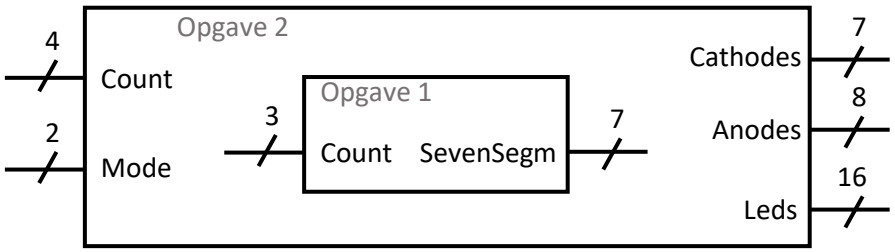
Opgave 2

Overzicht opdracht

Ontwerp een “manueel gestuurd” looplicht, gecombineerd met de snake opdracht van opgave 1. Je bent verplicht om opgave 1 in zijn geheel te gebruiken, door hiërarchie toe te passen (met component en port map). Bekijk zeker de tutorials, videolessen en/of voorbeeldoefeningen hierover als je niet weet hoe dit moet. Ook de testbench van opgave 1 gebruikt hiërarchie, dus daar kan je ook inspiratie uit halen.

Gebruik de twee meest rechtse dip switches als “Mode” ingang, de vier meest linkse dip switches als “Count” ingang, de 16 leds als looplicht en de 7-segment displays op het FPGA bord voor onder andere de snake van opdracht 1. In tegenstelling tot opgave 1, moet deze schakeling wél in de FPGA geconfigureerd worden. Dit heeft tot gevolg dat je de kathodes en de anodes van de 7-segment displays zal moeten aansturen op het FPGA bord.

Lees hiervoor zeker de handleiding van het Digilent Nexys bordje van p.22 tot 24, waar meer uitleg staat (en een schema) over de 7-segment displays. Hier kan je ook vinden of de anodes en kathodes van de displays actief hoog of laag zijn (‘1’ = aan of ‘1’ is uit?).



De twee Mode dip switches bepalen wat er juist moet gebeuren op de leds en 7-segment displays. In onderstaande tabel staan de verschillende modes opgesomd en daarna worden ze in meer detail besproken:

Mode	7-segment displays	leds
00	snake op alle displays	alles uit
01	alles uit	looplicht van rechts naar links
10	snake op lopende displays	looplicht van links naar rechts
11	cijfers op displays	alles aan

Mode 00

Wanneer “Mode” 00 is, dan moet op alle displays tegelijk dezelfde snake te zien zijn, volgens de tabel van in opgave 1. Nu bevat de Count ingang echter 4 bits in plaats van 3, dus nu kunnen er waarden groter dan 7 op de ingang komen. Als dit het geval is, dan mogen alleen de linkse vier displays een snake vertonen, terwijl de rechtse vier displays uit moeten blijven. Hoewel Count nu uit 4 bits bestaat en de Count ingang van opgave 1 slechts uit 3, kan je opgave 1 toch in zijn geheel als deelschakeling gebruiken. Tip: gebruik de MSB van Count om te detecteren wanneer Count groter is dan 7. Als je de handleiding van het FPGA bord gelezen hebt (zoals hierboven gevraagd), dan zou je moeten weten hoe je juist de anodes kan gebruiken om volledige 7-segment displays aan of uit te zetten.

Terwijl dit allemaal gebeurt op de 7-segment displays, moeten ondertussen alle 16 leds uit staan, onafhankelijk van de waarde op Count.

Mode 01

Voor een “Mode” gelijk aan 01, moeten alle 7-segment displays uit staan, onafhankelijk van de waarde op Count.

Ondertussen mag er slechts 1 ledje branden, afhankelijk van de waarde van Count. Als Count gelijk is aan 0, dan moet led 0 branden (de meest rechtse). Als Count gelijk is aan 1, dan moet led 1 branden (de tweede van rechts), enz… Als we er dus vanuit gaan dat een gebruiker achtereenvolgens alle waarden van 0 tot en met 15 aanlegt op Count, dan zou er dus een looplicht moeten verschijnen van rechts naar links. Voorlopig doen we dit dus nog manueel met de vier linkse dip switches. In de laatste opgave zullen we zelf deze waarden op Count genereren, zodat we dat niet meer manueel moeten doen.

