МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

Разработка КИХ-фильтра в среде Visual DSP++

Электронный лабораторный практикум

САМАРА

2012

Составители: **Корнилин Дмитрий Владимирович,**
Кудрявцев Илья Александрович

Разработка КИХ-фильтра в среде Visual DSP++ [Электронный ресурс] : электрон. лаб. практикум / Минобрнауки России, Самар. гос. аэрокосм. ун-т им. С. П. Королева (нац. исслед. ун-т); сост. Д. В. Корнилин, И. А. Кудрявцев. - Электрон. текстовые и граф. дан. (0,27 Мбайт). - Самара, 2012. - 1 эл. опт. диск (CD-ROM).

В лабораторном практикуме рассмотрены вопросы составления и отладки программы, реализующей КИХ-фильтр в среде Visual DSP++ на языке ассемблера и Си для процессоров цифровой обработки сигналов. Описаны приемы работы со средой программирования Visual DSP++. Лабораторный практикум обеспечивает подготовку по дисциплине «Методы цифровой обработки сигналов в радиотехнических системах» образовательных программ подготовки магистров по специальности 210400.68 "Радиотехника", реализуемых на радиотехническом факультете. Курс 5, 6, семестр А, В.

Лабораторный практикум разработан на кафедре радиотехнических устройств.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Процессоры Blackfin фирмы Analog Devices являются одними из самых производительных в своем классе. Высокая тактовая частота (до 750МГц), возможность выполнения параллельных операций, мощная поддержка ПДП обеспечивают высокую эффективность применения данных процессоров для реализации скоростных алгоритмов обработки сигналов в режиме реального времени.

16-ти и 32-разрядные операции, развитая система команд делают эффективным применение С-компиляторов для разработки программного обеспечения, что также сокращает затраты времени программиста на реализацию целого ряда рутинных операций. Вместе с тем для создания особенно критичных участков кода и эффективного использования архитектурных возможностей процессора программист должен владеть системой команд на уровне языка ассемблера.

При разработке сложных алгоритмов программист вынужден затрачивать значительные усилия на реализацию рутинных операций, например, организацию операций ввода-вывода, манипуляции с памятью, обработку прерываний и т.п., что требует длительного времени. Между тем, современные процессоры обработки сигналов обладают возможностью сократить «посторонние» усилия разработчика за счет использования операционной системы, автоматически решающей указанные задачи, и позволяющей сосредоточиться на разработке главного алгоритма, реализация которого составляет смысл проекта. Применительно к процессорам Blackfin такой операционной системой является VDK (Visual DSP Kernel), тесно интегрированная в одноименную среду разработки.

Процессор обладает целым рядом встроенных средств, облегчающих отладку программного обеспечения, в том числе режим эмуляции на базе интерфейса JTAG, что позволяет сократить сроки разработки конечного устройства при условии грамотного использования разработчиком имеющихся средств САПР.

Для создания окончательного варианта устройства на базе DSP Blackfin от разработчика требуются довольно значительные усилия по разработке схемотехнической и конструкторской частей проекта, причем стоимость и сроки разработки конструкции, включая печатную плату, могут быть весьма значительными. Сократить указанные затраты, распараллелить работу над проектом, а также снизить риск неоправданных потерь времени и средств на переработку проекта может использование отладочной платы типа EZ-LITE, на которой установлен процессор и наиболее популярные периферийные устройства. При использовании подобных средств «макетную часть» проекта удается провести без длительной и дорогостоящей стадии разработки (и изготовления) топологии печатной платы.

Методические указания позволяют студентам изучить основы программирования процессоров Blackfin в среде разработки Visual DSP++ с использованием такой отладочной платы и операционной системы VDK. Указания не претендуют на полноту описания особенностей процессора,

приводятся лишь краткие пояснения, необходимые для понимания приведенных фрагментов.

1 Создание проекта

Среда программирования Visual DSP предусматривает несколько режимов работы, включая работу с аппаратными отладчиками и симуляцию проекта. При запуске среды программирования обычно загружается последний проект в том режиме, с которым велась работа. При появлении диалогового окна, предлагающего выбрать режим работы, необходимо выбрать режим симуляции и тип процессора – ADSP-BF533. Впоследствии изменить режим работы достаточно легко с помощью переключения сессии (меню «Session»).

