

Sistemas Microcontrolados

Interrupções

Prof. Guilherme Peron

Interrupções

- Problema da latência
 - Tempo que o microcontrolador demora para acessar um periférico que já disponibilizou um dado útil.
- Formas de acesso aos periféricos
 - Ciclo cego (*blind cycle*);
 - Busy wait ou *polling*;
 - **Interrupção**;
 - **DMA** (direct memory access).

Interrupções

- O que é uma interrupção?



Interrupções

- Definição
 - Qualquer evento interno ou externo que obrigue o microcontrolador a suspender o que está fazendo para atender o evento que o interrompeu.
- Para que serve?
 - Executar uma tarefa de prioridade mais alta.
- Funcionamento
 - O programa é desviado para um outro ponto da memória de programa onde se encontra a rotina de atendimento à interrupção (como uma subrotina).
 - Após executar a rotina, o microcontrolador volta ao ponto imediatamente seguinte de onde foi interrompido.

Interrupções

- Qual a diferença entre um **BL** e uma **interrupção**?

Interrupções

- Qual a diferença entre um **BL** e uma **interrupção**?
 - O BL é uma instrução programada no *software* para acontecer em um momento específico. Já a interrupção, pode acontecer a qualquer momento.

Aspectos Fundamentais

- Existe uma chave geral para todas as interrupções, que habilita ou desabilita as interrupções. (Habilitado por padrão)
 - Registrador do core **PRIMASK** (flag **I**).
 - Se $I=0 \rightarrow$ Interrupções Habilitadas;
 - Se $I=1 \rightarrow$ Interrupções Desabilitadas.
- Existe uma fonte interrupção **independente** para cada periférico disponível no microcontrolador PortA, PortB, ..., PortQ, Timers, UARTs, I2Cs, SSIs...

Aspectos Fundamentais

- O controlador de interrupções separa cada um dos periféricos em **fontes** de interrupção
- Cada fonte de interrupção pode ser ativada ou desativada independentemente por software;
- Cada fonte de interrupção pode ter uma prioridade, em que o registrador do core **BASEPRI** previne interrupções com prioridades menores.
 - Exemplo: Se **BASEPRI** estiver configurado para 3, somente pedidos de interrupção com prioridade 0, 1 ou 2 serão atendidos, enquanto que maior ou igual a 3 serão postergados
 - Se **BASEPRI** estiver configurado em 0, todas as prioridades são permitidas.

NVIC

- As interrupções são controladas pelo **NVIC** (*Nested Vectored Interrupt Controller*);
- Descrito na seção 3.4 Datasheet;
- Cada fonte de interrupção tem um ID (**número da interrupção**) associado (Tabela 2-9 da página 116 *datasheet*)
- Para habilitar uma interrupção em um periférico, é necessário **habilitar** a sua fonte e configurar sua **prioridade** precisa ser **configurada a prioridade** e **habilitada a fonte** no NVIC.

Condições

- Para uma interrupção acontecer → 5 condições
 1. Bit **I** no registrador **PRIMASK** deve ser 0 → Chave geral ligada;
 2. **Fonte** de interrupção **habilitada** no **NVIC**;
 3. O prioridade da interrupção deve ser menor que o nível do **BASEPRI**, a menos que **BASEPRI** seja 0.
 - Prio da interrupção deve ser mais alta que BASEPRI
 4. A interrupção deve estar armada no registrador específico do periférico;
 5. Evento externo da interrupção deve acontecer.

Tratamento da Interrupção

- Quando uma interrupção acontecer, a execução do código principal é interrompida e o programa vai para a **rotina de tratamento de interrupção (ISR)**
- Esta rotina deve ser executada o mais rápido possível (evitar laços e iterações)
- Deve-se sair dela utilizando a instrução no *assembly* **BX LR** ou **return** em C.
- Para que outra interrupção do mesmo tipo ocorra, o seu flag de disparo deve ser limpo dentro **ISR** → ACK da interrupção

Tratamento da Interrupção

- Cada fonte de interrupção (assim como as exceções) está associada a uma posição na memória ROM de 32 bits chamada de **veto**r;
- Cada vetor deste, aponta para uma função de tratamento de interrupção → o endereço da ISR é escrito nestas posições da memória ROM.
- Os vetores estão no início da memória ROM, normalmente no arquivo **startup.s**
- Há até 240 possibilidades de fontes de interrupção que são listados a partir do endereço **0x0000.0040**.

