Arquitetura e organização de computadores

Arquivo: Estudo do microprocessador

Anderson Cottica

Erik R. Yamamoto

Para implementação do microprocessador no projeto da disciplina de Arquitetura e Organização de Computadores, escolhemos a família ISA RISC: MSP430 [MSP], cuja ROM tem comprimento de 15 bits (bits_ROM: 15). A figura abaixo, apresenta o diagrama de bloco da CPU de toda família MSP430x2xx.

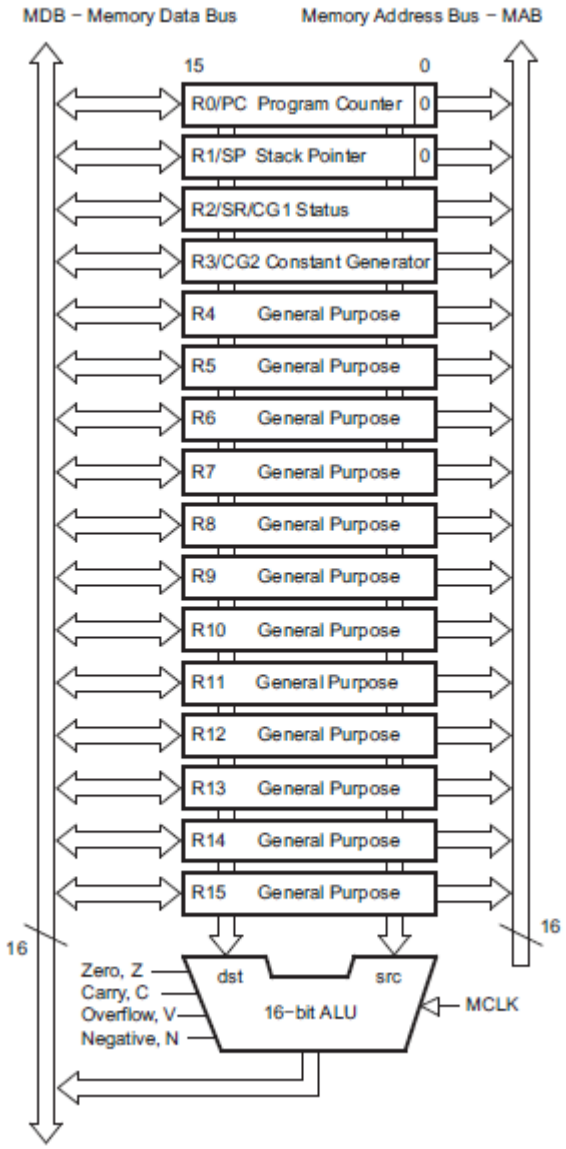


Figura 1 – Diagrama da CPU

Analisando o diagrama é possível perceber que tanto o barramento de endereços, quanto o barramento de memória são constituídos por largura de 16 bits. Da mesma forma, a unidade lógico-aritmética (ULA) do microprocessador é de 16 bits. Além disso, pelo recebimento dos registradores de fonte e de destino, após a realização das operações implementadas, a ULA tem a indicação de *Zero* (Z), *Carry* (C), *Overflow* (V) e *Negative* (N). A seguir, apresentamos a especificação dos componentes da CPU, bem como as instruções de operações a serem realizadas.

>> Registradores:

A CPU é constituída de 16 registradores. Cada registrador possui 16 bits. Os registradores R0, R1, R2 e R3 possuem funções dedicadas. Os registradores R4 a R15 são registradores de uso geral;

>> R0/PC – o registrador R0, juntamente com o PC (*program counter*) são de 16 bits. O PC é responsável por indicar a próxima instrução a ser realizada. Cada instrução usa um número par de bytes (dois, quatro ou seis), e o PC é incrementado em conformidade.

>> R1/SP – da mesma forma que a tamanho de R0, o registrador R1, possui 16 bits. SP (*Stack Pointer*) é o ponteiro de endereço da pilha. Ele é usado pela CPU para armazenar os endereços de retorno das chamadas de sub-rotinas e interrupções.

>> R2/SR/CG1 – R2 e SR (*Status Register*) também possuem o tamanho de 16 bits. O registro de status (SR) é usado como registrador de origem ou destino. CG1 (*Constant Generator 1)* são selecionadas com o registro-fonte de modos de endereçamento.

>>R3/CG2 – Como mencionado anteriormente, a constante 2, tem o tamanho de 16 bits e é usada como registro-fonte de modos de endereçamento.

>>R4~R15 – Todos os demais registradores são de 16 bits. Estes registradores são usados para as diversas operações executadas pela ULA, bem como a movimentação de dados.

>> Endereçamento de Registradores

Antes de apresentar as instruções para operação com registradores, apresentamos, na tabela 2, os modos de endereçamento possíveis para a arquitetura do processador. É possível perceber a característica do endereçamento de dados da arquitetura RISC. Apresentaremos a explicação e detalhamento somente das instruções a serem implementadas na construção de nosso processador, haja visto a grandiosidade de informações presentes no *datasheet*. Para maiores informações a respeito do modo de endereçamento faz-se necessário a consulta ao material integral deste estudo, cujo link está presente nas referências bibliográficas.

As/Ad	Addressing Mode	Syntax	Description
00/0	Register mode	Rn	Register contents are operand
01/1	Indexed mode	X(Rn)	(Rn + X) points to the operand. X is stored in the next word.
01/1	Symbolic mode	ADDR	(PC + X) points to the operand. X is stored in the next word. Indexed mode X(PC) is used.
01/1	Absolute mode	&ADDR	The word following the instruction contains the absolute address. X is stored in the next word. Indexed mode X(SR) is used.
10/-	Indirect register mode	@Rn	Rn is used as a pointer to the operand.
11/-	Indirect autoincrement	@Rn+	Rn is used as a pointer to the operand. Rn is incremented afterwards by 1 for .B instructions and by 2 for .W instructions.
11/-	Immediate mode	#N	The word following the instruction contains the immediate constant N. Indirect autoincrement mode @PC+ is used.

