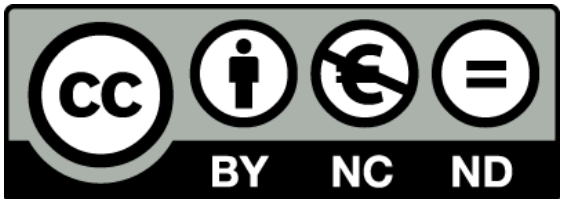
Laboratorio di Programmazione di Sistema

Fondamenti dei Linguaggi Assembly 2

Luca Forlizzi, Ph.D.

Versione 23.1



Stringhe binarie e bit

- In un *ASM-PM* i dati sono organizzati e manipolati in modo molto simile a come avviene nella corrispondente *ISA*, e vi sono anche similitudini con quanto accade ai livelli della Logica Digitale e della Microarchitettura
- Una *cifra binaria* è un elemento dell'insieme numerico $\{0, 1\}$
- Ogni dato è formato da una sequenza di cifre binarie, chiamato *stringa binaria*
- Anche se tutti i dati vengono memorizzati in un *ASM-PM* come sequenze di cifre binarie, i linguaggi *ASM* permettono di specificarli anche in modi più comodi per un programmatore umano

Stringhe binarie e bit

- La abstract machine di un *ASM-PM* memorizza dati in *dispositivi di memorizzazione* in essa contenuti
- I dispositivi di memorizzazione più semplici, chiamati *bit* sono in grado di memorizzare una cifra binaria
- I bit sono raggruppati in dispositivi più complessi
  - i *registri*, che contengono di solito da qualche decina a qualche centinaio di bit
  - la *memoria*, che contiene da migliaia a miliardi di bit

Parole

- Ogni dato (ovvero ogni stringa binaria) che un programma utilizza durante l'esecuzione deve essere necessariamente memorizzato in un insieme di bit
- Per ovvi motivi di efficienza, è preferibile che la abstract machine possa leggere o scrivere un dato mediante una singola operazione
- Ovvero che possa accedere mediante una singola operazione a tutti i bit dell'insieme che memorizza il dato

Parole

- Le abstract machine possono accedere mediante una singola operazione solo ad alcuni insiemi di bit, chiamati *parole*
- I bit a disposizione di una abstract machine sono quindi organizzati in parole, proprio per consentire l'accesso ad insiemi di bit mediante una singola operazione
  - Ciascun bit appartiene ad almeno una parola
  - Un bit può appartenere a diverse parole
- In *ASM*, le parole, usate singolarmente o in gruppi, hanno un ruolo simile a quello delle variabili degli *HLL*

Parole

- Data una parola P
  - La quantità dei bit che formano P viene detta *lunghezza* di P e indicata come LEN_P
  - A ciascun bit $b \in P$ è assegnato un diverso numero intero compreso tra 0 e $\mathrm{LEN}_P - 1$, detto *posizione* di b in P
- La stringa binaria formata dalle cifre binarie che sono memorizzate nei bit di una parola, disposte nell'ordine determinato dalle posizioni dei bit che le contengono, è il *contenuto* di tale parola
- Un linguaggio *ASM* consente di leggere e scrivere il contenuto delle parole
- La scrittura di un dato in una parola cancella il dato precedentemente contenuto nella parola stessa

Parole

- Per la maggior parte delle parole, è possibile sia leggere che scrivere il contenuto, ma vi sono eccezioni
  - Delle parole dette *read-only*, si può leggere il contenuto ma non scriverlo
  - Delle parole dette *write-only*, si può scrivere il contenuto ma non leggerlo
- Tentare di scrivere in una parola read-only o di leggere da una parola write-only, in alcuni *ASM-PM* non produce alcun effetto, mentre in altri può produrre malfunzionamenti o comportamenti non definiti

Parole

- Una parola è formata o da bit che si trovano tutti in registri, oppure da bit che si trovano tutti in memoria
- Quindi si distinguono 2 categorie di parole
  - *parole di registro*: formate da bit che appartengono ad uno o più registri della abstract machine
  - *parole di memoria*: formate da bit della memoria
- Le parole definite da un *ASM-PM*, spesso differiscono tra loro non solo per la categoria (di registro o di memoria) ma anche per altre caratteristiche, ad esempio per il numero di bit che esse contengono
- Un *ASM-PM* dunque definisce insiemi di parole che condividono le stesse caratteristiche, chiamati *formati di dato*

