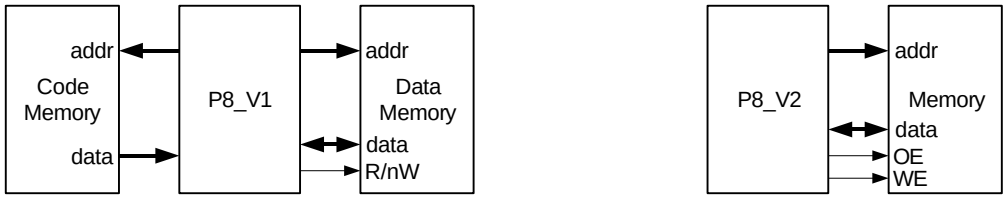
12. P8_V2

O módulo de controlo do P8_V1, não levou em consideração os tempos de reacção dos dispositivos endereçados e escritos. No sentido de corrigir estas deficiências e de aproximar o P8_V1 das arquitecturas reais, vamos construir uma nova versão que denominaremos P8_V2, que continuando a ser uma arquitectura de ciclo único, em relação à anterior apresenta as seguintes alterações:

- Implementação do controlo de modo a produzir as necessárias temporizações no acesso aos dispositivos de memória;
- Passagem do modelo *Harvard* a *Von Neumann*.



a) b)

Figura 12-1 – Diagrama de blocos: Modelo Harvard e *Von Neumann*

O P8_V1 apresenta uma arquitectura Harvard, ou seja, o espaço de endereçamento de código é separado do espaço de endereçamento de dados como mostra a Figura 12-1 a). Como veremos adiante, esta divisão apresenta algumas inconveniências no que diz respeito à gestão do espaço de memória, razão pela qual as arquitecturas comerciais apresentarem arquitectura *Von Neumann*, ou seja, um único espaço de endereçamento para código e dados como se mostra na Figura 12-1 b), assim sendo, iremos introduzir esta alteração no P8_V2 no sentido de a aproximar às arquitecturas reais.
Antes de introduzirmos estas alterações, é importante ter presente o diagrama temporal dos ciclos de leitura (*read*) e de escrita (*write*) dos vários tipos de dispositivos presentes no espaço de endereçamento do CPU, bem como algumas características das máquinas estado síncronas.

12.1 Ciclos de acesso a dispositivos

O espaço de endereçamento do P8_V2 vai ser povoado por vários tipos dispositivo de memória e entrada/saída, todos eles de característica estática. Estes dispositivos apresentam diagramas temporais de leitura e escrita semelhantes, pelo que analisaremos somente os da memória RAM por serem dispositivos de leitura e escrita. As memórias RAM podem apresentar uma de duas formas de controlo:

- Um sinal **R/nW** que selecciona leitura ou escrita e outro **CE** que inibe ou desinibe a acção seleccionada.
- Um sinal **OE** para accionar a leitura e outro **WE** para accionar a escrita, sendo estes sinais activados em exclusão. Para controlo de consumo de energia e facilitar a concatenação de vários dispositivos, dispõe de um sinal **CE** para inibição e desinibição do dispositivo.

As memórias RAM disponíveis no mercado, permitem ser configuradas para as duas formas de controlo, no entanto, por razões que adiante estudaremos, iremos adoptar a segunda forma, o que levará a adicionar dois sinais ao CPU, um **RD** para controlo da leitura e outro **WR** para escrita.

Dados os tempos de propagação e reacção dos vários elementos que compõem a RAM (ver capítulo 8), os ciclos de leitura e escrita, terão que respeitar os tempos associados à descodificação de endereços e de *Set Up Time* e *Hold Time* dos registos constituintes. Na Figura 12-2 e **Figura 12-3** são apresentados os diagramas temporais de um ciclo de leitura e escrita com as siglas dos tempos normalmente utilizadas pelos fabricantes deste tipo de dispositivo.

