

EVTEK University of Applied Sciences  
Institute of Technology  
Degree Program in Information Technology

**Claudio M. Camacho**

**The PowerPC G4 Processor**

Computer Architecture

25 May 2008

Student 0604495

Supervisor: Seppo Haltsonen

## Contents

<b>1</b>	<b>Motivation</b>	<b>3</b>
<b>2</b>	<b>History of the PowerPC in Brief</b>	<b>3</b>
<b>3</b>	<b>Anatomy of the PowerPC G4</b>	<b>4</b>
3.1	Overview . . . . .	4
3.2	Functional Description . . . . .	6
<b>4</b>	<b>The Instruction Set</b>	<b>6</b>
4.1	Instruction Formats . . . . .	7
4.2	Addressing Modes . . . . .	8
<b>5</b>	<b>Memory Organization</b>	<b>9</b>
5.1	Memory Model in Brief . . . . .	9
5.2	The PowerPC Registers Set . . . . .	9
5.3	Special Registers . . . . .	10
5.4	Stack Specifics . . . . .	10
<b>6</b>	<b>Interrupts and Timers</b>	<b>11</b>
6.1	Timer Functionality . . . . .	11
6.2	Interrupts . . . . .	12
<b>7</b>	<b>Special Features: AltiVec</b>	<b>13</b>
<b>8</b>	<b>Conclusions</b>	<b>14</b>

## 1 Motivation

Based on the personal esteem of the author for RISC<sup>1</sup> processors, and due to the fact that the AIM<sup>2</sup> alliance resulted in the possibility for end users to purchase desktop workstations having a PowerPC processor, the author has chosen this concrete architecture as it is a suitable example, due to its broad availability in comparison to the well-known x86 architecture.

Moreover, the author itself is owner of a PowerPC G4 laptop computer, which ships with a PowerPC 7457 processor, thus being a highly enriching opportunity for widening the knowledge of a powerful architecture which used to serve as a multipurpose computer design, from portable laptops to Apple Xserve rack-nodes.

Finally, the answer to a probable question about why not MIPS<sup>3</sup> or Sun's SPARC<sup>4</sup> is simply a matter of availability. As it was mentioned above, the owning of a PowerPC G4 computer by the author makes it easier to perform actual tests, thus providing a real feedback for the writing of this document.

## 2 History of the PowerPC in Brief

IBM<sup>5</sup> has always been behind behind the scene manufacturing advance machines, being usually ahead of the rest of companies in the field of microprocessors development, although things are changing nowadays. Thus, near the year 1990, IBM was positioning its AIX<sup>6</sup> operating system on the market by the release of the PS/2 AIX version. Next, in 1990, IBM released a new machine named System/6000, which was given the name POWER, with which the saga of POWER computers started. POWER was an intended

---

<sup>1</sup>RISC stands for *Reduced Instruction Set Computer*.

<sup>2</sup>AIM is known as the *Apple-IBM-Motorla* alliance, which was formed in 1991 as a cooperative work between Apple Computer, IBM and Motorola, in order to create a computing standard based on the PowerPC architecture. [1]

<sup>3</sup>MIPS stands for Microprocessor without Interlocked Pipeline Stages, and it is a computer architecture developed at MIPS Technologies. It is a RISC processor which was developed at the beginning of the whole RISC-concept history, by John L. Hennessy and David A. Patterson.

<sup>4</sup>SPARC stands for Scalable Processor ARChitecture, which is a RISC processor designed by Sun Microsystems in 1985.

<sup>5</sup>IBM stands for the name International Business Machines Corporation, although usually referred as *Big-Blue*.

