

Aula 23

Exceções e interrupções:

Gestão de exceções e interrupções

Datapath multiciclo com suporte para as exceções provocadas por *overflow* e *opcode* desconhecido

A unidade de controlo com suporte para as exceções

Exemplos da metodologia de instanciação de exceções baseadas no datapath multiciclo analisado

Bernardo Cunha, José Luís Azevedo, Arnaldo Oliveira e Silva

Exceções e interrupções

O que são exceções?

Exceções são eventos que *break the sequence*, alteram o fluxo normal de execução do programa. Existem duas fontes de exceções:

Eventos **internos** ao CPU, inesperados e decorrentes da execução das próprias instruções

Overflow aritmético ou **fetch** de uma instrução com **opcode desconhecido** para a unidade de controlo são dois exemplos de exceções

Eventos **externos** ao CPU que surgem assincronamente com o funcionamento deste. Estes eventos são normalmente designados por **interrupções** (**Interrupts**). Tal como as exceções provocam uma alteração do fluxo de execução do programa

As fontes de interrupção são normalmente dispositivos que necessitam de ser atendidos (por exemplo o teclado quando uma tecla é premida)

Exceções e interrupções

Alguns eventos deste tipo podem ser observados na tabela seguinte.

Tipo de evento	Proveniência	Terminologia MIPS
Pedido de atenção por um dispositivo de I/O	Externo	<i>Interrupt</i>
Invocação do sistema operativo a partir do programa	Interno	Exceção
Overflow aritmético	Interno	Exceção
Tentativa de execução de uma instrução desconhecida	Interno	Exceção
Problema de funcionamento do hardware	Qualquer	Exceção ou interrupt

A detecção das condições de exceção e de interrupções podem condicionar a eficiência do CPU ao influenciar diretamente o período do sinal de relógio. Assim, fundamental considerar estas condições desde a fase inicial do projecto da unidade de controlo.

Exceções e operações básicas efectuadas pelo

Há duas operações básicas que têm de ser efectuadas pelo processador sempre que ocorre uma exceção:

1. Copiar o endereço da instrução que gerou a exceção para um registo a isso destinado. No **MIPS** esse registo chama-se **EPC** (Exception Program Counter)
2. Transferir a execução para a **rotina de tratamento da exceção** (carregando no *Program Counter* um endereço previamente definido)

A rotina de tratamento da exceção, para além de guardar o endereço da instrução que gerou a exceção, deverá determinar a causa, por forma a agir em conformidade

Exceções e Identificação da origem / causa

Para a identificação da causa da exceção e do tratamento, os dois mecanismos mais utilizados são:

A existência de um registo especial no qual é armazenado um código que identifica a causa da exceção *interrupt*. Esse registo pode ser acedido pela rotina de tratamento para determinar a causa

Este é o mecanismo usado pelo MIPS

O registo é designado **Cause Register**

A vectorização das exceções

Nesta estratégia, cada tipo de exceção (desencadeia, directa ou indirectamente, a passagem da execução do programa para um endereço distinto (e.g. exceção X causa transferência da execução do programa para o endereço A; exceção Y transfere a execução do programa para o endereço...))

Exceções e processamento na rotina de atendimento

A rotina de tratamento da exceção pode realizar várias tarefas possíveis:

Se a exceção corresponde a um evento que compromete a continuidade da execução do programa do utilizador, deve terminar a execução do mesmo (e.g. acesso a um endereço não alinhado)

Providenciar um serviço ao programa do utilizador

Tomar uma medida de reparação do evento que gerou a exceção (quando seja possível)

Retornar ao programa do utilizador, caso a exceção seja crítica, usando para isso a informação guardada no EPC

Exceções integradas no ciclo do MIPS

Na análise que iremos fazer a seguir, vamos considerar dois tipos de exceções:

Exceções geradas pela tentativa de execução de uma instrução inválida (campo *OPCODE* desconhecido da unidade de controlo)

Exceções geradas pela ocorrência de uma operação aritmética

O mecanismo para atendimento de *interrupts* é semelhante. Há, no entanto, uma diferença fundamental:

Exceções A instrução que gerou a exceção não é executada, e a execução passa de imediato para a rotina de tratamento da exceção