12.1.1 Ciclo de leitura

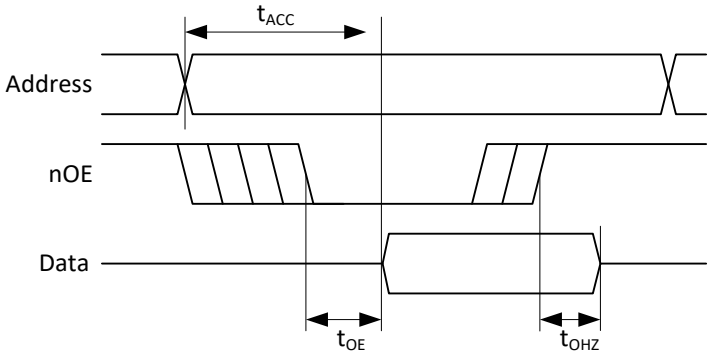


Figura 12-2 – Ciclo de leitura

t_{ACC} (*time ACCess*) tempo máximo para acesso à informação.

t_{OE} (OE to Output *valid*) tempo que media entre a activação do sinal nOE e a presença no data bus, de informação estável em baixa impedância posta disponível pelo dispositivo.

t_{OHZ} (*Output Disable to Output High* Z) tempo que media entre a desactivação do sinal nOE e libertação do data bus por parte do dispositivo.

12.1.2 Ciclo de escrita

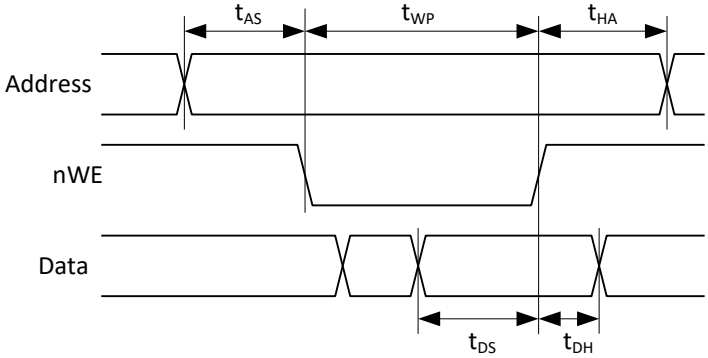


Figura 12-3 – Ciclo de Escrita

t_{AS} (*Address Setup Time*) intervalo de tempo mínimo a respeitar entre o estabelecimento de um endereço e a activação do sinal nWE.

t_{WP} (*Write Pulse Width*) duração mínima do sinal nWE.

t_{DS} (*Data Setup Time*) intervalo de tempo mínimo a respeitar entre o estabelecimento de informação valida no data bus e a desactivação do sinal nWE.

t_{DH} (*Data Hold Time*) tempo mínimo durante o qual ainda se torna necessário manter os dados estáveis no bus após ter terminado o sinal de nWE.

t_{HA} (*Hold Address*) tempo mínimo durante o qual ainda se torna necessário manter o endereço estável após ter terminado o sinal de nWE.

Como se pode observar no diagrama temporal do ciclo de escrita, ver Figura 12-3, só é possível activar o sinal WE após a estabilização do endereço. Por outro lado é necessário manter o endereço e os dados estáveis depois de desactivar o sinal WE. No caso do P8_V1, esta especificação da memória de dados não é respeitada. Quanto ao ciclo de leitura, o comportamento do P8_V1 produz um conflito no bus de dados, pois quando passa de leitura a escrita não espera que o dispositivo de memória liberte o bus de dados (t_{OHZ}).
Se observarmos o diagrama temporal do ciclo de leitura, ver Figura 12-2, não existe nenhum problema se variarmos o endereço quando o sinal RD está activo, no entanto, como veremos mais adiante, o espaço de memória do CPU vai ser povoado por outro tipo de dispositivos além da memória, como sejam dispositivos para entrada e saída de dados para o exterior do sistema. Se imaginarmos um dispositivo que está a receber uma cadeia de caracteres, e que cada vez que é lido pelo CPU entrega um carácter da cadeia recebida, facilmente se percebe que a leitura descontrolada de um endereço de memória é neste caso nefasta. Por esta razão o CPU deverá gerar tempos de guarda em torno do sinal RD relativamente à variação dos endereços.

12.2 Temporização dos sinais de saída

Dado que é necessário gerar temporizações para os vários sinais, leva a que a implementação do módulo de controlo do P8_V2, recorra a uma máquina de estados síncrona.
Como já foi estudado anteriormente, nas máquinas de estado síncronas, as evoluções de estado dão-se no momento da transição do sinal de *clock*, com o intervalo mínimo de um período do sinal *clock*. Os tempos dos sinais de saídas que estas máquinas podem gerar estão assim limitados ao período deste *clock*. No entanto se dispusermos de um sinal em contra fase com o *clock*, poderemos produzir acontecimentos com duração de meio período de *clock*. Assim sendo e, como acontecia no P8_V1, iremos gerar dois sinais $\phi 1$ e $\phi 2$, em contra fase com *duty cycle* de 50%, sendo $\phi 1$ para o módulo de controlo e $\phi 2$ para o módulo funcional.