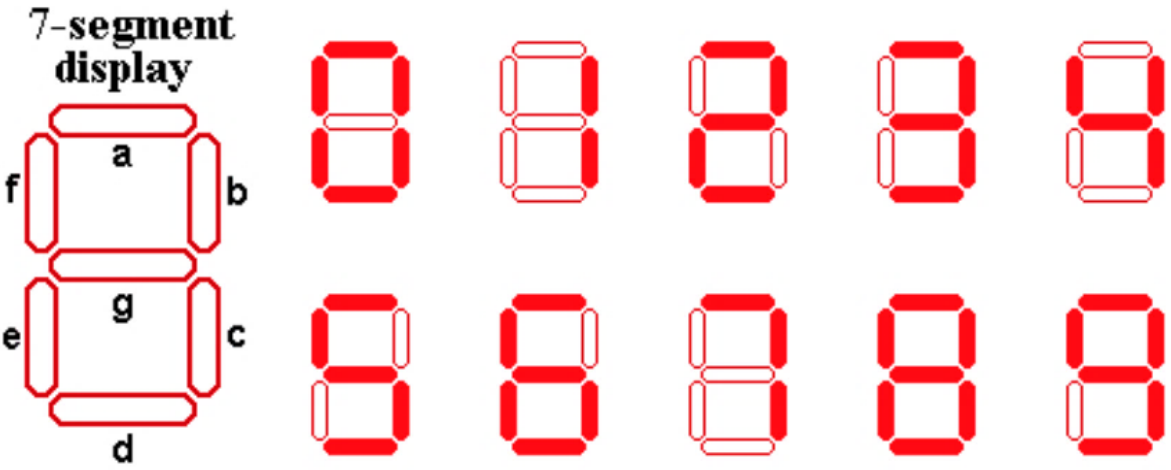
Mode 10

In deze mode moet er terug een snake verschijnen op de 7-segment displays, maar ditmaal telkens op slechts één display tegelijk. Als Count 0 is, dan verschijnt de snake op de meest linkse display, bij Count gelijk aan 1 op de display rechts ervan, bij Count gelijk aan 2 op derde display van links, enz. Vanaf waarde 8 op Count gebeurt net hetzelfde, maar dan van rechts naar links (Count=8: meest rechtse display, Count=7: display links ervan, enz). Dus er moet dezelfde waarde naar de kathodes gestuurd worden voor Count gelijk aan 0 en 8, Count gelijk aan 1 en 9, 2 en 10, enz. Enkel de anodes zullen verschillen.

Ondertussen gebeurt er op de leds net hetzelfde als bij mode 01, maar dan van links naar rechts. Als Count gelijk is aan 0, moet de meest linkse led oplichten en bij Count gelijk aan 15 de meest rechtse, zodat er een looplicht van links naar rechts zou verschijnen, wanneer een gebruiker achtereenvolgens de waarden 0 tot en met 15 op Count zou zetten.

Mode 11

In deze laatste mode moeten de overeenkomstige cijfers verschijnen op alle 7-segment displays tegelijk. Dus als Count gelijk is aan 0, moet het cijfer 0 op alle displays tegelijk verschijnen, Count gelijk aan 1 geeft cijfer 1 op alle displays, enz. Voor Count waarden groter dan 9 moeten alle displays volledig uit blijven. Hieronder kan je ter referentie alle cijfers vinden die afgebeeld moeten worden, naargelang de waarde op Count.



Ondertussen moeten alle leds aan blijven, ongeacht de waarde op Count.

Nadat je de correcte werking van je schakeling gecontroleerd hebt in simulatie, configureer je de FPGA en test je nogmaals je schakeling, maar ditmaal op het bordje zelf. Leg de 16 verschillende binaire waarden aan op de vier linkse dip switches en kijk of je 7-segment displays én leds zich correct gedragen volgens de mode die je hebt ingesteld op de twee rechtse dip switches.

