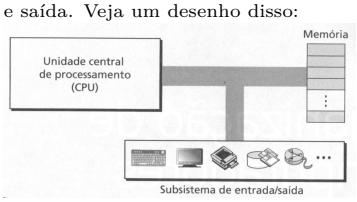
CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76201 ANA GLORIA DE SOUZA BRITO
24CC1203 - 1 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 68719476736 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`237 225 240 221 336 751 796 737 780 212 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

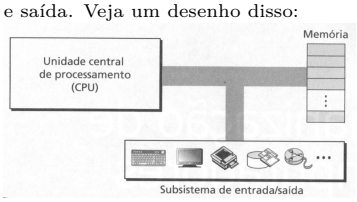
3. Um interpretador do computador acima descrito recebeu o programa
`209 230 234 202 376 335 438 759 698 510 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76218 ANDRE SANCHES 24CC1203 - 2
/ _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 524288 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`218 204 221 239 658 367 445 345 321 477 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

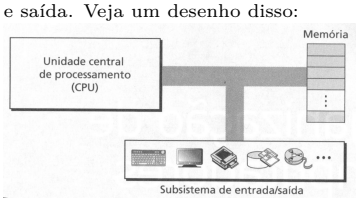
3. Um interpretador do computador acima descrito recebeu o programa
`211 213 232 218 390 704 559 701 668 601 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76225 CASSIO COELHO PAZ 24CC1203 - 3
/ _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 134217728 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`224 222 231 216 455 330 650 323 568 209 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

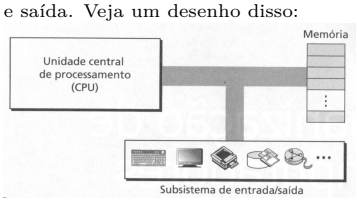
3. Um interpretador do computador acima descrito recebeu o programa
`219 237 237 224 556 223 761 391 285 749 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76232 CAUA DOS SANTOS COSTA
24CC1203 - 4 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 33554432 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`206 230 226 202 768 641 270 715 432 475 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

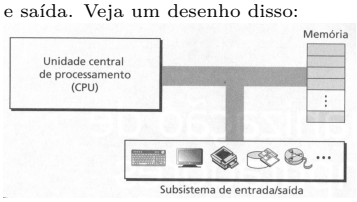
3. Um interpretador do computador acima descrito recebeu o programa
`237 235 212 240 501 209 493 640 739 607 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76249 CAUE DAS NEVES MACHADO
24CC1203 - 5 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 68719476736 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`202 210 214 209 674 599 611 739 793 243 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

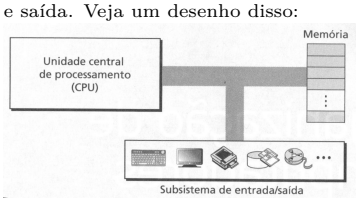
3. Um interpretador do computador acima descrito recebeu o programa
`216 238 236 238 202 441 731 722 696 590 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76256 CLEVERSON LUIZ VOITACH
JUNIOR 24CC1203 - 6 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 68719476736 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
231 208 208 225 223 799 741 249 264 626 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

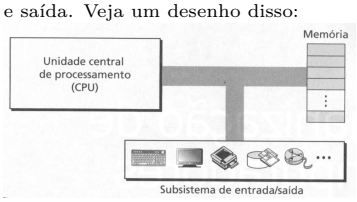
3. Um interpretador do computador acima descrito recebeu o programa
202 237 238 225 692 305 221 710 433 365 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76263 DAVI HONORIO ROSA 24CC1203 - 7
/ _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 67108864 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`230 226 204 238 231 403 670 696 241 445 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

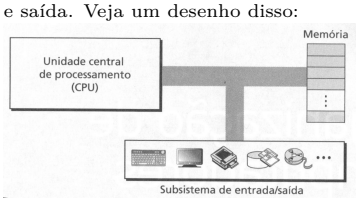
3. Um interpretador do computador acima descrito recebeu o programa
`202 208 207 210 402 627 711 265 326 530 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76270 DORA DOS SANTOS PELLANDA
24CC1203 - 8 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 4194304 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`205 203 207 207 429 750 712 627 745 667 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

