UNIVERSITÀ DEGLI
STUDI DI VERONA

ARCHITETTURA DEGLI ELABORATORI

Elaborato Assembly - A.A. 2021/2022

BOTTEGA MARICA - VR474005
VINCENZI BRUNO - VR471499
ZAMUNER SERENA - VR472442

Indice

Testo dell’elaborato

Descrizione:

Si descriva un programma che simuli il sistema di telemetria del videogame F1.
Il sistema fornisce in input i dati grezzi di giri motore (rpm), temperatura motore e velocità di tutti i piloti presenti in gara per ogni istante di tempo.
Ogni campo è diviso da una virgola usata come separatore.
Ogni riga del file di input è così composta:
<tempo>,<id_pilota>,<velocità>,<rpm>,<temperatura>

Obiettivo:

Si scriva un programma in Assembly che restituisca i dati relativi al solo pilota indicato nella prima riga del file, in base a delle soglie indicate.
Vengono definite tre soglie per tutti i dati monitorati: LOW, MEDIUM, HIGH.
Il file di output dovrà riportare queste soglie per tutti gli istanti di tempo in cui il pilota è monitorato.
Le righe del file di output saranno strutturate nel seguente modo e ordine:
<tempo>,<livello rpm>,<livello temperatura>,<livello velocità>
Inoltre, viene richiesto di aggiungere alla fine del file di output una riga aggiuntiva
che contenga, nel seguente ordine: il numero di giri massimi rilevati, la temperatura
massima rilevata, la velocità di picco e infine la velocità media.
La struttura dell’ultima riga sarà quindi la seguente:
<rpm max>,<temp max>,<velocità max>,<velocità media>

Variabili

Nel programma, oltre ad aver utilizzato le variabili relative ai nomi e cognomi dei piloti fornite dal testo dell'elaborato, abbiamo utilizzato le variabili .string LOW, MEDIUM e HIGH per la stampa dei livelli dei parametri (velocità, rpm, temperatura).
Non abbiamo utilizzato ulteriori variabili perché abbiamo scelto di implementare l'intero programma tramite l'utilizzo dei registri e dello stack.

Funzioni

Sommario delle funzioni utilizzate nel programma:
Funzione **atoi**
Funzione **virgola**
Funzione **find_max_rpm**
Funzione **find_max_temp**
Funzione **find_max_vel**
Funzione **write_time**
Funzione **write**
Funzione **write_end**

ATOI:
La funzione ATOI converte la stringa puntata dal registro ESI in un intero, salvandolo poi nel registro EDX.
Nel programma viene utilizzata più volte per controllare se il parametro ID è quello associato al pilota inserito nella prima riga e se i parametri velocità, temperatura e rpm sono di livello LOW, MEDIUM oppure HIGH.

VIRGOLA:
La funzione VIRGOLA aumenta lo spiazzamento della stringa puntata dal registro ESI in modo tale che il registro ECX sommato al registro ESI punti alla prima virgola che trova.

FIND_MAX_RPM:
Nello stack è presente il numero di giri motori massimi che viene aggiornato con la funzione FIND_MAX_RPM. Quando questa funzione viene chiamata, confronta il dato presente nello stack e il dato candidato in input. Se il candidato è maggiore, lo stack viene aggiornato.

FIND_MAX_TEMP:
Nello stack è presente la temperatura massima che viene aggiornata con la funzione FIND_MAX_TEMP. Quando questa funzione viene chiamata, confronta il dato presente nello stack e il dato candidato in input. Se il candidato è maggiore lo stack viene aggiornato.

