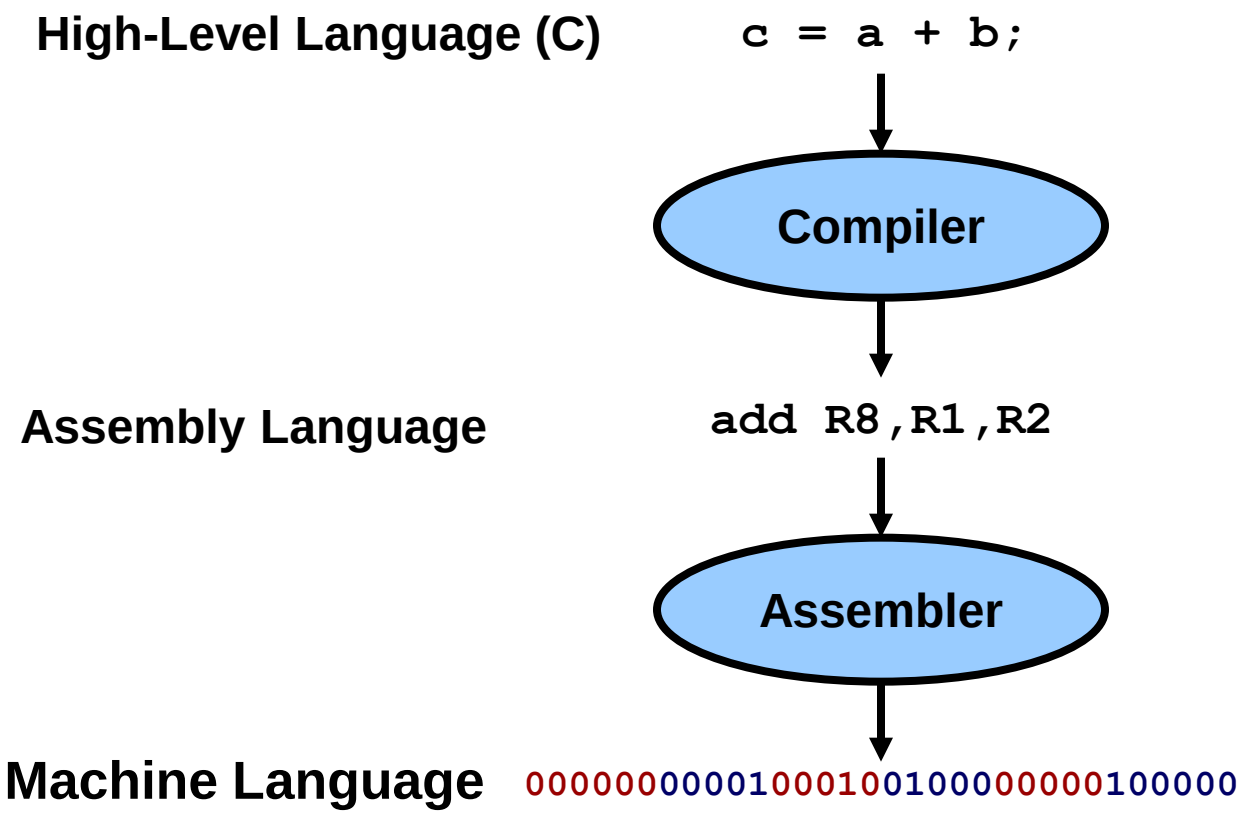
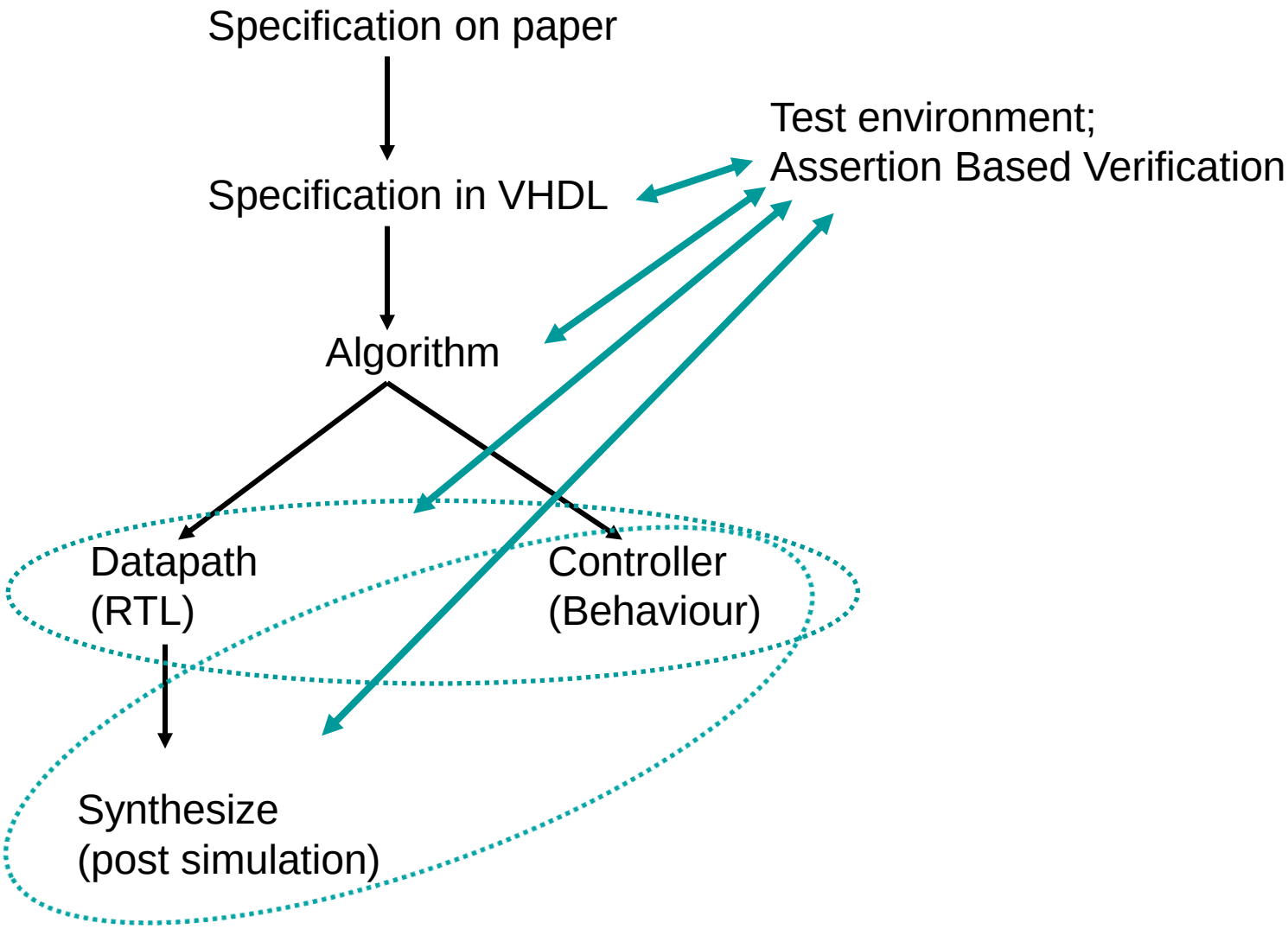
Design a simple MIPS
and the requirements for your project