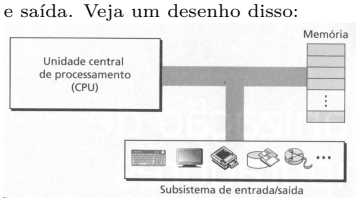
3. Um interpretador do computador acima descrito recebeu o programa
`232 234 222 203 371 479 386 518 640 584 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76287 ENZO KAUAN CARVALHO
24CC1203 - 9 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 1073741824 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
223 201 212 235 350 212 695 686 795 712 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

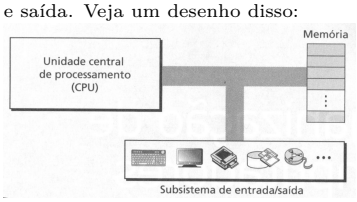
3. Um interpretador do computador acima descrito recebeu o programa
228 237 214 220 225 668 248 337 522 754 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76294 FABRICIO JAWORSKI FERREIRA
24CC1203 - 10 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 34359738368 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`233 205 232 220 450 750 298 795 544 551 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

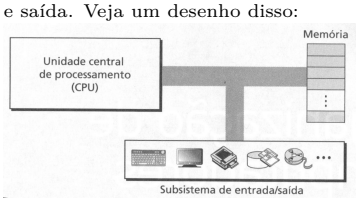
3. Um interpretador do computador acima descrito recebeu o programa
`220 214 218 230 246 505 677 232 348 468 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76306 FELIPE FERNANDES DOS SANTOS
24CC1203 - 11 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 1048576 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`213 226 231 201 740 329 672 427 243 248 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

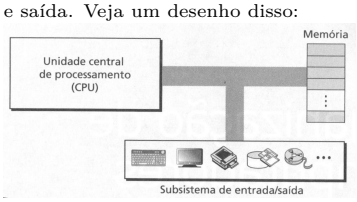
3. Um interpretador do computador acima descrito recebeu o programa
`237 224 215 239 793 775 536 748 353 361 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76313 GABRIEL BORDIGNON 24CC1203 -
12 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 262144 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
208 221 213 240 217 246 271 698 743 515 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

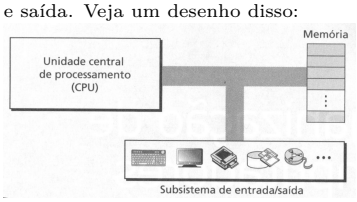
3. Um interpretador do computador acima descrito recebeu o programa
204 221 209 221 748 236 308 323 422 364 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76320 GEOVANNI TEIXEIRA ARAUJO
24CC1203 - 13 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e 2^{30} = 1.073.741.824. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 524288 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
222 240 210 202 579 333 587 444 618 646 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

3. Um interpretador do computador acima descrito recebeu o programa
236 203 228 204 321 562 212 277 396 515 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

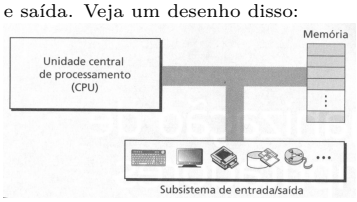
R:_____

203-76320 -

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76337 GUILHERME CAVALER KUSTER
24CC1203 - 14 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 34359738368 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`238 216 236 227 690 417 479 444 468 252 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

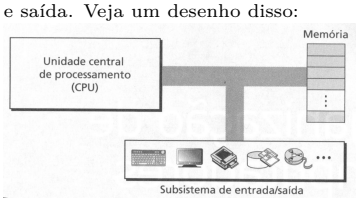
3. Um interpretador do computador acima descrito recebeu o programa
`215 221 204 209 579 676 484 356 766 354 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76344 GUILHERME LUCIANO DE MELO
24CC1203 - 15 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 131072 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`211 224 240 229 582 271 370 512 731 266 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

