Student Papers in Computer Architecture, 2004

Sandra Taha, Tommy Löfstedt and Ola Ågren (editors)

Technical report, 2005
UMINF 05.27
ISSN 0348-0542

DEPARTMENT OF COMPUTING SCIENCE
UMEÅ UNIVERSITY
SE-901 87 UMEÅ
SWEDEN

UMINF 05.27 ISSN 0348-0542

Umeå's Ninth Student Workshop in Computer Architecture

This book is the proceeding from Umeå's Ninth Student Workshop in Computer Architecture. It contains a collection of short technical articles about past and present processors, medias, busses and protocols. The articles have been written by the students of the autumn 2005 offering of the course in Computer Architecture.

Introduction

Another year has gone by since the last student workshop in Computer Architectur, and the time has come for a new one. The main topic of the course is still the same (as can be seen above) but one major and some minor changes has been done in the formatting department. This is the first year that we generate one continuous pdf file containing all the reports. This means that a slightly different style file had to be used, and we have had to enforce a stricter set of allowable things to do (no user-defined definitions outside of the bibliographies, labels must be unique, etc.)

The main purpose of this book is to give an overview of the current (and the not so current) state of Computer Architecture as well as to show the effort put into the course by the students. My wish is that the reader might find something interesting to them, something that is worth further research and therefore further enlightenment.

I hope you'll find it a pleasurable book to read.

September 2005
Program committee: Taha, Löfstedt and Ågren

A Note on Trademarks and Registered Trademarks

Contents

Part I

Processors

1. AMD Athlon 64

1.1 Introduction

The AMD Athlon 64 was released on September 23, 2003 and is the first 64-bit processor on the market fully compatible with the 32-bit legacy x86 architecture. This makes any Athlon 64 based system compatible with all existing 32-bit x86 software such as Windows and all its software. The AMD Athlon 64 is used in desktop computers.

The AMD Athlon 64 is built upon the x86-64 architecture which is a extension of the existing x86 architecture and it was developed by Advanced Micro Devices, Inc. as an attempt to create a seamless transaction between the 32- and 64-bit computing platforms. This is because today it would be almost impossible to go to a new pure 64-bit platform, which would require that all software was ported and recompiled. And that is not done overnight [2].

The first 64-bit AMD processor was the Opteron. It was released in April 2003 with clock speeds of 1.4, 1.6 and 1.8 GHz. The 2.0 GHz Athlon 64 and the 2.2 GHz Athlon 64 FX was released in September 2003. The latest addition to the AMD64 family is the dual-core Athlon X2 that was introduced in June 2005. It came in 4 different versions, 4200+, 4400+, 4600+ and 4800+ [4].

1.2 Overview

1.2.1 Registers

When the processor is run in the x86 32-bit legacy mode the processor will have eight 32-bit general purpose registers, eight 64-bit MMX and floating point registers, eight 128-bit SSE extension registers a 32-bit flag register and a 32-bit instruction pointer. This is normal for any modern x86 processor but when the processor is run in 64-bit mode there will be eight more general purpose registers totaling 16 and they will all be 64 bits, there will also be 16 SSE extension registers available and the instruction pointer will be extended to 64 bits [2].

1.2.2 Instruction Formats

The processor can be run in two major modes. The first is 32-bit legacy mode which makes the processor able to run an unmodified 32-bit OS natively. The other mode is long mode where all the x86-64 extensions is enabled. This mode is divided into two submodes, compatibility mode and 64-bit mode. The compatibility mode can be run by a 64-bit OS with 64-bit programs and at the same time run all old 32-bit x86 software which can be run in x86 protected mode. When run in compatibility mode the programs will see it as a x86 protected mode while the OS can use all the 64-bit mechanisms on the processor. The second submode of long-mode is the 64-bit mode which is made for the pure 64 bits applications and in this mode the program can access all the 64-bit extensions.

The processor also fully implements SSE1/2, MMX, 3DNow! [1, 2].

Mode		Operating System Required	Application Recompile Required	Defaults[1]			
				Address Size (bits)	Operand Size (bits)	Register Extensions[2]	GPR Width (bits)
Long Mode[3]	64-Bit Mode	New 64-bit OS	yes	64	32	yes	64
	Compatibility Mode		no	32		no	32
				16			
Legacy Mode[4]		Legacy 32-bit or 16-bit OS	no	32	32	no	32
				16	16		

1. Defaults can be overridden in most modes using an instruction prefix or system control bit.
2. Register extensions includes eight new GPRs and eight new XMM registers (also called SSE registers).
3. Long mode supports only x86 protected mode. It does not support x86 real mode or virtual-8086 mode. Also, it does not support task switching.
4. Legacy mode supports x86 real mode, virtual-8086 mode, and protected mode.

Figure 1.1: Processor operating modes (from [2])

1.2.3 Instruction Set Extensions

3DNow!

3DNow! is a multimedia extension created by AMD in 1998 for their K6 processors. It's an addition of SIMD instructions to the x86 instruction set, to better cope with the vector-processing requirements of many graphic-intensive applications. It was orginally developed to extend the integer only MMX instruction set to allow floating-point calculations.

3DNow! have been updated two times. The first "Enhanced 3DNow!" was introduced with the first generation of Athlons and only minor additions were made. The second was introduced with the first Athlon XP processors where they combined their 3DNow! with Intel's SSE1 and called it "3DNow! Professional" [6].

SSE1, SSE2 & SSE3

Streaming SIMD Extensions (SSE) was developed by Intel as a reply to AMD's 3DNow! It also replaced the by Intel's generally disappointing MMX set instructions which had two major flaws. First it re-used existing floating-point registers making the CPU unable to work with both floating-point and SIMD data at the same time, and second it only worked with integers. SSE fixed the second issue, that it didn't use floating point numbers, but the the first issue was not fixed until the release of SSE2.

SSE also adds eight 128-bit registers called XMM0 through XMM7 that contains four 32-bit single precision floating-point numbers.

SSE2 adds support for 64-bit double precision floating-point and for 64-, 32-, 16- and 8-bit integer operations on the eight XMM registers, this made the MMX instruc-

tion set redundant. This was on the version that Intel developed for their Pentium 4 in 2001. AMD also added support for SSE2 to their Opteron and Athlon 64 in 2003 but they also extended the SSE2 with 8 more XMM registers to a total of 16 to be used while running in 64-bit mode.

SSE3 was introduced with the 64-bit Pentium 4 which added 13 new instructions that simplified the implementations of a number of DSP and 3D operations. The SSE3 is not supported in the 64-bit AMD processors [6].

SIMD

Single Instruction, Multiple Data (SIMD) is a computing term that refers to operations that efficiently handling of large quantities of data, e.g. adding the same value to large number of data points. SIMD is commonly referred as multimedia instruction set. MMX, SSE, SSE2, SSE3 and 3DNow! are all SIMD instruction sets [6].

1.3 Memory Hierarchy

1.3.1 Instruction, Data and 2nd Level Cache

As many modern computers, the memory hierarchy consist of three layers. The fastest is the processor cache, which is made of SRAM, main memory comes second which is made of DRAM and last and slowest is storage memory which is a magnetic disk.

The processor cache is divided into two parts, L1 cache, or primary cache where the instructions and registers are stored and L2 cache, or secondary memory, which is the link between the L1 cache and main memory. SRAM that the processor cache is made of have a typical access time of 0.5 – 5 ns.

Data that is needed by the processor, for example programs, is loaded into the main memory. DRAM have a typical access time of 50 – 70 ns.

In the storage memory is data such as the OS, programs and files stored with a typical access time is 5 – 20 ms.

The AMD Athlon 64 has a 128 kB L1 cache, where 64 kB is for instructions and 64 kB for the registers. It also has 1 MB on-die L2 cache [4, 5].

1.4 Execution

The AMD Athlon 64 processors has 3 integer execution units, where only one have a multiplier and each with an eight entry scheduler. One floating-point execution unit with three pipelines (FADD, FMUL, FMISC) and a 36 entry scheduler.

1.4.1 Pipeline Organization

The first step is to fetch 16 bytes of instruction code from the instruction cache, then it will pick three instructions from the data stream and decode them either as a direct path operation (RISC like) or a vector path operation. The vector path operation is a more complex CISC like operation, which has to be loaded from a microcode ROM

as a microprogram. The hammer architecture introduces a new class of instructions, the double dispatch operations. They are split into two individual operations at the end of the decode stage, thus making the processor capable of starting six operations in a three-way pipeline. The instructions are then packed back into three operations in the following pack stage before the final decode stage.

The operations are then sent to the execution schedulers where they await their dependencies. The schedulers operates in an out-of-order manner meaning, that they will execute a operation in their buffer as soon as their dependencies are solved. When the operations has been executed they will be placed in an reorder buffer where they will wait for all the operations that should have been executed before themselves, prior to being stored in the registers [3, 4].

1.4.2 Branch Prediction

The AMD Athlon 64 has dynamic branch prediction with a 2,048 entry Branch History Table (BHT) and uses the 2-bit Smith Algorithm. The BHT contains information if a branch was recently taken or not. The 2-bit Smith algorithm uses 2 bits to remember if a branch was taken or not, which implies that a branch has to be predicted wrong two times in a row to change prediction for the branch. This implies that you get a much more accurate branch prediction [4, 5].

1.5 References

[1] ADVANCED MICRO DEVICES. (AMD Athlon 64 processor) Key architectural features, June 17, 2005. http://www.amd.com/us-en/Processors/-ProductInformation/0,,30_118_9485_9487%5E9493,00.html.

[2] CLEVELAND, S. AMD x86-64 technology. White paper, Advanced Micro Devices, Aug. 2000. Available at http://www.amd.com/us-en/assets/-content_type/white_papers_and_tech_docs/x86-64_wp.pdf.

[3] HANS DE VRIES. Understanding the detailed architecture of amd's 64 bit core, Sept. 21, 2003. http://chip-architect.com/news/-2003_09_21_Detailed_Architecture_of_AMDs_64bit_Core.html#4.

[4] LUDLOFF, C. sandpile.org – AA-64 implementation – AMD K8. Web page, July 26, 2004. http://www.sandpile.org/impl/k8.htm.

[5] PATTERSON, D. A., AND HENNESSY, J. L. *Computer Organization & Design: the Hardware/Software Interface*, second ed. Morgan Kaufmann Publishers, Inc., San Francisco, California, 1998.

[6] WIKIPEDIA. 3DNow! – Facts, Info, and Encyclopedia article. Web page, Aug. 31, 2005. http://www.absoluteastronomy.com/encyclopedia/3/3d/3dnow3.htm.

2. AMD Athlon X2 Dual-Core

2.1 Introduction

The AMD Athlon X2 Dual-Core processor was released on the ninth of May, 2005 and was a natural successor to the AMD Athlon 64, which in turn was based upon the much celebrated Athlon XP series. AMD's goal has since the late 1990s, when they announced their current architecture, been to put multiple cores on a single processor. The AMD64 architecture, which the Athlon X2 Dual-Core processor is based upon, was designed from the ground up to accommodate the use of multiple cores on a single processor [2].

2.1.1 Overview of the Processor

The Athlon X2 Dual-Core processor has, opposed to Athlon 64 two logical cores for symmetric multiprocessing. Because the Athlon 64 X2 actually consist of two Athlon 64 cores mounted on one and the same chip with some additional control logic, it has inherited the register structure from the AMD64 architecture which is used in Athlon 64. The AMD64 architecture extends the legacy of x86 architecture with sixteen new 64-bit general-purpose integer registers, sixteen new 128-bit XMM[1] registers for enhanced multimedia which includes SSE/SSE2/SSE3[2] and full 64-bit virtual addressing with 52 bits physical memory addressing. Each core in the processor is equipped with 128 KB L1 cache, 64 KB for instructions and 64 KB for data. It also has, depending on model, 512 KB or 1024 KB L2 cache [1].

When using the shared resources, the two cores communicate via the special Crossbar-switch and the System Request Queue. They interact with each other on the same level in order to solve cache coherency issues without the involvement of the system bus or the memory bus. Furthermore, communication with the I/O peripherals and the hard drives is done using a single HyperTransport link running at 2 GHz, 1 GHz full duplex to be more precise, and communication with the RAM via the shared Memory Controller at a speed of 6.4 GB/sec [4, 3]. Because the Memory Controller is built in, the processor does not have to go via the northbridge to access the RAM, this means that requests reach RAM faster and data is returned from the RAM more quickly. One effect of the built in Memory Controller is that it benefits from the frequency of the processor. Thus, when the processor clockspeed increases the Memory Controller will go faster, see Figure 2.1 on the following page [5].

[1] eXtended Memory Management.
[2] Streaming SIMD Extensions.

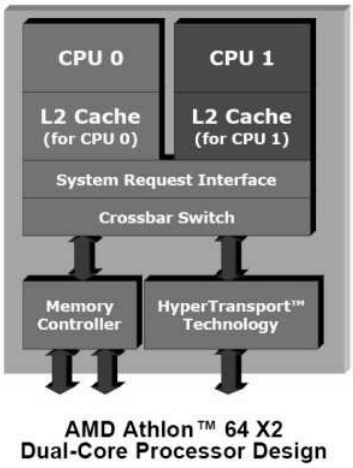


Figure 2.1: AMD Athlon X2 Dual-Core (from [4])

2.2 Execution

2.2.1 Pipeline

The Athlon X2 Dual-Core has the same pipeline as the Athlon 64. When compared to the old Athlon XP, it is evident that the pipeline has been increased from ten to twelve stages, see Table 2.1 on the next page. The new stages allow the processor to scale to higher clockspeeds then the Athlon XP. The main difference between Athlon X2 Dual-Core and Athlon XP pipelines lies in the first stages [5].

The fetch phase of the Athlon 64 X2 Dual-Core processor fetches 16 byte of instructions from the L1 cache. The fetch phase is divided in two stages to decouple the clockspeed from the L1 cache. In the Pick stage of the pipeline the processor moves the 16 bytes into a 32-byte buffer, appending the new 16 byte with the previous 16-byte group. Then the processor scans the 32-byte buffer after instructions and sorts the instructions into one of two types; those who can be decoded by the Fastpath decoder and those who must be decoded by the microcode engine. The Fastpath decoder and the microcode engine translates the x86 instructions into an internal format that looks like a RISC-format which is easier for the processor to manage and schedule. The Fastpath decoder can handle the shortest and simplest x86 instructions and it can handle up to three instructions at the same time. The microcode engine handles the more complex instructions but it can only handle one instruction per clock cycle [5].

2.2.2 Branch Prediction

The branch prediction of the Athlon 64 X2 architecture has, just like the Athlon XP, two main structures consisting of a local history table and a global history table. The local history table keeps track of the execution history of each individual branch in order to predict whether the branches will be taken or not taken. The global history table keeps

	Athlon XP	Athlon 64 X2 Dual-Core
1	Fetch	Fetch1
2	Scan	Fetch2
3	Align1	Pick
4	Align2	Decode1
5	Decode1 (EDEC)	Decode2
6	Decode2 (IDEC/Rename)	Pack
7	Schedule	Pack/Decode
8	AGU/ALU	Dispatch
9	L1 Address Generation	Schedule
10	Data Cache	AGU/ALU
11		Data Cache 1
12		Data Cache 2

Table 2.1: AMD Athlon XP and Athlon X2 Dual-Core pipelines (from [5])

track of the history of a large number of branches and predicts which way each of these will go by correlating them to the execution histories of the other branches in the table. Furthermore, a branch selector table chooses between the two tables depending upon which of the two prediction schemes that has produced the most accurate result for each branch. This way, superpipelined machines like the Athlon 64 X2 can be provided with just what it needs; a very accurate branch prediction [5].

When compared to the Athlon XP, the number of branch selectors of the Athlon 64 X2 is twice as many and the size of the global history table has quadrupled to over 16 000 entries, which results in an five to ten percent improvement in branch prediction accuracy over the Athlon XP [5].

When the branch predictor decides that a particular branch should continue to execute, it needs to know the address, which can either be specified in the branch instruction itself or be the result of a calculation to which it should jump. In cases where the branch target is the result of a calculation, the Athlon 64 X2 stores the calculated target address in yet another essential structure in the Athlon 64 X2's, as well as the Athlon XP's, branch prediction units, that is called the branch target buffer. As a result, no recalculation of the branch target is needed when, later on in a program execution, a particular branch is reencountered and predicted as taken. Instead it retrieves the address from the branch target buffer [5].

Both the Athlon XP's and the Athlon 64 X2's branch target buffer can hold information about 2048 branches. The branch target buffer of the Athlon 64 X2 is backed up by a branch target address calculator that diminishes the penalty caused by a wrongful prediction when the branch target buffer is holding the incorrect branch target for a branch. This is done by comparing the result given by the branch target address calculator with the branch target produced by the branch target buffer; if the two do not match, then the branch predictor calls mispredict and thereby eludes penalty cycles caused by a wrongful prediction mentioned above [5].

2.3 Comparing to the Intel Pentium D

The Pentium D is Intel's contribution to the dual core technology and is the main competitor to the Athlon 64 X2 Dual-Core. The main difference between these two is that the Pentium D has a much longer pipeline, 31 steps compared to 12 steps of the Athlon X2, which in turn leads to an advantage in clock frequency for the Pentium D. The higher clock frequency of the Pentium D has some disadvantages though, it has a larger power consumption then the Athlon X2 and is therefore generating more heat. Another difference is that the Pentium D needs a new Northbridge because it must handle the communications between the cores. All communication between the RAM and the processor must also go through the Northbridge because the Pentium lacks the integrated Memory Controller.

Generally it seems that most people that have compared these two processors have come to the same conclusion; the Pentium D is the best choice when running multiple applications simultaneously on a system due to its HyperThreading technology but when running a single application or two applications the Athlon 64 X2 is the best choice with up to 30 % better performance [6].

2.4 References

[1] ADVANCED MICRO DEVICES. *Athlon 64 X2 Dual-Core Product Data Sheet*, May 2005. Also available at http://www.amd.com/us-en/assets/content_type/-white_papers_and_tech_docs/33425.pdf.

[2] ADVANCED MICRO DEVICES. *Multi-core processors - the next evolution in computing*, May 2005. Also available at http://multicore.amd.com/WhitePapers/-Multi-Core_Processors_WhitePaper.pdf.

[3] FREEMAN, V. Athlon 64 x2 4800+ processor review. Web page, June 22, 2005. http://www.sharkyextreme.com/hardware/cpu/article.php/3514901.

[4] GAVRICHENKOV, I. AMD athlon 64 x2 4800+ dual-core processor review. Web page, Aug. 5, 2005. http://www.xbitlabs.com/articles/cpu/display/athlon64-x2.html.

[5] STOKES, J. Inside AMD's Hammer: the 64-bit architecture behind the Opteron and Athlon 64. Web page, Feb. 1, 2005. http://arstechnica.com/articles/paedia/cpu/amd-hammer-1.ars.

[6] VÖLKEL, F., TÖPELT, B., AND SCHUHMANN, D. Stress test: Athlon 64 x2 4800+ vs. pentium 840 ee. Web page, June 24, 2005. http://www.tomshardware.com/cpu/20050603/index.html.

3. ARM7

3.1 Introduction

The ARM7 family of processors is one of the most widespread in the world. It is used in many types of embedded systems, especially in portable devices due to its low power consumption and reasonable performance[1]. The ARM architecture features a 32-bit RISC[2] processor with 37 pieces of 32-bit registers, 16 of which is available for the programmer. Due to its simple structure it has reasonably good speed/power consumption ratio.

The ARM company was founded in 1990 after a collaboration project between Apple and Acron on the existing ARM[3] that produced such significant results that it made the companies come together and start the new ARM[4] company. This work eventually led to the release of the ARM6 family which is a predecessor to the ARM7 family. The most successful implementation so far has been the ARM7TDMI which can be found in millions of mobile phones, mp3 players, portable game consoles, to name only a few. Today ARM processors account for 75 % of all 32-bit embedded CPUs [5].

Figure 3.1: ARM7TDMI (from [1])

[1] MIPS/Watt.
[2] Reduced Instruction Set Computer.
[3] At the time Acron RISC Machines.
[4] Advanced RISC Machines.

3.2 Registers

The ARM7 is a load-store architecture. This means that in order to perform any data processing instructions the data first has to be moved from the memory into a central set of registers. After this the instructions can be executed and the data moved back into the memory. The ARM register structure is dependent of what type of mode is selected. There are seven different modes available in the ARM processor, each of which has their own register availability. In the table below is an overview of the registers available in the different states.

sys & usr	**fiq**	**svc**	**abt**	**irq**	**und**
R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7
R8	* R8_fiq	R8	R8	R8	R8
R9	* R9_fiq	R9	R9	R9	R9
R10	* R10_fiq	R10	R10	R10	R10
R11	* R11_fiq	R11	R11	R11	R11
R12	* R12_fiq	R12	R12	R12	R12
R13	* R13_fiq	* R13_svc	* R13_abt	* R13_irq	* R13_und
R14	* R14_fiq	* R14_svc	* R14_abt	* R14_irq	* R14_und
R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)

Table 3.1: Register table for different modes (from [2])

Explanations of the different modes in Table 3.1[5]:

- User (**usr**): Normal program execution state.
- FIQ (**fiq**): Data transfer state.
- IRQ (**irq**): Used for general interrupt services.
- Supervisor (**svc**): Protected mode for operating system support.
- Abort mode (**abt**): Selected when data or instruction fetch is aborted.

[5] Register banking (fields marked with an *) means that there are several copies of the register and whether one is visible depends upon which mode the ARM is in. The reason for having these banked registers is to get faster interrupt handlers and OS services, as these can use their own banked registers without having to save user mode registers.

- System (**sys**): Operating “privilege” mode for user.
- Undefined: (**und**): Selected when undefined instructions are fetched.

As we can see in the table there are 16 32-bit registers available in normal (usr) ARM-mode. R0 to R12 are standard registers. The R13 register is always used for the stack pointer, R14 is the link register (it holds the address to return to from a function) and R15 is the program counter (referred to as PC from now on). The R16 is the Current Program Status Register (CPSR) which holds information about the current state the processor is in.

3.3 Instruction Set

The 32-bit ARM instruction set is fully conditional. This means that every instruction is executed with one of sixteen conditional mnemonic in front of them. For instructions that don’t need any condition the prefix AL (Always) is used. The advantage with this is that you get a diverse instruction set with the possibility to execute every instruction with a condition, 16 different mnemonics that can be applied to the beginning of the instruction. Another benefit with this is that if an instruction does not satisfy the condition it will be handled like a NOP. This is done to make sure a smooth flow through the pipeline is kept.

3.4 Pipelining

The ARM7 has a three-stage pipeline, consisting of fetch, decode and execute. This allows the CPU to execute one instruction while decoding a second and fetching a third instruction.

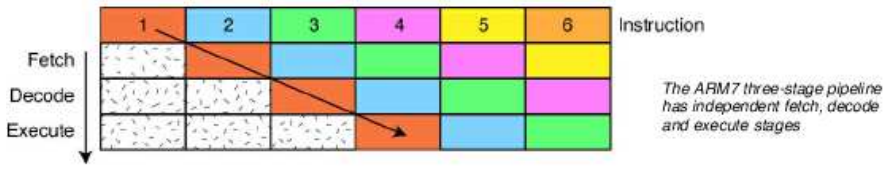


Figure 3.2: Illustration of the pipeline in ARM7 (from [3])

The pipeline performs at its best in linear code, as soon as a branch is encountered the pipeline is flushed and needs to be refilled before full execution speed is resumed. Because of the use of a pipeline the programmer needs to be careful when modifying the PC and remember that it is eight bytes ahead of the instruction being executed.

3.5 Thumbs

As mentioned before, the ARM7 is a 32-bit processor, but with the use of Thumbs[6] it has a second 16-bit instruction set. This makes it possible for the processor to cut the memory use with about 30 % when dealing with 16-bit code. On the other hand the normal ARM instruction set is about 40 % faster when dealing with 32-bit code [4]. So the advantage of Thumbs is the possibility to switch between the two instruction sets when dealing with different code.

3.6 References

[1] ADVANCED RISC MACHINES LIMITED. ARM7TDMI. Web page, Sept. 18, 2005. http://www.arm.com/products/CPUs/ARM7TDMI.html, date visited given.

[2] ATMEL. *Embedded RISC Microcontroller Core*, Nov. 1999. Also available at http://www.atmel.com/dyn/resources/prod_documents/0673s.pdf.

[3] TREVOR MARTIN. *The Insider's Guide To The Philips ARM7-Based Microcontrollers*. Hitex (UK) Ltd., 2005. Also available at http://www.hitex.co.uk/arm/lpc2000book/lpc-ARM-book_srn.pdf.

[4] VILLE PIETKÄINEN. ARM Architecture. PDF presentation, 2002. http://tisu.mit.jyu.fi/embedded/TIE345/luentokalvotEmbedded_3_ARM.pdf.

[5] WIKIPEDIA. ARM architecture, Sept. 18, 2005. http://en.wikipedia.org/wiki/ARM_architecture, date visited given.

[6] Thumbs is to be found in all ARM7 processors with a T following the ARM7.

4. ARM926EJ-S

4.1 Introduction

The ARM design was started on in 1983 by Acron Computers Ltd. and in 1985 the team released development samples called ARM1, and the first real production systems called ARM2 the following year. The ARM2 was possibly the simplest useful 32-bit microprocessor in the world at the time, with only 30,000 transistors.

In the late 1980s, Apple Computer started working with Acron on newer versions of the ARM core. The work was so important that Acron spun off the design team in 1990 into a new company called Advanced RISC Machines. This work would eventually turn into the ARM6 with the first models released in 1991. Apple used the ARM6-based ARM 610 as the basis for their Apple Newton PDA in 1994, while Acron used it as the main CPU in their RiscPCs.

The most successful implementation has been the ARM7TDMI with hundreds of millions sold for use in cellular phones. DEC licensed the design and produced the StrongARM. At 233MHz this CPU only consumed 1 watt off power.

The ARM7 family also includes the ARM7TDMI-S and ARM7EJ-S cores and the ARM720T macrocell, each of which has been developed to address different market requirements. This family was later developed into the ARM9 series [10].

4.1.1 ARM9 Processor Family

The ARM7 has only 3 pipeline stages, this was later increased to 5 stages for the ARM9. The ARM9 processor family is built around the ARM9TDMI processor and incorporates the 16-bit Thumb instruction set, which improves code density by as much as 35 % [7].

The ARM926EJ-S processor, which was released on June 13, 2001 [2], and can for example be found in the cellular phone V800 by Sony Ericsson, has extensions such as Jazelle and Digital Signal Processing (DSP). The ARM DSP extensions that offer enhanced 16- and 32-bit arithmetic capabilities within the functionality of the CPU, improves performance and flexibility.

ARM's Java acceleration technology consist of two parts: Jazelle [9], a hardware component and JTEK, a highly optimized software component. These components must be together to provide a complete Java acceleration solution. ARM Jazelle technology provides hardware Java acceleration to deliver significantly higher performance. It dynamically executes some 90 % [8] of Java bytecode directly on the core to give both performance improvement and power savings.

IBM, Texas Instruments, Nintendo, Philips, Atmel, Sharp, Samsung, etc have also licensed the basic ARM design for various uses. The ARM chip has become one of the most used CPU designs in the world, found in a wide variety of devices such as: cell phones, cars, toys, routers etc. Today it accounts for over 75 % [5] of all 32-bits embedded CPUs.

4.2 Overview of the Processor

The different details of the ARM926EJ-S that are discussed in this section can be observed in Figure 4.1. More in-depth information can be found in [3, 4, 6].

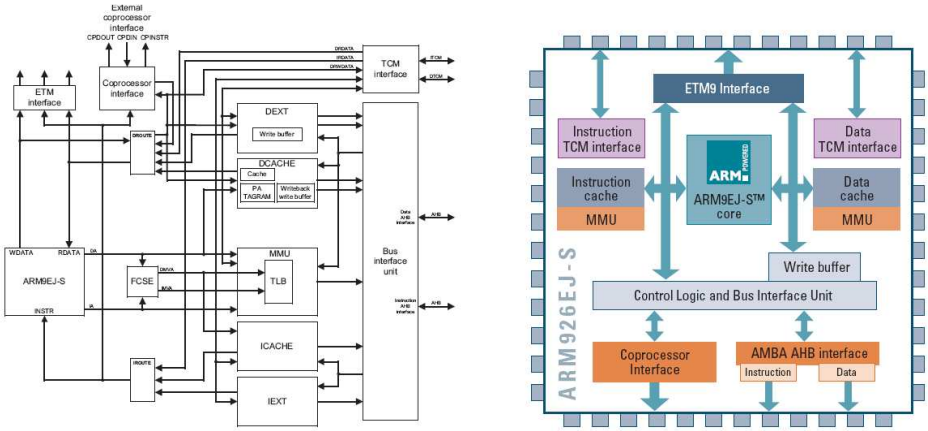


Figure 4.1: ARM926EJ-S overview (from [3, 6])

4.2.1 Tightly Coupled Memory (TCM)

There is a separate interface, the TCM, which the processor provides access too. The TCM can be used for storing instructions and data. It is designed to allow single cycle access that can be extended by a wait signal which in turn allows nonzero wait state memory to be used, and for DMA-type access to occur.

There is no restriction on the type of memory or memory system attached to the interface other than that it must not contain read-sensitive locations. The physical addresses and write accesses are protected based on the information stored in the Memory Management Unit (MMU). The TCMs are typically used to store critical code and data for which deterministic access times are required, which is not not always the case for code or data that resides in a cacheable region of memory [4].

4.2.2 Memory Management Unit (MMU)

The MMU supports a demand page virtual memory system required by operating systems such as Linux and Windows CE. The MMU provides the access protection mechanism for all memory access. The address translation, protection and region type information is stored in a series of page tables in main memory. The MMU contains hardware for automatically reading and processing these tables, and storing the resulting Translation Look-aside Buffer (TLB) [4].

4.2.3 Caches

The ARM926EJ-S uses separate caches for instructions and data, allowing both an instruction fetch and a load/store access to take place simultaneously. The cache is constructed using standard compiled SRAM blocks, allowing a wide range of processes and libraries to be targeted. The size of each cache can be selected in 4, 8, 16, 32, 64 and 128 kB range. The caches are virtually addressed, with each cache line containing eight words. A cacheable memory region can be defined as either being write-back (copy-back) or write-through. This is controlled in the page table entry for the region [4].

4.2.4 Coprocessor

ARM926EJ-S supports the connection of an on-chip coprocessor trough an external coprocessor interface. All ARM processors feature a coprocessor interface to allow the developers the flexibility to enhance the feature set of the core, adding support for new features and extensions to the instruction set [4].

Such an extension is the DSP instruction set that provides the following features:

1. 16-bit data operations
2. Saturating, signed arithmetic – on both 16- and 32-bit data
3. Enhanced Multiply Accumulate (MAC) operations

Multiply instructions are processed using single-cycle 32x16 implementation. There are 32x32, 32x16 and 16x16 multiply instructions or Multiply Accumulate (MAC), and the pipeline allows one multiply operation to start each cycle.

4.2.5 Registers

The processor core consists of a 32-bit datapath and associated control logic. The datapath contains 32 registers coupled to a full shifter, Arithmetic Logic Unit (ALU), and multiplier. At any time, 16 registers are visible to the user. The remainder are banked registers used to speed up exception processing [3].

4.3 Execution

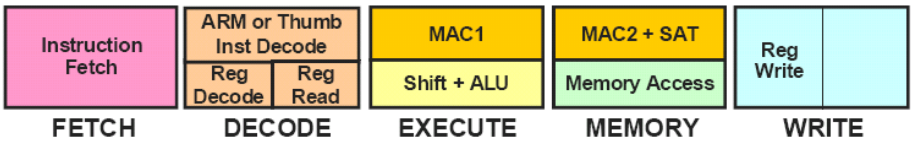


Figure 4.2: ARM926EJ-S pipeline (from [1])

The processor core implements a 5-stage pipeline design. It has a Harvard architecture, meaning that data accesses do not have to compete with instruction fetches for the use of one bus. Result-forwarding makes it faster in the sense that results from the ALU and data loaded from memory gets fed back immediately to be used by the following instruction.

In this pipeline design, dedicated pipeline stages have been added for memory access and for writing results back to the register bank, see Figure 4.2 on the preceding page. Register read has been moved back into the decode stage. These changes allow for higher clock frequencies by reducing the maximum amount of logic which must operate in a single clock cycle.

