EE-334
Digital System Design

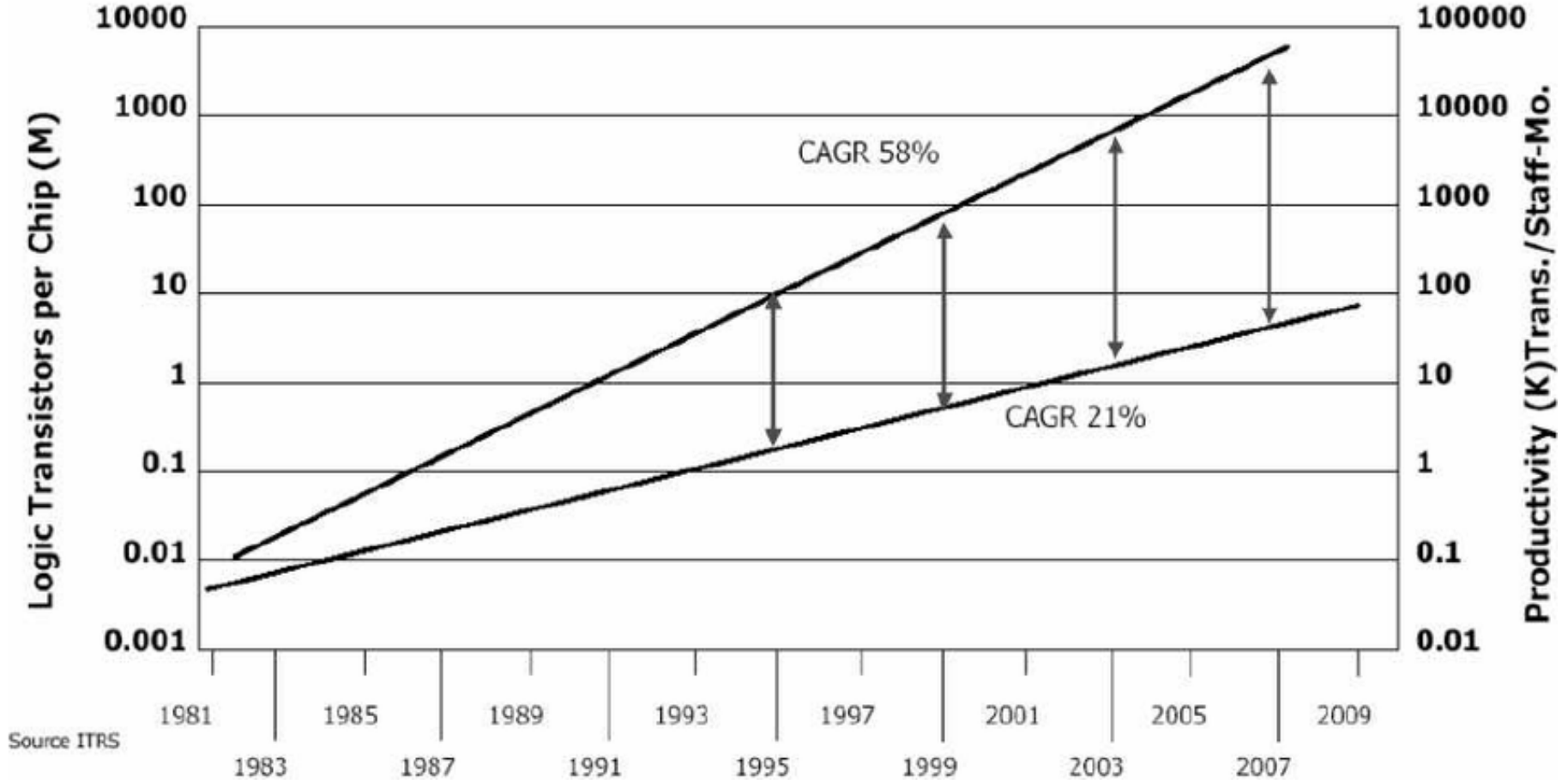
Custom Digital Circuits

Intro to Hardware Description Languages

Andreas Burg

Productivity Gap

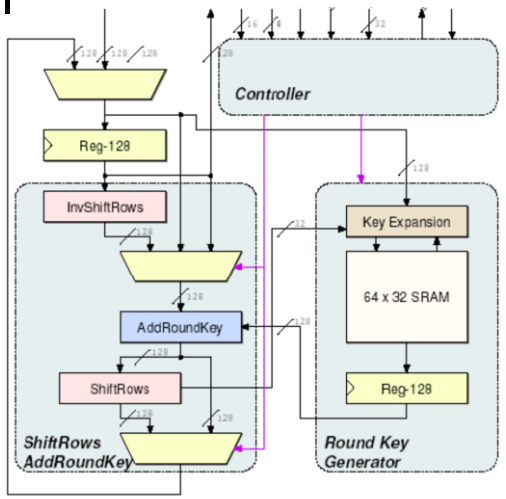
- **Early days of VLSI design: designers can not keep up with** the technology progress and **the available resources** (# of transistors)



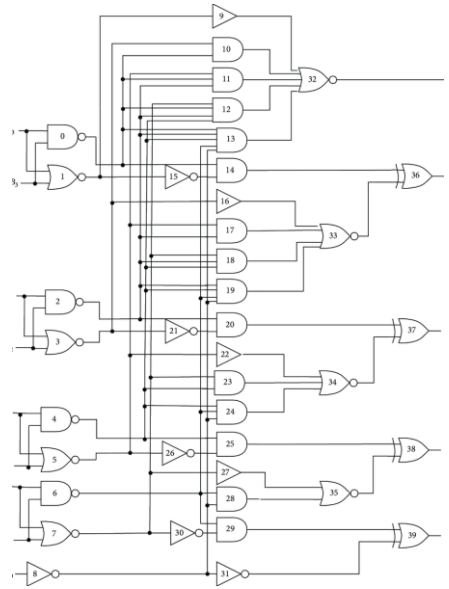
Need for Hardware Description Languages (HDLs)

- Recap: "**Digital Circuits are best explained with Block Diagrams**"
  So why do we need a Hardware Description Languages ??

- **Block diagrams** are great **for developing** and documenting **a high-level architecture**, **but** they…
  - **require abstraction to be readable** and meaningful
    - become chaotic once they include many details
