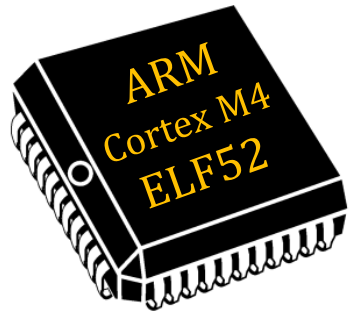


- GPIO
- PLL
- SysTick
- Referências

Microcontroladores: (ELF52)

Prof: DaLuz



GPIO

General Purpose Input/Output



- GPIO
- PLL
- SysTick
- Referências

- *General Purpose Input/Output*
- Para que servem?



GPIO

General Purpose Input/Output



- GPIO
- PLL
- SysTick
- Referências

- Para que servem?
- Para trocar informação digital com o mundo externo
- Exemplo:
 - Controlar LEDs
 - Controlar chaves



GPIO

Pinos do TM4C:

- A função regular de um pino é realizar I/O.
- Entretanto a maioria dos pinos têm uma ou mais funções alternativas:
 - UART
 - SSI (SPI)
 - I²C
 - *Timer*
 - PWM
 - ADC
 - Comparador Analógico
 - USB
 - Ethernet
 - CAN

UTFPR

- GPIO

- PLL

- SysTick

- Referências

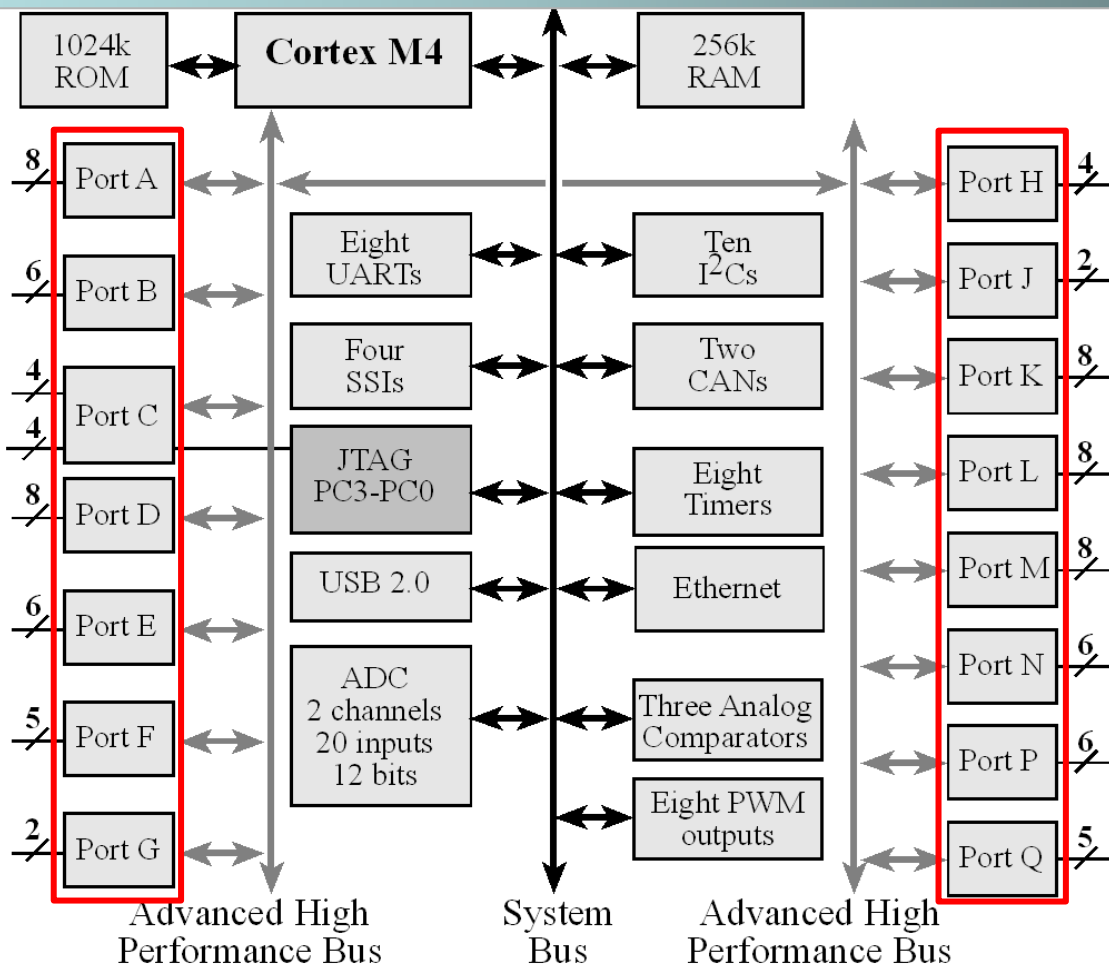


GPIO



- GPIO
- PLL
- SysTick
- Referências

Pinos do TM4C:





GPIO

Pinos do TM4C:

- Além de **GPIO**, os pinos de **I/O** podem ser associados a até sete funções alternativas.
- Exemplo: PA0
 - I/O Digital
 - Entrada Serial
 - *Clock* I2C
 - *Captura do Timer*
- Há funções que podem ser mapeadas em mais de um pino
- Há funções que só existem em um pino

UTFPR

- GPIO

- PLL

- SysTick

- Referências



GPIO

Pinos do TM4C:

- A placa EK-TM4C1294XL tem **duas** chaves e **quatro** LEDs mapeados nos GPIOs
 - Chaves de usuário → **Lógica negativa** e necessitam habilitar um **resistor de pull-up** (PUR)
 - LEDs de usuário → **Lógica positiva**

