TKDM – A Reconfigurable Co-processor in a PC’s Memory Slot

Christian Plessl and Marco Platzner
Computer Engineering Lab, ETH Zurich, Switzerland
December 17, 2003
FPT’03, Tokyo, Japan
Motivation

- FPGA based co-processors promise high raw-speedup for many application kernels
- Effective performance often moderate
  - *bandwidth* limitations of IO bus
    → limits throughput
  - communication *latency*
    → communication overheads
  - one *kernel* does not make an application
    → system-level impact ≠ raw speedup
Coupling of FPGA based co-processors within PCs
Memory vs. IO interface speed

- SDRAM DDR 400MHz: 3200 MByte/s
- SDRAM SDR 133 MHz: 1064 MByte/s
- SDRAM SDR 100MHz: 800 MByte/s
- PCI 64/66MHz: 528 MByte/s
- PCI 32/33MHz: 132 MByte/s
- USB2.0: 60 MByte/s
- FireWire400: 50 MByte/s

*Raw speed [MByte/s]*
Observation / Idea

• Every balanced computing system must have a fast memory interface
  – memory module interface is standardized, e.g., SDR/DDR SDRAM (DIMM), RAMBUS
  – memory interface is always one order of magnitude faster than fastest IO interface

• Attach IO intensive devices to memory bus
  → high-bandwidth, low-latency communication
Design goals for TKDM

- FPGA co-processor board
  - attached to DIMM memory bus (dual-inline memory module)
  - state-of-the-art FPGAs
  - application domain: streaming data processing
- on-board RAM
  - for data and configuration buffers
- fast (re-)configuration
  - via DIMM bus
  - full configuration, partial configuration, and readback
- system integration
  - integration in Linux (drivers and user libraries)
  - firmware support for streaming applications
  - hide complexity of memory bus protocol from user
Related work

• Pilchard [Leong et. al, Chinese Univ Hong-Kong, FCCM’01]
  - based on single Virtex / Virtex-E
  - configuration via external JTAG connection
  - no on-board RAM
  - used in numerous projects, mainly for cryptography

• Nuron AcB
  - product announcement
    • single FPGA, 64MB on-board SDRAM
    • configuration and data transfers via DIMM bus
  - company was acquired by Intel in late 2001
Hardware architecture

• Challenges
  – FPGA feed through delay is too high connect on-board SDRAM directly to DIMM bus on-demand
  – Configuration and data transfers take place over DIMM bus
  – Prevent OS from initialzing the TKDM module
TKDM hardware design

- 6 layer, impedance controlled PCB board, 13x9cm (2.5x height of normal DIMM module)
- FPGAs: AL-FPGA: XC2V1000, T-FPGA: XC2V1500
- 4 SDRAMs, 133MHz, 16MByte, 32bit width
TKDM hardware architecture (1)
TKDM hardware architecture (2)

- abstraction layer FPGA (AL-FPGA)
  - Virtex2 XC2V1000 FG456
  - configured from EEPROM
  - memory controller
    - handle timing sensitive DIMM protocol
    - DMA controller for on-board SDRAMs
    - provide simple interface to on-board SDRAM for T-FPGA
  - configuration controller:
    - partial reconfiguration and readback of T-FPGA (SelectMAP)

- target application FPGA (T-FPGA)
  - any Virtex2 in FG676 or FG456 package
  - implements application
  - uses simplified interface for SDRAM access provided by AL-FPGA
TKDM hardware architecture (3)

- on-board SDRAM
  - 4 banks of SDRAM
  - connected to AL-FPGA or T-FPGA
  - switch implemented by high-speed bus-switches
  - not directly accessible via DIMM bus

- DC/DC converters
  - TKDM can be powered via DIMM bus, or via connector
  - generate FPGA core and IO voltages
TKDM soft and firmware architecture

- Reuse firmware functions for a whole class of applications
- Layered design
  - *fixed function* soft and firmware functions
  - *domain specific* sw libraries and firmware
  - *application specific* software and associated hardware (cores)
Layered services: fixed function

- fixed function firmware and kernel drivers
  - configuration services
    - configuration of T-FPGA (full, partial)
    - verification: readback
  - memory services
    - bus-switch control
    - controller for on-board SDRAM
  - status services
    - initialize / reset firmware
    - provide access to firmware status for CPU (buffer fill levels, configuration status of T-FPGA, firmware version, etc)
  - kernel functions only
    - no direct access for applications
Layered services: domain specific

- domain specific firmware and libraries
  - domain definition
    - applications that have similar data access patterns
    - e.g., processing of streaming data, video processing, etc.
  - firmware functions
    - data pre-processing, packing/unpacking
    - buffer management
    - simplified data access for T-FPGA
  - user space libraries
    - export a domain-specific subset of kernel functions to applications
    - simplify programming due to restrictions imposed by domain
Integration with PC and Linux

- memory controller tries to setup RAM at boot time
  - TKDM module does not identify itself as a RAM module at boot time to prevent initialization
  - later, the memory controller maps TKDM into Linux address space
  - caching disabled to prevent consistency problems
  - all accesses to TKDM using memory mapped IO
- driver implemented with Linux kernel modules
  - fixed functions module, domain-specific modules
Example for domain-specific firmware: streaming data processing

- application, e.g., encryption of data streams
- interleaved processing for streaming applications
- hide processing latency
Implementation: domain-specific firmware: streaming data processing
Evaluation

• First benchmarks on Pentium 3 500 MHz mainboard (i840), 100MHz SDRAM clock
  – 128 MB/s write to TKDM
  – 53 MB/s read from TKDM
• Significantly slower than maximal throughput of 800 MB/s (6.25x, 15x resp.)
• Caused by ‘pessimistic’ behavior of mainboard in the case of deactivated caching
  – SDRAM only fast if bursts are used
  – Memory controller issues bursts of 4 words but uses only first word if caching is deactivated
  – enabling caching increases performance
Status

- TKDM board implemented
- Fixed function firmware running
- Domain specific firmware and API for streaming under development
- Performance below potential due to mainboard’s memory controller
Ongoing and Future work

• improving communication performance
  – porting firmware to Pentium4 mainboard with 133MHz DIMM memory clock
  – use finer cache control provided by Pentium4 (try to avoid turning of caching completely)

• applications
  – complete data streaming software and according libraries
  – implement case-studies to get real-world system-level performance data
Questions

• Feel free to ask questions!

• Further information:

  Christian Plessl
  plessl@tik.ee.ethz.ch
  http://www.tik.ee.ethz.ch/~plessl