4.4 Performance

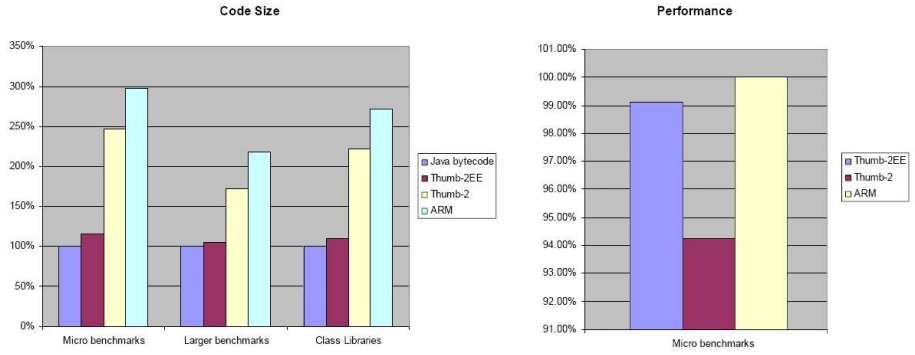


Figure 4.3: ARM926EJ-S performance comparison of both size and speed (from [9])

Figure 4.3 illustrates results from a compilation of Java bytecode using the ARM Ahead Of Time (AOT) compiler. The compiler output includes ARM code, Thumb-2 code and Thumb-2EE (Jazelle RCT) code, the results are shown normalized against the original Java bytecode. Note that this compiler is highly optimized for code density rather than performance, so code bloat for all ISAs is the low end of the possible range [1].

4.5 Special

Some of the richer features of this processor is that it can execute 80 % of javas code directly and that it has extensions for both encoding and decoding of MPEG4.

4.6 References

[1] ADVANCED RISC MACHINES LTD. *Product Comparison*, Sept. 2, 2000. Also available at http://www.arm.com/comparison-arm7-arm9-v1.pdf.

[2] ADVANCED RISC MACHINES LTD. ARM discloses technical details of ARM926EJ-S core. Web page, June 13, 2001. http://www.us.design-reuse.com/news/news631.html.

[3] ADVANCED RISC MACHINES LTD. *ARM926EJ-S Product Overview*, Sept. 19, 2001. Also available at http://www.arm.com/pdfs/DVI0035B_926_PO.pdf.

[4] ADVANCED RISC MACHINES LTD. *ARM926EJ-S Technical Reference Manual*, Jan. 26, 2004. Also available at http://www.arm.com/pdfs/DDI0198D_926_TRM.pdf.

[5] ADVANCED RISC MACHINES LTD. ARM CPU:s information. Web page, Sept. 19, 2005. http://www.arm.com/products/CPUs/index.html, date visited given.

[6] ADVANCED RISC MACHINES LTD. ARM926EJ-S. Web page, Sept. 19, 2005. http://www.arm.com/products/CPUs/ARM926EJ-S.html, date visited given.

[7] ADVANCED RISC MACHINES LTD. ARM926EJ-S architecture. Web page, Sept. 19, 2005. http://www.arm.com/products/CPUs/architecture.html, date visited given.

[8] ADVANCED RISC MACHINES LTD. Foundry program. Program flyer, Sept. 19, 2005. Also available at http://www.arm.com/pdfs/foundry%20program%20flyer%2099-6.pdf.

[9] PORTHOUSE, C. *Jazelle technology: ARM acceleration technology for the Java Platform*. Advanced RISC Machines Ltd, Sept. 2004. Also available at http://www.arm.com/pdfs/Jazelle%20RCT%20White%20Paper%20_final%201.0_.pdf.

[10] WIKIPEDIA. ARM architecture. Web page, Sept. 21, 2005. http://en.wikipedia.org/wiki/ARM_family.

5. Intel Pentium Pro

5.1 Introduction

The Pentium Pro processor by Intel is a sixth-generation x86 microprocessor that replaced its predecessor the original Pentium processor. The difference between the Pentium Pro and its predecessor is, despite the name, actually quite big. Based on the then new P6-core Pentium Pro uses the same core as the later Pentium II, Pentium III and Pentium M. So when the Pentium Pro was introduced in 1995 it was a big step for the Pentium processor family. Pentium Pro introduced several unique architectural features that had not been seen in any PC processor before.

5.2 Overview of the Processor

Using a new way of processing instructions and several other technical advances the Pentium Pro achieves approximately 50 % higher performance than a Pentium of the same clockspeed. Some of the advances that contribute to the increased performance are [4]:

1. **Superpipelining:** To compare with the Pentium processor, which uses 4 execution steps in pipelining, Pentium Pro uses 14.
2. **Out of Order Completion:** The executions can be completed out of order.
3. **Register renaming:** This makes parallel performance of the pipelines better.
4. **Speculative Execution:** To reduce pipeline stall time in the core, the Pentium Pro uses speculative execution.
5. **Superior Branch Prediction Unit:** The Pro has a branch target buffer that is two times the Pentium's and its accuracy is increased.
6. **Wider Address Bus:** With a 36 bits address bus the Pentium Pro can address up to 64 GB of memory.
7. **Integrated Level 2 Cache:** A then new level 2 cache memory is used in the Pentium Pro. It uses integrated level 2 cache, which runs at the full processor speed, instead of using the motherboard-based cache that runs at the speed of the memory bus. The cache memory of Pentium Pro is also non-blocking, this makes the processor continue without waiting on a cache miss.
8. **32-Bit Optimization:** When the Pentium Pro was released most software used 32-bit code and because of that the Pentium Pro is optimized for that.

5.2.1 Implementations

When the Pentium Pro was released it was mainly designed for high-end servers and workstations, because it was optimized for 32-bit code, preferably those computers that was using windows NT and UNIX systems. This made the processor poorly suitable for 16-bit code. When the Pro was released it was not well positioned to compete in the desktop market, because most desktops used Windows 3.1, which is using only 16-bit

code, and Windows 95, which was new at that time and therefore also using most 16-bit programs. To make a new processor improvement suitable for the desktop market Intel released Pentium MMX as the successor for the original Pentium. Today Pentium Pro still is popular for servers because its low costs [6].

5.3 Memory Hierarchy

5.3.1 Register

The registers of the processor is the at the top level of the memory hierarchy. It is the register that provides the fastest access to data possible on the 80x86 CPU. The register is also the smallest memory in the hierarchy of the CPU with a size of 32 bits for the Pentium Pro. The register is a very expensive memory [1].

5.3.2 Level 1 Cache

The next level of memory in the CPU is the level 1 cache. It is a fixed size memory provided by Intel and it can not be expanded. The size of this memory is 16 KB for the Pentium Pro, 8 KB for data and 8 KB for instruction. The cost per byte of level 1 cache is much lower than the register [1].

5.3.3 Level 2 Cache

Below the level 1 cache in the hierarchy is the level 2 cache. The Pentium Pro has integrated level 2 cache which makes the cache run at the same speed as the processor (usually 180 or 200 MHz) instead of just the system bus speed (60 or 66 MHz). Because of the integrated cache memory it is not possible to expand this memory without replacing the whole CPU. Another drawback of this type of cache memory is that it is very expensive to manufacture the large chip required for the level 2 cache. This is the main reason why Intel did not use integrated level 2 cache when designing the next processor, the Pentium II [1].

5.4 Execution

The Pentium Pro uses the three independent-engine approach which consists of a Fetch/-Decode unit, Dispatch/Execute Unit and a Retire Unit.

5.4.1 Instructions

The handling of instructions in Pentium Pro converts the Complex Instruction Set Computer (CISC) x86 instructions into Reduced Instruction Set Computer (RISC) micro-ops. This conversion is to reduce the inherent limitations in the x86 instructions set, such as irregular instruction encoding and register-to-memory arithmetic operations.

It passes the micro-ops to a out-of-order execution engine that determines if the instructions are ready for execution and else if not, the micro-ops are shuffled to prevent pipeline stalls [5].

5.4.2 Pipelining

The Pentium Pro was the first in the x86 family that implements superpipelining. The 14 stages trades less work per pipe stage for more stages. The pipe stage time is 33 percent less then the original Pentium, that results in a higher clock rate on any given process. The 14 stages are dived into three sections, called the three independent-engine approach. Eight stages consist of the in-order front end section, this handles decoding and issuing of instructions. There are three stages which executes the instructions called out-of-order core. The last three stages handles the in-order retirement. Superpipelining provide simultaneous, or parallel, processing within a CPU. It overlaps operations by moving data or instructions into a conceptual pipe with all stages of the pipe processing simultaneously. As an example, while one instruction is being executed, it is decoding the next instruction [2, 5].

5.4.3 Dynamic Execution

Also a performance improvement for the Pentium Pro is the dynamic execution. This includes branch prediction, data flow analysis and speculative execution. These functions reduces the wasted clock cycles, by making predictions about the program flow. Consider memory latency problems when an instruction to memory is called and the CPU must wait for the data to be read from main memory and return it before executing an instruction on the data. This CPU stalls while waiting for this data. To avoid this latency problem, the Pentium Pro processor “looks-ahead” into the instruction pool at subsequent instructions and will do useful work rather than be stalled. This approach has the side effect that instructions are typically executed out-of-order. A call to memory will take many internal clocks, so the Pentium Pro processor core continues to look ahead that could be speculatively executed, and is typically looking 20 to 30 instructions ahead. Dynamic Execution can be summarized as optimally adjusting instruction execution by predicting program flow, having the ability to speculatively execute instructions in any order, and then analyzing the data flow to choose the best order to execute [2, 5].

5.5 Performance

As already mentioned the Pentium Pro processor is designed to deliver optimal performance for 32-bit software. This made it especially suited for software like CAD, 3D, and multimedia applications. The processor became very popular for high performance desktops and workstations, as well as for large database and enterprise applications on servers. The processor delivers outstanding integer and floating-point performance. Many different benchmark tests exists and Table 5.1 on the following page displays the results from a few of them for the original Pentium, Pentium Pro and Pentium 2.

Table 5.1: Benchmarks, higher values is faster

Processor	CPUmark32	Norton SI32	iCOMP 2.0
Pentium 200MHz	382	44	142
Pentium Pro 150 MHz	420	70	168
Pentium Pro 166 MHz	465	78	186
Pentium Pro 180 Mhz	497	81	197
Pentium Pro 200 MHz(256 KB)	553	90	220
Pentium Pro 200 MHz(512 KB)	611	98	240
Pentium II 233 MHz	640	115	267

As we can see from Table 5.1 the difference between the original Pentium 200 MHz and the Pentium Pro 200 MHz is bigger than the difference between the Pentium Pro 200 MHz and the later Pentium II 233 MHz. This tells something about how fast the Pentium Pro processor was. The high performance had some back draws, and, as mentioned above, the high prize for the integrated level 2 cache was one of them [3].

5.6 References

[1] HYDE, R. *The Art of Assembly Language Programming*. No Starch Press, San Francisco, CA, Sept. 2003, ch. 6: Memory Architecture. Also available at `http://webster.cs.ucr.edu/AoA/Windows/HTML/MemoryArchitecture.html`.

[2] INTEL CORPORATION. *Pentium Pro Family Developer's Manual*, 1996. Also available at `ftp://download.intel.com/design/archives/processors/pro/-docs/24269001.pdf`.

[3] INTEL CORPORATION. *Pentium Pro Processor Performance Brief*, June 1997. `ftp://download.intel.com/design/archives/processors/pro/docs/-24276806.pdf`.

[4] KOZIEROK, C. M. Intel Pentium Pro ("P6"). Web page, Apr. 17, 2001. `http://pcguide.com/ref/cpu/fam/g6PPro-c.html`.

[5] PC TECHNOLOGY GUIDE. Pentium Pro. Web page, Sept.3, 2005. `http://www.pctechguide.com/02procs_Pentium_Pro.htm`.

[6] WIKIPEDIA. Pentium Pro. Web page, Aug. 16, 2005. `http://en.wikipedia.org/wiki/I686`.

6. Intel Pentium III processor

6.1 Introduction

The Pentium III processor is created by Intel, and was released in the end of February, 1999. It is based on the same architecture as its predecessor Pentium II, which in turn was built on the Pentium Pro core. The instruction set used by Pentium III is the IA-32 (Intel Architecture 32 bits), but with the addition of SSE (Serial SIMD Extension). Another difference between Pentium II and Pentium III is the controversial PSN (Processor Serial Number), a unique ID number hardcoded in each processor. Just like Pentium II, Pentium III also exists in two special versions. The slower and cheaper Celeron, and the faster, but more expensive Xeon. Pentium III has now been superseded by the newer Pentium 4 processor, but it is still used in a lot of places [7, 8].

6.2 General Information and History

Pentium III was first introduced in a version called Katmai. It was pretty much the same as Pentium II, but with the addition of the Streaming SIMD Extension (SSE), and an improved level 1 cache controller. It had a speed of 450 and 500 MHz. Later a version called Coppermine was released. It had an improved level 2 cache with lower latency, which made it a lot faster than Katmai. Versions with processor speeds of more than 1 GHz was created. Also the stalls that earlier existed in the instruction pipeline was fixed. Finally Intel released a third version, called Tualatin. It had as much as 512 KB level 2 cache, and existed with processor speeds of 1.4 GHz [8].

One big advantage Pentium III has compared to the older Pentium II processor is the addition of SSE. This is a set of instructions created to extend the MMX instruction set, that was introduced on Pentium II. Both MMX and SSE includes SIMD (Single Instruction, Multiple Data), which allows the processor to apply an instruction to large quantities of data in parallel. This is used to accelerate multimedia applications, since Intel saw a trend in the market requiring just that. Even though MMX was similar to SSE it had the big drawback that it could not handle floating point operations, which SSE could [7, 8, 9].

Another addition to the Pentium III was a Processor Serial Number (PSN), which was a unique ID number hardcoded into each processor. Even though it was possible to disable it, it was shown that web sites could still access the PSN. This caused groups of users, that did not want to be identified that easily, to file complaints against Intel. Eventually Intel had to accept this massive resistance, and declared that no future Intel processors would have the PSN. It is however still included in the Pentium III processors [5].

Intel also released a cheaper version of the Pentium III processor, called Celeron. It was more or less identical to Pentium III, with the significant difference that the level 2 cache was only half as big. This caused Celeron to be slower. On the other hand, a faster and more expensive version called Xeon was released. It had a much larger level 2 cache, up to 2 MB [8].

Since Pentium III processors are not created for some special area or machine, it can be found in regular home computers and laptops. It is also the processor used in Microsoft's gaming console X-box. A special variant of the Tualatin version called Pentium III-S was mostly used in servers where power consumption mattered [8].

6.3 Processor Overview

The first Pentium III processors to be released had frequencies ranging from 450 to 600 MHz. This increased rapidly, and eventually processors with frequencies up to 1.4 GHz existed. The system bus runs at 100 or 133 MHz [8, 4].

The processor has two separate level 1 caches. Both are 16 KB, and one is used for instructions, and the other for data, see Figure 6.1. The level 2 cache is 512 KB on most versions of the Pentium III, but some exist where the cache is only 256 KB, and some where the cache is much larger [3, 4].

There are four different kinds of registers. Eight 32 bit registers for general purpose. Eight 128 bit registers for SSE. Eight 64 bit registers for MMX, and finally eight 80 bit floating points registers [4].

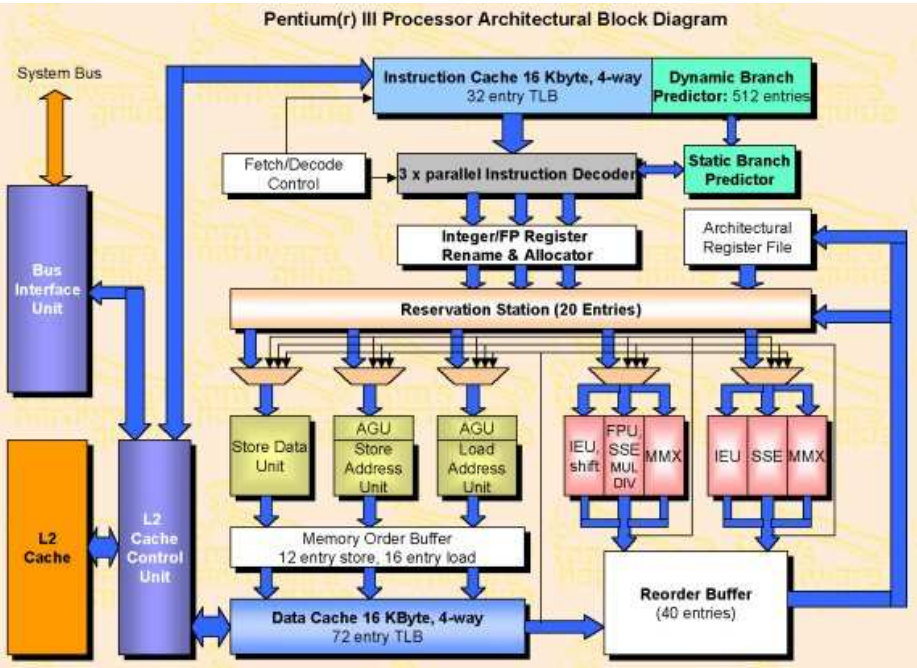


Figure 6.1: The Pentium III architecture (from [6]).

6.4 Architecture and Execution

Pentium III uses the Intel P6 core, which features an out-of-order execution unit. A window is opened on all instructions that are not yet executed, which allows the execute phase of the processor to get a better overview of the stream of instructions. This gives it a better ability to schedule the execution of the instructions optimally. The P6 core breaks the x86 instructions into simpler instructions, called micro-ops. This task is performed by three parallel decoders. When the micro-ops are decoded they are sent to the Reservation Stations (RS). There they will wait until their data operands are available, and are then sent to an execution unit. The P6 core has five execution units, designed to execute different sets of instructions. Two handles ALU, MMX and SSE instructions. One handles Load, one Store Address and one Store Data instructions (see Figure 6.2). When a micro-op has been executed it is returned to the Reorder Buffer, where it will wait for retirement. In this stage data is written back to the memory, and the micro-ops will then retire in-order [6].

There are two problems with this approach, that needs to be handled in order to increase the performance of the processor. The first is that the execution units all have a well-defined set of operations to execute. If a large bulk of instructions of the same kind are sent, that execution unit will cause delays, while the other units are idle. To solve this, instructions often needs to be rearranged to create a balanced stream, that make sure that all five execution units has about the same amount of work. The other problem is to decrease the number of dependency bounds on the micro-ops, to make sure that they do not stall, waiting for their data operands to be available [6].

Pentium III has twelve pipeline steps, plus two out-of-order steps. It also supports Branch Prediction, to further increase the performance [4].

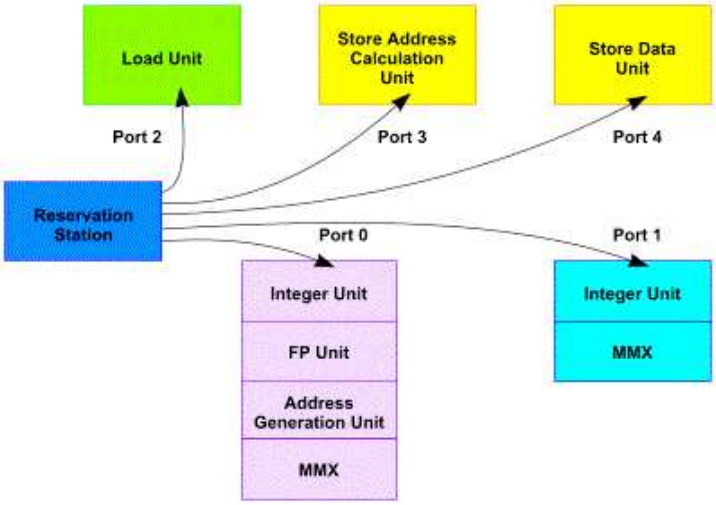


Figure 6.2: The five execution units (from [6]).

6.5 Performance

The performance of the Pentium III processor has been measured in two different ways; SpecFP95 and SpecInt95. The test performed by the Standard Performance Evaluation Corporation (SPEC) was made on 500 MHz processors, and gave the result of an average ratio of 20.7 for SpecInt95, and 14.7 for SpecFP95 [2, 1].

6.6 References

[1] INTEL CORPORATION. SPECfp95: Intel SE440BX2 motherboard (500 MHz, Pentium III processor). Web page, Apr. 21, 2000. http://www.spec.org/cpu95/results/res99q1/cpu95-19990315-03454.html.

[2] INTEL CORPORATION. SPECint95: Intel SE440BX2 motherboard (500 MHz, Pentium III processor). Web page, Apr. 21, 2000. http://www.spec.org/osg/cpu95/results/res99q1/cpu95-19990315-03453.html.

[3] INTEL CORPORATION. Hardware Design: Archives - Intel Pentium III Processor. Web page, July 6, 2005. http://www.intel.com/design/archives/processors/PentiumIII/index.htm.

[4] LUDLOFF, C. sandpile.org – IA-32 implementation – Intel P3 (incl. Celeron and Xeon). Web page, July 12, 2003. http://www.sandpile.org/impl/p3.htm.

[5] THE CENTER FOR DEMOCRACY & TECHNOLOGY. A History of Privacy Issues: Intel Pentium III Processor Serial Number. Web page, Sept. 18, 2005. http://www.cdt.org/privacy/issues/pentium3/, date visited given.

[6] TOMMESANI, S. Intel Pentium III microarchitecture. Web page, 2000. http://www.tommesani.com/IntelPentiumIII.html.

[7] WALKER, A. Pentium III and a brief history of PC processors. *Cyberwalker.net* (Apr. 15, 1999). Available at http://www.cyberwalker.net/columns/apr99/041599.html.

[8] WIKIPEDIA. Pentium III. Web page, Sept. 21, 2005. http://en.wikipedia.org/wiki/Pentium_III.

[9] WIKIPEDIA. Streaming SIMD Extensions. Web page, Aug. 8, 2005. http://en.wikipedia.org/wiki/Streaming_SIMD_Extensions.

7. Intel Pentium D with Dual-Core

7.1 Introduction

Codenamed *Smithfield*, the Pentium® D was first released on May 26th, 2005, with clockspeeds of 2.8, 3.0, and 3.2 GHz. Its predecessor is Pentium® 4 with *Hyper Threading* technology. Other predecessors are the Pentuim® family [3].

7.2 Overview

The Pentuim D processor dual-core features two Pentium 4 Prescott processors on the same die. Initially, the Pentium D processor did not support *Hyper Threading* although some Extreme Editions do [1].

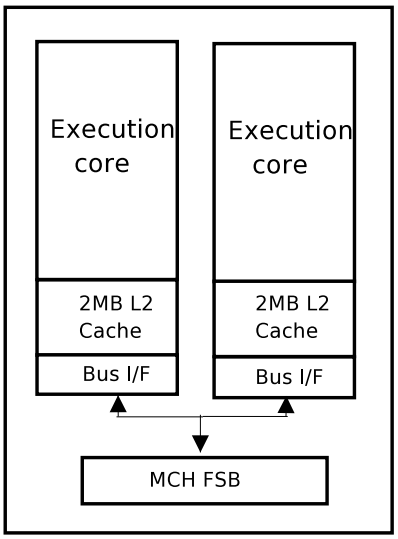


Figure 7.1: Pentium D *Dual-core* from [2]

7.2.1 Specification

The main feature of the Pentium D processor is the fact that it has two executing cores. The difference between that and Pentium 4 with *Hyper Threading* is that with the Pentium 4, processing threads are processed in the same core but with more efficiency than the normal Pentium processors (without *Hyper Threading*). With Dual-core technology, each processing thread is processed in a separate core, making it more easy and more effective to run different applications at the same time (see Figure 7.1). More-

over, Pentium D has Intel's EM64T technology which enables the processor to support 64-bit operating systems and applications. In addition to that, it:

1. Implements the 64-bit architecture.
2. Has 230 million transistors.
3. Features Enhanced Intel Speedstep®. Processor voltage and frequency adjust based on the needs of the application at hand which can result in less power usage and heat production.
4. Has 800 MHz Front Side Bus. High speed between the processor and the other peripherals.
5. Features Execute Disable Bit. Can help to prevent some classes of viruses and worms that exploit buffer overrun vulnerabilities. EDB will render the system more secure.
6. Features 90 nm Process Technology. Newest generation manufacturing process with technologies to help increase processor performance.
7. Has Level 1 cache (two 16 kB Data Caches and two 12 kB Micro-op Execution Trace Caches.
8. Has 2×1 MB Level 2 cache. Each processor core has its own 1 MB cache. This results in reducing the amount of Front Side Bus traffic.

The Pentium D is obviously optimal for server applications. But that is not its only major; it can be used with video editing and other demanding applications. Yet today's programs do not fully support this architecture although Windows and other operating systems such as Suse are compatible. Despite that, Pentium D is compatible with both 32-bit and 64-bit systems [1].

7.3 Registers

The Intel Pentium D processor supports EM64T® technology. EM64T increases the linear address space for software to 64 bits and support physical address space up to 40 bits. The technology also introduces a new operating mode referred to as IA-32e mode. IA-32e mode works in two ways; It either runs a 64-bit system and 32-bit programs or 64-bit operating system with programs that support 64-bits. These are the benefits with 64-bit mode [1]:

1. 64-bit flat linear addressing.
2. 8 additional general purpose registers(GPR).
3. 8 additional registers for streaming.
4. 64-bits-wide GPRs and instruction pointers.
5. Uniform byte-register addressing.
6. Fast interrupt-prioritization mechanism.

7.4 Memory Organization

The memory that the processor addresses on its bus is called **physical memory**.

Physical memory is organized as a sequence of 8-bit blocks, each of which is assigned a unique address called **physical address**. Intel EM64T supports physical address space greater than 64 Gbytes. The actual physical address size of IA-32 processors is implementation specific.

When employing the processor's memory management facilities, programs do not directly address physical memory. Instead, they access memory using on of three memory models [1]:

1. **Flat memory model**– Memory appears to a program as a single, continuous address space which is called a linear address space.
2. **Segmented memory model**– Memory appears to a program as a group id independent address spaces called segments. Code, data, and stacks are typically contained in separate segments. In order for a program to address a byte segment, it issues a logical address that consists of a segment selector and an offset.
3. **Read-address mode memory model**– This model is for the Intel 8086 processor. It is supported to provide compatibility with existing programs written to be run on the 8086 model.

7.4.1 Virtual Memory

With the flat or the segmented memory model, linear address space is mapped into the processor's physical address space either directly or through paging. When using direct mapping (paging disabled), each linear address has a one-to-one correspondence with physical address. Therefore, linear addresses are sent out on the processor without translation. While when using IA-32 architecture's paging mechanism, linear address space is divided into pages which are mapped to virtual memory. These pages are then mapped as needed into physical memory. When an operating system or executive uses paging, the paging mechanism is transparent to an application program. All that the application sees is linear address space [1].

7.5 Execution

The Pentium D processor uses the NetBurst® micro architecture. The Arithmetic Logic Units(ALU) run at twice the processor frequency. Basic integer operation can be dispatched in 1/2 processor clock tick. The processor uses Hyper-Pipelined Technology, which means it has deep pipelines to increase the clock rates, frequency headroom and scalability. The processor also provides an Advanced Dynamic Execution which is a Deep, out-of-order, speculative execution engine. It can manage up to 126 instructions at once, up to 48 loads and more than 24 stores in pipeline. It also has an enhanced branch prediction capability. It reduces the misprediction penalty associated with deeper pipelines and it uses a 4K-entry branch target array and an advanced branch prediction algorithm. The Processor has a new cache subsystem. The first level caches has an advanced execution trace cache that stores decoded instructions. The advanced

execution cache trace removes decoder latency from main execution loops and integrates path of program execution flow into a single line. The L1 cache is a low latency data cache. The L2 cache is a full-speed, unified 8-way level 2 on-die advance transfer cache and its bandwidth and performance increases with the processor family [1].

7.6 Performance

In Table 7.1 are some values that Pentium D processors presented.

Table 7.1: Three Pentium D processors are tested. Test is taken from [2]

Processor	MIPS	MFLOPS
Pentium D 840	17407	4621 FPU / 7950 SSE2
Pentium D 840(4.02 GHz)	18113	5737 FPU / 8711 SSE2
Pentium D 840 64-bit	18191	4970 FPU / 8770 SSE2

7.7 Conclusion

The Pentium D processor with dual-core is a good step towards faster systems. That is why it is important for the next generation operating systems to make use of this technology so that its power can be properly used.

7.8 References

[1] INTEL CORPORATION. *IA-32 Intel Architecture Software Developer's Manuals, Volume 1: Basic Architecture*, Sept. 2005. Available for download at `ftp://download.intel.com/design/Pentium4/manuals/25366517.pdf`.

[2] MIKE D. Intel Pentium D 840 dual-core LGA775 processor review. *pcstats.com* (Aug. 2005). Available at `http://www.pcstats.com/artvnl.cfm?articleID=1838`.

[3] WIKIPEDIA. Pentium D. Web page, Sept. 20, 2005. Available at `http://en.wikipedia.org/wiki/Pentium_D`.

8. Intel Itanium (IA-64 Merced)

8.1 Introduction

Itanium, also called Merced, is the first generation of processors built on the IA-64 architecture. It is the result of a cooperation between Intel and Hewlett-Packard whose goal was to create a 64-bit post-RISC architecture for heavy server applications. It was therefore intended to replace the Xeon series as Intel's foremost server CPU [7].

When the processor was released in June 2001 its performance didn't live up to expectations and had difficulties getting into the market [7].

8.2 Overview

When it was shipped in June 2001 it was available at the speeds 733 and 800 MHz manufactured with a 180 nm process. The system bus used was running at a mere 266 MHz which would prove to be one of Itanium's weak points. That is 33 % slower than the bus used by the Intel Pentium 4 sold at the same time. The CPU consists of approximately 25M transistors excluding the L3 cache which is off-die [7, 5].

At its core the Itanium is basically a RISC design but with multiple functional units enabling it to execute 6 instructions in each cycle (6-wide super scalar). What makes this architecture stand out from the ordinary RISC design is the way instructions are fed and decoded by the CPU [2].

8.2.1 Instruction Set

The instruction format for sending instructions to the CPU is called very long instruction word (VLIW). Each of these words is 16 bytes long and is called a bundle. A bundle is organized in three 41-bit instruction slots and a template field. Each instruction can take one or two slots depending on the size needed. The template field contains information about the instructions in the bundle and it tells if the bundle can be executed in parallel with the next bundle [4].

With this design the CPU can be expanded in later versions to handle more parallel instruction without changing the instruction set. In addition to the basic instructions the IA-64 architecture also contains instructions for multimedia and floating point operations. It is also possible to add more instructions by programming using an extensible firmware interface (EFI) program [6].

8.2.2 Registers

Itanium has a generous amount of registers, 256 of them, 128 each of 64 bits integer (r0 through r127) and 82 bits floating point registers (f0 through f127). Register r0 and register f0 is always set to 0 removing the need for the constant 0. The register f1 always hold the value 1. The CPU also has a feature called register rotation which is

used to rename registers in hardware to make as efficient use of registers as possible, thus improve performance [6].

8.3 Memory

8.3.1 Caches

The cache memory of the Itanium consists of 2 or 4 MB off-die L3 cache, 96 KB L2 cache and 32 KB L1. The L1 cache is divided in half between instructions and data. As can be seen in Figure 8.1 the latency of the L3 cache is quite high which is one of the main bottlenecks of the CPU. The other latencies in the memory hierarchy is also quite high and the bandwidth of the memory bus is a bit poor. All these things were vastly improved in the next generation of the CPU [3, 7].

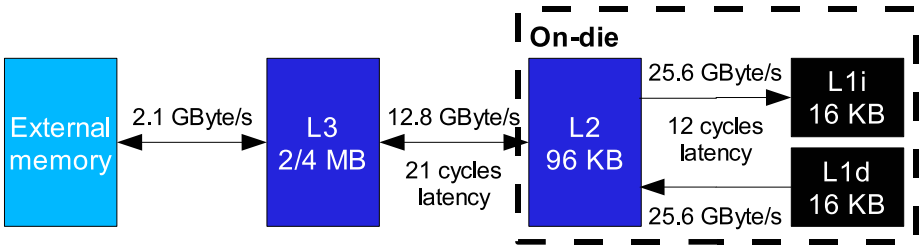


Figure 8.1: An illustration of the memory organization of the Itanium.

