פרק 8 : PWM באמצעות מערכת PCA

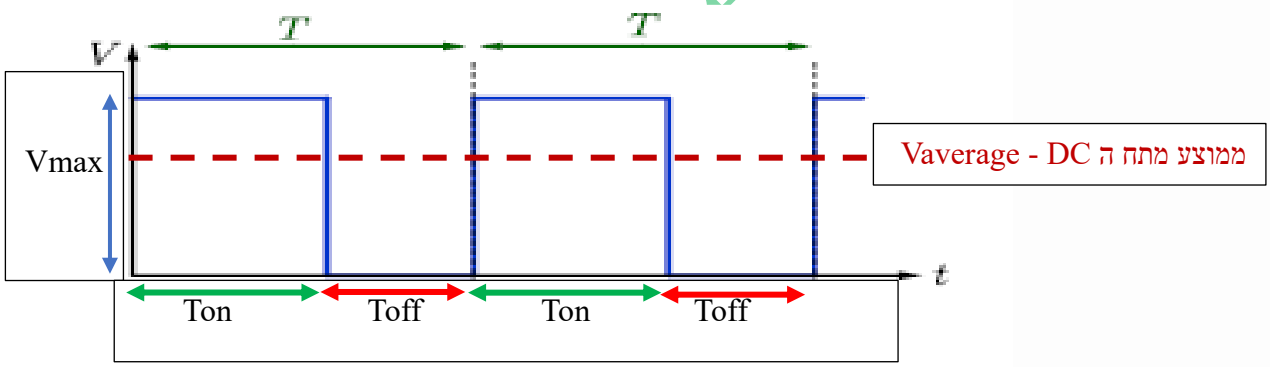
Pulse Width Modulation - PWM - אפנון רוחב דופק- עם PCA - Programmable Counter
Array - מערך המונה בר התכנות .

1. א מה זה PWM ?

אפנון רוחב פולס - Pulse Width Modulation PWM היא טכניקה שבה הערך הממוצע של מתח הכניסה מותאם על ידי שליחת סדרה של פולסים OFF-ON. מתח ממוצע זה פרופורציונלי לרוחב הפולסים .

ה PWM משמש לשליטה על מהירות מנועים חשמליים, עוצמת ההארה של נורות LED , טעינת סוללות ועוד. הדבר מאפשר לשלוט בכמות האנרגיה המועברת לרכיב בצורה מדויקת יותר משיטות אחרות.

מחזור עבודה/מנת פעולה – Duty Cycle הוא היחס בין הזמן של ה ON לחלק בזמן ON+OFF . האיור הבא מראה דוגמה של PWM . אפשר גם לומר שמחזור הפעולה הוא פרק הזמן בו מערכת פעילה ביחס לזמן המחזור השלם.



איור 1 : דוגמה של PWM

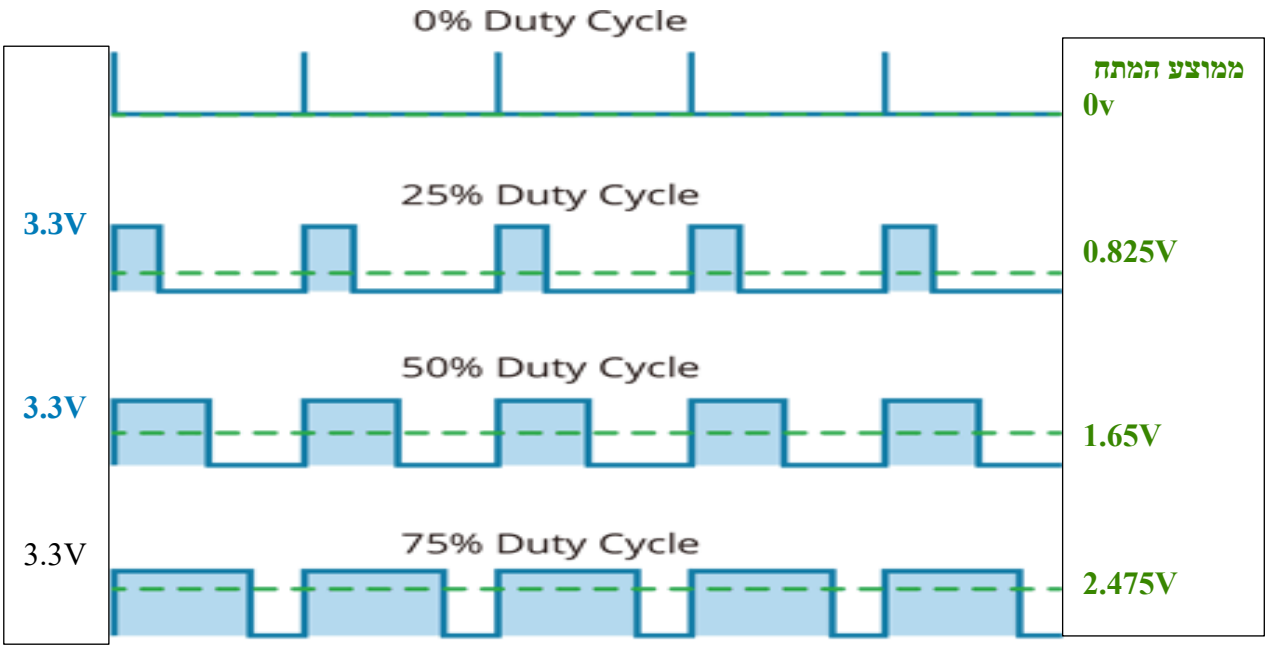
הנוסחה לחישוב מחזור העבודה/מנת פעולה/ Duty Cycle היא :

Duty Cycle = Ton / Toff = Ton / (Ton+Toff)

הנוסחה לחישוב מתח ה DC הממוצע של צורת גל PWM היא:

$$V_{average} = Vmax * T_{on} / T = Vmax * T_{on} / (Ton + Toff)$$

האיור הבא מתאר את ה Duty Cycle במספר מצבים של Duty Cycle. הציר האנכי הוא מתח והציר האופקי הוא זמן. בהנחה שהמתח המקסימלי המגיע ממיקרו בקר C8051F380 הוא מתח של 3.3V = **Vmax** (שזהו מתח ספק הכוח המגיע להפעלת המיקרו בקר) נקבל את ממוצעי ה DC הבאים :



איור 2 : דוגמאות ל Duty Cycles ב PWM .

אם מספקים גל PWM למנוע DC (או כל צרכן אחר כמו מנורות , מנועי סרוו וכו') אז ככל שמחזור העבודה גבוה יותר, כך המתח הממוצע המופעל על המנוע גבוה יותר ונקבל עלייה במהירות המנוע. ככל שמחזור העבודה קצר יותר, כך המתח הממוצע המופעל על המנוע נמוך יותר, וכתוצאה מכך נקבל ירידה במהירות המנוע.

בהנחה שהמנוע מקבל את צורת הגל הראשונה Duty Cycle = 0% הוא לא יסתובב כלל. ככל שאחוז ה Duty Cycle גדול יותר ממוצע ה DC גדול יותר והמנוע יסתובב במהירות גבוהה יותר.

אם המתח למנוע מגיע ממיקרו C8051F380 שמתח ההפעלה שלו 3.3V אז בדוגמה של ה Duty Cycle של ה 25% ממוצע ה DC יהיה 0.825 וולט. בדוגמה של ה 50% ממוצע ה DC יהיה 1.65 וולט ובדוגמה של ה 75% הוא יהיה 2.475 וולט.

1.ב מה זה מערך מונה בר תכנות ?

במילה בר תכנות הכוונה שניתן לתכנת אותו ובהמשך נשתמש גם במילה בר תכנות וגם במילה מתוכנת.

ה- PCA מורכב מטיימר / מונה - Timer/Counter - אשר משמש כבסיס הזמן למערך של חמישה מודולים של השוואה/לכידה - capture/compare . כלומר יש לנו מונה אחד ומערך של 5 מודולים להשוואה/לכידה.

1.ג מה זה capture/compare mode - אופן לכידה/השוואה ?

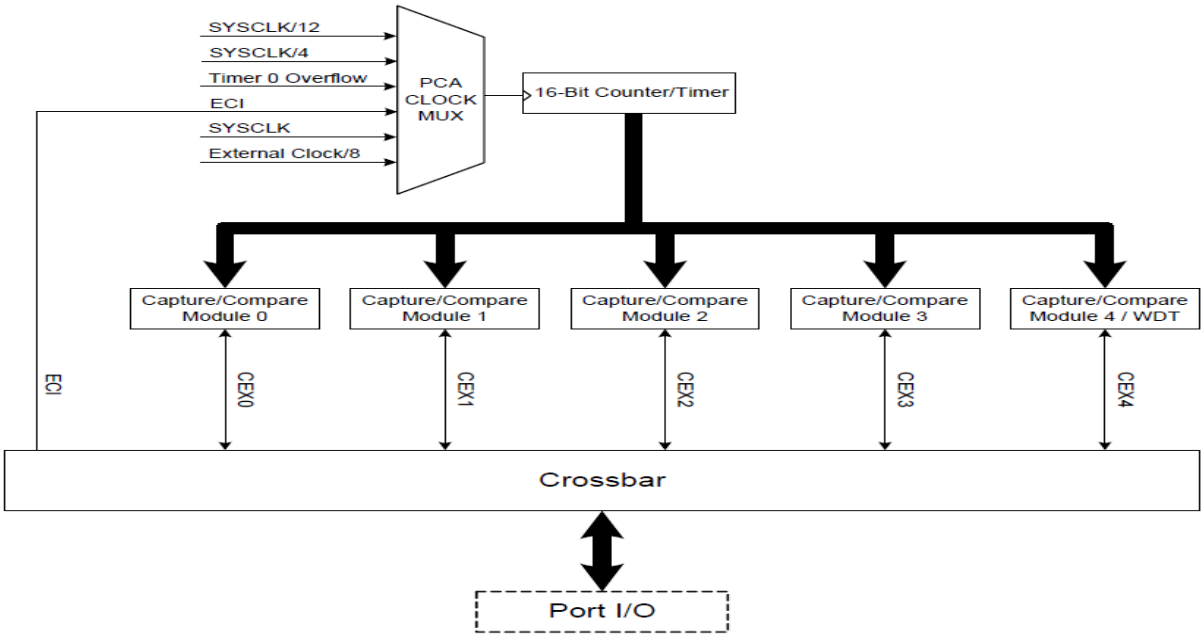
מצב לכידה – capture - מאחזר ערך טיימר בהתבסס על אירוע אות. במילים פשוטות כאשר מתקבל אות/טריגר הערך שבמונה/טיימר מועבר אל רגיסטרים שנבחר.

מצב השוואה – compare - מתבצעת השוואה בין הערך של מונה/טיימר ובין ערך שהכנסנו להשוואה. כאשר הערך במונה שווה לערך הרצוי לנו נקבל הודעה כמו פסיקה או העברת דגל ל 1 .

2. ה PCA במיקרו C8051F380

במיקרו בקר C8051F380 יש מערך מונה שניתן לתכנות (PCA0) שמספק פונקציונליות משופרת של טיימר תוך שהוא דורש פחות התערבות של המיקרו בקר מאשר מונה/טיימר סטנדרטי (כמו טיימר 0 או 1 או 2). ה-PCA מורכב ממונה/טיימר Timer/Counter ייעודי של 16 סיביות וחמישה מודולים של לכידה/השוואה - capture/compare - גם הם של 16 סיביות. לכל מודול לכידה/השוואה יש קו קלט/פלט משלו הנקרא n (CEXn) הוא מספר בין 0 ועד 4 המתחבר דרך הקרוסבר ליציאה קלט/פלט שהמשתמש בוחר. כל מודול מחובר לאחד ההדקים של פורט 0 (זוהי ברירת המחדל).

האיור הבא מתאר את הסכמה המלבנית של מערך המונה המתוכנת :



איור 3 : סכמה מלבנית של מערך המונה בר התכנות.

באיור רואים טיימר קאונטר של 16 ביטים - 16-bit Counter/timer. זהו המונה שסופר את פולסי הספירה . הוא מקבל את פולסי הספירה דרך מולטיפלקסר – PCA CLOCK MUX. ה MUX מקבל פולסי ספירה מ 6 מקורות. את מצב הספירה שלו הוא מעביר למערך של 6 מודולים של השוואה/לכידה Capture/Compare Module ממודול 0 ועד 4 . מודול 4 משמש גם כטיימר כלב שמירה **WDT- Watch Dog Timer והוא מאופשר בזמן ה RESET**. כל מודול יש הדק הנקרא CEXn (n מ 0 עד 4) שיכול להתחבר דרך הקרוסבר לאחד מההדקים החיצוניים של המיקרו בקר.

הערה חשובה: מודול 4 של ה PCA עשוי לשמש כטיימר כלב שמירה (WDT), והוא מופעל למצב זה לאחר איפוס RESET של המערכת. **הגישה לאוגרי PCA מסוימים מוגבלת כאשר מצב WDT מופעל.**

המונה / טיימר יכול להיות מופעל על ידי בסיס זמן שניתן לתכנות שיכול לבחור בין שישה מקורות המגיעים ל MUX :

- שעון מערכת - SYSCLK
- שעון מערכת חלקי ארבעה - SYSCLK/4
- שעון מערכת חלקי שתים עשרה – SYSCLK/12

- מקור שעון המתנד החיצוני חלקי 8 – External Clock / 8
- גלישת טיימר 0 - Timer 0 OverFlow
- אות שעון חיצוני בפין הקלט של ECI.

כל מודול לכידה/השוואה יכול להיות מוגדר לפעול באופן עצמאי באחד משישה מצבים:

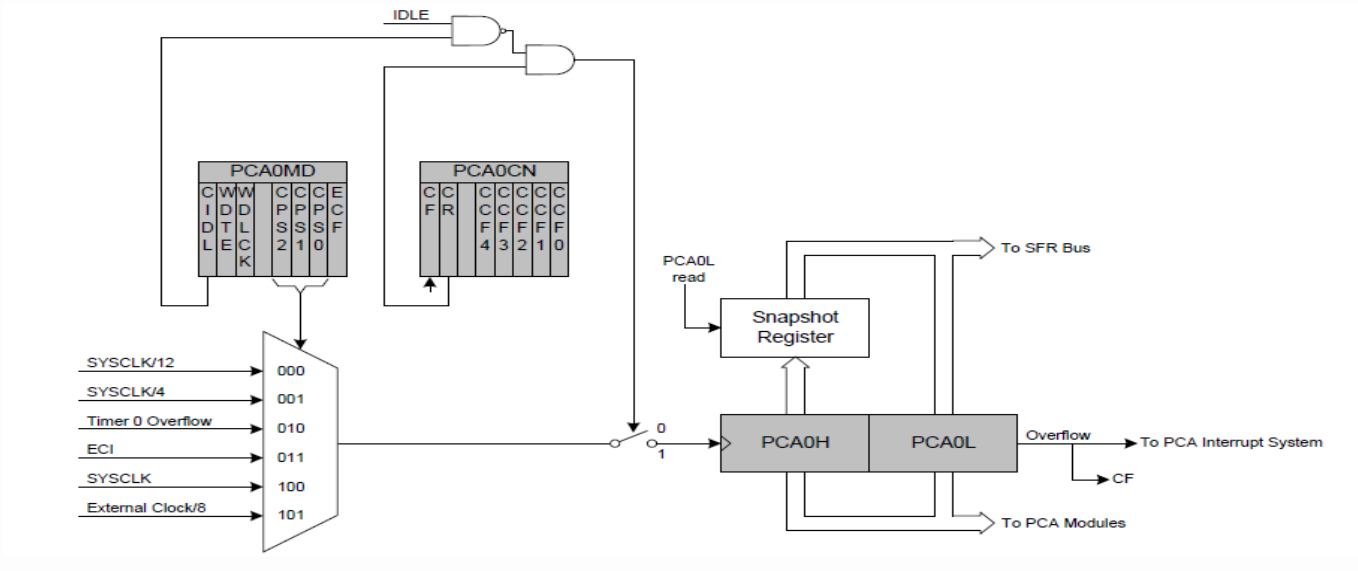
- לכידה המופעלת על ידי קצה/שפה – Edge (ירידה או עלייה).
- טיימר תוכנה.
- יציאה מהירות גבוהה.
- יציאת תדר.
- PWM של 8 סיביות.
- PWM של 16 סיביות .

האופציה של שעון מתנד חיצוני הוא אידיאלי לפונקציונליות של שעון בזמן אמת (RTC) ומאפשר ל PCA להיות מופעל על ידי מתנד חיצוני מדויק בזמן שהמתנד הפנימי מניע את מערכת השעון .

ה-PCA מוגדר ונשלט באמצעות אוגרים הנמצאים ב SFR – Special Function Registers – הרגיסטרים של הפונקציות המיוחדות של המיקרו בקר.

3. תרשים מלבנים - איך בנוי ה PCA Counter/Timer - מונה/טיימר של ה PCA ?

האיור הבא מתאר את תרשים המלבנים של ה PCA :



איור 4 : תרשים מלבנים של PCA Counter/Timer

באיור רואים במרכז בצבע כהה את המונה/טיימר עצמו.

בצד שמאל למעלה רואים את 2 הרגיסטרים השולטים על ה PCA ונקראים PCA0MD ו PCA0CN .

בצד שמאל באיור רואים את ה MUX הנשלט על ידי 3 ביטים של רגיסטר PCA0MD שקובעים מאיפה ומהו תדר פולסי הספירה.

מונה / טיימר PCA שבמרכז האיור הוא של 16 סיביות ומורכב משני מונים של 8 סיביות : PCA0L ו- PCA0H . PCA0H הוא הבית הגבוה (MSB) של טיימר/קאונטר של 16 סיביות ו- PCA0L הוא הבית הנמוך (LSB). קריאה ראשונה של PCA0L נועלת אוטומטית את הערך של PCA0H לאוגר snapshot -"תמונת מצב" . הקריאה הבאה של PCA0H מתבצעת מאוגר "תמונת המצב". הקריאה הראשונה/ההתחלתית של אוגר PCA0L מבטיחה קריאה מדויקת של מונה PCA0 כולו של 16 סיביות. קריאת PCA0H או PCA0L אינה מפריעה לפעולת הספירה של המונה.

