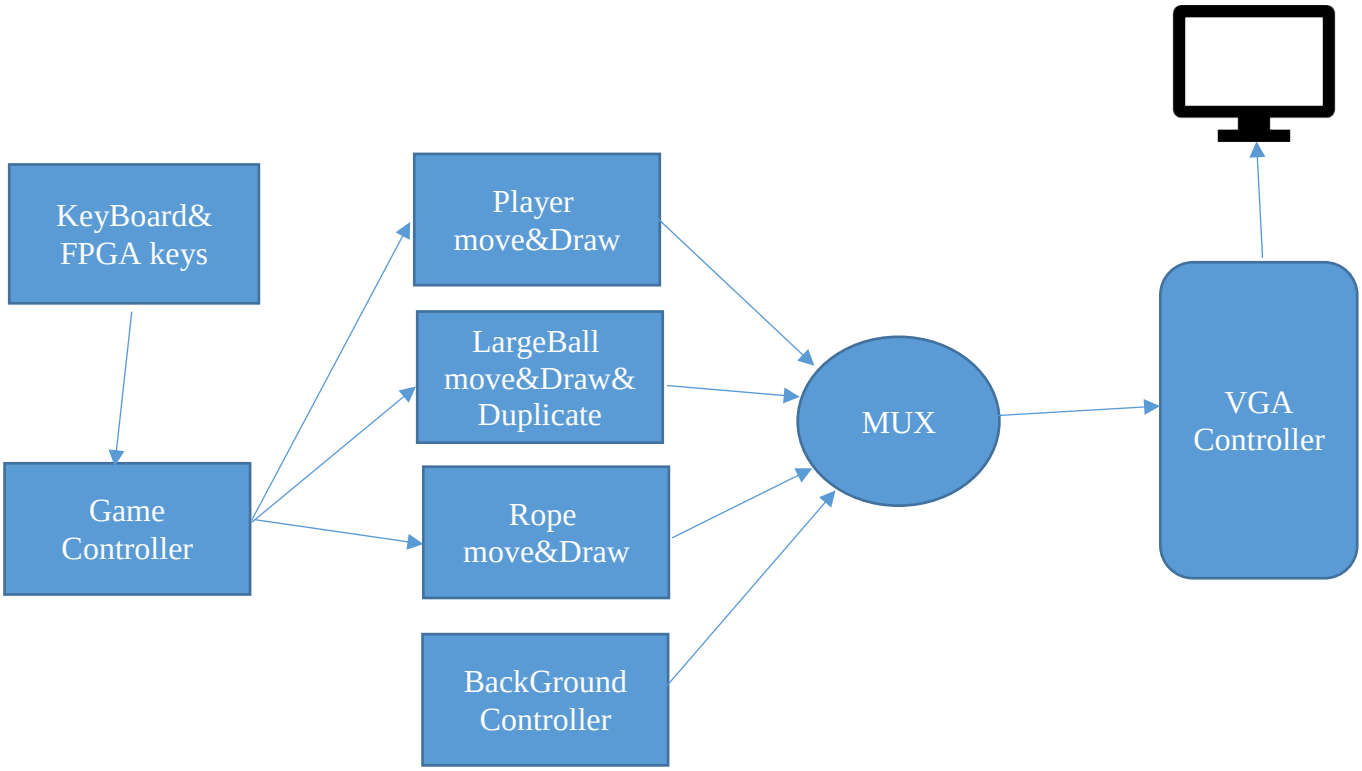
תאר מה יעשה ה PIPE,

כעת נרצה שיהיה מנהל משחק, אשר כל אותות המשחק הרלוונטיים יעברו דרכו, שתפקידו יהיה לנהל את הרכיבים האחרים כנדרש באופן מסודר.
לדוגמא: כאשר יש התנגשות בין כדור לשחקן. נרצה שמנהל המשחק יאפס את מיקום השחקן למיקומו ההתחלתי אשר נקבע על ידי רכיב תנועתו.

