Examen computerarchitectuur

Woensdag 22 juni 2005, 8u30

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 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)

Doe de volgende omzettingen:

Bitpatroon	**Type**	**Decimale waarde**
1001 1010	8-bit 2-complement	
1000 0110	Floating point 1|3|4	
	12 bit 10-complement packed BCD	-89
	8 bit teken-grootte	78

Vraag 2 (4 punten)

Beschouw het volgende C-programma

```
int p(int n, int m)
{
  if (n == 0 && m == 0)
   return 1;
  else if (n >= m)
    return p(n / 2, m) + 1;
  else
    return p(n, m-1) + 1;
}

int g;

main()
{
  g = p(3,6);
}
```

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

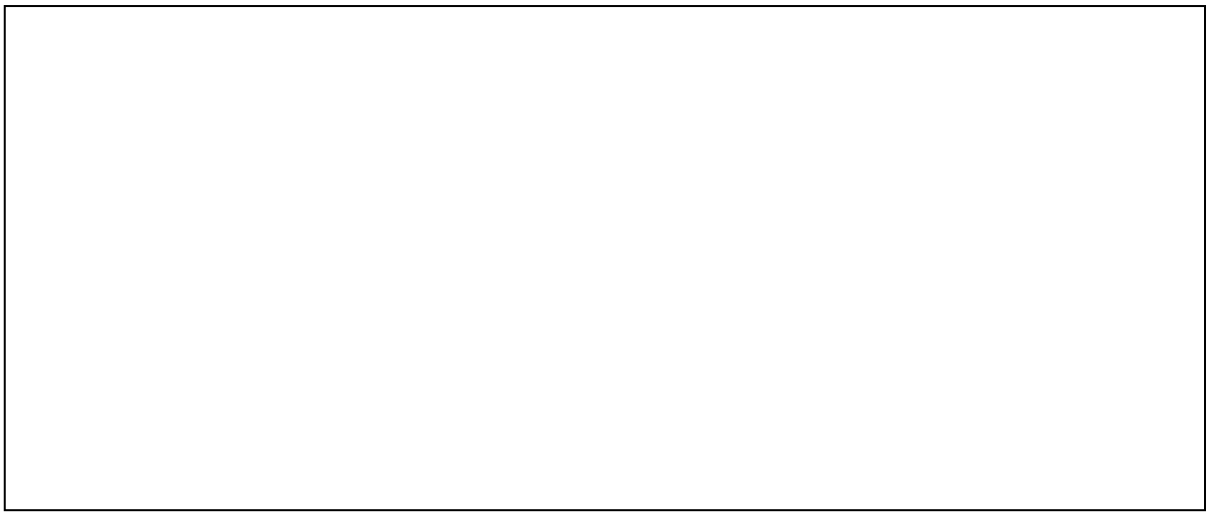
```
p:    pushl %ebp
      movl  %esp, %ebp
      movl  8(%ebp), %edx
      testl %edx, %edx
      movl  12(%ebp), %eax
      jne   .L3
      testl %eax, %eax
      jne   .L3
      movl  $1, %eax
      jmp   .L2
.L3:  cmpl  %eax, %edx
      jl    .L5
      subl  $8, %esp
      pushl %eax
      movl  %edx, %eax
      shrl  $31, %eax
      leal  (%eax,%edx), %eax
      sarl  $1, %eax
      pushl %eax
      call  p
      incl  %eax
      addl  $16, %esp
      jmp   .L2
.L5:  subl  $8, %esp
      decl  %eax
      pushl %eax
      pushl %edx
      call  p
      incl  %eax
      addl  $16, %esp
.L2:  leave
      ret
```

```
main:
      pushl %ebp
      movl  %esp, %ebp
      subl  $32, %esp
      pushl $5
      pushl $3
      call  p
      incl  %eax
      movl  %eax, g
      addl  $16, %esp
      leave
      ret
```

Teken de controleverloopgraaf van dit programma (2 punten)