Interrupções A passagem para a rotina de tratamento da interrupção só acontece quando for concluída a instrução que estava a ser executada no momento em que a interrupção surge

Exceções integradas no ciclo do MIPS

Para suportar a gestão dos dois tipos de exceções, o nosso *datapath* multi-ciclo precisa de ser ligeiramente alterado. Essas alterações podem ser sintetizadas da seguinte maneira:

Criação de dois novos registos: **EPC** e **Cause Register**. No caso deste último, apenas um bit será usado para identificar a causa

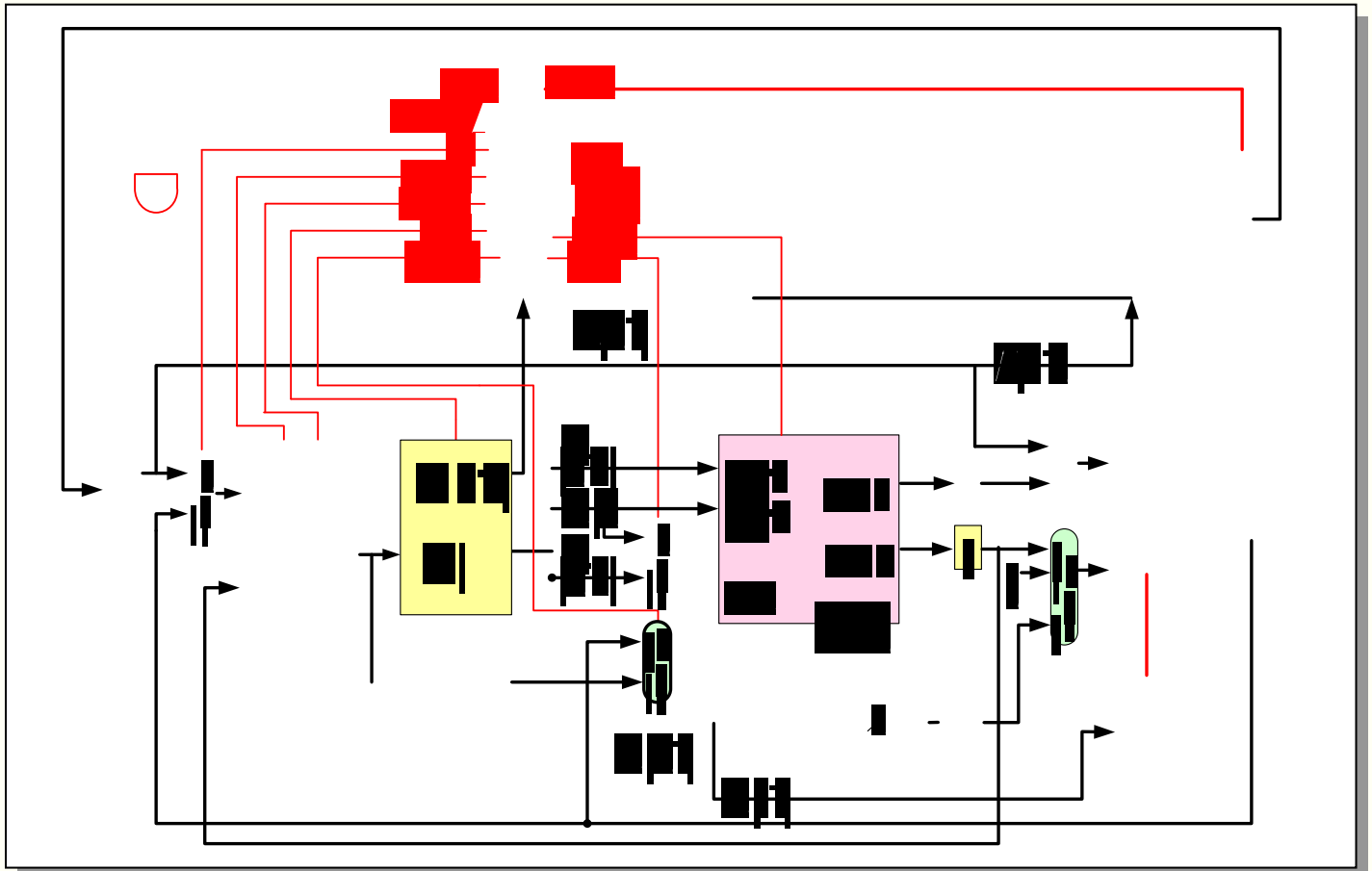
O valor 0 indicará uma instrução desconhecida