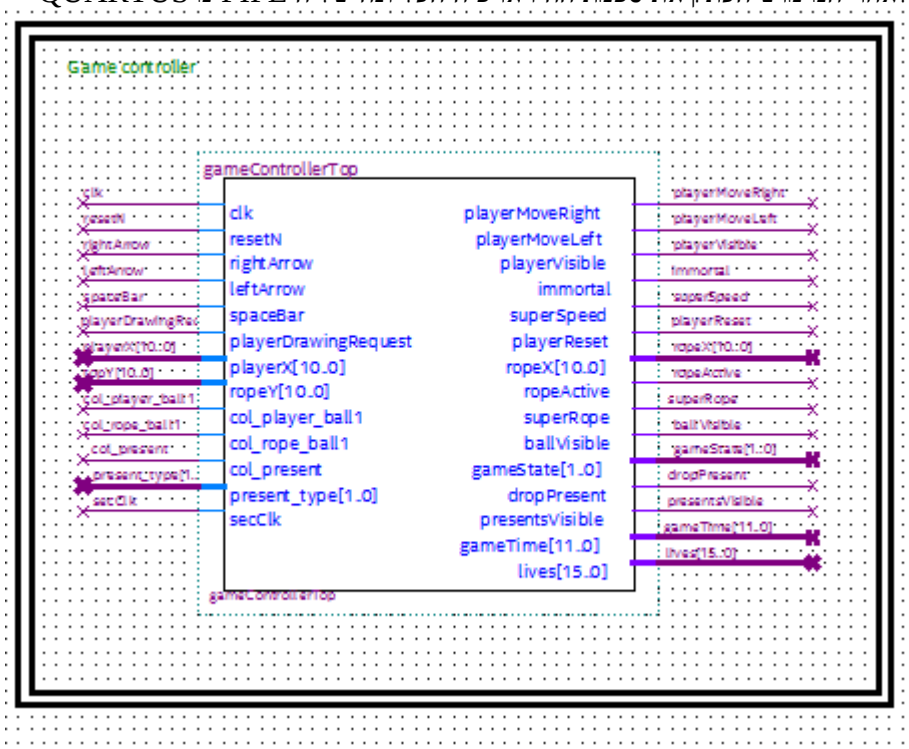
בנוסף, נוסיף חבל אשר יוכל להעלים את הכדור כאשר יפגע בו.

הערה:
הוספנו רק את מנהל המשחק בתמונה מטה. כל שאר המודולים הרלוונטיים נמצאים למעלה.

העתק לכאן את סכמת המלבנים הכללית וסמן עליה את המכלולים המשתתפים בביצוע ה PIPE



לאחר המימוש העתק את סכמת ההיראכיה העליונה של ה PIPE מ QUARTUS



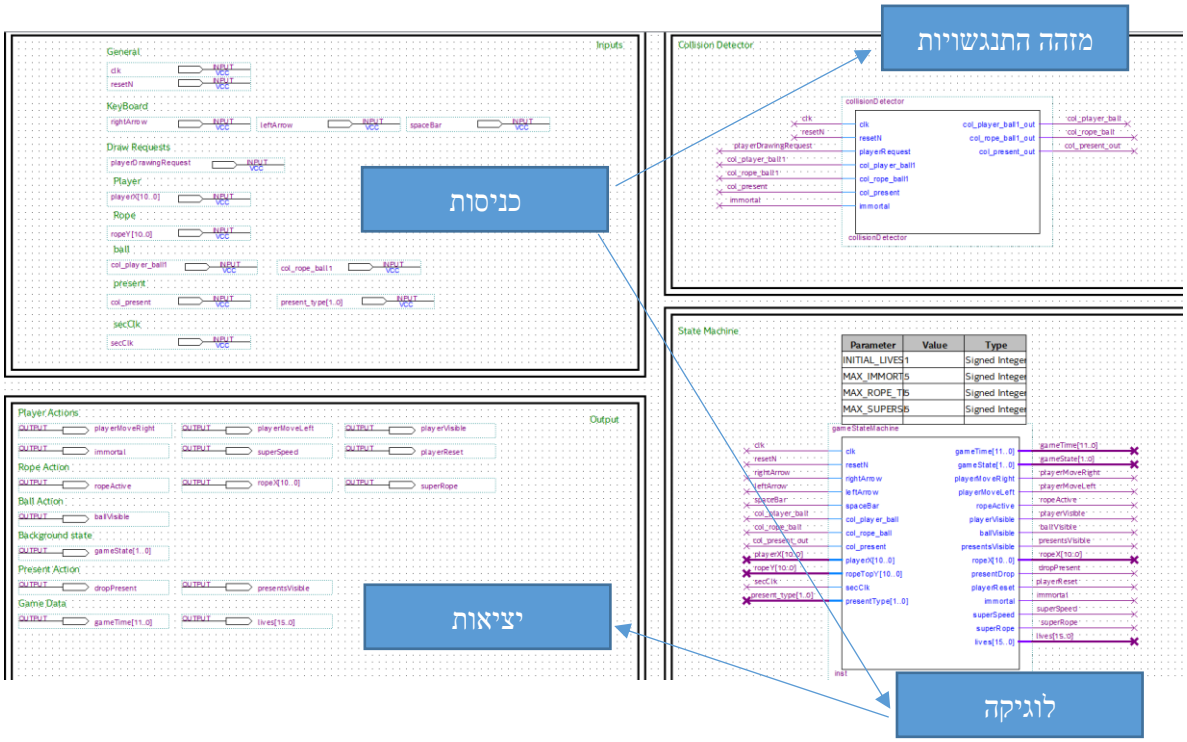
7 תיאור מפורט של שני מודולים -(כמו במצגת)

שימו לב שיש להקפיד לשים מודול אחד לכל סטודנט- (שיהיה תכנון שלו ועליו הוא יסביר)
יש לקחת מודולים מסובכים, רצוי כאלה המכילים המכילים מכונת מצבים , ולא קוד טרוויאלי
לכל מודול יש לבצע את הסעיפים שלהלן.

