Introduction UART

Chapter 22
Communication Protocols

• Parallel Vs. Serial
  – Parallel: Faster / More expensive
  – Serial: Cheaper / slower
Communication Protocols

• Timing in serial communication
  – Synchronous Serial
  – Asynchronous Serial

• Rules of Asynch. Protocols
  – Synch bit
  – Parity bit
  – Baud rate
  – Data bit
Parity and Baud Rate

- **Parity**
  - Even Parity: \(10101010101\) → 6 one’s → EP=0
  - Off Parity: \(1010101010\) → 5 one’s → OP=0

- **Baud Rate**
  - Pulse per second
  - In digital world: Bits per second
    - 9600 baud → 9600 bps or 104 µs per bit.

- **Synch bit**
  - Required to define the beginning/end of the data
    - the **start bit** [LOW] and the **stop bit** [HIGH]

---

9600 8N1 – 9600 bps/8 bits per data / No parity / One Stop bit
Always has start bit
Serial Communication

• Universal Asynchronous Receiver and Transmitter (UART)
  – UART is programmable.

• Asynchronous
  – Sender provides no clock signal to receivers
  – FT232R converts the UART port to a standard USB interface
Data Frame

- Sender and receiver uses the same transmission speed
- Data frame
  - One start bit
  - Data (LSB first or MSB, and size of 7, 8, 9 bits)
  - Optional parity bit
  - One or two stop bit

\[
\text{Overhead } \% = 1 - \frac{\text{Useful Data}}{\text{Total Data}}
\]

\[
9600/(1 + 8 + 1 + 1) = \sim 862 \text{ bytes/second}
\]

In this case how many frames are required to send out 100,000 bits? How long does it take to transmit 100,000 bits at 57,600 baud rate?
What are we transmitting? What is the baud rate?

Start here!
Assume LSB first

1 stop bit, 8 data bits, no parity.
Transmitting 0x32 and 0x3C

1 start bit, 1 stop bit, 8 data bits, no parity, baud rate = 9600
Communication Modes

• A serial interface can have different communication modes
  – Full Duplex - means both devices can send and receive simultaneously.
  – Half-Duplex - means serial devices must take turns sending and receiving.
Differential Vs. Single Ended

Single-ended Signaling

D

0v

Noise

Noise sources: (1) induction picked up on the wired, (2) voltage difference between two grounds

Differential Signaling

D^+
D^-
Twisted-pair wires

Noise

Requires more wires but eliminates noise

D^+ - D^-
UART

- A universal asynchronous receiver/transmitter (UART) is a block of circuitry responsible for implementing serial communication.
- USART – Supports synchronous communication.
Multiple Bus Connections
### UART Protocols

#### Different Voltage Levels

<table>
<thead>
<tr>
<th>Standard</th>
<th>Voltage signal</th>
<th>Max distance</th>
<th>Max speed</th>
<th>Number of devices supported per port</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS-232</td>
<td>Single end (logic 1: +5 to +15V, logic 0: -5 to -15 V)</td>
<td>100 feet</td>
<td>115Kbit/s</td>
<td>1 master, 1 receiver</td>
</tr>
<tr>
<td>RS-422</td>
<td>Differential (-6V to +6V)</td>
<td>4000 feet</td>
<td>10Mbit/s</td>
<td>1 master, 10 receivers</td>
</tr>
<tr>
<td>RS-485</td>
<td>Differential (-7V to +12V)</td>
<td>4000 feet</td>
<td>10Mbit/s</td>
<td>32 masters, 32 receivers</td>
</tr>
</tbody>
</table>
Interfacing

• Note that in general we SHOULD use something like Maxim's MAX232 in order to ensure voltage compatibility between the PIC and the RS232 or the terminal.

• However, it turns out that by INVERTING polarity of the signals on TX and RX pins of USART, it is possible to interface to the terminal.
Interfacing to a PC

- Download a PC terminal software such as Hyper Terminal or RealTerm ([https://sourceforge.net/projects/realterm/](https://sourceforge.net/projects/realterm/)) If you only have a USB port you may need a USB/ Serial Cable and driver
- Set the Hyper Terminal to 9600, N,1,0
**STM32F401RET6 - 64 Pins - on the Nucleo-64**