<sup>6</sup>AIX stands for Advanced Interactive eXecutive, which is an Unix-based operating system for being run on IBM RISC computers.

acronym for *Performance Optimization With Enhanced RISC*. [2]

This first POWER architecture had remarkable features -as of that time- such as 8KB instruction cache (I-cache) and up to 64KB data cache (D-cache), one floating-point unit capable of performing floating-point multiply-add (FMA) instruction per cycle, basic built-in 3D capabilities, and much more. This processor was made of 800k transistors in a single silicon chip and it was able to perform 30 million of instructions per second (30MHz). Afterwards, several improved models followed the first POWER processor, including a multi-node architecture named SP1. [2]

Subsequently, in 1993, Apple, IBM and Motorola would create an improved version of the old POWER, which supported higher frequencies of operation and was able to execute up to three instructions per cycle. This would become the first PowerPC of the history, named PowerPC 601. Moreover, in the following months, IBM released an enhancement of this processor under the name of POWER2. In this case, the major features were the inclusion of a FPU (Floating-Point Unit) with two 64bit execution units, where floating-point multiply-add instructions were executed within one cycle. [2]

Between the years 1994 and 2000, IBM accounted for an important number of high-performance processors, among POWER3 and POWER3-II were created. POWER3 was the first step towards the future, by unifying the PowerPC and the POWER2 architecture in a new breed of 64bit computers, which was able to execute two floating-point multiply-add instructions per cycle and included high-bandwidth buses (up to 256bit buses). However, POWER4 was about to come. [2]

POWER4 was the processor that broke the 1GHz frontier, and it was made of 174-million transistor in a single chip. The PowerPC G4 architecture is, in principle, based on the IBM's POWER4 microprocessor, with slight differences. [2]

### **3 Anatomy of the PowerPC G4**

#### **3.1 Overview**

Concretely, the PowerPC 7457 is a restricted 32-bit PowerPC implementation (with a super scalar PowerPC core), which includes the following main features: [3, 21]

- AltiVec technology (128-bit-wide vector execution unit)
- Dual 32KB cache for data and instructions
- 512KB on-chip level 2 cache with a clock speed ratio of 1:1
- High bandwidth bus (MaxBus, also compatible with the 60x bus)
- Fully symmetric multiprocessing capability

