Processoren 2014

Week 7: Assembly, linking and loading

Uiterste inleverdatum: 12 januari 2015

Lever de volgende opgaven in via email naar je studentassistent. Zorg dat [Proc]Week7 in het subject van je email staat zodat we die niet over het hoofd kunnen zien.

1) a) Vertaal het onderstaande programma naar C, C++ of Java.

```
          .START      0
          READ        [ARG1],R1
          READ        [ARG2],R2
LOOP:     SUBf        R1,R2,R0
          SUB.L       R2,R1,R2
          SUB.G       R1,R2,R1
          JUMP.NZ     LOOP
          WRITE       R1,[RESULT]
          HALT
ARG1:     .DATA       ...
ARG2:     .DATA       ...
RESULT:   .DATA       ...
```

   b) Om te begrijpen wat het programma doet, is het nuttig om dit programma met een paar argumenten uit te proberen. Wat is het resultaat van dit programma met de argumenten:

   - 42 en 8
   - 252 en 105
   - 10 en 9

   c) Voor de liefhebber: dit algoritme is voor het eerst zo'n 2300 jaar geleden opgeschreven en draagt nog steeds de naam van de oorspronkelijke auteur. Het zou niet moeilijk moeten zijn om met de antwoorden van onderdeel b te achterhalen wat dit algoritme doet. Wat berekent het? Werkt het ook met negatieve argumenten?

2) We geven eerst een licht gewijzigde formulering van deze vraag die vorig jaar gesteld is op het tentamen van 24 januari 2014.

   Schrijf een programma in practicum assembly dat het minimum en maximum berekent van de elementen van een array dat start op de locatie array waarvan de lengte gegeven wordt door de locatie length. Het minimum en het maximum dienen aan het eind naar de locaties min en max weggeschreven te worden. De elementen van het array zijn één machinewoord groot (dwz. 32 bits) en moeten signed behandeld worden. Dit programma zal dus als volgt eruit zien.

```
        .START  0
        ...                     # dit gedeelte moet je zelf schrijven
        WRITE   ..., [min]
        WRITE   ..., [max]
        HALT
min:    .DATA   0
max:    .DATA   0
length: .DATA   ...             # Lengte van het array
array:  .DATA   ...             # Het eerste element van het array
        .DATA   ...             # Het tweede element van het array
        ...                     # Et Cetera
```

a) Formuleer eerst in pseudo C, C++ of Java een algoritme dat hetzelfde doet.

b) Formuleer nu aan de hand van je beschrijving van onderdeel a het programma in practicum assembly.

3) Een linker leest 4 objectbestanden om ze tot een executable te linken. De bestanden zijn 1100, 200, 1300 en 1700 bytes groot. De doelprocessor gebruikt byte adressering. Aangenomen dat ze in de genoemde volgorde ingeladen worden, wat wordt dan voor elk objectbestand de relocatie constante (d.w.z. welke offset moet in elk van de bestanden gebruikt in elke interne referentie om ze goed te reloceren)?