The new Intel® Xscale™ Microarchitecture

Nuno Ricardo Carvalho de Sousa

Departamento de Informática, Universidade do Minho
4710 - 057 Braga, Portugal
nuno.r.sousa@iol.pt

Abstract. In embedded systems, performance and power consumption are the most important criteria to define a good processor chip. The new Intel® Xscale™ microarchitecture, an evolution from StrongARM™ microarchitecture, combines these two features, as will be detailed in this communication. We will also see the advanced techniques used by this microarchitecture core to achieve a high level of efficiency.

1 Introduction

Nowadays, most microprocessors are in embedded systems, not in PC's. Embedded products become part of our everyday items: cellular phones, video games, Personal Digital Assistants (PDA) and much more. Although PC processors seem to generate much of all the excitement in the press, it is the other 98 percent – the embedded processors – that are technologically leading the way. This required a new design of microprocessors.

The performance of these embedded microprocessors rivals that of PC's of just few years ago. With clock frequencies up to 400 MHz, these chips offer performance, with a very economical electrical consumption [1].

A microprocessor's architecture defines the instruction set and programmer's model for any processor that will be based on that architecture. Different processor implementations may be built to comply with the architecture. Each processor may vary in performance and features, and be optimized to target different applications. In this document we will see with more detail Intel's 80200, the first microprocessor that use Xscale, and the new Intel PXA250 application processor.

At the core of the PXA250 is an Intel Xscale core-based microprocessor. Intel Xscale is a 32-bit RISC microarchitecture based on Advanced RISC Machines (ARM). ARM-based and Intel Xscale technology are completely binary compatible. So software and software-developed tools designed for older ARM processors also work on newer Intel Xscale core-based processors. The Intel Xscale RISC microarchitecture is noted for its efficiency. It obtains high performance, minimal number of silicon transistors, which requires less power to operate the chip. It also means the chip processor itself will take up less silicon, making it smaller and less expensive to manufacture in volume.

2 Evolution of the ARM™ Architecture

The ARM processor is a great enhancement to processor designing in the embedded system market. This product not only offers high performance, but also it achieves small die size and low power.

Some basic hardware features of the ARM processor are [2]:

- It is a Reduced Instruction Set Computer i.e. RISC.
- It has a large uniform register file.
- It contains a load/store architecture where data-processing operations only operate on register contents and not directly on memory contents.
- Simple addressing modes.
- Uniform and fixed-length instruction fields, to simplify instruction decode.

The special feature of ARM processors is that they implement two instruction sets:

1. 32-bit ARM Instruction Set
2. 16-bit Thumb Instruction Set

The advantages of having two instruction sets are manifold. The Thumb instruction set is a re-encoded subset of the ARM instruction set. It is designed to increase the performance of ARM implementations that use a 16-bit or narrowed memory data bus and to allow better code density than ARM. This can be related to traditional RISC processors, which have a fixed 32-bit instruction set and hence each instruction occupies 4 bytes in memory [3]. This result in larger amount of ROM required in the system which means poor code density and hence more power consumed.

Another feature in the ARM processor is the inclusion of I/O controller units on the chip. A lot of power is consumed in transferring of data such as that related to a video output to the off chip controller unit. ARM, which has an extremely small die size, incorporated the controller unit onto the same die. This results in lesser switching costs and faster execution.

The introduction of a new architecture does not replace existing architectures, or make them redundant.

At each major revision of the ARM architecture, significant features have been added. Between major architecture revisions, new features have been included as variants on the architectures. The key letters appended to the core names indicate specific architecture enhancements within each implementation [4].

- V3 introduced 32-bit addressing, and architectures variants:
  - T – Thumb state: 16-bit instruction execution.
  - M – long multiply support (32 x 32 => 64 or 32 x 32 + 64 => 64). This feature became standard in architecture V4 onwards.
- V4 added halfword load and store
- V5 improved ARM and Thumb interworking, count leading-zeroes (CLZ) instruction, and architecture variants:
  - E – enhanced DSP instructions including saturated arithmetic operations and 16-bit multiply operations.
  - J – support for new Java state, offering hardware and optimized software acceleration of bytecode execution.
  - Floating-point instructions.