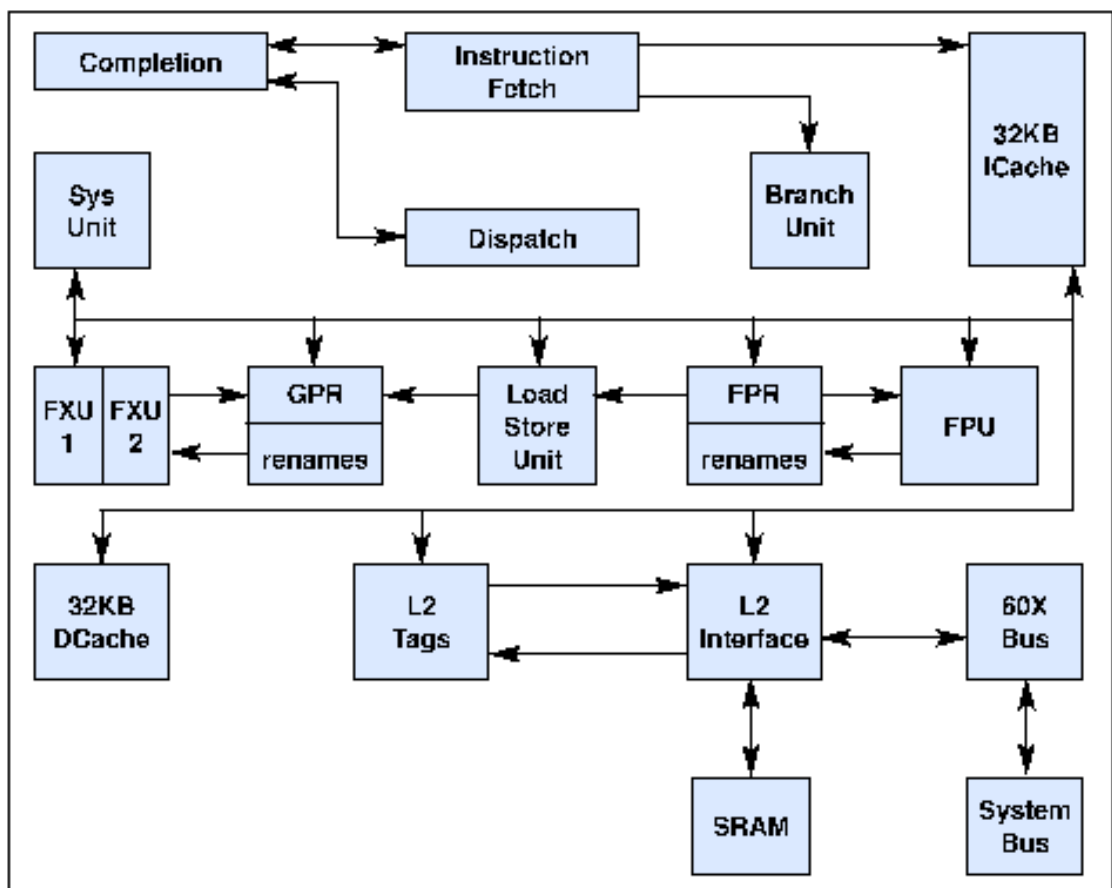


Figure 1: PowerPC 750 Overview, by IBM [4, 3].

Figure 1 shows the logic organization and structure of the PowerPC 750 Processor, which is architecturally similar to the PowerPC 7457. As it can be depicted, there are two floating-point execution units: FXU1 and FXU2. Furthermore, other characteristics mentioned above, such as a dual 32KB cache for data and instructions, a 60x-compatible bus, etcetera.

### 3.2 Functional Description

The PowerPC 7457 microprocessor is a CMOS copper-based processor, which provides eases for reducing the circuitry in size (thus requiring less cooling and less power consumption) while delivering faster performance operations [4, 2]. Due to the limitations of this paper, the detailed characteristics of the PowerPC 7457 processor will be covered in brief. For further information, please recur to the references and sources of this document. Please consider Table 1, which provides a comprehensive description of the processor's subsystems and their main features:

<b>Subsystem</b>	<b>Features</b>
Power Management Unit (PMU)	Dynamic power management and integrated thermal assistance.
Instruction Fetching and Dispatch Unit	Four instructions fetched per clock, two instructions dispatched per cycle, 4-stage pipeline (fetch, dispatch, execute and complete).
Load-Store Unit	One-cycle cache access, cache and TLB <sup>7</sup> instructions execution, alignment and number denormalization, hit under reload instruction.
Fixed-point Execution Unit	One-cycle add, subtract, shift or rotate, hardware-based multiply/divide, thirty-two 32bit general purpose registers.
Floating-point Execution Unit (FPU)	Thirty-two 64bit floating-point registers.
Memory Management Unit (MMU)	8 block address translation registers, fast-trap mechanism for software reload TLB, both big- and little-endian addressing support.
Cache Unit	32KB (32-byte line), 8 way set associative instruction- and data-cache, copy-back and write-through data cache, hardware-based data coherency.
Bus Interface Unit	General purpose system configuration, 32-bit address and 64-bit data bus, on-bus parity checking, LSSD <sup>8</sup> fast reset.

Table 1. PowerPC 750 Functional Description [4, 3].

## 4 The Instruction Set

The PowerPC processor has three well-differentiated instruction classes:

- Branch instructions
- Fixed-point instructions
- Floating-point instructions

The PowerPC architecture is mostly binary-compatible with the POWER architecture, although some instructions may vary. [6, 12]

## 4.1 Instruction Formats

In PowerPC, all the instructions are four bytes long and word-aligned. The bits 0 to 5 specify the OP code, whereas some instructions may have an extended OP code (XO). The remaining bits in the instruction define different fields according to concrete instruction format. [6, 18]

