Il Livello Microarchitettura (iii)

A cura di:
Luca Breveglieri Giuseppe Pozzi

DEI, PoliMI, Milano

luca.breveglieri,giuseppe.pozzi@polimi.it

- versione dell'11 aprile 2003 -

Il modello di memoria e i sottoprogrammi del processore IJVM

Strutturazione della memoria

- La memoria centrale è strutturata, ovvero è suddivisa in aree (non necessariamente contigue) aventi usi diversi
- Un'area può contenere, a seconda dell'uso assegnatole: codice eseguibile, costanti (cioè dati fissi), dati (cioè variabili), riferimenti ad altre aree (cioè puntatori), o altri tipi di informazioni ancora

La memoria centrale di IJVM

- Il processore IJVM dispone di uno spazio di memoria centrale di 4 Gbyte, cioè 4 G $\times$ 8 bit (parole da 8 bit)
- È anche possibile vedere la memoria come un vettore di 1 G $\times$ 32 bit, ovvero strutturata con parole da 32 bit

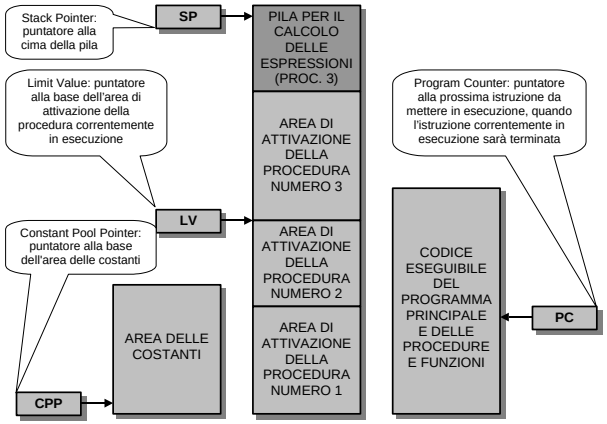
Memoria e istruzioni

- Le istruzioni macchina del processore IJVM non vedono direttamente gli indirizzi della memoria centrale
- Esse operano implicitamente sulla memoria, utilizzando degli appositi registri come puntatori (PC, SP, LV, CPP e registri altri ancora)
- Per esempio: le operazioni logico-matematiche vengono calcolate tramite la pila delle espressioni (registro SP)

Modello di memoria di IJVM

- La memoria centrale del processore IJVM è suddivisa in 4 aree principali (non necessariamente tutte contigue):
  - area delle costanti (Constant Pool Area)
  - pila delle aree di attivazione (Frame Stack Area)
  - pila per il calcolo delle espressioni (Operand Stack Area)
  - area del codice eseguibile (Executable Code Area)

Modello di memoria di IJVM



Area delle costanti

- **Area delle costanti**: contiene costanti numeriche e di altro tipo, stringhe fisse di caratteri e puntatori ad altre aree di memoria
- L'area viene inizializzata quando il programma viene caricato in memoria
- Il programma non può modificare il contenuto dell'area
- Il programma accede all'area tramite il registro puntatore CPP

Area di attivazione

- **Area di attivazione del programma principale, della procedura e della funzione**: contiene alcuni puntatori di servizio (vedere lezioni successive), usati nella gestione del meccanismo di chiamata-ritorno di sottoprogramma, contiene i parametri passati alla procedura (o alla funzione) all'atto della sua chiamata e contiene lo spazio riservato per le variabili locali della procedura

Area di attivazione

- L'area di attivazione viene allocata sulla cima della pila quando la procedura associata entra in esecuzione
- La procedura correntemente in esecuzione usa le variabili locali e i parametri contenuti nella propria area di attivazione
- Quando la procedura correntemente in esecuzione termina, l'area di attivazione in cima alla pila viene eliminata

Area di attivazione

- Sopra l'area di attivazione del programma principale, procedura o funzione correntemente in esecuzione (si noti che sopra quest'area di attivazione c'è sempre spazio libero per fare crescere ulteriormente la pila), viene costruita la pila per il calcolo delle espressioni, nei momenti in cui il programma deve calcolare un'espressione logico-matematica

