UNIVERSITÀ DEGLI STUDI DI VERONA

ELABORATO ASM

Architettura degli Elaboratori
Laboratorio

Corso di Laurea in Informatica

A.A. 2022/2023

PROGETTO:
Progettazione di un menu per la gestione del menu cruscotto di un'automobile

Studenti:

Alessia	**Gennari**	VR488137
Mattia	**Pacchin**	VR461870

SOMMARIO

SPECIFICA DEL PROGETTO

Il presente elaborato descrive la progettazione di un codice in Assembly AT&T per la gestione di un cruscotto di un'automobile.

CARATTERISTICHE DEL CODICE

Il menù si può accedere in due modalità: utente e supervisore.

Nella modalità utente è possibile visualizzare il menu composto nel seguente modo:

1. Setting automobile:
2. Data: 15/06/2014
3. Ora: 15:32
4. Blocco automatico porte: ON
5. Back-home: ON
6. Check olio

Nella modalità supervisore (si accede aggiungendo al nome dell'eseguibile il codice 2244) il menu visualizzato sarà il seguente:

1. Setting automobile (supervisor):
2. Data: 15/06/2014
3. Ora: 15:32
4. Blocco automatico porte: ON
5. Back-home: ON
6. Check olio
7. Frecce direzione
8. Reset pressione gomme

In entrambe le modalità sarà possibile spostarsi tra i vari menu con le frecce su e giù).
Premendo la freccia a destra sarà possibile modificare i punti 4, 5, 6, 7 e 8.

- *Per i punti 6 e 8 verrà visualizzato un messaggio di conferma.*
- *I punti 4 e 5 potranno essere modificati premendo freccia in su o freccia in giù, questo modificherà il loro valore da ON a OFF e viceversa.*
- *Per modificare il punto 7 si richiede di inserire un numero, di default è 3 il massimo è 5 ed il minimo 2, se verrà inserito un numero maggiore di cinque verrà salvato cinque, se invece viene inserito un numero minore di due verrà salvato due.*

SUDDIVISIONE DEI FILE

Il codice assembly è diviso in due file.

main : contiene le variabili globali, controlla se è stata attivata la modalità supervisore e dà il via al funzionamento del cruscotto chiamando una prima volta l'etichetta *index_position_message* e poi entra nel loop che chiama l'etichetta navigate_menu.

navigate_menu : richiama la funzione move, se questa ha dato risultato due controlla che l'index corrente sia modificabile in caso positivo viene messo un flag a true. In caso move non dia come risultato 1 per la variabile submenu (quindi 1 o 0 nel registro %dl) verrà aggiornata la variabile index con l'indice del menù in cui ci troveremo (tenendo conto se si è oppure no in modalità supervisore).

move : è la funzione che si occupa di prendere in input le frecce e restituire 1 o 0 nel registro %dl (freccia su o giù), 1 nel registro %cl per freccia a destra.

set_blinkers : prende in input il numero desiderato di blinks, controlla se è nel range corretto (da 2 a 5). Se è maggiore di cinque verrà impostato a cinque; se è minore di due verrà impostato a due.

index_position_message : uno switch case che a seconda della posizione dell'index visualizza l'impostazione corretta del menù. Nel caso in cui ci troviamo in una voce che ha un sottomenù, viene effettuato un controllo per verificare se è stato premuto il tasto freccia destra e in caso positivo richiama la funzione per poter modificare il valore (es. per modificare i lampeggi richiama l'etichetta set_blinkers).

VARIABILI

DESCRIZIONE DELLE VARIABILI ED IL LORO SCOPO

Riportiamo di seguito le variabili principali per il corretto funzionamento del cruscotto:

ind : variabile globale che salva la posizione (index) nel menù. Inizialmente ad 1, viene modificata dalla funzione navigate_menu .

submenu : variabile globale che assume valore 1 se è stato premuto il tasto freccia destra per una funzione del menù modificabile, altrimenti vale 0.

door_lock : variabile globale che salva il valore della funzionalità del menu *blocco automatico porte* (4). Di default a 1, che corrisponde ad ON, se viene premuta freccia destra e poi freccia su/giù il valore si modifica e va a 0 (OFF). Se invece il valore salvato è zero, premendo freccia su/giù la variabile va a uno.

back_home : variabile globale che contiene il valore della funzionalità *Back-Home* del menù. Si modifica con lo stesso funzionamento della variabile door_lock.

blinkers : variabile globale che salva il numero di lampeggi dell'automobile in autostrada (funzione *frecce direzione*). Questa variabile viene modificata dalla funzione set_blinkers.

Per ogni funzione vengono usate delle altre variabili temporanee che servono per salvare per breve tempo i valori, fare calcoli e poi restituirlo.

DESCRIZIONE DELLE MODALITA' DI PASSAGGIO DELLE VARIABILI TRA FUNZIONI

Non vengono passate dal *main.s* al *move.s* e viceversa delle variabili in quanto gran parte del codice viene eseguita dal nel *main.s*.

