1. UART

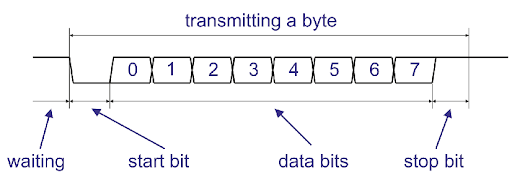
A *universal asynchronous receiver transmitter* (UART) é uma comunicação série largamente usada para comunicar entre dois dispositivos, como sistemas embebidos, microcontroladores e computadores. Na comunicação UART, a comunicação direta entre dois dispositivos faz-se através de duas linhas de dados como representado na figura.

Uma imagem com texto, relógio

Descrição gerada automaticamente

Esquema de ligação entre 2 UARTs

Este periférico transmite dados de forma assíncrona, ou seja, não necessita de um sinal de *clock* para se manter sincronizado. Em vez disso, o emissor acrescenta bits ao pacote a ser transmitidosinalizando o início e o fim dos dados a ser transferidos, tal como mostrado na figura.



Estrutura de um pacote de dados enviado por UART

Quanto ao recetor, após detetar o *start* bit, começará a ler os bits que se seguem a uma determinada frequência, denominado *baud rate*. Expresso em bits por segundo (bps), este é definido como sendo uma medida de velocidade de transmissão. Ambas as UART têm de operar ao mesmo *baud rate* para não ocorrer erros na transmissão, antes da inicialização da comunicação, ambas as UARTs têm de ser configuradas com o mesmo *baud rate* e com a mesma estrutura de pacotes.

Na STM32F767ZI encontram-se disponíveis 8 módulos UART do quais 4 (USART1, USART2, USART3 e USART6) podem ser configurados em modo síncrono. A figura mostra as funcionalidades que cada UART, presente na placa, possui.

Uma imagem com mesa

Descrição gerada automaticamente

Figura de funcionalidades existentes para cada UART

Cada UART pode ser configurada para comunicação assíncrona ou síncrona (no caso das UARTs previamente referidas), pode ser ainda configurada em *full-duplex* ou *half-duplex*, com baud-rate até 27 Mbits/s (quando *clock source* da UART é o *clock* do sistema) ou com deteção automática do baud rate. A estrutura dos pacotes de dados é modificável, podendo escolher ter 7,8 ou 9 bits de dados, 1 ou 2 stop bits e uma definição da ordem da transmissão de dados com MSB-*first* (bit mais significativo primeiro) ou LSB-*first* (bit menos significativo primeiro). Para além disso, existem mais 14 *interrupt flags* para controlo de fluxo de dados e erros. Possui, ainda, a possibilidade de funcionar com o DMA para comunicação continua e para *buffers* de dados recebidos e transmitidos, com o intuito de serem guardados ou adquiridos da SRAM. As UARTs podem também configuradas para a utilização do padrão de comunicação série RS-232 ou o padrão RS-485. Os principais elementos das UARTs localizadas na STM32F767ZI, ~~e~~ bem como as suas interações encontram-se descritas na figura.

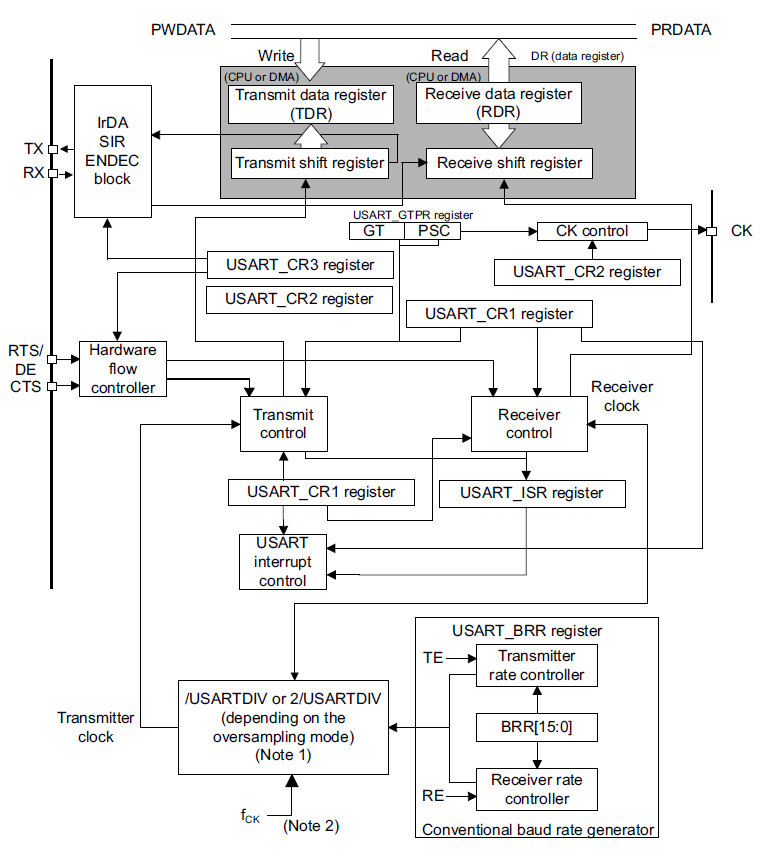


Diagrama de blocos da USARTs

O objetivo da UART neste projeto é para possibilitar a comunicação com um módulo Bluetooth HC-05 [ref?]. Ao consultar o *datasheet* [ref HC-05 Datasheet] deste verifica-se que o formato da comunicação série é: oito bits de dados, um stop bit e nenhum de paridade podendo ainda suportar várias frequências de comunicação. Porém, dado que o modo de operação do módulo utilizado é o mais simples entre os disponíveis (*Data mode* e *AT Commands*), este opera a um *baud rate* de 9600 bits/s.