Background

Assumption: you are familiar with

- Assembly
- Computer organization
- (and of course with VHDL)


High-Level Language (C)
c = a + b;
Compiler
Assembly Language
add R8,R1,R2
Assembler
Machine Language
00000000001000100100000000100000


Specification on paper
Specification in VHDL
Algorithm
Test environment;
Assertion Based Verification
Datapath
(RTL)
Controller
(Behaviour)
Synthesize
(post simulation)

Specification MIPS

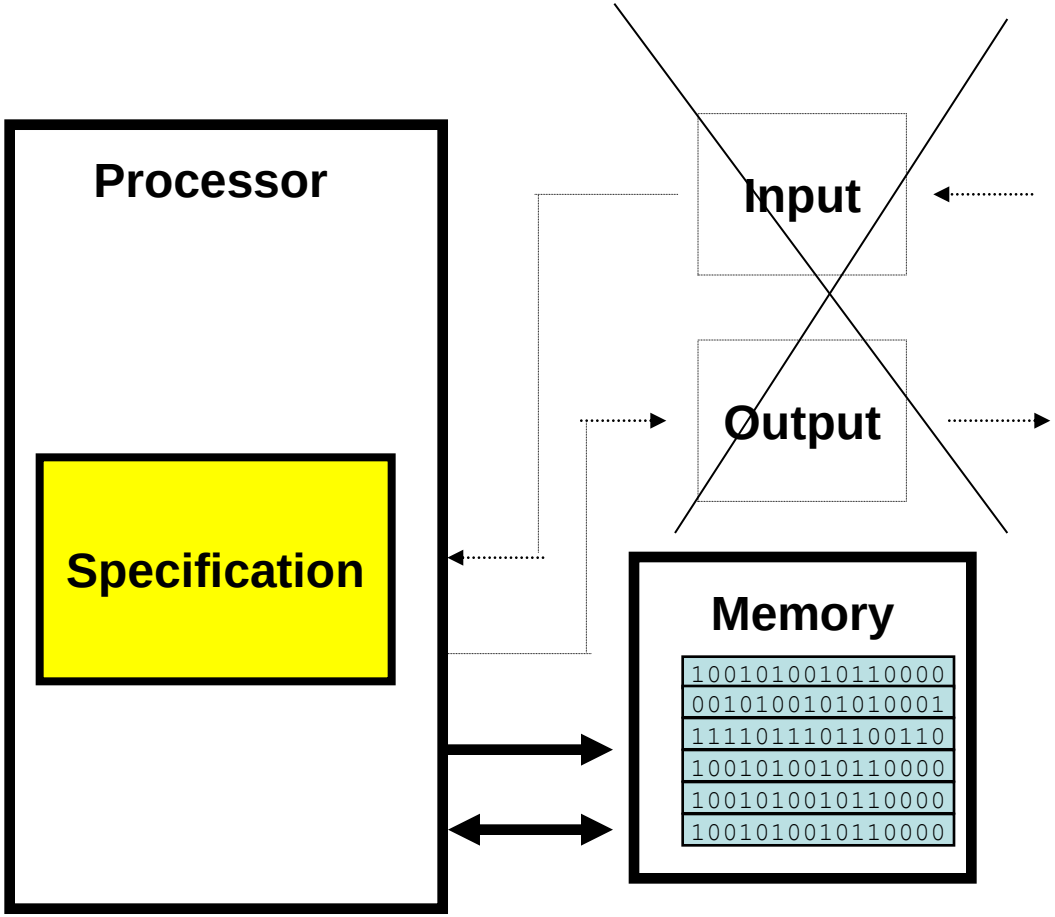
- Instruction set MIPS, e.g.
  - Arithmetic and Logical Instructions
  - Comparison Instructions
  - Conditional Branch Instructions
  - Data Transfer Instructions
  - Jump Instructions
  - Multiplication and Division instruction
- A subset (~10 instructions) for the MIPS is to be designed. The design should be optimized for the subset.
- Since we have no keyboard and no display be sure that data is in memory and the result is also stored in memory.
- See Blackboard for a location of the “paper specification” of the MIPS.

Keep it simple

- No pipeline
  - Not all instructions need to have the same number of clock cycles!
  - Your MIPS should not be too large (area) and not power hungry. Take this into account when you make design decisions.
- No interrupts
- No stack operations
- It should be a synchronous system; only one clock; no gated clocks;
- No bit-level adder algorithms! Use a behavioral description for the adder.
- Use a datapath width of 32 bits (including 32 bits operations if required).
- Use uni-directional databus (I/O) in stead of the bi-directional databus.

