

ASSIGNMENT WEEK 3

Titel van het document

30 september 2013

Student:
Sjoerd Wenker
10617558

Docent:
drs. A. van Inge

Cursus:
Architectuur &
Computerorganisatie

Vakcode:
5062ARCO6Y

1 Introductie

Dit is het LabReport van de opdrachten van week 3 van Architectuur & Computerorganisatie.

2 Resultaten

2.1 Assignment 1

1a Het resultaat van 0x59 vermenigvuldigd met 0x14 in binary format:

Multiplicand	0x59		1011001
Multiplier	0x14	x	<u>10100</u>
			0000000
			0000000
			1011001
			0000000
			<u>1011001</u>
			11011110100

1b Het resultaat van 0x59 gedeeld door 0x14:

	100	Quotient
Divisor 10100	1011001	Dividend
	<u>-10100</u>	
	10	
	100	
	<u>1001</u>	
	1001	Remainder

2.2 Assignment 2

2a Zie Bijlage A (assignment2a.wasm)

2b Zie Bijlage B (assignment2b.wasm)

2c Bij het eerste programma is het aantal clock ticks gelijk aan het aantal instructies en is het CPI dus gelijk aan 1.0

Bij het tweede programma is de CPI hoger dan 1.0

2.3 Assignment 3

3a Bij Multiplicand is een hexadecimaal getal ingevuld. Bij Multiplier een binair getal. Als dit omgedraait is, komt er een totaal andere waarde uit. **3b**

Een multiply instruction duurt meerdere clockticks. CPI is dus meer dan 1.

3c Er zal een extra control line moeten komen die aangeeft welke data naar het register moet. Ook zijn er meer datalines nodig voor het resultaat van de multiplier.

3d De kloktijd moet minimaal 430 ps zijn, omdat deze multiplier er zolang over doet.

3e OpCode voor de MULT instructie: 011000. De Control moet met behulp van AND componenten en multiplexers controleren of de instructie gelijk is aan 011000.

3f De control moet aangeven of de waarde van de multiplier of de ALU naar het register moet. Met een multiplexer kan dan de goede data doorgegeven worden.

3g Met een AND Component tussen Start en de Control zou dit gemakkelijk opgelost kunnen worden. Deze AND poort heeft dan als input de Start en het omgekeerde van Ready.

3 Discussie

3.1 Assignment 3g

Het was mij niet helemaal duidelijk hoe ik met behulp van de Editor extra hardware kan toevoegen aan Multiplier_plus_clock.sim-pl

4 Bijlage A: assignment2a.wasm

```
@include "Mips.wasm"
.data MyRegisters : REGISTERS
0: WORD Zero 0
1: WORD Multiplicand 0x59
2: WORD Multiplier 0x14
3: WORD Product 0
.data MyMemory : DATAMEM
0x00 : WORD result
.code MyCode: MIPS, MyMemory
```

```
ADDI $4, $0, 0
ADDI $5, $0, 32
ADDI $7, $7, 2
```

```
Start:
ANDI $8, $2, 1
BEQ $8, $0, Shift
ADD $3, $3, $1
```

```
Shift:
SLL $1, $1, $7
SRL $2, $2, $7
ADDI $4, $4, 1
BNE $4, $5, Start
```

```
Stop:
```

5 Bijlage B: assignment2b.wasm

```
@include "Mips.wasm"
.data MyRegisters : REGISTERS
0: WORD Zero 0
1: WORD Multiplicand 0x59
2: WORD Multiplier 0x14
3: WORD Product 0
.data MyMemory : DATAMEM
0x00 : WORD result
.code MyCode: MIPS, MyMemory
```

```
ADDI $7, 7, 2
```



Start :

ANDI\$8, \$2, 1

BEQ\$8, \$0, *Shift*

ADD\$3, \$3, \$1

Shift :

SLL\$1, \$1, 7

SRL \$2, \$2, \$7

BNE \$2, \$0, Start