Работа №8

Использование интерфейса UART в МК 1986BE9x.

Цель работы: изучить особенности использования контроллера последовательного интерфейса UART в МК 1986BE9x.

Задачи:

- изучить функциональный состав модуля MDR_UART;
- получить навыки программирования контроллера UART;
- закрепить на практике особенности программирования выводов общего назначения микроконтроллера для работы в конкретных режимах;
- закрепить на практике особенности использования системы прерываний микроконтроллера;
- закрепить навыки работы с дисплеем МЭЛТ 12864;
- познакомиться со структурой представления времени из библиотеки.
- закрепить навыки работы с библиотеками Common Microcontroller Software Interface Standard (CMSIS).

Используемое оборудование:

1. Отладочная плата с микроконтроллером Миландр MDR1986BE92QI/MDR1986BE93У.
2. Комплект Программатора:
   2.1 Программатор JLINK (USB-JTAG);
   2.2 Кабель USB 2.0 A – B;
   2.3 Шина (20 проводников).
3. Источник питания 5В (или дополнительный USB-кабель);
4. Комплект Логического анализатора:
   4.1 Логический анализатор;
   4.2. Кабель USB 2.0A – mini-B;
   4.3 Шина 10 проводников.
5. Элемент питания CR2032;
6. Преобразователь USB-TTL.

Используемая документация:

1. Техническое описание на ядро Миландр MDR1986BE9x (файл «1_Тех_описание_ядро_1986BE9X.pdf или по ссылке с сайта разработчика https://ic.milandr.ru/upload/iblock/a33/ioaf9ygfmq1lbxfhd5aad0mukg3dc93s/1986%D0%92%D0%959X.pdf)

2. Выводы отладочной платы микроконтроллера 1986BE92QI (файл «2_Выводы_платы_1986BE92QI.pdf или по ссылке* с сайта разработчика https://ic.milandr.ru/upload/iblock/8f6/8f67b8b736b3ec94edbbeb4777a9c4db.zip)

3. Выводы отладочной платы микроконтроллера 1986BE93 (файл «2_Выводы_платы_1986BE93У.pdf или по ссылке* с сайта разработчика https://ic.milandr.ru/upload/iblock/782/782c4c3b486d6f8d92995e9a44a94401.zip)

При загрузке схемотехнической документации с сайта Milandr (информация о выводах платы), загружается zip архив, который содержит 1 или 2 файла со схемой размещения выводов микроконтроллера на плате и несколько схемотехнических файлов для разводки и печати платы.

Теоретическая часть

UART

UART (Universal Asynchroanous Receiver/Transmitter) – последовательный асинхронный интерфейс передачи данных. Для передачи используются два проводника RX (Receiver - приемник) и TX (Transmitter – передатчик), которые отвечают за передачу информации в противоположных направлениях. Обычно в передаче также используется линия GND. Схема соединения двух устройств по UART представлена на рисунке 8.1.

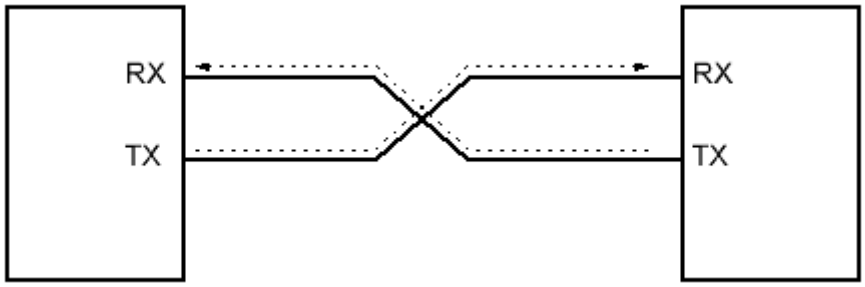

Рисунок 8.1 – Подключение двух устройств

Передача данных в UART выполняется следующим образом. В момент, когда линия свободна, на ней установлена 1. Первый бит пакета – стартовый, равен 0. Далее передаётся информационная часть, обычно вначале идут младшие биты. Информационная часть может содержать от 5-ти до 9-ти бит (в некоторых случаях даже до 12-ти). После, если потребуется, передаётся бит чётности для контроля целостности данных (если признак чётности, вычисляемый приёмником, не совпадет с передаваемым битом паритета, приёмник будет понимать, что данные переданы некорректно). Передача завершается стоповыми битами, которых может быть один или два. Протокол передачи данных по UART представлен на рисунке 8.2.