Для создания нового проекта целесообразно воспользоваться предлагаемым мастером создания проекта «Project wizard», вызвать который можно через меню «File/New/Project». В появившемся диалоговом окне укажите имя проекта (латинскими символами), путь к директории проекта и его тип. В качестве типа укажите «Standard Application». Попутно отметим, что можно также выбрать «Multi-threaded application using VDK», что понадобится нам в дальнейшем.

Далее щелкните левой кнопкой мыши по значку «Output type» и в появившемся окне выберите процессор ADSP BF533, в окне «Project output type» нужно указать «Executable file».

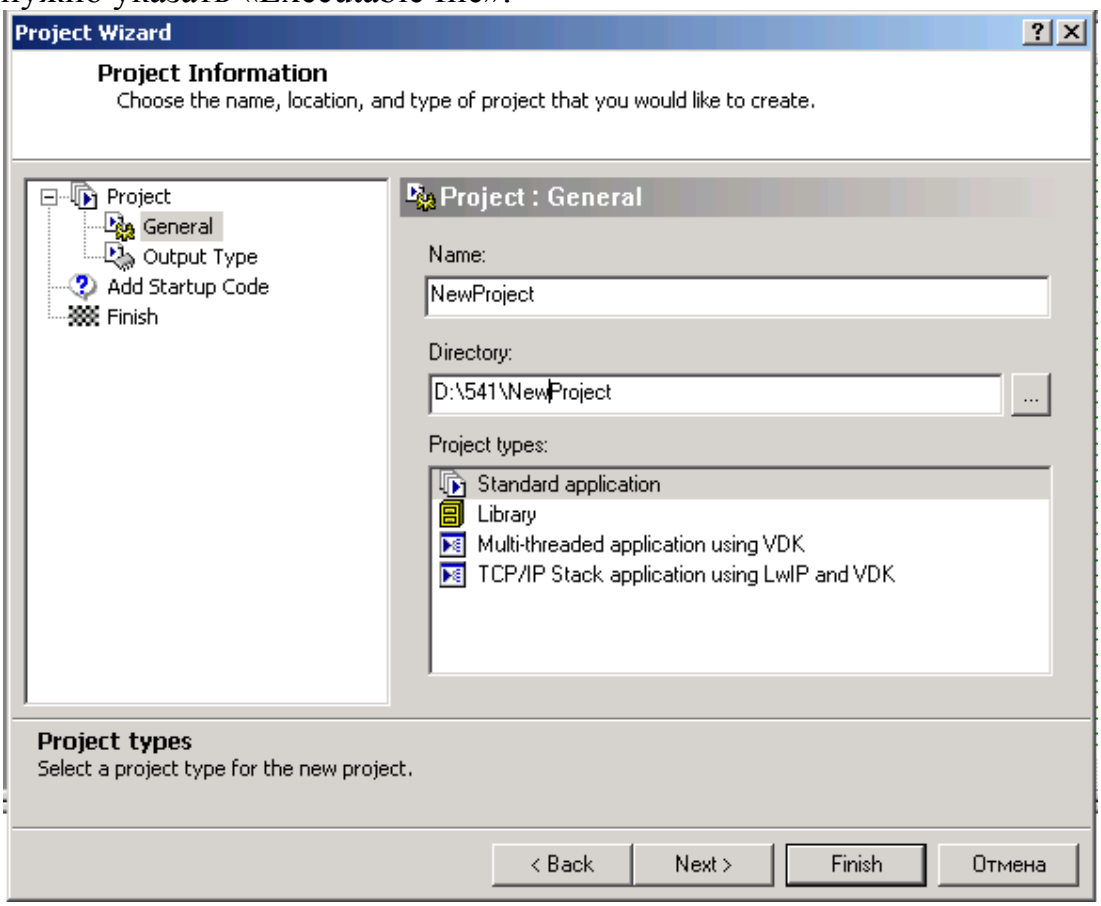


Рисунок 1 – Окно мастера создания проекта

После этого щелкните левой кнопкой мыши по значку «Add startup code» и далее откажитесь от внедрения в проект типового стартового кода (выбор «No»). По завершению всех этих операций нажмите кнопку «Finish».

В левой части окна появилось окно, в котором отображается структура проекта, но пока ничего нет, так как мы еще не включили в состав проекта никаких файлов с исходным кодом.

