Theorie week 2 – Microprocessorarchitectuur

Deze week werk je verder aan leerdoel 1 dat is beschreven in de cursushandleiding.

Je gaat deze les leren:

- hoe een CPU globaal is opgebouwd en werkt;
- hoe een CPU machinecodeinstructies uitvoert volgens de zogenoemde Von Neumann-cyclus;
- hoe de stack en de stack pointer gebruikt worden bij het uitvoeren van functieaanroepen en returns;

Studiemateriaal week 2

De theorie die je deze week moet bestuderen kun je vinden in de volgende paragrafen van je boek[1]:

7.4 CPU

Verdieping week 2

De in deze paragraaf besproken onderwerpen bieden verdieping aan op de in het boek gegeven informatie. De in deze paragraaf besproken onderwerpen behoren echter niet tot de toetsstof.

In figuur 7.31 op pagina 134 van je boek staat een conditiecoderegister (CCR) afgebeeld waarvan alleen het N-, Z-, V- en C-bit in de tekst worden besproken. Het betreft hier het CCR van de nu verouderde 6809 microprocessor[2]. Het E-, F- en I-bit hebben te maken met het interruptsysteem van de 6809 en zijn verder niet interessant. Het H-bit komt echter ook in de statusregisters van vele andere processoren voor[3]. Zoek uit wat de betekenis is van het H-bit en waarvoor dit bit wordt gebruikt. Welke naam heeft het bit met dezelfde functionaliteit

[1] Leo van Moergestel. *Computersystemen en embedded systemen*. 4de ed. Boom, 2016. ISBN: 9-789058-754233.

[2] Zie https://en.wikipedia.org/wiki/Motorola_6809

[3] Bijvoorbeeld in de Atmel[4] AVR, de Intel x86 (en dus ook de i7) en de Microchip PIC. In de laatste twee van deze architecturen heeft dit bit echter een andere naam.

[4] Atmel is in 2016 overgenomen door Microchip. De AVR microcontroller is vooral bekend omdat hij gebruikt wordt in het bij veel hobbyisten bekende Arduino bordje.

in het statusregister van de Intel x86 architectuur? Welke naam heeft het bit met dezelfde functionaliteit in het statusregister van de Microchip PIC architectuur?

Op pagina 141 van je boek wordt gesproken over instructies die de mode van een CPU veranderen (van supervisor naar user mode), zonder dat deze modes verder worden uitgelegd of besproken. Zoek uit waarom veel processoren over verschillende modes (zoals supervisor en user mode) beschikken en waarvoor deze modes worden toegepast.

Opgaven week 2

Gebruik bij het maken van de onderstaande opgaven in eerste instantie alleen je boek. Maak alleen als je er met behulp van het boek niet uitkomt, gebruik van andere bronnen zoals het internet. Bij de toets mag je immers ook alleen je boek maar gebruiken.

2.1 De Von Neumann-cycles bestaat uit vier stappen die steeds herhaald worden: fetch, decode, execute en store. We gaan in deze opgave uit van een processor die is opgebouwd zoals weergegeven in figuur 7.30 op pagina 133 van je boek. Verklaar je antwoorden!

- **A** In welke stappen wordt de ALU mogelijk gebruikt?
- **B** In welke stap(pen) wordt het instruction-register beschreven.
- **C** In welke stap(pen) wordt het satusregister mogelijk uitgelezen en/of beschreven.
- **D** In welke stap(pen) worden de general purpose registers mogelijk beschreven.
- **E** In welke stap(pen) worden de general purpose registers mogelijk uitgelezen.
- **F** De program counter wordt verhoogd in de fetch stap. In welke stap kan de program counter ook aangepast worden?

2.2 Beschouw figuur 7.34, figuur 7.36 en de bijbehorende uitleg. Bepaal voor elk van de onderstaande bewerkingen de benodigde waarde van het control word.

- **A** R3 = R3 + R0
- **B** R2 = R2 – 1
- **C** R1 = not R1

D R0 = R0 exor R2

2.3 Een 16-bits variabele met de waarde 0xCDEF in een little endian systeem heeft als startadres 0xFF00FF00. Wat wordt er opgeslagen op de adressen 0xFF00FF00 en 0xFF00FF01?

