ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ

1/2001

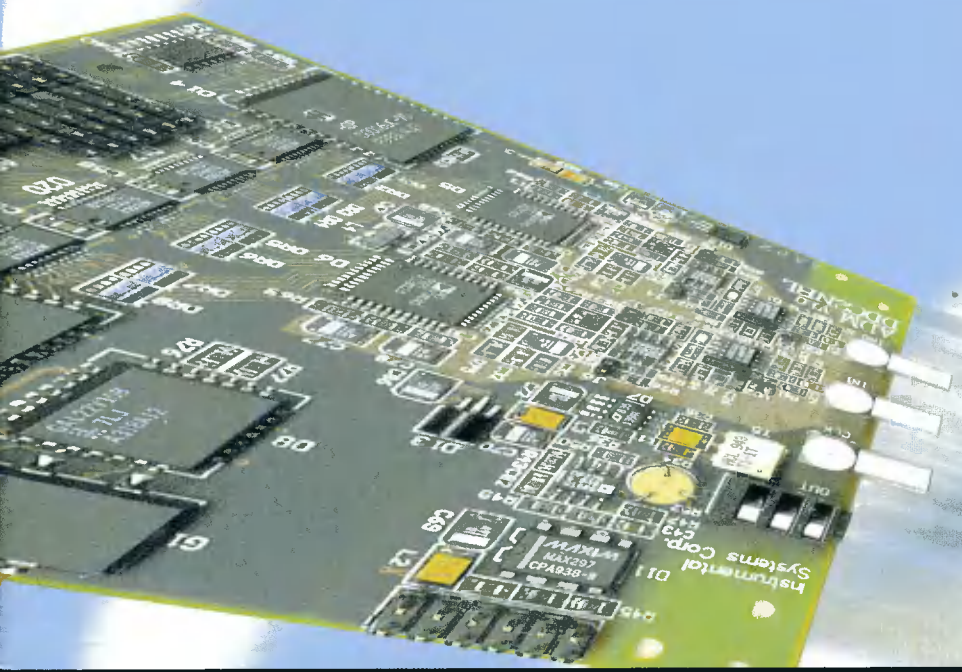
НАУЧНО - ТЕХНИЧЕСКИЙ ЖУРНАЛ

МУЛЬТИМЕДИА: проблемы и перспективы внедрения

ОБРАБОТКА РЕЧИ

ЦИФРОВАЯ ФИЛЬТРАЦИЯ

СИГНАЛЬНЫЕ ПРОЦЕССОРЫ



ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ

№ 1/2001

Адрес редакции:
103064 Москва, ул. Казакова, 16, к. 1202
Тел./факс: (095) 267-0662
E-mail: pl@com2com.ru

Цифровая обработка сигналов
№ 1/2001, с. 1 – 64

Издатель: ООО "ЛТИС"
Генеральный директор
П.Л. Серебрянников

Дизайн и верстка С.Г.Тюканова
Корректура Е.В. Карасева
Н.Н. Сергеева

Издание зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещания и средств массовых коммуникаций
Свидетельство о регистрации
ПИ № 77-1488 от 14.01.00

В НОМЕРЕ:

Ю.Б. Зубарев, В.П. Дворкович, В.А. Козлов, С.Е. Артамонов

Мультимедиа – проблемы и перспективы внедрения систем в России

1. Основные положения

Технической основой создаваемой информационной среды должны стать современные мультимедийные системы, обеспечивающие цифровую передачу динамических изображений, речи, звука, иных данных по каналам с различной пропускной способностью (видеотелефон, стационарная и мобильная телеконференц-связь, интерактивные телевизионные системы и пр.).

Внедрение мультимедийных систем различного назначения при минимальных капитальных и эксплуатационных затратах решает проблемы:

- создания высококачественных систем интерактивного цифрового телевизионного вещания при удовлетворении постоянно возрастающих запросов на частотные присвоения систем связи без пересмотра частотных планов;
- разработки и внедрения принципиально новых систем мобильного телевидения;
- создания принципиально новых систем опроса общественного мнения;
- обеспечения деятельности органов государственной власти;
- создания мобильной телеконференц-связи между центральными учреждениями с удаленными центрами и районами, а также удаленных районов между собой;
- обеспечения сбора и распространения информации различного экономического и политического характера, распространяемой органами власти среди населения, популяризации проводимых властями программ, акций, мероприятий;
- обеспечения деятельности ГО и ЧС, оперативного контроля объектов и дистанционного управления по устранению аварий и ЧС;
- оптимизации лечебной и профилактической деятельности, создания систем мобильной телемедицины;
- создания системы дистанционного обучения на базе ведущих ВУЗов, расширения системы подготовки абитуриентов и пр.;
- поддержки малого бизнеса, проведения рекламных мероприятий;
- организации системы Internet для сельских школ и др.

Требования замены морально и физически устаревших основных технических средств вещания вызывают необходимость внедрения новых прогрессивных комплексов и систем. В этих обстоятельствах наиболее целесообразным и экономичным является переход к цифровым системам.

В России только за счет использования имеющихся технических средств в режиме передачи многопрограммного цифрового телевидения экономический эффект, по самым скромным оценкам, может составить несколько миллиардов рублей в год.

Внедрение мультимедийных систем в России связано с решением пяти наиболее важных основных проблем:

- эффективного использования мирового опыта по созданию и применению систем цифровой обработки и передачи информации;
- разработки стандартов России по системам формирования и передачи по каналам связи различных видов цифровой информации;
- разработки и внедрения собственных мультимедийных систем и соответствующей аппаратуры;
- обучения персонала для обеспечения эффективной эксплуатации таких систем;
- создания средств метрологии мультимедийных систем.

Необходимость и эффективность внедрения цифровых систем в России связаны с требованиями:

- достижения на современном этапе развития видео- и аудиоинформационных систем нового уровня качества воспроизведения изображений и звука;
- удовлетворения постоянно возрастающих запросов на частотные присвоения систем связи без пересмотра частотных планов;
- развития цифровых систем передачи видео- и звуковых данных при существенном увеличении объема передаваемой сопутствующей информации;
- постепенной замены существующего линейного, передающего, коммутационного и др. оборудования, систем технической видеозаписи и хранения информации на цифровые, более прогрессивные, и т.д.;
- создания высоконадежных систем передачи информации с эффективной системой защиты от несанкционированного приема;
- интегрирования России в общемировую систему телекоммуникаций с учетом тенденций перехода к мультимедийным системам многопрограммного телевидения и телевидения высокой четкости в глобальном масштабе.

Решение этих проблем, по сути, может устранить наше отставание в развитии информационных технологий.

2. Что означает эффективное использование мирового опыта по созданию и применению систем цифровой обработки и передачи информации?

Проводимые во всех технически развитых странах разработки алгоритмов и аппаратуры сокращения объема и рационального пакетирования видео-, аудио- и сопутствующей информации являются основой создания систем эффективного использования каналов связи, сохранения действующих частотных планов, высвобождения значительной части частотного пространства для передачи потребителям дополнительных видов услуг – видеотелефона, мобильной и стационарной телеконференц-связи, многопрограммного интерактивного телевидения, телевидения высокой четкости, многопрограммного звукового вещания и др.

Разработка алгоритмов и соответствующей аппаратуры цифрового сжатия различных видов информации для их передачи по каналам связи как альтернативы аналоговым системам проводится уже более 20 лет практически во всех развитых странах мира. Был получен ряд важных результатов как в плане разработки алгоритмов сжатия, включая стандарты JPEG (JPEG-2000), MPEG-1, MPEG-2, MPEG-4 (видео), H.261, H.263, H-263+ и др. для статических и динамических изображений различного разрешения, Layer-2, Layer-3, AC-3, MPEG-4 (аудио), G.729, G.728, G.723-1 и др. для звуковой информации и речи, так и в плане создания действующих комплексов.

В качестве поставщиков цифровых кодеров/декодеров ТВ-информации, выполненных на специализированных БИС и реализующих стандарт MPEG, можно назвать значительное количество известных фирм, например, таких, как Array Microsystems, Compression Labs Inc., COMSAT Lab, Hewlett-Packard, IBM, JVC Professional Products Co., Matsushita Elec. Industrial Co., NEC, NTL, Toshiba Corp., Sony Corp., TV/COM International, BNI, Satellite Transmission Systems Inc., и др.

Большое количество известных фирм занимается проблемами реализации видеоконференц-связи как на уровне создания соответствующих аппаратных средств, так и ее программного обеспечения. К ним можно отнести компании Datapoint, Intel, Picture Tel, Polycom, Sony Electronics, VCOM, VideoServer, VistaCom, White Pine, Zydacron и др.

Из этого обширного, но существенно не полного перечня фирм, занимающихся перспективными разработками цифровых систем передачи видеоинформации, видно, насколько дружно весь мир движется к реализации таких систем. Можно привести еще больший список фирм, освоивших массовое производство БИС и устройств декодирования цифровой информации.

Проблема создания высококачественной системы цифровой передачи видеоинформации – это, главным образом, проблема быстрого и эффективного ее сжатия, наиболее ресурсопотребляющей части всей системы. Ее эффективность существенно влияет на качество воспроизводимой информации.

Кодирующее устройство решает сложные задачи в реальном масштабе времени и реализует большое количество операций, определяющих способы цифрового преобразования сигналов изображения и звукового сопровождения. Декодер, как правило, менее сложен и должен получать и восстанавливать переданные данные.

На основании изучения международного опыта можно предложить приведенные в *табл. 1*

Таблица 1

Приложения	Формат изображения, цветовая модель	Звуковой формат	Стандарт, профиль, уровень	Битовый поток
ТВЧ	1920:1152 (4:2:0)	2/0 стерео + 2 моно	MPEG-2, Главн. Профиль, высокий уровень	24.0 – 31.0 Мбит/с
Многоканальное ТВ	720:576 (4:2:0)	2/0 стерео +2 моно	MPEG-2, Главн. Профиль, главный уровень	5.0 – 8.0 Мбит/с
“Видео на дом”	352:288 (CIF) (4:2:0)	1 моно	MPEG-1, MPEG-4, H.263/H.263+	512 – 1024 кбит/с
Телеконференц-связь	352:288 (CIF) 176:144 (QCIF) (4:2:0)	1 или 2 вокодерных канала	H.263/H.263+, MPEG-4, G.729, G.728, G723.1	64 – 512 кбит/с
Видеотелефон	176:144 (QCIF) (4:2:0)	1 вокодерный канал	H261/H263, G729, G723.1	32 – 64 кбит/с

варианты использования цифровых и звуковых форматов в отечественных системах ЦТВ.

Сложнейшие проблемы связаны с реализацией эффективных методов передачи закодированной информации в каналах связи. Это касается выбора эффективных методов помехоустойчивого кодирования и используемых методов модуляции несущих.

Так, например, для систем распределения многопрограммного цифрового ТВ-вещания в Европе разработана система DVB, которая охватывает спутниковые (DVB-S), кабельные (DVB-C), наземные (DVB-T) средства передачи телевидения. В стандарте реализован принцип использования при различных способах передачи одинаковых методов кодирования сигналов, мультиплексирования, системы коррекции ошибок на первом этапе передачи, что обеспечивает максимальную совместимость разных систем.

В США для обеспечения передачи сигналов телевидения высокой четкости создана система наземного телевизионного вещания ATSC.

Международными стандартами охвачены также такие системы распределения телевизионных программ, как MMDS, LMDS, MVDS. В первой из них стандарт практически аналогичен стандарту DVB-T, а во второй и третьей – DVB-S. Однако в системах LMDS, MVDS не требуется использование помехоустойчивого кодирования.

Следует заметить, что общий объем стандартов, разработанных международными организациями (ITU, ETSI, CINELEC и др.), составляет более 10 000 страниц текстов на английском языке, существенная часть из них недоступна российским специалистам по причине отсутствия средств на их приобретение. Эти стандарты постоянно изменяются по мере совершенствования систем обработки и передачи информации. Ярким примером является появление нового стандарта JPEG-2000, который предусматривает использование алгоритмов wavelet-преобразования.

Исходя из изложенного следует, что важнейшей задачей является разработка высокоэффективных алгоритмов сжатия и передачи мультимедийной информации с использованием зарубежных материалов в качестве задающих общее направление работ, создание высококачественных систем кодирования информации в рамках согласованных международных стандартов.

3. Зачем нужна разработка стандартов России по системам формирования и передачи по каналам связи различных видов цифровой информации?

Цифровые мультимедийные системы обладают следующими преимуществами:

- организация высокоэффективных систем телеконференц-связи и видеотелефона;
- передача многопрограммного телевидения стандартного разрешения в каждом из стандартных каналах вещания;
- передача телевидения высокой четкости в стандартных каналах вещания;
- существенное повышение качества передачи изображений и звука в каналах с различной пропускной способностью;
- возможность использования более низких мощностей передающих средств, существенная экономия электроэнергии;
- гибкость при планировании сетей, позволяющая варьированием параметров системы обеспечивать качественный прием, необходимый объем дополнительных услуг, требуемую зону охвата территории;
- цифровое сопряжение сетей с компьютерными Internet-сетями;
- эффективность использования частотного ресурса за счет передачи мультиплексированных потоков данных, а также организации одночастотных сетей;
- применение цифровых методов защиты передаваемой информации от несанкционированного приема;
- достаточно низкая цена внедрения за счет возможностей использования существующих инфраструктуры аналоговых сетей, антенных и кабельных систем;
- предоставление абонентам в зоне обслуживания цифровой сети набора “обязательных” (информация об общественно значимых учреждениях, погода, расписания движений транспорта и др.) и дополнительных (реклама, видео по запросу и др.) услуг, зависимых от потребностей рынка и возможностей операторов.

Канал связи может рассматриваться как “контейнер данных”, обеспечивающий передачу различных сигналов: информационных сообщений, одной или более ТВ-программ, данных по запросу пользователя. Плотность заполнения “контейнера” данными зависит от обеспечиваемой скорости передачи.

При рациональном мультиплексировании система позволяет оператору потока данных в несколько раз увеличить число программ и дополнительных услуг, предлагаемых зрителям, без использования дополнительного частотного ресурса.

Например, при эфирном вещании в полосе частот 8 МГц метрового или дециметрового диапазонов можно обеспечить передачу до 4 и более ТВ-программ стандартной четкости студийного качества.

Использование же двух смежных каналов позволит передавать до 10 таких ТВ-программ за счет статистического мультиплексирования цифровой информации.

Важнейшим вопросом является разработка отечественных стандартов формирования, передачи и приема цифровой информации с учетом особенностей построения мультимедийных систем в России.

Новые стандарты должны разрабатываться с перспективой, а не фиксировать и, тем более, закреплять нашу техническую отсталость. Они должны стимулировать разработчиков и производителей передовой технологической базы мультимедийных систем.

Исследования характеристик существующих систем цифрового телевидения позволяют сделать вывод о целесообразности ориентации России на систему DVB как наиболее отвечающую принципам организации систем связи в России. Приведенные в стандартах DVB параметры и характеристики систем спутникового, кабельного и наземного цифрового телевидения основаны на использовании принципов кодирования MPEG-2, допускают множество методов канального кодирования (особенно в части сверточного кодирования) и различные варианты модуляции (от QPSK до 256-QAM для кабельного телевидения и использования от QPSK до 64-QAM различного количества несущих 2к или 8к OFDM).

