

UNIVERSITÀ DEGLI STUDI DI VERONA – A.A. 2019/20
CORSO DI LAUREA IN INFORMATICA

ELABORATO ASSEMBLY – ARCHITETTURA DEGLI ELABORATORI

FERRONATO MARTA - VR443649
MASSAGRANDE MARCO - VR446285
VANINI SAMANTHA - VR443381

CONTENUTO

Ottimizzazione di un codice in linguaggio C che controlla un dispositivo per la gestione intelligente di un parcheggio suddiviso in settori.

L'ottimizzazione prevede la gestione della componente esecutiva del programma mediante un modulo in linguaggio assembly (sintassi AT&T), al fine di migliorare il throughput del sistema.

Viene inoltre presentato un makefile che permette di velocizzare le funzioni di compiling, linking, permette di aggiungere rapidamente moduli aggiuntivi e di impostare celermente il setup per il debugger (ddd).

FUNZIONAMENTO

Il sistema prevede i seguenti INPUT:

- SETTORE-NPOSTI (prime 3 righe)
- TIPO-SETTORE (funzionamento automatico)

E i seguenti OUTPUT:

- SBARRE
- NPOSTIX
- LIGHTS

(La descrizione dei segnali è reperibile all'interno del file "elaborato_ASM.pdf")

Il filone esecutivo è gestito a cicli: **ogni ciclo si occupa dell'analisi di una riga di file input** (composta da un numero variabile di caratteri) **e la composizione della corrispondente riga di output** (composta da 16 caratteri, compreso il carattere \n).

I passi di esecuzione possono essere così riassunti:

1) **Inserimento del numero di posti occupati** per settore al momento dell'attivazione del sistema di gestione automatica del parcheggio. L'inserimento può avvenire in qualsiasi ordine di settore.

2) **Controllo della validità delle righe ricevute in ingresso, ed eventuale apertura della sbarra di ingresso oppure uscita.** Nel caso in cui una riga non rispetti la sintassi indicata non viene eseguita nessuna operazione e le sbarre rimarranno chiuse, così come nel caso di ingresso in parcheggio pieno.

3) **Aggiornamento del numero totale di auto presenti** all'interno del parcheggio dopo l'eventuale operazione di ingresso/uscita.

4) **Confronto del numero di auto presenti in parcheggio con la capienza massima**, nel caso in cui il parcheggio risulti totalmente occupato viene posto ad 1 il segnale LIGHTS relativo al settore in oggetto.

