Efficient Communication Mechanisms for DSP-based Multimedia Accelerators

Robert Baumgartl, Hermann Härtig
Department of Computer Science
Dresden University of Technology, Germany

{robert.baumgartl, hermann.haertig}@inf.tu-dresden.de

DSPs as Multimedia Accelerators

DSPs are designed to:

- provide vast amount of processing power
- handle continuous data streams
- operate under real-time conditions

➡ **ideally suited for Multimedia algorithms**

DSP Plug-In Board Interfaces

Past

<u>ISA interface:</u>

- 8 bits wide, 8 MHz
- no DMA

- DSP plug-in boards as development systems only
- small amount of data between host and DSP
- no real-time demands

Present

<u>PCI bus standard:</u>

- 32 or 64 bits wide, 33 MHz
- Busmaster DMA

- high sustained data transmission rates possible
- new application areas of DSP boards: accelerating host applications

Lessons learned from a first prototype*

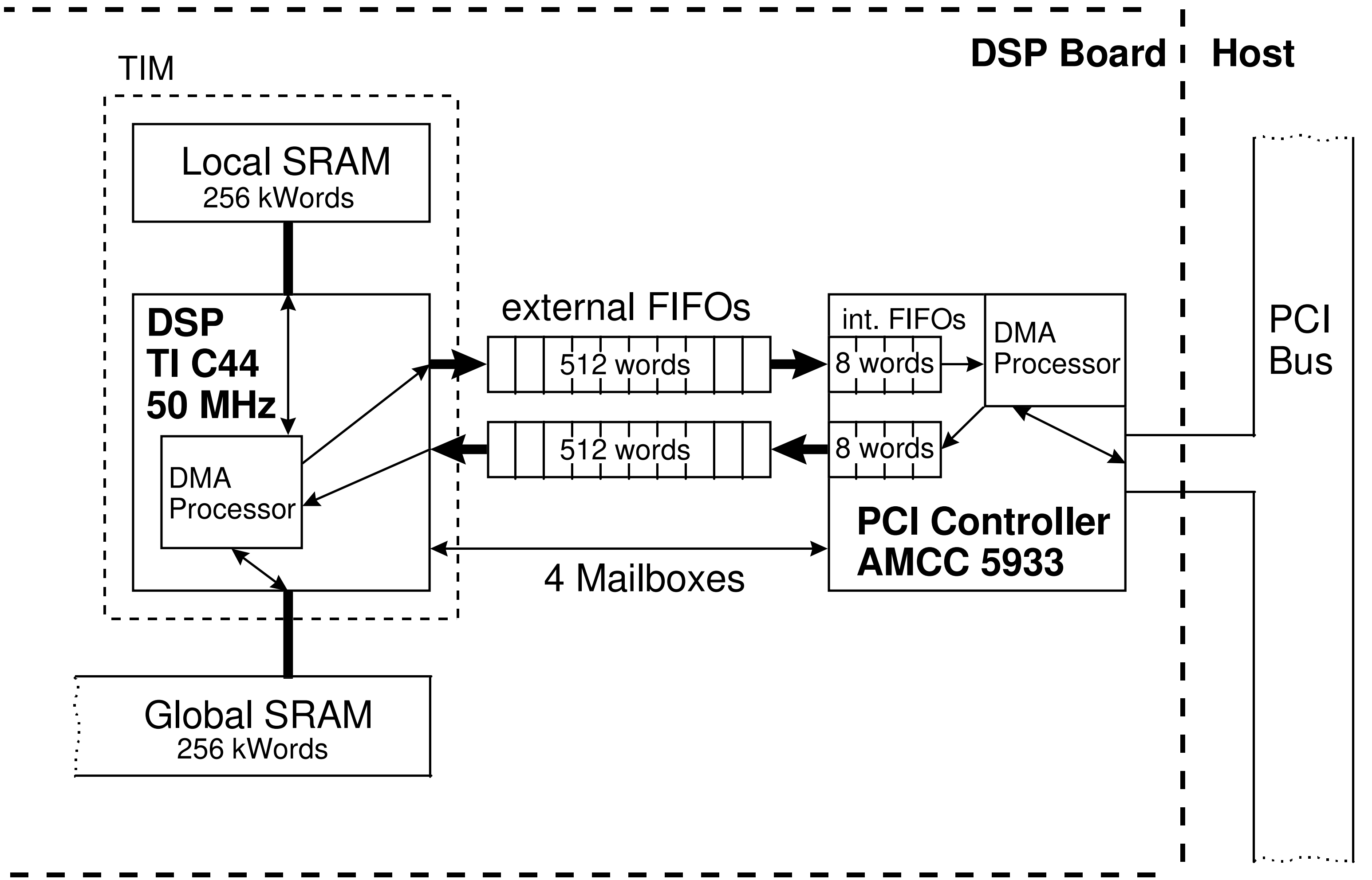
Acceptable solution requires:

- hardware interface as fast as possible
- inter-processor-communication services as efficient as possible
- DSP's processing power at least equal to the CPU's
- Host's operating system must provide mechanisms for data exchange between DSP processes and host applications

---

* Baumgartl, R; Härtig, H.: On the Integration of DSP Hardware into a Microkernel-based Operating System, Proc. ICSPAT'96, Boston, 1996

Hardware Configuration



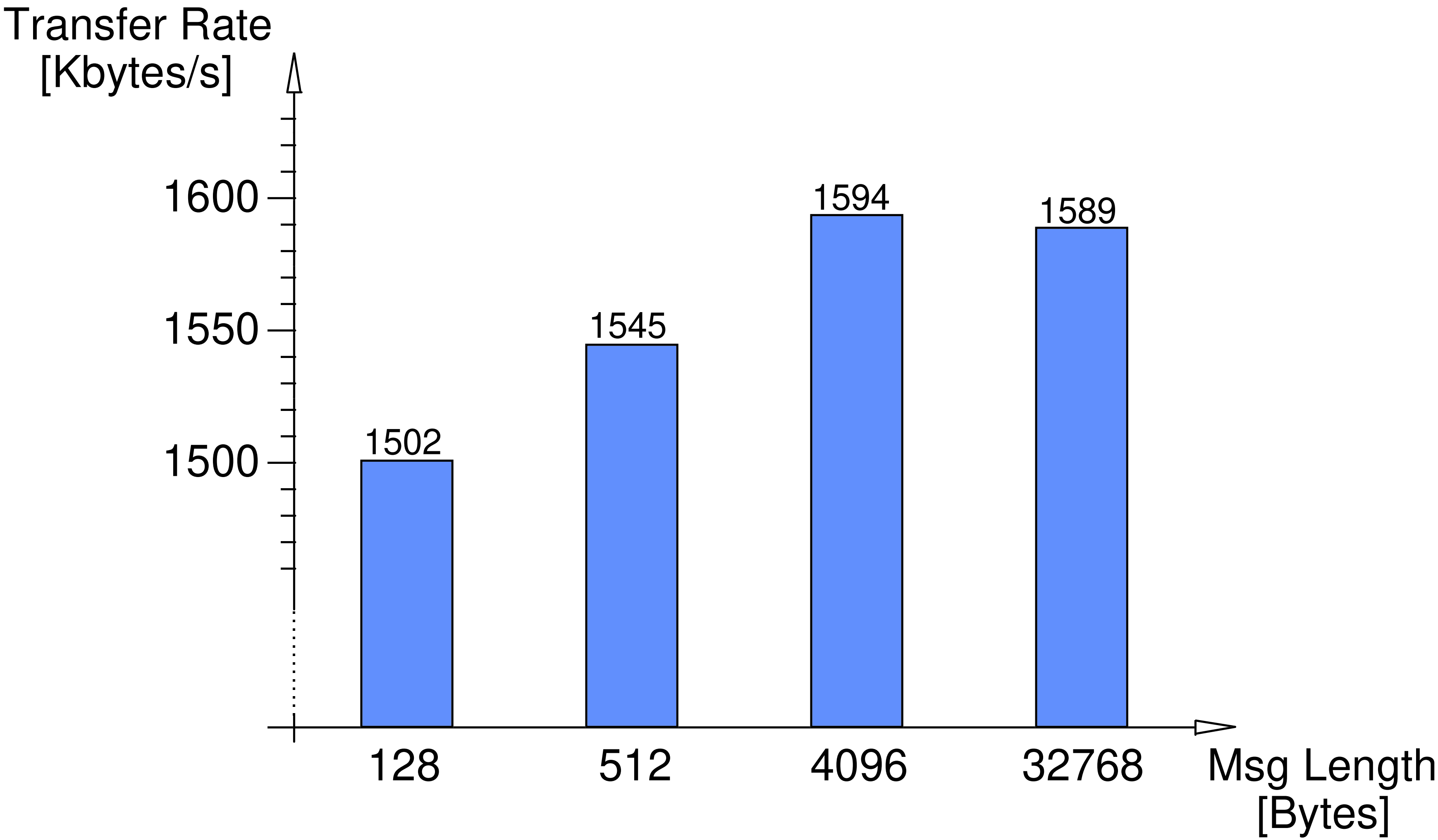
- Pentium Pro 200
- 64 MB RAM
- Chipset: Natoma
- PCI