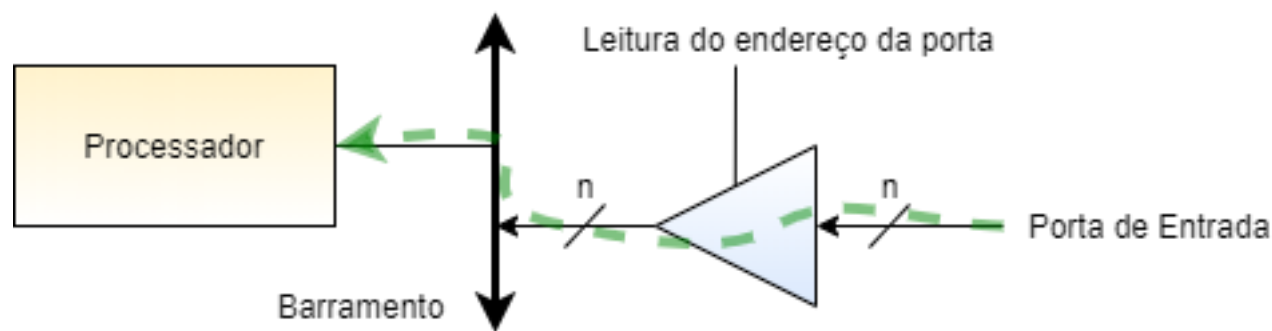




GPIO

Pino em função de entrada Digital:

- Porta de entrada permite o *SW* ler sinais digitais externos.
- Um ciclo de leitura ao endereço da porta retorna o valor de todas as entradas naquele momento.
- O *driver tri-state* direciona o sinal de entrada para o barramento de dados.



(Adaptado de VALVANO, J.)

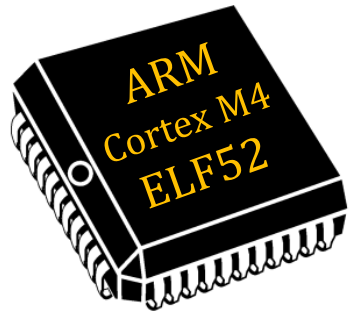
UTFPR

- GPIO

- PLL

- SysTick

- Referências



GPIO

Pino em função de entrada Digital:



- GPIO
- PLL
- SysTick
- Referências

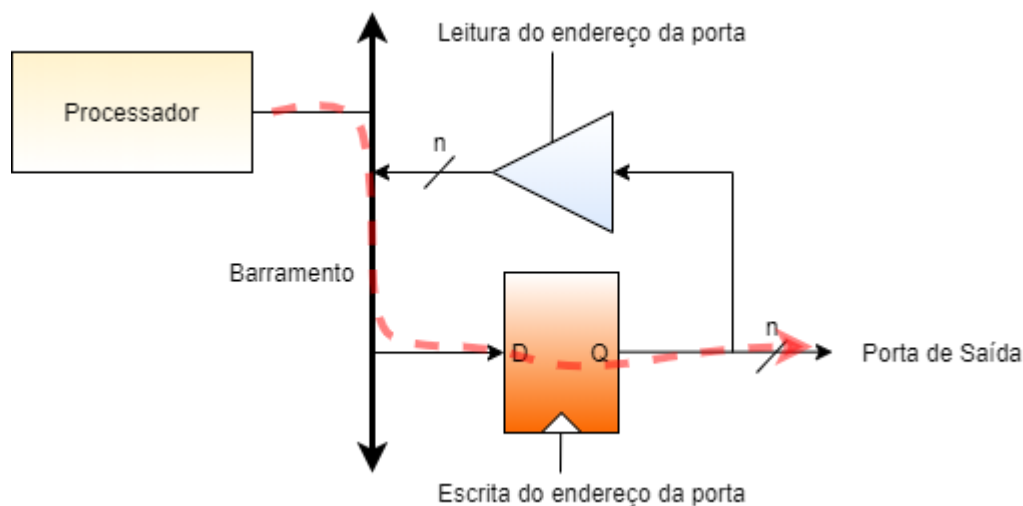
- Para fazer um pino de entrada escrever **0** no **registrador de direção**
- Desta forma um acesso de escrita não tem efeito nenhum
- A maioria dos pinos são tolerantes a 5V de entrada
Valores entre 2V e 5V serão considerados **ALTOS**
Valores entre 0V e 1,3V serão considerados **BAIXOS**



GPIO

Pino em função de saída Digital:

- Porta de saída permite o SW escrever sinais digitais externos, mas também permite ler o que foi escrito
- Um ciclo de escrita no endereço porta escreve os valores nos pinos de saída
- Para fazer um pino de saída escrever **1** no **registrador de direção**

