Digital Signal Processors: fundamentals & system design

Lecture 1

Maria Elena Angloletta
CERN

Topical CAS/Digital Signal Processing
Sigtuna, June 1-9, 2007
Lectures plan

Lecture 1 (now!)
introduction, evolution, DSP core + peripherals

Lecture 2
DSP peripherals (cont’d), s/w dev’pment & debug.

Lecture 3
System optimisation, design & integration.

M. E. Angoletta, “DSP fundamentals & system design – LECTURE 1”, CAS 2007, Sigtuna 2/45
Chapter 1: Introduction

Chapter 2: DSP evolution & current scenery

Chapter 3: DSP core architecture

Chapter 4: DSP peripherals
Chapter 1 topics

Introduction

1.1 Overview

1.2 Use in accelerators
1.1 Overview

**Digital Signal Processor (DSP):**
μ-processor for DSPing applications.

**Characteristics**

- **Real-time** data processing
- **High throughput**
- **Deterministic** operation
- **Re-programmable** by s/w

**Key enabling technology for many electronics products:**

- **Comms:** broadband & wireless
- **Automotive:** audio, driver assistance...
- **Consumer:** security, entertainment, toys
- **Instrumentation:** medical, test/measurement ...
- **Military/aerospace:** radar, target detection ...
1.2 Use in accelerators

- Diagnostics
- Machine protection
- Control

beam [LLRF]
power supplies
motors

DSP system example: classical controls structure.
Chapter 2 topics

DSP evolution & current scenery

2.1 DSP evolution

2.2 Current mainstream DSPs

Summary
2.1 DSPs evolution: h/w features

**Development**
- Harvard architecture
- Data format:
  - early '80s: fixed point
  - late '80s: floating point (often non IEEE).
- DMA
- Fixed-width instruction sets

**Consolidation**
- Parallel architectures
- Fewer manufacturers
- Multiprocessing support
- Late '90s: improved debug capabilities (ex: RTDX)
- Trend: wider/few families for code compatibility.

---

M. E. Angoletta, “DSP fundamentals & system design – LECTURE 1”, CAS 2007, Sigtuna 8/45
2.1 DSP evolution: s/w tools

- Spectacular evolution!
- Advanced compilers
  - Deal with h/w complexity
  - Efficient high-level languages
- Graphical programming
  - MATLAB
  - NI LabVIEW DSP Module (Hyperception RIDE)
- High-performance simulators, emulators & debugging facilities
  - High-visibility into target (~no interferences)
  - Multiple DSP development & debugging in same JTAG chain.

Code Composer for TI 'C40 DSPs (CERN, 1999)
### 2.1 DSPs evolution: device integration

<table>
<thead>
<tr>
<th></th>
<th>1980</th>
<th>1990</th>
<th>2000</th>
<th>≥ 2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Die size [mm]</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>5</td>
</tr>
<tr>
<td>Technology [μm]</td>
<td>3</td>
<td>0.8</td>
<td>0.1</td>
<td>0.02</td>
</tr>
<tr>
<td>MIPS</td>
<td>5</td>
<td>40</td>
<td>5000</td>
<td>50000</td>
</tr>
<tr>
<td>MHz</td>
<td>20</td>
<td>80</td>
<td>1000</td>
<td>10000</td>
</tr>
<tr>
<td>RAM [Bytes]</td>
<td>256</td>
<td>2000</td>
<td>32000</td>
<td>1 million</td>
</tr>
<tr>
<td>Price [$]</td>
<td>150</td>
<td>15</td>
<td>5</td>
<td>0.15</td>
</tr>
<tr>
<td>Power [mW/MIPS]</td>
<td>250</td>
<td>12.5</td>
<td>0.1</td>
<td>0.001</td>
</tr>
<tr>
<td>Transistors</td>
<td>50000</td>
<td>500000</td>
<td>5 million</td>
<td>60 million</td>
</tr>
<tr>
<td>Wafer size [in.]</td>
<td>3</td>
<td>6</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>
2.1 DSPs evolution: device integration [2]

**TI 'C6713**
- Technology: 0.13 μm/6-level metal CMOS.
- Voltages: core supply = 1.26 V; I/O supply = 3.3 V.
- Core frequency: up to 225 MHz.
- Packages: 256-pin ball grid array or 208-pin plastic quad flatpack.

- Operating voltage decrease (5 V → 1.5 V):
  - Lower power consumption
  - Faster logic level transitions

**Electromigration**: higher speed increases performance but decreases chip durability
2.2 Current mainstream DSPs

3 main manufacturers: Texas Instruments (TI), Analog Devices (ADI) & Freescale semiconductor (formerly Motorola).

