CORSO DI ARCHITETTURA DEGLI ELABORATORI
PROGRAMMAZIONE ASSEMBLY: PROGETTO

26 LUGLIO 2007

CONSEGNA Sabato 15 settembre 2007

Lo studente programmi nel linguaggio Assembly 8088 presentato a lezione un micro 8088 incaricato di controllare un sistema di riconoscimento degli accessi.

Descrizione del sistema
Il sistema è schematizzato in Figura 1. Il blocco 8088 incorpora anche la logica di controllo delle varie periferiche. I segnali di controllo sono indicati in tratteggio.

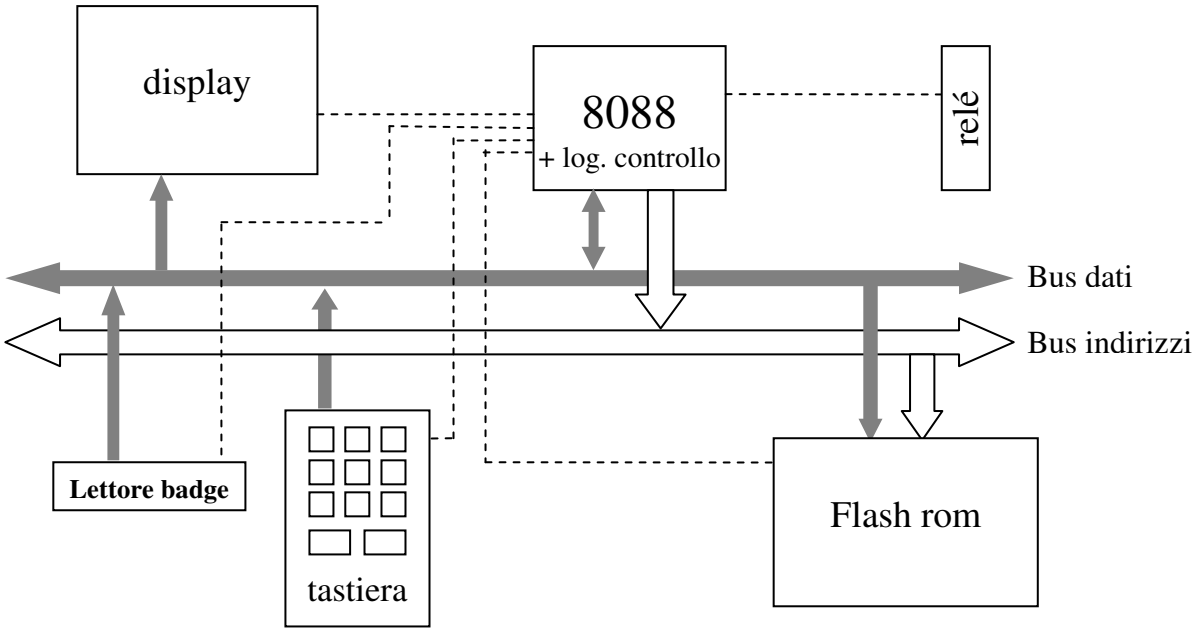

Figura 1: Descrizione del sistema.

Compito dello studente è esclusivamente la programmazione in Assembly del micro. Il sistema è composto da due dispositivi di input (lettore di badge e tastiera), due dispositivi di output (display e relé) e una memoria di massa (flash rom).
Il sistema controlla l'apertura di una porta. L'utente inserendo il badge nel lettore avvia la procedura di verifica a tre passi.
Passo 1: lettura del *nome_utente* (alfanumerico) dal badge
Passo 2: richiesta di inserimento del *pin* tramite tastiera
Passo 3: verifica esistenza utente nel database contenuto nella flash rom e della corretta associazione *nome_utente* ↔ *pin*; apertura della porta azionando il relé.

Display e tastiera
La lettura della tastiera e la scrittura sul display sono realizzate con le cinque funzioni implementate nell'interprete (Figura C.7 del libro di testo). Tastiera e display sono emulati dalla consolle dell'elaboratore usato.
Output:

- Mostra il display e il tastierino numerico
- Il display varia a seconda dello stato del sistema. Il tastierino non cambia. *-nota: non avendo a disposizione un controllo a matrice del display bisogna riscrivere il display ogni volta che cambia di stato. Ecco perché il display varia a seconda dello stato e il tastierino numerico viene sempre visualizzato allo stesso modo. Non essendo possibile svuotare lo schermo è opportuno inserire molte righe bianche (ad esempio 100) prima di riscrivere lo schermo. -*

Input

- Il tastierino è composto da 9 tasti con i numeri da 1 a 9, un tasto per lo 0, uno per INVIA e uno per CANCELLA..