Formati di Dato

- Come abbiamo detto, si definice *parola* un insieme di bit a cui la abstract machine è in grado di accedere simultaneamente mediante una singola operazione
- Un *formato di dato* è una parte di un *ASM-PM* che definisce la struttura e le caratteristiche di un insieme di parole
- Il numero di diversi formati di dato di un *ASM-PM* varia, tipicamente, da poche unità fino a una dozzina
- Si noti che in questa lezione descriviamo i formati di dato in relazione agli *ASM-PM*, ma tipicamente un computer presenta gli stessi formati di dato ai livelli 2, 3 e 4

Formati di Dato

- Per brevità, se p è una delle parole dell'insieme definito dal formato **F**, diremo che p *appartiene* ad **F**, o che **F** *contiene* p
- In dettaglio, un formato di dato **F** stabilisce
  - La *lunghezza* $\mathrm{LEN}_{\mathbf{F}}$ delle parole contenute nel formato, ovvero il numero di bit che formano ciascuna parola che appartiene al formato
  - Gli *identificativi* delle parole contenute nel formato, ovvero delle stringhe alfanumeriche che consentono di identificare le parole contenute nel formato
  - Per ogni parola $p \in \mathbf{F}$, a ciascun bit $b \in p$ è assegnato un diverso numero intero compreso tra 0 e $\mathrm{LEN}_{\mathbf{F}} - 1$, detto *posizione* di b in p

Formati di Dato

- Un formato di dato non definisce la semantica delle parole, ovvero non attribuisce un significato ai bit che formano le parole
- La semantica delle parole viene definita dall'*interpretazione di dato*, ovvero un algoritmo che, a partire dal contenuto di una parola, calcola un valore, appartenente ad un determinato dominio, detto *valore* della parola
- Negli *ASM-PM*, l'interpretazione di dato è attribuita, ad ogni diverso accesso ad una parola, dall'istruzione che effettua l'accesso, come spiegheremo meglio in seguito

Formati di Dato

- Un formato **F** può contenere sia parole di registro che parole di memoria
- È possibile che alcuni aspetti dell'organizzazione dei bit all'interno delle parole, siano diversi per parole di registro e per parole di memoria, anche se esse appartengono allo stesso formato
- Ma parole di registro e parole di memoria che appartengono allo stesso formato, hanno sempre la stessa lunghezza

Formati di Dato Generali e Speciali

- È utile classificare i formati di dato rispetto alla quantità di istruzioni che li usano
  - Chiamiamo *generali* i formati di dato che vengono usati da molte istruzioni *ASM*
  - Al contrario, chiamiamo *speciali* i formati di dato utilizzati da poche istruzioni
- Nella maggior parte degli *ASM-PM*
  - I formati di dato generali contengono sia parole di registro che parole di memoria
  - I formati di dato speciali possono contenere solo parole di registro, o solo parole di memoria, oppure parole di entrambe le categorie

Identificativi

- Come vedremo in seguito, ogni istruzione *ASM* indica un'operazione da effettuare e una o più parole da usare per leggere i dati necessari e per scrivere i dati prodotti
- Una parola viene indicata da un'istruzione specificando
  - un formato di dato
  - una stringa chiamata *identificativo*
- Se una parola p appartiene al formato **F**, allora un identificativo di p consente di distinguere univocamente p tra tutte le parole di **F**
- Ovvero ogni parola q di **F** diversa da p, ha identificativo diverso dall'identificativo di p

Identificativi

- Si noti che
  - Due parole p e q appartenenti a due formati diversi, possono avere lo stesso identificativo
  - Due diversi identificativi possono identificare, in un formato **F**, la stessa parola (in questo caso si dice che i due identificativi sono *alias* l'uno dell'altro)
- Per i formati di dato generali e per alcuni formati di dato speciali
  - Gli identificativi per parole di registro sono stringhe alfanumeriche, chiamate *nomi*
  - Gli identificativi per parole di memoria sono stringhe binarie, chiamate *indirizzi di memoria*

Registri

- In questa presentazione, discutiamo in dettaglio l'organizzazione e le modalità di accesso ai registri, mentre lo studio della memoria è oggetto di future presentazioni
- I registri sono dispositivi di memorizzazione costituiti da un numero relativamente piccolo di bit, che di solito è pari a qualche decina e in rari casi arriva a poche centinaia
- In un tipico computer, sono i dispositivi di memorizzazione più veloci
- I registri hanno modalità di accesso specifiche e diverse da quelle della memoria
- La quantità totale di bit presente in tutti i registri è molto minore della quantità di bit contenuta dalla memoria

Registri

- In relazione al tipo di informazione che memorizzano, i registri si classificano in

