מבנה מחשבים ספרתיים

361.1.4191

Final report LAB1

בר קופפרשמיד 318912193

רותם ארביב 208065052

המשימה

הוספת מצב חדש לקוד המערכת המבוסס מודל שכבות אבסטרקציה וגרעין הפעלה FSM לתמיכה ביכולת הרחבת המערכת, תחזוק המערכת ומימוש מצב חדש תוך שמירה על עמידות המערכת.

המצב החדש יהווה את state4. המצב מוגדר כך שבלחיצה על לחצן PB3 המערכת נדרשת להדפיס על גבי הלדים את הערך ה ASCII של איברי המחרוזת name איבר אחר איבר עם השהייה של 250ms.

name היא מחרוזת המכילה איברים מטיפוס BYTE של שם מלא של אחת הסטודנטיות ובמקרה שלנו Rotem Arbiv.

דרך הפתרון

הוספנו למערכת המצבים מצב חדש state4 למימוש המשימה. כניסה למצב זה מתרחשת בעת לחיצה על לחצן PB3 אותו חיברנו לרגל P2.3 וקינפגנו בהתאם בשכבת הbsp.

הלדים כבר מחוברים ל- PORT1 בערכה האישית ול- PORT9 בערכת המעבדה בהתאם לדרישות הדוח המכין, לכן יש צורך לקנפג רק את רגל P2.3.

נדרש לקנפג את הרגל המתאימה (הרביעית) ב P2SEL (P2SEL.3) להיות אפס לפעולת הרגל במצב I/O, לקנפג את P2DIR.3 להיות אפס כלומר במצב INPUT, לקנפג את P2IES.3 לערך 0 (Pull Down), לקנפג את P2IE.3 לערך 1 כדי לאפשר פסיקה ואיפוס P2IFG.3 כדי לנקות את הדגל אם מורם (יורם בלחיצה על הכפתור).

קינפוג בשכבת bsp:

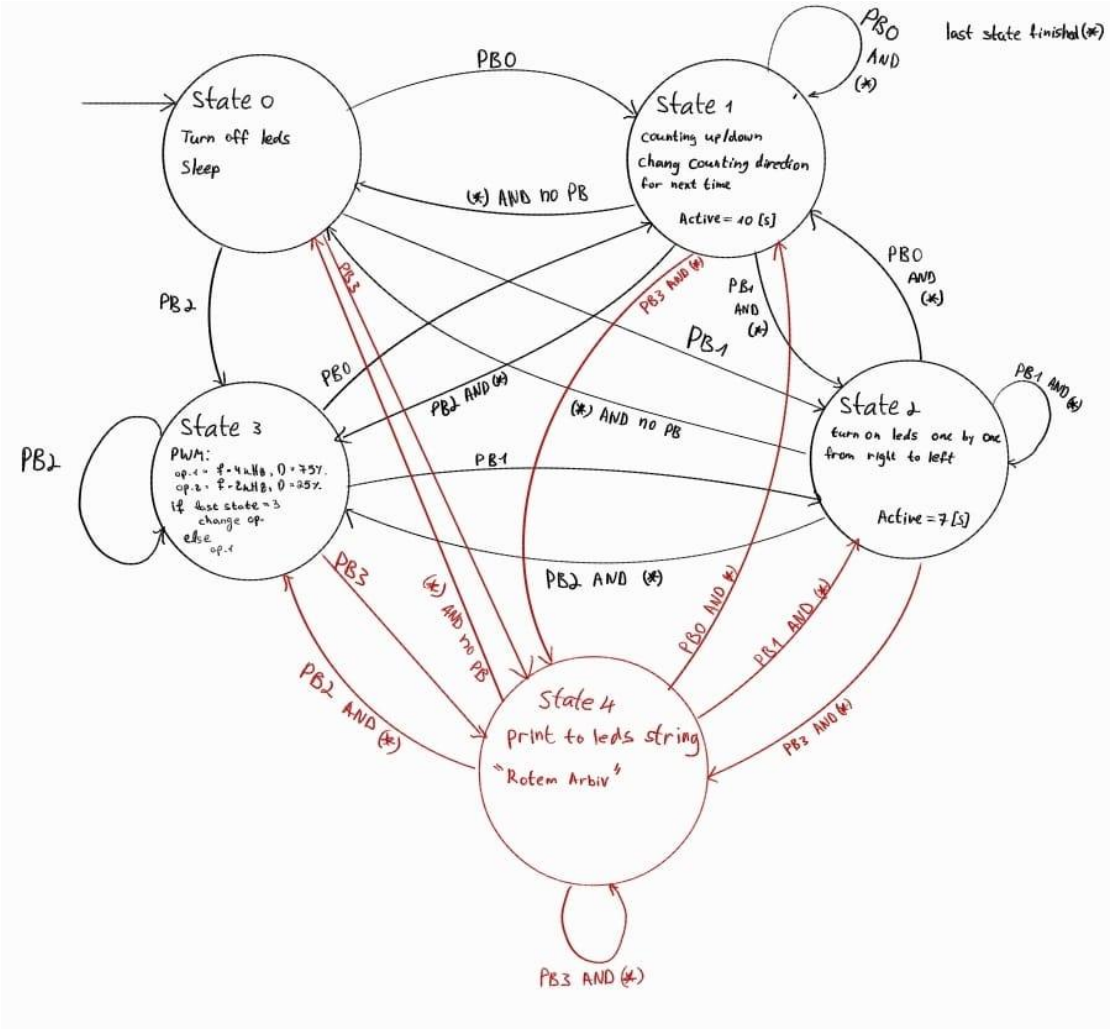
bsp.c:

```
// PushButtons Setup
PBsArrPortSel &= ~0x0F;            // I/O
PBsArrPortDir &= ~0x0F;            // input
PBsArrIntEdgeSel |= 0x03;          // pull-up mode
PBsArrIntEdgeSel &= ~0x0C;         // pull-down mode
PBsArrIntEn |= 0x0F;
PBsArrIntPend &= ~0x0F;            // clear pending interrupts
```