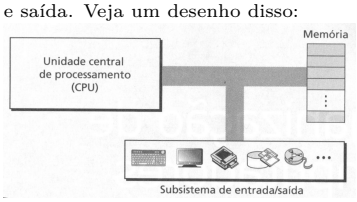
3. Um interpretador do computador acima descrito recebeu o programa
`205 209 209 212 560 255 230 708 427 645 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76351 HENRI KIM PINHEIRO
RODRIGUES 24CC1203 - 16 _____ / _____ /

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 131072 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`201 235 203 239 759 662 239 359 468 652 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

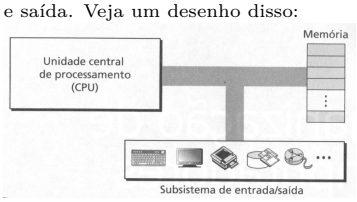
3. Um interpretador do computador acima descrito recebeu o programa
`203 240 202 228 206 433 570 713 390 673 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76368 HENRIQUE NOGUEIRA RIBEIRO
24CC1203 - 17 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 8388608 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
228 222 220 213 612 593 630 305 637 682 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

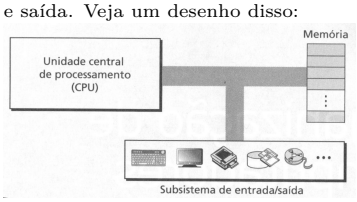
3. Um interpretador do computador acima descrito recebeu o programa
229 239 201 223 349 378 575 209 619 594 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76494 HENRIQUE RANKEL
CRENCHIGLOVA 24CC1203 - 18 _____ /
/ _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 16777216 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`229 230 232 201 774 480 685 552 536 611 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

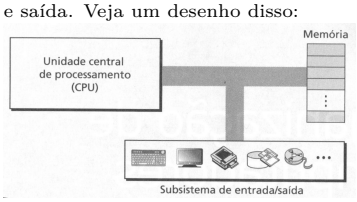
3. Um interpretador do computador acima descrito recebeu o programa
`207 226 240 216 785 621 692 703 642 464 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V: 2.30 76375 JOAO FRANCISCO LEONEL DE SOUSA 24CC1203 - 19 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 4294967296 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
233 203 216 224 757 423 548 229 637 795 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

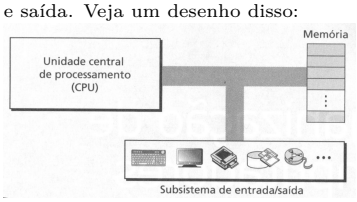
3. Um interpretador do computador acima descrito recebeu o programa
233 208 208 209 444 445 526 213 312 360 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76382 LANA ALESSANDRA GALEGO
VEIGA 24CC1203 - 20 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 1048576 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`237 233 240 231 581 408 772 503 635 674 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

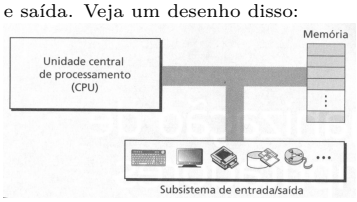
3. Um interpretador do computador acima descrito recebeu o programa
`205 218 237 209 633 536 243 497 450 275 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76399 LORENZO QUIRINO DE SALLES
24CC1203 - 21 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 268435456 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`234 237 225 232 341 441 719 295 612 681 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

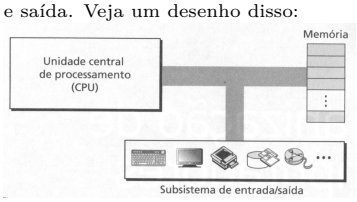
3. Um interpretador do computador acima descrito recebeu o programa
`220 202 223 224 786 252 305 691 471 394 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR – 03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V: 2.30 76401 LUAN TEIXEIRA DIBE 24CC1203 - 22 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 34359738368 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`218 219 230 218 432 335 555 401 615 565 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