2.4 Bij deze opgave gaan we uit van een big endian processor met 16 adreslijnen en 8 datalijnen. Een instructie bestaat uit één of meer bytes. Een CALL instructie is 3 bytes lang, de laatste twee bytes bevatten het 16-bits adres waar de subroutine begint. Een RET instructie is 1 byte lang. De JMP instructie kan gebruikt worden om de program counter te laden. Deze instructie is 3 bytes lang, de laatste twee bytes bevatten het 16-bits adres dat in de program counter wordt geladen. De program counter en de stack pointer zijn (uiteraard) beide 16 bits breed. De program counter wijst naar de volgende instructie die zal worden uitgevoerd. De stack pointer wijst naar de eerste vrije plaats op de stack en groeit richting adres 0x0000. In figuur 1 is een programma gegeven wat deze processor uitvoert. In figuur 2 is de stack weergegeven.

A Welke waarde bevat de stack pointer als de program counter nadat het programma is gestart voor de eerste keer de waarde 0x2008 bevat?

B Teken de inhoud van de stack op het bij deelopgave A gegeven moment.

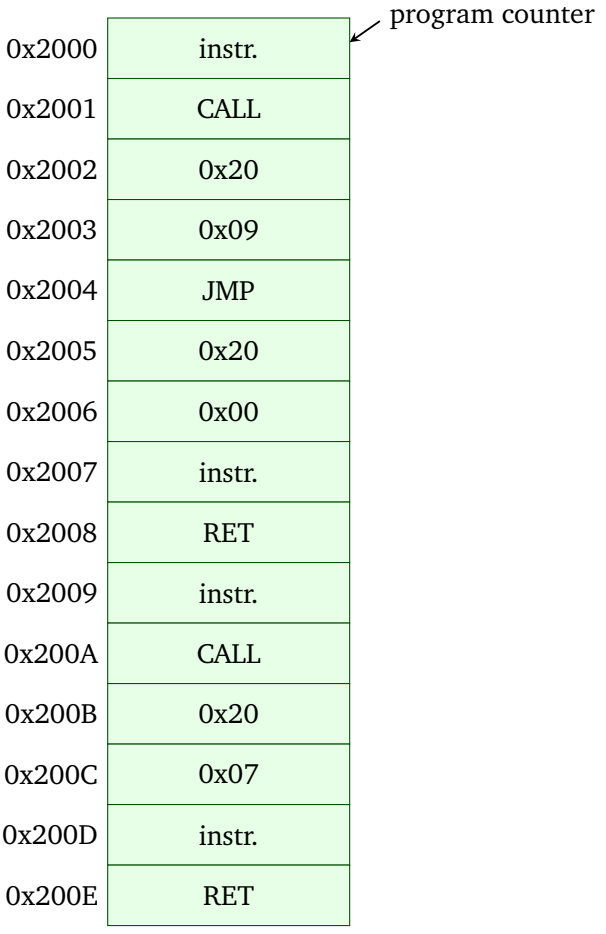
C Welke waarde bevat de stack pointer als de program counter nadat het programma is gestart voor de eerste keer de waarde 0x2004 bevat?

D Welke waarde bevat de stack pointer als de program counter nadat het programma is gestart voor de tweede keer de waarde 0x2008 bevat?

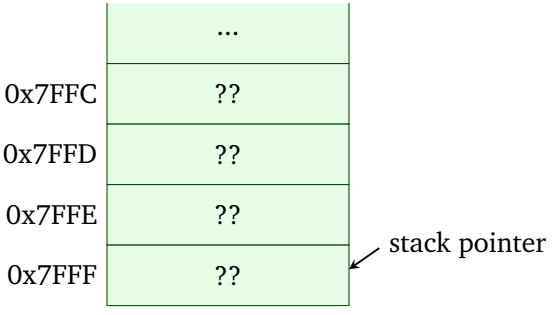
E Teken de inhoud van de stack op het bij deelopgave D gegeven moment.

F Een stack wordt ook wel een LIFO buffer genoemd. De afkorting staat voor Last In, First Out. Geef een verklaring voor deze naam, met het in figuur 1 gegeven programma als voorbeeld.

2.5 Op pagina 144 van je boek staat: “Voor de meeste CPU’s zal de stack pointer bij het opslaan van een returnadres verlaagd worden; we zeggen dat de stack naar beneden groeit”. Dit is inderdaad de meest voor de hand liggend aanpak.



Figuur 1: Het programma dat uitgevoerd wordt.



Figuur 2: De stack op het moment dat het programma begint.

