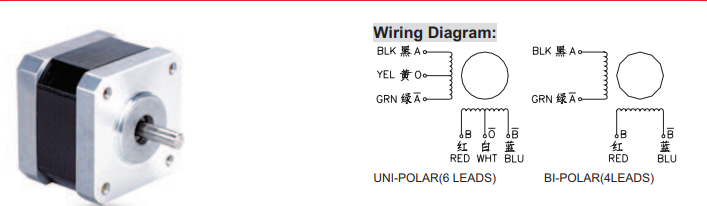
**דו''ח מעבדה**

**מעבדה 2 – בקר מנוע צעד**

מגישים: 342851284 איליה פורטוס

206698433 ארד שפירא

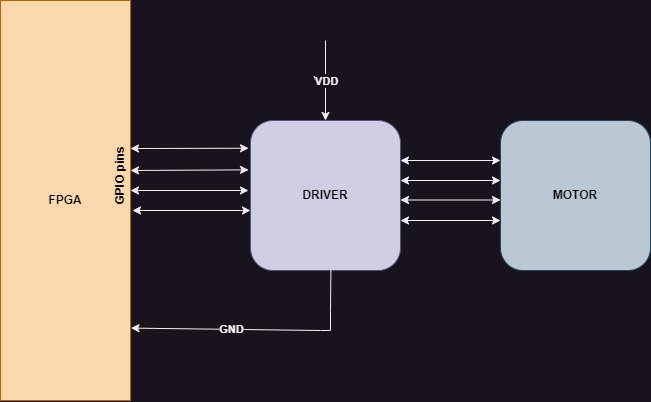
1. **סקירה על מנוע צעד**

מנוע צעד הוא סוג של מנוע המופעל על ידי מתח DC. ההזזה של המנוע מתקיימת על ידי הפעלת מתח על הסלילים אשר מזיזים את המנוע בצעדים קבועים. הסלילים מקבלים מתח וכתוצאה מכך נוצר שדה מגנטי אשר מזיז את ציר המנוע בצעד. על ידי הפעלת מתח על הסלילים השונים בסדר מסוים ניתן להזיז את המנוע בסיבובים שלמים בהתאם לפרמטרים (מהירות המנוע, כיוון התנועה, גודל הצעד וכדומה).

הפעלת המנוע: המנוע זז ב- בצעד מלא וב- בחצי צעד. לכן להשלמת סיבוב מלא נדרשים בהתאם 200 ו-400 צעדים. הסלילים מופעלים על ידי הספקת סדרות הבאות:

* צעד מלא עם כיוון השעון: 1000, 0010, 0100, 0001
* צעד מלא נגד כיוון השעון: 0001, 0100, 0100, 1000
* חצי צעד עם כיוון השעון: 1000, 1010, 0010, 0110, 0100, 0101, 0001, 1001
* חצי צעד נגד כיוון השעון: 1001, 0001, 0101, 0100, 0110, 0010, 1010, 1000

בצעד מלא הסלילים מופעלים אחד-אחד, לעומת זאת עבור צעד חלקי מפעילים סליל ראשון, אז מפעילים את השני בלי לכבות את הראשון ולבסוף מפסיקים הספקת מתח על הראשון ומפעילים רק על השני. אופן פעולה של חצי צעד מאפשר תנועה יותר חלקה.

חיבור המנוע לכרטיס: בחיבור ל-FPGA השתמשנו בדרייבר חיצוני על מנת לספק את המתח הדרוש. להלן מוצגת דיאגרמת החיבור:

1. **דיאגרמת הבלוקים של המעגל**

להלן מוצגת דיאגרמת הבלוקים של המעגל. בה ניתן לראות את החילוק לאזורים לוגיים:

1. לוגיקה של שינוי מהירויות

2. לוגיקה של רבע סיבוב

3. הצגת מהירות למסך 7-segment

A picture containing text, screenshot, diagram, circuit

Description automatically generated4. מכונת מצבים האחראית על הזזת המנוע.

המימוש שבחרנו מאפשר מודולריות גבוהה של המערכת, כלומר ניתן בקלות להוריד/להוסיף פעולה כלשהי למנוע בלי לפגוע בשאר המערכת.

1. **הסבר של המימוש**

כעת נעבור על כל המודולים ונראה מהי הפעולה של כל אחד מהם.

