Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Московский государственный технический университет им. Н. Э. Баумана»

Изучение работы микропроцессорной системы

Методические указания к лабораторным работам по курсу «Электронные устройства и преобразователи элементов и систем»

Составил: доц. Полынков А.В.

Москва 2015 год

Цели работы:

- изучение схемы построения устройства с микроконтроллером;
- изучение функциональных возможностей аппаратно-программного устройства с микроконтроллером;
- ознакомление с основными этапами проектирования устройства с микроконтроллером;
- получение навыков работы по программированию, отладке и работе с электронным устройством с микроконтроллером с использованием интегрированной среды разработки.

Назначение устройства. Электронное устройство с микроконтроллером предназначено для измерения линейных ускорений по трем взаимно перпендикулярным осям и вывода информации в персональный компьютер (ПК) через интерфейс USB.

Состав лабораторной установки:

- Плата электронного устройства с датчиком линейных ускорений **LIS3LV02DQ** (фирма STMicroelectronics), микроконтроллером **AT89C51ED2** (фирма Atmel) и микросхемой преобразователя интерфейса USB-UART **FT232R** (фирма FTDI).
- Программная среда **µVision** фирмы Kiel для разработки и отладки программ для микроконтроллера.
- Программа **FLIP** фирмы Atmel для программирования Flash памяти микроконтроллера AT89C51ED2.

Основные теоретические сведения

1. Функциональная схема устройства

Функциональная схема устройства приведена на рис. 1.

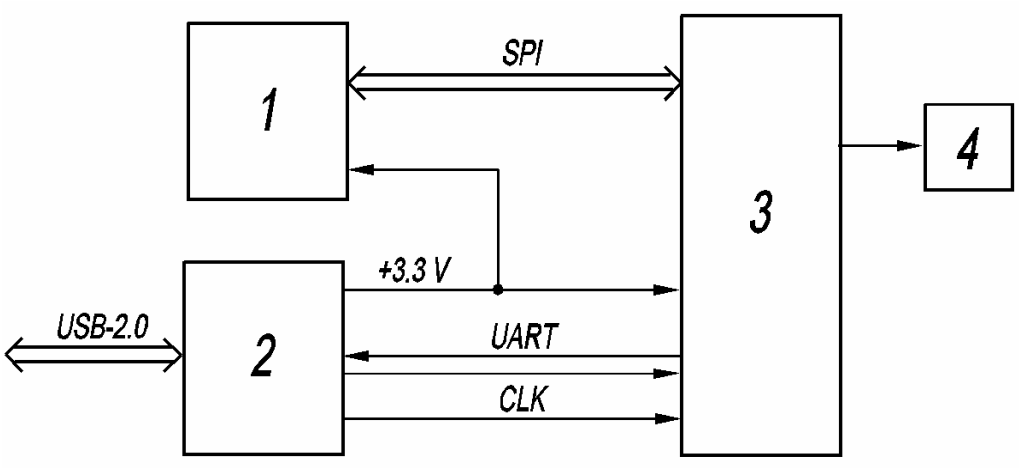

Рис. 1. Функциональная схема устройства

Электронное устройство с микроконтроллером представляет собой функционально связанные между собой микросхемы датчика линейных ускорений 1, преобразователя интерфейса USB-UART 2 и микроконтроллера 3. Для индикации работы устройства используется светодиод 4.

Передача данных между преобразователем интерфейса 2 и микроконтроллером 3 осуществляется по последовательному интерфейсу UART. Управление и съем информации с датчика линейных ускорений 2 осуществляется микроконтроллером 3 через последовательный интерфейс SPI.

Питание преобразователя интерфейса 2 осуществляется непосредственно от шины USB. Питание датчика 1 и микроконтроллера 3 осуществляется от линейного стабилизатора напряжения, входящего в состав преобразователя 2.

2. Основные элементы устройства

2.1. Датчик линейных ускорений

В качестве датчика линейных ускорений используется микросхема LIS3LV02DQ (фирма STMicroelectronics).

Микросхема LIS3LV02DQ представляет собой трех осевой акселерометр, который состоит из чувствительного элемента и схемы измерения отклонения чувствительного элемента от нейтрального положения и интерфейсного блока, передающего по последовательным интерфейсам I2C/SPI измеренное ускорение.

Чувствительный элемент акселерометра выполнен из кремния по технологии МЭМС, разработанной компанией ST.

Интерфейсный элемент выполнен по технологии CMOS, которая позволила получить высокую степень интеграции схемы.

Акселерометры имеют выбираемый пользователем диапазон измерений ±2g и ±6g и способны измерять ускорение по всем осям с полосой пропускания 640Гц. Полоса пропускания может быть выбрана в соответствии с требованиями их применения.

Способность проводить самотестирование, путем формирования электростатического воздействия на чувствительный элемент акселерометра, позволяет пользователю программно проверить работоспособность акселерометра, оценив изменение выходного сигнала при включении самотестирования.

В интерфейсном блоке можно задать пороговое значение для ускорения, превышение которого хотя бы по одной оси вызовет генерацию прерывания акселерометром.

