Sistemas Microcontrolados

UART

Prof. Guilherme Peron

Funções Alternativas

- Muitos dos terminais (pinos) das portas de E/S do microcontrolador TM4C1294 possuem funções alternativas que dão acesso direto ao hardware de periféricos, tais como:
  - Interfaces seriais;
  - Conversores analógico-digital;
  - Temporizadores/contadores de eventos.

Funções Alternativas

- O datasheet do TM4C1294 lista as funções alternativas por pino de E/S na Tabela 10-2 (p. 743).
  - Esta tabela deve ser utilizada em conjunto com as informações do capítulo específico do manual sobre o periférico que se deseja utilizar

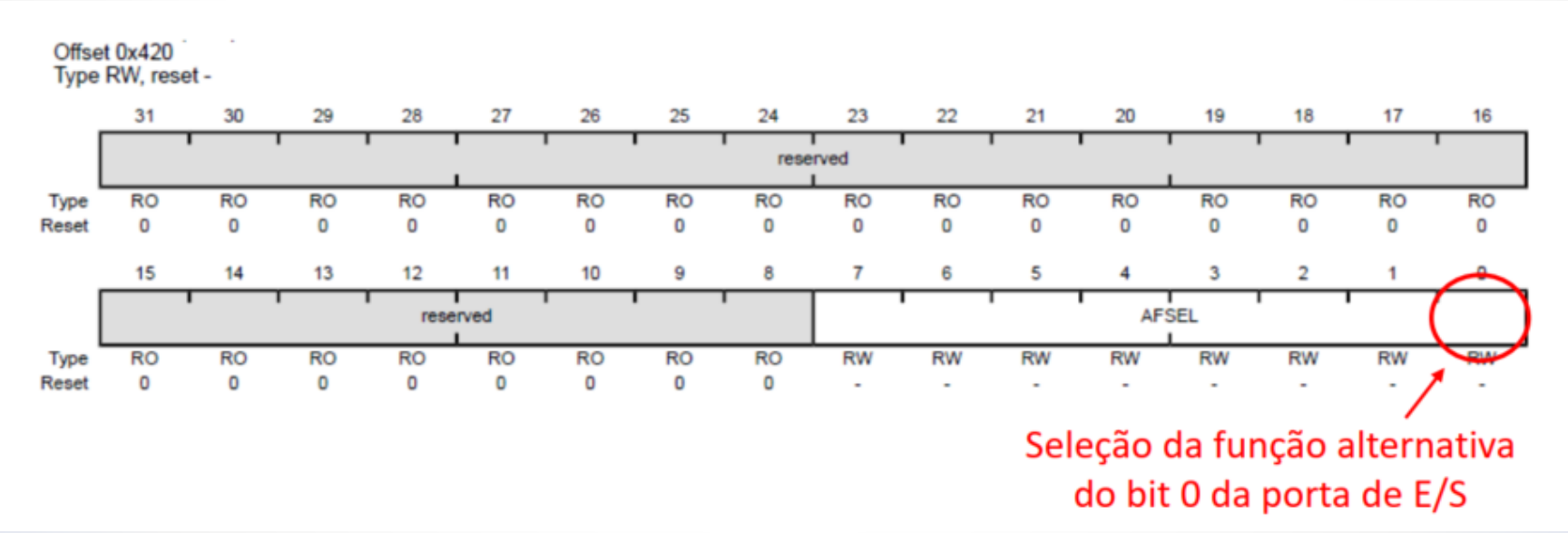
Funções Alternativas

IO	Pin	Analog or Special Function[a]	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[b]											
			1	2	3	4	5	6	7	8	11	13	14	15
PA0	33	-	U0Rx	I2C9SCL	T0CCP0	-	-	-	CAN0Rx	-	-	-	-	-
PA1	34	-	U0Tx	I2C9SDA	T0CCP1	-	-	-	CAN0Tx	-	-	-	-	-
PA2	35	-	U4Rx	I2C8SCL	T1CCP0	-	-	-	-	-	-	-	-	SSI0Clk
PA3	36	-	U4Tx	I2C8SDA	T1CCP1	-	-	-	-	-	-	-	-	SSI0Fss
PA4	37	-	U3Rx	I2C7SCL	T2CCP0	-	-	-	-	-	-	-	-	SSI0XDAT0
PA5	38	-	U3Tx	I2C7SDA	T2CCP1	-	-	-	-	-	-	-	-	SSI0XDAT1
PA6	40	-	U2Rx	I2C6SCL	T3CCP0	-	USB0EPEN	-	-	-	-	SSI0XDAT2	-	EPI0S8
PA7	41	-	U2Tx	I2C6SDA	T3CCP1	-	USB0PFLT	-	-	-	USB0EPEN	SSI0XDAT3	-	EPI0S9

- Exemplo: o pino **PA0** pode ser configurado como os seguintes sinais: **U0Rx** (UART0) ou **I2C9SCL** (I2C9) ou **T0CCP0** (GPTM0) ou **CAN0Rx** (CAN0)

Funções Alternativas

- As funções alternativas para cada pino podem ser definidas no seu respectivo registrador GPIOAFSEL (GPIO Alternate Function Select) – p. 771



Funções Alternativas

- Uma vez definidas funções alternativas de um pino, a especificação das funções deve ser feita no seu respectivo registrador GPIO PCTL (GPIO Port Control) – p. 788

Comunicação Paralela x Serial

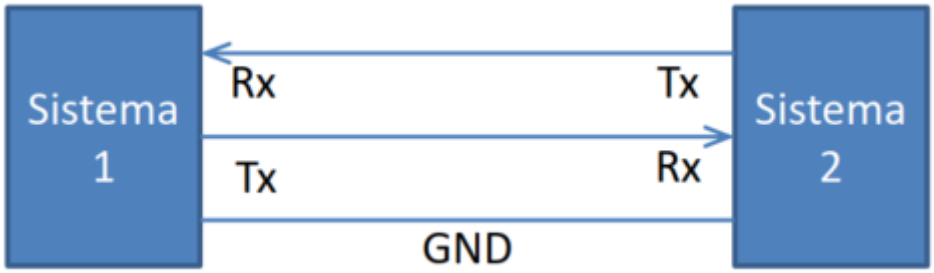
- Interface paralela: todos os bits de informação são transferidos simultaneamente utilizando canais separados para cada bit
- Interface serial: os bits de informação são transferidos um a um ao longo do tempo utilizando um único canal
  - Síncrona (canal de dados + sinal de sincronismo);
  - Assíncrona (apenas canal de dados).