Mostly used for accelerators - TI & ADI

TI DSP families: TMS320Cxxxx
- 'C2x: digital signal controller.
- 'C5x: power-efficient.
- 'C6x: high-performance.

ADI DSP families:
- SHARC: first ADI family (now 3 generations).
- TigerSHARC: high-performance for multiprocessor systems.
- Blackfin: high-performance, low power.
### 2.2 Current mainstream DSPs [2]

#### Low-cost fixed point DSPs

**Applications:** Audio, controls, power supplies, consumer.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Freescale DSP563xx</td>
<td>4-47</td>
<td>24</td>
<td>275</td>
<td>24 K-648 K</td>
<td>SHARC, 'C67</td>
<td>Only mainstream DSP with 24-bit fixed point.</td>
</tr>
<tr>
<td>Freescale DSP5685x</td>
<td>3-20</td>
<td>18</td>
<td>120</td>
<td>20 K-612 K</td>
<td>'C28x</td>
<td></td>
</tr>
<tr>
<td>TI 'C24x / 'C28x</td>
<td>2-8 3-14</td>
<td>16 32</td>
<td>40 150</td>
<td>13 K-69 K 40 K-294 K</td>
<td>DSP5685x</td>
<td>Hybrid μcontroller-DSP</td>
</tr>
<tr>
<td>TI 'C55x</td>
<td>4-17</td>
<td>16</td>
<td>300</td>
<td>80 K-376 K</td>
<td>Blackfin</td>
<td></td>
</tr>
</tbody>
</table>

(*): Q3 2006 price for 10K units.
## 2.2 Current mainstream DSPs [3]

### High-performance fixed point DSPs

**Applications:** Telecom infrastructure, automotive, video.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ADI Blackfin</td>
<td>5-60</td>
<td>16</td>
<td>750</td>
<td>84 K-328 K</td>
<td>'C55x, 'C64x</td>
<td></td>
</tr>
<tr>
<td>Freescale MSC71xx/MSC81xx</td>
<td>13-184</td>
<td>16</td>
<td>300 1000</td>
<td>88 K-472 K 10.7 M</td>
<td>'C64x, Blackfin</td>
<td>Most chips are quad-core.</td>
</tr>
<tr>
<td>TI 'C64x / 'C64x+</td>
<td>15-208 180-260</td>
<td>16</td>
<td>1000</td>
<td>160 K-2112 K</td>
<td>MSC81xx/ MSC71xx</td>
<td>Adds 4- or 8-MAC features to 'C62x</td>
</tr>
</tbody>
</table>

(*): Q3 2006 price for 10K units.
### 2.2 Current mainstream DSPs [4]

**Floating point DSPs**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ADI TigerSHARC</td>
<td>130-205</td>
<td>32 float. &amp; 16 fixed point</td>
<td>600</td>
<td>512 K-3 M</td>
<td>‘C67x, SHARC</td>
<td>VLIW + SIMD. On-chip DRAM.</td>
</tr>
<tr>
<td>ADI SHARC</td>
<td>5-15</td>
<td>32</td>
<td>200</td>
<td>512 K-768 K</td>
<td>‘C67x</td>
<td>SIMD + multiprocessing</td>
</tr>
<tr>
<td>TI 'C67x/'C67x+</td>
<td>12-30</td>
<td>32</td>
<td>300</td>
<td>16 K-288 K</td>
<td>TigerSHARC, SHARC</td>
<td>Floating point version of 'C62x</td>
</tr>
</tbody>
</table>

(*): Q3 2006 price for 10K units.

Applications: Military, imaging, audio.
Chapter 2 summary

- DSPs born early ’80s: fast, real-time, deterministic processing.

- **Spectacular** evolution in software tools.
  - Compilers allow efficient high-level.
  - Graphical DSP programming (rapid prototyping).

- Great evolution in device integration.
  - Operating voltage decrease: lower power consumption & faster logic.
  - Core frequency increase.
  - Beware: *electromigration*.

- Main families for accelerator applications: ADI & TI.
Chapter 3 topics

DSP core architecture

3.1 Introduction
3.2 Fast data access
3.3 Fast computation
3.4 Numerical fidelity
3.5 Fast-execution control

Summary
3.1 DSP core architecture: intro

Shaped by predictable real-time DSPing!

\[ \sum_{k=0}^{M} a_k \cdot x(n-k) \]

- **Requirements**
  - **3.2 Fast data access**
    - High-BW memory architectures.
    - Specialised addressing modes.
    - Direct Memory Access (DMA).
  