- Linux

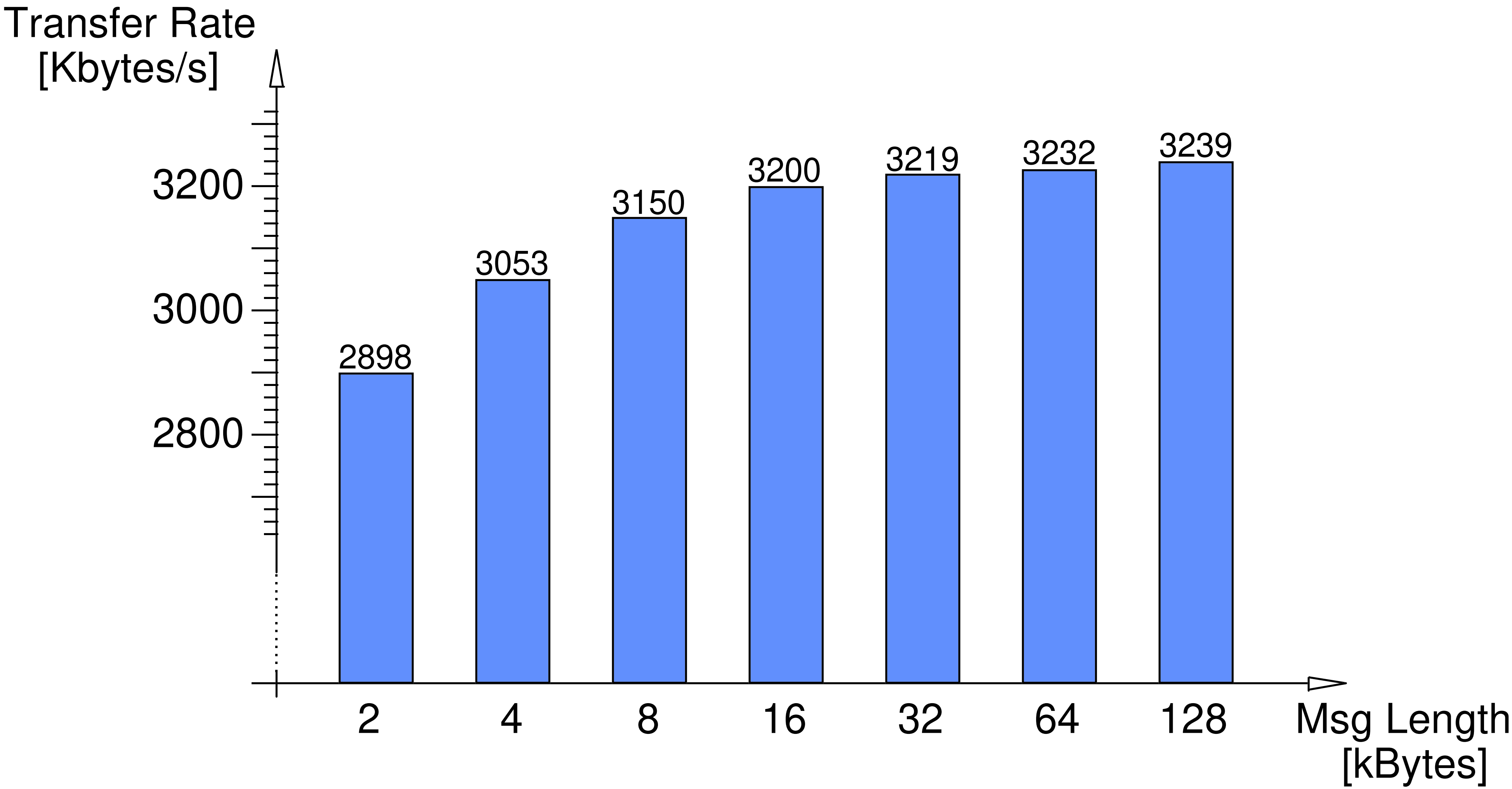
Hardware Mechanisms for Communication

Mailboxes:	Shared Memory:	DMA:
• parallel interface • individual 8-32 bits wide memory words • synchronization via Semaphore flags (HW) • CPU-intensive • slow • timing exactly predictable • cheap hardware	• 2-64 kBytes dual-ported RAM • synchronization via Mailboxes or Semaphore flags • CPU-intensive • fast • timing exactly predictable • expensive hardware	• autonomous transfer • bus system-dependent • small CPU load • very fast • timing hardly or only inexactly predictable • transfer rate dependent from bus load • expensive hardware
achievable: 1.5 Mbytes/s (32 Bit)	*no transfer rate measurements done*	*theoretical limit: 132 Mbytes/s (PCI)*

Mailbox communication performance



Busmaster DMA performance (unoptimized)



Optimization

Mailboxes: ca. 1.5 MBytes/s

⇩

DMA: ca. 3.2 MBytes/s

but ...