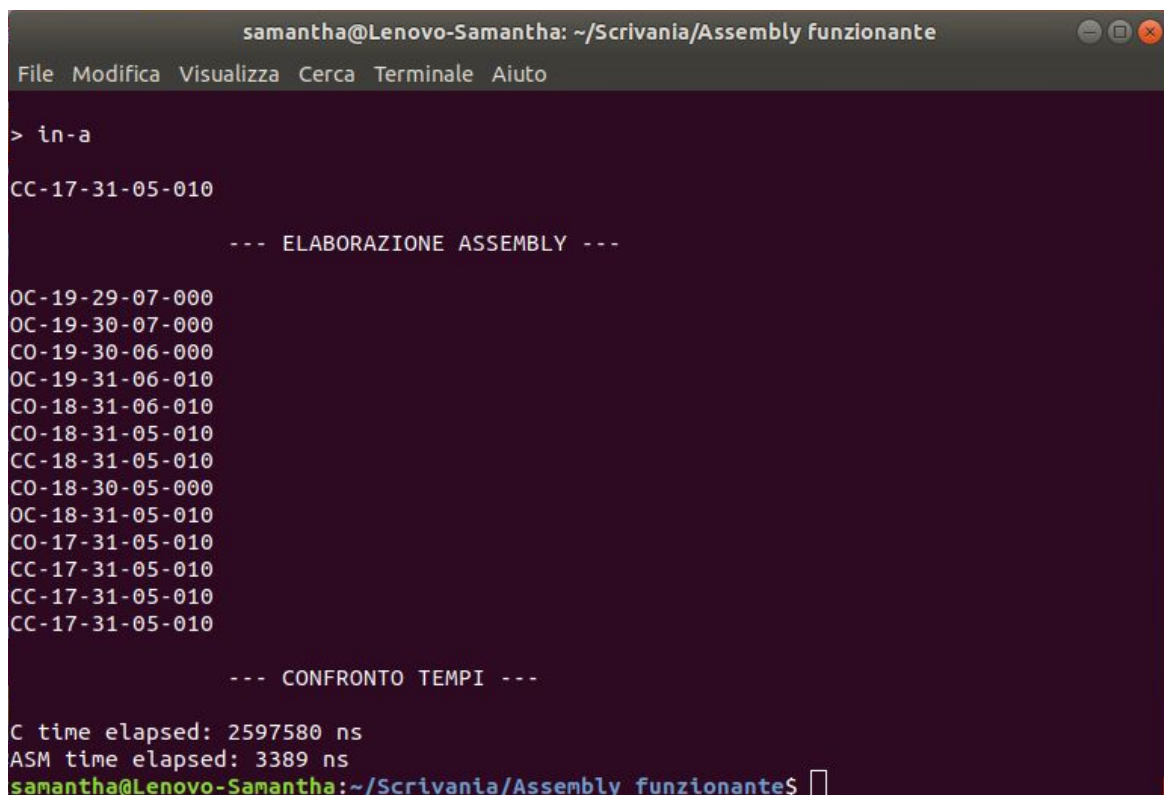
VERSIONE FINALE

La versione finale compie tutti i passaggi, descritti nella parte relativa al funzionamento, in un unico file con il fine di massimizzare l'incremento di prestazioni.

Il makefile è stato realizzato in questa direzione, ma è sempre integrabile con eventuali moduli e funzioni aggiuntive oltre a quelle già inseriti.

Le statistiche sulle prestazioni della versione finale figurano nella sezione “prestazioni” di questo documento.