  - **3.3 Fast computation**
    - MAC-centred.
    - Pipelining.
    - Parallel architectures (VLIW, SIMD).
  
  - **3.4 Numerical fidelity**
    - Wide accumulator regs, guard bits ..
  
  - **3.5 Fast-execution control**
    - H/w assisted zero-overhead loops, shadow registers ...

M. E. Angoletta, “DSP fundamentals & system design – LECTURE 1”, CAS 2007, Sigtuna 18/45
3.2 Fast data access

a) High-BW memory architectures

Harvard + cache = **Super Harvard Architecture** → SHARC

- Builds upon Harvard architecture & improves throughput.
- More buses than for Von Neumann: efficient *but* expensive.
- Instruction cache: loops instructions pre-fetched & buffered.
  → memory BW used for data fetch.
- Data cache on newer DSPs.
3.2 Fast data access

a) High-BW memory architectures - cont'd.

<table>
<thead>
<tr>
<th>Access time [ns]</th>
<th>h/w</th>
<th>Size [Byte]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>~5 transistor/cell</td>
<td>16K-32K</td>
</tr>
<tr>
<td>5-10</td>
<td>~2 transistor/cell</td>
<td>512K-2M</td>
</tr>
<tr>
<td>10-50</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Hierarchical memory architecture

**Cache limitations:** unpredictability of cache hits → difficult worst case scenario prediction.

**NB:** user may lock cache for deterministic execution of critical sections.
3.2 Fast data access example: 'C6713

'C6713 cache

- 2-level cache architecture, Level1 (L1) & Level2 (L2).

- L1 cache: 8 KByte total
  - 4 KByte program cache (L1P)
  - 4 KByte data cache (L1D)

- L2 cache: 256 Kbyte total
  - 64 KByte dual cache: mapped memory, cache or combination of the two.
  - 192 KByte mapped SRAM.

TI 'C6713 cache architecture
3.2 Fast data access example: ‘C6713 [2]

‘C6713 memory mapping

TMS320C6713

0000_0000
256 kB Internal Program / Data

0180_0000
Peripheral Regs

8000_0000
128 MB External

9000_0000
128 MB External

A000_0000
128 MB External

B000_0000
128 MB External

FFFF_FFFF

‘C6713 DSK (*):

8 MB SDRAM

256 kB FLASH

CPLD

Available via Daughter Card Connector

CPLD:
- LED's
- DIP Switches
- DSK status
- DSK rev#
- Daughter Card

(*): development board for DSP lab.
3.2 Fast data access

b) Specialised addressing modes

Data Address Generator (DAG) manages address update in DSP-common patterns.

- Addressing examples:
  - Circular: read & write pointers managed by h/w.
  - Bit-reversed: FFT-computation.
3.2 Fast data access

b) Specialised addressing modes - cont'd

Circular addressing example: buffer wrap-around

Bit-reversing addressing: data flow
3.2 Fast data access

c) Direct Memory Access (DMA)

DMA coprocessor: memory transfers without DSP core intervention

- DMA transfers:
  - data
  - program (for code overlay)

- Multiple channels (different priority).
- Arbitration DSP core–DMA for colliding memory access.
### 3.2 Fast data access

**c) Direct Memory Access (DMA) - cont’d.**

<table>
<thead>
<tr>
<th>Setup DMA</th>
<th>Do something else</th>
<th>Process data</th>
<th>Setup DMA</th>
<th>Do something else</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move external memory data</td>
<td></td>
<td></td>
<td>Move external memory data</td>
<td></td>
</tr>
</tbody>
</table>

**DMA setup**
- register- or RAM-based.
- typical info for setup

- **DMA transfer**: triggered by **DSP core** or by **event**.
- **DMA can generate interrupt** when transfer completed.
3.2 Fast data access

c) Direct Memory Access (DMA) – cont’d.

FLEXIBLE & POWERFUL CONFIGURATIONS

Examples:

**Chained DMA**

DMA transfer completion starts new transfer.

<table>
<thead>
<tr>
<th>Destination memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>End</td>
</tr>
<tr>
<td>6</td>
</tr>
</tbody>
</table>

Reset address

**Data formatting**

Multi-dimensional data transfers available

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7</td>
<td>8</td>
</tr>
<tr>
<td>8 9 10 11 12 13 14</td>
<td>9</td>
</tr>
<tr>
<td>15 16 17 18 19 20 21</td>
<td>10</td>
</tr>
<tr>
<td>22 23 24 25 26 27 28</td>
<td>11</td>
</tr>
<tr>
<td>29 30 31 32 33 34 35</td>
<td>15</td>
</tr>
<tr>
<td>36 37 38 40</td>
<td>40</td>
</tr>
<tr>
<td>40</td>
<td>...</td>
</tr>
</tbody>
</table>

2-D transfer

M. E. Angoletta, “DSP fundamentals & system design – LECTURE 1”, CAS 2007, Sigtuna 27/45
3.3 Fast computation

a) MAC-centered

3.3 Fast computation

a) MAC-centered cont'd

Registers: intermediate & final results, counters. Shadow registers in some DSPs (ex: ADI SHARC) for fast context switch.