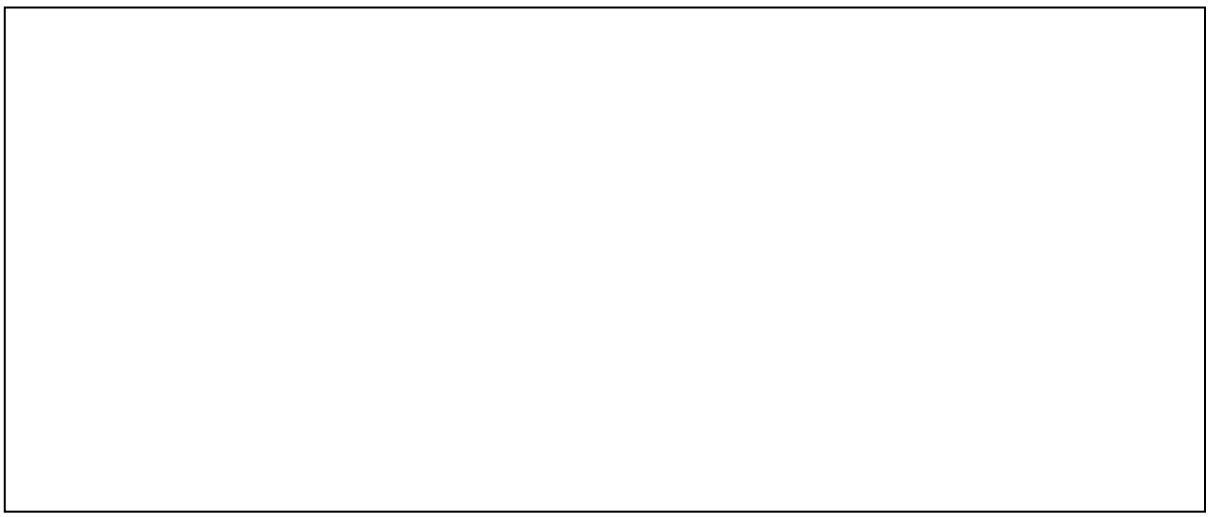
Leg de volgende codesequentie uit (1 punt)

```
movl  %edx, %eax
shrl  $31, %eax
leal  (%eax,%edx), %eax
sarl  $1, %eax
```



Leg de volgende codesequentie uit (1 punt)

```
pushl $5
pushl $3
call  p
incl  %eax
```

