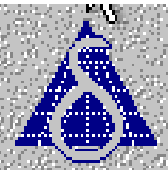


Lesson 1

Introduction

This lesson will introduce VHDL and synthesis, and explain how they can be used together to design hardware. You will first learn what VHDL is used for, then learn how synthesis is used to automate gate level design.

What is VHDL?

VHDL stands for the VHSIC Hardware Description Language. VHSIC is an abbreviation for Very High Speed Integrated Circuit, a project sponsored by the US Government and Air Force begun in 1980 to advance techniques for designing VLSI silicon chips.

VHDL was developed in order to provide a standard format for handling large volumes of design data.

VHDL

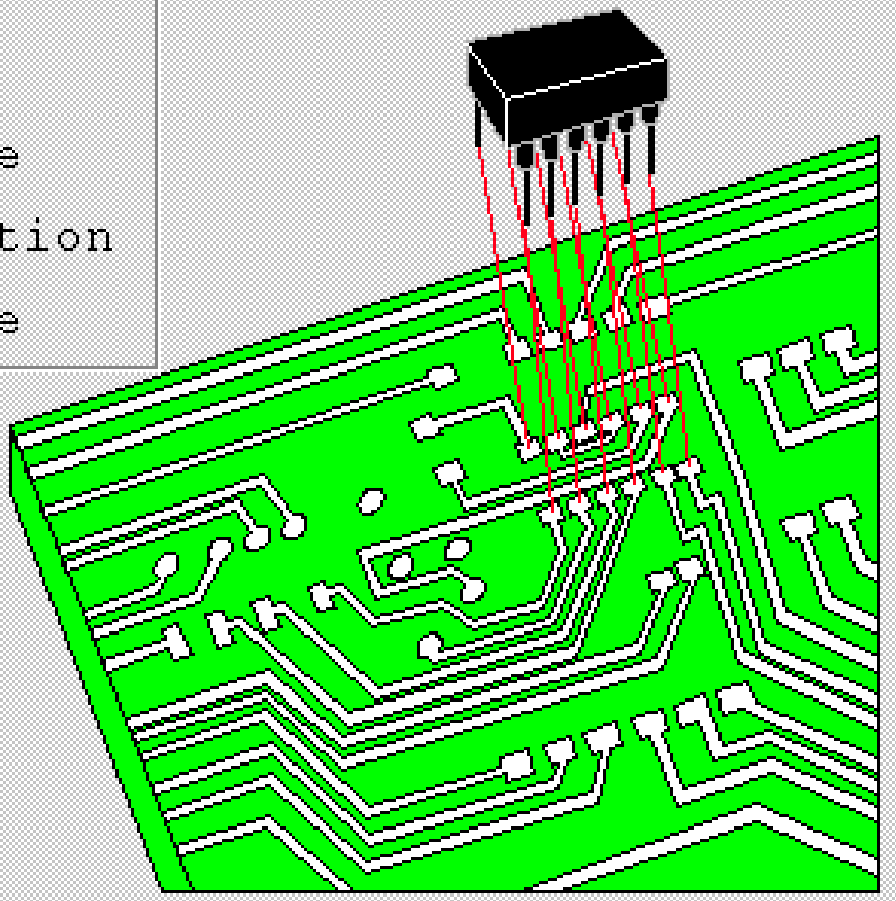
the

VHSIC

Hardware

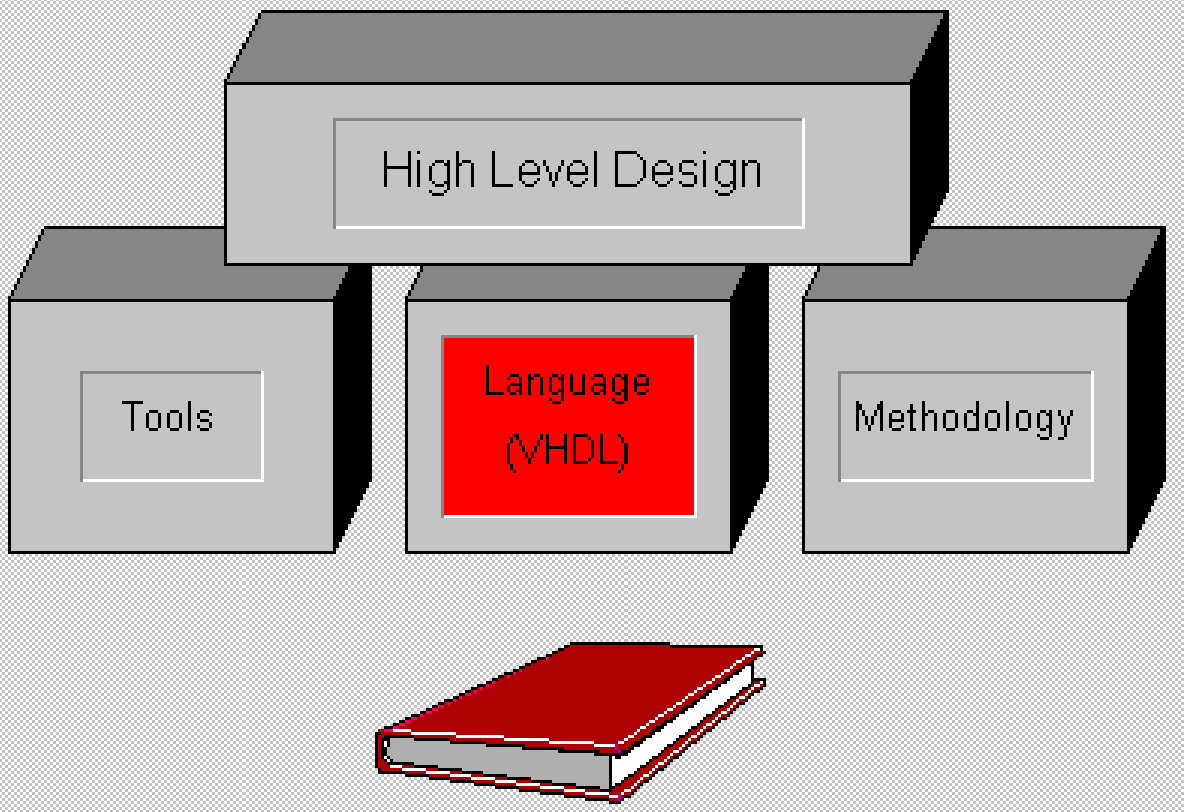
Description

Language



What is VHDL?

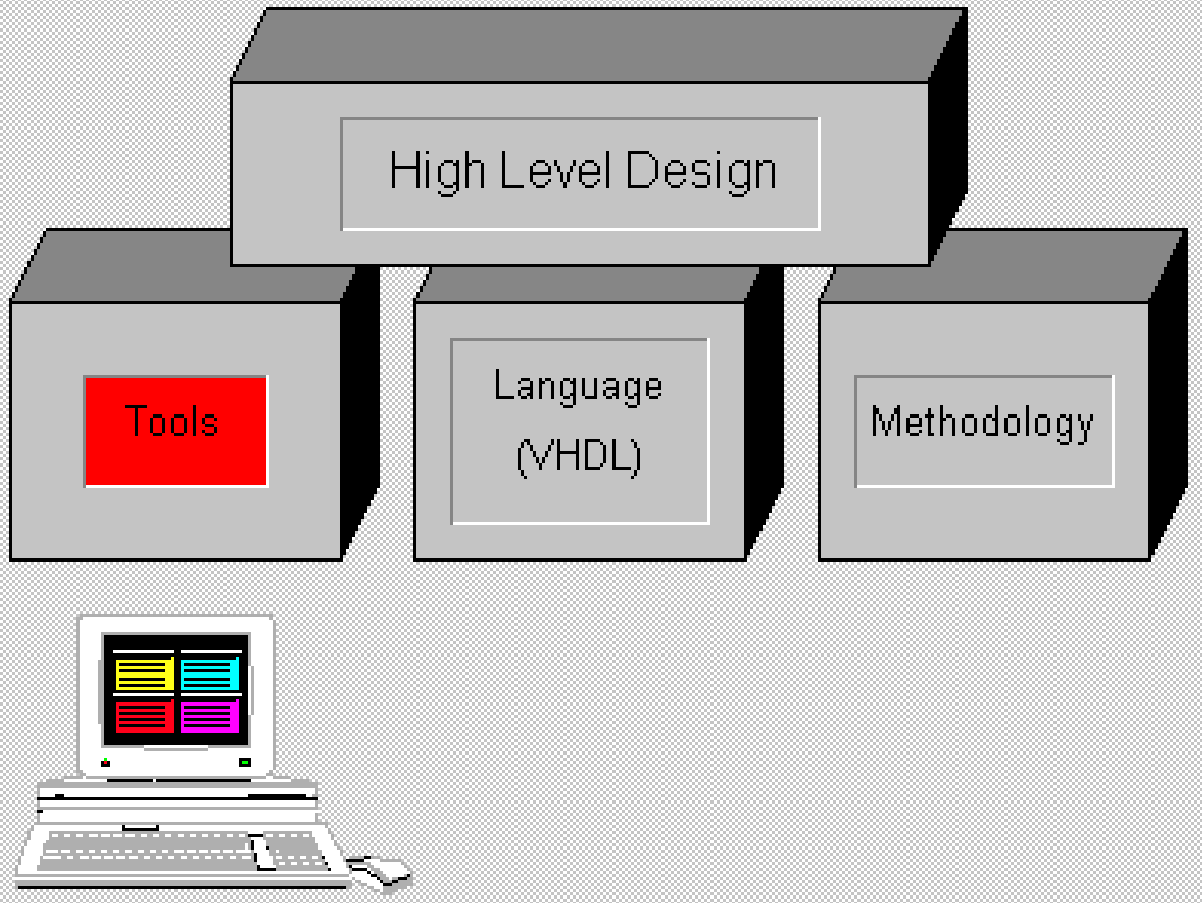
VHDL is a language used to describe the structure and behaviour of electronic systems. VHDL is a notation, and is precisely and completely defined by the Language Reference Manual (or LRM). VHDL is an IEEE standard and is non-proprietary. This sets VHDL apart from other hardware description languages, which are to some extent defined by the behaviour of the tools that use them.



What is VHDL?

Design tools are an essential requirement for high level design, but VHDL is not a toolset. The Language Reference Manual does not define a simulator, but unambiguously defines what each simulator must do with each part of the language.

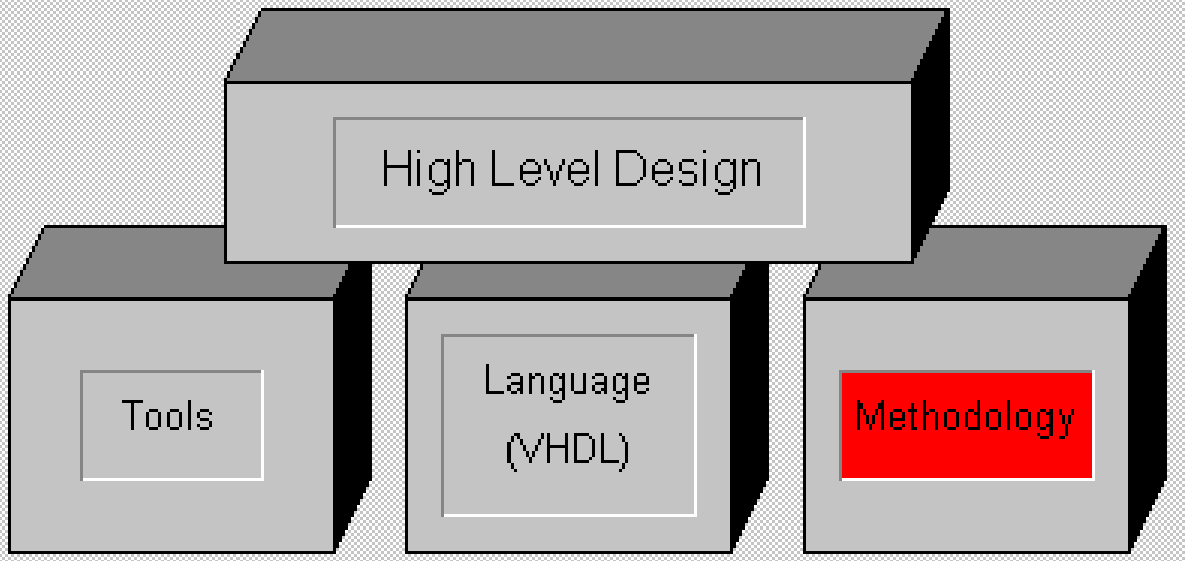
Simulation and synthesis are the two main kinds of tools which operate on the VHDL language.

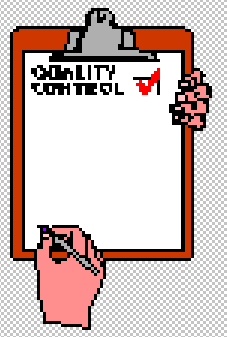


What is VHDL?

VHDL is not an information model or a methodology. VHDL does not constrain the user to one style of description. VHDL allows designs to be described top down, bottom up or middle out!

VHDL can be used to describe hardware at the gate level or in a more abstract way. Successful high level design requires a language, a tool set and a suitable methodology.

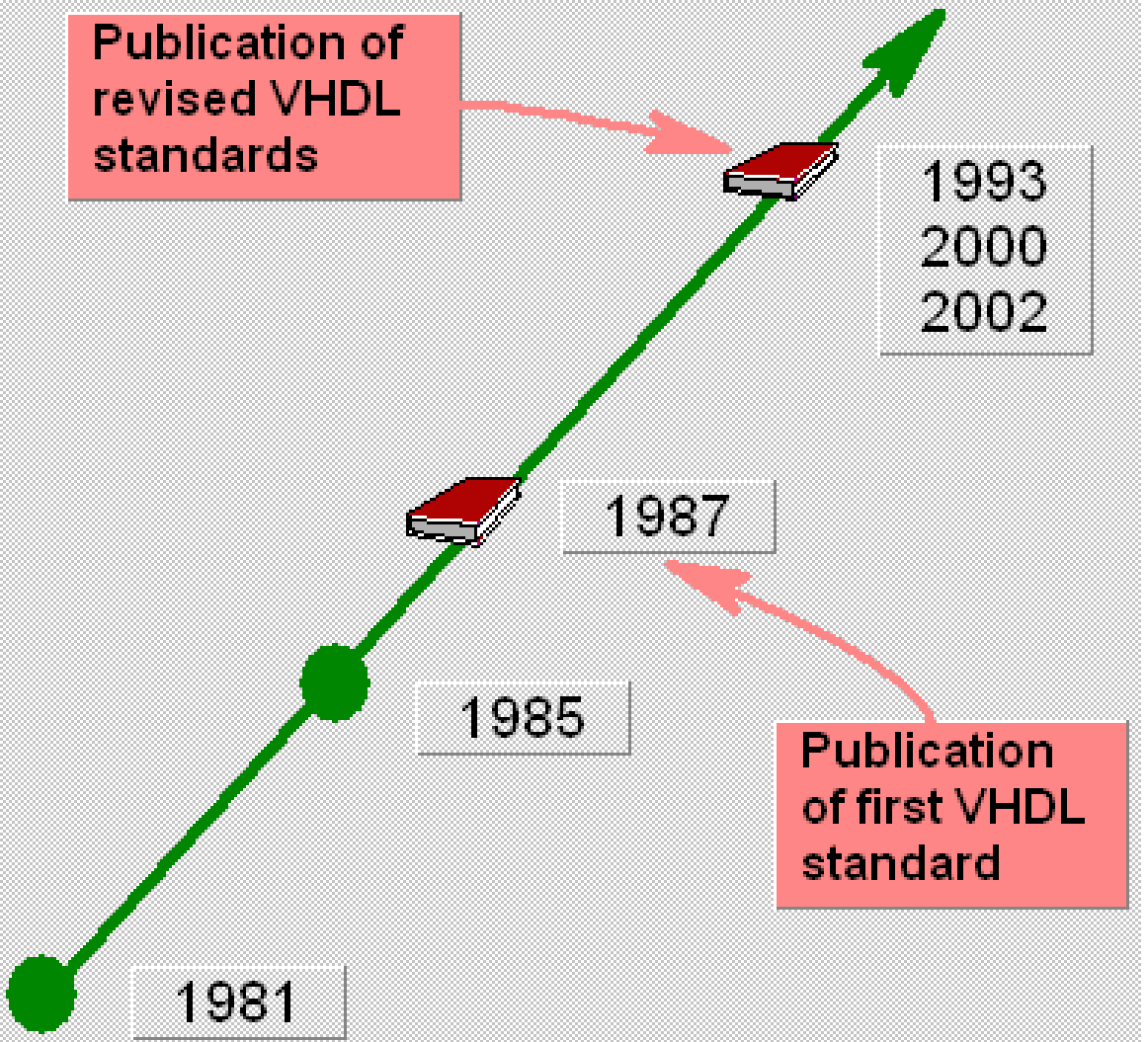




History of VHDL

The development of VHDL was begun by the US Department of Defense in 1981. A baseline language was published in 1985 so that tool development could begin in advance of the standard.