В отечественных стандартах должны быть строго определены варианты цифровой обработки изображений и звука в аппаратно-студийных комплексах различного назначения (от телевидения высокой четкости до домовых систем "видео по запросу"), варианты использования различных методов канального кодирования в спутниковых и наземных системах ТВ-вещания, варианты применения различных систем фазовой манипуляции несущих в наземных и кабельных системах ТВ-вещания, применения вариантов OFDM. Особое значение имеют проблемы определения набора используемых мощностей передающих средств наземного и спутникового ЦТВ.

При разработке отечественных стандартов, кроме того, должны быть решены проблемы использования систем точных частот и точного времени, что является особенно важным, учитывая территориальные особенности России.

Могут быть предложены новые методы планирования сетей телевизионного вещания, связанные с возможностью эффективного использования смежных каналов, поскольку при передаче 4 программ в стандартном ТВ-канале статистическое уплотнение позволит обеспечить до 10 программ в двух смежных каналах и порядка 15 – 16 программ в трех соседних каналах. В этом плане Россия может предложить варианты совершенствования стандартов DVB.

В рамках создания отечественных стандартов должны быть решены вопросы использования и передачи дополнительной информации в составе цифрового потока ЦТВ с учетом спецификаций DVB.

4. Разработка и внедрение собственных мультимедийных систем и соответствующей аппаратуры

Несмотря на уже начавшееся широкое внедрение цифровых мультимедийных систем, процесс разработки более эффективных алгоритмов сжатия визуальной и звуковой информации продолжается.

Поскольку имеющиеся стандарты определяют лишь цифровой поток кодируемых данных и не регламентируют принципы его формирования, имеется возможность более эффективной обработки сигналов в рамках этих стандартов.

Большое внимание научной общественности и фирм – производителей систем цифровой обработки изображений занимают проблемы создания альтернативных систем сжатия видеоинформации:

- векторного кодирования,
- вейвлет (Wavelet)-преобразования,
- нейрокомпьютерной обработки,
- фрактального кодирования и иных комбинированных систем, потенциально обеспечивающих более высокие коэффициенты сжатия.

Основные трудности связаны с тем, что лежащие в основе алгоритмов сжатия преобразования содержат весьма большое количество математических операций на каждый дискретный отсчет изображения и при обработке динамической последовательности кадров в реальном масштабе времени требуют такого быстродействия, которое может быть обеспечено лишь супер-ЭВМ с соответствующей системой распараллеливания операций.

Несмотря на наличие значительного количество публикаций в мире по тематике сжатия видеоинформации, практически все эти публикации содержат лишь самое общее описание методов сжатия без раскрытия наиболее существенных деталей используемых алгоритмов.

Информации же о конкретной совокупности операций, составляющих суть этих алгоритмов, в силу действия законов по интеллектуальной собственности невозможно обнаружить не только в открытых публикациях, но и в конкретных материалах фирм-разработчиков.

В связи с этим важнейшей задачей проведения фундаментальных работ в России по данной тематике является задача полного овладения теорией и математическими методами сжатия изображений, используя имеющиеся зарубежные материалы в качестве информации, задающей общее направление работ. А далее, с учетом полученного опыта по освоению стандартных алгоритмов, разработка более эффективных систем.

Например, к числу неэффективно решенных проблем в разработке кодера MPEG можно отнести вопросы:

- распознавания смены сюжета и связанное с ним разбиение последовательности изображений на I-, P- и B-кадры,
- повышения коэффициента сжатия I-кадров,
- расширения зоны и точности оценки и компенсации движения деталей динамических изображений и др.

В рамках проведенных в НИИР исследований предложены и реализованы весьма эффективные алгоритмы статистической оценки наличия в последовательности кадров смены сюжета и изменения структуры блока одновременно обрабатываемых кадров. Разработаны принципы повышения эффективности обработки I-кадров на основе анализа допустимых искажений их структуры и изменения на этой базе матрицы квантования различных блоков кадра.

Проведена разработка ускоренного алгоритма анализа движения, что позволило существенно увеличить зону поиска вектора движения, сократить требуемую вычислительную мощность и обеспечить компенсацию быстродвижущихся деталей, повышая при этом качество воспроизведения динамических изображений. Использование предложенного способа позволяет оценить движение не только с предусмотренной в MPEG-2 точностью до полпиксела, но с любой заданной точностью.

На указанные способы цифровой обработки изображений получены российские патенты.

Следует заметить, что большинство кодирующих устройств, производимых рядом фирм, обеспечивают весьма высокое качество передачи телевидения стандартного разрешения при битовом потоке порядка 8 – 9 Мбит/с и использовании Главного профиля и Главного уровня. Работы, проведенные в НИИР по совершенствованию алгоритмов обработки изображений, позволяют обеспечить такое же качество передаваемых изображений при битовом потоке порядка 6 – 7 Мбит/с.

В НИИР проводятся также компьютерные исследования и по ряду других перспективных методов сжатия, в частности по вейвлет-преобразованию, фрактальному кодированию и др. Внедрение результатов этих исследований позволит дополнительно сократить объем передаваемой информации и обеспечить более эффективное использование частотного пространства.

Накопленный теоретический багаж позволил перейти к его практической реализации.

В частности, в НИИР создана система компьютерной телеконференц-связи/видеотелефонии VPHONE, предназначенная для проведения телеконференций между стационарным узлом и удаленными стационарными или мобильными терминалами, а также между удаленными терминалами в режиме точка-точка в пределах территории, охваченной каналами связи системы.

Организация такой системы связи с центром и/или друг с другом позволяет реализовать системы интерактивного, в самом деле, телевизионного вещания, опроса общественного мнения в реальном масштабе времени, дистанционного интерактивного обучения, видеосвязи для использования в чрезвычайных ситуациях, дистанционного медицинского обслуживания, видеотелефонии и др.

Система является программно-аппаратным комплексом, позволяющим в реальном масштабе времени осуществлять видео- и аудиосвязь между пользователями по каналам связи от 32 до 1024 кбит/с. Благодаря использованию новых оригинальных алгоритмов быстрого кодирования, защищенных российскими патентами (№ 2122295, 2137194), удалось создать высококачественные системы компьютерной телеконференц-связи при весьма эффективном сжатии видео- и аудиоинформации и с использованием каналов заданной пропускной способности. В созданной реализации используются все заложенные в международных стандартах возможности видео- и аудиокомпрессии, включая реальную компенсацию движения между соседними кадрами динамического изображения.

Краткие технические характеристики системы:

- поддерживаемые протоколы сжатия видеоданных: H.261; H.263;
- поддерживаемые протоколы сжатия аудиоданных: G.723.1; G 728, G 729;
- передача видеоизображения, звука и произвольных дополнительных данных по каналу связи от 32 до 1024 кбит/с при частоте кадров изображений от 5 до 25 кадров/с;
- защита от несанкционированного доступа к передаваемой информации за счет использования собственных протоколов мультиплексирования и шифрования.

Особенности реализации

Благодаря реализации оригинальных алгоритмов кодирования система обеспечивает максимальное качество изображения при передаче по каналу связи с заданной пропускной способностью.

Рабочая станция системы выполнена в виде персонального компьютера, к которому подключены видеокамера, телевизионный и/или компьютерный мониторы, источник звукового сигнала и устройство его усиления. На компьютер установлен программно-аппаратный комплекс VPHONE, обеспечивающий обработку, кодирование и полнодуплексную передачу видеоизображения и звука.

Межкадровое сжатие видеосигнала осуществляется с использованием компенсации движения с точностью до 0,5 пиксела в пределах +31 пиксел на кадр. Внутрикадровое сжатие осуществляется с применением переменных параметров кодирования в зависимости от характера изображения в данном участке кадра.

Динамический подбор суммарного коэффициента сжатия осуществляется в зависимости от характера изображения и текущей пропускной способности канала передачи.

Используемый эффективный алгоритм анализа движения позволяет сократить число операций в 30 – 100 раз и осуществить реальную компенсацию движения между кадрами.

Особенностью алгоритмов внутрикадрового кодирования является адаптивный выбор коэффициентов квантования спектра дискретного косинусного преобразования в зависимости от характера изображения в данном участке кадра. Кроме адаптивного квантования применяется алгоритм выбора оптимального способа кодирования каждого макроблока (без или с предсказанием движения), а также алгоритм подбора суммарного размера закодированного кадра в зависимости от пропускной способности канала. В результате удается достичь максимального сжатия при сохранении высокого визуального качества изображения.

Структура и состав системы

В состав системы телеконференц-связи входят:

- центральный узел администрирования сети (ЦУС);
- мобильные или стационарные (удаленные) терминалы с программно-аппаратными комплексами (МПАК).

Предусматриваются два режима работы системы:

- телеконференция и/или видеотелефонная связь МПАК между собой в режиме точка-точка;
- телеконференция и/или видеотелефонная связь ЦУС с заданным количеством (4 и более) МПАК.

Видеосигналы во втором режиме могут использоваться для проведения широковещательных ТВ-телеконференций. В этом случае несколько видеосигналов, принимаемых от удаленных терминалов, расположенных в различных регионах, микшируются на центральной студии в стандартный телевизионный сигнал, который потом передается по телевизионному каналу.

Каналы связи

Система может использовать различные каналы связи: спутниковые, радиорелейные, Internet, стандартные коммутируемые телефонные линии и другие.

Для работы системы необходимо использование соответствующего приемопередающего оборудования, поддерживающего протоколы компьютерной передачи данных TCP/IP и обеспечивающего заданную пропускную способность (32 – 1024 кбит/с) в полнодуплексном режиме. Данное оборудование подключается к персональному компьютеру МПАК через соответствующий интерфейс.

Центральный узел сети

При использовании центрального узла сети (ЦУС) обеспечивается телеконференция между центральной студией и одновременно четырьмя или более МПАК. В таком режиме несколько пользователей могут участвовать в совместной конференции с центральной студией. Общая конференция при этом может транслироваться по центральному телевизионному каналу. В состав программно-аппаратного комплекса ЦУС входят: сервер сети; приемопередающие рабочие станции, оборудованные требуемым сетевым интерфейсом, обеспечивающим связь МПАК; рабочая станция, кодирующая цифровой поток, и другие устройства сопряжения с оборудованием ввода/вывода видео и звука.

Сервер обеспечивает поддержку специальных протоколов маршрутизации пакетов в сети (используется протокол OSPF динамической маршрутизации, что позволяет автоматически менять конфигурацию сети при появлении и отключении клиентов). Другая функция сервера – обеспечение стабильной маршрутизации пакетов в режиме видеосвязи между клиентами.

Мобильный или стационарный абонентский программно-аппаратный комплекс

В состав МПАК входит персональный компьютер, оборудованный устройством кодирования видео и сопряжением с соответствующим сетевым интерфейсом. Персональный компьютер выполняет все операции, необходимые для кодирования и декодирования аудио- и видеосигналов. В компьютере устанавливается плата аппаратного ускорителя видеокодера, выполняющая часть необходимых вычислительных операций.

Сервер является ведущим маршрутизатором в образующейся сети.

На нем непрерывно работает специальный пакет программ, позволяющий удаленным станциям связываться с центральной студией и вести диалог. Сервер позволяет также транслировать сигнал кодера сразу нескольким абонентам (как правило, это стандартный режим при ведении телемостов, когда есть гость в студии или ведется трансляция из зала). Сервер автоматически отслеживает появление и отключение клиентов. Мобильные станции передают свои пакеты через сервер, а декодеры их принимают и формируют видеосигнал. Такое устройство системы позволяет клиентам динамически подключаться и отключаться без уточнения их IP-адреса для декодеров. Это обеспечивает также проведение диалога сразу с несколькими передвижными станциями, если имеется соответствующая пропускная способность канала связи. В режиме работы точка-точка участвуют лишь две станции, и наличия студии не требуется.

Поскольку вся система построена по принципу Internet, то возможно подключение внешних абонентов через сервер. При наличии внешней сети открывается доступ в эту сеть и передвижным станциям. К примеру, если внешней сетью будет весь Internet, то передвижные станции получают доступ в него, будучи клиентами локальной сети центральной студии, а внешние абоненты получают возможность видеосвязи с передвижными станциями.

Основой успешного использования системы является ее относительная в сравнении с традиционными телевизионными средствами дешевизна; арендные платежи за каналы связи составляют лишь десятые доли тех затрат, к которым "привыкло" телевизионное вещание в процессе проведения телемостов.

Реализация данной системы позволяет уверенно утверждать о нашей возможности создания СБИС, необходимых для производства кодирующих и декодирующих устройств для видеотелефонов, телеконференций, многопрограммного телевидения и телевидения высокой четкости. Разработка и производство СБИС могут быть осуществлены в России на базе созданных алгоритмов и имеющихся патентов по уменьшению вычислительной мощности и распараллеливанию архитектуры.

При внедрении систем цифрового телевидения важное внимание необходимо уделить разработке рекомендаций по повышению качества ТВ-вещания.

Используемые в настоящее время системы цифрового распределения ТВ-программ неэффективны, поскольку часто применяется принцип формирования ТВ-программы в стандарте PAL со спектром сигнала яркости до 5 МГц, затем этот сигнал декодируется, преобразуется в цифровой поток, передается по каналу связи и наконец перед излучением в эфир преобразуется в сигнал системы SECAM. В результате таких преобразований спектр сигнала яркости ограничивается частотой порядка 3 МГц, и воспроизводимое изображение соответствует качеству VHS. Такой подход при внедрении современных систем цифрового телевидения неприемлем.

Следует считать приоритетным для специализированных научных учреждений, радио- и микроэлектронной промышленности, сопряженных с ними областей производства проведение разработок и создание средств передачи и приема наземного, кабельного и спутникового цифрового ТВ- и звукового вещания.

Внедрение цифрового телевидения предполагает, в общем случае, замену парка передающих и приемных средств, хотя существуют и варианты использования имеющегося аналогового оборудования. Отечественные производители должны своевременно освоить технологию производства такого оборудования и создать аппаратуру, конкурентоспособную на мировом рынке.

В этом плане разработке подлежат кроме систем кодирования информации:

- системы мультиплексирования цифровых сигналов;
- аппаратура формирования радиосигналов в соответствии со стандартами DVB-T, DVB-S, DVB-C и др.;
- соответствующие радиопередающие средства;
- приемная аппаратура профессионального и бытового применения.

5. Приставка к телевизору, цифровой телевизор или компьютер, реализующий все мультимедийные функции?

В настоящее время широко дебатируются вопросы, связанные с проблемами использования различных устройств, реализующих функции цифровых абонентских устройств.

Наиболее простым представляется использование Set-top-box (STB), обеспечивающей преобразование цифровых сигналов в аналоговый сигнал SECAM или PAL (по видеочастоте или на радиочастоте), воспринимаемый аналоговыми телевизионными устройствами. Как уже было отмечено ранее, такой подход явно нерационален по двум причинам:

- требуется использование дополнительных кодирующих и модуляционных систем;
- резко снижается качество воспроизводимых изображений из-за специфических свойств используемых аналоговых преобразований.

Более приемлемым представляется создание специализированных цифровых телевизоров, хотя и этот вариант тоже не решает всех проблем: для реализации мультимедийных функций требуется различная номенклатура таких абонентских устройств, существенно отличающихся по структуре построения и цене. Существенным недостатком этого варианта является его жесткая привязка к используемым стандартам обработки информации и вследствие этого сложность их замены на более эффективные, создаваемые в процессе научных исследований.

