Montador integrado em Javascript para Ramses e Cesar 2024

Manual

Resumo - Este documento apresenta a descrição simplificada do formato de instruções e diretivas para o montador de duas passagens escrito em Javascript e integrado nos simuladores Ramses 2024 e Cesar 2024. O montador usa o mesmo código, e chama os montadores de linha específicos de cada máquina para processamento das instruções. Além dos mnemônicos, o formato das instruções difere principalmente pelos modos de endereçamento de cada arquitetura.

* Neste texto, as palavras rótulo, identificador e símbolo (em tabela de símbolos) podem ser tomadas como sinônimos. Termos diferentes foram usados por coerência com sua função ou com o tratamento normal que os caracteriza em outras disciplinas.

Formato das Linhas - O arquivo fonte em linguagem de montagem consiste em um conjunto de linhas independentes, de instruções da arquitetura ou diretivas de montagem, com o seguinte formato, onde cada um dos 4 elementos é opcional:

[rótulo:] [operação] [operandos] [;comentários]

Rótulo - se presente, consiste em um identificador e terá como valor o endereço de memória onde a instrução ou dado seguinte será colocado. É caracterizado pelo símbolo ':' (dois pontos) ao final, e pode estar separado da operação por mais espaços ou tabulação;

Operação - mnemônico de uma das instruções da arquitetura Ramses ou Cesar, ou uma das diretivas do montador. É separado dos operandos por espaço, tabulação ou vírgula;

Operandos - se a linha atual for um mnemônico, os operandos podem ser um registrador, indicação de um ou dois endereços de operando, registrador e deslocamento, além de códigos de condição, dependendo da arquitetura e da instrução, e estes elementos também estarão separados entre si por espaços, tabulação ou vírgula. Se a linha atual for uma diretiva, ela pode ter um ou mais operandos, com formato especificado adiante.

Registrador - indica um três registradores da arquitetura Ramses, A ou RA, B ou RB, X ou RX, ou um dos 8 registradores da arquitetura Cesar, R0 a R7, sendo que R6 = SP e R7 = PC. O montador não distingue entre caracteres maiúsculos ou minúsculos para todos os nomes, seja de registrador, rótulo, mnemônico ou numeração hexadecimal.

Modos de endereçamento da arquitetura Ramses - um dos 4 modos de endereçamento será usado nas instruções que precisam indicar o endereço de um dos operandos (ou de desvio) na arquitetura Ramses, e têm os seguintes formatos:

direto:	LDR A <i>nn</i>	; indica que o operando está no endereço <i>nn</i>
indireto:	LDR A <i>nn</i> ,I	; indica que o endereço do operando está no endereço <i>nn</i>
imediato:	LDR A # <i>nn</i>	; indica que o operando é o valor <i>nn</i> (endereço é PC+1)
direto:	LDR A <i>nn</i> ,X	; indica que o operando está no endereço <i>nn</i> + valor de RX

Modos de endereçamento da arquitetura Cesar - um dos 8 modos de endereçamento originais ou um dos dois modos derivados (pós-incrementados direto e indireto usando R7) será usado nas instruções que precisam indicar o endereço de operando (ou de desvio) na arquitetura Cesar, e têm os seguintes formatos:

registrador:	R_i	; indica que o operando está no registrador i
reg. pós-incrementado:	$(R_i)+$	; o endereço do operando está no registrador i , e $R_i=R_i+2$
reg. pré-decrementado:	$-(R_i)$	; Faz $R_i=R_i-2$ e o end. do operando é o novo valor de R_i
indexado:	$ddd(R_i)$	; operando está no endereço $ddd +$ valor de R_i
registrador indireto:	(R_i)	; o endereço do operando está no registrador i
pós-incrementado indireto:	$((R_i)+)$	; o endereço do operando está na posição R_i+2 de memória
pré-decrementado indireto:	$-(R_i)$	; o endereço do operando está na posição R_i-2 de memória
indexado indireto:	$(ddd(R_i))$	; end. do operando está na posição de memória $ddd + R_i$
imediatos:	$\#nnn$	; o operando é o valor nnn ; igual a $(R7)+$, nnn
absoluto:	aaa	; o endereço do operando é aaa ; igual a $((R7)+)$, aaa

Onde ddd , nnn e aaa são valores constantes de 16 bits como definido abaixo. O registrador e os modos de endereçamento (com valores) que indicam os operandos, podem estar separados por espaços, tabulação ou vírgula. No modo imediato, o símbolo $\#$ não pode estar separado de nnn nem por vírgula e nem por qualquer tipo de espaço.