Opmerkingen/tips

Vermits je nog geen geheugenelementen gezien hebt, is deze opgave opgesteld om volledig te maken met combinatorische processen en concurrent statements (die "achter de schermen" door Vivado ook vervangen worden door combinatorische processen). Zorg ervoor dat je de regels van een combinatorisch proces respecteert! Kijk zeker na of je geen latches hebt in je ontwerp. Bekijk de videoles en/of cursus VHDL over combinatorische processen voor meer informatie. Begrijp alle warnings die Vivado je geeft in het Messages venster. Dan kan je de latches niet missen!

Het correct schrijven van combinatorische processen in VHDL is een belangrijke competentie voor dit vak! Zorg ervoor dat je dit volledig onder de knie hebt. Indien dit niet het geval is, dan vraag je het aan mij!

Voorzie ook een testbench waarmee je de correcte werking van je schakeling kan nakijken. De testbench moet je niet mee afgeven, maar helpt je wel om fouten op te sporen tijdens je ontwerp. Vergeet in je simulatie niet de inwendige signalen toe te voegen aan je waveforms. Dat is net de kracht van een simulatie, anders zie je enkel de toplevel signalen en die kan je net zo goed nakijken als je ontwerp op het bordje geconfigureerd staat. Bekijk zeker de videoles over simuleren in Vivado.

Hiërarchie in VHDL is een belangrijk concept dat je moet begrijpen. Als dit niet het geval is, dan vraag je het aan mij!

Vermits je meerdere vhdl bestanden gebruikt (met hiërarchie), vergeet dan ook niet dat je ze allemaal moet indienen!

Beschrijving: implementatie in de FPGA

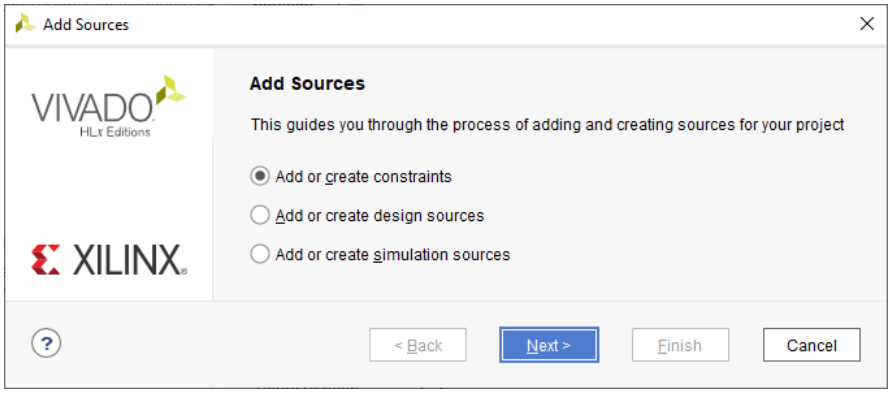
Ondertussen heb je gezien hoe je een digitale schakeling kan beschrijven en simuleren in VHDL. Om je schakeling nu effectief in de Artix 7 FPGA op het Digilent Nexys bord te krijgen, heeft Vivado nog wat extra informatie van je nodig. Je kan heel de onderstaande uitleg ook als demo bekijken in de laatste hoofdstukken van de videoles Vivado design flow.

In het blokschema kan je bijvoorbeeld zien dat je ontwerp 6 ingangen heeft, die op het bord verbonden zouden moeten worden met dip switches (de 4 meest linkse met Count en de 2 meest rechtse met Mode). Xilinx kan onmogelijk voor alle mogelijke fabrikanten van FPGA borden weten hoe zij hun FPGA's verbinden met de andere componenten op het bord. Hiervoor moet de fabrikant van het bord (Digilent in dit geval) een zogenaamd .xdc bestand voorzien. Op de website van Digilent kan je deze downloaden voor het betreffende bordje (Master XDC Files). Deze staat in een Github opgeslagen, wat het downloaden voor sommigen misschien wat bemoeilijkt. Als je op de link klikt van het betreffende bestand (Nexys A7 100T), dan krijg je de inhoud van het bestand te zien. Om dit te downloaden, kan je met de rechtermuisknop klikken op de "Raw" knop en de "link opslaan als..." Een andere manier is manueel alle tekst selecteren en kopiëren naar een leeg .xdc bestand op je computer.[1]

[1] Stel je Windows best in om bestandsextensies weer te geven, zodat je geen "blabla.xdc.txt" bestand maakt of dergelijke. Hier staat een uitleg hoe je dat kan doen.

