# The Kernel Accelerator Device

*-reconfigurable computing for the kernel-*

Lecture held at 21C3 in Berlin

by

Ludwig Jaffé

www.openhardware.de

# Why do we want the KAD?

- Many things we like are too slow
  - drive encryption
  - cryptoanalysis
  - DSP-functions for video-transcoding ....
- How to solve computing-intensive problems?
  - The conventional way:
    - throw more mips at the problems
      - and transform yor computer room into a sauna club -
  - The smarter way
    - use reconfigurable hardware ...

# What is the KAD?

- The KAD is
  - a *reconfigurable computer subsystem*.
  - user-configurable to do computing intensive jobs in hardware
  - as easy to use as a kernel module in linux
  - a piece of open source hardware
- The KAD consists of
  - PCI-Card with FPGAs
  - compiled vhdl (and/or verilog) code which does certain jobs
  - special kernel modules to make the KAD useable for the masses

# How does it work?

1. The user loads a special kernel module she wants to use.
   (e.g. loop-AES with KAD-Support)
2. The kernel module sets up communication to KAD via PCI
3. The kernel module configures the KAD to do the dirty job with firmware precompiled from open-source hdl-code.
   (e.g. it loads an AES-IP-Core with some glue logic into the FPGAs)
4. The kernel module does the dirty job by mostly transfering data between other kernel components and the KAD
   (using memory mapped IO this means quite low CPU-load.)
5. The user unloads the special kernel module
   (the module wipes the keys -if any- inside the FPGAs and erases them)

# What does reconfigurable computing?

## General-purpose vs. reconfigurable computers 1/2

- General-purpose computers:
  - program a fixed component (CPU) to do the things.
  - step in *sequence* through a set of instructions in the dimension of **time:**

    Example: Adding two numbers using a CPU:

```
MOV     A,23        ;Load Register A
MOV     B,42        ;Load Register B
ADD     A,B         ;The sum is in Register A
MOV     Result,A    ;Store the Result
```

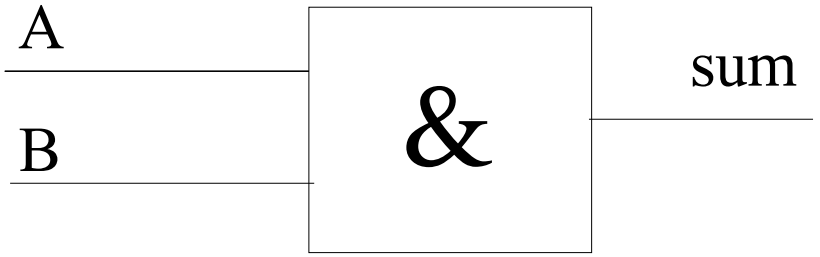
    If we want to add (a+b and c+d) we need to execute the program twice which means that we need *twice the time*.

# What does reconfigurable computing?

## General-purpose vs. reconfigurable computers 2/2

- Reconfigurable computers:
  - program a programable component (FPGA) to do the things.
  - compute using configued functional units and interconnects.
  - compute in *parallel* specific, configured operations in the dimension of **space.**

Example: Adding two numbers using a FPGA:




If we want to add (a+b and c+d) the same time we need two adders, which means that we need *twice the space*.