Besides, instructions must be aligned. This means that the starting address of each data item must be a multiple of four (word-aligned), thus existing the possibility for certain bits in the item to be don't-care values, acting as padding. In reference to the byte-ordering, PowerPC supports both big- and little-endian, when moving information from registers to the memory, although big-endian is the default byte ordering. PowerPC uses two bits in the MSR which tell the byte ordering. [7, 95-100]

In spite the classification of instructions by type, they are typically classified by class as well. In PowerPC exist three classes of instructions:

**Defined** A normally defined instruction, with a known OP code.

**Reserved** Special-purpose instructions.

**Illegal** Unknown OP-code instructions, neither defined nor reserved.

PowerPC has 17 different formats of instructions for optimizing the compactness of the execution performance. The most important and well known are the **I-Instructions** (for two-operand arithmetics), the **B-Instructions** (for conditional branching) and the **D-Instructions** (for data copy). However, there are many different versions of these three major presented sets, which have the goal of enriching and improving performance in any case, by creating a specialized way of executing certain type of instructions.

Figure 2 represents an example of an I-Instruction, which uses two-operands arithmetics, the instruction **add**, whose OP code is 31.

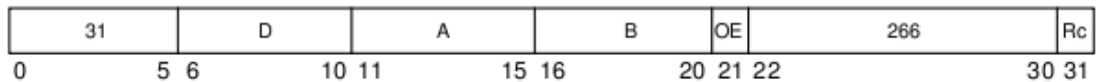


Figure 2: PowerPC **addx** Instruction Format [7, 346-552].

As it can be seen from Figure 2, the first 6 bits (0-5) define the OP code, as usual in PowerPC architectures. The D-operand is the destination of the result, whereas A and B are the operands of the instruction. The *OE* and *R<sub>C</sub>* fields are used for defining which special register may be affected or not by the result, such as the Condition Register or the XER Register.

## 4.2 Addressing Modes

PowerPC, as common as in other processor, presents several ways of referencing data in instructions, which may be classified by the type of instruction accordingly [9, 43]:

- Load and store instructions
  - Indirect
    - \* 16-bit displacement to be added to the base register
    - \* Replacement of the base register with the new address
  - Indirect indexed
    - \* Reference to base and index registers
    - \* EA contains the sum of the contents
- Branch instructions
  - Absolute
  - Relative
  - Indirect
- Arithmetic instructions
  - Direct (operands in registers)
  - Immediate (operands in the instruction)
  - Floating-point must be always used by direct and not immediate

## 5 Memory Organization

The PowerPC architecture has a built-in Memory Management Unit (MMU) which allows to address up to 4096 Mbytes (32-bit physical space). [7, 269-270]

### 5.1 Memory Model in Brief

The PowerPC MMU supports for demand-paged virtual memory, which enables the execution of programs larger than the size of the physical addressable memory. Virtual memory uses the concept of virtual address space, which must be larger than the physical address space, where the address translation flow from 32-bit to a 51-bit virtual space (performed by the MMU).

The main memory is divided by the MMU into regions (address spaces of 256Mbytes) and blocks (address spaces from 128Kbytes up to 256Mbytes). The virtual memory segments can be divided into 4K pages. The operating system must create a page descriptor (Page Table Entry -PTE-), which is then used by the MMU to perform the translation to the physical address.

### 5.2 The PowerPC Registers Set

The PowerPC architecture can be divided into three organizational levels of registers: the user instruction set architecture (UISA), the virtual environment architecture (VEA) and the operating environment architecture (OEA). PowerPC provides register-to-register transfers for all instructions, however register-to/from-memory are performed only using the specific load and store instructions. All the registers are 32-bit long, except for the floating-point registers. [7, 55-71]

Operands are accessed from general-purpose registers (GPRs) and floating-point registers (FPRs). The PowerPC provides 32 general-purpose registers, 32 floating-point registers, a 32-bit 4-fields-divided condition register (CR or  $R_c$ ), a floating-point status and control register (FPSCR), a XER register which tells about overflows/carries and data bytes transferred in load/store instructions, a link register (LR) used for branching and a count register (CTR) appropriate for looping situations. [7, 55-71]

### 5.3 Special Registers

The OEA registers set level encloses a wide variety of supervisor-level registers which are briefly depicted subsequently.