The Xscale architecture is compliant with the ARM Version 5TE ISA instruction set. The Xscale core implements the integer instruction set architecture of ARM V5, but does not provide hardware support to floating point instructions.

3 Architecture Overview

As the successor to Intel's StrongARM microarchitecture, a general-purpose embedded RISC microarchitecture, the new microarchitecture is an Internet focused RISC solution compliant with the ARM instruction set.

The core has been designed with power saving techniques that power-up a functional block only when it is needed. Low power modes are selectable by programming and the core has been designed to enable dynamic clocking. The core's clock frequency can also be adjusted by programming. This enables software to conserve power by matching the core clock frequency to the current workload [5].

Together with design enhancements added to the core, this technology allows the Intel Xscale microarchitecture to operate over a wide range of speed and power, producing performance that varies from 40-mW/185-MIPS at 150 MHz to 900-mW/1000-MIPS at 800 MHz.

The Xscale processor, that has been developed based on the ARM technology, also supports ARM and Thumb Instruction Set.

The core features of the Xscale processor are as follows:

- 7-8 stage Intel Superpipelined RISC technology to achieve high speed and low power.
- Intel Dynamic Voltage Management. Dynamic voltage and frequency on-the-fly scaling allows applications to utilize the right blend of performance and power.
- Intel Media Processing Technology. Multiply-accumulate coprocessor performs two simultaneous 16-bit SIMD multiplies with 40-bit accumulation for efficient media processing.
- Power management unit gives power savings via idle, sleep and quick wake-up modes.
- 128-entry branch target buffer. Keeps pipeline filled with statistically correct branch choices.
- 32 Kb instruction cache and 32 Kb data cache. Keeps local copy of important data to enable high performance and low power.
- 32-entry Instruction Memory Management Unit. Enables logical-to-physical address translation, access permissions, I-Cache attributes
- 32-entry Data Memory Management Unit. Enables logical-to-physical address translation, access permissions, D-Cache attributes.
- 4-entry Fill and Pend Buffers. Promotes Core efficiency by allowing non-blocking and "hit-under-miss" operation with D-Caches.
- Performance Monitoring Unit. Furnishes two 32-bit event counters and one 32-bit cycle counter for analysis of hit rates, etc.
- Debug Unit. Uses Hardware Breakpoints and 256-entry Trace History Buffer (for flow change messages) to debug programs.
- 8-entry Write Buffer. Allows the Core to continue execution while data is written to memory.
- 2 Kb mini-data cache to avoid thrashing of the d-cache.

We can see in figure 1 the major functional blocks of the Intel Xscale Microarchitecture core. It gives a brief high-level overview of these blocks [6].

In this figure we may see for example the MAC (Multiply Accumulator) that supports single cycle throughput for 16-bX32-b operations and 16-b SIMD operations for audio processing. The MAC encodes 16-b of the multiplier in the first cycle and encodes 12-b of the multiplier for the rest of the cycles with four stages. Two 40-b accumulators increase the multiply instruction throughput by avoiding data dependencies without requiring high circuit speed, thus limiting power. Forty-bit accumulators increase performance and precision of audio coding algorithms by allowing infrequent overflow checking.

To obtain high performance, the microprocessor core utilizes a simple scalar pipeline and a high-frequency clock. To avoid the potential power waste of a superscalar approach, functional design and validation complexity is decreased at the expense of circuit design effort. To avoid circuit design issues, the pipeline partitioning balances the workload and ensures that no one pipeline stage is tight. The main integer pipeline is seven stages, memory operations follow an eight-stage pipeline, and when operating in thumb mode an extra pipe stage is inserted after the last fetch stage to convert thumb instructions into ARM instructions. Since thumb mode instructions are 16-b, two instructions are fetched in parallel while executing thumb instructions [7].