PCI theoretical maximum transfer rate
= 132 Mbytes/s ??

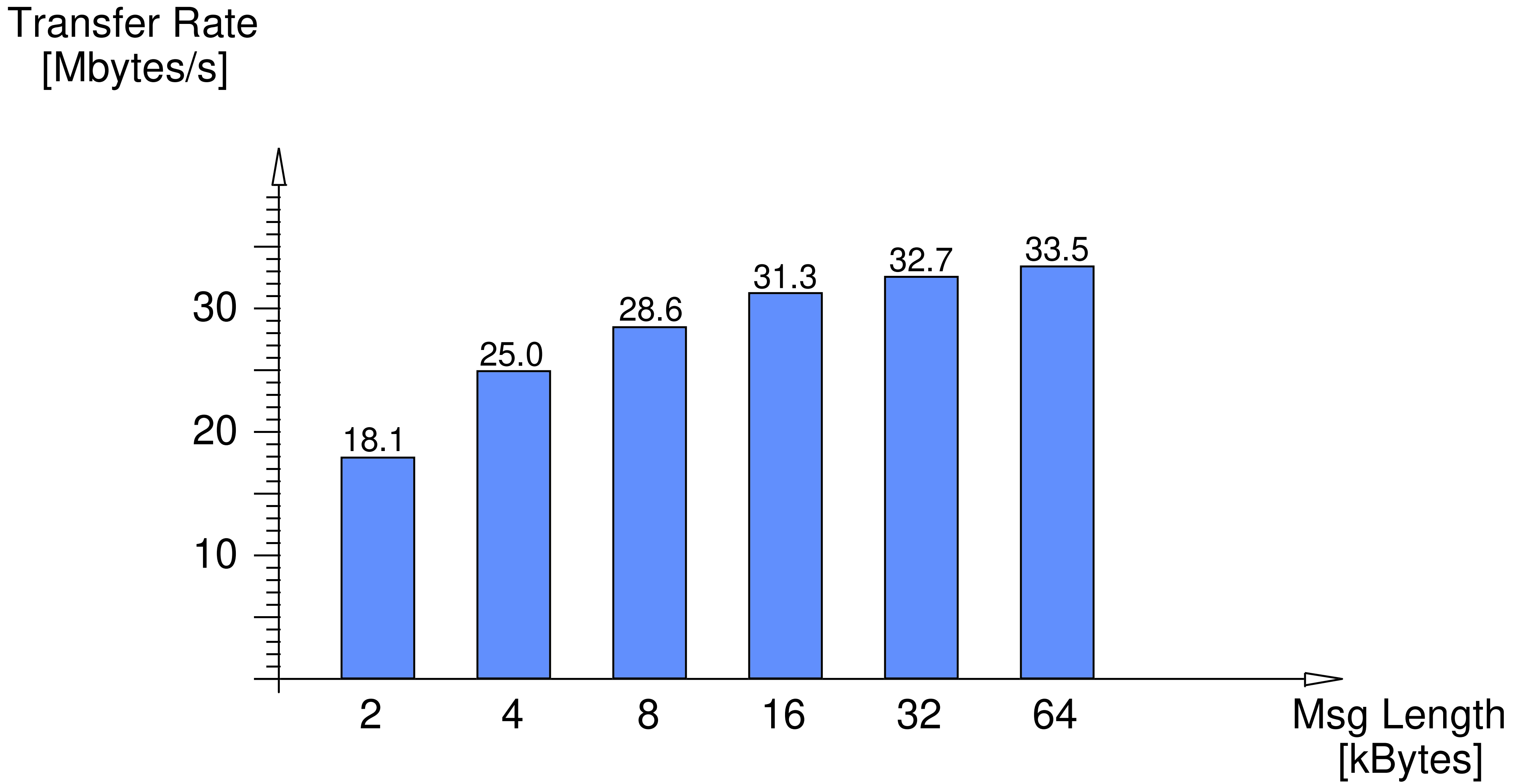
Analyzing and optimizing:

1. switch from C to Assembly language
2. optimize register usage
3. remove uunnecessary code out of loop
4. employ hardware looping
5. exploit Harvard architecture
6. instruction scheduling

Result:

Innermost transmission loop reduced from **15 to 1 cycles**.

