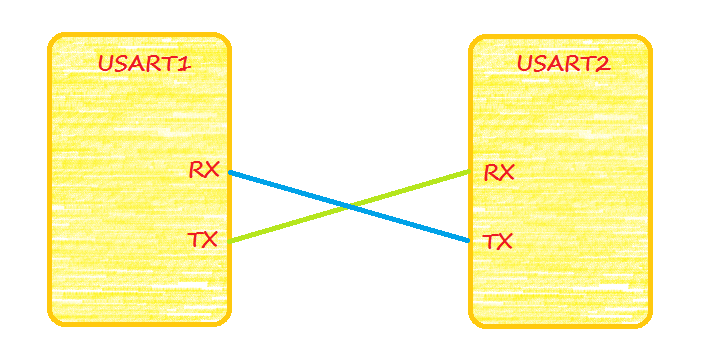
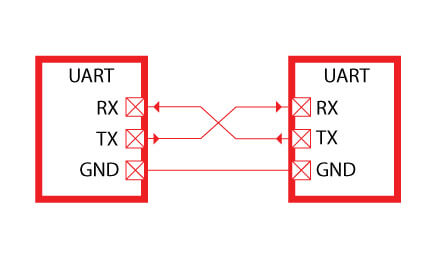
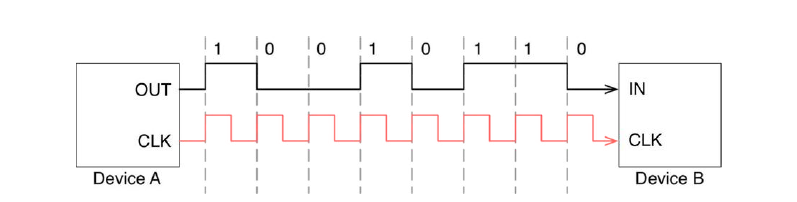
**UART&USART**

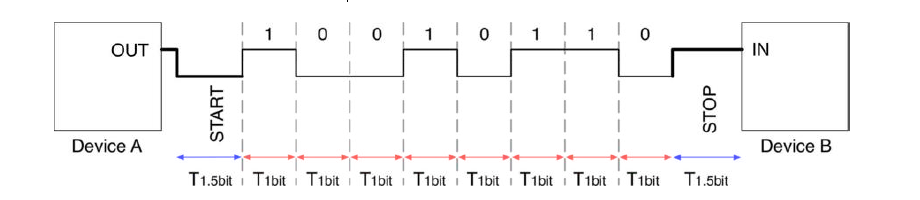
Günümüzde elektronik endüstrisinde çok sayıda seri iletişim protokolü ve donanım arabirimi bulunmaktadır. **UART (Universal Asynchronous Receiver Transmitter)**, bilgisayar veya mikrodenetleyicilerin çevre birimleri ile arasında haberleşmeyi sağlayan haberleşme protokolüdür. Asenkron olarak çalıştığı için herhangi bir “clock” ihtiyacı duymaz. **USART (Universal Synchronous Asynchronous Receiver Transmitter)** ise hem senkron hem de asenkron olarak çalışabilir. UART’a göre daha gelişmiş bir protokoldür. Haberleşme mantıklı aynı şekilde çalışır ancak USART aynı zamanda senkron haberleşmeleri de gerçekleştirebilir. Yeni çıkan bir mikroişlemcinin datasheet’ine baktığınız zaman bu birimleri genelde USART birimi olarak görüyoruz çünkü USART aynı zamanda UART’ı da kapsayan bir birim olarak tasarlanmıştır. Hemen hemen tüm STM32 MCU'ları en az iki UART / USART arabirimi sağlar.

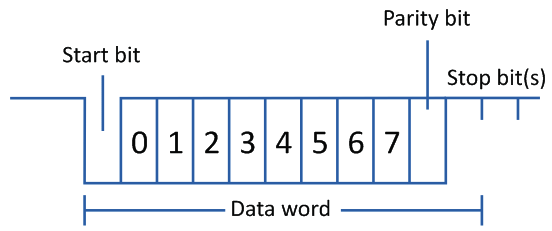


 UART haberleşmesini gerçekleştirirken ilk olarak **baudrate** (**veri taşıma hızı**) ayarlanması gerekir. Veri taşıma hızı çeşitli aralıklarda olabilir ancak piyasada yaygın olarak kullanılan baudrate’ler **4800, 9600, 57600, 115200** ve mikroişlemciler için çok fazla tercih edilmese de **921600**. Verinin saniyede ne kadarlık byte’ını taşıyacağını belirlememize yarar. Örneğin veri taşıma hızımızı 115200 seçersek bu bizim için saniyede yaklaşık olarak 115200 byte veri iletimi sağlayacaktır.

Bu haberleşme tipini kullanabilmemiz için alıcı ve vericinin veri taşıma hızlarının (baudrate) aynı olması gerekiyor. Bunun sebebi ise aktarım sırasında oluşabilecek hataları minimuma indirmek. Ayrıca haberleşme sırasında doğacak hataları en aza indirmek için start bit, stop bit ve isteğe bağlı olarak parity bit kullanılmaktadır.

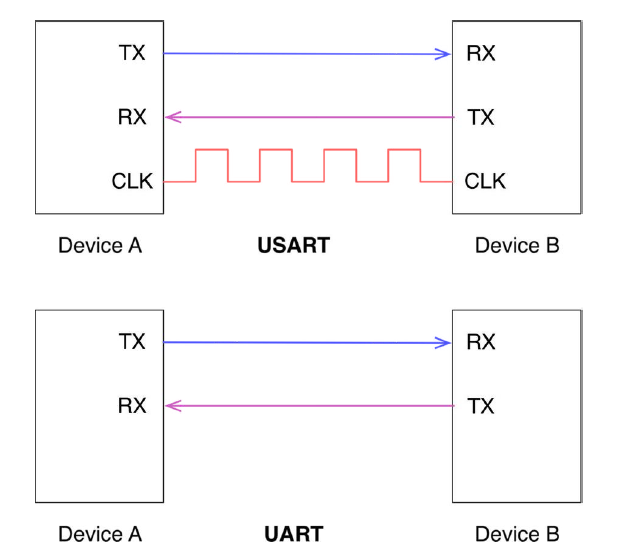
**Synchronous Transmission**



 **Asynchronous Transmission**

0 1 0 1 0 1 1 0 0

0 0 1 1 1 0 0 0 1



**UART Veri Gönderme:**

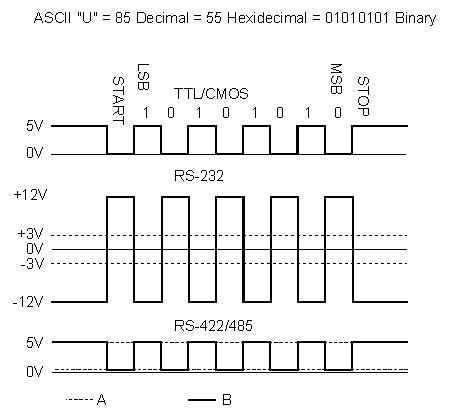
* Verilerimiz en düşük değerlikli bitten en yüksek değerlikli bite doğru gönderilir.
* Gönderici(transmit) veri göndermediği zaman lojik 1 seviyesindedir.

Veri göndereceğinde lojik 0 seviyesine kendini çeker.

* Start bitinden sonra veriler bit-bit gönderilir, en sonda ise stop biti ile veri gönderme sonlanır.
* Stop bitten sonra kendisini lojik 1 seviyesine çeker.

**UART Veri Alma:**

* Alıcı(Receive) lojik seviyesini 1’den 0’a çektiğinde veri alımı başlar.
* Gelen bitleri baundrate süresinde alır, buffer adlı değişkenine yazar.
* Stop biti ile buffera datalar yazılmış olur, ilgili flag set edilir.
* Flag ile buffer’ın doluluk oranı tespit edilir, yeni veri yolu başlatılmış olur.

Bir UART / USART, sinyalleme yöntemini tanımlar, ancak voltaj seviyeleri hakkında hiçbir şey söylemez. Bu, bir STM32 UART / USART'ın neredeyse VDD'ye eşit olan MCU I / O'ların voltaj seviyelerini kullanacağı anlamına gelir (bu voltaj seviyelerine TTL voltaj seviyeleri olarak da adlandırılması yaygındır). Bu voltaj seviyelerinin, kartın dışındaki seri iletişime izin vermek için çevrilme şekli, diğer iletişim standartlarına talep edilir. Örneğin, EIA-RS232 veya EIA-RS485, zamanlamalarına ve anlamlarına ve konektörlerin fiziksel boyutuna ve pin çıkışına ek olarak sinyal voltajlarını tanımlayan gerçekten popüler iki standarttır.

RS232 ve RS485 gibi bazı iletişim standartları, özel bir Donanım Akış Kontrol hattı kullanma imkanı sağlar. Örneğin, RS232 arabirimini kullanarak iletişim kuran iki cihaz Gönderme İsteği (RTS-Request To Send) ve Gönderme İşlemini Temizle (CTS-Clear To Send) adlı iki ek satırı paylaşabilir: gönderen, alıcıyı veri giriş hattını izlemeye başlaması için sinyal gönderen RTS'sini ayarlar. Veri için hazır olduğunda, alıcı, göndereni veri göndermeye başladığını ve gönderenin köle veri çıkış hattını izlemeye başlamasını sağlayan tamamlayıcı satırı olan CTS'yi yükseltir.



**UART&USART Terimler:**

**• BaudRate:** Verinin saniyede ne kadarlık byte’ının taşınacağını ifade eder.

**• WordLength:**İletilen veya alınan veri bitlerinin sayısını belirtir. Bu alan UART\_WORDLENGTH\_8B veya UART\_WORDLENGTH\_9B değerini alabilir, yani 8 veya 9 veri biti içeren bir UART paketi üzerinden iletim yapabiliriz. Bu sayı, başlatma ve durdurma bitleri gibi iletilen bitleri içermez.

**• StopBits:** Bu alan iletilen stop bitlerinin sayısını belirtir. UART\_STOPBITS\_1 veya UART\_STOPBITS\_2 değerini varsayabilir; bu, çerçevenin sonunu belirtmek için bir veya iki durdurma biti kullanabileceğimiz anlamına gelir.

**• Parity:** Parity, hata denetiminin çok basit bir şeklidir. İki çeşittir: tek veya çift. Eşlik bitini üretmek için tüm veri bitleri toplanır ve toplamın düzgünlüğü bitin ayarlanıp ayarlanmadığına karar verir. Parity isteğe bağlıdır ve çok yaygın olarak kullanılmaz. Gürültülü ortamlar arasında iletim için yararlı olabilir, ancak aynı zamanda veri aktarımını biraz yavaşlatır ve hata işlemeyi uygulamak için hem gönderen hem de alıcı gerektirir.

-UART\_PARITY\_NONE

-UART\_PARITY\_EVEN

-UART\_PARITY\_ODD

**• Mode:** RX veya TX modunun etkin veya devre dışı olduğunu belirtir.

-UART\_MODE\_RX

-UART\_MODE\_TX

-UART\_MODE\_TX\_RX

• **HwFlowCtl:** RS232 Donanım Akış Kontrol modunun etkin veya devre dışı olup olmadığını belirtir.

-UART\_HWCONTROL\_NONE

-UART\_HWCONTROL\_RTS-Request To Send(RTS)

-UART\_HWCONTROL\_CTS -Clear To Send(CTS)

-UART\_HWCONTROL\_RTS\_CTS

**• OverSampling:** Yüksek frekansa sahip bir sinyali örnekleme tekniğidir. Aşırı örnekleme alanı, -UART\_OVERSAMPLING\_16 değerinin her kare biti için 16 örnek gerçekleştirdiğini veya -UART\_OVERSAMPLING\_8 değerinin 8 örnek gerçekleştireceğini varsayabilir.