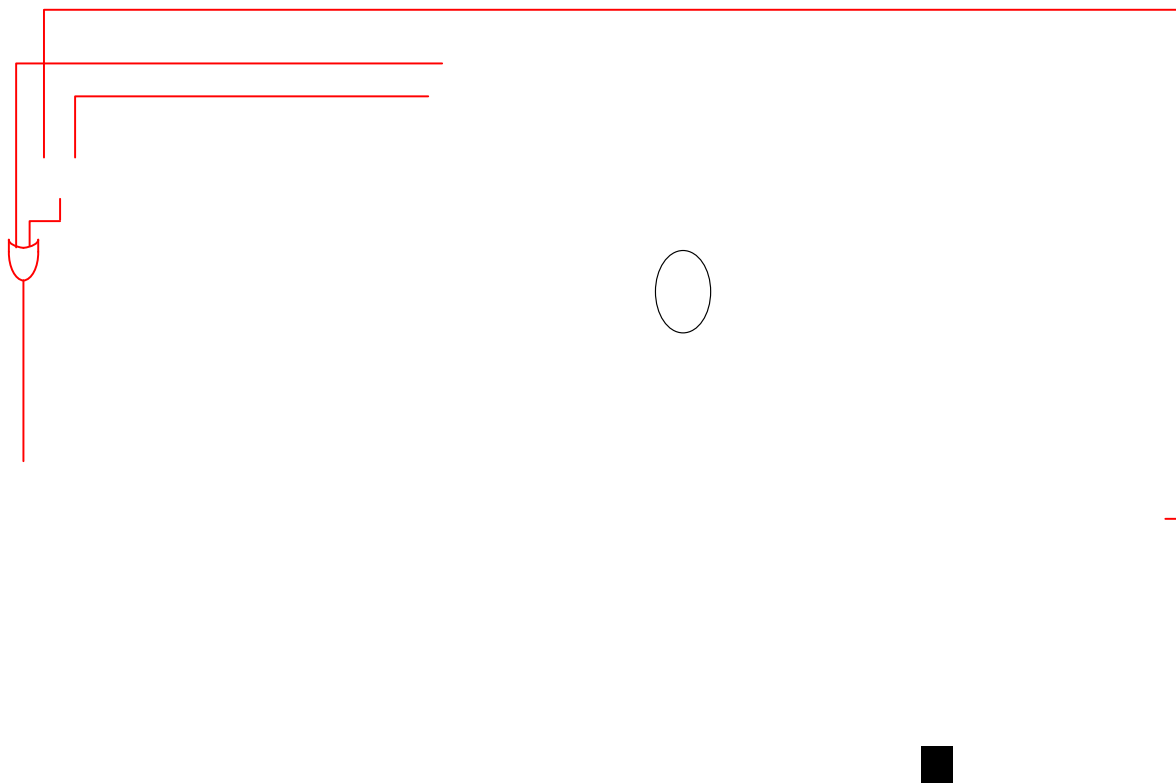
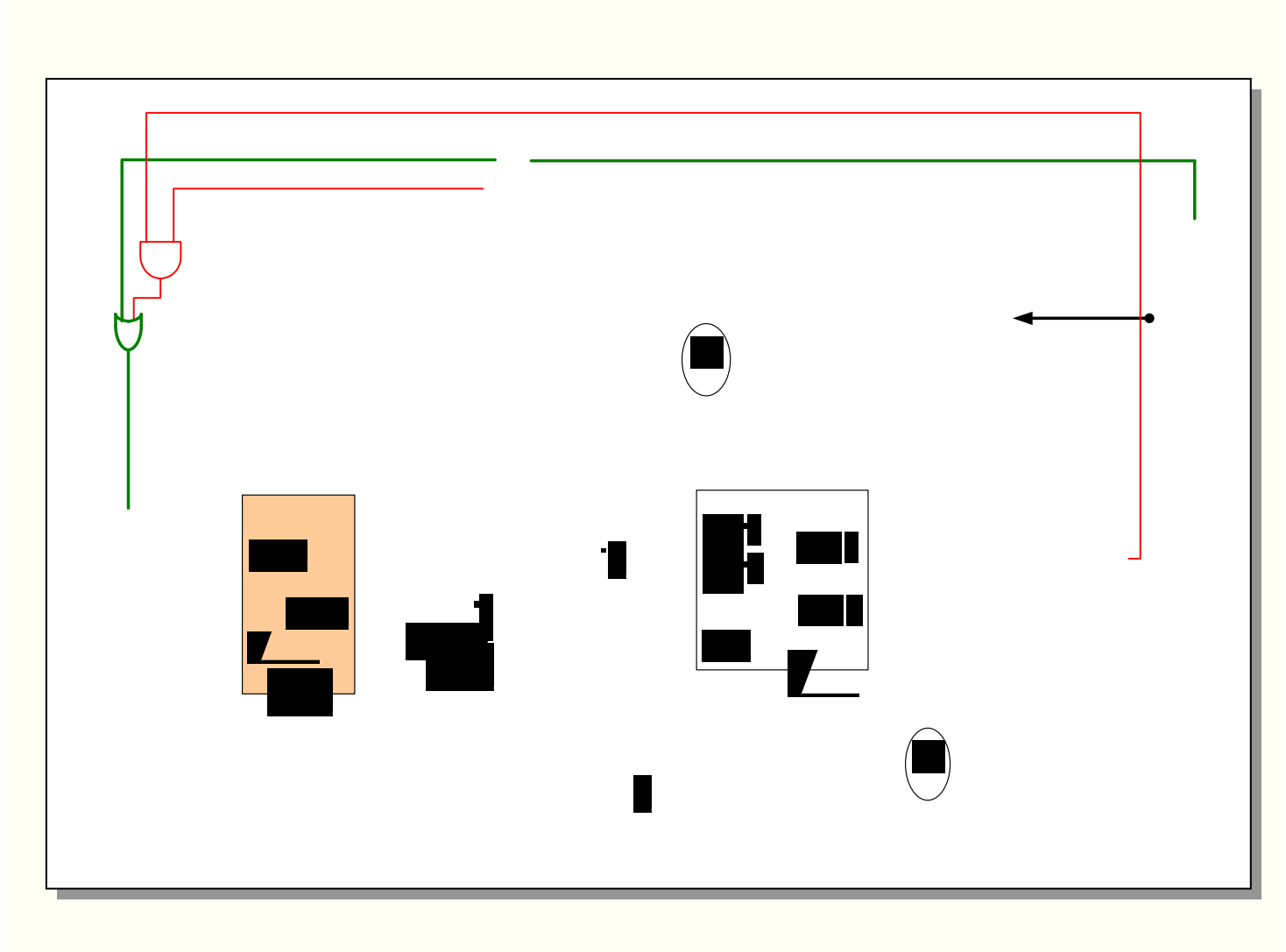
O valor 1 indicará a ocorrência de um overflow

