

חלק תיאורטי:

1. PxSEL: ברירה בין מודולי החומרה המשמשים באותה רגל בקר.
PxDIR: קביעת כיוונית רגל הבקר ואת שימוש
PxOUT: קביעת ערך לוגי במוצא ברגל הבקר (v0-0 לוגי, v3.3-1 לוגי)
PxIN: קריאת ערך מתח לוגי ברגל הבקר
2. לאחר ביצוע PxSEL - RESET נקבע על מצב ברירת המחדל שלו והוא '0' (ברוב המקרים- תלוי במעבד) וזהו המצב המתאים ל- I/O, כלומר input\output.
3. אפסנו את כל הביטים בPxDIR ולאחר מכן עשינו OR עם 0x55 שזה בבינארי '1' בכל המקומות הזוגיים. כך קיבלנו שכל הזוגיים הם במצב output ובמקומות האי זוגיים במצב input.

```
P9SEL &= ~0xFF
P9DIR &= 0x55
P9DIR |= 0xAA
```

4. לפי הנתון זמן מחזור בגל ריבועי, החלק של '1' הוא זמן של 0.5 ms. (אחד חלקי 2)
זמן מחזור של MCLK הוא 2^{-20} sec ולכן כדי לקבל 0.5 ms שהייה על '1' נצטרך:
 $0.5 \cdot 2^{-20} = 524.288 \approx 524$ מספרי מחזורים של MCLK על '1', וזה אותו מספר מחזורים עבור '0'.
5. פסיקה הינה אות חשמלי המתקבל בCPU מרכיב חומרה או משורת קוד, המאפשר לשנות את סדר ביצוע הפקודות ללא בקרה מותנת. הצורך בפסיקה עולה כאשר אנו רוצים להריץ קוד ללא תשואל קבוע של המעבד – כך המעבד פנוי לעסוק בדברים אחרים.
6. כאשר הקוד שלנו עובד בצורת תשואל (polling) המעבד מבדק הרבה אנרגיה כדי לבדוק האם הגיעה קריאה לביצוע מותנה (מתג, חיישן ועוד...). הדבר מהווה שימוש לא נכון של CPU ומונע ממנו להיות מסוגל לעסוק בדברים אחרים. אך במקרה של פסיקה נוכל להכניס את המעבד למצב שינה כך שיתעורר לביצוע המותנה רק במקרה של פסיקה, או לאפשר לו לעסוק בנושאים פחות דחופים עד לקבלת פסיקה.
7. קיימים 3 סוגים של פסיקות:
א. פסיקה חיצונית- פסיקה הנגרמת מרכיב חומרה ואינו תלוי במימוש התוכנה.
ב. פסיקה פנימית- פסיקה הנגרמת משורת קוד או פעולה צפויה במעבד אותה אנו יודעים לצפות את התממשותה.
ג. פסיקת תוכנה- פסיקה שעולה בעקבות דגל בקוד.
8. CPU אופני עבודה שונים המשמשים אותנו ליעול צריכת האנרגיה של המעבד. כך מעבד המחובר למקור אנרגיה מוגבל יוכל להחזיק זמן רב יותר עם אותה כמות אנרגיה.
בCPU שלנו קיימים 6 אופני עבודה:
א. AM (Active Mode) - המצב הסטנדרטי בו כל רכיבי המעבד עובדים בצורה מלאה.
ב. LPM0 (Low Power Mode 0) - CPU מושבתים. צריכת הזרם יורדת משמעותית יורדת ב-500% לכן ברוב המקרים יהיה נוח להשתמש במצב זה. הבקר עדיין יזהה פסיקות
ג. LPM1 (Low Power Mode 1) - מתנד CPU MCLK DCO מושבתים. נשתמש מתב זה כאשר נצטרך צריכת זרם של עד 55 μA .
ד. LPM2 (Low Power Mode 2) - מתנד CPU MCLK DCO מושבתים. נשתמש מתב זה כאשר נצטרך צריכת זרם של עד 17 μA .
ה. LPM3 (Low Power Mode 3) - מתנד CPU MCLK DCO מושבתים. נשתמש מתב זה כאשר נצטרך צריכת זרם של עד 0.9 μA .
ו. LPM4 (Low Power Mode 4) - מתנד CPU MCLK DCO מושבתים. נשתמש מתב זה כאשר נצטרך צריכת זרם של עד 0.1 μA .
9. כדי לקנפג את רגל הבקר P2.0 כך שב falling edge תתבצע בקשת פסיקה נבצע את השלבים הבאים:

```
P2SEL &= ~0x01
P2DIR &= ~0x01
P2IFG &= ~0x01
P2IES |= 0x01
```

P2IE |= 0x01

10.

