**Introduction**

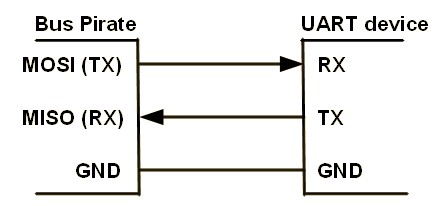
In the electronic world, a lot of thing that can be controlled by electronic and the purpose is to make human have convenient and better live. If the thing was controlled by electronic, that must have a sense and controller to monitor the situation. Furthermore, in between the controlled and sense, another important thing that is communication interface. The communication interface is a hardware device that usually to assist hardware to communicate another hardware. For example, if the file or photo in smart phone that need send to computer, the USB cable is one of way to assist smart phone send the data to computer. To fulfill the market demand, the communication interface was developed to produce different kind of capability protocol such as USB, UART, SPI and so no.

In this report, the UART communication interface is discussed and investigated to have a better understanding and learn how to apply UART device. The UART was investigated in STM32F429 board.

The UART is term for “Universal Asynchronous Receiver Transmitter”. The “Universal” indicates that data format and transmission speeds are configurable and “Asynchronous” indicates transmission of data without the use of an external clock signal. The UART performs serial-to-parallel conversion on data received from a peripheral device and parallel-to-serial conversion on data received from the CPU. The UART includes control capability and processor interrupt system that can be customized by programmer to service the high-priority instruction.

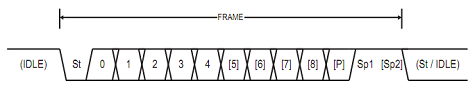
Besides that, the UART also includes a programmable baud generator that can be configure the speeds of transmission by use the equation that is given by data manual to calculate programmed of value.

**Theory**



Basically, the UAST is bidirectional communication that requires a minimum of two pin such as transmission line (TX) and reception line (RX) as show in diagram X. At the same diagram, the ground pin (GND) is also common pin for any UAST because this pin to ensure the ground level between master and slave is the same. Otherwise, when transmission is occurred, the transmit signal is received by slave that can not to read.

**UAST Data Formal Description**



The UAST transmits or received data in serial form. In diagram XX, that is show the common data frame formal. At begin, the first bit is called “start bit” (St) at low state to warn other peripheral device that should started to receive data.

From 0th bit to 8th bit in frame, these data was set by software. The word length may be selected as being either 8 or 9 bit by programming the master. After end of data (8th bit), the parity bit is appeared to let other peripheral receiver checks the received data that is correct or not.

At the end, the last bit is called “stop bit” (Sp) at high state to warn other peripheral device that should stop to receive data. The length of stop bit may be selected 1 or 2 bit by programmer.

**Transmission**

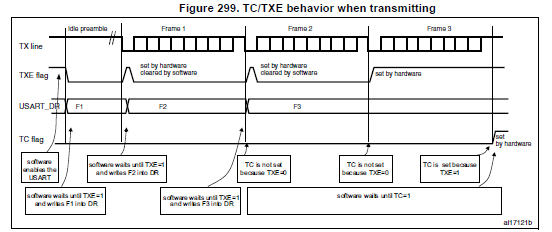
This instruction is referred by STM32F4XX data manual. Some of the specific term that is used in the STM32F4XX data manual.

The UART transmitter section includes a Transmit Data Register (TDR) and a Transmit Shift Register. When a transmission is taking place, a write instruction to the UAST Data Register (DR) stores the data in the TDR register and which is copied in the shift register at the end of the current transmission.

When no transmission is taking place, a write instruction to the UAST Data Register place the data directly in the shift register, the data transmission starts, and the TXE bit is immediately.

If a frame is transmitted (after the stop bit) and the TXE bit is set, the TC bit goes high. An interrupt is generated if the TCIE bit is set in the USART\_CR1 register.

After writing the last data into the USART\_DR register, it is mandatory to wait for TC=1 before disabling the USART or causing the microcontroller to enter the low-power mode.



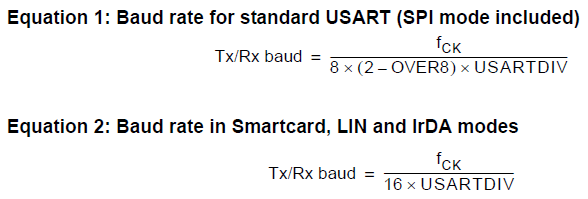
**Reception**

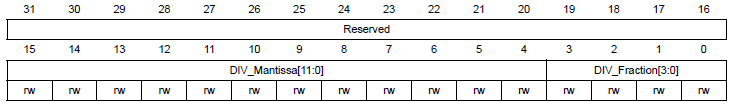
The UART receiver section includes a Receive Data Register (RDR) and a Receive Shift Register. When a character is received, the RXNE bit is set. It indicates that the content of the shift register is transferred to the RDR. In other words, data has been received and can be read.

The error flags can be set if a frame error, noise or an overrun error has been detected during reception.

**Result and Discussion**

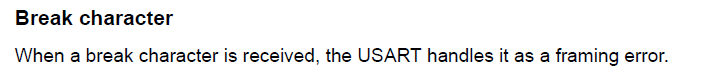
Baud Rate Algorithm



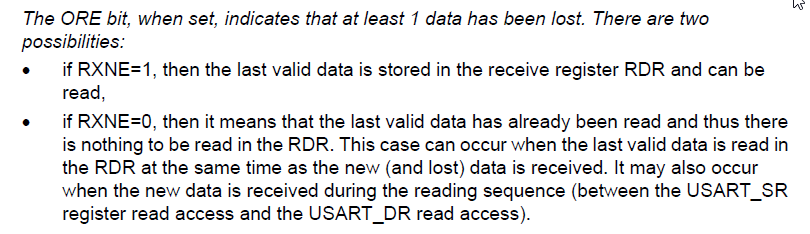


Word Length

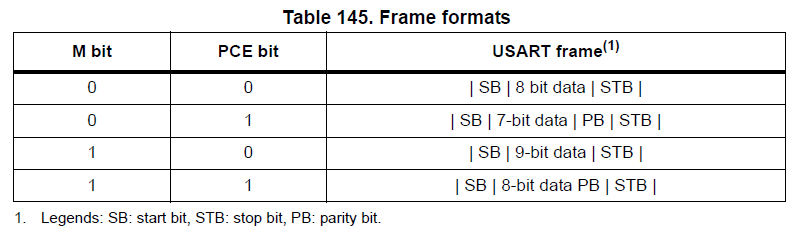
Framing Error



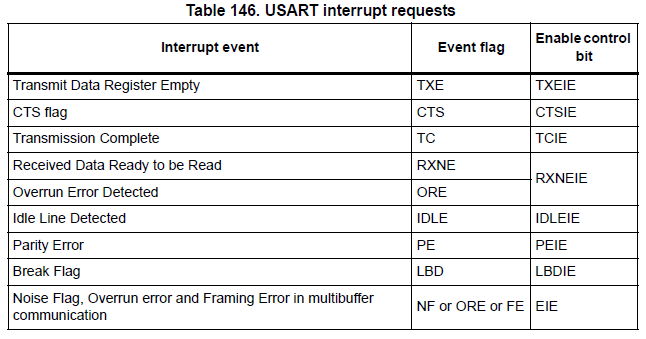
Overrun Error



Parity Control



Interrupt Event



Discussion