USART

Универсальный синхронно-асинхронный последовательный приемопередатчик (USART) – гибкое средство связи.

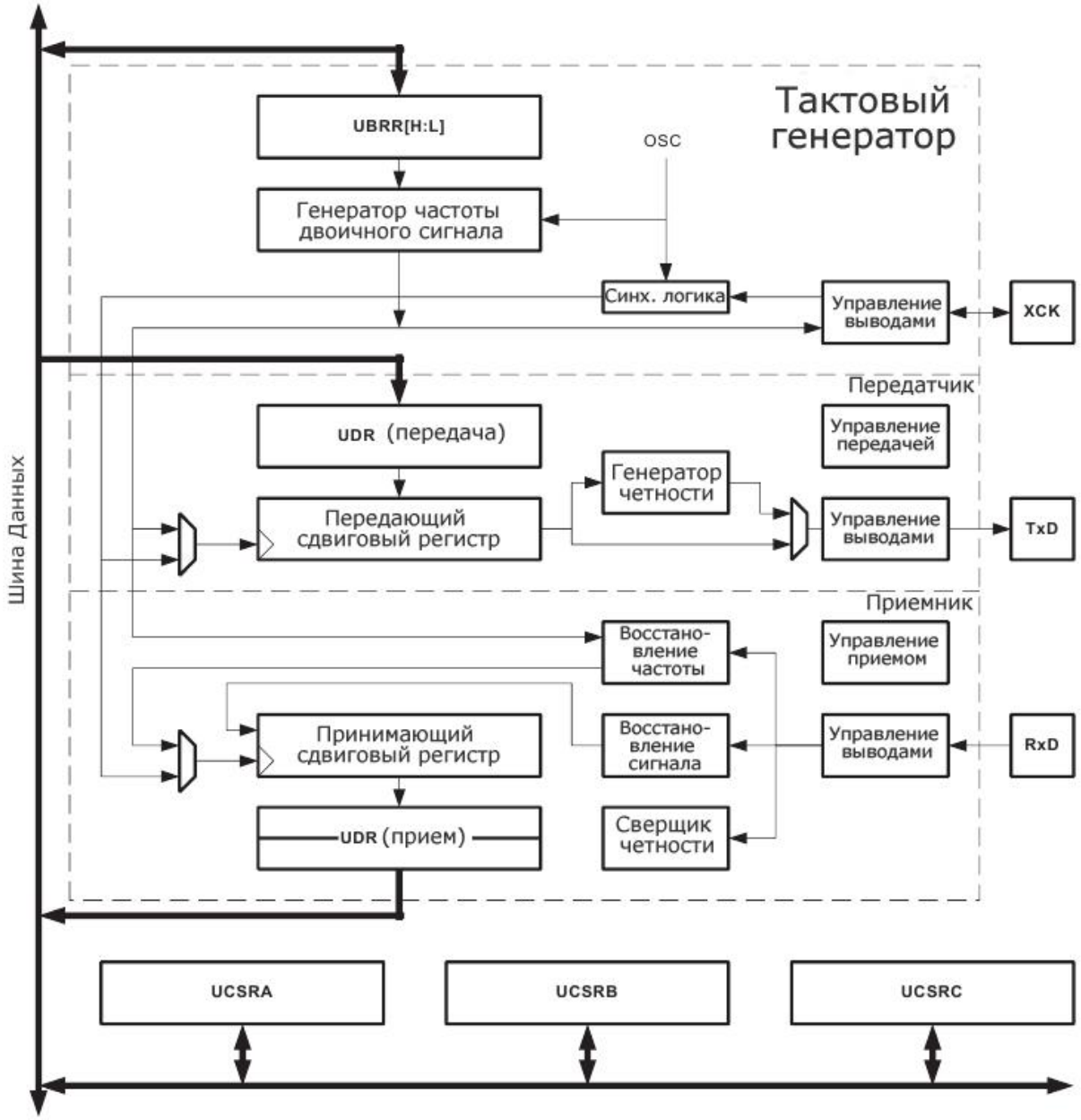
Основные плюшки:

- Дуплекс (независимые регистры приема и передачи);
- Синхронная и асинхронная работа;
- Синхронная передача с тактированием ведомого или ведущего;
- Высокое разрешение генератора сигнала;
- Поддержка бит данных от 5 до 9, 1 или 2 стоп бит;
- Аппаратная проверка четности;
- Обнаружение переполнения данных;
- Обнаружение ошибок структуры данных;
- Фильтр помех, обнаружение ложного старт-бита, цифровой ФНЧ;
- Три раздельных прерывания;
- Режим мультипроцессорной связи;
- Режим асинхронной передачи с двойной скоростью.

Обзор

Структурная схема USART представлена на рис. 69. Доступные регистры и порты ввода-вывода отмечены жирным начертанием.

рис. 69.



Пунктиром выделены три основных блока: тактовый генератор, передатчик и приемник. Управляющие регистры доступны всем блокам. Тактовый генератор состоит из

синхронизирующей схемы для внешнего источника тактирования (при ведомом режиме синхронной передачи) и непосредственно генератора. Вывод XCK используется только в режиме синхронной работы. Передатчик состоит из буфера, последовательного сдвигового регистра, генератора четности, и управляющей логики, определяющей режимы работы. Буфер поддерживает длительную непрерывную передачу без пауз между данными. Приемник – наиболее нашпигованный блок USARTa из за своих схем восстановления данных и коррекции частоты. Схемы восстановления используются для приема в асинхронном режиме. Также приемник включает сверщик четности (?) управляющую логику, сдвиговый регистр и двухуровневый регистр приема данных UDR. Приемник поддерживает те же форматы данных, что и передатчик, и может определять ошибки формата, переполнения и четности.

USART против UART – совместимость.

USART полностью совместим с UART касательно:

- Расположения бит во всех регистрах;
- Частоты двоичного сигнала;
- Работы передатчика;
- Работы буфера передатчика;
- Работы приемника.