Vetores de Interrupção

- Algumas posições das exceções (Extraído do startup.s)

```
EXPORT __Vectors
__Vectors ; address ISR
DCD StackMem + Stack          ; 0x00000000 Top of Stack
DCD Reset_Handler            ; 0x00000004 Reset Handler
DCD NMI_Handler              ; 0x00000008 NMI Handler
DCD HardFault_Handler        ; 0x0000000C Hard Fault Handler
DCD MemManage_Handler        ; 0x00000010 MPU Fault Handler
DCD BusFault_Handler         ; 0x00000014 Bus Fault Handler
DCD UsageFault_Handler       ; 0x00000018 Usage Fault Handler
...
DCD SVC_Handler              ; 0x0000002C SVCall Handler
DCD DebugMon_Handler         ; 0x00000030 Debug Monitor Handler
DCD 0                        ; 0x00000034 Reserved
DCD PendSV_Handler           ; 0x00000038 PendSV Handler
DCD SysTick_Handler          ; 0x0000003C SysTick Handler
DCD GPIOPortA_Handler        ; 0x00000040 GPIO Port A
DCD GPIOPortB_Handler        ; 0x00000044 GPIO Port B
DCD GPIOPortC_Handler        ; 0x00000048 GPIO Port C
DCD GPIOPortD_Handler        ; 0x0000004C GPIO Port D
DCD GPIOPortE_Handler        ; 0x00000050 GPIO Port E
...
```

Tratamento da Interrupção

- A rotina de tratamento da interrupção pode ser declarada em qualquer arquivo e em qualquer lugar do código;
- Declarar um *label* com o mesmo nome do vetor, para declarar a função de tratamento da interrupção (**ISR**);
- Lembrar de fazer o EXPORT desta função no arquivo que for feita a declaração.

Interrupções Aninhadas

- Uma **interrupção aninhada** (*nested interrupt*) acontece quando uma interrupção de maior prioridade suspende uma **ISR**.
- A **prioridade** define a ordem de execução se duas interrupções acontecerem ao mesmo tempo. Ela define também se uma interrupção pode suspender outra de menor prioridade.

Interrupções Pendentes

- Se:
 - O flag de interrupção estiver ativado, mas as interrupções estiverem desabilitadas ($I=1$);
 - O nível de prioridade não é alto o suficiente;
 - A fonte não está habilitada;

O pedido **não** é perdido, ele vai para fila de **interrupções pendentes**.