- Configuration registers
  - Machine state register (MSR): defines the processor state.
  - Processor version register (PVR): read-only identifier of the processor.
- Memory management registers
  - Block-address translation registers (BAT).
  - SDR1 register: specifies the page base for virtual memory translation.
  - Segment registers (SR): contain the segment descriptors.
- Exception handling registers
  - DSISR and data address register (DAR), for handling alignment.
  - SPRG registers for the use of the operating system.
  - Save and restore registers (SRR0 and SRR1) for program interruption.
  - Floating-point exception register (FPECR).
- Other registers (include timing, data breakpointing, and other features).

[7, 55-94]

### 5.4 Stack Specifics

The PowerPC uses a stack which grows from higher to lower addresses. The specialty of the PowerPC stack is that it is only managed by the stack pointer, without a frame pointer, which provides a fixed-size stack. Such stack pointer is held in a general-purpose register, the GPR1, typical of RISC processors. Furthermore, there are no dedicated instructions nor specific addressing modes supporting the stack, so it has to be used manually by the programmer using the GPR1. [10]

The lack of dedicated stack pointer enhances performance by reducing the access to memory. Then, a link register (LR) is used to hold the returning address from a subroutine call, where the SRR0 and SSR1 registers are used for the program exceptions. [10]

## 6 Interrupts and Timers

### 6.1 Timer Functionality

The PowerPC G4 uses a time base register, which is a 64-bit register whose purpose is to count ticks, depending on the driving frequency of the processor implementation. The time base register consists of two 32-bit parts, the TBU (Time Base Upper bits) and the TBL (Time Base Lower bits). Thereof, as the time base register counts with an increment of one every tick, the TBU is incremented by one when the TBL reaches  $0xFFFFFFFF$ . [7, 596-597]

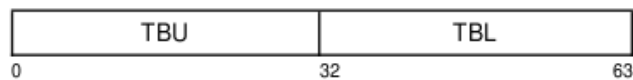


Figure 3: PowerPC Time Base Register, by IBM [12, 67].

Analogously, there exists a real-time clock (RTC) for the POWER architecture which ensures that certain amount of counts is done in a given period of time, such as a guarantee. However, the very PowerPC G4 does not implement an RTC itself, and hence the TB (time-base register) does not guarantee the exact periodicity in accuracy. For instance, the POWER RTC ensures that one count is done within the exact time of 10 *addi* instructions, whereas the TB cannot be guaranteed in a similar fashion. [7, 596-597]

When the time base register overflows, there is no explicit notification as such. Nevertheless, PowerPC G4 provides mechanisms for generating an interrupt when the driving frequency is updated and for finding what is that frequency, thus allowing the operating system to estimate time base calculations and timestamps generations. [11, 37-39]

As a supplementary approach, the PowerPC G4 processors provides a decremter, which is a register updated at the same rate as the time base register is updated. When the decremter is supposed to go below the value zero, an exception is fired which acts as an interrupt for timing purposes. The decremter is a 32-bit register containing a signed integer, whose values can be read into the general purpose registers using the instruction *mfdec Rx*. [12, 67-69]

## 6.2 Interrupts

As in most platforms, the PowerPC G4 processor provides several mechanisms for interrupting the system, offering two different models, one for ordered interrupts and another one for unordered interrupts. The PowerPC G4 processor works in a coherent way, so that the interrupting mechanism must respond to a model of synchronization. Therefore, the save/restore registers SRR0 and SRR1 are used by the hardware for this purpose. [12, 51-52]

Consequently, SRR0 is set to point to the following instructions which has all preceding instructions completed, and where all the instructions which follow after the one pointed by SRR0 are uncompleted. Furthermore, the instruction pointed by the SRR0 may be completed or not. The PowerPC G4 supports four different type of interrupt sources: a system reset, a machine check, an external interrupt or a decremter exception (timer interrupt). [12, 51-52]