12.3 Ciclos de acesso a memória do P8_V2

Como já vimos anteriormente, no P8_V1, o processamento é concretizado pela sucessão das acções *fetch execute*, existindo meio *clock* entre a leitura da instrução e a sua execução. Mantendo a mesma lógica, ou seja, após a leitura de uma instrução (*fetch*), passamos à execução e assim sucessivamente e, considerando que meio *clock* é suficiente para os tempos t_{OHZ}, t_{ACC}, $t_{AH,}$ t_{AS} e t_{DH}, poderemos pensar num diagrama temporal para a actividade dos *buses* de endereços, dados e controlo do CPU como é mostrado na Figura 12-4. Este diagrama temporal tem em conta os diagramas temporais característicos dos dispositivos de memória.

A Figura 12-4 apresenta três exemplos: um ciclo de *fetch* para uma instrução que tem execução interna ao CPU; o *fetch* de uma instrução cuja execução implica a leitura de um dado da memória; o *fetch* de uma instrução cuja execução implica a escrita de um dado na memória.

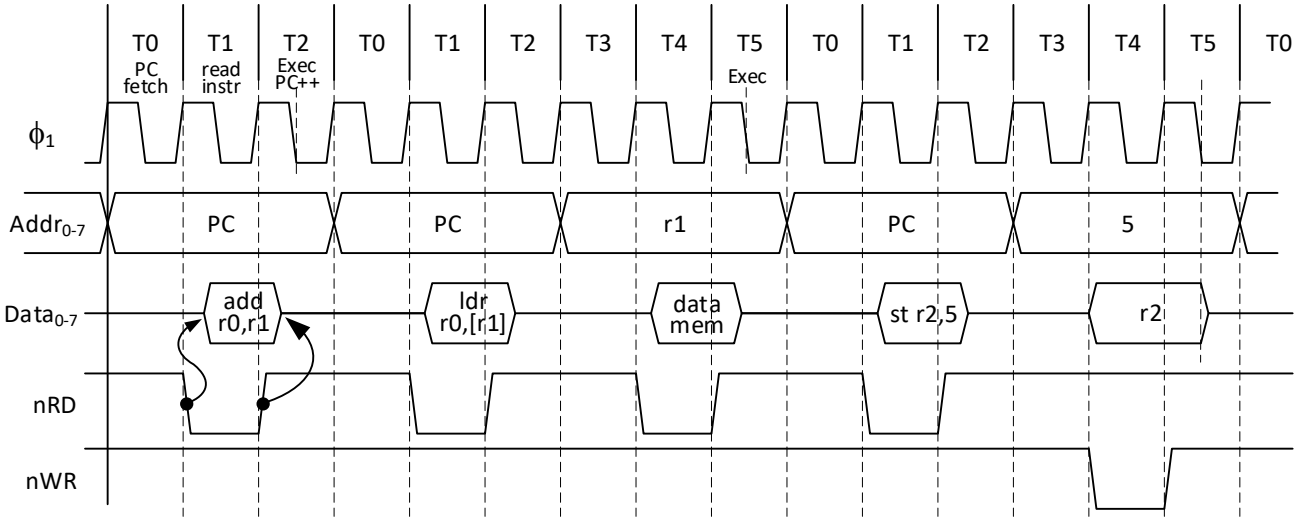


Figura 12-4 – Diagrama temporal de actividade dos *buses*

Estrutura do P8_V2

A nova especificação do P8_V2 implica alterações ao módulo funcional do P8_V1 como a seguir se descreve:

- Como se pode observar na Figura 12-4, no ciclo máquina de leitura de memória (*fetch* ou *execute*), o sinal RD fica activo durante um período de *clock*. Terminado o sinal de RD o CPU vai consumir a informação lida a meio do próximo *clock*, ou seja, com ϕ2. Assim sendo é necessário preservar esta informação para além do RD o que implica adicionar um registo do tipo transparente LATCH, denominado por MBR (*Memory Buffer Register*). A introdução do registo MBR como elemento de recepção dos dados vindos do exterior, tem também como vantagem apresentar ao exterior um único elemento da estrutura, definindo assim, uma impedância característica de entrada do bus de dados do CPU;