Alguns Tipos de Serial

- CAN – Controller Area Network;
- I2C – Inter-Integrated Circuit;
- SSI (SPI) – Synchronous Serial Interface;
- USB – Universal Serial Bus;
- USART – Universal Synchronous/Asynchronous Receiver/Transmiter;
- **UART – Unversal Asynchronous Receiver/Transmitter.**

UART

- Sistema de comunicação serial full-duplex que utiliza apenas dois sinais:
  - TX (Transmissão);
  - RX (Recepção).



Funcionamento - TX

1) Quando inativo, o transmissor mantém a linha de dados em nível ALTO;

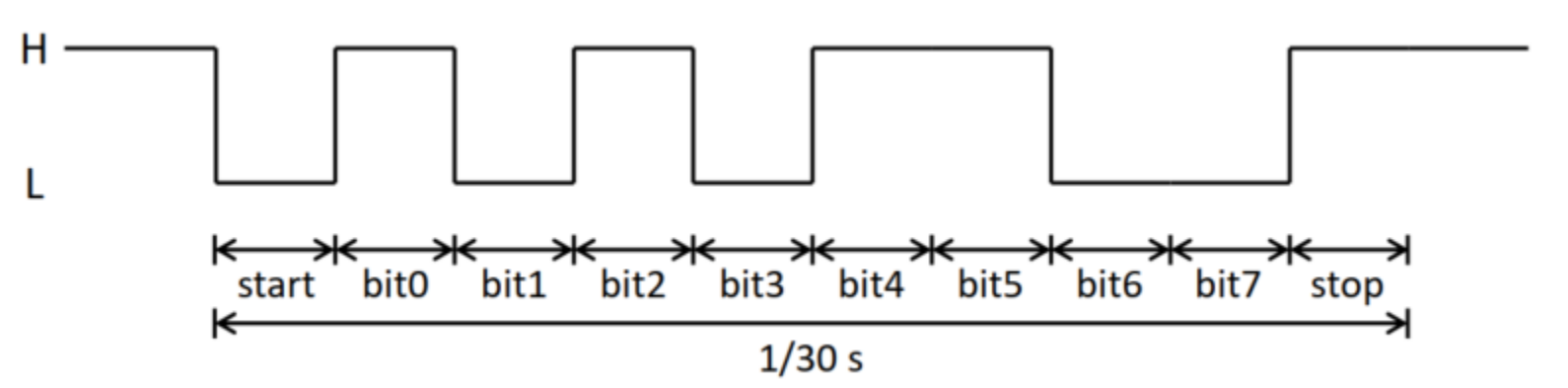
2) O início de transmissão é marcado por um start bit em nível BAIXO;

3) O pacote de dados é transmitido após o start bit, começando pelo bit menos significativo (lsb);

4) O final de transmissão é marcado por um ou dois stop bits em nível ALTO;

Funcionamento - TX

- A duração de cada bit é determinada pela taxa de transmissão, dada em bits por segundo.
- Exemplo de transmissão do caracter ASCII ‘5’ (0x35) com 8 bits de dados e 1 stop bit a uma taxa de transmissão de 300 bps.



Funcionamento - RX

1) Quando nada está sendo recebido, a linha de dados permanece em nível ALTO;

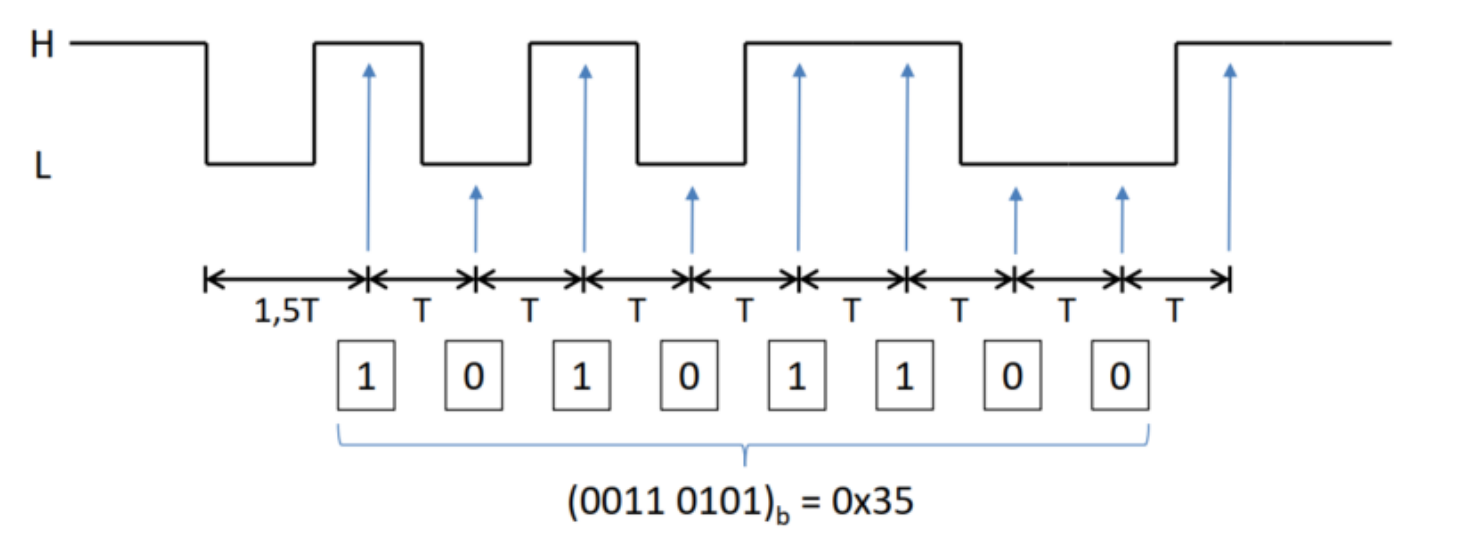
2) O receptor reconhece a borda de descida do start bit e sincroniza seu sinal de clock;

3) Após 1,5 período de clock, o receptor começa a amostrar a linha de dados a cada clock;

4) A última amostragem corresponde ao último stop bit (nível ALTO).