7.1 [Game Controller] - [עידו]

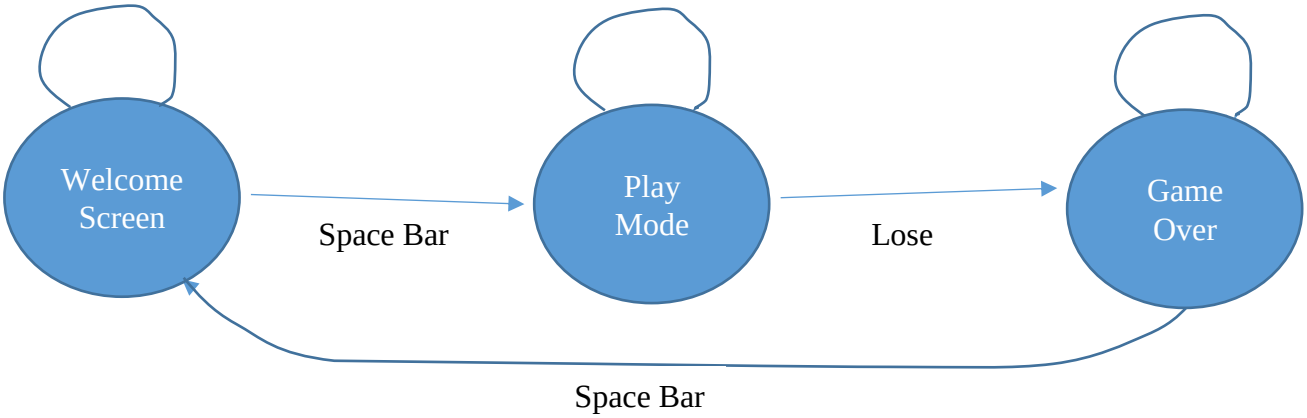
7.1.1 דיאגרמת מלבנים (תהליכים)

תאר את המודול כתהליך אחד או יותר.