כאשר המונה/טיימר גולש מהספירה המקסימלית שלו שהיא 0xFFFF ל - 0x0000 דגל הגלישה - Overflow – המסומן כאן גם (CF- Carry Flag) ב- PCA0MD עובר ל 1 לוגי ונוצרת בקשת פסיקה (בצד ימין באיור ביציאה מ PCA0L). אם אפשרנו פסיקות CF נקבל פסיקת PCA . מאפשרים לדגל CF ליצור בקשת פסיקה כאשר נשים בסיבית ECF ב - PCA0MD ל 1. ECF היא הסיבית הנמוכה ברגיסטר PCA0MD שבצד שמאל למעלה באיור) .

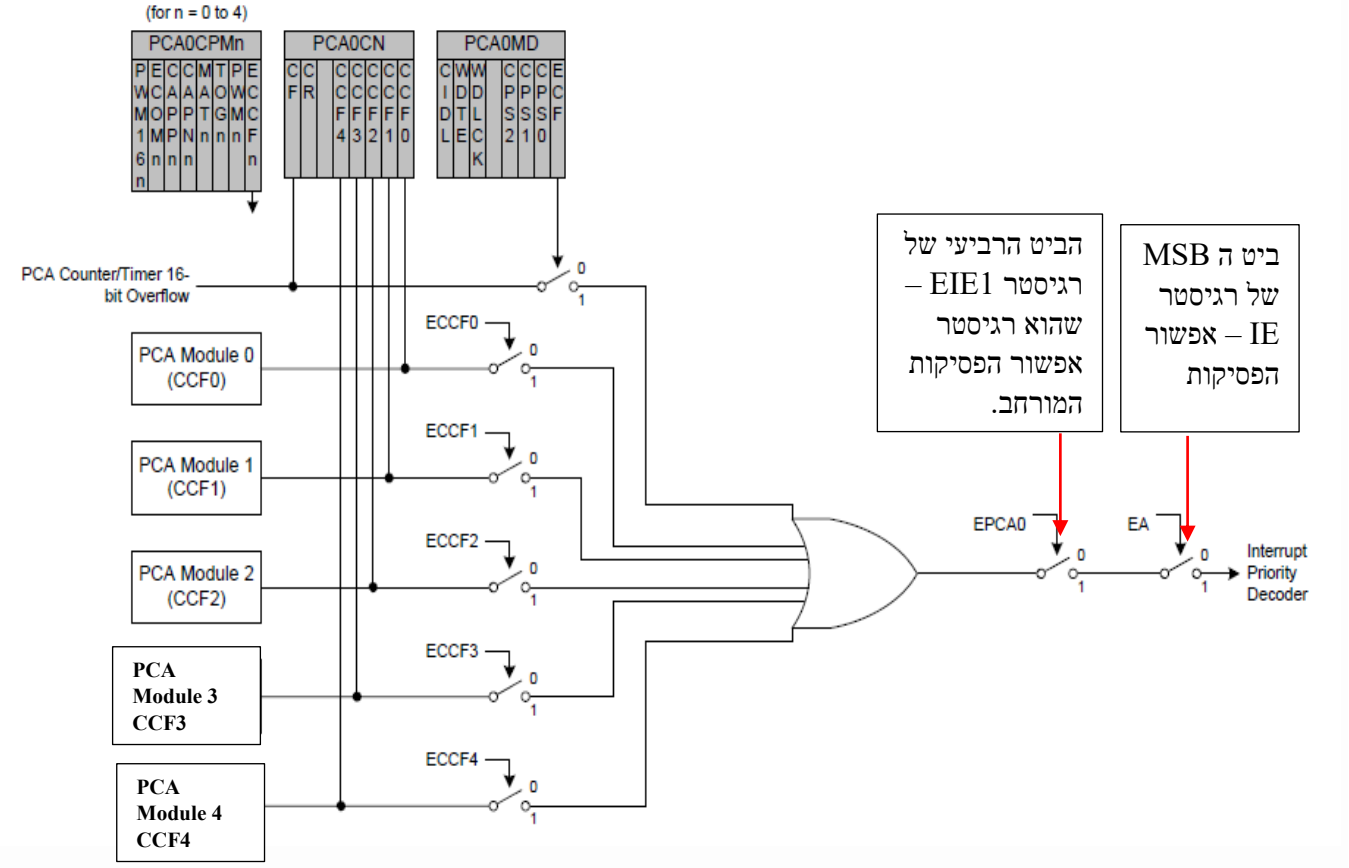
סיבית CF אינה מתאפסת אוטומטית על-ידי החומרה כאשר עוברים לפונקציית הפסיקה וצריך לאפס את הסיבית על-ידי תוכנה.

ניקוי סיבית CIDL באוגר PCA0MD מאפשר ל- PCA להמשיך בפעולה רגילה בזמן ש- ה- CPU נמצא במצב לא פעיל.

כדי לסגור את המפסק שבמרכז האיור ולאפשר לפולסי הספירה להגיע למונה צריך ששער ה AND (באיור הוא בצד שמאל למעלה) יוציא '1' . לשם כך צריך לשים '1' בביט CR ברגיסטר PCA0CN (ביט מספר 6) ולשים 0 בביט CIDL (ביט 7 ברגיסטר PCA0MD) או לא לעבוד במצב סרק IDLE=0.

4. הפסיקה של PCA0 (לא בתוכנית הלימודים)

הפסיקה המתקבלת מ PCA0 יכולה להתקבל מ **6 מקורות**. האיור הבא מתאר תרשים המלבנים של קבלת פסיקה ב PCA0 :



איור 5 : תרשים מלבנים של פסיקת PCA0 .

ישנם שישה מקורות שיכולים לשמש ליצירת פסיקת PCA0 והם נכנסים לשער ה OR במרכז האיור :

- גלישה של מונה PCA הראשי (CF) שעובר ל 1 בזמן גלישה של המונה PCA0 של ה 16 סיביות (הכניסה העליונה לשער ה OR). כדי לאפשר פסיקת גלישה של PCA0 יש לסגור את המפסק המופעל על ידי ביט ECF ברגיסטר **PCA0MD** .
- כאשר מתבצעת לכידה/השוואה בכל אחד מ 5 המודולים . לכל מודול יש פסיקה משלו - (CCF0, CCF1, CCF2, CCF3 ו-CCF4) - המוגדרים בהתאם למצב הפעולה של אותו מודול. כדי לקבל בקשת פסיקה מאחד המודולים יש לשים 1 בביט ה LSB – **ECCFn** - של רגיסטר **CPA0CPMn** כאשר n הוא מספר 0 עד 4 של אחד מ 5 המודולים .

פסיקת הגלישה של PCA0 היא פסיקה מספר 11 של המיקרו בקר.

למעלה באיור רואים שיש 2 רגיסטרים שנקראים PCA0MD ו PCA0CN ועוד רגיסטר הנקרא CPA0CPMn כאשר n מציין מספר בין 0 ל 4 , כלומר יש 5 רגיסטרים כאלו.

PCA0MD – PCA0 MoDe – אופן ה PCA - אחראי על אפשור פסיקת גלישה (מעבר של הטיימר/קאונטר מ 0xffff ל 0x0000) על מקור/המקום ממנו מגיעים פולסי הספירה, על התדר של הפולסים והוא מתואר בהרחבה בהמשך.

PCA0CN – PCA0 CoNtrol – בקרת PCA0 – יש לו ביט בקרה המפעיל או מפסיק את פעולת ה PCA ועוד 6 דגלים המציינים מאיפה הגיעה בקשת פסיקה (אחד מ 6 המקורות שלמעלה). בהמשך נרחיב על רגיסטר זה.

CPA0CPMn – CPA0 Capture Compare Mode n – n הוא מספר בין 0 ל 4 כי יש 5 רגיסטרים כאלו. ביט ה LSB בכל אחד מ 5 הרגיסטרים האלו קובע האם לאפשר פסיקת לכידה/השוואה. גם על רגיסטר זה נרחיב בהמשך.

כדי ליצור פסיקת PCA0 :

- כדי לקבל פסיקה כאשר יש גלישה מהמונה עצמו יש לשים '1' בביט האפשור (ביט מספר 0 ברגיסטר PCA0MD הנקרא ECF .
- לשים '1' בביט - ECCFn (עבור כל n מ 0 עד 4 של כל מודול). זהו ביט האפשור לכל מודול כאשר מתקיים מצב Capture/Compare .

פסיקות PCA0 חייבות להיות מאופשרות באופן גלובלי על ידי סיבית EA=1 ברגיסטר אפשור הפסיקות **IE** ובסיבית **EPCA0** שהיא סיבית 4 ברגיסטר EIE1 שהוא רגיסטר - **Extended Interrupt Enable 1 – אפשור פסיקה מורחבת מספר 1** . כדי לאפשר את פסיקות ה PCA0 (אם רוצים ללא פסיקות נוספות) יש לרשום : א. EA=1; או IE=0x80 ו ב. EPCA0=1; עבור ביט בודד או עבור הרגיסטר EIE1 ; EIE1=0x10 .

הרגיסטר EIE1 עצמו נראה כך:

Bit	7	6	5	4	3	2	1	0
Name	ET3	ECP1	ECP0	EPCA0	EADC0	EWADC0	EUSB0	ESMB0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

איור 6 : רגיסטר EIE1 .

ברגיסטר זה יש את אפשור הפסיקות הבא (מהביט הנמוך לגבוה):

- פסיקת SMBUS0 .
- פסיקת SMBUS1 .
- פסיקת חלון ההשוואה של ADC0 .
- פסיקת סיום ההמרה של ADC0 .
- פסיקת PCA0 .
- פסיקת משווה 0 . (במיקרו בקר C8051F380 יש 2 משווים)
- פסיקת משווה 1 .
- אפשור פסיקת טיימר 3 .

5. הרגיסטרים הקשורים למערך המונה בר התכנות PCA0

ישנם 2 רגיסטרים השולטים על PCA0 ונקראים PCA0MD ורגיסטר PCA0CN ועוד 5 רגיסטרים עבור המערך של 5 המודולים - PCA0CPn (n הוא מספר בין 0 ל 4 ומציין את מספר המודול). כמו כן בעזרת רגיסטר XBR1 קובעים איזה הדק של פורט 0 יהיה הדק CEXn שיכול להיות גם הדק כניסה או הדק יציאה (תלוי באופן/ מצב העבודה עם PCA0).

5.1 רגיסטר האופן – PCA0MD - PCA Mode

ברגיסטר זה נגדיר את אופן העבודה עם ה PCA0 , בסיס הזמן של הספירה - כלומר את המקור של פולסי הספירה שיגיעו למונה ואת התדר שלהם, האם לאפשר את טיימר כלב השמירה – Watch Dog Timer , האם לאפשר פסיקת גלישת מונה והאם ה PCA0 יעבוד במצב סרק – IDLE.

הרגיסטר נראה באיור הבא :

Bit	7	6	5	4	3	2	1	0
Name	CIDL	WDTE	WDLCK		CPS[2:0]			ECF
Type	R/W	R/W	R/W	R	R/W			R/W
Reset	0	1	0	0	0	0	0	0

איור 7: רגיסטר האופן PCA0MD .

ביט 7 – CIDL - **PCA Counter/Timer Idle Control.**

בקרת מצב סרק של טיימר/קאונטר PCA0 . מציין אופן פעולה של PCA כאשר ה- CPU נמצא במצב לא פעיל.

אם נשים בביט 0 אז PCA0 ימשיך לפעול כרגיל כאשר המיקרו בקר נמצא במצב סרק. אם נשים 1 אז פעולת PCA מופסקת כאשר המיקרו בקר נמצא במצב סרק.

ביט 6 – WDTE – **Watchdog Timer Enable** – אפשור כלב השמירה .

אם נשים בביט '1' אז מודול 4 משמש כטיימר כלב השמירה. 0: טיימר כלב שמירה מושבת.

ביט 5 – WDLCK - **Watchdog Timer Lock** – נעילת כלב השמירה

סיבית זו נועלת או לא נועלת את אפשור כלב השמירה.

כאשר שמים '1' ב WDLCK – לא ניתן להפסיק את כלב השמירה עד לאיפוס – reset - המערכת הבא.

אם נשים 0 אז טיימר Watchdog לא נעול ואפשר להעביר אותו למצב לא מאופשר. אם נשים 1: אפשור כלב שמירה נעול ולא ניתן להפסיק את כלב השמירה.

ביט 4 – ללא תפקיד כלשהו.

ביטים 1 עד 3 CPS[2:0] – בחירת הדופק של הטיימר/קאונטר **PCA Counter/Timer Pulse Select**

בעזרת הסיביות CPS0–CPS2 בוחרים את בסיס הזמן - מה המקור ומה התדר - שיגיע לטיימר/קאונטר כפי שמוצג בטבלה הבאה :

CPS2	CPS1	CPS0	Timebase	בסיס הזמן
0	0	0	System clock divided by 12	בסיס הזמן מחולק ב 12
0	0	1	System clock divided by 4	בסיס הזמן מחולק ב 4
0	1	0	Timer 0 overflow	גלישת טיימר 0
0	1	1	High-to-low transitions on ECI (max rate = system clock divided by 4)	מעבר מגבוה לנמוך ב ECI (קצב מקסימלי = שעון המערכת לחלק ב 4
1	0	0	System clock	שעון המערכת
1	0	1	External oscillator source divided by 8*	תדר מתנד חיצוני מחולק ב 8
1	1	x	Reserved	רזרבה
Note: External oscillator source divided by 8 is synchronized with the system clock.				

טבלה 1 : בחירת בסיס הזמן של הדופק המגיע למונה

את תדר היציאה של הגל הריבועי כאשר CSP2=CSP1=CSP0=0 ניתן לקבל מהנוסחה:

$$\mathbf{PWM_{frequency} = f_{osc}/(12*256)}$$

$$\mathbf{PWM_{frequency}} = 48*10^6/(12*256) = 15625\text{Hz}$$

תדר הגביש הוא 48 מגה הרץ. הוא מחולק ב 12 ולכן מגיע למונה תדר של 4 מגה הרץ. המונה סופר כל פעם עד 256 ונותן גלישה ולכן נחלק ב 256 ונקבל 15625 הרץ.

ביט 0 – ECF - **PCA Counter/Timer Overflow Interrupt Enable** – אפשור פסיקת גלישה

סיבית זו קובעת האם לקבל פסיקת גלישה של הטיימר/קאונטר של ה PCA - Overflow Timer/Counter PCA (CF).

0: לא לאפשר את פסיקת CF. 1: פסיקת גלישת PCA/טיימר מאופשרת .

5.2 רגיסטר PCA0CN – PCA0 CoNtrol – בקרת PCA0

רגיסטר הבקרה של ה PCA הוא רגיסטר שביט אחד שלו מפעיל או מפסיק את פעול ה PCA , ביט נוסף הוא ללא שימוש ו 6 הביטים הנוספים שלו משמשים כדגל עבור פסיקת גלישה או כאשר נוצרת לכידה/השוואה באחד מ 5 המודולים. האיור הבא מתאר את הרגיסטר :

Bit	7	6	5	4	3	2	1	0
Name	CF	CR		CCF4	CCF3	CCF2	CCF1	CCF0
Type	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

איור 8 : רגיסטר **PCA0CN**

ביט 7 – CF : גלישה של טיימר / קאונטר PCA

זהו ביט המשמש כדגל והוא עובר ל 1 על ידי החומרה ומציין שטיימר/קאונטר PCA גולש מ- 0xFFFF ל - 0x0000. אם אפשרנו פסיקת גלישת PCA המיקרו בקר עובר לפונקציית השרות המטפלת בפסיקה (פסיקה מספר 11) . סיבית זו אינה מנוקה באופן אוטומטי על ידי החומרה ויש לאפס את הביט בעזרת תוכנה כדי להכין את ה PCA לשים 1 בביט בגלישה הבאה.

ביט 6 – CR : Run Control – בקרת ריצה

בקרת הפעלת מונה / טיימר PCA CR. סיבית זו מאפשרת/לא מאפשרת את מונה/קאונטר ה- PCA.

אם נשים 0 בביט : מונה / טיימר לא מאופשר, כלומר לא עובד. אם נשים 1: מונה / טיימר מופעל.

ביט 5 – ללא תפקיד.

5 הביטים מ 0 עד 4 – CCF0 עד CCF4: PCA Module n Capture/Compare Flag - דגל מודול n לכידה/השוואה.

N הוא מספר בין 0 ל 4 . לכל 5 הביטים תפקיד דומה. כל אחד מהביטים הוא עבור אחד מ 5 המודולים של הלכידה/השוואה . זהו ביט המשמש כדגל והוא עובר ל 1 על ידי החומרה כאשר מתרחשת התאמה או לכידה במודול n . אם אפשרנו פסיקה עוברים לפונקציית פסיקת PCA. סיבית זו אינה מנוקה באופן אוטומטי על-ידי החומרה ויש לאפס אותה על-ידי תוכנה.

5.3 רגיסטר PCA0CPMn - PCA0 CaptureqCompare Mode – רגיסטר אופן לכידה/השוואה של PCA0 .

במיקרו בקר יש 5 מודולים ולכל מודול יש את הרגיסטר PCA0CPMn כאשר n הוא מספר בין 0 ל 4 . בעזרת רגיסטר זה ניתן להגדיר כל מודול לפעול באופן עצמאי באחד משישה מצבי פעולה:

- מופעל על ידי קצה לכידה.
- טיימר תוכנה.
- פלט במהירות גבוהה.
- פלט תדר.
- אפנן רוחב פולס של 8 סיביות.
- אפנן רוחב פולס של 16 סיביות .