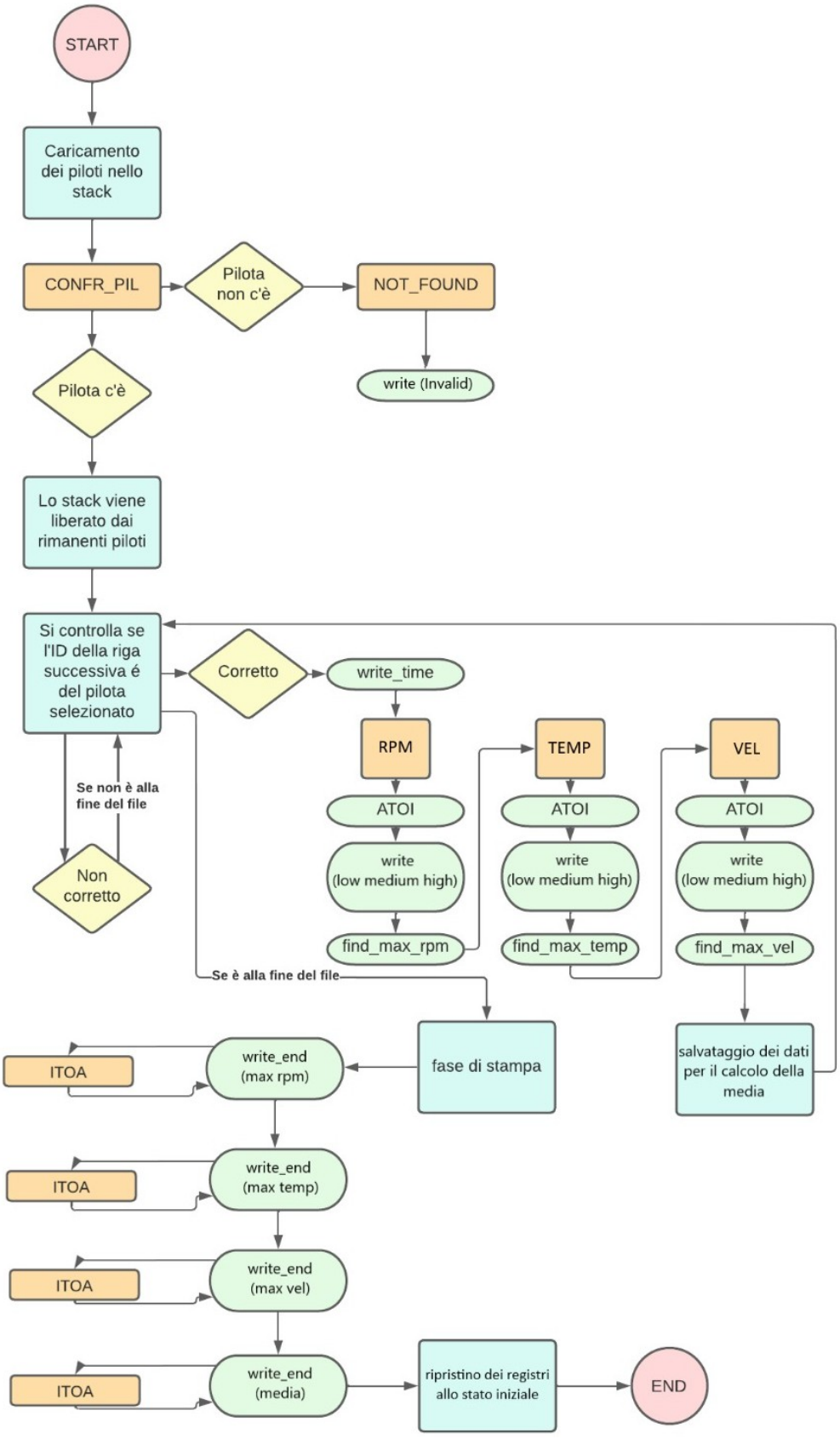
FIND_MAX_VEL:
Nello stack è presente la velocità massima che viene aggiornata con la funzione FIND_MAX_VEL. Quando questa funzione viene chiamata, confronta il dato presente nello stack e il dato candidato in input. Se il candidato è maggiore lo stack viene aggiornato.

WRITE_TIME:
La funzione WRITE_TIME prende il tempo dal registro ESI, che punta al file di input, e lo scrive nel registro EDI, il quale punta al file di output.

WRITE:
La funzione WRITE scrive nel registro EDI, puntatore al file di output, ciò a cui punta il registro ESI: il livello (HIGH - LOW - MEDIUM) di rmp, temperatura e velocità, o invalid,

WRITE_END:
La funzione WRITE_END converte in stringa ciò che è contenuto nel registro EAX e lo scrive in EDI, che punta al file di output, stampando così i massimi (max rpm, max temperatura e max velocità) e la media.