Criação de um conjunto suplementar de sinais que permitam operar sobre aqueles dois registos

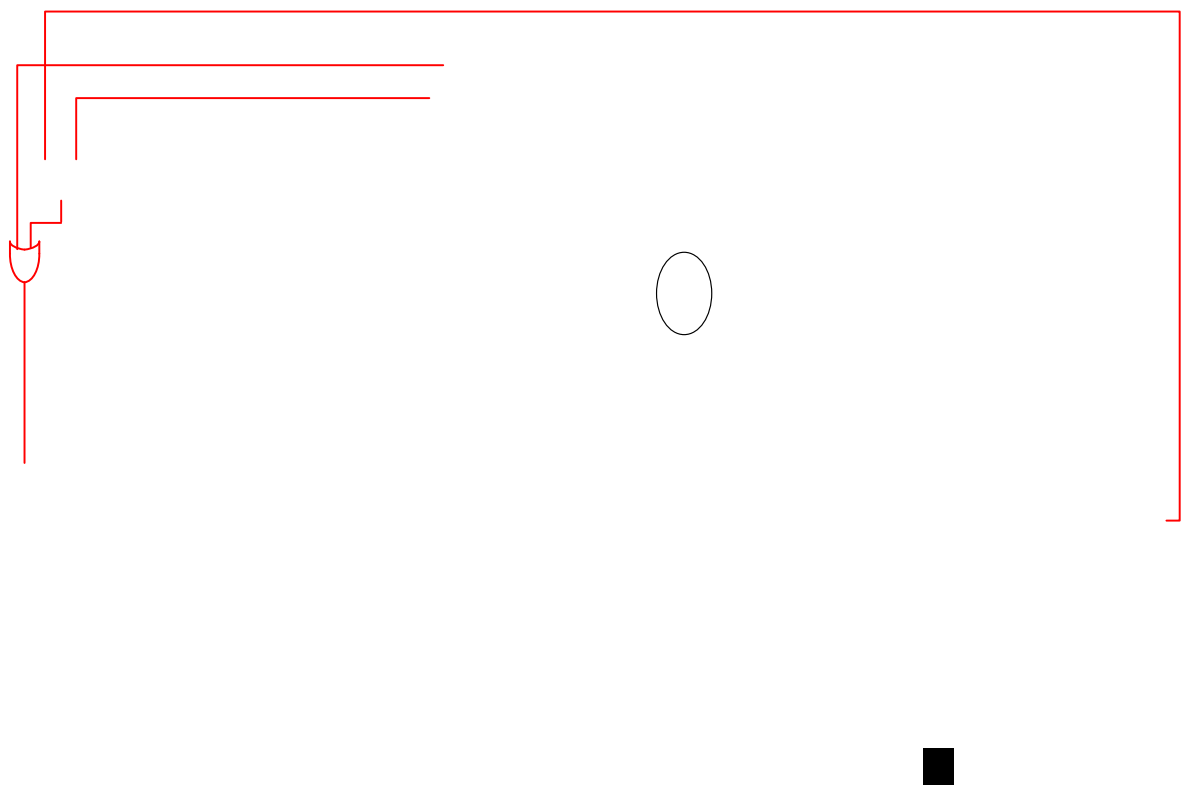
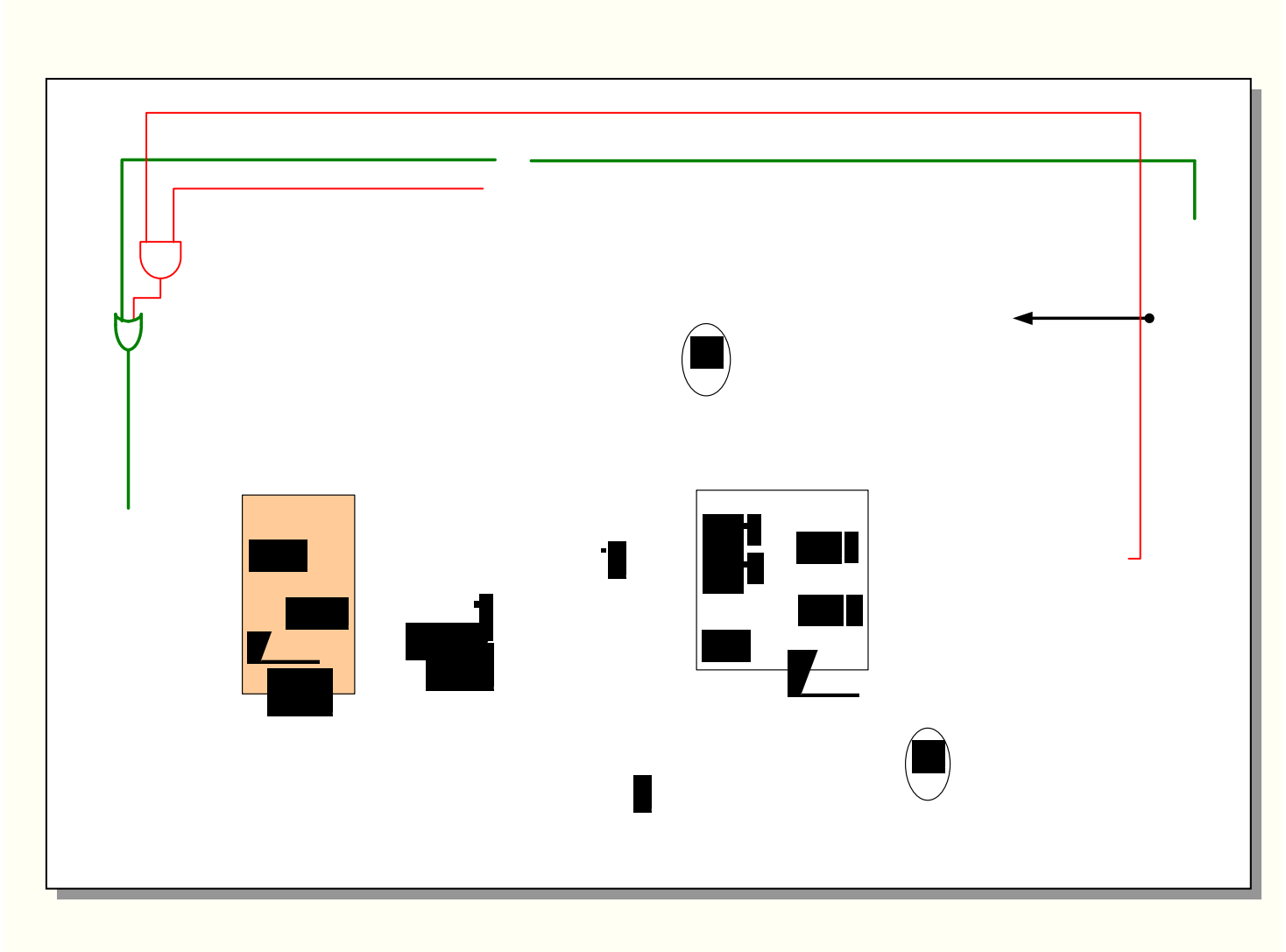
Criação de um mecanismo que permita escrever no **EPC** da primeira instrução da rotina de atendimento à exceção (isto é, esse endereço é **0x8000180**)