All rights to the language definition were given away by the DoD to the IEEE in order to encourage industry acceptance and investment.



History of VHDL

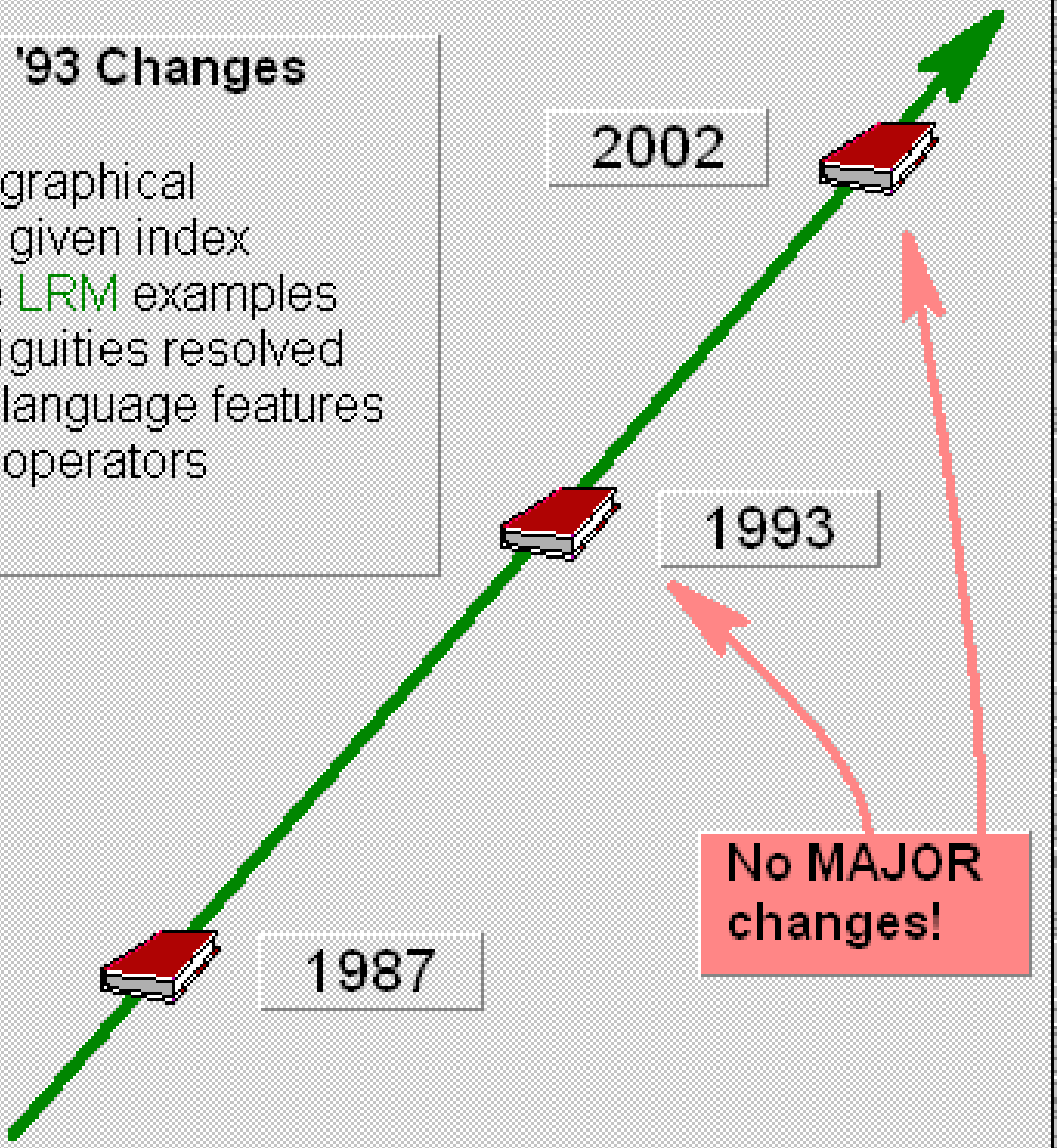
As an IEEE standard, VHDL must undergo a review process every five years to ensure its ongoing relevance to the industry.

A decision was taken that the revision process would not radically change the language for the '93 version.

So, errors and ambiguities in the original LRM were corrected, and some new features were added and simplified.

VHDL '93 Changes

* Typographical
* LRM given index
* More LRM examples
* Ambiguities resolved
* New language features
* New operators
  ...



History of VHDL

VHDL PaceMaker covers both VHDL '87 and VHDL '93. The differences between the two languages are highlighted where appropriate. As a general rule, VHDL '93 will make your life easier!

The example opposite is taken from the lesson on Design Entities and shows how VHDL '93 syntax is indicated in VHDL PaceMaker.

```
G1: entity WORK.INV(ARCH)
   port map (SEL, SELB);

G2: entity WORK.AOI(V2)
   port map (SEL, A, SELB, B, F);
```

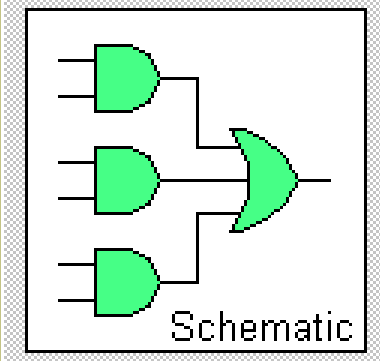
VHDL '93 onwards!

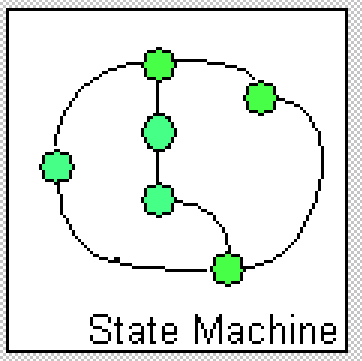
Implications of using VHDL

Using VHDL for hardware design means designing in a programming language.

This has implications for the design process, the tools, the methods used and skills to be learned.

VHDL can directly replace the schematics used in traditional FPGA design. But it can also represent the function of those components.






```
-- State Registers
R1:FF port map(clk,...);
R2:FF port map(clk,...);
R3:FF port map(clk,...);
-- Output Decoding
G1:ND2 port map(s,...);
```

VHDL Structure

Traditional low level design

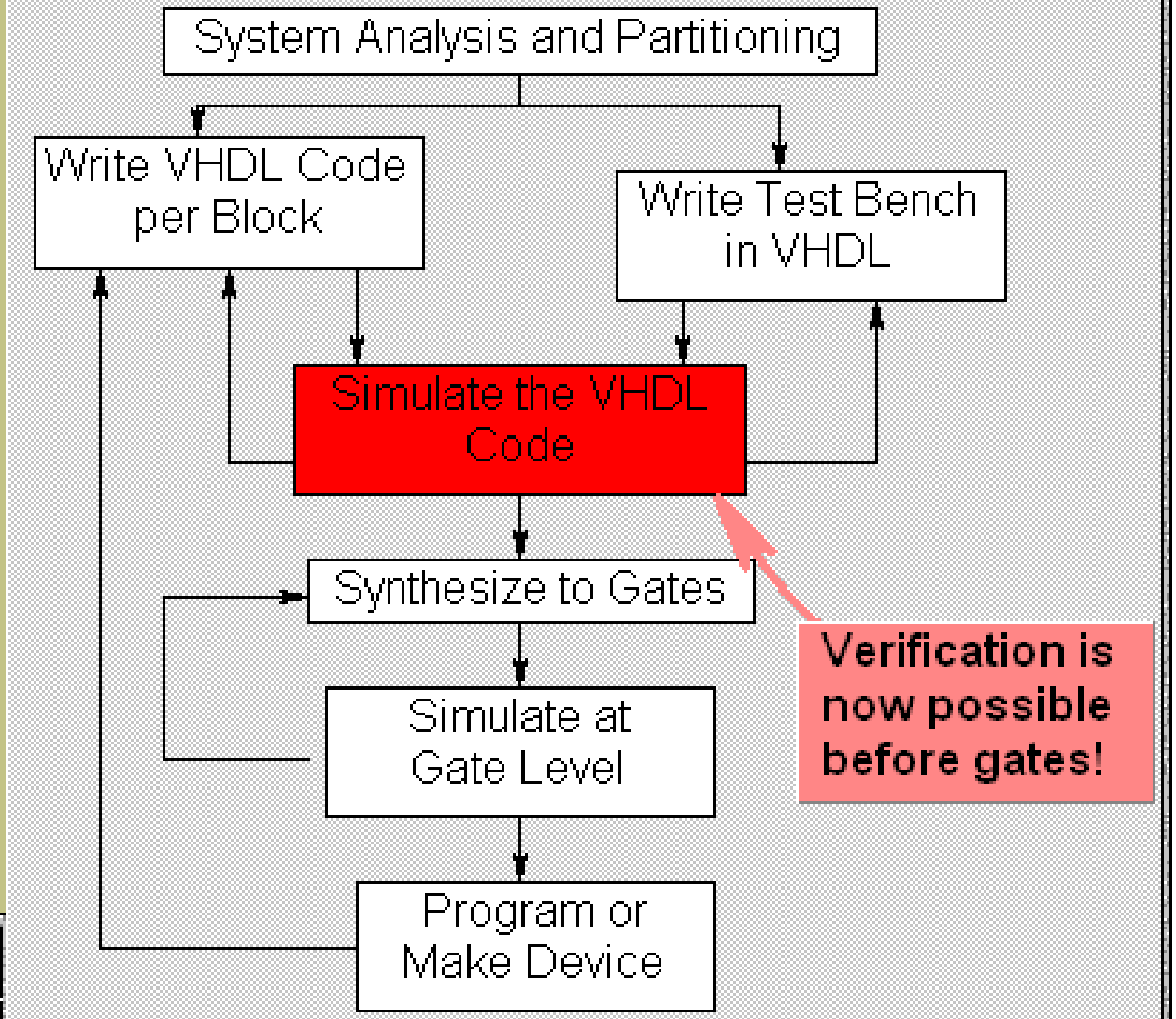
VHDL will mean changes to the:

- Design process
- Tools used
- Methods used
- Skills required

Implications of using VHDL

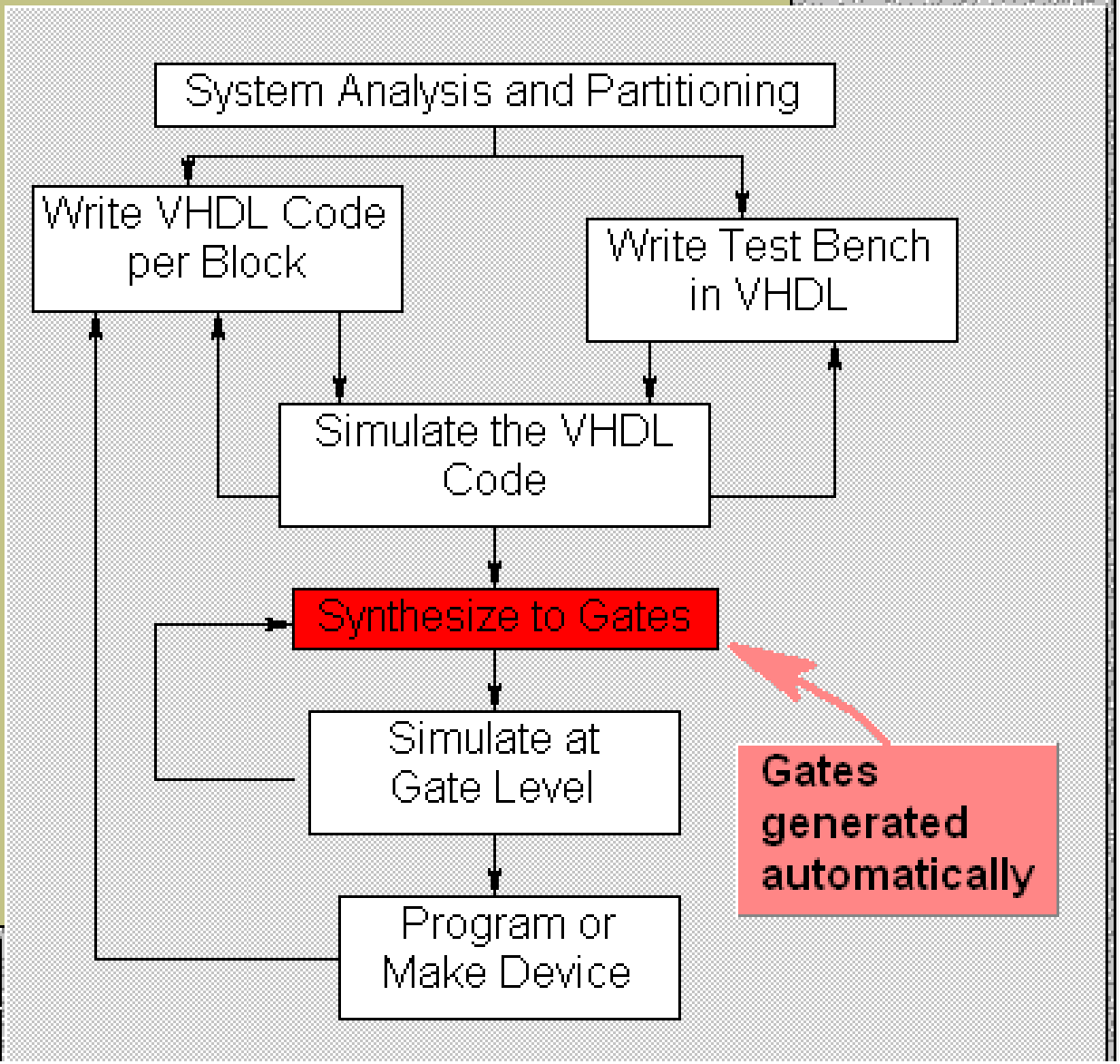
VHDL will affect the way the design is described. The productivity benefits of VHDL come from describing the design at a higher level.

Use of VHDL enables the design to be proved by simulation early in the design process, i.e. before the gate level design!