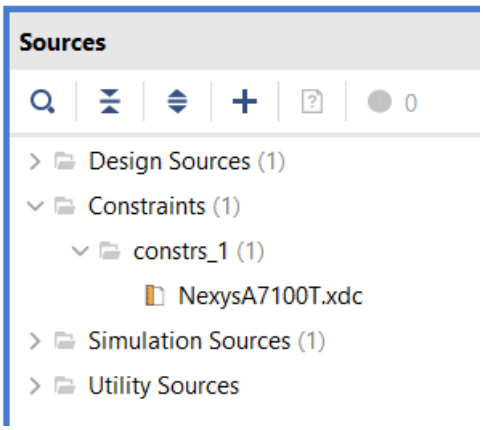
Als dat gelukt is, dan voeg je het .xdc bestand nog toe aan je project door opnieuw op het plusteken te klikken in je “Sources” venster, zoals in vorige opgaven.

Ditmaal kies je de optie die je nog niet hebt gehad, namelijk “Add or create constraints...”.



In plaats van “Create File” kies je vervolgens “Add Files” om het .xdc bestand aan je project toe te voegen. Als je “Copy constraints files into project” aanvinkt, voor je op “Finish” klikt, dan maakt Vivado een kopie aan van het origineel en bewaart het ergens onder een van je projectmappen. Let dus op dat je verderop de juiste kopie bewerkt en niet het origineel.

Het .xdc bestand kan je nu terugvinden onder “Constraints” in het “Sources” venster:



Nadat je het .xdc bestand hebt toegevoegd aan je project, dan moet je het nog bewerken, vermits alle lijnen nog als commentaar gelabeld staan. Dubbelklik op het bestand in het “Sources” venster om het te bewerken. Verwijder het commentaarteken (#) aan het begin van de lijnen die je nodig hebt. Voor de dip switches zal je onder andere de volgende lijnen moeten bewerken:

```
##Switches
#set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { SW[0]
}]; #IO_L24N_T3_RS0_15 Sch=sw[0]
#set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { SW[1]
}]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
...
```

De rode commentaartekens hierboven zal je dus moeten verwijderen. Merk ook op dat hiermee de VHDL poort “SW” (bit 0) verbonden wordt met pin J15 van de FPGA en dus ook (door Digilent) met dip switch SW0 op het bord. Stel nu dat je VHDL poort niet “SW” heet, maar bijvoorbeeld “Mode”, dan zal je dit dus ook moeten aanpassen in je XDC bestand: “SW[0]” (de meest rechtse dip switch) moet dan “Mode[0]” worden, terwijl “SW[15]” (de meest linkse dip switch) in onze schakeling verbonden moet worden met “Count[3]”.

VHDL is niet hoofdlettergevoelig, maar XDC wel! Let hierop als je je VHDL poorten laat overeenstemmen met je XDC poorten.

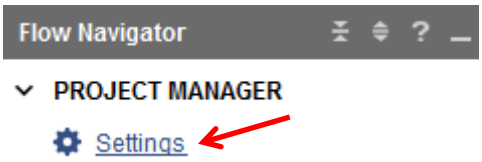
Merk op dat er op iedere lijn nog een tweede commentaarteken (#) staat met daarachter nog wat informatie over de FPGA pin. Deze mag je ter info laten staan, maar dat hoeft niet.

Dit moet je dus doen voor al de in- en uitgangen van je toplevel (zonder testbench) VHDL-bestand.