PRESTAZIONI



```
samantha@Lenovo-Samantha: ~/Scrivania/Assembly funzionante
File Modifica Visualizza Cerca Terminale Aiuto

> in-a

CC-17-31-05-010

    --- ELABORAZIONE ASSEMBLY ---

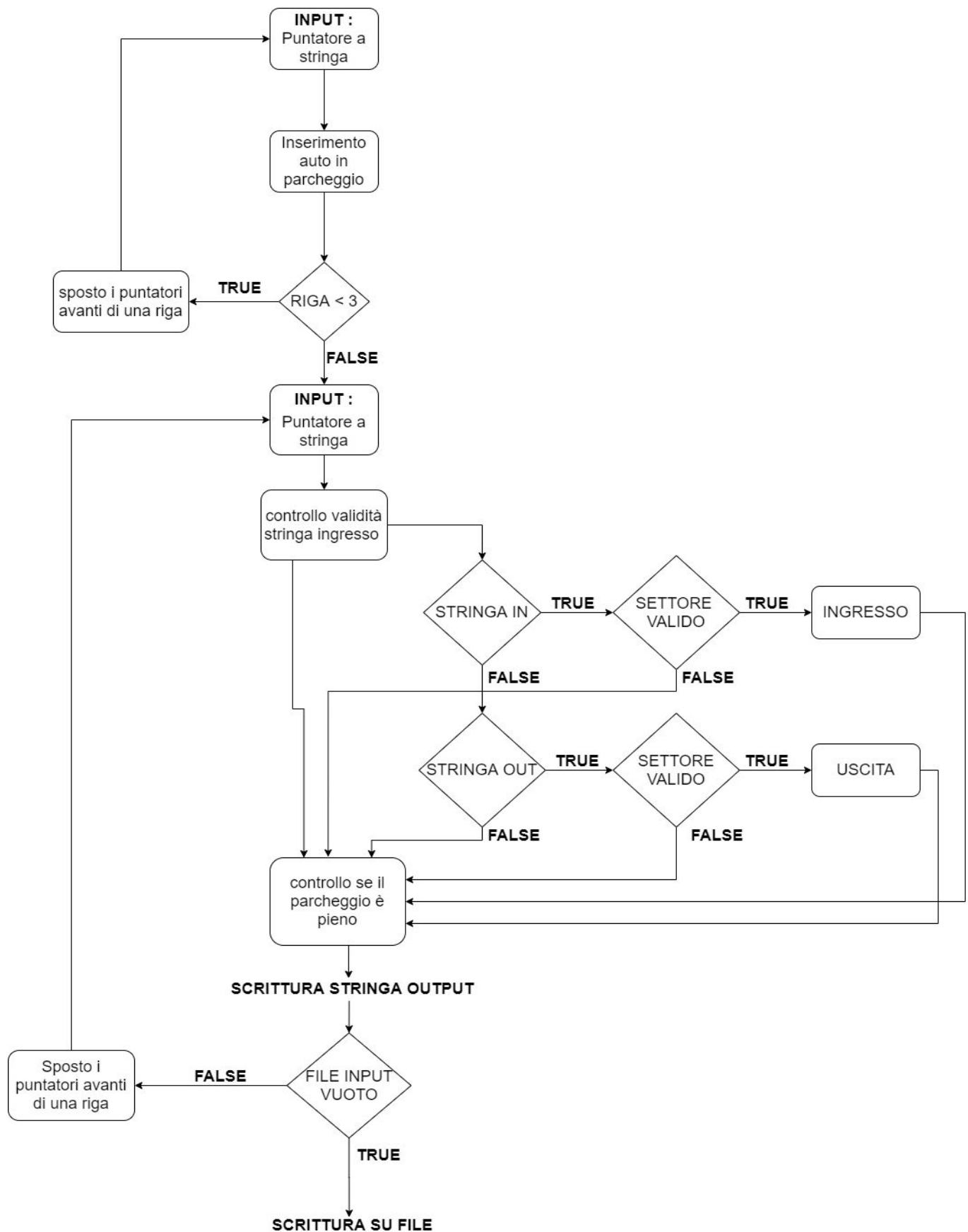
OC-19-29-07-000
OC-19-30-07-000
CO-19-30-06-000
OC-19-31-06-010
CO-18-31-06-010
CO-18-31-05-010
CC-18-31-05-010
CO-18-30-05-000
OC-18-31-05-010
CO-17-31-05-010
CC-17-31-05-010
CC-17-31-05-010
CC-17-31-05-010

    --- CONFRONTO TEMPI ---

C time elapsed: 2597580 ns
ASM time elapsed: 3389 ns
samantha@Lenovo-Samantha:~/Scrivania/Assembly funzionante$
```

Nella versione finale del progetto, l'incremento a cui si è arrivati (nel caso ottimo) è pari al **76647.39%**

DIAGRAMMA DEL FUNZIONAMENTO



DESCRIZIONE DELLE SCELTE PROGETTUALI

- L'implementazione della parte in assembly avviene tramite una funzione esterna di tipo void che lavora utilizzando gli indirizzi dei vettori **BUFFERIN** e **BUFFEROUT_ASM**, utilizzando il passaggio per side-effect.
- Il valore delle prime 3 righe, in cui si inserisce il numero di posti occupati all'avvio, si assume che sia compreso tra i range di valori validi, così come si assume che la sintassi di queste righe sia sempre corretta.
- Nel caso in cui si tenti di uscire da un parcheggio vuoto, le sbarre saranno mantenute abbassate e non avverrà nessun tipo di operazione sui valori dei parcheggi
- All'interno del codice assembly è stata dichiarata, all'interno della sezione .data, una sola costante contenente il numero 10, utilizzato durante le fasi di conversione da intero a carattere (e viceversa)
- Il numero dei posti occupati è stato mantenuto all'interno di registri : %eax per A, %ebx per B ed %ecx per C