לכל מודול יש רגיסטרים משלו באזור ה SFR . אוגרים אלה משמשים להחלפת נתונים עם המודול ולהגדרת מצב העבודה/פעולה של המודול. האיור הבא מתאר כיצד נראה הרגיסטר **PCA0CPMn** .

Bit	7	6	5	4	3	2	1	0
Name	PWM16n	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

איור 9 : מבנה רגיסטר PCA0CPMn .

ביט 7 - PWM16n - אפשור אפנון רוחב פולס PWM16n של 16 סיביות .

סיבית זו מאפשרת לקבוע מצב עבודה של 16 סיביות כאשר מצב אפנון רוחב דופק – PWM - מופעל.

אם נשים בביט 0 - נבחר PWM של 8 סיביות. אם נשים 1 - נבחר PWM של 16 סיביות.

ביט 6 – ECOMn - Comparator Function Enable – אפשור פונקציית השוואה.

סיבית זו מאפשרת את פונקציית ההשוואה עבור מודול PCA מספר n (מספר בין 0 ל 4) כאשר היא מוגדרת ל- 1. כלומר אפשור המשווה במודול.

ביט 5 - CAPPn - Capture Positive Function Enable - אפשור פונקציית לכידה חיובית.

כאשר נשים '1' - סיבית זו מאפשרת את הלכידה כאשר יש קצה חיובי עבור מודול PCAn .

ביט 4 - CAPNn - Capture Negative Function Enable - אפשור פונקציית לכידה שלילית.

כאשר נשים '1' - סיבית זו מאפשרת את הלכידה כאשר יש קצה שלילי עבור מודול PCAn.

ביט 3 - MATn - Match Function Enable - אפשור פונקציית התאמה .

כאשר היא מוגדרת ל- '1' הסיבית מאפשרת את פונקציית ההתאמה עבור מודול PCAn כאשר אפשרות זו מופעלת, התאמות של מונה PCA עם אוגר לכידה/השוואה של המודול גורמות ל – CCFn (פסיקת capture/ compare).

ביט 2 - TOGn - Toggle Function Enable – אפשור פעולת toggle (לעבור ממצב למצב – החלפת מצבים) .

סיבית זו מאפשרת את פונקציית ההחלפה עבור מודול PCAn כאשר היא מוגדרת ל- 1. כאשר אפשרות זו מופעלת, התאמות של מונה PCA עם אוגר לכידה/השוואה של מודול n גורמות ללוגיקה רמה בהדק CEXn להחלפה. אם סיבית PWMn מוגדרת גם ללוגיקה 1, המודול פועל במצב יציאת תדר.

ביט 1 - PWMn - Pulse Width Modulation Mode Enable – אפשור מצב אפנון רוחב דופק – pwm .

סיבית זו מאפשרת את הפונקציה PWM עבור מודול PCA מספר n כאשר היא מוגדרת ל- 1. כאשר אפשרות זו מופעלת, אות אפנון רוחב פולס יוצא בהדק CEXn. PWM של 8 סיביות פועל אם שמנו 0 בביט PWM16n; אם שמנו 1 אז נקבל 16 סיביות של PWM. אם סיבית TOGn מוגדרת גם היא, המודול פועל במצב יציאת תדר.

ביט 0 - ECCFn - Capture/Compare Flag Interrupt Enable – אפשור קבלת פסיקה מיחידת ה PCA .

אם שמים 0 בביט ממסכים את פסיקות CCFn. אם שמים 1 מאפשרים בקשת פסיקת לכידה/השוואה של דגל כאשר CCFn מוגדר.

הערה: כאשר סיבית WDTE מוגדרת ל- 1, לא ניתן לשנות את אוגר PCA0CPM4 ומודול 4 פועל כ- טיימר כלב שמירה. כדי לשנות את התוכן של אוגר PCA0CPM4 או את הפעולה של מודול 4 יש להפסיק את טיימר כלב השמירה.

הטבלה הבאה מסכמת את הגדרות הסיביות באוגר PCA0CPMn המשמש לבחירת PCA לכידה/השוואה של מצב ההפעלה של המודול. הגדרת סיבית ECCFn באוגר PCA0CPMn מאפשרת את פסיקת CCFn של המודול.

הטבלה הבאה מתארת את הגדרת הביטים עבור המודולים של הלכידה/השוואה של PCA0.

אופן הפעולה		PCA0CPMn							
	מספר הביט	7	6	5	4	3	2	1	0
Capture triggered by positive edge on CEXn	לכידה מופעל על ידי עלייה ב CEXn	X	X	1	0	0	0	0	A
Capture triggered by negative edge on CEXn	לכידה מופעל על ידי ירידה ב CEXn	X	X	0	1	0	0	0	A
Capture triggered by any transition on CEXn	לכידה מופעל על ידי ירידה או עלייה ב	X	X	1	1	0	0	0	A
Software Timer	טיימר תוכנה	X	B	0	0	1	0	0	A
High Speed Output	יציאה במהירות גבוהה	X	B	0	0	1	1	0	A
Frequency Output	יציאת תדר	X	B	0	0	0	1	1	A
8-Bit Pulse Width Modulator	אפנן רוחב דופק של 8 ביטים	0	B	0	0	C	0	1	A
16-Bit Pulse Width Modulator	אפנן רוחב דופק של 16 ביטים	1	B	0	0	C	0	1	A

הערות:

1. X = לא משנה (אין הבדל בפעולה עבור מודול בודד אם בביט יש 1 או 0).
2. A = איפשור פסיקות עבור מודול זה (פסיקת PCA מופעלת ב- CCFn שמוגדרת ל- 1).
3. B = כאשר נשים 0, המשווה הדיגיטלי כבוי. עבור מצבי יציאה במהירות גבוהה ויציאת תדר, ההדק המשויך לא יחליף מצב (toggle). בכל אחד ממצבי PWM, פעולה זו יוצרת מחזור עבודה של 0% (היציאה = 0).
4. C = כאשר נשים בו 1 - אירוע התאמה יגרום לדגל CCFn עבור הערוץ המשויך להיות 1.

טבלה 2 : הגדרת הביטים עבור המודולים של הלכידה/השוואה של PCA0.

5.4 רגיסטר XBR1 לקביעת הדק CEXn

עד עכשיו הכרנו את ביט ה XBARE של הרגיסטר XBR1 (ביט 6 של הרגיסטר). כאשר שמים בביט זה '1' הוא מפעיל את הקרוס בר ומחבר בין הרכיבים ההיקפיים והדקי היציאה של המיקרו בקר. בתוכניות שכתבנו היינו רושמים XBR1=0x40; והיינו מאפשרים את פעולת הקרוס בר. האיור הבא מתאר את רגיסטר הקרוסבר :

Bit	7	6	5	4	3	2	1	0
Name	WEAKPUD	XBARE	T1E	T0E	ECIE	PCA0ME[2:0]		
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

איור 10 : רגיסטר XBR1

הביטים הקשורים למערך המונים המתוכנת הם הביטים 0 , 1 , 2 הנקראים PCA0ME[2:0] שהם קיצור של : **PCA0 M**odule **E**nable . ביטים אלו קובעים את ההדק של פורט 0 (מ P0.0 עד P0.3) שאליו מחובר CEX.

הטבלה הבאה מתארת הקצאה של הדקי PCA0 להדקי הרכיב.

מצב הביטים	איזה הדק של ה PCA0	הדק ברירת המחדל
000	אין הדק שמתחבר החוצה.	
001	CEX0	P0.0
010	CEX0 , CEX1	P0.0 , P0.1
011	CEX0 , CEX1 , CEX2	P0.0 , P0.1 , P0.2
100	CEX0 , CEX1 , CEX2 , CEX3	P0.0 , P0.1 , P0.2 , P0.3
101	CEX0 , CEX1 , CEX2 , CEX3	P0.0 , P0.1 , P0.2 , P0.3
110	רזרבה	
111	רזרבה	

טבלה 3 : הקצאת הדקי PCA0 להדקי המיקרו בקר

6. מצבי הפעולה של PCA0

למונה/קאונטר PCA0 יש את 8 מצבי העבודה שבטבלה הקודמת. נסביר את מצבי העבודה השונים.

6.1 מצב פעולה Edge-triggered Capture Mode - מצב לכידה המופעל על ידי EDGE – שפה

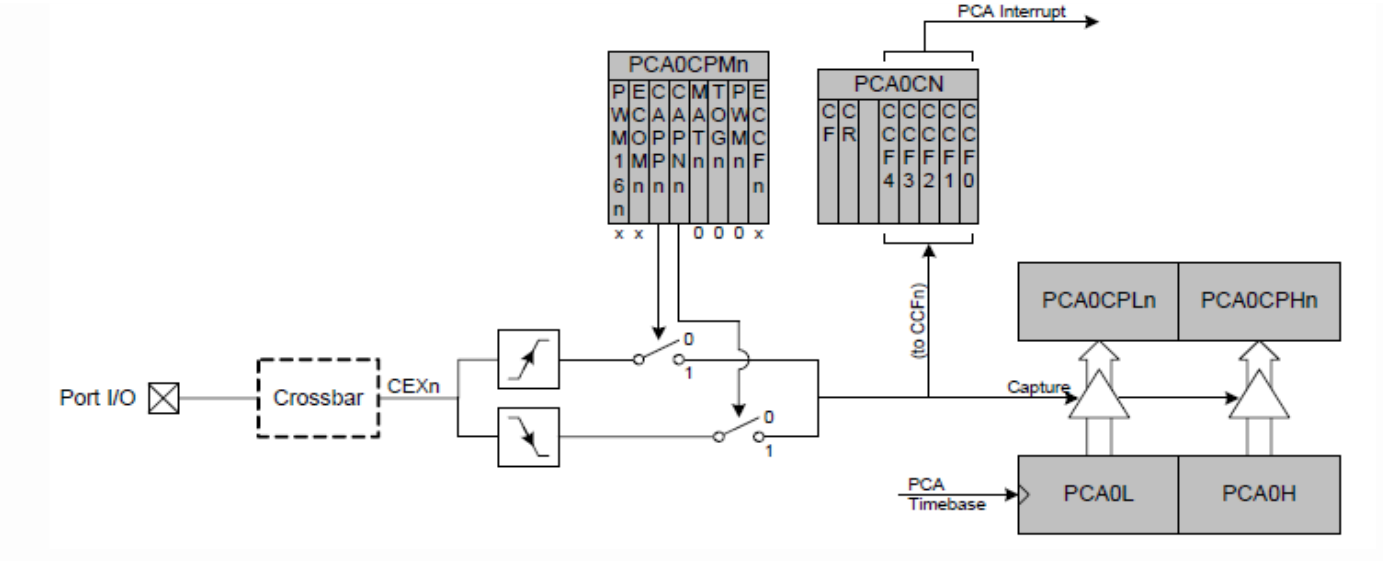
הערה - לא בתוכנית הלימודים בכתה יג

מצב פעולה זה מתאר את 3 המצבים הראשונים שבטבלה הקודמת.

במצב זה מעבר חוקי (עלייה או ירידה) בהדק CEXn גורם ל- PCA ללכוד את הערך של מונה/קאונטר PCA ולטעון אותו לאוגר לכידה/השוואה של 16 סיביות של המודול המתאים (PCA0CPLn ו- PCA0CPHn). סיביות CAPPn ו- CAPNn באוגר PCA0CPMn משמשות לבחירת סוג המעבר שמפעיל את הלכידה: מעבר מנמוך לגבוה (קצה חיובי / עלייה) או מעבר מגבוה לנמוך (קצה שלילי/ירידה) או כל אחת משתי האפשרויות (ירידה או עלייה) .

כאשר מתרחשת לכידה, דגל לכידה/השוואה (CCFn) ב- PCA0CN עובר ל 1. נוצרת בקשת פסיקה אם פסיקת CCFn עבור מודול זה מופעלת. הסיבית CCFn אינה מאופסת באופן אוטומטי על-ידי החומרה כאשר מטפלים בפסיקה ויש לאפס את הביט על ידי תוכנה. אם גם סיבית CAPPn וגם סיבית CAPNn עוברות ל אז ניתן לקרוא ישירות את מצב הדק היציאה המשויך ל- CEXn כדי לקבוע האם העלייה או הירידה גרמה ללכידה.

האיור הבא מתאר מצב לכידה של ה PCA .



איור 11 : תרשים מצב לכידה של ה PCA – PCA Capture Mode

באיור מתואר מצב לכידה של אחד המודולים מ 0 עד 4 והוא מסומן ב n . רואים את החלק הגבוה והחלק הנמוך של המונה/קאונטר PCA0 בצד ימין למטה . הוא מקבל את פולסי הספירה הנקראים באיור PCA Timebase .

מעליו רואים את רגיסטרי הלכידה PCA0CPLn (החלק הנמוך – 8 ביטים נמוכים של רגיסטר הלכידה) ואת PCA0CPHn (8 הביטים הגבוהים של רגיסטר הלכידה).

רואים שהמצב נשלט על ידי שני הרגיסטרים:

א. PCA0CPMn הקובע אם מצב הלכידה יתבצע על ידי עליה או על ידי ירידה או בעלייה וגם בירידה .

ב. PCA0CN הקובע האם נקבל פסיקה מהמודול n (מודול 0 עד 4).

באיור מצד שמאל רואים את ההדק שממנו תבוא בקשת הלכידה. ההדק מתחבר דרך הקרוס בר ונקרא CEXn . **3 הסיביות הנמוכות של XBR1 קובעות את ההדקים של פורט 0 שמשמשים כ CEXn .**

האות CEXn מבצע לכידה (capture) על ידי עלייה - אם שמנו '1' בביט CAPPn (המפסק המתאים נסגר) או על ידי ירידה אם שמנו בביט CAPNn '1' (המפסק המתאים נסגר) . אם נשים ב 2 הביטים '1' (2 המפסקים נסגרים) ונקבל לכידה גם כשיש עלייה וגם כשיש ירידה.

אות הלכידה מגיע ונקרא באיור Capture ומאפשר העברת הערך שב PCA0 לרגיסטרי הלכידה. החלק הנמוך של ה PCA לחלק הנמוך של רגיסטר הלכידה והחלק הגבוה של PCA0 לחלק הנמוך של רגיסטר הלכידה.

האות Capture מגיע גם לביט המתאים ל CCFn , שם '1' בביט המתאים וגם יוצא למערכת הפסיקות של ה PCA – באיור הוא נראה למעלה ונקרא PCA Interrupt. אם פסיקת PCA מאופשרת (פסיקה מספר 11) אז התוכנית עוברת לפונקציה המטפלת בבקשת הפסיקה.

בפונקציית הפסיקה ניתן לבדוק מאיזה רגיסטר לכידה הגיעה הפסיקה ומה הערך ברגיסטר הלכידה ובהתאמה לבצע או לא לבצע פעולות שרוצים.

הערה : כניסת אות CEXn צריכה להישאר בגבוה או בנמוך לפחות 2 מחזורי שעון מערכת כדי שהחומרה תזהה זאת.

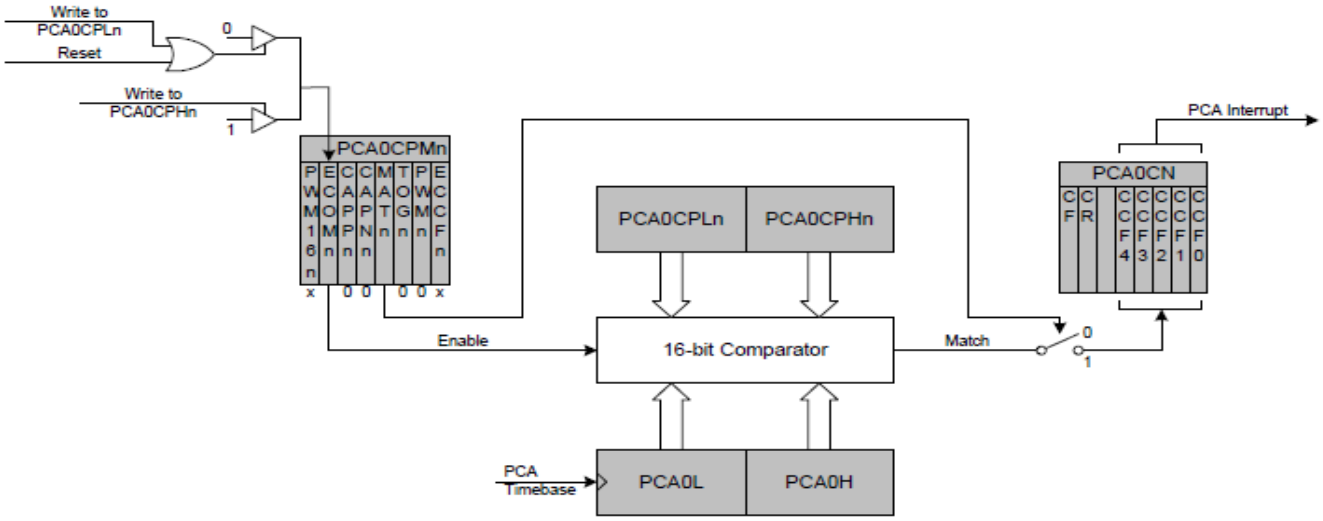
6.2 SOFTWARE TIMER – טיימר תוכנה (לא בתוכנית הלימודים)

במצב **טיימר תוכנה** ערך טיימר/קאונטר ה - PCA מושווה לערך שיש ברגיסטרי הלכידה/השוואה של 16 הסיביות של המודול הנקראים PCA0CPHn ו- PCA0CPLn.

כאשר מתרחשת התאמה, דגל לכידה/השוואה (CCFn) ב- PCA0CN עובר ל '1'. נוצרת בקשת פסיקה אם פסיקת CCFn עבור מודול זה מופעלת. סיבית CCFn אינה מתאפסת באופן אוטומטי על-ידי החומרה כאשר עוברים לפונקציית השיגרה של הפסיקה ויש לנקות אותו על ידי תוכנה.

כדי לקבל מצב **טיימר תוכנה** נשים '1' בסיביות ECOMn ו - MATn באוגר PCA0CPMn.

