Esercizio Cache

Andrea Bartolini – a.bartolini@unibo.it

FP Example: Array Multiplication

- C = C + A × B
  - All 32 × 32 matrices, 64-bit double-precision elements
  - DGEMM (Double precision GEneral Matrix Multiply)
- C code:

```
void mm (double c[][],
         double a[][], double b[][]) {
  size_t i, j, k;
  for (i = 0; i < 32; i = i + 1)
    for (j = 0; j < 32; j = j + 1)
      for (k = 0; k < 32; k = k + 1)
        c[i][j] = c[i][j]
                + a[i][k] * b[k][j];
}
```

  - Addresses of `c`, `a`, `b` in `x10`, `x11`, `x12`, and `i`, `j`, `k` in `x5`, `x6`, `x7`

FP Example: Array Multiplication

- RISC-V code:

```
    mm:...
        li    x28,32          // x28 = 32 (row size/loop end)
        li    x5,0            // i = 0; initialize 1st for loop
L1:     li    x6,0            // j = 0; initialize 2nd for loop
L2:     li    x7,0            // k = 0; initialize 3rd for loop
        slli  x30,x5,5        // x30 = i * 2**5 (size of row of c)
        add   x30,x30,x6      // x30 = i * size(row) + j
        slli  x30,x30,3       // x30 = byte offset of [i][j]
        add   x30,x10,x30     // x30 = byte address of c[i][j]
        fld   f0,0(x30)       // f0 = c[i][j]
L3:     slli  x29,x7,5        // x29 = k * 2**5 (size of row of b)
        add   x29,x29,x6      // x29 = k * size(row) + j
        slli  x29,x29,3       // x29 = byte offset of [k][j]
        add   x29,x12,x29     // x29 = byte address of b[k][j]
        fld   f1,0(x29)       // f1 = b[k][j]
```

FP Example: Array Multiplication

…

```
slli    x29,x5,5      // x29 = i * 2**5 (size of row of a)
add     x29,x29,x7    // x29 = i * size(row) + k
slli    x29,x29,3     // x29 = byte offset of [i][k]
add     x29,x11,x29   // x29 = byte address of a[i][k]
fld     f2,0(x29)     // f2 = a[i][k]
fmul.d  f1, f2, f1    // f1 = a[i][k] * b[k][j]
fadd.d  f0, f0, f1    // f0 = c[i][j] + a[i][k] * b[k][j]
addi    x7,x7,1       // k = k + 1
bltu    x7,x28,L3     // if (k < 32) go to L3
fsd     f0,0(x30)     // c[i][j] = f0
addi    x6,x6,1       // j = j + 1
bltu    x6,x28,L2     // if (j < 32) go to L2
addi    x5,x5,1       // i = i + 1
bltu    x5,x28,L1     // if (i < 32) go to L1
```

- La cpu dispone di una cache dati a 2 vie da 16KiB complessivi e linee da 64 byte, gestita con stato MESI e con politica di scrittura Write-Around in caso di miss.
- Si considerino A,B,C immagazzinate in memoria a partire rispettivamente dagli indirizzi: 0x0100 8000, 0x0100 A000, 0x0100 C000
- 1) Si disegni la mappa della memoria
- 2) Si analizzi la dinamica della cache dati, e, tenendo ben presente che il sistema ha un solo caching agent, si risponda in modo preciso, schematico, conciso e tabellare ai seguenti quesiti:
  - Quali sono gli indici di set e linea, e i tag associati ad A,B,C per il primo e l'ultimo blocco contenenti le matrici?
  - Quante linee di cache occuperanno nel loro insieme? Possono i due vettori e la variabile stare per intero e simultaneamente in cache?
- 3) Si consideri la dinamica della cache nel calcolo della prima iterazione i=0, j=0, k=0 nel calcolo del primo elmento di C, disegnando lo stato MESI, il contenuto della cache e il valore del bit LRU dopo ogni operazione elementare (Load e Store) e si indichi il numero di accessi, il numero di miss e il numero di cicli di writeback e gli eventuali cicli di write allocate.
- 4) Si indichi il numero di accessi, il numero di miss e il numero di cicli di writeback e gli eventuali cicli di write allocate, nonché lo stato MESI della cache al termine del calcolo del primo elemento di C[0][0]. (si riporti il contenuto del primo ed ultimo set della cache)