A Bij een CPU waarbij de stack naar beneden groeit wordt de stack pointer door de assembleeprogrammeur of door de C-compiler in het begin van een programma vaak op het laatste plekje in het RAM geplaatst. Verklaar waarom de stack pointer vaak op deze manier geïnitialiseerd wordt.

B Waarom is het de meest voor de hand liggende aanpak om de stack naar beneden te laten groeien?

2.6 Als bij de in opgave 2.4 beschreven processor erg veel data op de stack wordt geschreven zal de stack pointer uiteindelijk andere (in het RAM opgeslagen) data overschrijven. Dit wordt een stack overflow genoemd.

A Een stack overflow kan soms heel moeilijk op te sporen zijn. Leg uit waarom?

B Geef een zo kort mogelijk programma in de stijl van figuur 1 dat (uiteindelijk) altijd een stack overflow zal veroorzaken.

C Geef een zo kort mogelijk C-programma dat (uiteindelijk) altijd een stack overflow zal veroorzaken. Test dit programma op je favoriete microcontroller.

2.7 Maak vraag 7.7 tot en met 7.11 op pagina 157 van je boek. Deze vragen hebben allemaal betrekking op figuur 7.58 dat is weergegeven op diezelfde pagina.

2.8 VERDIEPENDE OPDRACHT

Op pagina 132 van je boek staat: “Nu zijn veel CPU’s in staat om op byteniveau te adresseren. Daarmee wordt bedoeld dat ook CPU’s met een brede databus van bijvoorbeeld 64 bits (dus 8 bytes) in staat zijn toch een enkele byte aan te duiden.”. Dit houdt in dat elke byte in het geheugen zijn eigen adres heeft, maar dat de CPU indien gewenst meerdere bytes in één keer kan benaderen. Het maximaal aantal in één keer te benaderen bytes wordt bepaald door de breedte van de databus. In figuur 7.29 is weergegeven hoe dit gerealiseerd kan worden voor een CPU met een databus van 64 bits en adressen van 32 bits. In dit figuur is te zien hoe de adreslijnen A0 tot en met A2 vervangen zijn door de byte enable lijnen BE0 tot en met BE7. Intern in de CPU worden de byte enable signalen afgeleid van de adreslijnen A0 tot en met A2 en van het aantal bytes dat de CPU wil benaderen.

A Hoeveel bytes kan deze CPU in totaal adresseren?

B Hoeveel bytes kan deze CPU in één keer inlezen of wegschrijven? Let op! In de laatste zin op pagina 132 van je boek staat: "We adresseren dus 64 bytes eenheden". Dit is echter *niet* correct! Corrigeer deze zin in je boek.

C Geef de benodigde waarden van de adreslijnen A3 tot en met A31 en de byte enable signalen BE0 tot en met BE7 als de CPU één byte wil inlezen van adres 0x00FF00F8.

D Welke acht datalijnen moet de processor bij deelopgave C inlezen?

E Geef de benodigde waarden van de adreslijnen A3 tot en met A31 en de byte enable signalen BE0 tot en met BE7 als de CPU één byte wil inlezen van adres 0x00FF00FB.

F Welke acht datalijnen moet de processor bij deelopgave E inlezen?

G Geef de benodigde waarden van de adreslijnen A3 tot en met A31 en de byte enable signalen BE0 tot en met BE7 als de CPU twee bytes wil inlezen van adres 0x00FF00FC en adres 0x00FF00FD.

H Welke zestien datalijnen moet de processor bij deelopgave G inlezen?

I Geef de benodigde waarden van de adreslijnen A3 tot en met A31 en de byte enable signalen BE0 tot en met BE7 als de CPU vier bytes wil inlezen van adres 0x00FF00FC tot en met adres 0x00FF00FF.

J Welke datalijnen moet de processor bij deelopgave I inlezen?

K Geef de benodigde waarden van de adreslijnen A3 tot en met A31 en de byte enable signalen BE0 tot en met BE7 als de CPU acht bytes wil inlezen van adres 0x00FF00F8 tot en met adres 0x00FF00FF.

L Welke datalijnen moet de processor bij deelopgave K inlezen?

M Als deze CPU een 64-bits getal (b.v. een variabele van het type `double` in de programmeertaal C) in één keer in moet kunnen lezen dan moet het beginadres van deze variabele deelbaar zijn door 8. Leg uit waarom.