1.1 Описание исследуемой программы

Программа представляет собой реализацию алгоритма несложного цифрового фильтра с конечной импульсной характеристикой (КИХ). Смысл обработки данных в таком фильтре (на примере фильтра 9-го порядка) представлен на рисунке 2. Здесь z^{-1} означает задержку сигнала на 1 такт, $x(n)$ - отсчеты входного сигнала, $y(n)$ - выходного. $h(n)$ - отсчеты импульсной характеристики фильтра, или его коэффициенты.

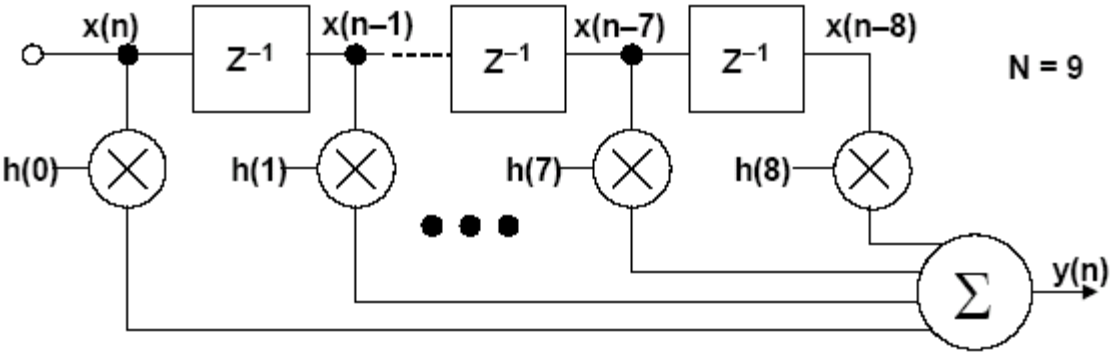


Рисунок 2 – Структурная схема КИХ фильтра

Не вдаваясь в подробности алгоритма, отметим, что фильтры с КИХ нашли довольно широкое распространение в технике цифровой обработки сигналов и их параметры определяются набором коэффициентов. Коэффициенты нашего фильтра (h(0), h(1)…) описываются в файле «fir_coeff.h».

Процедура обработки сигнала реализована на языке ассемблера и находится в файле «fir.asm». Программа, с помощью которой тестируется изучаемый фильтр, написана на языке Си и находится в файле «fir_test.c», который сопровождается заголовочным файлом «filter.h». Массив входных данных, представляющий собой набор чисел типа «short» и имитирующий входной сигнал, содержится в файле «fir_input.h».

Программа на языке ассемблера вызывается из основной программы на языке Си и принимает аргументы с помощью стандартных соглашений, подробно описываемых в файле помощи по работе с компилятором C/C++. Данные, обрабатываемые программой и коэффициенты описаны как «short», что естественно для реализации на базе 16-разрядного процессора.

Отметим также использование структуры, что характерно, в основном, для программ на языке Си, однако может применяться и в программах на ассемблере. Структура представляет собой фрагмент памяти, состоящий из отдельных полей, с точки зрения языка Си, структура является объектом, инкапсулирующим некоторый набор данных, связанных замыслом программиста. К отдельным полям структуры можно обращаться, указывая имя структуры и имя поля, разделенные точкой или символом «->». Подробнее с возможностями работы со структурами можно ознакомиться в руководстве по

компилятору. В нашем случае описание структуры и макроса, инициализирующего ее, содержится в файле «filter.h».

В файле на языке ассемблера, помимо самого текста, содержатся некоторые директивы:

.section program - указывает, что данный фрагмент должен размещаться в памяти программ;

.global _fir– указывает на глобальную видимость идентификатора _fir, что необходимо для работы линковщика;

Все эти файлы находятся на жестком диске компьютера (путь указывается преподавателем). Включите файлы fir_test.c и fir.asm в состав проекта, для чего щелкните правой кнопкой мыши по имени проекта в окне проекта и из контекстного меню выберите пункт «Add File(s) to project».

2 Компиляция и отладка программы

Для компиляции проекта выберите из меню команду «Project/Build project». Результаты компиляции будут отображены в окне «Output». После успешного завершения компиляции среда разработки автоматически перейдет в режим отладки, точка останова будет установлена на первый выполняемый оператор программы. Справа появится окно дизассемблера, в котором отображаются инструкции ассемблера в том виде, как они выполняются процессором.