Самым рациональным решением является применение компьютерных систем, содержащих соответствующие тюнеры.

Прогресс в этой области в последние годы и его развитие в ближайшие 5 – 10 лет поразителен. Индустрия микропроцессоров входит в XXI век с системами Itanium фирмы Intel и архитектурой x86(x86-64) фирмы AMD – 64-разрядными версиями при тактовых частотах выше 1 ГГц. Эти процессоры обеспечат скорость обработки информации в несколько миллиардов операций в секунду. В соответствии с краткосрочными прогнозами в 2001 году новые видеокарты бу-

дут иметь геометрический процессор, обеспечивать поддержку DVD, цифровой интерфейс DVI и иметь производительность выше гигапикселов в секунду. На типичном компьютере нижнего ценового уровня будет использоваться оперативная память (возможно DDR DRAM) в 128 – 192 Мбайт. Большинство винчестеров будут иметь частоту вращения шпинделя порядка 10 000 об/мин (в несколько раз выше нынешнего) и обладать объемом 35 – 50 Гбайт. Предполагается, что к 2005 году их емкость достигнет 500 Гбайт (объем, обеспечивающий память закодированных по стандарту MPEG-2 стандартных телевизионных программ на 7 суток).

В настоящее время и в дальнейшем ПК делятся на два ценовых класса – "дешевые" и более "серьезные" компьютеры. Стоимость "дешевых" компьютеров без учета цены дисплея составляет порядка $300 – 400, предполагается, что эта планка будет еще ниже.

Следует заметить, что эффективность использования компьютеров связана также с возможностью постепенного наращивания мультимедийных функций путем замены плат компьютеров на более совершенные и дополнения их новыми платами и программами. При этом весьма привлекательным является использование программных декодеров, что обеспечивает возможность совершенствования стандартов кодирования, не изменяющих аппаратные средства абонентов.

Особым преимуществом применения компьютерной технологии является использование различных дисплеев: от 14" до 60" по диагонали, от настольных до настенных вариантов, от стандартного разрешения до разрешения высокой четкости и др. Цена таких дисплеев может колебаться от $150 до 5000 – 10 000 от потребительских свойств и пожеланий абонента. Изменение типа дисплея при этом практически не требует замены аппаратной части.

6. Обеспечить выпуск требуемых специалистов и обучение эксплуатационного персонала для обслуживания мультимедийных систем

Серьезной проблемой при внедрении мультимедийных систем в России является подготовка специалистов-эксплуатационников данного профиля. Отечественная литература, посвященная изложению современных способов цифровой обработки мультимедийной информации, практически отсутствует, эта важнейшая область не отражена в ныне существующих программах обучения специалистов. Функцией обучения специалистов теории и практике мультимедийных систем, помимо создания представления о весьма сложных преобразованиях сигналов в процессе сокращения объема информации, должно стать привлечение молодых специалистов России к исследованиям еще не решенных проблем в этой перспективной области.

Состояние и перспективы развития информационных технологий в начале XXI века характеризуются становлением и широким практическим использованием техники цифровой обработки сигналов – одной из самых динамичных и быстроразвивающихся технологий в мире телекоммуникаций и информатизации общества. Цифровая обработка сигналов (ЦОС) – это ИНФОРМАТИКА реального времени, призванная решать задачи приема, обработки, сокращения избыточности и передачи информации в реальном времени.

Методы формирования и передачи мультимедийной информации вызывают повышенный интерес ученых и специалистов, работающих в различных областях, таких, как, связь и системы управления, радиотехника и электроника, акустика, радиовещание и телевидение, измерительная техника, и др. Ежегодно во всем мире проводятся десятки международных научно-технических конференций и семинаров, посвященных решению актуальных проблем. Постоянно растет число предприятий, организаций и научных центров, использующих в своих разработках мультимедийные методы и соответствующие технологии.

Хотелось бы надеяться, что начавшееся широкое внедрение в практику мультимедийных систем – от видеотелефонии до телевидения высокой четкости – послужит началом реализации планов обучения специалистов данного профиля.

В области организации науки и подготовки кадров важнейшим является формирование сети научно-исследовательских центров по мультимедийным проблемам, обладающих необходимым научным и техническим потенциалом для решения как фундаментальных проблем обработки сигналов, так и широкого круга прикладных задач.

7. Обеспечить создание средств метрологии мультимедийных систем

В России и странах СНГ сложилась крайне тяжелая ситуация, вызванная повсеместным недостатком, а часто и отсутствием измерительной аппаратуры, необходимой для разработки, настройки и поддержания технических характеристик систем передачи информации в состоянии, обеспечивающем их качественное функционирование.

Особые проблемы возникают в связи с началом широкого внедрения систем и аппаратуры цифровой обработки и передачи компрессированных сигналов по каналам связи. Разрабатываемые системы требуют создания соответствующих контрольных и измерительных средств.

С внедрением мультимедийных систем потребность в использовании измерительной аппаратуры будет возрастать, поскольку требуется создание принципиально новых способов и средств контроля и измерений. При этом необходимо, чтобы эти средства были совместимы и с традиционными аналоговыми системами.

Метрологическая безопасность России – залог создания высококачественной мультимедийной аппаратуры и ее эффективного использования.

На современном этапе развития техники выполнение требований метрологии связано с созданием виртуальных измерительных систем на базе использования персональных компьютеров в качестве устройств анализа и организации структуры систем формирования и обработки измерительной информации.

При этом обеспечиваются:

- реализация патентно-чистых способов измерений с применением специально разработанных оптимальных измерительных сигналов и процедур их обработки, позволяющих резко повысить точности и быстродействие измерений;
- существенное увеличение функциональных возможностей создаваемых измерительных средств, а также полностью цифровая реализация;
- снижение практически на порядок цены приборов, что достигается их реализацией на базе общедоступных персональных компьютеров, дополнительно комплектуемых соответствующими блоками (платами) ввода и вывода измерительной информации и программными продуктами.

Начало в разработке принципиально новых приборов на базе использования персональных компьютеров было положено при создании в НИИР видеоанализатора ВК-1/2 и аудиоанализатора компьютерного АК-1, обеспечивающих как генерацию стандартных и новых измерительных сигналов, испытательных эталонов, так и анализ их искажений, осциллографический, спектральный и векторный анализ сигналов и их составляющих.

8. Внедрение мультимедийных систем

Как было отмечено выше, современные тенденции информатизации общества свидетельствуют о постепенном сращивании телевидения и компьютерных сетей, образовании единого пространства мультимедийной информации. В связи с этим вопросы развития цифрового телевещания необходимо рассматривать в тесной взаимосвязи с проблемами информатизации.

Большое внимание в мире уделяется созданию мультимедийных пользовательских систем, объединяющих возможности компьютерных средств и возможности приема цифрового телевидения и проигрывания оптических видеодисков. Не вдаваясь в детали ведущихся дискуссий на тему, что должно лежать в основе подобных мультимедийных систем – компьютер или телевизор, можно предположить, что наиболее оптимальным решением для подобной системы является интегрированная компьютерная платформа, обладающая гибкостью компьютерного программирования и необходимыми функциями цифрового телевизора. Кроме того, платформа должна обеспечить эффективную интеграцию глобальных и локальных компьютерных сетей и телевизионных коммуникаций.

С точки зрения максимального удовлетворения пользовательских потребностей платформа должна обеспечивать целый набор компьютерно-телевизионных услуг:

- прием цифрового телевидения;
- видео-по-запросу;
- проигрывание дисков DVD;
- прием дополнительной цифровой информации;
- доступ в Internet с предоставлением услуг электронной почты;
- IP-телефония и видеоконференц-связь;
- поддержка сетевых игр и пр.

Создание подобной компьютерной платформы позволит эффективно соединить два на первый взгляд независимых процесса – переход к цифровому телевещанию и развитие информатизации общества. Более того, определенный шаг в развитии одного из процессов способен стимулировать рост другого.

Аппаратные решения для мультимедийных систем

Для поддержки перечисленных информационных услуг в настоящее время могут использоваться как универсальные персональные компьютерные платформы, так и развиваемые в настоящее время многими зарубежными компаниями (Motorola, Philips, Texas Instruments и др.) устройства Set-top-box (STB). Многие компании, выпускающие подобные устройства, часто идут по пути прямой интеграции компьютерных решений, построенных на базе существующих универсальных микропроцессоров, с аппаратурой для приема и декодирования цифрового видео. Подобные решения, как было сказано выше, не всегда приводят к значительному удешевлению средств, поддерживающих перечисленные выше сервисные функции. Анализ же этих функций позволяет сделать вывод, что разработка целевой аппаратуры для их поддержки может позволить существенно снизить стоимость телевизионных компьютерных приставок.

Время, когда большинство пользователей в России сможет приобрести персональный компьютер для удовлетворения перечисленных выше потребностей, еще, к сожалению, не настало. Кроме того, современные тенденции разви-

тия компьютерных технологий ориентированы на постоянную модернизацию компьютерных решений как на периферийном, так и на системном уровне в силу постоянно растущих "аппетитов" производителей компьютерной техники и программного обеспечения. Это приводит к постоянному вытеснению морально устаревших вычислительных средств. Использование в настоящее время, например, компьютера с 486-м микропроцессором значительно затрудняется, учитывая отсутствие возможности подключения к нему различного современного периферийного оборудования – звуковых, графических и сетевых карт и т.д., а также проблем с установкой современных программных средств.

Все это позволяет, видимо, сделать вывод о рациональности использования в настоящее время в России компьютерных приставок к телевизионным устройствам, обладающих, как минимум, перечисленным выше набором функциональных возможностей. В дальнейшем по мере появления на рынке доступных по стоимости телевизионных компьютеров компьютерные приставки к телевизорам будут постепенно заменяться на более сложные компьютерные устройства.

Накопленный опыт в областях цифрового видео и построения компьютерных систем уже в настоящее время может быть эффективно использован для построения сравнительно недорогих телевизионных приставок, обладающих базовым набором функций и сервисов мультимедийных систем. На первых этапах возможно применение зарубежной элементной базы для построения недорогих сетевых мультимедийных компьютеров с постепенной заменой в дальнейшем на отечественную.

Коммуникации

Внедрение мультимедийных систем позволит более эффективно использовать существующие телекоммуникации.

Подключение жилых домов к компьютерной сети возможно уже на настоящем этапе с использованием радиоканалов с пропускной способностью до 2 Мбит/с, технологий xDSL с парами телефонных линий. При этом внутри каждого жилого дома создается локальная компьютерная сеть Ethernet10base или Ethernet100base для обслуживания конечных пользователей.

Группы домов или микрорайоны могут подключаться с использованием оптоволоконных линий. На более глобальных уровнях возможно использование также и спутниковых каналов связи.

Прямой канал для подключения к Internet может быть организован и с использованием физического уровня телевизионных кабелей, что позволит значительно разгрузить существующие в настоящее время телекоммуникации. Обратный канал не требует широкой пропускной способности и может быть организован с применением xDSL-технологии.

С другой стороны, организация внутри жилого дома или группы домов компьютерной сети позволит обеспечить и дополнительные услуги цифрового телевидения и видео-по-запросу.

Поэтапное внедрение мультимедийных систем и развитие цифрового телевещания

Необходимость перехода в России на цифровое телевещание не вызывает сомнений. Однако ограниченность финансовых ресурсов российских телевизионных компаний и низкий покупательский спрос населения делает невозможным скачкообразный переход на использование цифровых технологий, требующий закупки и производства новых телепрограмм, глобальной замены как телевещательного оборудования, так и телевизионных приемников. Единственным подходом к постепенному переходу на цифровые технологии является поэтапное развитие и внедрение комплексных мультимедийных систем. Предоставление комплекса мультимедийных услуг может существенно повысить заинтересованность участвующих в процессе перехода на цифровые технологии сторон.

Развитие мультимедийных систем, обладающих перечисленными выше возможностями по предоставлению различных телевизионных и информационных сервисов, позволит значительно повысить заинтересованность конечных пользователей в их приобретении и использовании.

Заинтересованность Internet-провайдеров в развитии подобных систем может заключаться как в возможности увеличения пропускных способностей телекоммуникаций за счет использования телевизионных коммуникаций, так и в увеличении числа пользователей и расширении набора предоставляемых информационных услуг.

Для телевизионных компаний переход на цифровое вещание означает существенное снижение затрат на аренду каналов, возможность развития новых каналов и телепрограмм, повышение качества и объема предоставляемых услуг.

Процесс перехода может быть поддержан как "снизу" путем развития локальных компьютерных сетей, подключения к Internet, использования возможностей местных и кабельных телестудий, так и "сверху" за счет развития новых телепрограмм и телеканалов и вещания их в цифровой форме.

Для осуществления подобной программы необходимо проведение организационной работы по координации всех заинтересованных сторон – телевизионных каналов, компаний и студий, крупных Internet-провайдеров, поставщиков телекоммуникационного оборудования, компаний, разрабатывающих и производящих телевизионную и компьютерную технику.

С.Н. Кириллов, М.В. Степанов, О.Л. Виноградов

Синтез адаптивных цифровых фильтров по комбинированному методу наименьших квадратов

Разработан алгоритм адаптации коэффициентов цифровых фильтров на основе комбинированного метода наименьших квадратов (МНК), включающего в себя взвешенные средний квадрат сигнала ошибки и средний квадрат изменения сигнала ошибки. Показано, что предложенный алгоритм по сравнению с алгоритмом МНК имеет меньшие дисперсии сигнала ошибки и оценки градиента, меньшее время сходимости, а также обеспечивает большую устойчивость к таким мешающим факторам, как аддитивные помехи, изменение мощности и динамического диапазона собственных чисел корреляционных матриц входных сигналов.

Введение

В ряде радиотехнических систем широко применяются устройства осуществляющие обработку сигналов в адаптивных цифровых фильтрах [1, 2]. Такие устройства обеспечивают эффективное функционирование систем в условиях априорной неопределенности о сигнально-помеховой обстановке. Основными требованиями, предъявляемыми к адаптивным цифровым фильтрам, являются получение минимально возможных времени адаптации и сигнала ошибки на выходе, а также устойчивость к мешающим факторам. К мешающим факторам, существенно влияющим на результаты синтеза, следует отнести аддитивные помехи, а также изменение мощности и динамического диапазона собственных чисел корреляционных матриц входных сигналов [2]. Как известно, наиболее простым и в то же время эффективным алгоритмом адаптации коэффициентов цифровых фильтров является алгоритм, построенный на основе МНК. Однако в ряде случаев [1] алгоритмы МНК могут иметь неквадратичную рабочую функцию с локальными минимумами и областями неустойчивости, а также обладать высокой чувствительностью к перечисленным выше мешающим факторам.

В [3, 4] были предложены алгоритмы оптимизации коэффициентов цифровых фильтров на основе блоковых оценок по комбинированному критерию среднего квадрата ошибки (СКО), включающего в себя взвешенные средний квадрат сигнала ошибки и средний квадрат изменения сигнала ошибки. Как было показано использование этих алгоритмов позволило увеличить устойчивость процесса адаптации, уменьшить динамический диапазон и дисперсию сигнала ошибки, снизить влияние мешающих факторов. Рассмотрим возможность использования предложенного в [3, 4] подхода к синтезу алгоритмов оптимизации коэффициентов адаптивных цифровых фильтров на основе комбинированного МНК.