Figura 2 – Modo de endereçamento dos Registradores

>> *Register Mode*: trata-se da movimentação de dados entre registradores. Por exemplo:

```
MOV R10,R11
```

Esta instrução move o conteúdo do registrador R10 para o registrador R11. É importante notar que o conteúdo do registrador R10 não é afetado. Da mesma forma é necessário salientar que R10 é o registrador de origem e R11 é o registrador de destino. Neste modo de endereçamento, o próximo endereço apontado pelo *Program Counter* é incrementado em 2 unidades, isto é, o novo PC é formado de PC(anterior +2)

>> *Indirect register mode*: Move o conteúdo do endereço de origem (conteúdo de R10) para o endereço de destino (conteúdo de R11). Por exemplo:

```
MOV @R10,0(R11)
```

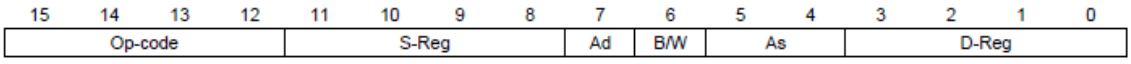
>> *Immediate mode*: este modo de endereçamento é a forma de carregar uma constante imediata em um registrador. Por exemplo:

```
MOV #45h,R9
```

Esta instrução move a constante imediata 45h, para o registrador R9. Desta forma, R9 estará com o valor de 45h para posterior operação.

>> Operação com Registradores

A figura 3 apresenta as operações aritméticas e de manipulação de dados possíveis de serem realizadas entre registradores. Da mesma forma, nesta seção serão detalhadas apenas as operações implementadas em nosso processador.



Mnemonic	S-Reg, D-Reg	Operation
MOV(.B)	src,dst	src → dst
ADD(.B)	src,dst	src + dst → dst
ADDC(.B)	src,dst	src + dst + C → dst
SUB(.B)	src,dst	dst + .not.src + 1 → dst
SUBC(.B)	src,dst	dst + .not.src + C → dst
CMP(.B)	src,dst	dst - src
DADD(.B)	src,dst	src + dst + C → dst (decimally)
BIT(.B)	src,dst	src .and. dst
BIC(.B)	src,dst	not.src .and. dst → dst
BIS(.B)	src,dst	src .or. dst → dst
XOR(.B)	src,dst	src .xor. dst → dst
AND(.B)	src,dst	src .and. dst → dst

Figura 3 – Instrução para operações (Formato I)

Para o caso da instrução de soma entre registradores, usa-se o mnemônico `ADD fonte, destino`. Esta instrução faz a soma do valor guardado nos registradores destino e fonte e reescreve no registrador destino com o novo valor. O valor guardado no registrador fonte não é afetado. Para o caso de subtração, o mnemônico aplicado é o seguinte: `SUB fonte, destino`. O operando de origem é subtraído do operando de destino adicionando a fonte por complemento de 1 do operando e a constante 1. O operando fonte não é afetado. O conteúdo anterior do destino é perdido. A indicação (.B) faz referência a operação para Bytes.

Para realizar os saltos, ou desvios incondicionais, as instruções são simples, conforme a figura 4.

Mnemonic	S-Reg, D-Reg	Operation
JEQ/JZ	Label	Jump to label if zero bit is set
JNE/JNZ	Label	Jump to label if zero bit is reset
JC	Label	Jump to label if carry bit is set
JNC	Label	Jump to label if carry bit is reset
JN	Label	Jump to label if negative bit is set
JGE	Label	Jump to label if (N .XOR. V) = 0
JL	Label	Jump to label if (N .XOR. V) = 1
JMP	Label	Jump to label unconditionally

Figura 4 – Instrução para operações (Formato J)

O mnemônico `JMP Label` faz um salto incondicional para a parte do programa especificado. Este recurso é fundamental quando necessário chamar uma função específica. Já o mnemônico `JC Label` faz o salto para Label quando o bit indicativo de carry é setado.

Os saltos condicionais suportam os *branchs* do programa em relação ao PC e não afetam os bits de status. É possível um intervalo de salto de - 511 a +512 *words* em relação ao valor do PC nas instruções de salto. Os 10 bits menos significativos de offset do PC são tratados como 10 bits assinalados que são duplicados e adicionados ao *Program Counter*, ou seja:

$$PC_{novo} = PC_{anterior} + 2 + PC_{offset} \times 2$$

>> Operação de Comparação

O operando CMP, faz uma comparação entre os registradores de fonte e destino. A sintaxe é: `CMP src,dst`. O operando fonte é subtraído do operando destino. É realizada a adição de 1s complementado o operando fonte mais 1. Os dois operandos não são afetados e o resultado não é armazenado. Somente o status dos bits são afetados. Na prática acontece a seguinte operação: dst + NOT.src + 1 ou, (dst – src). Como exemplo, podemos realizar a comparação entre R5 e R6:

```
CMP R5,R6 ; R5 = R6?
JEQ EQUAL ; YES, JUMP
```

Se os registradores são iguais, o programa continua na *label* EQUAL.

Para as funções de leitura e escrita na memória temos as diretivas

ST: carrega o valor do registrador fonte no endereço dado pelo registrador de destino.

LD: carrega o valor no endereço dado pelo registrador de destino, no registrador fonte.

Referência Bibliográfica

- Datasheet Texas Instrument - MSP430x2xx Family. Disponível em: <http://www.ti.com/lit/ug/slau144j/slau144j.pdf> Acesso em: 06/10/2017