Отладчик предусматривает режим пошагового выполнения программы, запуска, работу с точками останова, имитацию прерываний и т.п. Все эти особенности аналогичны стандартным возможностям отладчиков, изученных ранее.

2.1 Графическое представление результатов выполнения программы

Объектом операций исследуемой программы является файл данных, представляющий собой массив значений отсчетов входного сигнала фильтра ($x(n)$). Результатом выполнения программы является массив выходных значений сигнала $y(n)$. Среда разработки Visual DSP позволяет просмотреть графическое представление этих данных во временной и частотной области, что позволяет разработчику наглядно оценить результаты своей работы.

В меню «View» выберите пункт «Debug Windows/Plot/New», после чего появится окно выбора графических данных.

Введите название («Title») и укажите имя набора «In», далее с помощью кнопки «Browse» выберите объект «IN». В окне «Count» введите значение 128 и укажите тип данных «short», после чего нажмите «Add». В окне «Data Sets» должно появиться имя набора данных. Повторите эту же процедуру для набора «Out», после чего окно должно принять вид, представленный на рисунке 3.

После нажатия кнопки «ОК» появится окно с изображением содержимого обоих массивов. Если программа еще не запускалась, массив «Out» содержит нулевые значения. Запустите программу (нажатием F5) и в окне отобразится новое содержимое массива «Out» (рисунок 4). Смысл полученного результата очевиден: богатый гармониками входной сигнал после фильтрации представляет собой практически единственную гармонику.

С помощью курсора можно выделить фрагмент изображения для детального исследования, с помощью кнопок-стрелок на клавиатуре можно просматривать значения сигналов по одной выборке. Для этого необходимо выбрать в контекстном меню пункт «Data cursor».

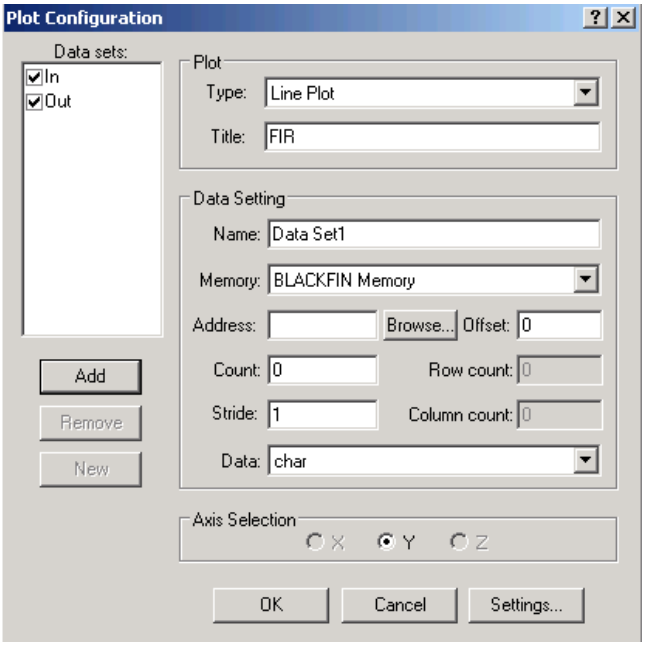


Рисунок 3 – Окно выбора графических данных

Можно перейти и в частотную область, где можно видеть спектральную диаграмму сигналов. Для этого в контекстном меню выберите пункт «Modify Settings» и вкладку «Data processing». Далее выделите имя набора в окне «Data Sets» и в окне «Data Process» укажите «FFT magnitude». В окне «Sample Rate» введите значение 10000 и нажмите «ОК».

