Examen computerarchitectuur

Vrijdag 8 juni 2007, 14u00

Prof. Koen De Bosschere

Naam, Voornaam: Richting:

Belangrijk

1. Vergeet niet uw naam en voornaam te vermelden.
2. Schrijf de antwoorden in de daarvoor voorziene ruimte. Bereid uw antwoord voor in het klad, en schrijf het naderhand over. De antwoorden zijn meestal kort.
3. Het examen duurt 3 uur.
4. Gelieve geen rode inkt te gebruiken.
5. Het examen is open boek.
6. U mag geen computer of rekenmachine gebruiken bij het oplossen van de vragen.

Veel succes!

Schrijf hier eventuele opmerkingen die van belang kunnen zijn bij de quotering (ziekte, topsport, gemaakte afspraken, enz.).

Vraag 1 (4 punten)

Gegeven de volgende getallen: a) 7 b) 0,5 c) -0,125 d) 0 e) -3.

Wat is de binaire voorstelling van deze 5 getallen? (negatieve getallen mogen met een min-teken geschreven worden).

a)

b)

c)

d)

e)

Ontwerp het kleinste vlottende-kommaformaat (analoog aan IEEE 754) dat in staat is om de vijf voorgaande bitpatronen exact te representeren.

Ontworpen formaat: | |

a)

b)

c)

d)

e)

Vraag 2 (4 punten)

Gegeven de volgende gegevensstructuur in C.

```
int b[31][62];
int g;
```

en de volgende for-lus:

```
g = 0;
for (i=0; i<31; i++)
  g += b[i][i*2];
```

Schrijf de adresexpressie neer die hoort bij de instructie uit de lus, en vereenvoudig deze zover mogelijk (veronderstel dat de waarde van i zich in ebx bevindt, de waarde van g zich in edx bevindt, dat de matrix b zich rij per rij in het geheugen bevindt, en dat een int 32 bit groot is).

Schrijf de assemblerinstructie(s) voor de IA32 neer die deze adresexpressie implementeren.

Vraag 3 (4 punten)

Beschouw het volgende C-programma

```
#include <stdio.h>
#include <malloc.h>

struct vector {
  int x;
  int y;
} a;

struct vector *add(struct vector a, struct vector b)
{
  struct vector *t;
  t = (struct vector*)malloc(sizeof(struct vector));
  t->x = a.x + b.x;
  t->y = a.y + b.y;
  return t;
}

int main()
{
  struct vector b, *z;
  a.x = 1;
  a.y = 2;
  b.x = 4;
  b.y = 5;
  z = add(a, b);
  printf("%d, %d\n", z->x, z->y);
}
```

Met als code die door de gcc-compiler gegenereerd wordt:

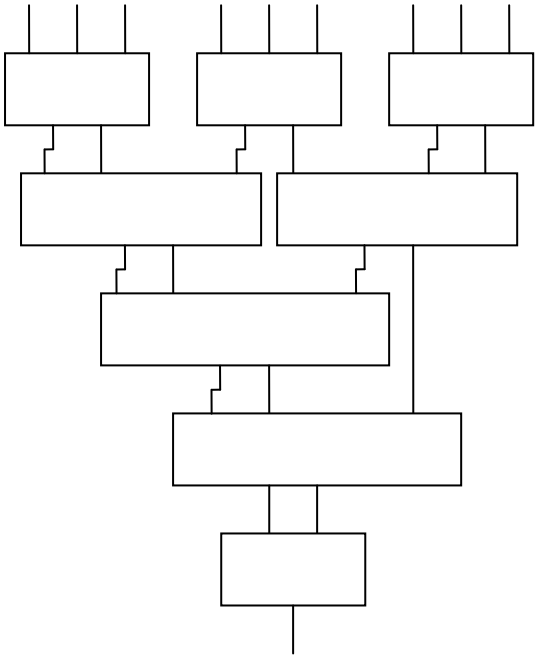
```
add:  pushl %ebp
      movl  %esp, %ebp
      pushl %edi
      pushl %esi
      pushl %ebx
      subl  $24, %esp
      movl  8(%ebp), %eax
      movl  12(%ebp), %edx
      movl  16(%ebp), %ecx
      movl  20(%ebp), %ebx
      movl  %edx, %edi
      movl  %eax, %esi
      movl  %ebx, -16(%ebp)
      movl  %ecx, %ebx
      pushl $8
      call  malloc
      addl  %ebx, %esi
      movl  %esi, (%eax)
      addl  -16(%ebp), %edi
      movl  %edi, 4(%eax)
      leal  -12(%ebp), %esp
      popl  %ebx
      popl  %esi
      popl  %edi
      leave
      ret
main:
      pushl %ebp
      movl  %esp, %ebp
      andl  $-16, %esp
      movl  $1, a
      movl  $2, a+4
      pushl $5
      pushl $4
      pushl a+4
      pushl a
      call  add
      addl  $16, %esp
      pushl 4(%eax)
      pushl (%eax)
      pushl $.LC0
      call  printf
      leave
      ret

.LC0:
      .string     "%d, %d\n"
```