**Table 9. Alternate function mapping**

<table>
<thead>
<tr>
<th>Port</th>
<th>AF00</th>
<th>AF01</th>
<th>AF02</th>
<th>AF03</th>
<th>AF04</th>
<th>AF05</th>
<th>AF06</th>
<th>AF07</th>
<th>AF08</th>
<th>AF09</th>
<th>AF10</th>
<th>AF11</th>
<th>AF12</th>
<th>AF13</th>
<th>AF14</th>
<th>AF15</th>
</tr>
</thead>
<tbody>
<tr>
<td>PA0</td>
<td>-</td>
<td>TIM2_CH1/ TIM2_ETR</td>
<td>TIM5_CH1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>USART2_CTS</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
<td></td>
</tr>
<tr>
<td>PA1</td>
<td>-</td>
<td>TIM2_CH2</td>
<td>TIM5_CH2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>USART2_RTS</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
<td></td>
</tr>
<tr>
<td>PA2</td>
<td>-</td>
<td>TIM2_CH3</td>
<td>TIM5_CH3</td>
<td>TIM9_CH1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>USART2_TX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
<td></td>
</tr>
<tr>
<td>PA3</td>
<td>-</td>
<td>TIM2_CH4</td>
<td>TIM5_CH4</td>
<td>TIM9_CH2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>USART2_RX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
<td></td>
</tr>
<tr>
<td>PA4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SPI1_NSS</td>
<td>SPI1_NSS</td>
<td>SPI1_NSS/ I2S3_WS</td>
<td>USART2_CK</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
<td></td>
</tr>
<tr>
<td>PA5</td>
<td>-</td>
<td>TIM2_CH1/ TIM2_ETR</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SPI1_SCK</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
<tr>
<td>PA6</td>
<td>-</td>
<td>TIM1_BKIN</td>
<td>TIM3_CH1</td>
<td>-</td>
<td>SPI1_MISO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
<tr>
<td>PA7</td>
<td>-</td>
<td>TIM1_CH1N</td>
<td>TIM3_CH2</td>
<td>-</td>
<td>SPI1_MOSI</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
<tr>
<td>PA8</td>
<td>MCO_1</td>
<td>TIM1_CH1</td>
<td>-</td>
<td>-</td>
<td>I2C3_SCL</td>
<td>-</td>
<td>-</td>
<td>USART1_CK</td>
<td>-</td>
<td>-</td>
<td>OTG_FS_SOF</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
<tr>
<td>PA9</td>
<td>-</td>
<td>TIM1_CH2</td>
<td>-</td>
<td>-</td>
<td>I2C3_SMBA</td>
<td>-</td>
<td>-</td>
<td>USART1_TX</td>
<td>-</td>
<td>-</td>
<td>OTG_FS_VBUS</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
<tr>
<td>PA10</td>
<td>-</td>
<td>TIM1_CH3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>USART1_RX</td>
<td>-</td>
<td>-</td>
<td>OTG_FS_ID</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
<tr>
<td>PA11</td>
<td>-</td>
<td>TIM1_CH4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>USART1_CTS</td>
<td>USART6_TX</td>
<td>-</td>
<td>OTG_FS_DM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
<tr>
<td>PA12</td>
<td>-</td>
<td>TIM1_ETR</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>USART1_RTS</td>
<td>USART6_RX</td>
<td>-</td>
<td>OTG_FS_DP</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
<tr>
<td>PA13</td>
<td>JTMS_SWDIO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SPI1_NSS</td>
<td>SPI1_NSS</td>
<td>SPI1_NSS/ I2S3_WS</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
<td></td>
</tr>
<tr>
<td>PA14</td>
<td>JTCK_SWDCLK</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
<tr>
<td>PA15</td>
<td>JTDI</td>
<td>TIM2_CH1/ TIM2_ETR</td>
<td>-</td>
<td>-</td>
<td>SPI1_NSS</td>
<td>SPI1_NSS</td>
<td>SPI1_NSS</td>
<td>SPI1_NSS</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>EVENT OUT</td>
</tr>
</tbody>
</table>
### Table 9. Alternate function mapping (continued)

<table>
<thead>
<tr>
<th>Port</th>
<th>AF00</th>
<th>AF01</th>
<th>AF02</th>
<th>AF03</th>
<th>AF04</th>
<th>AF05</th>
<th>AF06</th>
<th>AF07</th>
<th>AF08</th>
<th>AF09</th>
<th>AF10</th>
<th>AF11</th>
<th>AF12</th>
<th>AF13</th>
<th>AF14</th>
<th>AF15</th>
</tr>
</thead>
<tbody>
<tr>
<td>PB1</td>
<td>TIM1_CH2N</td>
<td>TIM3_CH3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB3</td>
<td>JTDOSWO</td>
<td>TIM2_CH2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SPI1_SCK</td>
<td>SPI3_SCK/I2S3_CK</td>
<td></td>
<td>I2C2_SDA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB4</td>
<td>JTRST</td>
<td>TIM3_CH1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SPI1_MISO</td>
<td>SPI3_MISO</td>
<td>I2S3ext_SD</td>
<td></td>
<td>I2C3_SDA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB5</td>
<td></td>
<td></td>
<td>TIM3_CH2</td>
<td></td>
<td></td>
<td></td>
<td>SPI1_SMBE</td>
<td>SPI3_MOSI/I2S3_SD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB6</td>
<td></td>
<td></td>
<td>TIM4_CH1</td>
<td>I2C1_SCL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB7</td>
<td></td>
<td></td>
<td>TIM4_CH2</td>
<td>I2C1_SDA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB8</td>
<td></td>
<td></td>
<td>TIM4_CH3</td>
<td>TIM10_CH1</td>
<td>I2C1_SCL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB9</td>
<td></td>
<td></td>
<td>TIM4_CH4</td>
<td>TIM11_CH1</td>
<td>I2C1_SDA</td>
<td>SPI2_NSS/I2S2_WS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB10</td>
<td></td>
<td></td>
<td>TIM2_CH3</td>
<td></td>
<td></td>
<td>I2C2_SCL</td>
<td>SPI2_SCK/I2S2_CK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB11</td>
<td></td>
<td></td>
<td>TIM1_BKIN</td>
<td></td>
<td></td>
<td>I2C2_SMBE</td>
<td>SPI2_SCK/I2S2_CK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB12</td>
<td></td>
<td></td>
<td>TIM1_CH1N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB13</td>
<td></td>
<td></td>
<td>TIM1_CH2N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB15</td>
<td>RTC_REFN</td>
<td>TIM1_CH3N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Setting the Baud Rate

• Suppose the processor clock $f_{PCLK}$ is 16MHz, and the system is oversampled by 16 ($OVER8 = 0$),

$$UARTDIV = \frac{f_{PCLK}}{8 \times (2 - OVER8) \times \text{Baud Rate}}$$

$$= \frac{16 \times 10^6}{8 \times (2 - 0) \times 9600} = 104.1667$$

• Thus $UARTDIV$ is 104.1875, which is encoded as 0x683.

• desired baud rate 9600

$$\text{Baud Rate} = \frac{16 \times 10^6}{8 \times (2 - 0) \times 104.1875} = 9598$$
Setting the Baud Rate

Thus $\text{USARTDIV}$ is 104.1875, which is encoded as 0x683.

$\text{OVER8} = 0$
# Register Mapping for UART

| Offset | Register  | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0x00   | USART_SR |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|        | Reset value |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x04   | USART_DR |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|        | Reset value |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x08   | USART_BRR |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|        | Reset value |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x0C   | USART_CR1 |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|        | Reset value |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x10   | USART_CR2 |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|        | Reset value |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x14   | USART_CR3 |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|        | Reset value |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x18   | USART_GTDR |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|        | Reset value |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |

Table 87. USART register map and reset values
UART Connection

To send and receive data using UART we can use several methods including: polling, interrupt
Transmitting and Receiving A Character (Polling) — Ref Manual section 19.3.2

1. Enable the USART by writing the UE bit in USART_CR1 register to 1.
2. Program the M bit in USART_CR1 to define the word length.
3. Program the number of stop bits in USART_CR2.
4. Select DMA enable (DMAT) in USART_CR3 if Multi buffer Communication is to take place. Configure the DMA register as explained in multibuffer communication.
5. Select the desired baud rate using the USART_BRR register.
6. Set the TE bit in USART_CR1 to send an idle frame as first transmission.
7. Write the data to send in the USART_DR register (this clears the TXE bit). Repeat this for each data to be transmitted in case of single buffer.
8. After writing the last data into the USART_DR register, wait until TC=1. This indicates that the transmission of the last frame is complete. This is required for instance when the USART is disabled or enters the Halt mode to avoid corrupting the last transmission.

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 (as well as its associated error flags).
- An interrupt is generated if the RXNEIE bit is set.
- The error flags can be set if a frame error, noise or an overrun error has been detected during reception.
- In multibuffer, RXNE is set after every byte received and is cleared by the DMA read to the Data Register.
- In single buffer mode, clearing the RXNE bit is performed by a software read to the USART_DR register. The RXNE flag can also be cleared by writing a zero to it. The RXNE bit must be cleared before the end of the reception of the next character to avoid an overrun error.
Transmitting and Receiving A Character (Interrupts)

- See section 19.4 USART interrupts
UART Interrupt: Receiving Data

Hardware sets TXE flag if TDR is copied to TX shift register.

Hardware sets RXNE flag if RX shift register is copied to RDR.

Set by hardware. Reading RDR automatically clears RXNE.

Receive Register Not Empty Signal (RXNE)

Processor Core

```c
void USART1_IRQHandler(void) {
    ...
    if(USART1->ISR & USART_ISR_RXNE) {
        buffer[counter] = USART1->RDR;
        counter++;
    }
    ...
}
```
Bluetooth

`Bluetooth to Serial Port Module`

- Vcc
- Reset
- Key
- TX
- RX
- GND
- LED

2.4GHz radio

STM32 Discovery Kit

UART TX
UART RX
GND

STM32 Discovery Kit
A little about Bluetooth

- Operates at the ISM frequency band
- Bluetooth is divided into 79 channels of different frequencies.
- A Bluetooth device, hops frequency at a rate of 1600 hops per second, randomly selecting a channel of 1 MHz to operate.
A little about Bluetooth

- Operates at the ISM frequency band
- Bluetooth is divided into 79 channels of different frequencies.
- A Bluetooth device, hops frequency at a rate of 1600 hops per second, randomly selecting a channel of 1 MHz to operate.

https://sites.google.com/site/nearcommunications/adaptative-frequency-hopping
Bluetooth PICONET

- Each Bluetooth device has a 48-bit address: 0x 0018 E4 0C68 0A