Implications of using VHDL

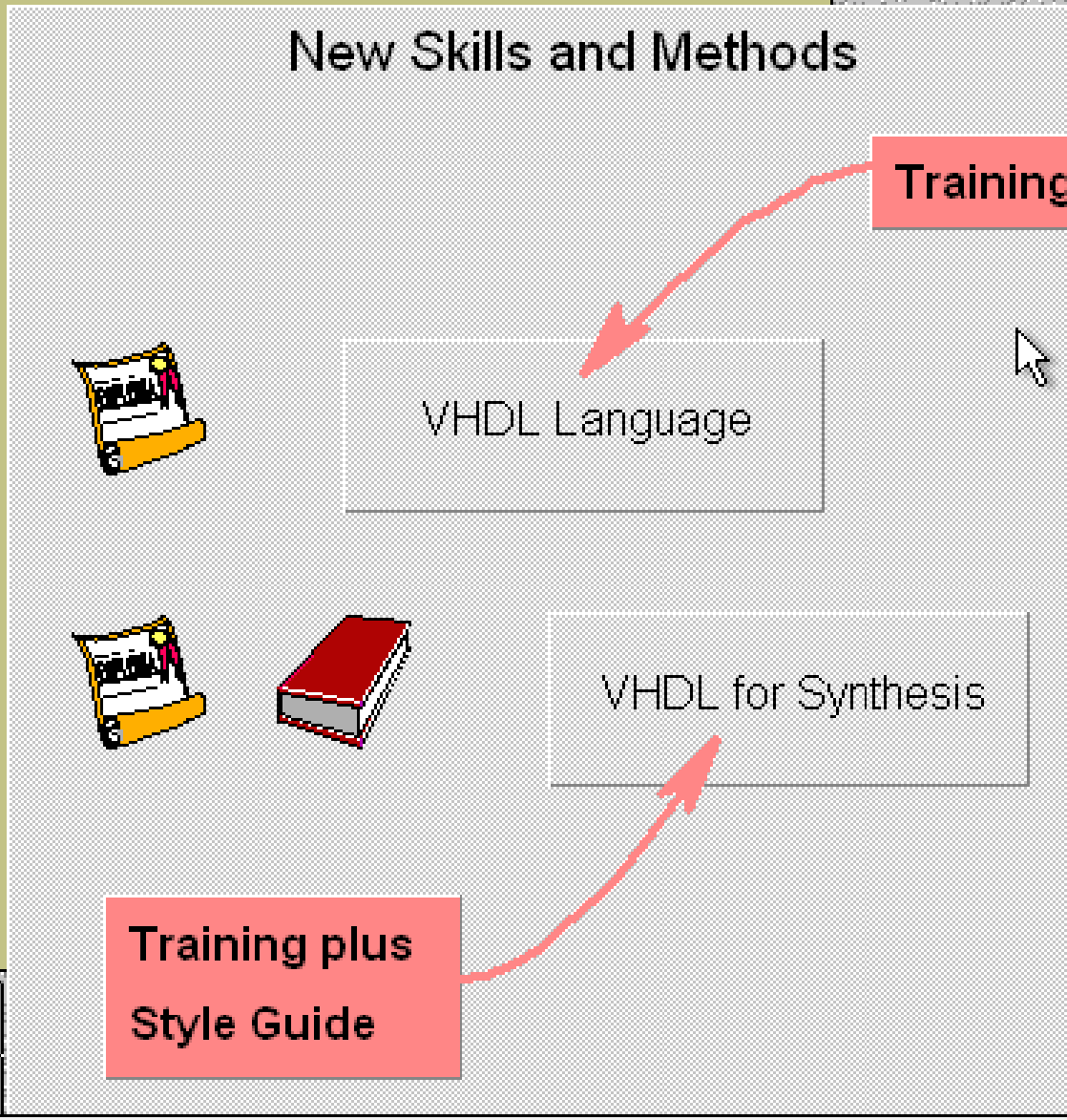
The power of applying Synthesis in the design process is that the implementation, i.e. the gates, are automatically generated from the high level description.



Implications of using VHDL

In order to use VHDL effectively, the engineer must not only learn the syntax of the language, but also how to write VHDL code that will be an efficient and effective style for synthesis.

A little later you will see why good style is so important for synthesis!



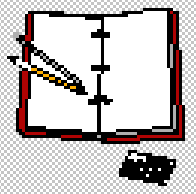
Implications of using VHDL

The engineer must also learn how to write and simulate VHDL test benches.

There are other skills that must be acquired or adapted including some which relate to good software design practice.

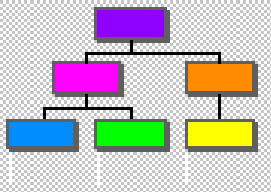
However, many of the design principles required for traditional hardware design still hold good in the design world of VHDL!

New Skills and Methods



VHDL Test Benches

Methodology

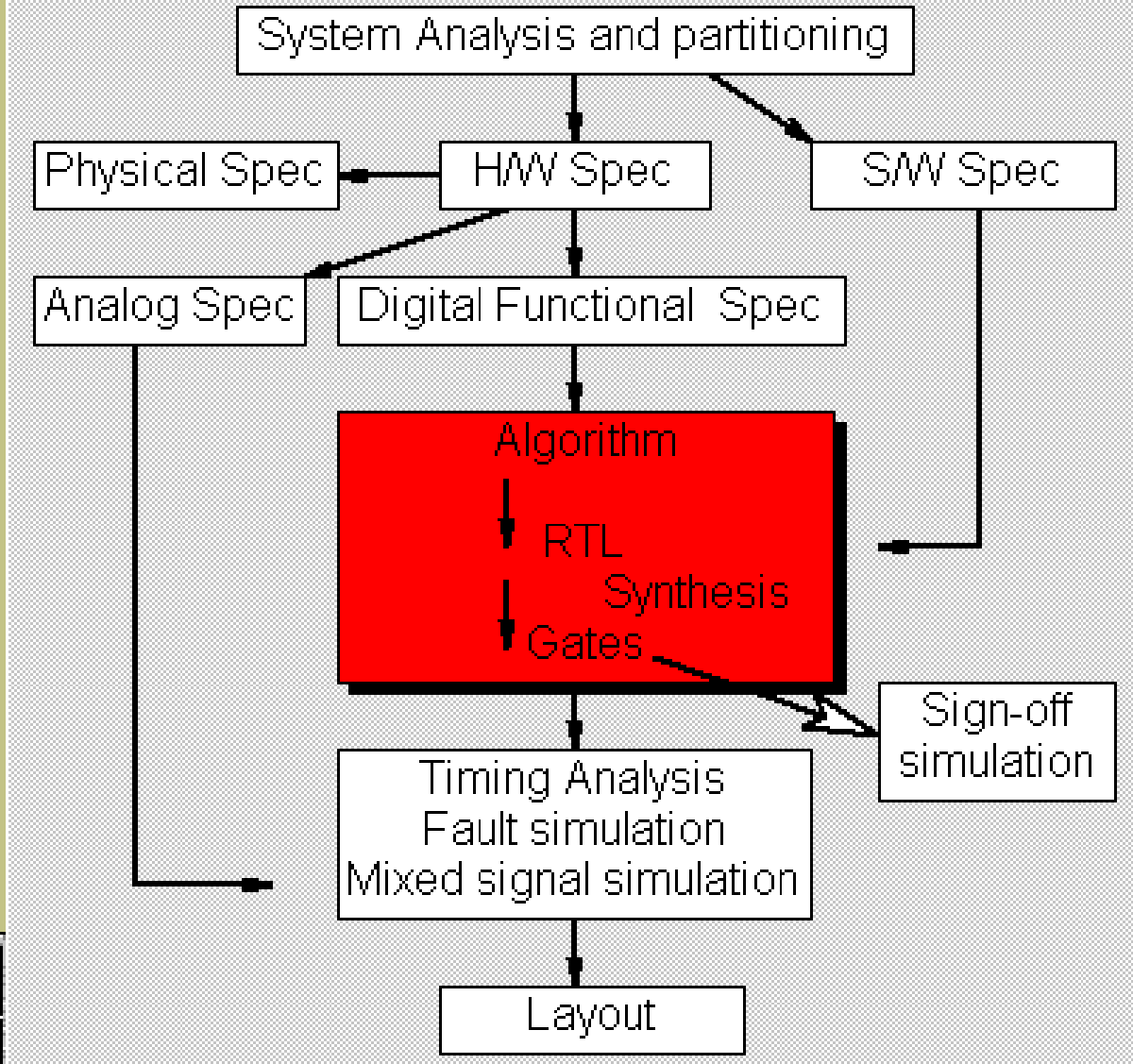


Design Management

Discipline

Scope of VHDL

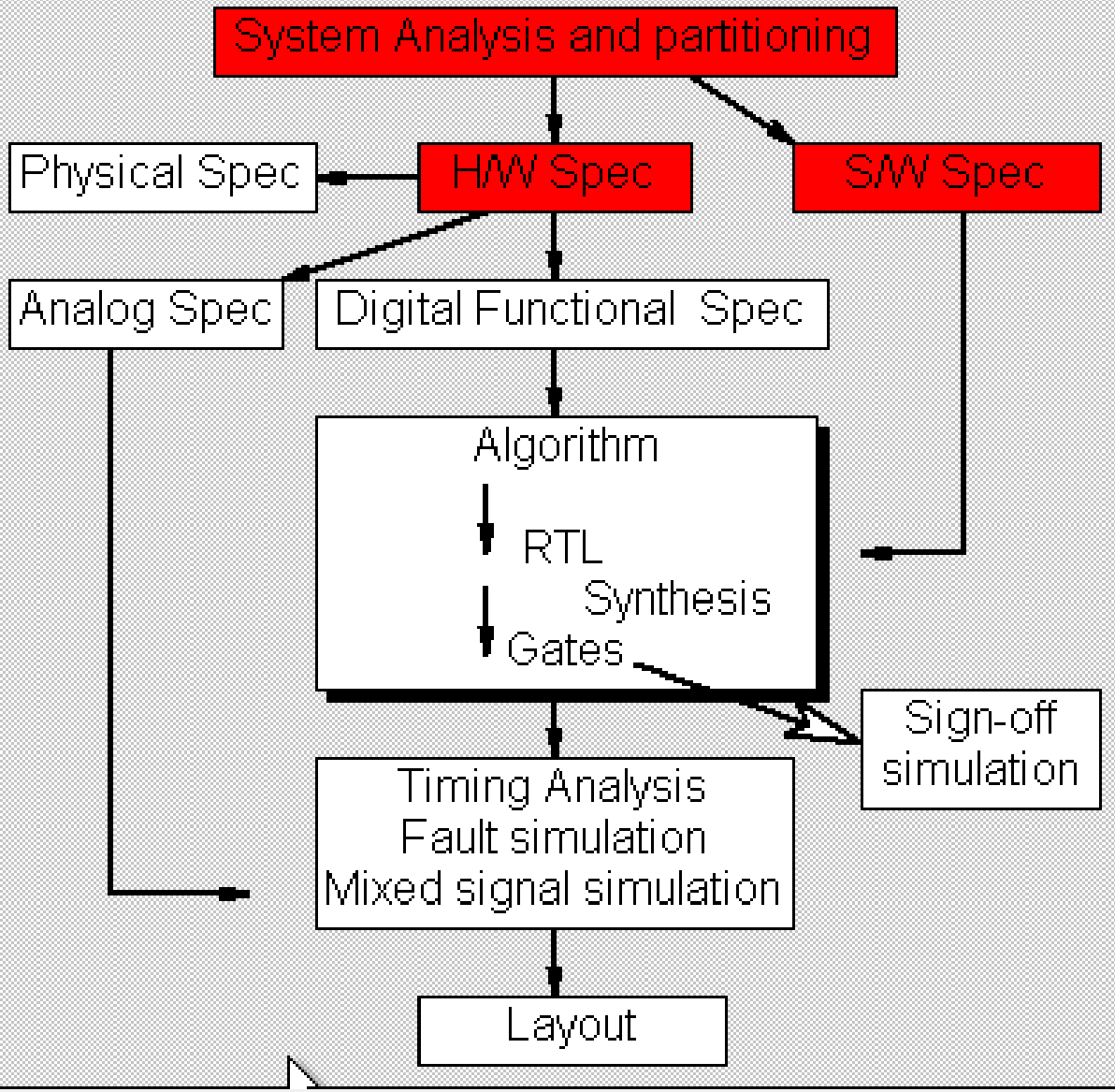
The diagram shows a very simplified view of the electronic system design process incorporating VHDL. VHDL has the greatest impact on the central portion of the diagram shown in red - the design of digital ASICs, FPGAs and CPLDs.



Scope of VHDL

VHDL can be used as a high level modelling language to aid in system analysis and partitioning, or as a way of validating the system specification.

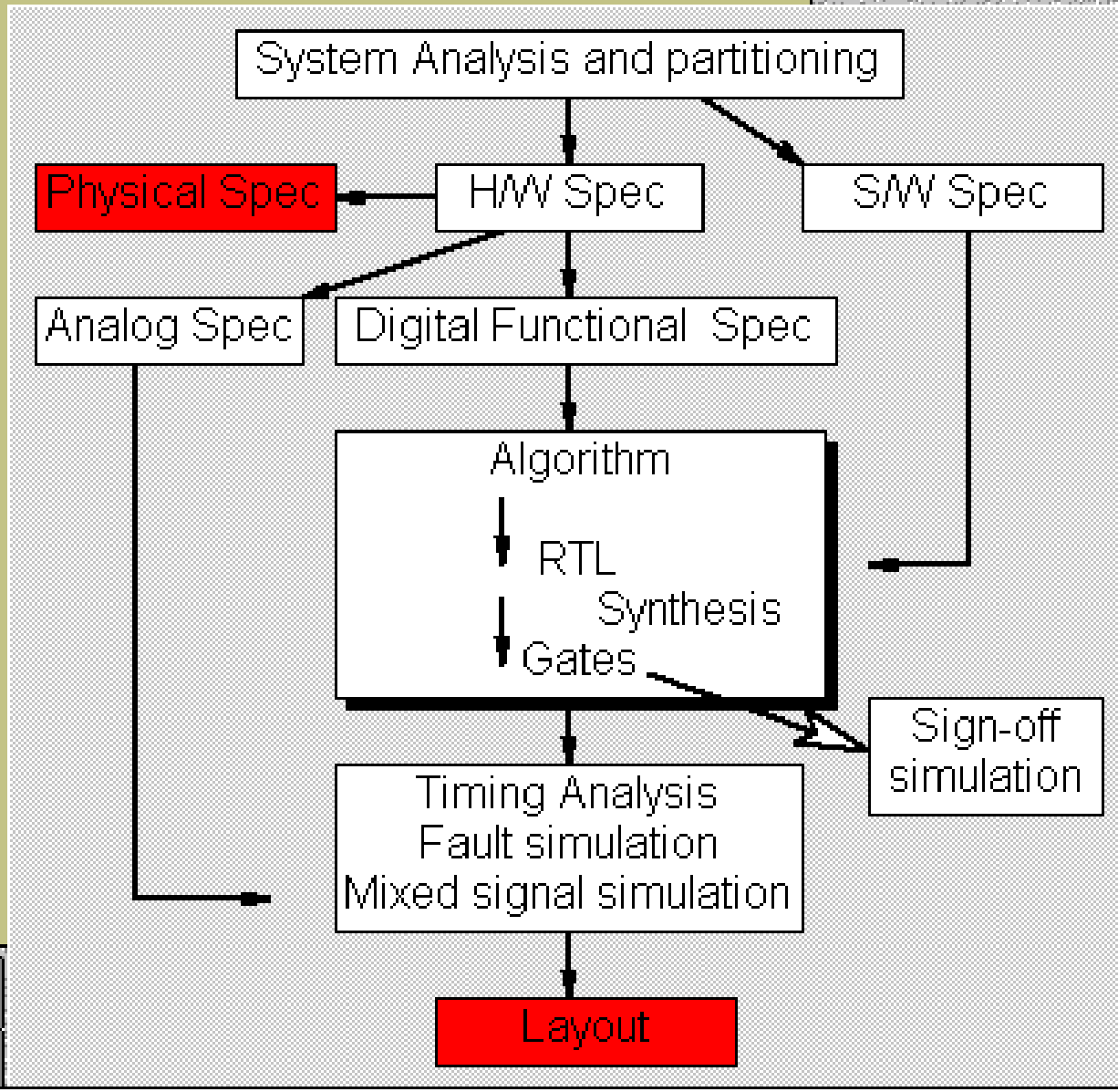
Other computer languages such as 'C' are equally useful for this purpose. The only good reason to use VHDL here would be to have a single simulation environment from specification through to implementation.



