FACULTEIT DER NATUURWETENSCHAPPEN, WISKUNDE EN INFORMATICA

Wouter Geraedts

Processen & Processoren

Radboud Universiteit Nijmegen

Overzicht

Welkom op het 5^{e} werkcollege van Processen & Processoren!

- Uitwerkingen vorige opgavenserie
- Behandelen (oefen)opgaven

Opgave 1

Bereken de volgende bitshifts en bitrotaties. Je mag aannemen dat het altijd om 8-bits binaire getallen gaat:

(a)	1101 1001	ROL 2
(b)	1110 1010	SHL 0
(c)	1000 0000	ROR 5
(d)	0101 0101	SHR 7
(e)	1100 0100	SHR 3
(f)	0001 0100	SHL 1

Opgave 1 2

Bereken de volgende bitshifts en bitrotaties. Je mag aannemen dat het altijd om 8-bits binaire getallen gaat:

(a)	1101 1001	ROL 2	=	0110 0111
(b)	1110 1010	SHL 0	=	1110 1010
(c)	1000 0000	ROR 5	=	0000 0100
(d)	0101 0101	SHR 7	=	0000 0000
(e)	1100 0100	SHR 3	=	1111 1000
(f)	0001 0100	SHL 1	=	0010 1000

Opgave 2

De practicum-processor heeft geen bitshift-instructie, maar wel rotatie-instructies. Hoe kun je toch een bitshift-operatie uitvoeren?

Opgave 2

De practicum-processor heeft geen bitshift-instructie, maar wel rotatie-instructies.
Hoe kun je toch een bitshift-operatie uitvoeren?

Met een bitmask kun je de *lege* plekken opvullen.

SHL => 0'en, gebruik van de AND.
SHR => sign-bit van het oorspronkelijke getal.
Bij negatief getal, gebruik van de OR.

Opgave 3

De vlaggen worden gezet op basis van het resultaat van de laatste ALU-berekening. Kunnen de vlaggen ook gebruikt worden om twee getallen (in registers) met elkaar te vergelijken? Zo ja, hoe?

Opgave 3

Ja, dat kan!

Opgave 3

Ja, dat kan! We hebben namelijk o.a. een Negative- en een Zero-flag.

- Als $x - y = 0$, dan moet $x = y$. (Z-flag)
- Als $x - y < 0$, dan moet $x < y$. (N-flag)
- Als $x - y > 0$, dan moet $x > y$. (Anderzijds)

Zie ook specificatie practicum-processor.

Opgave 4

Wat kun je bij overïňĆow zeggen over het voorteken van het resultaat?

Opgave 4

Wat kun je bij overïňĆow zeggen over het voorteken van het resultaat?

Deze is dan omgedraait. Met overflow: $x < y$ als n xor o.

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

1. $a + b = c$

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

1. $a + b = c$
2. we rekenen in het decimale stelsel; dit is dan te herschrijven als:

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

1. $a + b = c$
2. we rekenen in het decimale stelsel; dit is dan te herschrijven als:
3. $a_0 \cdot 10^0 + \ldots + a_n \cdot 10^n + b_0 \cdot 10^0 + \ldots + b_n \cdot 10^n = c_0 \cdot 10^0 + \ldots + c_n \cdot 10^n$

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

1. $a + b = c$
2. we rekenen in het decimale stelsel; dit is dan te herschrijven als:
3. $a_0 \cdot 10^0 + \ldots + a_n \cdot 10^n + b_0 \cdot 10^0 + \ldots + b_n \cdot 10^n = c_0 \cdot 10^0 + \ldots + c_n \cdot 10^n$
4. stel, we rekenen nu in restklassen modulo 9, dan is het volgende gelijk:

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

1. $a + b = c$
2. we rekenen in het decimale stelsel; dit is dan te herschrijven als:
3. $a_0 \cdot 10^0 + \ldots + a_n \cdot 10^n + b_0 \cdot 10^0 + \ldots + b_n \cdot 10^n = c_0 \cdot 10^0 + \ldots + c_n \cdot 10^n$
4. stel, we rekenen nu in restklassen modulo 9, dan is het volgende gelijk:
5. $a_0 + \ldots + a_n + b_0 + \ldots + b_n = c_0 + \ldots + c_n$