Funcionamento - RX

- Se as frequências de clock do transmissor e do receptor forem iguais, as amostragens serão efetuadas no meio da duração de cada bit:



Funcionamento

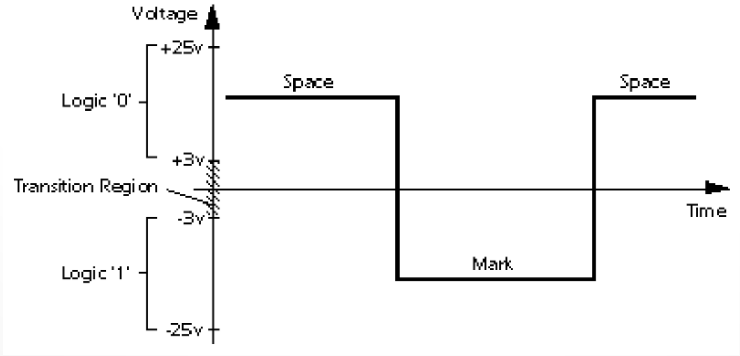
- Configurações do transmissor e do receptor devem ser iguais:
  - Taxa de transmissão (frequência de clock): há uma tolerância de aproximadamente ± 5%;
  - Número de bits de dados;
  - Número de stop bits;
  - Bit de paridade.

Bit de Paridade

- Pode ser acrescentado ao dado, destinado a detecção simples de erros. A paridade simples detecta 1 erro, mas não corrige.
  - Paridade par: número par de bits no estado 1, incluindo o bit de paridade;
  - Paridade ímpar: número ímpar de bits no estado 1, incluindo o bit de paridade
  - Exemplo: caractere ASCII ‘A’ é 0x41: 01000001b tem 2 bits no estado 1
    - Se for usada a paridade ímpar: acrescenta-se mais um bit ‘1’ (3 bits ‘1’ é ímpar): 101000001b
    - Se for usada a paridade par: acrescenta-se mais um bit 0 (2 bits 1 é par): 001000001b

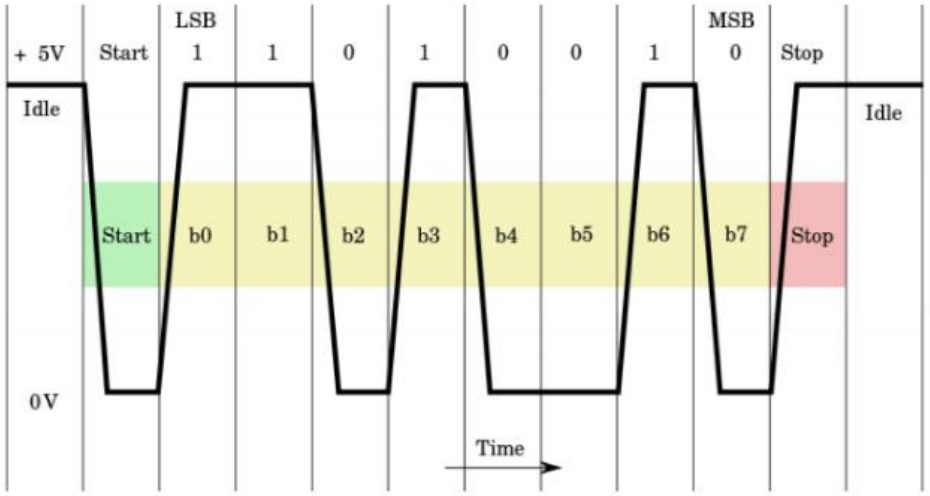
RS-232C

- O padrão da UART TTL opera de 0 a +5V
  - 0V: Nível lógico 0
  - 5V: Nível lógico 1
- Há, no entanto, o padrão RS-232C adotado pelos computadores que operam com tensões
  - Nível lógico 0: Entre +3V e +25V (usualmente +12V)
  - Nível lógico 1: Entre -3V e -25V (usualmente -12V)