Scope of VHDL

Perhaps it is as well to know what VHDL cannot be used for! VHDL is **not** normally used to describe physical information, layout, manufacturing information, packaging, physical or electrical operating conditions, worst case timing, power consumption, thermal analysis, reliability, availability or price!

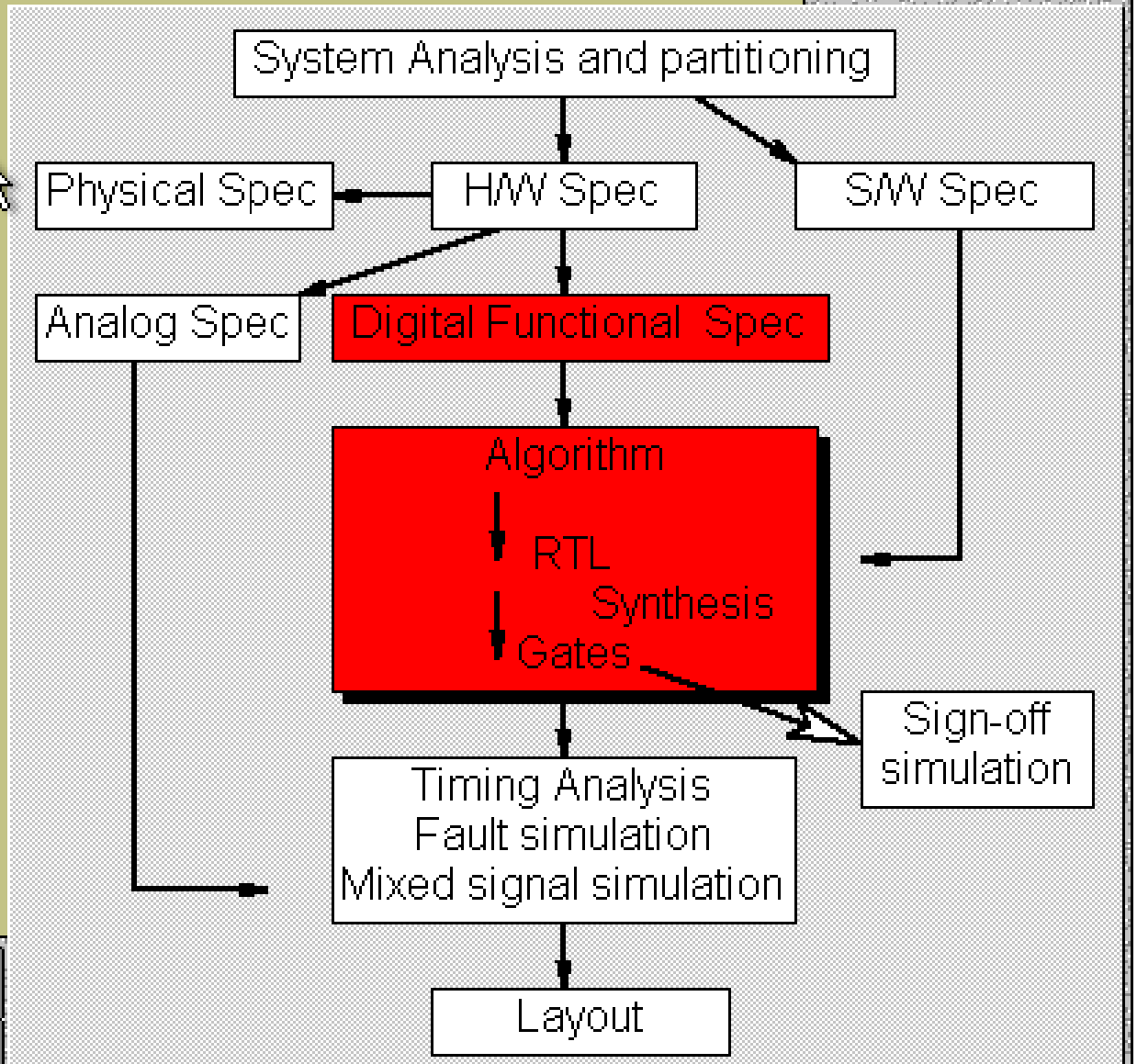
Well, no language is perfect.



Scope of VHDL

With VHDL, the focus is on the functionality of the system, and that of the digital hardware in particular.

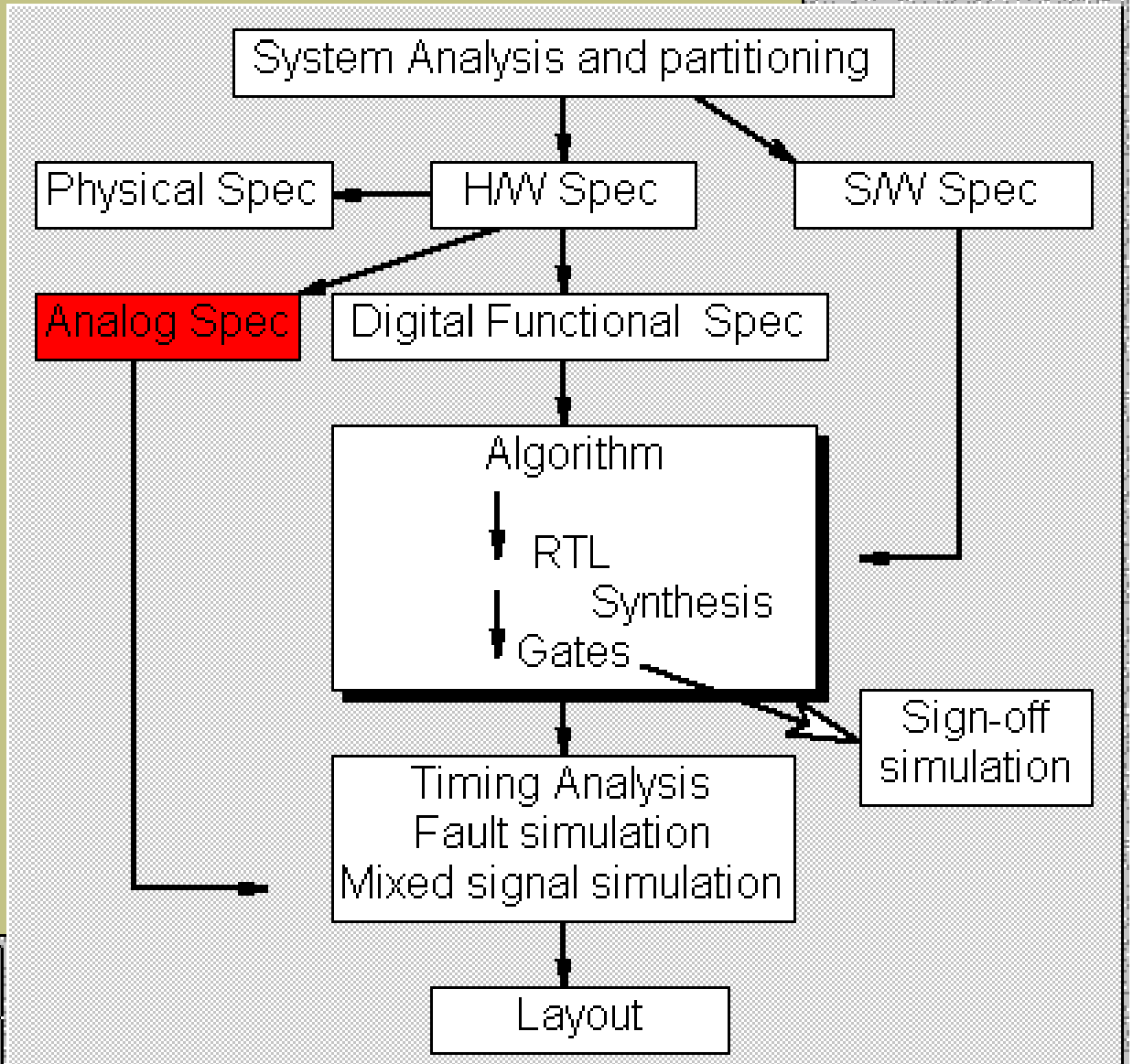
VHDL really gives a very narrow view of your system. But despite this, VHDL is a rich and powerful language which can be used to describe hardware at several different levels of abstraction, as we will see shortly.



Scope of VHDL

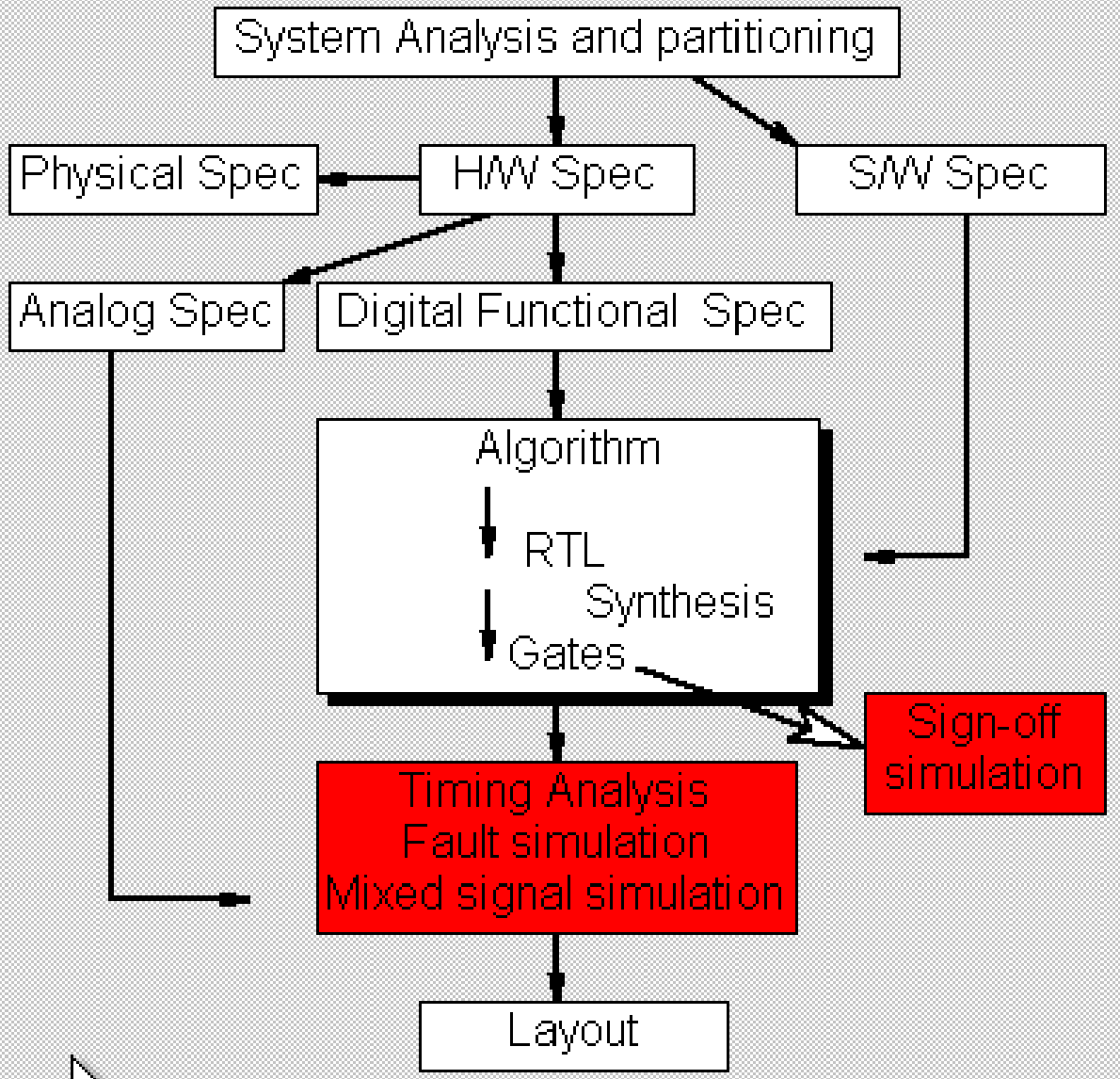
VHDL's power and generality have allowed it to be used to model and simulate analog circuitry such as D to A converters.

However there is now an Analog VHDL standard which allows analog and mixed signal simulation.



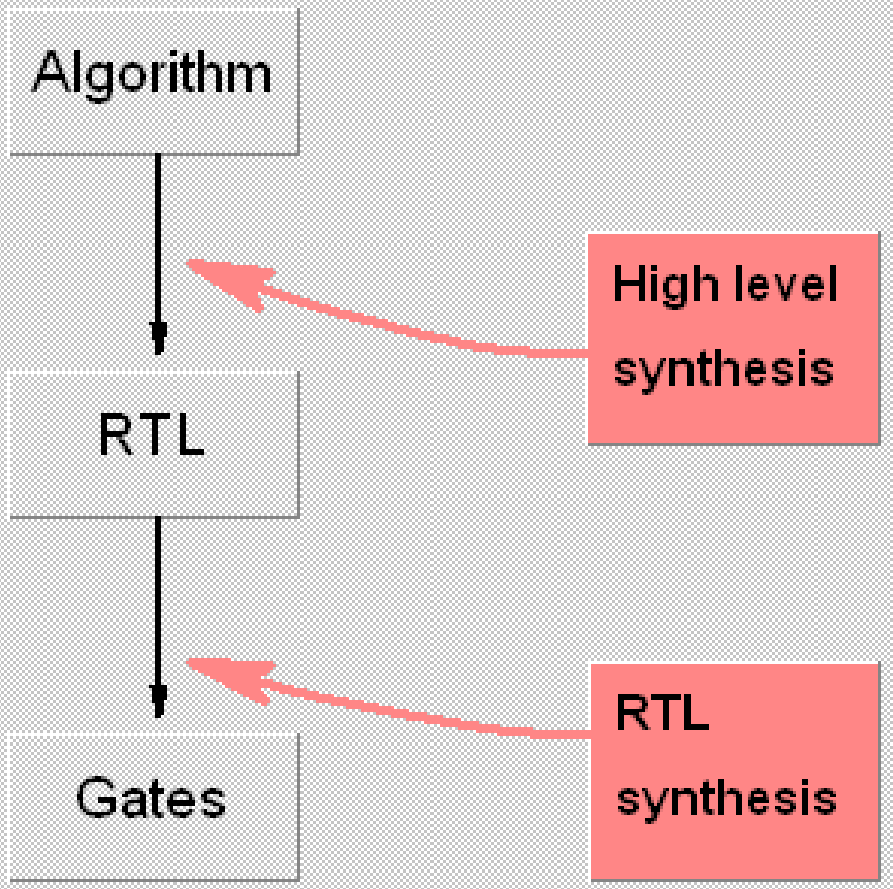
Scope of VHDL

Until recently, VHDL has not been used for sign off simulation, or for gate level timing or fault simulation. The situation is rapidly changing, however, as a new standard called VITAL comes on-line. VITAL is the VHDL Initiative Toward ASIC Libraries, and permits fast, accurate gate level timing simulation in VHDL. A front-to-back design flow using VHDL is now possible.