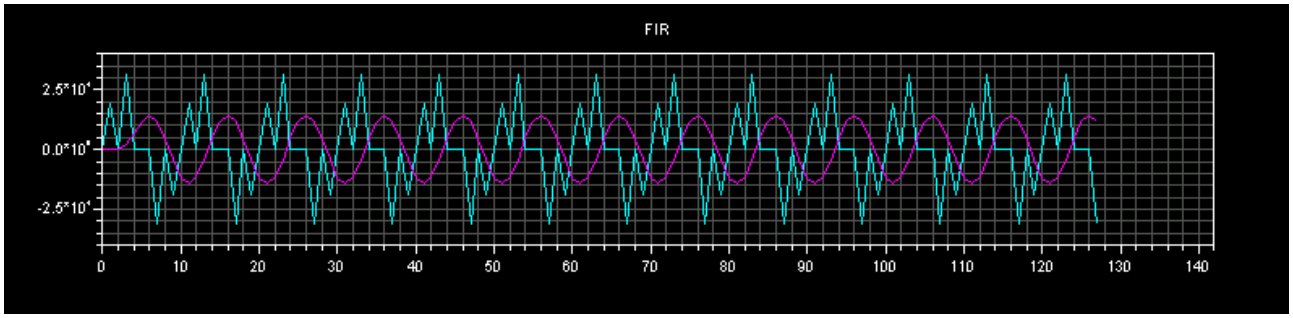


Рисунок 4 – Графическая интерпретация входных и выходных сигналов фильтра

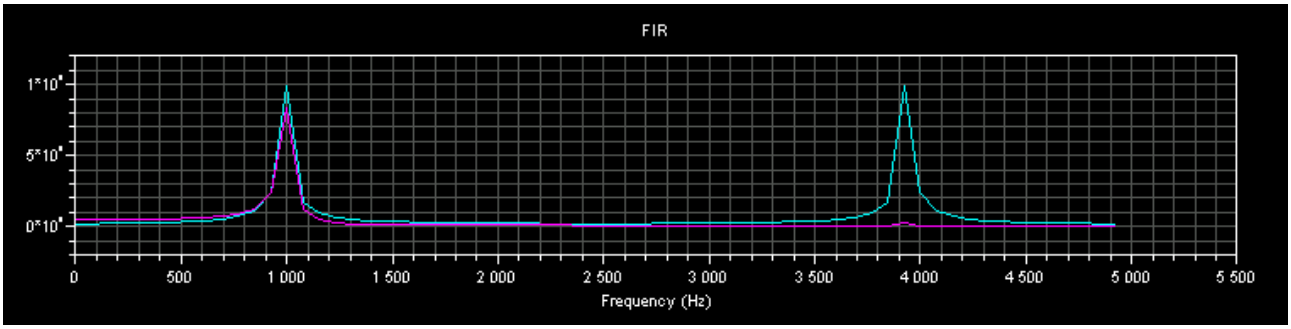


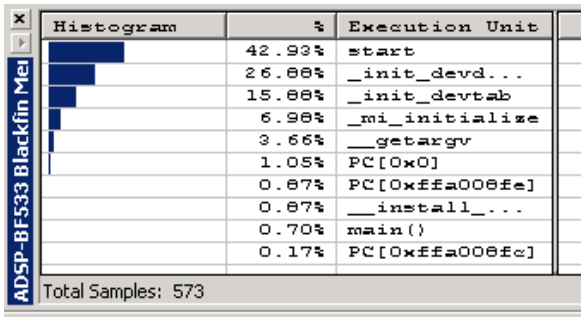
Рисунок 5 – Спектральная диаграмма сигналов на входе и выходе фильтра

Обратите внимание на задержку выходного сигнала во временной области и подавление высших гармоник сигнала на спектральной диаграмме.

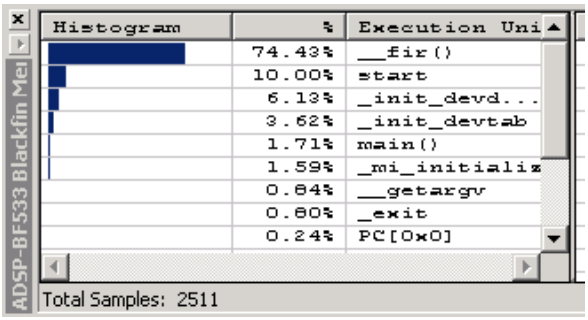
Можно также просмотреть АЧХ фильтра, если в качестве набора данных выбрать массив коэффициентов фильтра h (длиной 8) и режим отображения в частотной области.

2.2 Анализ программы

Выберите в меню пункт «Tools/Linear profiling/New» и откройте новое окно результатов профилировки. Далее перезагрузим программу «File/Load program» (необходимо указать имя исполняемого модуля с расширением «dxe»). Обратите внимание, что хотя программа еще не начала выполняться, в окне профилировки уже есть какие-то результаты. В данном случае это код, автоматически «приписанный» компилятором и инициализирующий процессор в рамках стандартной процедуры, свойственной Си-программам.



