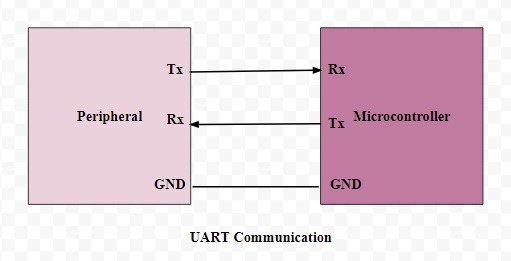
UART

1. Khái niệm

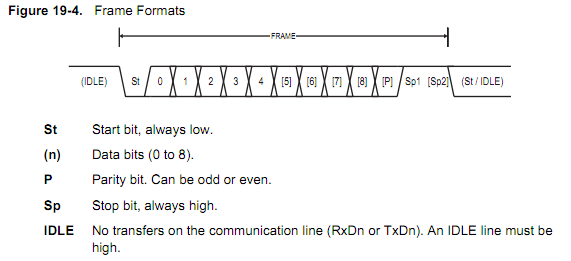
* UART (Universal Asynchronous Reciver/Transmister) một bộ truyền nhận nối tiếp không đồng bộ cho MCU và các thiết bị ngoại vi , sử dụng hai đường Tx và Rx để truyền và nhận dữ liệu và thường được dùng để giao tiếp với các module như Zigbee, Bluetooth, Wifi…
* Chuẩn UART là chuẩn giao tiếp điểm và điểm, nghĩa là trong mạng chỉ có hai thiết bị đóng vai trò là truyền hoặc nhận



1. Cách thức hoạt động của UART

* UART là truyền thông song công tức là tại 1 thời điểm đồng thời có thể truyền và nhận.
* UART là giao thức truyền không đồng bộ : chỉ gồm các đường truyền dữ liệu không có xung clock các thiết bị chỉ có thể hiểu nhau nếu được cấu hình giống nhau (đặc biệt là Baund rate phải được cài đặt giống nhau ở cả gửi và nhận ).

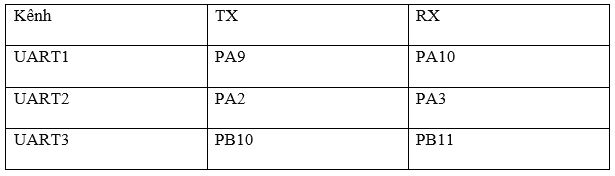
1. Định dạng tập tin



* **Frame:**Ngoài việc giống nhau của tốc độ baud 2 thiết bị truyền nhận thì khung truyền của bên cũng được cấu hình giống nhau. Khung truyền quy định số bit trong mỗi lần truyền, bit bắt đầu “Start bit”, các bit kết thúc (Stop bit), bit kiểm tra tính chẵn lẻ (Parity), … có 2 trường hợp của khung truyền
  + Idle frame: Đường truyền UART ở mức “1”, để xác nhận hiện tại đường truyền dữ liệu trống, không có frame nào đang được truyền đi.
  + Break frame: Đường truyền UART ở mức “0”, để xác nhận hiện tại trên đường truyền đang truyền dữ liệu, có frame đang được truyền đi.
* **Start – Bit:** Start-bit Bit đầu tiên được truyền trong một frame, bit này có chức năng báo cho bên nhận rằng sắp có một gói dữ liệu truyền đến. Đường truyền UART luôn ở trạng thái cao mức “1” cho đến khi chip muốn truyền dữ liệu đi thì nó gởi bit start bằng cách kéo xuống mức “0”. Như vậy start bit giá trị điện áp 0V và phải bắt buộc có bit start trong khung truyền.
* **Data:** Data hay dữ liệu là thông tin mà chúng ta nhận được trong quá trình truyền và nhận. Data trong STM32 có quy định khung truyền là 8bit hoặc 9bit. Trong quá trình truyền UART, bit có trọng số thấp nhất (LSB – least significant bit – bên phải) sẽ được truyền trước và cuối cùng là bit có ảnh hưởng cao nhất (MSB – most significant bit – bên trái).
* **Parity bit:** Parity dùng để kiểm tra dữ liệu truyền có đúng hay không. Có 2 loại Parity đó là Parity chẵn (even parity) và parity lẽ (odd parity). Bit này không bắt buộc.
* **Stop bits:** Stop bits là một bit báo cáo để cho bộ truyền/nhận biết được gói dữ liệu đã được gửi xong. Stop bits là bit bắt buộc phải có trong khung truyền.
* **Baudrate:** Số bit truyền được trong 1s, ở truyền nhận không đồng bộ thì ở các bên truyền và nhận phải thống nhất Baudrate. Các thông số tốc độ Baudrate thường hay sử dụng dể giao tiếp với máy tính là :600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 56000, 57600, 115200….