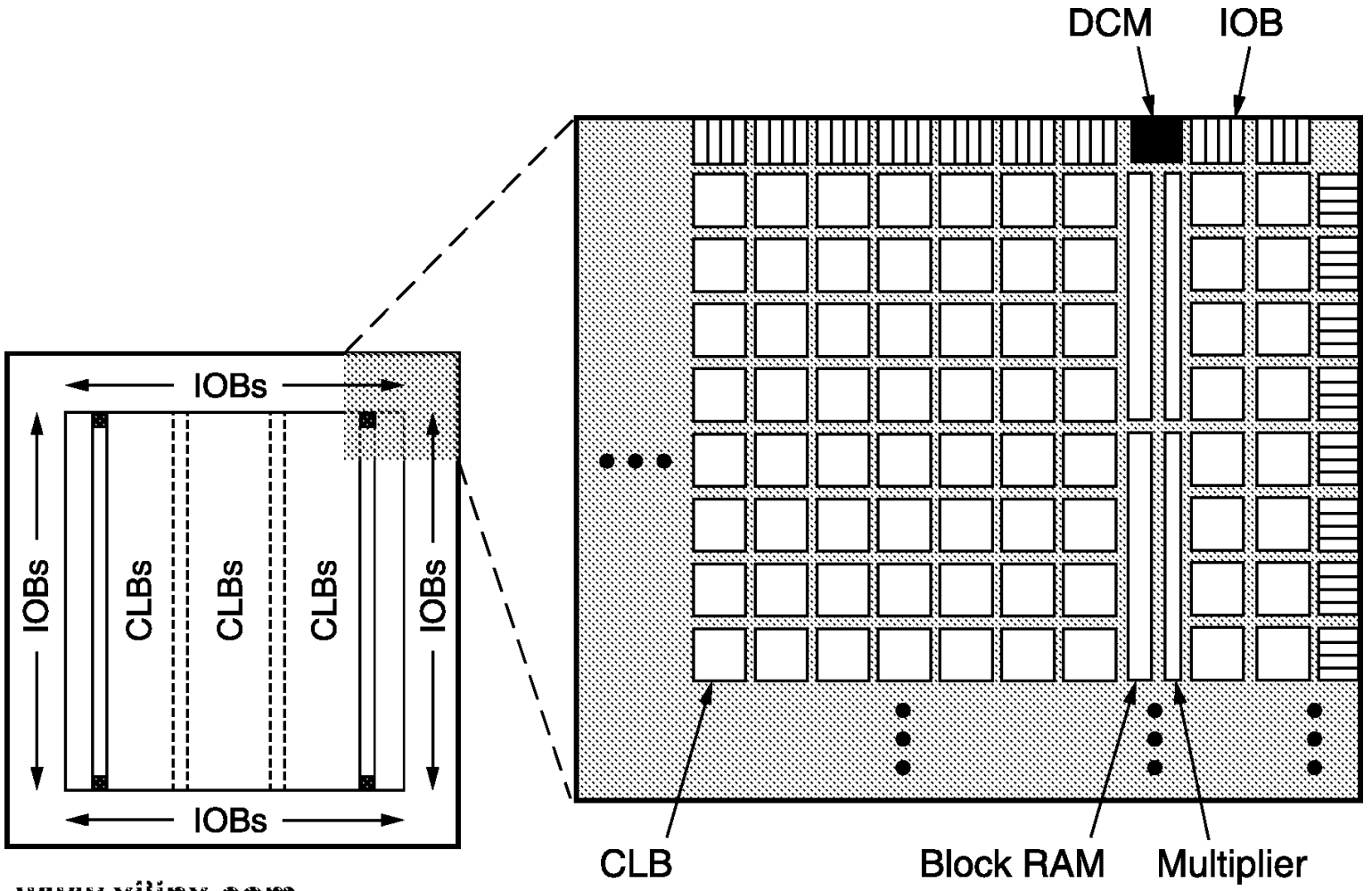
# What is so special about FPGAs?

FPGA

- means **F**ield **P**rogramable **G**ate **A**rray
- consists of
  - Logic Blocks (the idea is similar but the implementations differ slightly)
    - called *Logic Elements (LE)* by Altera