LIS3LV02DQ выпускается в пластиковом корпусе для поверхностного монтажа (SMS корпус) и специфицируются на температурном диапазоне от -40ºC до +80ºC.

LIS3LV02DQ позволяет решать множество задач:

- детектирование свободного падения;
- активация функций в портативных устройствах движением;
- противоугонные системы и инерциальная навигация;
- устройства ввода для игр и виртуальной реальности;
- мониторинг и компенсация вибраций.

Функциональная схема датчика показана на рис. 2.

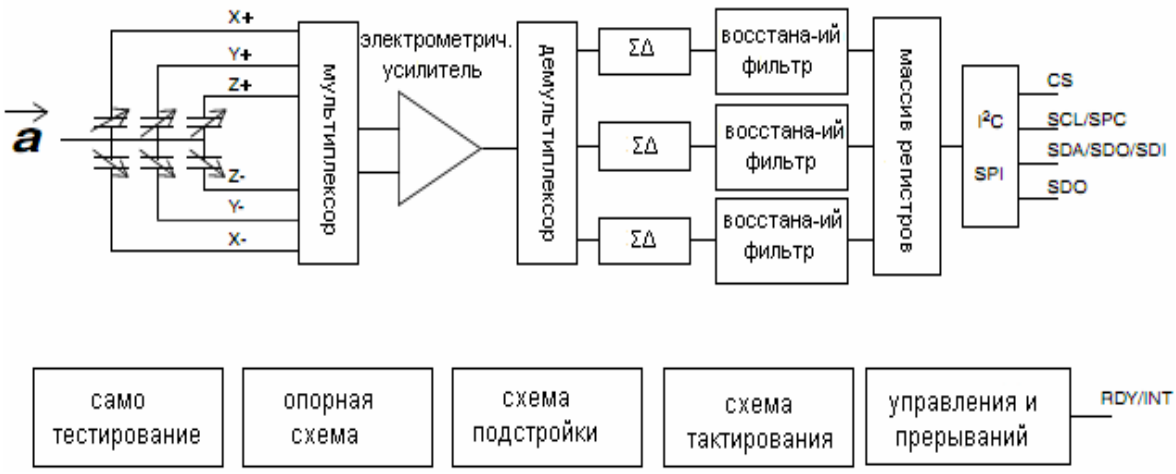

Рис. 2. Функциональная схема датчика линейных ускорений

2.2. Преобразователь интерфейса

В качестве преобразователя интерфейса используется микросхема FT232R (фирма FTDI). Микросхема позволяет осуществить взаимодействие микроконтроллера и ПК через стандартный интерфейс компьютера USB-2.0. При подключении к USB разъему компьютера она

автоматически формирует все необходимые данные для установления связи между ПК и микросхемой. После установки драйверов микросхемы в компьютере формируется дополнительный виртуальный последовательный порт, через который возможен обмен данными между ПК и микроконтроллером. Кроме этого с помощью микросхемы осуществляется тактирование микроконтроллера сигналом CLK частотой 12 МГц. Для питания микроконтроллера и датчика линейных ускорений используется выходной сигнал стабилизатора напряжения +3.3 В.

Функциональная схема преобразователя показана на рис. 3.