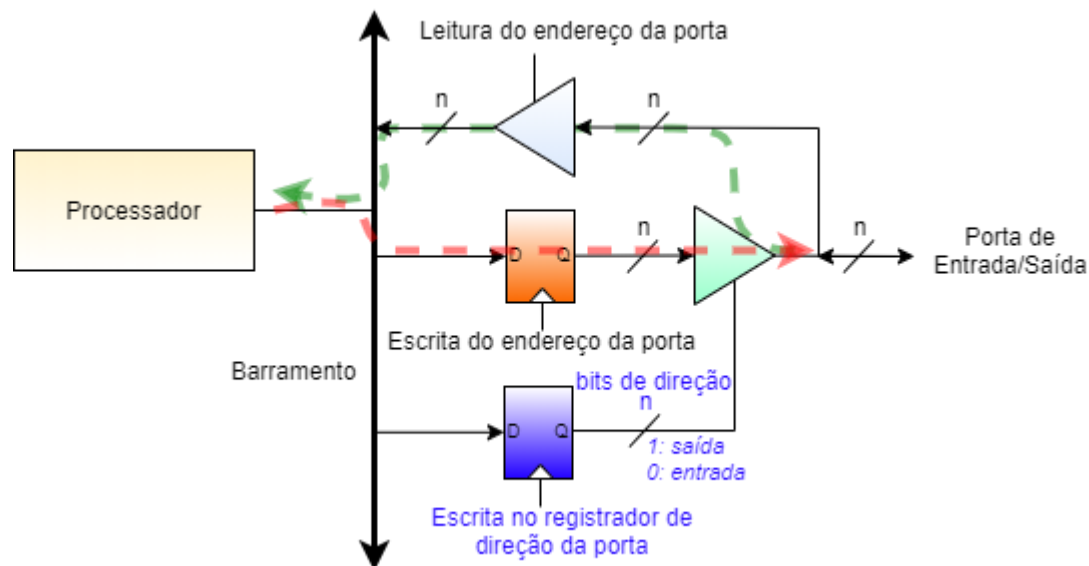




GPIO

Pino em função de saída Digital:

- Os pinos de GPIO se comportam a uma porta paralela
 - Múltiplos sinais podem ser acessados ao mesmo tempo
 - Mecanismo simples que permite ao SW interagir com dispositivos externos



UTFPR

- GPIO
- PLL
- SysTick
- Referências



GPIO

Programação:

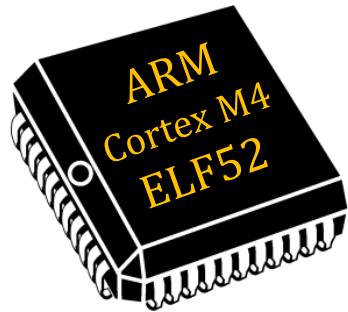
- Como acessar os GPIOs na Tiva?
- Capítulo 10 do Datasheet.
- Funções especiais na Tabela 10-2.

Capítulo 10 pg. 742:

http://www.elf52.daeln.com.br/Pdfs/Datasheet_TM4C1294NCPDT_v1.01.pdf



- GPIO
- PLL
- SysTick
- Referências



- GPIO
- PLL
- SysTick
- Referências

GPIO

Programação:

- Como acessar os GPIOs na Tiva?
 - Na Tiva e em vários outros microcontroladores as portas de I/O **são mapeadas em memória**;
 - Cada porta deve seguir uma série de configurações na memória (em registradores) antes de ser utilizada
 - Para escrever e ler nos pinos de cada porta também deve-se escrever ou ler em endereços específicos da memória.
 - Realizar operações de LDR e STR



GPIO

Programação:



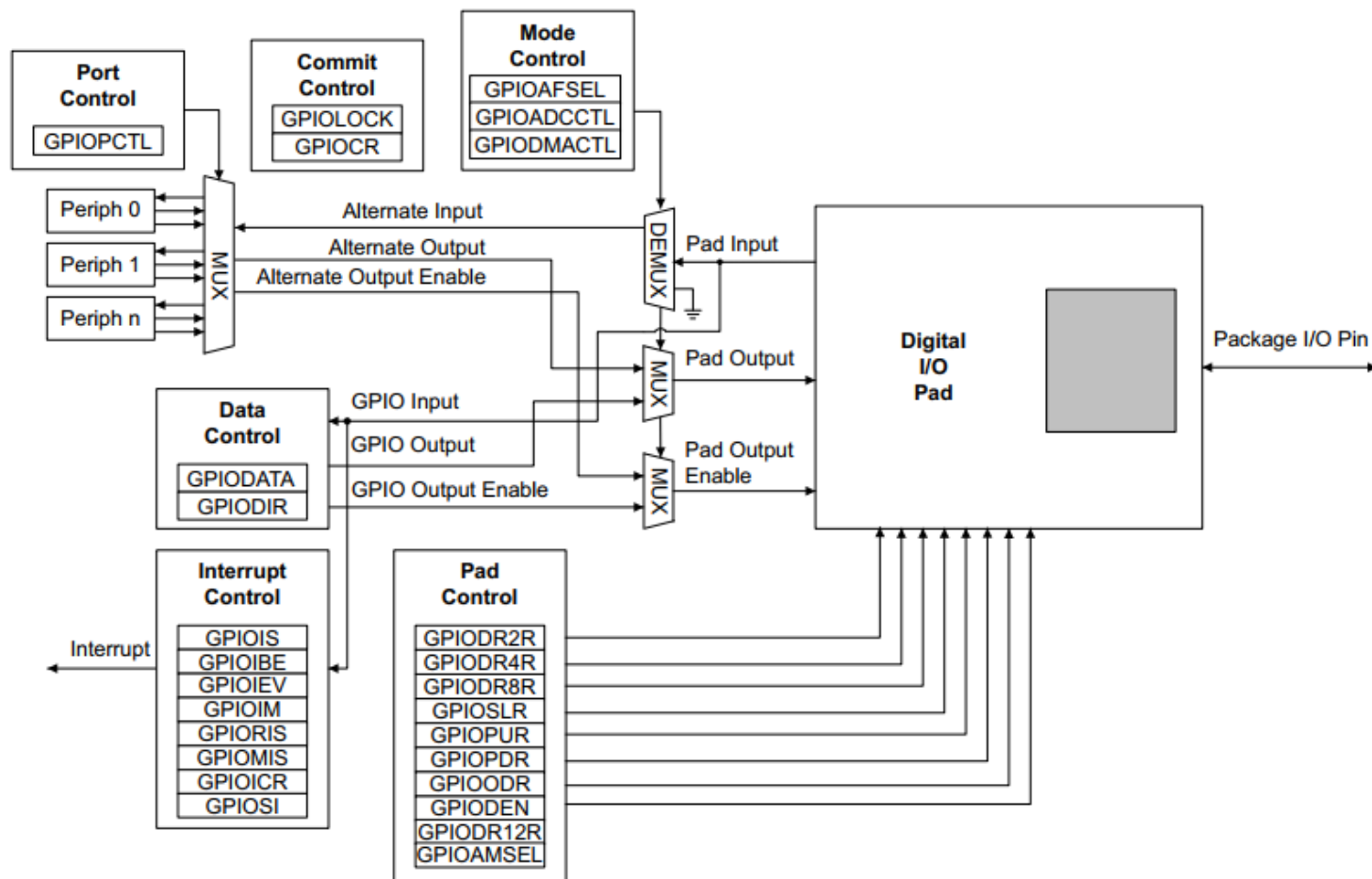
- GPIO
- PLL
- SysTick
- Referências

- As operações com I/O mapeado em memória se parecem com operações com memória, mas não agem igual memória (**Registrador Especial**)
 - Alguns bits são *read-only*
 - Alguns bits são *write-only*
 - Alguns bits só podem ser setados (1)
 - Alguns bits só podem ser limpos (0)



GPIO

Registradores:



- GPIO
- PLL
- SysTick
- Referências



- GPIO
- PLL
- SysTick
- Referências

GPIO

Registradores:

- Cada registrador segue o endereço base de uma porta + o endereço de configuração
- Endereços base de cada porta (Datasheet Seção 10.5 - pag 759)

GPIO Port	Endereço Base	GPIO Port	Endereço Base
GPIO Port A	0x4005.8000	GPIO Port J	0x4006.0000
GPIO Port B	0x4005.9000	GPIO Port K	0x4006.1000
GPIO Port C	0x4005.A000	GPIO Port L	0x4006.2000
GPIO Port D	0x4005.B000	GPIO Port M	0x4006.3000
GPIO Port E	0x4005.C000	GPIO Port N	0x4006.4000
GPIO Port F	0x4005.D000	GPIO Port P	0x4006.5000
GPIO Port G	0x4005.E000	GPIO Port Q	0x4006.6000
GPIO Port H	0x4005.F000		



- GPIO
- PLL
- SysTick
- Referências

GPIO

Registadores:

- *Direction Register (GPIODIR)*
 - Especifica se os pinos são de entrada ou saída. 1 bit por pino.
- *Alternate Function Register (GPIOAFSEL)*
 - Especifica se alguma função alternativa será utilizada. 1 bit por pino.
- *Digital Enable Register (GPIODEN)*
 - Se o pino deve ser utilizado como entrada ou saída digital. 1 bit por pino.
- *Analog Mode Select Register (GPIOAMSEL)*
 - Especifica se o pino será usado como entrada analógica. 1 bit por pino
- *Port Control Register (GPIOPCTL)*
 - Especifica qual a função alternativa (tabela 10-2 do *datasheet*) será utilizada. 4 bits por pino.



- GPIO
- PLL
- SysTick
- Referências

GPIO

Registradores:

- *Data Register (GPIODATA)*
 - Realiza entrada e saída na porta. 1 bit por pino.
- *Run Mode Clock Gating (RCGCGPIO) pag. 382*
 - Habilita o *clock* de cada porta. Obrigatório para habilitar uma porta. 1 bit por porta.
- *Peripheral Ready (PRGPIO) pag. 499*
 - Indica se a porta de GPIO já está pronta para o uso. 1 bit por porta.



- GPIO
- PLL
- SysTick
- Referências

GPIO

Programação do GPIO

- Passo-a-passo para ativar uma porta como entrada e saída (Resumo da **seção 10.4** pg. 753 do *DS*)
 1. Ativar o *clock* para a porta setando o bit correspondente no registrador **RCGCGPIO** e, após isso, verificar no **PRGPIO** se a porta está pronta para uso.
 2. Desabilitar a funcionalidade analógica, limpando os bits no registrador **GPIOAMSEL**
 3. Selecionar a funcionalidade de GPIO limpando os bits no registrador **GPIOCTL**
 4. Especificar se o pino é de entrada ou saída limpando ou setando, respectivamente os bits no registrador **PIODIR**



- GPIO
- PLL
- SysTick
- Referências

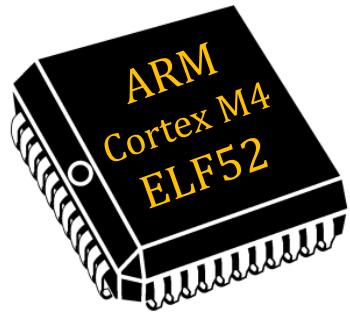
GPIO

Programação do GPIO

- Passo-a-passo para ativar uma porta como entrada e saída (continuação)
 5. Como o objetivo é utilizar os pinos como GPIO e não função alternativa limpar os bits correspondentes no registrador **GPIOAFSEL**
 6. Habilitar a funcionalidade de entrada e saída digital no registrador **GIODEN**

(Opcional) Habilitar um resistor de *pull-up* para entrada importante para operação com chaves no registrador **GPIOPUR**

E depois que inicializamos uma GPIO como fazemos para ler ou escrever na GPIO?



- GPIO
- PLL
- SysTick
- Referências

GPIO

Programação do GPIO

- *Data Register (GPIO DATA)*
 - Através do **Data Register** realiza-se a leitura e escrita do valor desejado dos pinos de cada porta.
 - Um **STR** para o endereço do **DATA Register** fará com que os pinos sejam modificados, ou seja, é realizada uma **ESCRITA** nos pinos.
 - Um **LDR** do endereço do **DATA Register** fará com que os pinos sejam lidos, ou seja, é realizada uma operação de **LEITURA**.