7.1.2 דיאגרמת מצבים bubble diagram

לתהליכים אותם מימשת בעזרת מכונת מצבים, צייר את דיאגרמת המצבים



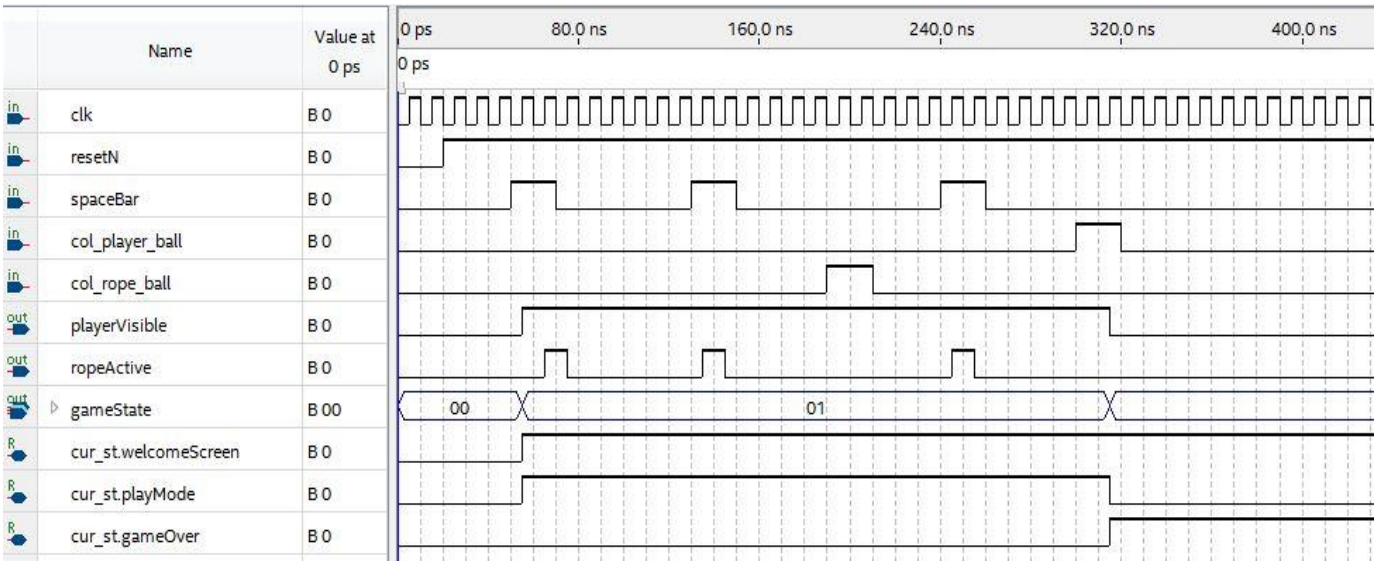
7.1.3 פרט את המצבים העיקריים –

שם המצב	פעילות עיקרית	לאיזה מצב עוברים מהמצב הנוכחי ובאילו תנאים
Welcome screen	מציגים מסך פתיחה עם ניקוד מקסימלי שהתקבל וממתינים להתחלת המשחק	כאשר השחקן לוחץ על מקש רווח עוברים ל play mode.
Play mode	המשחק רץ על המסך והטיימר של המשחק מאופס, נקודות החיים מאותחלות לערך שנקבע מראש. השחקן יכול לזוז לצדדים, לירות חבל, לפגוע בכדורים ולקחת מתנות.	כאשר נקודות החיים מגיעות ל – 0 עוברים למצב gameOver.
Game over	מציגים על המסך את הניקוד שנצבר במהלך המשחק האחרון.	כאשר השחקן לוחץ על מקש רווח עוברים ל – welcome screen.

7.1.4 מסך(י) סימולציה

יש לבדוק את כל הכניסות והיציאות, כל מקרי הקצה וכל המקרים המיוחדים.
אם יש צורך, הצג את תוצאות הסימולציה במספר חלונות. מעל כל חלון כתוב מה הוא בודק. סמן בעזרת חיצים על דיאגרמת הזמנים, את מקום הבדיקה.
וודא שבחלון הסימולציה רואים את רשימת האותות ואת ציר הזמן.

ניתן לראות בסימולציה שמצב המשחק עובר ממסך הפתיחה ל – play mode בעת לחיצה על מקש רווח והשחקן פעיל. לאחר מכן, לחיצה נוספת על מקש רווח משגרת את החבל בעזרת סיגנל ropeActive. לאחר מכן ברגע שהשחקן פוגע בכדור (ע"י סיגנל col_player_ball) המשחק נגמר והמצב עובר ל gameOver.



7.2 [Ball duplicate] - [אוהד]

7.2.1 דיאגרמת מלבנים

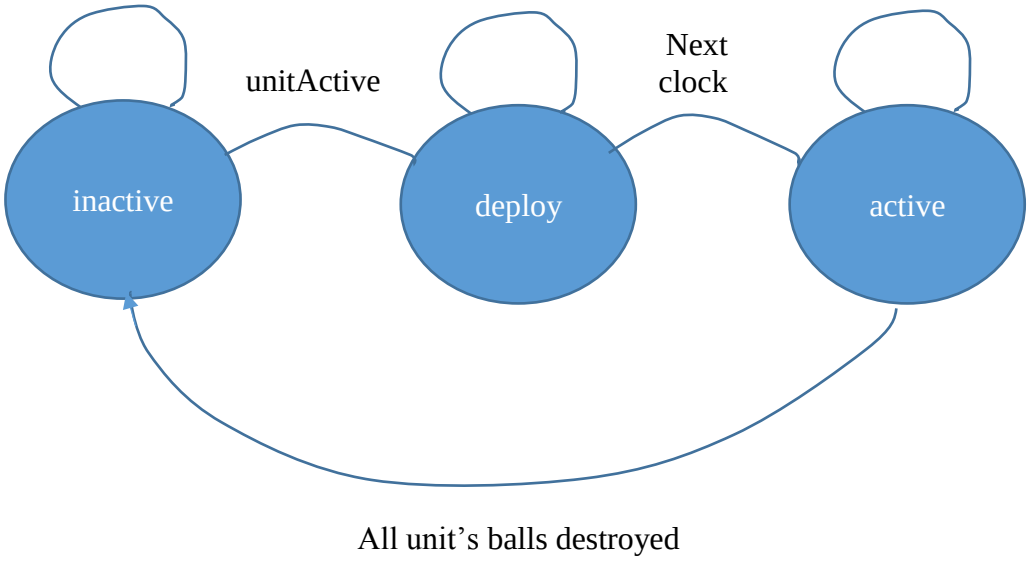
תאר את המודול כתהליך אחד או יותר.

מודול הכדור המתפצל הינו מסובך ולכן החלטנו לממש אותו כ-TOP נפרד עם Controller & MUX משלו. על מנת לתמוך ב-8 פיצולים – נממש יחידת ציור של כדור אשר מקבלת מיקום התחלתי מהירות התחלתית ואותות נראות.

נציב 15 יחידות כאלה בסכמת ה-TOP של הכדור הגדול ונתכנן בקר אשר יהיה אחראי על הצגת הכדורים ומהירותם בהתאם למהלך המשחק.