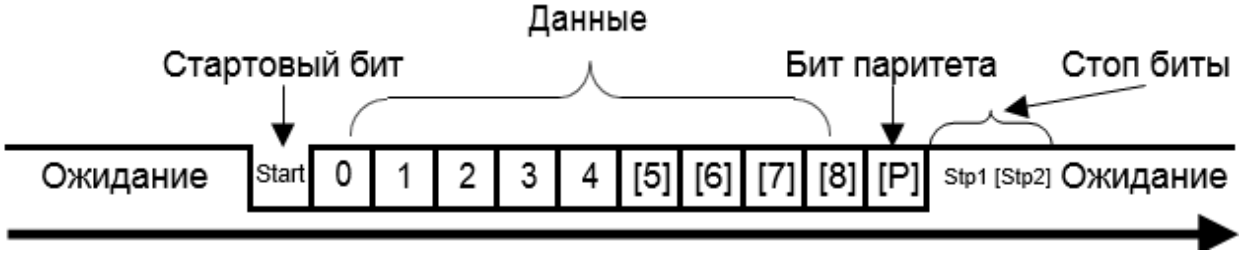

Рисунок 8.2 – Передача данных в UART

Скорость передачи определяется количеством бит, передаваемых в секунду (количество бит в секунду также называют бодами). Существует общепринятый ряд стандартных скоростей: 300; 600; 1200; 2400; 4800; 9600; 19200; 38400; 57600; 115200; 230400; 460800; 921600 бит/с.

Для управления потоком данных UART используется программный или аппаратный метод.

В случае **программного метода** информация о готовности устройства принимать данные или о необходимости остановить передачу передаётся по тем же каналам, что и данные. Принимающая сторона программно разделяет данные и управляющие сигналы в соответствии с принятым протоколом.

Аппаратное управление может использоваться некоторыми медленными устройствами или устройствами с простой схемной реализацией, однако оно потребует двух дополнительных линий для подключения устройства. При использовании аппаратного метода интерфейс UART предусматривает возможность использования дополнительных сигналов **CTS** и **RTS.** Подключение устройств с использованием аппаратного управления передачей изображено на рисунке 8.3.

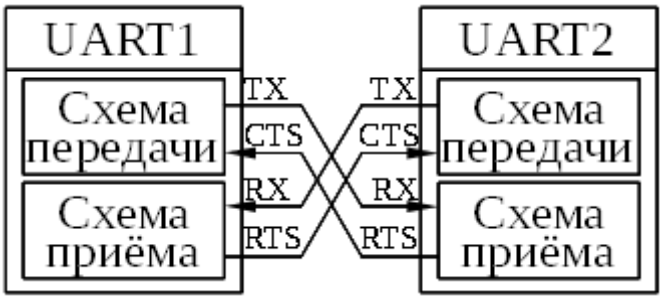

Рисунок 8.3 – Аппаратное управление

CTS (Clear To Send) – устанавливает принимающая сторона при готовности к приёму данных, активный уровень 0.

RTS (Request To Send) – запрос на отправку данных от передающей стороны для управления потоком данных.

Перед отправкой данных передатчик устанавливает сигнал **RTS** в 0. Если на **CTS** будет также установлен низкий уровень, передача происходит, иначе нет. Если сигнал **CTS** будет установлен во время передачи

информации, текущая передача всё равно будет завершена перед остановкой.

В некоторых случаях UART может поддерживать сигнал синхронизации, который сопровождает биты данных по отдельной линии передачи и по его изменениям приёмник фиксирует значения с линии данных. При такой реализации интерфейс обозначается как USART.