8.3.2 Virtual Memory

Itanium supports the following page sizes: 4 KB, 8 KB, 16 KB, 64 KB, 256 KB, 1 MB, 4 MB, 16 MB and 256 MB. It implements 54 virtual address bits and 44 physical address bits [5].

8.4 Execution

8.4.1 Pipeline

The pipeline of the Itanium is a highly parallel 10-stage pipeline. It is able to execute up to 6 instructions in every clock cycle. Figure 8.2 on the facing page shows the different stages in the pipeline and Table 8.1 on the next page contains brief explanations of each stage [2, 1].

The pipeline is divided into two groups of units, the front-end and the back-end. The front-end consists of the IGP, FET and ROT units which is responsible for fetching instruction bundles and doing branch prediction. It also does pre-fetching which hides fetch latency. The back-end consists of the other units and is responsible for the execution of the instructions. The communication of bundles is done via a decoupling

buffer with room for 8 bundles. This decoupling of the front-end and the back-end makes it possible for the front-end to keep running, fetching and pre-fetching if the back-end would stall [2, 1].

Figure 8.2: The 10 stages of the Itanium pipeline

Table 8.1: Explanations of the stages in the Itanium pipeline.

State	Explanation
IPG	Instruction Pointer Generation. An address for the instruction pointer is generated.
FET	Fetches bundles and passes them on to the next step.
ROT	Determines which order bundles will be placed in the buffer which passes them on to the back-end of the pipeline.
EXP	Disperses the instructions in the bundles to 9 issue ports.
REN	Register renamer.
WLD	Word-line decode.
REG	Read operands from the register file.
EXE	Instruction execution.
DET	Exception detection.
WRB	Write back.

8.4.2 Out-of-Order Execution

To create an efficient super scalar architecture it is necessary to implement an out-of-order decoder system which analyses the dependency between instructions to see what instructions can be executed i parallel. This is also the case in the Itanium, however instead of using some complex logic to accomplish this Itanium relies on the compiler to perform this task. The compiler is responsible for organizing the instructions so they can be executed in parallel to the greatest extent possible. This method is called explicitly parallel instruction computing (EPIC) [1, 6].

8.4.3 Branch Prediction

In the Itanium design the complexity of branch prediction has been moved from the CPU to the compiler (EPIC). The CPU relies on the compiler to put information about this into the code being executed. The benefit of this is that since the compiler has a lot more time to decide on a branch and also access to the complete code it can often make better decisions. It also speeds up the execution of the code since the same decision doesn't have to be done every time the code is being executed. However the

runtime behavior of a program is not always obvious in the code that generates it. This sometimes makes it impossible for the compiler to correctly predict what probably could be predicted at runtime by some logic on the CPU itself [1, 6].

Using the information handed from the compiler the pre-fetch engine of the Itanium can see to that the instructions needed can be fetched from the L1 cache [1, 6].

8.5 Performance

The Itanium running at 800 MHz scores about 50 points in Spec95int and about 90 points in Spec95fp. That was, at the time of launch, not enough to persuade most of the customers. The reasons for this failure are many, a few has already been mentioned earlier in Section 8.3.1 on page 34. The main reason is probably the lengthy delays which dogged the development an resulted in the processor being old before if even reached the market [6, 7].

8.6 References

[1] HODGIN, R. C. Intel's Itanium. *geek.com* (June 8, 2001). Available at http://www.geek.com/procspec/features/itanium/ (visited Sept. 22 2005).

[2] INTEL CORPORATION. *Intel Itanium Processor Microarchitecture Overview*, Oct. 1999. Also available at http://www.dig64.org/More_on_DIG64/-microarch_ovw.pdf.

[3] INTEL CORPORATION. *Intel Itanium processor reference manual for software optimization*, Nov. 2001. Also available at http://www.intel.com/design/itanium/downloads/24547403.pdf.

[4] INTEL CORPORATION. *Intel Itanium processor-specific application binary interface*, May 2001. Also available at http://www.intel.com/design/itanium/downloads/24537003.pdf.

[5] INTEL CORPORATION. *Intel Itanium product brief*, 2001. Also available at http://www.intel.com/design/itanium/itanium/ItaniumProdBrief.pdf.

[6] WIKIPEDIA. IA-64, Sept. 1, 2005. http://en.wikipedia.org/wiki/IA-64 (visited Sept. 22 2005).

[7] WIKIPEDIA. Itanium, Sept. 22, 2005. http://en.wikipedia.org/wiki/Itanium (visited Sept. 22 2005).

9. IBM PowerPC G5

9.1 Introduction

PowerPC 970FX, also known as PowerPC G5, is a 64-bit processor from IBM used in Apple computers. This is a paper with some relevant information about G5. Apple introduced the G5 in June 2003, and made it one of the first 64-bit processors for the personal computer market [2]. G5 is based on the core from IBM's POWER4 that uses the same vector engine, VMX, as in G5's predecessor G4 [4].

PowerPC 970 is also called G5. The difference between 970 and 970FX is that 970 is built using a 130 μm fabrication process and 970FX a 90 μm [4]. In Apples white pages they are referring to the 970FX as G5 [2].

Figure 9.1: PowerPC G5 (from [4])

9.2 Overview of the Processor

G5 uses the Double Data Rate frontside bus for faster communication between memory controller and processor. Instead of transporting data in one direction at the time, G5 uses a dual-channel frontside bus that has 32 bits in each direction [2]. For a 2.7 GHz G5, the frontside bus operates at 1.35 GHz and get a theoretical bandwidth of 10.8 GB per processorsecond.

In Figure 9.2 on the next page you can see an abstract view over G5's architecture [2].

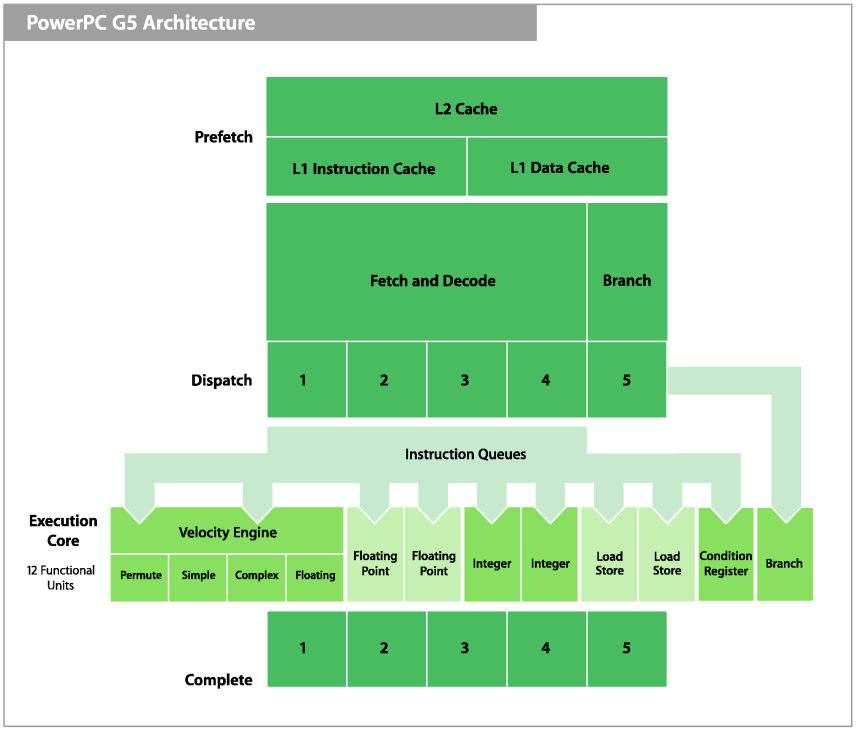


Figure 9.2: G5 Architecture (from [2])

9.2.1 Implementations

The G5 is used on the desktop PC Power Mac G5, the server Xserve G5 and the laptop iMac G5 [1].

9.3 Memory

Because memory addresses are computed in 64-bit registers capable of expressing integers up to 16 exabytes, G5 can theoretically address 2^{64} bytes of virtual memory[1] [2]. In practice, the physical memory is 42 bits or 2^{42} bytes, which is 4 terabytes.

In fetch and decode, the second level cache is 512 kB at a rate up to 64 GB per second. The L1 instruction cache is at 64 kB, direct-mapped from the L2 cache [2].

For virtual memory handling, the storage description register (SDR1) specifies the page table base address used in virtual-to-physical address translation [3].

[1] 16 exabytes $= 2^{64}$

9.4 Execution

The execution core consists of 12 functional units and are explained in Table 9.1.

Table 9.1: Execution core

128-bit velocity engine	A vector processing unit that support parallel internal operations. It simultaneously process 128 bits of data in four 32-bit integers, eight 16-bit, sixteen 8-bit or four 32-bit single-precision floating-point values [2].
Floating-point units	G5 completes at least two 64-bit calculations per clock cycle. Each of its two floating-point units can perform both an add and a multiply with a single instruction, as well as square root calculations [2].
Integer units	Two integer units that perform both 32- and 64-bit integer instructions.
Load/store units	Two load/store units that perform memory access operations on L1, L2 cache or main memory if appropriate [2].
Condition register unit	The condition register can hold up to eight condition codes and its purpose is to improve the flow of data. Branch instructions, for example, can consult the condition register for results in earlier operations [2].
Branch prediction unit	The unit for branch prediction and speculative execution to keep processing constantly in use. The branch prediction unit uses three component logic to reduce pipeline bubbles and maximize processor efficiency. The result of each prediction is captured in three 16 kB history tables, local, global and selector. Local branch prediction takes place when individual instructions are fetched into the processor. Global branch prediction occurs at current time, referring to the operation before and after. The selector branch identifies which of the global or local branches is most accurate [2].

9.5 References

[1] APPLE COMPUTER, INC. G5 processor. Web page, Sept. 21, 2005. http://www.apple.com/g5processor/, date visited given.

[2] APPLE COMPUTER, INC. *PowerPC G5*. White paper, Apr. 2005. Also available from http://images.apple.com/powermac/pdf/PowerPC_G5_WP_04272005.pdf.

[3] IBM. IBM PowerPC 970FX RISC Microprocessor User's Manual. Web page, July 19, 2005. http://www-306.ibm.com/chips/-techlib/techlib.nsf/techdocs/AE818B5D1DBB02EC87256DDE00007821/$file/-970FX_user_manual.v1.5.205JUL19.pdf.

[4] WIKIPEDIA. PowerPC 970. Web page, Sept. 3, 2005. http://en.wikipedia.org/wiki/PowerPC_G5.

10. Microchip PIC18F2550

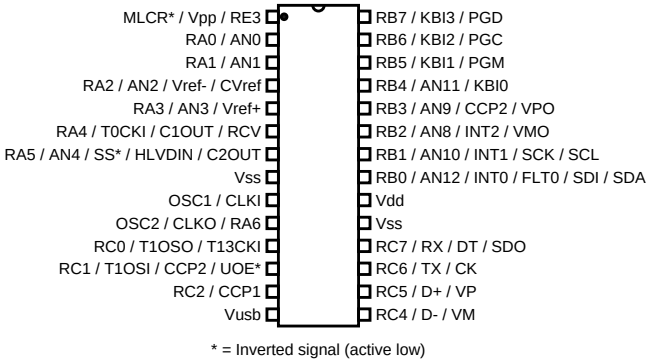


Figure 10.1: Pin diagram of PIC18F2550 (based on [4])

10.1 Introduction

The base for the first design of the PIC originated from Harvard University. A Harvard graduate, Howard H. Aiken, was the first to design a computer with separated data and program memory [1]. This design was adapted by General Instruments for usage in their peripheral interface controller (PIC).

The first PIC-product was named PIC 1650, and was released in 1977 [2]. The microprocessor was equipped with internal programmable ROM capable of holding up to 512 instructions, 32 registers, 32 I/O ports, and could operate at frequencies up to 1 MHz resulting in 250 kI/s[1].

In 1985 General Instruments spun off their micro electronics division into Arizona Microchip Technology, and the new company abandoned almost everything which by this time was out-of-date. The PIC however was upgraded with EPROM and was given much more room for program memory.

In time, the PIC products evolved and got features like WatchDog Timer (WDT), Brown-out Reset (BOR), built-in data transfer protocols like serial and parallel ports, and internal oscillators. Nowadays equipped with nanoWatt technology, USB-ports, RF transceivers and self-programmable program memory, the PIC are one of the most generally usable micro-controllers [5].

PIC18F2550 was released in 2004–2005 and is one of the most advanced, as in most well-equipped, PICs so far. Some of the features include: USB 1.1/2.0 slave port with internal pull-up resistors, internal oscillator, nanoWatt technology, 16 I/O ports supporting currents up to 25 mA, 4 internal timers and 10 A/D inputs. The only thing needed to run the chip is power, between 2.0 and 5.5 volts [5].

[1] Except for instructions changing the PC or if a conditional test evaluates to **true**. These instructions take twice the time of a normal instruction.

Nowadays the PIC circuits also have an internal Phase-Locked Loop (PLL) frequency multiplier that multiplies the oscillator frequency by four, and since the CPU internally need four clock cycles to complete an instruction, the PLL increases the frequency of the command execution to the same frequency as the oscillator [3].

Internally the CPU needs 8 clock cycles to execute a command. Four to load it, and four to perform the actual execution. Due to pipelining, these steps are done simultaneously for two instructions. When instruction A is performed instruction B is read.

10.2 Overview of the Processor

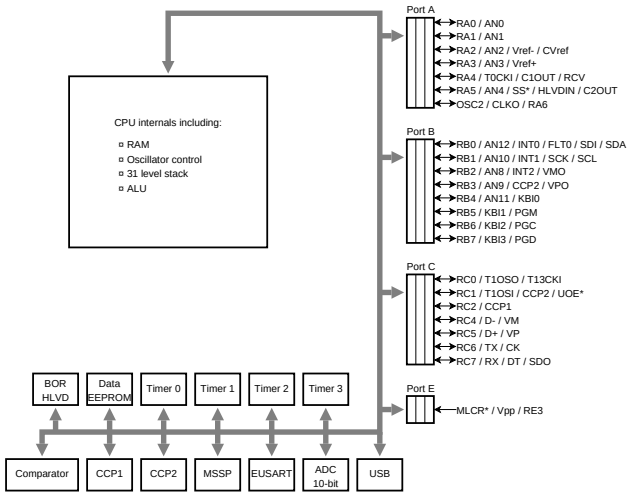


Figure 10.2: Overview of the PIC18F2550 (based on [4])

The main goal for the 18F2550 was to include all of the advantages in the whole PIC18-family, namely high computational performance along with large flash memory and high endurance at a low price. The result was the PIC18F2455/2550/4455/4550 family, a set of chips that is suitable for high-performance and power sensitive applications [4].

From PIC 1650 and forward the chips used 12 bits wide instructions and some chips even used 14 bits. This has recently been replaced by 16 bits to manage the increasing amount of instructions.

As seen in Figure 10.2 the circuit is well equipped, and in most applications many parts of the hardware is never used. This “overkill” approach makes the PIC usable in many different applications, a factor that is important when it comes to reducing costs, both for supplier and consumer.

The PIC18F2550 hasn’t been able to find its way into large productions yet, due to the recent release. The PIC family however can be found in a big variation of applications. The author has found PIC circuits inside one no-name-remote control for a TV

and a touch-pad for PCs. Considering the large number of I/O and analogue inputs, the PIC18F2550 family is likely to be found in memory card readers for PC, and measure and control applications in the future.

Instructions are all word-sized, and most commonly uses only one word (16 bits). The first part is the Opcode, which is followed by options, memory addresses and plain values. Opcodes differs in size to leave enough space for literals, but still being unambiguous [3].

When moving data between two memory positions, the memory is addressed with 12 bits. To be able to address both positions in the same instructions Microchip added *2-word instructions*. These instructions are all one normal 16-bit instruction followed by a 16-bit "instruction" with OPCODE set to `1111`, followed by a 12-bit literal. These instructions all take twice the time to run, since two words of instructions must be read.

10.3 Memory Hierarchy

Since the PIC is a one-chip computer, the implementation of memory and registers differs in relation to other processors. All RAM is implemented as registers, as well as all controlling bits, like status of a the I/O pins direction. These registers are divided into a maximum of 16 "banks", each carrying 256 bytes of memory [3]. The PIC18F2550 family have a memory of 2048 bytes, resulting in eight banks, where the first five is settings that affect the chip in one way or the other. The other three can be used as general purpose RAM [4].

One huge benefit with this design of using registers as RAM is that there's no delay when reaching the memory. This is not completely true, since it's a two step operation getting a value. First the bank must be selected, and then the value must be read. This is not true either. In PIC18 there's something called "indirect addressing" implemented. This works almost as pointers in C, making it possible to read several values from different banks without changing bank.

The use of indirect addressing also prepares for future implementations supporting external memory.

10.4 Execution

Executing an instruction is divided into two steps. First the CPU needs to read it from the program memory, then perform the actual execution. This two-step approach is divided into two separate parts of the core, and pipelining is used. This means that while doing the real execution of the first instruction, the next one is read and prepared [3].

Using pipelining enables the CPU to execute instructions much faster by preparing the next instruction. But predictive reading of the next instruction will be sabotaged by branches. Microchip has selected the easiest way to deal with pre-reading before branches and jumps. As soon as the Program Counter is changed, the next instruction is flushed from the pipeline [3]. This is approach is called *predict-not-taken*.

10.5 Programming

This whole section is a short version of section 30 in [3].

Programming the PIC is done serially to minimize the pin-count needed since all used pins must be disconnected or fairly independent on the rest of the circuit. For example, when the programmer is applying 5 V to the PIC for programming, it's seldom enough to drive a cooling fan that's connected to V_{SS} and V_{DD}. Therefore, isolating the PIC as much as possible during programming is needed.

Five pins are used for programming, V_{SS}, V_{DD}, $\overline{\text{MCLR}}$ (V_{PP}), DATA and CLOCK. Programming mode is entered by holding CLOCK and DATA low while rising $\overline{\text{MCLR}}$ (V_{PP}) to approximately 12 volts and having V_{DD} connected to +5 V. During the programming and reading, the CLOCK-line is controlled by the programmer, and the DATA-line is bidirectional. Now data (program code and EPROM memory) can be written to the PIC, and if the previous program wasn't read-protected, that program can be read.

10.6 Author's Personal Reflections

The PIC18F2550 is an interesting processor based on many factors. The I/O pins are multiplexed with different abilities. The port RE3 can act as both reset and input pin depending on the configuration of the processor when programmed. Anotherexample is the RB0 pin, which has seven different abilities, namely: digital input, digital output, analogue input, external interrupt, SPI data input, I^2C data input, I^2C data output.

Each I/O pin can handle up to 25 mA, enough for LEDs and small relays. They are also protected with diodes to avoid damage caused by out-of-range voltage.

It's the ability to minimize surrounding hardware that makes the PIC so interesting. It can run LEDs directly, use power saving modes, has internal oscillators, peak at 2 MIPS, and all while running at 2.0–5.5 volts.

10.7 References

[1] BAYKO, J. The great CPU list, section one. Web page, Mar. 30, 2003. `http://www.sasktelwebsite.net/jbayko/cpu1.html`.

[2] GENERAL INSTRUMENTS CORPORATION. *1977 Data Catalog*, 1977. Excerpts available at: `http://www.idcomm.com/personal/ottosen/`.

[3] MICROCHIP TECHNOLOGY INC. *PICmicro® 18C MCU Family Reference Manual*, 2000. Also available at `http://ww1.microchip.com/downloads/DeviceDoc/-39500a.pdf`.

[4] MICROCHIP TECHNOLOGY INC. *PIC18F2455/2550/4455/4550 Data Sheet*, 2004. `http://ww1.microchip.com/downloads/en/DeviceDoc/39632b.pdf`.

[5] MICROCHIP TECHNOLOGY INC. Microchip Technology Inc., Home Page. Web site, Sept. 20, 2005. `http://www.microchip.com`, date visited given.

11. MIPS R5000

11.1 Introduction

In 1996 the R5000 processor was released by MIPS[1] Computer Systems Inc. It was designed to be the successor of the R4600, which was released two years earlier. The design goal MIPS had with the R5000 was to provide several advanced features at a low cost. The way this was accomplished was that they borrowed a lot of features from the more advanced R10000 processor, while they still managed to retain a low chip complexity [4, 1].

The role of the R5000 was to provide good performance for mid- to low-range segments of workstations. Its main strength was its good price/performance ratio in regards to graphics processing [4].

11.2 Overview

The R5000 is a 64-bit superscalar RISC[2] processor, but it is also capable of running in 32-bit mode. It has 32 integer registers and 32 floating-point registers, all of which are 64 bits wide. The processor can execute both a single precision floating-point instruction and a dissimilar (i.e. integer, load/store etc.) instruction simultaneously.

The R5000 was the third Rx000 processor to implement the MIPS IV instruction set. MIPS IV includes instructions which allows for single-precision multiply-add operations to be performed with a single instruction. This is mainly used to speed up graphic-intensive calculations (i.e. CAD). The new instructions contribute a lot to the increased performance compared to previous generations of the R-series. The R5000 is fully backwards compatible with all Rx000 64-bit processors [3].

The R5000 was used in, among other systems, Silicon Graphics' Indy computer, which competed in the lower end of the CAD and multimedia market in the mid nineties [4].

11.3 Memory Hierarchy

To achieve higher performance, the R5000 has an internal memory cache, as well as an external one.

11.3.1 Internal Cache

The on-chip cache consists of the 32 kB data cache (D-cache) and the 32 kB instruction cache (I-cache), which is a doubling of the size of the caches compared to the R4600. The increased cache size increases performance by a significant amount, since applications then can avoid accessing the slower level 2 cache (or the much slower RAM).

[1] Microprocessor without interlocked pipeline stages.
[2] Reduced Instruction Set Computer.

Both the D-cache and the I-cache are two-way set associative, virtually indexed and physically tagged. The 32 bytes, or 8 words, fixed line-size applies to both caches. The D-cache loads 8 bytes of data each cycle, which results in a bandwidth of 1.6 GB/s at 200 MHz [1].

11.3.2 External Cache

The R5000 has an optional level 2 cache that is either 512 kB, 1 MB or 2 MB in size. The processor has a dedicated secondary cache interface which is used to send signals between the secondary cache, the processor and the secondary cache tag RAM. The external cache supports both write-back and write-through data transfer protocols. The data in the secondary cache is accessed through a 64 bits wide system bus [3].

11.4 Architecture

The processor consists of two different co-processors, the Central Processing Unit (CP0) and the Floating-Point Unit (CP1). The diagram on the right gives an overview of the architecture [2].

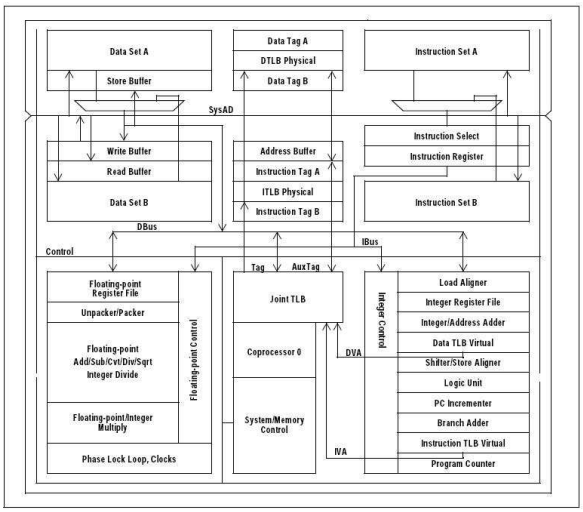


Figure 11.1: The R5000 architecture (from [2])

11.4.1 Central Processing Unit

The Central Processing Unit (CPU) performs all integer and system instructions. The CPU handles exception management, and also performs such tasks as translating virtual addresses into physical ones. CP0 controls the cache subsystem, power management and other vital responsibilities. The CPU has 32 general purpose registers (R0-R31). Two of the registers have assigned functions: R0 is hardwired to the value 0 and R31 is used as in implicit return address register. All registers are 64 bits wide [2].

11.4.2 Floating-Point Unit

The Floating-Point Unit (FPU) extends the CPU instruction set to include floating-point calculations. The FPU has 32 general purpose registers (FGR0-FGR31) that can be configured to be either 32 bits or 64 bits wide. The FPU conform to the IEEE754 Standard for Binary Floating-Point Arithmetic [2].

11.4.3 Cost Reductions

The R5000 was created with a low production cost in mind. Several aspects of the processor reflect this:

- No support for speculative execution.
- No support for dynamic branch prediction.
- No 128-bit secondary cache bus, unlike the R10000 and the R4000. The general 64-bit I/O-bus is used instead.
- The die is manufactured in a 0.32-micron process.
- It has a reduced number of pipelines compared to several other R-models.

All of these decisions lower the complexity of the chip, which results in a smaller die. The smaller the die, the lower the production cost [1, 3].

11.5 Pipeline

Every instruction the R5000 processes is sent through the five-stage pipeline. Having a pipeline allows the chip to process several instructions at the same time. The five stages are as follows:

1. Instruction fetch. Two instructions are fetched. The CPU checks which unit should handle the instruction.
2. Register access. The CPU accesses the register. Possible branch addresses are calculated.
3. Execute. The FPU and the integer unit executes the instructions.
4. Access data cache. The primary data cache is accessed by the instructions. Virtual address to physical address translation is performed.
5. Write-back. The processed information is written to the appropriate destination.

Note: The instructions are partially decoded when the instruction cache is refilled [1].

11.6 Instructions

There are three instruction formats: immediate (I-Type), jump (J-Type) and register (R-Type). The small number of instruction types simplifies instruction decoding immensely. All instructions are 32 bits long.

11.6.1 CPU Instruction Types

The CPU handles seven different types of instructions [2]:

- Load and Store: Moves data between general registers and the memory (I-Type).
- Computational: Performs arithmetic, logical, shift, multiply and divide operations on values in the registers (R-Type and I-Type).
- Jump and Branch: Performs flow control. Jumps are always made to an absolute address, while branches are made relative to the current program counter (I-Type, J-Type and R-Type).
- Co-processor: Performs operations in co-processors (I-Type).
- Co-processor 0: Performs operations such as memory and power management.
- Special: Performs system calls and breakpoint operations (R-Type).
- Exception: Causes a branch to the general exception handling vector, based on the result of a comparison (R-Type and I-Type).

11.7 Performance

The first release of the R5000 had a clock frequency of 150 MHz, while later models ran at 250 MHz [1, 3].

Table 11.1: A performance comparison

Parameter	R5000-200	Pentium 133	Pentium Pro 200
SpecInt95	5.5	4.1	8.1
SpecFP95	5.5	2.5	6.0

11.8 References

[1] HALFHILL, T. R. MIPS R5000: Fast, affordable 3-D. *Byte* (May 12, 1996). Available at `http://www.byte.com/art/9605/sec14/art1.htm`.

[2] INTEGRATED DEVICE TECHNOLOGY, INC. *IDT79RC5000 RISC Microprocessor: Reference Manual*, Sept. 26, 2000. Also available at: `http://www1.idt.com/pcms/tempDocs/RC5000_MA_33864.pdf`.

[3] MIPS TECHNOLOGIES, INC. R5000 product information. Web page, Sept. 20, 2005. `http://www.mips.com/content/PressRoom/TechLibrary/RSeriesDocs/`, date visited given.

[4] WIKIPEDIA. MIPS architecture. Web page, Sept. 8, 2005. `http://en.wikipedia.org/wiki/MIPS_architecture`.

12. MIPS R10000

12.1 Introduction

MIPS R10000 also called the T5, was released in the end of 1995. The creators, MIPS Technologies Inc., was a separate company earlier, but was on the verge of bankruptcy when Silicon Graphics (their largest client) bought the company.

The T5 processor is based on the same fifth generation RISC superscalar technology that was introduced earlier in 1994 with the MIPS R8000. R10000 is a general-purpose single-chip processor for desktop PCs, workstations and servers. Because of the better balance (than the R8000) between integer and floating-point performance the T5 offers a processor more suitable for mainstream applications. As the first single-chip superscalar processor from MIPS, the T5 represented a significant step forward for the Rx000 architecture. It also had a better price/performance ratio advantage compared to its opponents, the Alpha and the Pentium [2, 1].

12.2 Overview

The T5 builds on the preceding generations (R2000, R3000, R5000 and R4000) by incorporating five functional units, twice as much primary cache as the R4400, twice as many register and implementing many advanced mechanisms looked into more deeply further down in this document. The MIPS R10000 supports easy clustering of up to four processors with a single chip. Because of that, this processor is often used in large graphical computations.

12.3 Architecture

The R10000 has the following major features [4]:

1. It implements the 64-bit MIPS IV ISA and is backwards compatible with the I, II and III
2. It can decode four instructions in every pipeline cycle, appending them to one of three instruction queues (the integer, floating-point and address queues)
3. It has five execution pipelines connected to separate internal integer and floating-point execution (or functional) units.
4. It uses dynamic instruction scheduling and out-of-order execution (described in Section 12.3.3 on page 51)
5. It uses branch prediction and speculative execution (described in Section 12.3.4 on page 52)
6. It uses MIPS ANDRES[1]
7. It uses a precise exception model (exceptions can be tracked back to the instruction that caused them)

[1] Architecture with Non-sequential Dynamic Execution Scheduling.

8. It uses non-blocking cache, the processor can in advance execute load and store instructions if the data block isn't in the primary cache already
9. It has separate on-chip 32 kilobytes primary instruction and data caches
10. It has individually optimized secondary cache and System interface ports
11. It has an internal controller for the external secondary cache
12. It has an internal System interface controller with multiprocessor support

The T5 gain much of its power through the combination of dynamic scheduling, the non-blocking cache and the out-of-order execution. Thanks to these functions the processor can operate on the highest possible effect by executing instruction on available executions units without bother about the order of them [3].

Furthermore, the T5 can predict the outcome of branches and speculatively execute the code that follows. Only when all dependencies are resolved, the results of the completed instructions are retired and restored to their original sequences. Up to four results can be graduated every cycle. A four quadwords branch-buffer is used for the retirement of a wrong predicted branch, more about that in Section 12.3.4 on page 52.

Loads and stores in the T5 is non-blocking; i.e. cache misses won't stall the processor, all other parts not involved continues to work on independent instructions while as many as up to four cache misses are being evaluated. Hence, the R10000 is a particularly good CPU for busy database servers [1].

12.3.1 Cache and Execution Units

R10000 can fetch four 32-bits instructions from its 32 kB two-way set-associative instruction cache (also called the I-cache), i.e. there are two cache blocks for each set. The I-cache implements the LRU (Least Recently Used) replacement algorithm making it almost four times more efficient than the R4000. T5 also got an external secondary cache that can range from 512 kB up to 16 MB in size, actually, the primary cache is a subset of the secondary. Both caches use the same algorithm [1, 4]. Previous versions of the Rx000 series chip had scalar pipelines that always executed instructions as they where written in the program, in contrast the T5 has five totally independent pipeline execution units.

Cached instructions are 37 bits long because five extra bits are appended during the pre-decode-stage when instruction are being prefetched to the cache. The extra bits assist the decode-unit by classifying the instructions after attributes and assign them to appropriate execution unit. This step is not counted as a pipeline stage, so it is not considered in Table 12.1 on page 52.

After an instruction has been fetched from the I-cache it passes through a two-staged decoder, the actually decoding is taking one step and the other is for the "register renaming" mechanism explained in Section 12.3.3 on the facing page.