Area di attivazione

- Il registro LV, contenuto nel processore IJVM, viene usato per puntare alla base dell'area di attivazione della procedura correntemente in esecuzione
- Il registro SP, contenuto nel processore IJVM, punta alla cima della pila, ovvero:
  - alla cima dell'area di attivazione corrente
  - oppure alla cima della pila delle espressioni, nei momenti in cui questa non è vuota, cioè quando è in corso il calcolo di un'espressione

Pila delle espressioni

- **La pila per il calcolo delle espressioni**: è una zona di spazio libero, di dimensioni prefissate, riservata per fare crescere la pila per il calcolo delle espressioni logico-matematiche
- Quando il compilatore genera le sequenze di istruzioni che calcolano le espressioni usando la pila, si assicura che la pila non possa crescere in modo eccessivo rispetto allo spazio di memoria lasciato libero

Area del codice eseguibile

- **Codice eseguibile del programma principale e delle procedure e funzioni**: contiene il codice eseguibile del programma principale e delle eventuali procedure e/o funzioni facenti parte del programma complessivo da eseguire; almeno il programma principale è sempre presente, mentre procedure e/o funzioni sono facoltative

Area del codice eseguibile

- L'area del codice eseguibile viene caricata in memoria centrale quando l'intero programma entra in esecuzione
- Quando il programma principale esegue l'istruzione di terminazione, in linea di principio il contenuto dell'area di codice eseguibile diventa inutile e potrebbe essere cancellato

Area del codice eseguibile

- Il registro PC, contenuto nel processore IJVM, viene usato per puntare, all'interno dell'area di codice eseguibile, alla prossima istruzione macchina da mettere in esecuzione, quando l'istruzione macchina correntemente in esecuzione sarà terminata (e supponendo che l'istruzione macchina corrente non sia un'istruzione di salto)

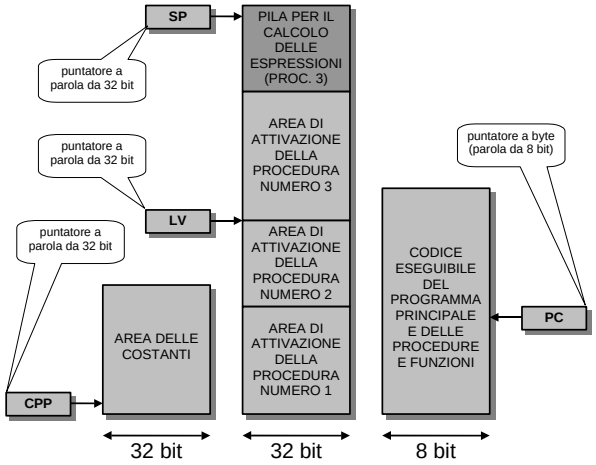
Precisazioni sui registri

- I registri CPP, LV e SP, contenuti nel processore IJVM, sono puntatori a parole da 32 bit, non a byte, cioè:
  - leggere alla locazione puntata p. es. da SP significa leggere la parola da 32 bit (non il byte) situato in cima alla pila
  - leggere alla locazione puntata p. es. da LV + 1 significa leggere la parola da 32 bit (non il byte) situata subito dopo la parola da 32 bit puntata direttamente da LV

Precisazioni sui registri

- Il registro PC, contenuto nel processore IJVM, è un puntatore a byte, non a parole da 32 bit, cioè:
  - leggere alla locazione puntata da PC significa leggere un byte (non una parola da 32 bit) facente parte dell'istruzione da mettere in esecuzione
  - leggere alla locazione puntata da PC + 1 significa leggere il byte (non la parola da 32 bit) successivo all'ultimo byte letto, facente parte dell'istruzione da mettere in esecuzione

