Lab 1 - Kennismaking met SystemC en het mmMIPS model

Het doel van dit lab is de kennismaking van studenten met:

- de hardware beschrijvingstaal SystemC (RTL-niveau),
- het modelleren van microprocessor hardware in SystemC,
- het simuleren van SystemC modellen,
- het modificeren van de instructie verzameling van een microprocessor,
- de software die daarvoor gebruikt kan worden.

Hiervoor wordt het niet synthetiseerbare SystemC model van een **single-cycle mini-mini MIPS** processor (**mmMIPS**) gebruikt. Tevens wordt de software beschreven in "Software mMIPS-lab" gebruikt.

Het is de bedoeling dat de studenten dit model en deze software op hun laptop hebben geïnstalleerd volgens de instructies in "Software mMIPS-lab Setup" voordat zij naar dit lab komen.

Opdrachten

Dit lab bestaat uit drie opdrachten. In de eerste twee opdrachten ga je kennis maken met de single-cycle mini-mini MIPS en zijn simulatie omgeving. In de laatste opdracht ga je de mmMIPS instructie set uitbreiden. Alleen de **derde** opdracht moet afgetekend worden.

1. Kennismaking met het mmMIPS model en de software

Je gaat in deze eerste opdracht kennis maken met het mmMIPS model en de software die in dit practicum wordt gebruikt. Hiervoor gaan je het volgende programma simuleren met het SystemC model van de mmMIPS. Het programma realiseert de volgende berekening: a=b+c-d. Het programma leest de invoer uit het data (RAM) geheugen. Het resultaat wordt ook in het data geheugen opgeslagen.

1.asm

```
.set noat
.set noreorder
.align 2
lw $16, 0($0)       # $s0 = b = memory[0]
lw $17, 4($0)       # $s1 = c = memory[4]
lw $18, 8($0)       # $s2 = d = memory[8]
add $16, $16, $17   # b += c
sub $16, $16, $18   # (b+c) -= d
sw $16, 12($0)      # memory[12] = (b+c)-d
```

Om deze opdracht uit te voeren kun je het onderstaande stappenplan volgen.

1. Start *Visual C++* met het project *single_cycle_mmmips*. Dit project staat in de folder d:\mmips\mips\single_cycle_mmmips.
2. De mmMIPS processor moet nu gecompileerd worden. In *Visual C++* kun je met behulp van de menu opties **Build** en **Build single_cycle_mmmips** het SystemC model van de mmMIPS processor compileren.
3. Het programma dat op de mmMIPS moet draaien (*1.asm*) is al reeds gecompileerd in het bestand (*1.bin*). Om dit programma daadwerkelijk op de mmMIPS te draaien moet je de volgende stappen uitvoeren:

(a) Open een Cygwin shell en type het volgende commando:
cd d:/mmips/mips/single_cycle_mmmips

(b) Kopieer *1.bin* naar *mips_rom.bin*.

(c) Vertaal de hexadecimale programma code voor de a=b+c-d berekening terug naar MIPS assembler instructies, door het volgende commando uit te voeren:

disas mips_rom.bin

Vergelijk het resultaat van dit commando, dat je op het scherm kan bekijken, met het programma voor de berekening a=b+c-d zoals dat staat beschreven in het bestand *1.asm*.

4. Als volgende stap moet het data geheugen van de mmMIPS processor voorzien worden van de juiste gegevens. Dit doe je door in het programma *HDD Hex Editor* de juiste invoer (b=20, c=35,d =10) in het RAM geheugen te plaatsen. Hiervoor moet je de volgende procedure volgen:

(a) Open het bestand *mips_ram.bin* in *HDD Hex Editor*.

(b) Vervang de inhoud van het bestand door nullen.

(c) Plaats de hexadecimale getallen: 14, 23 en 0a in de eerste drie woorden van het RAM geheugen, dwz. in de bytes nr. 4, 8 en 12.

5. Als laatste stap ga je nu het programma dat de berekening a=b+c-d uitvoert op het gecompileerde SystemC model van mmMIPS simuleren. Dit doe je door de volgende stappen uit te voeren:

(a) Open een Cygwin shell en type het volgende commando:
cd d:/mmips/mips/single_cycle_mmmips

(b) Simuleer het programma in mips_rom.bin op de gegevens in het bestand mips_ram.bin met het model van de mmMIPS (single_cycle_mmmips.exe), door het volgende commando uit te voeren:

./single_cycle_mmmips.exe 200

Het argument (200) dat aan het programma wordt meegegeven geeft de duur van de simulatie in cycles aan. Aan het eind van de simulatie wordt de inhoud van het data geheugen opgeslagen in het bestand *mips_ram.dump*.

(c) Bekijk met *HDD Hex Editor* de resultaten van de simulatie in het bestand *mips_ram.dump*. Het resultaat van de berekening staat in het vierde woord (byte 16) van het geheugen.