Тем не менее, приемная буферизация получила два усовершенствования, что сказывается на совместимости в определенных случаях:

- Добавлен второй регистр. Два регистра работают как FIFO буфер (first in – first out, первый вошел – первый ушел). Поэтому регистр UDR должен читаться единожды для каждого принятого фрагмента данных. Флаги ошибок (FE и DOR), а также девятый бит данных (RXB8) также сохраняются в приемный регистр. Поэтому биты статуса должны читаться до чтения регистра UDR.
- Сдвиговый регистр приемника может работать как третий буфер. Это достигается сохранением данных в регистре

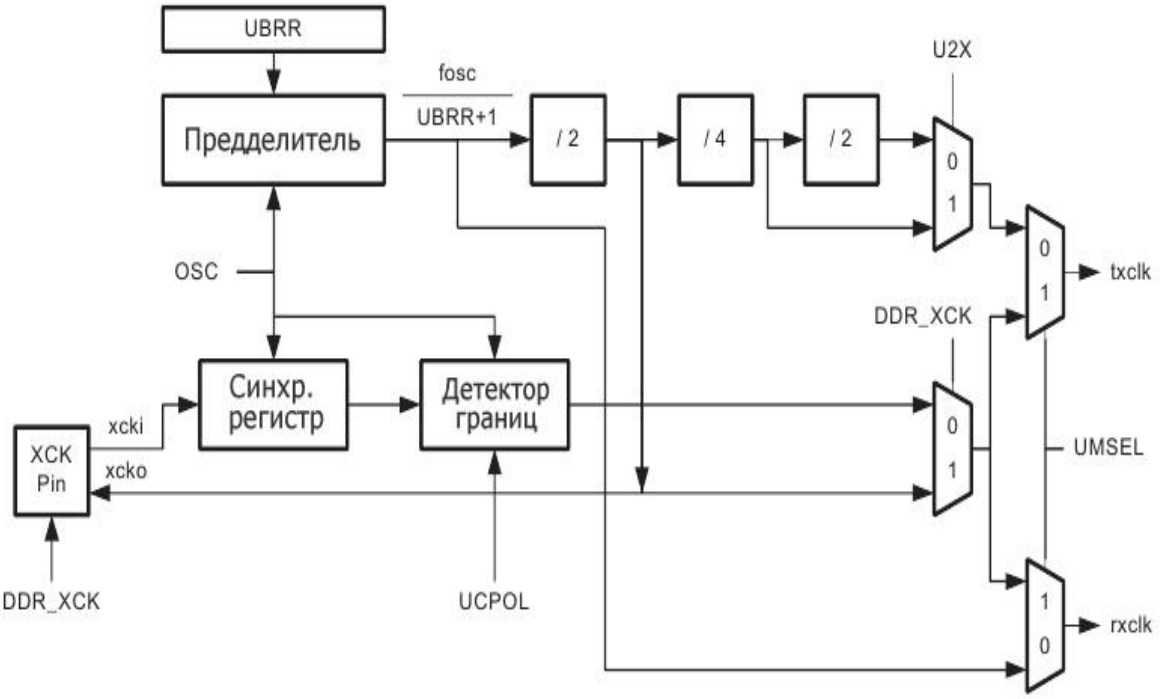
до обнаружение следующего старт-бита. Это делает USART более устойчивым к ошибкам переполнения (DOR);
- Следующие регистры были переименованы: CHR в UCRZ2 и OR в DOR.

Генерация частоты.

Тактовый генерирует частоту как для приемника, так и для передатчика. USART поддерживает четыре режима: нормальный асинхронный, асинхронный двойной скорости, синхронный ведущий и синхронный ведомый. Бит UMSEL в регистре UCSRC (USART Control and Status Register C – регистр управления и статуса) определяет синхронный или асинхронный режим. Двойная скорость определяется в регистре UCSRA битом U2X. При синхронном режиме (UMSEL = 1) регистр определения входа-выхода Data Direction Register определяет внешнюю или внутреннюю синхронизацию состоянием регистра для вывода XCK (DDR_XCK). Вывод XCK задействован только в синхронном режиме.

На рисунке 70 показана структурная схема логики генератора частоты.

Рис. 70.



Сигналы:

txclk – Синхронизация передатчика (внутренний);

rxclk – Синхронизация приемника (внутренний).

xcki – Вход с вывода XCK. Используется для синхронного ведомого режима.

fosc – системная тактовая частота.

Внутренняя генерация частоты.

Внутренняя генерация частоты используется для асинхронного и ведущего синхронного режимом. Предделитель генератора работает от тактовой частоты. Регистр скорости сигнала UBRR (USART Baud Rate Register) определяет частоту работы внутреннего генератора. Предделитель представляет собой счетчик, периодически считающий до нуля. Импульс генерируется каждый раз при достижении нуля. Передатчик делит частоту внутреннего генератора на 2, 8 или 16, в зависимости от режима, а приемник получает эту частоту

напрямую и использует для тактирования блоков восстановления. Формулы расчета частоты приведены в табл. 60.

Табл. 60.

Режим	Формула для частоты	Формула значения регистра UBRR
Нормальный асинхронный (U2X = 0)	$BAUD = \frac{f_{osc}}{16(UBBR+1)}$	$UBBR = \frac{f_{osc}}{16BAUD} - 1$
Асинхронный с двойной скоростью (U2X = 1)	$BAUD = \frac{f_{osc}}{8(UBBR+1)}$	$BAUD = \frac{f_{osc}}{8(UBBR+1)}$
Синхронный ведущий	$BAUD = \frac{f_{osc}}{2(UBBR+1)}$	$BAUD = \frac{f_{osc}}{2(UBBR+1)}$

Примечание:

BAUD – скорость сигнала, бит/сек;

f_{osc} – системная тактовая частота, Гц;

UBBR содержит старший и младший регистры UBRRH и UBRRL (0 -4095)

Режим двойной скорости.

Скорость передатчика может быть удвоена установкой бита U2X в регистре UCSRA. Установка этого бита возымеет результат только в режиме асинхронной передачи.

Установка этого бита сокращает предделитель вдвое, эффективно удваивая скорость, но понижая безошибочность

приемника. На передатчик негативных воздействий не оказывается.

Внешняя синхронизация.

Внешняя синхронизация используется в синхронном ведомом режиме. Обработка сигнала с вывода XCK регистром синхронизации и детектором границ занимает два такта ЦПУ, поэтому частота внешней синхронизации должна быть по крайней мере вчетверо меньше тактовой частоты:

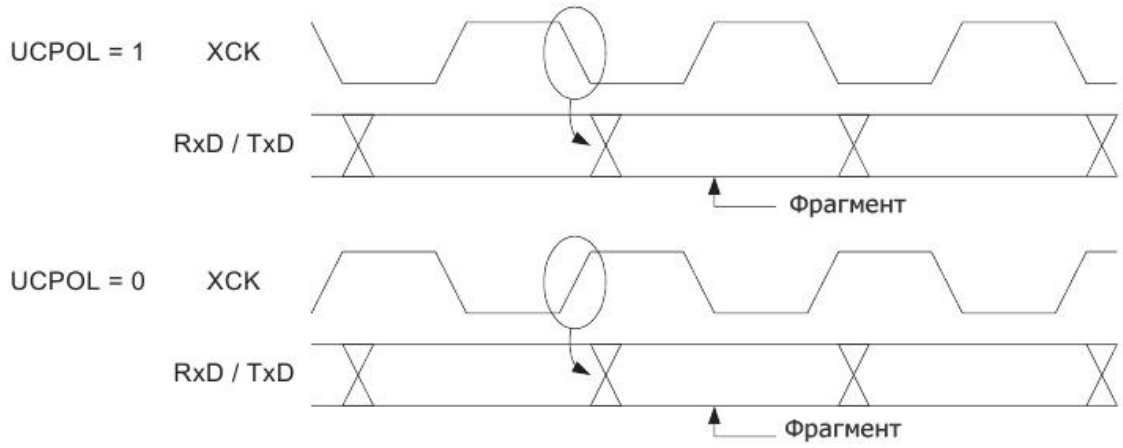
$$f_{XCK} < \frac{f_{osc}}{4}$$

Стабильность тактовой частоты зависит от источника тактирования. Рекомендуется увеличить отступ частот, чтобы избежать потери данных из-за отклонений частоты.