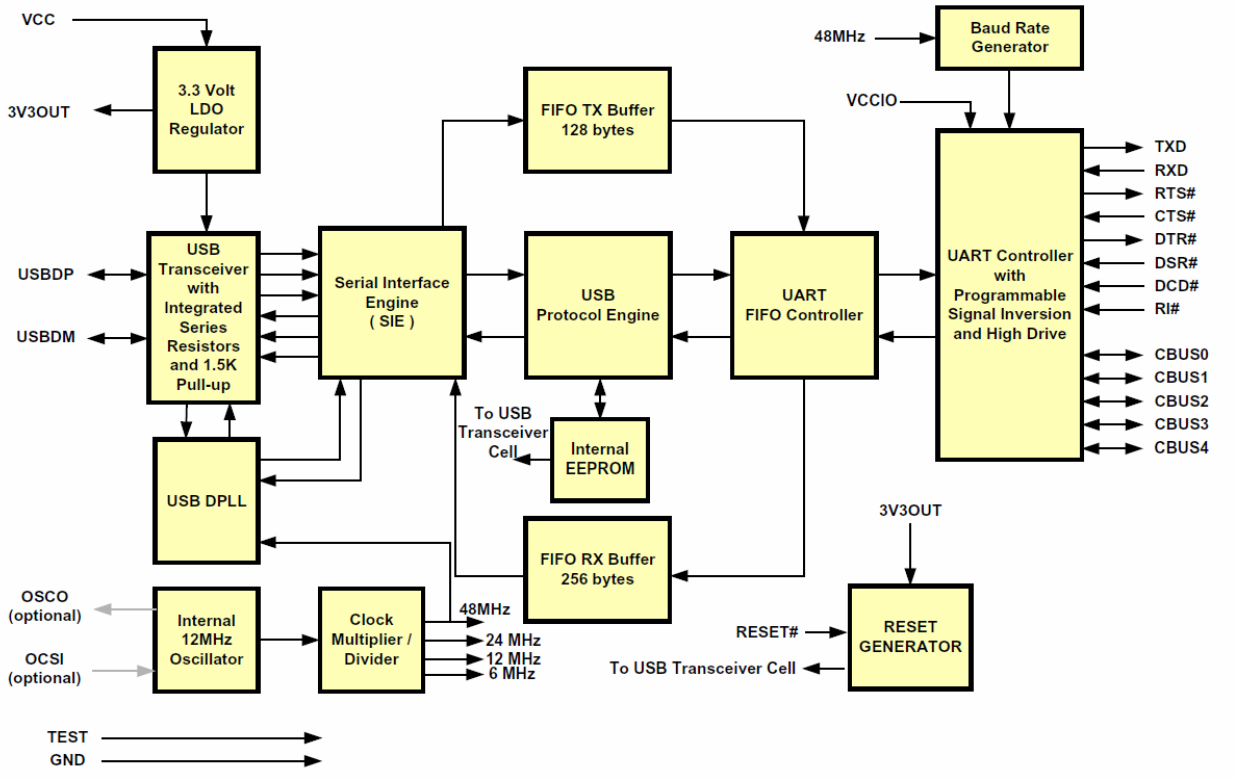

Рис. 3. Функциональная схема преобразователя интерфейса

2.3. Микроконтроллер

В схеме используется микроконтроллер AT89C51ED2 (фирма Atmel). Микроконтроллер относится к семейству MCS-51 и имеет следующие основные характеристики:

- четыре 8-битных порта ввода/вывода (в корпусе с 44 выводами);
- три 16-битных таймера/счетчика;
- 256 байт внутренней основной оперативной памяти (ОЗУ);
- 9 источников прерывания программы с четырьмя уровнями приоритета;
- интегрированный монитор питания (POR/PFD);
- встроенный программатор со стандартным напряжением питания микроконтроллера;
- высокоскоростную архитектуру, позволяющую использовать тактовую частоту 60 МГц (при напряжении питания 5 В и стандартном режиме работы процессора);

- 64 К байт встроенной Flash памяти программ и данных с возможностью побайтной и страничной (128 байт) записи и стирания;
- 1792 байта дополнительного ОЗУ с возможностью программного расширения внутреннего стека;
- 2048 байт внутреннего электрически перепрограммируемого ПЗУ для хранения данных;
- двойной регистр указателя адреса данных;
- интерфейс клавиатуры с формированием прерываний на порте 1;
- SPI интерфейс;
- программируемый массив 16-разрядных счетчиков (PCA);
- полнодуплексный UART со специальным генератором тактовой частоты;
- режим уменьшения электромагнитного излучения;
- режим контроля потребляемой мощности;
- аппаратный сторожевой таймер (Watchdog Timer);
- один источник питания в диапазоне от 2.7 В до 5.5 В;
- температурный диапазон работы от -40 ºС до +85 ºС.

Функциональная схема микроконтроллера показана на рис. 4.