7.2.2 דיאגרמת מצבים

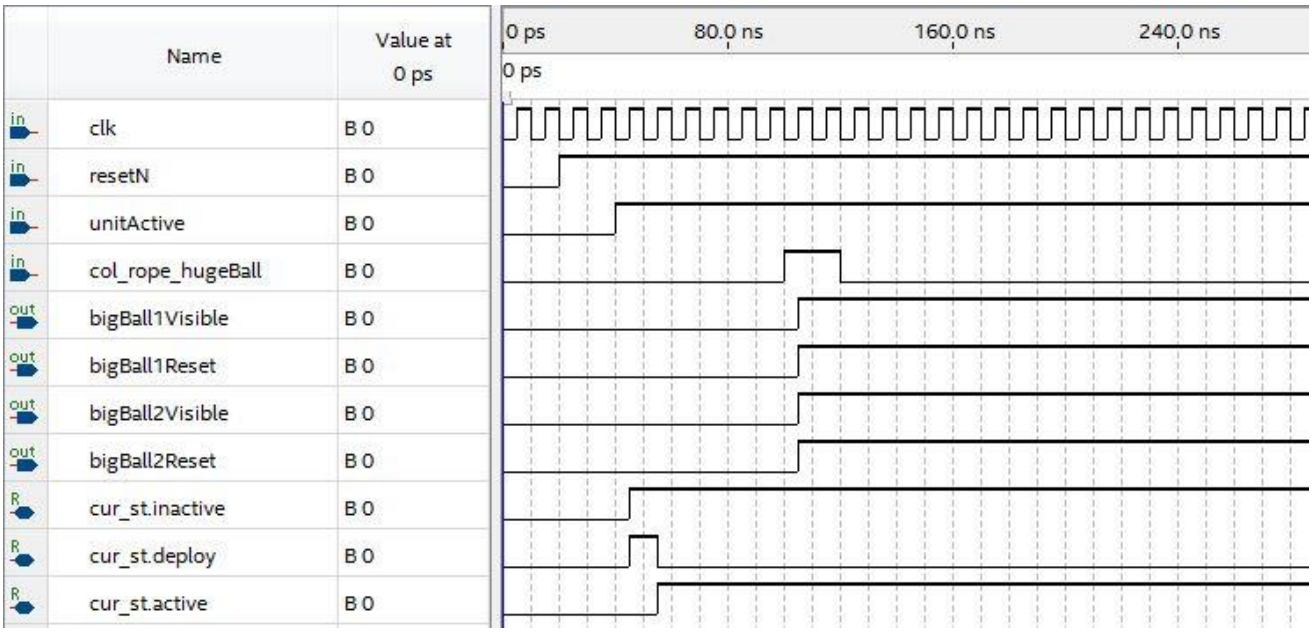
לתהליכים אותם מימשת בעזרת מכונת מצבים, צייר את דיאגרמת המצבים



7.2.3 מסך(י) סימולציה

יש לבדוק את כל הכניסות והיציאות, כל מקרי הקצה וכל המקרים המיוחדים.
אם יש צורך, הצג את תוצאות הסימולציה במספר חלונות. מעל כל חלון כתוב מה הוא בודק. סמן בעזרת חיצים על דיאגרמת הזמנים, את מקום הבדיקה.
וודא שבחלון הסימולציה רואים את רשימת האותות ואת ציר הזמן.

בסימולציה הבאה ניתן לראות שהפעלת הכדור ע"י unitActive מעבירה את מצב המכונה ל deploy ולאחר מכן ל active כמצופה. לאחר מכן העלאת הסיגנל col_rope_hugeBall גורם לכדורים הקטנים יותר להופיע ולבצע אתחול (ballVisible/ballReset) כמצופה.