Синхронный режим работы.

В синхронном режиме вывод XCK работает как вход (ведомый) или выход (ведущий). Биты данных разделяются по границам синхроимпульсов, как по фронтам, так и по спадам. Простое правило: прием данных дискретизируется по противоположной границе импульса вывода XCK от передачи.

Рис. 71.



Бит UCPOL регистра UCRSC определяет, какая из границ импульса на выводе XCK используется для фрагментирования, и какая для получения данных. На рис. 71 изображено как

получение данных происходит при UCPOL = 1 по фронту и при UCPOL = 0 по спаду. А фрагментирование напротив при UCPOL = 0 по фронту и при UCPOL = 1 по спаду.

Фреймы данных.

Фрейм последовательных данных состоит из бит данных, бит синхронизации и опционально из битов четности. USART поддерживает 30 различных вариантов фреймов следующей структуры:

- 1 старт бит;
- от 5 до 9 бит данных;
- ничего, или четный/нечетный бит четности;
- 1 или 2 стоп-бита.

Фрейм начинается со старт-бита. Следующими приходят биты данных, до 9. В конце биты четности, если включены и стоп-биты. За одним фреймом может тут-же последовать другой или же линия может быть переведена в режим ожидания (высокий уровень). Возможный формат фрейма показан на рис. 72.

Рис. 72.

St – старт-бит, всегда 0;
(n) – биты данных;
P – биты четности, могут быть четными и нечетными;
Sp – стоп бит(ы), всегда 1.

Формат данных задается битами UCSZ2:0, UPM1:0, USPS в регистрах UCSRB и UCSRC. Формат задается одновременно для передатчика и приемника. Биты UCSZ2:0 (USART Character Size) задают количество бит данных; UPM1:0 (USART Parity Mode) включают биты четности и определяют их четность; USPS (USART Stop Bit Selection) определяет, один или два бита

четности будет завершать фрейм. Приемник всегда игнорирует второй стоп-бит. Наличие низкого уровня на стоп-бите вызывает ошибку данных FE (Frame Error).

Определение битов четности.

Бит четности определяется результатом операции *исключающего или* всех бит данных. Если выбрана нечетная четность, то также производится инверсия (НЕ). Связь между битами данных и битами четности такая:

$$P_{even} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 0$$

$$P_{odd} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 1$$

P_{even} – четный бит четности;

P_{odd} – нечетный бит четности;

d_n – n-ый бит данных.

Биты четности располагаются между битами данных и стоп-битами.

Инициализация USART.

Перед использованием USART нужно инициализировать. Инициализация обычно заключается в установке скорости и формата фрейма. Для того, чтобы использовать прерывания в работе USART, прерывания должны быть глобально запрещены на время инициализации.

При переинициализации, затрагивающей изменение скорости и формата, стоит удостовериться, что передача данных временно не ведется. Флаг TXC используется, чтобы удостовериться в том, что передатчик завершил передачу, а флаг RXC показывает наличие непрочитанных данных в принимающем буфере. TXC должен быть сброшен перед записью в UDR (передачей). Далее приведен пример инициализации на ассемблере и С. Оба примера эквивалентны. В примерах устанавливается асинхронный режим без опроса (прерывания выключены) и фиксированный

формат фреймов. Скорость передается как аргумент функции. В ассемблерном варианте как псевдоним скорости используются регистры r17:r16.

Assembly Code Example

```
USART_Init:
; Set baud rate
out UBRRH, r17
out UBRRL, r16
; Включить приемник и передатчик
ldi r16, (1<<RXEN)|(1<<TXEN)
out UCSRB,r16
; Формат фрейма: 8 бит данных, 2 стоп-бита
ldi r16, (1<<URSEL)|(1<<USBS)|(3<<UCSZ0)
out UCSRC,r16
ret
```

C Code Example

```
#define FOSC 1843200// Clock Speed
#define BAUD 9600
#define MYUBRR FOSC/16/BAUD-1
void main( void )
{
...
USART_Init ( MYUBRR );
...
}
void USART_Init( unsigned int ubrr)
{
/* Установка скорости */
UBRRH = (unsigned char)(ubrr>>8);
UBRRL = (unsigned char)ubrr;
/* Включить приемник и передатчик */
UCSRB = (1<<RXEN)|(1<<TXEN);
/* Формат фрейма: 8 бит данных, 2 стоп-бита */
UCSRC = (1<<URSEL)|(1<<USBS)|(3<<UCSZ0);
```

Более продвинутый вариант инициализации может включать и формат фрейма как параметр, и не только, но и прерывания и т.д. Также, большинство программ используют фиксированные настройки скорости и регистров управления. Для таких программ код инициализации можно поместить непосредственно в код самой программы или модуль всеобщей инициализации.

Передатчик USART включается установкой бита TXEN регистра UCSRB. Когда передатчик включен, вывод TxD перехватывается приемопередатчиком и используется для последовательной передачи данных. При синхронной передаче, вывод XCK также перехватывается. Инициализация должно быть проведена до любых передач.

Передача фреймов с 5-8 битами данных.

Передача осуществляется загрузкой данных в регистр передачи. ЦПУ может загрузить буфер передачи через запись в регистр UDR, Затем данные из буфера поступят в сдвиговый регистр, когда тот будет готов к передаче нового фрейма.

Следующие примеры кода показывают простую функцию передачи данных с использованием флага UDRE (USART Data Register Empty). При использовании фрейма с менее чем восемью битами данных, значительная часть данных регистра UDR игнорируется. В ассемблерном коде, данные для передачи держаться в регистре R16.

Assembly Code Example

```
USART_Transmit:
; Ждем очистки буфера
sbis UCSRA,UDRE
rjmp USART_Transmit
; Помещаем данные(r16) в буфер, отправляем данные
out UDR,r16
ret
```

C Code Example

```
void USART_Transmit( unsigned char data )
{
/* Ждем очистки буфера*/
while ( !( UCSRA & (1<<UDRE)) )
;
/* Помещаем данные в буфер */
UDR = data;
}
```

Функция ждет опустошения передающего буфера, периодически проверяя флаг UDRE. Если используется прерывание Data Register Empty, процесс прерывания пишет данные в буфер.