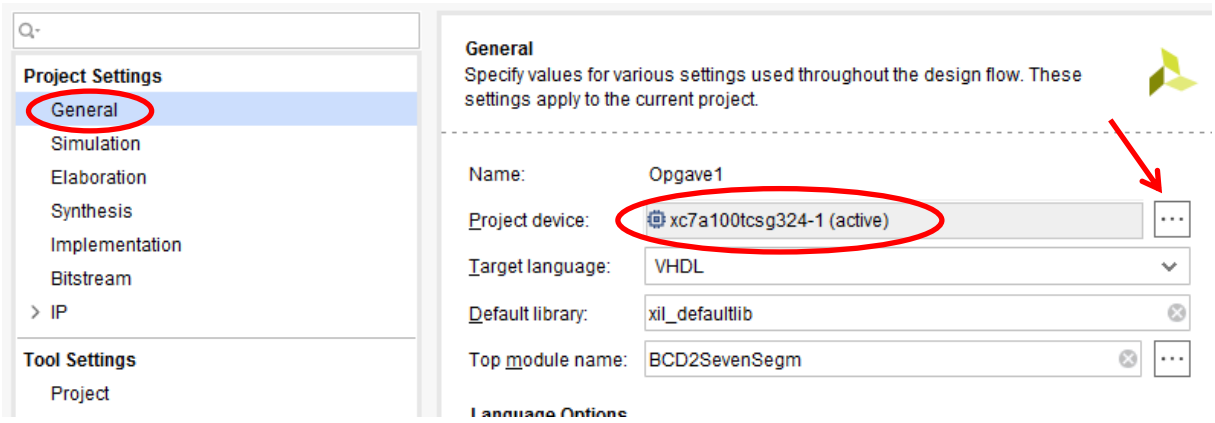
Tot slot zie je in bovenstaand voorbeeld hoe je een "array"-poort (std_logic_vector, unsigned, signed,...) moet verbinden in XDC, namelijk met vierkante haakjes. Mode is in het voorbeeld hierboven in VHDL waarschijnlijk een std_logic_vector(1 downto 0), waarvan iedere bit uiteraard afzonderlijk verbonden moet worden met een pin op de FPGA. Als je een VHDL in- of uitgang hebt die géén array is (bv. std_logic), dan plaats je ook geen vierkante haken, maar rechtstreeks de naam van je in- of uitgang.

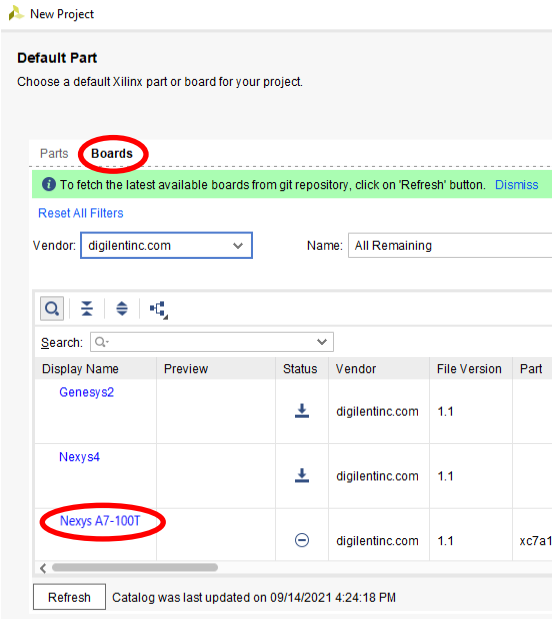
Het systeem van poorten verbinden aan de hand van XDC is een belangrijk concept voor dit vak. Indien je dit niet helemaal begrijpt, dan moet je het vragen aan mij!

Er is nog één ding dat Vivado moet weten vooraleer het je ontwerp in een FPGA kan steken, namelijk het type FPGA dat je zal gebruiken. Dit stel je in bij "Settings" in de "Flow Navigator":



In de "General" sectie kan je achter "Project device" de correcte FPGA kiezen. Voor het Digilent Nexys bord is dit de Artix 7 100T, codenaam xc7a100tcsg324-1. Je kan dit ofwel overtypen ofwel via de "browse" puntjes rechts uit een lijst kiezen.



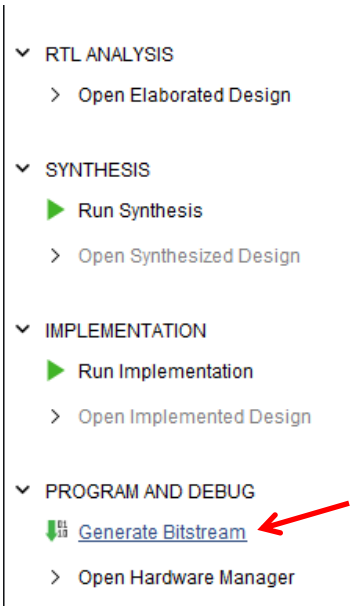


Voor toekomstige projecten kan je deze stap vereenvoudigen door tijdens de “Create New Project” wizard in de laatste stap de vendor “digilentinc.com” te kiezen, wat de lijst van FPGA borden onderaan filtert, zodat je gemakkelijker het gewenste bord “Nexys A7-100T” kan selecteren.