(d) Bekijk en analyseer met *WinWave* de waveforms van verschillende mmMIPS signalen tijdens de simulatie van het programma. Je kunt deze waveforms openen door te dubbel klikken op het bestand *mips.vcd* in de folder d:\mmips\mips\single_cycle_mmmips. Selecteer met behulp van **Search** en **Signal Search Tree** van het *WinWave* menu de volgende signalen voor deze analyse:

- *bus_pc* - program counter output
- *bus_imem1* - instruction memory output
- *bus_decoder_instr_25_21* - address of REG 1
- *bus_decoder_instr_20_16* - address of REG 2
- *bus_registers_1* - ALU input
- *bus_mux2* - ALU input
- *bus_alu_result* - ALU output
- *bus_dmem1* - data memory output
- *clock* - clock

(e) Verander de gegevens in *mips_ram.bin* en voer het commando *./single_cycle_mmmips.exe 200* nog een keer uit met deze nieuwe gegevens. Bekijk hierna opnieuw de resultaten met *HDD Hex Editor* en *WinWave*.

2. Simulatie van de berekening a=(b+c)*d

Je gaat in deze tweede opdracht een programma simuleren op de mmMIPS processor dat de berekening a=(b+c)*d realiseert. Net als in de vorige opdracht leest het programma de invoer (b=30, c=70 en d=5) uit het data geheugen (mips_ram.bin). Het resultaat van de berekening wordt aan het einde van het programma weer in het data geheugen opgeslagen.

2.asm

```
.set noat
.set noreorder
.align 2
lw $16, 0($0)           # constant 1 (loop increment) = memory[0]
lw $17, 4($0)           # b = memory[4]
lw $18, 8($0)           # c = memory[8]
lw $19, 12($0)          # d = memory[12]
add $17, $17, $18       # b += c
sub $20, $20, $20       # result = 0
sub $21, $21, $21       # count = 0
LOOP:
beq $21, $19, END       # if count == d goto END
add $20, $20, $17       # result += (b+c)
add $21, $21, $16       # count += 1
beq $1, $2, LOOP        # goto LOOP
END:
sw $20, 16($0)          # memory[16] = result
```

Om de MLT instructie te simuleren moet je de volgende stappen uitvoeren:

1. Kopieer *2.bin* naar *mips_rom.bin*.
2. Controleer met *disas* of de code in *2.bin* overeenkomt met de assembler instructies in *2.asm*.
3. Plaats de hexadecimale getallen 01, 1e, 46 en 05 (die corresponderen met de decimale getallen: 1, 30, 70 en 5) in de eerste vier woorden (byte nummers 4, 8, 12 en 16) van het data geheugen (*mips_ram.bin*).
4. Voer het commando *./single_cycle_mmmips.exe 2000* uit. Let op: de simulatie duur is nu 2000 cycles.
5. Bekijk de resultaten met *HDD Hex Editor* en *WinWave*.

3. Uitbreiding van de mmMIPS met een *MLT* (*) instructie

In deze opdracht ga je de mmMIPS uitbreiden met de multiply (*MLT*) instructie. Deze instructie realiseert de vermenigvuldiging-operatie * in hardware. Op de mmMIPS met de uitgebreide instructie verzameling ga je vervolgens het programma simuleren dat de berekening a=(b+c)*d realiseert. Dit programma maakt gebruik van de *MLT* instructie. Het programma in MIPS assembler language is beneden afgebeeld en de corresponderende hexadecimale programma code die in het bestand *3.bin* staat.

3.asm

```
.set noat
.set noreorder
.align 2
lw $16, 0($0)           # constant 1 (loop increment) = memory[0]
lw $17, 4($0)           # b = memory[4]
lw $18, 8($0)           # c = memory[8]
lw $19, 12($0)          # d = memory[12]
add $17, $17, $18       # b += c
mlt $17, $17, $19       # (b+c) *= d (not a real MIPS instruction!)
sw $17, 16($0)          # memory[16] = result
```

Om de uitbreiding van de mmMIPS te testen moet je de volgende stappen uitvoeren:

1. Bestudeer hoe de ADD instructie is geïmplementeerd in de ALU en ALUctrl van het mmMIPS model in *single_cycle_mmmips*. Een MLT kan op dezelfde manier geïmplementeerd worden. Om dit te doen heb je de functiecode en opcode nodig die voor de MLT instructie gebruikt wordt in *3.bin*. Gebruik het commando *disas 3.bin* om te achterhalen wat de functiecode en opcode van de MLT instructie zijn. De MLT instructie is geen officiele MIPS instructie. Daarom staan de functiecode en opcode die hiervoor gebruikt zijn niet in het boek "Computer Organization & Design: The Hardware / Software Interface".

2. Voeg de MLT instructie aan het mmMIPS model toe.

3. Kopieer *3.bin* naar *mips_rom.bin*.

4. Voer het commando *./single_cycle_mmmips.exe 2000* uit.

5. Bekijk de resultaten met *HDD Hex Editor* en *WinWave*.

6. Vergelijk de waveforms die je krijgt uit de simulaties van het programma dat de berekening a = (b+c)*d realiseert op de mmMIPS met en zonder de speciale MLT instructie. Wat is het meest belangrijke resultaat dat verkregen is door het uitbreiden van een processor met de instructie MLT?