Levels of Abstraction

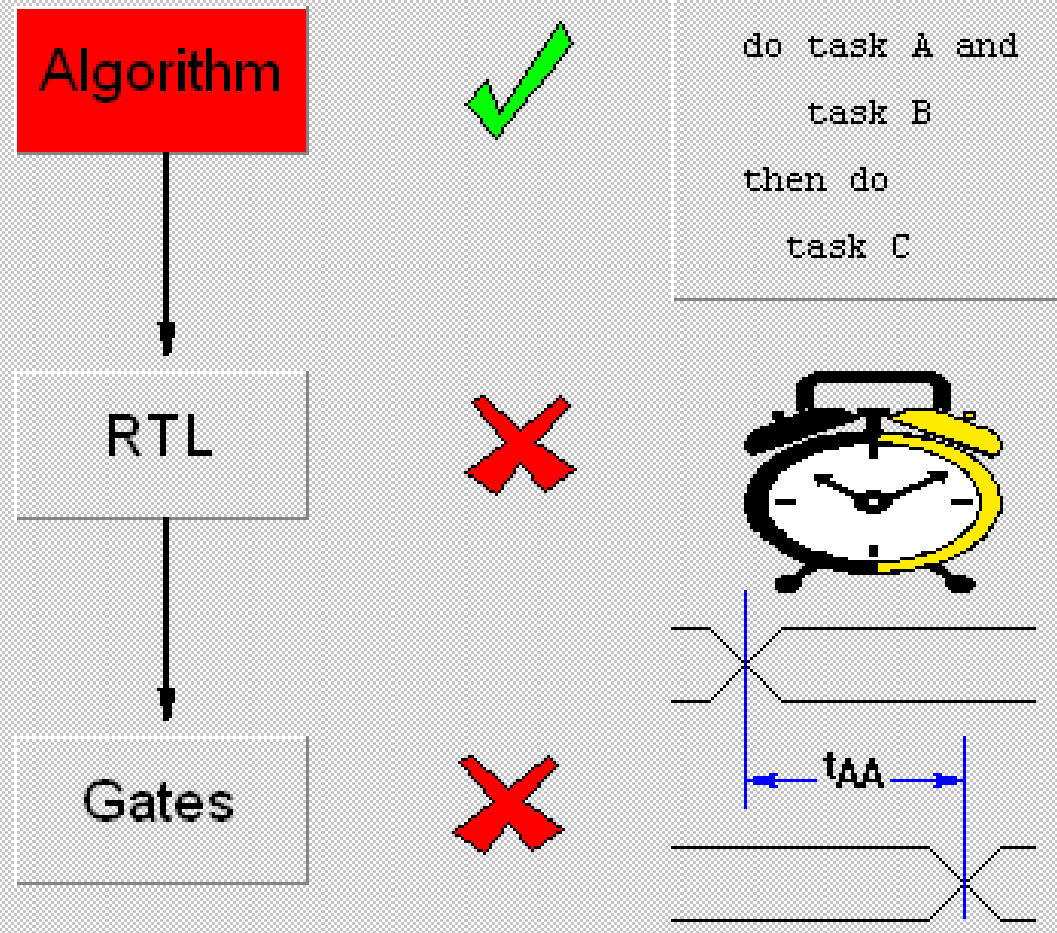
VHDL can be used to describe hardware at many levels of abstraction. When considering the application of VHDL to ASIC design, it is helpful to identify and understand the three levels of abstraction shown. Algorithms are unsynthesizable, RTL is the input to synthesis, gate level is the output from synthesis. The difference between these levels can be understood in terms of timing.



Levels of Abstraction

A pure algorithm consists of a set of instructions that are executed in sequence to perform some task.

A pure algorithm has neither a clock nor detailed delays. Some aspects of timing can be inferred from the partial ordering of operations within the algorithm.



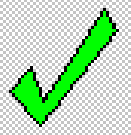
Levels of Abstraction

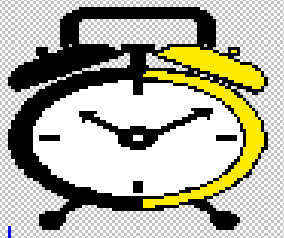
An RTL description has an explicit clock. All operations have been scheduled to occur in specific clock cycles, but there are no detailed delays below the cycle level.

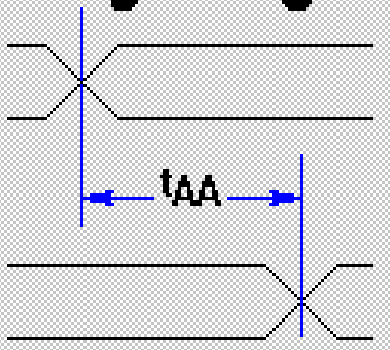
Algorithm

```
while Ready
   do task A and
      task B
   then do
      task C
```

RTL





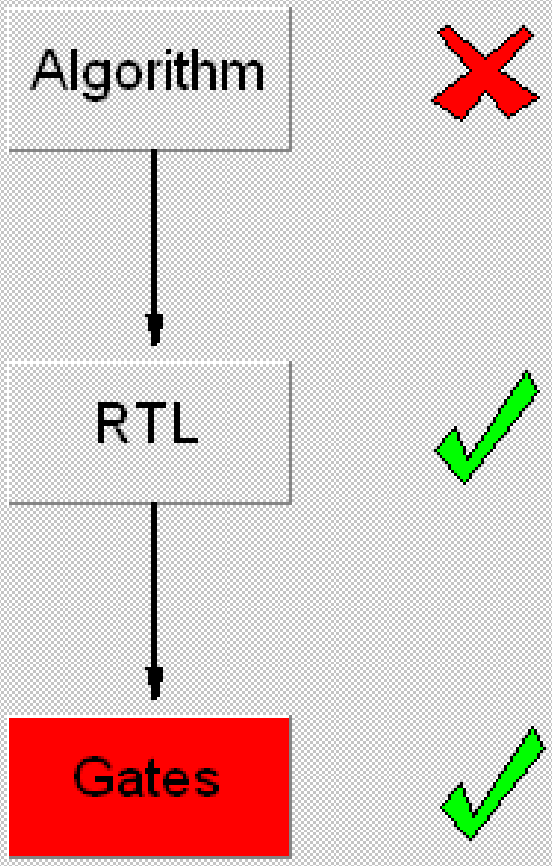


Gates

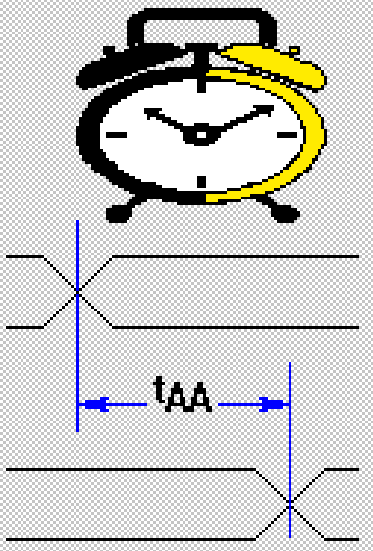
Levels of Abstraction

A gate level description consists of a network of gates and registers instanced from a technology library, which contains technology specific delay information for each gate.

The original clocking scheme defined at the RTL level is maintained by the synthesis tool.

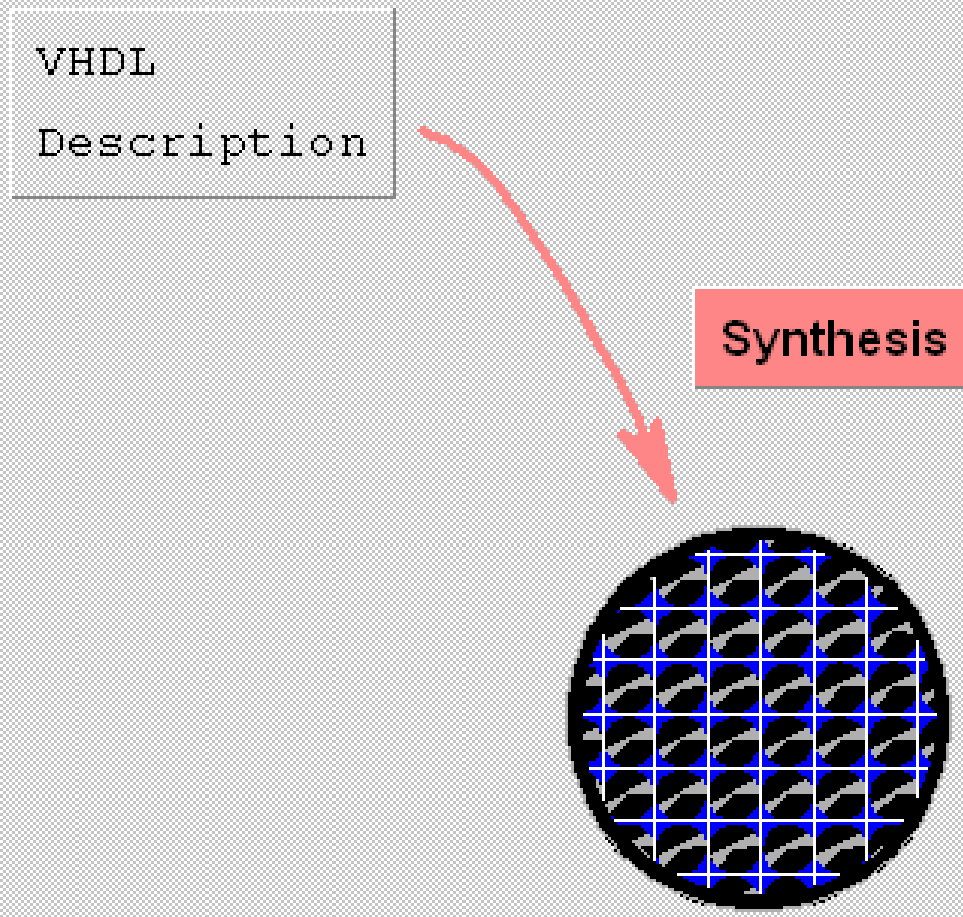



```
while Ready
   do task A and
      task B
   then do
      task C
```



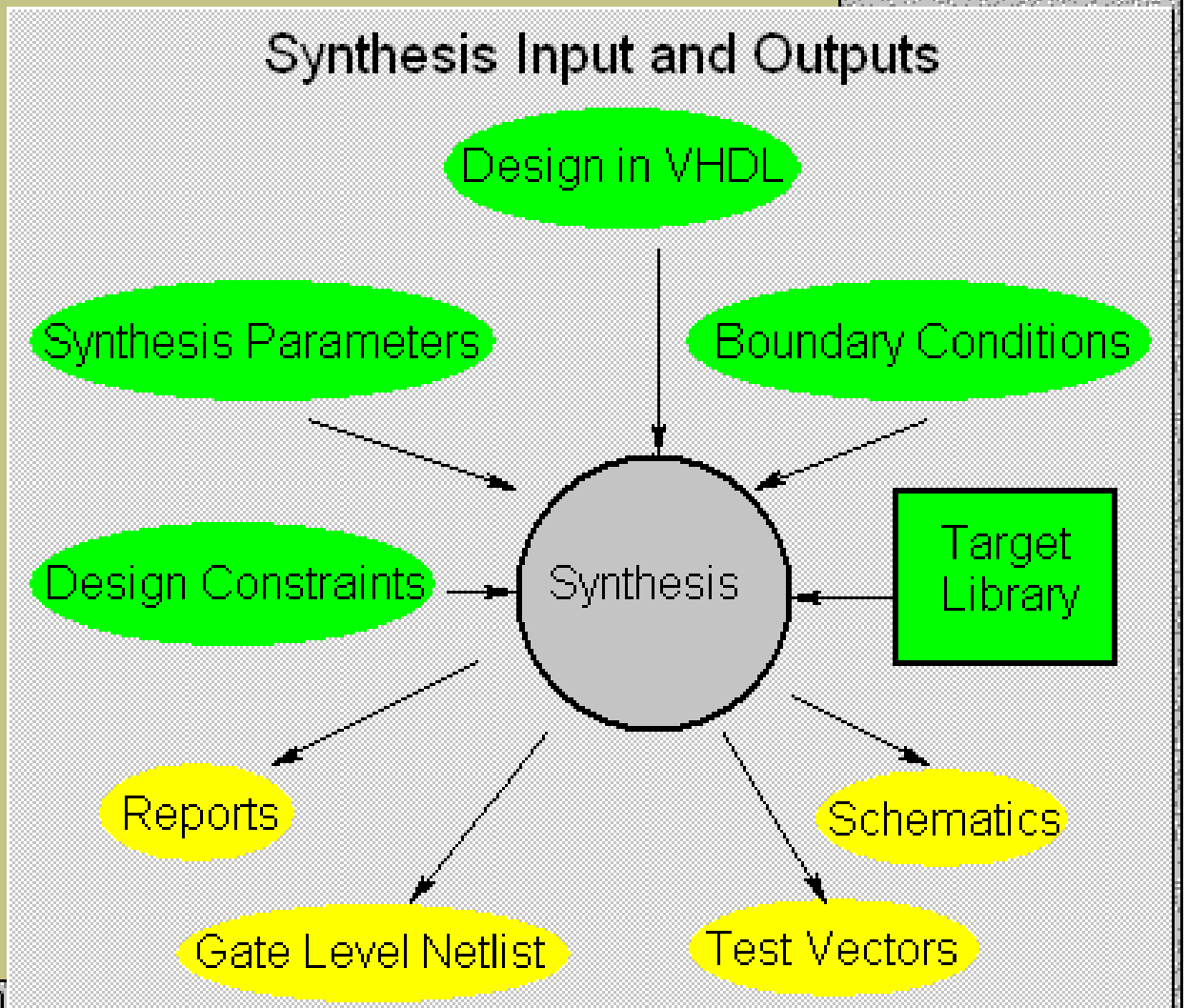
What is Synthesis?

Synthesis is a broad term often used to describe very different tools. Synthesis can include silicon compilers and function generators used by ASIC vendors to produce regular RAM and ROM type structures. Synthesis in the context of this course refers to generating random logic structures from VHDL descriptions. This is best suited to gate arrays and programmable devices, e.g. FPGA's.



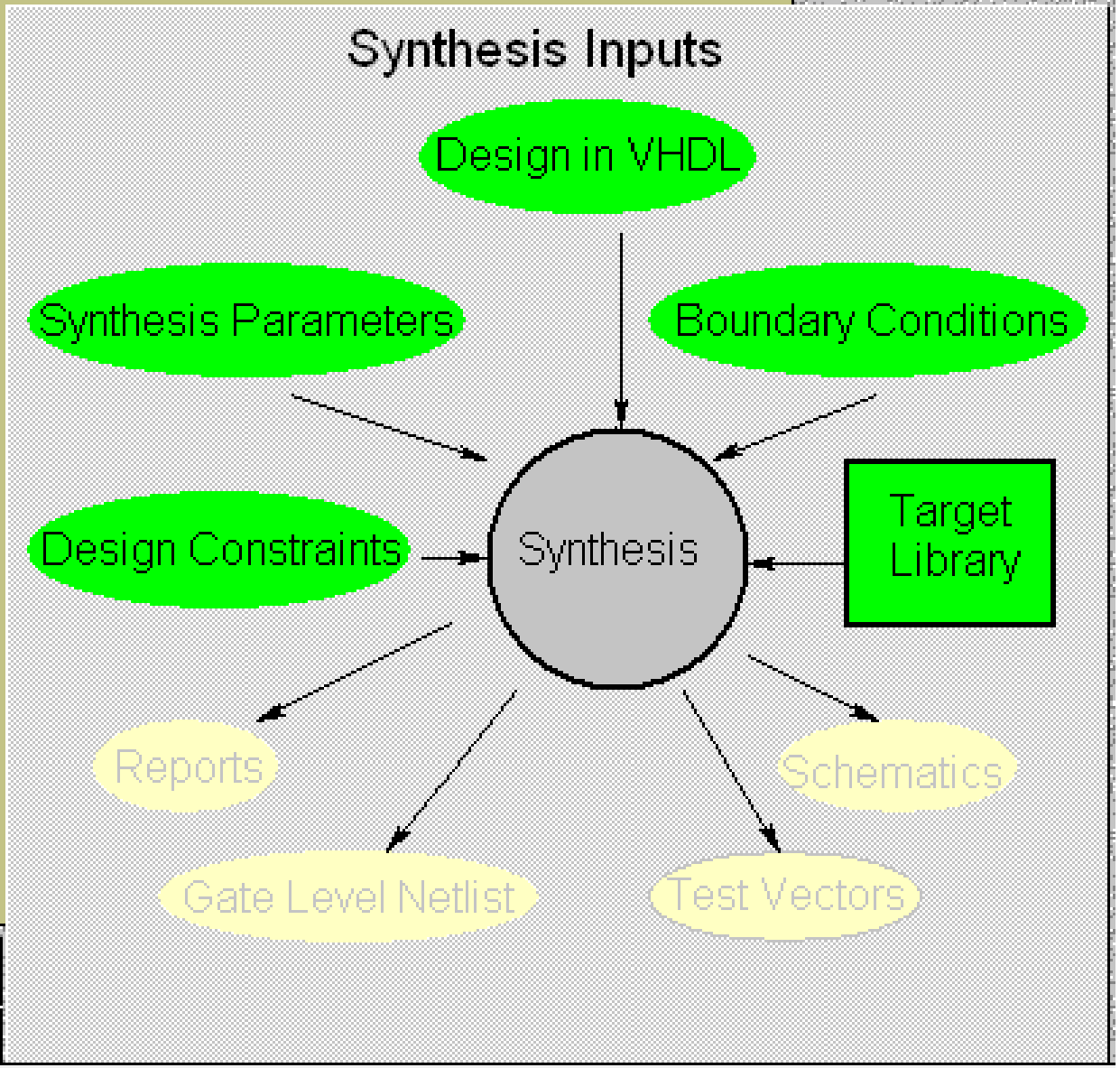
What is Synthesis?