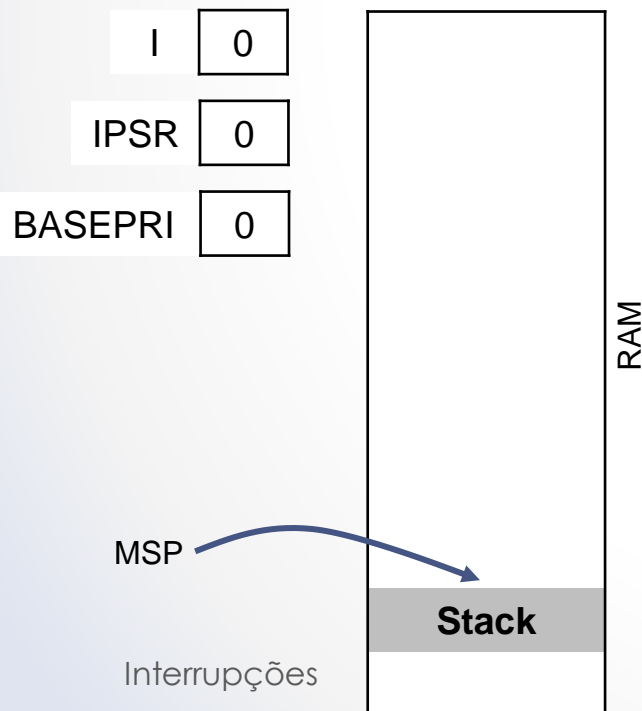
Troca de Contexto

- O que acontece quando ocorre uma interrupção?
 1. A instrução em execução é terminada;
 2. O processador entra no modo de **exceção** (modo *handler*) e a execução do programa corrente é **suspensa**.
 - 8 registradores são empilhados automaticamente na pilha (**R0**, **R1**, **R2**, **R3**, **R12**, **LR**, **PC** e **PSR** com R0 no topo)
 3. **LR** é setado para um valor específico significando que uma rotina de tratamento de interrupção (**ISR**) está sendo tratada (bits [31:8] para 0xFFFFFFFF e bits [7:1] especificam o tipo de interrupção, bit 0 sempre será 1)
 4. Registrador do core **IPSR** (xPSR) é setado para o **número da interrupção processada**
 5. **PC** é carregado com o endereço do **ISR**

Exemplo de Interrupção

- Interrupção gerada por borda na porta C
 - Supor que a interrupção da Porta C seja configurada como prioridade nível 2.

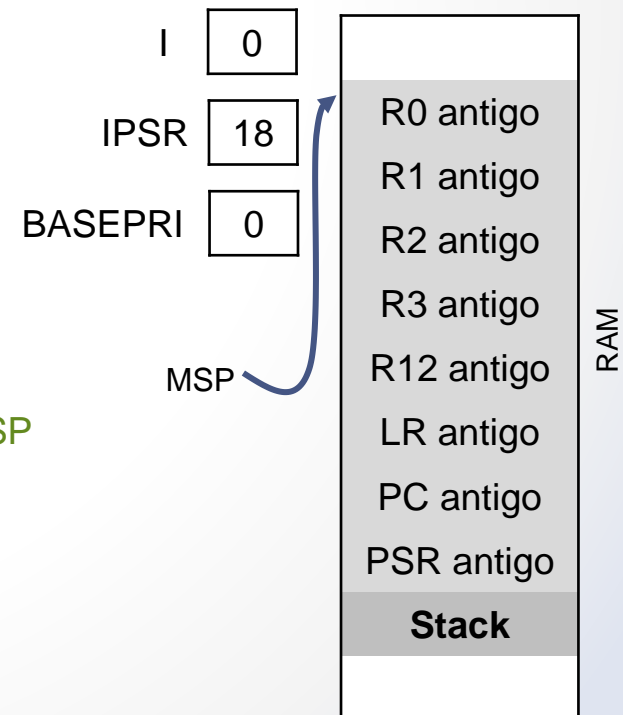
Antes da interrupção



Troca de contexto

- a) Termina a instrução
 - b) Empilha os registradores (R0, R1, R2, R3, R12, LR, PC e PSR)
 - c) PC={0x00000048}
 - d) Seta IPSR=18
 - e) Seta LR=0xFFFFFFF9
- F9 → Ao retornar, volta para o modo normal (*thread*) e usa o MSP como *stack pointer*

Depois da interrupção



Chave Geral

- Para ligar e desligar a chave geral das interrupções há duas funções já declaradas no arquivo startup.s. Basta chamá-las do código:

- Para desligar:

```
DisableInterrupts      CPSID I      ;set I  
                        BX          LR
```

- Para ligar:

```
EnableInterrupts      CPSIE I      ;disable I  
                        BX          LR
```

- Lembrar que as interrupções são **ligadas** por padrão.

Registradores do NVIC

(Seção 2.4 Datasheet)