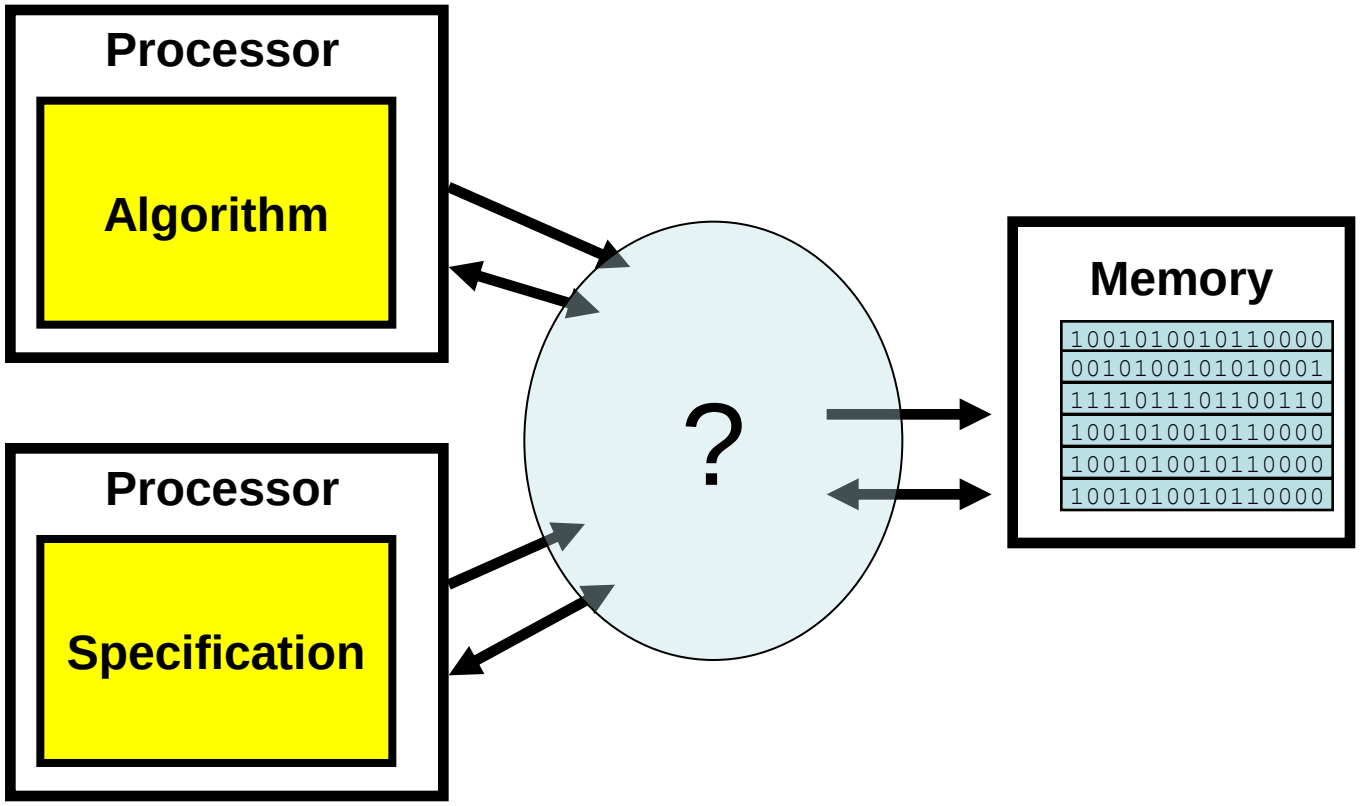
Specification in VHDL

- Write a specification in VHDL for your subset MIPS processor
  - Include a structural description with a memory that includes your test program (I'm not very good in interpreting assembly, so I expect an explanation of the test patterns in the report).
  - Give a behavioral description of (part of) the memory
    - Be careful .. If the whole memory is modeled in VHDL your design will simulate very slow (if loaded at all)!
- Test environment
  - Use PSL in relevant parts
  - Include (and motivate) a test program that test each instruction for design faults!
- An example of a behavioral description of a funny processor is found on: www.cs.utwente.nl/~molenkam/ods/spec_funny_processor/


Processor
Specification
Input
Output
Memory
1001010010110000
0010100101010001
1111011101100110
1001010010110000
1001010010110000
1001010010110000