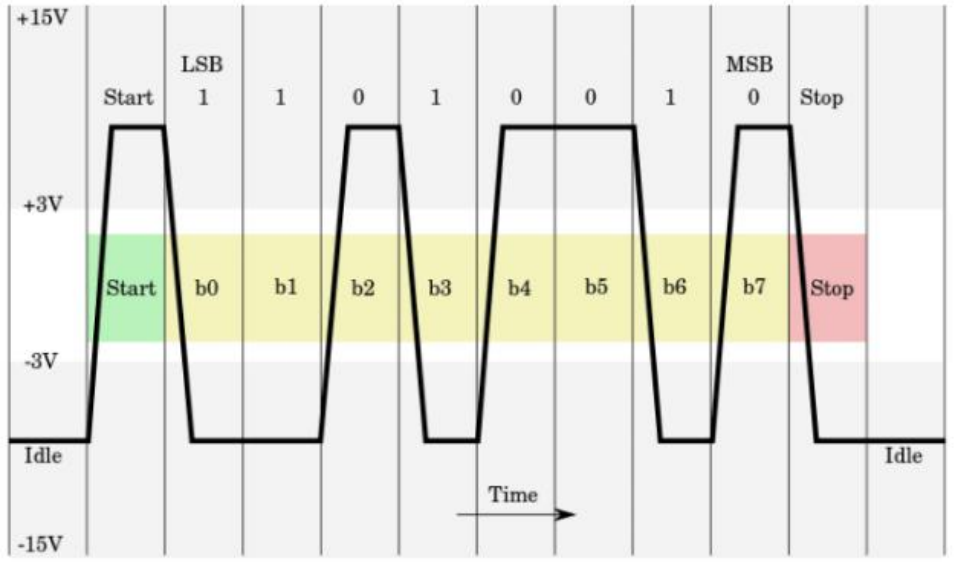


TTL x RS232

TTL



RS232

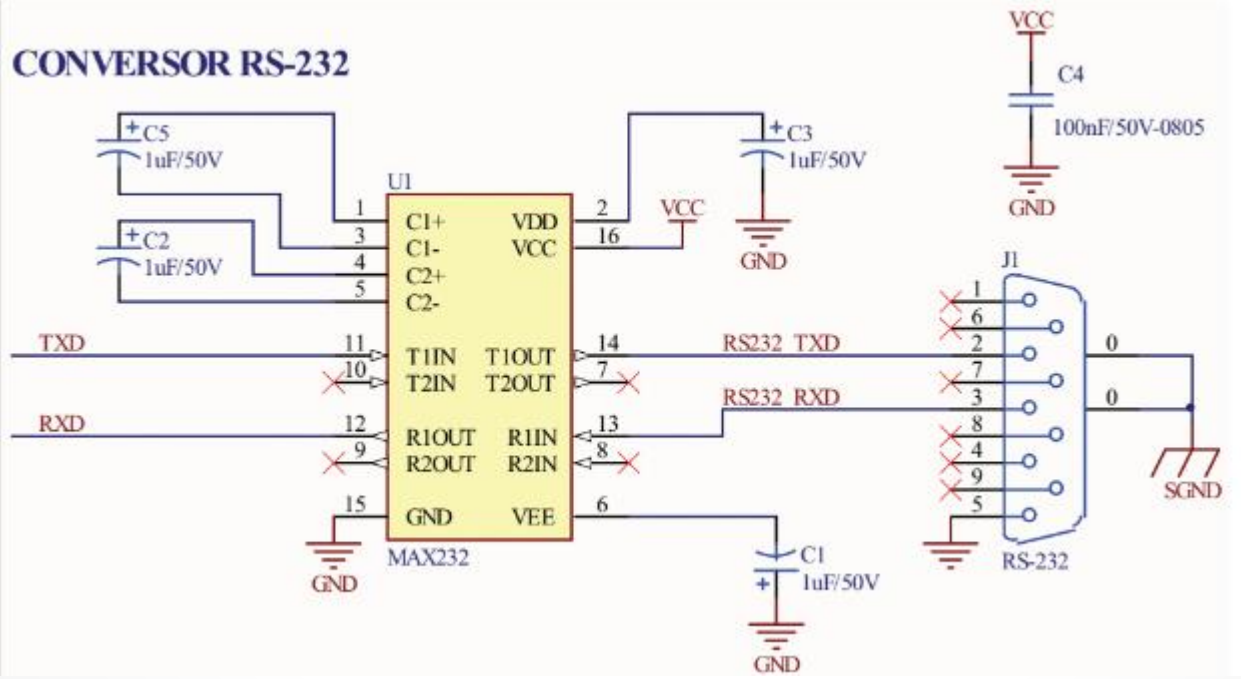


RS-232C

- Além disso, define sinais adicionais de *handshake* entre dois tipos de dispositivos:
  - DTE = data terminal equipment (computador)
  - DCE = data communication equipment (modem)
- Além dos sinais TX e RX, são definidos os sinais RTS/CTS, DSR/Dtr, DCD e RI para controle de fluxo de dados por hardware.

Camada Física

- Para conversão entre TTL e RS232, utiliza-se o circuito MAX232



Substituições

- Os novos computadores não possuem mais porta serial
- Utiliza-se conversores:
  - serial → USB
  - serial → bluetooth

Padrões de Config

- Velocidade *(Baud-Rate)* [bits/s]
  - 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
- Número de bits
  - 5, 6, 7 ou 8
- Paridade
  - Par, ímpar ou sem paridade
- Stop Bits
  - 1 ou 2

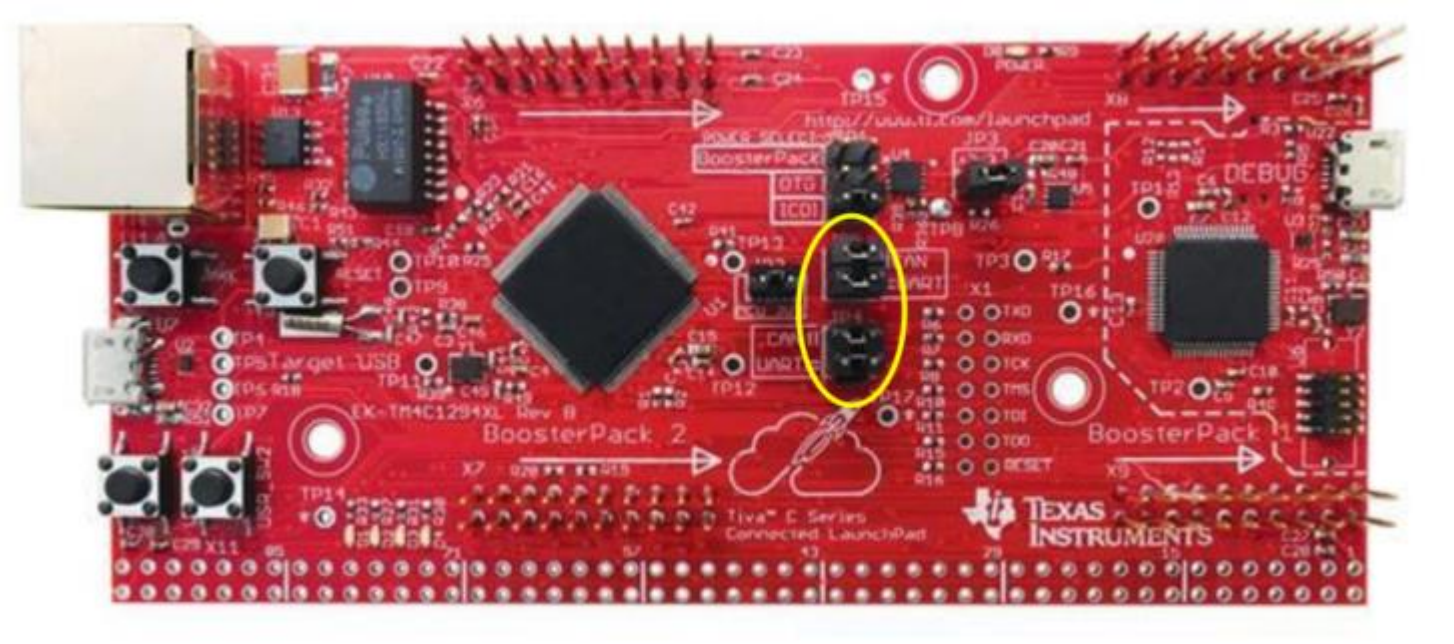
UART no kit EK-TM4C1294XL

Kit EK-TM4C1294XL

- O depurador do kit possui um canal de comunicação serial detectado pelo PC hospedeiro como uma porta COM do tipo “Stellaris Virtual Serial Port”
- Mesma interface do *debugger*
- Na Tiva, a UART0 está conectada aos pinos PA0 (U0Rx) e PA1 (U0Tx) e que por sua vez já passam por um conversor USB