* A screenshot of a computer program

  Description automatically generated with low confidence*step\_motor\_top*

A picture containing text, font, screenshot

Description automatically generated

זהו מודול ה-top בו מתקיימות הקריאות למודולים אחרים עם הכניסות המתאימות. הלוגיקה היחידה שנמצאת במודול (שורה 48) היא בחירה בין סיגנל מהמתג on ובין quarter\_out – ביט המוצא של מודול של רבע סיבוב. על ידי שער OR אנו קובעים איזה סיגנל יהיה ה-enable של מכונת המצבים של המנוע.

* A screen shot of a computer program

  Description automatically generated with low confidence*posedge\_detector*

המודול אחראי על זיהוי posedge של לחיצה על כפתור שינוי המהירות. מכיוון שהלחיצה היא הרבה יותר ארוכה ממחזור השעון של הFPGA, רצינו לזהות את תחילת הלחיצה ולאחר מכן לשלוח פולס לאורך מחזור שעון אחד אשר ייקבע לדרגה הבאה שהתרחשה הלחיצה.

* A screenshot of a computer program

  Description automatically generated with medium confidence*speed\_selector*

A screen shot of a computer program

Description automatically generated with low confidence

המודול כולל את מכונת המצבים האחראית על החלפת מהירויות. הוא מקבל פולס של מחזור שעון אחד מהמודול הקודם, ובהתאם מחליף את המצב שלו. ישנם 6 מצבים בהתאם למהירויות. כשאר מגיעים למצב של 10/60 rpm, מחליפים את משתנה ה-flag וחוזרים חזרה בהתאם לתנאי בכל מצב (flag=1 – קדימה, אחרת אחורה).

* A screenshot of a computer program

  Description automatically generated with medium confidence*speed\_controller\_sm*

**A screenshot of a computer program

Description automatically generated with medium confidence**

המודול מקבל את ערך המהירות מהמודול הקודם (3 ביט) ובהתאם לגודל הזה וגודל הצעד (input step) מוציא את מספר עד עליו נצטרך לספור. המספר הזה מתקבל אח''כ במחלק השעון (ראה עמוד הבא). הערה: חישבנו את המספרים האמיתיים לפי פרמטרי המנוע, אך בהצבה של המספרים האלה לא קיבלנו את המהירות הרצויה (מדדנו בפועל את כמות הסיבובים לדקה שהמנוע עושה). לכן חילקנו את כל אחד מהמספרים ב-2, שזה נותן בדיוק את המהירויות הרצויות.

* A screen shot of a computer code

  Description automatically generated with medium confidence*clock\_divider*

זהו המודול של מחלק שעון. הוא מקבל את המספר עד עליו צריך לספור (count\_to) ובהתאם למהירות שהתקבלה מייצר את השעון המחולק אשר ישמש במכונת המצבים של המנוע. המודול בנוי כמו counter רגיל שראינו בקורס, רק אם הבדל אחד של החלפת ערך במקום הוצאת אותו חוצה (שורה 20). השינוי הזה גורם לכך שהמוצא של מודול לא יראה כמו רצף פולסים אלא שיהיה שעון עם מחזור שעון ארוך מהזה של FPGA.

כעת נעבור למודולים אשר קשרים לשליטה על רבע סיבוב.

* *negedge\_detector*

A picture containing text, screenshot, font, display

Description automatically generated

המודול בנוי באותה צורה כמו posedge\_detector מקודם אך עם שינוי בהשמה של הערך בסוף. המודול נועד לזיהוי לחיצה על כפתור של quarter. כמו מקודם, הלחיצה הרבה יותר ארוכה ממחזור השעון, לכן נזהה את תחילת הלחיצה ונשלח פולס למשך מחזור שעון אחד.