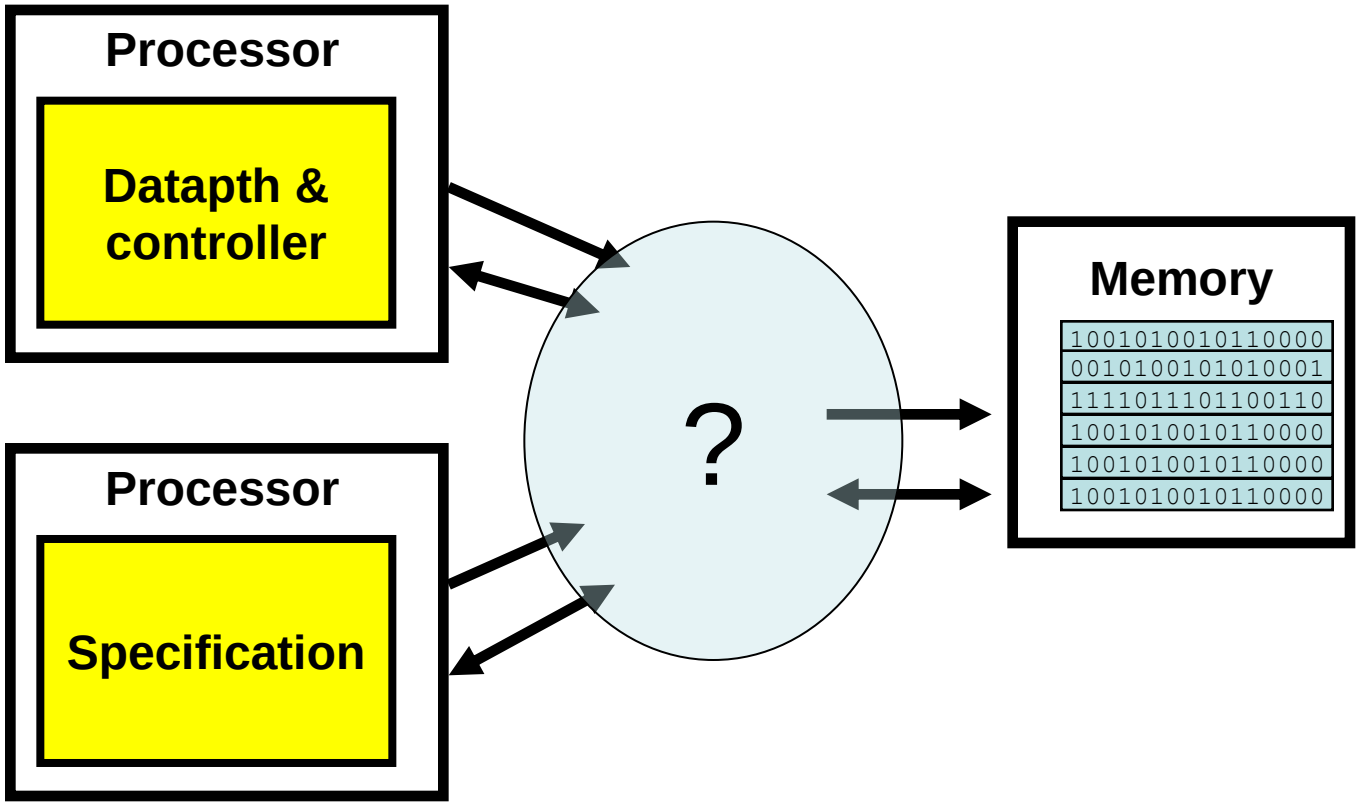
Algorithm

- For a complete design flow of a very simple ALU go to: http://wwwhome.cs.utwente.nl/~molenkam/ods/simple_ALU/
- Give algorithms for instructions (for some instructions it is not really possible, but for others it is).
- You may use behavioral description of an adder; i.e. a+b (don't give algorithms for it like ripple adder, cla adder, ..)
- You have to test your design with the algorithm included.


Processor
Algorithm
Processor
Specification
?
Memory
1001010010110000
0010100101010001
1111011101100110
1001010010110000
1001010010110000
1001010010110000

Datapath + Controller

- For a complete design flow of a very simple ALUgo to: http://wwwhome.cs.utwente.nl/~molenkam/ods/simple_ALU/
- The previous complete design also shows how the datapath and controller are written in VHDL
- And again .. test your design


Processor
Datapth & controller
Processor
Specification
?
Memory
1001010010110000
0010100101010001
1111011101100110
1001010010110000
1001010010110000
1001010010110000

Synthesize the datapath & post simulation

- The datapath description is at RTL (Register Transfer Level)
- The datapath description should be synthesizable. Synthesize it with *QuartusII*.
- The behavioral description of the controller is probably not synthesizable.
- A simulation of the processor with the behaviour of the controller and the synthesized datapath is possible.

  Examine how the type *control_bus* is realized. It is tool dependent (and version of the tooling) how this is done. Find a clever way to deal with this.

  (Note: this is also the reason way at the top level of a design often only the types *std_logic* and *std_logic_vector* are used.)

Final Report

- Should include a CD (or other media) with your design steps and a description how I can reproduce your test of the (I prefer a script file that compiles the design) and configurations to test:
  - specification in VHDL
  - algorithm
  - datapath + controller
- Report should include the design decisions, and short motivation
- Motivation of the test set
- Results of the realized datapath with Quartus II. Perform also a post simulation of your processor (probably you have to make changes due to the type control_bus).
- Motivation of how PSL is used
- Include also the (non) solved problems you had/have
- Report in Dutch or English
- Start writing your report as soon as possible