    - called *Configurable Logic Blocks (CLB)* by Xilinx
  - programable Routing fabric
  - universal I/O-Cells
  - some extras such as PLLs, dedicated RAM, dedicated functions (e.g. Multipliers)

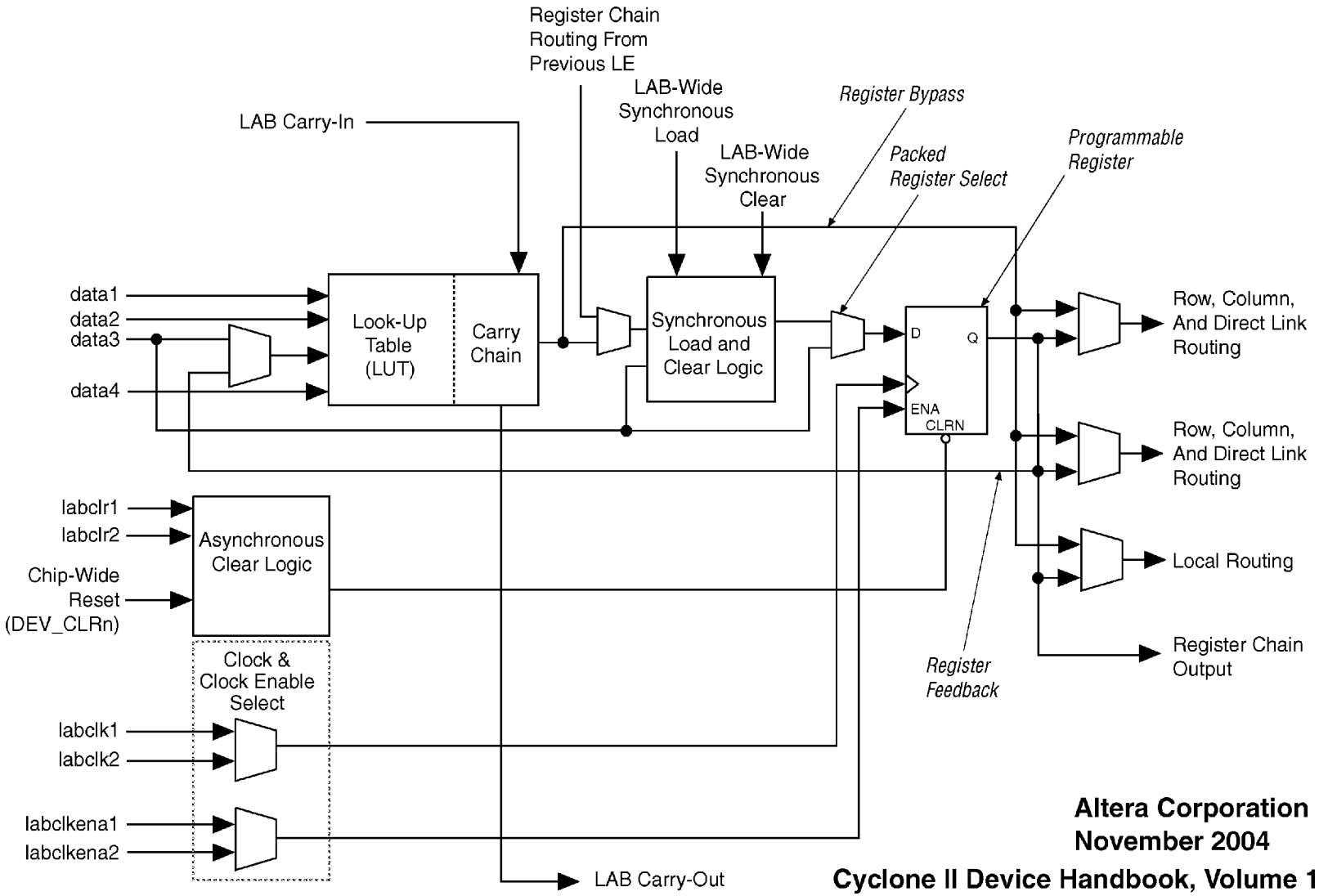
# What is so special about FPGAs?