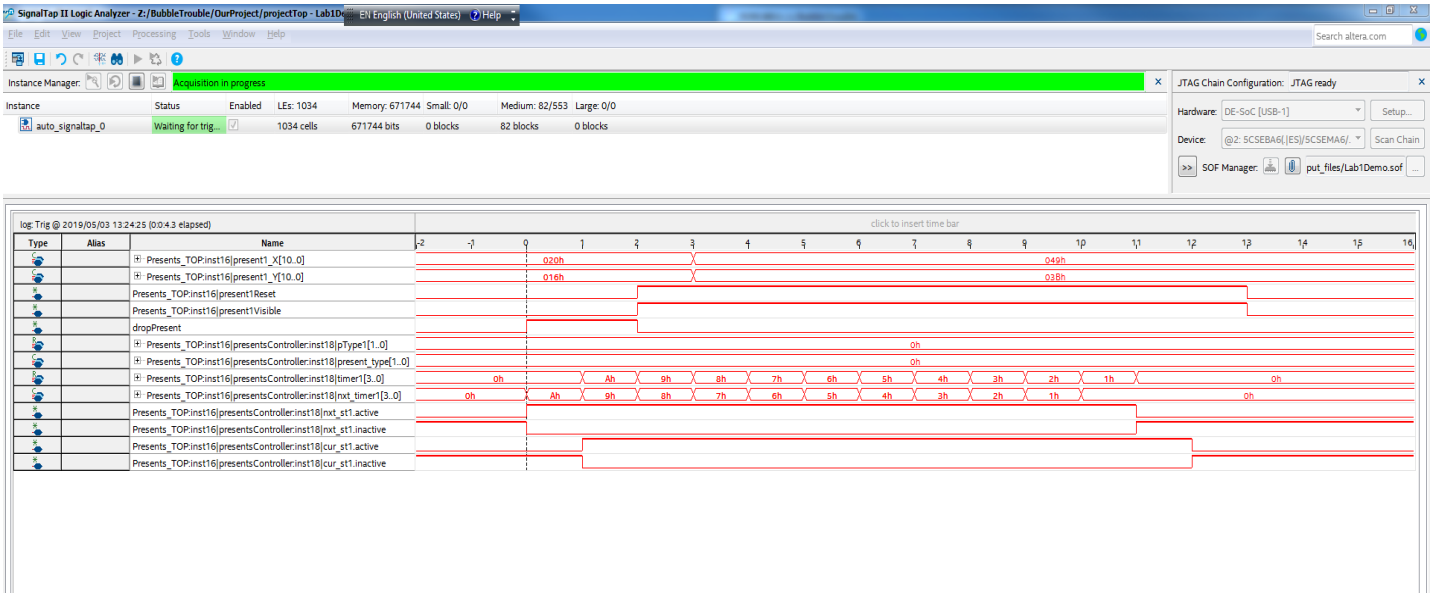


8 (S.T.) Signal Tap

אם השתמשת ב .S.T לזהות באג בחומרה, צרף מסך של ה .S.T בו זיהית את הבאג. הסבר מה היה הבאג, כיצד זיהית אותו וכיצד תקנת אותו.

אם לא השתמשת ב .S.T לזיהוי באג בחומרה, צרף מסך של ה .S.T בו מתבצעת פעולה סינכרונית והסבר אותה.

שימו לב יש למלא חלק זה במהלך העבודה ולא לצאת ידי חובה אחרי שסיימתם



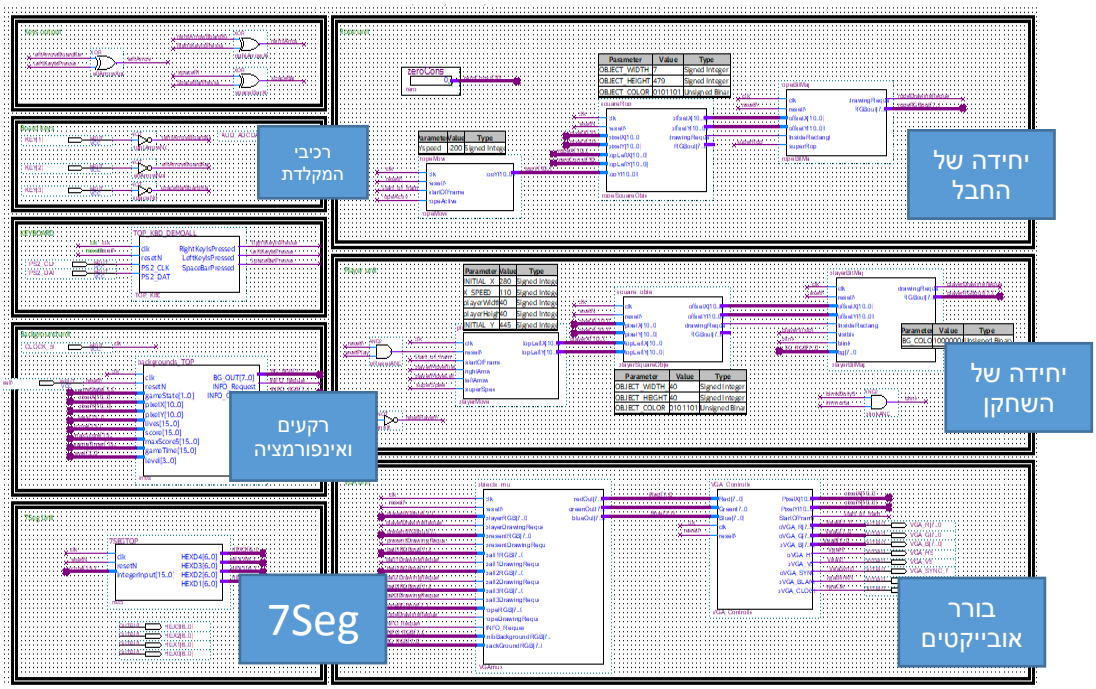
<u>הבעיה</u>: מתנה לא הופיעה על המסך ברגע פגיעה בכדור.