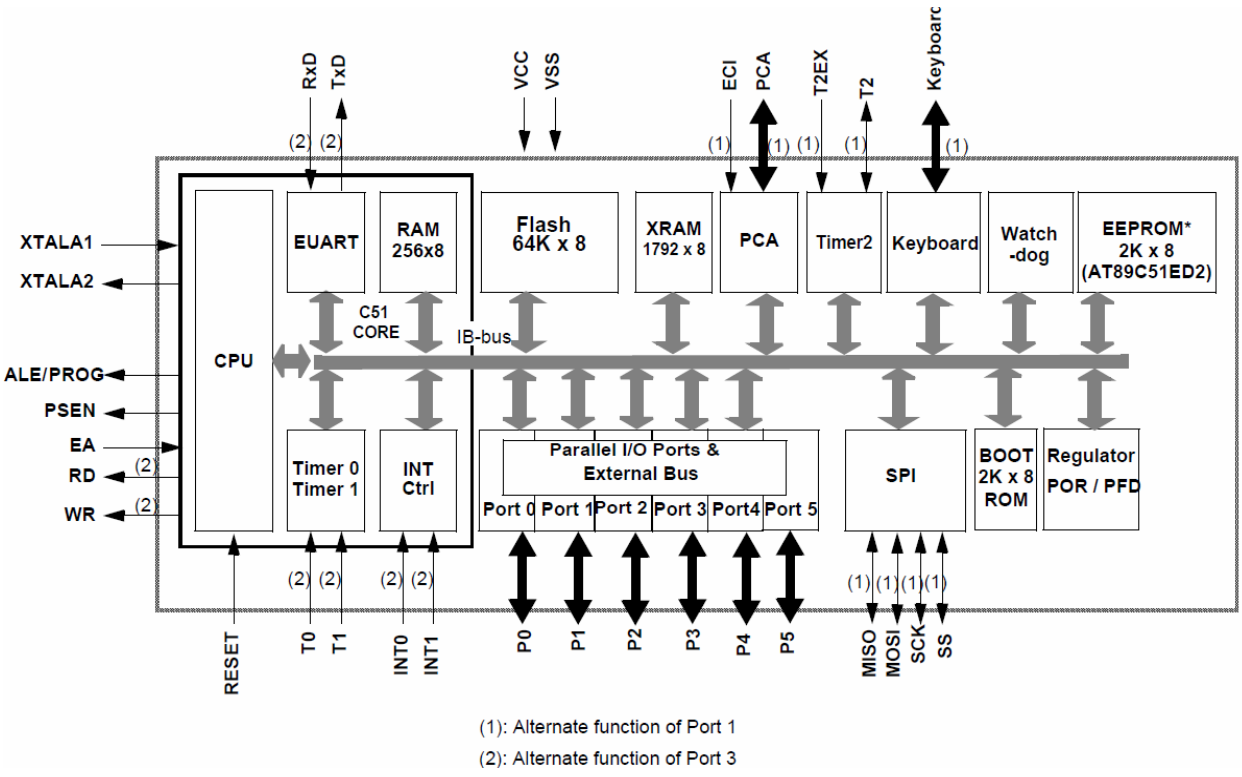

Рис. 4. Функциональная схема микроконтроллера

3. Электрическая схема прибора

Принципиальная электрическая схема устройства показана на рис. 5.

Подключение устройства к компьютеру осуществляется через USB разъем X1. Конденсаторы C1, C2, C3 и дроссель L1 служат для фильтрации сигнала на шине питания USB.

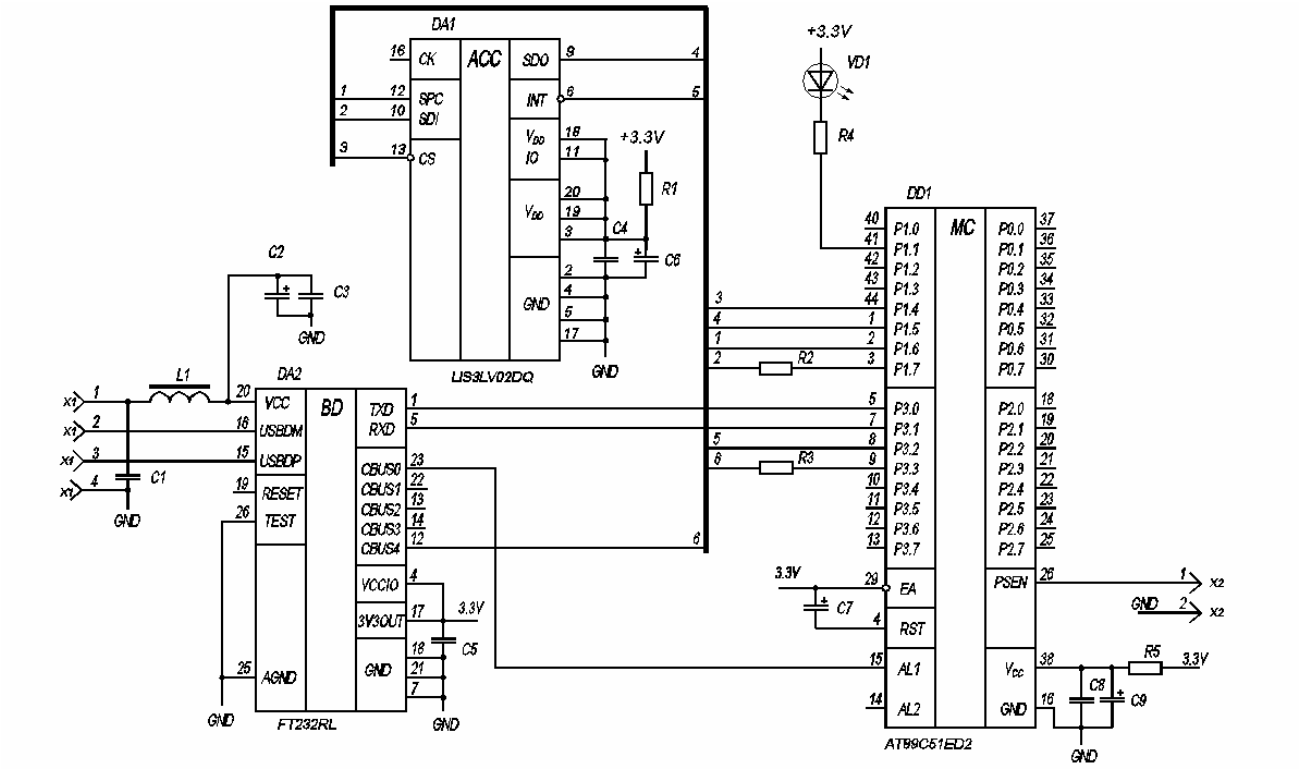

Рис.5. Принципиальная электрическая схема устройства

Разъем X2 используется при программировании Flash-памяти микроконтроллера DD1.

Резистор R1 и конденсаторы C4, C5 служат для фильтрации напряжения питания датчика линейных ускорений DA1.