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

1. $a + b = c$
2. we rekenen in het decimale stelsel; dit is dan te herschrijven als:
3. $a_0 \cdot 10^0 + \ldots + a_n \cdot 10^n + b_0 \cdot 10^0 + \ldots + b_n \cdot 10^n = c_0 \cdot 10^0 + \ldots + c_n \cdot 10^n$
4. stel, we rekenen nu in restklassen modulo 9, dan is het volgende gelijk:
5. $a_0 + \ldots + a_n + b_0 + \ldots + b_n = c_0 + \ldots + c_n$
6. dit komt overeen met de berekening van de negenproef

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

1. $a + b = c$
2. we rekenen in het decimale stelsel; dit is dan te herschrijven als:
3. $a_0 \cdot 10^0 + \ldots + a_n \cdot 10^n + b_0 \cdot 10^0 + \ldots + b_n \cdot 10^n = c_0 \cdot 10^0 + \ldots + c_n \cdot 10^n$
4. stel, we rekenen nu in restklassen modulo 9, dan is het volgende gelijk:
5. $a_0 + \ldots + a_n + b_0 + \ldots + b_n = c_0 + \ldots + c_n$
6. dit komt overeen met de berekening van de negenproef
7. als de negenproef niet klopt, kan de vergelijking in stap 5 niet kloppen

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

1. $a + b = c$
2. we rekenen in het decimale stelsel; dit is dan te herschrijven als:
3. $a_0 \cdot 10^0 + \ldots + a_n \cdot 10^n + b_0 \cdot 10^0 + \ldots + b_n \cdot 10^n = c_0 \cdot 10^0 + \ldots + c_n \cdot 10^n$
4. stel, we rekenen nu in restklassen modulo 9, dan is het volgende gelijk:
5. $a_0 + \ldots + a_n + b_0 + \ldots + b_n = c_0 + \ldots + c_n$
6. dit komt overeen met de berekening van de negenproef
7. als de negenproef niet klopt, kan de vergelijking in stap 5 niet kloppen
8. de vergelijking is echter gelijk met de additie-vergelijking, dus klopt deze

Opgave 5

Bewijs d.m.v. restklassen dat de negenproef werkt.

1. $a + b = c$
2. we rekenen in het decimale stelsel; dit is dan te herschrijven als:
3. $a_0 \cdot 10^0 + \ldots + a_n \cdot 10^n + b_0 \cdot 10^0 + \ldots + b_n \cdot 10^n = c_0 \cdot 10^0 + \ldots + c_n \cdot 10^n$
4. stel, we rekenen nu in restklassen modulo 9, dan is het volgende gelijk:
5. $a_0 + \ldots + a_n + b_0 + \ldots + b_n = c_0 + \ldots + c_n$
6. dit komt overeen met de berekening van de negenproef
7. als de negenproef niet klopt, kan de vergelijking in stap 5 niet kloppen
8. de vergelijking is echter gelijk met de additie-vergelijking, dus klopt deze
9. dus klopt de negenproef

Oefenopgaven

Op naar de (oefen)opgaven!