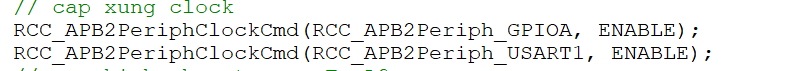
1. Lập trình uart trên stm32

\*STM32F1 có 3 kênh UART:

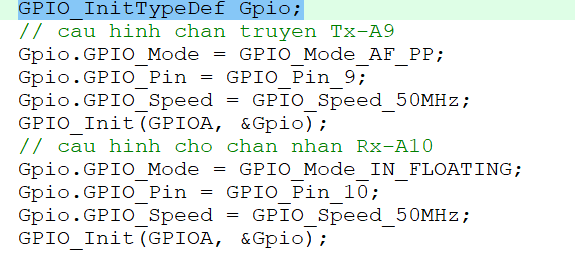


Vd viết trương trình truyền chuỗi kí tự

* Cấp xung clock

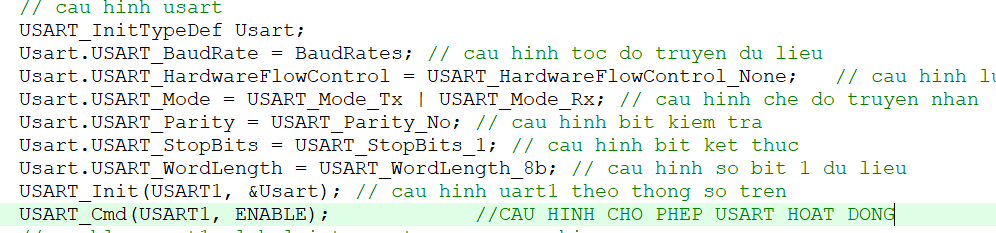


* Cấu hình thông số cho chân truyền –TX và nhận –RX



Cấu hình cho chân A9 là chân để truyền dữ liệu và chân A10 là chân để nhận dữ liệu.

* Cấu hình cho UART và cho phép hoạt động



Ta sử dụng các hàm:

+) USART\_BaudRate = A để cấu hình tốc độ truyền bit trong 1 s ( thông thường là A = 9600 ).

+) USART\_HardwareFlowControl để cấu hình luồng điều khiển phần cứng (chủ yếu không sử dụng nên = USART\_HardwareFlowControl\_None )

+) USART\_Mode = A để cấu hình các chế độ

A =

* USART\_Mode\_Tx : chế độ truyền
* USART\_Mode\_Rx: chế độ nhận

+) USART\_Parity = A để cấu hình bit kiểm tra

A =

* USART\_Parity\_No : Không sử dụng bít kiểm tra
* USART\_Parity\_Even : Sử dụng bít kiểm tra chẵn
* USART\_Parity\_Odd : Sử dụng bit kiểm tra lẻ

+) USART\_StopBits = A để cấu hình bit kết thúc

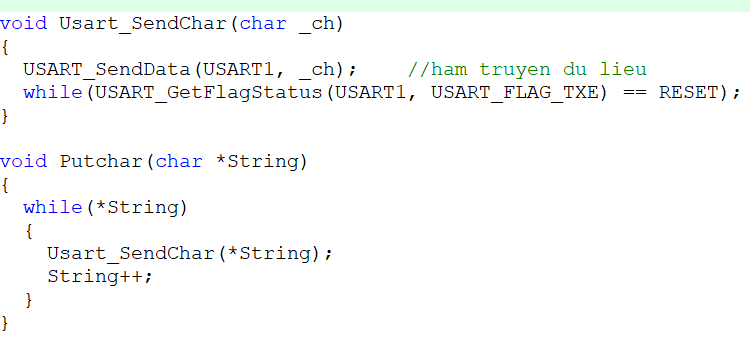
* USART\_StopBits\_1 số bit kết thúc là 1.

