МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

МОСКОВСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

А.Н. Богаченков

# ЦИФРОВЫЕ УСТРОЙСТВА И МИКРОПРОЦЕССОРЫ

**Лабораторные работы 1, 2, 3**

Методические указания

по выполнению лабораторных работ для студентов, обучающихся

по специальностям и направлениям

210400.62, 210601.65, 210700.62, 211700.62

МОСКВА МИРЭА 2016

Подготовлено на кафедре радиосистем передачи информации Богаченков А.Н.

Цифровые устройства и микропроцессоры. Лабораторные работы 1, 2, 3 : Методические указания по выполнению лабораторных работ. — М.: МИРЭА, 2016.

Методические указания содержат описания 3-х лабораторных работ, в ко- торых изучаются аппаратные и программные средства процессоров с ядром ARM.

Материал предназначен для студентов очной и очно-заочной форм обуче- ния по специальностям и направлениям: 210400.62 «Радиотехника», 210601.65

«Радиоэлектронные системы и комплексы», 210700.62 «Инфокоммуникацион- ные технологии и системы связи», 211700.62 «Конструирование и технология электронных средств».

Материал может быть использован при изучении дисциплин «Цифровые устройства и микропроцессоры», «Цифровые устройства в телекоммуникациях»,

«Микропроцессоры и программируемые логические интегральные схемы»,

«Микропроцессоры и микроЭВМ в конструкции и технологии электронных средств», а также для самостоятельной работы при освоении базового курса ка- федры.

Электронный ресурс, редакция 09.2016

ФГБОУ ВО «МИРЭА»

119454, Москва, пр. Вернадского, д. 78

© МИРЭА, 2016

## Лабораторная работа № 1

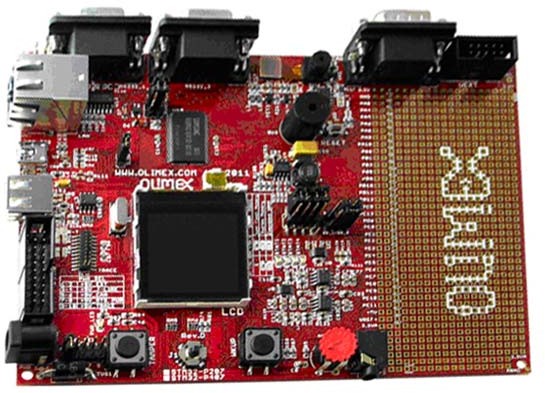
### Общее знакомство с процессорами с ядром ARM Cortex-M4

**и средой разработки программного обеспечения**

Целью лабораторной работы является ознакомление: с основными характеристика- ми процессоров, использующих ядро ARM Cortex-M4, со средой разработки программного обеспечения, с форматами и диапазонами обрабатываемых чисел, особенностями выпол- нения операций на языке Си, формами представления выходной информации. Проверка работы производится в режиме симуляции и на отладочной плате.

## Лабораторный макет

Внешний вид отладочной платы и основные компоненты, задействованные при вы- полнении лабораторного практикума, показаны на рисунке:



USB-интерфейс для загрузки

рабочей программы (используется также для питания платы)

Интерфейс

USB OTG

Интерфейсы RS-232 Интерфейс CAN

Соединитель расши- рения UEXT

Звуковой излучатель

(зуммер)

Кнопка сброса

(RESET)

4 индикатора состояний

Отладочный

интерфейс JTAG

Графический дисплей 132x132 точек

Кнопка (TAMPER)

общего назначения

Джойстик

(4 позиции +

кнопка выбора)

Кнопка (WAKEUP)

общего назначения,

выход из польз. программы

Аудиокодек с усилителями

Регулятор аналогового уровня (триммер)

Двухканальный аналоговый выход с аудиокодека

Функциональная схема макета и его описание приведены в учебном пособии «Про-

цессоры с ядром ARM в лабораторном практикуме и курсовом проектировании» [4].

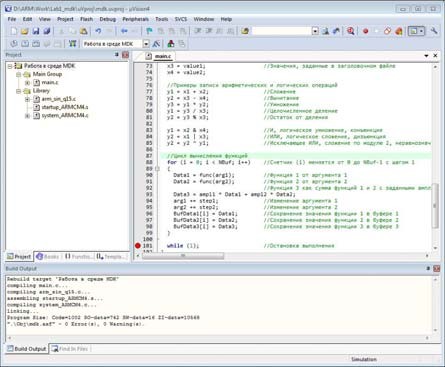
## Среда разработки

Для разработки программ для процессоров с ядром ARM используется интегриро- ванная среда MDK-ARM (µVision 4 или µVision 5) компании Keil (www.keil.com). Про- граммный пакет имеет в составе базу данных о всех выпущенных процессорах с данным ядром, систему управления созданием проектов, интеллектуальный редактор для подго- товки исходных текстов, библиотеки функций (математических, ввода-вывода, обработки сигналов и др.), компилятор-компоновщик программ на языках C/C++ и ассемблера, от- ладчик-симулятор, отладчик с использованием внутрисхемного эмулятора, расширения для оптимизации процесса проектирования, быстрого создания устройств с различными интерфейсами. Свободно распространяемая версия, не требующая лицензирования, имеет ограничение на объем скомпилированного кода — 32 килобайта.

Пользовательский интерфейс программного пакета MDK-ARM имеет различный вид в зависимости от режима работы: производится работа с файлами проекта или отладка. Вид по умолчанию показан ниже. При отладке может быть открыто большое число допол- нительных окон: несколько областей памяти, стек, локальные переменные, графический

анализатор, регистры ядра, регистры всех периферийных устройств, консольный вывод и др.

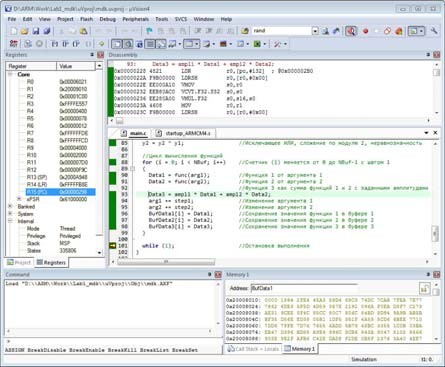
Режим подготовки проекта Режим отладки



**Окно проекта**

**Окно редактора исходных текстов**

**Результаты компиляции**



**Дизассемблер**

**Операци- онные регистры**

**Исходный текст**

**Команды, сообщения**

**Переменные, память**

Основные действия при работе в среде:

|  |  |  |  |
| --- | --- | --- | --- |
| **Операция** | **«Горячая»**  **клавиша** | **Кнопка** | **Меню** |
| Компиляция проекта | **F7** |  | Project/Build |
| Настройки проекта | – |  | Project/Options for Target... |
| Переход в режим отладки, возврат из режима отладки | **Ctrl+F5** |  | Debug/Start/  Stop Debug Session |
| Вставка/удаление точки останова | **F9** |  | Debug/Insert/Remove Breakpoint |
| Шаговое выполнение с заходом в вызываемые функции | **F11** |  | Debug/Step |
| Шаговое выполнение без захода в вызываемые функции | **F10** |  | Debug/Step Over |
| Непрерывное выполнение | **F5** |  | Debug/Run |
| Остановка выполнения | – |  | Debug/Stop |

## Загрузка и выполнение рабочей программы

После сброса процессор выполняет находящуюся во Flash-памяти специальную программу-загрузчик, позволяющую по USB интерфейсу загрузить из компьютера в опе- ративную память процессора рабочую программу пользователя. После компиляции проек- та в среде MDK-ARM и отсутствии ошибок автоматически запускается утилита ARM Loader, при нажатии в ее окне кнопки "Load & Run" машинный код передается в отладоч- ную плату и запускается на выполнение.