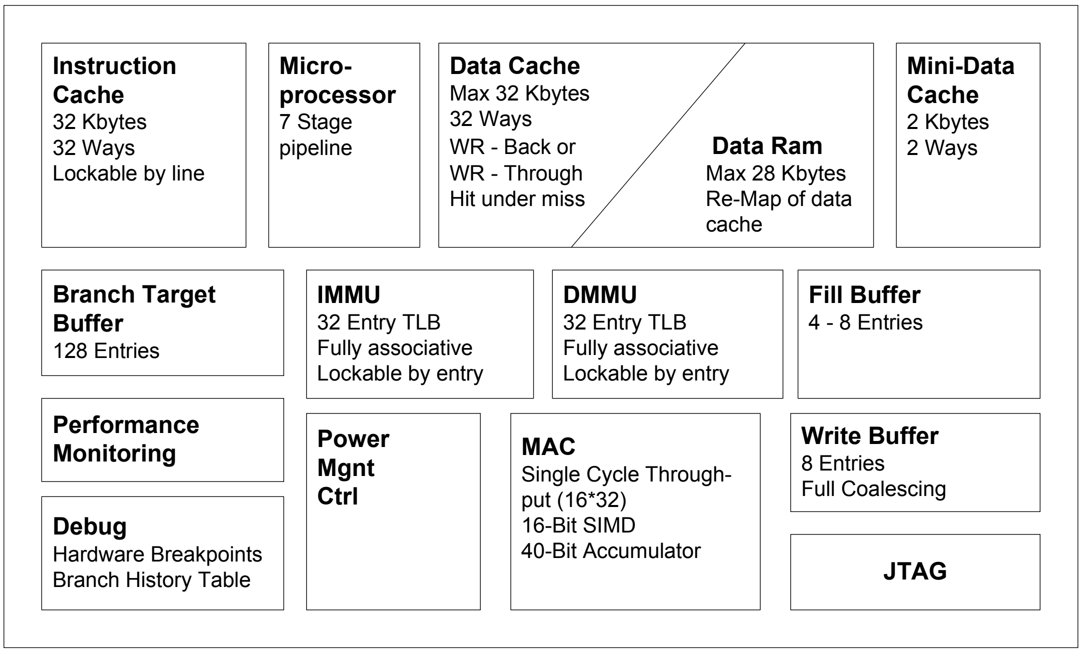

Fig. 1 Intel Xscale Microarchitecture Features

4 Intel Xscale Enhancements

The core of Xscale microarchitecture enhancements was the Intel dynamic voltage management (low power), the Intel media processing technology (multimedia) and the Intel superpipelined microarchitecture (performance).

Intel dynamic voltage management provides system-level power management through dynamic voltage and frequency scaling and new, low power modes. The result being that applications can on the fly use the performance they need when they need it, then quickly reduce the power and frequency to save battery life. Additionally, Intel's designers achieved low power through customized circuits, making extensive use of clock gating, leakage suppression circuitry, and a variety of special circuit techniques support low voltage operation, and balanced pipeline partitioning adds greater efficiency [8].

Intel media processing technology is the first in a series of planned media enhancements. Utilizing ARM coprocessor space, the new enhancements include 16-bit SIMD (Single Instruction-Stream Multiple Data-Stream) multiplies with 40-bit accumulation for efficient media processing. This technology was designed to accelerate multimedia and communications software. Basically, it embedded basic DSP (Digital Signal Processing) capabilities in the processor and included new data types to accelerate calculations common in audio, 2D and 3D graphics, video and data communications algorithms. The SIMD extensions are instructions that reduce the overall number of instructions required to execute a particular program task. As a result, they can perform good performance by accelerating a large range of applications, including video, speech, and image processing. The Xscale microprocessor is a general-purpose processor enhanced with digital signal processing capabilities. With its low power needs and good heap dissipation, is well suited for use in low-end solutions, such as access gateways for the small office market segments delivered by service providers through core networks applications [9].

Intel superpipelined technology enables performance approaching 1 GHz and better throughput and interrupt handling capabilities. This improved pipeline technology includes 7-stage integer/8-stage memory superpipelined for fast clock rates, dynamic branch prediction, extensive data bypassing, and high-speed custom circuits. This chip feature allows the processor to scale to the higher clock speeds. It is an important issue because lines up instructions to be processed.

