Architetture dei Sistemi di Elaborazione

4 settembre 2024

Nome e Cognome_____ _____

Matricola_____ _____

Il codice compila senza errori: sì [x] no [ ]

Ho provato il progetto in emulazione: sì [ ] no [x]

Ho provato il progetto su board: sì [x] no [ ]

L'ambiente di debug è stato utilizzato : sì [x] no [ ]

Desidero ritirarmi [ ]

Leggere con attenzione:

- Occorre sviluppare un progetto ARM usando l'IDE KEIL µVision.
- Effettuare login su propria area al LABINF ed usare il software disponibile per editare, compilare e debuggare il codice.
- Utilizzare l'area desktop sul computer del LABINF per creare il vostro progetto.
- Utilizzare la scheda LANDTIGER o l'emulatore con tutte le non-idealità abilitate per debuggare il progetto
- Sono inibiti tutti gli accessi ad internet.
- Si possono utilizzare progetti esistenti, prelevati dalla propria chiavetta USB, ed è possibile consultare materiale cartaceo.
- Entro l'orario di consegna, occorre finalizzare il salvataggio di tutti i file (valido anche per la parte di modern architecture) e **copiarli nella propria area personale Z:/ all'interno della cartella che contiene le tracce**. Le consegne in ritardo (con file salvati oltre l'orario massimo di consegna) non vengono considerate valide e conducono in ogni caso all'insufficienza.
- In caso non sia possibile compilare con successo il progetto consegnato, la prova sarà considerata insufficiente. **Si richiede di predisporre l'ambiente di debug con le watch ed i breakpoint che permettano di seguire il flusso del programma**.

1) Si consideri una variabile **VAR**, un vettore **VETT** di N elementi, contenenti valori interi senza segno espressi su 32 bit. N è una costante dichiarata a tempo di compilazione, grande a piacimento (si consiglia di debuggare con N = 8). La variabile VAR ed il vettore VETT saranno dichiarati di tipo **unsigned int**, gli elementi di VETT sono tutti impostati a 0 all'avvio delle funzionalità.
2) Si devono generare gli elementi del vettore, utilizzando i timer TIMER1 e TIMER2:
   - Il TIMER1 deve essere programmato in modo da scatenare ciclicamente una interruzione ogni 1,7 secondi
   - Il TIMER2 deve essere programmato in modo che implementi un conteggio di 14,6 secondi usando un clock in ingresso di 50 MHz in modo ciclico ma senza scatenare interruzioni
   - Allo scadere del conteggio di TIMER1, il valore attuale di TIMER2 viene salvato in VAR.

3) Alle pressione del pulsante INT0, il valore corrente di VAR viene salvato nella prima posizione libera di VETT qualora il valore corrente sia diverso da quello salvato in precedenza. Quando VETT satura, la seguente funzione assembly (ottimizzata con istruzioni condizionali) viene eseguita:

unsigned int avg_vett(unsigned int VETT[], unsigned int dim, char* flag);

La funzione avg_vett deve:

- calcolare la media aritmetica dei valori in VETT
- contare e restituire il numero di valori in VETT il cui valore sia superiore alla media
- restituire indirettamente flag, in modo che assuma il valore 1 (uno) se il numero di valori di VETT sopra la media è pari, 0 (zero) altrimenti.

4) Il valore restituito dalla funzione deve essere presentato usando tutti i LED.
   - Se flag = 0, allora il contenuto negato dell'LSB (Least Significant Byte) del risultato della funzione viene mostrato fisso in valore binario sui LED
   - Se flag = 1, allora il LED 6 lampeggerà con un periodo di 1400 ms (acceso per 700 ms, spento per 700 ms) mentre il resto dei LED sono spenti.
5) L'output deve essere mostrato fino alla successiva esecuzione della funzione assembly. Una volta che la funziona assembler viene eseguita, VETT viene riportato a 0 (tutti gli elementi sono riportati al valore 0). Durante la visualizzazione del risultato è quindi possibile operare col INT0 per produrre i nuovi elementi di VETT.