The MSR flag may be set to zero, meaning that the hardware completely ignores the external and decremter interrupts. In PowerPC, all interrupts are considered of precise nature, except the Imprecise Mode Floating-Point Enabled Exception type Program Interrupt. Moreover, for interrupt processing purposes, there is an interrupt vector associated with each type of interrupt, which keeps the list of instructions to be executed when the concrete interrupt takes place. Then, when an interrupt occurs, the processor state must be saved in certain registers and the execution is resumed at the interrupt vector location where it was stalled. [12, 51-54]

Briefly, the event of interrupting provokes the SRR0 to be loaded with an address of execution dependent on the type of interrupt. In addition, the MSR is modified so that certain properties are affected in the hardware while treating the interrupt. The PowerPC G4 includes a wide set of interrupt types for any general-purpose software platform, where the most common interrupt types are: *system reset interrupt*, *machine check interrupt*, *data storage interrupt*, *data segment interrupt*, *instruction storage interrupt*, *instruction segment interrupt*, *external interrupt*, *alignment interrupt*, *program interrupt*, *FP unavailable interrupt*, *decremter interrupt*, *system call interrupt*, *trace interrupt* and *performance monitor interrupt*. [12, 51-80]

**Note:** The PowerPC G4 interrupt facilities is something worthy to be further described

in a technical paper. However, the constraints of this document do not allow for detailed documentation about each type of interrupt available in the PowerPC G4 processor.

## 7 Special Features: AltiVec

AltiVec, although usually known also as *Velocity Engine*, is a native vector extension embedded in the PowerPC G4 (and other processors), which expands the functionality by adding a 128-bit vector execution unit able to work in parallel and coherently with the existing PowerPC fixed-point and floating-point registers. Therefore, G4 processors are able to perform up to 16 operations in one cycle period, which is highly suitable for multimedia streaming and vector graphics calculations requiring high-bandwidth data transfers. [13]

The vector execution unit therefore works in parallel with the whole G4 processor, being able to allocate vectors of 4, 8 or 16 elements long, depending on the data size. The operations defined for this vector unit are SIMD<sup>9</sup>, and hence operations are performed at once on several data elements in a single instruction. In addition, the vector execution unit includes a set of 32 128-bit registers where the data for the AltiVec operations is stored. In fact, the velocity engine provides an extension of the instruction set for the PowerPC architecture in order to facilitate the data processing and transferring from a to its vector storage registers. [14]

In contrast to the classical PowerPC RISC instruction topology, the velocity engine provides an instruction anatomy consisting of up to three source operands and an unique destination operand, where all operands must be vector registers. Nevertheless, the store/load instructions may have a different structure, according to their nature and purpose, as well as other few instructions with immediate addressing included in the instruction itself. [14]

Finally, it is of interest to mention that the AltiVec technology introduces a set of 162 new vector instructions which may be classified in several groups according to their purpose: intra-element arithmetic operations, intra-element non-arithmetic operations, inter-element arithmetic operations and inter-element non-arithmetic operations. [14]

---

<sup>9</sup>By and large, SIMD stands for Single-Instruction Multiple-Data in Computer Engineering.

## 8 Conclusions

Among this paper has been mentioned and remarked several features of the PowerPC G4 RISC processor that are considered primary and of importance and interest. Moreover, it has been spotted several times along the document that the constrains of this report were narrowing and, therefore, crucial details have been sometimes omitted in favor of abstraction and summarization.

Nevertheless, the PowerPC G4 processor has been roughly covered on its main features and wise conclusions may be evaluated after the analysis of the technology available in this type of architecture. First, and as it was presented at the beginning of this paper, the PowerPC G4 has been one of the most interesting alternatives to the x86 architecture between 2000 and 2004, delivering the power of a RISC processor with the enhancement of vectorial parallel processing (AltiVec). In fact, the x86 processor had to match this pace by introducing SSE<sup>10</sup> for SIMD calculations.