Parole di memoria e registri



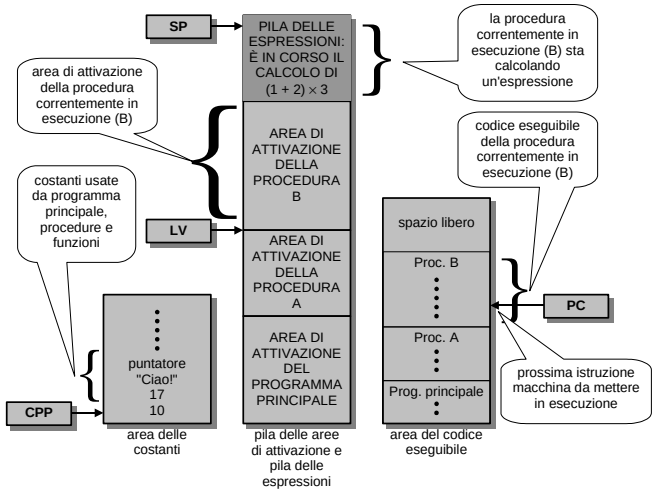
Esempio di allocazione memoria

- Si considera una programma Java (ma varrebbe anche per un programma C) costituito da:
  - programma principale (main)
  - funzione A, chiamata dal prog. principale
  - funzione B, chiamata dalla funzione A
- Il prog. principale è in realtà anch'esso una funzione, che viene chiamata per prima da parte del sistema operativo

Esempio di allocazione memoria

- La funzione A esegue alcune elaborazioni, poi chiama la funzione B
- La funzione B esegue il calcolo di un'espressione, poi termina
- Il prog. principale e le funzioni A e B fanno uso di alcune costanti
- La figura illustra lo stato delle varie aree di memoria della macchina IJVM, mentre la funzione B è in esecuzione e sta appunto calcolando l'espressione

Esempio di allocazione memoria



Osservazioni

- Il registro LV (Limit Value) punta alla base dell'area di attivazione della funzione B, correntemente in esecuzione
- Il registro SP (Stack Pointer) punta alla cima della pila delle espressioni, che contiene dei valori perché è in corso, da parte di B, il calcolo di un'espressione
- Il registro PC (Program Counter) punta nell'area di codice eseguibile di B, alle istruzioni che calcolano l'espressione

Sottoprogrammi

- Il linguaggio macchina IJVM ammette come sottoprogrammi solo funzioni (non procedure), similmente al linguaggio C
- Una funzione IJVM restituisce sempre un numero intero come valore di uscita
- Se la funzione va usata come procedura, il valore restituito può essere ignorato (benché sia presente)

Passaggio a sottoprogramma

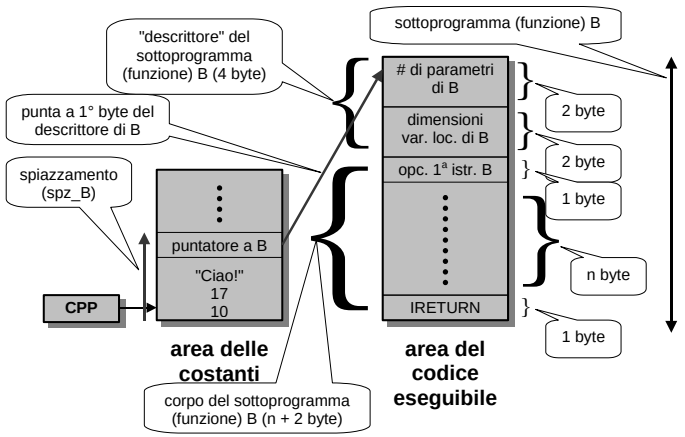
- Il processore IJVM dispone di due istruzioni macchina per la gestione del meccanismo di passaggio a sottoprogramma (funzione):
  - INVOKEVIRTUAL spiazzamento: il (sotto)programma chiamante attiva il sottoprogramma specificato tramite l'argomento spiazzamento
  - IRETURN: il sottoprogramma chiamato ritorna al (sotto)programma chiamante, restituendogli un numero intero

Specifica del chiamato

- Per specificare la collocazione e le caratteristiche del sottoprogramma da chiamare, l'istruzione INVOKEVIRTUAL adotta un formato di specifica apposito
- Tale formato si basa su un "descrittore" del sottoprogramma da chiamare, contenente informazioni necessarie per una sua corretta messa in esecuzione

Esempio

INVOKEVIRTUAL spz_B



Funzionamento della chiamata

- Il sottoprogramma chiamante scrive sulla cima della pila il riferimento all'oggetto OBJREF (pleonastico) e i parametri da passare al sottoprogramma chiamato
- Il sottoprogramma chiamante esegue l'istruzione:
  - INVOKEVIRTUAL spiazzamento

  dove spiazzamento indica (vedere la struttura descrittore) il sottoprogramma chiamato cui passare l'esecuzione

