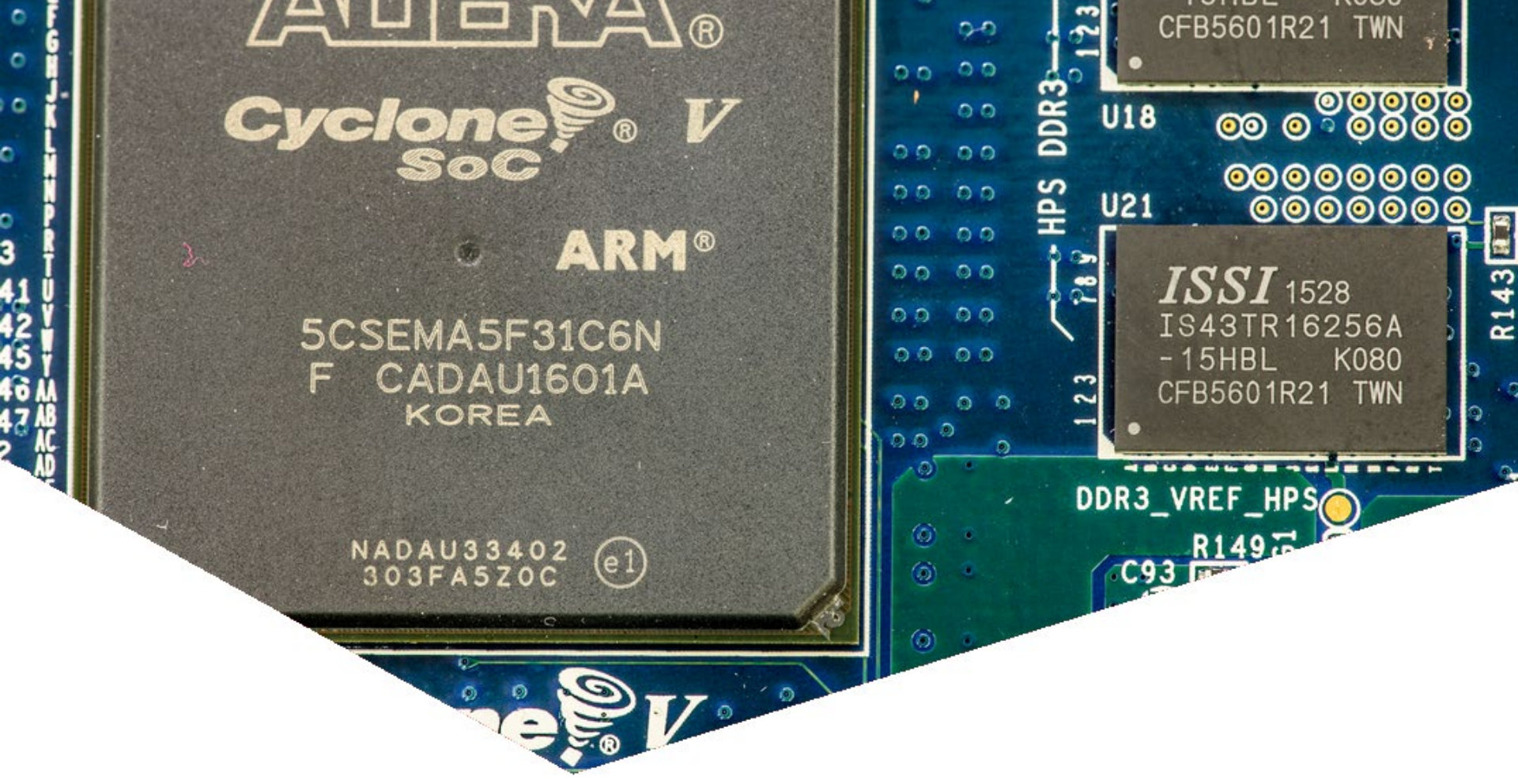


CSC10 Week 5: HLS & OpenCL

Leerdoelen CSC10

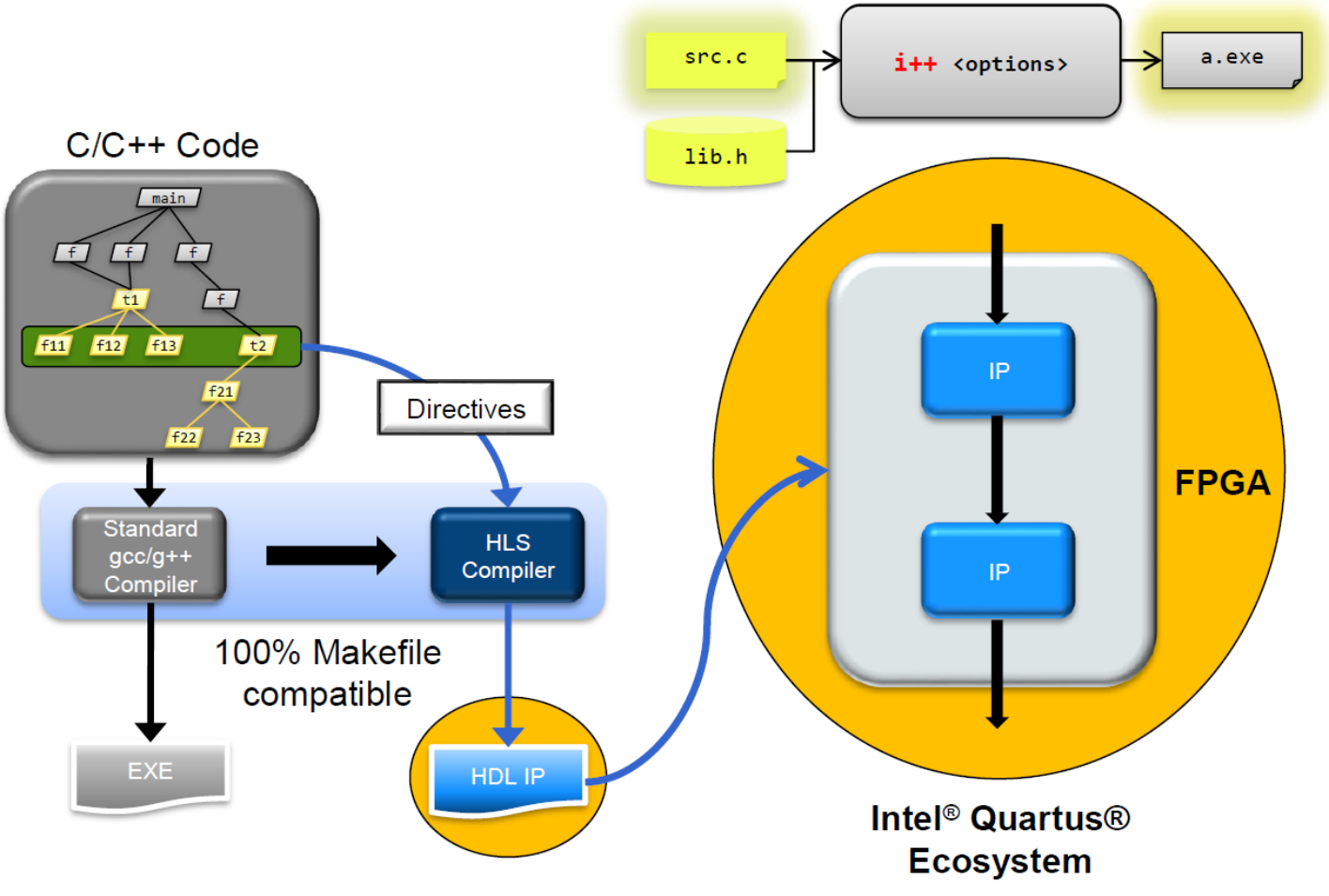
Je bent na het volgen van deze cursus in staat om:

- in VHDL een hardware module te ontwerpen en implementeren met een memory bus interface zodat deze module, vanuit een soft of hard core processor, memory mapped te programmeren is;
- een embedded systeem op een FPGA te ontwerpen en implementeren bestaande uit een soft core, software, bestaande hardware modules en zelf in VHDL geïmplementeerde hardware modules;
- op dit embedded systeem een RTOS toe te passen;
- een embedded systeem op een FPGA te ontwerpen en implementeren bestaande uit een hard core, software die draait onder Linux, bestaande hardware modules en zelf in VHDL geïmplementeerde hardware modules;
- te beslissen of bepaalde functionaliteit van een embedded applicatie beter op een soft core, op een hard core of in hardware geïmplementeerd kan worden;
- verschillende vormen van High Level Syntheses met de voor- en nadelen van deze vormen te benoemen.

High Level Synthesis

- Het genereren van HDL-code uit C-code
- Waarom?
  - Debuggen van software is veel sneller dan hardware
  - Functies zijn eenvoudiger te specificeren in software
- Het gegenereerde component kan worden toegevoegd aan het project via Platform Designer of de top level architecture HDL file.