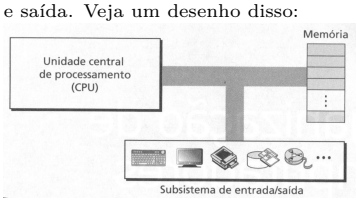
3. Um interpretador do computador acima descrito recebeu o programa
`214 227 220 233 448 670 448 306 235 725 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76418 MURILO DIRCKSEN CAMPOS
24CC1203 - 23 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 68719476736 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`202 219 229 228 790 354 551 294 767 759 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

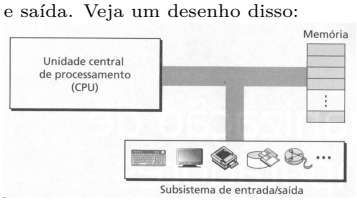
3. Um interpretador do computador acima descrito recebeu o programa
`230 237 220 217 260 336 324 424 255 796 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V: 2.30 76425 OLAVO AUGUSTO ROSSI 24CC1203 - 24 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 536870912 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`218 220 221 213 579 420 224 401 472 477 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

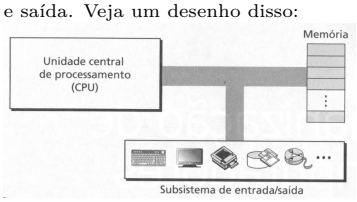
3. Um interpretador do computador acima descrito recebeu o programa
`219 216 237 231 202 239 441 617 448 635 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76432 PAULO GUSTAVO TORRES
24CC1203 - 25 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 8589934592 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`208 217 229 216 721 565 436 697 584 694 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

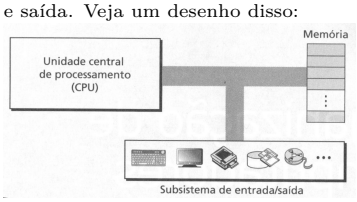
3. Um interpretador do computador acima descrito recebeu o programa
`224 217 206 205 799 469 429 343 543 352 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76449 PEDRO CAPOTE DA CONCEICAO
SOAR 24CC1203 - 26 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 4194304 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
220 226 202 227 693 295 329 739 649 572 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

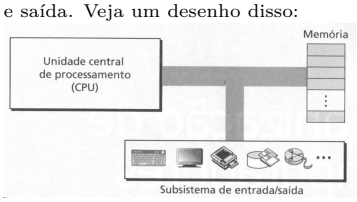
3. Um interpretador do computador acima descrito recebeu o programa
207 206 223 229 564 389 432 449 401 284 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76456 PEDRO HENRIQUE DE SOUZA
24CC1203 - 27 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 268435456 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`219 206 203 233 346 306 687 665 220 672 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

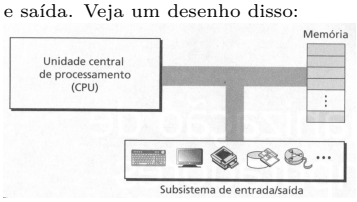
3. Um interpretador do computador acima descrito recebeu o programa
`219 240 222 219 393 796 764 510 291 402 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76463 PEDRO HENRIQUE STAATS
24CC1203 - 28 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador

Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 8589934592 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`217 227 216 227 538 609 306 657 597 508 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

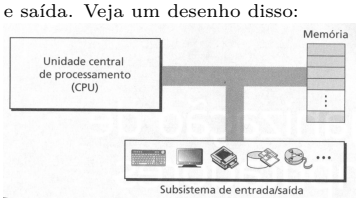
3. Um interpretador do computador acima descrito recebeu o programa
`208 205 217 213 246 303 530 347 615 394 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76470 SARAH FERNANDES CHELIGA
24CC1203 - 29 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 16777216 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
208 228 225 210 208 304 647 654 595 730 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