Als je digilentinc.com niet kan selecteren, dan heb je de Digilent board files nog niet geïnstalleerd. Je kan dit rechtstreeks vanuit de wizard doen door onderaan op de knop “Refresh” te klikken. Hierna zou je naar het juiste bord moeten kunnen scrollen en “Install” ( ) moeten kunnen kiezen. Vanaf dan is het bord altijd beschikbaar in de lijst en wijzigt het icoon in . Vanaf dan moet je dus nooit meer de individuele FPGA selecteren, maar kan je ineens het volledige bord Nexys A7 kiezen uit de lijst. Moest je dit tijdens de “Create New Project” wizard vergeten zijn of fout gedaan hebben, dan kan je het achteraf nog steeds wijzigen door de procedure te volgen die beschreven staat op Blackboard onder “Practicum> Informatiebronnen> Xilinx en Digilent> Het Digilent Nexys bord installeren in Vivado”.

Nu alle nodige informatie toegevoegd is aan het project, kan je Vivado je ontwerp laten interpreteren om er een digitale schakeling van te maken en de FPGA hiermee te configureren.

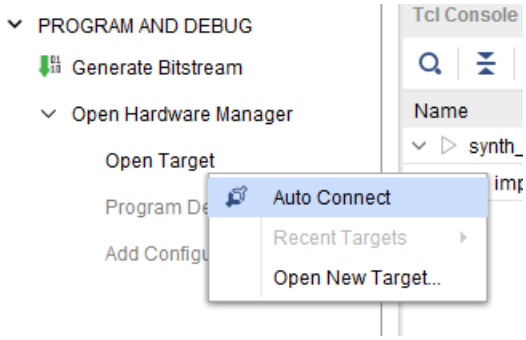
Dit doe je door op “Generate Bitstream” te klikken onder “PROGRAM AND DEBUG” in de “Flow Navigator”. Nu zal Vivado automatisch alle stappen doorlopen om uiteindelijk te komen tot het genereren van een bitstream, die vervolgens gebruikt kan worden om de FPGA te configureren:



- RTL analysis: het interpreteren van je VHDL code en omzetten naar Booleaanse algebra.
- Synthesis: het omzetten van deze Booleaanse algebra naar algemene logische poorten (EN-, OF-,…) zonder rekening te houden met de specifieke logische poorten die in de FPGA zitten. Deze schakeling van logische poorten wordt een “netlist” genoemd.
- Implementation: het gebruik van de specifieke logische poorten in de ingestelde FPGA om de netlist van algemene logische poorten te kunnen implementeren. Zo kan een EN-poort met vier ingangen bijvoorbeeld vervangen worden door meerdere EN-poorten met minder ingangen, indien er geen EN-poort met vier ingangen beschikbaar zou zijn in de FPGA (in praktijk zal dit met lookup-tables gebeuren, maar daarover later meer). Voor deze stap is het dus belangrijk dat de juiste FPGA ingesteld staat voor je huidige project. De “algemene” digitale schakeling wordt als het ware “gemapt” op de specifieke FPGA.
- Tot slot wordt dan een bitstream gegenereerd, wat niet meer is dan een binair bestand met daarin een reeks zeer specifieke codes waarmee de FPGA geconfigureerd kan worden. Dit bestand kan je in je project terugvinden met de extensie “.bit”.

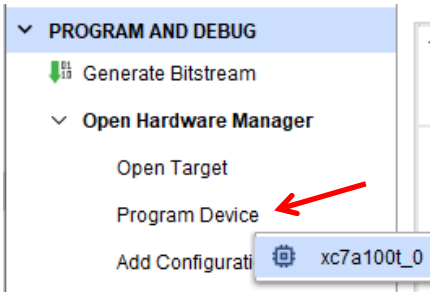
Je zou ook iedere stap afzonderlijk kunnen uitvoeren (elaboratie, synthese, implementatie, bitstream), maar Vivado detecteert normaal gezien automatisch welke stap(pen) er geüpdatet moet(en) worden.