**Внимание! Для перезапуска** рабочей программы (после модификации и компиля- ции) необходимо **осуществить сброс кнопкой WAKEUP или RESET и затем нажать "Load & Run"**. При возникновении различных сбоев нажать RESET.

### Порядок выполнения

**Перед началом работы выполнить** процедуру очистки рабочего каталога, исполь- зуя значок на рабочем столе **"Инициализация ARM".** Запустить среду разработки знач- ком **"Keil uVision"**.

## Ознакомление со средой разработки, структурой проекта, операторами языка Си

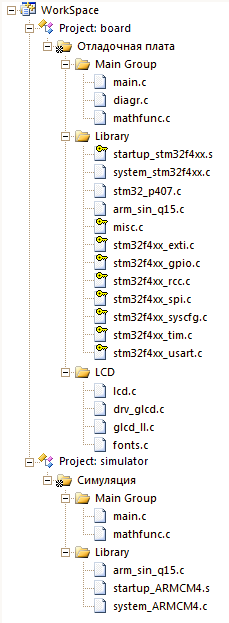
Файл одиночного проекта имеет расширение \***.uvproj** или \***.uvprojx**, мульти- проекта — \***.uvmpw**, открывается выбором пункта основного меню **Project/Open project**. Как правило, в проект входит большое число модулей, которые для удобства разбивают на группы. Проект может содержать следующие типы файлов:

|  |  |
| --- | --- |
| \*.**c**,  \*.**cpp** | Исходные тексты программы на языке C/C++, в лабораторном практикуме  достаточным является использование языка C. |
| \*.**h**,  \*.**inc** | Заголовочные (подключаемые) файлы, содержащие объявления общих типов данных, констант, функций. Эти файлы автоматически подключаются к про-  екту при наличии ссылок на них в исходных файлах (директива #include). |
| \*.**s**, \*.**a**,  \*.**asm** | Исходные тексты на языке ассемблера. |
| \*.**obj,\*.o** | Ранее откомпилированные модули (объектные модули). |
| \*.**lib** | Файлы стандартных или пользовательских библиотек. |
| \*.**txt** | Текстовые файлы для различных пояснений (компилятором игнорируются). |

С помощью пункта меню **Project/Open Project...** открыть проект

#### D:\ARM\Work\Lab1\_diagr\MDK‐ARM\multiproject.uvmpw

Просмотреть структуру проекта и входящие в него модули. Проект представляет собой так называемый мульти-проект: в од- ной рабочей области (Workspace) содержится в данном случае два проекта. Один – для реальной отладочной платы (board), другой – для симуляции (simulator).



В режиме симуляции работа процессора (фактически толь- ко его ядра) моделируется компьютерной программой. Для этой цели используется некий виртуальный процессор, который в настройках проекта обозначен как “Cortex-M4 FPU” (FPU – блок вычислений с плавающей точкой).

Проект кроме основного пользовательского файла **main.c** содержит несколько стандартных (библиотечных), из которых, как правило, всегда присутствуют следующие два:

**startup... .s** — файл с таблицей векторов прерываний, наиболее важным в этой таблице с меткой Vectors являются первые две записи, которые содержат адрес для регистра- указателя стека (SP) и начальный адрес программы, загружаемый в программный счетчик (PC).

**system. c** — файл, содержащий инициализационные дей-

ствия, производимые перед началом выполнения пользователь- ской программы, например, инициализацию тактового генератора процессора (после сброса он работает на минимальной частоте).

В состав проекта для отладочной платы входят библиотеч- ные файлы для программирования периферийных устройств, например:

**stm32\_p407.c** — подпрограммы для работы с компонентами отладочной платы

(кнопками, индикаторами и др.);

**misc.c** — обслуживание векторной системы прерываний ядра; **stm32f4xx\_exti.c** — конфигурирование прерываний от линий ввода-вывода; **stm32f4xx\_gpio.c** — программирование портов ввода-вывода; **stm32f4xx\_rcc.c** — конфигурирование системы синхронизации;

**lcd.c** — обслуживание графического дисплея и др.

В файле **main.c** имеется функция main(), с которой всегда начинается выполнение пользовательской программы. Так как файл является общим для обоих проектов, то разли- чия в работе учитываются использованием макроопределения SIMUL (макроопределение задано только в настройках проекта для симуляции).

Для переключения между проектами необходимо в окне проекта правой кнопкой мыши выделить строку **Project: board** или **Project: simulator**, выбрать пункт **Set as Active Project**.

Активизировать проект **simulator**, откомпилировать, перейти в режим отладки, **вы- полнить по шагам** арифметические и логические операции, несколько циклов вычисле- ния функций, наблюдая результат в:

* окне Call Stack – Locals;
* наводя курсор на имена переменных в исходном тексте.

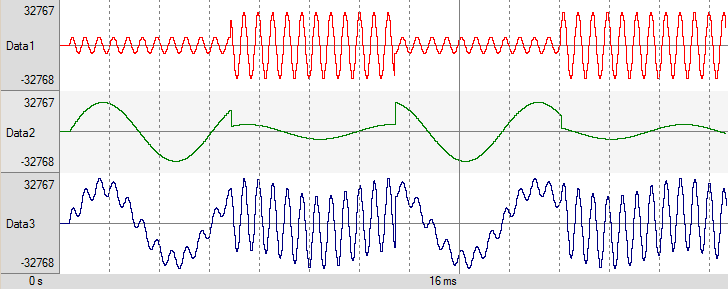
Сопоставить исполняемые операции на языке высокого уровня и соответствующие им команды ассемблера (дизассемблированный текст отображается в окне “**Disassembly**”). Заполнить и **привести в отчете** следующую таблицу, указав команды только самих ариф- метических и логических операций. Учесть, что один оператор языка Си обычно трансли- руется в несколько команд, среди которых, например — LDR (загрузка из памяти), STR (за- пись в память), MOV (пересылки), SXT (изменение разрядности), также учесть, что мнемони- ки команд могут иметь дополнительные суффиксы (например, S).