Резистор R5 и конденсаторы C8, C9 служат для фильтрации напряжения питания микроконтроллера DD1.

Конденсатор C7 формирует сигнал сброса микроконтроллера в момент включения питания.

Резистор R4 ограничивает ток, протекающий через светодиод VD1, который используется для индикации работы устройства.

Резисторы R2, R3 служит для коммутации соответствующих цепей между микросхемами DA1 и DD1.

4. Основные этапы проектирования микропроцессорных систем

При проектировании микропроцессорной системы необходимо решать вопросы организации обмена между устройствами системы и внешней средой, согласования функционирования устройств, имеющих различное быстродействие и принципы передачи данных.

Примерная последовательность этапов, типичных для создания микропроцессорной системы, включает:

Этап 1. Формализация требований к системе. На этом этапе составляются внешние спецификации, перечисляются функции системы, формализуется техническое задание, на систему, формально излагаются принципы работы системы в официальной документации.

Этап 2. Разработка структуры и архитектуры системы. На данном этапе определяются функции отдельных устройств и программных средств, выбирается микропроцессор, на базе которого будет реализована система, определяются взаимодействие между аппаратными и программными средствами, временные характеристики отдельных устройств и программ.

Этап 3. После определения функций, реализуемых аппаратурой, и функций, реализуемых программно, приступают к схемотехнической разработке системы и изготовлению опытного образца, и разработке программных средств.

Разработка и изготовление опытного образца состоят из разработки структурных и принципиальных схем, изготовления прототипа, автономной отладки.

Разработка программ состоит из разработки алгоритмов, написания текста исходных программ, трансляции исходных программ в объектные программы, автономной отладки.

Этап 4. Комплексная отладка и приемосдаточные испытания. Как правило, микропроцессорная система – это система реального времени, т.е. корректность её функционирования зависит от времени выполнения отдельных программ и скорости работы устройств системы. Поэтому система считается отлаженной после того, как рабочие программы правильно функционируют в реальном устройстве в реальных условиях. Дополнительным свойством, которым должны обладать средства комплексной отладки по сравнению со средствами автономной отладки, является возможность управления поведением системы и сбора информации о поведении в реальном времени. Средства отладки не должны влиять на правильность функционирования системы, вносить задержки, дополнительные нагрузки.

Существует пять основных приемов комплексной отладки микропроцессорной системы:

1) останов функционирования системы при возникновении определенного события;

2) чтение (изменение) содержимого памяти или регистров системы;
3) пошаговое отслеживание поведения системы;
4) временное согласование программ.

Комплексная отладка завершается приемосдаточными испытаниями, показывающими соответствие спроектированной системы техническому заданию.

Для облегчения процесса разработки программы используются интегрированные среды программирования. В состав интегрированной среды программирования включается определенный набор программных средств:

- редактор исходного текста;
- трансляторы с выбранного языка программирования;
- редакторы связей;
- загрузчики программного кода;
- отладчики программ и т.д.

Программирование микроконтроллера осуществляется в среде **µVision** фирмы Kiel для разработки и отладки программ для микроконтроллеров на языке ассемблера и С.

4.1. Создание программного проекта в интегрированной среде программирования

Работа с программными проектами начинается с создания нового файла проекта. В большинстве случаев программа состоит из нескольких программных модулей. Принцип разбиения единой программы на программные модули заключается в том, что для реализации работы с отдельными узлами аппаратуры пишутся отдельные подпрограммы, которые относительно слабо связаны с остальными частями программы. Эти подпрограммы можно выделить в отдельную программу, которую можно хранить в отдельном файле, и транслировать отдельно от остальной программы. Часто одни и те же модули могут входить в состав нескольких программ, выполняющих различные задачи, но использующие при этом одни и те же устройства. Часто программа пишется и отлаживается на оценочных платах, а используется в другом устройстве.

Для создания файла проекта в интегрированной среде разработки можно воспользоваться главным меню **(New µVision Project)** (См. рис. 5).

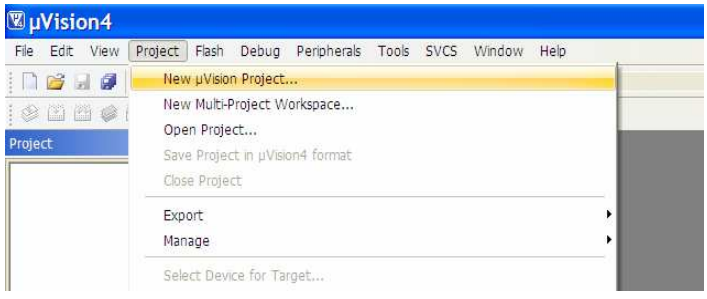

Рис. 5. Создание нового программного проекта

После создания новой директории и нового файла проекта, среда предлагает выбрать конкретную микросхему из семейства **MCS-51**, как показано на рис. 6 и 7.