## Structure of a FPGA



# What is so special about FPGAs

## What does a Logic Element look like?

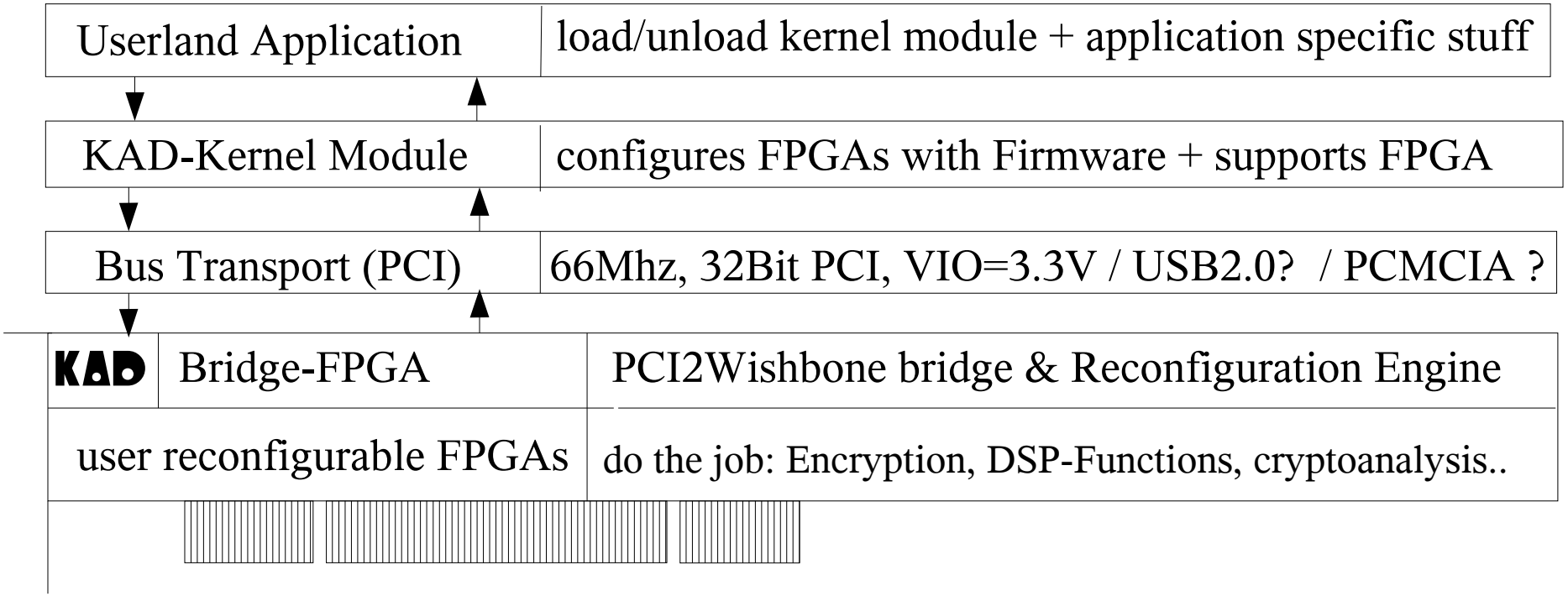



**Altera Corporation**
**November 2004**

**Cyclone II Device Handbook, Volume 1**

# The Architecture of the KAD <1/4>

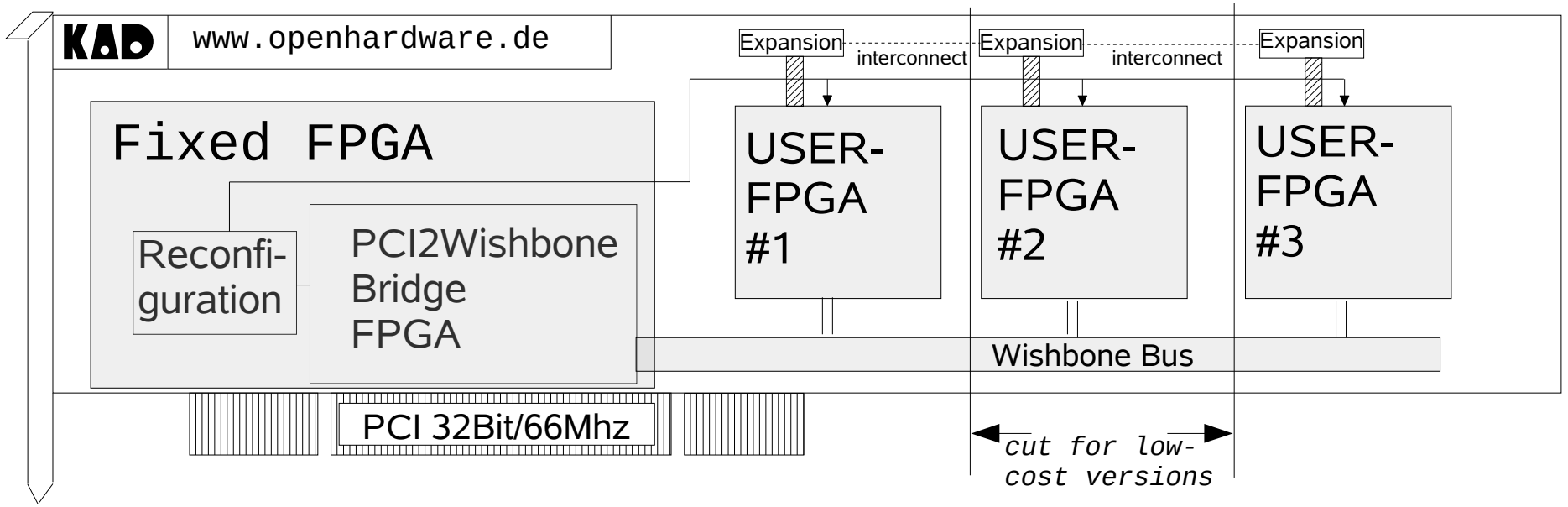
## The System-View



# The Architecture of the KAD <2/4>

## The Hardware-View

- pci2wishbone bridge
- wishbone SoC-Bus
- reconfiguration of SRAM-based FPGAs
- Expansion Ports
  - possible Interconnects using plugs and flat cable
  - expansion modules HW-Random, SRAM etc.



# The Architecture of the KAD <3/4>

- The Operating System -View
  - Userland interface
    - depends on intended application
  - KAD-Kernel modules
    - include pre-compiled VHDL as firmware
    - have firmware loader to configure the FPGAs
    - support their firmware inside the FPGAs
    - initiate and coordinate data transfers (memory mapped IO via PCI)
    - interface to other kernel modules and/or userland

# The Architecture of the KAD <4/4>

- The VHDL-View
  - Use of IP-Cores
    - provides fast time to “market” and lowers risks
    - open-source IP-cores are combined to form a system
    - additional vhdl-code does the needed interfacing and some special stuff like reconfiguration
  - How to compile, fit and simulate?
    - We use no-cost tools by the FPGA vendors (Altera vs. Xilinx)
  - VHDL-Versions are treated as part of the kernel- modules to prevent chaos and frustration

# Open Source Hardware

- Design Reusability
- Resources about Open Source Hardware
  - www.opencores.org (IP-Cores)
  - www.openhardware.de (KAD-Project)
  - www.opencollector.org (Info about open source HW)
- EDA-Tools
  - GNUeda vs. free closed source eda

# How can I participate?

- The KAD-Project needs
  - Architects to define the datailed KAD concept (20% done)
  - Hardware Designers (5% done)
  - VHDL-coders and integrators (0% done)
  - Kernel module coders (0% done)
  - Ideas for new applications (never done)