The diagram shows the various inputs and outputs to a typical synthesis tool. The target library, although an input to the synthesis process, is usually provided by the ASIC, FPGA or tool vendor, and contains a description of the functional and electrical characteristics of the cells which the technology provides. These would include simple logic gates, flipflops and possibly some multi-bit functions.



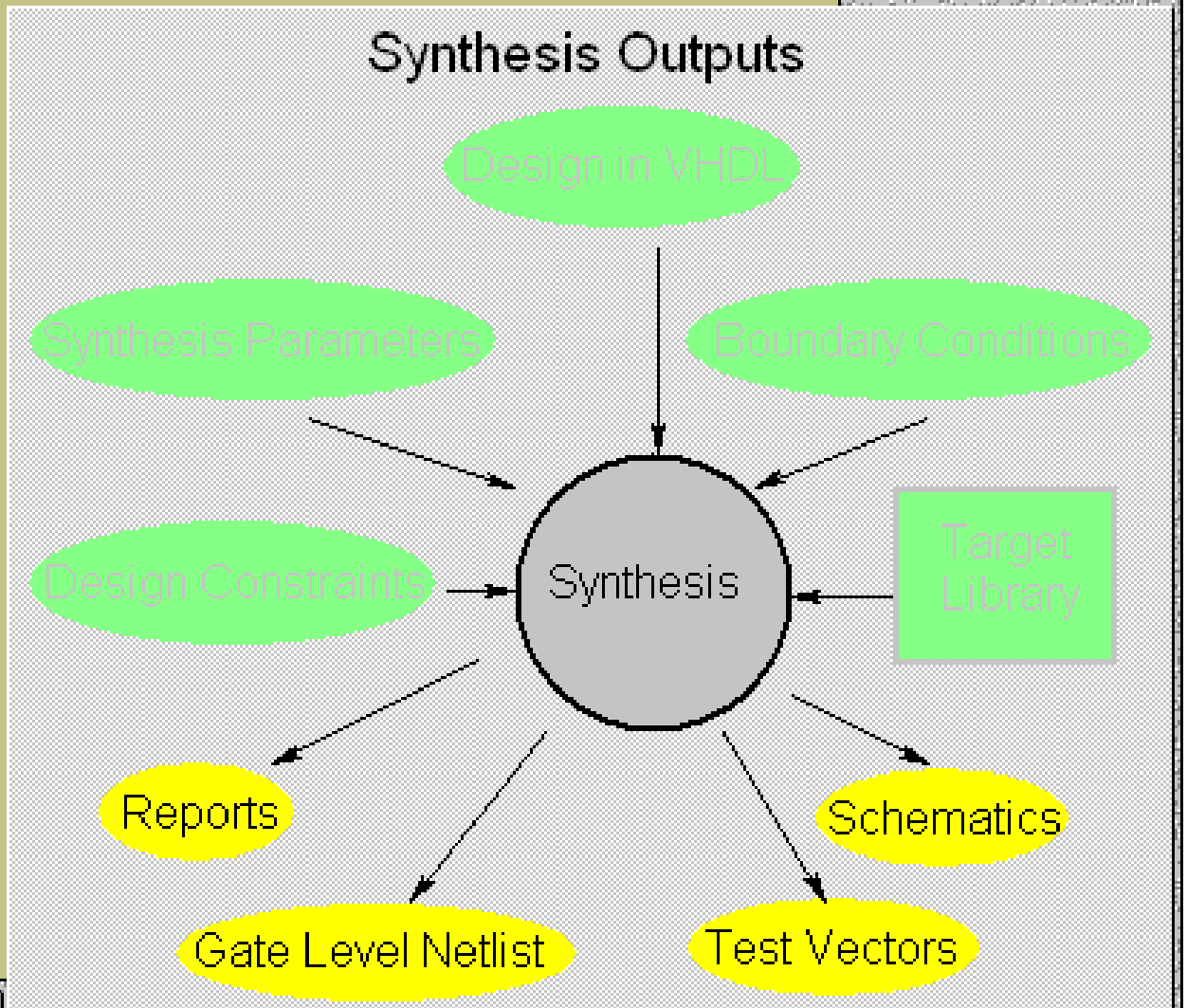
What is Synthesis?

The inputs to a synthesis tool include not only the VHDL, but also design constraints, boundary conditions and synthesis parameters and controls. For example, design constraints might include maximum clock period and maximum gate count; boundary conditions might include the capacitance on output pins; and synthesis parameters might include the CPU time allowed for optimization.



What is Synthesis?

The outputs from a synthesis tool include not only the synthesized design (usually in the form of a netlist), but also synthesized schematics and reports which aid in determining the quality of the design. Test synthesis tools will also output a set of automatically generated manufacturing test vectors (usually having modified the netlist to include test hardware).



What is Synthesis?

Synthesis is not a panacea! It is vital that everyone working on a project starts with realistic expectations of synthesis.

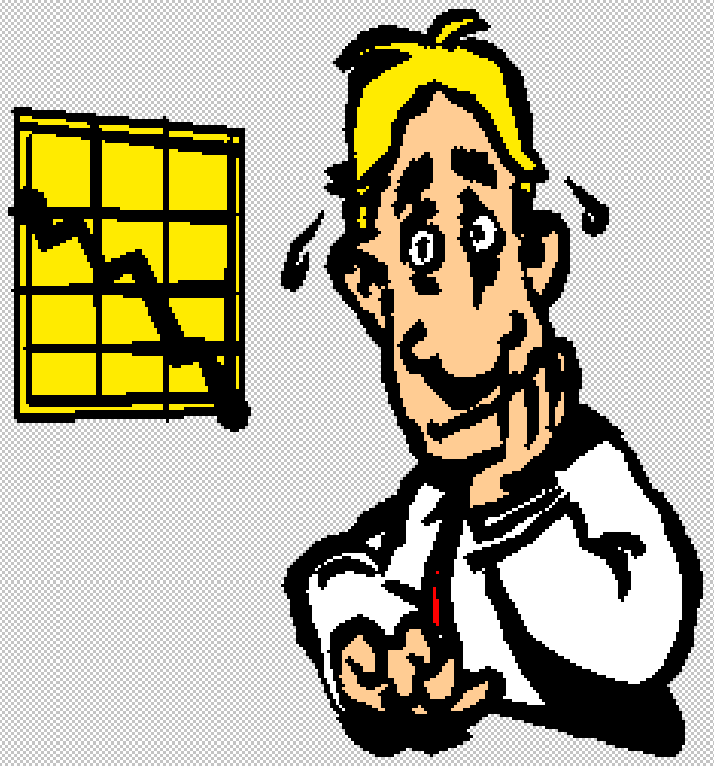
The definition of VHDL for simulation is cast in stone and enshrined in the Language Reference Manual. Other tools which use VHDL, such as synthesis, will make their own interpretation of the VHDL language.

Synthesis Caveats

Synthesis is sensitive to VHDL style

Rubbish in rubbish out

Synthesis does not replace human expertise



What is Synthesis?

The results of synthesis are very sensitive to the style in which the VHDL code is written.

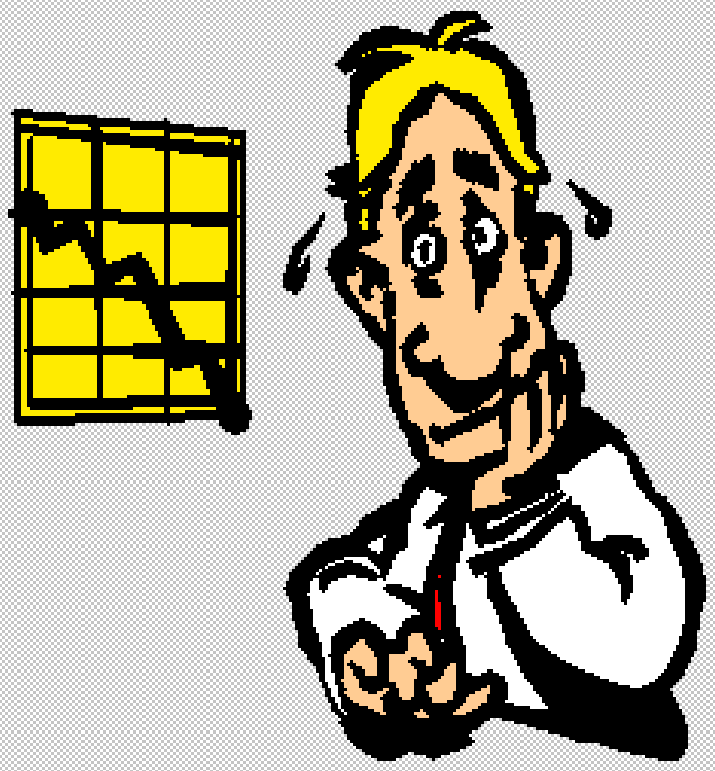
It is not sufficient that the VHDL is functionally correct; it must be written in such a way that it directs the synthesis tool to generate good hardware, and moreover, the VHDL must be matched to the idiosyncrasies of the particular synthesis tool being used.

Synthesis Caveats

Synthesis is sensitive to VHDL style

Rubbish in rubbish out

Synthesis does not replace human expertise



What is Synthesis?

With RTL synthesis, you the designer must be firmly in control of the design! Most of the design decisions are still made by the human, with the synthesis tool automating the gate level implementation.

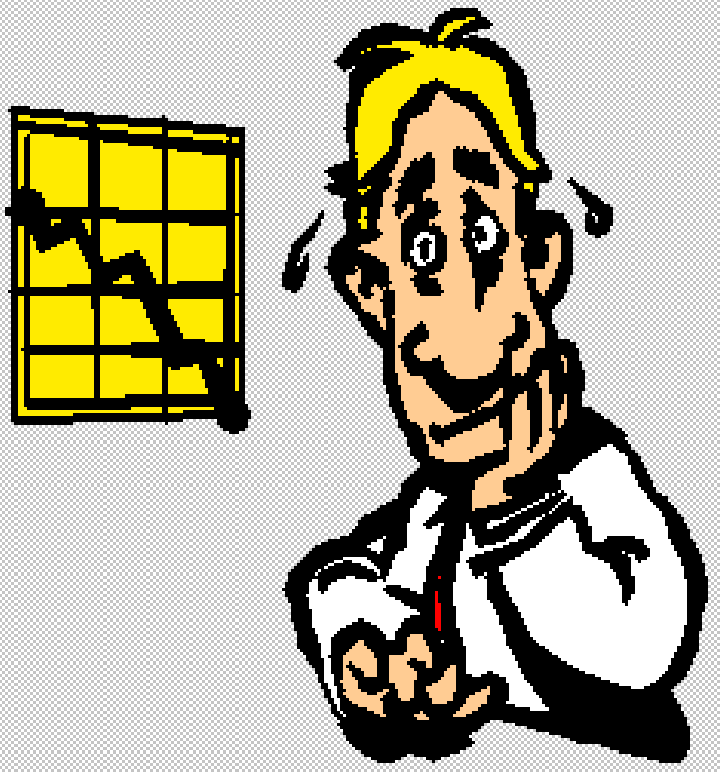
With synthesis, bad designers still produce bad designs, they just do so faster!

Synthesis Caveats

Synthesis is sensitive to VHDL style

Rubbish in rubbish out

Synthesis does not replace human expertise



What is Synthesis?

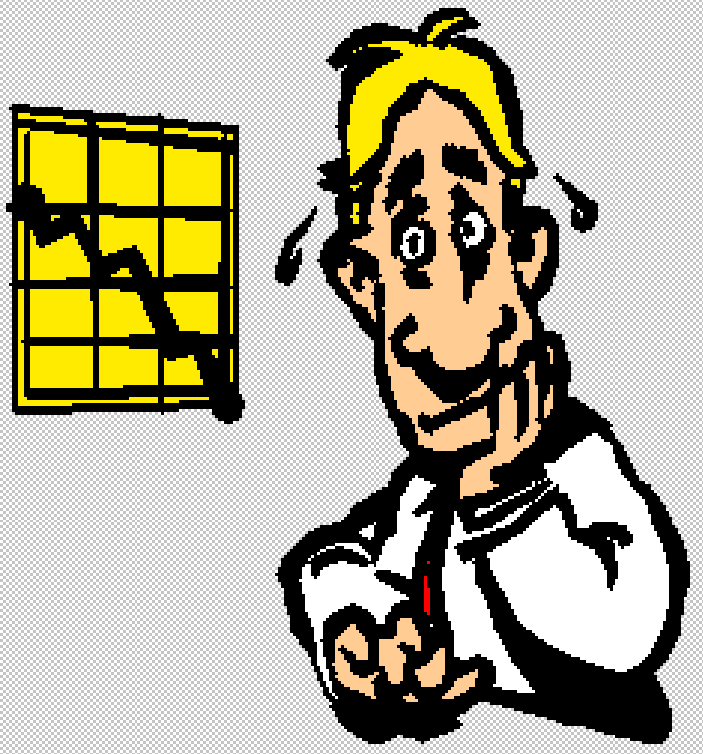
In many practical design situations, synthesis will produce results which equal or exceed those of the human designer. However, for designs that push the limits of the technology in terms of speed or density, synthesis alone is often not good enough, and manual intervention becomes necessary. Synthesis tools do not offer a substitute for human design knowledge and expertise.

