Tentamen Computerarchitectuur

Oefententamen, december 2016

Examinator: dr. K. F. D. Rietveld

- Het tentamen is **gesloten boek**, dus het is niet toegestaan om het tekstboek, slides of eigengemaakte aantekeningen te gebruiken.
- Alleen rekenmachines waarin geen teksten kunnen worden opgeslagen zijn toegestaan. Het gebruik van grafische rekenmachines is **niet** toegestaan.
- De vragen mogen worden beantwoord in het Nederlands en Engels.
- Beargumenteer al uw antwoorden. Aan antwoorden zonder uitleg of uitwerking worden geen punten toegekend.

- Bij het inleveren van het gemaakte werk zal u worden verzocht uw naam, studentnummer en aantal ingeleverde bladen te noteren op de presentielijst.

- Het aantal opgaven is 5 met een totaal van 20 onderdelen. Het tentamen bestaat uit 4 pagina's.
- Bij elk onderdeel staat tussen vierkante haken het aantal te behalen punten aangegeven. Het totaal aantal te behalen punten is 100.

Opgave I – Performance [15 punten]

We gaan uit van een RISC machine met een klokfrequentie van 2 GHz. Door middel van een representatieve benchmark zijn de operaties op deze machine gekarakteriseerd:

Operatie	CPI	Frequentie
Integer ALU	1	39%
FP ALU	8	13%
Loads	2	25%
Stores	2	11%
Branches	3	12 %

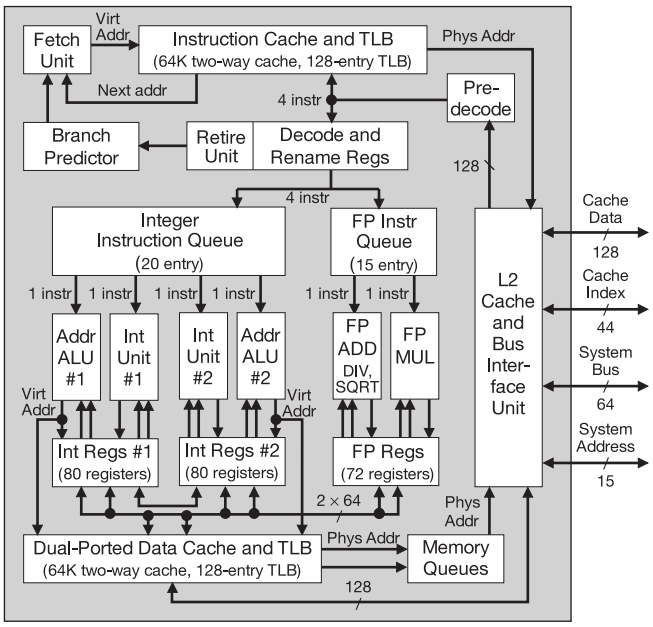
a. [5 punten] Wat is de overall CPI die wordt behaald?

b. [5 punten] Stel we kunnen de FP operaties met 40% versnellen. Bereken de nieuwe overall CPI, en de overall speedup die wordt behaald op de benchmark.

c. [5 punten] Een eenvoudige manier om de performance van een machine te verbeteren is door de klokfrequentie te verhogen. In de praktijk is echter gebleken dat de klokfrequentie niet eindeloos kan worden verhoogd. Leg uit waardoor dit wordt veroorzaakt.

Opgave II - Processor organisatie [15 punten]

In deze opgave bekijken we de Alpha 21264 processor:



Bron: L. Gwennap, Digital 21264 Sets New Standard.

a. [5 punten] Hoeveel instructies kunnen in deze processor maximaal tegelijkertijd in de "execute stage" zijn? En hoeveel nieuwe instructies kunnen per klokperiode worden opgestart ("issue")? Geef duidelijk aan waaruit dit blijkt.

b. [5 punten] Aan wat voor eigenschappen moet een programma voldoen om de piekcapaciteit van deze processor te kunnen halen? Geef een omschrijving van een programma waarmee de piekcapaciteit *juist niet* kan worden gehaald.

c. [5 punten] De data cache van deze processor, onderaan het figuur, is "dual-ported". Wat houdt dit in en wat zouden de consequenties zijn wanneer deze cache niet "dual-ported" was geweest?

Opgave III - Caching [25 punten]

De processor in de vorige opgave heeft een two-way (set associative) L1 data cache van 64KB. Een cache line is 64 bytes groot.

a. [5 punten] Leg het concept van een set-associative cache uit aan de hand van een vergelijking met een direct-mapped cache.

b. [5 punten] Bereken het aantal sets in deze cache en het aantal benodigde bits om de cache te kunnen indexeren.