- No P8_V2 o bus de endereços é estabelecido pelo conteúdo do registo PC na fase *fetch* e na fase execute, através de um multiplexer, pelo parâmetro **direct3**, ou pelos registos rb. Como se pode observar na Figura 12-4 o bus de endereço é estabelecido em $\phi 1$ e o módulo de controlo que estabelece os bits de selecção do multiplexer transita em $\phi 1$. No sentido de assegurar a estabilidade do bus de endereços ao longo de todo o ciclo é necessário adicionar um registo *latch* denominado MAR (*Memory Address Register*) e no qual, no início de cada ciclo de acesso à memória, é registado o endereço que se pretende aceder para leitura ou escrita;
- Na fase *fetch*, o processador lê da memória a instrução a executar. Como a instrução tem que permanecer na entrada do módulo de controlo até ao momento da execução, é necessário adicionar um registo para armazenar a instrução lida da memória. Este registo normalmente denominado por IR (*Instruction Register*), assegura a estabilidade dos vários sinais durante a fase de execução e da preparação do próximo valor do registo PC;
- Ao registo PC é adicionado uma entrada de *Enable* para condicionar a sua evolução a dois momentos: na fase *fetch* para incrementar de 1, e na fase execute para as instruções de *branch*.
- A acção de *Reset*, embora com início assíncrono têm finalização síncrona, A acção de *reset* tem a duração mínima de um *clock*;

12.4 Diagrama de blocos do P8_V2

Com as modificações assim propostas obtemos o diagrama de blocos do P8_V2 mostrado na Figura 12-5.

