

Esercitazioni di Reti Logiche

Raffaele Zippo

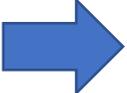
Corso di Laurea in Ingegneria Informatica, Università' di Pisa

a.a. 2020/21

Chi svolge il corso

- Prof. Ing. Giovanni Stea

- <http://www.iet.unipi.it/g.stea/>, giovanni.stea@unipi.it
- Ricevimento: lunedì 10.30-12.30 (orario valido per la durata del semestre)
- Necessario prenotarsi via email (evitare attese, impegni concomitanti)

 Ing. Raffaele Zippo (esercitazioni)

- raffaele.zippo@phd.unipi.it
- Ricevimento: lunedì 16.00-18.00 (orario valido per la durata del semestre)
- Necessario prenotarsi via email (evitare attese, impegni concomitanti)

Programma delle esercitazioni

- **Linguaggio Assembler**

- Presentazione ambiente di sviluppo
- Scrivere e debuggare programmi semplici
- Esercizi d'esame

- **Linguaggio Verilog**

- Presentazione ambiente di sviluppo
- Descrivere reti semplici
- Simulazione e verifica tramite testbench
- Esercizi d'esame

Ambiente di sviluppo

- Assemblatore GAS
- Debugger GDB
- Sistema operativo MS-DOS, emulato in DOSBox



Tutto incluso nel pacchetto
amb_GAS sul sito del corso

Editor: a propria scelta.

Io userò Visual Studio Code, con estensione “x86 and x86_64 Assembly”

Ambiente di sviluppo – *amb_GAS*

- Ambiente preconfigurato, per Windows
 - Su altro, dovrete configurarlo da voi

Installazione

1. Scaricare *amb_GAS.exe*
2. Eseguire ed estrarre in C:\amb_GAS (selezionato di default)
3. Eseguire C:\amb_GAS\INSTALLA.bat
 - Metterà in C:\WORK tutto il necessario
4. Eseguire C:\WORK\runDosBox.bat per avviare l'ambiente

Come assemblare

Il file sorgente (*.s) è un normale file di testo

Un editor adatto aiuta per

- Numeri di riga, utili per il debugging
- Evidenziare la sintassi

```
lezioni > 1 > ASM conta_bit.s
 1  #conteggio dei bit a 1 in un long
 2
 3  .GLOBAL _main
 4
 5  .DATA
 6  dato:      .LONG 0x0F0F0101
 7  conteggio: .BYTE 0x00
 8
 9  .TEXT
10  _main:     NOP
11  |         |     MOVB $0x00, %CL
12  |         |     MOVL dato, %EAX
13  |         |
14  comp:     CMPL $0x00, %EAX
15  |         |     JE fine
16  |         |     SHRL %EAX
17  |         |     ADCB $0x00, %CL
18  |         |     JMP comp
19  |         |
20  fine:     MOVB %CL, conteggio
21  |         |     RET
22
```

Come assemblare

Dalla finestra di DOSBox, eseguire *ASSEMBLE.BAT* con il percorso del file *.s*

```
C:\WORK>ASSEMBLE.BAT LEZIONI\1\CONTA_~1.S  
Press any key to continue.  
C:\WORK>_
```

Genera 2 file:

- L'eseguibile assemblato *.exe* nella stessa cartella del *.s*
- *C:\WORK\LISTATO.TXT* contenente l'output dell'assemblatore

Esecuzione

L'eseguibile assemblato si può lanciare (da DOSBox) come si farebbe con un programma compilato dal C

- Poco utile se non si usano input/output da terminale

```
C:\WORK>LEZIONI\1\CONTA_~1.EXE  
C:\WORK>_
```

Debugging

Per lanciare GDB, eseguire *DEBUG.BAT* con il percorso del file .exe

Utile per

- Vedere i risultati di programmi senza output a terminale
- Trovare gli errori nel codice, eseguendo passo passo
- Manuale (troppo) completo: <https://www.gnu.org/software/gdb/documentation/>

```
C:\WORK>DEBUG.BAT LEZIONI\1\CONTA_~1.EXE
GNU gdb 6.1.1
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "--host=i386-pc-msdosdjgpp --target=djgpp"...
Breakpoint 1 at 0x1e60: file LEZIONI\1\CONTA_~1.S, line 10.

Breakpoint 1, main () at LEZIONI\1\CONTA_~1.S:10
10      _main:      NOP
Current language:  auto; currently asm
(gdb)
```