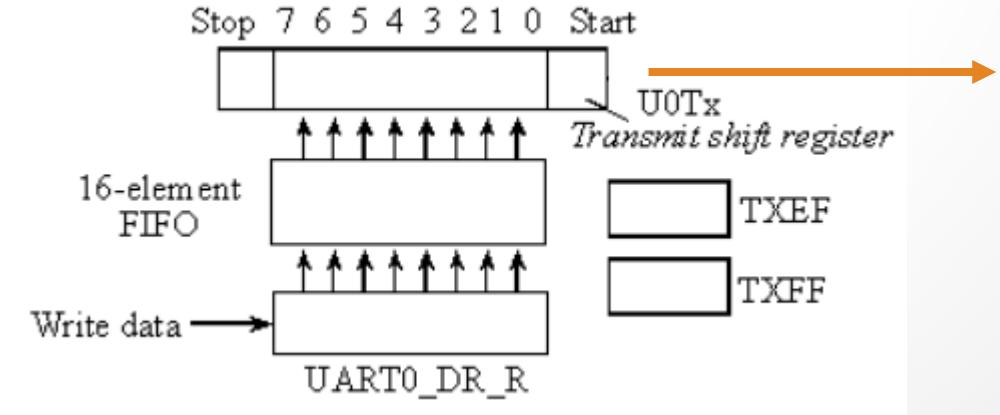
Kit EK-TM4C1294XL

- Os jumpers JP4 e JP5 determinam se a UART0 (=) ou a UART2 (II) estará conectada ao canal serial do depurador.



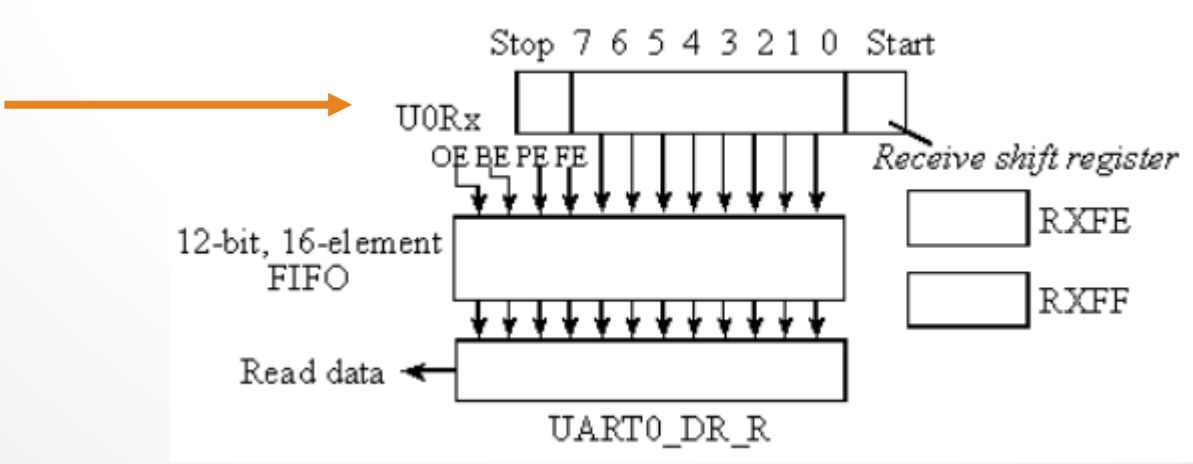
Transmissão UART

- Os transmissores têm uma FIFO de 16 elementos, monitorada por dois flags TXFF (TX FIFO Full) e TXFE (TX FIFO Empty);
- Uma transmissão é efetuada por meio de uma escrita no registrador de dados da UART;
- O software deve verificar se a fila de TX não está cheia antes de realizar uma escrita.