- GPIO
- PLL
- SysTick
- Referências

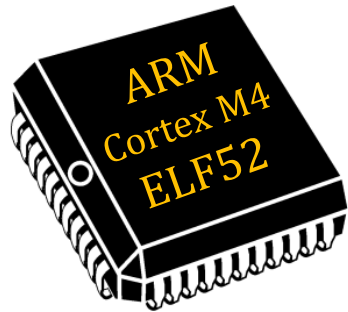
GPIO

Escrita e Leitura

- *Data Register (GPIO DATA)*

GPIO Port	Endereço
GPIO Port A	0x4005.83FC
GPIO Port B	0x4005.93FC
GPIO Port C	0x4005.A3FC
GPIO Port D	0x4005.B3FC
GPIO Port E	0x4005.C3FC
GPIO Port F	0x4005.D3FC
GPIO Port G	0x4005.E3FC
GPIO Port H	0x4005.F3FC

GPIO Port	Endereço
GPIO Port J	0x4006.03FC
GPIO Port K	0x4006.13FC
GPIO Port L	0x4006.23FC
GPIO Port M	0x4006.33FC
GPIO Port N	0x4006.43FC
GPIO Port P	0x4006.53FC
GPIO Port Q	0x4006.63FC



- GPIO
- PLL
- SysTick
- Referências

GPIO

Escrita e Leitura

- Entretanto se uma escrita é feita modificando todos os bits de uma porta corre-se o risco de sobrescrever outros pinos **indesejadamente**.
- Para evitar alterações em pinos indesejados há duas formas (escrita “**amigável**”):
 - Usar o trio: ***read-modify-write***
 - Usar **endereçamento de bit específico** (disponível em alguns microcontroladores). O ***Data Register*** apresenta uma estrutura complexa, permitindo o acesso individualmente de cada um dos bits ou de todos os bits da porta apenas modificando os endereços de acesso.



- GPIO
- PLL
- SysTick
- Referências

GPIO

Escrita e Leitura

- **Read-modify-write**

- Se desejar setar o pino PK7 para 1

```
GPIO_PORTK_DATA_R EQU 0x400613FC
LDR R1, =GPIO_PORTK_DATA_R      ;Carrega-se o endereço
LDR R0, [R1]                    ; Lê para carregar o valor
                                ; anterior da porta inteira
ORR R0, R0, #0x80                ; Faz o OR bit a bit para manter os
                                ; valores anteriores e setar só o bit
STR R0, [R1]                    ; Escreve o novo valor da porta
```

- Se desejar limpar o pino PK7 para 0

```
GPIO_PORTK_DATA_R EQU 0x400613FC
LDR R1, =GPIO_PORTK_DATA_R      ;Carrega-se o endereço
LDR R0, [R1]                    ; Lê para carregar o valor
                                ; anterior da porta inteira
BIC R0, R0, #0x80                ; Faz o AND negado bit a bit para manter
                                ; os valores anteriores e limpar só bit
STR R0, [R1]                    ; Escreve o novo valor da porta
```



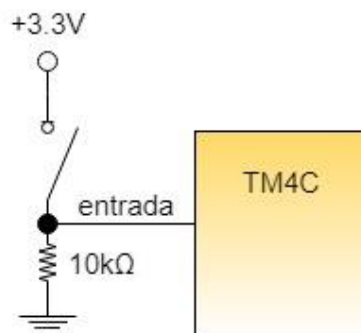

- GPIO
- PLL
- SysTick
- Referências

GPIO

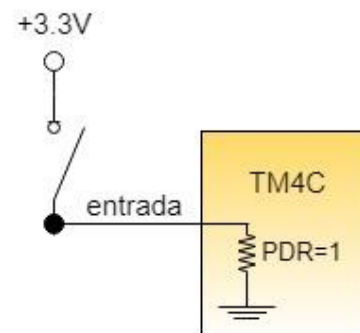
Chaves e LEDs

- Há as seguintes formas de interfacear com uma chave:

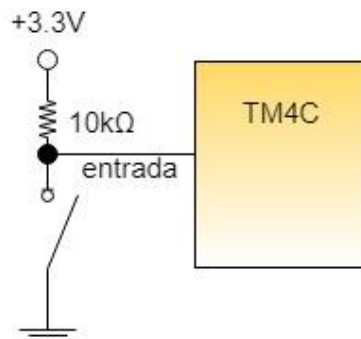
Lógica positiva, resistor externo



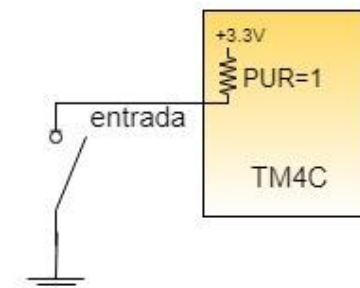
Lógica positiva, resistor interno



Lógica negativa, resistor externo



Lógica negativa, resistor interno





- GPIO
- PLL
- SysTick
- Referências

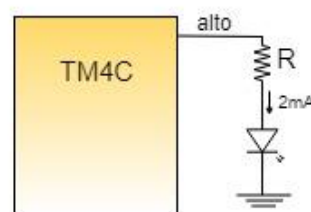
GPIO

Chaves e LEDs

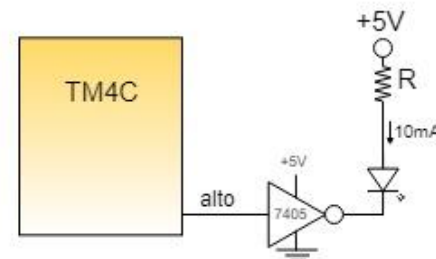
- Há as seguintes formas de interfacear com uma chave:

- Uma porta no TM4C1294 suporta no máximo 12mA, mas o microcontrolador não suporta todas as portas drenando / suprindo este máximo de corrente para todas as portas. (Ver *Datasheet* seção 27.3.2.1)
- O valor *default* é cada porta drenar/suprir 2mA por porta.
- *Se precisar que uma porta forneça mais corrente que ela suporta, deve-se utilizar um driver com circuito integrado ou transistor.*