Busmaster DMA performance (optimized)



Conclusions

1. Mailboxes:

- low data transfer rate
- predictable timing, small latency

⇨

- too slow for Multimedia streams
- ideal for OS communication

2. PCI Busmaster DMA:

- high data transfer rates possible, if
    - very efficient implementation
    - message length large enough

⇨

- suitable for Multimedia streams

- high overhead, limited minimum message length, unpredictable timing

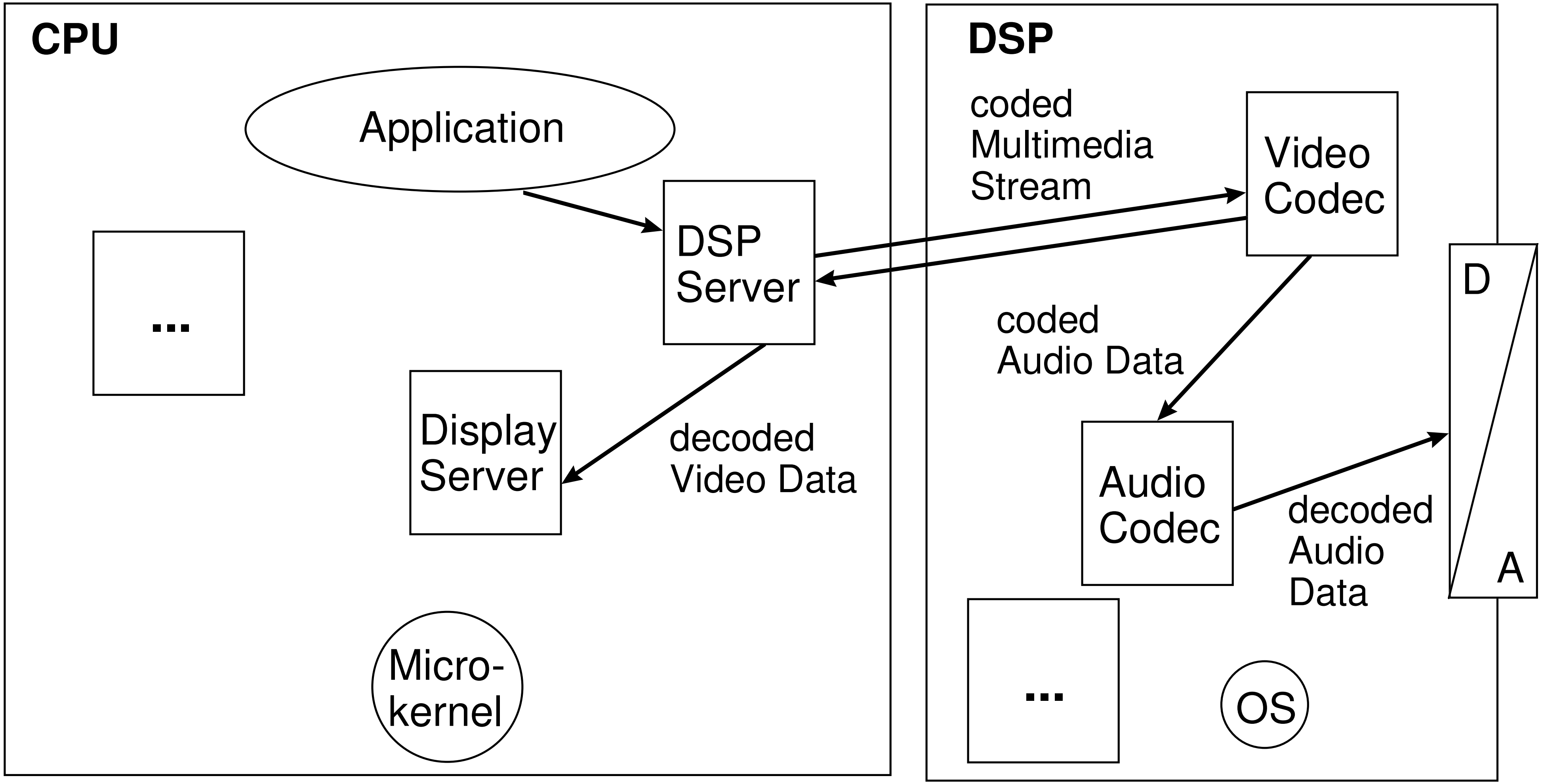
- must be complemented by another communication mechanism

The Future

- comparison of several bus systems
- statistical guarantees for DMA transfers
- Host OS: Linux ➝ Microkernel
- coprocessor-optimized DSP Kernel

Universal Multimedia Accelerator for PCs

Application Scenario



Comparison of Mechanisms