These solutions for the wireless world are good options of OS support for the convergence of voice, video, data, and Internet connectivity in handheld devices [8].

The Intel Xscale core in the PXA250 applications processor is an improvement over the previous Intel StrongARM processors, the Intel SA-1100 and the SA1110. Unlike the SA chips, the Intel PXA250 applications processor includes the ARM Thumb code-compression technology. For all its advantages, RISC architecture has one weakness: code density. Code density describes the space needed to store code (software) in memory. RISC chips generally have significantly larger programs than Complex Instruction Set Computing (CISC) processors requiring more memory to store the same amount of code. The Thumb technology improves this situation dramatically, compressing software density by about 30 percent over normal RISC code. The Intel PXA250 applications processor handles this compression automatically.

Other improvement in the Intel Xscale core is the dual-multiply/accumulate (dual-MAC) instruction. A MAC operation is a relatively new addition to most computers and microprocessors, borrowed from the Digital Signal Processor (DSP) world. MAC operations are so vital to many audio, video, and wireless applications, that one or two MAC instructions can dramatically benefit a chip's ability to run these multimedia applications [1].

Another enhancement was the use of the Memory Management Unit (MMU) to prevent programs from accessing any memory outside of the specific region (Address Space) to which they are assigned [10]. The MMU associates certain pages of physical memory with a virtual 'address space' that a program, or task, is able to access. A task can only access physical memory that has been 'mapped' to its virtual addresses: it is not possible for it to see any other memory. This not only prevents one program from causing another to malfunction, but also automatically identifies attempts to do so, making correction much easier.

5 Low Power Techniques

The first microarchitecture designs have emphasize performance and speed without significant concern about power dissipation. The main goal of Intel Xscale was the use of some techniques to reduce power consumption without lost of performance. These techniques may be implemented either by hardware or by software.

One of the hardware techniques to obtain low power consumption is clock gating. It is commonly implemented in commercial processors to reduce the switched capacitance associated with the clocks. The clock is the main consumer of power and becomes necessary to implement a technique that lowers power consumption by clocks [11]. With clock gating, a chip can execute an instruction without switching transistors or involving gates that aren't needed for the operation.

There are, however some problems associated with clock gating:

- Possibility that a disabled block may not power up on time.
- The problem of clock skew (different clock delays for different functional units) may worsen
- Great variation in current on successive clocks will lead to more noise problems.

The Intel Xscale core use other advanced circuit, process, and microarchitectural techniques to facilitate low power operation.

The CMOS power consumption equation is given by:

$$P = C * f * V^2$$

where, P = power consumed
C = switched capacitance
f = operating frequency
V = operating voltage

A processor's active power dissipation is proportional to the frequency and square of the voltage. In the V^2 term of the active power equation, voltage to the core is scalable from 0.75 V to 1.65 V. The core also incorporates back-bias circuitry to minimize leakage current during inactivity, and the core's static design – the clock can be reduced to dc – contributes linearly to active power savings.

Intel's 80200, offers additional power-saving opportunities by enabling rapid frequency and no-stall voltage changes and by providing a memory bus that runs asynchronously. Because the 80200 uses a PLL with a fast resynch time, it can switch frequencies or power modes in less than 20 μs. This enables system techniques in which the processor, while providing good interactive response, is usually asleep or operating at low frequency. For example, the processor can be in 'drowsy' mode (core dissipation less than 1 mW) until an event awakens it; then it processes the event at several hundred MHz and drops back in drowsy mode.

For voltage scaling to be effective, the processor must not spend excessive time waiting for the voltage to change. A shorter processor idle period translates into accomplishing more work. The 80200's core can run through a voltage change without the processor idling during the transition. Also, the voltage can follow a natural discharge curve rather than being stepped or precipitously dropped for the core, witch avoids wastefully forcing the voltage to a lower level and perhaps enables the use of a simpler power supply.