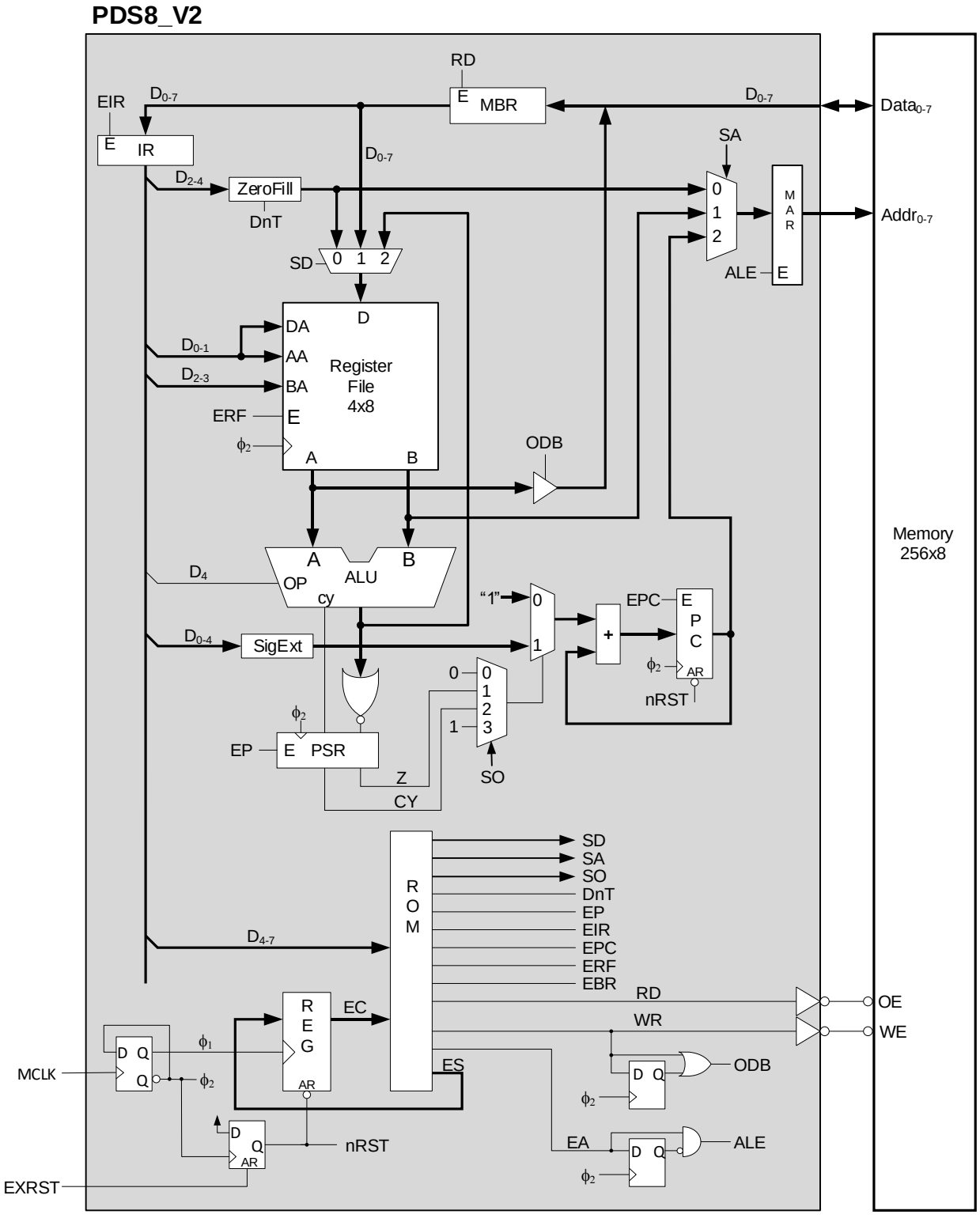


Figura 12-5 – Diagrama de blocos do P8_V2

Como podemos observar na Figura 12-5, para controlo do sinal ODB, foi necessário adicionar um *flip-flop* sincronizado com $\phi 2$, no sentido de assegurar que no ciclo de escrita o bus de dados só é desactivado meio *clock* depois de desactivado o sinal de WR.

Os registos constituintes da estrutura têm a seguinte especificação:

- O *file register* e os registos PSR e PC, são síncronos (*edge trigger*) com controlo de *Enable*;
- Os registos MBR, MAR e IR, são do tipo transparente LATCH;

Como se pode observar na Figura 12-5 todos os registos síncronos são afectados na transição descendente de *clock* ($\phi 2$), enquanto os vários sinais de selecção são estabelecidos na transição ascendente ($\phi 1$).

Descrição dos sinais de entrada:

A activação da entrada **EXRST** põe o registo PC, e o registo de estado corrente, associado ao controlo, com valor zero.

Descrição dos sinais de controlo:

- **SO** (*Select Offset*) determina se o próximo *fetch* se realiza no endereço dado por PC+1, ou em PC mais o parâmetro offset5 contido na instrução;
- **SD** (*Select Data*) selecciona qual a informação a carregar no *register file*. Para a instrução **`mov rd,const2`** selecciona a constante que é parâmetro da instrução. Nas instruções **`ldr`** selecciona o valor que está a ser lido da memória de dados. Nas instruções de processamento, selecciona o resultado da ALU.
- **DnT** selecciona se a constante a ser expandida para oito bits é de Dois ou Três bits.
- **SA** (*Select Address*) selecciona qual o parâmetro que estabelece o endereço. No caso de **`ldr/str direct`**, selecciona o parâmetro contido na instrução. Caso o modo de endereçamento seja indirecto, selecciona o registo rb. No início da fase *fetch* selecciona o registo PC.
- **ERF** (*Enable Register file*) permite a escrita no *register file*.
- **EP** (*Enable* PSR) permite a escrita das *flags* no registo PSR aquando das operações aritméticas.
- **EPC** (*Enable* PC) permite a escrita no registo PC.
- **EA** (*Enable* Address) controla a escrita no registo MAR aquando da preparação do endereço.
- **ALE** (Address *Latch Enable*) sinal com duração de meio *clock* que regista no MAR o endereço do ciclo de acesso.
- **EIR** (*Enable* IR) permite a escrita no registo IR.
- **RD** (*ReaD*) controla a leitura da memória, no sentido de informar o dispositivo endereçado que deve colocar em baixa impedância os dados a serem lidos pelo CPU.
- **WR** (*WRite*) controla a escrita na memória.
- **ODB** (Output Data Bus), dado que o bus de dados é bidireccional, este sinal controla a impedância de saída do bus de dados.
- **EC** (Estado Corrente) vector que representa o estado corrente da máquina de controlo.
- **ES** (Estado Seguinte) vector que representa o estado seguinte da máquina de controlo.

Módulo de controlo

O módulo de controlo apresenta o comportamento descrito pelo ASM da Figura 12-6. Este módulo controla os vários sinais da estrutura, no sentido de executar as várias instruções, e gerar os vários sinais do bus com o encadeamento estabelecido no diagrama temporal da Figura 12-4.