La Figura 2 è un esempio di ciò che visualizza il display aspettando che un utente inserisca il badge. Il display è 32x8 caratteri e circondato da *, come in Figura 1.
- l'area utile del display è quindi 30x6 caratteri. -

Figura 2: Esempio di schermata del display quando il sistema si trova nello stato iniziale: attesa che l'utente inserisca il badge.

Comportamento del programma

- Il programma non termina mai: aspetta un utente, autentica, torna ad aspettare un utente.
- Bisogna gestire i messaggi di errore: utente inesistente, password errata, ...
- Mostrato l'errore si torna al normale funzionamento.

Database
Simulare la flash rom (memoria non-volatile!) con la memoria principale e le operazioni sui file.
- suggerimento: forse serve una procedura di inizializzazione per la rom simulata caricando i dati da un file. Ad esempio il file rom.txt *, dove ogni riga contiene due stringhe, una corrispondente al nome utente e l'altra alla password, e rappresenta un'entry del database. Serve un'ulteriore terza*

stringa che abbia funzione di chiave numerica? Si lascia la risposta al progettista! Un possibile contenuto di rom.txt *potrebbe essere:*

```
monteirorossi     564t38my
34luca45u         23125552
lucacrupogno      34et4t17
```

oppure

```
1 monteirorossi 564t38my
2 34luca45u 23125552
3 lucacrupogno 34et4t17
```

Lettore

Il lettore di badge e il relé sono hardware non disponibile sull'elaboratore che lo studente usa per emulare il sistema: sarà necessario emularne il comportamento.

Il lettore di badge è una periferica che genera un segnale asincrono catturato dal micro e in seguito scrive il *nome_utente*, riportato nel badge, sul bus dati. La lettura del dato avviene con l'istruzione IN:

IN registro, indirizzo_periferica_di_input

Il registro di destinazione è AX, o AL, AH.
La IN è simile ad una MOV tra un registro e un indirizzo di memoria. La MOV e la IN agiscono sul segnale di controllo IO/M che abilita selettivamente l'I/O o la Memoria. Questo meccanismo permette di usare il bus dati e il bus indirizzi per accedere sia alla memoria sia all'I/O.
Lo studente emuli il comportamento del lettore come di seguito specificato. Si usi la testiera per generare l'evento asincrono: la pressione del tasto associato alla lettera **x**.
*- Il micro deve controllare - nel caso nostro - lo standard input aspettando di leggere la lettera **x** e scartando qualsiasi altra lettera. Cosa succede quando si inserisce un badge male? Il sistema genera un messaggio di errore tipo "Inserire la carta nel verso corretto". Un comportamento simile si deve avere premendo una lettera diversa dalla **x**. -*
Si associ un indirizzo al lettore scegliendolo nello spazio degli indirizzi di I/O e si scriva la procedura assembly **IN_2** che emuli l'istruzione IN. La IN_2, se invocata con l'indirizzo associato al Lettore, legge il *nome_utente* dal file testuale *badge.txt* e lo scrive nel registro indicato.
- ad esempio badge.txt contiene la stringa `34luca45u` *oppure la stringa* `lucacrupogno` -

Relé

Il relé, come il Lettore, è *mappato* nell'area degli indirizzi riservati all'I/O. Si genera quindi il segnale per aprire la porta tramite l'istruzione OUT:

OUT indirizzo_periferica_di_output, registro

Il registro sorgente è AX, o AL, AH.
La OUT, come la IN, è simile ad una MOV tra un registro e un indirizzo di memoria. Anche la OUT agisce sul segnale di controllo IO/M.
Lo studente emuli questo comportamento associando un indirizzo al relé e scrivendo la procedura **OUT_2** che emula l'istruzione OUT. La OUT_2, se invocata con l'indirizzo associato al relé, segnala l'apertura della porta scrivendo nel file testuale di log *porta.log* il *nome_utente* appena abilitato all'ingresso. Il file di log è inizializzato vuoto all'avvio del sistema.

NOTA: quanto non specificato in questo testo fa parte delle scelte del progettista, quindi dello studente!!! Si riportino le scelte, commentandole, nella relazione.

SI SVOLGANO I SEGUENTI PUNTI

Punto 1
Si disegni il diagramma degli stati del sistema. Il diagramma deve indicare anche gli stati d'errore.

Punto 2
Si dimensioni il sistema tenendo conto che deve gestire fino a 100 utenti, che la password è 8 caratteri e il nome utente è al massimo 16 caratteri. Si riporti la descrizione dello spazio di indirizzamento (indirizzi associati alla memoria principale, indirizzi riservati di sistema, indirizzi di I/O, ...).

Punto 3
Si scriva un applicativo assembly 8088 che gestisca il riconoscimento degli utenti e l'apertura della porta.

Punto 4
Fatto il punto 3, si permetta all'utente, dopo l'identificazione di aprire la porta o di cambiare la password.

