

Mundo Real: Power PC

Uma breve imerção sobre o power pc

Luís Filipe M. Loureiro | Elementos de Sistema | Maio de 2020

# Introdução

Em um primeiro momento pode causar uma certa estranheza que a capa de um estudo sobre um microprocessador, tenha a foto de um garoto jogando Playstation 3. Entretanto, o POWER PC, abreviação de *Performance Optimization With Enhanced RISC- Performance Computer,* apesar de ter sido implementado no *Macintosh* e no *iMac*, ambos da Apple, ele foi especialmente importante no mundo dos videogames, mais precisamente na

até seu uso atual, além de explorar sua arquitetura e instruções.

Histórico do Power PC

Antes de entrar na história propriamente do Power PC, é interessante fazer um curto recuo histórico para 1975. Nesse ano, na sede da IBM, ocorreu a finalização do desenvolvimento do Projeto 801. Liderado por John Cook, foi criado a família de arquitetura chamado RISC, *Reduced Instruction Set Computing.*

O RISC tornou-se uma alternativa ao CISC, *Complex Instruction Set Computing*, ele foi pensado de modo a diminuir o número de instruções suportáveis, aumentando, assim, a rapidez de processamento. Além disso, o processador RISC cada instrução é realizada em apenas um ciclo (*Program Counter*) e as instruções são separadas da memória (data).

Em 1982, a IBM iniciou o que ela chamava de *American Project* cujo objetivo era criar o processador mais rápido do mercado. Partindo da arquitetura RISC, foi criado um multi-chip chamado POWER, em 1990. O grande objeto a ser sanado nesse projeto era criar um processador que permitisse o maior número de ligações telefônicas ao mesmo tempo. Uma vez lançado no mercado, o POWER não se mostrou tão atrativo levando a IBM a pensar que o problema estava no fato do processador ser multi-chip, ou seja, seu hardware é seccionado em diversos microchips.

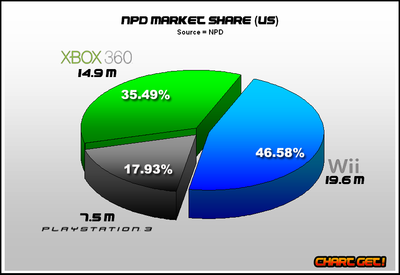
Diante do desafio de transformar o POWER em um single-chip, a IBM convidou a Apple e a Motorola para fazerem parte do projeto. A primeira se mostrava uma referência na arquitetura RISC enquanto a segunda apresentava grande experiência na fabricação de chips em larga escala. Tal surpreendente aliança recebeu o nome de AIM. Importante notar que nessa época havia um amplo domínio no mercado de chips por parte da Intel, com a venda do Intel 80286 e 80486, chips que integravam a grande maioria dos computadores pessoais que faziam uso da arquitetura RISC.

Nos primeiros anos de 1990, o POWER PC, resultado da aliança AIM, tornou-se um grande sucesso no mercado; Macintosh, Windows NT3.51 e Solaris utilizavam o processador. O POWER tornou-se um benchmark no mercado, por apresentar uma velocidade de processamento alta, pouca dissipação de calor e gasto de energia reduzido.

“Parte da cultura aqui é não ter uma cultura da IBM, Motorola ou Apple, mas ter a nossa cultura”, Russel Stanphill sobre a AIM durante o desenvolvimento do POWER

Entretanto, a empolgação com o POWER PC foi caindo no mercado ao final da década de 90, com o lançamento de novos processadores da Intel, deixando o POWER ultrapassado em termos de economia de energia. Aos poucos, o número de computadores que utilizavam o POWER como processador foi caindo, restando apenas o Macintosh nos anos 2000. Com a queda do domínio do chip, a relação em torno da aliança AIM também foi se deteriorando, tendo seu rompimento em 1998 marcada pela ríspida discussão entre Steve Jobs, que acabara de retornar à Apple, e o CEO da Motorola, Christopher Galvin.

A insistência da Apple em utilizar o POWER PC durou até 2003, quando passou a utilizar chips Intel, algo que permanece até os dias de hoje. O sucesso do POWER PC fica mais claro no mercado de consoles de videogames; praticamente 100% do market-share era composto por consoles que utilizaram processadores derivados do POWER PC.



**Imagem 1:** Market-share da sétima geração de consoles

Além dos consoles expostos na Imagem 1, o GameCube, Wii e Wii U também usaram processadores POWER PC.