|  |  |  |
| --- | --- | --- |
| **Операция** | **Язык C** | **Команда ассемблера** |
| Сложение | **+** | ADDS |
| Вычитание | **-** |  |
| Умножение | **\*** |  |
| Целочисленное деление | **/** |  |
| Остаток от деления | **%** |  |
| Инверсия | **~** |  |
| И (логическое умножение) | **&** |  |
| ИЛИ (логическое сложение) | **|** |  |
| Исключающее ИЛИ | **^** |  |

Выполнить построение графиков функций в следующем порядке (изложен также в виде комментария в файле main.c):

* Откомпилировать проект (F7).
* Инициировать режим отладки (Ctrl+F5).
* Поставить точку остановки на последний оператор – while(1) – должна присут- ствовать красная метка слева.
* Открыть логический анализатор: меню View / Analysis Windows / Logic Analyzer.
* Если левое поле анализатора пусто, в тексте программы последовательно выде- лить имена Data1, Data2, Data3 и перетащить их на это поле анализатора.
* Запустить программу на выполнение (F5).
* После остановки отрегулировать масштаб диаграмм полем Zoom (In/Out/All).
* Возвратиться из отладочного режима (Ctrl+F5).

Пример вывода с исходными значениями амплитуд сигналов:



Переключиться на проект **Project: board**. Проверить работу программы на реаль- ном устройстве (откомпилировать проект, в открывшемся окне ARM Loader нажать кноп- ку "Load & Run"). Рекомендуется дальнейшие действия производить также на макете.

Изменить амплитуды суммируемых функций, чтобы максимальное значение ам- плитуды было более единицы: ampl1 + ampl2 = 1.05...1.5.

**Привести в отчете** полученные диаграммы, объяснить возникшие «артефакты».

Не изменяя значений скорректированных амплитуд, реализовать режим ограниче- ния (насыщения), используя средства, заложенные в ядре Cortex-M4 — ассемблерную ин- струкцию SSAT или Си-функцию ssat. Определение данной функции:

int ssat(int val, unsigned int sat);

где val — значение, к которому применяется операция насыщения; sat — число бит кода

‐2

...2

(от 1 до 32); функция возвращает val, ограниченное диапазоном

sat‐1

sat‐1 ‐ 1.

**Привести в отчете** диаграмму в режиме насыщения, скорректированный текст программы.

## Построение графиков функций стандартной библиотеки Си

Скорректировать проект предыдущего пункта для отладочной платы, осуществить построение графика функции по одному из следующих заданий (номер варианта соответ- ствует номеру рабочего места или согласуется с преподавателем). Использовать то же са- мое число точек (NBuf), обеспечить масштабирование кривой на всё поле диаграммы, вы- вод названия функции. Не использовать насыщение. Если функция в каких-то точках стремится к бесконечности, ограничить (или обнулить) ее значения в некоторой окрестно- сти аргумента. **Привести в отчете** скорректированные фрагменты программы, скриншоты дисплея.

|  |  |
| --- | --- |
| Вариант |  |
| 2.1 | erf – функция ошибок. Диапазон изменения аргумента -2…2. |
| 2.2 | Секанс (выразить через cos). Диапазон изменения аргумента -…+, особые точки: -/2, +/2. |
| 2.3 | expm1 – смещенная экспонента. Диапазон изменения аргумента -3…3. |
| 2.4 | Функция sin(x)/x. Диапазон изменения аргумента -10…10. Реализовать кор- ректное поведение вблизи 0. |
| 2.5 | tan – тангенс. Диапазон изменения аргумента -…+, особые точки: -/2, +/2. |
| 2.6 | cosh – гиперболический косинус. Диапазон изменения аргумента -5…5. |
| 2.7 | Котангенс (выразить через sin, cos). Диапазон изменения аргумента  -3/4…+3/4, особая точка: 0. |
| 2.8 | fmod(x,2) – дробная часть. Диапазон изменения аргумента -10…10. |
| 2.9 | log10 – десятичный логарифм. Диапазон изменения аргумента 0…10, особая точка: 0. |

|  |  |
| --- | --- |
| 2.10 | Коcеканс (выразить через sin). Диапазон изменения аргумента -3/4…+3/4,  особая точка: 0. |
| 2.11 | acosh – ареакосинус. Диапазон изменения аргумента 0…10. Функция опреде- лена для аргумента > 1. |
| 2.12 | lgamma – натуральный логарифм от абсолютного значения гамма-функции. Диапазон изменения аргумента -10…10. |
| 2.13 | 3 – кубический синус. Построить 2-3 периода.  sin |
| 2.14 | cbrt – кубический корень. Диапазон изменения аргумента -100…1000. |
| 2.15 | logb – показатель аргумента. Диапазон изменения аргумента -10…10, особая точка: 0. |
| 2.16 | tgamma – гамма-функция. Диапазон изменения аргумента -0.01…0.01. |

## Индивидуальное программирование вычислительных операций

Выбрать вариант по номеру рабочего места или получить от преподавателя (при необходимости согласовать также исходные данные).

Для всех параметров (кроме констант) определить переменные, числовые значения задать в программе компактным блоком. Учесть, что при вычислениях необходимо ис- пользовать систему единиц СИ. В заданиях с вариацией исходных значений не дублиро- вать расчетную формулу, а включить ее в цикл или в функцию. Осуществить вывод на дисплей исходных данных и конечных результатов.

В качестве шаблона использовать следующий проект (убедиться в работе демон- страционного примера, удалить/заменить строки, помеченные как Тест, добавить свои операторы):

#### D:\ARM\Work\Lab1\_indiv\MDK‐ARM\individ.uvmpw

Продемонстрировать результаты работы преподавателю. **Привести в отчете** фор- мулы, исходные значения, полученные результаты (в виде снимка дисплея), текст основ- ной программы.

* 1. Рассчитать 3 точки оконной функции Барлетта-Ханна:

 

*w**n*  *a*0

* *a*1

 *n*  0.5  *a*

*N* 1 2

 cos 2  *n* 

*N* 1

 

для *a*0  0.62;

*a*1  0.48;

*a*2  0.38; N = 100; n = 10, 50 и 90.

рями:

* 1. Рассчитать резонансную частоту параллельного колебательного контура с поте-

1

*L C*  *R*2

*L C* 

*f*   ,

2 *LC*

при *L* = 15 мкГн, *С* = 22 пФ, *R* = 0.5 кОм и 10 Ом. Вывести значение частоты с единицей измерения без использования степенного множителя в двух форматах: с максимальной точностью и с округлением до 3-4 знаков.

* 1. Рассчитать полный импеданс последовательной RLC-цепи:

*Z*  ,

*R* 2  *L*  



1 2



*C* 

где   2

*f* ; *f* = 10 и 50 кГц; *L* = 2 мГн; *С* = 330 нФ; *R* = 200 Ом.