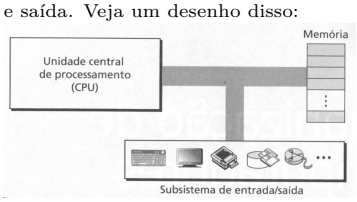
3. Um interpretador do computador acima descrito recebeu o programa
233 228 208 232 648 347 555 487 761 548 100
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

CEP - UP - UTFPR - PUCPr - UFPR –
03/06/2024 - 13:25:06.4
Prof Dr P Kantek (pkantek@gmail.com)
Arquitetura de computadores I VIVXq05b, V:
2.30 76487 WILLIAN IELEM PAVOSKI
24CC1203 - 30 _____ / _____ / _____

Arquitetura de computadores I

Pode-se dividir – grosso modo – um computador em três subsistemas: unidade central de processamento, memória principal e subsistema de entrada e saída. Veja um desenho disso:



Unidade Central de Processamento: Conhecida pela sigla CPU, é o processador principal. É o componente mais caro de um computador. Pode ainda ser subdividido em outros 3 componentes: ULA (unidade lógico-aritmética), unidade de controle e conjunto de registradores. A unidade de controle é responsável por decodificar, interpretar e executar as instruções de máquina (– aqui a grande contribuição de Von Neumann). A unidade aritmético-lógica simplesmente (?!) implementa a aritmética que vimos no ensino fundamental para dentro da máquina, de modo milhões ou bilhões de vezes mais rápido que nós lá no fundamental e mesmo hoje. Finalmente, os registradores são localizações de armazenamento ultra- rápido, independentes, para manipulações aritméticas temporárias. Por exemplo, uma operação de máquina pode ler um conteúdo da memória e guardá-lo num registrador determinado. Outra, pode somar dois registradores e uma terceira pode devolver um registrador para dentro da memória.

Eis alguns exemplos de instruções de máquina
Instruções aritméticas e lógicas: Adição (ADD): soma dois valores e armazena o resultado em um registrador ou na memória. Subtração (SUB): subtrai um valor de outro e armazena o resultado em um registrador ou na memória. Multiplicação (MUL): multiplica dois valores. Divisão (DIV): divide um valor por outro. AND: executa a operação lógica AND bit a bit entre dois valores. OR: executa a operação lógica OR bit a bit entre dois valores. NOT: inverte os bits de um valor.
Instruções de controle de fluxo: Jump (JMP): transfere o controle da execução para outra instrução em um endereço específico. Conditional Jump (JNZ, JE, JG, etc.): transfere o controle da execução para outra instrução apenas se uma condição específica for verdadeira. Call: chama uma subrotina e salva o endereço de retorno para que a execução possa voltar após a subrotina ser concluída. Return: retorna da subrotina para o local de onde foi chamada.
Instruções de acesso à memória: Load (LOAD): copia um valor da memória para um registrador. Store (STORE): copia um valor de um registrador para a memória.
Instruções de entrada e saída: Read: lê dados de um dispositivo de entrada e os armazena em um registrador ou na memória. Write: escreve dados em um dispositivo de saída a partir de um registrador ou da memória.
Instruções especiais: Move (MOV): copia um valor de um registrador para outro. Compare (CMP): compara dois valores e define flags de status que indicam o resultado da comparação. Halt: interrompe a execução do programa.

A memória principal consiste de um conjunto de localizações de armazenamento, cada uma com um identificador único chamado *endereço*. Dados são transferidos de e para a memória em grupos de bits chamados *palavras*. Uma palavra pode ser um conjunto de 8, 16, 32, 64 (e crescendo) bits. Por questões históricas, a palavra de 8 bits é chamada *byte* ou em português *octeto*.

Apesar de programadores usarem nomes para identificar trechos de memória, a nível de hardware, o único identificador válido é o endereço. O número total de localizações (endereços) é chamado espaço de endereçamento. Por exemplo, uma memória com 1GByte e palavras iguais a 1 byte, tem um espaço de endereçamento que varia entre 0 e $2^{30} = 1.073.741.824$. Como os computadores operam em padrões binários, isso significa que neste exemplo serão necessários 30 bits para compor qualquer endereço.

