AE 2020-21 Prova di ammissione all'orale – 2o appello, 3 Feb 2021

Si scrivano due funzioni in ASMv7:

- una `int convdigit(char c)`, che converte un carattere che rappresenta una cifra esadecimale (quindi un carattere in [0-9] o [a-f]) nel suo corrispondente valore. La rappresentazione ASCII dei caratteri in questione è riportata in tabella. Per le cifre "alfabetiche" si deve assumere di avere solo lettere minuscole. Il carattere passato come argomento è garantito essere nel range corretto;

Carattere	0	1	2	3	4	5	6	7	8	9		a	b	c	d	e	f
Codice ASCII	48	49	50	51	52	53	54	55	56	57		97	98	99	100	101	102

- una `int convstring(char * s)`, che converte una stringa di caratteri che rappresenta un numero in esadecimale in un intero (stesse assunzioni di prima, sui singoli caratteri, la stringa è una sequenza di caratteri terminata dal NULL (codice ASCII = 0)). La funzione fa uso della `convdigit` per convertire i singoli caratteri. L'algoritmo da seguire per la conversione è quello classico.

Si richiede che le due funzioni:

- rispettino tutte le convenzioni ARMv7 per parametri e utilizzo di registri;
- utilizzino, se possibile, i soli registri temporanei (r0-r3 ed eventualmente r8);
- siano ottimizzate per l'esecuzione sul processore single cycle.

Al termine si valuti quanti cicli di clock richiede la conversione di una stringa di *c* caratteri su un processore single cycle.

Quando si è terminato, le due funzioni e il numero dei cicli di clock devono essere copiati (in modo SOLO TESTO) nei box relativi nel compito google classroom.

Nella pagina che segue c'è un programma C che potete utilizzare per testare la correttezza delle vostre routine.

Per la conversione di una stringa si può utilizzare un codice che inizializza il risultato parziale a 0 e poi, una cifra alla volta, partendo dalla più significativa, finchè ce ne sono:

- moltiplica il risultato parziale per 16,
- calcola il valore della cifra che stiamo considerando (con la `convdigit`),
- somma il valore della cifra al parziale.

Per provare le due funzioni, si può utilizzare il codice C che segue:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

extern int convstring(char *);

int check(char * s) {
  int i, n, ret;
  ret = (1==1);
  n = strlen(s);
  for(i=0; i<n; i++)
    if(!(s[i]>='0' && s[i]<='9') && !(s[i]>='a' && s[i]<='f'))
      ret = (1==0);
  return ret;
}

int main(int argc, char ** argv) {

  int i;
  if(argc == 1) {
    char * x[4] = {"00d","ff", "ff00", "1000"};
    for(int i=0; i<4; i++)
    printf("Stringa %s converte in %d\n",
   x[i], convstring(x[i]));
  } else {
    for(int i=1; i<argc; i++)
      if(check(argv[i]))
        printf("Stringa %s converte in %d\n",
       argv[i], convstring(argv[i]));
      else
        printf("La stringa %s non è stringa esadecimale valida\n", argv[i]);
  }
  return(0);
}
```

Compilando questo programma (main.c) con il vostro file dove sono definite le funzioni (fun.s) si ottiene un eseguibile che prende quanti parametri volete (>1) che siano stringhe esadecimali e ne stampa le conversioni eseguite con la routine assembler. Se non vengono passati parametri, si stampano le conversioni delle stringhe inserite nell'array x.