Цель работы – исследовать свойства алгоритмов оптимизации коэффициентов адаптивных цифровых фильтров, синтезированных на основе комбинированного МНК.

Постановка задачи

В адаптивных цифровых устройствах с ограниченной апертурой желательно использовать алгоритмы на основе минимизации динамического диапазона сигнала ошибки, а не алгоритм МНК. Однако минимизация функционалов, связанных с абсолютной величиной ошибки на выходе адаптивного фильтра, приводит к необходимости использования сложных численных методов, не всегда сходящихся к устойчивым решениям. По этой причине целесообразно исследовать свойства алгоритмов, синтезированных на основе комбинированного МНК, учитывающего ограничение на изменение сигнала ошибки.

Рассмотрим задачу идентификации неизвестных параметров формирующего фильтра (ФФ) при заданных параметрах адаптивного фильтра (АФ). Пусть на входы ФФ и АФ поступает процесс, представляющий собой реализацию белого гауссова шума $x(n)$ (*рис. 1.*) На выходной сигнал ФФ $d(n)$ аддитивно воздействует помеха $\nu(n)$ в виде белого гауссова шума. На основе разностного сигнала $e(n)$ по комбинированному МНК осуще-

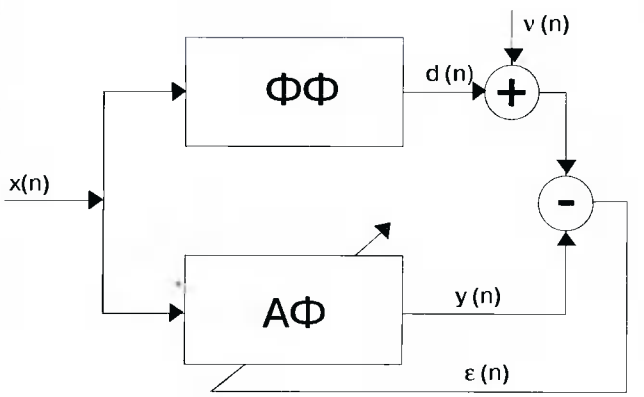


Рис. 1

ствляется подстройка параметров АФ под параметры ФФ.

В случае комбинированного МНК минимизации подвергается функционал вида:

$$J = [((1 - z)e_k + z\,\Delta e_{k-1}]^2, \qquad (1)$$

здесь $\Delta e_k = e_k - e_{k-1} = d_k - d_{k-1} - (y_k - y_{k-1})$, d_k , – эталонный сигнал.

Выходной сигнал рекурсивного АФ определяется соотношением

$$y_k = \sum_{p=0}^{P} a_p x_{k-p} + \sum_{q=1}^{Q} b_q y_{k-q} = W_k^T U_k, \qquad (2)$$

где $W_k = [a_{0k}\, a_{1k} \ldots a_{Pk}\, b_{1k} \ldots b_{Qk}]^T$ – вектор весовых коэффициентов адаптивного фильтра, $U_k = [x_k\, x_{k-1} \ldots x_{k-P}\, y_k \ldots y_{k-Q}]$ – вектор сигнала, P и Q – порядок рекурсивной и нерекурсивной ветвей фильтра соответственно. Алгоритм адаптации весовых коэффициентов определяется выражением

$$W_{k+1} = W_k - M\hat{\nabla}_k, \qquad (3)$$

где $\hat{\nabla}_k$ – оценка градиента на k-м шаге, $M = \mathrm{diag}[\mu_0 \ldots \mu_P \nu_1 \ldots \nu_Q]$ – матрица, содержащая параметры сходимости по каждой координате. В алгоритме МНК для оценки градиента используется само значение ошибки e_k^2 [1]. С учетом (2) и (3) минимизируемый функционал примет вид:

$$J = (d_k - W_k^T U_k - z\,(d_{k-1} - W_{k-1}^T U_{k-1}))^2. \qquad (4)$$

Оценка градиента на каждой итерации в случае рабочей функции подобного вида будет определяться соотношением

$$\nabla_k = \frac{\partial J}{\partial W_k} = \frac{\partial (d_k - W_k^T U_k - z(d_{k-1} - W_{k-1}^T U_{k-1}))^2}{\partial W_k} =$$

$$= -2[e_k - z e_{k-1}] \cdot \frac{\partial [W_k^T U_k - z(W_k^T + M\nabla_{k-1}) U_{k-1}]}{\partial W_k}. \qquad (5)$$

Вычисление входящих в выражение (5) производных производится по следующим рекуррентным формулам

$$\alpha_{m,k} = \frac{\partial W_k^T U_k}{\partial W_k} = x_{k-m} + \sum_{i=1}^{Q} b_{ik}\alpha_{m,k-i},$$

$$\Delta\alpha_{m,k} = \frac{\partial W_k^T U_{k-1}}{\partial W_k} = x_{k-1-m} + \sum_{i=1}^{Q} b_{ik}\Delta\alpha_{m,k-i}, \qquad (6)$$

$$\beta_{m,k} = \frac{\partial W_k^T U_k}{\partial b_{nk}} = y_{k-m} + \sum_{i=1}^{Q} b_{ik}\beta_{m,k-i},$$

$$\Delta\beta_{m,k} = \frac{\partial W_k^T U_{k-1}}{\partial b_{nk}} = y_{k-1-m} + \sum_{i=1}^{Q} b_{ik}\Delta\beta_{m,k-i}.$$

Если представить входящие в (6) соотношения в матричной форме

$$\mathrm{AB}_k = [\alpha_{0,k}\alpha_{1,k} \ldots \alpha_{p,k}\beta_{1,k} \ldots \beta_{q,k}]^T,$$

$$\Delta\mathrm{AB}_k = [\Delta\alpha_{0,k}\Delta\alpha_{1,k} \ldots \Delta\alpha_{p,k}\Delta\beta_{1,k} \ldots \Delta\beta_{q,k}]^T, \qquad (7)$$

то выражение (5) можно привести к виду:

$$\nabla_k = -2[e_k - z e_{k-1}]\left[\mathrm{AB}_k - z\Delta\mathrm{AB}_k - \frac{zM\partial(\nabla_{k-1}U_{k-1})}{\partial W_k}\right] \qquad (8)$$

Если предположить, что изменение градиента ∇_{k-1} относительно значений коэффициентов W_k незначительно, то выражение (8) можно представить следующим образом:

$$\nabla_k^{(1)} = -2[\varepsilon_k - z\varepsilon_{k-1}][\mathrm{AB}_k - z\Delta\mathrm{AB}_k]. \qquad (9)$$

Одним из путей снижения вычислительных затрат является вариант использования неполной формы оценки градиента (9) в соответствии со следующими формулами:

$$\nabla_k^{(2)} = -2\mathrm{AB}_k[e_k - z e_{k-1}], \qquad (10)$$

$$\nabla_k^{(3)} = -2e_k[\mathrm{AB}_k - z\Delta\mathrm{AB}_k]. \qquad (11)$$

Необходимо отметить, что оценки градиента (10), (11) были предложены в [1] на основе экспериментальных исследований в целях повышения устойчивости алгоритмов. Как нетрудно заметить, эти оценки являются частными случаями более общего выражения (9).

Результаты математического моделирования

Экспериментальные исследования синтезированных алгоритмов осуществлялись в соответствии со схемой, изображенной на *рис. 1*, при вычислении оценок градиента по формулам (9)...(11). В качестве показателей качества адаптации выступали дисперсия и модуль максимального отклонения разности сигналов АФ и ФФ, время сходимости алгоритма (по заданному уровню от начального состояния). Кроме того, исследовалась чувствительность синтезированных алгоритмов к таким мешающим факторам, как влияние адаптивной помехи $\nu(n)$, а также изменение мощности и динамического диапазона собственных чисел корреляционных матриц входного сигнала $x(n)$.

На *рис. 2* для случая оценки градиента (9) представлены зависимости нормированной дисперсии (σ/σ_0) и модуля максимального отклонения (ММО) ошибки (Δ/Δ_0) от значения параметра z в момент времени, соответствующей 10% ошибке для стандартного алгоритма МНК, здесь σ_0 и Δ_0 дисперсии и модуль максимально-

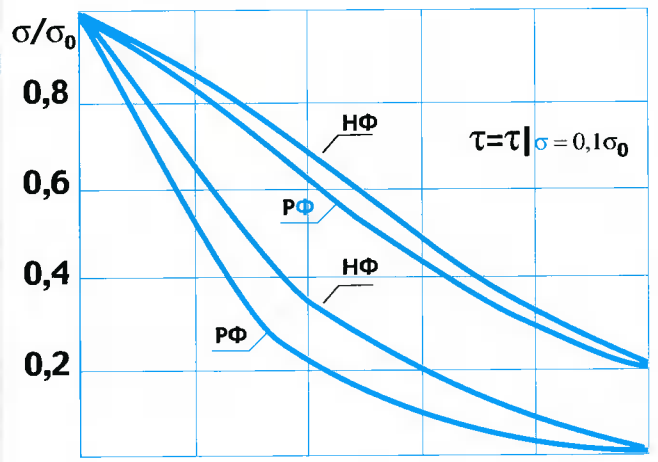


Рис. 2

го отклонения при z=0. Графики получены усреднением результатов 100 обучающих кривых для нерекурсивных фильтров (НФ) порядка P = 6 и рекурсивных фильтров (РФ) порядка Q = 2. Графики для нормированного ММО ошибки соответствуют наихудшим значениям из 100 реализаций. Анализ показывает возможность значительного, в 10 раз снижения дисперсии ошибки и в 5 раз ММО. При этом увеличивается не менее чем в 2,5 раза точность определения коэффициентов АФ, оцениваемая как среднеквадратическое отклонение $\eta = \sum \Delta W^2 / \sum W^2$ найденных коэффициентов от истинных, где ΔW – отклонение найденных коэффициентов от истинных.

На *рис. 3* приведены зависимости нормированного времени сходимости алгоритма (τ/τ_0), где τ – время сходимости комбинированного алгоритма МНК, определяемое по уровню 10% дисперсии ошибки стандартного МНК, τ_0 – время сходимости стандартного алгоритма.

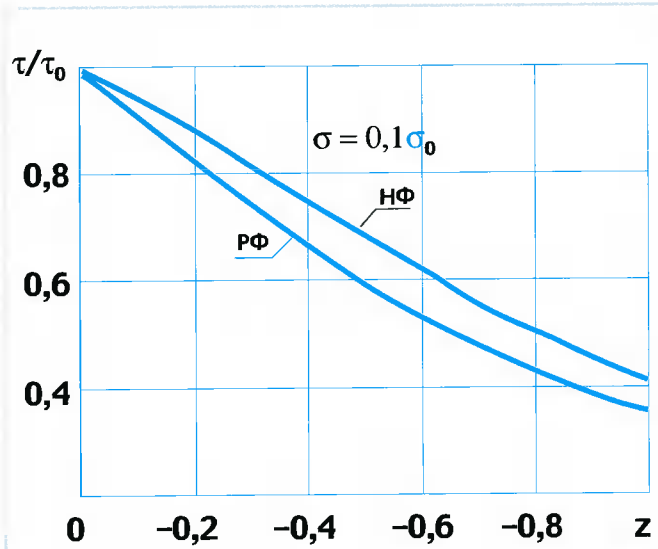


Рис. 3

Из анализа *рис. 3* следует, что применение комбинированного алгоритма МНК позволяет уменьшить время сходимости почти в 2,5 раза.

Исследования влияния на характеристики предложенного алгоритма аддитивной помехи ν(n) при изменении отношения сигнал – шум от 50 до 15 дБ показали, что при этом отношение $\sigma/\sigma_0 \approx 4$ и уменьшается не более чем на 15%, что позволяет говорить о робастности синтезированного алгоритма (9).

Кроме того, показана возможность уменьшения в 1,5...3,5 раза параметра сходимости μ при заданном значении τ и дисперсии оценки градиента в 1,1 раза. Это увеличивает устойчивость и точность процесса адаптации [1]. Изменение мощности и динамического диапазона собственных чисел корреляционной матрицы входного сигнала x(n) на 20 дБ при использовании алгоритма (9) приводило к увеличению дисперсии сигнала ошибки в 1,5...2 раза меньшему, чем для алгоритма МНК.

Как показали исследования, алгоритмы (10), (11) проигрывают алгоритму (9) не более чем на 15...20%, как по точности оценок параметров, так и по скорости сходимости. Реализация алгоритма (9) для адаптации коэффициентов АФ потребует 7P+15Q+9 операций сложения-умножения, что примерно в 1,5 раза больше, чем для стандартного алгоритма МНК (5P+9Q+5).

Таким образом, показана высокая эффективность и низкая чувствительность к мешающим факторам комбинированного алгоритма МНК, учитывающего первую конечную разность сигнала ошибки на выходе адаптивного цифрового фильтра. Алгоритм (9) по своей эффективности приближается к алгоритму последовательной регрессии (2,5P²+4P операций сложения-умножения [2]), обладая при этом значительно меньшими вычислительными затратами.

Литература

1. Уидроу Б., Стирнз С. Адаптивная обработка сигналов. М.: Радио и связь, 1989, 440 с.
2. Адаптивные фильтры. / Под ред. Коуэна К.Ф.Н., Гранта П.М. М.: Мир, 1988, 392 с.
3. Кириллов С.Н., Степанов М.В. Комбинированный критерий оптимизации коэффициентов адаптивных фильтров с конечной импульсной характеристикой. // Радиотехника. 1999, № 2, с. 39 – 41.
4. Кириллов С.Н., Степанов М.В. Реализация устройств цифровой обработки сигналов по комбинированному критерию среднего квадрата ошибки. // Цифровая обработка сигналов, 1999, № 2.
5. Кириллов С.Н., Шустиков О.Е., Степанов М.В., Виноградов О.Л. Синтез адаптивных цифровых устройств по заданным показателям качества. // 2-я Международная конференция "Цифровая обработка сигналов и ее применение". Тез. докл. Т.1. М.: МЦНТИ 1999, с.262 – 265.

В.Г. Санников

Статистическая идентификация модели речевого сигнала в пространстве состояний

В работе [1] исследовалась задача статистической идентификации параметров АР-модели голосового тракта источника речеобразования. Однако более адекватно акустику речеобразования можно описать, если использовать непараметрические модели, основу которых составляет модель речеобразования в пространстве состояний [2].

В данной работе исследуется проблема синтеза модели речеобразования в пространстве состояний путем анализа реализаций наблюдаемого речевого сигнала. Эта проблема в настоящее время изучается многими исследователями и достаточно успешно реализуется в высококачественных кодеках речи, получивших название в английской литературе как: a Class of Analysis-by-Syntchesis Predictive Coders for High Quality Speech Coding [3].

Решение указанной проблемы осуществляется на основе синтеза следящей системы с обратной связью по ошибке выхода, удовлетворяющей обобщенному критерию гиперустойчивости Попова.

Постановка задачи

Рассмотрим вначале упрощенную модель речеобразования Фанта (*рис.1*), содержащую голосовой тракт (ГТ), на вход которого поступает сигнал от источника голосового возбуждения (ИГВ).

Рис. 1. Упрощенная модель речеобразования Фанта

Здесь $u(t)$ – входное воздействие ГТ, называемое управляющим воздействием, $s(t)$ – отклик ГТ, доступный наблюдению. В реальных условиях функционирования ГТ нестационарен и имеет стохастическую динамику. Отклик ГТ – это речевой сигнал, представляющий собой случайный процесс с ограниченной дисперсией.