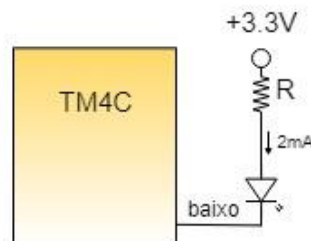
Lógica positiva, corrente baixa



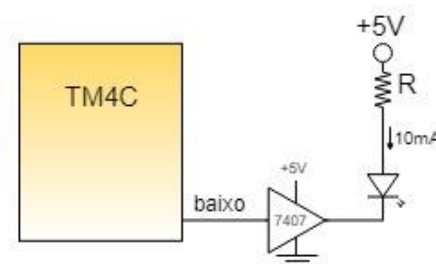
Lógica positiva, corrente alta



Lógica negativa, corrente baixa



Lógica negativa, corrente alta





- GPIO
- PLL
- SysTick
- Referências

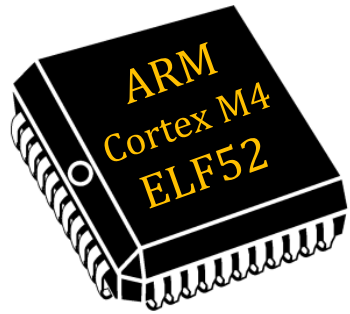
GPIO

Exercício:

Verificar a inicialização da porta J e porta F. Os pinos J0 e J1 estão ligados às chaves tácteis USR_SW1 e USR_SW2, respectivamente e os pinos F4 e F0 estão ligados aos LEDs 3 e 4, respectivamente.

- Baixar e abrir o projeto [GPIO1](#) do site da disciplina.
- Fazer o build e executar passo-a-passo para verificar a inicialização das portas.
- Executar e testar pressionando os dois botões e verificar se os LEDs acendem.
- Fazer um **fluxograma** do código. {Opcional}

<http://www.elf52.daeln.com.br/Labs/GPIO.pdf>



GPIO

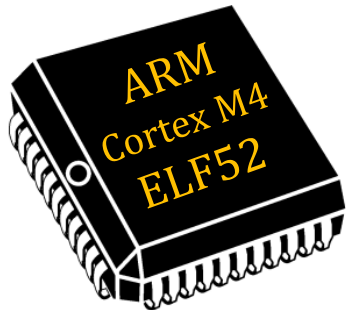
Exercício (Passo a Passo):

A seguir será demonstrado o passo-a-passo para a inicialização dos GPIOs J e F para este exercício;

Dica: correspondentes à seção 10.6 do datasheet.



- GPIO
- PLL
- SysTick
- Referências



GPIO

Exercício (Passo a Passo):

1. Como vamos utilizar os ports F e J, **setar** os bits 5 e 8 no registrador **RCGCGPIO** e depois **esperar** enquanto os bits 5 e 8 do registrador **PRGPIO** não estão **setados**.

General-Purpose Input/Output Run Mode Clock Gating Control (RCGCGPIO)

Base 0x400F.E000

Offset 0x608

Type RW, reset 0x0000.0000

SYSCTL_RCGCGPIO_R **Setar estes bits**

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
Reset	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	R14	R13	R12	R11	R10	R9	1	R7	R6	1	R4	R3	R2	R1	R0
Type	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Q	P	N	M	L	K	J	H	G	F	E	D	C	B	A

Pag 382

General-Purpose Input/Output Peripheral Ready (PRGPIO)

Base 0x400F.E000

Offset 0xA08

Type RO, reset 0x0000.0000

SYSCTL_PRGPIO_R **Testar, enquanto NÃO sejam 1**

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
Reset	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	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	R4	R3	R2	R1	R0
Type	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
	Q	P	N	M	L	K	J	H	G	F	E	D	C	B	A

Pag 382



GPIO

Exercício (Passo a Passo):

2. a) Como vamos utilizar os pinos J0 e J1 como entrada digital zerar pelo menos os bits 0 e 1 do **GPIOAMSEL** do Port J.

UTFPR

- GPIO
- PLL
- SysTick
- Referências

Pag 786

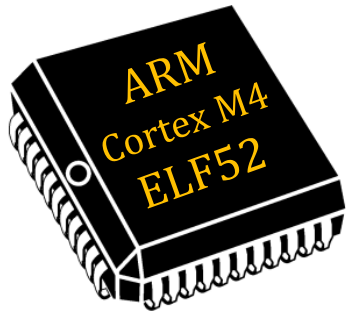
GPIO_PORTJ_AHB_AMSEL_R

Zerar estes bits

	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								GPIOAMSEL							
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

00

J1 J0



GPIO

Exercício (Passo a Passo):

2. b) Como vamos utilizar os pinos F0 e F4 como saída digital zerar pelo menos os bits 0 e 4 do **GPIOAMSEL** do Port F.



- GPIO
- PLL
- SysTick
- Referências

Pag 786

GPIO_PORTF_AHB_AMSEL_R

Zerar estes bits

	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											AMSEL				
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
												F4	F3	F2	F1	F0



GPIO

Exercício (Passo a Passo):

3. a) Como vamos utilizar os pinos J0 e J1, como GPIO, ou seja, sem função alternativa, **zerar** os bits correspondentes ao J0 e ao J1 do **GPIOCTL** do Port J.