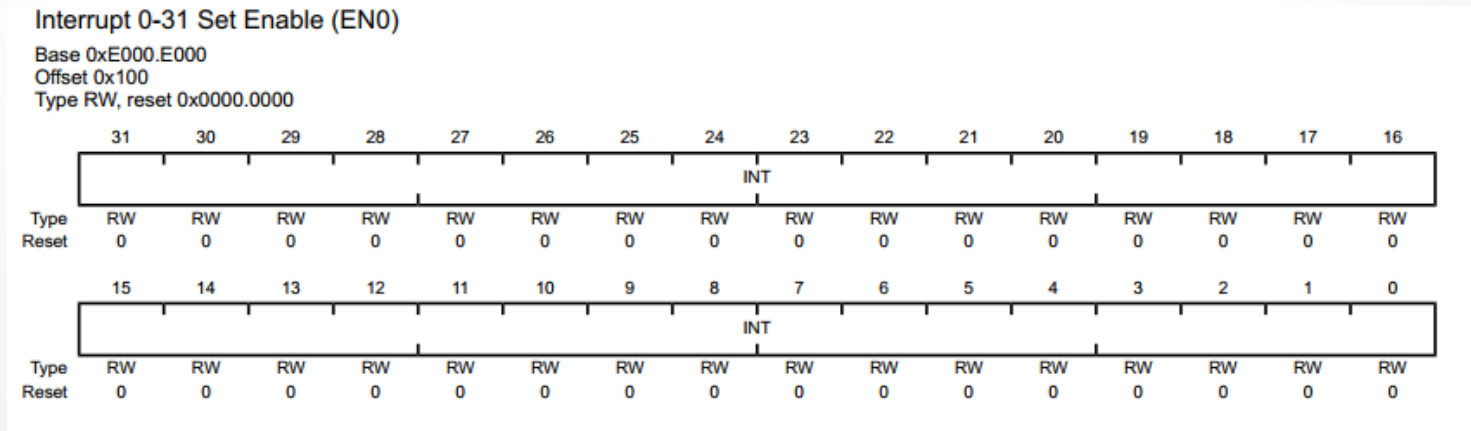
Registradores do NVIC

1) *Interrupt Set Enable* → **ENx**

- Controlam a **habilitação** das fontes de interrupção. Cada bit habilita uma interrupção, que cujo número correspondente pode ser encontrado na Tabela 2-9 da página 116 do *datasheet*.

OBS: Para desabilitar utiliza-se outro registrador.

- Exemplo:



Registradores do NVIC

1) *Interrupt Set Enable* → ENx

Registrador	Interrupções	Endereço
EN0	0 a 31	0xE000.E100
EN1	32 a 63	0xE000.E104
EN2	64 a 95	0xE000.E108
EN3	96 a 113	0xE000.E10C

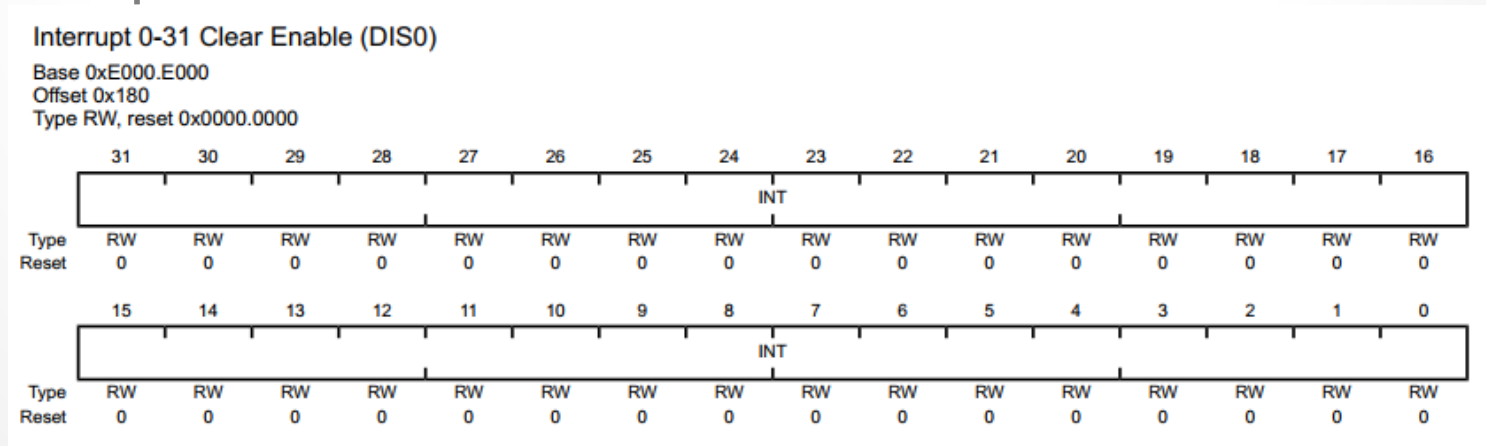
Registradores do NVIC

2) *Interrupt Clear Enable* → DISx

- Controlam o **desligamento** das fontes de interrupção. Cada bit desabilita uma interrupção, que cujo número correspondente pode ser encontrado na Tabela 2-9 da página 116 do *datasheet*.

OBS: Para habilitar utiliza-se o registrador anterior.