* 1. Определить корни квадратного уравнения 0.1234*x*2 + 5.6710-3*x* + 8.910-5 .

*x*  .

* *b*  *b*2  4*ac*

2*a*

Проверить (программными средствами): если уравнение не имеет вещественных корней, изменить знак одного из коэффициентов, на дисплей вывести также результаты проверки.

* 1. Найти коэффициент передачи активного фильтра верхних частот на двух часто-

тах , отстоящих на ±25% от частоты среза 0  1 *RC* :

*K* 2*R*2*C* 2

*k*  ,

1  *CR*2 2  3  *K* 2 *CR*2

где *K* = 2 (коэффициент усиления); *С* = 22 нФ; *R* = 1.5 кОм.

* 1. Вычислить значение функции sin по приведенной ниже приближенной формуле и с использованием библиотечной функции для значения аргумента, соответствующего 30º (перевести в радианы).

sin *x*  *x*  *x* 

3

3!

*x x*7

.

5



5! 7!

* 1. Найти значение автокорреляционной функции для  = 0.5 мс :

( ) 

1 *N* 1 *f* *T*

*i*   *f* *T i*

  

*f* (*t*)  sin 2 *t* 



*i* 0 

*N*



  

  

*N*

*N*

, где

 

 *T* 

, *T* = 1 мс, N = 50.

* 1. Рассчитать емкость двухпроводной линии длиной *l* = 1.5 км, расстояние между проводниками 5 мм, радиус проводников 0.2 и 0.6 мм, диэлектрическая проницаемость ва- куума  = 8.8510-12 Ф/м, относительная диэлектрическая проницаемость = 2.5 :

0

*C*  2  0   *l* .

ln*d r* 

* 1. Рассчитать энергию излучения (формула Планка):

*u**T*

 2  *h*  *c* 2

5

 1 ,

exp*hc* *k**T*   1

где *с* = 2.998108 м/с — скорость света; *k* = 1.3810-23 Дж/К — постоянная Больцмана; *h* = 6.62610-34 Джс — постоянная Планка;  = 700 нм — длина волны (для красного света); *T* = 273 и 373 K — абсолютная температура.

* 1. Рассчитать три точки вольтамперной характеристики p-n перехода для напря- жений *u* = –0.5, 0 и +0.5 В:

*i*  *iS*

 exp *u*

 *T*





1



, где

  *kT*

*T q*

— температурный потенциал;

*is* = 1 мкА — ток насыщения; *T* = 280 K — абсолютная температура; *k* = 1.3810-23 Дж/К —

постоянная Больцмана; *q* = 1.610-19 Кл — элементарный заряд.

* 1. Найти три точки Гауссовой функции для *µ* = -0.5, 2 = 2 и вариации *x* = –1.5,

–0.5, 0.5:



*f*  1





exp

 2



*x*   2 

 2 2  ,

* 1. Рассчитать три точки Гауссова радиоимпульса:

*y*  exp  *t* 2 cos2 *f t*  ,

*C*

где  = 4.5 — коэффициент длительности; *f С* = 20 МГц — частота несущей; *t* = –50 нс, 50 нс, 200 нс — временные отсчеты.

* 1. Вычислить значение экспоненциальной функции, аппроксимированной рядом Маклорена, и той же функции с использованием стандартной библиотеки для аргумента x = 3.3:

*x* 3 *xn*

*e*   .

*n*!

*n*0

* 1. Найти коэффициенты взаимной индуктивности двух антенных катушек с па- раметрами: числа витков *N*1,2 = 5 и 2; радиусы витков *R*1,2 = 55 и 33 мм; расстояния между катушками *x* = 1, 5, 20 мм; µ0 = 4 10-6 Вс / Ам — магнитная постоянная:

  *N*  *R*2  *N*  *R*2 

*M*  0 1 1 2 2 .



*R*  *x* 

2 2

3

2

2

* 1. Определить коэффициент ослабления электромагнитной волны при прохожде- нии через слой земли толщиной *d* = 5 и 15 м:

 *X*  *B* 

1

 2

1  



  109 2

*B*





*k*  *e* *pd* , где

*p*    

1 ;

 2  

  

*X* = 0.810-16 2 *f* ; *B* = 0.556   *f* ;  = 310-2 См/м — проводимость земли;  = 5 — диэлек- трическая проницаемость земли; *f* = 10 МГц — частота волны.

* 1. Найти значения тока транзистора, соответствующие фазам 0º, 20º, 90º, 180º, 320º входного колебания, при аппроксимации переходной вольтамперной характеристики кусочно-линейной функцией:

 0, при *U* 0  *Um*  cos  *t*   *UH*

*i*   

*U*  *U*  ,

 *S* *U* cos  *t*   *H* 0  ,



0 *m*

*H*

при *U*  *U*  cos  *t*   *U*

 *m* 

 

*Um* 

где *U*0 = 2.5 В — постоянная составляющая входного колебания; *Um* = 4 В — амплитуда входного колебания; *UH* = 1.5 В — напряжение отсечки; *S* = 90 мА/В — крутизна харак- теристики.

**Содержание отчета**

Таблицы, диаграммы/скриншоты с заголовками и объяснениями возникающих особенностей (указаны в соответствующих пунктах выполнения работы).

Полный исходный текст программы с внесенными коррекциями (п. 1, 2), коррекции рекомендуется выделить, например, другим шрифтом или вручную.

Текст основного модуля и результаты работы программы по индивидуальному за- данию (п. 3).

Внимание! При оформлении текстов программ применять только моноширинные шрифты (Courier New, Consolas, Lucida Console и т.п.). Исключить автопереносы строк. При необходимости использовать альбомную ориентацию страницы. Образец оформления

* в файлах с текстами программ.

### Контрольные вопросы

1. Перечислите возможности среды разработки программного обеспечения.
2. Как осуществляется загрузка программы в реальный процессор?
3. Какие файлы и с какой целью входят в состав проекта?
4. Как задается начало рабочей программы? Какие дополнительные действия осуществ- ляются перед ее стартом?
5. Что такое режим симуляции, какие действия возможны в этом режиме?
6. Расскажите о порядке графической визуализации данных и сигналов.
7. Каковы причины нарушения «плавности» изменения сигналов при увеличении их ам- плитуды?
8. Какие форматы данных обрабатываются процессором с максимальной производитель- ностью? Как осуществляется работа с другими типами данных?
9. Каковы причины возникновения некорректных результатов при арифметических опе- рациях? Какие рекомендации по их исправлению?
10. Когда используют программирование на языках высокого уровня и когда на ассембле- ре?
11. Дайте определение основным характеристикам процессора.
12. Какие виды памяти и какого объема встроены в процессорный кристалл, какую память можно подключить дополнительно, каков может быть ее объем?
13. Как распределяется адресное пространство процессора?
14. Охарактеризуйте внутренние регистры процессора (количество, разрядность, назначе- ние).
15. Какие основные группы команд используются в процессорах с ядром ARM?
16. Дайте характеристику основных компонентов отладочной платы.