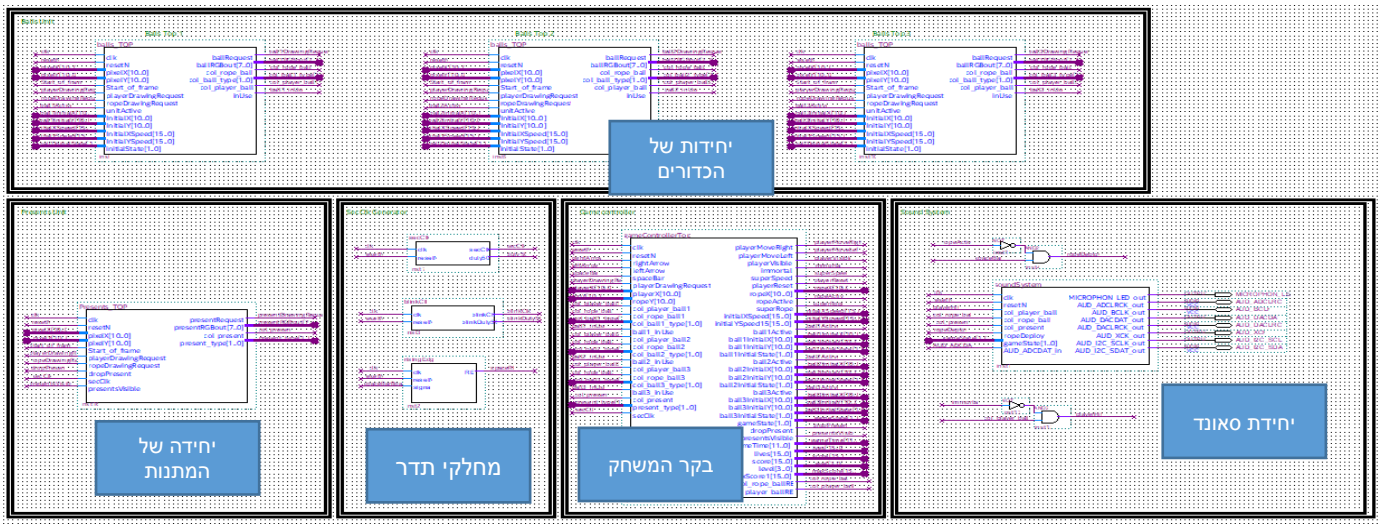
<u>הפתרון</u>: למתנה הוגדר פרק זמן שבה היא "חיה" כלומר כעבור פרק זמן כלשהו המתנה נעלמת מהמסך. ניתן לראות שמונה הזמן "timer1" סופר אחורה בכל מחזור שעון של הכרטיס במקום בכל שנייה. כך זיהינו את הבעיה ותיקנו אותה כך שהמונה סופר אחורנית לפי מחלק התדר של השניות.