Предположим, что математическая модель голосового тракта, подлежащего идентификации, описывается в дискретном времени стандартной канонической формой Коши [4]

$$s_k = A_k s_{k-1} + B_k u_{k-1}, \quad k = 0,1,2\ldots, \qquad (1)$$

с начальными условиями s_0 и u_0. Здесь s_k и u_{k-1} – выходной и входной ($n\times 1$) – векторы ГТ, в общем случае это выборки случайных процессов, реализуемые в k и $k-1$ моменты времени; A_k и B_k – вещественные ($n\times n$) матрицы, медленно изменяющиеся во времени, элементы которых ограничены по значениям. Полагаем, что при любых k матрица управления B_k является невырожденной.

Используя методику работы [5], рассмотрим задачу идентификации модели (1) по результатам слежения за вектором наблюдений s_k. С этой целью, полагая, что на начальном интервале обучения вектор управления u_{k-1} известен, для вектора s_k сформируем вектор прогноза

$$p_k = A^*_{k-1} s_{k-1} + B^*_{k-1} u_{k-1}, \qquad (2)$$

где A^*_{k-1} и B^*_{k-1} – оценки в $k-1$ момент времени матриц A_{k-1} и B_{k-1} исследуемого ГТ, p_k – выходной сигнал фильтра-предсказателя модели ГТ, рассматриваемой в виде некоторой системы слежения за динамикой сигнала s_k, $k = 1, 2\ldots$

Если на начальном интервале наблюдения матрицы A^*_{k-1} и B^*_{k-1} каким-либо образом определены (вопрос о методике их нахождения будет рассмотрен ниже), то в соответствии с соотношением (2) получаем следующий алгоритм синтеза сигнала управления в блоке регулятора синтезируемой модели ГТ

$$u_{k-1} = (B^*_{k-1})^{-1}(p_k - A^*_{k-1} s_{k-1}), \qquad (3)$$

где p_k – входное воздействие, а u_{k-1} – отклик регулятора.

Вычитая из наблюдаемого сигнала s_k вектор прогноза p_k, сформируем вектор погрешности предсказания

$$e_k = s_k - p_k. \qquad (4)$$

Найдем матричные параметры A^*_k и B^*_k фильтра-предсказателя путем минимизации среднеквадратической погрешности предсказания $\mathrm{M}e_k^{tr}e_k = \mathrm{M}||e_k||^2$, где M – знак математического ожидания, $||.||$ – знак нормы вектора, tr – знак транспонирования вектора.

Допустим, что матрицы A^*_k и B^*_k удовлетворяют следующим рекуррентным соотношениям

$$A^*_k = A^*_{k-1} + \Delta^A_k, \quad B^*_k = B^*_{k-1} + \Delta^B_k, \qquad (5)$$

где Δ^A_k и Δ^B_k – малые приращения отслеживаемых матриц идентифицируемого голосового тракта.

Воспользовавшись этими оценками, сформируем оценку вектора наблюдений

$$s^*_k = A^*_k s_{k-1} + B^*_k u_{k-1}. \qquad (6)$$

Вычитая из наблюдаемого вектора s_k его оценку s^*_k, находим вектор погрешности выхода системы слежения

$$\varepsilon_k = s_k - s^*_k. \qquad (7)$$

Теперь математическую модель голосового тракта, получаемую по результатам слежения за его наблюдениями s_k, $k = 1,2...$, с учетом соотношений (5) – (7), можно представить в виде

$$s_k = A^*_k s_{k-1} + B^*_k u_{k-1} + \varepsilon_k. \qquad (8)$$

Здесь ε_k – вектор возмущающих воздействий, как правило, не полностью известная вектор-функция, реализация случайного процесса, относительно которой известны чаще всего лишь ее вероятностные характеристики. Например, это может быть вектор гауссовских независимых величин.

Блок-схема исследуемой системы слежения, построенная в соответствии с данными предпосылками, приведена на *рис.2*.

Здесь А/Д – блок аналого-дискретного представления отклика ГТ, входом которого является речевой сигнал $s(t)$, а откликом векторная последовательность s_k; ОПИ – оценщик погрешности выхода ε_k системы слежения (СС); ФП – фильтр-предсказатель; Р – регулятор; ЛЗ – многоотводная линия задержки.

Данная система относится к классу замкнутых систем с обратной связью по сигналу погрешности слежения ε_k. Важнейшим свойством такого вида систем является требование их устойчивости [4 – 8].

Задача, которую требуется решить, заключается в отыскании такого адаптивного алгоритма оценки матриц перехода A^*_k и управления B^*_k, который позволяет проверить, обладает ли система слежения требуемыми свойствами устойчивости и при каких условиях.

Адаптивная оценка параметров системы слежения

Рассмотрим задачу адаптивного управления матричными параметрами A^*_k и B^*_k следящей системы, основываясь на критерии гиперустойчивости Попова [5]. Применительно к детерминированным дискретным по времени системам управления этот критерий гласит: *система с обратной связью асимптотически гиперустойчива, если погрешность ε_k (ошибка слежения) удовлетворяет условию:*

$$\sum_{k=0}^{k1} \varepsilon_k^{tr} \varepsilon_k = \sum_{k=0}^{k1} ||\varepsilon_k||^2 \le \delta^2, \ \delta = \text{const} \ \forall k1. \qquad (9)$$

Так как голосовой тракт относится к классу стохастических систем, то использование критерия (9) невозможно из-за случайности процесса $s(t)$, вектора наблюдений s_k, а следовательно, и случайности вектора ε_k. Обобщением критерия (9) на стохастический случай является применение операции усреднения по множеству реализаций для входящих в него компонент. Кроме того, для учета различного влияния разных компонент вектора ε_k на среднеквадратическую погрешность идентификации модели (1) рационально ввести в критерий устойчивости неотрицательно определенную известную весовую $(n \times n)$-матрицу V_k.

Теперь критерий устойчивости для рассматриваемой задачи можно сформулировать так: *если система слежения с обратной связью удовлетворяет условию*

$$M\{\sum_{k=0}^{k1} \varepsilon_k^{tr} V_k \varepsilon_k\} = \sum_{k=0}^{k1} M||\varepsilon_k||^2_V \le \delta_0^2, \ \delta_0 = \text{const} \ \forall k1, \qquad (10)$$

то она будет асимптотически стохастически гиперустойчивой.

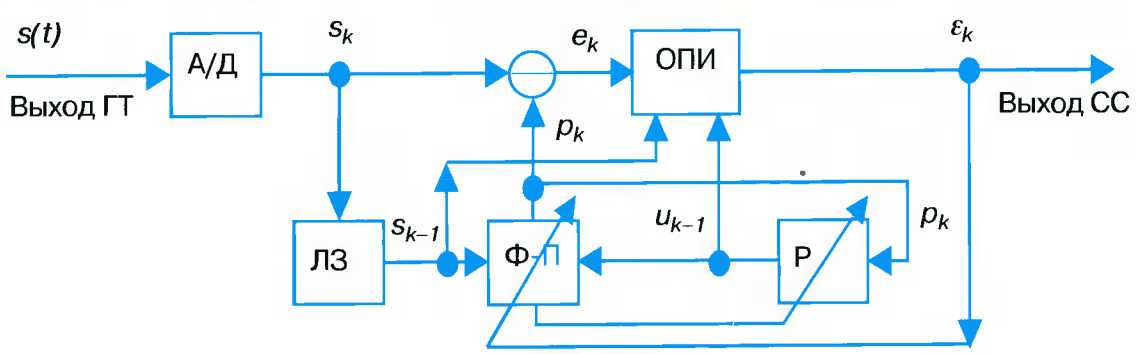


Рис.2. Блок-схема системы слежения

Здесь $||\varepsilon_k||_V$ – норма вектора в евклидовом пространстве с весом V.

Соотношение (10) характеризует ограничение, накладываемое на норму случайной последовательности $\nu_k = \varepsilon_k^{tr} V_k \varepsilon_k$, $k = 0, 1, 2,...$, определяемую в координатном гильбертовом пространстве l_2 случайных последовательностей.

Основываясь на данном критерии, сформулируем требования для оценки матриц приращений Δ^A_k и Δ^B_k, введенных в соотношении (5). С этой целью, учитывая соотношения (1), (6) и (7), находим

$$\varepsilon_k = (A_k - A^*_k)s_{k-1} + (B_k - B^*_k)u_{k-1}, \quad k = 0,1,2... \quad (11)$$

Умножая $(-\varepsilon_k^{tr})$ на $V_k\varepsilon_k$, усредняя это произведение и подставляя в соотношение (10), приходим к следующему неравенству

$$\sum_{k=0}^{k1} M\{\varepsilon_k^{tr} V_k (A^*_k - A_k)s_{k-1} + \varepsilon_k^{tr} V_k (B^*_k - B_k)u_{k-1}\} \geq -\delta_0^2. \quad (12а)$$

Из (10) и (12а) следует, что при положительных и отрицательных матрицах приращений $(A^*_k - A_k) = \Delta^A_k$ и $(B^*_k - B_k) = \Delta^B_k$ справедливо условие

$$-\delta_0^2 \leq \sum_{k=0}^{k1} M\{\varepsilon_k^{tr} V_k \Delta^A_k s_{k-1} + \varepsilon_k^{tr} V_k \Delta^B_k u_{k-1}\} \leq +\delta_0^2 \quad \forall\, k1. \quad (12б)$$

Данное неравенство можно усилить, если матрицы приращений Δ^A_k и Δ^B_k построить в виде следующих произведений

$$\Delta^A_k = \varepsilon_k s^{tr}_{k-1} V_k, \quad \Delta^B_k = \varepsilon_k u^{tr}_{k-1} V_k. \quad (13)$$

Подставляя (13) в (12б), приходим к следующему неравенству

$$\sum_{k=0}^{k1} M\{\varepsilon_k^{tr} V_k \varepsilon_k (s^{tr}_{k-1} V_k s_{k-1} + u^{tr}_{k-1} V_k u_{k-1})\} =$$

$$= \sum_{k=0}^{k1} M||\varepsilon_k||^2_V (||s_{k-1}||^2_V + ||u_{k-1}||^2_V) \leq \delta_0^2 \quad (14)$$

Итак, можно сделать вывод о том, что рассматриваемая система слежения с обратной связью будет асимптотически статистически гиперустойчивой, если матрицы A^*_k и B^*_k оцениваются в соответствии с соотношениями

$$A^*_k = A^*_{k-1} + \Delta^A_k = A^*_{k-1} + \varepsilon_k s^{tr}_{k-1} V_k, \quad (15)$$

$$B^*_k = B^*_{k-1} + \Delta^B_k = B^*_{k-1} + \varepsilon_k u^{tr}_{k-1} V_k, \quad (16)$$

где при малых приращениях учтены следующие приближения

$$\Delta^A_k = (A^*_k - A_k) \cong (A^*_k - A^*_{k-1}),$$

$$\Delta^B_k = (B^*_k - B_k) \cong (B^*_k - B^*_{k-1}).$$

Для практической реализации данных соотношений необходимо определить правило расчета ошибки слежения ε_k. С этой целью на основе (2) – (6), а также (13) преобразуем соотношение (7) к виду

$$\varepsilon_k = s_k - A^*_k s_{k-1} - B^*_k u_{k-1} = e_k - (\Delta^A_k s_{k-1} + \Delta^B_k u_{k-1}) =$$

$$e_k - \varepsilon_k (s^{tr}_{k-1} V_k s_{k-1} + u^{tr}_{k-1} V_k u_{k-1}),$$

где $e_k = s_k - (A^*_{k-1} s_{k-1} + B^*_{k-1} u_{k-1})$.

Из данного соотношения после ряда преобразований находим следующую взаимосвязь между погрешностями выхода и прогнозирования системы слежения

$$\varepsilon_k = c_k e_k, \quad (17)$$

где c_k – изменяющаяся во времени скалярная величина, равная показателю взаимосвязи между рассматриваемыми погрешностями и определяемая соотношением

$$c_k = (1 + s^{tr}_{k-1} V_k s_{k-1} + u^{tr}_{k-1} V_k u_{k-1})^{-1} = (1 + ||s_{k-1}||^2_V + ||u_{k-1}||^2_V)^{-1}. \quad (18)$$

Покажем теперь, что матрицы приращений Δ^A_k и Δ^B_k, определяемые соотношением (13), приводят к статистической некоррелированности векторов s_{k-1}, u_{k-1} с ε_k. С этой целью найдем средние величины $Ms^{tr}_{k-1}\varepsilon_k$ и $Mu^{tr}_{k-1}\varepsilon_k$. На основе (11) и (13) после несложных преобразований приходим к следующим условиям:

$$Ms^{tr}_{k-1}\varepsilon_k \times M(||s_{k-1}||^2_V + ||u_{k-1}||^2_V - 1) = 0,$$

$$Mu^{tr}_{k-1}\varepsilon_k \times M(||s_{k-1}||^2_V + ||u_{k-1}||^2_V - 1) = 0.$$

Для реальных речевых сигналов эти равенства выполняются тогда и только тогда, когда удовлетворяются условия:

$$Ms^{tr}_{k-1}\varepsilon_k = 0, \quad Mu^{tr}_{k-1}\varepsilon_k = 0,$$

но это и доказывает сделанное выше утверждение о статистической некоррелированности векторов s_{k-1} и u_{k-1} с ε_k.

Сходимость оцениваемых параметров

Сходимость матричных параметров A^*_k и B^*_k системы слежения к параметрам A_k и B_k модели голосового тракта проверим на основе теоремы о сходимости систем при выборочных средних [7]. Для этого вначале выведем уравнения рассогласования параметров моделей (1) и (6). Введем вместо матриц $A_k = [a_{k,ij}]$, $B_k = [b_{k,ij}]$ и

$A^*_k = [a^*_{k,ij}]$, $B^*_k = [b^*_{k,ij}]$ обобщенные векторные параметры

$$d^{tr}_{k,i} = (a_{k,i1}, a_{k,i2}, ..., a_{k,in}; b_{k,i1}, b_{k,i2}, ..., b_{k,in}),$$

$$i = 1...2n , \qquad (19а)$$

$$d^{*tr}_{k-1,i} = (a^*_{k-1,i1}, a^*_{k-1,i2}, ..., a^*_{k-1,in};$$

$$b^*_{k-1,i1}, b^*_{k-1,i2}, ..., b^*_{k-1,in}), \; i = 1...2n. \qquad (19б)$$

Определим регрессионный и обобщенный регрессионный векторы

$$x^{tr}_{k-1} = (s_{k-1,1}, s_{k-1,2}, ..., s_{k-1,n}; u_{k-1,1}, u_{k-1,2}, ..., u_{k-1,n}), \qquad (20а)$$

$$y^{tr}_{k-1} = x^{tr}_{k-1} W_k , \qquad (20б)$$

где W_k – составная матрица размера $(2n \times 2n)$, на главной диагонали которой стоят весовые матрицы V_k, а внедиагональные матрицы нулевые.