Multiplier: one instruction cycle multiplication (& accumulation).

Arithmetic Logic Unit (ALU): one instruction cycle ops. (basic arithmetic & logical).

Shifter (simple / barrel): shifts (scaling) & rotates.
3.3 Fast computation

b) Pipelining

- Instruction execution divided into stages.
- Execution of stages for different instructions overlapped in time.

**Basic pipeline stages**

<table>
<thead>
<tr>
<th>Stage</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fetch</strong></td>
<td>Generate program fetch address</td>
</tr>
<tr>
<td></td>
<td>Read opcode</td>
</tr>
<tr>
<td><strong>Decode</strong></td>
<td>Route opcode to functional unit</td>
</tr>
<tr>
<td></td>
<td>Decode instruction</td>
</tr>
<tr>
<td><strong>Execute</strong></td>
<td>Execute instruction</td>
</tr>
</tbody>
</table>
3.3 Fast computation

b) Pipelining - cont'd

- Pipeline full → fast instruction fetch → L1 & L2 cache.

M. E. Angoletta, “DSP fundamentals & system design – LECTURE 1”, CAS 2007, Sigtuna 31/45
3.3 Fast computation

b) Pipelining – cont’d

- Processors may add more sub-stages.
  - Smaller steps → faster processor clock speed

C6713 pipelining

- 4 fetch stages. Fetch-packet = 8 instructions.
- 2 decode stages.
- Up to 10 execution stages.

Pipelining limitations:

- Efficient BUT complex to program → compiler/scheduler effort.

- **Branch effects**: flow change (branch/interrupt) → pipeline flush

- **Resource conflicts**: two or more instructions need same h/w.

- **Data hazards**: instruction needs result of previous instruction.

  → difficult worst case scenario prediction.
3.3 Fast computation

c) Parallel architectures

Increased parallelism improves performance.

**Very Long Instruction Words (VLIW):**

- **Instruction-level parallelism (ILP):** multiple execution units, each executes its own instruction.
- Innovative architecture - first used in 'C62xx (1997) VelociTI.
- “Multi issue” DSPs: many instructions issued @same time.

**Single Input Multiple Data (SIMD):**

- **Data-level parallelism (DLP):** one instruction performs same operation on multiple data sets.
- Technique used within other architectures (ex: VLIW).
- “Single-issue”: one instruction issued @same time.
3.3 Fast computation

c) Parallel architectures: VLIW

- Simple & regular instructions set.
- Wide “global” instructions.
- **Deterministic** behaviour: scheduling @compile-time (i.e. static) \_\_\_@processing-time.

TI 'C6000 VLIW architecture

**Plus**
- Increased performance for many algorithms
- Scalable.
- Good compiler target.

**Minus**
- Deep pipelines & long latencies: peak performance elusive.
- High memory use/power consumption.
- Assembly: complex to hand-optimise.
3.3 Fast computation

c) Parallel architectures: SIMD

- Each instruction performs *lotsa* work.
- May support multiple data width.
- SIMD can be ON/OFF

SIMD typical architecture: SHARC “Hammerhead”

- Effective on large data blocks.
- Applicable to other architectures, ex: TigerSHARC (VLIW + SIMD).

**Plus**

**Minus**

- Parallel algorithms only → reorganisation penalties.
- High program-memory usage.

M. E. Angoletta, “DSP fundamentals & system design – LECTURE 1”, CAS 2007, Sigtuna 35/45
3.4 Numerical fidelity

- Wide accumulators/registers for precision & overflow avoidance: guard bits.
- Overflow/underflow flags.
- Saturated arithmetic when overflowing.
- Floating point arithmetic: high dynamic range/precision.