- Exemplo:



Registradores do NVIC

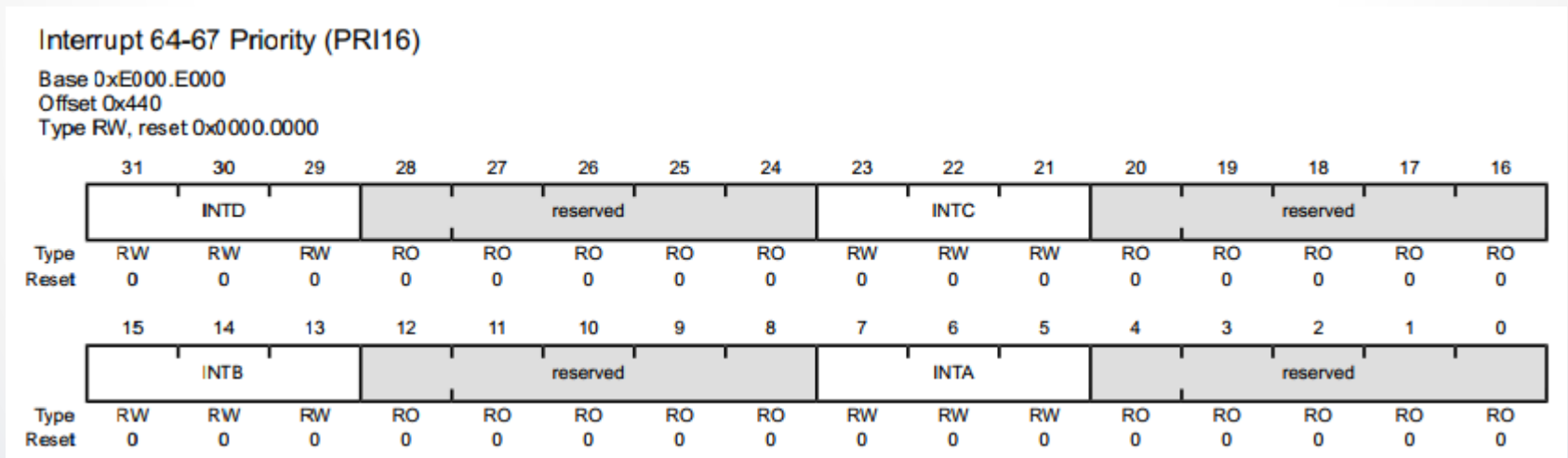
2) *Interrupt Clear Enable* → DISx

Registrador	Interrupções	Endereço
DIS0	0 a 31	0xE000.E180
DIS1	32 a 63	0xE000.E184
DIS2	64 a 95	0xE000.E188
DIS3	96 a 113	0xE000.E18C

Registradores do NVIC

3) *Interrupt Priority* → **PRIx**

- Os registradores de prioridade configuram a **prioridade** de 4 em 4 fontes de interrupção (8 bits para cada fonte, mas **somente os bits entre 5 a 7 são usados**)
- Exemplo: PRI16 (NVIC_PRI16_R):



Registradores do NVIC

3) *Interrupt Priority* → **PRIx**

Registrador	Interrupções	Endereço
PRI0	0 a 3	0xE000.E400
PRI1	4 a 7	0xE000.E404
PRI2	8 a 11	0xE000.E408
...
PRI26	104 a 107	0xE000.E468
PRI27	108 a 111	0xE000.E46C
PRI28	112 e 113	0xE000.E470

Registradores do NVIC

- Além de configurar no NVIC, cada periférico possui sua própria configuração para habilitar a interrupção

Interrupções nos GPIO

Interrupções nos GPIO

- Utilizadas para reconhecer quando um *hardware* altera o estado de 1 para 0 ou 0 para 1.
- Há dois tipos:
 - Por borda
 - Por nível
- Qual a diferença entre esperar por *polling* e interrupção?

Interrupções nos GPIO

- Além dos registradores do NVIC, os seguintes registradores do GPIO também realizam o controle das interrupções a nível de **periférico**.
- Similarmente a outros registradores do GPIO, cada bit controla um pino do port.