С учетом данных обозначений формулы (15), (16) приводятся к единому виду

$$d^*_{k,i} = d^*_{k-1,i} + \varepsilon_{k,i}\, y_{k-1} = d^*_{k-1,i} + c_k\, y_{k-1}(s_{k,i} - x^{tr}_{k-1} d^*_{k-1,i}) = d^*_{k-1,i} + c_k\, y_{k-1} x^{tr}_{k-1}(d_{k,i} - d^*_{k-1,i}). \qquad (21)$$

Уравнение (21) представляет собой пример обобщенного градиентного алгоритма [8]. Для его решения обозначим через $\Delta_{k,i} = (d_{k,i} - d^*_{k,i})$, $i = 1...2n$, – расхождения параметров или ошибки идентификации моделей (1) и (6). Тогда с учетом того, что $(d_{k,i} - d^*_{k-1,i}) = \Delta_{k-1,i}$ и $(d^*_{k,i} - d^*_{k-1,i}) = (\Delta_{k-1,i} - \Delta_{k,i})$, уравнения рассогласования моделей (1) и (6) можно представить в виде следующей разностной процедуры

$$\Delta_{k+1,i} = (I_{2n} - c_k\, y_k\, x^{tr}_k)\Delta_{k,i} = \left(I_{2n} - \frac{W_k^{tr} x_k x^{tr}_k}{1 + x^{tr}_{k-1} W_k x_{k-1}}\right)\Delta_{k,i} = F_k\, \Delta_{k,i} \qquad (22)$$

с переходной матрицей $\Phi_{k+n,k} = F_{k+n-1} \times F_{k+n-2} \times ... F_{k+1} \times F_k$. В (22) I_{2n} – единичная матрица размера $2n \times 2n$.

При $W = I_{2n}$ в работе [7] показано, что если идентифицируемый объект детерминирован с постоянными, но первоначально неизвестными векторами параметров, то за исключением одного, равного

$$1 - (x^{tr}_{k-1} x_{k-1}) \times (1 + x^{tr}_{k-1} x_{k-1})^{-1} = 1 - ||x_{k-1}||^2 \times (1 + ||x_{k-1}||^2)^{-1},$$

остальные $(2n-1)$ собственные значения матрицы F_k равны 1. Очевидно, при весовой матрице W, отличной от единичной, указанное собственное значение будет равно: $1 - ||x_{k-1}||^2_W / (1 + ||x_{k-1}||^2_W)$. Оно, как нетрудно заметить, лежит в интервале (0,1). Отсюда следует сходимость данного градиентного алгоритма идентификации. Действительно, $x^{tr}_k \Delta_{k,i} \to 0$ при $k \to \infty$, т.е. разность между реальным речевым сигналом, формируемым голосовым трактом, и его оценкой, вырабатываемой следящей системой, стремится к нулю независимо от x_k . При этом, если для некоторого положительного α , целого K и всех целых l выполняется неравенство

$$\sum_{k=l}^{l+K} x_k x_k^{tr} > \alpha I_{2n} , \qquad (23)$$

то $||\Delta_{k,i}||$ стремятся к нулю экспоненциально [7].

Прежде чем проанализировать свойства сходимости линейной нестационарной модели (22), заметим, что, хотя векторы x_k и y_k, а следовательно, и матрица $R_k = y_k\, x^{tr}_k$ изменяются достаточно быстро, векторные параметры $\Delta_{k,i}$, $i = 1,..., 2n$, изменяются медленно, если R_k есть последовательность ограниченных $(2n \times 2n)$-матриц, а значение c_k достаточно мало. Малость величины c_k подтверждается соотношением (18). Этот факт позволяет сформулировать условия сходимости модели (22) в терминах "выборочных средних" [7]

$$\langle R \rangle_l = \frac{1}{K_l} \sum_{k=k1}^{kl-1} R_k , \quad kl = k1 + K_l \quad \forall\, k1 , \qquad (24)$$

по конечным интервалам

$$0 < K_m \le K_l \le K_M < \infty \quad \forall\, l \subset Z . \qquad (25)$$

Теперь можно сформулировать достаточное условие экспоненциальной сходимости модели (22). Если для последовательности выборочных средних $\langle R \rangle_l$, определенных для всех $l \subset Z$ соотношением (24), существует постоянная положительно определенная матрица $P = P^{tr}$, удовлетворяющая неравенству

$$P \langle R \rangle_l + \langle R^{tr} \rangle_l P \ge I_{2n} , \quad \forall\, l \subset Z, \qquad (26)$$

то найдется $c^* > 0$, такое, что модель (22) будет экспоненциально устойчива при всех c_k $(0, c^*)$.

В работе [7] доказывается, что свойство сходимости модели (22) эквивалентно неравенству $\Delta^{tr}_{l,i} P \Delta^{tr}_{l,i} < \Delta^{tr}_{l-1,i} P \Delta^{tr}_{l-1,i}$, $i = 1,..., 2n$, " $l \subset Z$. Причем данное неравенство выполняется, если величина c^* удовлетворяет условию: $c^* = \inf p_l^{-1} > 0$, где $p_l = 7||P||\rho_l^2$. Здесь $||P||$ – норма матрицы P, а величина ρ_l определяется так

$$\rho_l = (K_l)^{-1} \sum_{k=k1}^{kl-1} ||R_k|| , \quad kl = k1 + K_l . \qquad (27)$$

В результате показывается, что последовательность равномерно ограниченных сжимающих отображений $\langle\Phi\rangle_l$ при всех l удовлетворяет неравенству

$$||\langle\Phi\rangle_l|| \le \exp(-c^*) \quad \forall\, l \subset Z, \qquad (28)$$

и, следовательно, исследуемая модель (22) сходится экспоненциально.

В (25) переходная матрица $\langle\Phi\rangle_l = \Phi_{kl,\,k1}$, $kl = k1 + K_l$, связана со средней выборочной матрицей $\langle R\rangle_l$ соотношением

$$\langle\Phi\rangle_l = I_{2n} - \alpha K_l \langle R\rangle_l + (\alpha K_l)^2 M_l, \qquad (29)$$

где при выполнении условия: $|\alpha| K_l \rho_l \le 0.5$, для M_l справедлива оценка $M_l \le 2\rho_l^2$.

Структурная схема системы адаптивного слежения

Подробная структурная схема рассматриваемой системы при $V_k = I_n$ приведена на *рис.3*. Она содержит блоки, соответствующие блок-схеме на *рис.2*, но с более конкретной детализацией, включающей такие элементы, как векторные вычитающие устройства, линии задержки (z^{-1}), сумматоры, перемножители, делители. В данной схеме введены обозначения: БНО – блок начальных оценок, ПРР – переключатель режимов работы, А/Д – блок аналого-дискретного представления отклика голосового тракта, ОПИ – оценщик погрешности ε_k выхода системы слежения (погрешности идентификации), Р – регулятор.

Можно выделить три последовательных режима работы данной системы:

0 – режим формирования начальных условий;
1 – режим идентификации параметров голосового тракта;
2 – режим управления (идентификационный режим синтеза следящей системы, отображающей эквивалентную модель речеобразования).

На начальной стадии работы системы (ПРР→0) требуется оценить момент $k = 0$, для которого формируются матрицы A_0, B_0 и вектор управления u_0. Полагая $B_0 = I_n$ единичной матрицей размера (n×n), а $u_0 = 0$ нулевым вектором, для оценки матрицы A_0 воспользуемся упрощенной АР-моделью ГТ. Применяя разработанный автором в работе [1] метод идентификации АР-модели, находим начальные оценки параметров $\{a_i\}$. На основе полученного век-

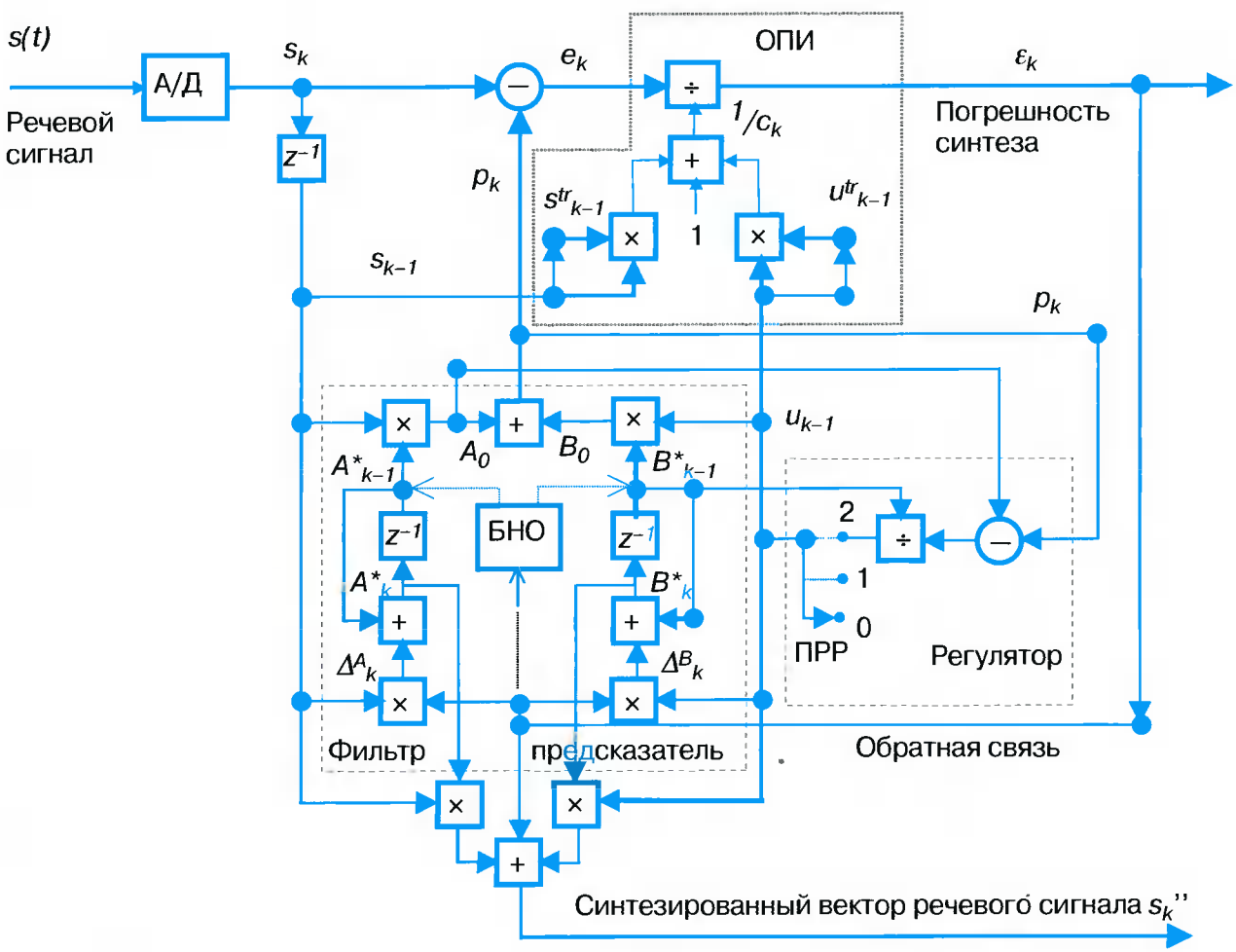


Рис. 3. Структурная схема системы синтеза речевого сигнала через анализ

тора параметров $a^{tr} = (a_1, a_2,..., a_n)$ начальная матрица строится так

$$A_0 = \begin{bmatrix} 0 & & & \\ 0 & & I_{n-1} & \\ ... & & & \\ 0 & & & \\ a_n & a_{n-1} & ... & a_1 \end{bmatrix}.$$

Здесь I_{n-1} – единичная матрица размера $(n-1)\times(n-1)$.

Как только данная матрица сформирована, числу k_0 сопоставляется число $k = 0$, и фиксируется начальный вектор наблюдений $s^{tr}_0 = (s_{01}, s_{02},..., s_{0n})$.

В режиме идентификации (ПР→1) осуществляется рекуррентная оценка матриц A^*_k и B^*_k для $k>0$. Здесь вектор управления полагается единичным $u^{tr}_k = (1, 1,..., 1)$. Дальнейшая процедура идентификации модели ГТ осуществляется по рабочим циклам в реальном масштабе времени и в каждый k-й момент состоит из следующих шагов:

- формирование единичных векторов управления $u^{tr}_k = (1, 1,..., 1)$;
- измерение векторов наблюдений s_{k-1} и s_k;
- вычисление вектора прогноза p_k по (2);
- вычисление вектора погрешности предсказания e_k по (4);
- вычисление вектора погрешности идентификации ε_k по (17) с учетом оценки коэффициента c_k по (18);
- вычисление матриц A^*_k и B^*_k в соответствии с соотношениями (15) и (16).

Полагая, что процесс адаптации сходится, режим идентификации прекращается в тот момент, когда нормы разностных матриц $||\Delta^A_k||$ и $||\Delta^B_k||$ достигают некоторой пороговой величины, определяемой критерием точности идентификации.

В режиме управления (синтеза сигнала голосового возбуждения u^*_k) рабочий цикл в каждый k-й момент времени состоит из следующих шагов:

- на момент окончания режима идентификации фиксируются данные: s_{k-1}, A^*_{k-1}, B^*_{k-1}, p_k и данному моменту сопоставляется число $k = 0$ (осуществляется сброс счетчика тактов);
- вычисление вектора управляющего воздействия u_{k-1} по соотношению (3);
- измерение вектора наблюдений s_k для $k>0$;
- вычисление вектора e_k по (4);
- вычисление вектора ε_k по (17) с учетом оценки коэффициента c_k по (18);
- подключение (ПР→2) управляющего воздействия u_{k-1} к блоку фильтра-предсказателя;

Заключение. Предложен устойчивый адаптивный алгоритм работы следящей системы – модели речеобразования. Согласно данному алгоритму в соответствии с соотношением (8) синтезируется сигнал источника речеобразования на основе идентификации неизвестных параметров голосового тракта, моделью которого является линейная нестационарная стохастическая система. На основе теоремы об устойчивости при выборочных средних доказана сходимость параметров следящей системы к параметрам модели голосового тракта. Разработана структурная схема адаптивной следящей системы с предсказанием, не требующая априорных сведений о голосовом источнике и параметрах идентифицируемого голосового тракта и реализуемая на микропроцессорной элементной базе.

Литература

1. Санников В.Г. Устойчивый алгоритм статистической идентификации авторегрессионной модели речевого сигнала. // Системы и средства связи телевидения и радиовещания. 2000, №2, с. 36.
2. Омельченко В.А., Санников В.Г. Вероятностные и детерминистские модели каналов и проблемы передачи информации в электросвязи: Учеб. пособие. Киев: УМК ВО, 1992, 200 с.
3. The Past, Present and Future of Speech Processing. // IEEE Signal Processing Magazine. – May 1998. – p. 24-48.
4. Александров А.Г. Оптимальные и адаптивные системы: Учеб. пособие для вузов. М.: Высшая школа, 1989, 263 с.
5. Мартин-Санчес Х. Новое решение задачи адаптивного управления. // ТИИЭР, том 64, № 8, август 1976, с. 106 – 117.
6. Абдуллаев Н.Д., Петров Ю.П. Теория и методы проектирования оптимальных регуляторов. Л.: Энергоатомиздат ЛО, 1985, 240 с.
7. Андерсон Б., Битмид Р., Джонсон К. и др. Устойчивость адаптивных систем. / Пер. с англ. М.: Мир, 1989, 263 с.
8. Льюнг Л. Идентификация систем. Теория для пользователя. / Пер с англ. / Под ред. Цыпкина Я.З. М.: Наука, ГР ФМЛ, 1991, 432 с.