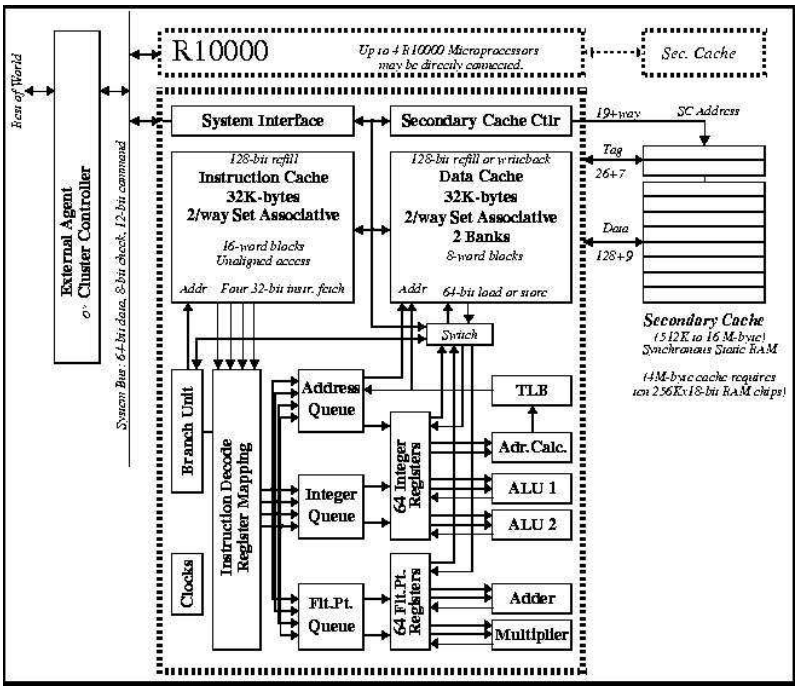


Figure 12.1: Block diagram of the R10000 processor (from [4])

12.3.2 Functional Units

The functional units are two integer arithmetic logic units (ALUs), one load/store and two floating-point units (FPUs) – one for addition and one for multiply/square/divide calculations. The two ALUs are quite alike, but only one can handle multiply while the other contains logic for verifying predicted branch, depending on integer register values. The load/store unit handles the address calculations and translation, it translates 44-bit virtual addresses to 40-bit physical addresses using the Translation Look-aside Buffer (TLB) as seen in Figure 12.1 [4].

12.3.3 Dynamic-Register Renaming

Dynamic-register renaming is a new invention for the Rx000 architecture, earlier versions of the processors had 32 integer and floating-points registers, T5 has 64 registers where each one of them is 64 bits wide. In each register file any of the 64 physical registers can be represented. Seen from the outside (by the programmer) the T5 still has 32 registers, even though it is twice as many. This feature is important for speculative and out-of-order execution and even branch prediction, allowing them to store intermediate results and data in the “invisible” registers [1].

12.3.4 Branch Prediction

"Branch prediction" works as follows: The T5 speculates on every branch trying to pick the right one of two or more choices (like in an if–then–else), up to four branches deep. At each of these, the T5 takes a "shadow map" as the register-rename map existed at that moment. If it was a wrong prediction, the T5 does not have to clear any buffers or flush any data, it can simply restore the appropriate shadow map as the working register-rename map. This can take from one to four cycles depending on when the processor realized it took the wrong branch [1, 4].

12.4 Performance and Data

MIPS R10000 has an up to 275 MHz internal clock, the external clock is programmable at 275, 250, 200, 133, 100, 67, 57 or 50 MHz (depending on version).

Table 12.1: Performance data of a 275 MHz MIPS R1000 (from [2])

Bits Instruction/Data	64
Clock frequency MHz	275
SPEC-95 Int/Fp	12/24
Units/issue	5/5
Pipe Stages int/ldst/fp	5/6/7
Cache Instruction/Data kilobyte Set Associative	32/32 2/2
Voltage	3.3
Effect W Peak	30
Transistors x1000000	6.8
Size mm^2	298

12.5 References

[1] HALFHILL, T. R. T5: Brute force. Web Page, Nov. 20, 1994. http://www.byte.com/art/9411/sec8/art6.htm.

[2] HALLGREN, A. MIPS R10000 processor. Web Page, May 24, 1997. http://hem.passagen.se/merlin3/r10000.htm.

[3] MIPS TECHNOLOGIES, INC. MIPS technologies announces. Web Page, Oct. 1994. http://sgistuff.g-lenerz.de/documents/r10000-announce.txt.

[4] MIPS TECHNOLOGIES, INC. *MIPS R10000 Microprocessor User's Manual*, Jan. 1997. Available for download from http://techpubs.sgi.com/library/tpl/-cgi-bin/download.cgi?coll=hdwr&db=bks&docnumber=007-2490-001.

13. MOS Technology 6502

13.1 Introduction

The MOS technology 6502 is an 8-bit microprocessor that started a revolution in computer projects in the late 1970s, which resulted in the low-cost revolution of personal home computers in the 1980s [7].

13.1.1 History

In 1975 a large group of designers involved in the creation of Motorola 6800 left the Motorola company to pursue their own ideas on CPU design. Since, at the time, there was no such thing as a "design only" firm, they joined MOS Technology and started working on a new CPU that would exceed the 6800 [16]. The resulting 6501 design was better than the 6800, but since it was pin compatible with the 6800 it was taken off the market almost immediately for legal reasons. As a result, the 6502 was released. In reality the 6502 was a replica of the 6501 except from the fact that the pins were rearranged so that it wasn't compatible with Motorola motherboards. The 6502 really hit the scene in 1976 when it was apparent that it was faster than most rivaling designs and sold for 1/6 the price, or less, than its competitors from larger companies such as Motorola and Intel [15]. This success derived from two factors, where the first was improvements in CPU production, which increased the success rate by 70 % or better and thus lowering the prize on the final products. The second factor was it's amazing performance. This was achieved with the help of a technique called "factory pipelining" (not to be confused with real pipelining which arrived in the early 1980s [14], which in short means that the CPU doesn't waste any memory cycles since new information was rolling in while the old ones was being digested [3].

13.2 Overview of the Processor

The CMOS 6502 is a so called *true 8-bit processor* since no data registers exceeds 8 bits, unlike the Z80 which was an 8-bit processor with 16-bit data registers [12]. The CMOS 6502 is an 8-bit, factory pipelined processor, with a 16-bit address bus. Inside was one 8-bit accumulator register, two 8-bit index registers, an 8-bit stack pointer, an 8-bit status register and a 16-bit program counter.

There are a total of three general purpose and two special purpose registers [9] whereas the accumulator is the heart of the system, since it handles all of the arithmetic and logic calculations. The X and Y registers are mainly used as an index in different types of index addressing modes, thus falls under general purpose registers. The stack pointer and the processor status register are more special purpose registers and software access to the stack was done via four implied addressing mode instructions whose function were to push or pop the accumulator or the processor status register [16].

All this combined defines it to what is commonly known as an index-machine [1]. When the 6502 was constructed RAM was faster than CPUs, so the idea to have fewer

registers and optimized RAM-access rather than increase the number of registers on the chip made it both faster and cheaper than its competitors [16].

The small number of registers was at first taken as a limitation but proved swiftly not to be, since the registers could be loaded and used so quickly that it rarely proved an inconvenience. One of the things that made the 6502 such a powerful competitor on the market was its zero page addressing mode. The zero page mode could be used to hold 16-bit pointers for *indirect addressing* [3] and let you specify a memory location addressable with only one byte instead of the normal two for all other memory locations. In a way this means the 6502 can be seen as having 256 registers [2].

There are 56 different instructions. They are implemented by a hardwired logic array that is only defined for valid opcodes. This is a bit of a dubious feature and it will be discussed further in the section on anomalies. Most of the 56 different instructions can use several addressing modes whereas some only use one [16]. The 6502 was one of the first processors to spread the little-endian-dogma (due to its popularity), which means that the 6502 processor expects addresses to be stored in *"little endian"* order, with the least significant byte first and the most significant byte second. The designers of 6502 felt it was more natural this way because when you do arithmetic manually, you start at the least significant part and work your way towards the most significant part. At the time of MOS 6502 this meant it was easier to implement an 8-bit processor using little-endian than big-endian [5]. Worth mentioning in regards to this is that our Intel x86 P4 of today use little-endian architecture whereas Sun SPARC, Motorola 68000, PowerPC 970 and IBM System/360 use big-endian architectures [13].

13.2.1 Addressing Modes

The 6502 processor had quite a few addressing modes for its time, with a total of 13 different addressing modes [4]. At most 64 kB of memory could be addressed [10].

Immediate Addressing The second byte of the instruction is used immediately in the operation. In other words, the instruction don't need to access any memory addresses except the next byte in the code.

Absolute Addressing The second byte of the instruction specifies the eight low order bits of the effective address while the third byte specifies the eight high order bits. Thus, the absolute addressing mode allows access to the entire 65536 bytes of addressable memory.

Zero Page Addressing The zero page instructions allow for shorter code and execution times by only fetching the second byte of the instruction and assuming a zero high address byte.

Implied Addressing The address containing the operand is implicitly stated in the operation code of the instruction.

Indirect Absolute Addressing Only used by the JMP-instruction. The second byte of the instruction contains the low order eight bits of a memory location. The high order bits of that memory location is contained in the third byte of the instruction.

The contents of the fully specified memory location is the low order byte of the effective address. The next memory location contains the high order byte of the effective address which is loaded into the sixteen bits of the program counter.

Absolute Indexed (X, Y) Addressing This form of addressing is used in conjunction with X and Y index register and is referred to as *Absolute X* and *Absolute Y*. The effective address is formed by adding the contents of X and Y to the address contained in the second and third bytes of the instruction. This type of indexing allows any location referencing and the index to modify multiple fields resulting in reduced coding and execution time.

Zero Page Indexed (X, Y) Addressing This form of addressing is used in conjunction with the index register and is referred to as *Zero Page X* and *Zero Page Y*. The effective address is calculated by adding the second byte to the content of the index register. Since this is a form of *Zero Page* addressing, the content of the second byte refers to a location in page zero.

Indexed Indirect Addressing In indexed indirect addressing, the second byte of the instruction is added to the content of the X index register. The result of this addition points to a memory location on page zero whose contents is the low order eight bits of the effective address. The next memory location in page zero contains the high order eight bits of the effective address. Both memory locations specifying the high and low order bytes of the effective address must be in page zero.

Indirect Indexed Addressing In indirect indexed addressing, the second byte of the instruction points to a memory location in page zero. The contents of this memory location is added to the content of the Y index register, the result being the low order eight bits of the effective address. The carry from this addition is added to the contents of the next page zero memory location. The result being the high order eight bits of the effective address.

Relative Addressing The relative addressing is used only with branch instructions and establishes a destination for the conditional branch. The second byte of the instruction becomes the operand which is an offset added to the contents of the lower eight bits of the program counter when the counter is set at the next instruction. The range of the offset is -128 to +127 bytes from the next instruction.

Accumulator Addressing This form of addressing is represented with a one byte instruction, implying an operation on the accumulator.

Table 13.1: Benchmark

Company, model, year	CPU	speed	MWIPS
Acorn, BBC B, 1982	MOS/Rockwell 6502	2,2 MHz	0.0031
Apple, Apple II, 1976	65C02	1 MHz	0.0011
Sinclair, Spectrum, 1982	Zilog Z80	3,54 MHz	0.00052

13.3 Performance

The 6502 did not have a lot of apparent speed and power under the hood, but a closer look showed you that its rather humble clock-speed of 1–4 MHz actually was comparable to machines having clock-speeds up to four times as high [11]. Mostly this can be attributed to the 6502's factory pipelined static processor, that uses the index and stack registers effectively with several addressing modes.

Furthermore the 6502 had a small delay in which it was guaranteed not to access the bus. This gave video display hardware time to read out a line of the screen without the 6502 having to pause, which in general terms meant an increase of 25 % in performance for the machine with a 6502.

When studying the 6502's performance [6] (shown in Table 13.1) it is important to remember that the processor is available in a wide range of versions with different clock speeds, registers and pins and therefore gives different benchmark results depending on its composition.

13.4 Anomalies

As mentioned earlier in Section 2 (Overview of the processor) the 6502s instruction decoding was only defined for valid opcodes. This resulted in roughly 32 undocumented opcodes which unfortunately triggered several valid instructions at once, leading to unexpected results [16].

The 6502's indirect jump instruction was broken. If the address was hexadecimal xxFF, the processor would not access the address stored in xxFF and xxFF+1, but rather xxFF and xx00. The 6510 did not fix this bug, nor was it fixed in any of the other NMOS versions of the 6502 such as the 8502 and the 2A03. Bill Mensch at the Western Design Center was the first to fix it, in the 65C02 CMOS derivative; he then went on to design the 65C816 processor, a 16-bit successor to the 65C02.

13.5 Implementations

One of the first "public" uses for the design was the Atari 2600 video-game console [16]. The 2600 used an offshoot of the 6502 called the 6507, which had fewer pins and could address only 8 kB of RAM as a result. Millions would be sold in this form.

MOS Technology's 6502 processor was chosen for the first Apple computers, not because it was powerful, but because it was cheap [8]. Introduced in 1975 at under $100 (compared with $375 for the similar Motorola 6800), the 6502 was a real bargain. Fifteen years later, the 6502 was still being used (in the Nintendo Entertainment System).

The efficient design of the 6502 also inspired the principal designers of the ARM RISC processor, and so the legacy of the 6502 may be said to transcend the original processor (family) since its spirit lives on in the ARM design [7]. It was used in several desktop computers as well as a lot of handheld and other embedded systems, and sold as an Intellectual Property (IP) block to be used in systems-on-chip products.

The 6502 design was originally second-sourced by Rockwell and Synertek and later licensed to a number of companies; it is still made for embedded systems

13.6 References

[1] ÅGREN, O. Dark: The virtual machines for the computer architecture course. Web page, Sept. 21, 2005. http://www.cs.umu.se/~ola/Dark/.

[2] BERGMANS, S. Sb-assembler. Web page, Sept. 21, 2005. http://www.xs4all.nl/~sbp/sbasm/6502.htm, date visited given.

[3] BUTTERFIELD, J. The new 6500 cpu's. *Compute!* (Sept. 1983). Available at http://www.commodore.ca/history/company/6502/6500cpus.htm.

[4] COMMODORE SEMICONDUCTOR GROUP. *6500 Microprocessors*, Oct. 1985. Also available at http://www.6502.org/archive/datasheets/mos_6500_mpu_nov_1985.pdf.

[5] GREEN, R. endian : Java glossary. Web page, Sept. 21, 2005. http://mindprod.com/jgloss/endian.html, date visited given.

[6] LONGBOTTOM, R. Whetstone benchmark history and results. Web page, Dec. 2004. http://homepage.virgin.net/roy.longbottom/whetstone.htm.

[7] NATIONMASTER. MOS technology 6502. Web page, Sept. 11, 2005. http://www.nationmaster.com/encyclopedia/MOS_Technology_6502.

[8] PATTERSON, J. The history of computers during my lifetime. Web page, July 12, 2003. http://www.pattosoft.com.au/jason/Articles/HistoryOfComputers/-1970s.html.

[9] RTK. Assembly in one step. Web page, Sept. 21, 2005. http://nesdev.parodius.com/6502guid.txt.

[10] VARIOUS AUTHORS. 6502. Web page, Sept. 21, 2005. http://www.everything2.com/index.pl?node=6502, date visited given.

[11] VARIOUS AUTHORS. The performance of computer architectures. Web page, Sept. 21, 2005. http://boinc-doc.net/site-common/lessons/-03-performance.php.

[12] WHEELER, D. A. Ways to implement computer languages on 6502. Web page, Sept. 21, 2005. http://www.dwheeler.com/6502/a-lang.txt.

[13] WIKIPEDIA. Endianness. Web page, Sept. 21, 2005. http://en.wikipedia.org/wiki/Endianness/, date visited given.

[14] WIKIPEDIA. MIPS architecture. Web page, Sept. 21, 2005. http://en.wikipedia.org/wiki/MIPS_architecture.

[15] WIKIPEDIA. MOS technology. Web page, Sept. 21, 2005. http://www.answers.com/topic/mos-technology-6502.

[16] WIKIPEDIA. MOS technology 6502. Web page, Sept. 21, 2005. http://http://en.wikipedia.org/wiki/MOS_Technology_6502.

14. MOS Technology 6510

14.1 Introduction

14.1.1 History

Released in 1982, the MOS Technology 6510 was contained in the best-selling computer of all time, the Commodore 64. What made the computer so popular was the low price and the MOS 6510 definitely contributed to making it cheap. Typically, the CPU ran on a clock speed of 1 MHz but it was designed to run at up to 3 MHz. The original MOS 6510 was based on NMOS-technology (Negative-channel Metal Oxide Semiconductor) and in 1985 the faster HMOS-version (High performance NMOS) of the CPU was developed. It was renamed to 8500 and became integrated into the new and improved version of the Commodore 64, the C64c. As noted below, the 6510 had very few registers and this reflects the fact that RAM actually was faster than CPUs at the time of construction. Thus, to increase performance optimizing RAM access rather than adding more registers made sense [8].

14.1.2 Predecessors

The MOS 6510 had the same internal architecture as the very successful predecessor MOS 6502 (1975) which was included in the Apple II, VIC-20 and several Atari models. The 6510 basically added an external clock and six I/O pins to the MOS 6502 architecture [8].

14.2 Overview of the Processor

14.2.1 Registers

The MOS 6510 has six registers; the 8-bit accumulator, the stack-pointer, the index-registers X and Y, the processor status register and the 16-bit program counter (see Figure 14.1 on the next page) [6].

Accumulator

The accumulator is in contrast to the X and Y registers directly connected to the ALU. All arithmetic and logical operations are performed using this 8 bit register. The accumulator has direct connection with the arithmetic and logic unit which is not the case with the X and Y registers.

Stack Pointer

The stack register is a 8 bit offset to the 256 byte stack page. To push or pop data to/from the stack the 6510 make use of TSX and TXS instructions which transfers the value on the stack to/from the X register.

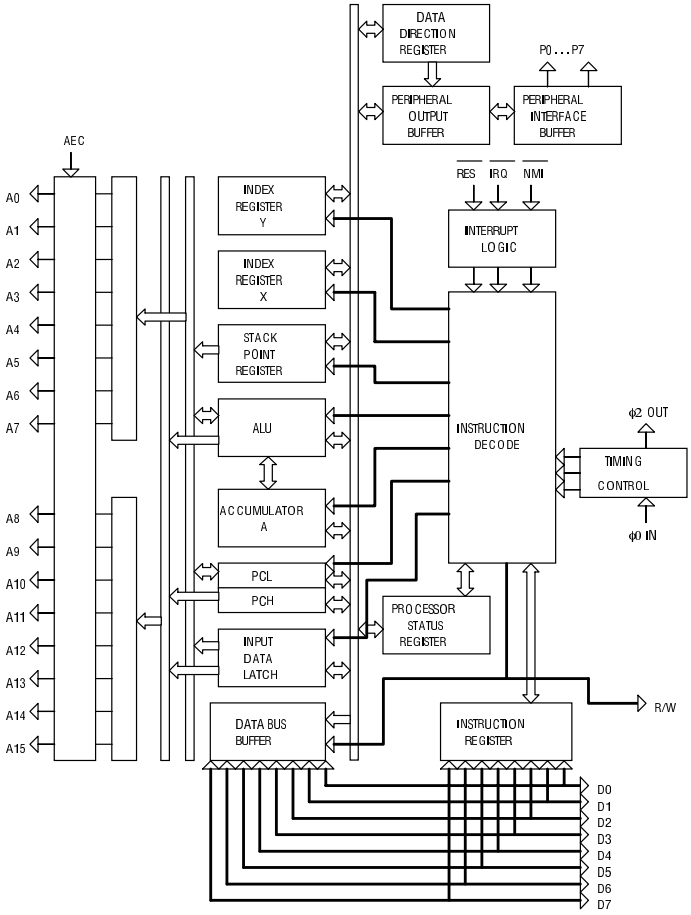


Figure 14.1: MOS 6510 block diagram (from [7])

Index Registers X and Y

The X and Y registers allow indexing in the memory. It is possible to index in the zero page (the first 256 bytes in the data memory) using one of the registers or combine the two to allow indexing in the whole data memory.

Program Counter

The PC register is the only 16 bit register in the 6510. It is divided into two halves, PCL (low-) and PCH (high-) bits. The PC points to the next instruction to be executed.

Processor Status Register

The 8-bit processor status register holds 8 flags which are signaling the state of the processor. These are the flags:

1. Carry flag - is used in additions and subtractions to calculate numbers bigger than 8 bits.
2. Zero flag - is set when the result of an arithmetic operation is zero.
3. Interrupt disable flag - is set when a hardware interrupt is triggered via the IRQ signal.
4. Decimal mode flag - is used for setting decimal mode for addition and subtraction.
5. Break flag - is always set except when a hardware interrupt is being processed.
6. Unused flag.
7. Overflow flag - is set if the result of a arithmetic operation does not fit in a 8-bit signed integer.
8. Negative flag - is set if the result of a arithmetic operation was less than zero.

14.2.2 Busses

The processor has an 8-bit bi-directional data bus and the direction of data is controlled via a read/write signal. The address bus, on the other hand, has 16 bits (see Figure 14.1 on the facing page) [3].

14.2.3 Instructions

There are 56 instructions and all operations has either zero or one operand corresponding to instruction lengths of either two or three bytes. Instruction lengths are very much depending on the addressing mode used and also affects the amount of clock cycles required to decode an instruction. The aim in the 6510 is therefore to minimize instruction length which makes the zero page an important feature [5].

14.2.4 Addressing Modes

As mentioned before, the MOS 6510 makes heavy use of the RAM instead of having several registers to operate on. To facilitate the heavy memory addressing, it has 13 addressing modes. An important addressing mode is zero page addressing where addressing is performed in the 256-bytes zero page mentioned above. Zero page addressing enables shorter instructions because the zero page address can be stored in just 1 byte in contrast to higher addresses which must occupy 2 bytes thus takes more time to execute. The zero page can also provide 16-bit pointers for indirect addressing which makes it, in a sense, quite similar to an extra 128 registers [2].

14.3 Memory Hierarchy

14.3.1 Data Memory

The Commodore 64 has an addressable memory space of 64 kilobytes which is largely occupied by the external RAM (see Figure 14.2) [3].

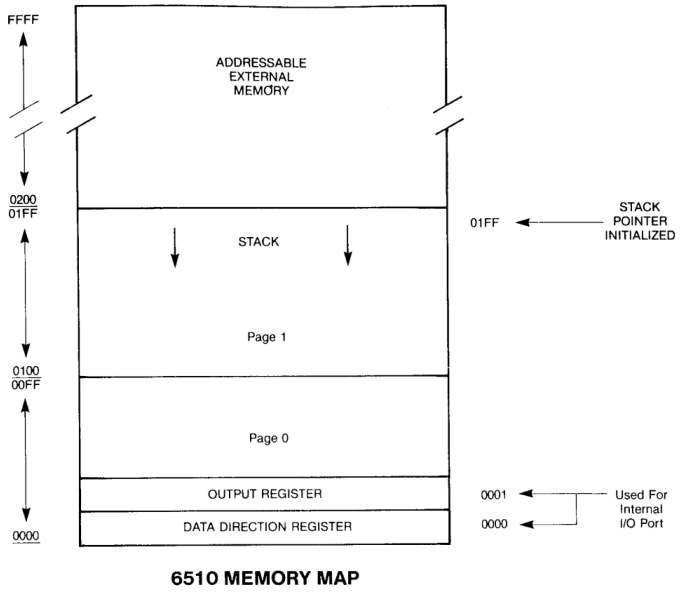


Figure 14.2: MOS 6510 memory map (from [3])

14.4 Execution

14.4.1 Steps

The 6510 always perform at least two reads for each instruction, the opcode and the next byte is always fetched. This is made even if the instruction consists of just one byte which may seem a bit odd. The third byte is fetched only if the instruction consists of three bytes. The next step is depending on the addressing mode used in the instruction, for example if indexed addressing is being used the value of the index register (X or Y) is added to the operand while in an immediate instruction the value to perform a calculation on is already fetched. The remaining steps execute the instructions once the operand has been fetched. Branch instructions has specific steps [6].

14.4.2 Pipelining

The processor uses a primitive type of pipelining. If an instruction doesn't store data to memory on the last cycle the next instruction is started, which means that the opcode is fetched. This makes the processor a little bit faster and still, the problems with pipelining are avoided. The 6510 lacks support of branch prediction [6].

14.4.3 Performance

The MOS 6510 performs a 0.0205 MIPS in dhystone MIPS v1.1 [1]. The Apple II which has the same cpu but runs at a slightly higher clock speed achieves 0.0210 MIPS in the same test. A CPU of today has somewhere around 5000 MIPS (Pentium 4 2.8 GHz) [4].

14.5 References

[1] ABURTO, A. dhrystone. Web page, Dec. 2, 1996. http://performance.netlib.org/performance/html/dhrystone.data.col0.html.

[2] BUTTERFIELD, J. The new 6500 cpu's. *Compute!* (Sept. 1983). Available at http://www.commodore.ca/history/company/6502/6500cpus.htm.

[3] COMMODORE SEMICONDUCTOR GROUP. *6510 Microprocessor with I/O, data sheet*. Norristown, Pennsylvania, Nov. 1982.

[4] JAY, J. Pentium 4 2.8GHz and Intel D845EBT motherboard – review: Synthetic benchmarks. *ActiveWin* (2004). Available at http://www.activewin.com/reviews/hardware/processors/intel/p428ghz/benchs.shtml.

[5] MANSFIELD, R. *Machine Language For Beginners*. Small System Services, Inc, 1983.

[6] MÄKELÄ, M. Documentation for 6502/6510/8500/8502 instruction set. Web page, June 3, 1994. http://www.atarihq.com/danb/files/64doc.txt.

[7] SUNDELL, H. Korrekt emulering av mikrodator C64 i systemoberoende mjukvara. Master's thesis, Göteborgs University, Sweden, 1996. Available at http://www.cs.chalmers.se/~phs/md1phs.pdf.

[8] WIKIPEDIA. MOS Technology 6502. Web page, Sept. 11, 2005. http://en.wikipedia.org/wiki/6502.

15. Motorola 68HC11

15.1 Introduction

In the mid 70s, Motorola released several 8-bit microprocessors. The development of the 68HC11 started with 6800, after that followed 6801, 6805 and 6809. It is designed as a microcontroller unit (MCU) which means that the CPU, memory and I/O are fitted on one single chip. The letters HC in the name means that the processor is built with HCMOS technique (High-Density Complementary Metal Oxide Semiconductor). This makes the processor faster and uses less power compared to CMOS.

15.2 Overview

Since the 68HC11 is a single-chip computer, it has all the things a computer system needs on one single chip. Apart from the CPU, memory and I/O, the HC11 also has built-in random-access memory (RAM), read-only memory (ROM), electrically erasable programmable ROM (EEPROM), A/D converter, pulse accumulator and oscillator. Also a serial peripheral interface (SPI) for synchronous communication and a serial communications interface (SCI) for asynchronous communications is built in, see Figure 15.1.

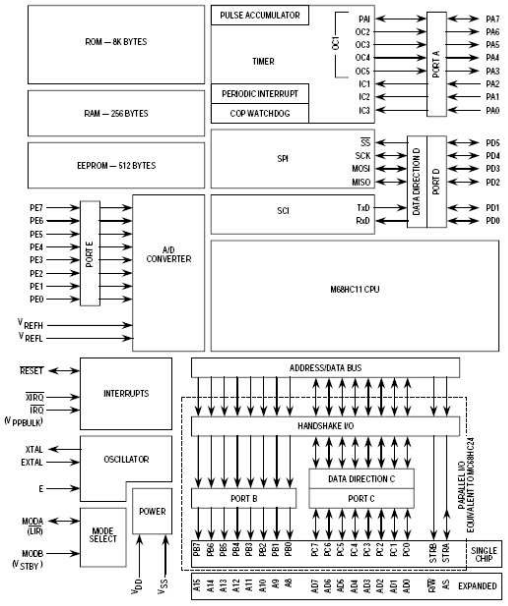


Figure 15.1: Block Diagram (from [2])

15.3 Central Processing Unit

The CPU is responsible for executing all software instructions in their programmed sequence. The 68HC11 CPU can execute all the instructions of the earlier M6800 and M6801 instructions, and more than 90 new instruction opcodes. The biggest change from the M6800 and M6801 is the addition of a second 16-bit index register (Y) [2].

15.3.1 Registers

The 68HC11 registers consists of the accumulators (A, B), that also can be used as a double accumulator D, index registers X and Y, program counter (PC), stack pointer (SP) and the condition code register (CCR), see Figure 15.2.

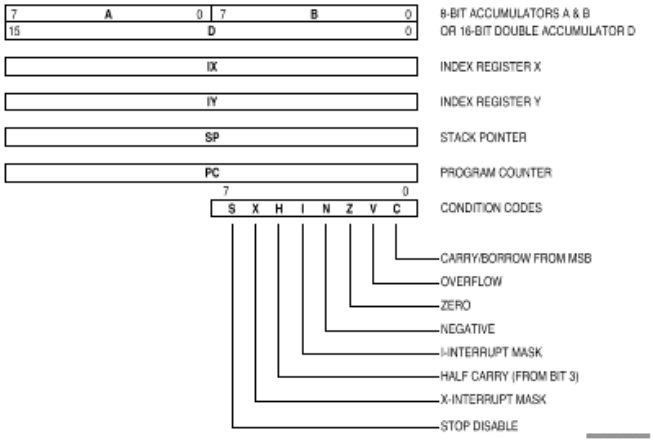


Figure 15.2: MC68HC11 Programmers model (from [2])

Accumulators

The accumulators A and B are general-purpose 8-bit registers that are used to hold operands and results of arithmetic calculations. It is possible to use both accumulators as a single 16-bit double accumulator (D).

Index Registers

Index registers X and Y are 16-bit registers that are used for indexed addressing mode, where the registers are used as pointers when accessing the memory.

Program Counter

The program counter (PC) is a 16-bit register that contains the address of the next instruction to execute.

Stack Pointer

The Stack pointer is a 16-bit register that holds the address of the first available space on the stack.

Condition Code Register

The Condition Code Register (CCR) is an 8-bit status register where the bits can be thought of as flags. The flags indicates status from the last operation.

The Z bit indicates if the last operation resulted in zero. If the result of an arithmetic or logic operation is zero the flag will be true, and if the operation results in other than zero the flag will be false.

The N bit reflects the state of the most significant bit (MSB) of a result. If an operation results in MSB=1, the flag will be set, otherwise it will not.

The C bit usually indicates carry from addition, or borrow from subtraction, but it is also used as error indicator of multiply and divide operations.

The V bit indicates if a two-complements overflow has occurred from an operation.

The STOP (S) bit is used to enable or disable the STOP instruction. When the CPU encounter the STOP instruction and the S bit is set, it will treat it as a NOP (no operation) operation and continue to the next instruction.

15.3.2 Addressing Modes

In the 68HC11, six modes can be used to reference memory: immediate, direct, extended, indexed, inherent and relative.

Immediate (IMM)

In the immediate addressing mode, the actual argument is immediately following the instruction. For example `LDAA #18` loads accumulator A with 18. The '#' is used to separate immediate addressing from other addressing modes.

Extended (EXT)

In extended addressing mode ("Direct" in standard terminology [1]), which has an instruction length of 3 bytes, the effective address appears explicitly in the two bytes following the opcode. For example `LDAA $1098` loads the accumulator with the value contained in $1098.

Direct (DIR)

Direct addressing ("Page zero" in standard terminology [1]), has an instruction length of 2 bytes and works the same way as extended addressing, except that it is only possible to access the operands in the $0000–$00FF area of the memory. The benefit of this is that the address can be specified with only one byte. Variables that are used frequently are often put in this memory area to save time.

Indexed (INDX, INDY)

In indexed addressing mode, the index register X or Y is used when calculating the effective address. The effective address is variable and depends on the contents of register X (or Y) and an 8-bit unsigned offset contained in the instruction. For example, assume that index register X contains $ABCD. The instruction `LDAA 3,X` will result in that the contents of address $ABD0 is loaded into accumulator A.

Inherent (INH)

In inherent addressing mode ("Implied" in standard terminology [1]) everything needed for execution of an instruction is known by the CPU. For example `INCB` will increase accumulator index B with one, or `ABA` will add accumulator B to accumulator A.

Relative (REL)

Relative addressing (Page Relative in standard terminology [1]) is only used for branching instructions. Branch instructions generate two machine code bytes. One for the opcode and one for the relative offset. The offset byte is a signed two-complement offset with a range from -126 to +127 bytes. If the branch condition is true, the offset is added to the contents of the program counter to form the effective branch address.

15.4 On-Chip Memory

The 68HC11 includes random-access memory (RAM), read-only memory (ROM), and electrically erasable programmable ROM (EEPROM) memory. The A family of the HC11 contains 256 bytes of RAM for storage of variables and temporary information.