Alguns tipos de memória:
RAM: Random access memory. Compõe a maior parte da memória e pode ter qualquer endereço lido e/ou gravado. É volátil, o que significa que é estável enquanto houver energia.
ROM: Read-only memory, **PROM: Programable ROM**, **EPROM: Erasable PROM**, **EEPROM: Electrically EPROM**.

Além destas memórias, constuma-se trabalhar com uma hierarquia de memórias: na base a memória principal (grande e barata), seguida pela memória cache, e no topo os registradores (caros, logo poucos e ultra-rápidos). O conceito de cache é intermediário e se beneficia de um fenômeno chamado localidade de referência temporal/espacial. Esta regra também é conhecida como Princípio de Pareto ou regra 80-20.

O subsistema de entrada e saída contempla a comunicação do computador com o mundo externo. Este subsistema ainda pode se dividir em 2 grupos: aquele sem armazenamento (teclado, vídeo, mouse, impressora, joystick...) e os com armazenamento (discos, SSDs, pendrives...). Outra maneira de olhar para os E/S com armazenamento é como memórias, já que podem armazenar dados para posterior recuperação. Sendo assim, eles podem entrar na parte inferior da hierarquia de memórias, já que não são voláteis e tem custos muito inferiores aos da memória. Os discos magnéticos estão divididos em setores e trilhas. Contam com acesso aleatório. A menor unidade de transferência disco ⇔ CPU é o bloco. Blocos grandes minimizam o tempo de transferência, mas pioram o desperdício de espaço no disco. A decisão quanto ao tamanho do bloco é tomada em tempo de formatação (que poderia ser comparada ao desenho das marcas de um estacionamento feito antes de ele começar a ser usado). Os SSDs (Solid State Disks) tem mais ou menos as mesmas características dos discos magnéticos, mas por não terem partes móveis estão menos sujeitos a defeitos além de terem tempos de acesso menores.

Há aqui um outro tipo de dispositivo (que rapidamente está perdendo relevância) que são os discos óticos: CD-ROMs, CD-R e os DVDs. Neste caso, o disco é tratado como uma longa tira de informações binárias (tudo a ver com sua origem: a gravação de música).

Interconexão entre subsistemas: Esta abordagem é importante já que os 3 subsistemas (CPU, memória e E/S) precisam trocar informações todo o tempo. A CPU e a memória estão conectadas por 3 barramentos: dados, endereços e controle. O barramento de dados transmite 1 bit por linha e havendo 64 linhas, a transferência se dará à base de 64 bits ao mesmo tempo. O barramento de endereços é usado para acessar a memória. Usa n bits para acessar o endereço 2^n e portanto precisa n linhas. Este valor tem a ver com o espaço de endereçamento. Finalmente o barramento de controle indica o tipo de operação que é desejada a cada ciclo. Novamente deve haver m linhas para comandar 2^m operações distintas. deve-se notar que tanto a memória quanto a CPU são dispositivos 100% eletrônicos, sem nenhum tipo de movimento envolvido.

Já a conexão memória ⇔ E/S é distinta e sobretudo fortemente dependente de QUAL é o dispositivo de E/S acessado. Para uniformizar tais acessos (que são inerentemente distintos) usa-se o conceito de controlador (ou interface). Aqui reinam o SCSI (Small computer system interface), antigo, mas ainda usado em alguns casos, USB (universal serial bus), que conecta muitas coisas (teclados, mouses, discos, pendrives, câmeras...), passando por SATA (serial ATA, usado em discos rígidos e SSDs) e NVMe (Nonvolatile Memory Express, ostentando velocidade e desempenho excepcionais para SSDs). Este último é uma tecnologia ainda em desenvolvimento. ATA é a sigla *Advanced Technoloy Attachment*, marca comercial do lançador a AST em 1986.