Передача фреймов с девятью битами данных.

Если размер данных установлен в 9 бит (UCSZ = 7), последний бит должен быть записан в TXB8 регистра UCSRB раньше, чем младший бит данных попадет в регистр UDR. Следующие примеры иллюстрируют передачу в режиме 9 бит.

Assembly Code Example

```
USART_Transmit:
; Ждем обнуления буфера передатчика
sbis UCSRA,UDRE
rjmp USART_Transmit
; копируем 9й бит из r17 в TXB8
cbi UCSRB,TXB8
sbrc r17,0
sbi UCSRB,TXB8
; Помещаем данные в буфер, отправляем фрейм
out UDR,r16
ret
```

C Code Example

```
void USART_Transmit( unsigned int data )
{
/* Ждем обнуления буфера передатчика */
while ( !( UCSRA & (1<<UDRE))) )
;
/* Копируем 9й бит в TXB8 */
UCSRB &= ~(1<<TXB8);
if ( data & 0x0100 )
UCSRB |= (1<<TXB8);
/* Помещаем данные в буфер, отправляем фрейм */
UDR = data;
}
```

Девятый бит может быть использован для индикации адреса в мультипроцессорном режиме.

Флаги и прерывания передатчика.

С передатчиком связаны два флага, показывающих его состояние: UDRE (USART Data Register Empty) и TXC (Transmit Complete). Оба флага можно использовать для генерации прерываний.

Флаг UDRE показывает, что буфер готов принять новые данные. Этот бит устанавливается в 1, если передающий буфер пуст, и в 0, если буфер содержит данные для передачи.

Если установлен флаг прерывания по пустому буферу (UDRIE в регистре UCSRB), прерывание будет генерироваться каждый раз, когда флаг UDRE установлен в 1. UDRE обнуляется записью в UDR.

Флаг TXC устанавливается в 1, когда весь фрейм полностью отправлен из сдвигового регистра и нет данных в передающем буфере. Флаг автоматически обнуляется при генерации прерывания «Передача завершена». Флаг TXC полезен для полудуплексного режима (напр. RS485), когда требуется освободить шину передачи данных незамедлительно для осуществления приема.

Если установлен бит TXCIE (Transmit Complete Interrupt Enable) регистра UCSRB, прерывание генерируется каждый раз, когда флаг TXC устанавливается в 1.

Генератор четности.

Генератор четности вычисляет бит четности для фрейма данных. Когда установлен бит UPM1 в 1, управляющая логика передатчика добавляет бит четности после битов данных.

Отключение передатчика.

Отключение передатчика (установка TXEN в 0) не даст результатов, пока присутствуют данные в сдвиговом регистре или передающем буфере. После отключения, вывод TxD более не перехватывается.

USART приемник. Прием данных.

Приемник включается установкой бита RXEN регистра UCSRB в 1. Когда приемник включен, вывод RxD перехватывается приемником и работает как последовательный вход. Инициализация должна проводиться до приема данных. В синхронном режиме вывод XCK также перехватывается приемопередатчиком.

Прием с 5-8 битами данных.

Приемник начинает прием данных сразу же, как на вывод RxD попадает корректный старт-бит (всегда 0). Все данные попадают в сдвиговый регистр, пока не будет принят первый старт-бит. Второй стоп-бит, если он есть, игнорируется. Данные из сдвигового регистра попадают в приемный буфер, откуда могут быть прочитаны чтением регистра UDR.

Следующий пример кода иллюстрирует прием данных с чтением флага RXC (Receive Complete). При использовании менее восьми бит данных, часть регистра UDR устанавливается маской с нулями. Перед приемом необходимо провести инициализацию.

Assembly Code Example

```
USART_Receive:
; Ожидание данных
sbis UCSRA, RXC
rjmp USART_Receive
; Получение и извлечение из буфера
in r16, UDR
ret
```

C Code Example

```
unsigned char USART_Receive( void )
{
/* Ожидание приема */
while ( !(UCSRA & (1<<RXC)) )
;
/* Извлечение данных из буфера */
return UDR;
}
```

Функция ожидает появление данных, циклически проверяя флаг RXC, затем читает данные из буфера.

Получение с 9 битами данных.

Если установлен режим девяти бит данных (UCSZ = 7), девятый бит должен быть прочитан из RXB8 регистра UCSRB раньше, чем младшие биты UDR. Правило заодно относится к битам FE, DOR и PE. Сперва читаются данные из UCSRA, только затем из UDR. Чтение UDR изменит все указанные биты.

Пример кода иллюстрирует функцию приема данных с 9ю битами.

Assembly Code Example

```
USART_Receive:
; Ждем отправки данных
sbis UCSRA, RXC
rjmp USART_Receive
; Получаем статус и 9й бит
in r18, UCSRA
in r17, UCSRB
in r16, UDR
; Возвращаем -1 в случае ошибки
andi r18,(1<<FE)|(1<<DOR)|(1<<PE)
breq USART_ReceiveNoError
ldi r17, HIGH(-1)
ldi r16, LOW(-1)
USART_ReceiveNoError:
lsr r17
andi r17, 0x01
ret
```

C Code Example

```
unsigned int USART_Receive( void )
{
unsigned char status, resh, resl;
/* Ждем приема данных */
while ( !(UCSRA & (1<<RXC)) )
;
/* Получаем статус и 9й бит */
/* Только потом читаем из регистра */
status = UCSRA;
resh = UCSRB;
resl = UDR;
/* В случае ошибки возвращаем -1 */
if ( status & (1<<FE)|(1<<DOR)|(1<<PE) )
return -1;
/* Извлекаем 9й бит и возвращаем результат */
resh = (resh >> 1) & 0x01;
return ((resh << 8) | resl);
}
```

Флаг Receive Complete и прерывание приема данных.

Флаг RXC (Receive Complete) показывает наличие непрочтенных данных в приемном регистре установкой в 1. Если передатчик выключен, приемный буфер сброшен и флаг постоянно установлен в 0.

Прерывание приема данных включается установкой бита RXCIE регистра UCSRB в 1 и генерируется каждый раз, когда флаг RXC устанавливается в 1.

Флаги ошибок приема.

С приемником связаны три флага ошибок: FE (Frame Error), DOR (Data Overrun) и PE (Parity Error). Все три флага доступны из регистра UCSRA. Флаги ошибок должны быть прочитаны до чтения приемного буфера, иначе они будут изменены. Флаги ошибок доступны только для чтения. Ни один флаг ошибок не может генерировать прерывания.

Флаг FE устанавливается в 1 получении некорректного стоп-бита. Может использоваться для отслеживания рассинхронизации или ошибок обработчиков протоколов.

Флаг DOR устанавливается при переполнении приемного буфера и сообщает о пропущенных фреймах между прошлым и последующим чтением из UDR.