Registradores do GPIO

- **GPIOIS** – Borda ou nível;
- **GPIOIBE** – Uma borda apenas ou ambas as bordas;
- **GPIOIEV** – Borda de subida ou borda de descida, nível alto ou nível baixo;
- **GPIOIM** – Habilita a interrupção;
- **GPIORIS** – Indica se as condições para a interrupção aconteceram mesmo se não está habilitada no GPIOIM;
- **GPIOMIS** – Indica que as condições engatilharam uma interrupção no periférico. Neste caso, ela está habilitada no GPIOIM;
- **GPIOICR** – Ao setar o bit, realiza a limpeza do GPIORIS e GPIOMIS, (ACK da interrupção) permitindo uma nova interrupção.

Registradores

- Para configurar se a interrupção é por borda ou nível e qual(is) da(s) borda(s) utiliza-se os seguintes registradores:

GPIOIS	GPIOIBE	GPIOIEV	Modo
0	0	0	Borda de descida
0	0	1	Borda de subida
0	1	-	Ambas as bordas
1	0	0	Nível Baixo
1	0	1	Nível alto

Registradores

- Para habilitar/desabilitar a interrupção utilizar o registrador **GPIOIM**, **setar** o bit do respectivo pino para **habilitar** a interrupção e **limpar** para **desabilitar** a interrupção
- Quando uma condição de interrupção acontece, o sinal do estado da interrupção pode ser visto em dois locais:
 - **GPIORIS**: As condições foram atendidas, mas a interrupção não foi necessariamente enviada para o controlador de interrupções
 - **GPIOMIS**: Mostra somente as condições que são permitidas ser passadas para o controlador de interrupções

Registradores

- Para permitir outra interrupção no mesmo periférico, é obrigatório setar o bit correspondente do registrador **GPIOICR**, no início da ISR;
- Este processo se chama ACK (acknowledgment) da interrupção;
- Quando o **GPIOICR** é setado, o **GPIORIS** e o **GPIOMIS** são limpos. Estes últimos são READ-ONLY, ou seja, não podem ser escritos, somente lidos.

Registradores

- Uma interrupção de GPIO irá acontecer quando:
 - A chave geral estiver habilitada (bit I do PRIMASK tem que ser 0)
 - A fonte da interrupção estiver habilitada no **NVIC**;
 - O nível da fonte de interrupção do **GPIO** for menor que o **BASEPRI** (a interrupção mais prioritária que BASEPRI);
 - O bit do pino correspondente no **GPIOIM** estiver setado;
 - O evento acontecer, por exemplo borda ou nível. Neste caso o flag no **GPIORIS** será setado automaticamente

Configuração

- Sequência para habilitar uma interrupção no GPIO (Além da inicialização do GPIO):
 1. Desabilitar a interrupção no registrador **GPIOIM**;
 2. Configurar o tipo de interrupção (0=borda, 1=nível) no registrador **GPIOIS**;
 3. Configurar **GPIOIBE** (0=borda única, 1=borda dupla), **GPIOIEV** (0= nível baixo ou borda de descida, 1=nível alto ou borda de subida);
 4. Limpar o registrador GPIORIS, escrevendo 1 no **GPIOICR**;
 5. Habilitar a interrupção no registrador **GPIOIM**;
 6. Habilitar a interrupção no NVIC;
 7. Setar a prioridade no NVIC.

Tratamento da Interrupção

- A função de tratamento da interrupção (**ISR**) pode ser declarada em qualquer arquivo, no entanto deve ter o mesmo nome que as declarações no arquivo *startup.s*. Não esquecer do **EXPORT**;
- A primeira coisa a se fazer na ISR é realizar o ACK. Setar o bit respectivo no registrador **GPIOICR**, para limpar o **GPPIORIS** e **GPPIOMIS**;
- Como as portas têm mais de um pino, se houver interrupções **ativadas** em mais de um pino na porta, verificar através do registrador **GPPIORIS** para saber de qual pino é a interrupção e depois realizar o tratamento para a interrupção naquele pino.