The 80200's bus interface runs asynchronously – the core can run at a frequency unrelated to the bus clock. In fact, two clock signals are input to the device, allowing system software to change core frequency as needed without concern for bus operation. If the bus frequency is a fixed fraction of the core frequency, as is often the case, a change in core frequency might necessitate waiting for the bus to clear.

The asynchronous relationship of the bus and core clocks also enables static or dynamic optimization of the bus frequency. A RAM region with a 25-ns access time requires three 100-MHz clock cycles per access, or one 33 MHz clock cycle. A system with predictable accesses to this memory might choose to drop the bus frequency and realize a linear improvement in power without incurring any loss in throughput [12].

In conclusion, this is a method, which allows the processor to dynamically alter its operating voltage at run-time depending on the present load of work on the processor. Just lowering the frequency for low power is not effective. Because the power consumed will be the same as the execution time of the process will increase. So to effectively reduce power, voltage should also be reduced.

Other technique to reduce power consumption may be performed by programmers through software. If a program can detect when the processor will not be used for a short time, they can put part of, or the full processor to sleep. Even short naps of few microseconds are worthwhile for maximizing energy saving. For example, a handheld computer can take naps in between each tap on the keyboard; even the fastest typists are slow compared to a microprocessor. The processor might be asleep for as much as 95 percent of the time, yet the user would notice no difference in the device's operation. Lighter naps (called 'idle mode') drop the processors power consumption by a factor of two or three, to about 100 mW. Longer naps ('sleep mode') can slash power to just 50 µs [1].

6 Conclusions

Embedded processors have certainly made dramatic advances in the past few years. Developing a processor that is fast, low cost and miserly with power is a challenge few companies can undertake.

Recently, processors have begun to trade off performance for power, while using operating system, microarchitecture, and circuit techniques to minimize power consumption. In my opinion, I think Intel Xscale chips are a good market option because it combines perfectly performance with power control. A pair of AA batteries can now power a 400 MHz microprocessor and it's peripherals for hours or even days. This level of efficiency it is possible using advanced techniques to archive it.

Today, the average consumer can carry around more processing power in there pocket than NASA put on the moon in the 1970's. Power for communications, power for compressing and processing video, power for music and games it is all there in the palm of your hands [1].

References

[1] Intel: The Intel PXA250 Applications Processor,

http://www.intel.com/design/pca/applicationsprocessors/whitepapers/25031201.pdf (2002)

[2] David, Seal: ARM Architecture Reference Manual, Second Edition, Addison-Wesley (2001)

[3] Oliver Gunasekara: Developing a Digital Cellular Phone using a 32-bit microcontroller, http://www.arm.com/support.nsf/htmlall/80256BC1005E148480256A8C004AB6A3/$File/digitalc ellularphone.pdf?OpenElement (2002)

[4] Brash, David: The ARM Architecture Version 6 (ARMv6), http://www.arm.com/support/56vf7h/$file/ARMv6_Architecture.pdf (2002)

[5] Intel: Intel Xscale Microarchitecture – Technical Summary, ftp://download.intel.com/design/intelxscale/XScaleDatasheet4.pdf (2000)

[6] Intel: Intel Xscale Core – Developer's Manual, ftp://download.intel.com/design/intelxscale/27347301.pdf (2000)

[7] Lawrence T. Clark: An embedded 32-b Microprocessor Core for Low-Power and High-Performance Applications, IEEE Journal of solid-state circuits, vol. 36 No 11 November 2001

[8] Rogers, David, English, Joe: Intel, Introducing Intel Xscale Microarchitecture (2000)

[9] Intel: Next-Generation Media Processing for the Modular Network, White Paper, http://www.intel.com/network/csp/pdf/7786.pdf (2002)

[10] Hills, Green: Integrity RTOS Uses Intel Xscale Microarchitecture to Deliver High Reliability, http://www.ghs.com/download/whitepapers/integrity-rtos-xscale.pdf (2002)

[11] Mehta, Abhishek: Technics for Low Power Processing – A Survey (Spring 2001)

[12] Morrow, Mike: Microarchitecture Uses a Low Power Core, IEEE Computer, (April 2001)