A USB pode ser *hot-swappable* (=conectado e removido sem ser necessário reinicializar o computador), e admite o conceito de árvore, sendo o controlador imaginado como o nodo raiz. A versão 2.0 do padrão USB admite até 127 nodos nessa árvore. A USB utiliza um cabo de 4 linhas: duas conduzem eletricidade (+5V e terra) para dispositivos de baixa potência. As outras duas são trançadas (para diminuir o ruído) e conduzem dados, endereços e sinais de controle. A interface USB utiliza quatro tipos de conector físico: o retangular A (conecta o controlador), o quadrado B (conecta o dispositivo), enquanto os mini-A e mini-B conectam dispositivos menores em tamanho. A versão 2.0 define 3 taxas de transferência: 1.5 Mbps, 12 Mbps e 480 Mbps. Os dados são transferidos em pacotes. Já o padrão USB 3.0 apresenta as velocidades de 5, 10 e 20 Gbps.

Interpretador Um certo computador tem 10 registradores e 1000 palavras de RAM. Cada registrador ou cada endereço da RAM pode conter um inteiro de 3 dígitos variando entre 0 e 999. As instruções são codificadas como um inteiro de 3 dígitos e armazenadas em RAM. São elas:

1??	geralmente 100, ordem para parar
2du	coloque o valor u em R_d ($R_d = u$)
3du	adicione u ao R_d ($R_d = R_d + u$)
4du	multiplique o R_d por u ($R_d = R_d * u$)
5du	coloque o valor do R_u no R_d ($R_d = R_u$)
6du	adicione o valor do R_u no R_d ($R_d = R_d + R_u$)
7du	multiplique o R_d pelo valor do R_u ($R_d = R_d * R_u$)
8du	não será usado neste exercício
9du	não será usado neste exercício
0du	não será usado neste exercício

Todos os registradores inicialmente contém 0. O conteúdo inicial da RAM é lido da entrada. A primeira instrução a ser executada está no endereço 0 da RAM. Todos os resultados devem ser reduzidos ao modulo 1000.

Entrada Cada entrada consiste de uma seqüência de números significando o conteúdo da RAM a partir do endereço 0. Endereços não referidos aqui são inicializados com 0. O sinal de fim de dados é um único 0.

Saída Na maratona original, a saída era o número de instruções executadas (incluindo a instrução PARE). Pode-se assumir que todo programa deverá parar. **Entretanto, nesta folha**, a resposta esperada é a soma dos registradores R0+R1+R2+R3. **Exemplo**

```
entrada
299 492 495 399 492 495 399 283 279 689 100
230 202 216 207 349 287 782 100
238 208 222 210 357 496 580 462 303 100
0
saída
0, 13, 21
```

A seguir, uma interpretação dos comandos referentes a primeira linha acima:

```
000.299=coloque 9 em R9          fica R9=0009
001.492=multiplique R9 por 2     fica R9=0018
002.495=multiplique R9 por 5     fica R9=0090
003.399=some 9 em R9             fica R9=0099
004.492=multiplique R9 por 2     fica R9=0198
005.495=multiplique R9 por 5     fica R9=0990
006.399=some 9 em R9             fica R9=0999
007.283=coloque 3 em R8          fica R8=0003
008.279=coloque 9 em R7          fica R7=0009
009.689=some a R8 o valor de R9 fica R8=0002
010.100=Parando...
```

☞ Para você fazer

1. Um computador tem palavras de 1 byte e 131072 bytes de memória. Quantos bits são necessários para endereçar qualquer byte na memória ?

R:_____

2. Um interpretador do computador acima descrito recebeu o programa
`201 227 206 233 671 276 588 287 743 423 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

3. Um interpretador do computador acima descrito recebeu o programa
`218 229 215 201 389 711 547 697 489 358 100`
Após sua execução qual a soma de R0+R1+R2+R3 ?

R:_____

203-76487 -