    - are not well suited to describe low-level details
  - are **difficult to parameterize**
    - every change requires significant effort
    - limit re-use of generic components
  - **Can not** easily **express higher abstraction** levels



vs.

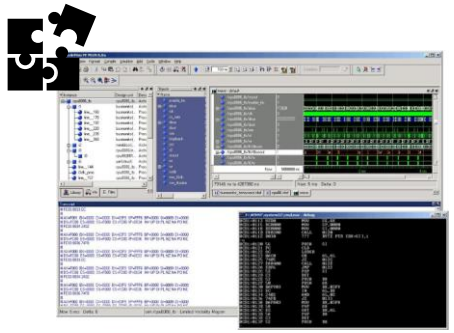


  - **Block diagrams** are **for high-level specification**, but **detailed (complete)** design **specification requires a** more flexible, more expressive, and **more capable formalism.**

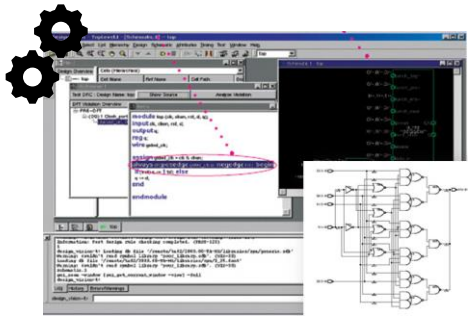
The Need for Hardware Description Languages

- A **formal language** was **required to describe hardware** more **efficiently** and more precisely **in all its details**.
  - Purpose is a **refinement of block diagrams**, **not a replacement**