In terms of performance and consume, PowerPC G4 processors have typically tent to be of lower frequency and faster internal buses, which always denotes high-performance while keeping a low consume. Nonetheless, it is noticeable from the contents of this document that the PowerPC G4 processor may become more complex to program than any other CISC processor, architecturally speaking. This complexity yields an argument for the performance boost and the obviousness of lower frequencies, meaning less consume.

Aside, and turning to the results of the paper overall, the author is somewhat satisfied about the acquired knowledge on the PowerPC G4 processor. However, this paper only presents some major points of the PowerPC G4 which need to be studied and discussed further, in order to fully understand the PowerPC architecture and G4 design as a whole and its internal behavior and implementation. Thus, the author is looking forward to further tamper with such processor and discover which improvements could be added to this document in order to write a new revision with more technical details and observations.

---

<sup>10</sup>SSE stands for Simple SIMD Extensions, and it is a CPU-side technology introduced in Intel architectures in 1999.

## References

- 1 Charles R. Moore and Russell C. Stanphill. The PowerPc Alliance. *Communications of the ACM*; June 1994, 37(6):25-27.
- 2 International Business Machines Corporation. 27 Years of IBM RISC. [online]. International Business Machines Corporation; 2002.  
URL: [http://archive.rootvg.net/column\\_risc\\_.htm](http://archive.rootvg.net/column_risc_.htm)  
Accessed 24 March 2008.
- 3 Apple Inc. iBook Developer Note. Apple Computer, Inc; April, 2004.
- 4 IBM. PowerPC 750 Microprocessors. United States of America: Internet Business Machines Corporation; 1999.
- 5 Translation Lookaside Buffer is property of the World Intellectual Property Organization. The TLB description is available [online] at:  
URL: <http://www.wipo.int/pctdb/en/wo.jsp?wo=2007073624&IA=WO2007073624&DISPLAY=>  
Accessed 24 March 2008.
- 6 Joe Wetzel Poughkeepsie. PowerPC User Instruction Set Architecture, Book 1 - Version 2.01. IBM; September, 2003.
- 7 International Business Machines Corporation. PowerPC Microprocessor Family: The Programming Environments for 32-Bit Microprocessors. IBM Microelectronics Division; February, 2000.
- 8 Steve Hoxey, Faraydon Karim, Bill Hay and Hank Warren. The PowerPC Compiler Writer's Guide. IBM - Warthman Associates; 1996.
- 9 Adrian J. Pullin. Instruction Sets and Addressing Modes. [online]. December, 2002.  
URL: [http://www.ece.eps.hw.ac.uk/Modules/B35dq1/slides/InstructionSet\\_AddressinModes/sld](http://www.ece.eps.hw.ac.uk/Modules/B35dq1/slides/InstructionSet_AddressinModes/sld)  
Accessed 30 March 2008.
- 10 Embedded Concepts and Solutions, Inc. PowerPC Technical Tidbits. [online]. 2002.  
URL: <http://www.go-ecs.com/ppc/ppctek1.htm>  
Accessed 30 March 2008.
- 11 Joe Wetzel Poughkeepsie. PowerPC User Instruction Set Architecture, Book 2 - Version 2.01. IBM; December, 2003.
- 12 Joe Wetzel Poughkeepsie. PowerPC User Instruction Set Architecture, Book 3 - Version 2.01. IBM; September, 2003.
- 13 Apple Computer, Inc. Velocity Engine. [online]. Apple Computer, Inc; 2008.  
URL: <http://developer.apple.com/hardware/drivers/ve/index.html>  
Accessed 5 April 2008.
- 14 Freescale Semiconductor, Inc. AltiVec Execution Unit and Instruction Set Overview. [online]. Freescale Semiconductor, Inc; 2004-2008.  
URL: <http://www.freescale.com/webapp/sps/site/overview.jsp?nodeId=0162468rH3bTdGmKqW>  
Accessed 5 April 2008.