UART лежит в основе нескольких интерфейсов: COM, RS232, RS422, RS423, RS485. С точки зрения программиста, они практически одинаковы, разница проявляется на физическом уровне. Логические выводы контроллера RX и TX подключаются к специальной микросхеме – драйверу интерфейса, который и выполняет согласование логических уровней. Например, RS232 кодирует 0 напряжением +5 - +15В, логическая 1 напряжением -5 - -15В.

Особенности контроллера UART

Контроллер UART может быть запрограммирован для использования как в качестве универсального асинхронного приемопередатчика, так и для инфракрасного обмена данными (SIR). Содержит независимые буферы приема (16x12) и передачи (16x8) типа FIFO (First In First Out – первый вошел, первый вышел), что позволяет снизить интенсивность прерываний центрального процессора. Программное отключение FIFO позволяет ограничить размер буфера одним байтом. Программное управление скоростью обмена. Обеспечивается возможность деления тактовой частоты опорного генератора в диапазоне (1x16 – 65535x16). Допускается использование нецелых коэффициентов деления частоты, что позволяет использовать любой опорный генератор с частотой более 3,6864 МГц. Поддержка стандартных элементов асинхронного протокола связи – стартового и стопового бит, а также бита контроля четности, которые добавляются перед передачей и удаляются после приема. Независимое маскирование прерываний от буфера FIFO передатчика, буфера FIFO приемника, по таймауту приемника, по изменению линий состояния модема, а также в случае обнаружения ошибки. Поддержка прямого доступа к памяти. Обнаружение ложных стартовых бит. Формирование и обнаружения

сигнала разрыва линии. Поддержка функция управления модемом (линии CTS, DCD, DSR, RTS, DTR и RI). Возможность организации аппаратного управления потоком данных.

Наличие идентификационного регистра, однозначно идентифицирующего модуль, что позволяет операционной системе выполнять автоматическую конфигурацию.

Программируемые параметры

Следующие ключевые параметры могут быть заданы программно:

– скорость передачи данных – целая и дробная часть числа (от 0 до UARTCLK/16 Бод);

– количество бит данных (5-8);

– количество стоповых бит (1-2);

– режим контроля четности (формирование и контроль четности (проверочный бит выставляется по четности, нечетности, имеет фиксированное значение, либо не передается);

– разрешение или запрет использования буферов FIFO (глубина очереди данных – 16 элементов или один элемент соответственно);

– порог срабатывания прерывания по заполнению буферов FIFO (1/8, 1/4, 1/2, 3/4 и 7/8);

– частота внутреннего тактового генератора (номинальное значение – 1,8432 МГц) может быть задана в диапазоне 1,42 – 2,12 МГц для обеспечения возможности формирования бит данных с укороченной длительностью в режиме пониженного энергопотребления;

– режим аппаратного управления потоком данных.

Общая структурная схема UART в МК1986ВЕ9х представлена на рисунке 8.4.