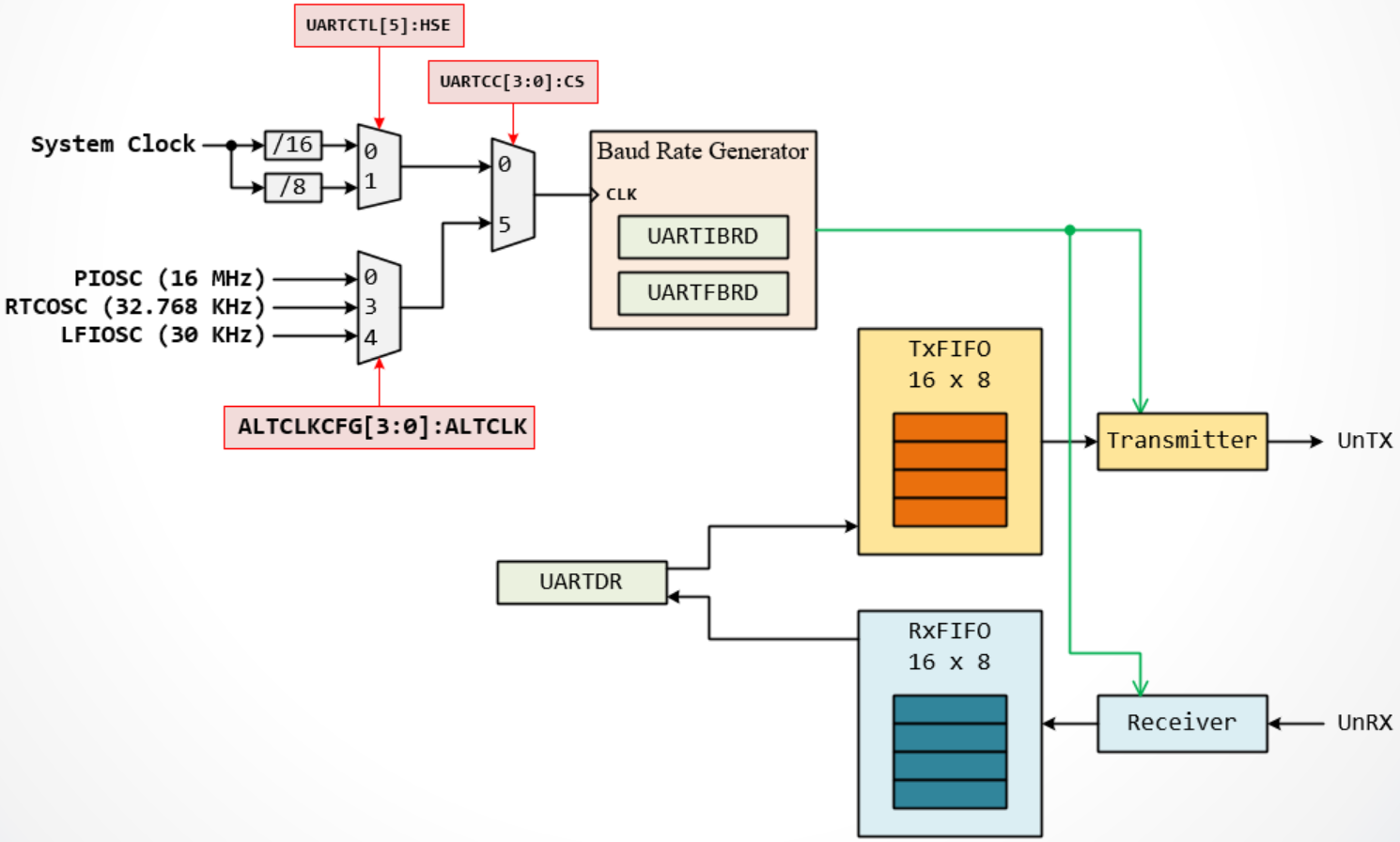
Recepção UART

- Os receptores têm uma FIFO de 16 elementos, monitorada por dois flags RXFF (RX FIFO Full) e RXFE (RX FIFO Empty);
- Uma recepção é efetuada por meio de uma leitura no registrador de dados da UART
- O software deve verificar se a fila de RX não está vazia antes de realizar uma leitura.



Esquemático UART

- 8 UARTs



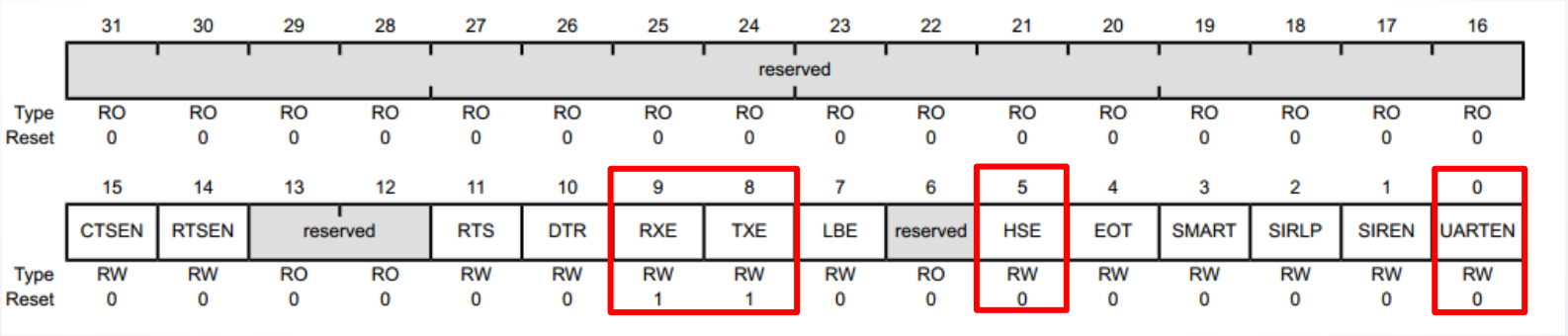
Fonte: https://bityli.com/WCQbk

Registradores UART

- Para ativar a UART, o *clock* da respectiva UART tem que ser ativado no registrador RCGCUART. Cada bit ativa uma UART.
- Verificar o bit da UART respectiva no registrador PRUART para saber se está pronta para o uso.

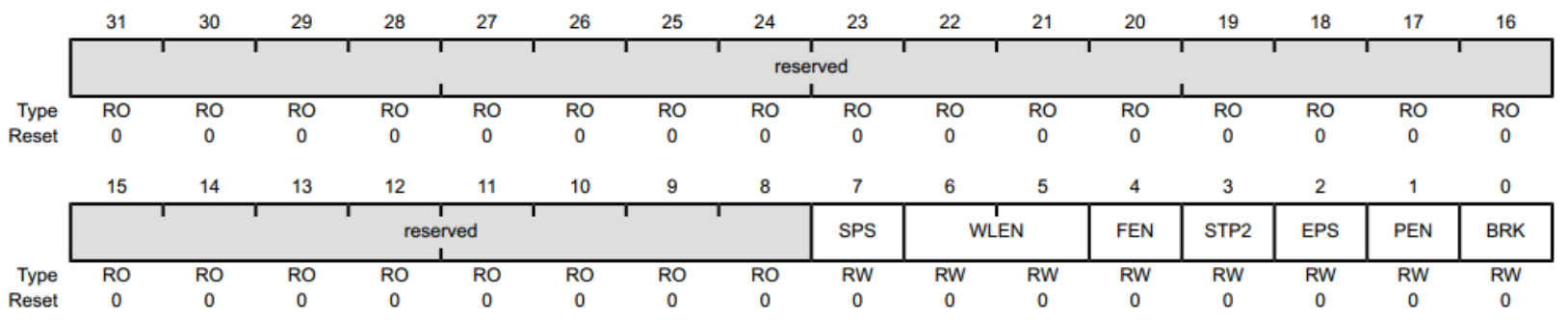
Registradores UART

- O registrador UARTCTL controla a UART:
  - TXE é o bit para habilitar o transmissor;
  - RXE é o bit para habilitar o receptor;
  - UARTEN habilita a UART como um todo. **Antes de realizar a configuração deve-se desabilitar este bit.**



Registradores UART

- O registrador UARTLCRH:
  - WLEN controla o tamanho da palavra a ser enviada
    - 5 a 8 bits
  - FEN habilita as filas;
  - STP2 habilita 2 *stop bits*;
  - EPS e PEN controla e habilita a paridade.