GPIO_PORTJ_AHB_PCTL_R

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	PMC7				PMC6				PMC5				PMC4			
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PMC3				PMC2				PMC1				PMC0			
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	-	-	-	-	-	-	-	-	0	0	0	0	0	0	0	0
									J1				J0			

Pag 788

Zerar estes bits



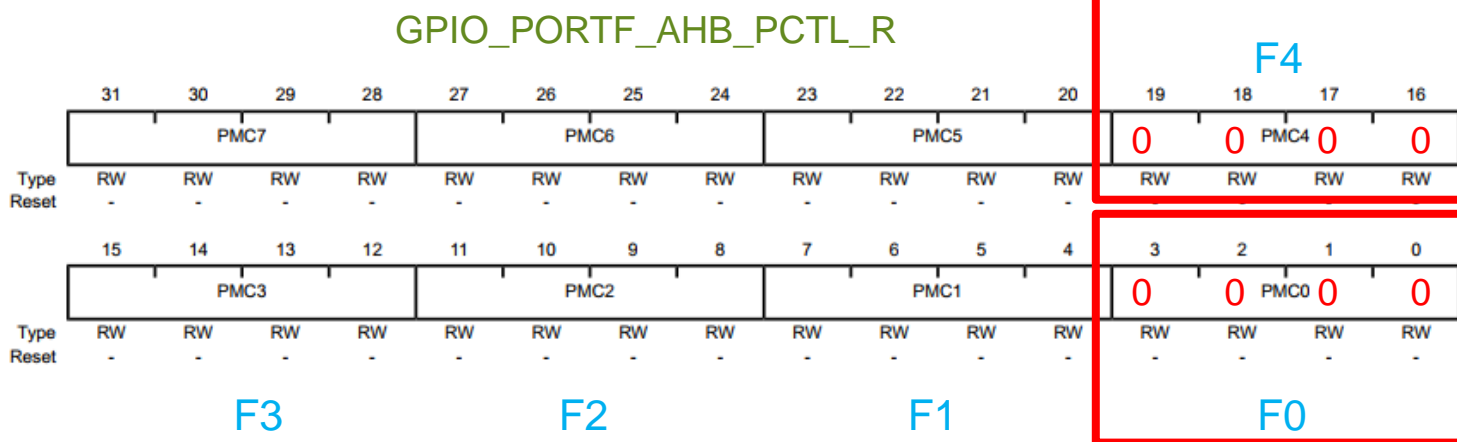
- GPIO
- PLL
- SysTick
- Referências



GPIO

Exercício (Passo a Passo):

3. b) Como vamos utilizar os pinos F0 e F4, como GPIO, ou seja, sem função alternativa, **zerar** os bits correspondentes ao F0 e ao F4 do **GPIOPCTL** do Port F.



Pag 788

Zerar estes bits



- GPIO
- PLL
- SysTick
- Referências



GPIO

Exercício (Passo a Passo):

4. a) Como vamos utilizar os pinos J0 e J1 como entrada digital, zerar os bits 0 e 1 do **GPIODIR** do Port J.

GPIO_PORTJ_AHB_DIR_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								DIR							
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
															0	0
															J1	J0

Pag 760

Zerar estes bits

UTFPR

- GPIO
- PLL
- SysTick
- Referências



GPIO

Exercício (Passo a Passo):

4. b) Como vamos utilizar os pinos F0 e F4 como saída digital, **setar** os bits 0 e 4 do **GPIODIR** do Port F.

GPIO_PORTF_AHB_DIR_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	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
												F4	F3	F2	F1	F0

Pag 760

Setar estes bits



GPIO

Exercício (Passo a Passo):

5. a) Como vamos utilizar os pinos J0 e J1 como GPIO, desabilitar a função alternativa, **zerar** pelo menos os bits 0 e 1 do **GPIOAFSEL** do Port J.

GPIO_PORTJ_AHB_AFSEL_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								AFSEL							
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
															J1	J0

Pag 770

Zerar estes bits



GPIO

Exercício (Passo a Passo):

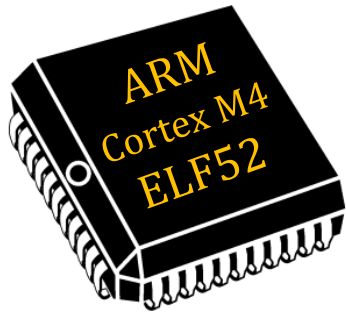
5. b) Como vamos utilizar os pinos F0 e F4 como GPIO, desabilitar a função alternativa, **zerar** pelo menos os bits 0 e 4 do **GPIOAFSEL** do Port F.

GPIO_PORTF_AHB_AFSEL_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	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	-	-	-	0	-	-	-	0
												F4	F3	F2	F1	F0

Pag 770

Zerar estes bits



GPIO

Exercício (Passo a Passo):

6. a) Como vamos utilizar os pinos J0 e J1 como GPIO, habilitar a função digital, **setar** os bits 0 e 1 do **GPIODEN** do Port J.

GPIO_PORTJ_AHB_DEN_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								DEN							
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	-	-	-	-	-	-	1	1
															J1	J0

Pag 782

Setar estes bits

UTFPR

- GPIO
- PLL
- SysTick
- Referências



GPIO

Exercício (Passo a Passo):

6. b) Como vamos utilizar os pinos F0 e F4 como GPIO, habilitar a função digital, **setar** os bits 0 e 4 do **GPIODEN** do Port F.



- GPIO
- PLL
- SysTick
- Referências

GPIO_PORTF_AHB_DEN_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	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	-	-	-	1	-	-	-	1
												F4	F3	F2	F1	F0

Pag 782

Setar estes bits



GPIO

Exercício (Passo a Passo):

7. Como vamos utilizar os pinos J0 e J1 como entrada para chaves e não há resistor de *pull-up* externos, habilitar os *pull-up* internos, **setar** os bits 0 e 1 do **GPIOPUR** do Port J.