Pelo fato da produção do Playstation 3 e do Xbox-360 terem sido descontinuadas, a produção do POWER PC globalmente foi drasticamente reduzida. Atualmente, o processador é encontrado no Wii U (ainda produzido) e na linha de produtos Amiga da marca Commodore. Inclusive, o lançamento do Amiga One há dois anos, utiliza o POWER PC como processador.

Arquitetura do Power PC

Desde a sua criação, já foram criados centenas de *branches* de POWER PC, ou seja, houve modificações em sua arquitetura para atender determinado objetivo. Entretanto, em todos os POWER PCs eles apresentam instruções de 32 bits de largura e há poucas instruções para manipular a data diretamente na memória, assim como no Z01, primeiramente é copiado a informação da RAM para um registrador, manipulado, retornando, assim, para a memória.

Apesar do tamanho da instrução ser fixa para todos os POWER PCs, a sua largura pode variar; um exemplo disso são o POWER PC G5 e o POWER PC 604, com 64 bits e 32 bits de largura, respectivamente. Entretanto, desde o princípio do desenvolvimento do processador, foi considerado a possibilidade de uma variação na largura, dessa maneira, aplicativos pensados para processadores de 32 bits também poderiam ser interpretados para os de 64 bits.

Devido ao fato de o POWER PC ter passado por diversos aprimoramentos e mudanças, nesse estudo será estudado a arquitetura do POWER PC 604, lançado em 1994. Interessante notar que o processador supracitado, desde a sua primeira versão apresenta diferenças significativas em relação ao Z01, estudado em aula. Sendo elas:

- Possibilidade de executar 4 instruções por ciclo

- Possibilidade de executar instruções fora da ordem

- Instruções ocorrendo em paralelo no *hardware*

A arquitetura do POWER PC pode ser divido em dez unidades funcionais:

1. *Program Counter* ou IAR (*Instruction Adress Register*) é o *clock* do processador e funciona de maneira incremental
2. Unidade de Predição de Desvios (BPU) é responsável por providenciar à *Fetch Unit* um desvio no IAR
3. Unidade de Busca (*Fetch Unit*) a partir do endereço provindo do IAR ou do BPU, ela providencia as instruções a serem executadas
4. Unidade de Decodificação (*Dispatch Unit*) atua na decodificação das instruções e despacha para as unidades de execução apropriadas
5. Unidade de Controle (CRU) executa todas as instruções lógicas e de controle de fluxo relativas aos registros de controle
6. Unidades Lógicas Aritméticas (ULA) realiza as operações aritméticas
7. Unidade de Término das Instruções (*Completion Unit*) controla a atualização do *hardware* e finaliza as instruções executadas
8. Unidade de ponto flutuante (FPU): executa operações em ponto flutuante, ou seja, bits
9. Unidade de Leitura/Escrita (LSU) atua na transferência de dados provindo dos registradores e memória para a saída (resultado)
10. Cache de dados e instruções: Informações guardados em vetores de 32 bits acerca das instruções e dos dados

Assim como o Z01, o POWER PC 604 também apresenta um *Program Counter* e uma ULA. Ademais, a unidade funcional BPU assemelha-se ao comando *jmp* do processador estudado em aula.

Os registradores no POWER PC 604 são distribuídos da seguinte maneira:

- General-purpose registers (GPRs): há 32 GPRs e eles servem para armazenamento rápido de dados

