Telairity Semiconductor

TVP400 Vector DSP Microcontroller

Howard Sachs
Siamak Arya, Richard Dickson, Jane Perera
AGENDA

- DSP Processor landscape & goals
- DSP architecture
- Floor plan
- Memory centric design
- Implementation
- Conclusion
Current DSP Landscape

- **DSP Chips**…..Focus on algorithm
  - TI TMS320C54xx

- **DSP/Microcontroller Chips**…..Algorithm + System control
  - TI-OMAP , TI TMS320C6xxx

- **L licensable Hard/Soft Cores**
  - Embedded SOC applications
    - **DSP Building Blocks**…..Application Specific
      - Partial solution
      - 3DSP SP-20 (no memory)
    - **DSP Cores**…..Algorithm
      - ParthusCeva Teak
    - **DSP/Microcontroller Cores**….Algorithm + System control
      - SH-DSP
      - Tricore 2
TVP400 Goals

- **Embedded DSP/Microcontroller for SOC implementations**
- **Performance**
  - 600MHz clock rate
  - Vector & Scalar operations (single issue)
  - Operations in flight
    - 8, 16-bit vector ops + 8 vector loads + 4 vector stores, 2 scalar op + 1 scalar load/store
  - 8-16bit vector ops/clock +1 scalar op/clock
  - FFT 256 point, complex = 2.1 us
  - FIR 64 taps=29ns/result-continuous
- **Delivered as a hard core**
  - Portable to any factory or process node
    - Less than 2MM of effort
  - Initial process node 0.13um
    - Die size of 4mm x 4mm
    - ~20 Million transistors
- **Market focus**
  - Industrial, Imaging
DSP/Microcontroller Architecture

- 4-pipe vector DSP architecture
- Independent Scalar Unit
- 128 K Byte on-chip SRAM
- 16K Byte microcontroller data cache
- 8K Byte DSP data cache
- 8KB DSP/Microcontroller instruction cache
- 1M Byte on-chip instruction ROM
- ~16 mm² in 0.13um technology
Vector DSP architecture

- Each vector unit has 16 vector registers
  - Each vector pipe or unit contains
    - 16 vector registers, each vector register has 32 elements, elements are 16 bits wide
    - One instruction causes up to 32 elements to be operated on
      - One element per clock/vector operation
  - All 4 pipes can operate at the same time
    - 4 instructions
    - 1 instruction
Vector DSP architecture

- 4 vector units
  - Operates as 4 independent vector units
    - 4 instructions
      - 4 vector units * 32 elements/vector unit = 128 operations
  - Operates as a 4 wide vector
    - 1 instruction
    - 4 vector units * 32 elements/vector unit = 128 operations

- Two 16 bit vector operations/vector unit/clock
  - 8 vector data operations/clock
  - 8 vector load operations/clock
  - 4 vector store operations/clock
Programming Methodology

- Application is written in C and C++
  - Vectorizing compiler

- Critical sections are coded in extended C or Assembly
  - Extended C programming
    - Memory type extension, like XY memory, to map data to and from SRAM
    - Data type extension for Fixed Point
    - Intrinsic functions for vector operations
  - Programming in Assembly
    - Access to full instruction set
    - Access to Assembler

- Performance Simulator
  - Debugging
  - Performance analysis
Software Tools

- **RTOS**
  - ThreadX
  - Linux

- **Compiler**
  - Apogee C/C++ compiler
  - Crescent Bay Software VAST vectorizer

- **Assembler**

- **Debugger(s)**
  - Instruction Set Simulator
  - Analyzer for investigating vector performance
  - Macro assembler
Scalar Unit

- 2-Register Files, each with
  - 32, 32-bit registers
  - 2 read, 1 write

- Scalar issue & execution units
  - 32-bit Single issue
  - 3 address

- DSP Data Cache
  - 8 KB, 4 way SA
  - 16 Byte line
  - Write through

- Microcontroller Data Cache
  - 16KB, 4 way SA
  - 16 Byte line
  - Copy back

- Instruction Cache
  - 8 KB, 4 way SA
  - Pre-fetch and lock
  - 16 Byte line
Vector Registers & Functional Units

- **4-Vector pipes**
  - 4-Loads/pipe
  - 2-Stores/pipe

- **16-Vector registers**
  per pipe

- **Each vector Register**
  has 32, 16-bit elements

- **2-Adders**
  - 24-bit accumulators
  - Round, shift

- **1-MAC**
  - 40-bit accumulator
  - Round, shift
12- Memory operations
- 8- Reads
- 4-Writes