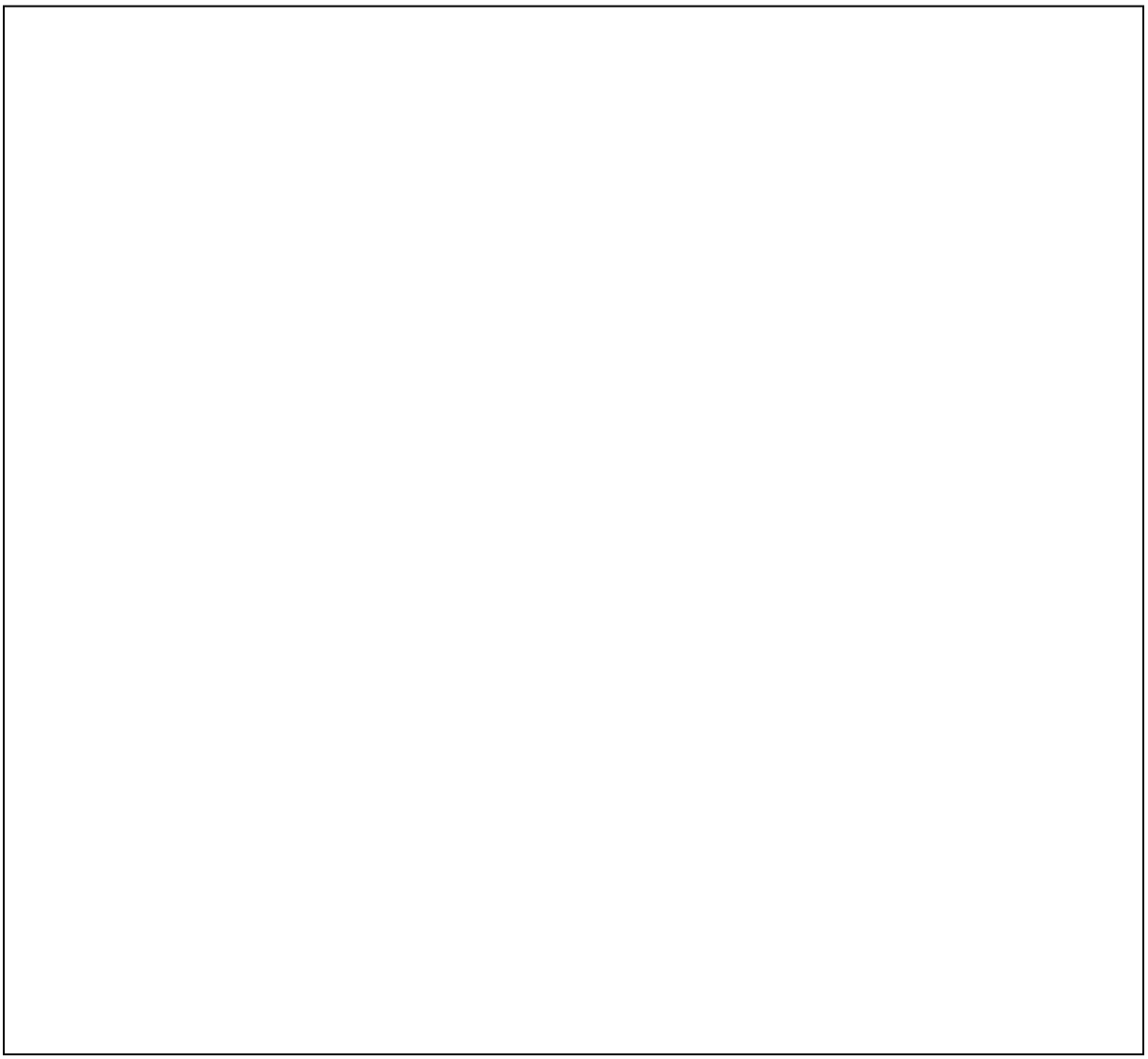


Vraag 3 (4 punten)

Optellers kunnen willekeurig breed gemaakt worden door ze in cascade te schakelen. Een 16-bit opteller gebaseerd op 4-bit ripple carry opteller zal 32 poortvertragingen nodig hebben om zijn resultaat te berekenen. De snelheid van het totale circuit wordt bepaald door de snelheid waarmee de overdracht tussen de individuele optellers kan doorgegeven worden.

Het doorgeven van deze overdracht tussen de verschillende 4-bit optellers kan echter versneld worden door de meest beduidende optellingen op voorhand (speculatief) uit te voeren (eenmaal met overdracht 0 en eenmaal met overdracht 1), en naderhand via een multiplexer te kiezen voor één van beide oplossingen. Dit wordt een carry select opteller genoemd.

Teken hieronder een dergelijke 16-bit opteller uitgaande van een 4-bit opteller, en bereken de totale poortvertraging als U ervan uitgaat dat de 4-bit opteller een ripple carry opteller is en de multiplexer 1 poortvertraging introduceert (b.v. door gebruik te maken van tri-state buffers).



Vraag 4 (4 punten)

Gegeven een lokale geschiedenisgebaseerde voorspeller met een geschiedenislengte van 3 bits en 2-bit saturerende tellers in de tabel. Na een opwarmperiode bevat de sprongtabel de volgende informatie.

000	00
001	01
010	11
011	11
100	00
101	00
110	10
111	11

Het programma voert vervolgens een sequentie van sprongen uit. A, B, C, D, E zijn de symbolische namen van de sprongen (je zou ze de adressen van de spronginstructies kunnen beschouwen, een + betekent dat de sprong in werkelijkheid genomen wordt, een – betekent dat de sprong niet genomen wordt). Vul nu de tabel verder aan: de geschiedenisbits per sprong (voor en na; de eerst keer dat een sprong voorkomt is de geschiedenis reeds ingevuld, de recentste sprongen staan rechts), geef aan wat de voorspelling zal zijn (met een + of een -) en geef tenslotte aan of de voorspelling juist was of fout. Vergeet niet om de saturerende tellers telkens aan te passen.

Sprong	Geschiedenis		Voorspelling +/-	Juist/Fout
	Voor	Na		
A+	000			
B-	101			
E+	100			
A+				
A+				
A-				
B+				
D+	111			
C-	101			
E+				
D+				
A-				
A+				
A-				

Vraag 5 (4 punten)

Gegeven het volgende programma:

```
for (i = 0; i < 4096; i++) {
   sum = 0;
   for (j = 0; j < 16; j++) {
      sum = sum + a[i][j] * h[j];
      /* load a[i][j] before h[j] in assembly */
   }
   b[i] = sum;
}
```

De arrays a[4096][16], b[4096], h[16] worden na elkaar in het geheugen gealloceerd (gealigneerd op het begin van een cacheblok) en alle andere veranderlijken worden in registers opgeslagen (elk array-element is 1 woord groot). De elementen van de array a worden als volgt in het geheugen opgeslagen a[0][0] a[0][1] …

Hoeveel leestoegangen en schrijftoegangen worden er door de uitvoering van dit programma gegenereerd.

leestoegangen	
schrijftoegangen	

Bereken het aantal L1 data cachemissers voor een (zeer kleine) direct mapped cache van 4 blokken van 4 woorden. De caches zijn initieel leeg en de schrijfstrategie is write through/write no allocate.

Bereken het aantal L1 data cachemissers voor een (zeer kleine) volledig associatieve cache (LRU) van 4 blokken van 4 woorden. De caches zijn initieel leeg en de schrijfstrategie is write through/write no allocate.