Nadat je een .bit bestand (bitstream) gegenereerd hebt, open je de “Hardware Manager” van Vivado door op “Open Target” te klikken nadat je “Open Hardware Manager” hebt opengevouwen onder “Generate Bitstream”. Vervolgens kies je “Auto Connect”.

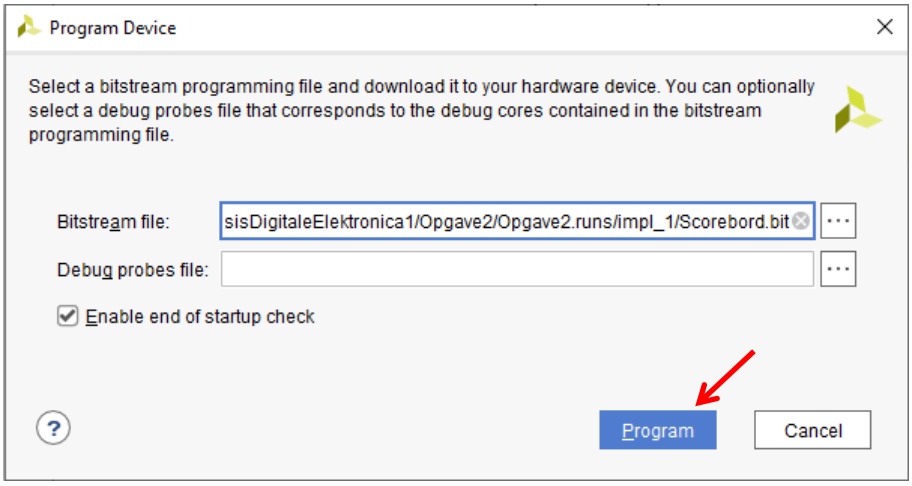


Uiteraard moet je hiervoor eerst het Digilent bordje via USB verbinden met je laptop en niet vergeten om de schakelaar op ON te zetten.

Vervolgens kan je dan op “Program Device” klikken en op de naam van de verbonden FPGA (xc7a100t):



In het venster dat dan opent, zou normaal gezien het bitstream bestand (.bit) al ingevuld moeten zijn. Als dat niet het geval is, dan heb je de bitstream nog niet gegenereerd.



Druk ten slotte op “Program” om de FPGA te configureren. Tijdens het configureren gaat het groene ledje “DONE” uit op het Digilent bord. Als alles goed gaat, zal het ledje terug oplichten wanneer hij klaar is.

Nu kan je dus de schakeling nogmaals testen, maar ditmaal op het bord zelf. Leg de 16 verschillende binaire waarden aan op de vier linkse dip switches en kijk of je 7-segment displays én leds zich correct gedragen volgens de mode die je hebt ingesteld op de twee rechtse dip switches.

Als alles werkt naar behoren en je hebt de correcte werking van je ontwerp nagekeken, dan moet je je opgave indienen in Blackboard. Dit doe je op de volgende manier:

- Je opent Blackboard en gaat naar de cursus van dit vak.
- Onder “Practicum” > “Indienen” staat er een link “Opgave 2”. Klik hierop.
- Dien vervolgens je VHDL bestanden én je XDC bestand in.
- Zowel je .vhd en je .xdc bestanden kan je terugvinden door er in het “Sources” venster op te dubbelklikken. Het bestand zal dan rechts openen en net onder het betreffende tabblad kan je dan de exacte locatie terugvinden.
- Eventueel kan je nog opmerkingen toevoegen in het betreffende tekstvak.
- Klik op “Verzenden” om in te dienen.

Je hebt pas ingediend als je de juiste bestanden vóór de deadline correct hebt ingediend! Als je de verkeerde bestanden hebt ingediend en je merkt dit pas na de deadline, dan kan je alsnog de juiste indienen, maar dan geldt dit niet meer als “tijdig”. Als je dus niet zeker bent, dan vraag je het aan mij.

Merk op dat het niet geldig is om je VHDL of XDC code te kopiëren in een Word, PDF, tekst- of eender welk ander bestand. Ik heb je “.vhd” en “.xdc” bestand nodig en NIETS ANDERS! Maak dus ook geen zip of dergelijke!