Domande Orali Calcolatori Elettronici 1° Appello 2022 – 12/01/2022

1) Perché non serve invocare la 'invalida_entrata_TLB(vaddr);' – dopo aver eseguito una unmap – all'interno della funzione che distrugge il processo? (perché tanto il processo verrà distrutto e non eseguirà più l'accesso a quegli indirizzi)
2) Traduzione in Assembler di:

```
1
2   class cc{
3       long a [8];
4
5       cc (const cc&){...}
6
7       cc f (int i){
8           cc tmp ;
9           tmp.a[1] = tmp.a[3] = i;
10          return tmp;
11      }
12  }
13
```

   Analisi delle varie possibili ottimizzazioni
3) Problemi di interazione tra BUS Master e Cache: consistenza dei dati presenti in memoria per via del fatto che il BUS Master può effettuare trasferimenti che coinvolgono la memoria RAM senza dover coinvolgere il processore (e cioè la cache). Tipologie di problemi e vari possibili soluzioni: sia hardware che software.
4) Traduzione in Assembler di:

```
class cc{
    long a[8]

    cc f(int i){
        cc tmp;
        tmp.a[1]=tmp.a[3]=i;
        return tmp;
    }

}
```

5) BUS Mastering: problematiche anche relative alla paginazione e memoria virtuale
6) Trasferimenti tra periferiche: il buffer passato dal processo deve essere per forza presente nella parte condivisa della memoria? Perché è necessario che lo sia nel caso in cui sia un processo esterno ad occuparsi del trasferimento dei dati da/verso la periferica.
7) Traduzione in Assembler di:

```
class cc{
    char c;
    int j;
    char c1;

    cc f(cc mioc){
        c = mioc.c1;
        j = mioc.j++;
        return mioc;
    }

}
```

8) BUS Mastering: conclusione dell'operazione, quali sono i possibili problemi connessi con l'invio da parte della periferica di un segnale di terminazione del trasferimento. (Come si fa a capire che una operazione sia terminata: soluzione hardware e software).

9) Traduzione in Assembler di:

```
struct s{
    short s[4];
}

class cc{
    char c;
    s s1;

    void f(cc &mioc){
        for(int i = 0; i < 4; i++){
            s1.s[i] = mioc.c;
        }
    }
}
```

10) Cosa è l'allineamento: definizione.
11) Finestra di traduzione, traduzione identità: motivazione e vantaggi.
12) Traduzione in Assembler in:

```
class cc{
    long a[8];
    cc& f (long *p){
        for(int i=0; i<8; i++){
            a[i]+=p[i];
        }
        return *this
    }
}
```

13) Quali sono i vantaggi di avere il bus pci?
    a. Come si fanno a mettere dei dispositivi sul bus pci?
    b. Quanti spazi di indirizzamento ci sono nel bus pci? (3: memoria, I/O, configurazione)
14) Politica di rimpiazzamento delle cacheline: LRU Last recently used. Nella cache di indirizzamento a 4 vie, quando tutte le vie sono piene, ne va liberata una.
    Come funziona l'algoritmo che approssima LRU che abbiamo visto?