Флаг PE устанавливается в 1 при получении некорректных битов четности. При выключенных битах четности всегда установлен в 0.

Сверщик четности.

Сверщик четности включается при установке бита UPM1. Четность или нечетность проверки выбирается битом UPM0. Чтение бита PE должно производиться до чтения регистра UDR.

Отключение приемника.

В противоположность передатчику, отключение приемника происходит незамедлительно. Недополученные данные безвозвратно теряются. Для отключение необходимо установить бит RXEN в 0. При выключенном приемнике, его регистры обнуляются, перехват вывода RxD прекращается.

Обнуление буфера приемника.

Буфер приемника обнуляется при его (приемника) отключении. Непрочтенные данные теряются. Если требуется очистить буфер приемника, не выключая его, то следует читать из UDR до обнуления флага RXC. Следующий пример показывает процедуру обнуления буфера приемника без выключения методом чтения из регистра UDR.

Assembly Code Example

```
USART_Flush:
sbis UCSRA, RXC
ret
in r16, UDR
rjmp USART_Flush
```

C Code Example

```
void USART_Flush( void )
{
unsigned char dummy;
while ( UCSRA & (1<<RXC) ) dummy = UDR;
}
```

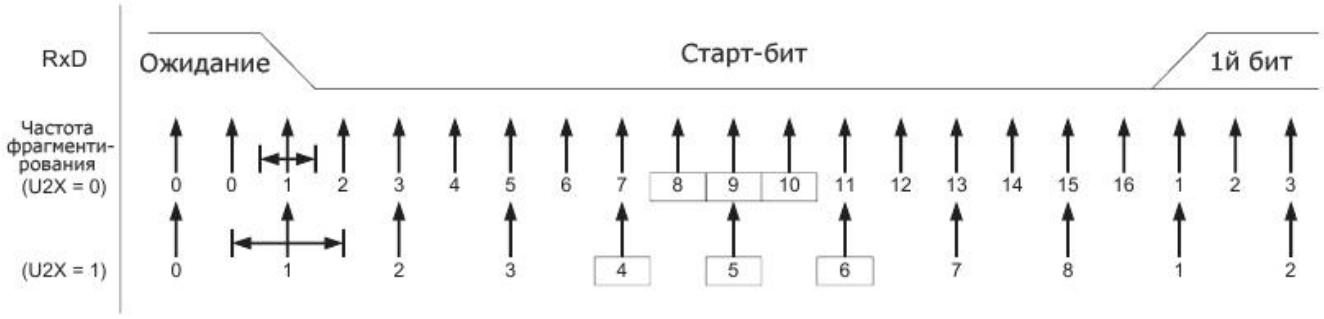
Асинхронный прием данных.

Приемопередатчик USART включает блоки восстановления частоты и данных. Блок восстановления частоты нужен для синхронизации внутренней частоты передачи данных с поступающими данными на выводе RxD. Блок восстановления данных фрагментирует сигнал и пропускает его через фильтр низкой частоты, что увеличивает помехоустойчивость приемника. Работа в асинхронном режиме приема данных зависит от точности генератора тактовой частоты, скорости приема и формата принимаемых фреймов.

Восстановление частоты в асинхронном режиме.

Блок восстановления частоты синхронизирует внутреннее тактирование с принимаемыми данными. На рис. 73 показан процесс фрагментирования старт-бита принимаемого фрейма. Частота восстановителя-фрагментатора в 16 раз больше частоты принимаемого сигнала, и в 8 раз больше для режима двойной скорости.

Рис. 73.



Когда блок восстановления частоты регистрирует изменение из высокого (ожидание) до низкого (старт-бит) уровня, регистрируется старт-бит.

Восстановление данных в асинхронном режиме.

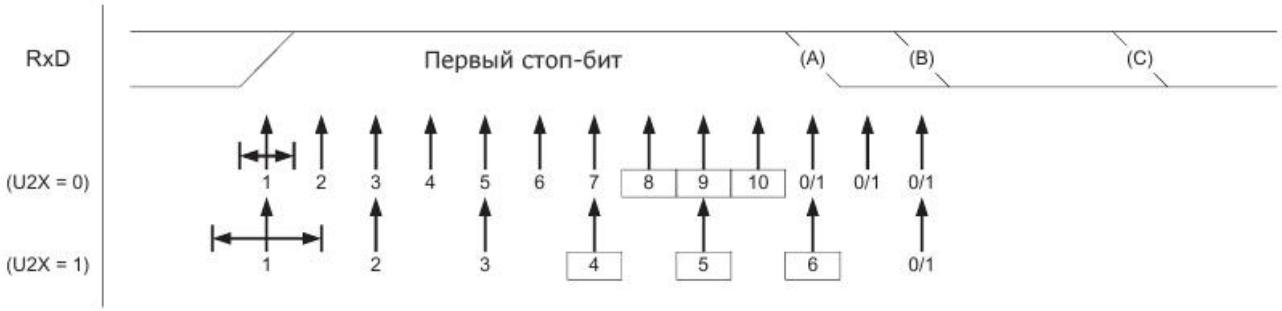
Когда приемник синхронизировался со старт-битом, начинается восстановление фрейма. Рис. 74 показывает, как происходит восстановление бит данных и бит четности.

Рис. 74.

Определение логического уровня принятого бита производится по анализу напряжения трех центральных фрагментов по следующей схеме: если хотя бы два фрагмента имеют высокий уровень, значит принята единица. Восстановления производится до принятия первого стоп-бита.

Рис. 75 показывает фрагментирование первого стоп-бита, и определение возможного начала следующего старт-бита.

Рис. 75.



Если стоп-бит регистрируется в низком уровне, устанавливается флаг FE. При стандартной скорости старт-бит ожидается в точке A, при двойной в точке B. Отметка C означает полную продолжительность стоп-бита. Раннее распознание старт-бита влияет на рабочий диапазон приемника (?)(The early start bit detection influences the operational range of the receiver).

Рабочий диапазон приемника.

Рабочий диапазон зависит от расхождения между частотой принимаемого сигнала и внутренней частотой приемника. Если передатчик генерирует сигнал слишком быстро или слишком медленно, приемник не сможет корректно определить начало фрейма.

Следующие уравнения можно использовать для вычисления максимальной и минимальной частоты корректно принимаемого сигнала:

$$R_{slow} = \frac{S(D+1)}{S-1+D\cdot S+S_F}$$

$$R_{fast} = \frac{S(D+2)}{S\ D+1\ \ +S_M}$$

где:

D – сумма бит данных и четности (от 5 до 10);

S – число фрагментов на принимаемый бит (S=16 для обычного режима и S=8 для режима двойной скорости);

S_F – стартовый фрагмент для схемы определения уровня (8 для нормальной скорости и 4 для двойной);

S_M – средний фрагмент для определения уровня (9 для нормальной скорости и 5 для двойной);

R_{slow} – значение для наименьшей частоты принимаемых данных;

R_{fast} – значение для наибольшей частоты принимаемых данных;

Максимальные допустимые ошибки представлены в таблицах 61 и 62. Режим двойной скорости имеет меньшую устойчивость к ошибкам.