The amount of ROM differs between the series of the HC11. The A family have 256 bytes of ROM for storage of user program instructions, compared to the E9 which has as much as 12 kilobytes. There are also members of the M68HC11 family with disabled ROM that instead uses external memory for the user program.

The M68HC11A8 has 512 bytes of EEPROM, whereas other members of M68HC11 family includes as much as 8.5 kilobytes. Under software control, data can be programmed into, or erased from the EEPROM. No extra power supply other than the normal 5 V_{dc} supply are needed for programming/erasing the on-chip EEPROM. The MC68HC11A8 was the first MCU to include CMOS EEPROM which is commonly used for semi-permanent data, such as calibration tables or product history information, it can also be used for program memory [2].

15.5 References

[1] DAGERMO, P. *Datorteknik – med inriktning mot 68HC11*, second ed. Studentlitteratur, 2001.

[2] MOTOROLA. *M68HC11 Reference Manual*.

16. Motorola 68000

16.1 Introduction

The Motorola 68000 (MC68000 / M68000 / 680x0) processor was released in 1979. It was developed from the 8-bit processors MC6800 and MC6809, released earlier in the 1970s. The MC68000 was used in Amiga and Atari computers as well as in several early Macintosh computers [2, 6].

The MC68000 is considered to be a 16-bit processor, due to the width of its data bus. The address bus, though, is 24 bits wide, meaning an addressable memory area of 2^{24} byte, i.e., 16 MiB. Internally, the processor is mainly a 32-bit processor, with 32-bit registers and operations. There are thus instructions for data fields of length 8, 16 and 32 bits, here called bytes, words and long-words, respectively [5].

The processor lacks instructions for floating points, but its architecture features the ability to use a coprocessor, mostly the MC68881 floating-point processor, providing up to 96-bit extended-precision format using 80-bit registers [2].

Following the original MC68000 were, among others, the 68020 with a 32-bit data bus and built-in cache memory, the 68030 with an internal memory management unit and the 68040 with onboard floating-point coprocessor [1].

16.2 Security

The MC68000 is at any instant in one of two modes: the supervisor mode or the user mode. By forcing user programs to operate only in the user state and by dedicating the supervisor state to the operating system, it is possible to build considerable security into a system, and to protect the operating system against errors in user programs. This arrangement has obvious advantages in a multiuser computer installation [1, 3].

When any exception occurs, generated by an instruction in a user program, the processor is forced into the supervisor state. User programs have no direct control over exception processing and interrupt handling [1].

16.3 Registers

The 68000 has eight data registers, named `D0` to `D7`. These are general, meaning that all operations that can be done on one of the data registers also can be done on the other data registers. Most operations involving data manipulation act on these registers [1].

There are also eight address registers, `A0` to `A7`. These are pointer registers, holding addresses to data elements in memory [1].

Register `A7` is somewhat special. It is the stack pointer (SP), used to address the parts of the memory storing data for subroutine calls and temporary data. Actually, for security reasons there are two different stack pointers: The USP (user stack pointer) and the SSP (supervisor stack pointer). Both are referenced to as `A7`, but depending on whether the processor is in supervisor mode or in user mode, it means different stacks.

One special instruction, `MOVE USP`, gives the supervising program access to the USP. Programs in user mode, however, have no access to SSP [6].

MC68000 has only two more special-purpose registers. The first is the program counter (PC) which contains the address of the next instruction to be executed. The second is a 16 bits wide register called the status register (SR), shown in Figure 16.1. Its eight most significant bits are called the system byte and stores the processor status, such as a supervisor/user mode bit and interrupt information. The system byte is only used by programs running in supervisor mode, such as the operating system. The other eight bits contain the condition codes that indicate the results of previous arithmetic and logical operations, such as carry and overflow information [1, 4].

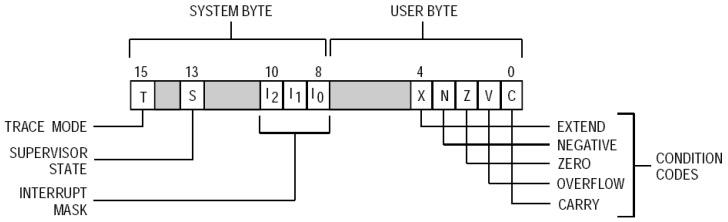


Figure 16.1: The flags or status bits of the status register (from [5])

.

16.4 Addressing Modes

On the MC68000, an instruction operand is specified by using one of 12 addressing modes. An addressing mode tells the CPU how to locate the data or address needed by the instruction. On the MC68020/30/40 processors, 18 addressing modes are available [2]. Some of them are shown in Table 16.1 on the next page.

16.5 Instructions

MC68000 instructions consist of an opcode and zero, one or two operands. The operands specify the source and the destination for the read and write cycles of the instruction. For example, `MOVE D1,D2` copies the content of register `D1` to register `D2` [2].

Operations on bytes, words and longwords are denoted by the addition of .B, .W, and .L, respectively, to the opcode. For example, the operation `ADD.L D0,D1` adds the 32-bit data in register `D0` to the 32-bit data in `D1` and puts the 32-bit result in `D1`. The operation `ADD.B D0,D1` adds the least significant 8 bits of `D0` to the corresponding 8 bits of `D1` and puts the result in `D1`. When a subsection of a data register is operated on, the remainder of the register is unaffected. For example, `ADD.B D0,D1` does not modify bits 8 to 31 of the destination register `D1` [1].

Table 16.1: Some of the addressing modes of the MC68000 processors

Mode	**Example**	**Meaning**
Direct mode	`A2`	An instruction directly accesses the contents of the address register
Indirect mode	`(A2)`	The operand specifies the contents of the memory location whose address is contained in A2.
Indirect with displacement	`(8,A2)`	The operand accesses the contents of address (A2) + 8
Indirect with index and displacement	`8(A2,D4)`	The operand accesses the contents of address (A2) + 8 + D4
Indirect with postincrement	`(A2)+`	The operand specifies the contents of the memory location whose address is contained in A2. After the memory access, the address register is increased with 1, 2 or 4, depending on operation.
Indirect with predecrement	`-(A2)`	Before the memory access, the address register is decreased with 1, 2 or 4, depending on operation. The operand then specifies the contents of the memory location whose address is contained in A2.

MC68000 is a CISC (Complex Instruction Set Computer) processor and as such it has many instructions [1]. These can be categorized into data movement, arithmetic and logical operations, program control and system control.

The number of basic instructions are more than 80. For example, in the data movement category, we have the instructions `MOVE`, `MOVEA`, `MOVE to CCR`, `MOVE to SR`, `MOVE from SR`, `MOVE USP`, `MOVEM`, `MOVEQ`, `MOVEP`, `LEA`, `PEA`, `EXG` and `SWAP`. If we were to count all variations of every instruction we would get significantly more than 80 instructions. For example the branch instruction `Bxx` is actually a collection of 16 different branching instructions, where `xx` can be `EQ` (equal), `GT` (equal), `VS` (overflow set), `PL` (plus), etc. [1].

In the arithmetics category, the processor also has instructions for binary-coded decimals (BCD). In the BCD form, decimal digits are represented as four bits, which means that a data register can hold a 8-digit decimal number. The supported instructions are addition, subtraction and negation of BCD numbers. This can for instance be useful in economics applications, which then don't have to convert from decimal form to binary form and back. BCD arithmetics also guarantees preserved accuracy [1, 6].

16.6 Assembler Language Example

The following example shows an implementation of a Fibonacci procedure in MC68000 assembler [6].

```
*************************************************
* Recursive Fibonacci-procedure
*   In: the number N in D0.W
*   Out: answer i D0.W
*************************************************

FIB     CMP.W   #1,d0       * simple case?
        BGT     DOFIB       * no, recursion
        RTS                 * done
DOFIB   MOVE.W  D0,-(A7)    * save current value of D0 to stack
        SUB.W   #1,D0       * decrement N to N-1
        JSR     FIB         * do F(N-1)
        MOVE.W  D0,-(A7)    * save result
        MOVE.W  2(A7),D0    * restore N
        SUB.W   #2,D0       * decrement N to N-2
        JSR     FIB         * do F(N-2)
        ADD.W   (A7)+,D0    * calculate sum
        ADD.L   #2,A7       * pop saved word
        RTS                 * return
```

16.7 References

[1] CLEMENTS, A. *Microprocessor systems design: 68000 hardware, software, and interfacing*, third ed. PWS Publishing Company, 1997.

[2] FORD, W., AND TOPP, W. *Assembly language and systems programming for the M68000 family*, second ed. Heath amd Company, 1992.

[3] HELLAWELL, N. *Coding the 68000*. Chartwell-Bratt, 1989.

[4] MOTOROLA, INC. *Motorola M68000 Family Programmers Reference Manual*, 1992. Also available at `http://www-clips.imag.fr/projet-systeme/68040/-68kprm.pdf`.

[5] MOTOROLA, INC. *M68000 8-/16-/32-Bit Microprocessors User's Manual*, ninth ed., 1993. Also available at `http://lars.nocrew.org/computers/-processors/M68000/MC68000/manual.pdf.bz2`.

[6] WILLIAMS, S. *Programmera 68000*. Pagina Förlags AB, 1987.

17. Ricoh 2A03

17.1 Introduction

The 2A03 processor is a modified version of the MOS 6502 processor. The MOS 6502 was released in September 1975. What differs the 2A03 from the 6502 is that the 2A03 processor lacks the Binary Coded Decimal (BCD) mode, that the 6502 has. The 2A03 can also handle sounds and work as a psuedo-Audio Processing Unit (pAPU) [1]. Because the MOS 6502 processor was much cheaper than the other processors when it entered the market, costing only $25 compared to Intel's 8008 and Motorola's 6800 which costed $179, it started selling directly [2]. And when Nintendo decided to enter the arcade game market they wanted a cheap console with good quality games, since the company's profit would be made from the games. So they altered the MOS 6502 into the 2A03 and got Ricoh to produce the chip after promising them a three-million chip order. So in 1983 the 2A03 chip entered the console market in America in the form of a little grey box called Nintendo Entertainment System (NES, see Figure 17.1) and was the top selling toy there in 1987 [1].

Figure 17.1: The Nintendo Entertainment System console (from [1])

17.2 Overview of the Processor

The 2A03 is an 8-bit processor with a 16-bit address bus and is a little endian processor which means that addresses are stored in the memory with the least significant byte first [1]. The clock frequency is roughly 1.79 MHz. When this processor was made, the RAM was faster then the CPU, so this processor uses fewer registers and focuses on RAM access instead. Even though this processor had a low clock frequency it was actually competing with other CPUs that was running with clock frequency four times as high, because the 6502 and the 2A03 are pipelined static processors. Other CPUs at this time were microcoded and had internal frequencies comparable with the 6502 (for instance the Z80s internal clock was divided by four) [2].

17.2.1 Registers

2A03 has 6 registers, an 8-bit accumulator register, two 8-bit index register, an 8-bit status register, an 8-bit stack pointer and a 16-bit program counter [2]. A closer look on the status registers, which contains 8 bits of different single bit flags, follows:

Carry Flag – If an overflow occurred in the last instruction the carry flag is set which allows the system to perform calculations with numbers longer than 8 bits.

Zero Flag – This flag is set if the result of the previous instruction was zero.

Interrupt Disable – This flag can be set to make the system ignore Interrupt Request (IRQ) interruptions until the flag is switched back.

Decimal Mode – The decimal mode flag is used to switch into BCD mode but since the 2A03 doesn't have a BCD mode this flag will be ignored.

Break Command – To indicate that a Break (BRK) instruction has been used, this flag is set causing an IRQ.

Overflow Flag – An overflow can occur if the last instruction resulted in an invalid two's complement. For example if two positive numbers are added and the result is negative. If this happens, the overflow flag will be set.

Negative Flag – If the sign bit (bit 7 of a byte) is set then this flag will be set as well.

17.2.2 Instructions

The 2A03 instructions can vary depending on what kind of addressing that mode is using. Therefore, some instructions come in multiples. So, basically there are 56 different instructions and 156 valid opcodes (operation codes) out of 256 possible [1]. But the 2A03's original processor (MOS 6502) is known to have roughly 32 undocumented opcodes that triggers many valid instructions at the same time, which can lead to unexpected results [2]. Depending on the addressing mode, the instructions are either one, two or three bytes long. The opcode is the first byte and the remaining bytes are operands. All instructions can be divided into several functional groups [1].

- Load/Store operations
- Register Transfer Operations
- Stack Operations
- Logical Operations
- Arithmetic Operations
- Increments/Decrements

- Shifts
- Jumps/Calls
- Branches
- Status Register Operations
- System Functions

There was a bug in the 6052 that wasn't fixed in the 2A03. The indirect jump instruction JMP, could jump to the wrong address if the address was hexadecimal xxFF. If the address was xxFF it would access xxFF and xx00 instead of xxFF+1 [2].

17.3 Memory

As mentioned before the 2A03 has a 16-bit address bus, capable of storing up to 64 kB. The first 256 addresses are part of the "zero page" or "direct page" mode. It allows users to access the first 256 memory locations with a single 8-bit address to allow quicker execution. Other addressing modes also included in the 2A03 was implied (1 byte instruction); absolute (3 bytes); relative (2 bytes); accumulator (1 byte); indirect,x and indirect,y (2 bytes); and immediate (2 bytes). For general purpose addressing the absolute mode was used. When conditional branch instruction was implemented for moving the program counter up to 128 bytes forward or backwards, relative addressing was used. In the accumulator mode the accumulator was used as an effective address, therefore it didn't need any operand data. Immediate mode used an 8-bit literal operand. For array processing or other kind of looping the indirect addressing mode was useful. Using indirect,y mode the 8 bits in the y register was added to a 16-bit base located somewhere in the zero page memory.

Instead of using the X and Y registers as "normal" index registers, they where used as offsets. So by incrementing X or Y you could index the array. Even though the array might be located at any position in the 16-bit address space, this was only a two-cycle 8-bit operation (given the 16-bit base read from the zero page) [2]. In the NES console the 2A03 accesses memory using buses. As shown in Figure 17.2 on the next page we can see that the memory is divided into three parts, the CPU's RAM, cartridges ROM and the I/O registers. To inform the components that the request is a read or a write, the 8-bit control bus is used. In order for the components to read or write from or into the selected address, the data bus is used. ROM is a read-only memory, and is accessed via a MMC to allow bank switching to occur. In order for the CPU to communicate with other components in the system, i.e. the Picture Processing Unit (PPU), the I/O register is used.

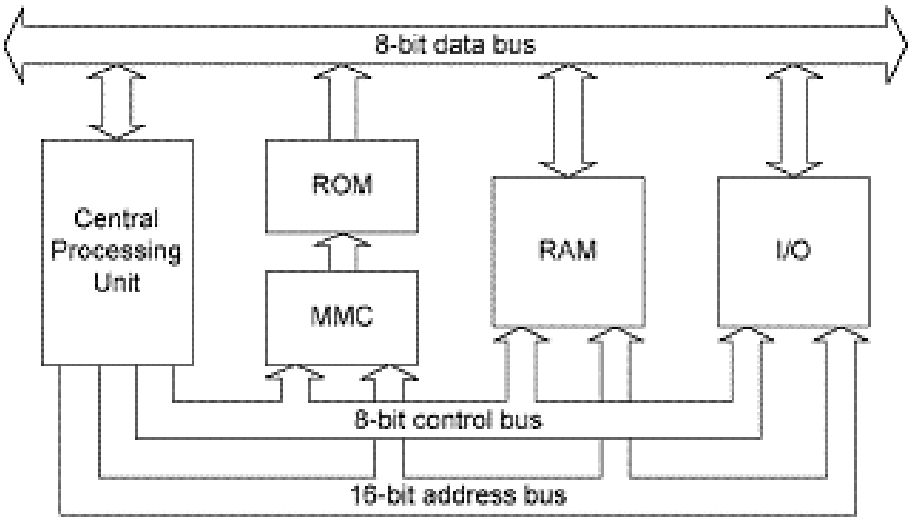


Figure 17.2: Processor Diagram (from [1])

17.4 References

[1] DISKIN, P. *Nintendo Entertainment System Documentation*, Aug. 2004. Also available at `http://nesdev.parodius.com/NESDoc.pdf`.

[2] WIKIPEDIA. Mos technology 6502. Web page, Sept. 20, 2005. `http://en.wikipedia.org/wiki/MOS_Technology_6502`.

18. Sun Niagara: A Multithreaded Processor

18.1 Introduction

The Niagara processor from Sun Microsystems has not yet been released. The exact time of release is not yet decided on, but sources tell that 8- and 6-core versions of Niagara will be presented before the end of 2005 [11]. Until then some information about Niagara will be presented here.

The Niagara processor is part of Sun's UltraSPARC family [8], but differs significantly from its predecessors with an unusual design that includes eight processing engines, each of which is capable of handling four threads simultaneously.

18.2 Overview of the Processor and Explanations

As previously mentioned, the Niagara has not yet been released, this, and the fact that Sun has not been very generous with details, means that the information collected here must not be regarded as the absolute truth.

A description of Niagara says that it is a "Blade processor including 8 Afara-created (simplified UltraSPARC I) cores on a die. Each core handle 4 threads and supports Switch on Event Multi-Threading (SoEMT)" [2].

Some quick specifications from the same source say the following:

L2 cache	3 MB
L1 cache	8x [16 kB I, 8 kB D]
Memory	4xDDR2 (20 Gb/s)
Pipeline	6-stage

The description and the specifications will be explained here.

Afara is short for Afara Websystems, Inc., a company acquired by Sun Microsystems in 2002 [10]. The company developed SPARC-based microprocessors. These processors were used in the previously mentioned UltraSPARC family.

SPARC stands for Scalable Processor Architecture, and "is an open set of technical specifications that any person or company can license and use to develop microprocessors and other semiconductors based on published industry standards" [9]. SPARC specifications are owned by an independent, non-profit organization called SPARC International, since 1989, when Sun Microsystems transferred it to them.

This means, Afara Websystems used the SPARC specifications to design their own processor, which later on has been used by Sun for its own processor family, the UltraSPARC family. The processor family has five members currently available, they are called UltraSPARC IIi, IIIi, III, IV and IV+.

SoEMT means that the processor switches task in case of long pauses, e.g. when the current task misses in the cache [7].

What we have now is a processor consisting of eight smaller and simplified processors, each of which are capable of handling four threads simultaneously and, in case of long pauses, switches between these threads. These smaller processors are based on the SPARC specifications.

The cache is used to save information needed often by the process currently run [6]. To be faster, many processors use two levels of caching. The first level of caches is implemented on the same die as the processor. The second level is often off-chip and is searched if the desired information can not be found on the first level cache. In the previously stated information about the Niagara, it can be seen that the processor uses two levels of caching. It seems like the second level cache is off-chip, based on the available information. The first level of cache storage is divided into two parts, one which handles instructions (I) and one which handles data (D) [4].

The memory of the Niagara is 4xDDR 2, DDR stands for Double Data Rate memory and 2 means it is the second generation of DDR. DDR is a SDRAM memory, SDRAM stands for Synchronous Dynamic Random Access Memory. The memory contains the instructions and data of a program, the RAM parts imply that memory accesses take the same time no matter what information to get [6]. Easily explained it can be said that a SDRAM releases 1 bit per clock cycle, DDR releases two bits per clock cycle and in DDR2 the rate of the clock cycle is doubled, thereby releasing bits faster [3].

One explanation of pipelining is “an implementation technique in which multiple instructions are overlapped in execution, much like an assembly line” [6]. A pipeline stage is, according to the same source, “A step in executing an instruction that occurs simultaneously with other steps in other instructions and typically lasts one clock cycle.” The fact that the pipelining used in the Niagara has six stages imply that each instruction is divided into six steps.

In summary, Niagara is a processor consisting of eight smaller, simplified, processors. These smaller processors can handle four simultaneous threads each. When executed, each instruction is divided into six steps and pipelined. Niagara has a quite fast memory and uses a two level cache.

18.3 Some Opinions about Niagara

The Niagara marks a new approach in the designing of processors since it is using multithreading. Previously there have been processors handling two threads simultaneously, but the 32 threads of Niagara yields a big step forward. Designers see three advantages with this [5] . Firstly, the processor is kept busy with real work from another thread while any one thread is waiting for data from a memory. Secondly, multicore CPUs help control signal speeding across the chip while the length of the signal paths are limited. Finally, today’s server software requires no changes to adapt to these new processors. On the other hand it is hard to predict how these chips will perform, it is also hard to design the synchronization of and the communication between threads.

In an article, Charlie Demerjian discuss other pros and cons of Niagara [1]. The article also reveals information about Niagara which Demerjian has extracted from a conversation with Sun’s Vice President Marc Tremblay.

The fact that Niagara features in-order execution results in that the cache size may be of less importance than it is in other architectures. The reason for this is that any other thread can be executed when the first thread is waiting; resulting in less penalty for memory misses. The backside is the ability to optimize how the program is executed.

The two kinds of caches also facilitate interaction between threads. To pass data between threads in the same core, just read and write to the level one cache, for data exchange between threads on different cores, use the level two cache in the same way. The improved data exchange will affect software design greatly.

The previously lessened penalty for memory misses results in the branch prediction being less aggressive, which in turn makes the development easier and the dies smaller.

If things can be passed between threads with no penalty, the OS in which the processes will be run has to be adjusted to that. Fortunately for Sun the two main OSes in which this processor will be used are Solaris, which is owned by themselves, and Linux, which is completely open.

According to Demerjian, Marc Tremblay repeatedly mentioned that Niagara was "network facing" not "data facing". The statement implies that Niagara will be good at, for example searching, web page serving and streaming media, and other services giving service to thousands of tasks a second.

Niagara also takes a new approach in efficient use of resources available. The aim is to keep all the resources in the chip as busy as they can be, for as long as there is data. This approach will avoid the typical peaky CPU behaviour where the processor hovers around zero and then spikes to near 100 %. The power consumption of Niagara will lie around 60 watts, much less than many other PC-CPUs on the market now.

All in all, Demerjian seems to think the future for the Niagara processor looks bright.

18.4 References

[1] DEMERJIAN, C. Sun's Niagara falls neatly into multithreaded place. *The Inquirer* (Nov. 2, 2004). Available at `http://www.theinquirer.net/?article=19423`.

[2] ENDIAN.NET. Sun Niagara. Web page, Sept. 18, 2005. `http://endian.net/details.asp?tag=Niagara`, date visited given.

[3] LOSTCIRCUITS. DDR II sacrificing latency. Web page, Jan. 6, 2003. `http://www.lostcircuits.com/memory/ddrii/`.

[4] MCMILLAN, R. Sun offers glimpse of Niagara. Web page, June 17, 2004. `http://www.infoworld.som/article/04/06/17/HNsunniagara_1.html`.

[5] MERRITT, R. Sun, IBM take processors multithreaded, multicore. *EE Times* (Feb. 28, 2003). Available at `http://www.eetimes.com/story/OEG20030228S0017`.

[6] PATTERSON, D. A., AND HENNESSY, J. L. *Computer Organization & Design: the Hardware/Software Interface*, second ed. Morgan Kaufmann Publishers, Inc., San Francisco, California, 1998.

[7] SEVERINOVSKIY, E. Intel hyper-threading technology review. Web page, Sept. 18, 2005. http://www.digit-life.com/articles/-pentium4xeonhyperthreading, date visited given.

[8] SHANKLAND, S. Sun's Niagara chip moves into testing. Web page, May 12, 2004. http://news.zdnet.co.uk/hardware/chips/-0,39020345,39154430,00.htm.

[9] SPARC INTERNATIONAL, INC. FAQ: What is SPARC? Web page, Dec. 7, 2003. http://www.sparc.com/faq.html.

[10] SUN MICROSYSTEMS, INC. Sun Microsystems aquires Afara Websystems, Inc. Web page, July 23, 2002. http://www.sun.com/smi/Press/sunflash/2002-07/-sunflash.20020723.1.html.

[11] VANCE, A. Sun's Niagara: speeds and feeds emerge. Web page, Sept. 9, 2005. http://www.theregister.co.uk/2005/09/09/niagara_many_cores.

19. Transmeta Crusoe

19.1 Introduction

In January 2000 a start-up company named Transmeta released a family of processors, named Crusoe™, with an architecture that differed from every other in the market at that time. Low power consumption were put before high performance, and the target was mobile and embedded systems. The goals were achieved by letting much of the work usually done in hardware be done in software instead, and thereby reducing the die size. By also integrating the northbridge on the processor chip, combined with another new technology called LongRun, power consumption was reduced even further. The Crusoe processors are x86-compatible [3].

19.2 Overview

The core of the Crusoe is a VLIW, Very Long Instruction Word, core with 128-bit or 64-bit long instructions in a 474-pin ceramic BGA, Ball Grid Array. Each 128-bit instruction word contains four operations which can be executed in parallel.

It has an in-order 7-stage integer pipeline and a 10-stage floating point pipeline. There are 64 internal integer registers, two integer ALU's and one floating point ALU [6].

19.3 Memory Hierarchy

The Crusoe is a pure RISC processor with 128 KB L1-cache and between 0 to 512 KB L2 cache depending on the model, see Figure 19.4 on page 83. Models without L2 cache are designed for use in handheld devices like PDA's and models with larger L2 cache are mostly used in laptops and tablet PC's.

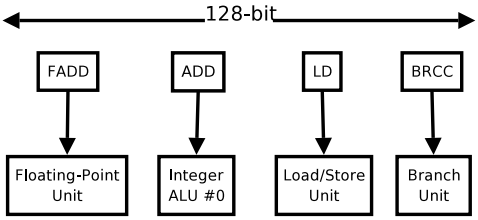


Figure 19.1: A VLIW molecule (from [5])

The instructions executed by the VLIW-core are called molecules since they consist of four instructions, called atoms (see Figure 19.1). They are encoded in little endian byte order [6].

19.4 Execution

The execution takes place in a 7-stage pipeline for integers and a 10-stage pipeline for floating point number, see Figure 19.2. The pipeline follows the model found in most RISC processors. Branch prediction is implemented in the Code Morphing software [2].

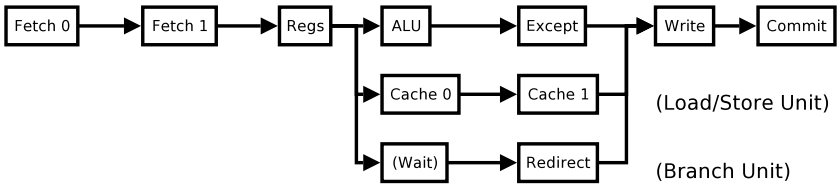


Figure 19.2: Pipeline (from [1])

- `Fetch0` The first 64 bits of a 64-bit to 128-bit bundle are fetched.
- `Fetch1` The second 64 bits are fetched (for 128-bit bundles only).
- `Regs` Read source registers and decode/disperse instructions.
- `ALU` Execute single cycle operations in ALU0 and ALU1.
- `Except` Complete two-cycle ALU0/ALU1 ops and detect exceptions.
- `Cache0` Initiate L1 data cache access based on register address.
- `Cache1` Complete L1 data cache access, TLB access and alias checks.
- `Write` Write results back to GPRs or store buffer.
- `Commit` Optionally latch the lower 48 GPRs into the shadow registers.

19.5 Code Morphing™

As mentioned earlier the Crusoe executes VLIW's in its core, and is compatible with x86. This is achieved through a software layer between the BIOS and the processor called Code Morphing, see Figure 19.3 on the next page. The Code Morphing software recompiles x86 machine code to VLIW code on the fly when executing a program and the translations are stored in a translation cache to speed things up. The most frequently used code will be optimized by the software for even higher performance. Since the translation is done in software, larger buffers and tables can be used compared to doing it in hardware, thus more history can be stored and branch prediction can be made more efficient than in hardware.

Since the VLIW's contain four instructions which can be executed in parallel the Code Morphing software can pack translated x86 instructions into a VLIW out of order,

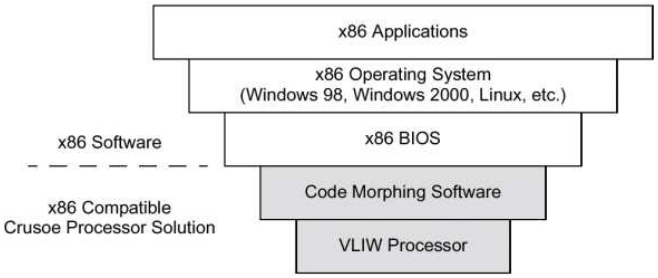


Figure 19.3: Layers of the Crusoe ISA (from [3])

to increase performance. This leads to some problems if an exception occurs, since all instructions before the one which caused the exception must be executed. This is solved by having a copy of all x86 registers, so-called shadow registers, and only if a VLIW executed without an exception the working registers are copied to the shadow registers. Otherwise, if an exception occurred, it will copy the shadow registers to the working registers and start all over again, executing the instructions in-order. This allows for proper exception handling though the x86 code is executed out of order [5].

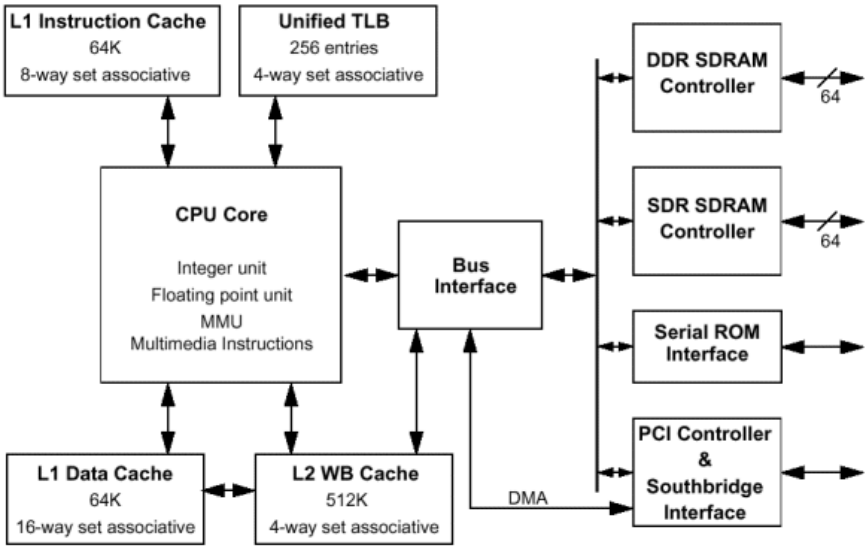


Figure 19.4: TM5600 block diagram (from [6])

19.6 LongRun™

To reduce the power consumption a feature called LongRun is introduced that can adjust both the clock frequency and the voltage level to the processor, the latter being a

feature not found in many other processors. When the processor is not fully utilized it can lower its speed and thereby also lower its voltage level since a higher speed require a higher voltage level. The energy consumed is linear to the clock frequency and quadratic to the voltage level. This means that a 10 percent reduction in clock frequency leads to a 30 percent reduction in power consumption due to the cubic relationship when lowering both speed and voltage [6].

19.7 Performance

Since the Code Morphing software optimizes frequently used code, tools that measure performance will sometimes be faster the second time they are run. Not many benchmarks takes this into consideration it seems, but still, a Transmeta Crusoe TM5800 running at 600 MHz is comparable to an Intel Pentium 3 running at about 550 MHz.

Another low-power rival is the Via C3 processor. A Crusoe TM5600 running at 600 MHz is almost twice as fast as a C3 also running at 600 MHz on floating point operations, and about equal for integer operations [4].

19.8 Applications

The Crusoe processor is used in many portable devices such as Sony VAIO Picturesooks and HP thin client t5700. Crusoe's successor, the Efficeon processor can be found in the ECS 532 15-inch notebook.

19.9 References

[1] ANONYMOUS. Crusoe exposed: Reverse engineering the Transmeta TM5xxx architecture I. Web page, Feb. 1, 2004. http://www.realworldtech.com/-page.cfm?ArticleID=RWT010204000000.

[2] ANONYMOUS. Crusoe exposed: Reverse engineering the Transmeta TM5xxx architecture II. Web page, Feb. 27, 2004. http://www.realworldtech.com/-page.cfm?ArticleID=RWT012704012616.

[3] GREENHALGH, P. Transmeta crusoe. Web page, Jan. 31, 2001. http://www.beyond3d.com/articles/transmeta/index1.php.

[4] SMITH, V. Preliminary benchmarks: VIA C3 versus Transmeta TM5600. Web page, Aug. 16, 2002. http://www.vanshardware.com/articles/2001/september/-010921_Transmeta_v_C3/010921_Transmeta_v_C3.htm.