Geração de *Baud-Rate*

- Para o Baud-Rate há dois registradores:
  - UARTIBRD: 16 bits para a parte inteira;
  - UARTFBRD: 6 bits para a parte fracionária.
    - Resolução de $1/(2^6) = 1/64$

$$BRD = \frac{UARTSysClk}{ClkDiv \times BaudRate}$$

$$ClkDiv = \begin{cases} 16, \text{ se bit HSE de UARTCTL=0} \\ 8, \text{ se bit HSE de UARTCTL=1} \end{cases}$$

Geração de *Baud-Rate*

Exemplo de cálculo:

- Para um *baudrate* de 19200bps, um clock de 80MHz e ClkDiv = 16 (HSE = 0)

$$BRD = \frac{UARTSysClk}{ClkDiv \times BaudRate} = \frac{80000000}{16 \times 19200} = 260{,}4167$$

UARTIBRD = 260

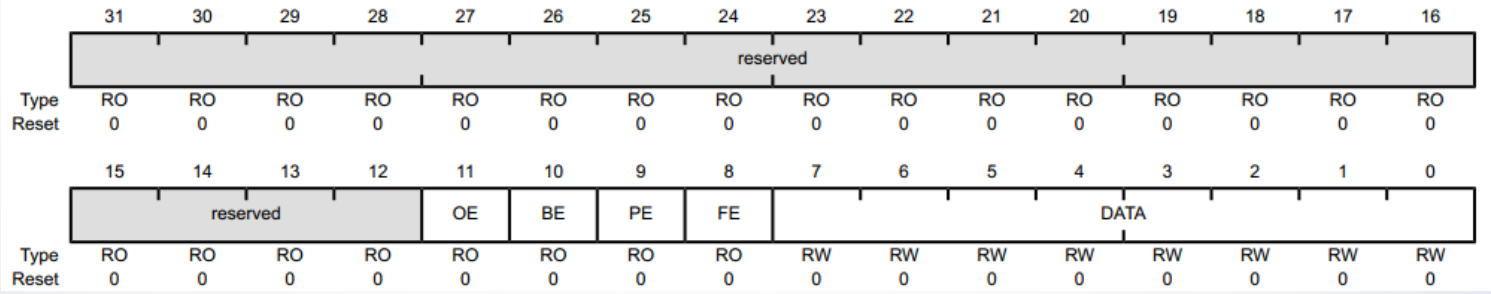
UARTFBRD = BFDF*64 = 0,4167*64 = 26,67 = 27

Geração de *Baud-Rate*

- ATENÇÃO: uma alteração no divisor de baud-rate deve ser seguida por uma escrita no registrador UARTLCRH para as alterações fazerem efeito.

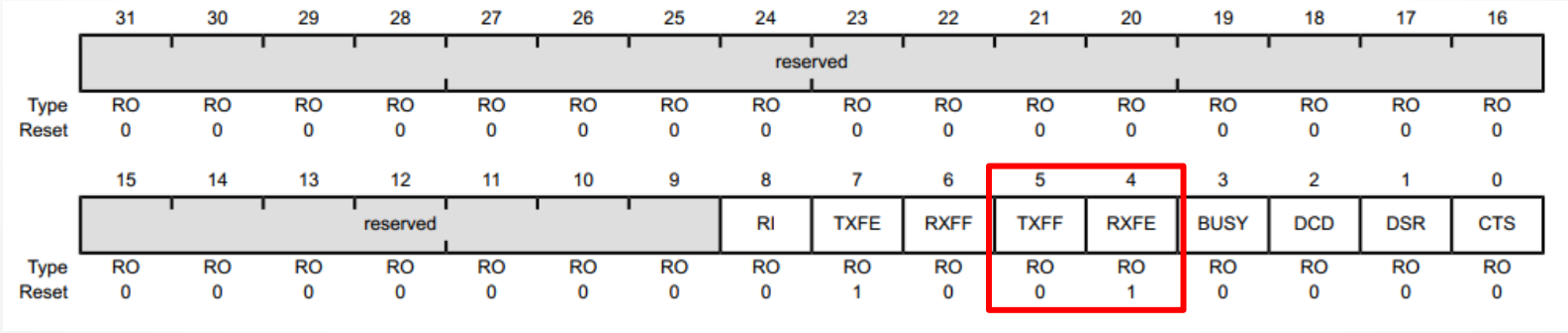
Registradores UART

- O registrador UARTDR é usado para transmitir o caractere (5 a 8 bits) ou para receber o caractere.
  - Escrita: transmissão de dados (TX)
  - Leitura: recepção de dados (RX) + sinalizadores de erro:
    - OE: ***overrun*** → recepção realizada com a FIFO cheia;
    - BE: ***break*** → a linha de recepção ficou presa em 0 por um tempo de recepção completo (start a stop bit);
    - PE: **parity** → paridade recebida não corresponde
    - FE: **frame** → caractere recebido não tem um stop bit válido.



Registradores UART

- Os *status* das *flags* das FIFOs podem ser vistos no registrador UARTFR.
  - TXFF: indica que não é possível escrita em UARTDR;
  - RXFE: indica que não é possível leitura de UARTDR;



Passo-a-passo (UART)

1. Habilitar o clock no módulo UART no registrador RCGCUART (cada bit representa uma UART) e esperar até que a respectiva UART esteja pronta para ser acessada no registrador PRUART (cada bit representa uma UART).
2. Garantir que a UART esteja desabilitada antes de fazer as alterações (limpar o bit UARTEN) no registrador UARTCTL (Control).
3. Escrever o *baud-rate* nos registradores UARTIBRD e UARTFBRD

Passo-a-passo (UART)

4. Configurar o registrador UARTLCRH para o número de bits, paridade, stop bits e fila
5. Garantir que a fonte de *clock* seja o *clock* do sistema no registrador UARTCC escrevendo 0 (ou escolher qualquer uma das outras fontes de *clock*)
6. Habilitar as flags RXE, TXE e UARTEN no registrador UARTCTL (habilitar a recepção, transmissão e a UART)

Passo-a-passo (GPIO)

