Design and development of an Ultra-Low Power Intel Architecture MCU Class SoCs

HotChips 2016

Peter Barry, contributions from Venkat Madduri & Raj Kothandaraman.
Overview

- Back to the future…
- Microcontroller Characteristics
- Intel® Quark™ Microcontroller D2000
- X86 tradeoffs for microcontrollers
- SoC level microcontroller tradeoffs.
1991...

- Yesteryears desk top is todays microcontroller.
- 33Mhz 486/Pentium Era – early ’90

Source: InfoWorld - 1991

https://books.google.com/books?id=0FAEAAAAMBAJ&printsec=frontcover#v=onepage&q&f=false
http://www.zdnet.com/article/1991s-pc-technology-was-unbelievable/
Overview

- Back to the future…
- Microcontroller Characteristics
- Intel® Quark™ Microcontroller D2000
- x86 ISA tradeoffs for microcontrollers
- SoC level microcontroller tradeoffs.
Microcontroller Characteristics

- Design targets for long life from battery
- Typically integrated Flash and SRAM.
- Mixed signal, DACs/ADCs/Comparators/Radios
- Duty Cycled use cases
  - ratio between low power and active states are critical.
- Application managed power state transitions.
Duty Cycling – active to sleep ratios

Instantaneous Power Dissipated = Current * Vbatt.

Battery life = Power density/power dissipated

Goal to minimize area under curve.

Vbatt Current

Active

Active

Low power sleep state

Time
Expanded Power analysis

Product architecture driven by process

- Time
- Time to go to low power state
- Time Spend on processing Loop
- Interrupt Entry Times
- Oscillator Ramp Up
- Dynamic Power for Active State
- Incremental Leakage for Active SoC portions
- Wake Logic
- SRAM Retention
- RTC/OSC
- Leakage
Overview

- Back to the future…
- Microcontroller Characteristics
  - Intel® Quark™ Microcontroller D2000
  - x86 tradeoffs for microcontrollers
  - SoC level microcontroller tradeoffs.
Entry Edge Controller: D2000

**Battery-operated**
- <35mW active, <10uW idle*
- Months/years activity dependent on use case

**Intel® Architecture Microcontroller**
- Extensible End Point
- 40pin QFN

**Scalable SW applications & Tools**
Intel® System Studio for Microcontrollers
- Free development tools, (GCC* Compiler, JTAG Debugger, flashing, optimized C and DSP libraries)
- Scalable Intel® Quark™ Microcontroller Interface API

**Hardened**
- -40 to 85°C ambient, 10 year reliability
- Long life availability

*Other names and brands may be claimed as the property of others.
Overview

- Back to the future…
- Microcontroller Characteristics
- Intel® Quark™ Microcontroller D2000
- x86 tradeoffs for microcontrollers
- SoC level microcontroller tradeoffs.
Tradeoffs

- Intel x86 ISA Compatibility
  - Instruction set has evolved over 30 years, what is the correct point.

- System level capability
  - Tradeoff of existing software

- Performance, area and power
Quark is an in-order 5 stage IA32 CPU with Pentium ISA.

Configurable SoftIP targeting area, power, performance and features for different applications.
## Quark D2000 Core Configuration Options

<table>
<thead>
<tr>
<th>Domain</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arch</td>
<td>Pentium ISA baseline.</td>
</tr>
<tr>
<td></td>
<td>PAEXD feature – disabled</td>
</tr>
<tr>
<td></td>
<td>SMEP feature – disabled</td>
</tr>
<tr>
<td></td>
<td>Local APIC – nested vectored controller</td>
</tr>
<tr>
<td></td>
<td>IOAPIC – Vectored interrupt controller.</td>
</tr>
<tr>
<td></td>
<td>No FPU</td>
</tr>
<tr>
<td>Bus Interface</td>
<td>AHB-Lite Fabric Interface</td>
</tr>
<tr>
<td>Target max</td>
<td>Frequency : 32Mhz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Micro Architecture</td>
<td>Enable Tightly/Closely Coupled Memories (CCM)</td>
</tr>
<tr>
<td></td>
<td>Instruction CCM Support</td>
</tr>
<tr>
<td></td>
<td>Data CCM – 8K</td>
</tr>
<tr>
<td></td>
<td>TLB Entries – 2/2</td>
</tr>
<tr>
<td></td>
<td>Data Cache Disabled</td>
</tr>
<tr>
<td>Implementation</td>
<td>Single/Dual Ported Memories</td>
</tr>
</tbody>
</table>
Application Instruction Set