Рис. 6. Диалоговое окно выбора микросхемы для программного проекта

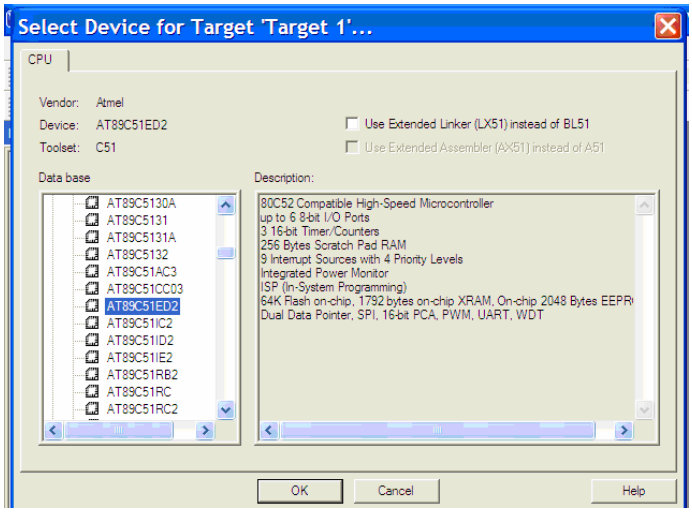

Рис. 7. Диалоговое окно выбора микросхемы для программного проекта

После нажатия кнопки «ОК», будет предложено добавить в рабочую папку и включить в проект файл начальной инициализации микроконтроллера (**Copy ‘STARTUP.A51’ to Project Folder and Add File to Project**). Следует выбрать вариант ответа «Нет». При этом в окно менеджера проекта приобретает вид, показанный на рис. 8.

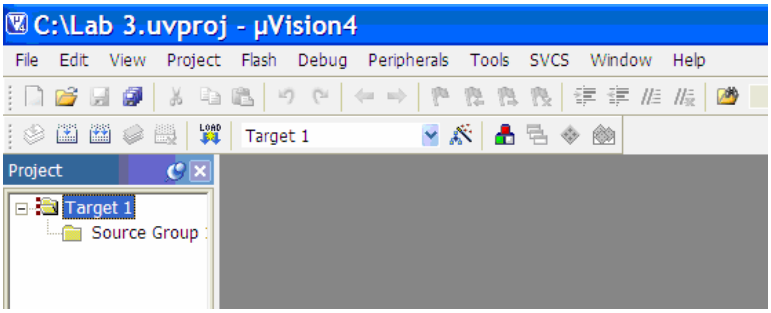

Рис. 8. Внешний вид окна менеджера проекта

Далее требуется настроить параметры программного проекта. Для этого необходимо перейти к изменению свойств проекта, как показано на рис. 9.

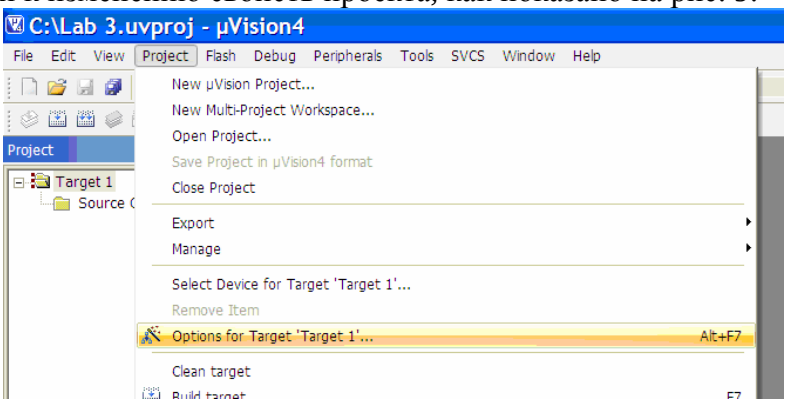

Рис. 9. Изменение свойств проекта

На рис. 10 показан вид диалогового окна изменения свойств проекта.

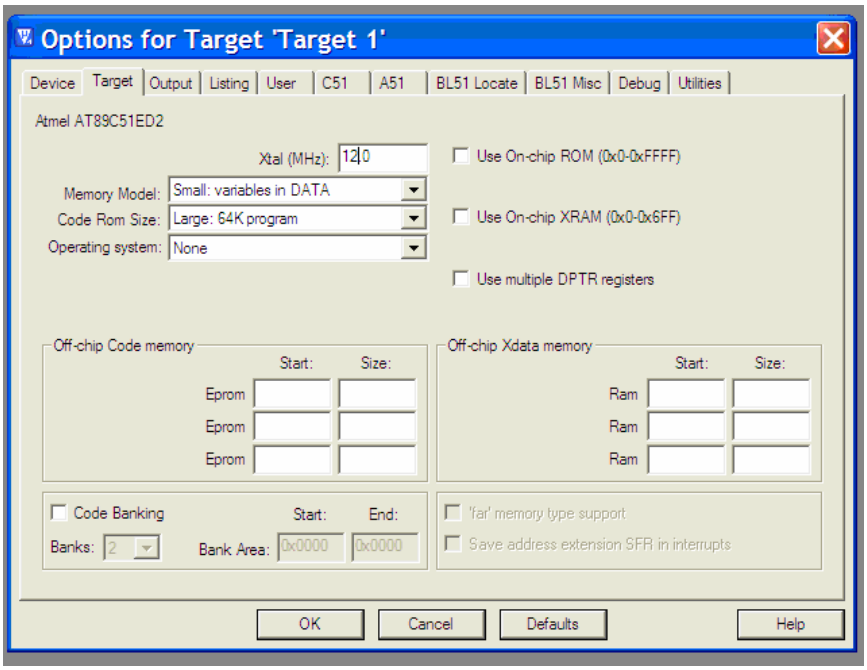

Рис. 10. Диалоговое окно настройки свойств программного проекта

В этом окне необходимо ввести параметры внешней памяти программ и памяти данных.