* *quarter\_count*

**A screenshot of a computer program

Description automatically generated with medium confidence**

A picture containing text, screenshot, software, font

Description automatically generated

המודול שולט על פעולה של רבע סיבוב. המודול מקבל את השעון המחולק בהתאם למהירות הנוכחית, את הפולס מהמודול הקודם שקובע את הלחיצה על הכפתור ואת גודל הצעד. בתוך המודול יש counter עם enable, כלומר אם זיהינו את הלחיצה, המודול מופעל, לפי גודל הצעד נקבע עד כמה לספור וה-counter מאותחל. המודול מוציא החוצה את השעון לאורך של כמות המחזורים (בהתאם לגודל הצעד) ואז 0. הערה: בגלל שחילקנו את כל המספרים במודול של speed\_controller\_sm פי 2, בהתאם נצטרך לספור לא עד 100 (חצי צעד) ו-50 מחזורים (צעד מלא), אלא ל-200 ו-100 בהתאם. בבדיקה המעשית ראינו שעם המספרים האלה המנוע אכן עושה רבע סיבוב כנדרש.

* A picture containing text, screenshot, software, number

  Description automatically generated*motor\_sm*

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with low confidence

זהו המודול אשר קובע את המצב הנוכחי של המנוע ושולח לו את 4 ביט של כניסה. המודול מכיל את מכונת המצבים אשר לפי גודל הצעד וכיוון התנועה (עם/נגד כיוון השעון) מחליף בין הצמבים. יש סה''כ 8 מצבים אפשריים – 4 ראשיים לצעד מלא ו-4 מצבי ביניים לחצי צעד. המנוע מקבל את השעון המחולק בהתאם למהירות מה-clock\_divider.

* *seven\_segment*

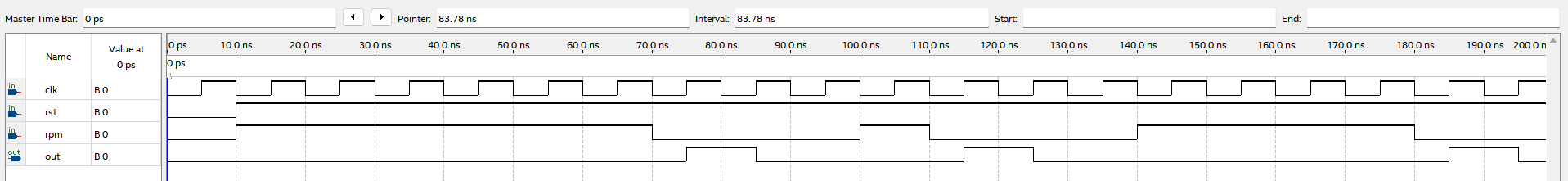
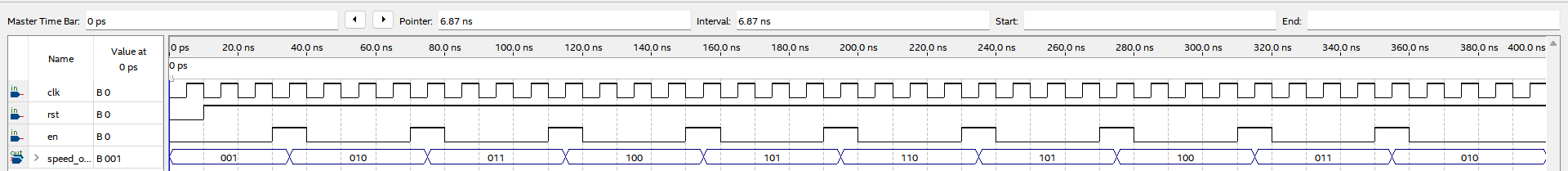
A picture containing text, screenshot, number, display