GPIO_PORTJ_AHB_PUR_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								PUE							
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	-	-	-	-	-	-	1	1

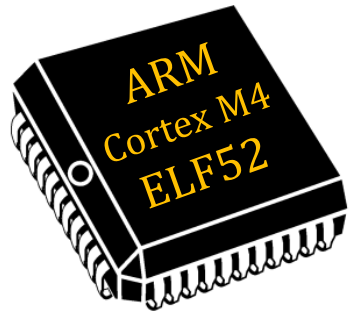
Setar estes bits J1 J0

Pag 777

Obs: Como os pinos F0 e F4 são saídas, não faz sentido habilitar os *pull-up* para eles.

UTFPR

- GPIO
- PLL
- SysTick
- Referências



GPIO

Exercício (Passo a Passo):

Em relação à leitura das chaves e escrita nos LEDs, devemos utilizar os registradores **GPIO DATA**. Para ler as chaves, vamos ler o registrador **GPIO DATA** do Port J. Para acender ou apagar os LEDs vamos escrever no registrador **GPIO DATA** do Port F.



- GPIO
- PLL
- SysTick
- Referências



GPIO

Exercício (Passo a Passo):

Como as chaves estão conectadas em resistores de *pull-up* internos, se as chaves estão soltas retornam o valor 1. Se as chaves estão pressionadas, retornam o valor 0, no registrador **GPIO_DATA** do Port J.

GPIO_PORTJ_AHB_DATA_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								DATA							
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

Se a chave **USR_SW1**, que está ligada ao pino **J0**, estiver solta, o bit 0, quando lido, retorna 1, se ela estiver pressionada o bit 0 retorna 0. Se a chave **USR_SW2**, que está ligada ao pino **J1**, estiver solta, o bit 1, quando lido, retorna 1, se ela estiver pressionada o bit 1 retorna 0.



GPIO

Exercício (Passo a Passo):

Como ativamos os dois pinos do port J, J0 e J1, de uma forma geral não conseguimos ler os bits separadamente. Só conseguimos ler **GPIO DATA** do Port J inteiro, que pode retornar os valores:

GPIO DATA (Port J)	Estado das chaves	
	USR_SW1 (J0)	USR_SW2 (J1)
11	SOLTA	SOLTA
10	PRESSIONADA	SOLTA
01	SOLTA	PRESSIONADA
00	PRESSIONADA	PRESSIONADA



- GPIO
- PLL
- SysTick
- Referências



GPIO

Exercício (Passo a Passo):

Para escrever nos LEDs, LED3 e LED4, que estão ligados respectivamente aos pinos F4 e F0, devemos escrever diretamente no Port F.

GPIO_PORTF_AHB_DATA_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								DATA							
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
												F4	F3	F2	F1	F0

Escrevendo 0 no bit F4, apaga-se o LED3. Escrevendo 1 neste bit, acende-se o LED3. Escrevendo 0 no bit F0, apaga-se o LED4, escrevendo 1 acende-se o LED4. Deve-se tomar cuidado, entretanto para ao escrever em um LED não interferir o outro (escrita amigável).



- GPIO
- PLL
- SysTick
- Referências



- GPIO

- PLL

- SysTick

- Referências

PLL

Função PLLINIT

- Normalmente a velocidade de execução de um microcontrolador pode ser determinada pelo cristal externo, no caso do kit é **25MHz**
- Muitos microcontroladores possuem um **PLL** que possibilita **ajustar a velocidade** de execução por *software*
- Normalmente a frequência é um *tradeoff* entre velocidade de execução e potência elétrica
- Para informações avançadas sobre o gerenciamento de *clock* (Seção 5.2.5 do DS)
- PLLINIT: Configura o *clock* para **80MHz**



- GPIO
- PLL
- SysTick
- Referências

SYSTICK

Função: SysTick_Wait1ms

- Periférico **contador** simples para gerar **atrasos** e gerar interrupções periódicas em todos os Cortex-M.
- Deve-se carregar esse contador com um valor de contagem.
- A cada **tick** (ciclo de *clock*), a contagem é decrementada.
- Se o PLL estiver configurado para **80MHz** então, o contador decrementa a cada **12,5 ns**.
- Quando o contador chega a 0, o periférico notifica que estourou a contagem.
- Função **SysTick_Wait1ms**: *delay* múltiplo de 1ms.



- GPIO
- PLL
- SysTick
- Referências

Exercício

Piscar um LED

Baseando-se no exemplo anterior, criar um projeto que pisque um LED (LED1 no pino PN1) a cada intervalo, quando pressionado um botão (USR_SW1 no pino PJ0).

- Baixar o projeto [GPIO2](#) do site da disciplina;
- Abrir o projeto no Keil MDK;
- Fazer um **fluxograma** do problema proposto; (Opcional)
- Modificar o arquivo gpio.s para inicializar os GPIO para uma chave e um LED;
- Modificar o arquivo main.s para fazer o que foi pedido no enunciado;
- Primeiramente faça apenas o LED acender;
- Depois que esta parte estiver pronta, faça o LED piscar;
- Para fazer o LED piscar utilize a rotina SysTick_Wait1ms.

<http://www.elf52.daeln.com.br/Labs/GPIO.pdf>



Referências:



- GPIO
- PLL
- SysTick
- **Referências**

Atividade Prática 2: Adaptado por Prof. DaLuz

http://www.elf52.daeln.com.br/Labs/Laboratorio_AP2.pdf

* Refs ↔ Renesas.com, Pixabay.com, wikimedia.org, flickr, community.arm.com, Undergraduated course Renesas / CWS71-Sistemas Embarcados (Prof. Douglas P. B. Renaux e Robson Linhares), ytchannel Gustavo W. Dernardin, *ARMv7-M Architecture Reference Manual*, CSW40-Sistemas Microcontrolados – Prof. Peron, toshiba.semicon-storage.com, microntrollerslab.com, lfelectronics.com.br