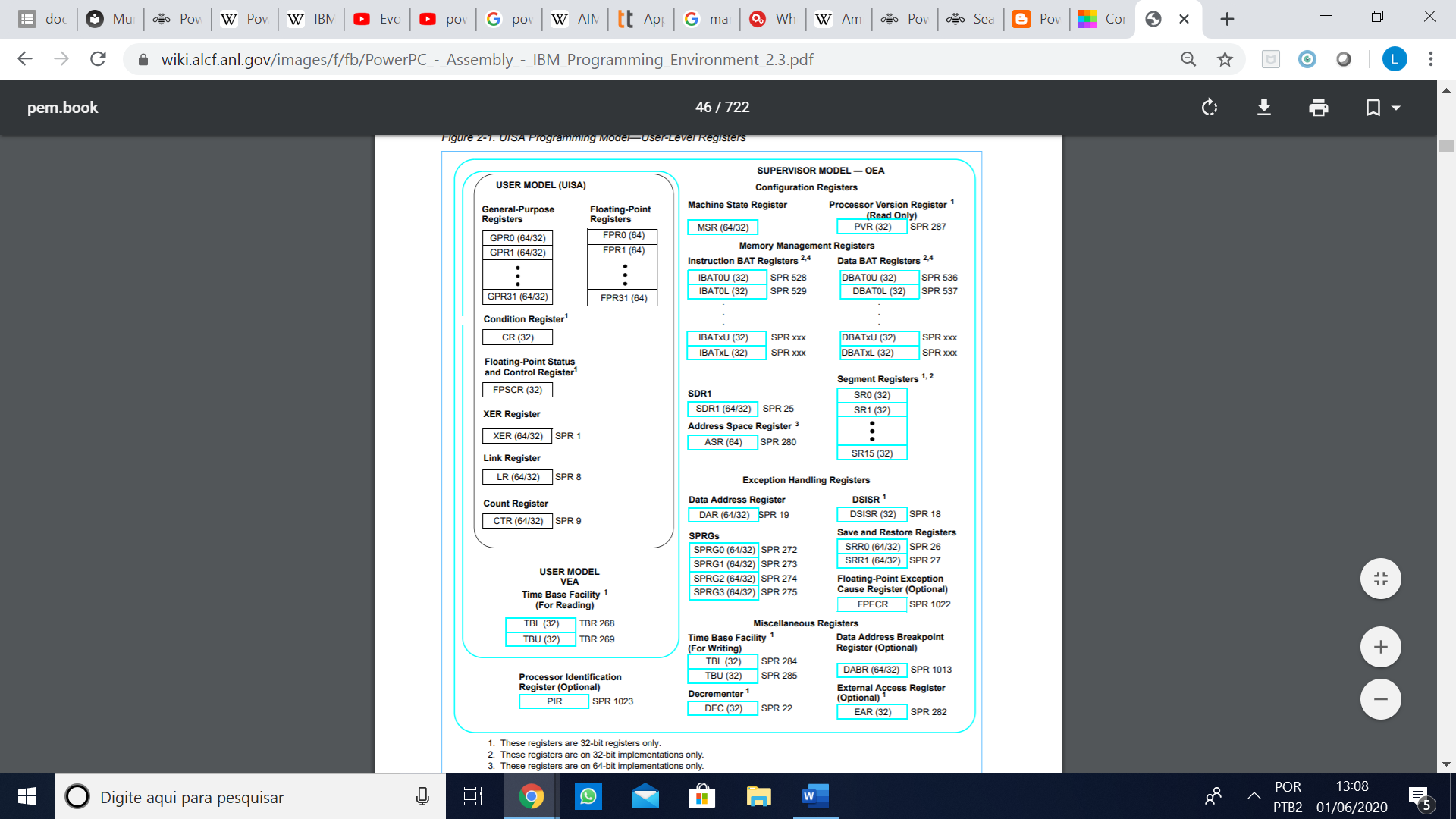
- Floating-point registers (FPRs): servem seja para o destino ou saída de instruções do tipo *floating points*

- Condition register (CR): armazena resultados de certas operações da ULA

- Floating-point status and control register (FPSCR): armazena exceções

-Fixed point exception register (XER): armazena loads e carry de operações da ULA

- Count register (CTR): armazena o loop de incremento do *Program Counter*

A Imagem 2 apresenta a relação dos registradores supracitados no POWER PC. Importante notar, que parte deles podem ser acessados explicitamente, como é o caso dos GPRs e FPSCRs, ou seja, por meio de instruções do tipo mtspr (*Move to Special-Purpose Register*) ou do tipo mfspr (*Move from SpecialPurpose Register*). Já os outros registradores são acessados implicitamente, ou seja, são utilizados para armazenamento temporário para a realização de operações na ULA.

**Imagem 2**: Registradores no POWER PC

Com relação à ULA do POWER PC, ela se mostra muito mais complexa que a do Z01, podendo realizar 26 operações. No documento que descreve o processador criado pela IBM [4] nas páginas 137-141 são listadas todas as operações realizadas. Nesse estudo serão citados duas delas: a *multiple low* e a *negate*. A primeira é dada pela instrução *mullw*, multiplica os registradores rA e rB, armazenando em rD. Já a *negate,* dada por *neg,* nega o valor de rA e armazena em rD.

Instruções do Power PC

Independente da largura do POWER PC em questão (32 ou 64 bits), as instruções são sempre de 32 bits, além disso, para qualquer operação entre registradores é necessário, na instrução, especificar o registrador de destino, ou seja, que irá armazenar o resultado de tal operação.