האיור הבא מתאר את אופן העבודה טיימר תוכנה.



איור 12 : תרשים של אופן טיימר תוכנה של ה PCA – PCA Software Timer Mode

במרכז האיור רואים משווה של 16 סיביות – 16 bit Comparator - המקבל מצידו התחתון את הערך שיש בטיימר/קאונטר PCA0 ובחלק העליון הוא מקבל את הערך שהכנסנו לרגיסטרי הלכידה/השוואה (PCA0CPLn ו PCA0CPHn) .

כדי ש PCA0 יבצע השוואה יש לשים '1' בביט ECOMn וגם '1' בביט MATn באוגר PCA0CPMn .

כאשר הערך של PCA0 שווה לערך שיש ברגיסטרים של הלכידה/השוואה נקבל '1' בקו ה Match שיוצא מהמשווה . היות והמפסק סגור כי שמנו '1' בביט MATn ונקבל '1' בביט CCFn ברגיסטר PCA0CN וגם בקשת פסיקה.

הערה חשובה לגבי לכידה/השוואה של אוגרים: בעת כתיבת ערך של 16 סיביות לרגיסטרי הלכידה/השוואה של PCA0 **יש לבצע כתיבה קודם כל לבית הנמוך ורק אחר כך לגבוה**. כתיבה ל- PCA0CPLn מנקה את סיבית ECOMn ל-0. כתיבה ל- CPHn0PCA מעבירה את ECOMn ל- 1. זה גם תפקיד החלק השמאלי למעלה באיור.

6.3 אופן יציאה מהירות גבוהה - High-Speed Output Mode (לא בתוכנית הלימודים)

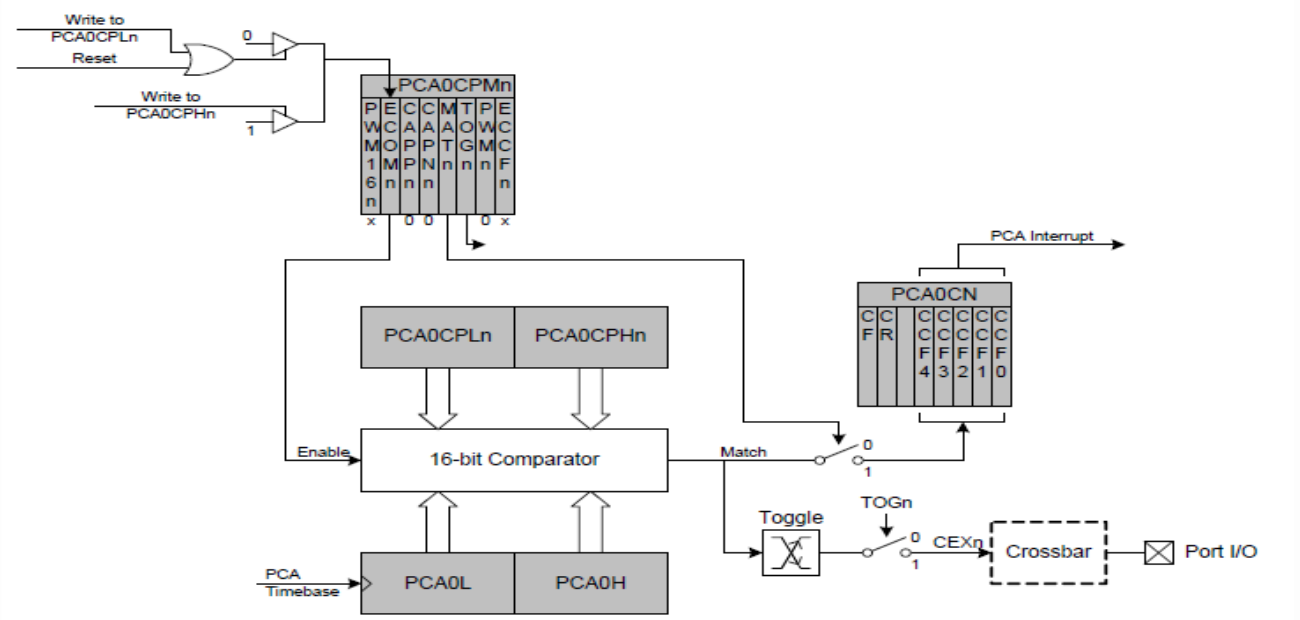
במצב יציאה במהירות גבוהה ההדק CEXn המשויך למודול מחליף את מצבו (מ 0 ל 1 ומ 1 ל 0) בכל פעם שמתרחשת התאמה בין מונה ה-PCA לבין אוגר הלכידה/השוואה של 16 סיביות של המודול (PCA0CPHn ו- PCA0CPLn). כאשר מתרחשת התאמה, דגל לכידה/השוואה (CCFn) ב- PCA0CN עובר ל '1' ונוצרת בקשת פסיקה . אם פסיקת CCFn עבור מודול זה

מופעלת. סיבית CCFn איננה מאופסת אוטומטית על-ידי החומרה כאשר עוברים לפונקציית הטיפול בפסיקה ויש לאפס אותה על ידי תוכנה.

כדי לעבוד במצב זה יש לשים '1' בסיביות TOGn, MATn ו- ECOMn באוגר PCA0CPMn.

אם ECOMn מאופסת ההדק המתאים ישמור על מצבו ולא יעבור למצב אחר.

הערה חשובה לגבי לכידה/השוואה של אוגרים: בעת כתיבת ערך של 16 סיביות לרגיסטרים של הלכידה/השוואה PCA0 הבית הנמוך תמיד צריך להיות כתוב הראשון. כתיבה ל- PCA0CPLn מנקה את סיבית ECOMn ל-. כתיבה ל- PCA0CPHn מגדירה את ECOMn ל- 1. האיור הבא מתאר תרשים של מצב פעולה זה:



איור 13 : אופן יציאה במהירות גבוהה של ה PCA - PCA High Speed Output Mode

אופן זה די דומה לאופן הקודם רק שכאן אפשר להוציא גם גל ריבועי בהדק CEXn (n מספר בין 0 ל 4 לפי המודול).

בצד ימין למטה רואים שהקו CEXn יוצא דרך הקרוס בר להדק חיצוני של המיקרו בקר.

כאשר ההערך שבמונה/קאונטר שווה לערך ברגיסטרים של הלכידה/השוואה יוצא '1' בקו Match . אם שמנו '1' בביט TOGn אז המספק נסגר ובכל התאמה דואג המלבן Toggle להחליף את המצב ובהדק Port/IO נקבל גל מרובע.

6.4 אופן יציאת תדר - Frequency Output Mode (לא בתוכנית הלימודים)

אופן יציאת תדר מייצר גל ריבועי בתדר ניתן לתכנות במודול שרוצים בהדק CEXn. הבית הגבוה של מודול לכידה/השוואה מכיל את מספר פולסי הספירה שה-PCA צריך לספור לפני שביציאה יש החלפת מצב. תדר הגל הריבועי מוגדר על ידי המשוואה :

$$F_{CEXn} = \frac{F_{PCA}}{2 \times PCA0CPHn}$$

משוואה 1: תדר הגל הריבועי באופן יציאת תדר

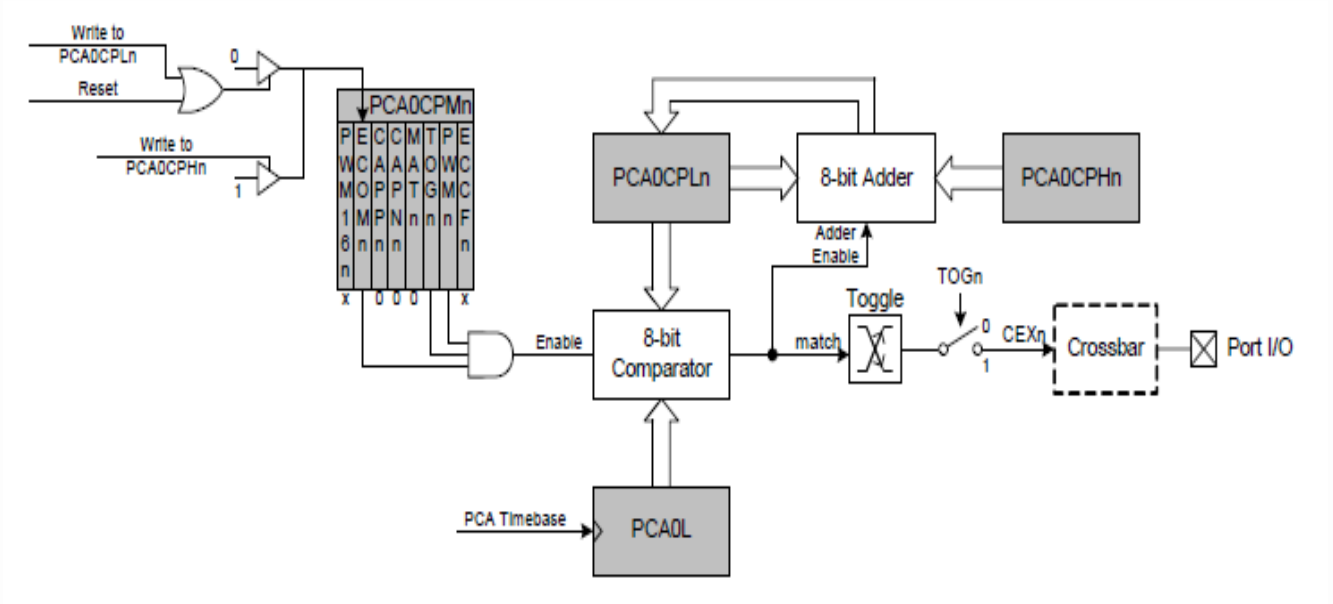
F_{CEXN} – התדר בהדק היציאה CEXn

F_{PCA} - התדר המגיע לספירה ל PCA0 ונקבע על ידי הביטים CPS2:0 ברגיסטר PCA0MD (רגיסטר האופן של PCA0).

PCA0CPHn – הערך בחלק הגבוה של רגיסטר ה לכידה/השוואה.

הערה: ערך של 0x00 באוגר PCA0CPHn שווה ל- 256 עבור משוואה זו.

האיור הבא מתאר תרשים של אופן יציאת תדר:



איור 14 : אופן יציאת תדר - PCA Frequency Output Mode

הביית הנמוך של מודול הלכידה/השוואה מושווה לביית הנמוך של ה PCA . כאשר יש התאמה הדק CEXn מחליף מצב וההיסט המוחזק בבית הגבוה של ה PCA מתווסף לערך התואם ב- PCA0CPLn.

לדוגמה: אם קבענו ש PCA0CPHn=50 אז החלפת מצב ראשונה תהיה כאשר החלק הנמוך של רגיסטר הלכידה/השוואה יהיה 50 וגם כאשר בחלק הנמוך של טיימר/קאונטר של PCA0 יגיע ל 50 . כאשר יש התאמה המלבן 8-bit adder יחבר את הערך 50 של PCA0CPLn עם הערך 50 שב PCA0CPHn ויעביר ל PCA0CPLn את הערך 100 . בינתיים המונה סופר וכאשר החלק הנמוך שלו יגיע ל 100 נקבל התאמה . שוב המלבן adder מחבר את PCA0CPLN+PCA0CPHn ויעביר 150 ל PCA0CPLn וכך הלאה.

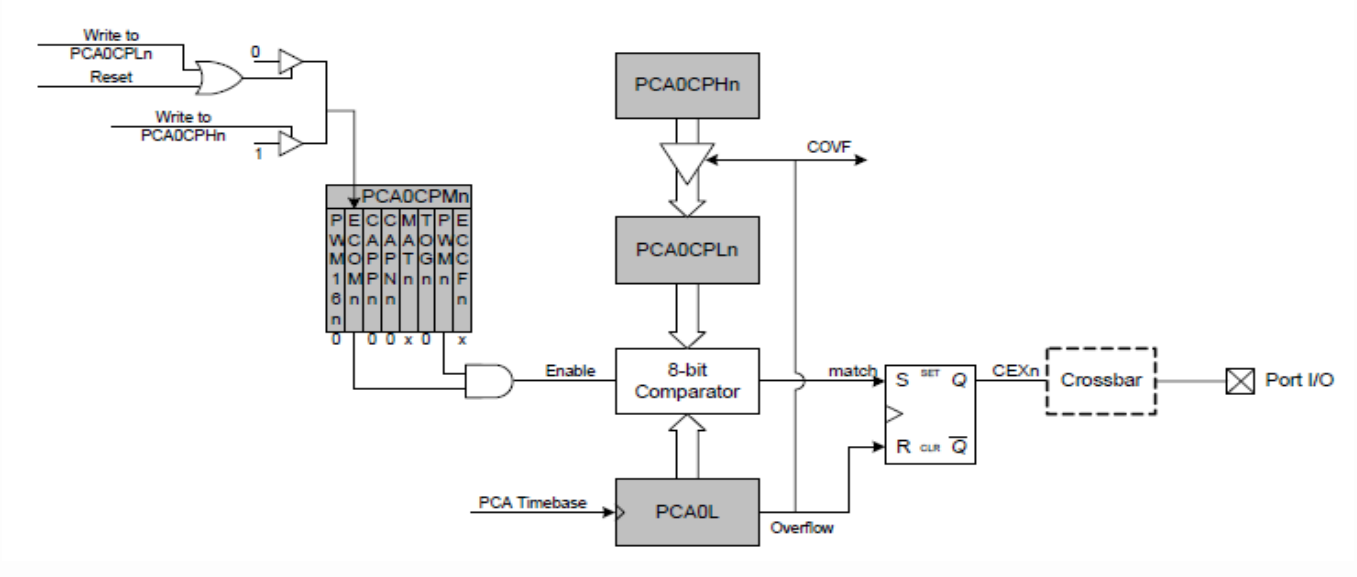
מצב יציאת תדר מופעל על-ידי כך שנשים '1' בסיביות ECOMn, TOGn ו- PWMn באוגר PCA0CPMn.

יש לשים לב שבדרך כלל יש להגדיר את סיבית MATn ל- 0 במצב זה. אם סיבית MATn מוגדרת ל- 1 אז דגל ה- CCFn עבור הערוץ יעלה ל '1' כאשר מונה PCA0 של 16 סיביות ורגיסטר הלכידה/השוואה של 16 הסיביות יהיו שווים .

6.5 אופן אפנן רוחב דופק של 8 ביטים - 8-bit Pulse Width Modulator Mode

מחזור העבודה – Duty Cycle - של אות הפלט PWM במצב PWM של 8 סיביות משתנה בעזרת רגיסטר - PCA0CPLn הביית הנמוך של רגיסטר ההשוואה/לכידה. כאשר הערך ב PCA0L - הבית הנמוך של טיימר/קאונטר PCA שווה ל ערך ב PCA0CPLn (הביית הנמוך של רגיסטר ההשוואה/לכידה) - הדק היציאה CEXn יקבל 1 . כאשר ערך הספירה ב- PCA0L גולש, הדק CEXn יקבל 0 .

האיור הבא מתאר את אופן העבודה של אפנון רוחב דופק של 8 ביטים.



איור 15 : אופן עבודה PWM של 8 ביטים.

באיור במרכז למטה רואים את PCA0L המקבל מצד שמאל שלו את פולסי הספירה. המשווה שמעליו (8 bit Comparator) מקבל בחלק התחתון שלו את הערך של הביית הנמוך של הטיימר/קאונטר (PCA0L) ובחלקו העליון את הערך שיש בביית הנמוך של רגיסטר PCA0CPLn .

כאשר יש שוויון בין 2 ערכים אלו יוצא בקו match פקודה להדק set של S-R F.F ומעבירה את הדק Q של ה FF ל '1' . הדק זה נקרא באיור CEXn ומתחברים דרך הקרוסבר להדק חיצוני שנקרא Port I/O (אחד ההדקים שנבחר בפורט 0).

הטיימר/קאונטר ממשיך לספור וכאשר הוא גולש (עובר מ 0xFF ל 0x00 ומתחיל לספור מהתחלה.) קו הגלישה - Overflow שביציאה שלו מגיע לרגל R של S-R F.F , שדואג ל '0' ביציאת ה Q שלו. כמו כן הקו עולה למעלה ונותן פקודת טעינה ל PCA0CPLn שנטען מחדש באופן אוטומטי לערך המאוחסן ב PCA0CPHn (למעלה במרכז האיור) שהוא הביית הגבוה של רגיסטר הלכידה/השוואה של המודול.

דוגמה : בהנחה שטענו את PCA0CPHn לערך 128 .

כאשר המונה מגיע ל 128 נקבל match והדק Q יקבל '1' ודרך הקרוס בר נקבל '1' בהדק היציאה שבחרנו (אחד מהדקי פורט 0).

המונה ממשיך לספור עוד 128 פולסים ואז גולש overflow ונותן פקודה ברגל R של ה FF ונקבל 0 בהדק היציאה.

128 פולסים היציאה הייתה ב '1' ו 128 פולסים היציאה הייתה ב '0' וקיבלנו Duty Cycle של 50% .

דוגמה נוספת : אם הערך ששמנו ב PCA0CPHn שווה ל 64 אז מ 0 עד 64 היציאה תהיה ב 0 ומ 64 עד 256 היא תהיה ב 1 .

קיבלנו Duty Cycle של 75% .

הנוסחה לקבלת Duty Cycle רצוי היא : (כדי לקבל באחוזים נכפיל ב 100%).

$$\text{Duty Cycle} = \frac{(256 - \text{PCA0CPHn})}{256}$$

משוואה 2 : משוואת ה Duty Cycle ב 8 bit PWM

כדי לעבוד באופן אפנון רוחב דופק של 8 ביטים נשים '1' בסיביות ECOMn ו - PWMn ב- אוגר PCA0CPMn . אם סיבית MATn מוגדרת ל- 1 דגל CCFn עבור המודול יקבל '1' בכל פעם שמתרחשת התאמה במשוואה של 8 הסיביות (קצה עולה).

