מעבדת זמן אמת מספר 3 בקורס מבנה מחשבים ספרתיים

במטלת זמן אמת של מעבדה זה התבקשנו להוסיף מצב עבודה חדש בנוסף למצבים אשר מימשנו בעבודת הבית
המצב מצריך להשתמש בDMA על מנת לבצע היפוך של מחרוזת אשר נתונה בעבודה, לאחר מכן על מנת לראות שהפעולה בוצעה כראוי נדרש להציג אותה על מסך הLCD (32 תווים ראשונים של המחרוזת)

ראשית כל, על מנת להוסיף מצב חדש לעבודה ולוודא שהוא עובד היטב עם המצבים השונים, הרחבנו את רוטינת הפסיקה לכלול גם את Push button 3, עבורו אנו שומרים את המצב שלנו כמשתנה גלובלי וקובעים כי state = state3, כאשר בקובץ התוכנית הראשית main.c אנו מבטלים פסיקות גלובליות עד שמסיימים את הפעולה של המצב ולאחר מכן עוברים למצב שינה. הסיבה לכך היא כי בעבודה מוגדר שלא ניתן לעבור למצבים אחרים תוך כדי ביצוע המצב הנוכחי.

כעת, על מנת להשתמש בDMA לביצוע המשימה, הגדרנו ערוץ אחד בלבד שלו בתצורה הבאה:

- מעבירים מידע בצורה של memory to memory, המקור הוא המערך המקורי (הכתובת של האיבר האחרון) והיעד הוא המערך החדש (כתובת התחלתית, על מנת לבצע היפוך)
- כל יחידה שמעבירים הינה בגודל BYTE (גודל של משתנה מסוג char)
- ההעברה היא מסוג block transfer, כלומר העברה יחידה שמורכבת ממספר העברות קטנות, במקרה הזה של משתנים מסוג char.
- הדרך בה מעבירים את הבלוק היא שאת כתובת המקור מורידים (decrement source) ואת כתובת היעד מעלים (Increment source)
- אפשור של הDMA
- קביעת הטריגר של הDMA להיות לפי רגיסטר CCR2 של Timer A, זה מצריך קנפוג נוסף של הטיימר (הפעלה שלו, מצב מנייה, קביעת ערך CCR0 על מנת לקבוע את קצב קבלת הטריגרים לDMA
  - הערה: במעבדה לאחר הסתכלות על הקוד והערה של המדריך, הבנו שעדיף לקבוע את הטריגר להיות לפי בקשה של הDMA ולא לפי טריגר של טיימר לדוגמה, באופן זה אנו מקבלים ביצוע אופטימלי של קבלת טריגרים (הDMA מבקש שליטה במידת הצורך) וחוסכים בשימוש בחומרה

לאחר מכן אנו ראינו כי קיבלנו את התוצאה הרצויה, כאשר המסקנה היא שרצוי להשתמש בDMA למשימות כאלו מפני שהוא רכיב ייעודי לכך אשר מבצע אותן מהר יותר מפני שאין צורך לביצוע תוכנית שלמה על ידי הCPU, כך אנו מקבלים ביצועים טובים יותר עם שימוש בפחות חומרה והקלה במימוש של התוכנית, בהתאם למה שראינו בהרצאות.

מגישים:
יעקב מסילתי 205671852
איילון קפל 207807025