### Библиографический список

* 1. Новожилов О.П. Основы микропроцессорной техники. В 2 т.— М.: ИП Радио- софт, 2007. Т. 1. — 432 с.
  2. Джозеф Ю. Ядро Cortex-M3 компании ARM. Полное руководство / Джозеф Ю; пер. с англ. А.В.Евстифеева. — М.: Додэка-XXI, 2012. — 552 с.
  3. Trevor Martin. The Insider's Guide to the STM32 ARM-based Microcontroller // Hitex (UK) Ltd (www.hitex.com), 2009. Перевод: Ознакомительное руководство по ARM- микроконтроллерам Cortex-M3 [Электронный ресурс]. — Режим доступа: [http://www.gaw.ru/html.cgi/txt/doc/micros/arm/cortex\_arh/index.htm.](http://www.gaw.ru/html.cgi/txt/doc/micros/arm/cortex_arh/index.htm)
  4. Богаченков А.Н. Процессоры с ядром ARM в лабораторном практикуме и кур- совом проектировании [Электронный ресурс] : Учебное пособие по выполнению лабора- торных и курсовых работ — М.: МИРЭА, 2016.
  5. STM32-P407 Development board. User Manual (на англ. языке) [Электронный ре- сурс], 2012. — Режим доступа: [https://www.olim](http://www.olimex.com/Products/ARM/ST/STM32-P407/)ex.com[/Products/ARM/ST/STM32-P407/.](http://www.olimex.com/Products/ARM/ST/STM32-P407/)
  6. Библиотека цифровой обработки сигналов и стандартной периферии: <http://www.st.com/content/st_com/en/products/embedded-software/mcus-embedded-> software/stm32-embedded-software/stm32-standard-peripheral-libraries/stsw-stm32065.html
  7. Иванов Р. Лидер по производительности среди ядер Cortex-M4 – STM32F4xx.
* Новости электроники, 2012, № 2, с. 17-22.
  1. Бугаев В., Мусиенко М., Крайнык Я. Визуализация возможностей: графический генератор кода STM32CubeMX — Новости электроники, 2014, № 11, с. 19-25.
  2. Керниган Б., Ритчи Д. Язык программирования C. : Пер. с англ. — М.: Изда- тельский дом "Вильямс", 2009. — 304 с.

## Лабораторная работа № 2

### Процессоры с ядром ARM Cortex-M4.

**Параллельные порты. Таймеры.**

Целью лабораторной работы является освоение работы с параллельными портами ввода-вывода общего назначения, таймерами, ознакомление с системой прерываний. Все задания выполняются в среде MDK-ARM (µVision 5) с практической проверкой на отла- дочной плате STM32-P407. Функциональная схема макета:

**+**

**+**

**+**

+3.3 B

**TRIMMER**

**TAMPER WAKEUP**

Джойстик

**LEFT RIGHT UP DOWN**

**SEL**

**CPU STM32F407**

**PC13 PA0**



**PG11 PG6 PG7 PG8**

**PG15**

**PC0**

**PF6 PF7 PF8 PF9**

**PB1**

**PA5 PC3 PD6 PD3 PB0**

**LED1** (зел.)

**LED2** (желт.) **LED3** (красн.) **LED4** (зел.)

**BUZZER**



(зуммер)

Графический дисплей

**SCK DIO CS RESET LED**

Интерфейс

RS-232

Аудиокодек

**RXD TXD CTS RTS**

Преобразо- ватель уровня

**PD8 PD9 PD12 PD11**

**PB3 PB5 PA15 PC7**

**SCLK SDIN LRCK MCLK**

Аудиовыход

Вход АЦП

**...**

(24 канала)

**PA4 PA5**

Выход ЦАП

(2 канала)

### Порядок выполнения

**Перед началом работы выполнить** процедуру очистки рабочего каталога, исполь- зуя значок на рабочем столе **"Инициализация ARM".** Запустить среду разработки знач- ком **"Keil uVision5"** .

**Напоминание**. Для загрузки рабочей программы в отладочную плату после успеш- ной компиляции проекта используется кнопка "Load & Run" в отдельном окне ARM Loader. Перед новой загрузкой необходимо завершить рабочую программу, как правило, кнопкой WAKEUP отладочной платы, или, в крайнем случае, кнопкой RESET.

## Ввод-вывод логических сигналов через параллельные порты

Изучить функциональную схему лабораторного макета (см. выше). Открыть проект

#### D:\ARM\Work\Lab2\_gpio\MDK‐ARM\gpio.uvproj

В проекте реализованы: ввод состояний джойстика и кнопок, вывод этих состояний на светодиодные индикаторы и звукоизлучатель. Используются библиотечные подпро- граммы временных задержек, построенных на основе подсчета пустых циклов.

Программирование портов, операции ввода-вывода осуществляются путем прямого обращения к регистрам периферийных устройств. Это способствует получению компакт- ного программного кода, но требует изучения технических руководств по форматам дан- ных в используемых регистрах. Другие подробности проекта описаны в виде комментария в основном файле main.c.

Откомпилировать, загрузить программу в отладочную плату, проверить работу, **привести в отчете** следующую таблицу, описав реакцию выходных устройств. При дина- мической реакции (мигание, переключение, звук) указать точные временные или частот- ные параметры (период/частота мигания/переключения, скважность и др.), определив их из текста программы и сопоставив с визуальным восприятием.

|  |  |  |
| --- | --- | --- |
| Воздействие на основной орган управления | Воздействие на дополнительный орган управления | Поведение выходных устройств с указанием временных или частотных параметров |
| Позиционирование джойстика | нет |  |
| Кнопка TAMPER |  |
| Кнопка выбора джойстика | нет | Яркость 1 Яркость 2 Время включ. состояния: ……. мс ……. мс Время выключ. состояния: ……. мс ……. мс  Период попарного переключения: ……. мс Период смены яркости: ……. мс |
| Кнопка TAMPER |  |

Для лучшего понимания действий программы рекомендуется внести некоторые коррекции: например, изменить соответствие позиций джойстика определенным индика- торам, изменить временные задержки. Оформление результатов этих действий не требует- ся. **В отчете привести** исходный текст программы.

## Использование таймеров для генерации временных интервалов и сигналов с широтно-импульсной модуляцией

Открыть проект

#### D:\ARM\Work\Lab2\_tim\_pwm\MDK‐ARM\timer.uvproj

В проекте продемонстрированы некоторые режимы работы таймеров. Два таймера с номерами TIM10, TIM11 сконфигурированы на генерацию сигналов ШИМ, выходы каналов сравнения таймеров непосредственно подключены к двум светодиодным индикаторам (LED1, LED2). Таймер TIM4 работает в базовом режиме, генерируя прерывания с заданной периодичностью, в подпрограмме же обслуживания этого прерывания осуществляется программное управление третьим светодиодным индикатором (LED3).