Табл. 61. Для нормальной скорости

D (биты данных и четности)	R_{slow}, %	R_{fast}, %	Максимальное количество ошибок, %	Рекомендованный допуск ошибок приема, %
5	93.20	106.67	+6.67/-6.80	±3.0
6	94.12	105.79	+5.79/-5.88	±2.5
7	94.81	105.11	+5.11/-5.19	±2.0
8	95.36	104.58	+4.58/-4.54	±2.0
9	95.81	104.14	+4.14/-4.19	±1.5
10	96.17	103.78	+3.78/-3.83	±1.5

Табл. 62. Для двойной скорости.

D (биты данных и четности)	R_{slow}, %	R_{fast}, %	Максимальное количество ошибок, %	Рекомендованный допуск ошибок приема, %
5	94.12	105.66	+5.66/-5.88	±2.5
6	94.92	104.92	+4.92/-5.08	±2.0
7	95.52	104.35	+4.35/-4.48	±1.5
8	96.00	103.90	+3.90/-4.00	±1.5
9	96.39	103.53	+3.53/-3.61	±1.5
10	96.70	103.23	+3.23/-3.30	±1.0

Рекомендации составлены исходя из предположения, что ошибки провоцируются как передатчиком, так и приемником в равной степени.

Режим мультипроцессорной связи.

Включение мультипроцессорного режима производится установкой бита MPCM (Multi-processor Communication Mode) регистра UCSRA в 1. В этом режиме производится фильтрация фреймов в приемнике. Фреймы без адресной информации игнорируются и не попадают в приемный буфер. Такая схема работы позволяет разгрузить ЦПУ в устройстве с множеством приемопередатчиков, использующих общую шину. Бит MPCM не влияет на передатчик.

Если приемник установлен в режим 5-8 бит данных, первый стоп-бит определяет содержит фрейм данные или адрес. Если приемопередатчик установлен в режим 9 бит данных, то девятый бит используется для идентификации адресных фреймов. Когда бит идентификации (первый стоп-бит или 9й бит) установлен в 1, фрейм считается адресным, когда в 0 – фреймом данных.

Мультипроцессорный режим позволяет связать одно ведущее устройство с несколькими ведомыми. Ведомые устройства сперва получают адресный фрейм, затем либо принимают следующие фреймы, либо игнорируют.

Использование мультипроцессорного режима

Для устройства, выступающего в качестве ведущего можно использовать 9-битный режим (UCSZ = 7). Девятый бит должен быть 1 для адресных фреймов и 0 для фреймов данных. В этой ситуации ведомые устройства должны быть также установлены в 9-битный режим.

Для мультипроцессорной связи годится следующая схема:

1. Все ведомые устройства настроены в мультипроцессорный режим (MPCM в UCSRA в 1);
2. Ведущее устройство отправляет адресный фрейм, ведомые его получают;
3. Каждое ведомое устройство получает адресный фрейм и определяет, принимать ему данные или игнорировать; в положительном случае бит MPCM устанавливается в 0.
4. Адресованное устройство принимает все фреймы данных, остальные их игнорируют; и так до тех пор, пока не будет передан следующий адресный фрейм; У игнорирующих устройств бит MPCM установлен в 1;
5. Когда последний фрейм данных принят, адресованное устройство устанавливает бит MPCM и ожидает адресные фреймы. Процедура повторяется с пункта 2.

Использование 5-8 битного режима с адресацией возможно, но неудобно и непрактично. Не следует использовать

инструкции SBI и CBI для установки или сброса бита MPCM. Бит MPCM имеет тот же адрес, что и флаг TXC, и может быть случайно изменен.

Доступ к регистрам UBRRH и UCSRC.

Регистр UBRRH имеет тот же адрес, что и UCSRC. Поэтому для доступа к этим регистрам требуются особые методы и средства.

Доступ к этим регистрам контролируется битом URSEL: 0 – доступ к UBRRH, 1 – UCSRC.

Пример доступа к этим регистрам:

Assembly Code Example

```
...
; Устанавливаем UBRRH в 2
ldi r16,0x02
out UBRRH,r16
...
; Устанавливаем биты USBS и UCSZ1 в 1
; Остальные в 0
ldi r16,(1<<URSEL)|(1<<USBS)|(1<<UCSZ1)
out UCSRC,r16
...
```

C Code Example

```
...
/* Устанавливаем регистр UBRRH в 2 */
UBRRH = 0x02;
...
/* Устанавливаем биты USBS и UCSZ1 в 1, и */
/* остальные в 0 */
UCSRC = (1<<URSEL)|(1<<USBS)|(1<<UCSZ1);
...
```

Чтение UBRRH и UCSRC.

Чтение – более заковыристая операция но и используется она крайне редко. Определяется она считаемой последовательностью. При первом чтении возвращается значение UBRRH. Если этот адрес читался в предыдущей операции, в текущей чтение вернет значение UCSRC. Пример:

Assembly Code Example

```
USART_ReadUCSRC:
; Читаем UCSRC
in r16,UBRRH
in r16,UCSRC
ret
```

C Code Example

```
unsigned char USART_ReadUCSRC( void )
{
unsigned char ucsrc;
/* Читаем UCSRC */
ucsrc = UBRRH;
ucsrc = UCSRC;
return ucsrc;
}
```

Описание регистров приемопередатчика USART.

UDR – USART Data Register – Регистр данных.

Буфер как приемника, так и передатчика делят один адрес. Доступ к ним осуществляется через регистр UDR. При чтении из UDR данные читаются из регистра приемника RXB, При записи в UDR, данные попадают в регистр передатчика TXB.

Запись в регистр передатчика возможна только при установленном бите UDRE регистра UCSRA.

UCSRA – USART Control and Status Register A – Первый регистр управления и состояния.

Бит	7	6	5	4	3	2	1	0
	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM
Чтение /запись	R	RW	R	R	R	R	RW	RW
По умолчанию	0	0	1	0	0	0	0	0

- 7 – RXC – USART Receive Complete – Прием завершен
  Этот бит устанавливается в 1 как только появляются непрочитанные данные в буфере приемника, и в 0, когда

буфер пуст. При выключенном приемнике всегда установлен в 0. Может использоваться для генерации прерывания (см. описание на бит RXCIE).