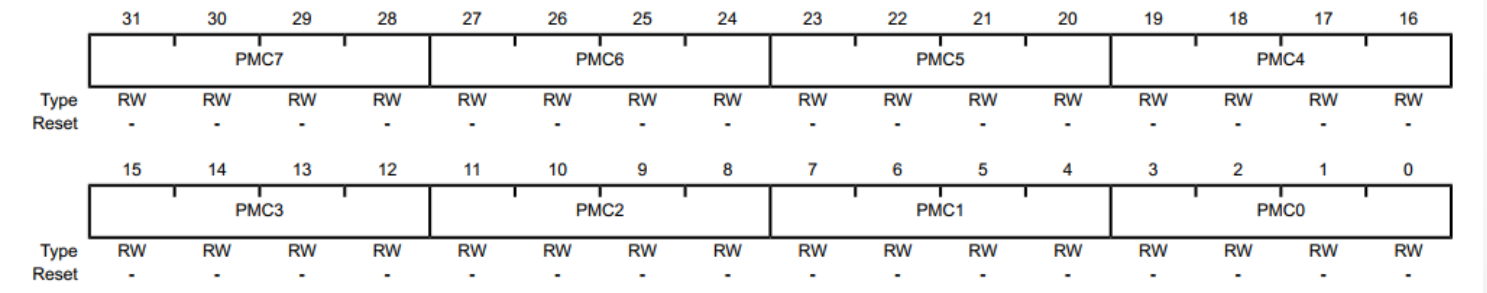
7. Habilitar o clock no módulo GPIO no registrador RCGGPIO (cada bit representa uma GPIO) e esperar até que a respectiva GPIO esteja pronta para ser acessada no registrador PRGPIO (cada bit representa uma GPIO).
8. Desabilitar a funcionalidade analógica no registrador GPIOAMSEL.

Passo-a-passo (GPIO)

9. Escolher a função alternativa dos pinos respectivos TX e RX no registrador GPIOPCTL (verificar a tabela 10-2 no datasheet páginas 743-746)

Lembrando que cada 4 bits deste registrador configura um pino.

Pino 7 Pino 6 Pino 5 Pino 4



Pino 3 Pino 2 Pino 1 Pino 0

Passo-a-passo (GPIO)

9. Escolher a função alternativa dos pinos respectivos TX e RX no registrador GPIOPCTL (verificar a tabela 10-2 no datasheet páginas 743-746)

Por exemplo a UART0 da placa está mapeada nos pinos PA0 e PA1:

IO	Pin	Analog or Special Function[a]	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[b]											
			1	2	3	4	5	6	7	8	11	13	14	15
PA0	33	-	U0Rx	I2C9SCL	T0CCP0	-	-	-	CAN0Rx	-	-	-	-	-
PA1	34	-	U0Tx	I2C9SDA	T0CCP1	-	-	-	CAN0Tx	-	-	-	-	-

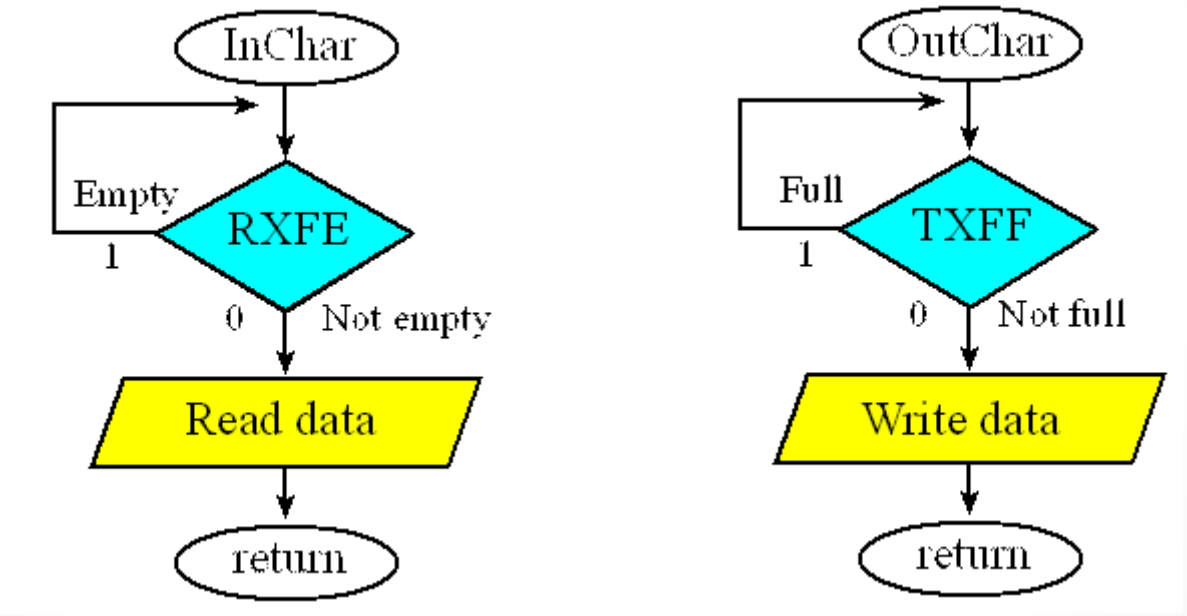
Assim, como a função alternativa do pino 0 e pino 1 é a 1 de cada um então, o registrador GPIOPCTL do Port A (GPIO_PORTA_AHB_PCTL_R) deve receber o valor 0x11 (00010001 em binário).

Passo-a-passo (GPIO)

10. Habilitar os bits de função alternativa no registrador GPIOAFSEL nos pinos respectivos à UART.
11. Configurar os pinos como digitais no registrador GPIODEN.

Recepção e Transmissão

- Para escrita e leitura da porta serial por *busy-flag*:



Recepção e Transmissão

- Para fazer a recepção, criar uma função que:
  - Realiza *polling* no bit RXFE (0x10) *(FIFO empty)* do registrador UARTFR;
    - Enquanto for 1, não há dados para serem lidos;
    - Quando for 0, o conteúdo do registrador UARTDR para uma variável de 8 bits.
- Para fazer a transmissão, criar uma função que:
  - Realiza *polling* no bit TXFF (0x20) *(FIFO full)* do registrador UARTFR;
    - Enquanto for 1, não pode transmitir / FIFO cheia;
    - Quando for 0, copiar o byte a ser enviado para o registrador UARTDR.

Comunicação com o PC

- O funcionamento da UART pode ser verificado utilizando-se um aplicativo emulador de terminal, como o Putty ou Tera Term, instalado no PC hospedeiro do kit.
  - Todo caracter recebido pela serial do PC é mostrado na janela do emulador de terminal e todo caracter correspondente a uma tecla pressionada no teclado do PC é transmitido.