+) USART\_WordLength = C: để cấu hình số bit dữ liệu

* USART\_WordLength\_8b : số bit là 8
* USART\_WordLength\_9b : số bit là 9

+) USART\_Cmd(B,C); cấu hình cho phép bộ uartx hoạt động

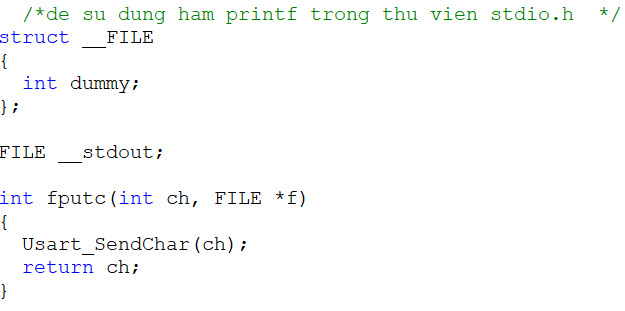
* B: uartx
* C : Enable / disable
* Chương trình gửi chuỗi kí tự



Sử dụng hàm truyền dữ liệu : USART\_SendData(A,B)

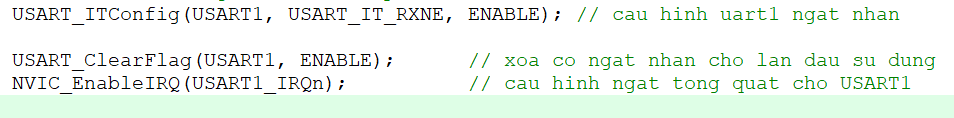
* A : UART1 : sử dụng bộ uart1
* B : data : dữ liệu muốn truyền đi

Hoặc ta có thể sử dụng hàm printf trong thư viện “stdio.h”



Vd viết chương trình truyền nhận dữ liệu điều khiển led

* Đầu tiên ta cần cấu hình như truyền chuỗi kí tự
* Để nhận dữ liệu một cách hiệu quả và chính xác ta cần sử dụng ngắt USART.
* Để sử dụng ngắt ta cần cấu hình ngắt cho chức năng bằng hàm



* USART\_ITConfig(A,B,C) : cấu hình (không ) cho phép ngắt

A : USARTx

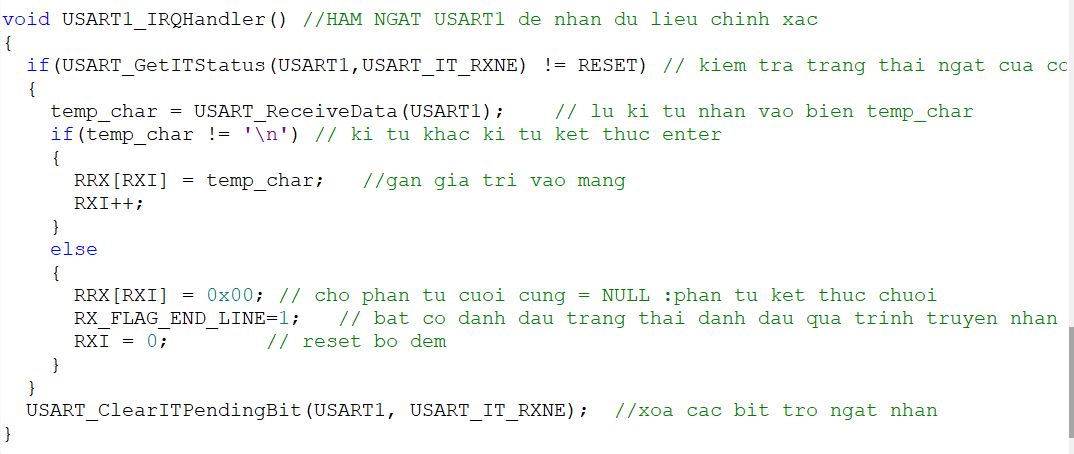
B: USART\_IT\_RXNE : ngắt nhận

C : Enable : cho phép

* NVIC\_EnableIRQ(A) : cấu hình ngắt tổng quát

A: USART1\_IRQn : ngắt tổng quát cho usart1

* Sau đó là sử dụng hàm ngắt nhận



* + Kiểm tra trạng thái của cờ bật qua hàm USART\_GetITStatus(USART1,USART\_IT\_RXNE) != RESET
  + Gán giá trị nhận vào biến temp\_char.
  + Nếu kiểm tra gia tri khac enter (“\n”) ta tiếp tục gán giá trị từ temp\_char vào mảng và tăng biến đếm RXI
  + Nếu ngược lại quét được giá trị enter ( “\n” ) -> gán cho phần tử cuối bằng NULL (đánh dấu phần tử kết thúc chuỗi ) và bật cờ đánh dấu trạng thái và reset bộ đếm