Design flow



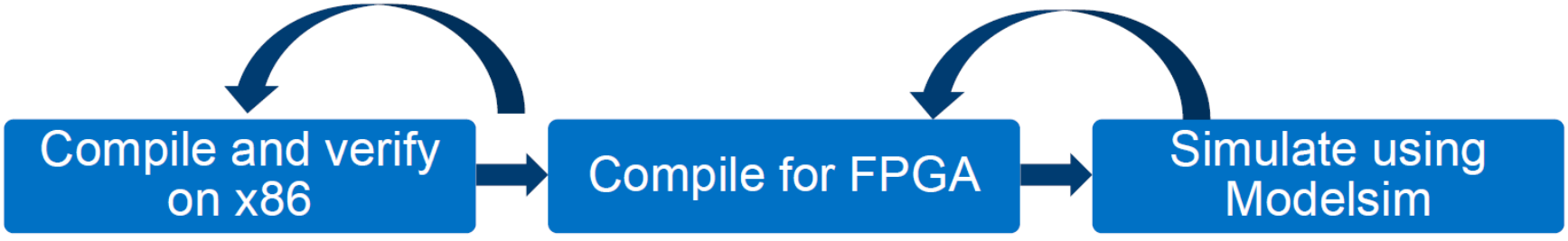
Voorbeeld a+b


```
#include "HLS/hls.h"
#include "assert.h"
#include "HLS/stdio.h"
#include "stdlib.h"

component int accelerate(int a, int b) {
      return a+b;
}

int main() {
      srand(0);
      for (int i=0; i<10; ++i) {
            int x=rand() % 10;
            int y=rand() % 10;
            int z=accelerate(x, y);
            printf("%d + %d = %d\n", x, y, z);
            assert(z == x + y);
      }
      return 0;
}
```

`accelerate()` becomes an FPGA component

`main()` becomes testbench for component `accelerate()`

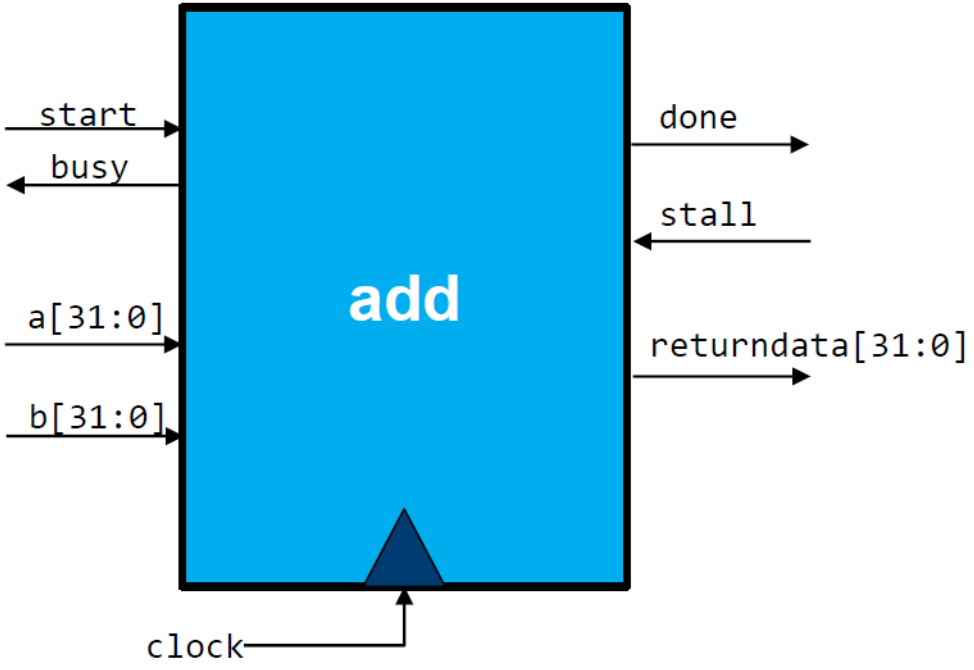
Stappen



Iterate on the algorithm	Examine the FPGA reports	Test functionality
Functional verification	Iterate on the architecture of the design	Test latency and performance (through verification stats)
Debugging using gdb/valgrind	Use the reports as feedback on what the bottlenecks are	