Ademais, um ponto curioso sobre as instruções no POWER PC é que, diferente do Z01, os registradores são referenciados apenas pelo seu número, não pelo nome. Assim, 4 pode tanto se referenciar ao *register* 4, como ao *floating point* 4, tornando, por vezes, confuso para um iniciante em POWER PC. Tal característica das instruções pode ficar mais claro no exemplo 1.

addi 3,3,1 (1)

O exemplo 1 apresenta todos os aspectos supracitados da instrução em um POWER PC, nele ocorre uma adição do *register* 3 com o número 1 e é armazenado o resultado no *register* 3. Destarte, uma operação entre registradores demanda que seja especificado um registrador que armazenará o resultado, no caso o *register 3,* e tanto *floating points* como registradores são referenciados apenas pelo número.

Um outro ponto que o exemplo 1 traz é a letra “i” ao final de *add.* Tal letra se deve ao fato de a instrução ser do tipo *immediate*, ou seja, ela diz que a operação deve ser feita assim que recebida e, nessa tipo de instrução, a operação é sempre feita entre um registrador e um número.

add 3,3,1 (2)

O exemplo 2 é muito semelhante ao exemplo 1, com exceção do fato de não ser *immediate*. Essa pequena diferença muita completamente a instrução, não mais ocorre a soma do *registe*r 3 com o *floating point* 1, no exemplo 2, o *register* 3 é somado com o conteúdo no *register* 1 e o resultado é armazenado no *register* 3.

É necessário mencionar também os *mnemonics,* como é o caso do li. Tal instrução é muito semelhante ao leaw do Z01, em que um valor é adicionado à um registrador. Entretanto, o li não reflete o que ocorre fisicamente como o *hardware*, por isso não recebe o nome de instrução. Quando se escreve um *mnemonic,* apesar do computador entender, uma série de comandos são disparados.

li 3,1 (3)

O exemplo 3 carrega o valor de 1 para o *register* 3, entretanto, o processador entenderá o comando como o exemplo 4.

addi 3,0, 1 (4)

Assim, o que ocorre de fato com o hardware é somar o número zero c0m 1 e armazenar no *register* 3. Um outro ponto importante é entender o *machine code*, ou seja, o arquivo bin que é gerado a partir de uma instrução, algo que está sendo visto no Projeto G da matéria. Como já dito, cada instrução é transformada em um arquivo de 32 bits, que é dividido como mostra a Tabela 1.

|  |  |
| --- | --- |
| Opcode | 6 bits |
| Source Register | 5 bits |
| Destine Register | 5 bits |
| Value | 16 bits |
| Total | 32 bits |

**Tabela 1:** Distribuição dos bits de uma instrução

Dessa maneira, como só há 16 bits de espaço para o value, uma instrução de carregamento de um número que seja superior a acarretará um erro.

As instruções do POWER PC incluem as básicas da arquitetura RISC [7], além de outras: cache control, sincronização, tipos de dados e base de tempo. Ao todo há algumas dezenas de instruções. Não cabe nesse estudo a reprodução de todas as instruções, ela pode ser encontrada no registro do POWER PC [6] a partir da página 136.

Análise de um código do Power PC

O código abaixo retirado do próprio site da IBM [8] adiciona dois valores, soma-os e se tem como saída o resultado da operação como um *status code*.

Antes de analisar propriamente o código é necessário ter um entendimento de uma seção, algo que não foi visto no estudo do Z01. As principais seções e seus respectivos significados são dados na tabela 2.

|  |  |
| --- | --- |
| .data | Contém a data pré-inicializada |
| .text | Contém o próprio código (texto) |
| .opd | Contém as declarações oficiais e requeridas para programar em POWER PC |

**Tabela 2:** Significado de cada função

Assim, é possível entender as seções como análogos às funções que, ao serem chamadas, retornam informações.

