Università degli Studi di Verona

ELABORATO ASM

Architettura degli Elaboratori

Berardo Raffaele (VR)

Falezza Alberto (VR)

Zampieri Davide (VR421649)

DESCRIZIONE DEL PROGETTO

Riprendendo la situazione presentata nell'elaborato precedente, si deve ottimizzare un codice in linguaggio C che controlla un dispositivo per la gestione intelligente del consumo di energia elettrica all'interno di un sistema domotico mediante l'uso di Assembly.
Il programma deve essere lanciato dalla linea di comando rispettando la seguente sintassi:

```
$ ./controller testin.txt testout.txt
```

Il programma legge il contenuto di *testin.txt*, che contiene in ogni riga i seguenti valori:

RESET-LOAD

- **RESET [3]**: contiene la sequenza dei comandi di riarmo degli interruttori; nell'ordine RES_GEN, RES_DW e RES_WM (senza spazi)
- **LOAD [10]**: stato di accensione (1=ON, 0=OFF) dei carichi elettrici. Ogni carico ha un suo consumo istantaneo associato. Il carico complessivo del circuito è dato dalla somma di tutti i carichi accesi contemporaneamente. L'ordine ed il consumo di ogni carico sono:

Forno	Frigo	Aspira-polvere	Phon	Lava-stoviglie	Lava-trice	4xlamp 60 W	4xlamp 100 W	HI-FI	TV
2 kW	300 W	1200 W	1 kW	2 kW	1800 W	240 W	400 W	200 W	400 W

Il programma restituisce i risultati del calcolo in *testout.txt*, in cui ogni riga contiene:

INT-TH

- **INT [3]**: indica lo stato di attivazione (1=ON, 0=OFF) degli interruttori; in ordine INT_GEN, INT_DW e INT_WM (senza spazi)
- **TH [2]**: indica la fascia di consumo istantanea

SCRITTURA DELL'APPLICAZIONE

VARIABILI

- Sono state utilizzate 3 variabili per salvare lo stato (1=ON, 0=OFF) degli interruttori. Esse sono **int_gen**, **int_dw**, **int_wm**
- È stata utilizzata 1 variabile per salvare il numero di cicli in cui il sistema si trova in overload. Tale variabile è stata chiamata infatti **overload**
- Si è deciso di utilizzare altre 2 variabili: una per tenere memorizzata la posizione che si è arrivati a controllare nella stringa di input, e un'altra per sapere in che posizione si dovrà scrivere nella stringa di output. Esse sono state chiamate **input_pos** e **output_pos**

FUNZIONI ASSEMBLY

Nel programma principale (*controller_asm.s*) richiamiamo i seguenti sottoprogrammi:

- **spento**: scrive *"000-00\n"* sulla stringa di output un carattere alla volta (nel caso in cui la macchina sia spenta)
- **carico**: calcola il carico complessivo del circuito, ossia la potenza sviluppata dagli elettrodomestici accesi al ciclo corrente
- **scrivi_int**: scrive 0 o 1 a seconda del valore dell'interruttore
- **fascia**: scrive, tramite l'utilizzo della codifica ASCII, la fascia in cui ci si trova dato un corrispondente carico