Затем необходимо установить выходные параметры проекта. Для этого требуется открыть закладку **Output**. Выходной загрузочный файл ***.hex** будет создаваться при выборе соответствующей опции. Для создания файлов объектных кодов и файлов листинга можно создать отдельные папки (**OBJ** и **LST**). Обычно эти файлы располагаются внутри папки проекта.

Для настройки параметров компиляции выбирается закладка **C51** (программы на С) или **A51** (программы на ассемблере). В закладке настраивается уровень оптимизации транслируемого программного модуля и цель оптимизации (скорость исполнения или объём кода).

После настройки свойств проекта в диалоговом окне, это окно закрывается нажатием кнопки **OK**.

Теперь можно подключать к программному проекту файлы с исходным текстом программных модулей. Для этого можно выбрать опцию добавления файлов к проекту в окне менеджера проектов. Если проект состоит из нескольких программных модулей, то опцию добавления файлов нужно повторить нужное число раз.

4.2. Трансляция программ и программных проектов

В программном проекте может быть несколько программных модулей, поэтому имеется возможность нескольких видов трансляции:

- трансляция только одного выбранного файла модуля. Это позволяет обнаруживать и исправлять синтаксические ошибки в отдельном модуле.
- трансляция всего проекта. При этом происходит связывание всех объектных модулей, полученных после трансляции отдельных файлов. При возникновении ошибки транслирования проекта необходимо изменить текст в программном модуле, из-за которого возникает ошибка и заново его транслировать. При отсутствии ошибок создается загрузочный файл.

В составе интегрированной среды программирования для поиска синтаксических ошибок удобнее пользоваться не файлом листинга, а окном **Build** (расположено в нижней части рабочей области), где выводятся сообщения об ошибках. При этом если дважды щёлкнуть мышью по сообщению об ошибке в окне **Build**, то в окне текстового редактора будет выделена строка программы, где была обнаружена данная ошибка. Трансляция программного модуля производится нажатием кнопки **Build target**, как показано на рис. 11.

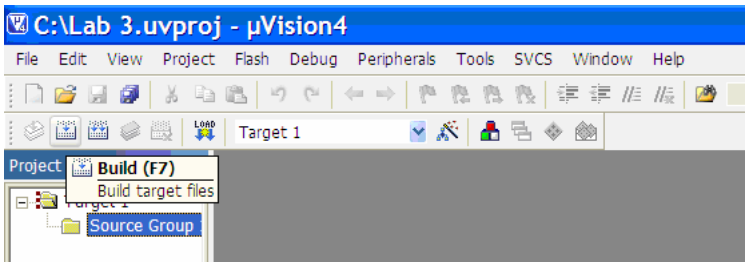

Рис. 11. Трансляция программного модуля проекта

4.3. Отладка программ во встроенном отладчике программ

Отладка программ заключается в проверке правильности работы и устройства. Программа, не содержащая синтаксических ошибок, тем не менее, может содержать логические ошибки, не позволяющие программе выполнять заложенные в ней функции. Логические ошибки могут быть связаны с алгоритмом программы или с неправильным пониманием работы устройств системы, подключенных к портам микроконтроллера.

Для отладки программ обычно применяются три способа:

1) Пошаговая отладка программ с заходом в подпрограммы;

2) Пошаговая отладка программ с выполнением подпрограммы как одного оператора;

3) Выполнение программы до точки останова.

Пошаговая отладка программ заключается в том, что выполняется один оператор программы и, затем контролируются те переменные, на которые должен был воздействовать данный оператор.

Если в программе есть уже отлаженные подпрограммы, то подпрограмму можно рассматривать, как один оператор программы и воспользоваться вторым способом отладки программ.

Использование точек останова позволяет пропускать уже отлаженную часть программы. Точка останова устанавливается в местах, где необходимо проверить содержимое переменных или проконтролировать, передается ли управление данному оператору.

4.4. Последовательные интерфейсы микропроцессорных систем

Последовательный интерфейс использует одну сигнальную линию для передачи данных, по которой биты информации передаются друг за другом последовательно. При последовательной передаче сокращается количество сигнальных линий, что упрощает соединение устройств системы и позволяет сделать интерфейсы помехозащищенными. При последовательной передаче каждый информационный бит должен сопровождаться импульсом синхронизации. Если импульсы синхронизации передаются от одного устройства к другому по выделенной линии, то такой интерфейс является синхронным. Генератор импульсов синхронизации располагается на стороне устройства инициирующего обмен информации – ведущее устройство (Master). Если передача синхроимпульсов между устройствами отсутствует, а передатчик и приемник содержат свой генератор синхроимпульсов, каждый работающий на близких частотах, то такой интерфейс называется асинхронным. В этом случае для синхронизации приема информации требуется передавать дополнительную служебную информацию.

