Integrazione di Sistemi Embedded
Laboratorio 03

Matteo Perotti 251453
Giuseppe Puletto
Luca Romani
Giuseppe Sarda

November 16, 2018

1 Esercizio 1

Il programma si compone di tre files .c e tre header files. Un ciclo infinito si occupa di leggere il prossimo carattere in arrivo salvandolo nella posizione più a destra di un array di caratteri lungo quanto il numero di caratteri del comando previsto più lungo. Prima della memorizzazione del carattere i dati in ogni cella dell'array vengono spostati di una cella a sinistra.

Ogni volta che un nuovo carattere è stato memorizzato, la funzione readCommand analizza il buffer per vedere se può essere arrivato un comando valido. Questa operazione è svolta andando a controllare la cella dell'array nella quale, se fosse stato ricevuto il comando più corto (draw a point), sarebbe salvato il carattere identificativo del comando stesso. In caso di mancato riscontro viene analizzata la prima cella dell'array, ovvero quella in cui potrebbe essere presente una delle altre due lettere identificative di un comando valido.

In caso di riscontro, i parametri relativi al comando vengono salvati nella struttura apposita, convertendo ogni carattere in intero. In seguito viene fatto un controllo sui dati salvati, per vedere se sono parametri validi, ossia se ogni coordinata è un numero compreso tra 0 e 127, e se il "modo" è un numero compreso tra 0 e 2.

Se il comando non viene riconosciuto o se i parametri non sono corretti, la funzione restituisce 0 e viene letto il prossimo carattere.

Nel caso in cui il comando venga riconosciuto e possegga parametri validi, allora viene chiamata la funzione corrispondente.

1.1 main.c

Il file main.c contiene la definizione della funzione main, la funzione principale del programma. Essa si occupa di definire l'array di char cmdBuffer in cui viene memorizzato un carattere alla volta e la struttura in cui i parametri del comando vengono memorizzati. Prima del ciclo, l'array di caratteri viene inizializzato con null characters per evitare che valori casuali possano portare a riconoscimenti errati di comandi mai ricevuti.

Nel ciclo infinito viene traslato il buffer di arrivo e viene salvato il nuovo carattere nell'ultima posizione. Il buffer è quindi passato alla funzione readCommand insieme alla struttura; a seconda dell'intero ritornato da quest'ultima funzione viene controllato il comando valido ed invocata la funzione corrispondente, oppure il ciclo riprende da capo.

Nel caso in cui la funzione readCommand dovesse ritornare una lettura corretta ma non fosse stato salvato un comando valido nella struttura, la funzione main restituisce un 1 per segnalare un errore.

1.2 read.c and read.h

Nel file header read.h sono presenti i comandi #define per aumentare la leggibilità e la manutenibilità del codice, insieme con la dichiarazione del tipo "basicCmd", ossia un tipo-struttura utile per la variabile in cui saranno salvati i vari parametri del comando, e la dichiarazione delle funzioni relative alla lettura del prossimo carattere dal periferico di input, al riconoscimento del comando e al salvataggio dei relativi parametri e alla conversione di un carattere ad intero.

readChar(void) La funzione ritorna il carattere che viene letto dal periferico di input, in questo caso lo standard input. Viene effettuata una chiamata a funzione "getc()" con arcomento "stdin" (il tutto corrisponde a "getchar()").

char2int(char charIn) La funzione riceve un carattere in ingresso e restituisce uno short int che corrisponde alla cifra rappresentata in ASCII. Se il carattere non è una cifra da 0 a 9, viene restituito 10, perché è un valore che aiuta il parser a capire se i parametri dell'ipotetico comando sono validi oppure no.

readCommand(char* cmdBuffer, basicCmd* cmdStruc_pt) La funzione riceve l'array buffer in cui è salvato il possibile comando, insieme con la struttura dati in cui i parametri del comando vengono salvati.

La funzione si occupa di controllare nelle posizioni chiave se è presente un identificativo del comando. Se è presente, la struttura viene aggiornata grazie all'utilizzo di char2int(char charIn).

In seguito viene eseguito un controllo sui parametri appena aggiornati: se sono validi, allora viene ritornato un 1.

In caso il comando non sia riconosciuto come completamente valido, viene ritornato un 1.

1.3 draw.c and draw.h

drawPoint

drawLine

drawEllipse

2 Esercizio 2

2.1 Dimensione del codice

x86_64

ARM

2.2 Script