Blocking.1

בתכנות זו קריאה לפונקציה או פעולה מסוימת חוסמת את המשך הביצוע של התוכנית עד שהפונקציה או הפעולה תושלם. במהלך הזמן הזה התוכנית לא תוכל להמשיך לבצע פעולות אחרות.

Non-blocking. 2

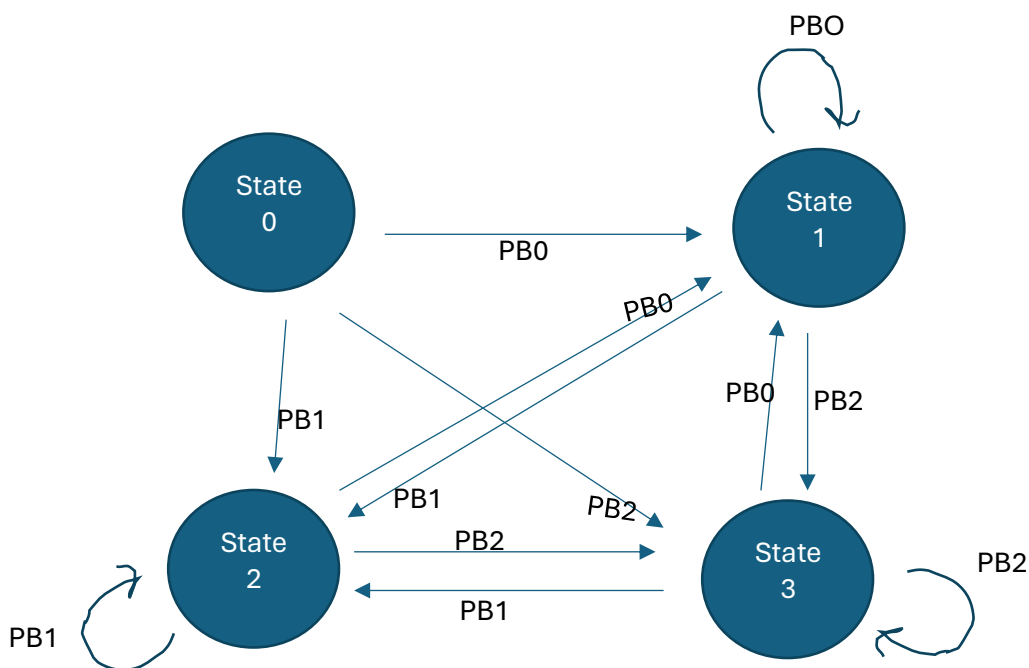
בתכנות זו קריאה לפונקציה או פעולה מסוימת אינה חוסמת את המשך הביצוע של התוכנית. התוכנית יכולה להמשיך לבצע פעולות אחרות בזמן שהפונקציה מבוצעת ברקע או מחכה לסיים.

Event-driven. 3

בתכנות זו הפעולות מבוצעות בתגובה לאירועים מסוימים שמתרחשים בתוכנה או במערכת. התוכנית ממתינה לאירועים כמו לחיצת כפתור או סיום פעולה כלשהי ומבצעת קוד מתאים בתגובה לאירועים אלו.

Interrupt-driven. 4

בתכנות זו פעולות מסוימות מתבצעות בתגובה לinterrupts שמתרחשות במערכת. כאשר פסיקה מתרחשת התוכנית מופסקת ועוברת לקוד שמטפל בפסיקה.



כאשר המעבר ממצבים 1 ו2 קורה רק אחרי סיום המצב.