- 6 – TXC – USART Transmit Complete – Передача завершена
  Этот бит устанавливается когда данные из сдвигового регистра передатчика отправлены, а новых данных для отправки нет. Может использоваться для генерации прерывания. Автоматически устанавливается в ноль при срабатывании прерывания Transmit Complete (см. описание на бит TXCIE).
- 5 – UDRE – USART Data Register Empty – Регистр данных пуст
  Бит оповещает о том, что регистр UDR пуст и готов к приему данных. Может использоваться для генерации прерывания (см. UDRIE).
- 4 – FE – Frame Error – Ошибка данных
  Устанавливается в 1 в случае приема низкого уровня в стоп-бите. . Должен читаться до регистра данных UDR.
- 3 – DOR – Data Overrun – Переполнение регистра приемника
  Бит устанавливается в 1, когда регистрируется переполнение регистра приемника. Должен читаться до регистра данных UDR.
- 2 – PE – Parity Error – Ошибка четности
  Устанавливается в 1 когда регистрируется некорректный бит четности. Актуально только при включенных битах четности (UPM1 = 1). . Должен читаться до регистра данных UDR.
- 1 – U2X – Double Transmission Speed – Режим двойной скорости
  Бит устанавливает двойную скорость в асинхронном режиме. В синхронном режиме следует устанавливать в 0.
  Установка в 1 перенастраивает делитель частоты с 16 на 8 при асинхронной работе.

- 0 – MPCM – Multi-processor Communication Mode – Режим мультипроцессорной связи
  Активирует режим мультипроцессорной связи. Если установлен в 1, все фреймы, кроме адресных, приемником игнорируются.

UCSRB – USART Control and Status Register B – Второй регистр управления и состояния.

Бит	7	6	5	4	3	2	1	0
	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8
Чтение /запись	RW	RW	RW	RW	RW	RW	R	RW
По умолчанию	0	0	0	0	0	0	0	0

- 7 – RXCIE – RX Complete Interrupt Enable – Разрешение прерывания по завершению приема
  Установка в 1 разрешает прерывание по флагу RXC.
- 6 – TXCIE – TX Complete Interrupt Enable – Разрешение прерывания по завершению передачи
  Установка в 1 разрешает прерывание по флагу TXC.
- 5 – UDRIE – USART Data Register Interrupt Enable – Разрешение прерывания пустого регистра данных
  Установка в 1 разрешает прерывание по флагу UDRE.
- 4 – RXEN – Receiver Enable – Приемник включен
  Установка в 1 включает приемник, перехват вывода RxD, флаги FE, DOR, PE. Выключение приемника происходит незамедлительно при сбросе бита в 0. При выключении буфер приемника очищается.
- 3 – TXEN – Transmitter Enable – Передатчик включен
  Установка в 1 включает передатчик и перехват вывода TxD. При выключении (установка в ноль) ничего не произойдет до тех пор, пока не завершится передача данных.
- 2 – UCSZ2 – USART Character Size – Размер данных
  Вместе с битами UCSZ1:0 используется для установки

количества бит данных во фреймах приемника и передатчика.

- 1 – RXB8 – Receive Data Bit 8 – Восьмой бит принятых данных
  При 9-битном размере данных последний бит сохраняется сюда. Читать следует до UDR.
- 0 – TXB8 – Transmit Data Bit 8 – Восьмой бит отправляемых данных.
  При 9-битном размере данных последний бит пишется сюда. Писать следует до записи в UDR.

UCSRC – USART Control and Status Register C – Третий регистр управления и состояния.

Бит	7	6	5	4	3	2	1	0
	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZO	UCPOL
Чтение /запись	RW	RW	RW	RW	RW	RW	RW	RW
По умолчанию	1	0	0	0	0	1	1	0

Регистр имеет тот же адрес, что и UBRRH.
См. соответствующую главу про доступ.

- 7 – URSEL – Register Select – Выбор регистра
  Переключает доступ между UCSRC и UBRRH. Для записи в UCSRC должен быть установлен в 0.
- 6 – UMSEL – USART Mode Select – Выбор режима
  0 – Асинхронный режим, 1 – синхронный.
- 5:4 – UPM1:0 – Parity Mode – Выбор четности
  Включают и настраивают генерацию и проверку четности.

UPM1	UPM0	Режим четности
0	0	Выключен
0	1	Зарезервирован
1	0	Включен, четная
1	1	Включен, нечетная

- 3 – USBS – Stop Bit Select – Установка стоп-битов
  0 – один стоп-бит, 1 – два стоп-бита.

- 2:1 – UCSZ1:0 – Character Size – Размер данных
  Определяют количество бит данных вместе с UCSZ2 в UCSRB для приемника и передатчика.

UCSZ2	UCSZ1	UCSZ0	Биты данных
0	0	0	5
0	0	1	6
0	1	0	7
0	1	1	8
1	0	0	зарезервировано
1	0	1	зарезервировано
1	1	0	зарезервировано
1	1	1	9

- 0 – UCPOL – Clock Polarity – Граница синхроимпульсов
  Используется только в синхронном режиме, в асинхронном следует установить в 0. Определяет фрагментацию сигнала на выводе XCK: 0 – по спаду импульса, 1 – по фронту.

UBRRH и UBRRL – регистры установки скорости передачи.

Бит	15	14	13	12	11	10	9	8
UBRRH	URSEL	-	-	-				
UBRRL								
Бит	7	6	5	4	3	2	1	0
Чт/Зп	RW	R	R	R	RW	RW	RW	RW
	RW	RW	RW	RW	RW	RW	RW	RW
По умолч.	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0

- 15 – URSEL – Register Select – Выбор регистра
  Переключает доступ между UCSRC и UBRRH. Для записи в UBRRH должен быть установлен в 0.
- 14:12 – Зарезервировано.
  Для совместимости с будущими контроллерами рекомендуется устанавливать в 0.
- 11:0 – UBRR – USART Baud Rate Register – Регистр установки скорости
  Эти биты составляют 12-битный регистр установки скорости USART. При изменении мгновенно

перенастраивают предделитель генератора частоты приемопередатчика.

