

Тестовое техническое задание (программирование)

Необходимо написать на языке C следующие функции-драйверы нескольких устройств, входящих в состав системы на кристалле:

- 1) Функция вывода строки с помощью контроллера UART с произвольным числом символов, последний символ маркируется как “\0”. Функция в качестве аргумента принимает указатель на эту строку
- 2) Функция приема с помощью контроллера UART произвольного числа символов, определяемого аргументом функции и загрузку в ОЗУ по адресу, передаваемому в качестве аргумента.
- 3) Функция запуска таймера на отсчёт заданного числа тактов, окончание счёта обрабатывается с помощью сигнала прерывания от таймера.

Требуется написать предлагаемые функции на языке C, допускается использование ассемблерных вставок архитектуры MIPS-32 rev.6 либо RV32I на выбор студента.

Ниже приведена справочная информация, содержащая программные модели контроллера UART и таймера.

Контроллер UART

Контроллер UART необходим для установки соединения по каналу UART. Скорость передачи составляет 921600 бод. Формат транзакции имеет вид «1 Start bit – 8 Data bits – 1 Stop bit».

Базовый адрес контроллера UART в системе = 0xAFF0_0000. Контроллер UART имеет несколько регистров, доступных программисту, формат и адреса которых представлен в таблице 1.

Таблица 1 – Регистры контроллера UART

Смещение	Название	Права записи	Значение по reset	Краткое описание
0x00	Status	R, R/W	0	Регистр состояния
0x04	Received	R	0	Принятый байт
0x08	ToTransmit	R/W	0	Отправляемый байт

Регистр Status

Регистр служит для управления контроллером UART и описан в таблице 2.

Таблица 2 - поля регистра Status (UART)

Разряд	Название	Права записи	Значение по reset	Описание
31:3	Reserved	R	0	Зарезервировано
2	Miss	R	0	Флаг потери данных
1	Send	R/W	0	Разрешение отправки
0	Rec	R	0	Флаг принятого байта

Бит **Rec**, взведенный в «1» означает, что регистр Received хранит принятые данные, которые еще не были считаны пользователем. Контроллер UART обладает специальным FIFO, хранящим приятные данные, глубиной 8 байт. Каждое считывание пользователем регистра Received удаляет один

актуальный байт из FIFO. Бит Rec принимает значение 0, если пользователь прочел последний актуальный байт и FIFO больше не содержит данных.

Бит **Send** является флагом разрешения отправки байта данных из регистра ToTransmit по каналу UART. При завершении передачи данный бит опустится в «0» автоматически.

Бит **Miss** – флаг пропущенных данных. «1» сигнализирует о том, что FIFO приема было переполнено в процессе получения новых данных и данные утеряны безвозвратно.

Регистр Received

Регистр хранит актуальный принятый байт и сопряжен с FIFO приема, хранящим до 8 принятых байт данных. Иными словами, регистр Received является вершиной FIFO. Каждое чтение этого регистра приводит к удалению самого старого принятого байта данных из FIFO. Чтение данного регистра имеет смысл только при **Rec** = 1 (см. регистр **Status**). Формат регистра описан в таблице 3.

Таблица 3 - поля Received

Разряд	Название	Права записи	Значение по reset	Описание
31:8	Reserved	R	0	Зарезервировано
7:0	Received	R	0	Принятые данные

Регистр ToTransmit

Регистр хранит записанный пользователем байт данных который надлежит передать по каналу UART. Формат регистра описан в таблице 4.

Таблица 4 - поля регистра ToTransmit

Разряд	Название	Права записи	Значение по reset	Описание
31:8	Reserved	R	0	Зарезервировано
7:0	ToTransmit	R/W	0	Данные для передачи

Таймер

Таймер необходим для контроля времени, затраченного на выполнение программы или ее части, путем отсчета тактовых импульсов. Таймер способен вести как инкрементируемый счет тактов, так и декрементируемый. Счет тактов может вестись как по одному, так и по любому другому значению, устанавливаемому делителем программно.

Базовый адрес таймера в системе = 0xAFF1_0000. Регистры таймера описаны в таблице 5.

Таблица 5 – регистры таймера

Смещение	Название	Права записи	Значение по reset	Описание
0x0	Status	R/W	0	Регистр статуса таймера
0x4	TMR	R/W	0	Значение счётчика
0x8	Divider	R/W	0	Регистр делителя счёта

Регистр Status

Регистр служит для управления таймером и описан в таблице 6.

Таблица 6 - поля регистра Status (Таймер)

Разряд	Название	Права записи	Значение по reset	Описание
31:4	Reserved	R	0	Зарезервировано
3	IRQ	R	0	Запрос прерывания
2	IRQen	R/W		Разрешение прерывания
1	Upcount	R/W	0	Разрешение инкрементируемого счёта
0	Downcount	R/W	0	Разрешение декрементируемого счёта

Downcount – бит разрешения декрементируемого счёта. При инициализации программистом в значение «1», значение регистра TMR уменьшается на 1 каждые N тактов, N определяется регистром Divider. Когда значение TMR достигнет 0, бит Downcount сбросится в «0» автоматически. Также этот бит может быть сброшен программно досрочно.

Upcount – бит разрешения инкрементируемого счёта. При инициализации программистом в значение «1», значение регистра TMR увеличивается на 1 каждые N тактов, N определяется регистром Divider. Когда значение TMR достигнет 0xFFFFFFFF, бит Upcount сбросится в «0» автоматически. Также этот бит может быть сброшен программно досрочно. Данный бит игнорируется при Downcount = 1.

IRQen – бит разрешения генерации прерываний.

IRQ - флаг прерывания. Данный сигнал генерируется при IRQen = 1 в случае, если в процессе декрементируемого счета регистр TMR достиг значения 0. В разработанной системе данный флаг подключен к входу флагов прерываний процессора с номером [0].

Регистр TMR – регистр, определяющий начальное значение отсчёта, а также текущее значение счётчика, если таймер ведёт счёт. Данный регистр может быть переписан программистом даже процессе счёта. Формат представлен в таблице 7.

Таблица 7 - поля регистра TMR

Разряд	Название	Права записи	Значение по reset	Описание
31:0	TMR	R/W	0	Значение счётчика

Регистр Divider – регистр-делитель, определяющий количество тактов, при прохождении которого регистр TMR будет изменять свое значение согласно исполняемой задаче. Формат представлен в таблице 8.

Таблица 8 - поля регистра Divider

Разряд	Название	Права записи	Значение по reset	Описание
31:20	Reserved	R	0	Зарезервировано
19:0	Divider	R/W	0	Значение делителя