Punto 5
Fatto il punto 3, si gestisca l'esistenza di un amministratore del sistema. L'amministratore, dopo esser stato riconosciuto ha accesso al menù per la gestione degli utenti: inserimento di un nuovo utente, visualizzazione di quelli esistenti e cancellazione di uno di questi. La navigazione nel menù è numerica, cioè ad ogni voce è assegnato un numero. Il sistema è fatto per gestire molti utenti e le dimensioni del display sono limitate: si deve poter scorrere la lista degli utenti. Gli utenti sono visualizzati in ordine di inserimento nel database.

Punto 6
Fatto il punto 5, dia la possibilità all'amministratore di visualizzare gli utenti in ordine di inserimento nel database o in ordine alfabetico.

Punto 7
Fatto il punto 5 si dia la possibilità all'amministratore di cercare un'utente e, trovato, poterlo cancellare.

Punto 8
Fatti tutti i precedenti punti si suggerisca un possibile meccanismo di sicurezza per non conservare le password degli utenti in chiaro. Si suggerisca una possibile implementazione e si commenti criticamente le potenzialità offerte (o mancanti) dall'assembly a disposizione.

I punti 6-7-8 sono facoltativi. In particolare il punto 8, se fatto, deve esser parte della relazione con accenni di codice. I punti 1-2-3-4 oppure 1-2-3-5 sono necessari per valutare positivamente l'elaborato.

NOTE PER LA CONSEGNA

La valutazione del progetto dipende strettamente dai seguenti vincoli:

- Il progetto deve essere consegnato entro le 24.00 della data indicata nella prima pagina.
- **Presenza di commenti.** Il codice assembly deve essere commentato in modo da agevolare la comprensione. Ogni funzione o macro deve esser preceduta da opportuni commenti che ne chiariscono la funzionalità e gli argomenti. **I progetti privi di commenti NON saranno valutati.**
- **Relazione**, di max. 4 pagine per facilitare la lettura del codice. **Riportare nella relazione le scelte progettuali.**
- **Codice strutturato.** I codice deve essere opportunamente strutturato facendo uso di chiamate a funzioni e di macro.
- **Ottimizzazione.** Il codice disordinato è difficile da verificare (debugging). Saranno valutate positivamente scelte progettuali che ottimizzino il codice. L'ottimizzazione deve riguardare anche i commenti: chiari e coincisi.
- **Lavoro di gruppo.** Il progetto deve esser svolto da gruppi di lavoro, composti da almeno due persone e al massimo tre, in maniera collaborativa da tutti i componenti del gruppo. **Ogni componente del gruppo deve conoscere le scelte progettuali e i dettagli implementativi.**
- **Confronto tra gruppi.** E' utile il confronto tra gruppi. Tuttavia, dovendo ogni gruppo fare le sue scelte progettuali, è fondamentale evitare scelte di massa. Saranno valutati negativamente progetti simili tra loro o con simili scelte progettuali.
- **Funzionamento.** Il progetto deve essere assemblabile e funzionante. Indicare nella relazione quali sono i Punti di questo testo svolti.

Modalità di consegna

- Preparare una cartella chiamata **"cognome1_cognome2_cognome3"** (dove **cognome1, cognome2, ...** sono i cognomi dei componenti il gruppo di lavoro; evitare spazi nei nomi) contenente le seguenti cartelle/file:
  1. cartella **"progetto"**: contiene i file SORGENTI del progetto
  2. file **"relazione.pdf"**: relazione PDF
  3. file **"sorgente.pdf"**: file PDF contenente i SORGENTI del progetto
  4. file .pdf col testo del progetto (questo file!)
- Comprimere la cartella **"cognome1_cognome2_cognome3"** in **"cognome1_cognome2_cognome3.zip"**
- Spedire il file .zip allegato ad un mail ai seguenti indirizzi (ad entrambi e con un solo mail) entro il termine di consegna:
  1. **biafelice@arces.unibo.it**
  2. **difelice@cs.unibo.it**
- Indicare come **soggetto** del mail: "**[ARCHI]cognome1_cognome2_cognome3"**
- Riportare eventuali commenti nella **relazione**, non nel testo del mail.
- Una mail che conferma il ricevimento verrà generata automaticamente nelle 24ore successive la consegna.

Problemi riscontrati durante il precedente appello per la consegna

- Mail inviati separatamente ai due indirizzi: genera confusione rendendo difficile capire chi ha consegnato.
- Mail (forse) spedite ma non arrivate. Alcuni account di posta possono avere malfunzionamenti o limiti nella dimensione della posta in uscita. Si consiglia di mettere in copia alla mail di consegna l'indirizzo di un componente del gruppo per verificarne l'invio.
- Mail spedite senza l'allegato.

Errori di questo tipo possono rendere il progetto non valido vanificando il lavoro fatto.