Hardware/Software Interactions on the Mpact Media Processor

Paul Kalapathy

Chromatic Research Inc.
Overview

• Reason for media processors
• Mpact Media Processor implementation
  – Hardware/Software architecture
• Examples of HW/SW interaction
Media Processors

- Provide high performance and quality for multimedia
- Permit flexibility for new or better multimedia algorithms
- Use silicon efficiently
- Achieve this through a combination of hardware and software architectures
Mpact:
7 Multimedia Functions

1. Video
- MPEG-1 real-time encode
- MPEG-1 decode (full screen, 30 fps)
- MPEG-2 decode (full screen, 30 fps)

2. 2D Graphics
- Windows GUI acceleration
- 1280 x 1024 x TrueColor, 75Hz
- VGA

3. 3D Graphics
- Windows 95 Direct3D
- Texture mapping
- Perspective correction

4. Audio
- MPEG audio
- Dolby AC-3 audio
- Wavetable synthesis
- Waveguide synthesis
- 3D sound and effects
- General MIDI
- FM synthesis
- Sound card compatibility

5. FAX/Modem
- 33,600 baud (V.34 bis)
- DSVD

6. Telephony
- Speakerphone
- Caller ID
- Voicemail
- H.320 (ISDN)
- H.324 (POTS)
- H.323 (Internet/LAN)

7. Videoconferencing
Value of Programmable MeP

- Proliferation of MM functions makes dedicated HW unreasonable
  - Gate count not cost effective
  - Intractable design and verification
  - Not all MM functions used simultaneously
    - Must re-use hardware
- Support new MM standards without new Si
- Faster time to market
  - Parallelize HW and SW efforts
Media/Host Processor

- Real-time OS required
  - Most popular host OS’s not real-time
- Microprocessor cost/gate very high
- Host processor arch tuned for general purpose computing
  - MM functions frequently not seamlessly integrated
  - Caches useless for streaming media data
  - VM not required for multimedia processing
  - Floating point use very limited in multimedia
Mpact mostly programmable

- Programmable
  - Media algorithms
  - Emulation of legacy HW
    - Sound card, VGA, COM ports
  - Codec control engine

- Hardwired
  - Bus interfaces
  - Display refresh
  - RDRAM memory controller
Mpact Processor Data Path

- RDRAM Access Control
- SRAM (4KB) 4 Read Ports
- Instruction Decode
- ALU group 1 (shifts & aligns)
- ALU group 2 (adds & logic)
- ALU group 3 (arithmetic & logic)

- Stage 1 of MUL
- Motion Estimate
- ALU group 4
- ALU group 5

792 bits (11 x 72)
Processor Arch. Tradeoffs

- No data cache needed
  - Poor locality of reference for streaming data
- Large multiport register file (512 x 72)
  - Hide/amortize memory access
  - 4R/4W ports needed for VLIW ISA
- High memory bandwidth (RDRAM)
  - Good for streaming data
  - Display refresh from same memory
  - Low pin count
• Huge data crossbar (11 GB/s)
• Result bypassing & forwarding
• Clock cycle limited by SRAM & DP paths
  – Reg file (SRAM) BW in excess of 4 GB/s
  – Higher clock rate achievable in technology
  – But, performance declines with DP pipelining
• Fixed dual-issue instruction dispatch
• Fixed-length instruction pairs
  – Concurrent or sequential execution
• VLIW-style DP controls
  – Single instructions control multiple ALUs
• Mem ops are ld/st variants with masking
  – Can ld/st 1-32 DWORDs per ld/st
• Explicit forwarding
  – ALU result registers architecturally visible
Mpact ISA

- MM data types: 9 (x8), 18 (x4), 24, 36 (x2)bits
- Flow control
  - Vector instructions
    - Vector length 0 to 255
  - Zero-overhead loops
    - Hardware loop count with no branch overhead
  - Traditional branches, jumps, calls
Mpact ISA