Funzionamento del rientro

- Il sottoprogramma chiamato mette sulla cima della pila il valore da restituire al sottoprogramma chiamante (un intero)
- Il sottoprogramma chiamato esegue l'istruzione:
  - IRETURN

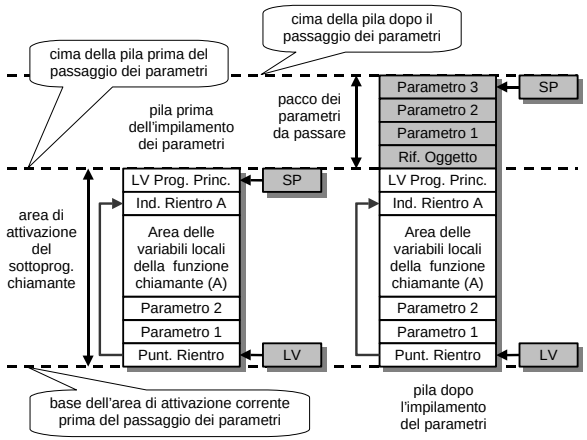
  che riporta l'esecuzione al sottoprogramma chiamante, all'istruzione successiva a INVOKEVIRTUAL

L'istruzione INVOKEVIRTUAL

- L'istruzione INVOKEVIRTUAL predispone l'ambiente per il passaggio dell'esecuzione al sottoprogramma:
  - impila l'area di attivazione del chiamato
  - inserisce nell'area di attivazione informazioni necessarie in seguito per gestire il rientro da sottoprogramma
  - aggiorna i registri LV, SP e PC del processore IJVM
  - lancia l'esecuzione del sottoprogramma

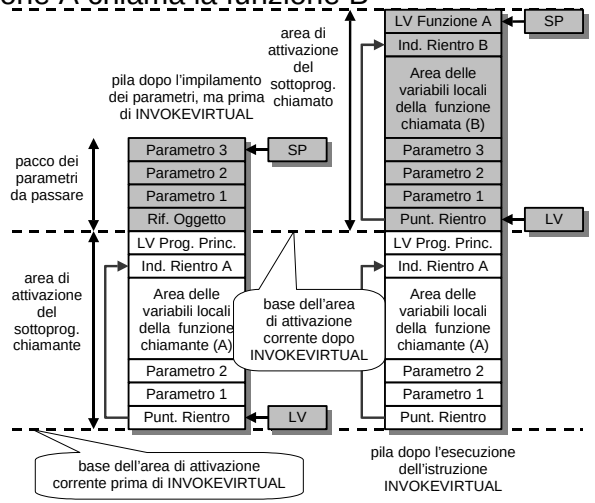
Chiamata a sottoprogramma (I)

la funzione A impila i parametri da passare alla funzione B



Chiamata a sottoprogramma (II)

la funzione A chiama la funzione B

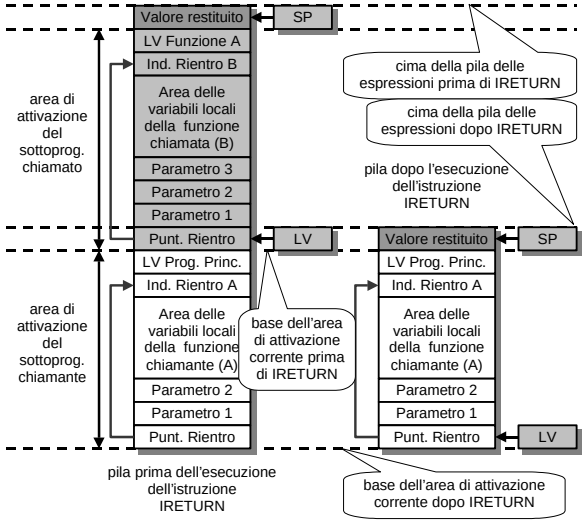


L'istruzione IRETURN

- L'istruzione IRETURN ricostruisce l'ambiente di esecuzione del sottoprogramma chiamante:
  - spila l'area di attivazione del chiamato
  - lascia sulla cima della pila del chiamante il valore restituito dal chiamato
  - ripristina i registri LV, SP e PC del processore IJVM ai valori precedenti la chiamata
  - riporta l'esecuzione al chiamante

Rientro da sottoprogramma

la funzione B termina e la funzione A riprende l'esecuzione; A riceve il valore restituito da B