[5] STOKES, J. Crusoe explored. Web page, Jan. 6, 2000. http://arstechnica.com/-articles/paedia/cpu/crusoe.ars.

[6] TRANSMETA CORPORATION. *Crusoe Processor Model TM5400*, Feb. 8, 2000. Also available at http://www.transmeta.com/crusoe_docs/-tm5400_productbrief_000802.pdf.

Part II

Media and Memory

20. Blu-ray

20.1 Introduction

Blu-ray is a new type of disc used to store data, something the creators hope will replace the DVD. A daily-used term like Blue-ray cannot be registered as a trademark, thus the e was dropped and the product named Blu-ray. The name Blu-ray refers to the blue laser technology used to read from and record to the disc. Ordinary CDs and DVDs use red laser, while blue laser allows greater density and storage capacity.

In this report we are going to tell you about the basics about the Blu-ray. You will get to know a bit about the short history of blue ray, what the blue laser is and the how and whys of the disc itself. We will end the report with a comparison to Blu-ray's greatest adversary, HD DVD, which is another technology using blue laser but with the same disc structure as an ordinary DVD.

20.2 History

Blu-ray, as well as its concurrent HD DVD, was developed to support recording of High Definition video. The 4.7 GB capacity of a DVD is simply not enough anymore.

The concept of Blu-ray was introduced in February 2002 by a large group of companies, including Dell, Hewlett-Packard, Hitachi, LG Electronics, Matsushita (Panasonic), Mitsubishi, Pioneer, Philips, Samsung, Sharp, Sony, TDK and Thomson. The current Blu-ray Disc Association consists of over 140 companies from all over the world. Blu-ray disc recorders are already sold in Japan, for recording HDTV. The product will probably be launched in USA early 2006 and shortly after in Europe [1].

20.3 Technical Information

20.3.1 Summary

Storage capacity (single-layer)	25 GB
Storage capacity (dual-layer)	50 GB
Laser wavelength	405 nm
Numerical aperture (NA)	0.85
Protection layer	0.1 mm
Data transfer rate	36.0 Mbps (1x)
Data transfer rate (movie application)	54.0 Mbps (1.5x)
Video compression	MPEG-2, MPEG-4 AVC, VC-1

Blu-ray supports multi-layer discs as well. Right now a disc can store two data-layers. The laser will first read one layer and then refocus on the second layer. This is

done automatically without any user inference. The format can easily support adding more layers, allowing the capacity to increase up to 200 GB.

The Blu-ray player is promised to maintain compatibility with DVD. An often used method to achieve this has been to simply switch the objective lens. Even if the difference of the thickness of layers between the two formats is big the problem can easily be solved by using this method. The Blu-ray developers are working on a laser lens is capable of both red and blue-violet laser. This is necessary to build portable players and notebook drivers [1].

20.3.2 Information

Despite the name, Blu-ray actually uses blue-violet laser to read and write to discs. The blue-violet laser has a shorter wavelength than the red laser used by CD and DVD, 405nm vs. 650nm.The idea of using a wavelength even shorter was considered, but it caused durability problems on the disc. Experiments has shown that optical discs and devices suddenly show a poor transmission factor when a shorter wavelength than 400 nm is used.

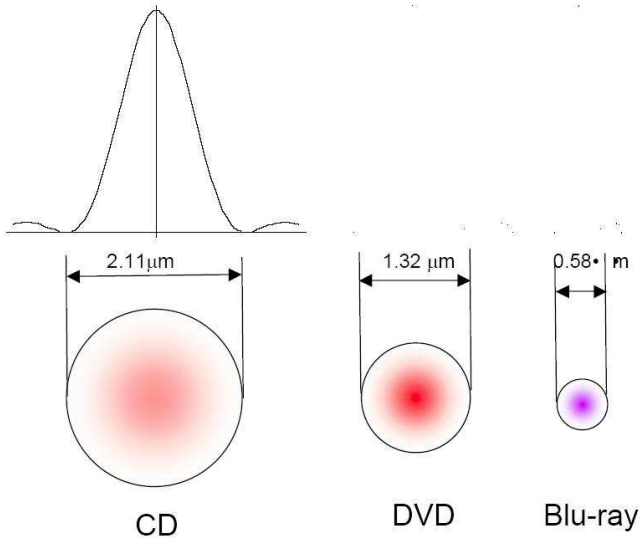


Figure 20.1: Comparision in beam size (from [2]).

The blue-violet laser allows better precision and thus more data of a disc on the same size as a DVD/CD. But changing to blue laser only increased the storage up to 12 GB. The thinner the cover layer of the disc is, the more easily the performance of the objective lens to converge the laser beam can be improved. In the DVD-standard the thickness of the cover is 0.6 mm, leading to a NA value of 0.6. NA, Numerical Aperture, measures the optical lens ability to gather light and resolve fine details at a fixed object distance and has a value between 0 and 1. A NA of 0 means that no light

is gathered, which means a high value is preferred (see Figure 20.2). Together with the wavelength the size of the laser beam is defined. A high NA and a short wavelength results in a small laser beam. That is what Blu-ray wanted to achieve, see Figure 20.1).

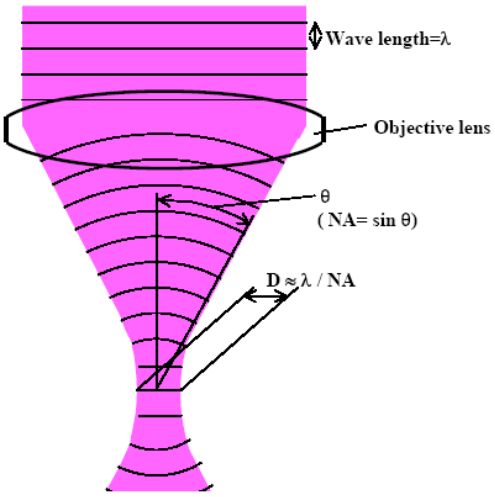


Figure 20.2: Numerical Aperture is defined as sin(θ). Where θ is half angle of converging light converged by an objective lens. Around 80% of light energy is converged in an area with diameter of λ / NA (from [2]).

Rather than to solve all the problems that a 0.6 mm thickness lead to it was decided to reduce the thickness to 0.1 mm. That lead to the much better NA value of 0.85, but also introduced a new set of problems. The recording laser is very close to the surface of the disc. The advantage of this is that the laser has less material to read though, thus the higher NA. But the disc becomes more vulnerable as well. The lens must be closer to the optical disc, which increases the chances of the lens accidentally hitting the disc. Protective measures can be taken by the hardware to sense danger and avoid a hit, but it will put more pressure on the manufactures. Also, the Blu-Ray discs will be much more sensitive to fingerprints and accidental scratches. But if the hard coating that has been developed keeps its promises the disc should be even more resistant to damage than the current DVD [2].

20.4 A Comparison to HD DVD

Unlike Blu-Ray HD DVD is more of a direct extension of DVD, using blue laser instead of red while still using the same thickness of the cover layer. That means the production costs will be lower for HD DVD and at the moment the production time is shorter as well. Despite that HD DVD is not yet for sale while Blu-ray already is used in Japan.

A great deal of powerful companies support Blu-ray, but more than 40% of the Hollywood movie-studios are behind HD DVD. Some companies, like Thomson Electronics, prefer not to take sides and show indication to support both formats. Compatibility

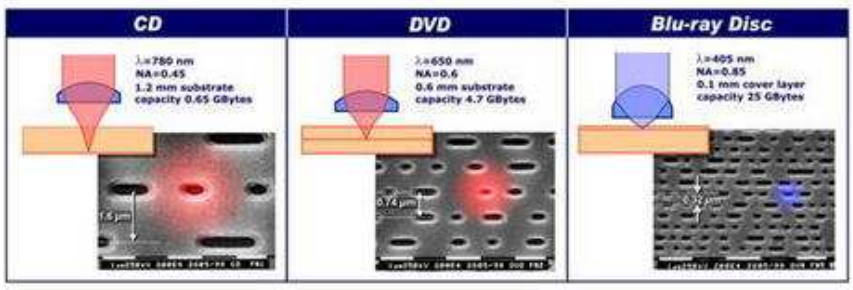


Figure 20.3: CD: λ=780 nm, NA=0.45, 1.2 mm substrate, capacity 0.65 Gbytes. DVD: λ=650 nm, NA=0.6, 0.6 mm substrate, capacity 4.7 Gbytes. Blu-ray: λ=405 nm, NA=0.85, 0.1 mm substrate, capacity 25 Gbytes (from [4]).

with CD/DVD will be available for both formats; compatibility between Blu-ray and HD DVD is another question. The technology to achieve this should be a small problem, but it is doubtful there is a will to produce it.

Looking at the facts Blu-ray seems like a more worthy follower to DVD. First, the storage capacity is 25 GB to 15 GB, the difference is even more noticeable when considering the dual-layer disc. Also, Blu-rays combination of high NA/low wavelength requires a lower rotation speed to reach the transfer rate of 36 Mbps that both formats have promised. But as always it comes down to money; HD DVD will be cheaper and easier to produce. Blu-rays far superior density is more space than the market currently requires.

At the moment there is no clear winner and both formats will be on the market. The new game-console Sony Playstation 3 will be using Blu-ray discs, something that might give them a slight advantage against the HD DVD. On the other hand, Microsoft's new X-box console might be using HD DVD in the future. But it is an interesting thing that Microsoft has not yet made any public promises [4, 3].

20.5 References

[1] BLU-RAY DISC ASSOCIATION. Blu-ray.com - everything you want to know about Blu-ray Disc. Web site, Sept. 20, 2005. `http://www.blu-ray.com`, date visited given.

[2] BLU-RAY DISC FOUNDERS. Blu-ray disc format - general. white paper, Blu-ray Disc Association, Aug. 2004.

[3] HD DVD PROMOTION GROUP. HD DVD Promotion Group. Web site, Sept. 20, 2005. `http://www.hddvdprg.com`, date visited given.

[4] SPATH, M., AND HERBERT. Blu-ray vs. HD DVD. Web page, Mar. 2005. `http://www.cdfreaks.com/article/186/`.

21. CD-RW

21.1 Introduction

Today almost every new computer has a DVD/CD burner installed. The burning of a CD requires a CD-R or CD-RW disc. The technology used when burning differs between the two types of discs but we will concentrate only on CD-RW. When burning a CD-R the laser creates pits and lands on the disc, the laser then uses them to read by notice the difference between the pits and lands. A burned CD-RW has different states in its material called amorphous and crystalline states, which resembles the pits and lands on the CD-R [2].

21.2 Time of Birth

In 1841 the research started with a sampling theorem by Augustin-Louis Cauchy which has evolved each year until 1970 when two workers at Philips completed the first glass disc and found out that a laser would be needed to read the disc. The laser was invented in 1958 and as late as 1980 the standard CD was founded by Philips and Sony. The need to be able to burn your own discs made the evolution continue and in mid 1997 the CD-RW was a big threat to the CD-R and superfloppy alternatives out on the market [3].

21.3 Overview

CD-RW stands for Compact Disk, Read and Write. Which means that you can both read and write to the CD. A regular CD is built up by a kind of plastic composition and can not be altered after the burning/pressing process. The CD-RW disk contains a dielectric material, a mix of silver, indium, antimony and tellurium. This kind of mix enable the re-writable capacity by melting and “freezing” the material over and over, thus creating pits and lands on the disc which reflects the light in different ways [4]. Because of the difference in material between the CD-R and CD-RW, some of the older CD-ROM drivers can not read RW disks. The main reason of this is that the material of the RW-disk does not reflect the laser as well as from CD-R disks.

21.4 Surface

In this section we will explain in detail the structure and composition of a CD-RW. Standard CDs measure 12 cm in diameter with a 15 mm diameter centre hole. The audio or computer data is stored from radius 25 mm (after the lead-in) to radius 58 mm maximum where the lead-out starts The first 4 mm of the disc is used by the SUA (system use area) which is a boot-section. The SUA-section is divided into two parts named Power Calibration Area (PCA) and the Program Memory Area (PMA).

21.4.1 PMA

The PMA tells where the data is stored on the disc, or rather the start and end of the stored data. It also store a temporary table of contents (TOC) for the individual packets on the disc that has been only partially recorded, a multisession. When it is closed/finalized, the TOC is written to the lead-in and the CD-RW can now be read by other devices than the CD-RW device.

21.4.2 PCA

The PCA is a sort of testing ground for the CD burner. In order to ensure that the write laser is set at the right level, the burner will make a series of test marks along the PCA section of the disc. The burner will then read over these marks, checking for the intensity of reflection in marked areas as compared to unmarked areas. Based on this information, the burner determines the optimum laser setting for writing onto the disc.

21.4.3 Information Area

After the SUA the information area starts, which is divided into three parts, lead-in, program area and lead-out. This is where the actual data is stored on the disc. When using multisession the disc will be divided as; lead-in, program area, lead-out, lead-in, program area, lead-out, etc.

Lead-In

The lead-in consists of a main channel and eight subcode channels labeled P-W where the Q-channel contains the TOC and time-codes. The TOC that is written when the burning process is done enables the reader to know where data is stored on the disc and is followed by a digital silence (empty area) in the main channel. The TOC can contain a maximum of 99 tracks even though the program area would fit more. For data there are no such limitations and the program area can be used fully. The P-channel indicates the start and end of each track and is used by simple audio players to know when the next track begins. In the subcode channels R-W there can be graphics that is shown when playing the music, e.g. title of song etc.

Program Area

Data stored on the disc is located on this section and usually consists of 330,000 sectors where each sector has 2048 bytes of data to be used. The capacity of the disc can be different depending on if multisession is used and depending on the disc, some discs have more sectors allocated on the program area thus increasing the total capacity of the disc and the use of multisession reduces the capacity by 13 Mb each time a new session is started. The increase in number of sectors is possible by using less track pitch, meaning the disc becomes “compressed”, because of this we also need to lower the scanning velocity. Since error detection is more needed for data than audio it follows that a disc containing data has less space for actual data than a disc using audio.

Lead-Out

After each burning session a lead-out is required to tell the CD-reader that the session has ended. The lead-out uses 13 Mb of digital silence, meaning that there are no data here and the reader knows that the data written this section is read. When using multi-session the burner also prepares a lead-in for the next session just after the lead-out on this session. The following sessions lead-out uses only 4 Mb instead of 13 Mb. Sometimes a disc uses overburning which means that a small part of the lead-out is used as program area. The lead-out consists of 6750 sectors (about 90 seconds or 15 Mb)

21.4.4 Layers

The recording layer is surrounded by two dielectric layers that act as an insulation for the lasers heat when it burns on the recording layer thus protecting the disc from being damaged. Above the upper dielectric layer is a reflective layer that reflects enough of the laser if the dielectric material has been crystallized so that the reader can determine a land. This layer is made of a mix of titanium and aluminum. At the bottom there is a polycarbonate substrate and a spiral groove that is used to guide the laser beam and to hold absolute time information and other data. The layers are illustrated in Figure 21.1.

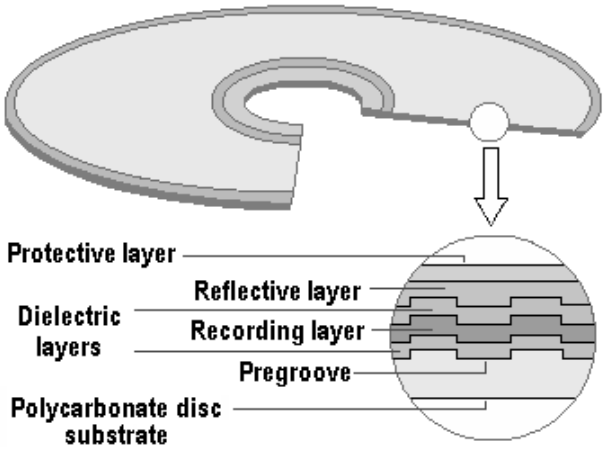


Figure 21.1: Layers on a CD-RW(from [1])

21.5 Phase Change

When a disc is burned the laser that is being used uses different strengths to alter the material on the disc, changing the material on the disc between an amorphous and crystalline state. The laser uses three different powers of the beam and uses them when reading, writing and erasing data on the disc. The highest power is called “Write Power” and converts the dot to an amorphous state which is the non-reflective state, it

heats the disc to a temperature of 500-700^o C which makes the atoms in the area to move rapidly in the liquid state, when if cooled fast enough the liquid state is "frozen in" and the amorphous state is obtained, this also makes the amorphous material to shrink leaving a pit where the laser dot was written. This results in a readable CD surface. When we want to erase data from the disc we use the middle-power called "Erase Power", which revert the atoms back to its crystalline state by heating the disc to around 200^o C. To read data the "Read Power" is used which is used only to read and therefor does not alter the material on the disc.

21.6 Specifications

UDF (Universal Disk Format) enables operating systems to read, write and erase data from optical media such as CD-RW that has been created by other operating systems. This specification is being used by CD-RW as an optional secondary standard. The UDF is a subsection in ISO[1]13346 which is an extension to the ISO 9660. The ISO 9660 limits the label to a maximum of eight letters (or numbers) all in upper-case, filenames to 31 characters and also limits to eight subdirectories, the UDF reduces these limitations and allows for e.g. 255 characters in filenames. At first UDF limited the write/erase capabilities of the discs to not be able to just rewrite old data as planned, we had to erase the entire disc and start over from the beginning again. In the updated UDF its possible to do a direct overwrite on discs and thereby enable drag and drop feature that is being used with CD-RW. It was planned to be used as an alternative to hard drives but it is not quite that easy to use [1].

21.7 References

[1] NEJSUM, R. Storage / CD-recordable / CD-rewritable. Web page, July 19, 2005. http://www.cephalon.dk/articles.htm.

[2] PHILIPS. *High Speed CD-RW*. White paper, Jan. 23, 2002. Also available at http://www.licensing.philips.com/includes/download.php?id=1095&-filename=1095.pdf.

[3] POHLMANN, K. C., AND DESPAIN, J. History of CD technology. Web page, 1999. http://www.oneoffcd.com/info/historycd.cfm.

[4] WIKIPEDIA. CD-RW. Web page, Sept. 17, 2005. http://en.wikipedia.org/wiki/CD-RW.

[1] International Organization for Standardization.

22. DVD

22.1 Introduction

The laser disc was invented by David Paul Gregg in 1958 and is the precursor to DVD. The audio tracks on the laser disc store audio signals digitally in the same way as compact discs (CD, released 1979), but video signals on a laser disc is not recorded digitally. Composite analog signals is translated into digital language and then decoded back into an analog format for display. It offered superior video and sound capabilities much like Video Home System (VHS, released 1976), prior to DVD.

The DVD was released in September 1995. DVD is an acronym for "Digital Versatile Disc" but it was originally "Digital Video Disc", that is to indicate its potential for non-video applications. It was a new optical disc storage with greater capacity of a CD. It can deliver data at a faster rate than a CD and the format allows for different kinds of storage.

In this study the purpose is to get a deeper understanding into its physical and application formats. Physically it can hold from seven times to over 25 times the digital data on a CD and it can also be used for different storage applications as video, audio or data storage as a DVD-video, DVD-audio and DVD-ROM format respectively. But first we are going to introduce applications of laser in such areas.

22.1.1 The Laser Appliance

DVD player uses laser to write or read from the disk. Laser is an acronym for "Light Amplification by Stimulated Emission of Radiation" [3]. The diode laser is the most compact among all commercial available laser products. Table 22.1 shows a list of diode laser wavelengths and their typical applications The products are important in many fields such as the following: Biomedicine, graphic arts, holography, optical data storage, trace element sensing, and telecommunications etc [5].

Table 22.1: Diode laser wavelengths and their typical applications

Wave length	Application area
635-660	Pointing, holography, DVD, CD
780	CD ROM
850-1310	Communications
1550	Communications, range findings
1625	Telecommunications testing

Special techniques are used to further narrow the range of wavelengths contained in the laser output and thus to increase the monochoromaticity. That it consist of light of almost a single color. That is to say the narrower the wavelength makes it possible to store more information on a disc [3].

22.1.2 The Physical Format

A DVD disc is made from a 0.6 mm thick disc of polycarbonate plastic coated with a much thinner reflective aluminum layer. Two such discs form a 1.2 mm double-sided disc. A single-layer DVD can store 4.7 Gigabyte (GB) in comparison to a CD which is less then 1 GB. So there are some reasons for DVD's greater data capacity with a different laser appliance [6]:

- Tighter track spacing with Smaller pit size

  Pits are small and tight depressions on the surface of a DVD disc that allow the laser pickup to distinguish between the digital 1's and 0's (see Figure 22.1.2). In order for a DVD player to read pits a laser with a smaller beam of light is used. By using a red laser at 650 nm (was 780 nm) wavelength and a numerical aperture[1] of 0.6 (was 0.45), the read-out resolution is increased by a factor of 1.65. This holds for two dimensions to 3.5. Despite sophistication on the beam, DVD also uses a more efficient coding method in the physical layer for error correction.

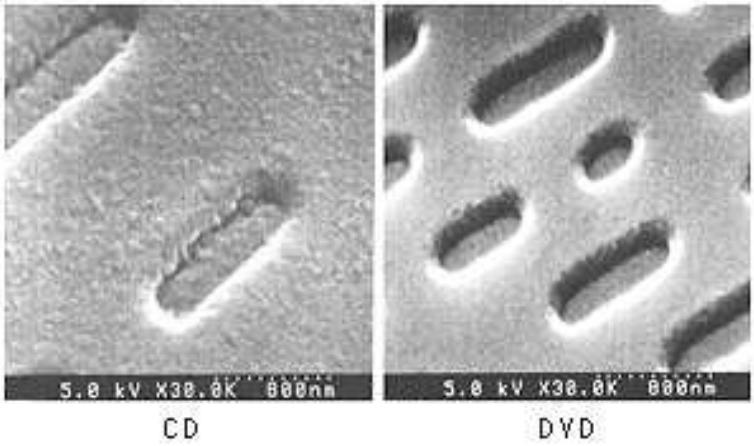


Figure 22.1: Photographs of pits recorded on CD and DVD media [4]

- Multiple layer capacity

  Some DVD discs may have up to four layers of information, with two layers on each side. To read information on the second layer on the same side, the laser focuses deeper into the DVD and reads the pits on the second layer. To read information from the other side of the DVD is cumbersome since it requires the user to flip the disc manually but the other positive point is that it does not need any backward function as in VHS.

  Based on DVD's double sided options, there are various disc construction formats which is represented in Table 22.2 on the facing page.

[1] Related to the lens that focuses pits thus reading 0.40 μm with 0.6 aperture.

Table 22.2: The number of sides and layers determines the disc capacity (from [6])

Name	sides and layers	capacity
DVD-5	single sided, single layer	4.7 GB
DVD-9	single sided, double layer	8.5 GB
DVD-10	double sided, single layer on both sides	9.4 GB
DVD-14	double sided, double layer on one side	13.3 GB
DVD-18	double sided, double layer on both sides	17.1 GB

22.1.3 The Application Format

Motion Pictures Experts Group (MPEG) is a professional association of motion picture engineers who develops standards for compressing audio and video signal, including the MPEG-1 and MPEG-2 video compression algorithms. Here we are going to find the effectiveness of these algorithms used. A DVD player may also able to recognize other formats as well [2].

- MPEG-1: A video compression algorithm that is part of the Video CD standard. MPEG-1 effectively compresses the video picture to about 1/140 of its original size.
- MPEG-2: A video compression algorithm that is part of the DVD-Video, Digital Broadcast Satellite, and Digital TV (including HDTV[2]) standard. MPEG-2 effectively compresses the video picture to about 1/40 of its original size. The picture quality from a MPEG-2 encoded source is superior to that of MPEG-1.

22.1.4 The Storage Formats and Quality

- DVD-Video: It holds movies which means video and sound. It provides up to 480 horizontal lines of resolution. In case CD, it uses compressed MPEG-1 video with 240 h. lines resolution which is roughly comparable to VHS [7]. This is a significant improvement over the 260 h. lines of resolution of standard VHS, and the 330 h. lines of resolution for broadcast TV.
  DVD-Video is compressed using MPEG-2. If the video information is left uncompressed, DVD's would contain only ten minutes of video footage, instead of several hours.
- DVD-Audio: It holds high-definition sound. It is the audio application format of the general DVD specification. It also concludes the following [6]:
  1. Pulse Code Modulation: PCM samples analog signals at regular intervals and encoding the amplitude value of the signal in a digital word. In CD format the word length is limited to 16 bits. The sampling rate is 44.1 kHz[3] [1]. The sampling rate must be at least two times the maxi-

[2]High-definition television provides higher resolution and increases the percentage of visual field.
[3]44,100 cycles or samples per second.

mum frequency signal to be reproduced. It is enough to produce audio up to 20 kHz.
2. High Resolution Stereo (2-Channel) Audio: DVD-Audio supports sampling rates of up to 192 kHz. It is more than four times the sampling rate of audio CD and up to 24-bit word length. The higher sampling rate means more accurate and realistic reproduction of the higher frequencies. Human hearing is sensitive in the range of 20 Hz to 20 kHz and the 192 kHz sampling rate is over nine times the highest frequency of human hearing.
3. Multi-Channel Audio: It refers to use of more than two channels. Stereo is equivalent to two-channel and Mono one. That is up to six, full-range, independent audio channels that can be recorded. And in terms of sampling rates and data words, multi-channel DVD-Audio can use up to 192 kHz and up to 24-bit word length. But practically uses 96 kHz sampling, because 6-channel audio uses three times the data capacity of two-channel stereo when both use the same sampling rate and word length. In that case, DVD-Audio uses a form of data compression in order to fit the high resolution stereo and/or multi-channel digital information.
4. Lossless Data Compression: DVD uses Meridian's lossless encoding/decoding algorithm in order to store the massive quantity of digital audio information [6]. No information is lost in the encoding and decoding process.

- DVD-ROM: It holds data, games, programs. Basically DVD-ROM is a data storage format just like CD-ROM. A DVD disc may contain any combination of DVD-Video, DVD-Audio, and/or DVD-ROM application content.

22.1.5 Recordable DVD Formats

The disc medium can be [6]:

- DVD-ROM ⇒ [Read Only Memory]
- DVD-R/RW ⇒ [R=Recordable once, RW = Re-Writable]
  Shorthand for "DVD dash Recordable and DVD dash Re-Writable".
- DVD+R/RW ⇒ [R=Recordable once, RW = Re-Writable]
  Shorthand for "DVD plus R and DVD plus RW". It uses a special metal alloy and can be switched back and forth between a crystalline phase and an amorphous phase, changing the reflectivity, depending on the power of the laser beam. Data can thus be written and re-written. The difference to DVD/RW is that it is better engineered.
- DVD-RAM ⇒ [Random Access Re-Writable]
  It is the same as a removable hard disk. Data can be stored in non-contiguous blocks, much like a computer hard disk. A single-sided 12-cm DVD-RAM disc holds 4.7 GB, so a double-sided 12-cm DVD-RAM disc holds 9.4 GB.
- DVD-R DL ⇒ [R=Recordable once, Double Layer]
  See Table 22.2 on the previous page.

22.2 Discussion

The DVD-Video format for movies is comparable with the Compact Disc (CD) for music and the major advantages to CD's are:

- Higher bit rate

  Typical data rate for DVD movies range from 3-10 Mbit/sec in comparison to 1 Mbits/sec in VHS-quality. Where minimum for audio is 4-8 kbit/sec (telephone) and for video is 16 kbit/sec (video-phone).

- Improved Picture Resolution Quality

  The DVD format provides up to 480 horizontal lines of resolution. This is a significant improvement over 260 horizontal lines of resolution of standard VHS and similarly for CD's.

- The Surround Sound Quality

  All DVD-Video discs include multiple channel sound, consisting of up to six channels of surround sound.

- Multiple Language Dialogues and Sound tracks

  Many DVD-Video movies contain multiple language options, each with its own dialogue. With up to eight languages or sound tracks.

- Random Access To Scenes

  Movies on DVD-Video are organized into chapters, similar to how songs are on tracks of an audio CD. It is easy to find favorite scenes.

- HD DVD (High Density Digital Versatile Disk) It is a triple-layer disc with 45 GB of storage and less expensive to manufacture then the Blu-ray Disk.

- Durable Disc Format

  DVD-Video is a great format to collect movies and other video titles. With proper handling it should last long time since there is no contact between the laser pickup and the DVD disc.

22.3 References

[1] ANTONINI, J. P. Dictionary: Sampling rate. Web page, Sept. 18, 2005. http://www.audiovideo101.com/dictionary/sampling-rate.asp, date visited given.

[2] BERKELEY MULTIMEDIA RESEARCH CENTER. Berkeley MPEG tools. Web page, Apr. 2005. http://bmrc.berkeley.edu/frame/research/mpeg/.

[3] GOLDWASSER, S. M. Tutorial on lasers. Web pages, Aug. 26, 2002. http://repairfaq.ece.drexel.edu/sam/CORD/leot/.

[4] PIONEER CORPORATION. DVD technical guide - chapter 2 physical format of read-only discs. Web page, Sept. 18, 2005. http://www.pioneer.co.jp/crdl/tech/dvd/2-e.html, date visited given.

[5] PRINCETEL, INC. Tutorial: Diode laser pigtailing. Web page, 2004. http://www.princetel.com/tutorial_pigtailing.asp.

[6] WIKIPEDIA. DVD. Web page, Sept. 19, 2005. http://en.wikipedia.org/wiki/Dvd.

[7] WVONG, R. Video CD FAQ. Web page, Oct. 28, 2004. http://www.geocities.com/Athens/Forum/2496/vcdfaq.html.

23. SDRAM

23.1 Introduction

A lot of processors are able to handle a lot of tasks solely dependent on the limited space their registers supply. But for more advanced and demanding tasks, more memory space is required. Over the years many different memory types have been used: Read Only Memory (**ROM**), Dynamic random access memory (**DRAM**, Extended data-out dynamic random access memory (**EDO-RAM**), Synchronous dynamic random access memory (**SDRAM**). And nowadays more modern variants of memory, such as: Double data rate synchronous dynamic RAM (**DDR-SDRAM**) [4]. The main focus of this report is SDRAM, but since all these memory types are so close related, the report will also bring forth more general facts about how memory are working.

This report is divided in a few subsections: **DRAM** Section 23.2 where we get a quick look at some interesting things about dynamic random access memory, we will also learn more about how the computer is using its memory. **SDRAM** Section 23.3 on the next page, a few differences between DRAM and SDRAM and some advantages with SDRAM is the main topic. In the **error detection/error correction** Section 23.4 on page 103, errors in memory is discussed; how to detect them and how to correct them. The **PC66/100/133** Section 23.5 on page 103 brings forth a little more information about what these figures stand for. **DDR** Section 23.6 on page 104 tells the tale of the most commonly used memory today, double data rate SDRAM.

23.2 DRAM

In Figure 23.1 on the next page you can see the big rectangular array of memory cells that add up to DRAM. The width of the array – the lines – are usually called *bitlines* or *wordlines*. Each bit have a unique address/position depending on the line and column it is located at [3]. DRAM, differs from SRAM in that it requires constant refreshing of the info in its memory cells [4]. This is because each and everyone of the memory cells use a capacitor to hold the information. To keep the capacitors loaded, a counter is going over the entire memory, over and over again constantly (this work is about 1 % to 2 % of all the work the memory does [2]) and read each bitline and write it down to the memory again. To manipulate the memory in some way, individual bits can not be changed directly. The entire bitline is read, then the bit is changed, and only then the entire bitline (with the change) is written back again [5]. The major difference between DRAM and SRAM is not, as one might think, the speed, but the price! Each SRAM cell requires 6 transistors, while a DRAM cell requires only one transistor and one capacitor, and since a capacitor is significantly smaller than 5 transistors, the cells requires less space, therefore DRAM is cheaper [5, 2].