До запуска программы



После запуска

Рисунок 6 – Окно профилировщика

Очистим окно профилировки командой «Clear profile» из контекстного меню и запустим программу на выполнение нажатием кнопки F5. Гистограмма и процентные значения затраченного времени покажут эффективность созданного кода.

3 Контрольные вопросы

1) Поясните основные требования к программе на языке ассемблера, вызываемой из Си-программы. Как передаются аргументы функции fir ?
2) Зачем в программе на языке ассемблера используются команды nop ? Попробуйте убрать их из исходной программы.
3) Что содержится в регистре R1 в начале выполнения программы fir ?
4) По какому адресу находится точка входа в создаваемую программу? Каковы адреса точек входа в функции main и fir ?
5) Где размещены исходные данные (массивы IN и OUT) в памяти процессора ?
6) Выясните, какой код предшествует выполняемой программе ? Определите его основные функции.
7) Как узнать точное время выполнения программы fir ?
8) Какие ограничения накладываются на инструкции выполняемые параллельно ? Сравните инструкции в исходном файле с дизассемблированным кодом.

9) Замените в исходном тексте на языке ассемблера параллельные инструкции на отдельные (закомментируйте строку с параллельной инструкцией и раскомментируйте соответствующие строки ниже). Исследуйте, как изменилось поведение программы, результаты профилировки, время выполнения.
10) Поясните принципы организации циклов с нулевыми непроизводительными затратами. Определите длительность выполнения процессором циклов, входящих в состав исследуемой программы.
11) Как использовать циклическую адресацию в процессорах Blackfin ?
12) Что такое исключения, чем они вызываются и как обрабатываются ?
13) Поясните основные принципы обслуживания прерываний в DSP Blackfin

4 Приложение

Структура процессора Blackfin

Ядро и периферийные устройства, а также взаимодействие между ними представлены на рисунке 7.

