# Processoren 2014

## Week 2: Getallenrepresentatie

Uiterste inleverdatum: 24 november

Lever de volgende opgaven in via email naar je studentassistent. Zorg dat [Proc]Week2 in het subject van je email staat zodat we die niet over het hoofd kunnen zien.

1) Converteer de volgende getallen naar binair, octaal, decimaal en hexadecimaal: $142_{\text{dec}}$, $357_{\text{oct}}$ en $B4_{\text{hex}}$.

2) Zij gegeven woordbreedte $k = 8$. Geef de twee-complement representatie van de waarden -42, 127 en -117.

3) Beschouw de optelling van de volgende paren getallen bij woordbreedte $k = 8$.

   a) 11001101 en 11101100

   b) 01101100 en 01001111

   c) 10101010 en 01011111

   d) 11001000 en 10101011

   Geef aan wat de uitkomst van elk van deze optellingen wordt. Bij welke van deze optellingen wordt er een carry en bij welke een overflow gegenereerd?

4) In het college hebben we laten zien dat een aftrekking te realiseren is door middel van het optellen van de inverse + 1.

   a) Wat betekent een carry out die 1 is bij zo'n aftrekking?

   b) In onze practicum processor zit een instructie, ADC (Add with Carry), die ons in staat stelt om een multiprecisie optelling uit te voeren d.w.z we gebruiken de carry out van een vorige 32 bits optelling die tijdelijk is bewaard en voeren die als extra invoerbit bij een volgende optelling als carry in aan de 32 bits opteller.

      Beredeneer waarom dit ook een correct resultaat geeft als we een multiprecisie aftrekking willen uitvoeren als we deze implementeren via het optellen van de inverse waarbij we geen harde 1 als carry in aan de opteller voeren maar ook de vorige carry out. In de practicum processor is dit de SBC (Subtract with Carry) instructie.

5) Neem dezelfde paren getallen als bij opgave 3 en bepaal het resultaat als je ze van elkaar zou aftrekken. Neem hiervoor de implementatie uit het college nl door de bitten van de tweede operand te inverteren en dan bij de eerste operand op te tellen + 1. Bij welke van deze aftrekkingen wordt er nu een carry en een overflow gegenereerd?

# Facultatief

Op dit moment zou je al een beginnetje kunnen maken met de implementatie van je practicum processor door een 32 bits adder/subtractor te maken d.w.z. een schakeling waarbij je met een controle lijn kunt aangeven of je met de schakeling wil kunnen optellen dan wel aftrekken. In de hades bibliotheek vind je in models/rtlib.arith een component Addc die een optelling van twee vectoren realiseert met een carry in en een carry out. De 32 bits optelling heb je dus al. Het enige wat je nu nog nodig hebt is iets om de tweede operand conditioneel te inverteren. Bedenk nu dat $a \oplus 0 = a$ en $a \oplus 1 = \overline{a}$. De benodigde componenten vind je in rtlib.logic en rtlib.io.

Om je schakeling te testen kun je hem embedden in een testbench d.w.z. je schakeling leg je als hierarchische component in een test omgeving waarbij je met de rtlib.io.Ipinvector de invoer maakt en rtlib.io.Opinvector de uitvoer kunt bekijken.

Dit onderdeel is niet verplicht, maar door nu al met Hades en de practicum processor te starten vermijd je haastige spoed aan het einde van het trimester. Jullie studentassistent kan je natuurlijk wel altijd om feedback vragen.