Diagramma di flusso



Codice in C

```c
#include <stdio.h>
#include <string.h>
#define N 20 //lunghezza max delle stringhe e numero dei piloti

//struttura del pilota
typedef struct {
    char  [N];
}pilota_t;

int livello_girimotore(int rpm);
int livello_temperatura(int temp);
int livello_velocita(int speed);

int main(void){
    char pilotaIn[N];
    int check = 0;
    int id;
    //array di strutture per dichiarare i piloti in gara
    pilota_t pilota[N] = {
        {"Pierre Gasly\0"},
        {"Charles Leclerc\0"},
        {"Max Verstappen\0"},
        {"Lando Norris\0"},
        {"Sebastian Vettel\0"},
        {"Daniel Ricciardo\0"},
        {"Lance Stroll\0"},
        {"Carlos Sainz\0"},
        {"Antonio Giovinazzi\0"},
        {"Kevin Magnussen\0"},
        {"Alexander Albon\0"},
        {"Nicholas Latifi\0"},
        {"Lewis Hamilton\0"},
        {"Romain Grosjean\0"},
        {"George Russell\0"},
        {"Sergio Perez\0"},
        {"Daniil Kvyat\0"},
        {"Kimi Raikkonen\0"},
        {"Esteban Ocon\0"},
        {"Valtteri Bottas\0"}
    };
    //input del pilota che si vuole monitorare
        scanf ("%[^\n]%*c", pilotaIn);
         for(int i = 0; i < N; i++){
             if(strcmp(pilotaIn, pilota[i].nomeCognome) == 0){
                 check = 1;
                 id = i;
                 break;
             }
        }
        if(check == 0){   //se il pilota non esiste ritorna invalido
            printf("Invalid\n");
            return 0;
        }
//-------------------------------------------------------------
    int temperatura = 0, vel = 0, rmp = 0, idk = 0;
```

```
    double tempo;
    char lvltemp = 0, lvlrmp = 0, lvlvel = 0;
    int maxtemp = -1, maxrmp = -1, maxvel = -1;
    int somma = 0, num = 0;
    do{
        scanf("%lf %d %d %d %d", &tempo, &idk, &vel, &rmp, &temperatura);
       //presa degli input finchè il tempo non è uguale a 0

        //livello dei giri del motore
        if(livello_girimotore(rmp) == 0)
      lvlrmp = 'L';
        if(livello_girimotore(rmp) == 1)
      lvlrmp = 'M';
        if(livello_girimotore(rmp) == 2)
      lvlrmp = 'H';

        //livello della temperatura
        if(livello_temperatura(temperatura) == 0)
      lvltemp = 'L';
        if(livello_temperatura(temperatura) == 1)
      lvltemp = 'M';
        if(livello_temperatura(temperatura) == 2)
      lvltemp = 'H';

        //livello della velocità
        if(livello_velocita(vel) == 0)
      lvlvel = 'L';
        if(livello_velocita(vel) == 1)
      lvlvel = 'M';
        if(livello_velocita(vel) == 2)
      lvlvel = 'H';

        if(id == idk){
            printf("%.5lf, %c, %c, %c\n", tempo, lvlrmp, lvltemp, lvlvel);
            num++;
            somma+=vel;
        }

        if(maxtemp < temperatura) maxtemp = temperatura;
        if(maxvel < vel) maxvel = vel;
        if(maxrmp < rmp) maxrmp = rmp;

    }while(tempo != 0); //finché il tempo in input non è 0

    //stampa dell'ultima riga
    printf("%d, %d, %d, %d\n\n", maxrmp, maxtemp, maxvel, somma/num);

    //fine programma
    return 0;
}

//-----------------------------------------------------------------
//funzioni per calcolare il livello
int livello_girimotore(int rpm){
     if(rpm <= 5000)
         return 0;
     else if(rpm > 5000 && rpm <= 10000)
         return 1;
     else if(rpm > 10000)
         return 2;
```

```
    return 333;
}
int livello_temperatura(int temp){
    if(temp <= 90)
        return 0;
    else if(temp > 90 && temp <= 110)
        return 1;
    else if(temp > 110)
        return 2;
    return 333;
}
int livello_velocita(int speed){
    if(speed <= 100)
        return 0;
    else if(speed > 100 && speed <= 250)
        return 1;
    else if(speed > 250)
        return 2;
    return 333;
}
```

Scelte progettuali

- Non abbiamo utilizzato variabili di tipo intero ma abbiamo implementato il programma solo attraverso l’uso dei registri e dello stack.

- Abbiamo deciso di scrivere il codice in un unico file per questioni di gestibilità.

- Abbiamo preso in considerazione il caso in cui se il pilota è valido ma non ci sono righe corrispondenti al suo ID allora verranno stampati degli zero in corrispondenza dei vari valori.