Teken de stapel op het ogenblik dat de functie malloc opgeroepen wordt (ga uit van een initiële waarde van esp van 10c hex). Verklaar de diverse velden op de stapel.

Optimaliseer de functie add. Je mag ervan uitgaan dat malloc enkel het resultaatregister aanpast.

Vraag 4 (4 punten)

Gebaseerd op het principe van een carry save vermenigvuldiger kan men ook een carry save opteller ontwerpen. Een *carry save opteller* (CSO) bestaat uit een parallelschakeling van volledige optellers. Een dergelijke opteller reduceert drie bitpatronen van gelijke lengte tot 2 bitpatronen van dezelfde lengte: de sommen en de overdrachten. Door de overdrachten over 1 positie naar links te verschuiven en bijkomend op te tellen kan men een reeks van getallen optellen. De laatste opteller moet uiteraard een traditionele opteller zijn (met slechts 1 uitgang). Hieronder staat een opteller getekend die 9 getallen optelt.



Indien het aantal poortvertragingen voor een CSO ***a*** bedraagt, en voor een traditionele opteller ***b***, hoeveel poortvertragingen veroorzaakt bovenstaand circuit dan?

Indien men 9 getallen zou willen optellen met traditionele optellers, hoeveel poortvertragingen zou dit veroorzaken?

Onder welke voorwaarde op ***a*** en ***b*** zal het CSO-circuit sneller zijn dan de traditionele oplossing?

Indien het circuit 4 bit getallen zou optellen, en de traditionele opteller is een carry-look-ahead opteller, wat zal dan de uitvoeringstijd van de beide oplossingen zijn ? En met een doorsijpelopteller?

	Carry save adder	Traditioneel optelcircuit
Carry look-ahead		
Doorsijpelopteller		

Vraag 5 (4 punten)

Gegeven een computersysteem met de volgende kenmerken

1 Instructiecache met een missrate van 2%
1 Datacache met een missrate van 4%
Percentage geheugeninstructies in de instructiestroom: 30%
Latentie naar het geheugen: 100 cycli

Je mag ervan uitgaan dat de toegangstijd naar de caches 1 cyclus is, dat de processor per cyclus 1 instructie kan uitvoeren indien alle gegevens zich in de caches bevinden (analoog met het escape model). Je mag ook aannemen dat een instructiecachemisser en een datacachemisser nooit in dezelfde cyclus zullen optreden.

Indien er een budget beschikbaar is om de computer te versnellen, en je kan dit budget gebruiken om (i) ofwel een processor te kopen die draait aan een dubbele snelheid (inclusief de caches, maar niet het geheugen), (ii) ofwel het geheugen te vervangen door een geheugen dat dubbel zo snel is, wat is dan de beste optie? Bespreek kwantitatief.

CPI op oude machine

CPI op machine met snelle processor

CPI op machine met snel geheugen

Op welke manier kan het computersysteem best versneld worden, en hoe groot is de bereikte snelheidswinst (in procenten).