- Operators
  - Rich set of shift/swap/mask instructions
  - Special purpose ops
    - Motion Estimation
    - IDCT (Inverse DCT - for video decompression)
    - BFY (butterfly - for FFT)
    - SHAQ (SHift & Align Quad - for GUI accel.)
    - ROP2, ROP3 (Raster-ops - for GUI accel.)
  - Variety of integer arithmetic ops
    - add, sub, cmp, mul, mac, etc.
vector1 \([\text{mac.b } \%0, \%32 ||| \text{ bfy.b } \%64, \%128]\)
– vector multiply-accumulate & sum/diff of registers
Mpact ISA example

[bsh.b @, @p0++,@p1++ ||| me.b @.1, ageF,%64]
- fragment of inner loop of video motion estimation
Software use of Hardware Resources

- Multimedia software split between x86 and Mpact-1
- Host/Mpact decision made on efficiency basis
  - API architectures force certain structures
    - E.g., GDI primarily unidirectional
  - Performance issues drive other structures
    - E.g., MPEG video/audio streams split by x86
Mediaware Architecture

- **DOS Application**
  - DirectPlay
  - TAPI.DLL
  - COMM.DRV
  - VCOMM.386
  - Port Virtualization on VxD
  - Resource Manager (RM)
    - Concurrency Mgmt
    - DSP Task Management
    - Heap & Resource Mgmt
  - MRK
    - Device Drivers
    - Task Dispatcher
    - System Monitor

- **Windows Application**
  - GDI
  - Direct 3D
  - Draw
  - Display Driver
  - MPEG MCI Driver
  - Direct Sound
  - Multimedia Drivers

- **DOS Application**
  - MMSYSTEM.DLL
  - Games & CODEC VxD

- **x86**
  - Modem Bit Pump
  - Graphics
  - MPEG
  - XAPM (Audio)

- **M1**
  - M1 Nodes
Mediaware Architecture

• RM/MRK Partitioning
  – Resource Manager (RM) - Host side - non-real-time
  – Mpact Real-time Kernel (MRK)- Mpact - real-time

• MRK Architecture
  – Real-time, nearest deadline scheduling
  – Pre-emptive scheduling multitasking
  – Interrupt driven
    • Host interrupts do not block Mpact processes, merely post event and exit
**Mpact Real Time Kernel**

- Critical requirement for quality delivery of concurrent multimedia
  - Providing immunity from system latencies and interrupt demands
    - Memory latency
    - PCI bus latency
    - Other arbitration latencies
  - Maintaining audio/video synchronization
  - No corrupted audio! (human ear too sensitive)
    - 3D audio has very tight synchronization and latency requirements
Mediaware Architecture

- Primary RM/ MRK IPC mechanisms
  - RDRAM data structures & queues
  - Hardware semaphores
  - Hardware queues for legacy emulation
Performance: GDI Acceleration

• Architecture
  – GDI command/data queue in RDRAM
    • GDI writes “undigested” DDI commands directly to queue
    • Allows immediate return from GDI calls
    • Mpact processes queue in order
  – Queued/non-queued request synchronization
    • Host memory MUTEX
    • Acquire MUTEX, write to queue, release MUTEX
GDI Acceleration cont’d

- Performance
  - RDRAM queue never fills running Winbench
  - Winbench performance limited by application/GDI production rate
Flexibility: Dolby AC-3

• Media processor programmability allows easy adoption of new algorithms
• Mpact-1 supports full DVD decode
  – MPEG2 video
  – Dolby AC-3 audio
• Algorithm specifications not complete at Mpact-1 tape-out
• Easily implemented in SW when defined
Conclusions

- Media processor advantages
  - Achieve high performance with a programmable architecture
  - Are flexible platforms for new multimedia algorithms
  - Provides real-time behavior which is inescapable for audio, modem, etc.
  - Have dramatically lower silicon area compared to equivalent hard-wired solutions