Synthesis Caveats

Synthesis is sensitive to VHDL style

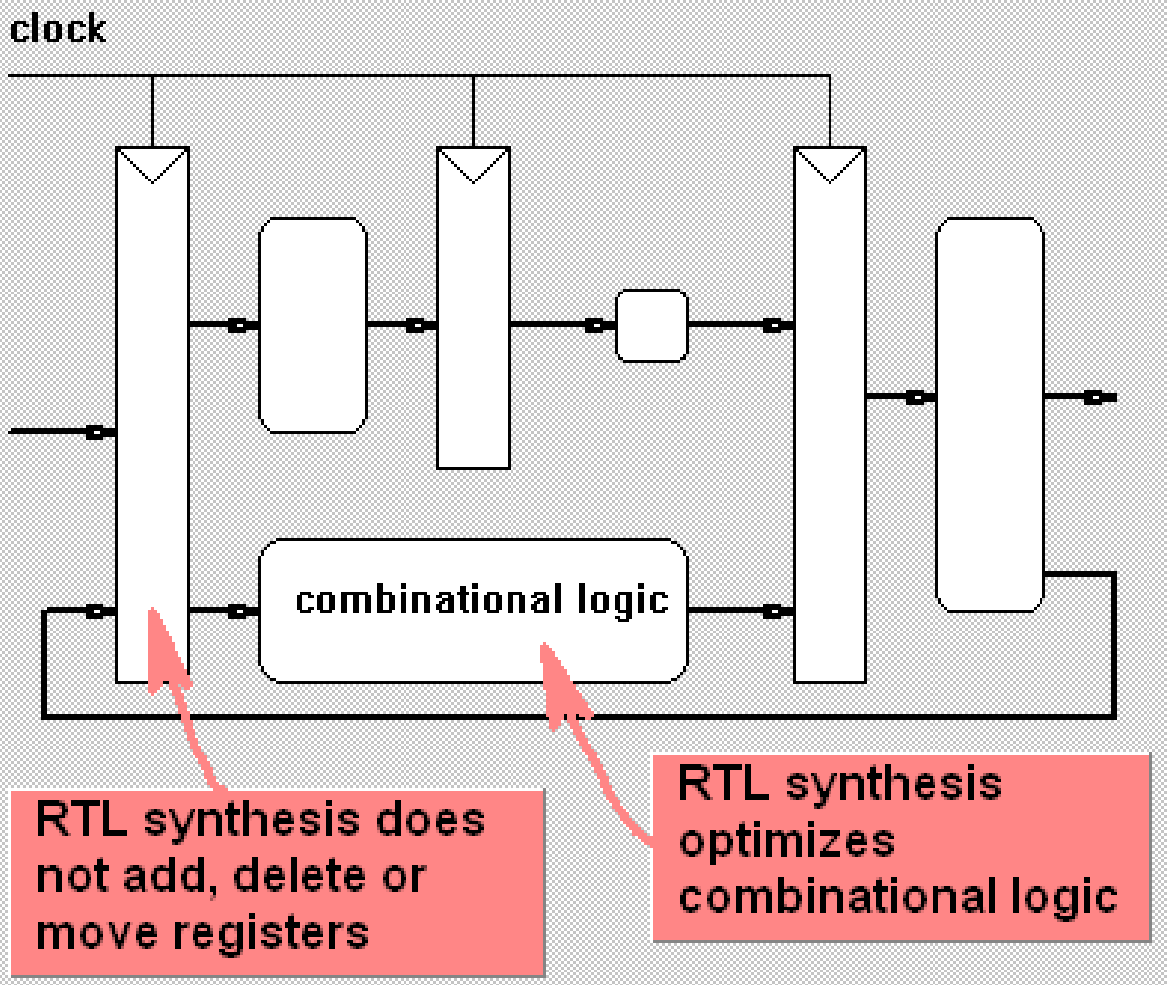
Rubbish in rubbish out

Synthesis does not replace human expertise



What is RTL Synthesis?

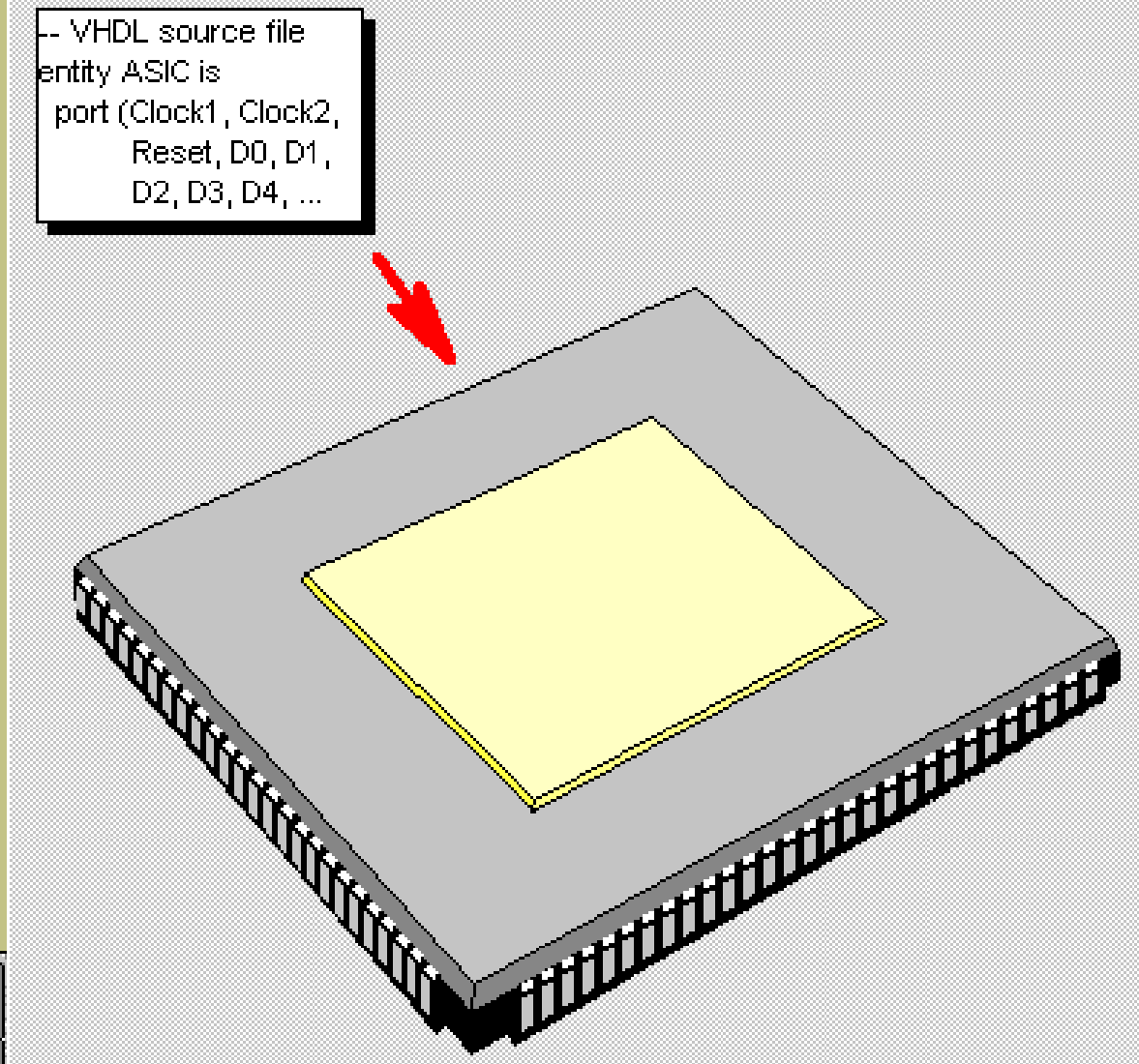
A Register Transfer Level (RTL) description (in VHDL) must explicitly synchronize all operations with a clock. The essence of RTL is that you define the registers and the transfers between those registers that occur on each clock tick (i.e. the combinational logic). RTL Synthesis tools infer the existence of registers directly from the VHDL source following a few simple rules.



ASICs and FPGAs

VHDL is used to design and synthesize gate array and standard cell ASICs, FPGAs and complex PLDs.

Although the same VHDL language is used to design ASICs and FPGAs, the philosophy you need to adopt when expressing the design in VHDL is rather different.

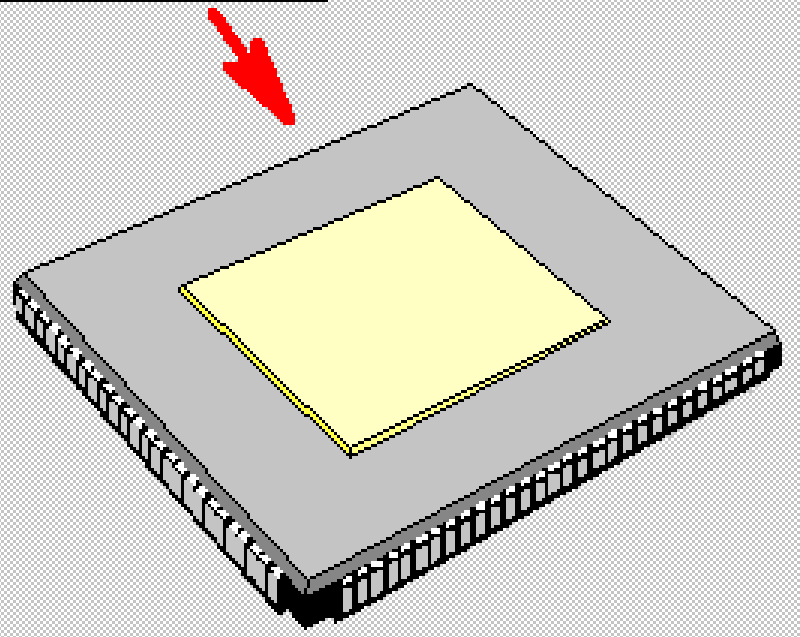


ASICs and FPGAs

To design ASICs with VHDL, you must write a good RTL description. The synthesis tool optimizes the combinational logic to exploit the types of cell available in the technology.

To create efficient FPGA designs, you **must** write the VHDL code to directly exploit the architectural features of the device. For example, logic function generators with limited fanin.

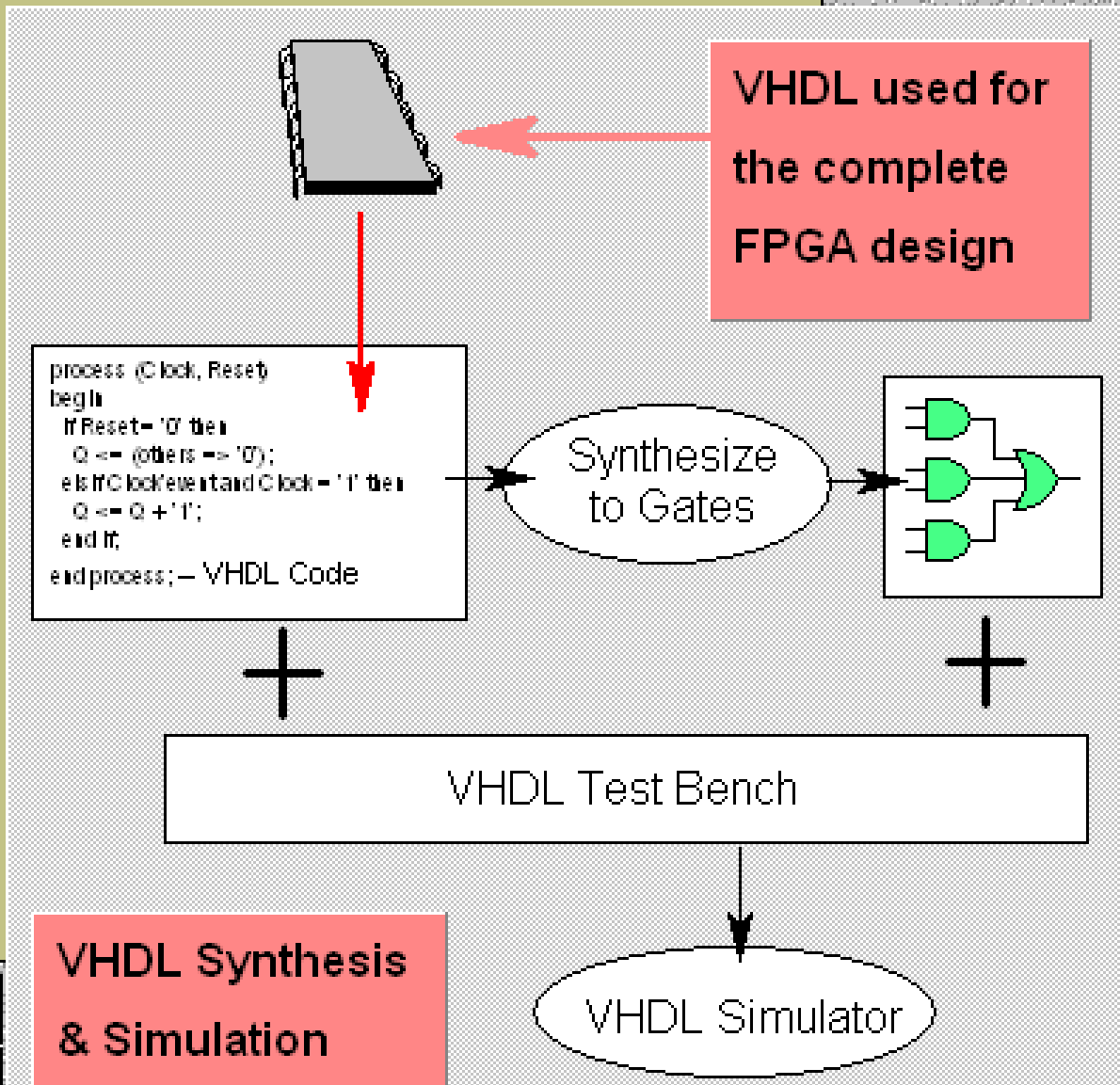
```
-- VHDL source file
entity FPGA is
  port (Clock1, Clock2,...

-- 1st logic block
F <= A xor B xor C xor D;

-- 2nd logic block
G <= (A nand B) or (C nor D);
```



Migrating to VHDL for FPGAs

With the increasing size of FPGAs, it becomes more difficult to capture the complete design using traditional methods. When the full design is captured using VHDL, you will need a VHDL simulator and synthesis tool.

You will need to develop a VHDL test bench which allows you to verify the FPGA design before and after synthesis.

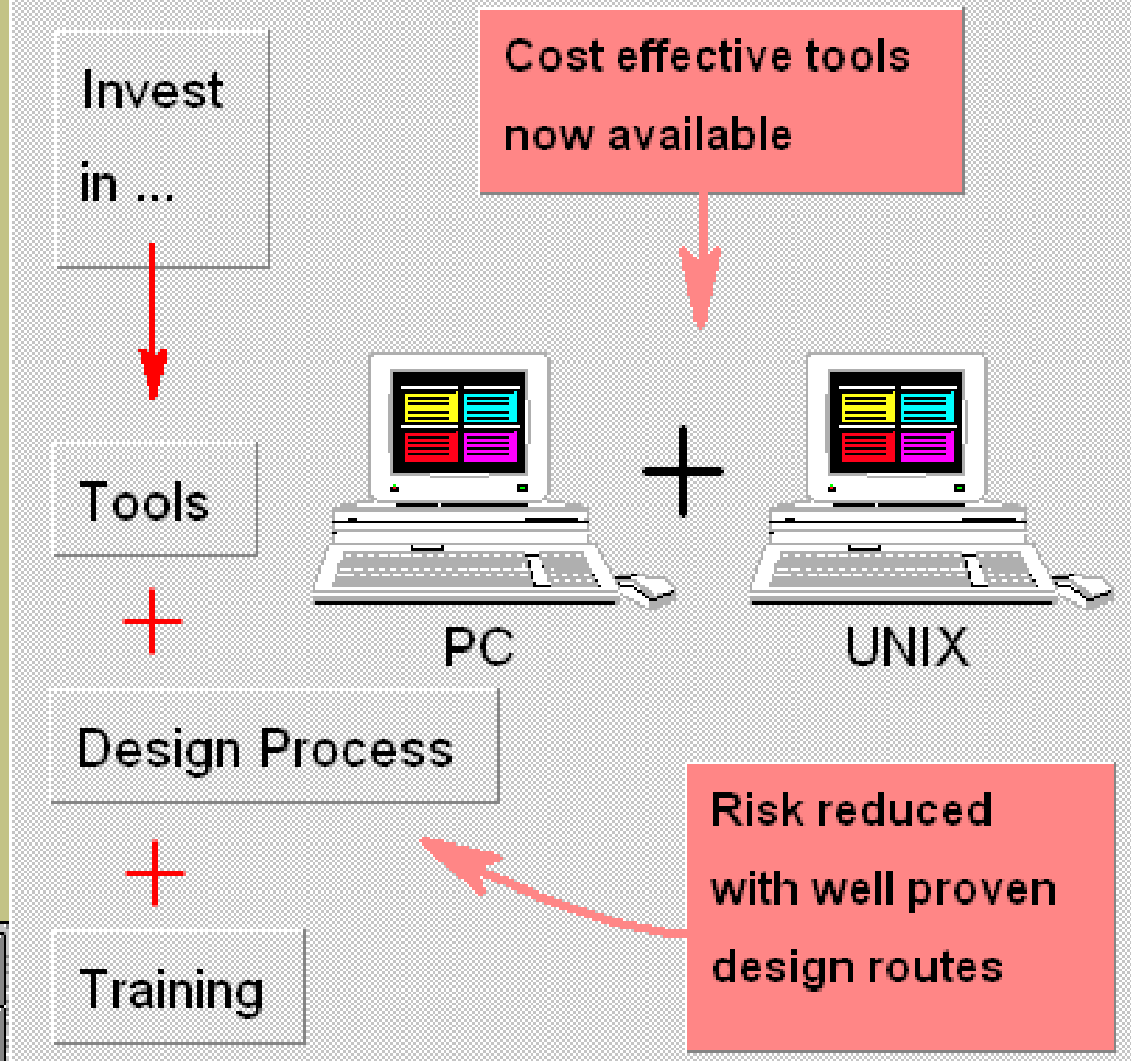


Investing in VHDL

There are many reliable and affordable simulation and synthesis tools available for both PCs and Unix workstations.