bsp.h:

```
// PushButtons abstraction
#define PBsArrPort          P2IN
#define PBsArrIntPend       P2IFG
#define PBsArrIntEn         P2IE
#define PBsArrIntEdgeSel    P2IES
#define PBsArrPortSel       P2SEL
#define PBsArrPortDir       P2DIR
#define PB0                 0x01
#define PB1                 0x02
#define PB2                 0x04
#define PB3                 0x08
```

לחיצה על כפתור PB3 משנה את המצב לstate4 ורוטינת הטיפול בפסיקה מתחילה בהתאם לגרף המצבים החדש :



<u>הגדרת המצב מהmain :</u>

```
52        case state4: // Print name
53          disable_interrupts();
54          PrintStr(name , 26175); // 250 ms delay
55          state = state0;         // When finish go back to sleep
56          enable_interrupts();
57          break;
```

שימוש בפונקציות disable_interupts(), enable_interrupt() מאפשרות כיבוי של אפשור הפסיקות הגלוגבלי (GIE) לפני הכניסה לטיפול בפסיכה והדלקה לאחר מכן כדי לא לאפשר למצב אחר "לחתוך" מצב זה עד להשלמת ההדפסה על גבי הלדים.

בשורה 54 מתבצעת קריאה לפונקציה השולחת להדפסה את המחרוזת שברצוננו להדפיס (מוגדרת בתחילת main) יחד עם מספר שיגרום להשהיה של 250ms בין הדפסת תו לתו הבא אחריו במחרוזת. הפונקציה מוגדרת ופועלת משכבת api.

לאחר מכן בשורה 55 חוזרים למצב 0 שמכבה את הלדים ומכניס את המערכת למצב שינה.

הגדרת הפונקציה בשכבת api:

```
//-----------------------------------------------------------------------
//            Real Time Task - Print String to LED
//-----------------------------------------------------------------------
void PrintStr(char str[] , unsigned int delay_ms){
    unsigned int i = 0;

    while (str[i] != '\0'){
        print2LEDs(str[i]); // Print char to LED
        delay(delay_ms); // delay
        i++;
    }
}
```

פונקציה זה עוברת בלולאה על כל איברי המחרוזת עד לסופה ומבצעת את ההדפסה של כל אות עם השהייה בין כל הדפסה להדפסה. הדפסת כל אות (או תו כמו רווח) וההשהיה בין כל הדפסה מתבצעים על ידי שתי פונקציות ייעודיות המוגדרות בשכבת HAL.

הגדרת הפונקציות בשכבת HAL:

```
void print2LEDs(unsigned char ch){
    LEDsArrPort = ch;
}
```

הדפסה של כל תו ללדים.

```
void delay(unsigned int t){   // t[msec]
    volatile unsigned int i;

    for(i=t; i>0; i--);
}
```

השהייה בין כל הדפסה.