Н. Абкаиров, А. Назаров, Л. Пурто

Создание программного кода для алгоритмов ЦОС: советы разработчика

Рассматриваются общие подходы к реализации алгоритмов цифровой обработки сигналов (ЦОС) на конкретных платформах, помогающие делать код переносимым, поддерживаемым, легко встраиваемым. Обсуждаются аспекты совместной разработки кода силами нескольких разработчиков.

Введение

В последнее время рост производительности DSP-процессоров позволяет реализовывать на них все более сложные проекты, которые уже не под силу создать одному программисту. Растет также необходимость повторного использования кода, написанного раньше, в других проектах и даже на других платформах. С учетом того, что на создание проектов отводится ограниченное количество времени и ресурсов, при проектировании и реализации даже не очень сложных алгоритмов ЦОС возникает необходимость придерживаться некоторых принципов, которые помогают сделать программный код переносимым, поддерживаемым и легко встраиваемым.

Многие фирмы, разрабатывающие ПО, имеют свои внутрифирменные стандарты или свод правил, позволяющие унифицировать методы создания и стиль кода. В последнее время один из крупнейших производителей DSP-процессоров, фирма Texas Instruments, выступила с инициативой стандарта на внешний интерфейс ЦОС-алгоритмов [1]. Эта инициатива касается в основном только внешней стороны кода в объектном виде, однако в рамках одной фирмы имеет смысл договориться о более подробном и строгом стандарте, которому должны следовать разработчики.

Стандартизация методов создания кода и его стиля, описываемая в этой статье, преследует следующие цели:

1. Обеспечить возможность разделения труда между программистами, работающими над одним проектом.
2. Дать набор общих правил, следуя которым разработчикам будет легче создавать код вместе, с лучшим качеством и с большим циклом жизни.
3. Облегчить последующую интеграцию этого кода в другие системы.
4. Обеспечить возможность повторного использования этого кода или его частей в других проектах.
5. Повысить качество кода.
6. Из всех возможностей языка С ограничить использование некоторых из них, чтобы уменьшить вероятность возникновения ошибок.

Естественно, что эти правила не являются догмой, они только дают обзор тех принципов, на которые стоит обратить внимание перед началом проектирования. Можно рассматривать эти правила как рекомендации или советы профессионала. Однако, как показывает опыт авторов, все-таки необходимо, чтобы все разработчики после совместного обсуждения строго придерживались этих правил хотя бы в рамках одного проекта.

В первой части статьи описывается порядок создания проекта и приводится пример этапов тестирования, а во второй даны некоторые полезные советы по программированию на языке С.

Этапы создания проекта

Общие принципы

Общие методы создания программного обеспечения применимы и при создании проектов,

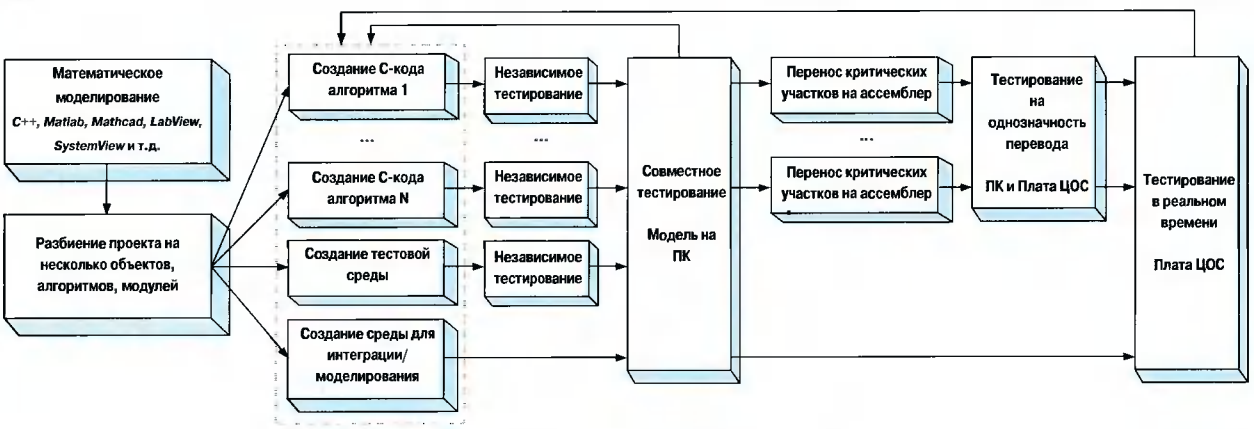


Рис. 1. Этапы создания проекта

ориентированных на цифровую обработку сигналов, однако в рамках данной статьи мы будем рассматривать только те этапы и аспекты, которые специфичны для области ЦОС.

На *рис. 1* показаны основные этапы проектирования, характерные для систем ЦОС.

При разработке многих алгоритмов ЦОС удобно и полезно уже в самом начале проектирования создать модель алгоритма на языке С. Если для начального этапа проектирования необходимо математическое моделирование на языке/инструменте более высокого уровня (C++, Matlab, LabView, SystemView и т.п.), то желательно впоследствии перенести алгоритм на язык С. Этот С-код впоследствии должен стать рабочим кодом самого алгоритма. Чтобы не запутаться в разных версиях кода (модельного и рабочего), не стоит использовать язык C++ даже на ранних этапах проектирования, потому что у многих DSP-процессоров нет C++ компиляторов либо они неэффективны. С другой стороны, ни в коем случае нельзя писать код сразу на ассемблере, потому что такой код намного труднее отладить и потом поддерживать.

После этапа математического моделирования необходимо разбить проект на несколько независимых объектов. Отдельно можно также выделить задачи создания тестовой среды (сред) для каждого объекта и среды для интеграции и моделирования всего проекта. Наличие таких отдельных задач позволяет распределить работу между несколькими программистами.

Необходимо приложить максимум усилий к тому, чтобы одни и те же С-файлы использовались как в тестовом проекте (модели), так и в рабочем коде. Это позволяет легко делать итерации, когда при отладке рабочего кода в реальной среде возникает необходимость что-то изменить в самом алгоритме.

Тестовая среда для интеграции/моделирования должна перебирать как можно более полное пространство тестовых векторов, и с помощью нее нужно добиться требуемых характеристик в проекте еще до переноса на платформу ЦОС. Если не получается достичь требуемых характеристик, то необходимо вернуться на этап проектирования С-кода алгоритма, а иногда даже на этап математического моделирования. Важно иметь в виду, что если тестовая среда не проверяет какие-либо ветви алгоритма, то есть очень большая вероятность, что там кроется ошибка. Поэтому стоит провести анализ кода и убедиться, что все ветви алгоритма тестируются. Если алгоритм был разбит на несколько независимых объектов, то нужно протестировать каждый такой объект по отдельности, а потом уже тестировать весь алгоритм. Только после этого можно приступать к переносу на другую платформу.

Для перенесения кода на конкретную платформу (обычно часть кода остается на языке С, а критические участки переводятся на ассемблер) в первую очередь необходимо создать тестовую среду для проверки однозначности перевода бит в бит (bit-exactness). Эта среда должна проверять правильность не только конечного результата работы алгоритма, но также и промежуточное состояние внутренних статических переменных алгоритма, потому что

а) многие ошибки при перенесении могут не привести к расхождению конечного результата на ограниченном наборе входных векторов;
б) зная состояние внутренних статических переменных, можно найти ошибку намного скорее.

Такая среда для проверки однозначности должна собираться и работать как на ПК, так и на платформе ЦОС, и выходные тестовые векторы, создаваемые этой средой на разных платформах, должны совпадать. В первую очередь нужно убедиться, что сам С-код (еще до включения ассемблерного кода) дает одинаковые результаты на разных платформах. Желательно проверить эту среду на нескольких разных компиляторах на ПК. Например, если изначально С-код создавался и работал в 32-разрядном компиляторе, а компилятор для платформы ЦОС – 16-разрядный, то перед тем, как переносить код на платформу ЦОС, нужно скомпилировать его на 16-разрядном компиляторе для ПК (например, Borland C 3.1) и убедиться, что тестовая среда выводит полностью одинаковые выходные векторы.

На последнем этапе – тестирование в реальном времени – желательно создать такую тестовую среду, которая будет максимально приближена к условиям работы разрабатываемого алгоритма. При обнаружении алгоритмических ошибок на этом этапе также возникает необходимость вернуться к этапу проектирования С-кода.

Пример этапов тестирования

Рассмотрим применение вышеизложенных этапов тестирования на примере проектирования алгоритма Fax Over IP. Этот алгоритм можно разбить на два независимых более простых алгоритма – факс-модем группы 3 (Fax G3) и протокол передачи факс-сообщений по IP сети ITU-T T.38. Соответственно при проектировании использовалось несколько разных тестовых сред.

I. Модель для независимой проверки приемника и передатчика факс-модема при проектировании алгоритма на ПК (*рис. 2*), состоящая из С-кода приемника и передатчика факс-модема, модели телефонного канала, среды управления моделированием и визуализации. В этой модели выходной сигнал передатчика подается на вход приемника через программную модель телефонного канала.

Управление состоянием факс-модема, управление состоянием эмулятора телефонного

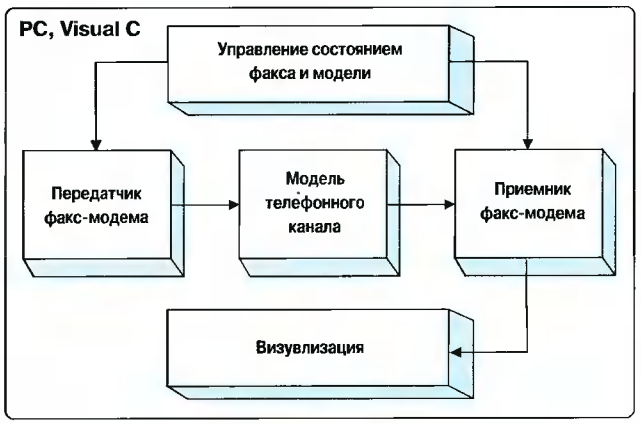


Рис. 2. Модель факс-модема на ПК

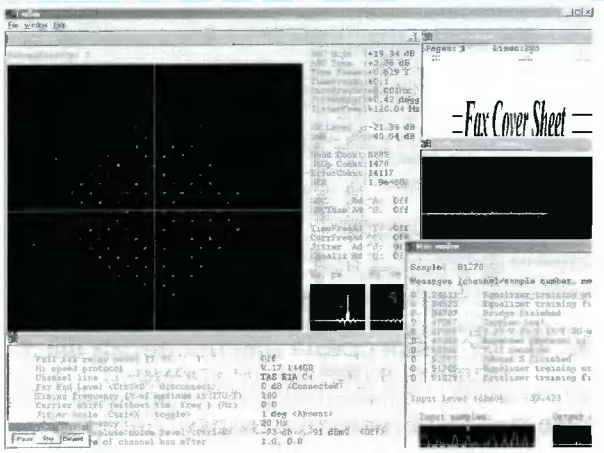


Рис. 3. Внешний вид среды визуализации модели на ПК

канала и отображение результатов моделирования производятся с помощью специальной среды моделирования/визуализации (*рис. 3*), код которой вынесен в отдельные модули и минимально связан с будущим рабочим кодом факс-модема.

II. Модель для независимой проверки приемника факс-модема на тестовых векторах (*рис. 4*), записанных при связи реальных факс-аппаратов на реальных линиях. Эта модель позволяет уже на ПК отладить работу приемника факс-модема (приемник обычно намного сложнее алгоритмически, чем передатчик) при работе с реальными факс-аппаратами.

III. Тестовая среда для проверки однозначности перевода С-кода на ассемблер (*рис. 5*) – в ней выход передатчика факс-модема подается на вход приемника. Главная цель данной среды – выводить состояние статических переменных приемника и передатчика с некоторым прореживанием по времени в файл (выходной тестовый вектор). Величину прореживания можно уменьшить, чтобы сократить размер выходного вектора, или увеличить, чтобы повысить разрешение по времени. Эта среда должна компилироваться и работать под несколькими компиляторами на ПК и на плате с DSP-процессором. Сначала нужно добиться, чтобы выходные векторы среды, скомпилированной под разными компиляторами на ПК, совпадали. Затем нужно добиться, чтобы совпадали тестовый вектор, сгенерированный на ПК, и тестовый вектор, сгенерированный на плате ЦОС на языке С. После этого можно постепенно заменять функции, которые требуют больше всего времени на разработку их ассемблерных аналогов, и после каждой замены нужно убедиться, что выходные тестовые векторы остаются одинаковыми.

Для этой среды полезно также создать вспомогательную утилиту, которая сравнивала бы выходные тестовые векторы, и если в них есть различия, то сообщала бы, в каких конкретно переменных и на каких тактах произошло расхождение.

Опыт показывает, что хотя на создание этой среды уходит некоторое время, оно быстро окупается благодаря возможности эффективно находить ошибки.

IV. Тестовая среда для проверки факс-модема в реальном времени (*рис. 6*). После перевода некоторых С-функций на ассемблер наступает момент, когда алгоритм может работать на выбранном DSP-процессоре в реальном време-

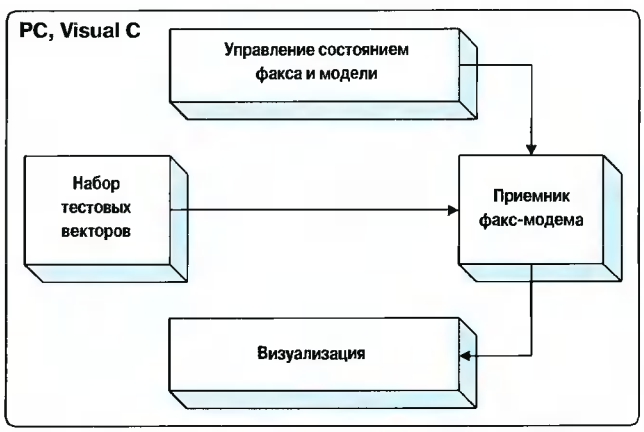


Рис. 4. Тестовая модель для проверки приемника факс-модема на тестовых векторах

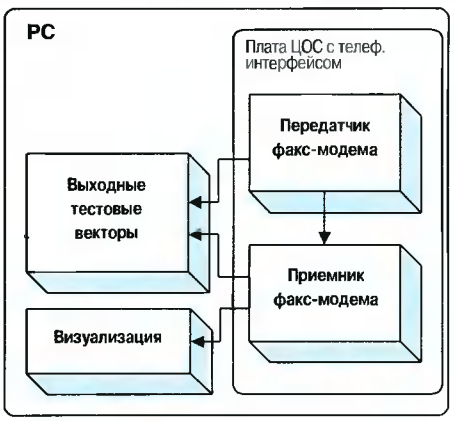


Рис. 5. Тестовая среда для проверки однозначности перевода с С на ассемблер

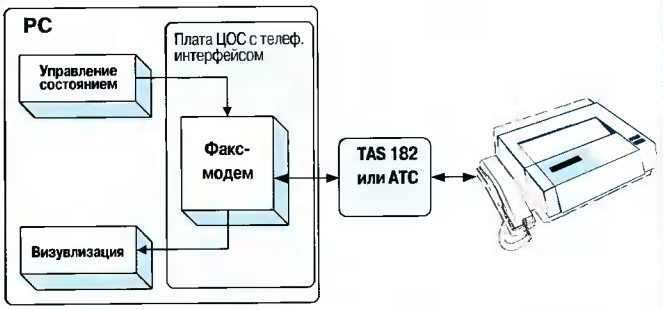


Рис. 6. Тестовая среда для проверки факс-модема в реальном времени

ни. Это позволит начать отладку, работая с реальными факс-аппаратами (даже если перевод на ассемблер еще будет продолжаться – чтобы распараллелить работу). В данной среде необходимо обеспечить передачу как можно большего количества отладочной информации из платы ЦОС в ПК, чтобы было легче отлаживать код в реальном времени. Отладочный интерфейс JTAG или, например, HPI-интерфейс (Host Port Interface) процессора TMS320C54, или IDMA процессора ADSP2181 позволяют выводить такую информацию, не замедляя работы самого процессора.