Gearing up for VHDL design need not be a risky venture. Many FPGA vendors have well proven design routes and integrated toolsets.

Training in VHDL and its application is still a vital part of your investment.

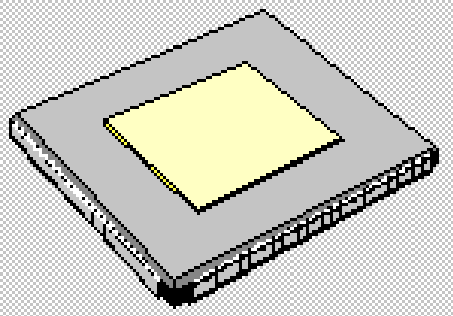


Synthesis for FPGAs

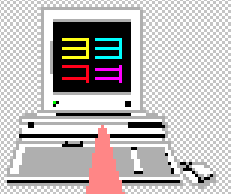
Whether you are designing ASICs or FPGAs, there is a large range of tools available.

High capability tools are usually targeted at ASIC design.

FPGA design tools are more focused. They are typically windows based, less flexible, low cost, and are part of a toolset tightly coupled with the chosen FPGA technology.



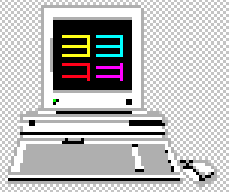
ASIC Design

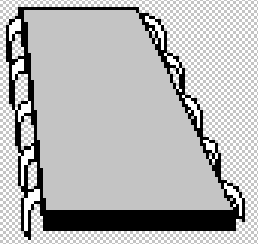


Less capable tools but, integrated with layout or fitting

High capability tools for synthesis and simulation

FPGA Design





Benefits of VHDL

VHDL can be used across a wide range of ASICs and programmable devices.

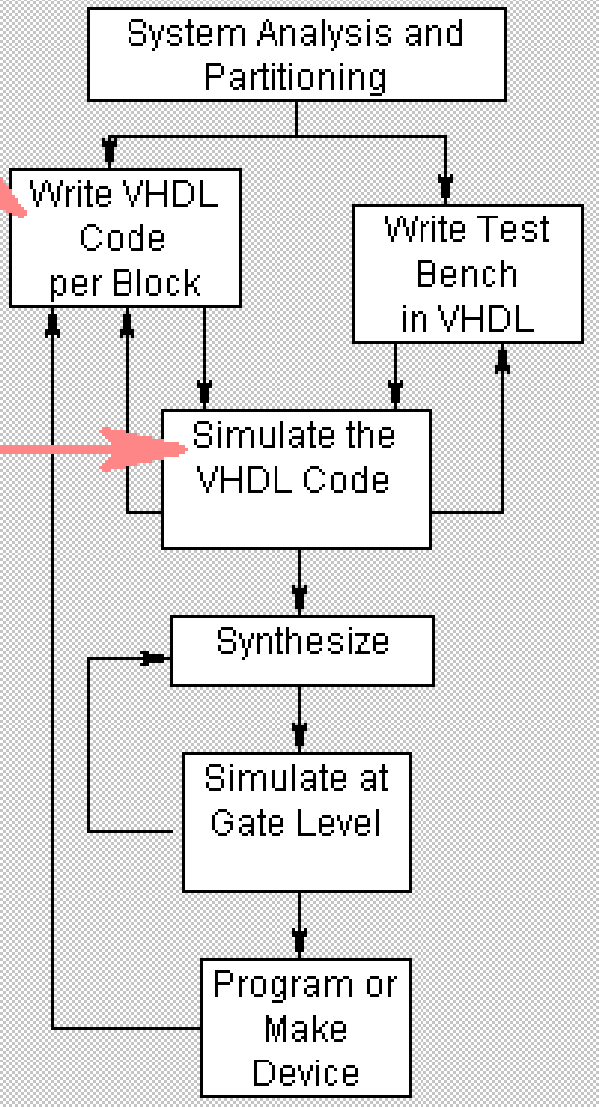
VHDL is not owned by any one company, so your investment is secure.

VHDL allows design proving before making or programming devices.

Synthesis automates gate level design.

VHDL is non-proprietary

Verify design before implementation



Benefits of VHDL

VHDL is now well supported by most ASIC and FPGA vendors.

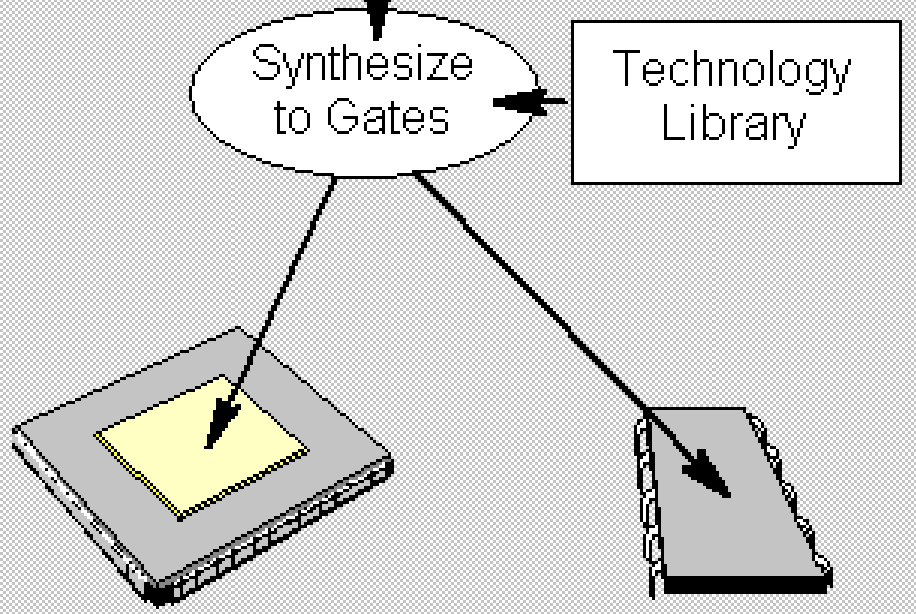
VHDL allows you to design in a technology independent manner. This allows easy migration from FPGAs to ASICs, and also design re-use.

So, VHDL protects you from changes in tools and technology.

Same VHDL code for ASIC or FPGA

```
process (Clock, Reset)
begin
  if Reset = '0' then
    Q <= (others => '0');
  elsif Clock'event and Clock = '1' then
    Q <= Q + '1';
  end if;
end process;
```

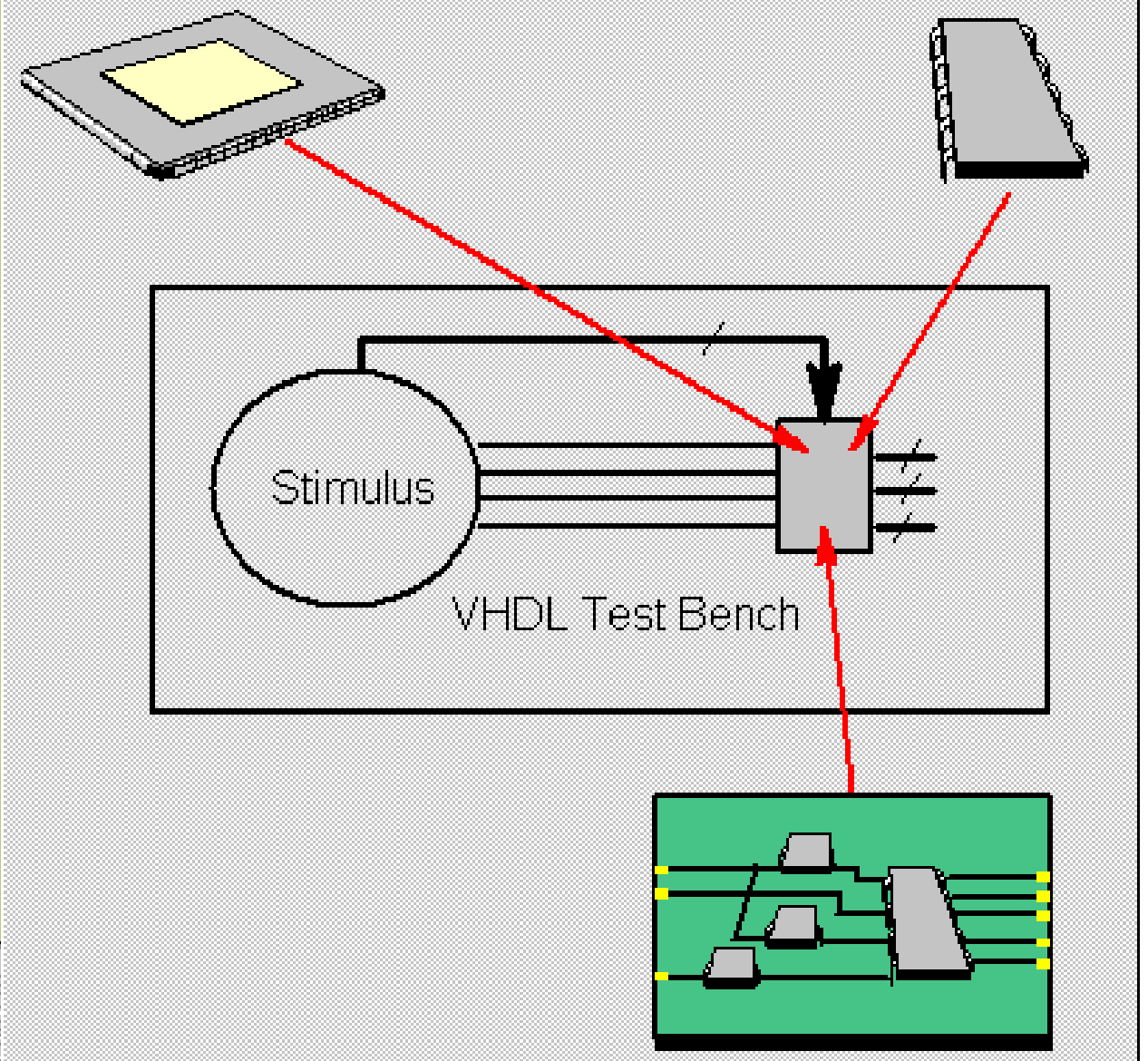
VHDL Code



Benefits of VHDL

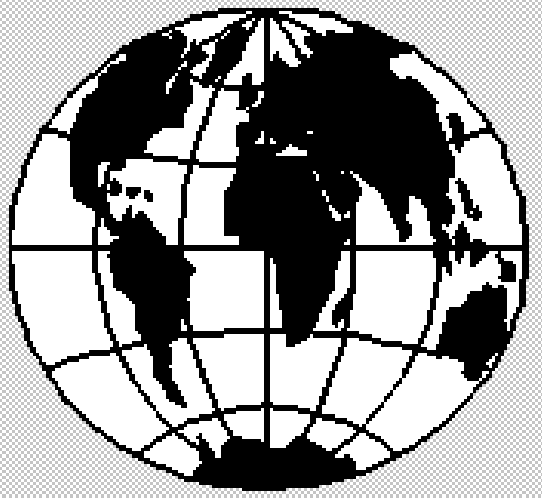
VHDL's programming language allows you to create powerful test benches to test your hardware components.

The test bench concept is very flexible. Test benches can be written to test parts of your system containing ASICs or FPGAs, or even to test complete circuit boards.



The VHDL World

Since the publication of the first VHDL standard in 1987, many other related standards have been required, various organizations have been set up to service VHDL users, and there is much Internet activity relating to VHDL.



IEEE Standards

VHDL Organizations

VHDL & Internet

The VHDL World

The IEEE is responsible for all standards relating to VHDL.

1076 is the VHDL Language Reference Manual.

1164 is a standard multivalued logic system which aids model interoperability. More about other standards later...

IEEE Standards

1076	VHDL
1076.1	Analog VHDL (Analog and Mixed Signal)
1076.3	VHDL Synthesis Package
1076.4	VHDL Timing Methodology (VITAL)
1164	VHDL Model Practices (Standard Logic)

The VHDL World

Through the Internet there is now access to a vast supply of information, and VHDL is no exception. The most asked question usually starts, "Where do I find out about ...?". The best place to start is the newsgroup comp.lang.vhdl and the web links shown opposite. Accellera and ECSI are both non-profit organisations serbing the EGA industry

Good surfing!

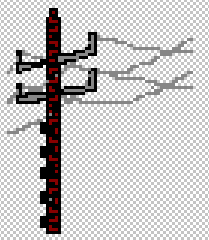
VHDL Organisations & Internet

Newsgroups

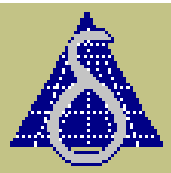
comp.lang.vhdl
comp.cad.synthesis

Web Sites of Interest

www.eda.org
www.accellera.org
www.ecsi.org



Doulos Web Page:
http://www.doulos.com/



Exercise 1

What do the letters RTL stand for?

- Register Transfer Level
- Resistor Transistor Logic
- Really Tough Language
- Register Transfer Language

Yes! RTL is a style of description which synthesis tools transform to gate level.

Exercise 2

Drag and drop the items from the top into the correct slots below.

Hint: First decide if each item is an input to or an output from synthesis.

Incorrect! This is a fragment from a gate level VHDL description. Try again

Load of external pin

Maximum delay path

if Rst then Reg<= 0;

ScanChain inserted

Area of a 2 input nor

Setup violation

Postscript diagrams

G1: nand2
G2: nor2

011101111
010111110

Flatten hierarchy

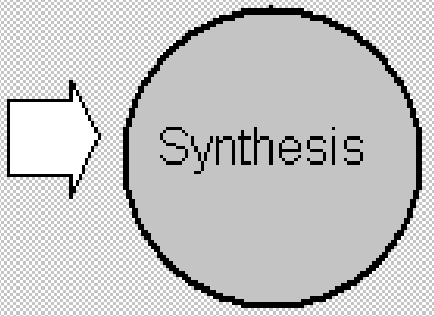
Synthesis Parameters

Target Library

Design Constraint

The Design

Boundary Conditions



Test Patterns

Output Schematic

Output Reports

Testability Changes

Gate Level Netlist

If you are not sure, have a guess!