Expressão (valor) constante - pode ser um literal numérico ou de caractere, um identificador ou uma expressão com os operadores aritméticos simples (+, -, *, /) entre literais e identificadores (**termos**). Esta expressão entretanto está limitada a ter apenas dois termos (e um operador) por compatibilidade com o montador Daedalus e também por não implementar precedência ou associação com parênteses, o que conflitaria com a sintaxe dos modos de endereçamento. As expressões não permitem espaços, exceto na diretiva EQU, pois espaços são separadores de outros elementos da linha e das instruções.

Literais numéricos - finalmente, os números podem ser representados diretamente por dígitos decimais, ou com dígitos hexadecimais ([0-9A-F]) se precedidos por um de dois prefixos: 'h' ou "0x", que são respectivamente o padrão do montador Daedalus e de linguagens como C/C++/JS.

Literais Caractere - Valores também podem ser especificados com literais de caractere ASCII, como 'A' para 65, 'a' para 96, e aceitam os escapes '\n', '\t', '\r', '\\' e '\ ' para 10, 9, 13, 92 e 39.

Deslocamentos na arquitetura Cesar - Os dados especificados como ddd nos modos de endereçamento indexados do Cesar são valores de 16 bits. Os deslocamentos $dd8$ especificados nas instruções de desvio (*branches*) e na instrução SOB são deslocamentos de 8 bits, em complemento de dois, em relação ao valor do PC ao final da carga da instrução atual, a serem somados (para *branches*) ou subtraídos (para SOB) na execução. No montador, esses valores podem ser dados de duas formas, ou como expressões puramente numéricas, dando um valor absoluto, ou com expressões que acessem algum símbolo (rótulo), sendo que nessa forma o montador calcula a diferença entre o PC 'atual' e esse rótulo destino para codificar a instrução. Exemplos:

BEQ	4	; salta 4 bytes pra frente
BEQ	fim	; salta para o rótulo fim (que deve estar definido em algum lugar)

Diretiva ORG - Especifica que a montagem passa a ser feita a partir do endereço dado como operando, na forma de um termo como descrito acima, para instruções ou dados seguintes.

Diretiva EQU - Define que o identificador usado como rótulo nesta linha não tem o valor do endereço atual de montagem como definido anteriormente, mas assume o valor dado como operando, na forma de expressão, como descrita acima. Serve para definir constantes auxiliares.

Diretivas DB e DW - Criam espaço para um dado de 8 bits (*byte*) ou 16 bits (*word*), com o valor inicial dado como operando, na forma de expressão como descrita acima.

Diretivas DAB e DAW - Criam espaço para um vetor (*array*) de dados de 8 bits (*bytes*) ou 16 bits (*words*), com o valores inicializados em uma de duas formas. Se o operando de DAB ou DAW for um número *n* entre colchetes, reserva *n bytes* (para DAB) ou palavras (para DAW) inicializadas com 0, a partir do endereço de montagem atual. Se o número dado como operando não estiver entre colchetes, interpreta os operandos como uma lista de expressões separadas por espaços, tabulações ou vírgulas, e reserva espaço para tantos dados quantos estiverem nessa lista, inicializando-os com os respectivos valores constantes. Além disso, essas diretivas aceitam inicialização por *strings*, dadas por sequências de caracteres entre aspas *simples* ('). Exemplos:

	ORG	0		; inicia montagem no endereço 0
atraso:	DB	5		; cria uma variável, seu endereço é 0, valor inicial 5
vetor:	DAB	2, 3, 'A', atraso+1		; vetor inicializado com os valores 2, 3, 65, 1
espaco:	DAB	[10]		; reserva uma área de memória de 10 bytes
	ORG	65500		; passa a montagem para o início do visor
display:	DAB	'version 1.0'		; inicializa 11 bytes com essa mensagem escrita

Bytes e Words - *Bytes* ocupam uma posição de memória, e palavras de 16 bits (*words*) ocupam duas posições consecutivas, no formato *Big-Endian*, ou seja, com o *byte* mais significativo antes, ou no endereço menor. Assim, por exemplo, a palavra 0xAABB será armazenada com o valor 0xAA no primeiro endereço e 0xBB no endereço seguinte.