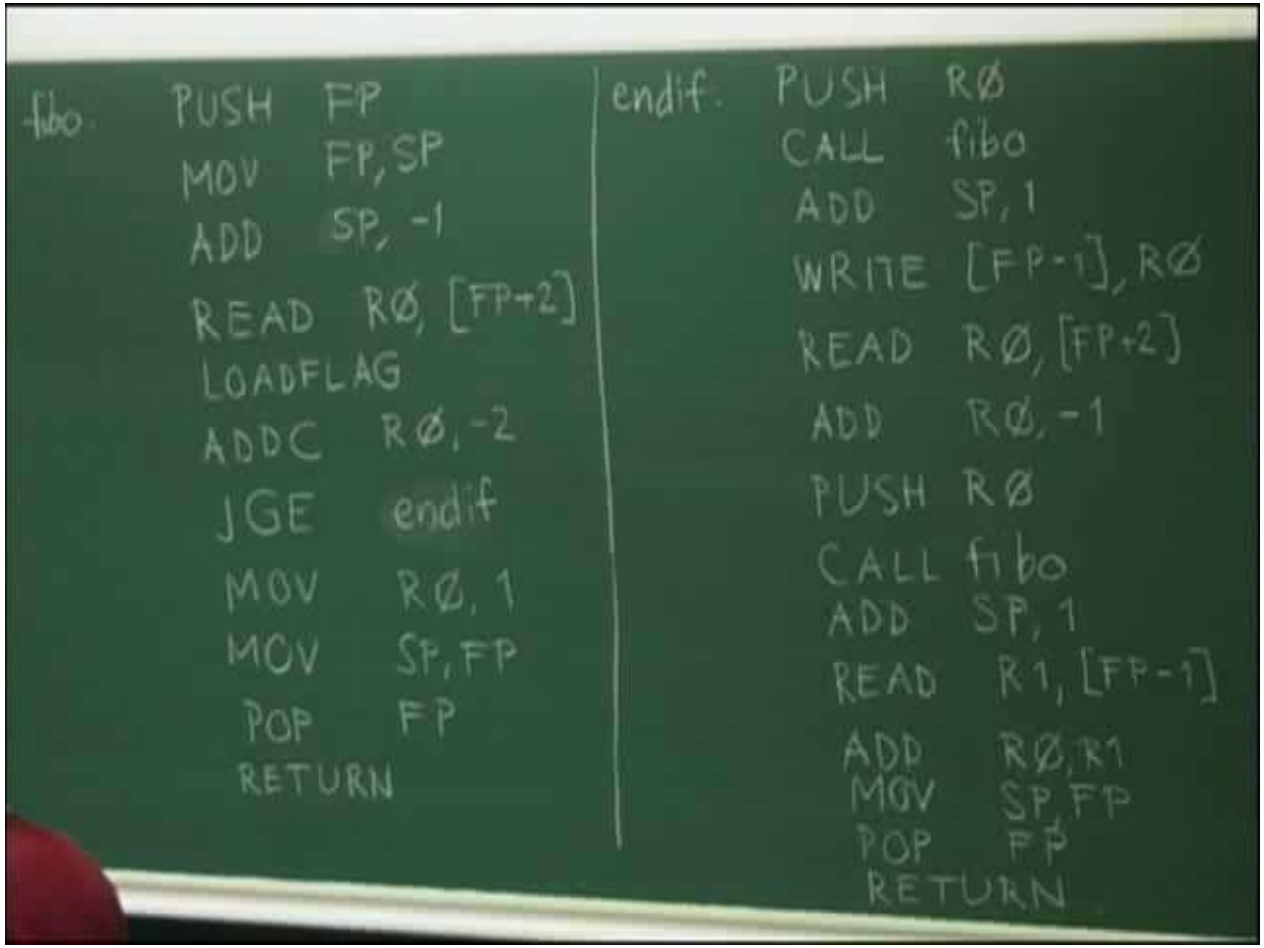
Opgave 1

Kijk naar het assembly-programma in de opname van het hoorcollege van 15 maart. David noemde rond 5:35 dat de practicum- processor de instructie `READ R0, [FP+2]` niet echt kan verwerken, maar kleinere stapjes moet maken. Welke andere instructies kan de practicum-processor niet direct uitvoeren? Welke kleinere stapjes kan de practicum-processor wel maken die daarmee overeenkomen?

Opgave 1 2

1. Ga naar de werkplaats
2. Ga naar de planning
3. Ga naar de opnames van het vorige college
4. David legt uit dat instructies in delen uitgevoerd moeten worden
5. Pak een videofragment waar zijn volledige programma op staat

Opgave 1 3


fibo. PUSH FP
MOV FP, SP
ADD SP, -1
READ RØ, [FP+2]
LOADFLAG
ADDC RØ, -2
JGE endif
MOV RØ, 1
MOV SP, FP
POP FP
RETURN
endif. PUSH RØ
CALL fibo
ADD SP, 1
WRITE [FP-1], RØ
READ RØ, [FP+2]
ADD RØ, -1
PUSH RØ
CALL fibo
ADD SP, 1
READ R1, [FP-1]
ADD RØ, R1
MOV SP, FP
POP FP
RETURN

Opgave 1 4

1. Ga naar de werkplaats
2. Ga naar het practicum
3. Download de beschrijving van de processor
4. Leg deze naast het screenshot
5. Zoek instructies die niet in de beschrijving staan

Opgave 1 5

Voor deze onbekende instructies dien je programma-fragmenten te schrijven die de functionaliteit emuleren.

We doen een voorbeeld: JGE

Opgave 1 6

LOADHI R_4, HI x
ADDIFT R_4, LO x

[ALU-operatie(s) t.b.v. vergelijking]

MOVIFGE R_7, R_4

(R_7 is de program-counter)

De overige opgaven

De overige opgaven doen we op het bord / demonstreer ik.

Einde

Fin