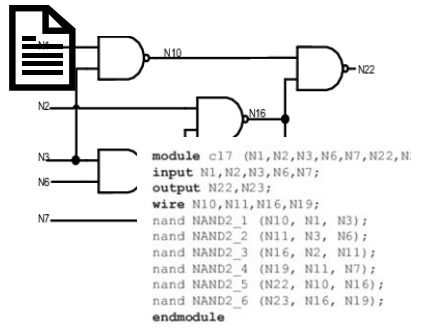
- **HDLs are the foundation for** the goal of **automating the design process** **with Computer Aided Design (CAD)** flows and **tools**.
- **CAD** for microelectronics is **referred to as Electronic Design Automation (EDA)**



Behavioural modelling for simulation



RTL design for automatic synthesis



Structural description of a design

Hardware Description Languages

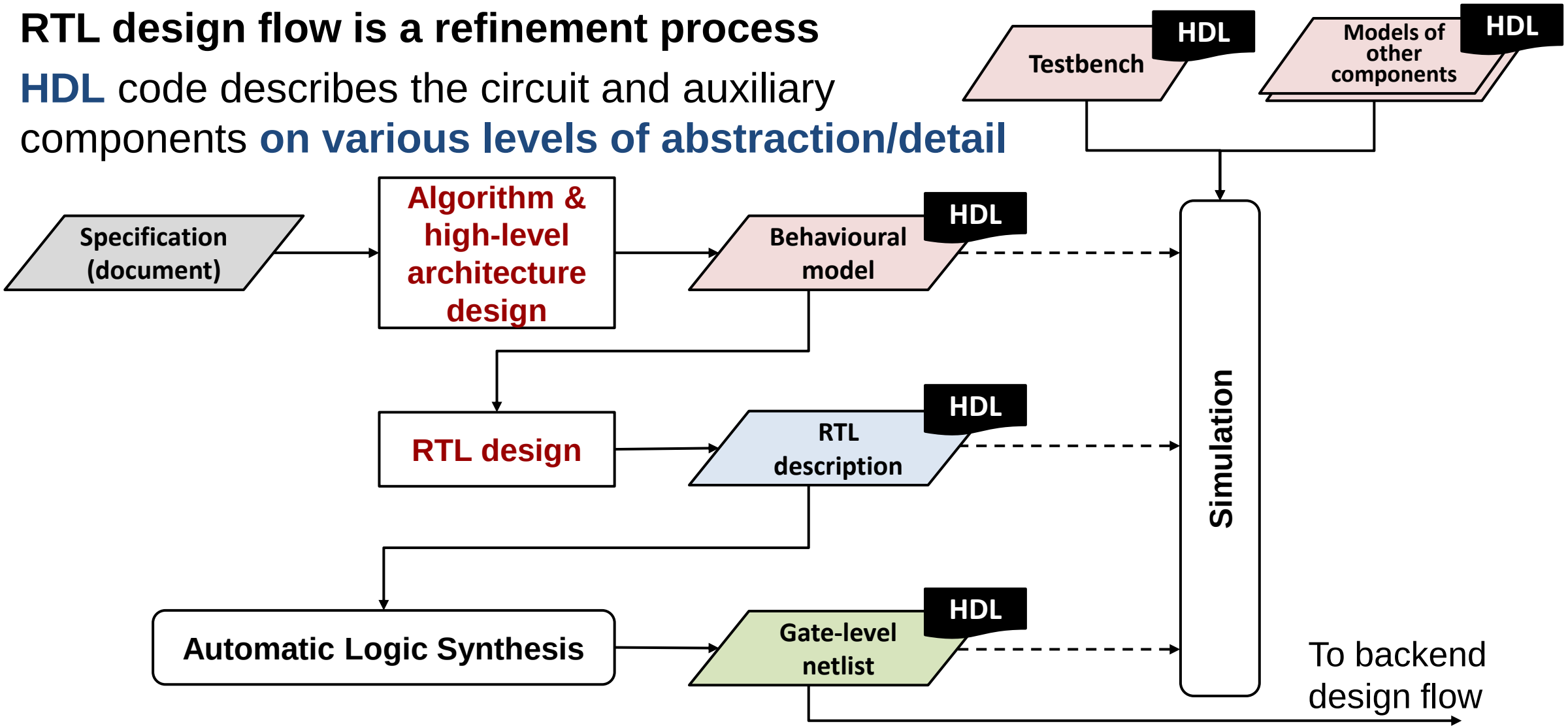
- **HDLs allow to describe parallel hardware on various abstraction levels**
  - Note the difference to most programming languages which struggle to describe parallelism