Tratamento da Interrupção

- Exemplo de tratamento de interrupção de GPIO para o pino PC4 (pode ser declarado em qualquer arquivo desde que faça o **EXPORT**)

GPIOPortC_Handler

```
LDR R1, =GPIO_PORTC_ICR_R
MOV R0, #0x10      ;bit 4 do PortC 00010000b
STR R0, [R1]       ;limpando a interrupção (ack)
;
; código qualquer
;
BX LR              ;retorno
```

Exemplo

Exemplo

- Interrupção por borda de descida no PM0
- Cada vez que acontece a interrupção (borda de descida), o LED do PN0 é alternado
- Para testar a borda de descida, ligamos o pino PM0 (entrada) no pino PK7 (saída)

Exercício

- Programar uma interrupção por borda de descida na USR_SW1 (PJ0) e uma interrupção por borda de subida na USR_SW2 (PJ1). Para que quando o usuário pressione a chave USR_SW1 acenda o LED1 (PN1) e quando soltar a chave USR_SW2 apague o LED1.

Exercício (Passo-a-passo)

A seguir será demonstrado o passo-a-passo para a configuração da interrupção no GPIO Port J;

Antes de configurar a interrupção, deve-se realizar a configuração dos GPIOs conforme a aula de GPIOs.

Exercício (Passo-a-passo)

- Passo 0
 - Configurar os ports **J** e **N**, conforme a aula de GPIO.
 - **J0** e **J1** → Entrada
 - **N1** → Saída

Exercício (Passo-a-passo)

1. Antes de configurar as interrupções, devemos desabilitar (para depois habilitar novamente ao final) no registrador **GPIOIM**. Como vamos usar os pinos J0 e J1, desabilitar os dois bits.

GPIO_PORTJ_AHB_IM_R

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DMAIMR		IMR				0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
															J1	J0

Zerar estes bits

Pag 764

Exercício (Passo-a-passo)

2. Como vamos capturar a interrupção durante o pressionamento ou liberação das chaves, a interrupção deve ser configurada como borda em ambos os pinos no registrador **GPIOIS**;

GPIO_PORTJ_AHB_IS_R

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								IS							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

J1

J0

Zerar estes bits

Pag 761

Exercício (Passo-a-passo)

3.a) Configurar borda única em ambos pinos no registrador **GPIOIBE**;

GPIO_PORTJ_AHB_IBE_R

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved															
Type	RO	RO	RO	RO	RO	RO	RC	RO	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
															0	0
															J1	J0

Zerar estes bits

Pag 762

Exercício (Passo-a-passo)

3.b) Configurar borda de descida para J0 e borda de subida para J1 no registrador **GPIOIEV**;

GPIO_PORTJ_AHB_IEV_R

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								IEV							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
															J1	J0

Pag 763

Exercício (Passo-a-passo)

4. Garantir que a interrupção será atendida limpando o GPIORIS e GPIOMIS, realizando o ACK no registrador **GPIOICR** para ambos os pinos. Setar os bits;

GPIO_PORTJ_AHB_ICR_R

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								IEV							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
															1	1
															J1	J0

Setar estes bits

Pag 769

Exercício (Passo-a-passo)

5. Ativar a interrupção em ambos os pinos do Port J, no registrador **GPIOIM**;

GPIO_PORTJ_AHB_IM_R

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved								DMAIMR		IMR				1	0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Setar estes bits

J1 J0

Pag 764

Exercício (Passo-a-passo)

6. a) Ativar a fonte de interrupção no NVIC.

Primeiramente consultar a Tabela 2-9 que começa na página 116 do datasheet, para saber o número da interrupção do Port J. A fonte do Port J é a número **51**.