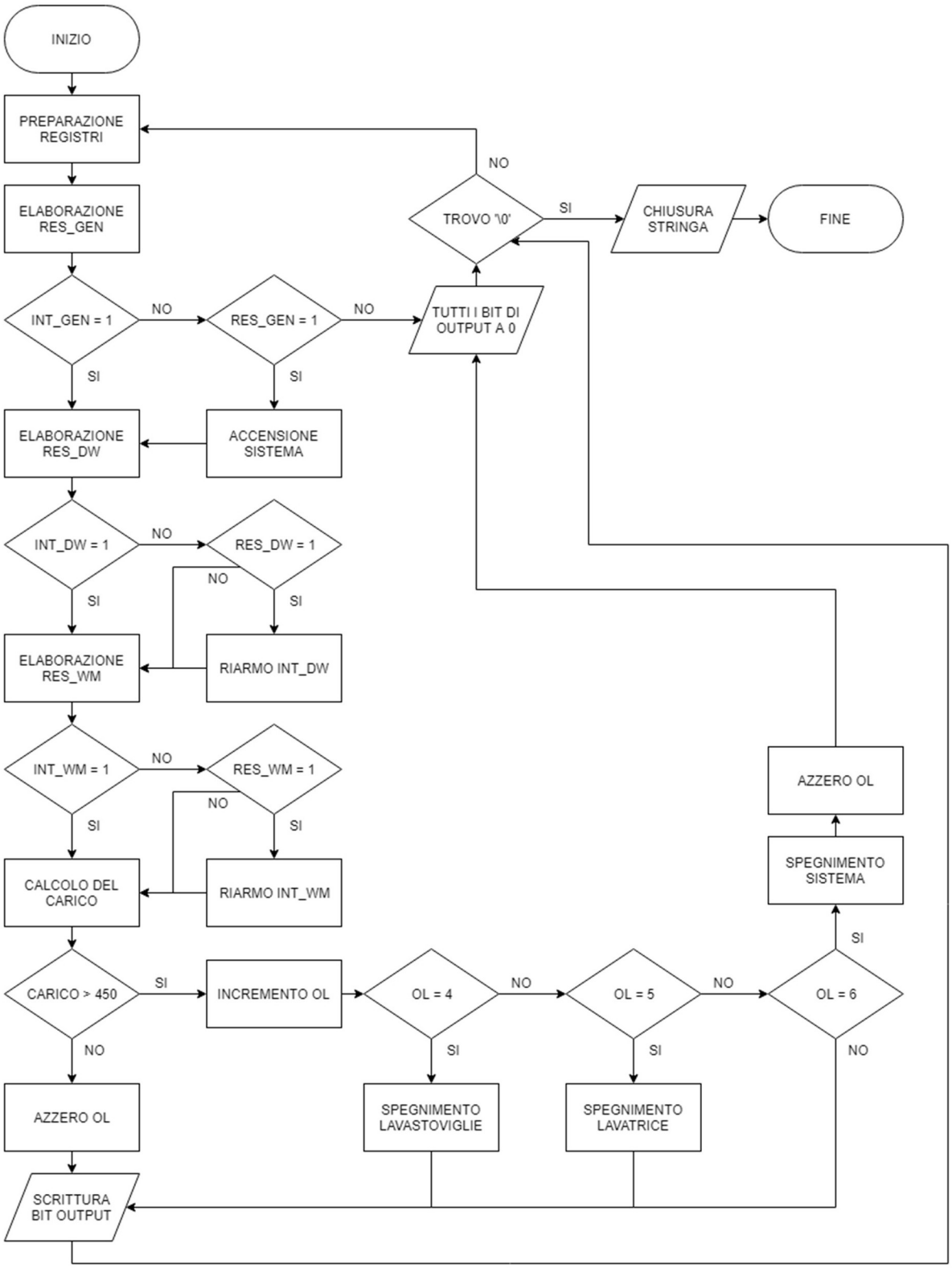
STRUTTURA DEL CODICE ASSEMBLY

Nel file *controller_asm.s* sono state utilizzate le seguenti etichette:

- **controller_asm:** scarica dallo stack i parametri
- **inizio:** inizializza i registri che verranno utilizzati
- **res_gen:** elabora il primo carattere della riga
- **res_dw:** elabora il secondo carattere della riga
- **res_wm:** elabora il terzo carattere della riga
- **calcola_carico:** chiama la funzione carico
- **controllo_ol:** aggiorna il contatore dei cicli in cui siamo in overload e controlla se è uguale a 4, 5 o 6
- **crea_output:** chiama le funzioni scrivi_int e fascia
- **ricomincia:** se il carattere successivo nella stringa di input è uguale a *'\0'* ho finito, altrimenti salto a inizio per elaborare la riga successiva

DIAGRAMMA DI FLUSSO

Il programma si compone di diverse fasi, che sono illustrate nel diagramma di flusso sottostante:



FUNZIONAMENTO DEL PROGRAMMA

Ad ogni ciclo di clock si controlla quali reset sono attivi (all'inizio del programma non si commutano gli interruttori a 1 fino a quando non viene attivato il reset generale).
Dopo la lettura degli interruttori si può calcolare il carico dato dall'accensione dei vari elettrodomestici. Questo carico, se maggiore di 450W, incrementa una variabile di overload che tiene conto del numero di cicli in cui si rimane in una situazione di sovraccarico. Nel caso in cui questa variabile raggiunga il valore di 4, si provvederà a disattivare e quindi spegnere la lavastoviglie. Se, una volta spenta, al ciclo seguente si dovesse verificare ancora una situazione di overload, la variabile sarebbe nuovamente incrementata e verrà spenta la lavatrice. Nel caso in cui si permanga ancora in questa situazione di sovraccarico, si provvederà a spegnere il sistema, tornando alla situazione iniziale.
Nel caso in cui, invece, si dovesse uscire dallo stato di overload anticipatamente (quarto o quinto ciclo) si provvederà al ciclo successivo ad azzerare la variabile di overload.

PASSAGGIO PARAMETRI DA C AD ASSEMBLY

La funzione che è stata implementata per ottimizzare il codice in linguaggio C è stata chiamata *controller_asm*, ed è stata inserita nella sezione di codice dedicata agli extern modules:

```
extern void controller_asm(char bufferin[], char bufferout_asm[]);
```

Alla funzione vengono passati due array di caratteri:

- **bufferin**, che contiene le righe di input
- **bufferout_asm**, che inizialmente è vuota e che viene poi modificata per side-effect inserendo le righe di output

Nel file Assembly corrispondente (*controller_asm.s*) i parametri vengono scaricati dallo stack e messi rispettivamente nei registri **ESI** ed **EDI**.

SCELTE PROGETTUALI

Sono state fatte le seguenti scelte progettuali:

- Quando il sistema deve rimanere spento, si somma 14 al contatore delle posizioni della stringa di input (**input_pos**) per andare a leggere direttamente la riga successiva
- I consumi dei dispositivi sono stati semplificati dividendo per 10