Come funziona IRETURN

- L'istruzione IRETURN inverte le operazioni dell'istruzione INVOKEVIRTUAL, ovvero:
  - ripristina i valori originali dei registri LV e PC, usando il "puntatore_rientro", lo "indirizzo_di_rientro" e il "LV Funzione" memorizzati nell'area di attivazione del sottoprogramma chiamato
  - aggiorna il valore del registro SP e lascia il valore restituito sulla cima dell'area di attivazione del sottoprogramma chiamante

Esempio

- Si supponga di avere un programma IJVM modularizzato come segue:
  - programma principale
  - funzione A (restituisce un numero intero)
  - funzione B (restituisce un numero intero)
- Il programma principale chiama la funzione A, che a sua volta chiama la funzione B
- Viene mostrato il processo di passaggio (chiamata-rientro) da A a B

Aree di attivazione

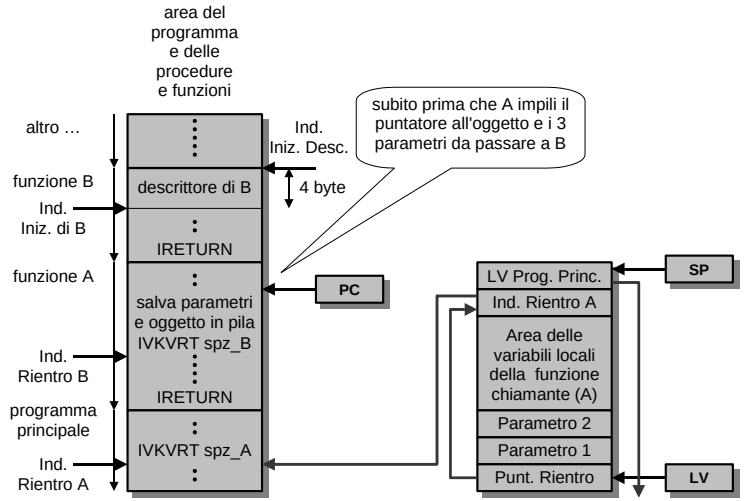
- La pila del processore IJVM contiene dunque l'area di attivazione del programma principale, e sopra questa quella della funzione A
- Il processo di chiamata installa l'area di attivazione della funzione B sopra l'area di A (e così disattiva l'area di A)
- Il processo di rientro disinstalla l'area di attivazione della funzione B, riattivando l'area di A

Simulazione della chiamata

Animazione

Fine

la funzione A chiama la funzione B

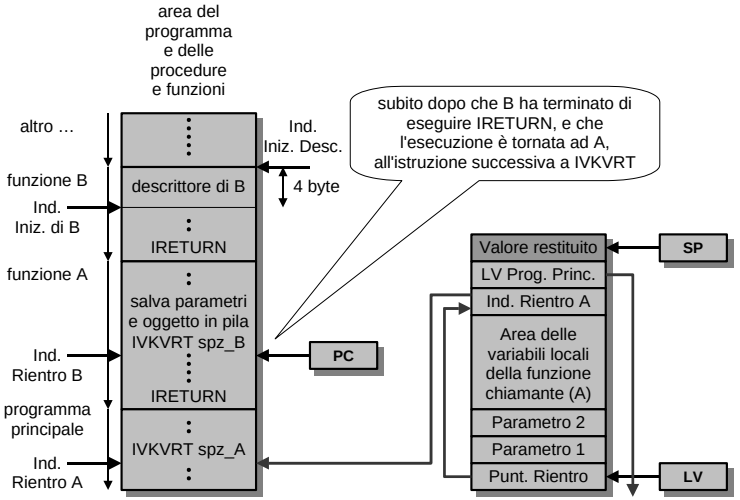


Simulazione del ritorno

Animazione

Fine

B termina e A riprende l'esecuzione



Osservazione

- Il processo di chiamata di una funzione viene visto dalla funzione chiamante come un calcolo di espressione:
  - alla chiamata la pila per il calcolo delle espressioni del chiamante è vuota
  - al rientro la pila per il calcolo delle espressioni del chiamante contiene solo il valore restituito dal chiamato (come se fosse stata calcolata un'espressione)
- Si possono pertanto calcolare facilmente espressioni contenenti funzioni