6.6 אופן 16-Bit Pulse Width Modulator Mode - אפנן רוחב דופק של 16 סיביות

ניתן להפעיל מודול PCA גם במצב PWM של 16 סיביות. האיור הבא מתאר את אופן זה.

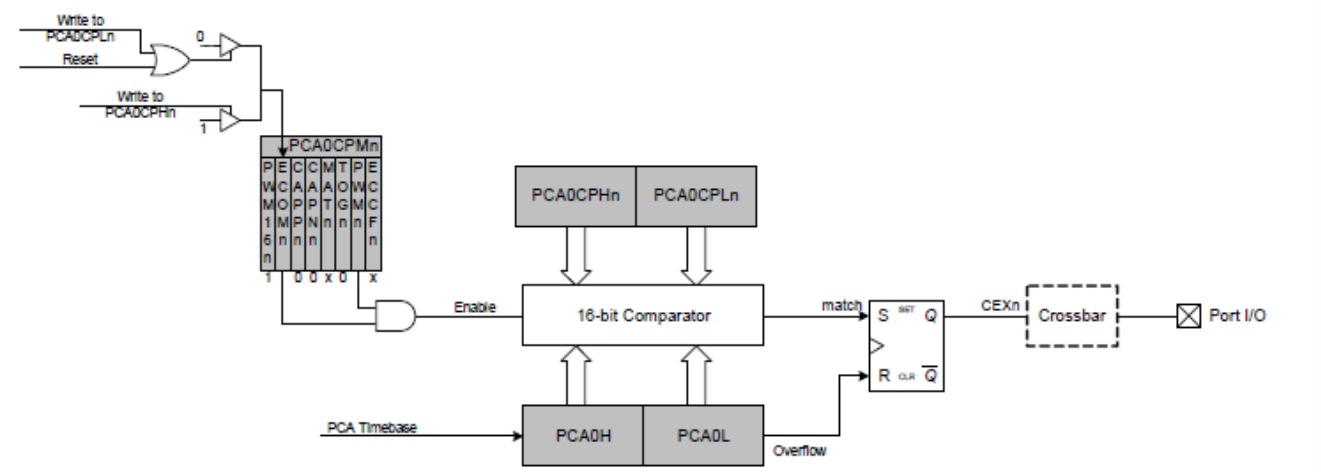


Figure 27.9. PCA 16-Bit PWM Mode

איור 16 : אופן PCA 16 bit PWM

האיור מתאר אופן פעולה די דומה לאופן הקודם של ה PWM של 8 ביטים.

כאן מתבצעת השוואה בין הערך של 16 הביטים של טיימר/קאונטר PCA0 והערך שנטען ל 16 הביטים של רגיסטר הלכידה/השוואה PCA0CPn . כאשר יש שוויון בין ערכים אלו יוצא אות בקו match המתחבר להדק s של SR FF . הדבר גורם ל '1' ביציאת CEXn היוצא להדק חיצוני שהוא אחד מהדקי פורט 0 של המיקרו בקר .

בהמשך הספירה כאשר המונה יגיע ל 0xFFFF ויעבור ל 0x0000 נקבל גלישה שמגיע להדק R של ה SR FF ודואגת ל '0' ביציאת ה Q שלו ולאפס בהדק החיצוני שבחרנו בפורט 0 .

הנוסחה לחישוב ה Duty Cycle היא : (כדי לקבל באחוזים נכפיל ב 100%).

$$\text{Duty Cycle} = \frac{(65536 - \text{PCA0CPn})}{65536}$$

משוואה 3 : משוואת ה Duty Cycle ב 16 bit PWM

דוגמאות :

אם רוצים PWM של 50% נטען את PCA0CPn בערך **32768** . 65536/2 = PCA0CPH0 = 0x80, PCA0CPL0=0x00

אם רוצים PWM של 75% נטען את PCA0CPn בערך **16384** . 65536/4= PCA0CPH0 = 0x40, PCA0CPL0=0x00

אם רוצים PWM של 25% נטען את PCA0CPn בערך **49152** . PCA0CPH0 = 0xC0, PCA0CPL0=0x00

ה Duty Cycle הנמוך ביותר הוא כאשר PCA0CPn = 65535 ואז נקבל :

Duty Cycle = (65536-65535)/65536 = 0.0000152587890625 = 0.00152587890625%

עבור PCA0CPn = 0 נקבל Duty Cycle של 100% .

כדי לקבל את אופן PWM של 16 סיביות נשים '1' בסיביות ECOMn , PWMn ו- PWM16n ב- רגיסטר PCA0CPM0.
עבור מחזור עבודה משתנה, יש להפעיל פסיקות התאמה (ECCFn = 1 ו- MATn = 1) כדי לסייע בסנכרון פעולות הכתיבה של האוגר לכידה/השוואה. אם סיבית MATn מוגדרת ל- 1, דגל CCFn עבור המודול יעלה ל '1' בכל פעם שתתרחש התאמה במשווה של 16 סיביות (קצה עולה). ניתן להשתמש בדגל CF ב- PCA0CN כדי לזהות את הגלישה .

הערה חשובה: גם כאן , בעת כתיבת ערך של 16 סיביות לאוגר הלכידה/השוואה PCA0 יש לכתוב קודם כל לבית הנמוך שלו ורק אחר כך לביית הגבוה שלו. כתיבה ל- PCA0CPLn מאפסת את סיבית ECOMn ל-0; כתיבה ל- PCA0CPHn שמה '1' ב ECOMn .

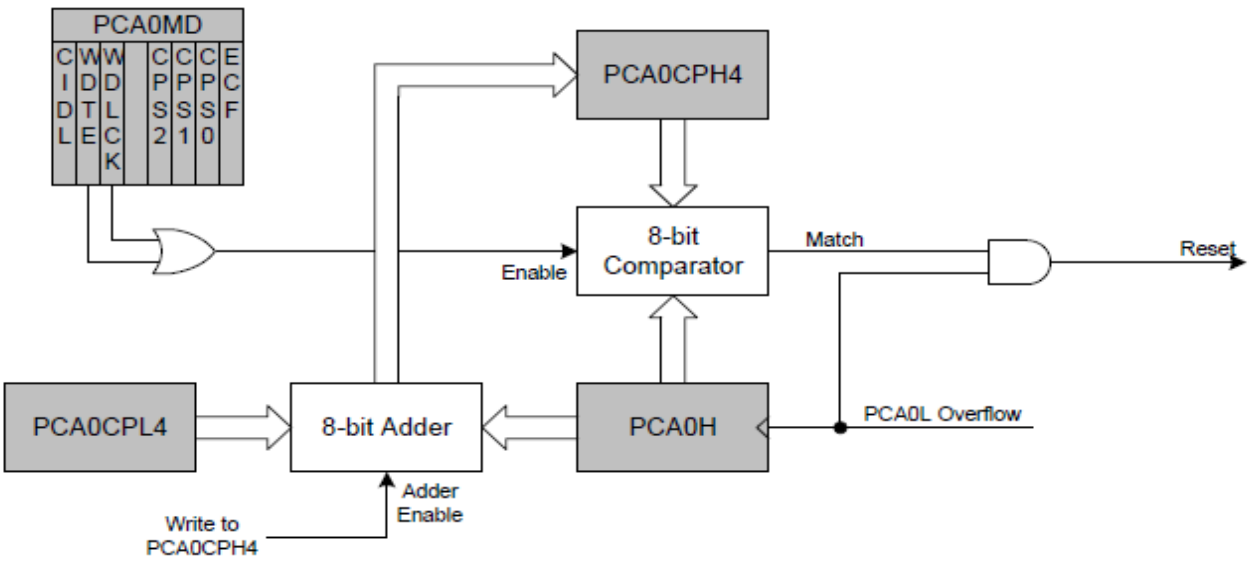
6.7. אופן Watchdog Timer Mode - טיימר כלב שמירה - WDT (לא בתוכנית הלימודים)

6.7.א כללי :

כלב שמירה הוא טיימר שניתן לתכנות והוא משמש לאיתור והתאוששות מתקלות מחשב. טיימרים של כלב שמירה נמצאים בשימוש נרחב במחשבים כדי להקל על תיקון אוטומטי של תקלות חומרה זמניות, וכדי למנוע מתוכנה שגויה או מרושעת (וירוס...) לשבש את פעולת המערכת.
במהלך פעולה רגילה, המחשב מפעיל מחדש באופן קבוע את הטיימר של כלב השמירה כדי למנוע ממנו לסיים את הספירה שלו, כלומר, "לסיים את הזמן הקצוב". אם עקב תקלת חומרה או שגיאה בתוכנית או סתם "התברברות" של התוכנית כתוצאה מרעש או כל סיבה שהיא , המחשב אינו מצליח להפעיל מחדש את כלב השמירה – טיימר כלב השמירה יסיים את הספירה שלו וייצור אות פסק זמן או פסיקה או RESET. האות או הפסיקה או ה RESET משמשים ליזום פעולות תיקון. הפעולות המתקנות כוללות בדרך כלל הצבת המחשב והחומרה המשויכת אליו במצב בטוח והפעלת אתחול מחדש של המחשב (RESET).

6.7.ב טיימר כלב השמירה במיקרו בקר שלנו :

הטיימר במיקרו בקר שלנו המשמש ככלב השמירה הוא רגיסטר לכידה/השוואה של מודול 4 - PCA0CPH4 . אם הוא עובר את הזמן שקבענו לו כגבול הוא נותן RESET והמיקרו בקר מתחיל את התוכנית מהתחלה.
ניתן להגדיר את ה-WDT ולהפעיל או להפסיק אותו לפי הצורך.
הערה: בהרבה תוכניות תרגול שכותבים סטודנטים שעדיין לא שולטים בטיימר כלב השמירה , כדאי בתחילת כל תוכנית להפסיק את הפעולה של טיימר כלב השמירה כדי לא לקבל פעולות reset לא רצויות בהרצת התוכנית.
האיור הבא מתאר את אופן כלב השמירה :



איור 17 : מודול 4 של ה PCA עם אפשור כלב שמירה

כאשר נשים '1' בסיבית WDTE – Watch Dog Timer Enable - אפשור טיימר כלב השמירה באוגר PCA0MD - מודול 4 פועל כטיימר כלב שמירה (WDT). ה '1' בביט זה עובר את שער ה OR ומגיע לקו Enable של המשווה ומאפשר לו פעולת השוואה. ניתן גם לשים '1' בביט הנעילה - WDLCK . סיבית זו נועלת או לא נועלת את אפשור כלב השמירה. כאשר שמים '1' בביט לא ניתן להפסיק את כלב השמירה עד לאיפוס – reset - המערכת הבא.

אם נשים 0 אז טיימר Watchdog לא נעול ואפשר להעביר אותו למצב לא מאופשר. אם נשים 1: אפשור כלב שמירה נעול ולא ניתן לבטל אותו עד לפעולת RESET .

גם כאן מתבצעת השוואה על ידי מלבן 8-bit Comperator בין הביית הגבוה של טיימר/קאונטר PCA0H ובין הביית הגבוה של רגיסטר הלכידה/השוואה של מודול 4 הנקרא PCA0CPH4 .

הבית הגבוה של מודול הלכידה/השוואה מושווה לביית הגבוה של ה PCA0 . כאשר יש התאמה , כלומר טיימר כלב השמירה עבר את הזמן שהוקצב לו יוצא מהמשווה בקו Match '1' וכאשר תהיה גלישה בביית הנמוך של PCA0 נקבל '1' גם בכניסה התחתונה של שער ה AND ונקבל ביציאת שער ה AND '1' שיגרום לפעולת RESET .

הבית הנמוך של מודול 4 מחזיק את ההיסט המשמש בעת ביצוע עדכוני WDT. טיימר ה Watchdog מופעל בעת איפוס.

יש הגבלה בכתיבה לרגיסטרים של ה PCA בזמן שטיימר ה Watchdog מופעל כי ה-WDT ייצור איפוס RESET זמן קצר לאחר תחילת ביצוע הקוד. כדי להימנע מאיפוס זה, יש להפסיק את אפשור ה-WDT במפורש (ובאופן אופציונלי לקנפג אותו מחדש ולאפשר אותו מחדש אם נעשה בו שימוש במערכת).

6.7.ג פעולת טיימר כלב שמירה :

כאשר WDT מופעל :

- PCA מועבר למצב ON .
- אסור לבצע כתיבה ל PCA0L ו PCA0H .

- הסיביות של מקור השעון CPS2–CPS0 לא פעילות.
- ביט בקרת הסרק (CIDL) של ה PCA לא פעיל.
- מודול 4 מאולץ לאופן טיימר תוכנה.
- לא ניתן לבצע כתיבה לרגיסטרי הלכידה/השוואה (PCA0CPM4) של רגיסטר 4 .

כאשר WDT מאופשר, הכתיבה לסיבית CR לא תשנה את מצב מונה ה- PCA. המונה יפעל עד להפסקת ה-WDT. סיבית CR - בקרת ריצת מונה PCA תקרא אפס אם ה- WDT מופעל אך תוכנת המשתמש לא הפכה את מונה ה- PCA לזמין.
אם מתרחשת התאמה בין PCA0CPH4 ל-PCA0H בזמן שה WDT מאופשר, ייווצר RESET. כדי למנוע RESET של WDT, יש לעדכן כל ערך ל PCA0CPH4.
בכתיבה ל PCA0CPH4 אז הערך ב PCA0H בתוספת הקיזוז/הערך המוחזק ב-PCA0CPL4 נטען לתוך PCA0CPH4.
ההיסט של 8 סיביות המוחזק ב- PCA0CPH4 מושווה לבית העליון של מונה PCA של 16 סיביות. ערך קיזוז זה הוא מספר גלישות PCA0L לפני RESET. עד 256 פולסי ספירה של PCA יכולים לחלוף לפני שגלישת PCA0L מתרחשת, בהתאם לערך של PCA0L בעת ביצוע העדכון.
הקיזוז הכולל לאחר מכן (בשעוני PCA) נתון על ידי המשוואה הבאה כאשר PCA0L הוא הערך של אוגר PCA0L בזמן העדכון.

$$\text{Offset} = (256 \times \text{PCA0CPL4}) + (256 - \text{PCA0L})$$

משוואת היסט טיימר של כלב שמירה בשעוני PCA - **Watchdog Timer Offset in PCA Clocks**
איפוס WDT נוצר כאשר יש גלישה ב PCA0L ויש התאמה בין PCA0CPH4 ו- PCA0H. בעזרת תוכנה אפשר לבצע RESET על ידי כתיבה של '1' לדגל CCF4 - (4.PCA0CN) בזמן שה-WDT מאופשר.

6.7.ד השימוש בטיימר כלב שמירה

כדי לקבוע את תצורת WDT יש לבצע את המשימות הבאות:

1. הפסקת ה- WDT על ידי כתיבת 0 לסיבית WDTE.
2. בחירת מקור שעון ה- PCA הרצוי (עם סיביות CPS0-CPS2).
3. לטעון ל PCA0CPL4 את ערך היסט עדכון WDT הרצוי.
4. להגדיר את מצב Idle של ה PCA . (לשים '1' בביט CIDL אם יש לעצור את ה- WDT כאשר ה מיקרו נמצא במצב Idle .
5. להפעיל את ה- WDT על ידי הגדרת סיבית WDTE ל- 1. 6
6. לעשות RESET של טיימר WDT על ידי כתיבה ל- PCA0CPH4.

לא ניתן לשנות את מקור שעון ה-PCA ואת הבחירה במצב סרק – Idle - כאשר ה-WDT מופעל. טיימר כלב השמירה מופעל על-ידי השמה של '1' לסיביות WDTE או WDLCK באוגר PCA0MD. כאשר נשים '1' בביט WDLCK לא ניתן לא לאפשר

את WDT עד ל RESET המערכת הבא. אם שמנו 0 בביט WDLCK ניתן להפסיק את פעולת טיימר כלב השמירה על ידי '0' בביט WDTE.

טיימר כלב השמירה מופעל לאחר כל RESET . ברירת המחדל של פולסי הספירה למונה PCA0 היא שעון המערכת חלקי 12, ברירת המחדל של PCA0L היא 0x00 וברירת המחדל של PCA0CPL4 היא 0x00. בעזרת משוואת היסט טיימר של כלב שמירה בשעוני PCA שבעמוד קודם, התוצאה היא WDT במרווחי זמן קצובים של 256 מחזורי שעון PCA או 3072 מחזורי שעון מערכת. הטבלה הבאה מפרטת כמה דוגמאות למרווחי פסק זמן עבור שעוני מערכת טיפוסיים :

System Clock (Hz)	PCA0CPL4	Timeout Interval (ms)
48,000,000	255	16.4
48,000,000	128	8.3
48,000,000	32	2.1
12,000,000	255	65.5
12,000,000	128	33.0
12,000,000	32	8.4
1,500,000[2]	255	524.3
1,500,000[2]	128	264.2
1,500,000[2]	32	67.6
32,768	255	24,000
32,768	128	12,094
32,768	32	3,094

טבלה 4 : מרווחי זמן עבור טיימר כלב השמירה.

התוצאות בטבלה הם בהנחה שתדר פולסי הספירה ל PCA הוא SYSCLK/12 וערך PCA0L בזמן העדכון הוא 0x00.

העמודה השמאלית בטבלה היא תדר השעון של המערכת הנקרא SYSCLK .

העמודה המרכזית היא הערך שנשים ב PCA0CPL4 (רגיסטר לכידה/השוואה של מודול 4).

העמודה הימנית היא מרווח הזמן של טיימר כלב השמירה מרגע העדכון ועד קבלת RESET אם אין עדכון חדש.

דוגמאות :

בשורה הראשונה רואים שעבור שעון מערכת SYSCLK של 48 מגה הרץ וטעינת הערך 255 ל PCA0CPL4 נקבל מרווח זמן של 16.4 מילי שניות.

בורה האחרונה רואים שאם SYSCLK 32,768 הרץ וטענו ערך של 32 ל PCA0CPL4 נקבל מרווח זמן של 3094 מילי שניות שהם 3.094 שניות.