- The **development of** today's **HDLs started in the 1980s**.
  - State of CAD – generators and proprietary HDLs
  - VHDL was the first HDL standard (requirements in 1981, established in 1987)

- Mainly **three HDLs** (plus their off-springs) are **established today**:
  - **VHDL**: since 1987, used more in Europe, but also globally
  - **Verilog**: established in 1995 and merged with **System Verilog** in 2009
  - **System C**: established in 2005 to build on the success of C++ for SW

The Complete RTL Frontend Design Flow

- **RTL design flow is a refinement process**
- **HDL** code describes the circuit and auxiliary components **on various levels of abstraction/detail**

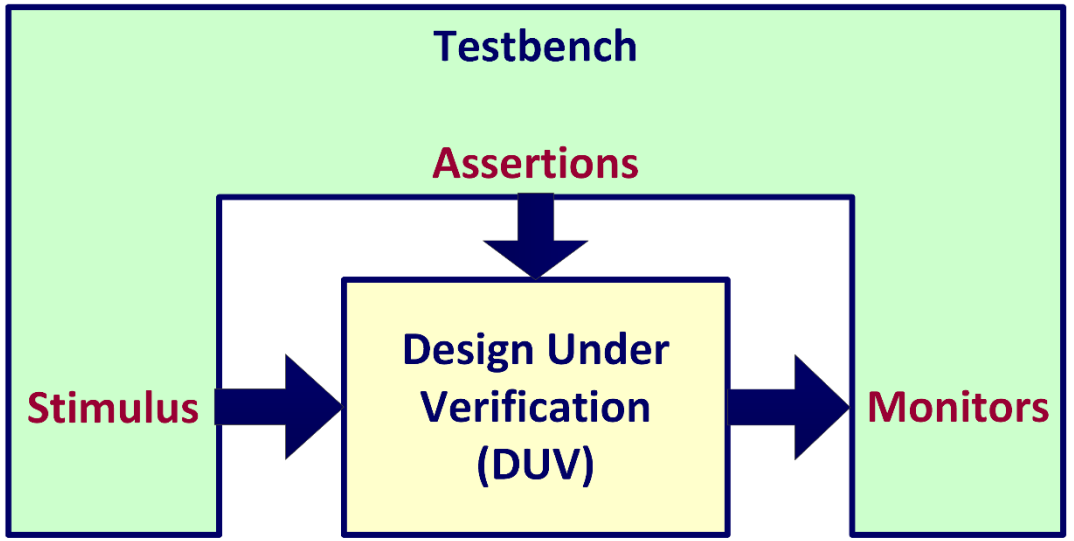


Modelling and Simulation

- **HDL** code can be **"executed" with an event-driven simulator**
  - Used for modelling and verification

- In addition to the Design **under Verification (DUV), simulations often involve also other HDL components**

- **Simulations are** usually **driven by TESTBENCHES** that
  - Instantiate the DUV
  - Generate stimuli as inputs to the DUV
  - Monitor and check the outputs of the DUV



RTL Design and Logic Synthesis

- Design on **register transfer level (RTL) describes only the RTL architecture**, but **not the details of the logic**.

- **Automatic tools perform**
  - **Analysis**: syntax check & translation of VHDL into some more easy to handle internal representation
  - **Elaboration**: expansion of parameterized constructs (e.g., multi-bit operators or loops)
  - **Logic synthesis & optimization**: derivation and optimization of Boolean equations
  - **Technology mapping**