Pesquisa 8 – PIO Interrupção

Exceções

2.1. Qual a diferença entre as exceções NMI e IRQ?

NMIs (**N**on-**M**askable **I**nterruptions) são exceções ativadas normalmente pelo hardware para sinalizar erros graves e que não podem ser ignoradas pelo sistema, como: falha de memória, corrupção de dados ou *reset* do sistema. Já IRQs (**I**nterruption **Req**uests) são interrupções geradas pelo hardware para permitir a execução do *Interrupt Handler*, responsável por lidar com eventos externos como acionamento de um botão ou chegada de dados pela internet.

Interrupções

3.1. Qual a diferença entre as exceções IRQ e ISR?

Como explicado acima, um IRQ e um pedido de interrupção gerado quando outra parte do sistema externa a rotina de execução principal necessita atuar. Já a ISR (**I**nterrupt **S**ervice **R**outine) e de fato a rotina que será executada quando um IRQ for aceito. Por exemplo: o envio de dados do HD para o computador e lento, impossibilitando a espera pelo carregamento completo dos dados. Quando a informação termina de ser carregada o HD gera um IRQ e a ISR de envio de dados e executada.

3.2. No ARM que utilizamos no curso, quantas são as interrupções suportadas e qual a sua menor prioridade?

No ARM do curso, são suportadas interrupções do número 0 ao número 239, totalizando 240 interrupções (ARM Cortex-M7 Devices Generic User Guide na seção Exception Model → Vector Table). De acordo com o manual, as interrupções são configuráveis em seu nível de prioridade, começando em 0 como a de maior prioridade e 239 de menor.

3.3. Descreva o uso do FIQ

O FIQ (Fast Interruption Routine) e um tipo de interrupção com maior prioridade do que os IRQs, não podendo ser interrompida por nenhum outro evento. Quando um FIQ e chamado, automaticamente todas as outras interrupções são mascaradas (ignoradas), da mesma maneira que, se um FIQ e um IRQ forem chamados ao mesmo tempo, o FIQ será executado primeiro.

3.4. No diagrama anterior, quem possui maior prioridade: IRQ ou FIQ?

FIQ.

3.5. No datasheet, secção 13.1 informa o ID do periférico que está associado com a sua interrupção. Busque a informação e liste o ID dos seguintes periféricos:

PIOA: 10
PIOC: 12
TC0: 23

3.6. O que aconteceria caso não limpemos a interrupção?
Se a interrupção não fosse limpa, a cada ciclo do programa a interrupção seria detectada e executada novamente, mesmo que ela fisicamente não esteja mais sendo realizada. Isso criaria um *loop* infinito no programa em questão.

3.7. O que é latência na resolução de uma interrupção (Interrupt Latency), o que é feito nesse tempo?
A latência é a quantidade de ciclos de *clock* que o processador leva para responder a uma interrupção. Nesse meio tempo, alguma operações de "salvamento" são realizadas, movendo para o stack o status do programa principal para que a rotina possa retornar para onde parou.

PIO – Interrupções

5.1. Qual deve ser a configuração para operarmos com interrupção no botão do kit SAME70-EK2?

Low-level detection.

5.2. Com base no texto anterior e nos diagramas de blocos descreva o uso da interrupção e suas opções.

As interrupções, "pausas" no programa principal, podem ser executadas quando se é detectado nível baixo (0), nível alto (1) ou borda de subida (de 0 para 1) e borda de descida (de 1 para 0).

5.3. Descreva as funções dos registradores:

PIO_IER / PIO_IDR: ativar e desativar, respectivamente, o PIO_IMR, responsável por indicar se há uma interrupção ativa.

PIO_AIMER / PIO_AIMDR: ativar e desativar, respectivamente, interrupções adicionais

PIO_ELSR: mostra o status da interrupção adicional (ativado/desativado)

PIO_FRLHSR: mostra status da opção de interrupção escolhida (rise/fall ou high/low)