7. דוגמאות לעבודה ב PWM

7.1 תוכנית להפעלת PWM של 8 ביטים

נרשום תכנית להפעלת המונה בר התכנות - PCA0 לעבודה עם PWM של 8 ביטים לקבלת גל מרובע בתדר 15.625 קילו הרץ עם Duty Cycle של 50%. לשם כך נעבוד עם מודול 0 של ה PCA .

כדי לקבל תדר כזה נעשה זאת על ידי החישוב הבא :

- תדר שעון המערכת הוא 48 מגה הרץ. נחלק אותו ב 12 ונקבל 4 מגה הרץ .
- במצב 8-bit PWM הטיימר/קאונטר סופר 256 פולסים ולכן התדר יהיה : $4*10^6 / 256 = 15625$ Hz.
- כדי לקבל Duty Cycle של 50% ניעזר בנוסחה :

$$\text{Duty Cycle} = \frac{(256 - PCA0CPHn)}{256}$$

ורואים שצריך לטעון לרגיסטר PCA0CPH0 את הערך 128 .

כדי לקבל Duty Cycle של 75% נטען לרגיסטר PCA0CPH0 את הערך 64 . לקבלת 25% : PCA0CPH0 =192 .

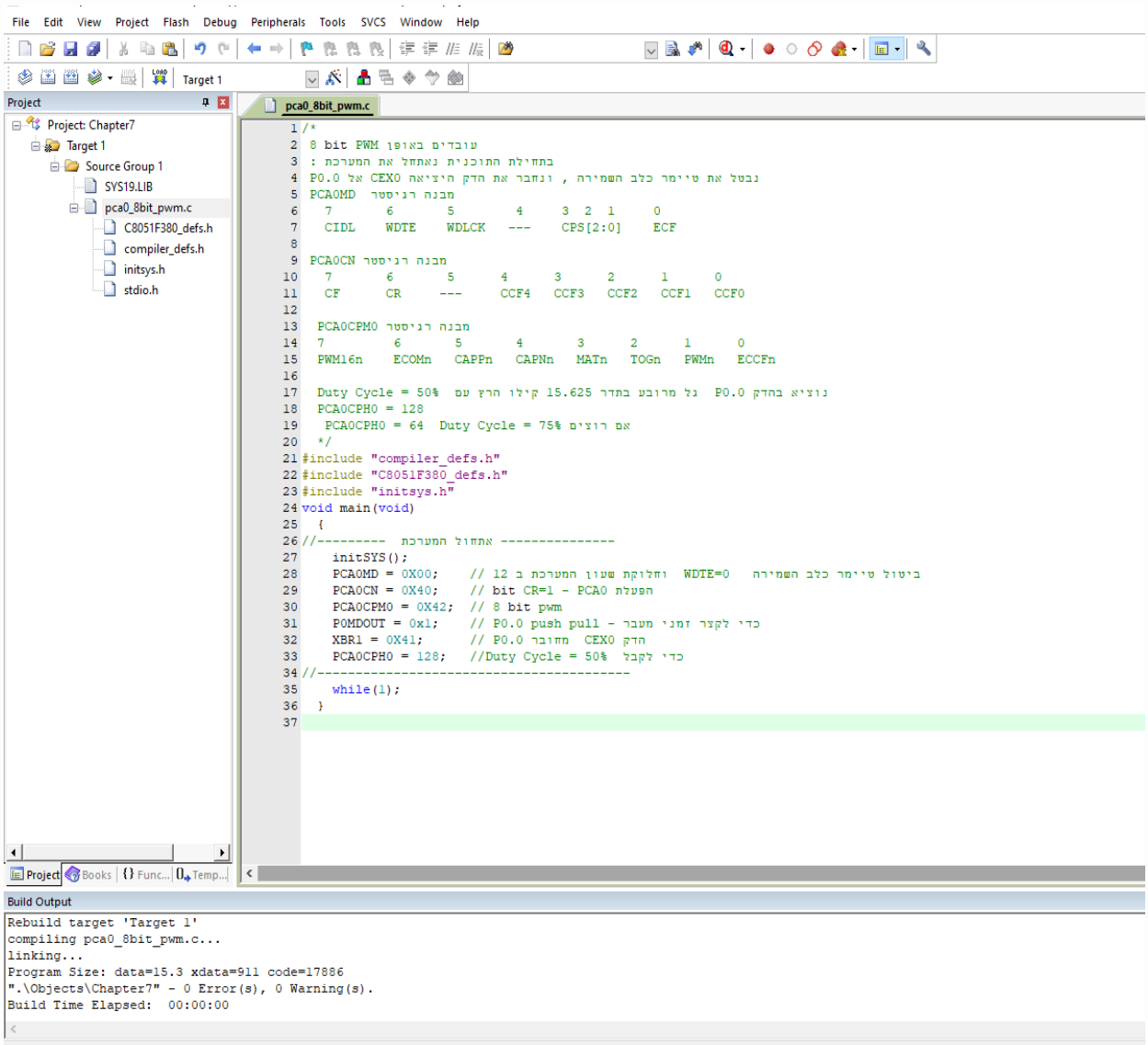
התוכנית נראית כך (נעזרנו בקבצי ה include של חברת "שיא מערכות" :

```
/*
8 bit PWM עובדים באופן
בתחילת התוכנית נאתחל את המערכת :
נבטל את טיימר כלב השמירה , ונחבר את הדק היציאה CEX0 אל P0.0
PCA0MD מבנה רגיסטר
 7     6     5        4    3  2  1    0
   CIDL   WDTE   WDLCK  ---   CPS[2:0]   ECF

PCA0CN מבנה רגיסטר
 7     6     5        4    3    2    1    0
   CF    CR    ---   CCF4  CCF3  CCF2  CCF1  CCF0

  PCA0CPM0 מבנה רגיסטר
  7      6     5    4    3    2    1    0
  PWM16n  ECOMn  CAPPn  CAPNn  MATn  TOGn  PWMn  ECCFn

Duty Cycle = 50%  נוציא בהדק P0.0 גל מרובע בתדר 15.625 קילו הרץ עם
PCA0CPH0 = 128
```

```
  PCA0CPH0 = 64          Duty Cycle = 75% אם רוצים
/*
 8 bit PWM עובדים באופן
 : בתחילת התוכנית נאתחל את המערכת
 P0.0 אל CEX0 נבטל את טיימר כלב השמירה , ונחבר את הדק היציאה
 PCA0MD  מבנה רגיסטר
      7         6           5            4      3  2  1        0
     CIDL    WDTE    WDLCK   ---    CPS[2:0]    ECF

 PCA0CN מבנה רגיסטר
     7       6       5       4        3         2         1         0
     CF      CR     ---    CCF4   CCF3   CCF2   CCF1   CCF0

    PCA0CPM0 מבנה רגיסטר
    7              6           5           4           3         2         1          0
    PWM16n    ECOMn   CAPPn   CAPNn   MATn   TOGn   PWMn   ECCFn

  Duty Cycle = 50%  גל מרובע בתדר 15.625 קילו הרץ עם  P0.0 נוציא בהדק
  PCA0CPH0 = 128
   PCA0CPH0 = 64          Duty Cycle = 75% אם רוצים
  */
#include "compiler_defs.h"
#include "C8051F380_defs.h"
#include "initsys.h"
void main(void)
    {
//---------  אתחול המערכת ---------------
            initSYS();
            PCA0MD = 0X00;    // 12 ב וחלוקת שעון המערכת  WDTE=0   ביטול טיימר כלב השמירה
            PCA0CN = 0X40;    // bit CR=1 - PCA0 הפעלת
            PCA0CPM0 = 0X42;  //   8 bit pwm
            P0MDOUT = 0x1;    // P0.0 push pull - כדי לקצר זמני מעבר
            XBR1 = 0X41;      // P0.0 מחובר  CEX0 הדק
            PCA0CPH0 = 128;   //Duty Cycle = 50%  כדי לקבל
```

```
//-----------------------------------------
        while(1);
}
```

בתוכנת ה µvision זה נראה כך :

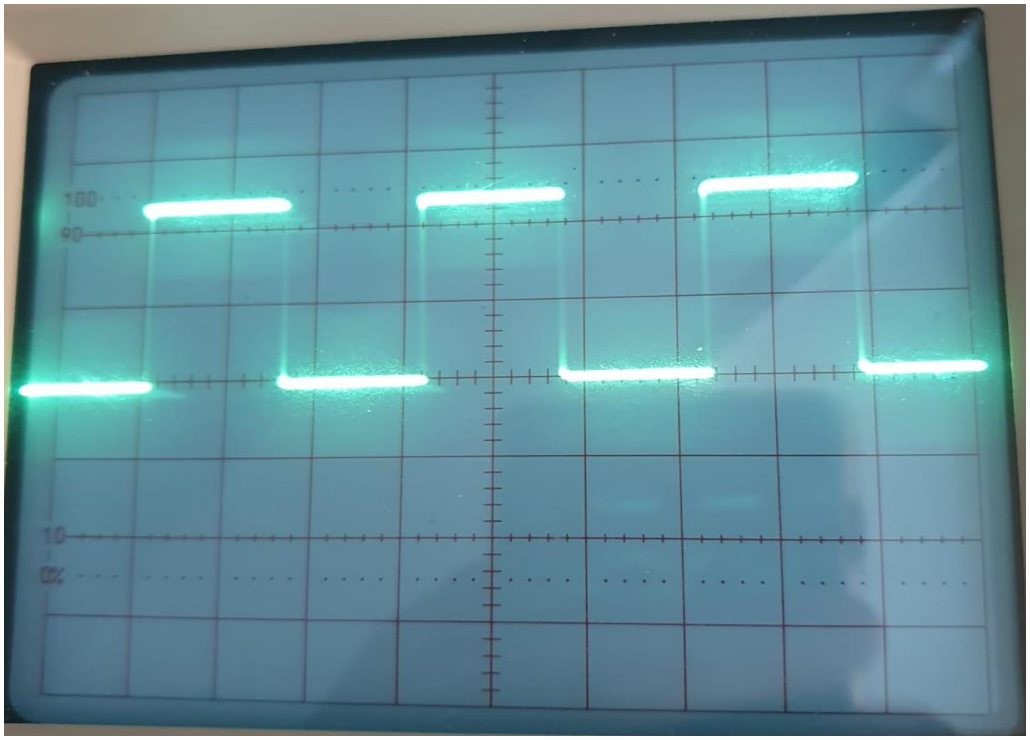


איור 18 : התוכנית 8 bit PWM בתוכנת µvision

האיור הבא מתאר את התמונה שקיבלנו במשקף התנודות.

מפסק ה VOLTS/DIV = 1v

מפסק TIME/DIV = 20µSEC



איור 19 : צורת הגל המתקבלת ב 8-bit PWM

רואים שקיבלנו אמפליטודה של כ 2.4 וולט ומחזור של 3.2 משבצות כלומר $3.2 * 2*10^{6} = 64*10^{-6}$.

מכאן שהתדר של הגל :

$f = 1 / 64*10^{-6} = 15625Hz = 15.625Hz$

כמו כן ניתן לראות ש Duty Cycle = 50% כי $T_{ON} = T_{OFF}$

7.2 תוכנית להפעלת PWM של 16 ביטים

התוכנית די דומה לתוכנית שמעליה . ההבדלים יהיו :

- יש לקבוע בעזרת PCA0MD מצב עבודה של 16-bit PWM (ביט ה MSB של הרגיסטר), כלומר :

```
PCA0CPM0 = 0XC2;  // 16 bit pwm
```

- תדר ה PWM משתנה ויהיה לפי החישוב הבא :

  תדר שעון המערכת הוא 48 מגה הרץ. נחלק אותו ב 12 ונקבל 4 מגה הרץ .

- במצב16-bit PWM הטיימר/קאונטר סופר 65536 פולסים ולכן התדר של ה PWM יהיה כ 61 הרץ לפי החישוב :

  $4*10^{6} / 65536 = 61.035$ Hz

- כדי לקבל Duty Cycle של 50% הערך שנעביר ל PCA0CP0 יהיה 32768 לפי הנוסחה :

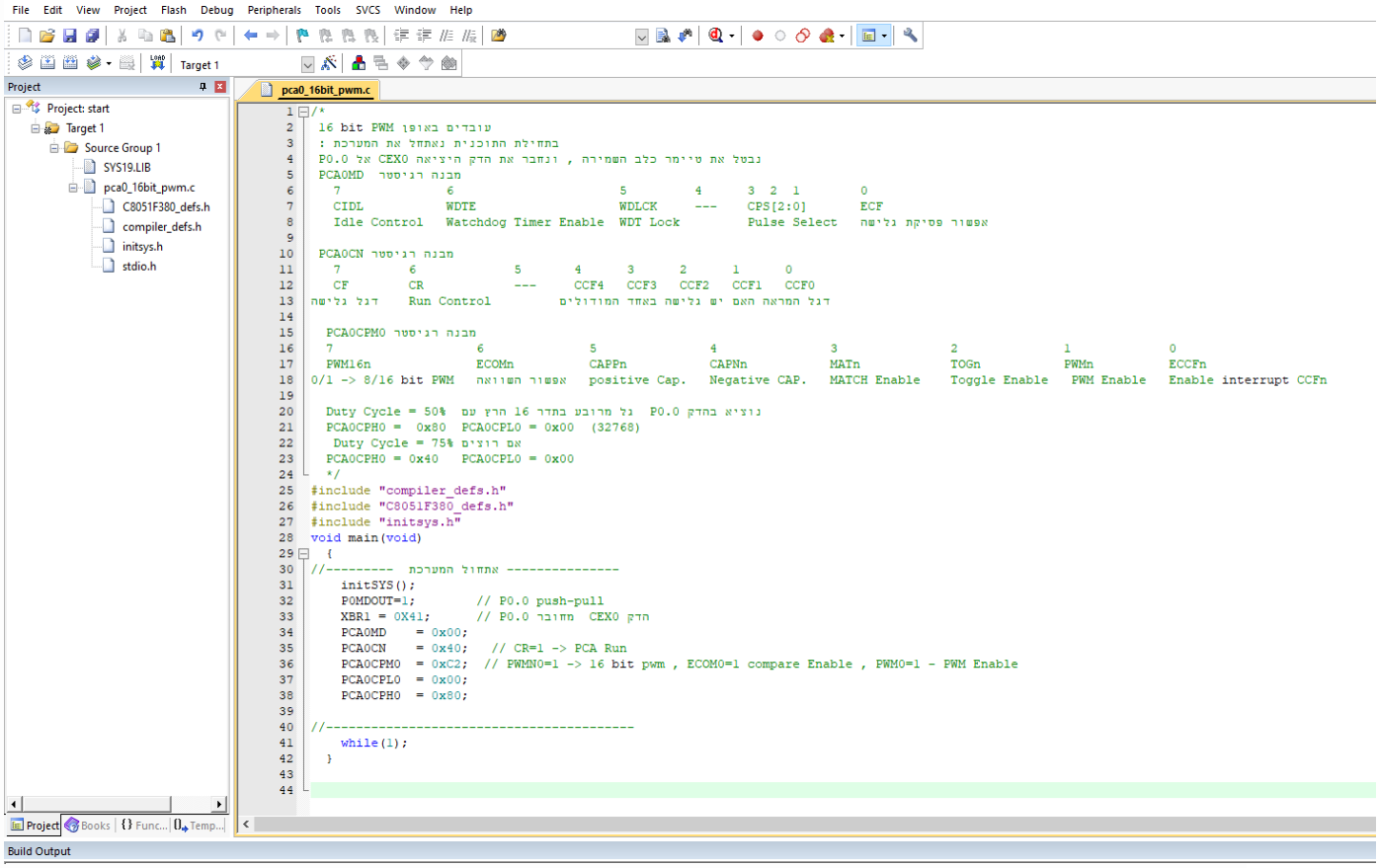
$$\text{Duty Cycle} = \frac{(65536 - PCA0CPn)}{65536}$$

התוכנית תיראה כך :

```
/*
 16 bit PWM עובדים באופן
 : בתחילת התוכנית נאתחל את המערכת
 P0.0 אל CEX0 נבטל את טיימר כלב השמירה , ונחבר את הדק היציאה
 PCA0MD מבנה רגיסטר
  7        6                          5          4     3  2  1      0
 CIDL      WDTE                       WDLCK      ---   CPS[2:0]     ECF
 Idle Control  Watchdog Timer Enable  WDT Lock        Pulse Select  אפשור פסיקת גלישה

 PCA0CN מבנה רגיסטר
  7      6          5      4    3    2    1    0
  CF     CR         ---    CCF4 CCF3 CCF2 CCF1 CCF0
 דגל גלישה  Run Control     דגל המראה האם יש גלישה באחד המודולים

 PCA0CPM0 מבנה רגיסטר
 7          6       5       4       3       2       1       0
 PWM16n     ECOMn   CAPPn   CAPNn   MATn    TOGn    PWMn    ECCFn
 0/1 -> 8/16 bit PWM  אפשור השוואה  positive Cap.  Negative CAP.  MATCH Enable  Toggle Enable  PWM Enable  Enable interrupt CCFn

  Duty Cycle = 50%  גל מרובע בתדר של כ 61 הרץ עם  P0.0 נוציא בהדק
  PCA0CPH0 =  0x80  PCA0CPL0 = 0x00     (32768)
        Duty Cycle = 75% אם רוצים
  PCA0CPH0 = 0x40   PCA0CPL0 = 0x00
 */
#include "compiler_defs.h"
#include "C8051F380_defs.h"
#include "initsys.h"
void main(void)
{
//---------  אתחול המערכת ----------------
initSYS();
P0MDOUT=1;       // P0.0 push-pull
```