Comentários - Um comentário inicia pelo caractere ';' (ponto-e-vírgula) e vai até o final da linha.

Identificadores - Identificadores são definidos no montador somente na posição de rótulos, tendo o valor do endereço de montagem atual ou o que for especificado pela diretiva EQU.

Montagem em duas passagens - Na primeira passagem o montador calcula e acompanha o tamanho das instruções e dos dados, calcula a posição dos rótulos e valor de todos identificadores, montando uma tabela de símbolos com seus valores. Isso permite que na segunda passagem se possa referenciar símbolos (identificadores) definidos mais abaixo, o que é indispensável para instruções de salto e chamada de sub-rotina definidas mais adiante. Mas a definição de valores dos símbolos e expressões também podem considerar outros símbolos já definidos anteriormente na mesma passagem. Assim, mesmo que cada expressão possa ter apenas um operador aritmético, você pode definir trechos para calcular expressões maiores como no exemplo a seguir:

a:	EQU	'A'+1
b:	EQU	a + 5
y:	EQU	b * 2

Diferenças para o montador Daedalus:

O novo montador procurou ter a maior compatibilidade possível com o montador Daedalus, para evitar que programas desenvolvidos em um deles fossem recusados pelo outro. Mas por coerência e flexibilidade, algumas concessões foram feitas, e então existem construções que são aceitas no novo montador e não aceitas no Daedalus, e vice-versa, as quais são listadas a seguir. Todos recursos foram testados considerando o montador WDaedalus V 1.0.6.1 de set/2017.

- O montador Daedalus trata todas as vírgulas como espaços, e aceita também vírgula entre o rótulo e o mnemônico de instrução (ou diretiva), e o montador JS não aceita;
- O montador Daedalus aceita o caractere # no início de cada número, e portanto aceita expressões como "#end+#1" ou mesmo "end+#1" para modos imediato e absoluto, respectivamente. No montador JS, o caractere # é indicador do modo de endereçamento, e pode aparecer apenas uma única vez no início do operando;
- Na definição dos dados pelas diretivas DB DW DAB e DAW e também como operandos de instrução, as expressões com um operador podem ter ambos operandos identificadores, ou um identificador e um número, mas o primeiro operando precisa ser um identificador no Daedalus. Essa restrição não existe no novo montador JS, que então aceita instruções como "LDR A #5+5" no Ramses ou "MOV #5+5,R1" no Cesar, que não são aceitas no Daedalus;
- O montador Daedalus não aceita a forma literal com escape '\\' para aspas simples;
- O novo montador aceita também os nomes alternativos RA, RB e RX (como são mostrados nos simuladores) para os registradores A, B e X da arquitetura Ramses;
- O novo montador aceita também o prefixo "0x" como formato para números em hexadecimal;
- A implementação atual do novo montador ainda não verifica se os caracteres usados nos identificadores pertencem ao conjunto [A-Za-z0-9_], e portanto aceita qualquer caractere diferente exceto os já usados para outra função (vírgula, dois pontos, #, ;, ...);
- O montador Daedalus considera os nomes dos registradores, todos os mnemônicos e diretivas como palavras reservadas, não sendo permitidos como rótulos. O novo montador não tem essa restrição. Esses nomes não podem ser confundidos porque ocorrem em posições diferentes, como antes do símbolo ':' ou depois dos mnemônicos ou diretivas. Então, embora não seja aconselhável, você poderia usar por exemplo "add" ou "equ" como rótulos e operandos, como no seguinte exemplo para o Cesar:

```

equ:  equ    20           ; define o identificador "equ" com o valor 20
add:  add    #3,equ      ; define o rótulo 'add' com o endereço atual dessa adição
                                ; e usa valor 20 como endereço (modo absoluto) de destino
                                mov    #0+r0,r1      ; move o valor do rótulo R0 para o registrador R1
r0:   jmp    add         ; retorna para o início fazendo um contador sem fim

```

Como regra geral você deve evitar usar qualquer recurso que seja uma exceção ou não seja aceito por algum dos montadores.