# Errata-Corrige del libro di testo

## Indice generale

All'inizio della seconda pagina dell'indice: il capitolo "Livello dei Processi" è il Cap. VI.

## Cap. IV

**Pag. 133**: secondo microprogramma: la seconda frase della microistruzione 0 è

**( = 1 0 0 ) reset RDY0, set ACK0, IN → B, IND → IND1, 2;**

La microistruzione 2 è:

2. **($ACK_1$ = 0 ), nop, 2;**

   **( = 1) B + A[IND1] → A[IND1], B + A[IND1] → OUT1,**
   **set $RDY_1$, reset $ACK1_1$, 0;**

**Pag. 147**: terza frase del primo microprogramma: la condizione logica è ( = 111).

## Cap. V

**Pag. 186**: aggiungere quanto segue alla fine della sez. 2.2:

Al file eseguibile è associato un *file di configurazione* contenente informazioni convenienti o necessarie al caricatore. Tra esse:

- quali blocchi di informazioni, sia istruzioni che dati, conviene caricare inizialmente in memoria principale in quanto aventi la più alta probabilità di essere riferite in una fase iniziale dell'esecuzione del processo ("insieme di lavoro", argomento che sarà trattato nel Cap. VIII);
- in quali posizioni sono contenute informazioni utili al caricatore o al gestore della memoria, come PCB e Tabella di Rilocazione;
- quali strutture dati sono condivise con altri processi, in modo da inizializzare correttamente la Tabella di Rilocazione nel caso che sino già state caricate in memoria principale. A tale scopo, ad ogni struttura condivisa deve essere associato un identificatore unico;
- quali informazioni devono essere caricate in memorie locali di specifiche unità di I/O, invece che nella memoria principale vera e propria (sistemi con Memory Mapped I/O, Cap. IX).

**Pag. 201**: sostituire la definizione dell'istruzione START_PROCESS con la seguente:

- *START_PROCESS RIC, Rtabril, Rcaptabril*

Invio alla *MMU* delle informazioni necessarie a riferire la Tabella di Rilocazione del processo che entra in fase di esecuzione, e, contemporaneamente, ripristino del contenuto di *IC* all'indirizzo logico della prima istruzione da cui il processo deve riprendere (iniziare) l'esecuzione. Ciò permette di effettuare, in maniera atomica, tutte le azioni necessarie ad iniziare l'esecuzione di un processo una volta che i registri generali siano stati ripristinati. Il registro di indirizzo RIC deve contenere il valore dell'immagine del contatore istruzioni letta dal PCB.

Il registro di indirizzo Rtabril contiene l'indirizzo logico della Tabella di Rilocazione del processo che entra in esecuzione, e Rcaptabril contiene l'entrata della Tabella Rilocazione di tale processo mediante la quale effettuare la traduzione degli indirizzi della Tabella di Rilocazione stessa. In tal modo, MMU può tradurre gli indirizzi della Tabella di Rilocazione allo scopo di accedere alla Tabella stessa.

Alternativamente, se l'architettura permette di lavorare anche su indirizzi fisici, Rtabril può contenere l'indirizzo *fisico* della Tabella di Rilocazione del processo che entra in esecuzione. In tal caso, il campo Rcaptabril non è significativo. La traduzione da indirizzo logico a indirizzo fisico è effettuata *a programma* prima di eseguire la START_PROCESS.

## Cap. VI

**Pag. 254**, nel codice di BUFFER_MANAGER il comando *while* va scritto come

**while not** end **do**

## Cap. VII

**Pag. 297**, terza riga: togliere la frase finale "L'architettura … fig. 9:".

## Cap. VIII

**Pag. 332**: righe + 4 e -8: sostituire "indirizzo logico" con "indirizzo di memoria principale".

**Pag. 332**: aggiungere alla sez. 3.3.3 quanto segue:

Se realizzata sul chip della CPU, non è più necessario che la cache secondaria sia interallacciata. Possiamo assumere che la cache secondaria abbia tempo di accesso uguale a *due cicli di clock* per parola, sia per tenere conto della maggiore capacità rispetto alla cache primaria, che per tenere conto di operazioni di traduzione dell'indirizzo. Sovrapponendo la lettura di una parola da cache secondaria alla scrittura della parola precedente nella cache primaria, il tempo di trasferimento del blocco vale soltanto:

$$T_{trasf} \sim 2\,\sigma\,\tau$$

Vale la pena di notare che, a parità di capacità complessiva $\gamma$, mantenere la distinzione tra cache primaria e secondaria on-chip è conveniente rispetto ad avere soltanto un'unica cache primaria di capacità $\gamma$. Infatti:

- la cache primaria contiene solo informazioni del processo in esecuzione, mentre la cache secondaria può mantenere informazioni di più processi, e quindi essere pronta a trasferire blocchi in/da cache primaria in caso di commutazione di contesto;

- i blocchi della cache secondaria hanno dimensione adatta al trasferimento con la memoria principale (o cache terziaria),
- la cache secondaria può trasferire blocchi dalla memoria principale (o cache terziaria) in parallelo all'esecuzione del processo e all'uso della cache primaria, ad esempio anticipando blocchi del processo in esecuzione o di altri processi.

## Cap. X

**Pag. 413**, il modulo M va scritto come segue:

… operazione di *k-unfolding*:

```
M ::    int A[m]; input stream: int s0; output stream: int s;
        { receive s0 from input_stream;;
          s = s0;
          for (i = 0; i < m/k; i++)                    }
                s = F (s, A[i]);                       }
          for (i = m/k; i < 2m/k; i++)                 }
                s = F (s, A[i]);                       }  k volte
          …                                            }
          for (i = (k-1)*m/k; i < m; i++)              }
                s = F (s, A[i]);                       }
          send s onto output_stream
        }
```

## Note integrative su parallelismo a livello firmware

Par. 1.2.1 Riga 5: $T_{2\text{-}id} = T_{F2} = 70\ \tau$ (invece di $80\ \tau$)

Par. 1.2.2: la quarta formula è per $T_{A2}$ (invece di $T_{A1}$).

Pagina 10, Riga 9 dal basso: tempo di servizio ideale (invece di tempo di servizio effettivo ideale).

Par. 1.2.4: nella figura va modificato $F_{11}$ (nodo a destra) con $F_{12}$.