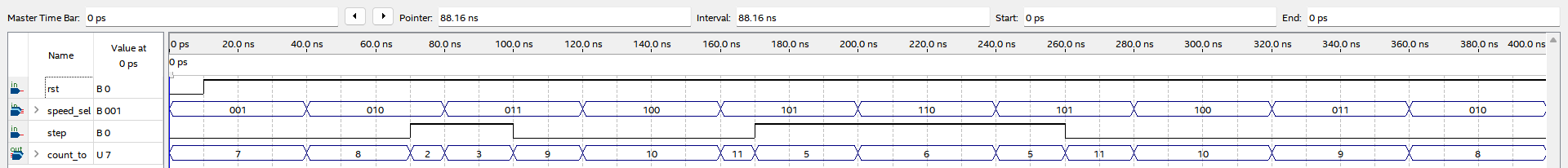
Description automatically generated

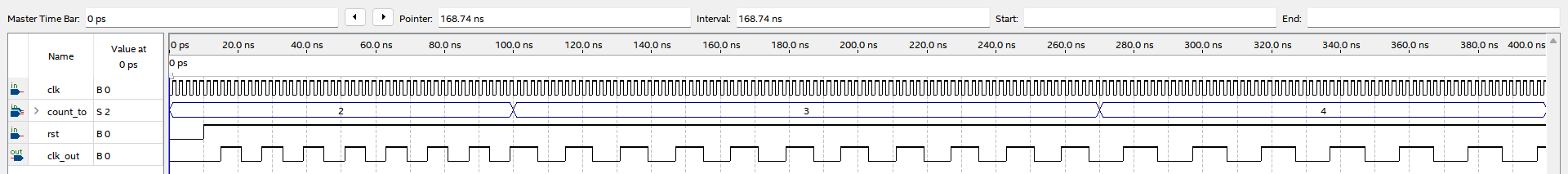
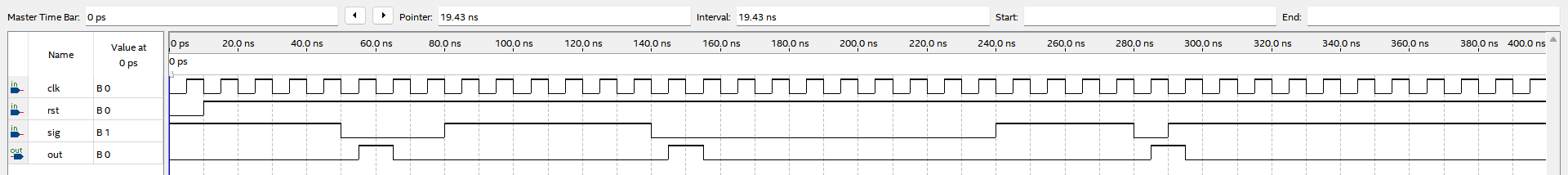
זהו המודול אשר מעביר את הערך של המהירות למספר אשר נשלח ל-7-segment display. יש שתי קריאות למודול – אחת לעשרות ואחת לאחדות. הקריאה לאחדות מקבלת פרמטר 0 תמיד (הרי המהירויות שלנו הם כפולות של 10) והקריאה לעשרות מקבלת את הערך מהמודול speed\_controller (אשר מוציא את הערכים 1 עד 6 בהתאם למהירות הנוכחית).

1. **סימולציה – דיאגרמות גלים**

להלן הסימולציות של כל המודולים:

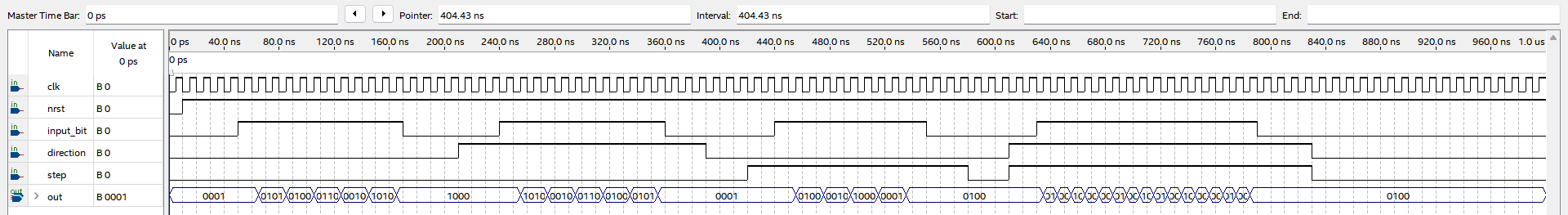
* *posedge\_detector*
* *speed\_selector*
* *speed\_controller\_sm*

הערה: על מנת להבחין בשינויים בסימולציה, שמנו ערכים קטנים יותר

* *clk\_divider*
* *****negedge\_detector*
* *quarter\_count*

A screenshot of a computer

Description automatically generated with low confidenceהערה: על מנת לבחון בשינויים בסימולציה, הקטנו את הפרמטרים מ-100 ו-200 ל-4 ו-8 בהתאם.

* *motor\_sm*

**5. סרטון של המנוע**

להלן מצורף קישור לסרטון המוכיח את פעולת המנוע בכל המצבים הנדרשים שלו: