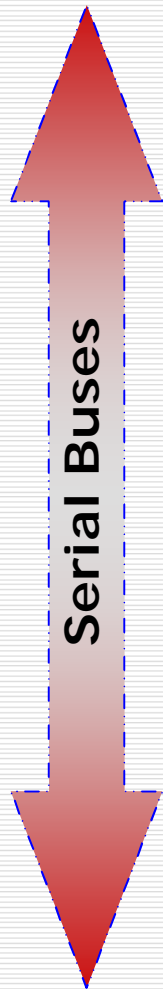


Komunikacja szeregową – UART:

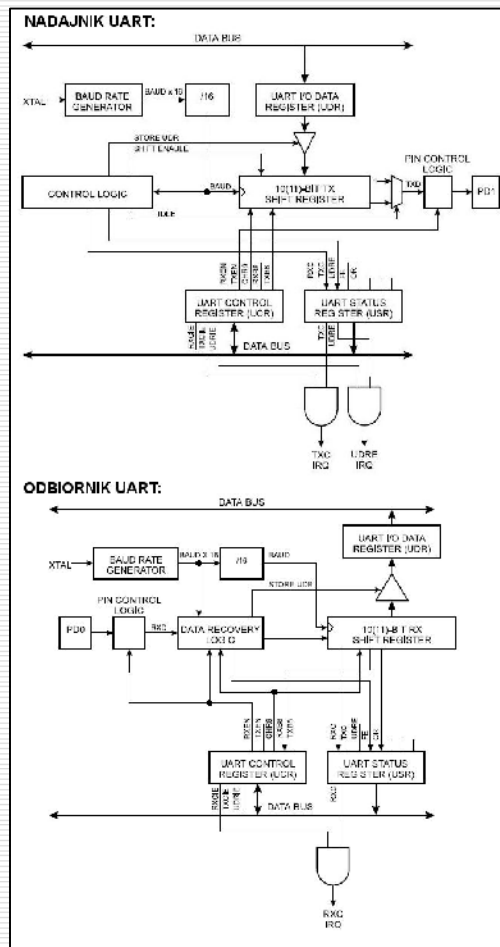


- UART – Universal Asynchronous Receiver Transmitter:
 - n Standard rozwinięty już w latach 60',
 - n Prosty, uniwersalny, dobrze udokumentowany,
 - n Wolna komunikacja: max. 1Mbit/s,
 - n Po jednym przewodzie komunikacyjnym w każdym z kierunków plus wspólna masa sygnałowa,
 - n Asynchroniczna – odbiornik i nadajnik muszą wcześniej znać szybkość transmisji,
 - n Bity START i STOP normują przesyłanie,
 - n Możliwe dołączenie informacji o parzystości (kontrola błędów).

Informacje: kom. szeregową

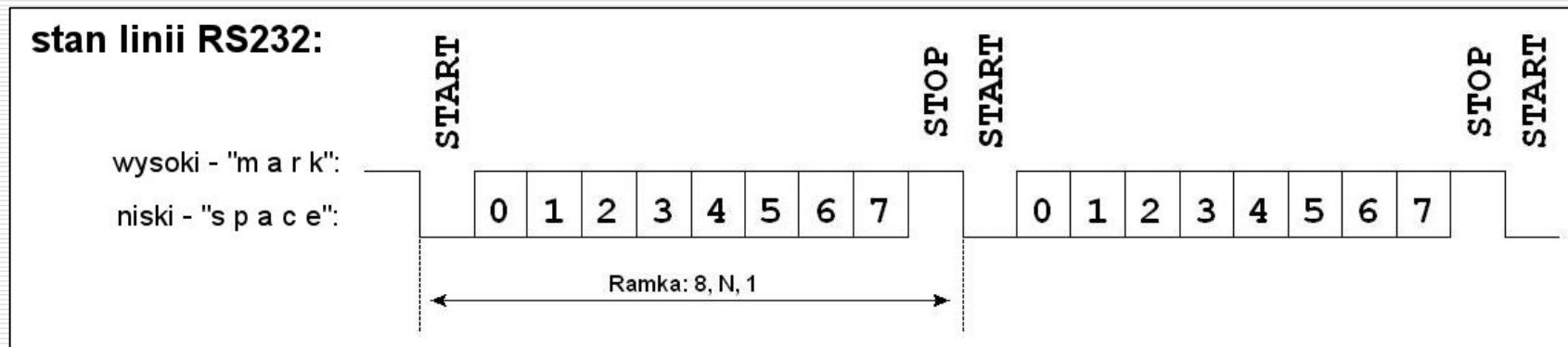
- Dwa podstawowe typy komunikacji szeregową:
 - n synchroniczna – równolegle z ciągiem bitów przesyłany jest sygnał synchronizujący, określający kiedy dane są ważne (valid),
 - n asynchroniczny – dane nie są związane z żadnym sygnałem synchronizującym – parametry transmisji trzeba ustawić wcześniej – ręcznie, a do danych trzeba dodać pewne informacje mogące zsynchronizować dane przesyłane i odbierane (np.: ramka 8, N, 1; 8, E, 2),
 - Nazewnictwo:
 - n Port dwukierunkowy (dwukierunkowy) – może równocześnie nadawać i odbierać dane (analogia: telefon),
 - n Port half-dwukierunkowy (jednokierunkowy) – jednoczesny odbiór i nadawanie nie jest możliwe (analogia: CB radio).
 - W AVR'ach mamy w pełni dwukierunkowy (dwukierunkowy, osobne rejestry nadawania i odbioru) USART – czyli transmisja asynchroniczna lub synchroniczna
-

Informacje: układ USART



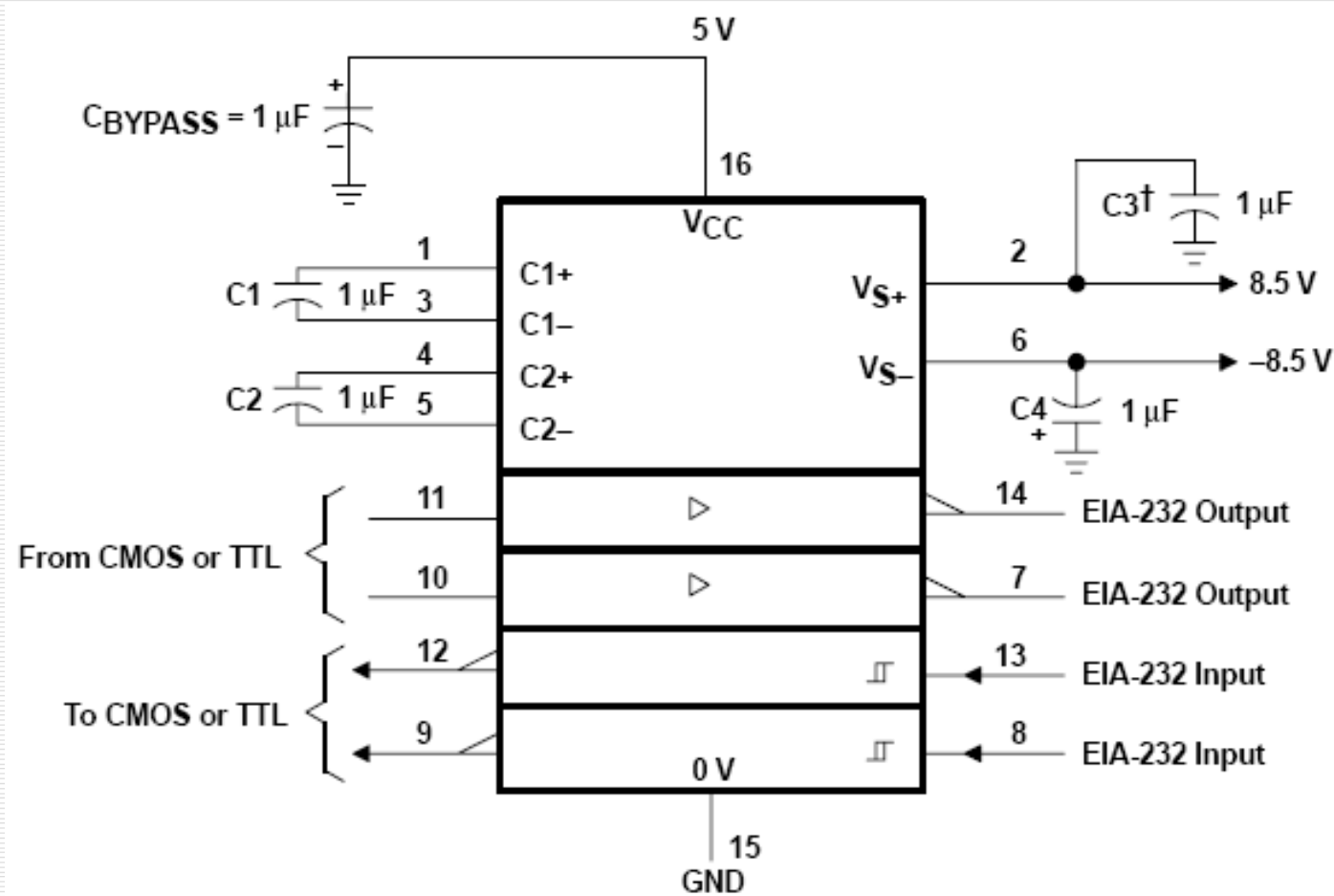
- **USART** – skrót od *Universal Synchronous Asynchronous Receiver/Transmitter* – jest to sprzętowy kontroler transmisji szeregowej, który zamienia dane równoległe (wpisywane do rejestru) na postać szeregową, która wysyłana jest do odbiornika, np. poprzez RS232 (po zmianie poziomów logicznych).
- USART wykonuje wszelkie zadania: synchronizację, sprawdzanie parzystości i inne niezbędne podczas transmisji.

Informacje: RS232 – specyfikacja



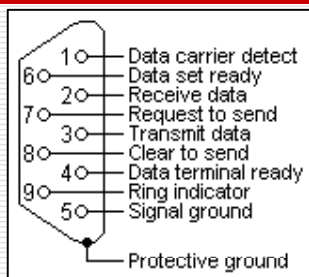
- Brak sygnału synchronizującego upraszcza znacznie transmisję, ale konieczne należy pamiętać, że:
 - n bity wysyłane są z daną częstotliwością (baud rate) i ta musi być jednakowa dla odbiornika i nadajnika,
 - n odbiornik może zacząć odbiór w nieodpowiednim momencie, (rozsynchronizowanie) - na ponowną synchronizację potrzeba czasu,
 - n potrzebne są dodatkowe bity aby właściwie zaznaczyć początek i koniec właściwych danych.
- W tym celu protokół określa następujące bity dodawane do transmitowanych danych: **bit startu, bit parzystości, bit stopu**

Układ aplikacyjny układu MAX232



źródło: TI MAX232 Datasheet

Informacje: RS232 - fizycznie

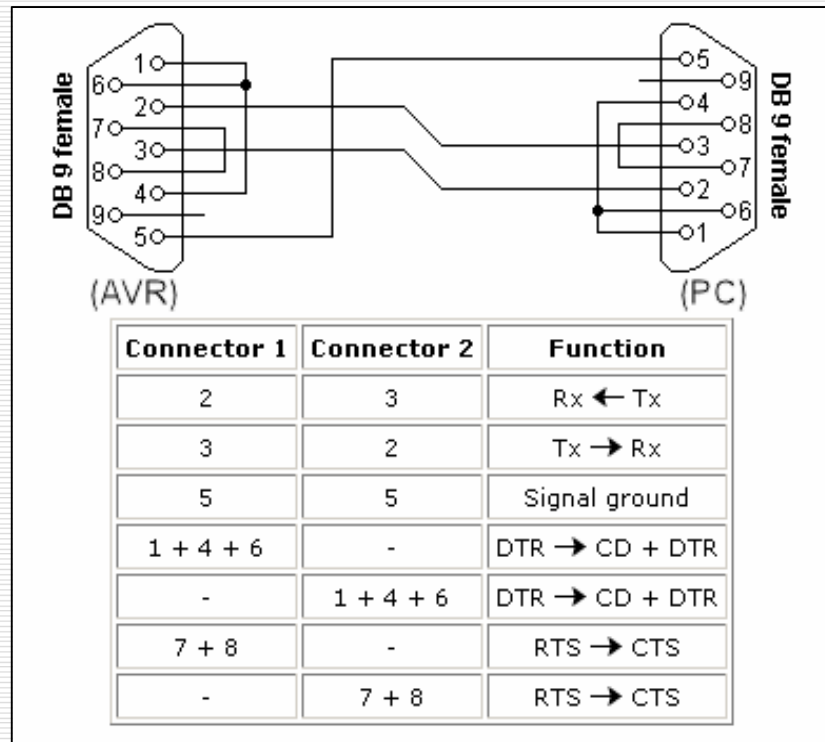


○ Połączenie z PC:

n fizyczny rozkład wyprowadzeń gniazda DB9,

oznacz.:	nazwa	opis	wypr.:
TXD	transmitted data	dane nadawane z DTE do DCE, przy braku transmisji stan MARK	3(M), 2(F)
RXD	received data	dane odbierane przez DTE z DCE, przy braku transmisji stan MARK	2(M), 3(F)
RTS	request to send	żądanie nadawania, sygnalizuje gotowość DTE	7
CTS	clear to send	gotowość do odbioru przez DCE	8
DSR	DCE ready	gotowość DCE do prowadzenia transmisji	6
DTR	DTE ready	gotowość DTE do prowadzenia transmisji	4
DCD	carrier detect	wykrycie nośnej, wykorzystywane przy modemach	1
SG	signal ground	masa sygnałowa, poziom odniesienia dla wszystkich sygnałów	5
RI	ring indicator	sygnał dzwonienia, gdy DTE jest modemem	9
GND	signal ground	masa sygnałowa dla wszystkich linii	5

Informacje: RS232 - fizycznie



- Połączenie z PC:
połączenie null-modem
 - n najłatwiejszy sposób na połączenie dwu urządzeń przez RS232 – do transmisji wykorzystywane są jedynie linie Tx (3), Rx (2) oraz GND - signal ground (5),
 - n zwarte sygnały kontroli transmisji – zakładamy, że zarówno odbiornik, jak i nadajnik są zawsze gotowe do wymiany danych.

Informacje: RS232 - fizycznie

Level	Transmitter capable (V)	Receiver capable (V)
Logical 0 space state	+5 ... +15	+3 ... +25
Logical 1 mark state	-5 ... -15	-3 ... -25
Undefined	-	-3 ... +3

Baud rate	Maximum cable length (ft)
19200	50
9600	500
4800	1000
2400	3000

- Napięcia na liniach:

- n Stan wysoki (mark state) - napięcie ujemne,
- n Stan niski (space state) - napięcie dodatnie

Napięcia te mają wpływ na maksymalną długość kabla, szybkość transmisji a przede wszystkim na mogące wystąpić błędy (np. przy zakłóceniach występujących w przemyśle lepszym rozwiązaniem jest RS485).

- Długość linii RS232:

- n Standard określa jednoznacznie: 50m.
- n Zmniejszając baud rate można jednak kabel wydłużyć – wg tabeli.

Kod przykładowy: Inicjalizacja

```
// !-- 1 --!
#define FOSC 14745600 // Clock Speed
#define BAUD 9600
#define MYUBRR FOSC/16/BAUD-1

void main( void ) {
    ...
    USART_Init ( MYUBRR );
    ...
}

void USART_Init( unsigned int ubrr ) {

    // !-- 2 --!
    UBRRH = (unsigned char)(ubrr>>8);
    UBRL = (unsigned char)ubrr;

    // !-- 3 --!
    UCSRB = (1 << RXEN) | (1 << TXEN);

    // !-- 4 --!
    UCSRC = (1<<URSEL) | (3<<UCSZ0);
}
```

Komentarz do kodu:

1. Definiując odpowiednie FOSC (podane w Hz) oraz BAUD unikamy szukania odpowiedniej wartości w tabelach na str. 159,

Inicjalizacja USART:

2. ustawiamy odpowiednią prędkość transmisji (przekazaną w definicji MYUBRR przy wywołaniu funkcji inicjalizacji),
3. włączamy odbiornik oraz nadajnik USARTu,
4. konfigurujemy parametry ramki interfejsu: 8 bitów danych, jeden bit stopu brak parzystości
UWAGA na bit URSEL!!!

źródło: ATmega8(L) Datasheet

Kod przykładowy: Transmisja

```
void USART_Transmit
    ( unsigned char data )
{
    // !-- 1 --!
    while ( !( UCSRA&(1<<UDRE)) )
        ;

    // !-- 2 --!
    UDR = data;
}
```

Funkcja wysyłająca pojedynczy bajt:

1. Pooling bitu UDRE z rejestru UCSRA – oczekiwanie na zwolnienie bufora danych mikrokontrolera (zabezpieczenie przed zamazaniem nie wysłanych jeszcze danych),
2. Zapisanie danych do rejestru UDR – wysyłka następuje automatycznie.

Kod przykładowy: Odbiór

```
unsigned char USART_Receive( void )
{
    // !-- 1 --!
    while ( !(UCSRA & (1<<RXC)) )
        ;

    // !-- 2 --!
    return UDR;
}
```

Funkcja odbierająca pojedynczy bajt:

1. Pooling bitu RXC z rejestru UCSRA – oczekiwanie na zakończenie odbioru danych przez blok USART
UWAGA: gdy dane nie zostaną nadane mikrokontroler zatrzymuje się w tym punkcie.
2. Odczyt odebranych danych z rejestru UDR oraz zwrócenie ich jako wartość funkcji.

Więcej o UART / Serial Port

- Wikipedia EN:
 - n <http://en.wikipedia.org/wiki/RS-232>
 - Beyond Logic, Interfacing The RS232 / Serial Port:
 - n <http://www.beyondlogic.org/serial/serial.htm>
 - Introduction to Serial Communications:
 - n <http://www.taltech.com/resources/intro-sc.html>
 - RS232 Tutorial:
 - n http://www.camiresearch.com/Data_Com_Basics/RS232_standard.html
-