- Retain instruction set compatibility for compiler/debugger/tool chain reuse.
- Binary compatibility lesser consideration.
- For the simplest MCU’s – we chose Pentium ISA baseline.
- Drive implementation of Core to meet area/power/performance targets.
System Level ISA

- Enumerate features using CPUID.

- Paging
  - Retained in design for protection not translation
  - Set TLBs to 2Instr/2Data.

- Segmentation
  - Retained as provides memory protection.
Interrupt Routing - Latency reduction.

- Traditional – vector assignment and priority
- MCU – fixed vector assignment and priorities
- Integrated tightly into core, IDT cached
Instruction TCM Tradeoffs.

- Product Cost
- CoreMark Performance
- Core Prefetch Power performance tradeoff
- Flash Controller Sleep states
- Wait states vs core speed
- Race to halt power
## Prefetcher (PRF) Sensitivity to CoreMark

<table>
<thead>
<tr>
<th>Freq (MHz)</th>
<th>CoreMark Performance (CM/MHz)</th>
<th>Active Power (mW)</th>
<th>CM/MHz/mW</th>
<th>Perf Loss</th>
<th>Pwr Reduc</th>
<th>Energy (mJ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefetcher ON</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>1.53</td>
<td>26.4</td>
<td>0.058</td>
<td>N/A</td>
<td>N/A</td>
<td>552.7</td>
</tr>
<tr>
<td>16</td>
<td>1.77</td>
<td>16.5</td>
<td>0.107</td>
<td>N/A</td>
<td>N/A</td>
<td>597.6</td>
</tr>
<tr>
<td>8</td>
<td>1.77</td>
<td>8.3</td>
<td>0.213</td>
<td>N/A</td>
<td>N/A</td>
<td>601.2</td>
</tr>
<tr>
<td>4</td>
<td>1.81</td>
<td>4.7</td>
<td>0.385</td>
<td>N/A</td>
<td>N/A</td>
<td>666.3</td>
</tr>
<tr>
<td>Prefetcher OFF (relative to Prefetch on)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>1.14</td>
<td>23.4</td>
<td>0.049</td>
<td>25%</td>
<td>11%</td>
<td>658.4</td>
</tr>
<tr>
<td>16</td>
<td>1.37</td>
<td>13.2</td>
<td>0.104</td>
<td>23%</td>
<td>20%</td>
<td>614.8</td>
</tr>
<tr>
<td>8</td>
<td>1.37</td>
<td>6.9</td>
<td>0.199</td>
<td>23%</td>
<td>17%</td>
<td>642.7</td>
</tr>
<tr>
<td>4</td>
<td>1.60</td>
<td>4.1</td>
<td>0.390</td>
<td>12%</td>
<td>13%</td>
<td>657.5</td>
</tr>
</tbody>
</table>

**Recommendation:** Prefetcher ON Always – more energy efficient
Von Neumann/Harvard – Data Fetches

- Lower latency direct paths to memory
- Self modifying code
  - ITCM Flash only, no cache.
- Literal Pools (immediate data)
- X86 Variable length instructions
- Data fetch to ITCM routed via AHB
- C env startup relocate data to SRAM
Core Sleep States

- Tradeoff between sleep power and exit latency.

- Clock Stopping
  - X86 Core/memory clock stop via Halt intr.
  - Wake on interrupt

- Power state – core – always on (no C6 supported)
Overview

- Back to the future…
- Microcontroller Characteristics
- Intel® Quark™ Microcontroller D2000
- x86 tradeoffs for microcontrollers
- SoC level microcontroller tradeoffs.
Power Management Efficiency.