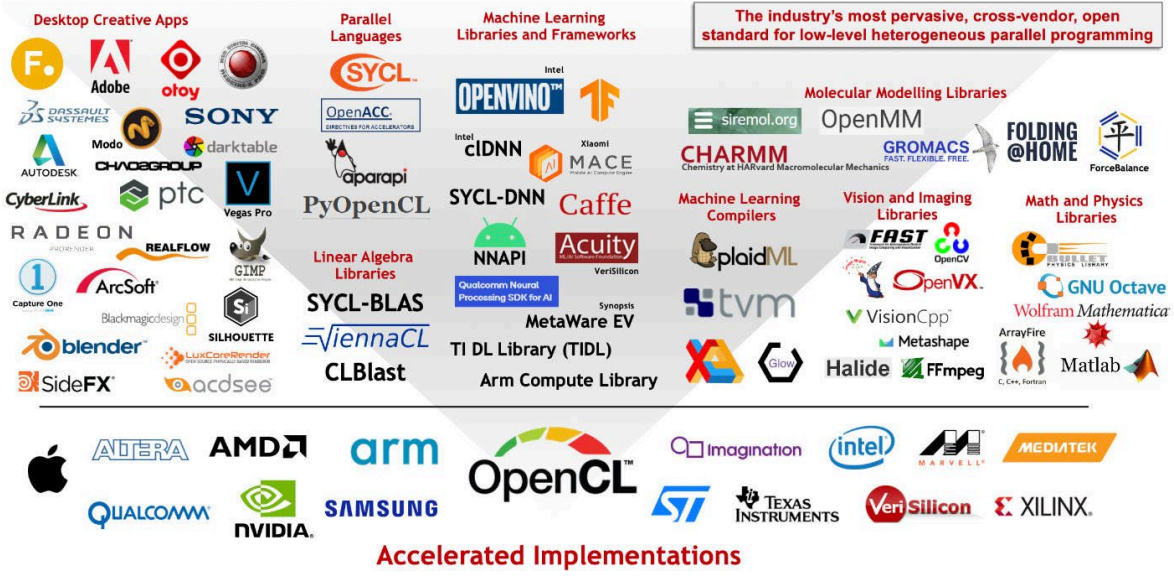
Component symbol

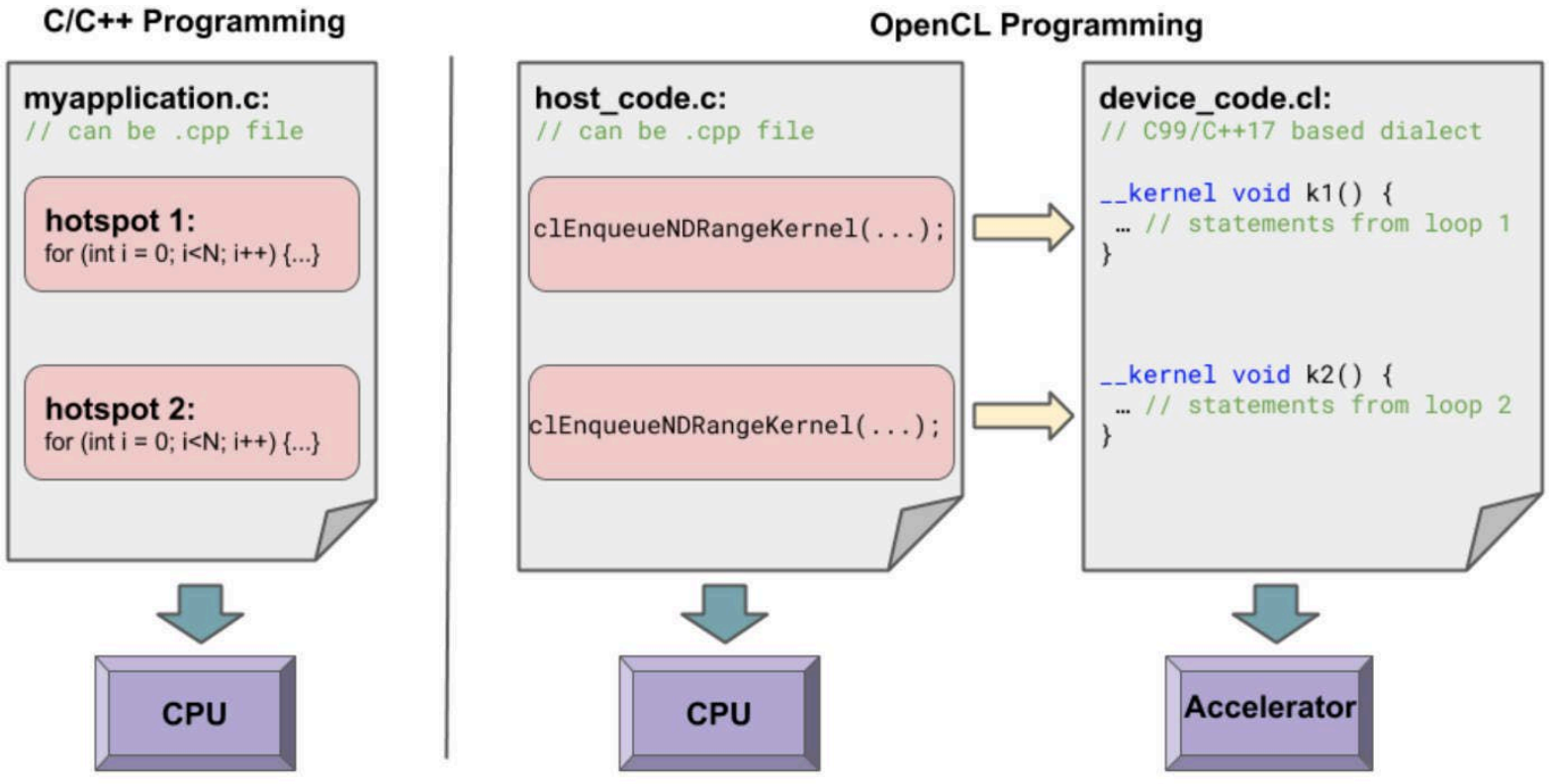

```
component int add(int a, int b) {
    return a+b;
}
```