Запустить программу на выполнение. Наблюдая работу индикаторов, изобразить

примерные временные диаграммы их переключения.

LED1

(зел.,

левый)

LED2

(желт.)

LED3

(красн.)

LED4

(зел.,

правый)

Из текста программы определить точные параметры генерируемых сигналов:

Таймер 10 (LED1). Период ... Коэффициент длительности ... Таймер 11 (LED2). Период ... Коэффициент длительности ... Таймер 4 (LED3). Период ... Коэффициент длительности ...

Не трогая управление индикатором LED3, добавить независимый канал управления индикатором LED4. Для этого использовать систему прерываний и свободный таймер с номером 2, 3, 5 или 7 (все указанные таймеры тактируются половинной системной часто- той).

Скорректировать программу в соответствии с одним из вариантов, приведенных да- лее в таблице. Продемонстрировать работу преподавателю.

На тех же или других диаграммах отобразить изменения. **Привести в отчете** диа- граммы, данные об исходных и измененных параметрах, исходные и модифицированные фрагменты программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Вариант | LED1 | | LED2 | | LED3 | LED4 |
| Период | Коэф.длит. | Период | Коэф.длит. | Период | Частота |
| 2.1 | 100 мс | 3 % | 3.2 с | 41 % | 543 мс | 2.1 Гц |
| 2.2 | 200 мс | 5 % | 3.0 с | 51 % | 654 мс | 3.2 Гц |
| 2.3 | 300 мс | 15 % | 2.8 с | 61 % | 765 мс | 4.3 Гц |
| 2.4 | 400 мс | 20 % | 2.6 с | 71 % | 865 мс | 5.4 Гц |
| 2.5 | 500 мс | 25 % | 2.4 с | 81 % | 987 мс | 6.5 Гц |
| 2.6 | 600 мс | 30 % | 2.2 с | 54 % | 432 мс | 7.6 Гц |
| 2.7 | 700 мс | 35 % | 1.95 с | 64 % | 321 мс | 8.7 Гц |
| 2.8 | 800 мс | 40 % | 1.85 с | 84 % | 210 мс | 9.8 Гц |
| 2.9 | 900 мс | 45 % | 1.65 с | 94 % | 109 мс | 10.9 Гц |
| 2.10 | 999 мс | 50 % | 1.45 с | 26 % | 98 мс | 40 Гц |
| 2.11 | 1.1 с | 55 % | 1.25 с | 21 % | 87 мс | 34.5 Гц |
| 2.12 | 1.2 с | 60 % | 1.15 с | 16 % | 76 мс | 32.1 Гц |
| 2.13 | 1.3 с | 65 % | 777 мс | 12 % | 65 мс | 23.4 Гц |
| 2.14 | 1.4 с | 70 % | 555 мс | 8 % | 54 мс | 20.2 Гц |
| 2.15 | 1.5 с | 75 % | 333 мс | 6 % | 43 мс | 16.6 Гц |
| 2.16 | 1.6 с | 80 % | 222 мс | 4 % | 32 мс | 13.3 Гц |

## Использование таймеров для генерации тональных сигналов

Открыть проект

#### D:\ARM\Work\Lab2\_tim\_ton\MDK‐ARM\timer.uvproj

В проекте задействован таймер TIM3, который настроен на генерацию импульсов с коэффициентом длительности 50 % (или скважностью 2) и частотой, изменяемой при воз- действии на органы управления (джойстик, кнопки). Сигнал напрямую передается звуко- вому излучателю. Генерируемые частоты соответствуют музыкальным тонам. **Привести в отчете** описание результата воздействия на каждый орган управления — длительность и частоту генерируемого сигнала.

В качестве простейших упражнений предлагается поменять тона, установленные по умолчанию, сгенерировать тона из более низких или высоких октав. На следующем шаге выполнить одно из следующих заданий:

* 1. При каждом нажатии на одну и ту же кнопку (джойстик) последовательно вы- давать различные тона, составляющие заданную в виде массива кодов музыкальную фразу.
  2. При однократном воздействии на орган управления сформировать заданную массивом кодов последовательность звуковых колебаний (режим музыкального звонка).
  3. Сформировать плавно увеличивающийся и уменьшающийся по частоте сигнал

(режим сирены).

**Привести в отчете** текст модифицированной программы.

## Использование таймеров для измерения временных интервалов

Открыть проект

#### D:\ARM\Work\Lab2\_tim\_sys\MDK‐ARM\timer.uvproj

Для отсчета времени выполнения фрагментов программы используется 24- разрядный системный таймер ядра ARM Cortex, работающий на той же частоте (168 МГц). Метки времени фиксируются функциями TimeMarkerX(). По нажатию кнопки TAMPER запомненные значения счетного регистра таймера выводятся на дисплей. Для повышения точности измеряемый фрагмент включен в цикл. **Привести в отчете** скриншот вывода на дисплей, таблицу с расчетными временами выполнения математических функций. При за- щите иметь текст исходной программы.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Функция | | Число тактов таймера для  цикла вычислений | Время всего цикла вычисления функции, мкс | Время однократного вычисления функции, мкс |
| 1. | … |  |  |  |
| 2. | … |
| … |  |

**Содержание отчета**

Таблицы, диаграммы, пояснения, исходные тексты программ со всеми произведен- ными коррекциями.

Внимание! При оформлении текстов программ применять только моноширинные шрифты (Courier New, Consolas, Lucida Console и т.п.). Исключить автопереносы строк. При необходимости использовать альбомную ориентацию страницы. Образец оформления

* в файлах с текстами программ.

### Контрольные вопросы

1. Изобразите на функциональной схеме лабораторного макета подтягивающие резисто- ры. Каково их назначение?
2. Для чего нужны резисторы, включенные последовательно со светодиодными индика- торами?
3. Какие параметры задаются при конфигурировании порта общего назначения? Воз- можная ли одновременная работа частей порта на ввод и вывод?
4. Опишите регистры, используемые при работе с портами и их отдельными разрядами.
5. Насколько точными и почему являются временные задержки на основе циклических операций?
6. Перечислите все возможные режимы работы таймера.
7. Какие параметры задаются при конфигурировании таймера для работы в базовом ре- жиме?
8. Какие параметры задаются при конфигурировании таймера для генерации ШИМ сиг- налов?
9. Каков предельный диапазон генерируемых частот и временных задержек в 16-

разрядном и 32-разрядном таймерах?