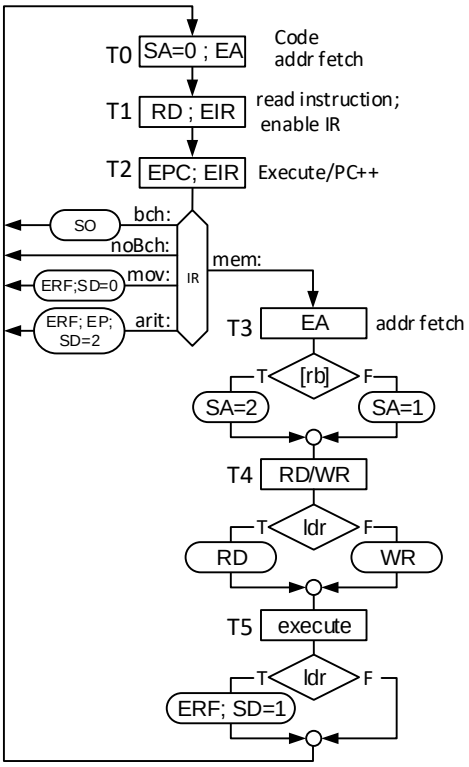


Figura 12-6 – *ASM-chart* – módulo de controlo

Estado 0: Estado inicial do controlo, atingido por activação da entrada externa EXRST e aqui permanecendo enquanto EXRST se mantiver activa. O controlo, neste estado activa o sinal EA, e selecciona a entrada 2 do multiplexer do MAR para iniciar MAR com o valor do PC que foi colocado a zero.

Estado 1: Neste estado é realizada a leitura da instrução, para tal o CPU activa o sinal de RD para ler da memória a instrução a executar.

Estado 2: O estado 2 corresponde ao estado de execução. Neste estado, se a instrução presente no IR é uma instrução **mov** activa ERF e selecciona no multiplex do *register file* a entrada 0. Se a instrução é aritmética activa ERF, EP e selecciona a entrada 2 do multiplex do *register file*. Para todas as instruções activa EPC e se a instrução é *branch* e a condição é verdadeira, activa SO.

Estado3: Preparação do endereço para leitura ou escrita da memória de dados.

Estado4: Se a instrução é **ldr** activa o sinal RD, caso contrário activa o sinal WR.

Estado5: Se a instrução é **ldr** realiza a fase de execução, activando EFR e seleccionando no multiplexe do *register file* a entrada 1.

12.4.1 ROM de descodificação

Na Tabela 12-1 está representada a programação da ROM de descodificação, na qual podemos observar o comportamento das várias saídas do controlo e a evolução de estados. Na implementação do controlo do P8_V2, iremos utilizar a ROM para conter o micro-code. Dado a característica do ASM do módulo de controlo e o reduzido número de estados, a geração de estado seguinte não recorre a um sequenciador, sendo o estado seguinte determinado pela ROM de micro-code. Por esta razão a dimensão da ROM será de 128x17, pois tem como bits de endereço, os 4 bits mais significativos do código da instrução, mais os 3 bits que codificam o estado corrente.

	I_7 I_6 I_5 I_4	EC_{0-2}	ES_{0-2}	DnT	SD	SA	SO	EA	EPC	ERF	EP	EIR	RD	WR
	- - - -	0	1	-	-	2	-	1	0	0	0	0	0	0
	- - - -	1	2	-	-	-	-	0	0	0	0	1	1	0
ldr/str	0,1,2,3,4,5	2	3	-	-	-	0	0	1	0	0	0	0	0
mov rd,imm2	6	2	0	1	0	-	0	0	1	1	0	0	0	0
cmp	7	2	0	-	2	-	0	0	1	0	1	0	0	0
add/sub	8,9	2	0	-	2	-	0	0	1	1	1	0	0	0
bzs offset5	a,b	2	0	-	-	-	1	0	1	0	0	0	0	0
bcs offset5	c,d	2	0	-	-	-	2	0	1	0	0	0	0	0
b offset5	e,f	2	0		-	-	3	0	1	0	0	0	0	0
ldr/str direct3	0,1,4,5	3	4	0	-	0	-	1	0	0	0	0	0	0
ldr/str [rb]	2,3	3	4	-	-	1	-	1	0	0	0	0	0	0
ldr	0,1,2	4	5	-	-	-	-	0	0	0	0	0	1	0
str	4,5,3	4	5	-	-	-	-	0	0	0	0	0	0	1
ldr	0,1,2	5	0	-	1	-	-	0	0	1	0	0	0	0
str	4,5,6	5	0	-	-	4	-	1	0	0	0	0	0	0

EC_{0-2} – Estado Corrente **ES_{0-2}** – Estado Seguinte **I_{3-7}** – bits do *Instruction Register*

Tabela 12-1 - Programação da ROM de descodificação