### IEEE 754 standard
(normalised single precision)

<table>
<thead>
<tr>
<th>s</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSB</td>
<td>31 30</td>
<td>23 22</td>
</tr>
<tr>
<td>LSB</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

e = exponent, offset binary, -126 < e < 127
s = sign, 0 = pos, 1 = neg
f = fractional part, sign-magnitude + hidden bit

Coded number \( x = ( -1)^s \cdot 2^e \cdot 1.f \)

Dynamic range \( \text{dB} \) = \( 20 \log_{10} \left( \frac{\text{largest value}}{\text{smallest value}} \right) \)

- Fixed point \( \sim 180 \, \text{dB} \)
- Floating point \( \sim 1500 \, \text{dB} \)
3.5 Fast-execution control

- Zero-overhead looping via specialised instructions (ex: RPTB).
- Fast/deterministic interrupt servicing. Important as DSP systems often interrupt-driven.
  - **Interrupts**: internal/external (DSP pins).
  - **Latency** = delay [interrupt pin active → first ISR instruction executed].
  - DSP stops current activity (if higher priority interrupt) & starts ISR.
  - DSP must save info related to previous activity (**context**).
  - Often many user-selectable interrupt dispatchers (→ saved context).

**Example: SHARC**
(ADSP21160M, 80 MHz, 12.5 ns cycle)

<table>
<thead>
<tr>
<th>Interrupt dispatcher</th>
<th>Cycles before ISR</th>
<th>Cycles after ISR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>183</td>
<td>109</td>
</tr>
<tr>
<td>Fast</td>
<td>40</td>
<td>26</td>
</tr>
<tr>
<td>Super-fast</td>
<td>34</td>
<td>10</td>
</tr>
<tr>
<td>Final</td>
<td>24</td>
<td>15</td>
</tr>
</tbody>
</table>

ISR uses secondary register set
3.6 DSP core example: TI 'C6713

Yellow box: C6713 DSP core. White boxes: parts common to all C6000 devices. Grey boxes: additional features on the C6713 DSP.

M. E. Angoletta, “DSP fundamentals & system design – LECTURE 1”, CAS 2007, Sigtuna  38/45
Chapter 3 summary

- DSP core characteristics shaped by predictable real-time DSPing.
- Fast data access
  - High-BW memory architecture
  - Specialised addressing modes
  - Direct Memory Access (DMA)
- Fast computation
  - MAC-centred
  - Pipelining
  - Parallel architectures (VLIW, SIMD)
- Numerical fidelity
  - wide accumulator registers, guard bits...
- Fast-execution control
  - h/w assisted zero-overhead loops, shadow registers...
Chapter 4 topics

DSP peripherals

4.1 Introduction
4.2 Interconnect & data in/out
4.3 Services
4.4 C6713 example
4.5 Memory interfacing
4.6 Data converter interfacing
4.7 DSP Booting

Summary
4.1 Introduction

- Available peripherals: important factor for DSP choice.
  - Interconnect & data in/out. Potential bottleneck!
  - Services: timers, PLL, power management, booting logic.

- Embedded peripherals:
  - 😊 Fast performance
  - 😒 Less flexible across applications
  - 😊 Low power consumption
  - 😒 Unit cost may be higher

- Terrific evolution
  - Few parallel & serial ports initially.
  - Now support for audio/video streaming applications.

- Often not enough pins on DSP chip → multiplexed! Select desired peripherals @ DSP boot [→ C6713 example].
4.2 Interconnect & data I/O

Some connectivity:
- GPIO
- Host Processor Interface
- PCI

Some serial interfaces:
- Serial Peripheral Interface (SPI):
- Universal Asynchronous Receiver-Transmitter (UART)
- Controller Area Network (CAN)
- Multichannel Buffered Serial Ports (McBSP)
- Multichannel Audio Serial Port (McASP)

Some parallel interfaces:
Linkports [ADI]: DSP-DSP or DSP-peripheral communication
4.3 Services

- **Timers**: pulses / interrupts generation.

- **Power management**: reduces clock to reduce power consumption.

- **DSP booting mode & configuration logic**: [see next lecture]

- **PLL controller**: generates clock for DSP core & peripherals from internal/external clock.

- **Interrupt selector**: Selects interrupts to send to the CPU. Can also change their polarity.

- **JTAG**: allows emulation & debug [→ chapter 7].

![JTAG 14-pin header]

---

4.5 Example: C6713 DSP

Yellow box: C6713 DSP peripherals. White boxes: parts common to all C6000 devices. Grey boxes: additional features on the C6713 DSP.

M. E. Angoletta, “DSP fundamentals & system design – LECTURE 1”, CAS 2007, Sigtuna 44/45
Chapter 4 (partial) summary

- Peripherals: wide range & important parameters for DSP choice.
- Interconnect & data I/O: serial + parallel interfaces.
- Services: PLL, timers, JTAG, power management...

TO BE CONTINUED TOMORROW