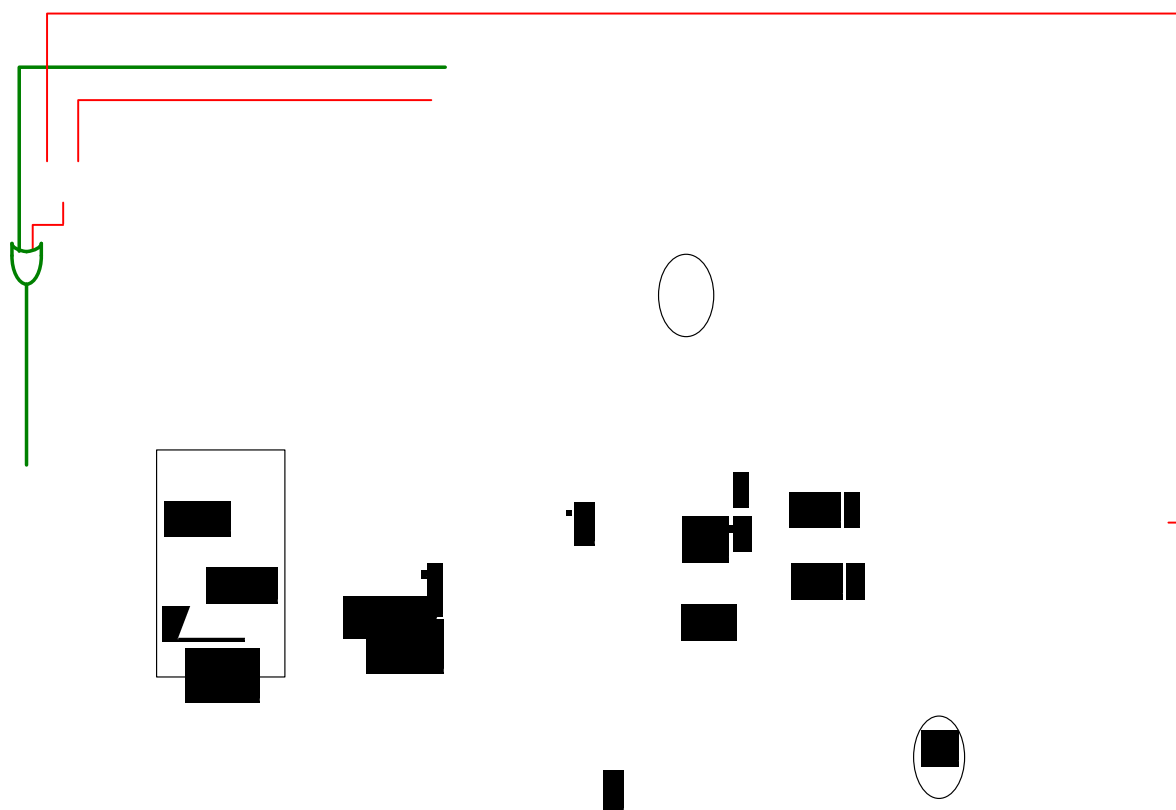
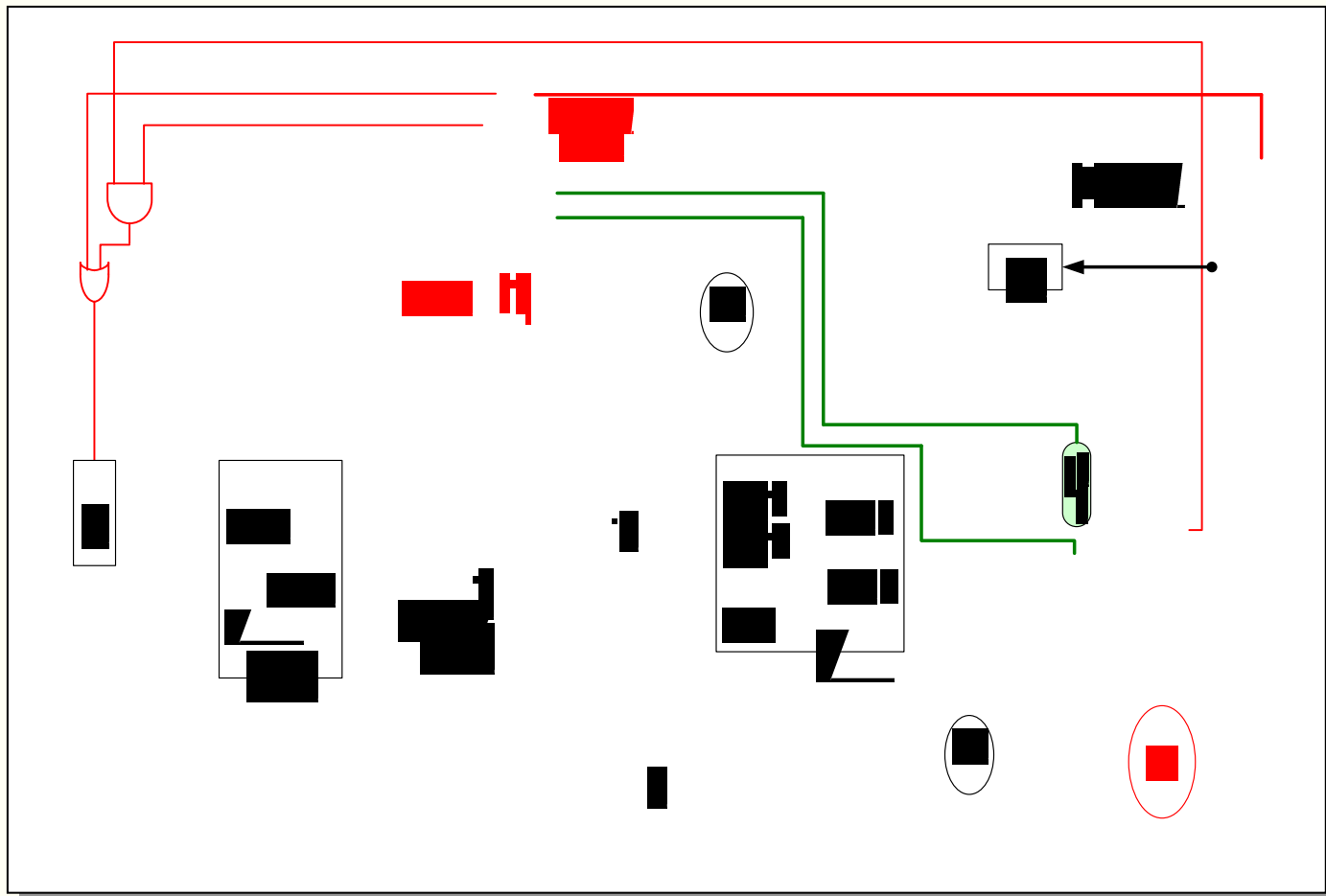
Os registos EPC e Cause podem ser acedidos através do coprocessador 0 do MIPS