1 .section ".opd", "aw" # chama a seção opd

2 .align 3 # Altera o Program Counter para um múltiplo de 8

3 .global \_start # Criação de um símbolo global start

4 \_start: #Chama o símbolo criado na linha acima

5    .quad .\_start, .TOC.@tocbase, 0 #Cria um vetor de 8bits e adapta o hardware para recebe-lo

6 .text # início do texto de instruções chamando a instrução text

7    li 3, 1   #load "1" no registrador 3

8    li 4, 2   #load "2" no registrador 4

9    add 3, 3, 4    #adição do conteúdo do registrador 3 com o conteúdo do 4 e armazena o resultado no registrador 3

10    li 0, 1   #load "1" no registraor 0 para system call (finalização do text e consequentemente do código)

11    sc # System Call

O código é iniciado com a seção. opd, necessária para o hardware receber os comandos. Como todo o carregamento dos registradores ocorrerá em vetores de 8 bits, é necessário configurar o Program Couter (PC) e isso é realizado pelo .align 3 (. Em seguida é criado um símbolo global \_start, algo requerido pela seção. opd, sendo que tal símbolo é inicializado na linha seguinte.

A linha 5 serve para criar um vetor de 8 bits (.quad) e como há espaço para 16 bits para um número em cada instrução no *machine code,* coloca-se um sinal @ para referir-se que será utilizado os 8 primeiros bits do vetor da instrução. Interessante notar que o uso de um sinal @ também é fundamental para alocar número que excedam 16 bits de largura, já que poderá ser criado um vetor maior, supondo 64 bits, e utilizar sinais @ como referência para determinadas partes do número.

A linha 6 chama a seção .text e inicia-se propriamente os comandos. A primeira instrução realizada é por meio do *mnemonic* li, carregando um vetor de 8 bits contendo o número “1” para o registrador 3. O mesmo é feito para o registrador 4.

Em seguida, é realizado a operação de soma e cabe notar que como a instrução é um add, ela não é do tipo *immediate*, assim, se refere ao conteúdo dos registradores. Por fim, para finalizar o código é necessário passar o valor “1” para o registrador 0 sinalizando o fim da instrução e chama-se o SystemCall (sc) . Essas questões são pré configuradas e são inicializadas com a seção .opd.

Conclusão

O estudo do POWER PC permitiu um maior entendimento de como ocorre um desenvolvimento de um processador e seu lançamento no mercado, além da acirrada competição pelo *market-share* entre as grandes empresas de tecnologia.

Ademais, esse estudo se mostrou extremamente interessante por aumentar os horizontes acerca da linguagem assembly, possibilitando entender e escrever códigos em um processador muito utilizado (pelo menos no passado).

Referências

[1] IBM. **Archived | PowerPC assembly**. Disponível em: https://developer.ibm.com/technologies/linux/articles/l-ppc/. Acesso em: 31 mai. 2020.

[2] WIKIPEDIA. **POWER PC**. Disponível em: https://en.wikipedia.org/wiki/PowerPC#32-bit\_PowerPC. Acesso em: 1 jun. 2020.

[3] IBM. PowerPC ™ 601 RISC Microprocessor Technical Summary: POWER PC. **NXP**: Freescale Semiconductor, Inc., IBM, v. 1, n. 1, p. 1-32, dez./2005. Disponível em: https://www.nxp.com/docs/en/data-sheet/MPC601.pdf. Acesso em: 2 jun. 2020.

[4] YOUTUBE. **Evolution of PowerPC Architecture, lecture by Michael W. Blasgen and Richard Oehler**. Disponível em: https://www.youtube.com/watch?v=kfem6lllSBI&t=172s. Acesso em: 03 jun. 2020.

[5] COSTA, F. D. O. PowerPC – A História de Produção de Chips da IBM: Power PC. **Universidade Estadual de Campinas**: subtítulo da revista, Campinas, v. 1, n. 1, p. 1-6, mai./2020.

[6] IBM. PowerPC® Microprocessor Family: The Programming Environments Manual for 32 and 64-bit Microprocessors: Power PC. **International Business Machines Corporation**: IBM, IBM, v. 1, n. 2.3, p. 1-722, mar./2005. Disponível em: https://wiki.alcf.anl.gov/images/f/fb/PowerPC\_-\_Assembly\_-\_IBM\_Programming\_Environment\_2.3.pdf. Acesso em: 3 jun. 2020.

[7] JACOB, Prof. Bruce. The RiSC-16 Instruction-Set Architecture: RiSC-16 Instruction Set. **ENEE 446**: Digital Computer Design, USA, v. 1, n. 1, p. 1-4, mar./2005. Disponível em: https://user.eng.umd.edu/~blj/RiSC/RiSC-isa.pdf. Acesso em: 1 jun. 2020

[8] IBM. **Programming concepts and beginning PowerPC instructions**. Disponível em: https://www.ibm.com/developerworks/library/l-powasm1/index.html. Acesso em: 3 jun. 2020.