Настройка скорости USART

Для стандартных номиналов кварцевых резонаторов и часто используемых скоростей передачи можно использовать данные из табл. 68. Для расчета ошибок используйте формулу:

$$Error\ \% = \frac{BaudRate_{Closest\ Match}}{BaudRate} - 1 \cdot 100\%$$

Таблица 68. Примеры настроек скорости

Скорость, бит/сек	f_{osc} = 1.0000 MHz				f_{osc} = 1.8432 MHz				f_{osc} = 2.0000 MHz			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	25	0.2%	51	0.2%	47	0.0%	95	0.0%	51	0.2%	103	0.2%
4800	12	0.2%	25	0.2%	23	0.0%	47	0.0%	25	0.2%	51	0.2%
9600	6	-7.0%	12	0.2%	11	0.0%	23	0.0%	12	0.2%	25	0.2%
14.4k	3	8.5%	8	-3.5%	7	0.0%	15	0.0%	8	-3.5%	16	2.1%
19.2k	2	8.5%	6	-7.0%	5	0.0%	11	0.0%	6	-7.0%	12	0.2%
28.8k	1	8.5%	3	8.5%	3	0.0%	7	0.0%	3	8.5%	8	-3.5%
38.4k	1	-18.6%	2	8.5%	2	0.0%	5	0.0%	2	8.5%	6	-7.0%
57.6k	0	8.5%	1	8.5%	1	0.0%	3	0.0%	1	8.5%	3	8.5%
76.8k	–	–	1	-18.6%	1	-25.0%	2	0.0%	1	-18.6%	2	8.5%
115.2k	–	–	0	8.5%	0	0.0%	1	0.0%	0	8.5%	1	8.5%
230.4k	–	–	–	–	–	–	0	0.0%	–	–	–	–
250k	–	–	–	–	–	–	–	–	–	–	0	0.0%
Max (1)	62.5 Kbps		125 Kbps		115.2 Kbps		230.4 Kbps		125 Kbps		250 Kbps	

Скорость, бит/сек	f_{osc} = 3.6864 MHz				f_{osc} = 4.0000 MHz				f_{osc} = 7.3728 MHz			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	95	0.0%	191	0.0%	103	0.2%	207	0.2%	191	0.0%	383	0.0%
4800	47	0.0%	95	0.0%	51	0.2%	103	0.2%	95	0.0%	191	0.0%
9600	23	0.0%	47	0.0%	25	0.2%	51	0.2%	47	0.0%	95	0.0%
14.4k	15	0.0%	31	0.0%	16	2.1%	34	-0.8%	31	0.0%	63	0.0%
19.2k	11	0.0%	23	0.0%	12	0.2%	25	0.2%	23	0.0%	47	0.0%
28.8k	7	0.0%	15	0.0%	8	-3.5%	16	2.1%	15	0.0%	31	0.0%
38.4k	5	0.0%	11	0.0%	6	-7.0%	12	0.2%	11	0.0%	23	0.0%
57.6k	3	0.0%	7	0.0%	3	8.5%	8	-3.5%	7	0.0%	15	0.0%
76.8k	2	0.0%	5	0.0%	2	8.5%	6	-7.0%	5	0.0%	11	0.0%
115.2k	1	0.0%	3	0.0%	1	8.5%	3	8.5%	3	0.0%	7	0.0%
230.4k	0	0.0%	1	0.0%	0	8.5%	1	8.5%	1	0.0%	3	0.0%
250k	0	-7.8%	1	-7.8%	0	0.0%	1	0.0%	1	-7.8%	3	-7.8%
0.5M	–	–	0	-7.8%	–	–	0	0.0%	0	-7.8%	1	-7.8%
1M	–	–	–	–	–	–	–	–	–	–	0	-7.8%
Max (1)	230.4 Kbps		460.8 Kbps		250 Kbps		0.5 Mbps		460.8 Kbps		921.6 Kbps	

Скорость, бит/сек	f_{osc} = 8.0000 MHz				f_{osc} = 11.0592 MHz				f_{osc} = 14.7456 MHz			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	207	0.2%	416	-0.1%	287	0.0%	575	0.0%	383	0.0%	767	0.0%
4800	103	0.2%	207	0.2%	143	0.0%	287	0.0%	191	0.0%	383	0.0%
9600	51	0.2%	103	0.2%	71	0.0%	143	0.0%	95	0.0%	191	0.0%
14.4k	34	-0.8%	68	0.6%	47	0.0%	95	0.0%	63	0.0%	127	0.0%
19.2k	25	0.2%	51	0.2%	35	0.0%	71	0.0%	47	0.0%	95	0.0%
28.8k	16	2.1%	34	-0.8%	23	0.0%	47	0.0%	31	0.0%	63	0.0%
38.4k	12	0.2%	25	0.2%	17	0.0%	35	0.0%	23	0.0%	47	0.0%
57.6k	8	-3.5%	16	2.1%	11	0.0%	23	0.0%	15	0.0%	31	0.0%
76.8k	6	-7.0%	12	0.2%	8	0.0%	17	0.0%	11	0.0%	23	0.0%
115.2k	3	8.5%	8	-3.5%	5	0.0%	11	0.0%	7	0.0%	15	0.0%
230.4k	1	8.5%	3	8.5%	2	0.0%	5	0.0%	3	0.0%	7	0.0%
250k	1	0.0%	3	0.0%	2	-7.8%	5	-7.8%	3	-7.8%	6	5.3%
0.5M	0	0.0%	1	0.0%	–	–	2	-7.8%	1	-7.8%	3	-7.8%
1M	–	–	0	0.0%	–	–	–	–	0	-7.8%	1	-7.8%
Max (1)	0.5 Mbps		1 Mbps		691.2 Kbps		1.3824 Mbps		921.6 Kbps		1.8432 Mbps	

Скорость, бит/сек	f_{osc} = 16.0000 MHz				f_{osc} = 18.4320 MHz				f_{osc} = 20.0000 MHz			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	416	-0.1%	832	0.0%	479	0.0%	959	0.0%	520	0.0%	1041	0.0%
4800	207	0.2%	416	-0.1%	239	0.0%	479	0.0%	259	0.2%	520	0.0%
9600	103	0.2%	207	0.2%	119	0.0%	239	0.0%	129	0.2%	259	0.2%
14.4k	68	0.6%	138	-0.1%	79	0.0%	159	0.0%	86	-0.2%	173	-0.2%
19.2k	51	0.2%	103	0.2%	59	0.0%	119	0.0%	64	0.2%	129	0.2%
28.8k	34	-0.8%	68	0.6%	39	0.0%	79	0.0%	42	0.9%	86	-0.2%
38.4k	25	0.2%	51	0.2%	29	0.0%	59	0.0%	32	-1.4%	64	0.2%
57.6k	16	2.1%	34	-0.8%	19	0.0%	39	0.0%	21	-1.4%	42	0.9%
76.8k	12	0.2%	25	0.2%	14	0.0%	29	0.0%	15	1.7%	32	-1.4%
115.2k	8	-3.5%	16	2.1%	9	0.0%	19	0.0%	10	-1.4%	21	-1.4%
230.4k	3	8.5%	8	-3.5%	4	0.0%	9	0.0%	4	8.5%	10	-1.4%
250k	3	0.0%	7	0.0%	4	-7.8%	8	2.4%	4	0.0%	9	0.0%
0.5M	1	0.0%	3	0.0%	–	–	4	-7.8%	–	–	4	0.0%
1M	0	0.0%	1	0.0%	–	–	–	–	–	–	–	–
Max (1)	1 Mbps		2 Mbps		1.152 Mbps		2.304 Mbps		1.25 Mbps		2.5 Mbps	

Перевел Ozze, 18.12.11