1. Опишите механизм измерения временных интервалов посредством таймеров. Каковы минимальный и максимальный интервал для системного таймера ядра? Какова по- грешность таких измерений и что ее обуславливает?
2. Как измерить длительность и период внешних сигналов?
3. Как осуществить амплитудную и частотную манипуляцию сигнала посредством тай- меров?
4. В программе п. 1 заменить библиотечные подпрограммы временных задержек на ос- нове циклических операций задержками, реализованными с помощью таймеров.
5. Дайте определения понятиям: запрос прерывания, вектор прерывания, обработчик прерывания.
6. Как осуществляется конфигурирование прерываний?

### Библиографический список

См. описание лабораторной работы № 1.

## Лабораторная работа № 3

### Процессоры с ядром ARM Cortex-M4.

**Аналого-цифровые периферийные устройства.**

Целью лабораторной работы является освоение работы с аналого-цифровой пери- ферией: АЦП, ЦАП, аудиокодеком, а также изучение принципов передачи данных по си- стеме прерываний и прямого доступа к памяти. Все задания выполняются в среде MDK- ARM (µVision 5) с практической проверкой на отладочной плате STM32-P407.

### Порядок выполнения

**Перед началом работы выполнить** процедуру очистки рабочего каталога, исполь- зуя значок на рабочем столе **"Инициализация ARM".** Запустить среду разработки знач- ком **"Keil uVision5"**.

**Напоминание**. Для загрузки рабочей программы в отладочную плату после успеш- ной компиляции проекта используется кнопка "Load & Run" в отдельном окне ARM Loader. Перед новой загрузкой необходимо завершить рабочую программу, как правило, кнопкой WAKEUP отладочной платы, или, в крайнем случае, кнопкой RESET.

## Измерения с помощью аналого-цифрового преобразователя

Открыть проект

#### D:\ARM\Work\Lab3\_adc\MDK‐ARM\adc.uvproj

В проекте демонстрируются: конфигурирование АЦП, диапазоны цифровых дан- ных в смещенном и дополнительном кодах, особенности оцифровки при наличии помех, режим "оконного" аналогового компаратора (или аналоговой "сторожевой схемы").

Исследование работы АЦП осуществляется в "статическом" режиме — для медлен- но меняющейся (практически постоянной) амплитуды сигнала, формируемой установлен- ным на плате потенциометром (триммером). Регулятор выдает положительное напряжение в пределах 0…3.3 В (в реальном устройстве при прохождении цепей, например с раздели- тельными конденсаторами/трансформаторами такой сигнал будет знакопеременным). Ре- зультаты измерений выводятся на графический дисплей отладочной платы в десятичном формате. Переключение режима усреднений производится джойстиком.

Наличие внутренних помех, нестабильностей при аналого-цифровом преобразова- нии, плохого качества опорного напряжения, внешних флюктуаций во входном сигнале и других факторов обуславливают нестабильность цифрового кода — так называемый шум младших разрядов. Оценить шум можно по величине изменения кода в течение некоторого времени (например, нескольких секунд). Произвести такую оценку для 3-4-х различных входных уровней, поместив в табл. 1 некоторое усредненное значение. Рекомендуется найти диапазон изменения кода в режиме выравнивания вправо, разность кодов перевести в двоичную систему, число значащих двоичных разрядов и будет соответствовать числу "шумовых". **Привести** таблицу в отчете.

Таблица 1. Оценка шумов при преобразовании

|  |  |  |
| --- | --- | --- |
|  | Диапазон изменения кода | Число "шумовых"  двоичных разрядов |
| Без усреднения |  |  |
| С усреднением |  |  |

Проверить работу АЦП во всем диапазоне уровней входного напряжения с вклю- ченным или выключенным режимом усреднений (для чтения стабильных показаний мож- но использовать одну из позиций джойстика, временно останавливающую измерения). Об- ратить внимание на поведение красного индикатора. В табл. 2 и 3 занести только коды в 10-ной системе — соответствующие ячейки обведены жирной рамкой, остальные парамет- ры рассчитать при домашней подготовке. Три строки с номерами 1, 5, 9 (первую, сред- нюю, последнюю), если не удается ручным регулятором установить заданный уровень, за- полнить теоретически. Уровень в строке 2 взять близким к уровню в строке 1, в строках 4 и 6 — близким к уровню в строке 5, в строке 8 — близким к уровню в строке 9. Одно- именные строки в таблицах должны соответствовать одному и тому же уровню (т.е. долж- ны заполняться одновременно).

Таблица 2. Напряжения и коды АЦП, выравненные влево

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | UВХ, В | UАЦП, В | Смещенный код | | Дополнительный код | |
| 10-чный | двоичный (16 бит) | 10-чный | двоичный (16 бит) |
| 1 |  | 3.3 | (макс.) |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  | 0 |  |
| 6 |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 9 |  | 0 | (мин.) |  |  |  |

Таблица 3. Напряжения и коды АЦП, выравненные вправо

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | UВХ, В | UАЦП, В | Смещенный код | | Дополнительный код | |
| 10-чный | двоичный (16 бит) | 10-чный | двоичный (16 бит) |
| 1 |  | 3.3 | 4095 |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  | 0 |  |
| 6 |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |
| 9 |  | 0 | 0 |  |  |  |

Различие напряжений UАЦП и UВХ (для АЦП), а также UЦАП и UВЫХ (для ЦАП) ил- люстрируется следующей схемой:



UВХ

UАЦП

АЦП



ЦАП

UЦАП

UВЫХ

Встроенные АЦП и ЦАП имеют соответственно входной и выходной диапазоны напряжений от 0 до опорного напряжения Vref. Для внешних устройств, подключаемых через разделительные конденсаторы, напряжения из-за отсутствия постоянной составляю-

щей будут знакопеременными. Рабочий режим АЦП при наличии разделительного кон- денсатора необходимо обеспечить дополнительно подачей на вход постоянного смещения.

**Привести в отчете** результаты измерений, формулы для расчета напряжений по одному из кодов.

Модифицировать программу для отображения на дисплее реальных уровней напряжения в вольтах, снимаемых с потенциометра и одного из внутренних источников (опорного или батарейного) или температурного датчика. Добиться, чтобы красный инди- катор сигнализировал об отклонении напряжения более, чем, например, 2.5 В ± 10% (чис- ло рекомендуется согласовать с преподавателем). Продемонстрировать работу, **привести в отчете** скорректированный текст программы.

## Вывод сигналов с использованием ЦАП и аудиокодека

Открыть проект

#### D:\ARM\Work\Lab3\_dac\MDK‐ARM\dac.uvproj

В проекте показан пример вывода периодических сигналов, заданных табличным способом, через одноканальный ЦАП и двухканальный аудиокодек. Особенность основ- ной программы состоит в том, что после проведения инициализации в ней не производит- ся никаких операций, процессорное ядро переводится в состояние "сна". Все действия — передача отсчетов сигнала в ЦАП, передача отсчетов в интерфейс аудиокодека, реакция на нажатие кнопок — осуществляются по системе прерываний. При возникновении соответ- ствующих запросов процессор пробуждается, выполняет подпрограмму обработки преры- вания и возвращается в спящий режим. Предусмотрена возможность еще большей раз- грузки процессора — передача отсчетов в режиме прямого доступа к памяти (DMA).