Dati	Memorizzano dati da usare nelle operazioni
Indirizzi	Memorizzano indirizzi di memoria
Dati/Indirizzi	Memorizzano dati o indirizzi di memoria
Stato	Memorizzano informazioni sullo stato del programma in esecuzione e/o sul funzionamento del dispositivo

Registri

- In relazione ai loro utilizzi si classificano in

  Specific purpose Possono essere usati per pochi scopi, spesso per uno solo; di conseguenza sono usati da poche istruzioni

  General purpose Possono essere usati per scopi differenti, e quindi da molte istruzioni diverse

  General purpose with special functions Sono *general purpose*, ma hanno alcune particolarità semantiche, in relazione a specifiche istruzioni
- I registri di stato sono molto spesso *specific purpose*
- I registri Dati, Indirizzi e Dati/Indirizzi possono essere *specific purpose* o *general purpose*
- Registri *general purpose* sono più flessibili, ma possono essere più lenti e rendere più complesso il dispositivo

Parole di Registro

- In ogni *ASM-PM*, l'intero contenuto di un registro è una parola
- Ovvero, per ciascun registro esiste un formato di dato che definisce come parola di registro l'intero contenuto del registro
- Può accadere che in un *ASM-PM* ci siano registri di lunghezze diverse: in tal caso, per ciascuna possibile lunghezza esiste un corrispondente formato di dato
- As esempio MC68000 ha tutti i registri costituiti da 32 bit, tranne uno che è invece costituito da 16 bit: di conseguenza MC68000 ha un formato di dato per parole di lunghezza 32 e uno per parole di lunghezza 16

Parole di Registro

- In aggiunta alle parole di registro che coincidono con l'intero contenuto di un registro, ce ne possono essere altre che coincidono con alcune parti di un registro
- In molti *ASM-PM*, detto **F** un formato di lunghezza N che definisce una parola coincidente con un intero registro R, esistono dei formati $\mathbf{F_1}, \mathbf{F_2}, \ldots$ tali che
  - hanno lunghezze (rispettivamente) $\frac{N}{2}, \frac{N}{4}, \ldots$
  - contengono parole formate da bit di R che hanno posizioni consecutive in R, e che vengono disposti nello stesso ordine in cui si trovano in R

Parole di Registro

- Ad esempio, un *ASM-PM* che ha registri di 32 bit, potrebbe avere
  - Un formato **F** di lunghezza 32 con parole che coincidono con i registri
  - Un formato **F'** di lunghezza 16 che, per ciascuna parola di registro $R \in \mathbf{F}$, contiene
    - una parola R_1 formata dai bit di R che hanno posizione compresa tra 0 e 15, disposti in modo che la posizione di ciascun bit in R_1 è uguale alla posizione dello stesso bit in R
    - una parola R_2 formata dai bit di R che hanno posizione compresa tra 16 e 31, disposti in modo che la posizione di ciascun bit in R_2 è pari alla posizione dello stesso bit in R diminuita di 16
  - Quindi, ad esempio, il bit che ha posizione 4 in R appartiene anche a R_1 con posizione 4, mentre il bit che ha posizione 22 in R appartiene anche a R_2 con posizione 6

Registri e Formati di Dato Generali in MC68000

- Ogni implementazione di MC68000 ha i seguenti registri
  - 8 registri *general purpose* di 32 bit ciascuno, detti *registri dati*
    - possono contenere dati interi
    - sono chiamati `d0`,`d1`,...,`d7`
  - 8 registri *general purpose* di 32 bit ciascuno, detti *registri indirizzi*
    - possono contenere sia dati interi, sia indirizzi di memoria
    - sono chiamati `a0`,`a1`,...,`a7`
    - il nome `sp` è un alias per `a7`
    - `a7` ha una funzioni speciale che verrà discussa in seguito
  - i registri *specific purpose* `pc` (di 32 bit) e `sr` (di 16 bit)
- Come opzione, può avere 8 registri *general purpose* di 80 bit ciascuno per dati floating point

Registri e Formati di Dato Generali in MC68000

- MC68000 ha 3 formati di dato generali
  - `long` di lunghezza 32
  - `word` di lunghezza 16
  - `byte` di lunghezza 8
- Ciascuno di questi 3 formati definisce sia parole di registro, i cui identificativi sono i nomi dei registri, che parole di memoria, i cui identificativi sono indirizzi di memoria
- Caratteristiche del formato `long`
  - Le parole contenute in questo formato sono chiamate `long` e hanno lunghezza 32
  - Le `long` di registro sono gli insiemi di tutti e 32 i bit dei registri dati e dei registri indirizzi