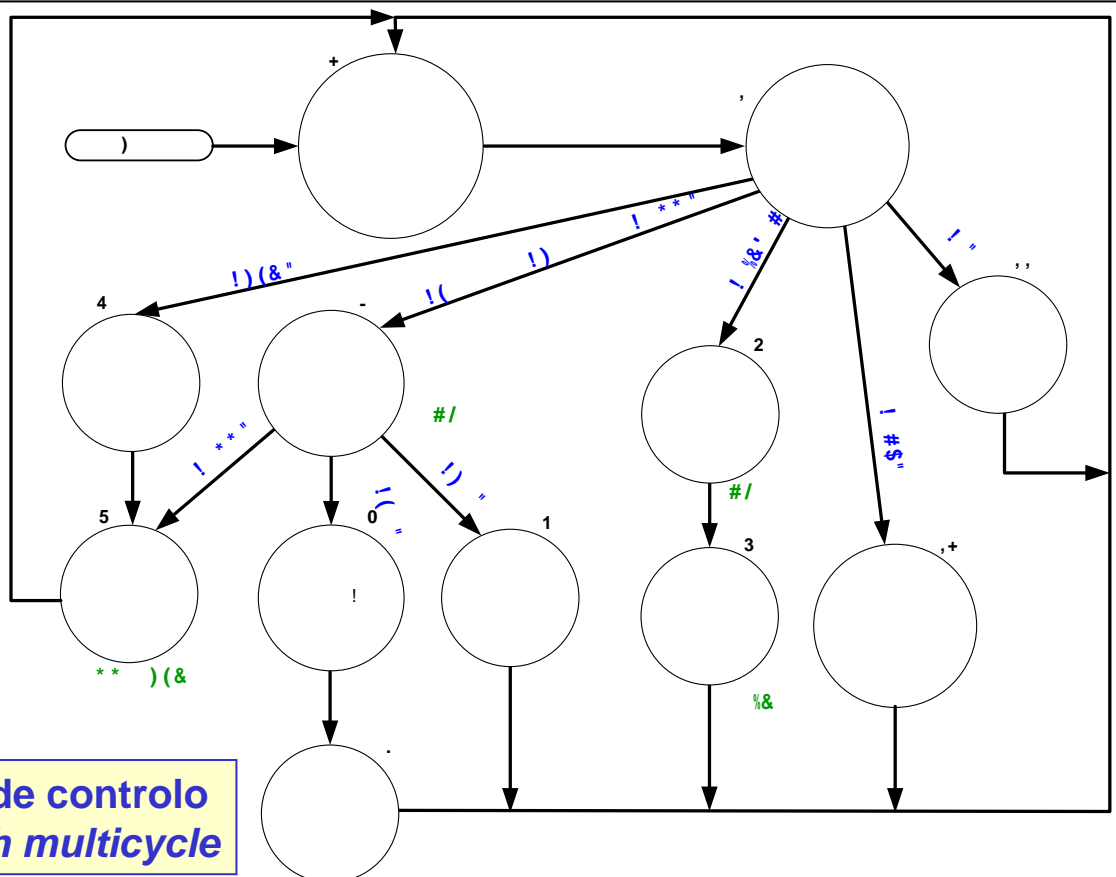
A unidade de controlo com suporte para as excepções

A unidade de controlo tem também de ser redesenhada para acomodar a gestão das excepções e a geração dos sinais de controlo

Como podemos observar *datapath*, a gestão dos dois tipos de excepção, considerados nestes exemplos, carece de um ciclo de relógio suplementar para cada um

Ao nível do diagrama de estados precisaremos de dois estados suplementares

Exercício: Para as alterações *datapath* que entender necessárias para que seja gerada uma excepção sempre que o endereço de acesso à memória (nas respectivas instruções) não deseja mais. Reflita essas alterações na unidade de controlo.



A unidade de controlo do *datapath* multiciclo