We willen graag de performance van de L1 cache verbeteren door de miss rate terug te dringen. Dit zou kunnen worden gedaan door de grootte van de cache te verdubbelen naar 128KB of door de associativity te verhogen van 2 naar 4. Deze verbetering mag niet ten koste gaan van de kloksnelheid van de processor.

c. [5 punten] Welke van deze twee suggesties zou u kiezen om deze verbetering te realiseren en waarom?

Stel nu we hebben één cache niveau met een hit time van 1 klokperiode (een klokperiode is 0.4 ns) en een miss penalty van 60 ns. Uit een karakterisatie van de software volgt dat de CPI met een perfecte cache 2 is en er ongeveer 1.6 memory references per instructie worden plaatsvinden. Het blijkt mogelijk om de miss rate terug te dringen van 3.4% naar 1.5%.

d. [5 punten] Wat is de "average memory access time" in beide gevallen? Wat voor speedup (in CPU-tijd) wordt er behaald met deze reductie van de miss rate?

e. [5 punten] Volgens de documentatie van de chip van Opgave II is de cache "non-blocking" en is er een "miss queue" aanwezig met 8 slots. Leg in je eigen woorden uit wat dit inhoudt en welke voordelen dit heeft.

Opgave IV - Instruction-Level Parallelism [25 punten]

In deze opgave bekijken we een 5-stage micro-architectuur (Instruction Fetch, Decode and Register Fetch, Execute, Memory, Write back). Elke stage kost 1 klokperiode. Er is *geen* forwarding geïmplementeerd. Gegeven is de volgende code:

```
Loop:
    LW    R1, 0(R2)
    ADDI  R1, R1, #1
    SW    R1, 0(R2)
    LW    R3, 0(R5)
    ADDI  R2, R2, #4
    SUB   R4, R3, R2
    BNZ   R4, Loop
```

a. [6 punten] Laat voor één iteratie van de loop zien hoe de pipeline per kloktik wordt gevuld. Indien er een stall optreedt, benoem dan ook de reden waarom.

b. [3 punten] Hoeveel kloktikken gaan er per iteratie verloren aan loop-overhead?

Met behulp van Tomasulo's algoritme proberen we tot een efficiëntere pipeline te komen. We gaan uit van het volgende:

- Een single-issue Tomasulo pipeline: IF/ID/IS/EX/WB (IS=Issue), IF, ID IS en WB kosten 1 kloktik, EX is afhankelijk van de functional unit.
- 1 Load/Store unit (3 kloktikken in EX, 5 reservation station slots voor loads, 3 reservation station slots voor stores).
- 1 Integer unit (1 kloktik in EX, 4 reservation station slots).
- De functional units zijn niet gepipelined.
- Resultaten worden gecommuniceerd via een Common Data Bus (CDB), 1 resultaat per klokperiode.
- De effective address calculation voor load/stores wordt binnen de load/store functional unit gedaan tijdens de EX stage.
- Branch instructions worden uitgevoerd via de Integer unit.
- We gaan ervan uit dat een branch altijd wordt genomen.

c. [10 punten] Laat nu voor 3 loop iteraties van het bovenstaande programma zien hoe dit wordt uitgevoerd door de Tomasulo pipeline. Geef een tabel waarin het volgende is opgenomen:

- Iteratie (loop iteratie nummer)
- Instructie
- IS (in welke kloktik de instructie issued)
- EX (in welke kloktik de instructie start met execute)
- WB (in welke kloktik de instructie met write result start, dus het schrijven naar de CDB).

d. [2 punten] Wat is het gemiddelde aantal kloktikken per loop iteratie dat wordt behaald met de uitvoering onder **c.**?

e. [4 punten] We gingen er tot nu toe van uit dat de branch altijd wordt genomen. Stel nu dat later voor een branch blijkt dat deze *niet* had mogen worden genomen. Leg uit wat er in zo'n geval moet gebeuren.

Opgave V - Parallelism [20 punten]

a. [5 punten] Leg het verschil uit tussen SIMD en multi-core processing.

b. [5 punten] Ga uit van een single-threaded programma, waarvan 70% kan worden geparallelliseerd. We hebben een machine met 40 processor cores tot onze beschikking. Wat is de theoretisch maximum speedup die kan worden behaald door dit programma te parallelliseren?

Stel we hebben een GPU architectuur met 10 SIMD processoren (multiprocessors) en elke SIMD processor bestaat uit 32 lanes. Elke lane kan zowel single-precision arithmetic als load/store instructies uitvoeren. De processor heeft een klokfrequentie van 1.5 GHz.

c. [5 punten] Bereken de theoretische pieksnelheid in GFLOPS/s die deze processor kan behalen.

d. [5 punten] Ook GPUs ontkomen niet aan het feit dat de geheugenlatency hoog is. De vele kleine GPU "cores" kunnen allen tegelijkertijd een load of store-operatie starten. Omschrijf het mechanisme dat in GPUs wordt gebruikt om deze latency te verbergen.