- Start/done: signaleerd wanneer de data op input/output valide is
- Busy/stall geeft het component de functionaliteit om de pipeline te beïnvloeden.

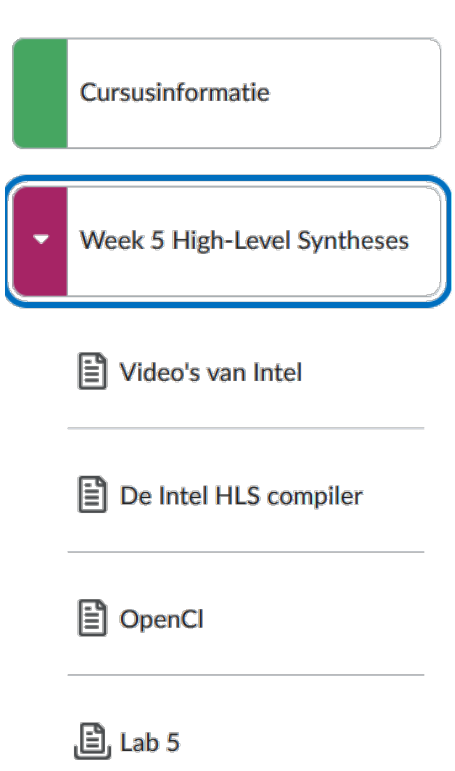
- Software framework zodat code kan worden uitgevoerd op een of meerdere (heterogene) processors.
- Oorsprong: Apple, onderhouden door Khronos Group Inc.




C/C++ Programming
OpenCL Programming
myapplication.c:
// can be .cpp file
hotspot 1:
for (int i = 0; i<N; i++) {...}
hotspot 2:
for (int i = 0; i<N; i++) {...}
CPU
host_code.c:
// can be .cpp file
clEnqueueNDRangeKernel(...);
clEnqueueNDRangeKernel(...);
CPU
device_code.cl:
// C99/C++17 based dialect
__kernel void k1() {
… // statements from loop 1
}
__kernel void k2() {
… // statements from loop 2
}
Accelerator

Aan de slag!

Aan de slag met Week 5 op Brightspace



Week 5 High-Level Syntheses

In deze minor heb je hardware leren ontwerpen met behulp van de hardware description language VHDL. In talen zoals VHDL en Verilog wordt hardware beschreven op het zogenoemde Register Transfer Level (RTL) abstractieniveau (zie eventueel: https://en.wikipedia.org/wiki/Register-transfer_level). Dit is een relatief laag abstractieniveau vergeleken met de abstractieniveaus die bij het schrijven van software worden gebruikt. Het ontwerpen van hardware m.b.v. VHDL of Verilog is daarom erg arbeidsintensief vergeleken met het ontwerpen van software. Ook de verificatie van de ontworpen hardware op RTL-niveau (bijvoorbeeld met behulp van ModelSim) neemt, zoals je hebt gemerkt veel tijd in beslag. Voor het ontwerpen van hardware op RTL-niveau is specifieke, gespecialiseerde kennis nodig waar niet veel ingenieurs over beschikken. Om die reden heeft men al lang de wens om C, C++, of MATLAB code automatisch om te kunnen zetten naar hardware. Dit wordt High-Level Synthesis genoemd (ie eventueel: https://en.wikipedia.org/wiki/High-level_synthesis). De voordelen van HLS t.o.v. RTL zijn:

Aan de slag!

Aan de slag met [Week 5 op Brightspace]