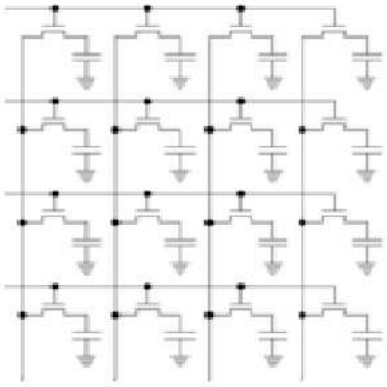


Figure 23.1: DRAM, bitlines in x-axis, read/write instructions y-axis (from [5])

23.3 SDRAM

SDRAM is not that different from DRAM, the difference is that DRAM is asynchronous and SDRAM is not. DRAM responds to any instruction immediately, and while that may sound like a good thing, SDRAM's approach to be synchronized with the clock pulse does have its advantages. The clock fuels a finite state machine which is able to pipeline incoming commands. Pipelining means that the memory chip can accept new commands before recent ones are done. And then you get a time for when the command will be executed, which makes it possible for the processor to continue with other work in the meanwhile [5]. Since SDRAM have an on-board counter, which allows the column part of the address to be increased very rapidly, big chunks of sequential bits can be fetched in very fast bursts, and supplied to the CPU as fast as it can take them [3].

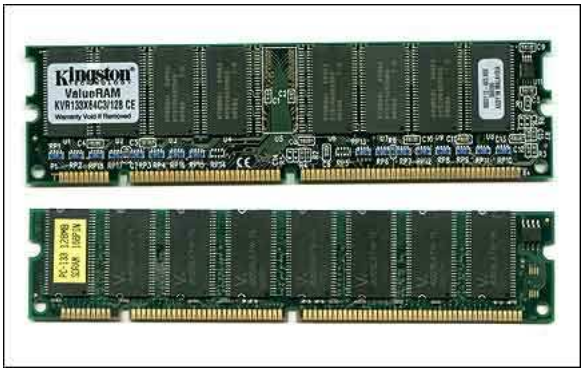


Figure 23.2: Two SDRAM chips (from [1])

23.4 Error Detection/Error Correction

23.4.1 Error Detection

In large quantity of memory, such as SDRAM, and due to the fragile state of the bits, there is a potential chance of corruption in the data. Most (if not all) computer systems check for errors in some way. A very naive and simple (fast, but unsafe) way to detect errors is the use of a "parity code". The value of the parity code depend on the number of 1s in a word, if the number of 1s is odd, the parity code is 1, otherwise it is 0 [2]. This way of error detection have its obvious pros and cons. Fast and simple, the disadvantages are a little more complicated. For one thing, the parity code will indicate that there is no error at all if there is a even number of errors. For example 1101 have the parity code 1, and the corrupted version of 1101, 1011 have two corrupted bits (second and third) but the parity code will still be 1. A little side-note is that there can also occur corruption in the parity code, making it blind to errors. But perhaps the most eminent disadvantage of parity code is that it does not say anything about where the error occurred or how it can be corrected.

23.4.2 Error Correction

Many systems work in such ways that they use extra bits to encode the data, typically such systems that are able to detect and correct errors use 7 or 8 bits for every 128 bits of data for error handling purposes.

> "A 1-bit parity code is a distance-2 code, which means that if we look at the data plus the parity bit, no 1-bit change is sufficient to generate another legal combination of the data plus parity... there is a distance of two between legal combinations of parity and data. To detect more than one error or correct an error, we need a distance-3 code." [2, page B-66]

With this information, its clear that with a distance-3 code, one are able to detect even 2-bit errors, but not correct them. But also correct 1-bit errors. This is because if the mechanism detect an error, it can check all 1-bit inverts for another working combination of bits, if it finds a working one, that will be the one intended in the first place. If there is no one, there have been more than one bit-corruption.

23.5 PC66/100/133

Basically, **PC100** means that the memory chip generally works with a clock cycle of about 10 ns. PC100 also says that the chip works stably at 100 MHz. There were many different PC100 chips made by many different companies. They were not all equal, the only requirement of a PC100 chip is that it should be able to work at 100 MHz [3].

> "The latency is a measurement of how long it takes for other hardware to return data to the RAM. The lower the latency rating, the better the chip, and the faster it will operate." [3]

PC133 is basically just another implementation of the same SDRAM, with the difference that PC133 have lower latency, and therefore runs on a faster bus [3].

23.6 DDR

DDR is a later version of SDRAM that was introduced to the average citizen around 2000, it have been the main memory used ever since (written 2005). While all types of SDRAM use a clock signal with a square wave (meaning that the signal is either high or low) to function, DDR-SDRAM differs from SDRAM in that it acts on both low-to-high transition and high-to-low transition. Therefore DDR gets twice as much work done in equal amount of time [5].

23.7 References

[1] GOOGLE. SDRAM. Web page, Sept. 18, 2005. http://fara.cs.uni-potsdam.de/~klihow/pics/sdram.jpg, picture search on google, third hit.

[2] PATTERSON, D. A., AND HENNESSY, J. L. *Computer Organization & Design: the Hardware/Software Interface*, second ed. Morgan Kaufmann Publishers, Inc., San Francisco, California, 1998.

[3] PC TECHNOLOGY GUIDE. DRAM. Web page, Sept. 18, 2005. http://www.pctechguide.com/03memory_DRAM.htm.

[4] TYSON, J., AND COUSTAN, D. How RAM works. Web page, Sept. 18, 2005. http://computer.howstuffworks.com/ram4.htm, date visited given.

[5] WIKIPEDIA. SDRAM. Web page, Sept. 18, 2005. http://en.wikipedia.org/wiki/SDRAM.

Part III

Busses and Protocols

24. Bluetooth

24.1 Introduction

Since the dawn of the computer age in the early 70s, cables have been the primary and only way of transferring data between devices such as mices, mobile phones, keyboards and printers. In the end of the 90s wireless solutions started to become more popular and the demand for a workspace without leashes grew. While companies that produced external devices and networking companies alike based their wireless products on ordinary radio wave technology, the Swedish company Ericsson did not. What Ericsson did was developing a new communication technology called Bluetooth, named after a renowned Danish king who united parts of Norway and Denmark during the late 900s. The name was chosen more as an indication of the role of the Nordic countries in the growing communication industry than as an explanation of the technology itself [3, 6].

The difference between Bluetooth and ordinary wireless technology is most apparent in terms of reliability, dynamics and automation. The technique itself was a minor breakthrough in wireless communication but what really made Bluetooth commonly accepted was the founding of the Bluetooth Special Interest Group, the Bluetooth SIG. In addition to Ericsson, big companies like IBM, Intel, Nokia and Toshiba where all a part in founding the SIG in the end of 1998. The list of companies involved has drastically increased since then and today over 2000 companies worldwide are attached to the group, developing and supporting the future of Bluetooth.

24.2 The Technique

Bluetooth is a standard used in many devices today. It can be found in almost any type of electronic equipment and makes it possible for the devices to communicate without any wires. The standard can be split into two categories:

1. **The specification**
   This section describes how the technique works and how the protocol architecture looks like. The information is gather from three different websites, namely Techworld [4], Bluetooth.org [1] and Palowireless [2].
2. **The profiles**
   The profiles define how the technique is used.

24.2.1 The Specification

The Bluetooth specification contains a protocol stack that describes the different parts that are necessary for this technique. The different components in this stack are described below.

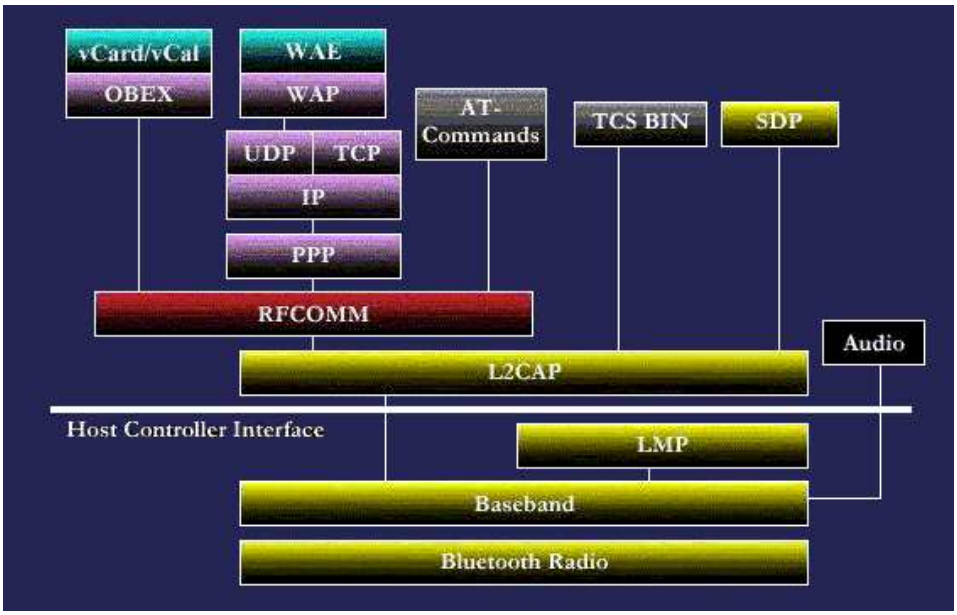


Figure 24.1: The Bluetooth layer architecture (from [2])

Radio

The radio layer is located at the bottom of the stack and defines the requirements of the transceiver. Bluetooth works at a frequency of 2.45 gigahertz, which is the same as for example WLAN. A problem that can occur when several devices uses the same frequency is that they interfere with each other, which makes it difficult for the devices to communicate. To solve this problem the Bluetooth standard takes on two big measures to avoid interference.

1. Devices that use the Bluetooth standard are intended to be used for communication within personal networks consisting of devices with short range requirement. Obvious examples are between the mobile phone and a PDA or between a computer and a printer. Because of the short range, the transmitter can operate on a low energy level. The short range networks are unlikely to interfere with other communications.
   The Bluetooth devices are categorized into three different classes. These classes differ in the output signal strenght and thereby also the range. The third class has a range of about 1 meter, devices from the second class are capable of sending up to 10 meters and those of the first class up to about 100 meters. Although weak signals often are transmitted, they can penetrate the walls of a house and make it possible to communicate with devices being placed in another room. Naturally the power consumption grows according to the strenght of the signal being transmitted. The power transmitted from a Bluetooth device range from 1 mW up to 100 mW. Compared to a mobile phone's maximum capacity of 3 W

this is obviously very low.

2. One way to avoid interference is to use frequency hopping. This technique makes it unlikely that several devices use the same frequency and has been implemented in the Bluetooth standard.
   A Bluetooth device has a certain range of frequencies that can be used while communicating. This range differs depending on where in the world you are and in Europe this range is between 2,400 and 2,483.5 MHz. The technique uses 79 different frequencies within this range that are randomly and individually chosen and are changing on a regular basis. The transmitter alternate the frequency 1600 times every second and this way the risk of two or more transmitters disrupting each other is greatly reduced. Frequency hopping will also result in a more effective use of the radio spectrum.
   The devices can limit their output power to the distance needed for the current communication. This is done by measuring the RSSI (Receiver Signal Strength Indicator) and vary the output power to match the minimum required signal strength, keeping the power consumption at a minimum.

Baseband

In the Bluetooth layer architecture, baseband is the physical layer and handles for example the physical channels and links, data direction and some security features.

When Bluetooth devices come close enough to communicate with each other, a special connection phase will be initialized. This conversation will determine if one unit needs to control the other and if the two devices are to share information. Once the connection has been established the units form a small network, called PAN (Personal Area Network) or piconet. Each piconet contains one master and at least one slave and also has a unique hopping sequence. To be able to allow several piconets in a small area, the units that are connected to each other in a piconet randomly hop frequencies in unison. Every Bluetooth device have a unique Bluetooth address and by using this address the master determines the hopping sequence of the whole piconet.

The Bluetooth technique is based on TDD (Time Division Duplex) which means that the channel is divided into time slots with each timeslot being 625 micro seconds long. The point of using TDD is that each device can transmit at full power during a small period of time, minimizing the risk of two units disrupting each other.

There are two different ways for a Bluetooth device to send data:

1. **SCO**
   The first is SCO (Synchronous Connection-Oriented) which is point to point link between a master and a slave. The link uses reserved slots and is therefore Connection-Oriented. The data is sent regularly in an even flow and is therefore intended to be used to transfer voice and sound. The maximum transfer rate is 64 kbits/s (v1.1) and does not support retransmissions.
2. **ACL**
   The second way to send data is using a ACL (Asynchronous Connectionless Link). This link is a point to multipoint link between the master and all the slaves in a piconet. With this link the sender also sends data in packets and is

capable of reaching the Bluetooth maximum transfer rate of 721 kbit/s in one direction and 57.6 kbits/s in the other.

A Bluetooth controller can work in two different modes, standby and connection. The default mode is standby and this causes the controller to use a minimal amount of power. The only thing running is the native clock and no communication between the units is taking place. While a device is in the connection state the master and the slave can exchange data within the piconet.

The connection state can also be divided into a few different modes and the devices will automatically change mode depending on the traffic.

Since several piconets can be running at the same time and place, any single device might be a member of more than one piconet. This phenomena is called Scatternet and means that several piconets are somehow connected to each other. One unit can be a slave in several piconets at the same time, but can only be the master of a single piconet at a time. It is also possible for two units to swap roles and this is called a master-slave switch.

The baseband layer also contains features for error correction and flow control.

Link Manager Protocol

The Link Manager Protocol main task is to handle the setup and configuration of the link and the authentication. To be able to do this the Bluetooth devises exchange PDUs (Protocol Data Units) containing network information. When a package is received the information is handled at this layer and the data is not propagated to the layer above. The protocol consists of a number of different PDUs and takes care of the following things:

1. Link Setup
2. Link Configuration
3. Security
4. Low Power Modes
5. Information Commands

Host Controller Interface

HCI is a part of the Bluetooth standard and serves as a command interface to the baseband and the link manager. It also makes it possible to access the hardware and the control registers.

This interface is used in three different parts of the Bluetooth device, namely The Host, the Transport Layer and the Host Controller, whose functionality also can be classified into three parts:

- **HCI Firmware**

  This part is found in the host controller which is the same as the Bluetooth hardware device. It handles the HCI commands for the hardware. This is done by using the baseband commands.

- **HCI Driver**

  The HCI driver is located in the host, which is a software unit. When an HCI event occurs the host will be notified. The host will then examine the package and discover which event that occurred.

- **Host Controller Transport Layer**

  The reason the host controller transport layer exists is because the HCI firmware and the HCI driver needs to send data between each other. This is done via this layer which can be, for example, USB.

To be able to handle the Bluetooth hardware this layer also provides a number of commands which makes it possible for the host to control the link layer connections to other Bluetooth units.

Logical Link Control and Adaptation Protocol

Logical Link Control and Adaptation Protocol is a layer above the HCI which provides services to the upper protocols in the host. Here the different protocols are identified and multiplexed up into the layer architecture. Packages sent into a piconet have a small MTU (Maximum Transfer Unit) compared to other networks and this results in a lot of multiplexing and demultiplexing operations being carried out.

RFCOMM protocol

This protocol is placed just above the Logical Link Control and Adaptation Protocol and serves as an abstraction between the layers above and the L2CAP. The protocol makes it unnecessary for the upper layers to adjust to the fact that the data is being transferred using the Bluetooth technique. Examples of layers above are PPP, IP and TCP.

Service Discovery Protocol

This protocol lies above the L2CAP and is used to discover which services are available from the Bluetooth host. Because of the fact that Bluetooth is a wireless technique its services can change during a session.

24.2.2 Profiles

Since Bluetooth can be used for many different tasks it needs something that can describe how the implementation for these can be done. In Bluetooth this is accomplished with different profiles. A profile defines options in each protocol in the layer architecture and can therefore be seen as a vertical slice of the protocol stack. The possibility to use profiles has been developed to facilitate for different manufactures and prevent problems between different products.

24.3 Security

Because the piconet structure of Bluetooth is based on regular ad hoc network techniques, it also suffers from the same weaknesses as they do. The term “ad hoc” refers to a technique where the network is formed on the fly with any of the units serving as master. Inside any network which lacks a fixed topology and a fixed infrastructure, security issues and vulnerability to attacks will be present. The information is based on [7] unless otherwise noted.

24.3.1 Availability

One of the weakest links in ad hoc networking security is the availability. Since the information is transmitted on the air and the fact that all devices are dependent on each other to relay the messages, denial of service attacks are easy to perform by jamming or otherwise interfering with the flow of information in the air.

Another factor in availability is the possible disruption of the routing algorithm. By feeding the network inaccurate information the whole routing-table could easily cause all units in the network to route through our malicious and non-relaying device.

A weakness that is unique to mobile networks is the energy of the devices. The Bluetooth technology supports different levels of power management and by sending inaccurate information in the piconet, all other units can be tricked into using the highest energy consumption level and thus cause the battery of the device to power out prematurely.

24.3.2 Authorization and Confidentiality

The authorization process in Bluetooth is based on symmetric keys and challenge-response strategy making sure the devices share the same secret key. The key generation process is briefly described below.

The nature of ad hoc networking distorts the boundaries between authorization and confidentiality. If the authentication process is weak there is no use for confidentiality as the destination of the confidential information could be malicious. Similarly, despite the level of authentication a failure in confidentiality will still compromise the security of the encrypted data as everyone has access to the ether.

Integrity can be discussed in a similar manner with the additional problem of radio interference.

There is also a flaw in the key handling system that threatens the authentication, confidentiality and integrity, described in the next section.

24.3.3 Key Handling

Bluetooth uses a intricate system of keys and the details of key handling and generation is beyond the scope of this article. Briefly described, it’s a system with every key being more or less derived from the only two static parts; the PIN code and the address of each device, the Bluetooth Device Address.

The PIN is entered by the user on both devices trying to establish a link. This means connecting more then a pair of units into a piconet is a time consuming task. Because of this, the complexity of the PIN used is often compromised.

An easy PIN code coupled with a fixed address for each device that can be obtained easily makes the key generation process based on weak premises. It has been shown that a 4 digit PIN can be cracked in 0.06 seconds using a normal 3 GHz computer [5].

As the units authenticate and later on communicate they use encryption based on device addresses and a Link Key. This key is chosen from the Unit Key of any participating device. The key is generated in each device as they are used for the first time. This key is rarely regenerated which compromises the security. If two units, A and B, communicate using A's unit key as the link key, any unit previously connected to A will know the unit key. By simply faking a device address the encryption key can be calculated and the confidentiality of the link is broken.

This is clearly also a breach of integrity as our malicious device can authenticate to either part of the link as being the other.

A possible solution could be making use of the popular Key Distribution Center infrastructure. This, however, is not supported in Bluetooth and has to be implemented at the application level.

24.3.4 Privacy

Since every Bluetooth unit has its own unique address, simply monitoring transmissions makes it possible to map and log the behaviour of the user carrying the device. This makes profiling a big issue which threatens the privacy.

24.4 Discussion

Our impressions of Bluetooth after writing this paper is that the potential of this technology is great. Cheap devices coupled with low energy consumption and complete interoperability ensures that Bluetooth is here to stay.

We feel that the piconet structure, however, suffers too many disadvantages to be used to form temporary networks containing several devices. Bluetooth is best suited for communication between two devices and preferably two that has a more or less permanent connection, like a mouse and a receiver.

In terms of security, the conclusion to be drawn has to be that bluetooth is not secure enough for things like money transfers and other sensitive information. The security provided should probably be sufficient for everyday use as the effort of breaching it probably exceeds the gain. The biggest weaknesses as we see it is the jamming, the routing protocol, the PIN code and the key management process.

Jamming of wireless communication and the faulty use of the PIN system are probably unavoidable, but both the routing protocol and the key management process can hopefully be adjusted to cope with anything but the most dedicated attackers.

We predict that Bluetooth in the future is more secure, has a longer range and will feature piconets that are easier to create and maintain. This could lead to Bluetooth becoming a serious challenger to other wireless techniques.

24.5 References

[1] BLUETOOTH SIG. The official Bluetooth membership site. Web site, Sept. 21, 2005. http://www.bluetooth.org, date visited given.

[2] DAID, C. M. Bluetooth tutorial - specifications. Web page, Sept. 21, 2005. http://www.palowireless.com/infotooth/tutorial.asp, date visited given.

[3] FRANKLIN, C. How Bluetooth works. Web page, Sept. 21, 2005. http://electronics.howstuffworks.com/bluetooth.htm, date visited given.

[4] MCKEAG, L. The nuts and bolts of Bluetooth. Web page, June 25, 2004. http://www.techworld.com/mobility/features/index.cfm?featureid=678.

[5] SHAKED, Y., AND WOOL, A. Cracking the Bluetooth PIN. Web page, May 2, 2005. http://www.eng.tau.ac.il/~yash/shaked-wool-mobisys05/.

[6] SUSNING. Bluetooth. Wiki page, Feb. 15, 2004. http://susning.nu/Bluetooth.

[7] VAINIO, J. T. Bluetooth security. Web page, May 25, 2000. http://www.niksula.cs.hut.fi/~jiitv/bluesec.html.

25. FireWire - IEEE 1394

25.1 Introduction

FireWire, a hardware and software standard for transporting data, is described in this paper. The questions what FireWire is, how it works, what areas it is working in and the difference between it and USB will hopefully be answered here.

FireWire, or its original name IEEE 1394, is a protocol for high-speed communications between peers. There are three existing protocols 1394, 1394a and 1394b, and one on the way 1394c.

25.2 Origins of Firewire

The cross-platform implementation of the high-speed data bus that can move large amounts of data between computers and peripheral devices, FireWire was originally developed by Apple Computer Inc. in 1986. The goal with the development was to find a cheap bus which could transfer digital sound. The existing solution at Apple could at this time not handle this. But to make the market more interested Apple realized the bus had to be able to handle more than just transferring digital sound fast. So the goal with the development became a bus that was very fast, easy to use and cheap to implement at the same time as it should be able to replace most other port in a PC (Personal Computer).

The first specification for this link was completed in 1987 and in 1990 a prototype of the FireWire-bus was ready. In this prototype the signals was sent over optic fiber a solution that was replaced by a copper cable to make the bus cheaper. At this point a group consisting of representatives from Apple, Texas Instruments, Stewart Connector, Molex, Adaptec and Western Digital within IEEE was founded to start the standardization of FireWire. In 1992 IBM affiliated to the group and the work of the group increased. The real interest for FireWire arrived when Apple demonstrated it at a show in Las Vegas in 1993. A year later in 1994 the “1394 Trade Association” was founded, an organization to facilitate the development of FireWire products [1].

In 1995 the Institute of Electrical and Electronic Engineers adopted FireWire as an industry standard. The FireWire technology was at that point named “IEEE 1394-1995 Standard for a High Performance Serial Bus”. PC developers as NEC, Compaq and Sony was slower but did as well implement the IEEE 1394 technology in their computers. But the name FireWire is Apples name for the technology and there are other names for the technology as well, for example Sony’s i.Link [4].

A new version of IEEE 1394, IEEE 1394a was developed in 2000. It includes specifications for 100, 200 and 400 Mbps transfer speed, cable power, hot plugging, plug-and-play and isochronous data mediation. IEEE 1394a supports a cable length up to 60 meters. The third version of IEEE 1394, IEEE 1394b came to the market in early 2003 and supports a cable length up to 100 meters and a speed of 800, 1 600 and 3 200 Mbps. Now days Apple Computer Inc. has agreed for others to use the name FireWire, so this name is now widely spread as the name of the technology [7].

25.3 How it Works - FireWire's Structure

So what exactly is FireWire and how does it work? Some features with FireWire are that it is hot-pluggable[1], plug-and-plug[2] and it also provide peer-to-peer[3] connections [4]. Because the protocol is peer-to-peer it allows the topology to be either a daisy chain, tree, star (see Figure 25.1) or a combination of them [3].

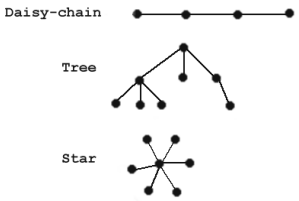


Figure 25.1: FireWire's different topologies

25.3.1 Isochronous and Asynchronous

FireWire has the advantage over other I/O connections that it supports both isochronous and asynchronous transfers. The asynchronous part takes longer and involves more steps than asynchronous since it demands that acknowledgment is sent and received before more packages can be sent. An other disadvantage is that, in asynchronous mode, if an other devise requests space on the bus, the data stream already sending will be interrupted. This is a great disadvantage when sending streaming media. Asynchronous transfer is on the other hand efficient when important packages are sent. For example receivers ID and audio volume. Other packages, like raw video or audio packages, are more suitable to send with isochronous transfer. In this case it is not important that every package gets through. It is more important that the packages gets to the other application in time. In this case the streaming media is guaranteed a consistent bandwidth [4]. Although 20 % of the bandwidth is always reserved for asynchronous transfer [1].

25.3.2 The Cable

The cable that FireWire use consists of two power conductors and two twisted pairs for data signaling. The two twisted pairs are shielded as well as the entire cable (see Figure 25.2 on the next page). Because FireWire offers power in the connection there is no need to take power from somewhere else. This is a great convenience for the user and makes peer-to-peer connections easier.

[1] You can add them to your PC without rebooting.
[2] No need for additional drivers.
[3] No need for a PC to monitor the transfer.

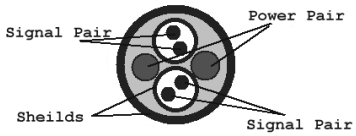


Figure 25.2: The FireWire cable (originally from [5])

25.3.3 The Layers

There are three protocol layers involved in a FireWire connection: The physical layer, the link layer and the transaction layer. To connect these three layer we also have a serial bus management process.

The physical layer provides the electrical and mechanical connection between the device and the cable. This layer also provides arbitration to make sure that all devices have fair access to the bus. The link layer takes the raw data from the physical layer and formats it into two types of packets: Isochronous or asynchronous. The transaction layer is involved only in the asynchronous transfer and takes care of the write, read and lock commands. Finally there is the serial bus management process that provides configuration control of the serial bus [5].

25.4 User Area

Areas where FireWire is used are external storage devices, digital video cameras, home theater, widescreen review, PC-board, PC-Card-Interface, High-speed Mass Storage, High-speed Printer and Scanner, DVD players, Digital Video Camera, Digital Still Camera, Set Top Box, Digital Imaging, Game console, Multimedia Home networking, etc. FireWire can for example be used to link electronic equipment to PCs, connect a VCR to a TV and share peripherals between PCs for example a hard drive. Established IP services such as AFP, HTTP, FTP, SSH and TCP/IP can all be used on FireWire to support new development [2].

25.5 Difference Between FireWire and USB

We have already discovered that FireWire is a great invention and can be used to do many things. But what about USB? Are they not just the same or is one of them better than the other? We have FireWire that runs at 800 Mbps (at the moment) and USB 2.0 that runs at 480 Mbps. Just by comparing these two rates we can see that FireWire is the winner. An other advantage is that FireWire works peer-to-peer which USB 2.0 does not. This means that with FireWire you can easily transfer music files, among many other things, directly between two peripherals without the need to connect a PC. This disadvantage have USB on the other hand recently taken care of. An advantage for USB is on the other hand that it is cheaper than FireWire and that it is more spread, meaning that USB already exists on many computers and not FireWire [6]. So which

one should you use? USB is probably the best choice for middle and low bandwidth applications, such as mice, keyboards, scanners and printers, where price is a priority. For high bandwidth applications, such as streaming media and disk drives, it might be more suitable to use FireWire [3] specially if they can achieve 3.2 Gbps as the promise will come. Hopefully all computers will have both FireWire and USB in the future so there will be no need to choose [6].

25.6 Pros and Cons with FireWire

FireWire is fast at transmitting data peer-to-peer and supports up to 63 devices at once with out any special software. It can also handle devices which needs a big amount of bandwidth and can guarantee data arrival. FireWire is at cheap bus that supports both asynchronous and isochronous transmission. FireWire is user friendly and plug-and-play, witch means the system configure automatic when a new unit is connected.

FireWire needs a fast CPU (Central Processing Unit) and is not as widely implemented as USB (Universal Serial Bus). FireWire 400 supports only a distance up to 4.5 meters between the units and the initiating time is relatively long [4].

25.7 References

[1] ANDERSSON, P. Överföring av digital video via FireWire. Master's thesis, Linköpings Tekniska Högskola, Feb. 8, 2002. LITH-ISY-EX-3182-2002, available at http://www.diva-portal.org/diva/getDocument?-urn_nbn_se_liu_diva-1009-1__fulltext.pdf.

[2] APPLE COMPUTER, INC. Web site, Sept. 19, 2005. http://www.apple.com, date visited given.

[3] DHIR, A., AND MOUSAVI, S. Home networking using new wires - IEEE 1394, USB and fast Ethernet technologies. White paper, Xilinx, Mar. 21, 2001. Available at http://direct.xilinx.com/bvdocs/whitepapers/wp134.pdf.

[4] GOWAN, M. How it works: IEEE 1394. *PC World* (Dec. 20, 1999). Also available at http://www.pcworld.com/howto/article/0,aid,14371,00.asp.

[5] HUFFMAN, G., AND MOORE, D. IEEE 1394: A ubiquitous bus. Web page, Mar. 1995. http://www.skipstone.com/compcon.html, original article presented at COMPCON '95 in San Francisco, CA on March 5–9, 1995.

[6] IKRAM, G. Firewire vs. USB: A simplified approach. Web page, Mar. 25, 2005. http://www.directron.com/firewirevsusba.html.

[7] TEENER, M. J. IEEE 1394b: Faster, longer and simpler. *Electronic News* (2001). http://www.reed-electronics.com/electronicnews/-index.asp?layout=article&articleId=CA91031.

26. I^{2}C-bus

26.1 Introduction

The I^{2}C bus, a control bus for communication between integrated circuits, was developed by Philips Semiconductors in the early 1980s. The original specification defines a bi-serial two-wire bus with a software protocol [1, 3]. Because of its simplicity, efficiency and low-cost design the bus has survived and maintained its position as the de facto standard Inter-IC-bus. Over the years the bus specification has been updated to meet the increasing demands in terms of speed and performance. It is commonly found in a broad variety of devices, such as temperature sensors, microprocessors and EEPROMs.

26.2 Overview

Devices are connected to the bus as either a master or a slave. At least one master device, usually a microprocessor, has to be present. Slave devices on the bus responds to queries from master devices. Examples of common slave devices are flash memory, LCD-displays, A/D and D/A converters. A slave can never be the initiator of a data transaction, only a responder. Several master devices may co-exist, but only one at a time can take hold of the bus. Three different transfer speeds are defined:

1. Standard-mode allowing for up to 100 kbit/sec.

2. Fast-mode allowing for up to 400 kbit/sec.

3. High-speed mode allowing for up to 3.4 Mbit/sec.

26.3 I^{2}C-bus Protocol

The bus consists of two wires that are labeled SDA (Serial Data Line) and SCL (Serial Clock Line). The two wires carry information between devices connected to the bus. Each device has a unique ID which is used for addressing specific devices on the bus. Addressing is done using seven bits. However, only the three least significant bits are programmable and the other bits are fixed depending of type of device and manufacturer. Furthermore, combinations starting with 1111 and 0000 are reserved, reducing the total number of unique addresses to 112. The 0000XXX address group is used for broadcasting on the bus and the 1111XXX group is reserved for future purposes. The latest specification supports 10-bit addressing, extending the original specification by exploiting the 1111XXX combination. This would theoretically allow for up to 1024 devices to be addressed. Electrical limitations must also be considered since it limits the number of possible connected devices to about 20-30. The broadcast address group

(0000XXXX) is used for hardware master devices, such as keyboards and other arbitrary devices lacking the ability of addressing slaves. The 0000XXX group is also used to write the programmable part of slave addresses.

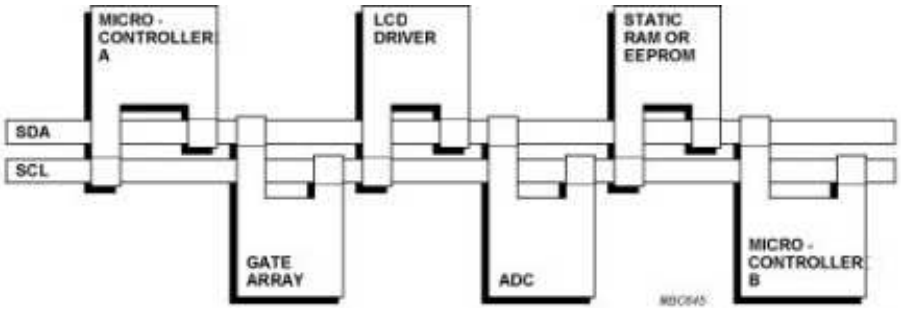


Figure 26.1: Example of devices connected to the two wires on the bus (from [4])

The master device initiates data transfers and is responsible for generating the clock signals on the SCL-line. The master can start a data transaction only if the bus is free. The bus is said to be free when both lines are high. During every clock pulse on the SCL-line one data bit is transferred. The signal on the SDL-line must remain stable during the high phase of the pulse.