Tuttavia passiamo due valori dal file *move.s* al file *main.s*: nel registro %dl salviamo il valore 1 o 0 che ci permette di decidere se muoverci in su o in giù nel menù; nel registro %cl salviamo il valore che ci permette di sapere se dovremo andare o meno nel sottomenù (freccia destra).

ATTUAZIONE

PSEUDO-CODICE AD ALTO LIVELLO

```
# Pseudocodice

// initiate some global variables shared by all functions
ind = 1 // index var
sub = 0 // submenu var
door_lock = 1 // bool, true if doors auto lock is on
back_home = 1 // bool, true if back home mode is on
blinkers = 3 // how many times blinkers blink (supervisor mode)

### MAIN(terminal input) // return int
    supervisor = 0
    if terminal input = 2244
        supervisor = 1

    index_position_message(supervisor);

    while (true)
        navigate_menu(supervisor)

    return 0

### INDEX_POSITION_MESSAGE(supervisor)
    switch (ind)
        case 1:
            if (supervisor = 1)
```

```
                print "1. Setting automobile (supervisor):"
            else
                print "1. Setting automobile:"
        case 2:
            print "2. Data: 15/06/2014"
        case 3:
            print "3. Ora: 15:32"
        case 4:
            if (sub)
                read = move()
                if (read = -1 or read = 1)
                    door_lock = not(door_lock)
                sub = 0
            if (door_lock)
                print "4. Blocco automatico porte: ON"
            else
                print "4. Blocco automatico porte: OFF"
        case 5:
            if (sub)
                read = move();
                if (read = -1 or read = 1)
                    back_home = not(back_home)
                sub = 0
            if (back_home)
                print "5. Back-home: ON"
            else
                print "5. Back-home: OFF"
        case 6:
            print "6. Check olio"
        case 7:
            if (sub)
                set_blinkers()
                sub = 0
            print "7. Frecce direzione: *blinkers*"
        case 8:
            if (sub)
                print "Pressione gomme resettata"
                sub = 0
            print "8. Reset pressione gomme"

### NAVIGATE_MENU(supervisor)
    read = move()
    if (pressed right arrow and index has a submenu)
        sub = 1
    else if (pressed up or down)
        ind = ind + read
        if (supervisor = true)
            if (ind < 1)
```

```
                ind = 8
            else if (ind > 8)
                ind = 1
        else
            if (ind < 1)
                ind = 6
            else if (ind > 6)
                ind = 1
    index_position_message(supervisor);

### MOVE() // return int
    get char c

    if (c = up)
        return -1
    else if (c = down)
        return 1
    else if (c = right)
        return 2

    return 0

### SET_BLINKERS()
    get n
    if (n > 5)
        blinkers = 5
    else if (n < 2)
        blinkers = 2
    else
        blinkers = n
```

SCELTE PROGETTUALI

PARTE GRAFICA

- **Visualizzazione voci del menu**: quando si scorre il menu si possono vedere le diverse voci con già il loro valore attuale (es. 2. Data: 15/06/2014 o 5. Back-Home: ON) senza dover andare nel sottomenu. Abbiamo fatto questa scelta dopo aver letto l'elaborato e abbiamo ritenuto che così facendo sarebbe stato più semplice visualizzare il valore delle variabili
- **Ristampa voce menu con valore modificato**: quando si va nel sottomenu di back-home, di blocco automatico porte e frecce direzione è possibile cambiare l'attuale valore delle variabili. Una volta completata l'operazione la voce viene visualizzata

nuovamente con il valore modificato. Abbiamo fatto questa scelta per poter dare un feedback immediato all'utilizzatore dell'avvenuta modifica del campo senza doverci ritornare.

CODICE

- **Divisione delle funzioni in due file**: abbiamo scelto di dividere le funzioni in due file per avere più chiarezza all'interno del codice.

CRITICITA' RISCONTRATE E SOLUZIONI

- **Passare le variabili tra funzioni**: se fosse stato un unico file sarebbe bastato salvare di volta in volta i valori modificati nelle variabili locali, ma nel nostro caso, con le funzioni in due file, questo non era possibile. Per questo abbiamo scelto di usare i registri %cl e %dl già precedentemente nominati per passare dei valori dal file *move.s* al file *main.s*.
- **Prendere gli elementi da cmd**: gli elementi passati da linea di comando vengono salvati dentro lo stack ed ognuno ha la sua posizione. Per controllare se ci fossero elementi oltre all'eseguibile è stato necessario prendere l'elemento interno a %esp, mentre per prendere il codice bisognava spostarsi di 8 bit da %esp e prendere il valore contenuto all'indirizzo.
- **Frecce**: per prendere come input da tastiera freccia su, giù e destra sono servite tre call ad interrupt ed ognuna prendeva una parte del codice dell'input. L'ultima contiene la lettera collegata al verso della freccia e ci permette di riconoscere l'input impartito.