Deste modo, configurou-se a USART1 para manusear pacotes com oito bits de dados sem paridade, 1 stop bit e a operar um *baud rate* de 9600 bits/s. De seguida, as interrupções da USART1 se encontram ativas para processar os dados assim que chegam ao destino.

1. SPI

A *Serial Peripheral Interface* (SPI) é um protocolo de comunicação de curto alcance criado para a troca de dados entre microcontroladores ou entre microcontroladores e sensores. O SPI é síncrono e necessita que o transmissor e o recetor estejam sincronizados através de uma linha de *clock* partilhada. Este suporta transferências de dados em três modos: *full-duplex* (2 linhas de dados), *half-duplex* (1 linha de dados bidirecional) ou *simplex* (1 linha unidirecional).

Os dispositivos conectados por SPI encontram-se numa relação *master-slave.* O *master* é, tipicamente, um microcontrolador que envia instruções ao *slave*. Este, em vários casos, trata-se de um sensor, um *chip* de memória ou um display.

De modo a proceder à comunicação *full-duplex* entre um dispositivo *master* e um dispositivo *slave,* estes devem ser conectados segundo a configuração apresentada na figura.

Uma imagem com texto

Descrição gerada automaticamente

Esquema de ligação entre Master e Slave.

O protocolo SPI especifica 4 linhas lógicas para a transferência de dados:

* SCLK: *Serial Clock*, uma saída proveniente do *master* para sincronizar as transferências de dados nas linhas MISO e MOSI;
* MOSI: *Master Out Slave In*, Saída de dados do *master* com destino ao *slave*;
* MISO: *Master In Slave Out*, Saída de dados do *slave* com destino ao *master*;
* SS: *Slave Select*, através desta linha o *master* pode selecionar com qual dispositivo pretende comunicar.

A STM32F767ZI possui 6 interfaces SPI que suportam comunicação *full-duplex*, *half-duplex* e *simplex*, permitindo enviar blocos de dados de 4 a 16 bits. Além disso, o *clock* e a ordem dos dadosé configurável por software, existem *interrupt flags* dedicadas à transmissão e receção de dados e cada interface possui 2 FIFOs de 32 bits (uma para receção e outra para a transmissão) com capacidade de uso conjunto com o DMA. Os principais elementos da interface SPI localizada na STM32F767ZI e as suas interações encontram-se descritas na figura.

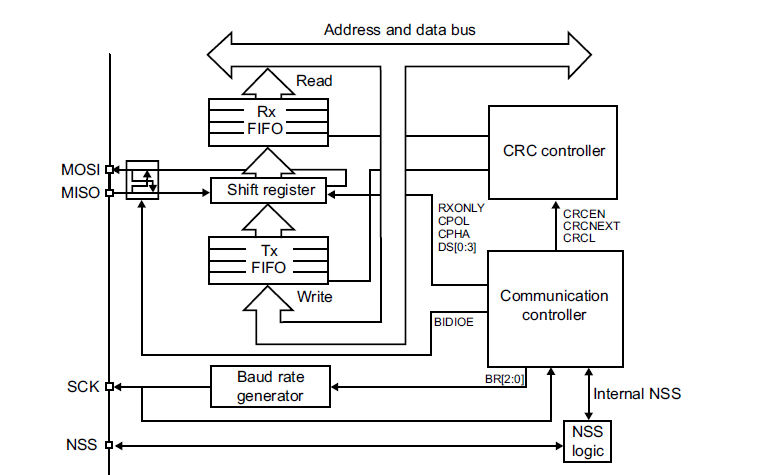


Diagrama de blocos das interfaces SPI

Pelo indicado na *datasheet* do chip MFRC522 [ref NXP ‑ MFRC522], este atua como *slave* e aguenta velocidades de transferência de dados até 10 Mbits/s. Para além disso, é mostrado que durante a receção e transmissão de dados o módulo envia o bit mais significativo primeiro (MSB) e que os dados transmitidos são estáveis durante a transição ascendente do *clock* e são alterados durante a transição descendente.

Como existem 6 interfaces disponíveis, optou-se por utilizar a interface SPI3, dado que os pinos que esta usa para a comunicação não entram em conflito com pinos usados por outros periféricos. Possuindo a informação necessária e consultando o *datasheet* do microprocessador referente ao SPI [ref – RM0410], configurou-se a interface SPI3 para funcionar com dados no formato Motorola com o tamanho de 8 bits em que o MSB é enviado primeiro. Também se configurou o *clock* da interface para a transferência de dados ocorrer a 3,375 Mbits/s ao colocar o *prescaler* a 16 (assumindo que o *clock* do barramento APB1 é 54 Mhz) e configurou-se a polaridade do *clock* (CPOL) a *low* e a fase deste (CPHA) a 0, de modo executar a captura dos dados provenientes do MFRC522 durante a transição ascendente.