Registri e Formati di Dato Generali in MC68000

- Caratteristiche del formato `word`
  - Le parole contenute in questo formato sono chiamate `word` e hanno lunghezza 16
  - Le `word` di registro sono
    - l'insieme di tutti e 16 i bit di `sr`
    - gli insiemi di bit dei registri dati e indirizzi che nel formato `long` hanno posizione compresa tra 0 e 15
  - Se un bit b è contenuto sia in una `word` di registro che in una `long` di registro, allora il formato `word` e il formato `long` assegnano a b la stessa posizione

Registri e Formati di Dato Generali in MC68000

- Caratteristiche del formato `byte`
  - Le parole contenute in questo formato sono chiamate `byte` e hanno lunghezza 8
  - I `byte` di registro sono
    - l'insieme dei bit di `sr` che nel formato `word` hanno posizione compresa tra 0 e 7
    - gli insiemi di bit dei registri dati che nel formato `word` hanno posizione compresa tra 0 e 7
  - Il formato `byte` non è valido per i registri indirizzi
  - Se un bit b è contenuto sia in un `byte` di registro che in una `word` di registro, allora il formato `byte` e il formato `word` assegnano a b la stessa posizione

Registri e Formati di Dato Generali in MIPS32

- MIPS32 definisce i seguenti registri
  - 32 registri *general purpose* di 32 bit ciascuno, detti *GPR*
    - possono contenere indirizzi di memoria o dati interi
    - i nomi sono i numeri da 0 a 31 e inoltre per tutti vi sono alias formati da stringhe di 2 o 3 caratteri alfanumerici
    - il registro 0 ha una funzione speciale: è read-only ed il suo contenuto è la stringa formata da 32 cifre uguali a 0
    - il registro 1 ha una funzione speciale che verrà discussa in seguito: per ora non verrà usato
    - il registro 31 ha una funzione speciale che verrà discussa in seguito
  - 2 registri *specific purpose* di 32 bit ciascuno, chiamati `LO` e `HI`, usati per operazioni di moltiplicazione o divisione
  - 32 registri *general purpose* di 32 bit ciascuno per dati floating point
  - il registro *specific purpose* `PC`
  - vari registri di Stato non interessanti in LPS

Registri e Formati di Dato Generali in MIPS32

- MIPS32 ha 2 formati di dato generali
  - `word` di lunghezza 32
  - `half` di lunghezza 16
- Anche MIPS32 ha un formato di lunghezza 8 chiamato `byte`, ma esso non è un formato generale perché è usato solo da poche istruzioni e pertanto verrà discusso in seguito

Registri e Formati di Dato Generali in MIPS32

- Caratteristiche del formato `word`
  - Il formato `word` definisce sia parole di registro che di memoria
  - Le parole contenute in questo formato sono chiamate `word` e hanno lunghezza 32
  - Le `word` di registro sono gli insiemi di tutti e 32 i bit dei registri
  - Gli identificativi delle `word` di registro sono i nomi dei registri
  - Gli identificativi delle `word` di memoria sono indirizzi di memoria, come verrà illustrato in seguito
- Caratteristiche del formato `half`
  - Il formato `half` definisce solo parole di memoria
  - Le parole contenute in questo formato sono chiamate `half` e hanno lunghezza 16
  - Gli identificativi delle `half` di memoria sono indirizzi di memoria, come verrà illustrato in seguito

Formato e Interpretazione di Dato

- Come detto, un formato di dato non stabilisce l'interpretazione di dato; essa viene invece stabilita da un'istruzione che accede ad un dato
- Nella maggior parte degli *ASM-PM*, ogni istruzione, per ciascun dato
  - Usa un'unica interpretazione di dato
  - Può usare diversi formati di dato
- Un formato di dato può essere associato (da istruzioni diverse) a diverse interpretazioni di dato, ma non necessariamente a tutte quelle possibili nell'*ASM-PM*
- Un'interpretazione di dato viene associata ad uno specifico formato di dato, tuttavia a formati differenti possono essere associate interpretazioni simili

Formato e Interpretazione di Dato

- Le interpretazioni di dato consentono di utilizzare stringhe binarie per rappresentare dati di varia natura
- Le più comuni interpretazioni di dato permettono di rappresentare
  - Numeri interi con un numero fissato di cifre
  - Indirizzi di memoria
  - Numeri floating point
- Le interpretazioni di dato per interi e indirizzi di memoria verrano studiate in future lezioni di LPS, ma si assume sin d'ora che ogni *ASM-PM* abbia una interpretazione di dato che permette di rappresentare valori appartenenti ad un intervallo di interi che comprende sia numeri negativi che numeri positivi
- Le interpretazioni di dato per numeri floating point sono al di fuori dell'ambito di LPS