- Power deliver efficient/mode depends on supply current.

<table>
<thead>
<tr>
<th>Config</th>
<th>Mode</th>
<th>Max Current</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.8v Normal</td>
<td>Switching</td>
<td>50mA</td>
<td>90% 50mA-10mA : 70% 10mA</td>
</tr>
<tr>
<td>1.8v Low Current</td>
<td>Linear</td>
<td>300μA</td>
<td>99%</td>
</tr>
<tr>
<td>1.35v Retention</td>
<td>Linear</td>
<td>300µA</td>
<td>99%</td>
</tr>
</tbody>
</table>

- 30% Efficiency delta – equates to 30% battery life, when dominated by idle power.
# D2000 Deep Sleep Power

<table>
<thead>
<tr>
<th>SoC Power State</th>
<th>VR</th>
<th>HYB OSC</th>
<th>RTC OSC</th>
<th>CPU</th>
<th>CMP</th>
<th>ADC</th>
<th>Deep Sleep Current Actual (µA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deep Sleep RTC AONPT Wake</td>
<td>iLR-1.8V</td>
<td>OFF</td>
<td>ON</td>
<td>HALT</td>
<td>OFF</td>
<td>OFF</td>
<td>3.4</td>
</tr>
<tr>
<td>Deep Sleep NoRTC Comparator Wake</td>
<td>iLR-1.35V</td>
<td>OFF</td>
<td>ON</td>
<td>HALT</td>
<td>1 ON</td>
<td>OFF</td>
<td>2.2</td>
</tr>
<tr>
<td>Deep Sleep NoRTC GPIO Wake</td>
<td>iLR-1.8V</td>
<td>OFF</td>
<td>OFF</td>
<td>HALT</td>
<td>OFF</td>
<td>OFF</td>
<td>1.9</td>
</tr>
</tbody>
</table>

Notes:
- Deep Sleep Current = PVDD Current + AVDD Current + IOVDD Current. Sum of all 3 inputs rails.
SoC Clocking

- Core Frequency Scaling (32/16/8/4Mhz)
  - Race to halt usually best, but running slower for longer better in some use cases.
  - 4Mhz Operation could fit in Low Power regulator modes max current.
  - Tradeoff of Power Islanding & Leakage.
Summary

- First iteration right ballpark in terms of performance/area/power/cost

- We Continue to
  - Continue to iterate on the micro architecture
  - Process related micro-architecture evolution
  - Analog IP evolution
Q&A
Disclaimer

Intel, the Intel logo, Quark are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

'Mileage May Vary' Disclaimer: Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Estimated Results Disclosure: Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.
<table>
<thead>
<tr>
<th>Micro-Operation</th>
<th>Prefetch</th>
<th>D1</th>
<th>D2</th>
<th>Execute</th>
<th>Write Back</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefetch from ITCM</td>
<td>Prefetch Code from ITCM</td>
<td>Opcode Decode</td>
<td>X</td>
<td>RF Read ALU Operation</td>
<td>Register Update</td>
</tr>
<tr>
<td>Load</td>
<td>Prefetch Code from ITCM</td>
<td>Opcode Decode</td>
<td>Linear Address Generation</td>
<td>TLB Lookup DTCM Read</td>
<td>Register Update</td>
</tr>
<tr>
<td>Store</td>
<td>Prefetch Code from ITCM</td>
<td>Opcode Decode</td>
<td>Linear Address Generation</td>
<td>TLB Lookup DTCM write</td>
<td>DTCM Write</td>
</tr>
<tr>
<td>Jump</td>
<td>Prefetch Code from ITCM</td>
<td>Opcode Decode</td>
<td>X</td>
<td>Taken/Not Taken</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Prefetch (PF) from Target Address for Tkn</td>
<td>D1</td>
</tr>
</tbody>
</table>