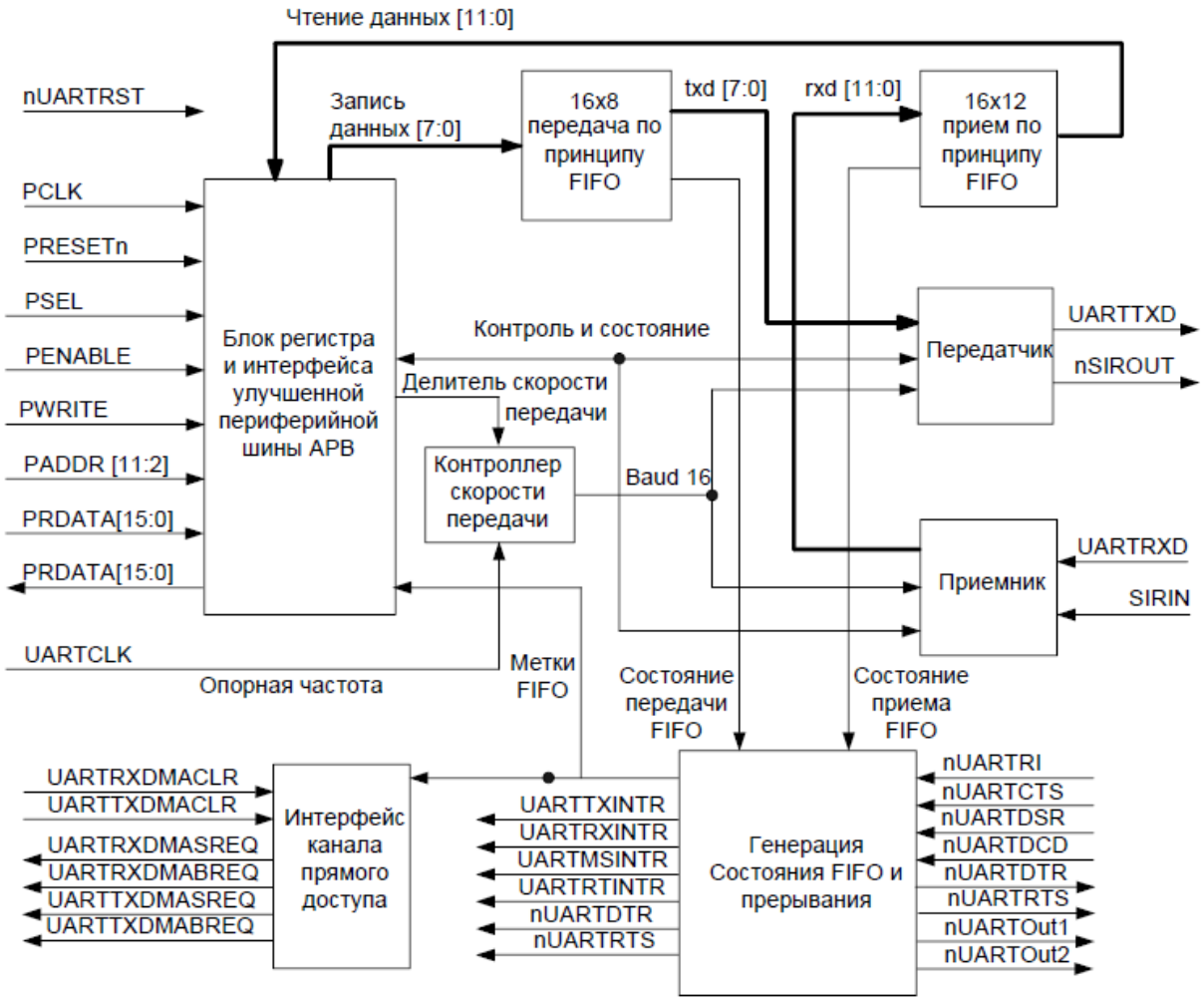

Рисунок 8.4 – Структурная схема UART

Сброс параметров контроллера UARTx

```
void UART_DeInit(MDR_UART_TypeDef* UARTx);
```

Инициализация параметров контроллера UARTx (возвращает параметр BaudRateStatus)

```
BaudRateStatus UART_Init(MDR_UART_TypeDef* UARTx, UART_InitTypeDef*
UART_InitStruct);
```

Инициализация значений структуры параметров контроллера UARTx

```
void UART_StructInit(UART_InitTypeDef* UART_InitStruct);
```

Включение/выключение модуля UARTx

```
void UART_Cmd(MDR_UART_TypeDef* UARTx, FunctionalState NewState);
```

Включение/выключение прерываний контроллера UARTx

```
void UART_ITConfig(MDR_UART_TypeDef* UARTx, uint32_t UART_IT, FunctionalState
NewState);
```

Проверка состояния прерывания UARTx

```
ITStatus UART_GetITStatus(MDR_UART_TypeDef* UARTx, uint32_t UART_IT);
```

Проверка маскирования (разрешения) прерывания UARTx

```
ITStatus UART_GetITStatusMasked(MDR_UART_TypeDef* UARTx, uint32_t UART_IT);
```

Сброс флага прерывания

```
void UART_ClearITPendingBit(MDR_UART_TypeDef* UARTx, uint32_t UART_IT);
```

Настройка параметров прямого доступа к памяти

```
void UART_DMAConfig(MDR_UART_TypeDef* UARTx, uint32_t UART_IT_RB_LVL,
uint32_t UART_IT_TB_LVL);
```

Включение/выключение прямого доступа к памяти

```
void UART_DMACmd(MDR_UART_TypeDef* UARTx, uint32_t UART_DMAReq,
FunctionalState NewState);
```

Отправить байт данных по UART
```
void UART_SendData(MDR_UART_TypeDef* UARTx, uint16_t Data);
```
Получить байт данных по UART
```
uint16_t UART_ReceiveData(MDR_UART_TypeDef* UARTx);
```
Разрыв линии
```
void UART_BreakLine(MDR_UART_TypeDef* UARTx, FunctionalState NewState);
```
Настройка параметров инфракрасного передатчика
```
void UART_IrDAConfig(MDR_UART_TypeDef* UARTx, uint32_t UART_IrDAMode);
```
Включение модуля UART в режиме инфракрасной передачи
```
void UART_IrDACmd(MDR_UART_TypeDef* UARTx, FunctionalState NewState);
```
Проверка состояния контроллера UART
```
FlagStatus UART_GetFlagStatus(MDR_UART_TypeDef* UARTx, uint32_t UART_FLAG);
```
Установка параметров тактирования UART
```
void UART_BRGInit(MDR_UART_TypeDef* UARTx, uint32_t UART_BRG);
```
Структура инициализации UART
```
UART_InitTypeDef
UART_BaudRate – значение тактовой частоты (скорости передачи данных)
UART_WordLength – количетво бит данных в передаче
UART_StopBits – количество стоп битов
UART_Parity – формат бита паритета
UART_FIFOMode – параметр включения FIFO
UART_HardwareFlowControl – аппаратный контроль приёма/передачи, включение и
выключение приёмника и передатчика
```
Длина слова данных в передаче UART
```
UART_WordLength5b – длина слова данных в передаче 5 бит
UART_WordLength6b – длина слова данных в передаче 6 бит
UART_WordLength7b – длина слова данных в передаче 7 бит
UART_WordLength8b – длина слова данных в передаче 8 бит
```
Количество стоповых бит UART
```
UART_StopBits1 – один стоповый бит
UART_StopBits2 – два стоповых бита
```
Биты паритета
```
UART_Parity_No - без проверки
UART_Parity_Even – проверка четности
UART_Parity_Odd – проверка нечетности
UART_Parity_1 – всегда 1
UART_Parity_0 – всегда 0
```
Включение буфера FIFO
```
UART_FIFO_OFF – буфер выключен
UART_FIFO_ON – буфер включен
```
Аппаратный контроль UART
```
UART_HardwareFlowControl_None – аппаратный контроль отключен
UART_HardwareFlowControl_CTSEn - Разрешение управления потоком данных по CTS.
1 – разрешено, данные передаются в линию только при активном значении сигнала
nUARTCTS.
UART_HardwareFlowControl_RTSEn - Разрешение управления потоком данных по RTS.
1 – разрешено. Запрос данных от внешнего устройства осуществляется только при
наличии свободного места в буфере FIFO приемника
UART_HardwareFlowControl_Out2 - Инверсия сигнала на линии состояния модема
nUARTOut2. В режиме оконечного оборудования (DTE) эта линия может
использоваться в качестве линии «сигнал вызова» (RI)
UART_HardwareFlowControl_Out1 - Инверсия сигнала на линии состояния модема
nUARTOut1. В режиме оконечного оборудования (DTE) эта линия может
использоваться в качестве линии «обнаружен информационный сигнал» (DCD)
UART_HardwareFlowControl_RTS - Инверсия сигнала на линии состояния модема
nUARTRTS
UART_HardwareFlowControl_DTR - Инверсия сигнала на линии состояния модема
nUARTDTR
UART_HardwareFlowControl_RXE – разрешение работы приёмника
UART_HardwareFlowControl_TXE – разрешение работы передатчика
UART_HardwareFlowControl_LBE – 0 запрещено;
```

1 – шлейф разрешен.
В режиме разрешенного шлейфа:
Если установлены бит SIREN=1 и бит регистра управления тестированием UARTTCR SIRTEST=1, то сигнал с выхода кодека nSIROUT инвертируется и подается на вход кодека SIRIN. Бит SIRTEST устанавливается в 1 для того, чтобы вывести устройство из полудуплексного режима, характерного для интерфейса SIR. После окончания тестирования по шлейфу бит SIRTEST должен быть установлен в 0.
Если бит SIRTEST=0, то выходная линия передатчика UARTTXD коммутируется на вход приемника UARTRXD.
Как в режиме SIR, так и в режиме UART, выходные линии состояния модема коммутируются на соответствующие входные линии.
После сброса бит устанавливается в 0

Прерывания UART

UART_IT_OE – прерывание по переполнению буфера
UART_IT_BE – прерывание по разрыву линии
UART_IT_PE – прерывание по ошибке контроля четности
UART_IT_FE – прерывание по ошибке в структуре кадра
UART_IT_RT – прерывание по тайм ауту приёма данных
UART_IT_TX – прерывание от передатчика
UART_IT_RX – прерывание от приёмника
UART_IT_DSR – прерывание по изменению состояния линии nUARTDSR
UART_IT_DCD – по изменению состояния линии nUARTDCD
UART_IT_CTS – по изменению состояния линии nUARTCTS
UART_IT_RI – изменению состояния линии nUARTRI

Количество слов в буфере FIFO UART

UART_IT_FIFO_LVL_2words – 2 слова
UART_IT_FIFO_LVL_4words – 4 слова
UART_IT_FIFO_LVL_8words – 8 слов
UART_IT_FIFO_LVL_12words – 12 слов
UART_IT_FIFO_LVL_14words – 14 слов

Режимы потребления энергии при работе с инфракрасным передатчиком

UART_IrDAMode_LowPower режим пониженного энергопотребления
UART_IrDAMode_Normal режим нормального энергопотребления

Флаги состояния UART_FLAG

UART_FLAG_RI – Инверсия линии nUARTRI
UART_FLAG_TXFE – Буфер FIFO передатчика пуст
UART_FLAG_RXFF - Буфер FIFO приёмника полон
UART_FLAG_TXFE – Буфер FIFO передатчика пуст
UART_FLAG_RXFF - Буфер FIFO приёмника полон
UART_FLAG_BUSY - UART занят
UART_FLAG_DCD - Инверсия линии nUARTDCD
UART_FLAG_DSR - Инверсия линии nUARTDSR
UART_FLAG_CTS - Инверсия линии nUARTCTS

Ошибки принимаемой информации

UART_Data_OE – переполнение буфера приёмника
UART_Data_BE – разрыв линии
UART_Data_PE – ошибка контроля четности
UART_Data_FE – ошибка в структуре кадра

Предделитель частоты UART

UART_HCLKdiv1 – деление частоты на 1
UART_HCLKdiv2 – деление частоты на 2
UART_HCLKdiv4 – деление частоты на 4
UART_HCLKdiv8 – деление частоты на 8
UART_HCLKdiv16 – деление частоты на 16
UART_HCLKdiv32 – деление частоты на 32
UART_HCLKdiv64 – деление частоты на 64
UART_HCLKdiv128 – деление частоты на 128

Практическая часть

Задание 7.1. Создайте проект по алгоритму из работы 1.

Добавьте в функцию SystemInit цикл, позволяющий задержать начало выполнения основной программы процессором на 5 секунд (файл system_MDR32F9Qx.c).

Правильно определите используемый JTAG-разъем в файле MDR32F9Qx_config.h.

Опишите процедуру инициализации модуля RST_CLK, обеспечивающую переход процессора на генератор HSE на частоте 8МГц.

Опишите процедуру инициализации выводов контроллера для работы со светодиодами: PC0 для 1986BE92QI или PF0 для 1986BE93У.

Опишите инициализацию таймера SysTick и загрузите в него значение для формирования секундных задержек (учтите сформированную частоту в рамках задания, не забудьте подать тактовый сигнал на SysTick).

Опишите процедуру обработчика прерывания от системного таймера. В теле обработчика опишите процедуру изменения состояния светодиода на противоположное. Вход в обработчик будет осуществляться раз в секунду, поэтому не нужно сразу выполнять операции вывода 1 и 0 и использовать задержки.

В основной программе обеспечьте тактирование используемых портов, инициализацию портов, глобальное разрешение прерываний.

Опишите процедуру инициализации выводов GPIO PORTA6, PORTA7 в режиме работы UART1 .

Опишите процедуру инициализации модуля UART в соответствии с примером:

```
UART_BRGInit(MDR_UART1, UART_HCLKdiv1);
UART_InitTypeDef UConf;
UConf.UART_BaudRate = 115200;
UConf.UART_StopBits = UART_StopBits1;
UConf.UART_WordLength = UART_WordLength8b;
UConf.UART_Parity = UART_Parity_No;
UConf.UART_HardwareFlowControl = UART_HardwareFlowControl_TXE;
UConf.UART_FIFOMode = UART_FIFO_OFF;
UART_Init(MDR_UART1, &UConf);
UART_Cmd(MDR_UART1, ENABLE);
```

```
UART_ITConfig(MDR_UART1, UART_IT_RX, ENABLE);
```

Прокомментируйте все строки процедуры инициализации.

Разрешите тактирование контроллера UART1 в основной программе.

В обработчике прерывания от системного таймера выполните отправку слова данных. Перед отправкой необходимо обеспечить проверку на занятость линии:

```
while (UART_GetFlagStatus(MDR_UART1, UART_FLAG_TXFE)==RESET);
```

Выполните сборку проекта. Загрузите программу в контроллер. Проанализируйте работу контроллера с помощью логического анализатора. Зафиксируйте реакцию системы в отчёте.

Проанализируйте работу контроллера с помощью монитора порта. Для передачи данных на компьютер используйте модуль преобразователь USB-TTL. При подключении учтите, что RX контроллера должен быть подключен к TX преобразователя и наоборот. Не забудьте подключить общий gnd. Зафиксируйте реакцию системы в отчёте.

Задание 7.2. В программе из задания 7.1 включите разрешение буфера TX. В процедуре обработчика прерывания SysTick опишите отправку 8 слов. Проверку занятости линии выполните только перед первой отправкой.

Выполните сборку проекта. Загрузите программу в контроллер. Проанализируйте работу контроллера, в том числе при отключении основного питания от платы. Зафиксируйте реакцию системы в отчёте.

Проанализируйте работу контроллера с помощью монитора порта. Для передачи данных на компьютер используйте модуль преобразователь USB-TTL. Зафиксируйте реакцию системы в отчёте.

Задание 7.3.

В программе из задания 7.1 в процедуре инициализации UART добавьте включение режима приёма информации и прерывания по приёму. Включите прерывание от модуля UART1. Опишите процедуру обработчика прерывания, в которой полученный с компьютера символ выводится на

дисплей. Вывод символов должен осуществляться построчно. Для инициализации дисплея воспользуйтесь работами 3, 4.

Проанализируйте работу контроллера с помощью монитора порта. Для передачи данных на компьютер используйте модуль преобразователь USB-TTL. Зафиксируйте реакцию системы в отчёте.

Задание 7.4.

В программе из задания 7.3 в процедуре инициализации UART добавьте включение буфера FIFO. В обработчике прерывания учтите объем данных, которые будет необходимо обработать за одно выполнение процедуры. Проанализируйте работу контроллера с помощью монитора порта. Для передачи данных на компьютер используйте модуль преобразователь USB-TTL. Зафиксируйте реакцию системы в отчёте.

Задание 7.5.

Измените программу из задания 7.3 (или 7.4) таким образом, чтобы контроллер получал значение системного времени в виде числа в формате uint32_t с компьютера, записывал его в счётчик RTC и выводил на экран.

Для инициализации модуля RTC воспользуйтесь примерами из работы №6.