Dati: *HLL* vs. *ASM*

- Gli *HLL* usano le variabili per memorizzare i dati, mentre gli *ASM* usano le parole
- Poiché negli *ASM* le parole hanno un ruolo simile a quello delle variabili negli *HLL*, è naturale tracciare un parallelo tra parole e variabili e, in particolare, confrontare i seguenti concetti, che presentano analogie ma anche differenze
  - *identificativi* delle parole $\leftrightarrow$ *nomi* delle variabili
  - *formato di dato* delle parole $\leftrightarrow$ *tipo di dato* delle variabili

Dati: *HLL* vs. *ASM*

- Per un confronto in generale tra parole e variabili, è opportuno distinguere tra parole di registro e di memoria
- Le parole di registro si differenziano dalle variabili di un *HLL*, in quanto
  - Esiste una quantità fissata di registri e di parole di registro, ciascuno con uno o più identificativi predefiniti
  - Pertanto le parole di registro non devono essere "dichiarate" in un programma *ASM*: sono automaticamente pronte all'uso
  - Tuttavia le parole di registro non sono inizializzate automaticamente
- Le parole di memoria sono più simili alle variabili di un *HLL*
  - Non esiste una quantità fissata di parole di memoria
  - Esistono costrutti *ASM* analoghi alle dichiarazioni di variabile, che consentono di specificare l'identificativo ed il contenuto iniziale di una parola di memoria

Dati: *HLL* vs. *ASM*

- Negli *HLL*, ogni programma contiene numerosi *scope*, ovvero parti del codice sorgente in cui un nome di variabile è valido
  - Esempi di scope in Java: definizioni di classe e di metodo
  - All'interno di ciascuno scope, il nome di una variabile la identifica univocamente, ovvero è diverso dai nomi di tutte le altre variabili, qualunque tipo esse abbiano
  - Un nome può essere utilizzato in scope diversi, con significati diversi
- **identificativo** $\leftrightarrow$ **nome di variabile**: prima differenza
  - Negli *HLL*, un nome è valido solo in alcuni scope, e può essere riferito a variabili diverse, in scope diversi
  - Negli *ASM*, l'identificativo di una parola è valido in tutto il programma ed è riferito sempre alla stessa parola

Dati: *HLL* vs. *ASM*

- **identificativo** $\leftrightarrow$ **nome di variabile**: seconda differenza
  - Negli *HLL*, all'interno di uno scope, un nome è diverso da quello di tutte le altre variabili, qualunque tipo esse abbiano
  - Negli *ASM*, l'identificativo di una parola è diverso da quello di tutte le altre parole dello stesso formato, ma può essere uguale a quello di parole che hanno formato diverso
- Negli *ASM*, per identificare in maniera univoca una parola, bisogna specificarne sia il formato, sia l'identificativo

Dati: *HLL* vs. *ASM*

- Negli *HLL*, il tipo di dato di una variabile (o di una costante) ne specifica
  - L'aspetto sintattico, ovvero la quantità di bit usati per formare la variabile e la loro organizzazione
  - L'aspetto semantico, ovvero l'algoritmo che, a partire dalle cifre binarie contenute nella variabile, permette di calcolare un valore, appartenente ad un determinato dominio
- **Formato di dato $\leftrightarrow$ tipo di dato**: prima differenza
  - Negli *HLL*, il concetto di tipo di dato integra gli aspetti sintattici e semantici del dato
  - Negli *ASM*, il formato di dato (che definisce gli aspetti sintattici) e l'interpretazione di dato (aspetti semantici) sono concetti distinti, benché ovviamente collegati

Dati: *HLL* vs. *ASM*

- **Formato di dato** $\leftrightarrow$ **tipo di dato**: seconda differenza
  - Negli *HLL* il tipo di dato è una proprietà di un dato, che determina quali operazioni sono valide o meno su di esso
  - Negli *ASM* formato ed interpretazione di un dato dipendono in parte dalla parola che lo memorizza e in parte dalle operazioni che vengono effettuate
- In particolare, negli *HLL* avviene il *type checking* che determina quali operazioni sono consentite su un dato
- Invece negli *ASM* non vi è *type checking* a limitare quali sono i dati a cui può essere applicata una determinata operazione