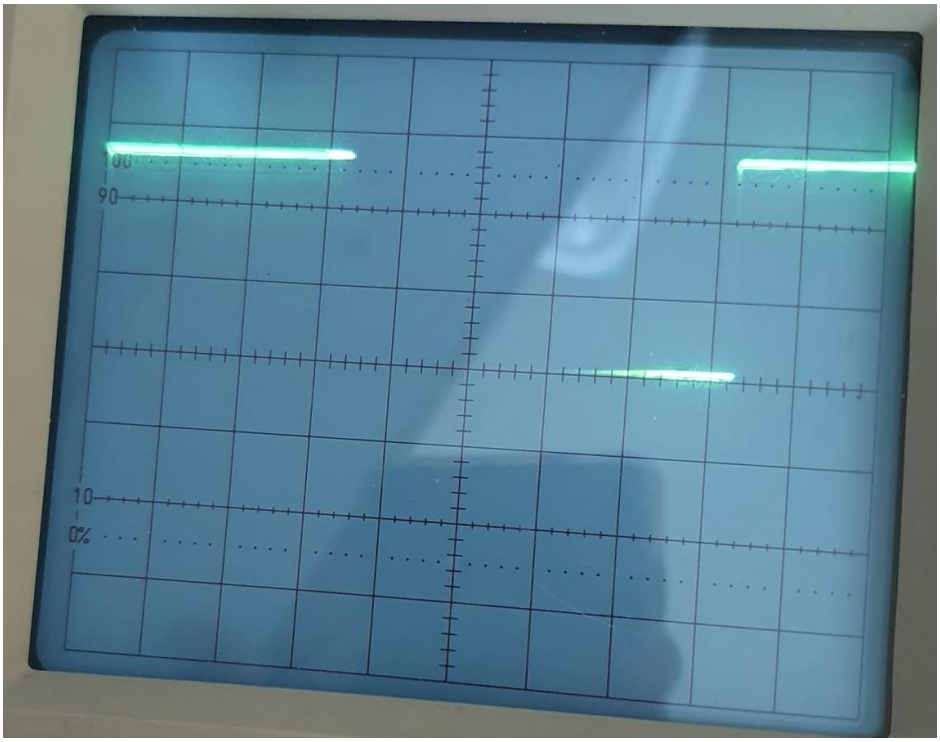
```
XBR1 = 0X41;      // P0.0 מחובר  CEX0 הדק
PCA0MD    = 0x00;
PCA0CN    = 0x40;   // CR=1 -> PCA Run
PCA0CPM0  = 0xC2;  // PWMN0=1 -> 16 bit pwm , ECOM0=1 compare Enable , PWM0=1 - PWM Enable
PCA0CPL0  = 0x00;
PCA0CPH0  = 0x80;
//----------------------------------------
while(1);
}
```

ובתוכנת ה UVISION היא נראית באיור הבא :



איור 20 : התוכנית 16 bit PWM בתוכנת µvision

צורת הגל המתקבלת במשקף התנודות (אוסצילוסקופ) נראית באיור הבא :



איור 21 : 16 bit PWM במסך משקף התנודות (אוסצילוסקופ).

המפסק VOLTS/DIV נמצא במצב 1 VOLTS/DIV . אמפליטודת המתח של ה PWM הוא ב 2.8 וולט.

מפסק TIME/DIV נמצא במצב 2 mSec/div . זמן המחזור הוא 8.2 משבצות כפול 2 מילי, כלומר 16.4 מילי שניות שהם תדירות של כ 61 הרץ לפי החישוב :

$1 / 16.4*10^{-3} = 60.975$ Hz

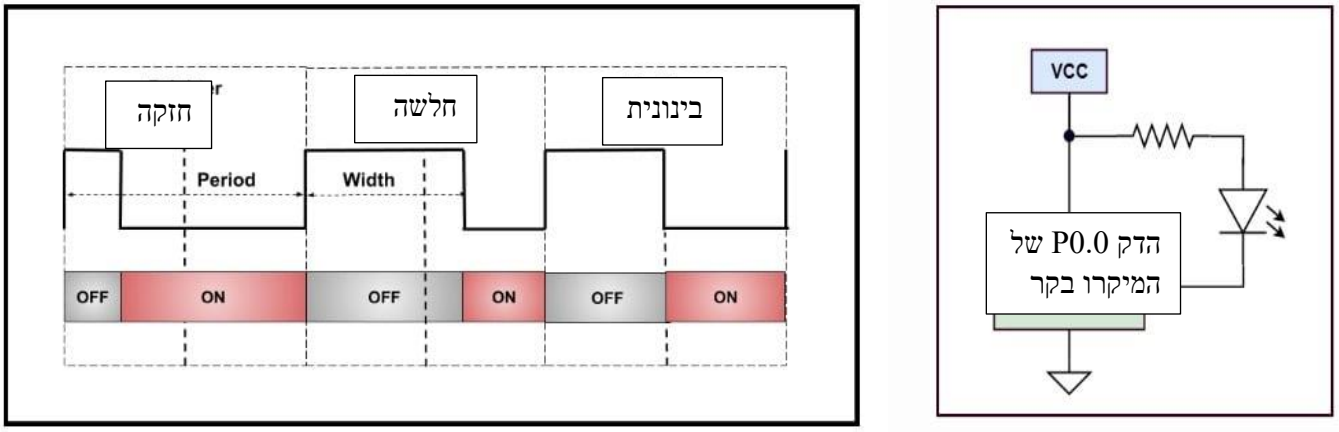
8. בקרה על עוצמת ההארה של לד או מהירות סיבוב של מנוע DC .

8.1 בקרה על עוצמת ההארה של לד

ל PWM יישומים רבים . אם נרצה לקבוע את עוצמת ההארה של לד או מהירות מנוע DC נוכל להשתמש ב PWM . ככל שה Duty Cycle של גל PWM גדול יותר אז הלד תאיר חזק יותר או המנוע יסתובב במהירות גבוהה יותר.

נניח שחיברנו לד ליציאת P0.0 של המיקרו בקר דרך נגד עבודה מתאים (כ 100 אוהם).

הערה: במקום לחבר לד ניתן לחבר את אות ה PWM למנוע DC קטן. אם המנוע דורש זרם גדול מ 20 מילי אמפר אז ניתן לחבר את המנוע דרך מגבר זרם.



איור 22 : חיבור לד ליציאת הדק העובד ב PWM . מצד ימין המעגל. מצד שמאל עוצמת הארה של הלד.

מהחלק הימני באיור רואים לד המתחברת להדק P0.0 של המיקרו בקר. (אפשר לחבר לכל הדק שנבחר מ P0.0 עד P0.3 של פורט 0 בעזרת תכנות של רגיסטר XBR1 עם הסיביות שלו הנקראות PCA0ME (**PCA0 M**odule **E**nable-).

כדי להדליק את הלד יש לתת ביציאת הדק P0.0 ערך של '0' (מצב SINK – הטבעת זרם) , כדי שתהיה סגירת זרם מה V_{cc} דרך הנגד והלד לאדמה .

בחלק השמאלי של האיור רואים את עוצמת ההארה של הלד עבור Duty Cycle מתאים. בחלק זה מתוארים 3 מצבים. כאשר זמן ה ON (צבע אדום) שונה האחד מהשני .

משמאל רואים זמן ON ארוך (Duty Cycle קטן) והלד דולקת בעוצמה חזקה.

במרכז זמן ה ON קצר והלד דולקת בעוצמה חלשה וה Duty Cycle גדול.

מצד שמאל זמן ה ON וזמן ה OFF שווים והלד דולקת ומאירה בצורה בינונית.

הדבר נראה קצת "הפוך" כי חשבנו שככל שה Duty Cycle יהיה גדול יותר הלד תאיר בעוצמה גדולה יותר אבל ההסבר הוא שכאן הלד נדלקת כאשר יש 0 ונכבית כאשר יש 1. אם היינו מדליקים את הלד על ידי '1' ביציאת ההדק אז היינו מקבלים שככל שה Duty Cycle גדול יותר הלד מאירה חזק יותר.

8.2 בקרה על מהירות מנוע DC

8.2.1 מנוע DC על קצה המזלג

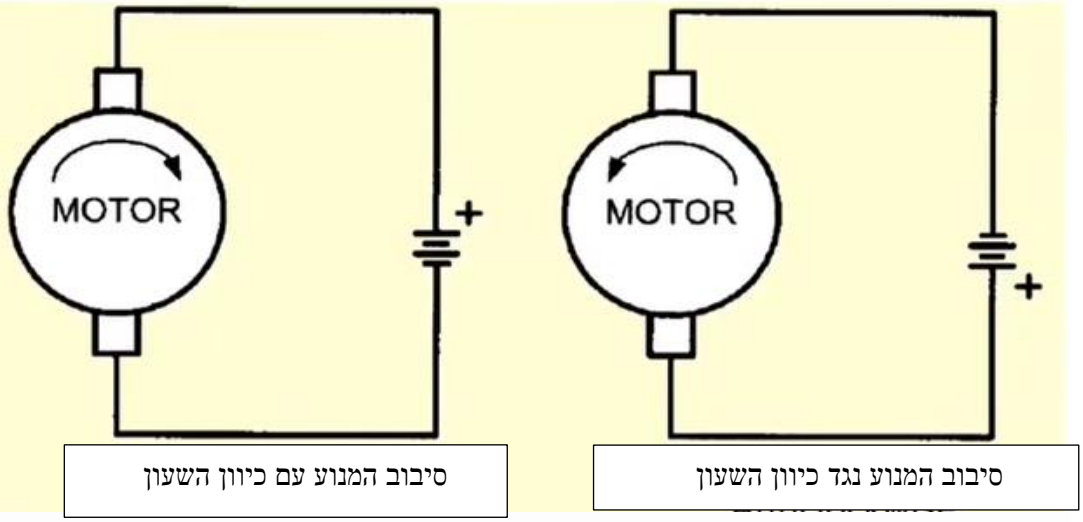
למנוע DC יש 2 הדקים הנקראים + (פלוס) ו – (מינוס). אם נותנים בין 2 ההדקים מתח המנוע מסתובב. כדי לשנות כיוון במנוע צריך להפוך את קוטביות המתח שנכניס ל 2 ההדקים ואז כיוון הזרם בהדקים שלו מתהפך והמנוע מסתובב לכיוון ההפוך ממקודם.

הזרם שמנוע DC צורך הוא בסביבות 100 מילי אמפר במנועים קטנים ועד אמפרים ועשרות אמפרים במנועים גדולים.

מהירות הסיבוב המקסימלי של המנוע נתונה על ידי היצרן ב RPM - Revolutions per minute - (סיבובים בדקה) .

המתח הנומינאלי של מנוע הוא המתח שבו המנוע מתוכנן לעבוד. מתח נמוך יותר גורם למהירות סיבוב נמוכה ומתח גבוה יותר יגביר את מהירות הסיבוב ויש להיזהר לא לעבור את המקסימום כדי שמנוע לא יישרף.

האיור הבא מראה את המתח שיש לספק למנוע כדי שיסתובב עם כיוון השעון או נגד כיוון השעון:



איור 23 : סיבוב המנוע עם כיוון השעון (בצד ימין באיור) או נגד כיוון השעון (בצד שמאל)

8.2.2 דוחפי זרם למנוע DC

מיקרו בקרים אינם יכולים לספק זרם בגודל של מאות מילי אמפרים ובוודאי שלא אמפרים ולכן נהוג להשתמש בדוחפי זרם שיודעים להוציא זרמים מתאימים להפעלת המנועים. סיבה נוספת היא שהפעלת המנוע הבנוי מסלילים והזרמת הזרם או הפסקת הזרם דרכו גורמים לכוח אלקטרו מגנטי היוצר מתחים הפוכים שיכולים לגרום נזק למיקרו בקר.

דוחפי זרם נפוצים הם רכיבים הנקראים L293D או L298N . הסבר מפורט על הרכיב L298N ניתן למצוא בקישור :

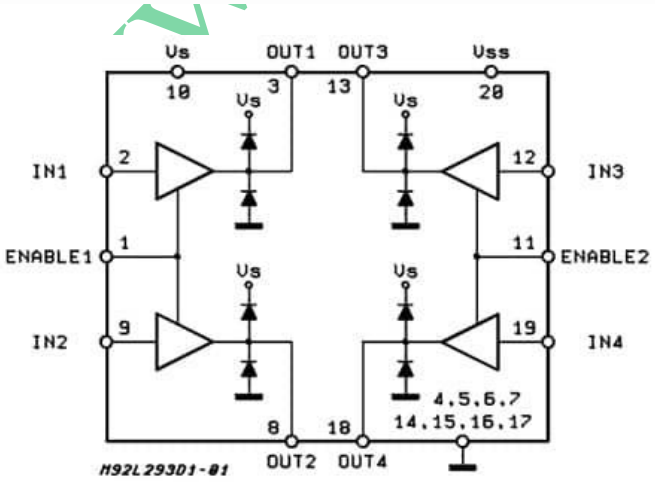
https://www.arikporat.com/wp-content/uploads/2024/02/298Module-1.pdf

בפרק זה נעבוד עם הרכיב/ג'וק L293D .

לג'וק L293D ניתן לתת מתח ספק מ 4.5 וולט ועד 36 וולט והזרם המקסימלי שהוא מסוגל לספק הוא עד 600 מילי אמפר.

לג'וק L293D יש דיודות הגנה ביציאות הרכיב ואילו לג'וק L293 אין דיודות הגנה בתוך הג'וק ויש להוסיף דיודות הגנה חיצונית).

הג'וק נקרא PUSH-PULL FOUR CHANNEL DRIVER WITH DIODES – דוחף PUSH-PULL ל 4 ערוצים עם דיודות . האיור הבא מתאר את סכמת המלבנים של הרכיב L293D וטבלת האמת עבור אחד הערוצים



INPUT	ENABLE (*)	OUTPUT
H	H	H
L	H	L
H	L	Z
L	L	Z

Z = High output impedance

איור 24 : סכמת המלבנים של הרכיב וטבלת האמת שלו :

מהאיור רואים שיש 4 מגברי זרם . אין כאן חיבור של גשר H "רגיל" כמו במגברי זרם למנועים אחרים. אפשר להגיד שהמבנה הוא חצי גשר H .

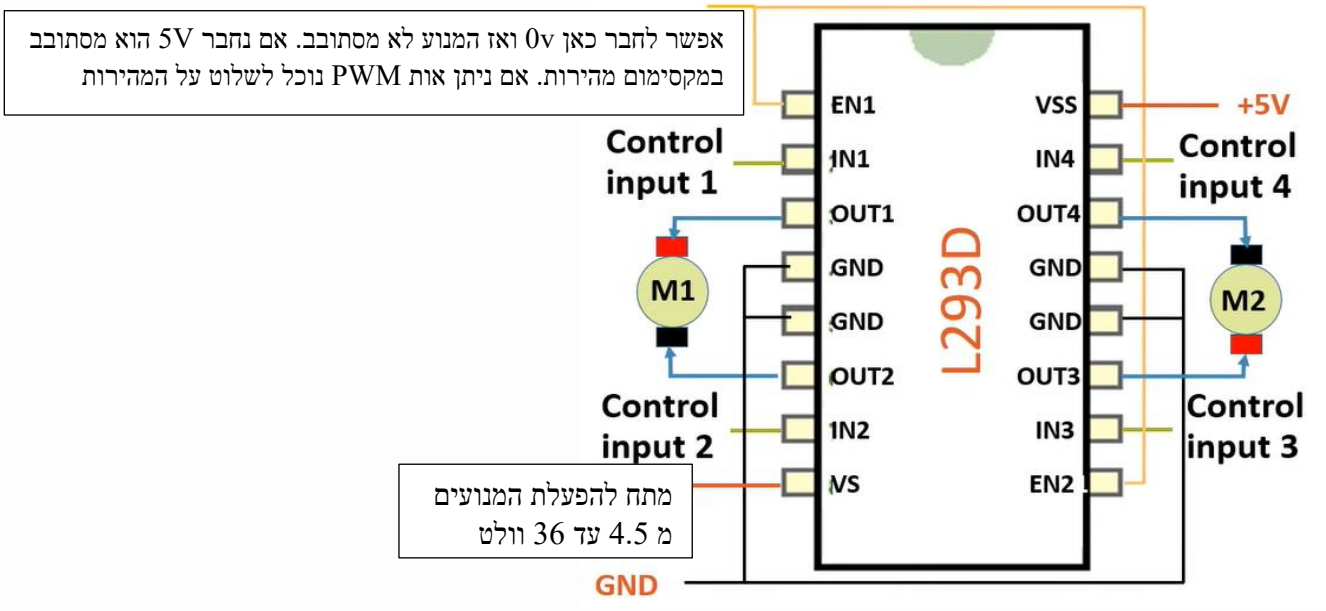
ניתן לחבר מנוע לאחד המגברים ואז הוא יסתובב בכיוון אחד ונוכל לשלוט על המהירות שלו בעזרת PWM .

ניתן לחבר את המנוע בין 2 יציאות של מגבר ואז ניתן לשלוט גם על כיוון הסיבוב וגם המהירות.

ביציאת כל מגבר יש דיודות הגנה כדי להגן על המגבר ממתחים הפוכים שמתפתחים בסלילי המנוע במיוחד במעברים מסיבוב המנוע לעצירה, כלומר במעבר במגבר מרוויה לקטעון .

לכל 2 מגברים יש הדק אפשור ENABLE . לפי טבלת האמת רואים שיש לשים '1' (HIGH) כדי שהמגבר יעבוד.

האיור הבא מתאר את ההדקים של הרכיב :



איור 25 : הדקי הרכיב L293D .

הג'וק בתצורת DIP הוא של 16 הדקים והוא יכול לדחוף זרם ל 2 מנועי DC . בצד שמאל של הג'וק רואים את הדקי הבקרה על מנוע אחד ומצידו השני את הדקי הבקרה על המנוע השני .

EN1, EN2 - הדק מספר 1 והדק מספר 9 הם הדקי האפשור (ENable) של הג'וק . 0 באחד מהדקים אלו או בשניהם לא מאפשר את הפעולה של האחד המנועים או שניהם. 1 בהדקים אלו מאפשרים את פעולת דוחף המנוע המתאים. לכן חיברנו את ההדקים האלו ל 5 וולט ובכך נאפשר את פעולת 2 המנועים. אם נחבר אותם ל-GND המנוע לא מסתובב. אם נחבר אותם ל-5 וולט המנוע יסתובב במהירות מקסימלית. אם נחבר אותם לפין דיגיטלי PWM וניתן ערך ביניים המנוע יסתובב במהירות מתאימה באופן יחסי למתח אותו נתנו.