Типичный представитель синхронного последовательного интерфейса – **SPI** (последовательный периферийный интерфейс). Этот интерфейс в работе используется для обмена информацией между микроконтроллером и микросхемой акселерометра. Обмен информацией начинается с началом передачи синхроимпульсов и завершается при передаче определенного числа синхроимпульсов. Чаще всего используется 8-битная посылка и данные представляются в шестнадцатеричном коде (HEX- код). Если необходимо передавать большее число

Типичный представитель асинхронного интерфейса – **UART** (универсальный асинхронный приёмопередатчик). Этот интерфейс в работе используется для организации обмена информацией между микроконтроллером и компьютером. При передаче по интерфейсу каждому байту данных предшествует старт-бит, сигнализирующий приемнику о

начале посылки, за старт-битом следуют биты данных. Завершает посылку стоп-бит, гарантирующий паузу между байтами данных. Старт-бит следующей передачи посылается в любой момент, то есть между передачами возможны паузы произвольной длительности. Внутренний генератор синхроимпульсов приёмника использует момент приёма старт-бита для обнуления счётчика-делителя опорной частоты. Так обеспечивается простой механизм синхронизации приёмника по сигналу передатчика.

Поскольку порт **UART** в асинхронном режиме предназначен для передачи символьной информации, то набор передаваемых символов представляют в виде таблицы кодов, где каждому символу соответствует последовательность бит из одного или двух байт. Наиболее распространенным является код **ASCII** (американский стандартный код для обмена информацией). Кроме информационных символов этот код содержит символы-команды для управления связью. В таблице 1 такие символы показаны двоеточием.

Таблица 1. Символы ASCII

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
1	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
2		!	“	#	$	%	&	‘	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	O
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	…

Таблица 1 содержит 8 столбцов и 16 строк, каждая строка и столбец пронумерованы в шестнадцатеричной системе счисления. Один символ кодируется байтом и в таблице представлено 128 символов. Остальными кодами (128- 255) представляются дополнительные символы (например, символы русского алфавита).

Плата электронного устройства с микроконтроллером **AT89C51ED2** (фирма Atmel) подключается к **USB** порту персонального компьютера через микросхему преобразователя интерфейса USB-UART **FT232R** (фирма FTDI).

5. Задания к работе

5.1. Вывод информации через последовательный асинхронный порт

1. Изучите настройки последовательного порта микроконтроллера **AT89C51ED2**.

2. Разработайте алгоритм программы, передающей через последовательный порт данные на персональный компьютер – согласованное с преподавателем сообщение. Скорость передачи и источник синхронизации UART задается преподавателем.

3. Определите значения регистров специальных функций (SRF), которые позволяют реализовать требуемые параметры передачи.

4. Войдите в интегрированную среду **µVision**. Создайте и настройте должным образом проект.

5. Разработайте и введите текст программы в соответствии с разработанным алгоритмом. При разработке программы можно использовать примеры программ для работы со стандартными устройствами микроконтроллера, которые имеются в составе среды.

6. Оттранслируйте программу и исправьте синтаксические ошибки.

7. Проверьте работу полученной программы в отладчике среды. При необходимости скорректируйте работу программы.

8. Загрузите полученный *.hex файл в плату электронного устройства. Для загрузки файла используется программа **FLIP** фирмы Atmel для программирования Flash памяти микроконтроллеров.

9. Запустите терминальную программу персонального компьютера и убедитесь, что устройство пересылает требуемую информацию.

5.2. Управление микросхемой акселерометра через порт SPI

1. Изучите систему команд для управления микросхемой акселерометра платы электронного устройства.

2. Разработайте алгоритм программы для настройки микросхемы акселерометра с требуемыми параметрами (согласовываются с преподавателем)

3. Определите значения регистров микросхемы акселерометра, которые позволяют реализовать требуемый режим его работы.

4. В интегрированной среде **µVision** создайте файл программы для взаимодействия с микросхемой акселерометра.

5. Отладьте и оттранслируйте программу взаимодействия микроконтроллера с микросхемой акселерометра.

6. В программу передачи данных из микроконтроллера в персональный компьютер внесите изменения, которые позволяют передавать в компьютер данные, полученные из микросхемы акселерометра.

7. Оттранслируйте проект (обе программы) и загрузите полученный *.hex файл в плату электронного устройства.

8. Убедитесь, что программа работает должным образом.

6. Содержание отчета

1. Цель работы.

2. Исходные тексты программ (модулей) в компьютере.

3. Загрузочные файлы, полученные на этапах работы.

4. Расчет значений регистров микроконтроллера и микросхемы акселерометра, реализующих требуемые режимы работы устройства.

5. Выводы по работе.