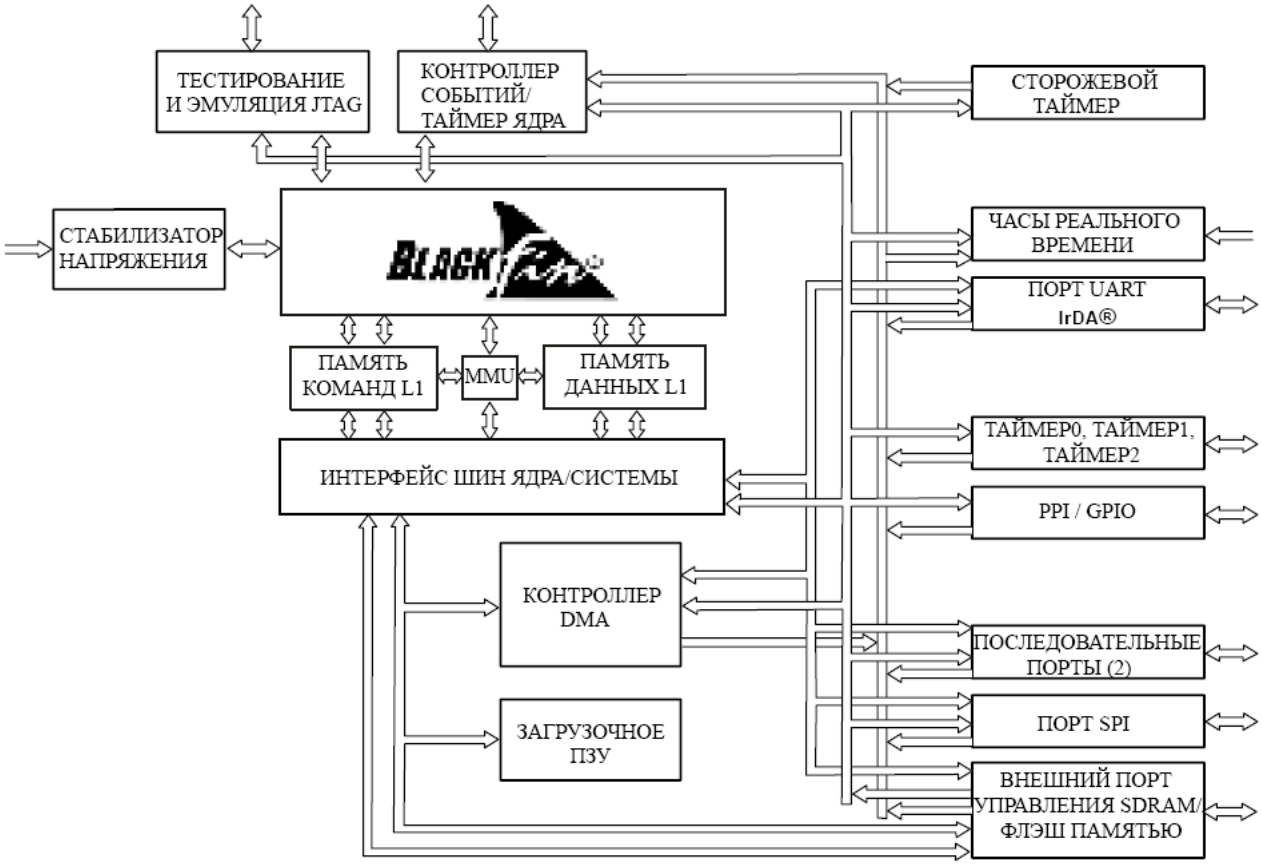


Рисунок 7 – Блок-схема процессора

Архитектура ядра процессора Blackfin является архитектурой с единым набором команд, включающей ядро обработки сигналов со сдвоенным блоком умножения-накопления, имеющей ортогональный набор команд, характерный для RISC-микропроцессоров, обладающей гибкостью команд типа SIMD и мультимедийными возможностями. Особенностью продуктов семейства Blackfin является динамическое управление питанием. Возможность изменения как напряжения питания, так и рабочей частоты позволяет оптимизировать потребление мощности в соответствии с конкретной задачей.

Периферийные устройства системы процессора включают:

- Параллельный периферийный интерфейс (PPI).
- Последовательные порты (SPORT)
- Последовательный периферийный интерфейс (SPI)
- Таймеры общего назначения
- Универсальный асинхронный приёмник-передатчик (UART)
- Часы реального времени (RTC)
- Сторожевой таймер

• Порт ввода/вывода общего назначения (программируемые флаги)

Эти периферийные устройства соединены с ядром несколькими шинами с высокой пропускной способностью, как показано на рисунке 7.

Параллельный периферийный интерфейс (PPI, Parallel Peripheral Interface) – это полудуплексный двунаправленный порт, поддерживающий передачу данных разрядностью до 16 бит. Он имеет выделенный вывод тактовой синхронизации, три мультиплексируемых вывода кадровой синхронизации и четыре выделенных вывода данных.

Последовательные порты (SPORT0 и SPORT1) обеспечивают интерфейс ввода/вывода с различными последовательными периферийными устройствами. Они поддерживают разнообразные протоколы последовательной передачи данных и могут обеспечивать прямое соединение процессоров в многопроцессорной системе.

Порт последовательного периферийного интерфейса (SPI, Serial Peripheral Interface) представляет собой четырехпроводной интерфейс с двумя выводами данных, выводом выбора устройства и выводом сигнала синхронизации. SPI является полнодуплексным синхронным последовательным интерфейсом, поддерживающим режимы ведущего и ведомого устройств и режим с несколькими ведущими устройствами.

Процессор имеет три идентичных 32-разрядных таймера общего назначения, таймер ядра и сторожевой таймер. Для каждого из таймеров общего назначения может быть индивидуально задан один из трёх режимов работы:

• режим широтно-импульсной модуляции (PWM_OUT),

• режим измерения периода и длительности импульса (WDTH_CAP),

• режим счётчика внешних воздействий (EXT_CLK).

Таймер ядра генерирует периодические прерывания, используемые в задачах синхронизации системы.

Универсальный асинхронный приёмопередатчик (UART, Universal Asynchronous Receiver Transmitter) – это полнодуплексное периферийное устройство, совместимое с интерфейсами UART индустриального стандарта. UART преобразует данные между последовательным и параллельным форматами. Последовательная передача (приём) выполняется по асинхронному протоколу, поддерживающему различные длины слов данных, различное количество стоп-битов и возможность формирования битов проверки чётности.

Блок **часов реального времени** (RTC, Real-Time Clock) процессора обеспечивает набор свойств цифровых часов, включающий функции будильника, секундомера и индикации текущего времени.

Процессор имеет 16 двунаправленных программируемых флагов (PFx) или выводов I/O общего назначения, PF[15:0]. Каждый вывод может индивидуально настраиваться на вход или на выход при помощи регистра направления флагов (FIO_DIR).