IN1,IN2 הן אותות הבקרה CONTROL INPUT למנוע 1 , ו **IN3,IN4** הם אותות הבקרה למנוע 2 . בהדקים אלו מתח הנחשב כ '0' הוא עד 1.5 וולט (VILmax) ומתח הנחשב כ '1' הוא ממתח מ 2.3 וולט (VIHmin) , כך שהוא מתאים לעבודה עם המיקרו C8051F380.

OUT1, OUT2 הם ההדקים אליהם נחבר את מנוע 1 ואילו **OUT3, OUT4** הם הדקי החיבור של מנוע 2 .

VSS הוא מתח הספק של המעגל הלוגי שברכיב. נחבר אותו ל 5 וולט . ניתן לחבר עד 36 וולט.

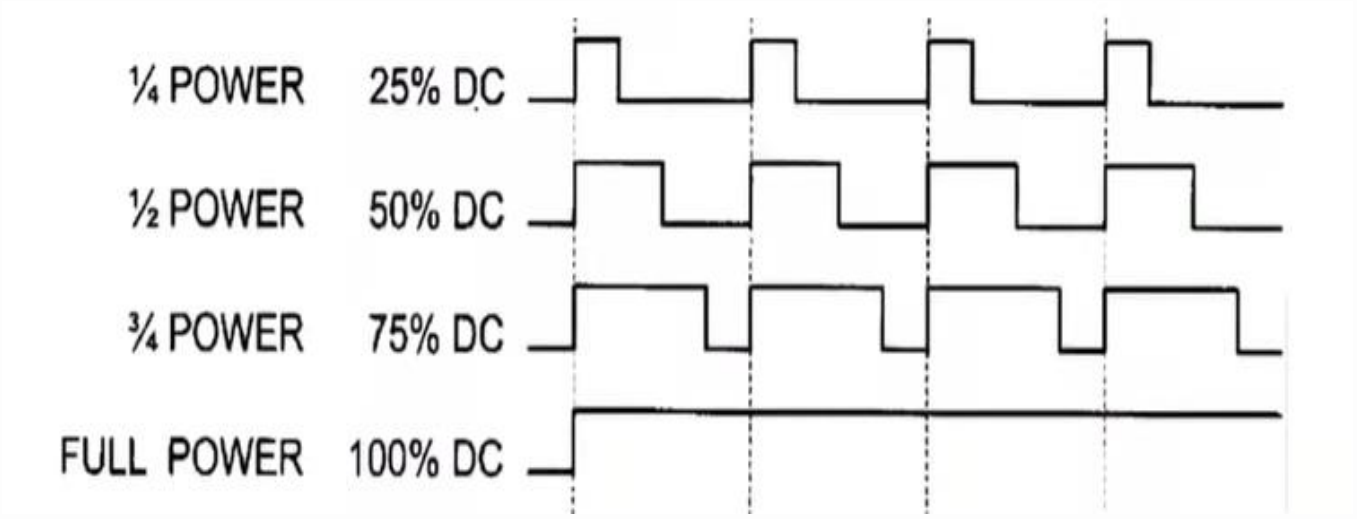
VS – הוא המתח של הספק שבעזרתו נסובב את המנועים (ניתן להכניס ממתח VSS ועד 36 וולט).

8.2.3 עבודה עם PWM ומנוע DC .

ישנן מספר דרכים לשלוט על מהירות של מנוע DC . דרך פשוטה ונפוצה היא בעזרת PWM .

אם היינו רוצים לסובב מנוע במהירות משתנה היינו צריכים לספק לו מתח ישר משתנה. מערכת אנאלוגית שמספקת מתח משתנה היא מורכבת ויש לה בזבוז הספק ונצילות נמוכה. הפתרון של PWM נותן אפשרות טובה . בשיטה זו מעבירים למנוע מתח ב Duty Cycle משתנה. הדבר גורם לממוצע DC שונה ומהירות המנוע משתנה. הנצילות במקרה כזה גבוהה בהרבה.

האיור הבא מתאר צורות גל של PWM עם Duty Cycle שונים :



איור 26 : צורות גל של PWM עם Duty Cycle שונים .

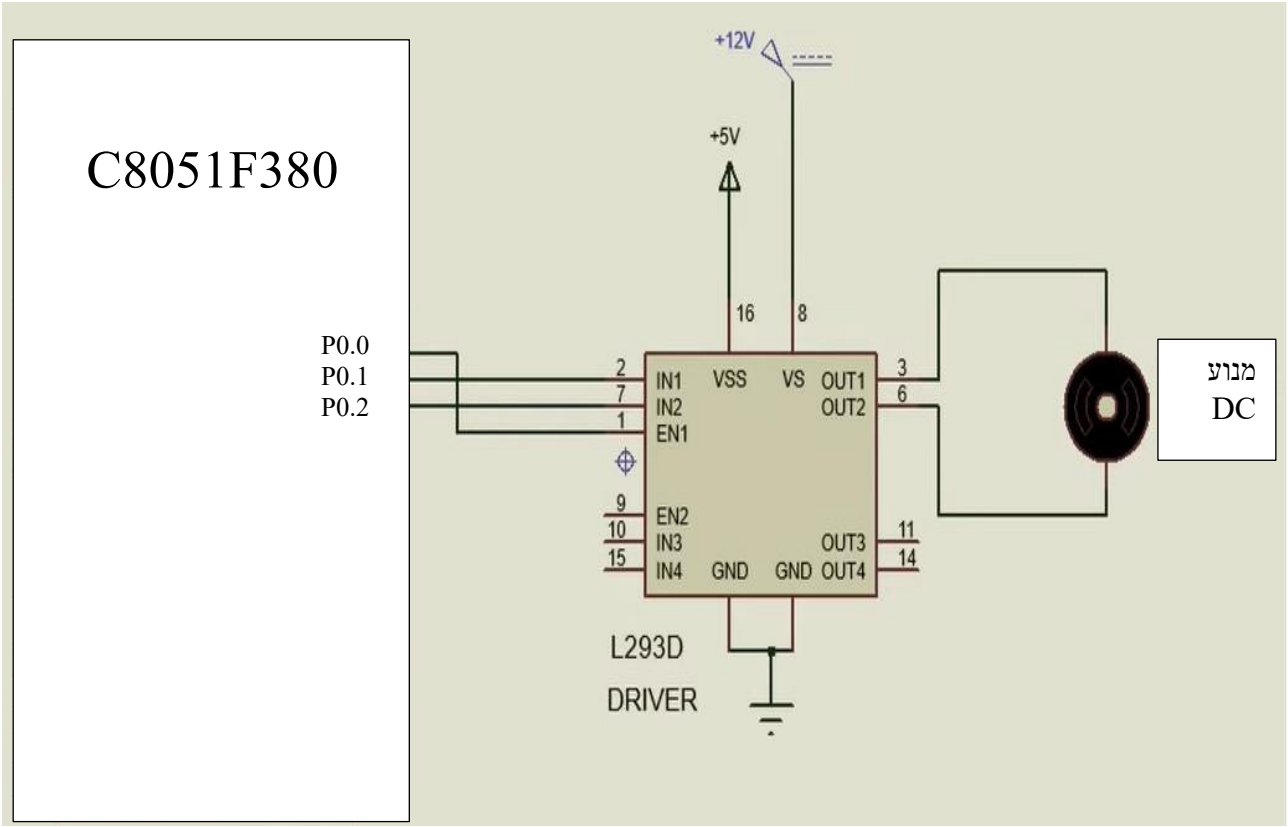
רואים שעבור Duty Cycle =100% (החלק התחתון באיור) ההספק המתקבל הוא מקסימלי/מלא ונקרא באיור שלמעלה FULL POWER .

נניח שהמנוע מסתובב במהירות של 100 RPM (סיבובים בדקה) עבור FULL POWER.

בהנחה שניתן לומר באופן כללי שמהירות המנוע נמצאת ביחס ישיר למתח שהוא מקבל אז עבור Duty Cycle = 25% הוא יקבל רבע מההספק ולכן יסתובב ברבע מהירות כלומר 25 סיבובים בדקה. עבור Duty Cycle של 50% הוא יסתובב ב 50 סיבובים לדקה וכך הלאה.

מומלץ להשתמש בתדירות PWM של לפחות 50 קילו הרץ עבור מנועי DC ללא מברשות Brushless DC Motors. תדירות של 80 קילו הרץ ויותר תהיה אפילו יותר מתאימה.

באיור הבא מתואר המעגל החשמלי שבו נשתמש :



איור 27 : חיבור מנוע DC אל דוחף זרם L293D .

ב P0.0 נכניס את אות ה PWM וה Duty cycle שלו יקבע את מהירות המנוע.

בעזרת P0.1 ו P0.2 נקבע את כיוון הסיבוב.

8.2.4 התוכנית

בדומה לתוכנית של הפעלת PWM ב 8 ביט שנמצאת בסעיפים קודמים גם כאן אתחלנו את הרגיסטרים הקשורים ל PCA בצורה דומה. ההבדל היחיד הוא שאת רגיסטר PCA0MD אתחלנו בערך 0x02 כדי לקבל תדר PWM של 46,875 Hz . הפקודה היא:

```
PCA0MD    = 0x02;    // חלוקת תדר השעון ב 4
```

גם כאן אתחלנו את PCA0CPH0 בערך 128 כדי לקבל Duty Cycle של 50% . הפקודה היא :

```
PCA0CPH0 = 128;      //  Duty Cycle = 50%  כדי לקבל
```

בהמשך התוכנית יש לולאה אין סופית שבה מריצים את המנוע בכיוון אחד, למשך 2 שניות , עוצרים אותו לשנייה אחת, מריצים את המנוע בכיוון ההפוך ושוב עוצרים לשנייה וחוזר חלילה.

לתוכנית ניתן להוסיף בקלות 5 מפסקים :

מפסק אחד של עצירת המנוע, מפסק שני של הרצת המנוע בכיוון אחד , מפסק שלישי להרצת המנוע בכיוון ההפוך. מפסק נוסף להגדלת מהירות המנוע על ידי הגדלת ה Duty Cycle של ה PWM – (הקטנת הערך שנעביר ל PCA0CPH0) ומפסק חמישי להקטנת המהירות על ידי הקטנת המהירות של המנוע על ידי הגדלת הערך שנעביר ל PCA0CPH0 .

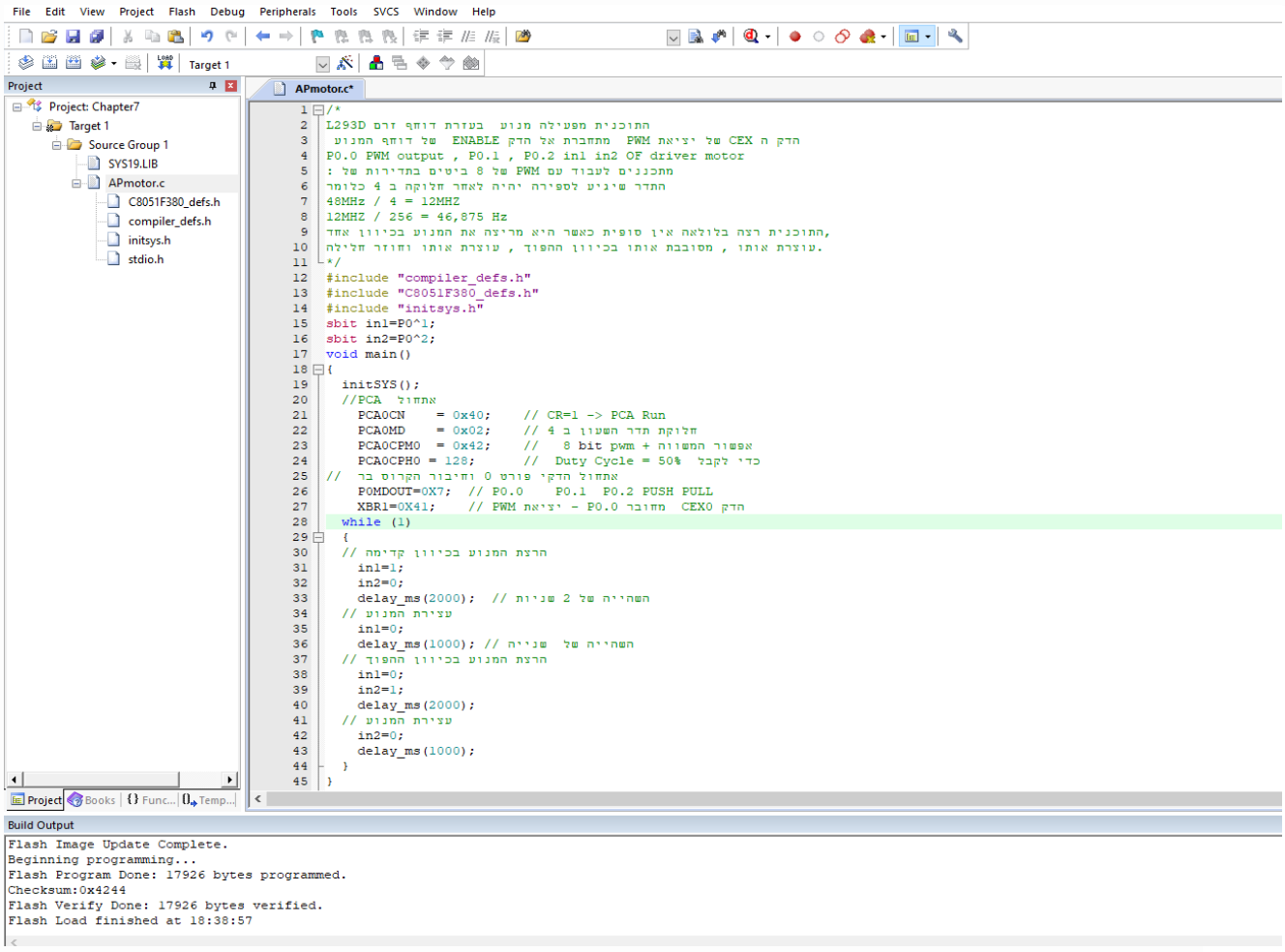
הגדלת המהירות תהיה עד Duty Cycle של 100% (PCA0CPH0 = 0) והקטנת המהירות תהיה עד ל Duty Cycle של 0% (PCA0CPH0 = 255) .

התוכנית נראית כך :

```
/*
התוכנית מפעילה מנוע  בעזרת דוחף זרם L293D
הדק ה CEX של יציאת PWM  מתחברת אל הדק ENABLE  של דוחף המנוע
P0.0 PWM output , P0.1 , P0.2 in1 in2 OF driver motor
מתכננים לעבוד עם PWM של 8 ביטים בתדירות של :
התדר שיגיע לספירה יהיה לאחר חלוקה ב 4 כלומר
48MHz / 4 = 12MHZ
12MHZ / 256 = 46,875 Hz
התוכנית רצה בלולאה אין סופית כאשר היא מריצה את המנוע בכיוון אחד,
עוצרת אותו , מסובבת אותו בכיוון ההפוך , עוצרת אותו וחוזר חלילה.
*/
#include "compiler_defs.h"
#include "C8051F380_defs.h"
#include "initsys.h"
sbit in1=P0^1;
sbit in2=P0^2;
void main()
{
 initSYS();
        //PCA  אתחול
   PCA0CN    = 0x40;   // CR=1 -> PCA Run
   PCA0MD    = 0x02;    // 4 חלוקת תדר השעון ב
   PCA0CPM0  = 0x42;            //   8 bit pwm + אפשור המשווה
   PCA0CPH0 = 128;      //  Duty Cycle = 50%  כדי לקבל
//  אתחול הדקי פורט 0 וחיבור הקרוס בר
                P0MDOUT=0X7;  // P0.0    P0.1  P0.2 PUSH PULL
          XBR1=0X41;                    // PWM יציאת - P0.0 מחובר  CEX0 הדק
        while (1)
        {
        // הרצת המנוע בכיוון קדימה
                in1=1;
```

```
            in2=0;
            delay_ms(2000);  // השהייה של 2 שניות
        // עצירת המנוע
            in1=0;
          delay_ms(1000);      // השהייה של  שנייה
        // הרצת המנוע בכיוון ההפוך
            in1=0;
            in2=1;
            delay_ms(2000);
        // עצירת המנוע
            in2=0;
            delay_ms(1000);
        }
}
```

התוכנית ב µVISION נראית כך :



איור 28 : התוכנית ב µVISION

9. ביבליוגרפיה :

1. https://www.silabs.com/documents/public/data-sheets/C8051F38x.pdf
2. https://www.allaboutcircuits.com/technical-articles/introduction-to-microcontroller-timers-pwm-timers/
3. https://www.electronics-tutorials.ws/blog/pulse-width-modulation.html

10. נספח : פרק 8 מסילבוס תוכנית הלימודים לכתה יג במגמת אלקטרוניקה מחשבים משרד החינוך

פרק 8 - PWM באמצעות מערכת PCA (Programmable Counter Array) 10) שעות)

8.1 מבנה הפנימי של מערכת ה-PCA ויישומו ליצירת PWM ל-8 סיביות.

8.2 קביעת הדקי המוצא של ה- PWM , אפשור, מספר הסיביות ותדר העבודה באמצעות סימון המצבים בחלון PCA ו-PORT I/O של תוכנת Configuration Wizard .

8.3 הכרת הרגיסטר PCA0CPHn לקביעת גורם המחזור לפי 8 סיביות.

8.4 כתיבת תכנית להפעלת PWM באחד המוצאים.

8.5 שימוש במוצא PWM 8 סיביות לבקרה על מהירות מנוע DC ועוצמת הארה של לד.

WW.