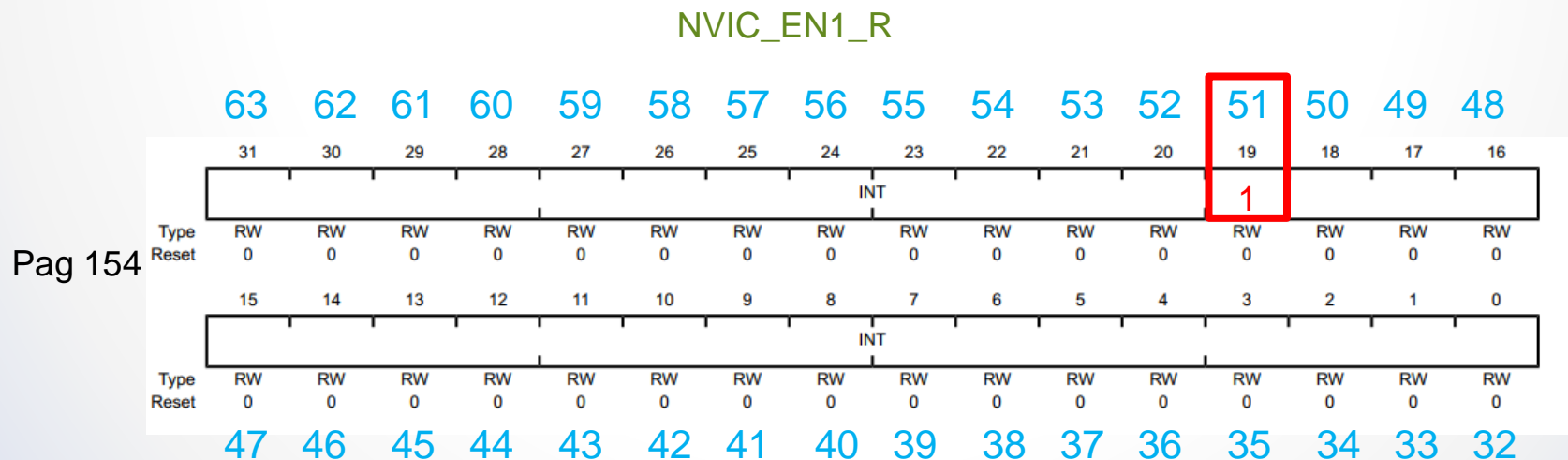
Table 2-9. Interrupts (continued)

Vector Number	Interrupt Number (Bit in Interrupt Registers)	Vector Address or Offset	Description
62	46	0x0000.00F8	ADC1 Sequence 0
63	47	0x0000.00FC	ADC1 Sequence 1
64	48	0x0000.0100	ADC1 Sequence 2
65	49	0x0000.0104	ADC1 Sequence 3
66	50	0x0000.0108	EPI 0
67	51	0x0000.010C	GPIO Port J
68	52	0x0000.0110	GPIO Port K
69	53	0x0000.0114	GPIO Port L
70	54	0x0000.0118	SSI 2
71	55	0x0000.011C	SSI 3

Exercício (Passo-a-passo)

6. b) Ativar a fonte de interrupção no NVIC.

Sabendo que o número da fonte de interrupção é o 51, encontrar qual ENx habilitará a fonte de interrupção do Port J. Da tabela 3-8 (página 146) encontra-se que o **EN1** habilita as interrupções 32 a 61. Setar o bit deste registrador que habilita a interrupção, no bit **19**.



Exercício (Passo-a-passo)

7. b) Configurar a prioridade da fonte de interrupção no NVIC.

Sabendo que o número da fonte de interrupção é o 51, encontrar qual PRIx configura a prioridade da fonte de interrupção do Port J. Da tabela 3-8 (página 146), encontra-se que o **PRI12** configura a prioridade das interrupções 48 a 51. Configurar os bits 29 a 31, com a prioridade desejada. Vamos supor que queremos setar a prioridade 5.

NVIC_PRI12_R

51															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	0	1	reserved					INTC			1 reserved				
Type	RW	RW	RW	RO	RO	RO	RO	RO	RW	RW	RW	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
49															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTB			reserved					INTA			reserved				
Type	RW	RW	RW	RO	RO	RO	RO	RO	RW	RW	RW	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
								48							

Dica: usar a instrução:

```
MOV    Rx, #5
```

```
LSL    Rx, #29
```

Exercício (Passo-a-passo)

- ISR
 - Verificar no arquivo startup.s qual é a função de tratamento de interrupção.
 - No caso é a **GPIOPortJ_Handler**.
 - Declarar o label **GPIOPortJ_Handler** em qualquer arquivo que não seja o startup.s, por exemplo gpio.s.
 - Primeiramente realizar um teste para saber qual interrupção foi gerada, se foi causada pelo J0 ou pelo J1, por meio do **GPIORIS**
 - Se o bit 0 estiver setado, foi causada pelo J0. Se o bit 1 estiver setado, foi causada pelo J1.
 - Realizar o ACK da interrupção escrevendo 1 no bit respectivo do registrador **GPIOICR**.
 - Escrever no pino para acender ou apagar o LED
 - Sair da interrupção com **BX LR**