Opgave 3

Overzicht opdracht

Deze opgave staat los van de vorige opgaven en heeft als doel het gebruik van optellingen in VHDL te begrijpen. De bedoeling is ook dat je leert werken met de package numeric_std (en de types unsigned en signed). **Daarom is het, uitzonderlijk in deze opgave, niet toegestaan om integers te gebruiken in je code**. Dit is puur om didactische redenen, want je mag normaal in VHDL wel degelijk met integers werken, wat heel dikwijls zeer handig is.

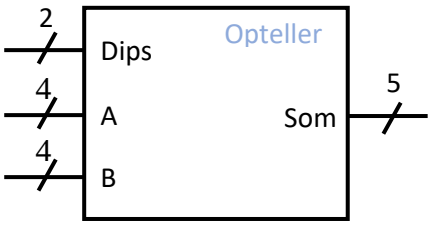
Enkel de dip switches en de leds worden gebruikt voor deze opgave:

- Als dip switch SW0 gelijk is aan
  - 0 dan moet je SW15-12 en SW11-8 beschouwen als 4-bit positief binaire getallen.
  - 1 dan moet je SW15-12 en SW11-8 beschouwen als 4-bit two's complement getallen.
- De som hiervan moet je tonen op LED15-11.
- Als dip switch SW1 gelijk is aan
  - 0 dan gebruik je 5 leds (LED15-11), zodat er nooit overflow kan optreden.
  - 1 dan gebruik je slechts 4 leds (LED14-11, terwijl LED15 uit blijft).

Begrijp wat er gebeurt in het geval van overflow in deze opgave. **Zorg ervoor dat je dit weet, want dit zal gevraagd worden op een test en/of op je verdediging van deze opgave!**

Beschrijving

Het blokschema ziet er als volgt uit, waarbij je uiteraard zelf de benamingen van de in- en uitgangen (en de component zelf) mag kiezen.



Let wel op dat je de juiste verbindingen legt in het .xdc bestand. Zo zal je bijvoorbeeld de meest linkse bit van A moeten verbinden met pin V10 (waar oorspronkelijk SW[15] stond in het .xdc bestand) en de meest rechtse bit van B met pin T8 (waar oorspronkelijk SW[8] stond). Zorg ervoor dat alle niet-gebruikte "SW" lijnen (SW7-2) in het .xdc bestand in commentaar staan, anders zal Vivado een foutmelding geven.

Zoals je in de theorie gezien hebt, moet je 1 bit méér voorzien voor het resultaat dan het grootste aantal bits van de ingangen om zeker te zijn dat je geen overflow kan hebben. In dit geval moet je dus 5 bits voorzien als je twee ingangen van 4 bits met elkaar wil optellen zonder kans op overflow. Het grootste positief binaire getal dat je kan voorstellen met 4-bits is namelijk 15, dus de grootst mogelijke som wordt dan 15+15=30, wat een 5-bit binair getal is (11110).

De "+" operator in VHDL gaat er echter standaard niet vanuit dat je een extra bit voorziet, dus hier zal je even wat moeten nadenken en uitzoeken hoe je dat juist moet forceren. Kijk hiervoor zeker even naar de voorbeeldoefeningen op Blackboard. Bovendien moet je ook even nakijken wat er gebeurt als je slechts vier bits voorziet voor de som en wat er dan juist gebeurt als er effectief

overflow optreedt. Kijk zeker na of de leds het gedrag vertonen dat je verwacht bij overflow voor zowel positief binaire als two's complement getallen.

Belangrijk! Het is niet de bedoeling dat je in VHDL een carry look ahead of ripple carry adder tot op poortniveau gaat ontwerpen! Dat is de kracht van VHDL: gebruik gewoon de "+" operator om de som te berekenen en Vivado zal dan zelf wel de juiste logica genereren. Bijvoorbeeld:

```
Som <= A + B;
```

Je zal in deze opgave problemen hebben met het omzetten van unsigned naar signed en vice versa, want stel dat je de ingangen A en B als unsigned definieert, moet je ze achteraf nog omzetten naar signed indien dip 0 (SW0) gelijk is aan 1. Hier zal je goed in de cursus (of de videoles over numeric_std) moeten opzoeken hoe je best kan converteren tussen deze verschillende types.