9 מימוש ההיראדכיה עליונה

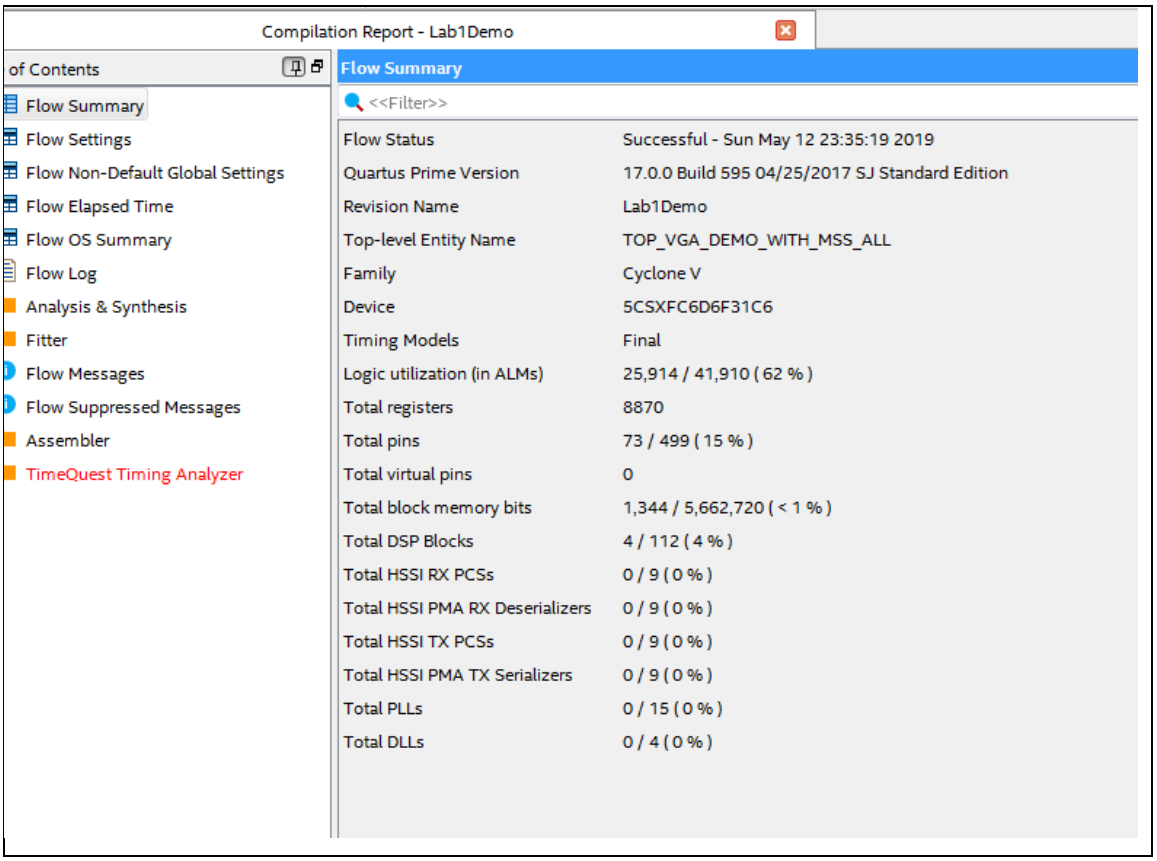
9.1 שרטוט

שרטוט מלבנים של ההיראכיה (העליונה של דהפרויקט – מצויר מעל תדפיס הקוארטוס – ראה דוגמא





9.2 צריכת משאבים



האם צריכת המשאבים (CELLS סבירה , לאן לדעתכם הלכו רב המשאבים
האם עמדתם בדרישת קומפילציה בפחות מ10 דקות ? לא עמדנו ב 10 דקות קומפילציה, אבל זאת מכיוון ששיפרנו בצורה משמעותית את התנהלות המשחק.

10 סיכום ומסקנות

עמידה בדרישות, קשיים , פתרונות אחרים, שימוש בכלים, מסקנות.

לאחר השקעה רבה הצלחנו לעמוד בכל הדרישות ואף להוסיף מטרות משלנו.

נוכחנו לגלות שהשימוש בsignal tap וב-in system memory הכרחי על מנת לדבג את המשחק.

אם לא היינו רוכשים מיומנות בכלים אלו לפני תחילת העבודה על הפרויקט אנחנו מאמינים שלא היינו מצליחים להגיע לתוצאה זו.

המסקנה העיקרית שלנו היא שעל מנת לבנות משחק מודולרי (כזה שמאפשר הוספת פיצ'רים בזמן סביר) חייבים לעבוד מסודר, הן בתכנון ה-TOP והן בקוד.

11 המלצות לשנה הבאה

הכל היה עשר, תודה רבה

12 נספחים: דפי נתונים, דפי מידע שונים בהם השתמשת.

https://www.doulos.com/knowhow/sysverilog/tutorial/datatypes/
נעזרנו בעמוד האינטרנט הנ"ל על מנת להבין את סוגי הסיגנלים אשר ניתן להשתמש בהם בשפת התכנות SystemVerilog.

SystemVerilog Data Types

This tutorial describes the new data types that Systemverilog introduces. Most of these are synthesisable, and should make RTL descriptions easier to write and understand.

Integer and Real Types

SystemVerilog introduces several new data types. Many of these will be familiar to C programmers. The idea is that algorithms modelled in C can more easiliy be converted to SystemVerilog if the two languages have the same data types.

Verilog's variable types are *four-state*: each bit is 0,1,X or Z. SystemVerilog introduces new *two-state* data types, where each bit is 0 or 1 only. You would use these when you do not need X and Z values, for example in test benches and as for-loop variables. Using two-state variables in RTL models may enable simulators to be more efficient. Used appropriately, they should not affect the synthesis results.

TYPE	Description	Example
bit	user-defined size	bit [3:0] a_nibble;
byte	8 bits, signed	byte a, b;
shortint	16 bits, signed	shortint c, d;
int	32 bits, signed	int i,j;
longint	64 bits, signed	longint lword;

Two-state integer types

Note that, unlike in C, SystemVerilog specifies the number of bits for the fixed-width types.

TYPE	Description	Example
reg	user-defined size	reg [7:0] a_byte;
logic	identical to reg in every way	logic [7:0] a_byte;
integer	32 bits, signed	integer i, j, k;

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

מלא את הטופס