В предыдущей лабораторной работе был продемонстрирован способ измерения времени выполнения фрагментов программы посредством таймера. Здесь представлен другой способ — регистрация с помощью осциллографа специально сформированных сиг- налов-меток. В частности, при передаче отсчета в ЦАП осуществляется вывод сигнала лог. "1" в начале обработчика прерывания и вывод лог. "0" по завершении обслуживания. Дли- тельность импульса позволяет оценить время активной работы процессора, пауза между импульсами — время бездействия. По этим данным легко определяется коэффициент за- грузки процессора (обычно в %). Также подобные сигналы-метки очень полезны для целей отладки, позволяя отследить прохождение определенных участков программы.

Проверить работу устройства, измерить с помощью осциллографа периоды сигна- лов на выходе аудиокодека. Осциллограммы **привести в отчете**, указав на них значения периодов.

Если имеется возможность регистрации сигналов с выхода ЦАП и отладочного сиг- нала-метки, произвести определение их периодов (а для сигнала-метки — и длительности). При отсутствии такой возможности воспользоваться иллюстрациями в файле с исходными текстами программ. **Привести в отчете** диаграммы сигналов с ЦАП и отладочного сигна- ла с необходимыми временными параметрами. При защите уметь объяснить различие сиг- налов ЦАП и кодека: по форме кривой, амплитуде, периоду. Определить коэффициент за- грузки процессора, **привести в отчете** соответствующие расчеты.

Если есть возможность наблюдения сигналов на выходе ЦАП, убрать символы комментария с определения "#define DMA\_ENA", перекомпилировать программу, прове- рить работу. **Указать в отчете** на изменения в сигналах, при невозможности эксперимен- тальной оценки сделать это теоретически.

Модифицировать программу для получения одновременно на выходе ЦАП и двух выходах аудиокодека трех разных сигналов (их форма должна соответствовать исходным сигналам). Продемонстрировать работу, **привести в отчете** скорректированный текст про- граммы.

* 1. **Дискретизация сигналов с помощью АЦП** Открыть проект **D:\ARM\Work\Lab3\_discret\MDK‐ARM\discret.uvproj**

В проекте реализована система ввода-вывода аналогового сигнала с потенциальной его цифровой обработкой. Исходный сигнал частотой Fсигн генерируется самим же про- цессором с помощью ЦАП, выход которого соединен с входом АЦП. Выходной сигнал формируется посредством аудиокодека. Изначально частота дискретизации АЦП выбрана значительно большей допустимой частоты дискретизации по теореме Котельникова.

Генератор сигналов

Вход Выход

DAC1

ADC1 ADC2

Обра- ботчик

Кодек

По осциллографу на выходе кодека определить частоту сигнала Fсигн, из програм- мы или посредством тестовых выходных сигналов-меток — частоту дискретизации АЦП (Fдискр). **Привести в отчете** осциллограммы сигналов при следующих частотах дискре- тизации, указав при этом реальную частоту наблюдаемого выходного сигнала:

* Fдискр >> Fсигн (исходный вариант);
* Fдискр = (2.1 – 2.3) Fсигн;
* Fдискр = (1.2 – 1.5) Fсигн.

Осуществить простейшую цифровую обработку сигнала только в одном из каналов

(если осциллограф позволяет регистрировать каналы раздельно).

*Обязательные варианты*:

а) произвести однополупериодное и двухполупериодное выпрямление (детектиро-

вание);

б) осуществить двухстороннее ограничение сигнала (сверху и снизу) на уровне

40…80% от максимальной амплитуды;

*Варианты, согласуемые с преподавателем*:

в) реализовать систему шумоподавления (обнулить сигнал уровня менее 5-20%);

г) осуществить амплитудную манипуляцию сигнала с некоторым программно за- данным периодом (в 5…10 раз большим периода сигнала);

д) осуществить умножение частоты входного сигнала в 2 или 4 раза;

е) измерить амплитуду (пиковое значение) входного сигнала, вывести результат на дисплей или включить индикатор при превышении некоторого уровня;

ж) реализовать систему автоматической регулировки — обеспечить одинаковую амплитуду выходного сигнала при различных амплитудах входного.

Продемонстрировать работу, **привести в отчете** диаграммы выходных сигналов до и после обработки, скорректированный текст программы.

**Содержание отчета**

Тексты заданий, таблицы, диаграммы, тексты программ (указаны в соответствую- щих пунктах выполнения работы)

Внимание! При оформлении текстов программ применять только моноширинные шрифты (Courier New, Consolas, Lucida Console и т.п.). Исключить автопереносы строк. При необходимости использовать альбомную ориентацию страницы.

### Контрольные вопросы

1. Перечислите основные характеристики АЦП. Какие параметры задаются при конфи- гурировании АЦП?
2. Опишите функционирование и приведите передаточную характеристику стандартного аналогового компаратора. Какими дополнительными свойствами обладает "оконный" аналоговый компаратор ("аналоговая сторожевая схема")?
3. Перечислите основные характеристики ЦАП. Какие параметры задаются при конфи- гурировании ЦАП?
4. Перечислите основные характеристики аудиокодека. Какие параметры задаются при его конфигурировании?
5. Чем обусловлено различие форм кривых одного и того же сигнала на выходе ЦАП и выходе аудиокодека?
6. Какими рекомендациями следует руководствоваться при выборе частоты дискретиза- ции и разрядности АЦП-ЦАП?
7. Покажите все необходимые компоненты на входе АЦП для работы со знакоперемен- ными сигналами, сигналами с большим диапазоном напряжений.
8. Расскажите о правилах подключения, алгоритме работы при наличии многоканальных сигналов.
9. Дайте определения понятиям: запрос прерывания, вектор прерывания, обработчик прерывания. Как осуществляется конфигурирование прерываний?
10. Опишите подробно процесс перехода на обслуживание прерывания и возврат из него. Как происходит обслуживание одновременно нескольких запросов?
11. Что такое прямой доступ к памяти? Какие модули могут работать в этом режиме? Как происходит работа одновременно нескольких каналов прямого доступа по общим ши- нам?
12. Сравните коэффициент загрузки процессора, например, в задачах генерации сигналов: а) без использования системы прерываний и DMA; б) при использовании системы прерываний; в) при использовании системы DMA.
13. Как обеспечить непрерывное аналого-цифровое и цифро-аналоговое преобразование, а также цифровую обработку нескольких сигналов одним процессорным ядром?

### Библиографический список

См. описание лабораторной работы № 1.