Arbitration is used to ensure that only one master has control over the bus. If a master detects a high signal on the SDA-line while transmitting a low signal the data transaction will be cancelled since this implies that another master is using the bus. The loosing master must then turn into slave-mode at once in case that it is the device being addressed. To avoid race conditions a number of precautions has to be taken [1].

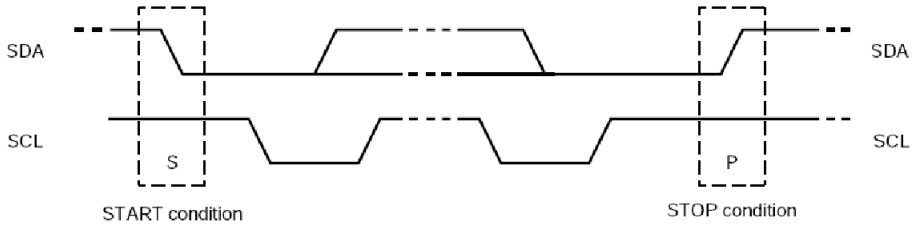


Figure 26.2: START and STOP definition (from [2])

Unique START and STOP conditions are defined as shown in Figure 26.2. The START condition is indicated by a high to low transition on SDA while SCL is high. A low to high on SDA while SCL is high indicates a STOP condition. When a successful START condition has been accomplished by a master the address of the requested slave device is broadcasted along with a single bit labeled R/W which indicates whether to data is to be read or written. The master then releases the SDA-line, waiting for the slave device to acknowledge by pulling the SDA-line to low. After the address has been acknowledged, data is transfered 8 bits at a time, each byte followed by an

acknowledge by the receiver. The last data byte in the transaction is marked by not sending an acknowledge. The master must then transfer the STOP condition and let go of the bus. The master also has the option to send a repeated START condition instead and maintain control over the bus. Figure 26.3 shows an example of two data transfers.

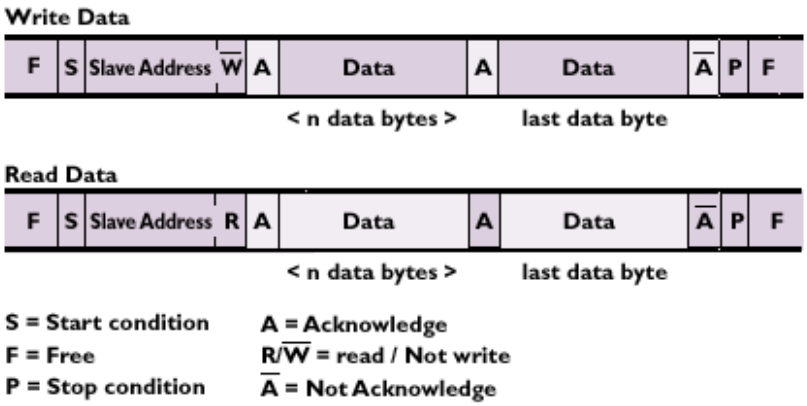


Figure 26.3: Example of data transactions (from [4])

In order for receiver devices to cope with high transfer rates, they have an option to hold the SCL low while processing the data received and thereby forcing the master into a wait-state. When the receiver is done processing the data it must release the SCL allowing the master to send more data. A master with limited performance can force the whole bus into its own pace by extending the low clock period.

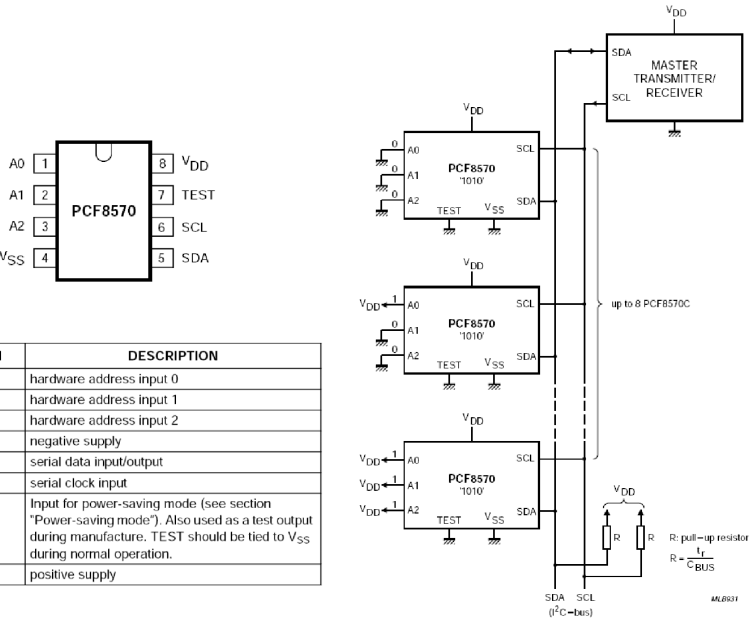
These are roughly the criteria that has to be met for a device to be I^2C compliant. Although, there are legal issues and manufacturers also need to acquire a license from Philips.

The device protocol layer is totally device specific and manufacturers have to provide their own protocol embedded in the data portion of the I^2C protocol.

26.4 Example Application

The Philips PCF8570 is a general purpose 256 $\times$ 8-bit static low-voltage RAM and can serve as RAM expansion for micro-controllers etc [2]. An example setup involving three of these can be seen in Figure 26.4 on the following page.

In this particular setup, each of the three devices is given a unique address by hard-wiring pins A0-A2 on the ICs. The other four bits of the address is fixed to 1010 for this type of device resulting in three unique addresses on the bus (1010000, 1010001 and 1010010). The device layer protocol used for the PCF8570 is almost trivial. After a successful START condition followed by addressing of the slave the master transmits the memory address it wishes to access. The slave device then sets its internal memory



SYMBOL	PIN	DESCRIPTION
A0	1	hardware address input 0
A1	2	hardware address input 1
A2	3	hardware address input 2
V_{SS}	4	negative supply
SDA	5	serial data input/output
SCL	6	serial clock input
TEST	7	Input for power-saving mode (see section "Power-saving mode"). Also used as a test output during manufacture. TEST should be tied to V_{SS} during normal operation.
V_{DD}	8	positive supply

Figure 26.4: Example application (from [2])

word address accordingly and acknowledges. Depending on whether the R/W bit was set or not, the slave sends or receives data bytes while incrementing the memory address after each acknowledged byte. This process continues until the STOP condition is signaled by the master.

26.5 References

[1] PHILIPS SEMICONDUCTORS. *I²C Peripherals.* Philips Semiconductors, 1995.

[2] PHILIPS SEMICONDUCTORS. *PCF8570 - 256 × 8-bit static low-voltage RAM with I²C-bus interface*, 1999. Also available at `http://www.semiconductors.philips.com/acrobat/datasheets/PCF8570_4.pdf`.

[3] PHILIPS SEMICONDUCTORS. *The I²C-Bus Specification*, 2001. Also available at `http://www.semiconductors.philips.com/acrobat/literature/9398/-39340011.pdf`.

[4] PHILIPS SEMICONDUCTORS. About the I²C-bus - Facts and figures. Web page, Sept. 16, 2005. `http://www.semiconductors.philips.com/markets/mms/-protocols/i2c/facts`, date visited given.

27. PCI Express

27.1 Introduction

The PCI Express bus is the successor of the popular local bus PCI (Peripheral Component Interconnect). Like its predecessor it is designed by PCI-SIG (PCI Special Interest Group). With its great performance it has been accepted as the next standard for high performance expansion cards such as graphics cards. It uses the existing PCI programming concepts but is based on a much faster serial communications system. That means that the PCI Express is not a bus in the traditional meaning [3, 8].

PCI Express is backwards compatible with PCI in software but not in hardware. This means that an operating system written for a PCI-based platform will work without any changes whatsoever on a PCI Express-based system. PCI Express will redefine how the internal layout of the computer is set up [5, 1]. Today, the north bridge is working with the memory/AGP and the south bridge is working with I/O. The communication is mainly performed through shared buses as can be seen in Figure 27.1.

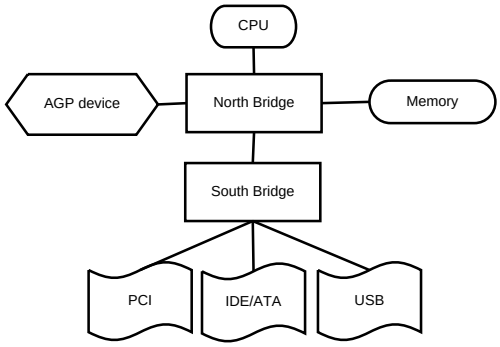


Figure 27.1: Internal layout of a regular system

With PCI Express there will be a single bridge, called the Host Bridge. With the memory controller moving in to the CPU the Host Bridge is taking over for both the north and the south bridge. This is illustrated in Figure 27.2 on the following page. Because the communication is serial the devices use point-to-point interconnections. That means that two devices communicating with each other has a dedicated channel that only those devices can use. For the transition time you have a PCI Express-to-PCI bridge and a PCI Express-to-PCI-X bridge that is connected to the Host Bridge so that older PCI and PCI-X devices still can be used [4]. The figures 27.1 and 27.2 on the following page does not represent the way it looks in all systems but they show the principles in a simple manner.

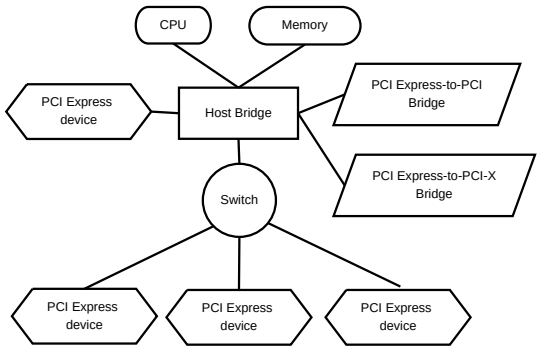


Figure 27.2: Internal layout of a PCI Express system

27.2 Usage

Because of the high performance and scalability of PCI Express it is capable of replacing PCI, AGP, PCI-X and PCMCIA. The idea is that all devices will use PCI Express, anything from Hot-Plug devices to high performing graphics cards [3, 8].

27.3 Hardware

The big difference between PCI Express and PCI is that PCI is a parallel bus while PCI Express is a serial bus. PCI has 32 parallel data pins which means that it transfers a whole 32-bit word at once. A serial connection is much cheaper to construct and the signal traces on the PCB (Printed Circuit Board) is easier to route since it is much narrower and is less susceptible to interference. A PCI Express connection is called a link and uses one or several serial bidirectional lanes each consisting of two pairs of wires. A PCI Express link can aggregate x1, x2, x4, x8, x16 or x32 lanes [2].

As mentioned in the introduction the PCI Express bus is actually not a bus in the common sense of the word since it does not connect multiple devices on the same physical media. PCI Express works as a switched point-to-point network, with switches that route command and data packets to the connected devices [3].

The signals are transferred over the lanes using LVDS signaling. LVDS (Low Voltage Differential Signaling) uses two wires to transfer a signal [7]. Regular signaling uses a voltage near 0V to indicate a zero and a voltage usually around +5V to indicate a one. LVDS transmits two similar voltages over the two wires and looks at the difference, gaining several benefits. Since noise usually affects both wires similarly the net effect of the noise is close to zero. It is cheaper and faster to adjust a current a little bit than to fully change it from 0 V to +5 V.

27.4 Protocols and Signaling

The PCI Express protocol encodes the data stream using 8b/10b encoding [6]. 8b/10b encodes 8 data bits into 10 bits which is then sent over the lane. The benefit of throwing 20 % of the bandwidth away on the encoding is that the 8b/10b encoding breaks up long runs of identical bits which might otherwise make the receiver lose synchronization on the data stream. The clock is embedded within the data stream to avoid a separate clock wire. As mentioned before, PCI Express is a switched network of point-to-point connections. When a device is plugged into a slot the newly connected device and the device or switch on the other end of the link negotiate how many lanes are available and throttle their transfer rates accordingly. The PCI Express bus is implementing the lower layers of the OSI[1] network layering standard (Figure 27.3), thus isolating the higher levels of functionality from the lower levels [1]. That way it is possible to enhance or exchange the physical level without any changes in the upper levels. This also allows software written in compliance with the PCI standard to work perfectly since the PCI Express stack has a regular PCI interface on top.

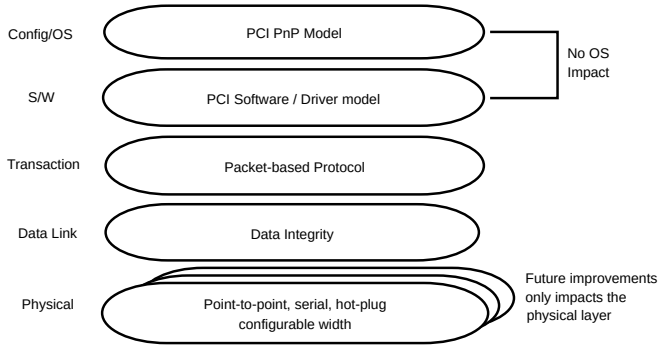


Figure 27.3: The OSI stack as implemented by PCI Express

27.5 Performance

PCI Express transfers data at a rate of 5 Gbps full duplex per lane, for a maximum of 160 Gbps on a x32 link [2]. A comparison between some of the popular buses can be seen in Figure 27.4 on the following page. In the future PCI-SIG predicts that they will be able to increase the signaling speed of a lane with up to four times the speed, approaching the theoretical maximum of copper. Since the bus is switched the switches have the capability of implementing QoS (Quality of Service). That way it is possible to prioritize certain kinds of traffic such as traffic to the video card for better gaming performance or prioritizing a capture stream from an audio card for lower latencies and skip protection [5, 4].

[1] Described at `http://en.wikipedia.org/wiki/OSI_model`.

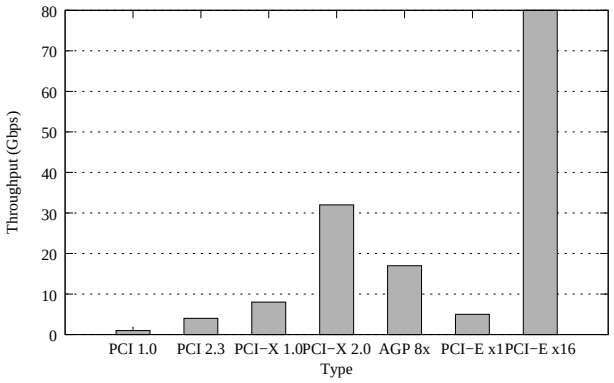


Figure 27.4: Performance of some popular buses

27.6 References

[1] BHATT, A. V. Creating a PCI Express^{TM} Interconnect. White paper, Technology and Research Labs, Intel Corporation, 2002. Available at http://www.pcisig.com/specifications/pciexpress/technical_library/-pciexpress_whitepaper.pdf.

[2] IBM. Introduction to PCI Express. Web page, Sept. 12, 2005. http://www.redbooks.ibm.com/Redbooks.nsf/0/dcd597d08215466385256f34006-f9705?OpenDocument, date visited given.

[3] LIMA, C. PCI Express Bus Tutorial. Web page, Sept. 6, 2005. http://www.hardwaresecrets.com/article/190.

[4] MICROSOFT CORPORATION. PCI Express and Windows. Web page, Oct. 22, 2004. http://www.microsoft.com/whdc/system/bus/PCI/PCIe_Windows.mspx.

[5] STOKES, J. PCI Express: An Overview. Web page, July 7, 2004. http://arstechnica.com/articles/paedia/hardware/pcie.ars/1.

[6] WIKIPEDIA. 8B/10B encoding. Web page, Sept. 19, 2005. http://en.wikipedia.org/wiki/8B/10B_encoding, date visited given.

[7] WIKIPEDIA. Low voltage differential signaling. Web page, Sept. 15, 2005. http://en.wikipedia.org/wiki/LVDS.

[8] WIKIPEDIA. PCI Express. Web page, Sept. 6, 2005. http://en.wikipedia.org/wiki/PCI_Express.

28. Serial ATA

28.1 Introduction

SATA Stands for Serial Advanced Technology Attachment and is primary a technology for communicating with hard drives. Its main predecessor is UATA (Ultra ATA), also known as IDE (Integrated Drive Electronics) or PATA (Parallel ATA). UATA has many shortcomings which has come to inhibit it. Among these shortcomings are crosstalk and clock skew.

Crosstalk is when the magnetic field generated by a wire interferes with a neighbouring wire. The other problem, clock skew, is when the signals on parallel wires arrive out of sync.

SATA solves these problems by using serial communication instead of parallel. Where parallel communications (used by UATA) sends several bits of data in parallel on multiple wires, SATA sends the bits sequentially instead. That is, data is sent one bit at the time over one wire. This eliminates cross talk since there is only one channel and not several to interfere with each other. Clock skew is also eliminated, since there is no longer multiple wires in which the clock pulses might arrive out of sync.

The minimization or elimination of problems like these allows for much higher clock rates, which in turn makes for higher throughput. Serial communication has therefore become increasingly popular for high speed transfers, as the clock frequency can be substantially higher than that in a parallel port. Other buses using serial communication include USB (Universal Serial Bus), FireWire (IEEE1394) and PCI-express.

The latest UATA standard, ATA-8, has a throughput of 133 MB/sec whilst the first generation SATA has a throughput of 150 MB/sec and the second generation has 300 MB/sec. SATA is designed to scale easily to meet future demands, and the third generation SATA is planned for 2007 with a throughput of 600 MB/sec [1].

SATA was developed by the independent organization SATA-II, renamed SATA-IO (Serial ATA International Organization) in 2004. SATA-II is also sometimes misused to reference the second generation SATA standard with the 300 MB/sec throughput. This use of the name SATA-II is however discouraged by SATA-IO [3].

One should also mention the SCSI (Small Computer System Interface), which is another way of communicating with drives. SCSI has a huge throughput (up to 640 MB/sec), but is very expensive and thus is mainly targeted for servers and large databases where high throughput to RAID:ed drives is important. SATA is more (like UATA) intended for desktop computers [5].

28.2 Technology

The SATA data connector consists of seven pins in a row, see Figure 28.1 on the next page. The connector is L-shaped so that it is impossible for the cable to be plugged in upside down.

The outermost pins and the middle pin are used for optional shielding purposes only, which leaves two pairs of pins for data transfer, one pair for each direction. Each

Figure 28.1: The SATA data connector [9].

pair sends data using Low Voltage Differential Signaling, LVDS. This means that bits are represented as a low voltage difference between the two wires. LVDS reduces the effects of noise substantially, since normally noise affects both wires equally, which leaves the voltage difference unaffected. This enables lower voltages which in turn makes way for higher frequencies, since it is faster with small changes in voltage [7]. The wires are normally twisted around each other in the SATA cable. Twisted pair is a well used technique for reducing the magnetic footprint of a wire by twisting it with another wire carrying opposite voltage. This makes the two wires magnetic fields cancel out almost completely, thus minimizing crosstalk between the pairs.

It is important that the clocks of the hard drive and controller is in perfect sync all the time. SATA does not have a separate clock wire, and so to keep the clocks synchronized it is important that the transferred bits alternates frequently. If they did not, it would be very hard to, as an example, distinguish nine sequential zeroes from ten. To solve this problem SATA sends data in 8b/10b encoding. This means that each byte (8 bits) are actually sent as 10 bits in a manner to make sure that there is not too many ones or zeroes in a row. When nothing is transmitted a pattern of alternating ones and zeroes are sent to keep the clocks synchronized [6].

Another difference between UATA and SATA is that UATA supports two devices per bus. This means that the two devices share the bandwidth. To handle two devices on one bus, a lot of extra communication is needed to negotiate who is to use the bus next. This adds overhead and can also hinder faster devices by being forced to wait for a slower device on the same UATA cable.

SATA is on the other hand purely point-to-point. Only one device may be connected to a SATA bus, and so the bus can be used more efficiently. This also eliminates the hassle of the master-salve relationship of UATA-devices, where one device would be the master of the other and be the one to control who was to talk on the bus [1].

The SATA protocol allows for hot-swapping. Hot-swapping is when a device can be plugged or unplugged at run time. When a SATA disk is connected to a running system the SATA controller will recognize the new device immediately.

28.3 Native Command Queuing

SATA supports Native Command Queuing, or NCQ. NCQ is a technique where the drive can rearrange the order of the commands given to it to optimize speed. When the

drive is instructed to read from several locations on the disk it can rearrange the order in which to read them so that it can read as many fragments as possible in one rotation. NCQ is best suited for devices that are often instructed to read data from distant places on the disk. This is quite common for database hard drives when multiple users want to access different parts of the disk. For normal home PC usages however, the gain from NCQ is questionable [8].

28.4 First-party DMA Controller

The SATA specification supports first-party DMA. DMA stands for Direct Memory Access and is a technique for copying data without the involvement of the CPU. In the case of SATA, first-party DMA means that the device takes control of the data transfer to and from memory over the memory bus. This is in contrast to third-party DMA (used by UATA), where the DMA-controller of the motherboard are responsible for the transfer. Giving DMA control to the SATA-device makes way for efficient use of Native Command Queuing, as the SATA device can send data on its own initiative in any order it sees fit [2].

28.5 The Power Cable

The SATA specification also includes a separate power connector for SATA devices. The power connector resembles the SATA connector with its distinct L-shape, but has 15 pins instead of seven (see Figure 28.2). The pins are used to supply current with different voltages, including 3.3 V, 5 V and 12 V. This was to to support different types and sizes of drives.

Figure 28.2: The SATA power connector [9].

28.6 External SATA

While SATA was intended for internal devices only, it was realized that it could work for external devices as well.

In 2004 SATA-IO released the External SATA (eSATA) standard. It only differs from the SATA specification by specifying new connectors and a shielded cable. The

main reason for defining a shielded cable was to allow for cable length up to two meters, instead of SATA's maximum length of one meter. To disallow internal unshielded SATA cables to be attached to an eSATA device the connector was also redesigned. The connector has the L-shape removed to guarantee that an unshielded internal cable can not be used with eSATA. The connectors was also made a lot more robust to sustain thousands of plugs and unplugs.

The performance of eSATA is equal to that of the SATA standards, which currently means speeds up to 300 MB/sec. This can be compared to that of other external connectors, such as FireWire (IEEE 1394b) which has a peak transfer rate of 100 MB/sec. However, FireWire external drivers are actually UATA drivers with a bridge chip that translates between FireWire and UATA. This has an substantial overhead which leaves the efficiency at about 80 MB/sec. An eSATA drive need no such bridge and will thus have no extra overhead. This means that external devices can be as fast as internal, making it possible for large RAID controlled external disk storage without using SCSI. See [4] for more information about eSATA.

28.7 References

[1] SATA-IO. Serial ATA - a comparison with Ultra ATA technology. White Paper, 2002. Available at `http://www.sata-io.org/docs/Serialata - a comparison with Ultra ATA Technology.pdf`.

[2] SATA-IO. Serial ATA: High speed serialized AT attachment. White paper, Jan. 7, 2003. Available at `http://www.serialata.org/docs/serialata10a.pdf`.

[3] SATA-IO. Dispelling the confusion: SATA II does not mean 3Gb/s. Web page, Sept. 22, 2005. `http://www.sata-io.org/namingguidelines.asp`, date visited given.

[4] SATA-IO. External SATA (eSATA): Features and benefits of a new external storage interface. Web page, Sept. 22, 2005. `http://www.sata-io.org/esata.asp`, date visited given.

[5] TECHNICAL COMMITTEE T10. SCSI storage interfaces. Web site, Oct. 3, 2004. `http://www.t10.org`.

[6] WIKIPEDIA. 8b/10b. Web page, Aug. 13, 2005. `http://en.wikipedia.org/wiki/8B/10B_encoding`.

[7] WIKIPEDIA. Low voltage differential signaling. Web page, Sept. 23, 2005. `http://en.wikipedia.org/wiki/LVDS`.

[8] WIKIPEDIA. Native command queueing. Web page, Sept. 19, 2005. `http://en.wikipedia.org/wiki/Native_command_queueing`.

[9] WIKIPEDIA. Serial ata. Web page, Sept. 17, 2005. `http://en.wikipedia.org/wiki/Sata`.

29. Universal Serial Bus

29.1 Introduction

This text present a cursory overview of the Universal Serial Bus (USB). This bus has become very popular over the last couple of years and is available in almost all new desktop computers. The number of different peripherals has practically exploded, making it possible to connect everything from digital cameras to small humidifiers using USB [5].

29.1.1 Background

As personal computers grew more and more popular in the beginning of the 1990s, it became clear that the existing ways to connect peripherals to a computer would not be adequate for future demands. They did not provide enough bandwidth, the many different connectors were fragile and often confusing for the inexperienced user. They where also severely limited in the number of peripherals that could be connected at a time, and they did not support plug-n-play.

The USB Implementers Forum, Inc. (USB-IF) was formed in 1995 to solve these problems by creating a unified bus technology to support a wide range of peripherals. Initially the USB-IF consisted of 340 companies with seven core companies; Compaq, Digital Equipment Corporation (DEC), IBM, Intel, Microsoft, NEC, and Northern Telecom [4].

Today USB-IF have more than 900 member companies and the Board of Directors consists of Agere Systems, Hewlett-Packard, Intel, Microsoft, NEC, and Philips.

29.1.2 Terminology

The USB specification contains some terminology that need explanation. These terms will be used in the rest of this text.

Function "A USB device that provides a capability to the host, such as an ISDN connection, a digital microphone, or speakers [3]."

Device Either a function, a hub, or a *compound device*. A compound device contains a hub and one or more functions.

Hub Like an Ethernet hub, the USB hub has one upstream port, which directly or indirectly is connected to a host, and typically several downstream port where devices can be plugged in.

The hub provide communication between the devices connected to it and the host. It also provides power management, configuration and monitoring for the host as well as transfer speed conversion in the case of USB 2.0 [2].

29.2 USB Versions

As computer and user needs have changed over time so has the available USB standards. Most notable are increases in transfer rates and addition of direct device to device communications.

Below are short descriptions of the most relevant existing USB specifications—the content of the rest of this text covers USB 2.0 though.

USB 1.1 Released in September 1998, this version of USB supports two speeds, called *low speed* and *full speed*. These allow transfer rates of 1.5 Mbps and 12 Mbps, respectively.

The low speed mode makes it possible to build very cheap USB devices since simpler components can be used. Low speed is also useful where a lighter, more flexible cable is of use, the prime example is the computer mouse. It is possible to make the cable lighter since interference is less of a problem in low speed mode, making twisted pair or heavily shielded cables unnecessary [2].

USB 2.0 This backwards compatible version of USB primarily provides higher performance, increasing the maximum transfer rate from 12 Mbps to 480 Mbps. Called *high speed*, this new limit is a response to the ever increasing demand for bandwidth where users wish to transfer videos and high resolution images in reasonable time.

The requirements on hubs became higher with the introduction of this new version since they must allow both older 1.1 and new 2.0 devices to share the same bus [2].

USB On-The-Go This specification is a supplement to USB 2.0 which adds features that are useful for mobile devices. The primary addition is support for point-to-point communication between devices, such as a camera and printer, without the need for a computer to sit in the middle. This is done by letting the devices negotiate for the role of host using new additions to the communication protocol.

USB On-The-Go also introduces new, smaller, cables and connections, as well as features that allow for better power efficiency which is very much of a priority when designing battery powered devices [6].

Wireless USB With the growth of GSM, WiFi, and other means of wireless communications, people have seen the benefit of not being bound by a wire. The Wireless USB (WUSB) standard wishes to add this benefit to computer peripherals and be to the desktop computer what Bluetooth is for mobile devices.

The WUSB 1.0 specification was released in May 2005 and the first devices using this standard are expected for the end of 2005. Using radio communications WUSB provides speeds of 480 Mbps at close range, approximately three meters [1].

29.3 Architecture

The design goals of the USB architecture focus on simplicity and expandability.

29.3.1 Topology

USB uses a tiered star topology where each star is built around a hub. An example topology using two hubs is shown in Figure 29.1. The maximum number of hubs in cascade is five, that is, the number of tiers will never exceed seven—the root hub, five non-root hubs and the peripheral. This limit is imposed due to various timing constraints. The maximum cable length is five meters which makes the maximum distance between a host and device 30 meters.

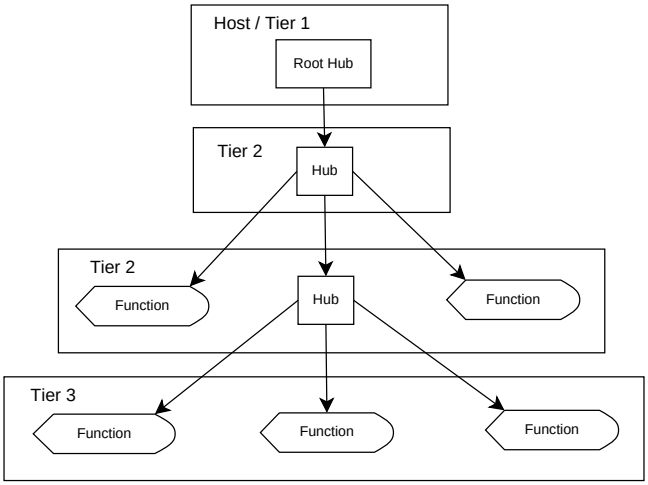


Figure 29.1: Example topology.

Only the physical connections in the USB system describes a tiered star topology, the logical connection between host and peripheral is the same regardless of the number of hubs that lie between them [3].

29.3.2 The Host

At the top of a USB system there is a host. The host contains one or more *Host Controllers.* A Host Controller acts as a layer between the host, the host's operating system, and the USB system.

Initially many computers had a single Host Controller but since only one device at a time can communicate with the Host Controller it has become popular to have several Host Controllers to increase the overall bandwidth.

The host has a responsibility to, among other things, detect when a device is connected or disconnected, manage data and control flow between host and devices, as

well as providing power to connected devices that require it. When a connect or disconnect has been detected by the host then the correct driver for the device should be loaded or unloaded.

A limitation of USB is that there is no support for broadcasting, that is, for a host to send a message to multiple devices it has to send the message to each device in turn. The similar—but more complex—IEEE-1394 (Firewire) standard does support broadcasting [2].

29.3.3 The Device

Each device must contain enough information to completely describe itself. This information is divided into three categories: Standard, class, and vendor. The standard category contains, for example, the vendor identification, device class, and information about the supported power management features. The device class identify the general purpose of the device, a few examples of device classes are hub, human interface, printer, imaging, or mass storage. The class category contains information specific to the device class. Under the vendor category it is possible to store any additional information that might be required for that particular device.

The device is, for example, responsible for detecting when it receives messages directed to the device as well as following the power management requirements stated in the USB specification [3].

29.4 References

[1] AGERE, HEWLETT-PACKARD, INTEL, MICROSOFT, NEC, PHILIPS, AND SAMSUNG. *Wireless Universal Serial Bus Specification, Revision 1.0*, May 12, 2005. http://www.usb.org/developers/wusb/docs/WUSBSpec_r10.pdf.

[2] AXELSON, J. *USB complete: everything you need to develop custom USB peripherals*, 3rd ed. Lakeview Research LLC, 2005.

[3] COMPAQ, HEWLETT-PACKARD, INTEL, LUCENT, MICROSOFT, NEC, AND PHILIPS. *Wireless Universal Serial Bus Specification, Revision 2.0*, Apr. 27, 2000. http://www.usb.org/developers/docs/usb_20_02212005.zip.

[4] KOON, J. The USB Vision: 10 Years Later. Web page, Mar. 25, 2005. http://www.everythingusb.com/timeline.html.

[5] THINKGEEK, INC. ThinkGeek :: USB humidifier. Web page, Sept. 27, 2005. http://www.thinkgeek.com/gadgets/electronic/742d/, date visited given.

[6] USB IMPLEMENTERS FORUM, INC. *On-The-Go Supplement to the USB 2.0 Specification, Revision 1.0a*, June 24, 2003. http://www.usb.org/developers/onthego/OTG1_0a(PDFs).zip.

List of Authors