128 memory banks

Each bank 512x16-bits

1 R/W per bank

14.4 GB/s bandwidth
Vector Bandwidth

4-Vector Functional Units

9.6 GB/S
8-16b

64 Vector Registers
32 elements/Reg

4.8 GB/S
4-16b

128 KB SRAM

4.8 GB/S
1-64b

9.6 GB/S
8-16b

19.2 GB/S
16-16b

Telairity
Floor Plan

1 MB ROM

32KB SRAM  32KB SRAM

Vector Registers
Vector Unit  8KB D CA
Scalar Unit  16KB MC D Cache

8KB I CA

32KB SRAM  32KB SRAM

Telairity
High Performance DSP’s

- Memory Centric Designs
- High performance algorithms consume BANDWITH!
- >15/16 of TVP400 core is composed of memories
  - SRAM, Register file, Cache, ROM
- Synthesis does not address memory implementation
  - Logic only
  - Memory compilers needed for customers use
- Memory compilers do not address high port counts
- Customized memories required for high performance DSP’s
  - Cost, time, risk are usually unacceptable
- Telairity’s building block approach solves this problem
  - No added cost
  - No added time
  - No risk
Building Block Implementation

- Developed by Telairity over a 3 year period
  - Large number of man years invested in technology
  - 400k gate DSP test chip completed 2002

- Pre engineered hard IP building blocks
  - IP Building blocks
  - Wiring, placement & gate sizing are optimized for
    - Speed, Portability
  - Over 200 total building blocks
    - Data path, Control
    - Multiported memories, ROM, RAM, Register files

- Core portability comes from
  - Building block portability
    - Factory to factory & process to process

- No surprises
  - Crosstalk
  - Power sag
  - Timing closure
Conclusions

- New DSP Vector architecture
  - Available as a hard core
    - Any process-any factory
      - No added cost
  - Easy to use
    - Guaranteed clock rate
    - Guaranteed area
    - No surprises
  - No additional memories to add
    - Lower cost
    - Shorter TTM

- Guaranteed Performance
  - 600 MHz clock rate
  - 256 point complex FFT in 2.1us
  - 64 tap FIR filter- 29ns/result
Technical Backup
Scalar/Vector Architecture Review

Scalar OP Scalar
A + B → C

Vector OP Vector
A_i + B_i → C_i
i = 1, n
4 Elements per Vector Register

Vector OP Scalar
A_i + B → C_i
i = 1, n
4-Pipe Vector Architecture

Vector OP Vector

Register + Register

Vector OP Vector

Register - Register

Vector OP Vector

Register \times Register

Vector OP Vector

Register \div Register
Accumulators

**Multiply Add Accumulate**
(40-bit Accumulator)

- Vector Element
- Vector Element
- Multiplier
  - 32-bit Product
  - 8-b Guard
  - 40-b
- 32-bit Accumulator
- Normalize
  - 16-bit Result

**Add Accumulate**
(24-bit Accumulator)

- Vector Element
- Vector Element
- Adder
  - 17-bit Sum
  - 8-b Guard
  - 24-b
- 16-bit Accumulator
- Normalize
  - 16-bit Result

*Telairity*
Instruction Pipeline

Scalar <- Scalar OP Scalar

(Src <- Sra OP Srb) (4-6 STAGES)

[IF][ID/RF][EX0]…[EX2][WB]

Load Scalar Register

(Rsa <- Cache)(5-6 STAGES)(cache hit)

[IF][ID/RF][AG][Ca][WB]  Load Word

[IF][ID/RF][AG][Ca][LA][WB]  Load half word or byte

Vector <- Vector OP Vector

(Vrc <- Vra op Vrb) (8-10 STAGES)

[IF][ID/RF][VD][VR][XB][EX0]…[EX2][XB][WB]

Vector <- Vector OP Scalar

(Vrc <- Vra op Sra)(8-10 STAGES)

[IF][ID/RF][VD/SR0][VR][XB][EX0]…[EX2][XB][WB]
Load Vector Register
(Vra <- SRAM) (11 STAGES)

[IF][ID/RF][AG/VD][XB][S0][S1][S2][S3][S4][XB][WB]

Store Vector Register
(SRAM <- Sra) (7 STAGES)

[IF][ID/RF][AG/VD][VR][XB][XB][S0]

Move Vector Register
(Vrc <- Vra) (7 STAGES)

[IF][ID/RF][VD][VR][XB][XB][WB]
Vector Chaining

```
LOAD    Vr1
LOAD    Vr2
MPY     Vr3<-Vr1*Vr2
ADD     Vr4<-Vr3+Vr4
```

```
IF   ID  AGXB  S0  S1  S2  S3  S4  XBWB
    ID  AGXB  S0  S1  S2  S3  S4  XBWB
    ID  VD   --  --  --  --  --  VRXB EXEX XBWB
    ID  VD   --  --  --  --  --  --  VRXB EXEX XBWB
```