Architectural Considerations for CPU and Network Interface Integration

C. D. Cranor; R. Gopalakrishnan; P. Z. Onufryk

IEEE Micro
Volume: 201 , Jan.-Feb. 2000 , Page(s): 18 -26

Advisor: Ing-Jer Huang
Student: Ming-Chih Chen
What’s the Problem?

- Due to the demands for internet and consumer appliances.
- Communication processor: integrates processing, networking, and system functions into a SOC.
- The primary challenges are minimizing cost and time to market.
- Reducing on-chip buffer is also to decreasing die size of a network processor.
Design Approaches

- Complex DMA controller and buffer management unit are the important considerations of designing network equipments, such as NIC, and network processor.
- UNUM supports extremely fast event processing and high performance data movement.
- The design of a communication processor often integrates IPs by licensing standard CPU and network interface cores.
- A multi-channel DMA design is both complex and time consuming.
• Only one DMA channel can be active at a time.
• A DMA state machine is used to store the state information associated with each DMA channel (SA, DA, byte count, and descriptor pointer).
• When a DMA channel becomes active, the controller transfers its state from RAM to the state machine. After the operation completes, the controller writes back the updated channel state to RAM.
Design Approaches (Cont.)

- The DMA controller must often be designed from scratch due to highly system dependencies.
- These system dependencies include:
  -- 1. Performance requirements
  -- 2. On-chip bus architecture
  -- 3. Memory controller design
  -- 4. The type and number of supported network Interfaces
Design Approaches (Cont.)

- Due to the transfer requirements of diverse DMA control and status information, channel customization interface-specific processing is used for data movement.
- Due to the complexity of a multi-channel DMA controller, a dedicated processor replaces data transfers of DMA’s.
- A communication processor for low-cost consumer applications should contain a single processor that performs all data movement, interface-specific processing, and application processing.
Multithreaded CPU for Event Processing

- Minimizing on-chip buffers results in small bus transfer thus increasing the number of events.
- Current processors service external events, using either polling or interrupts.
- A major component of interrupt latency is saving and restoring the state of the interrupted context.
Multithreaded CPU for Event Processing (Cont.)

- Techniques used to reduce this overhead include:
  -- coding interrupt service routines in assembly language to use a small number of registers,
  -- switching to an alternate register set for interrupt processing,
  -- saving processor registers to unused floating-point registers, and
  -- providing on-chip memory for saving and restoring state.
Multithreaded CPU for Event Processing (Cont.)

- UNUM consists of three major components:
  - An event mapper: used to initiate event service routine execution.
  - A context scheduler: used to issue instructions to the CPU pipeline.
  - A CPU pipeline: used to support concurrent execution of instructions from multiple contexts.
- \(31n \times 32\) register file: \(n\) is the number of supported hardware contexts.
- UNUM employs a traditional single-issue RISC pipeline and multithreading which used to reduce event service latency.
Data Movement Instructions

- Programmed I/O (PIO) generates memory-to-memory transfers that require twice the bus bandwidth of fly-by DMA operations.
- PIO operations:
  -- Non-cacheable loads and stores result in single-word data transfers that achieve poor bus utilization.
  -- Cacheable loads and stores that generate burst transfers result in data cache pollution.
Data Movement Instructions (Cont.)

- IBIU (Internal Bus Interface Unit) incorporates a data mover and aligner that segments the on-chip bus into a memory bus and an I/O bus.
- Data movement operation occurs:
  -- Issue data movement instruction.
  -- Either stall the CPU pipeline until the operation completes or allow the pipeline to continue execution from on-chip caches as long as they are no misses.
Fly-by transfers: data bypasses the data cache and does not pollute it.

TM2D: transfers data from memory to an interface.

TD2M: transfers data from an interface to memory.

To maintain cache consistency, the cache supplies dirty data during TM2D processing, and performs cache invalidates during TD2M.
Data Movement Instructions (Cont.)

- TD2C: loads data directly into the data cache from an interface, eliminating an unnecessary transfer through memory.
- TM2DD: discards dirty data from the data cache as it is written to a network interface, potentially eliminating an unnecessary future write-back.
Aligner

- Unaligned data transfers from memory to I/O device.
- The aligner uses a holding register, shifter, and multiplexer to align data as it flows from one bus to the other.
Simulation Results

Simulation constraints:

-- Cycle-accurate simulator of a UNUM-based communication processor.
-- 200-MHz UNUM processor.
-- Instruction cache: 8-Kbyte, two-way set-associative data cache.
-- Data cache: 2-Kbyte, two-way set-associative data cache.
-- 32-bit system bus, 100-MHz SDRAM.
-- Benchmarks were written in C.
Data Movement

• Three UNUM configurations: UNUM (hardware context switch with state preservation), fast interrupts (alternate register set with no interrupt state preservation), and normal processor interrupts.

40 Mbytes/sec, the cost of PIO dominates all other overheads.

The state of the service routine fits within a UNUM context, no state information needs to be loaded from memory.

Data cache miss had little effect.

40 Mbytes/sec, the cost of PIO dominates all other overheads.

Figure 5. Data movement performance.
ATM Soft-SAR

- ATM (AAL-5) Segmentation and Reassembly (SAR):
  -- Datagram, connectionless, variable transfer rate, unreal-time transfer service, such as FTP, and Telnet.
  -- Transmit and receive processing like MAC.
- The SAR software uses three hardware contexts:
  -- ATM receive event processing
  -- ATM transmit event processing
  -- ATM transmit cell scheduling
ATM Soft-SAR

Transmit throughput is lower because of the extra overhead associated with cell scheduling.

Figure 6. Maximum ATM SAR throughput.
Conclusion

- UNUM communication processor:
  -- 32-bit RISC processor (Single-issue).
  -- 31n x 32 register file, where n is the number of supported hardware contexts.
  -- Data movement instructions replace DMA processing.
    (Fly-by processing)
- Low cost and time-to-market are the primary challenges of SOC design.
- The trend of Net+ Processor is irresistible.