V. Модель Fax Over IP на ПК (*рис. 7*). После того как код факс-модема был отлажен независимо, можно начинать отладку кода Fax Over IP, который включает в себя факс-модем. В этой модели создаются два объекта Fax Over IP (T.38 + Факс-модем), которые связаны друг с другом через модель пакетной сети. На вход каждого из них подаются входные векторы с сигналами факс-аппаратов, а результат работы производится по выходным векторам.

VI. Тестовая среда для Fax Over IP в реальном времени (*рис 8*). Весь код Fax Over IP (T.38+ Факс-модем) погружается на платы ЦОС в двух разных ПК, связанных по IP. Это конечный этап тестирования, наиболее приближенный к реальным условиям, в которых будет использоваться код Fax Over IP. Как и в тестовой среде №IV, важно обеспечить передачу как можно большего количества отладочной информации из платы ЦОС в ПК.

Совместная работа над одним проектом

Все не генерируемые файлы проекта нужно заносить под систему поддержки версий, например Visual SourceSafe. Это позволит вести совместную разработку проекта сразу нескольким программистам, а также воспользоваться более ранними версиями файлов, если возникнет необходимость.

Время от времени необходимо ставить метки на рабочую версию кода в системе поддержки

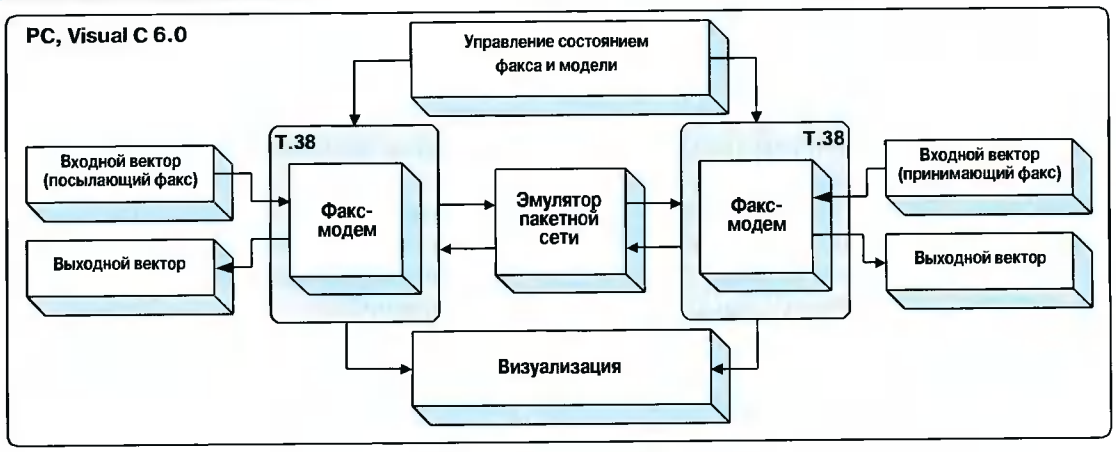


Рис. 7. Тестовая модель Fax Over IP на ПК

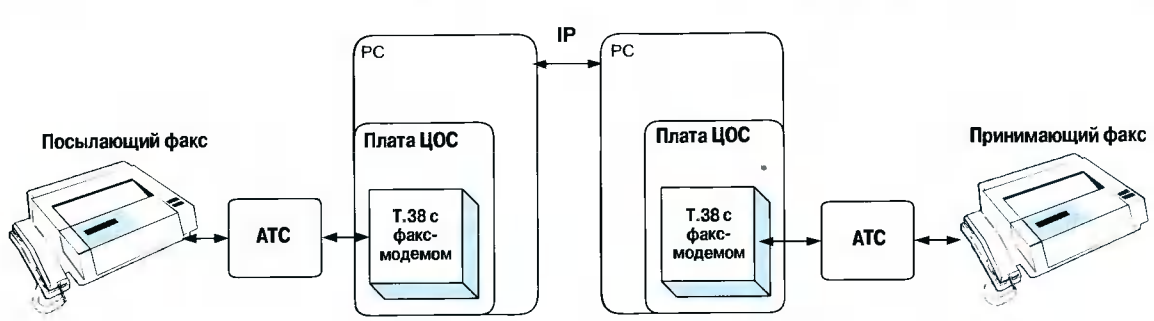


Рис. 8. Тестовая среда для Fax Over IP в реальном времени

версии, чтобы потом можно было вернуться к "хорошей" версии проекта.

В Visual SourceSafe 6.0 есть возможность разрешить правку (multiple check-out). Этой опцией нужно пользоваться очень осторожно, разрешать ее только опытным программистам, и только на этапе начального создания продукта.

Части кода, которые используются одновременно в нескольких проектах, должны копироваться в эти проекты через команду Share. Модифицировать такие файлы нужно с особой осторожностью, убеждаясь, что изменения не приводят к неработоспособности какого-либо из проектов, где эти файлы используются.

Необходимо время от времени делать резервное копирование всей базы данных Visual SourceSafe.

Системы поддержки версий часто опираются на атрибуты времени файлов, когда происходит занесение файлов в базу данных, поэтому важно, чтобы все участники проекта постоянно синхронизировали часы своего ПК с часами сервера, на котором находится база.

Некоторые полезные советы по программированию

Многие из приведенных ниже советов могут быть довольно очевидными, однако часто бывает удобно заранее убедиться, что все разработчики их понимают и согласны с ними.

Общие принципы

- Каждый заголовочный файл должен иметь защиту от повторного включения:

```
#ifndef __STDIO_H
#define __STDIO_H
...
#endif // __STDIO_H
```

- Все глобальные функции (по возможности) и константы должны быть описаны как extern в заголовочном файле, соответствующем С-файлу, в котором они объявлены (исключение могут составлять функции, у которых есть параметры такого типа, который может быть определен только после включения данного заголовочного файла). Все внутренние для данного объекта функции и константы, напротив, должны быть спрятаны с помощью слова static, и ссылки на них ни в коем случае не должны быть в заголовочном файле.
- Для обеспечения возможности использовать С-код в проекте, написанном на языке С++, желательно использовать не сам модификатор extern, а макрос типа EXTERN_C, который для С-проекта будет расширяться в просто extern, а для С++ – в extern "C".
- Если заголовочные файлы проекта находятся в разных поддиректориях проекта, то включать эти файлы нужно вместе с путем относительно верхней директории проекта, в кавычках, а в пути поиска H-файлов нужно указать только верхнюю директорию проекта:

```
#include "SystemHeaders\xdas.h"
```

- Нельзя использовать угловые скобки для включения заголовочных файлов проекта (то есть не системных файлов) во избежание неконтролируемости включения файлов из других путей.
- Тело каждого макроса должно быть обязательно обнесено круглыми или фигурными скобками. Параметры макроса – обнесены круглыми скобками (для каждого в отдельности):

```
#define Abs(x)   ( (x)<0 ? –(x) : (x) )
#define Print(s) {printf(s); }
```

- По возможности нужно "прятать" константы и имена функций, то есть ставить модификатор static перед ними, если они не используются за пределами своего файла. Это уменьшит вероятность пересечения с именами в других модулях.
- Все константы должны иметь модификатор const (иначе они будут занимать в два раза больше места – в секции .cinit и в .data), а по возможности еще и static. Константы, объявляемые внутри функций, обязательно должны иметь модификатор static const (иначе компилятор будет создавать эту константу как автоматическую переменную при каждом входе в функцию).
- Для ссылок на объекты, передаваемые в функцию по указателю, но не изменяемые внутри функции, необходимо использовать модификатор const.
- По возможности нужно избегать явного преобразования типа. Особенно это касается преобразования из void*.
- Не использовать без острой необходимости функции с переменным числом параметров – лучше в качестве параметра передавать ссылку на union.
- По возможности нужно избегать проверок на состояние и операторов switch в конечных автоматах, а пытаться реализовать такую логику с помощью вызова функций по указателям (виртуальные вызовы).

Принципы, касающиеся объектного программирования

Нижеизложенные принципы предназначены для использования объектно-ориентированного подхода на языке С. Хотя некоторые из них могут привести к расходованию дополнительных ресурсов, этим стоит пожертвовать ради лучшей организации кода.

Применять их стоит как на уровне "несамостоятельных" объектов, так и на уровне целых алгоритмов ("самостоятельных" объектов). "Самостоятельные" объекты должны удовлетворять не только этим принципам, но также и стандар-

ту XDAS фирмы Texas Instruments ([1]) для того, чтобы эти алгоритмы легко было интегрировать в другие системы. Обратите внимание, что приведенные ниже принципы помогут в дальнейшем легко сделать алгоритм совместимым с XDAS.

В дальнейшем изложении будем называть определение объекта классом, хотя в стандарте языка С и нет прямого соответствия этому понятию.

Наш опыт показывает, что инкапсуляция объектов, пусть и в урезанном виде, упрощает поддержку и сопровождение кода, а в ряде случаев дает выигрыш по памяти и быстродействию. За исключением, быть может, тривиальных объектов кода она позволяет разработчику сосредоточиться на поведенческом аспекте программы, а не на том, как устроена каждая конкретная функция.

- По возможности нужно стараться передавать функции только те переменные, которые ей необходимы, и модули нужно делать насколько возможно независимыми и самодостаточными.
- Все функции должны быть реентерабельными – все состояния объекта (статические переменные) должны быть во внешней структуре.
- Нельзя использовать глобальные и статические не константные переменные.
- Указатель на внешнюю структуру с переменными объекта (инстанцию) должен передаваться в функцию как первый параметр.
- Рекомендуется использовать указатель this или This как указатель на текущую инстанцию объекта.
- Нужно выделять класс в отдельный файл и в шапке указывать его связи.
- Нужно стремиться повторно использовать классы, пусть даже самые простые.
- Соблюдать единые правила именования классов и member-функций.
- Обязательно использовать конструкторы для класса.
- В конструкторах явно инициализировать ВСЕ переменные. Виртуальная таблица должна настраиваться конструктором.
- Класс должен допускать повторную инициализацию по умолчанию. В случае, если повторная инициализация в данный момент недопустима, она должна быть выполнена после вызова деструктора.
- Избегать явного использования переменных объекта другими объектами, заменяя обращения inline модификаторами и атрибутами.
- По возможности сокращать взаимосвязи между классами через структуру объектов – совсем слабые связи обрывать, сильные превращать в слабые.
- Использовать псевдонаследование: первый элемент класса – элемент базового класса, все функции базового класса через inline.
- Нужно стремиться к тому, чтобы отсутствовал жесткий порядок создания/инициализации взаимосвязанных объектов, то есть чтобы их можно было в любом порядке создать, а потом инициализировать.

Принципы, касающиеся отладки

- Все отладочные/визуализационные функции и переменные должны быть собраны в отдельных файлах. Все, что связанно с отладкой или моделированием в рабочем коде, должно быть обнесено директивами условной компиляции (типа IS_DEBUG или IS_MODEL, например) или вызываться через набор отладочных/модельных функций, которые в конечной системе будут заменены пустыми макросами.

В случаях, когда некоторые простые действия необходимо сделать только ради отладки/модели, можно заменять операторы соответствующими макросами:

вместо:	стоит использовать:
if((BaudCounter&1)==0) NewSignalPoint(A);	Model_DoIf((BaudCounter&1)==0) NewSignalPoint(A);

- Все инородные (модельные) функции в рабочем коде должны начинаться со стандартного префикса, например Model_XXX.

Принципы, касающиеся переносимости кода

Использование типов:

- Рекомендуется определить отдельные типы для целочисленных переменных, которые бы не пересекались со стандартными именами типов и которые позволяли бы однозначно задавать разрядность переменных. Например, на многих платформах будет справедливо такое переопределение:

```
typedef long int32;
typedef short int16;
typedef unsigned long uint32;
typedef unsigned short uint16;
```

В этом случае для переменных, которые обязательно должны быть 16-разрядными, нужно использовать тип int16; для 32-разрядных – тип int32; для беззнаковых типов использовать типы uint16, uint32 и т.д.

- Для переменных, разрядность которых не имеет значения, желательно использовать тип int – это обычно позволяет компилятору лучше оптимизировать код.
- Использовать enum вместо int там, где это имеет разумный смысл.
- Избегать арифметических операций с типом char, явно преобразуя его в int, так как реализации допускают как беззнаковое, так и знаковое расширения char.
- Символьная константа должна состоять только из одного символа

('A' ☺, 'AB' ☹). Символьные константы из двух и более символов нужно создавать так: 'A'+256*'B'+...

- В любых операциях, при вычислении которых промежуточный результат может превысить разрядность исходных аргументов, необходимо явно преобразовывать исходные аргументы к типу с большей разрядностью:

```
AInt16=BInt16*CInt16; ☹      AInt16=((int32)BInt16)*((int32)CInt16); ☺
AInt32=BInt16<<5; ☹          AInt16=((int32)BInt16)<<5; ☺
```

- Нужно инициализировать все переменные объекта (например, с помощью memset(this,0,sizeof(*this))).
- Побитовые сдвиги должны иметь только положительный аргумент:

```
C=-5; ☹                  if( C<0 )                    ☺
~~A=B>>C;~~                  A=B << –C;
                         else
                             A=B >> C;
```

- Не допускается объявление переменных или функций без явного указания типа:

```
~~extern I;~~ ☹          extern int I;                ☺
~~extern GetParam();~~   extern int GetParam();
```

- Нужно помнить, что размер объекта (типа) измеряется в единицах size_t, а не int. Везде, где нужно вычислить размер объекта, необходимо использовать sizeof().

Принципы, касающиеся реализации на платформах ЦОС

- С-код нужно писать сразу так, чтобы он учитывал специфику процессора ЦОС. Рекомендуется создать набор примитивных функций, моделирующих операции с насыщением, работу с кольцевыми буферами, и т.п. (пример такого набора можно найти в файле basop.c рекомендации G.723.1 [2]), и во всем коде осуществлять эти операции только через вызов этих примитивных функций. Это позволит сохранить логику работы кода при переносе на ассемблер.
- Переменные нужно распределять, как минимум, в двух структурах – для быстрой и медленной памяти.
- Переменные, которые никогда не используются одновременно, можно размещать в одной и той же физической области памяти для экономии. Это можно сделать, например, через их объединение внутри одной структуры.
- Линии задержки и буфера, к которым происходит интенсивное обращение, нужно делать по возможности кольцевыми. На некоторых платформах, чтобы уменьшить потери памяти, связанные с выравниванием, такие буфера должны располагаться в начале структуры.
- Нельзя заводить кольцевые буфера как автоматические переменные внутри функции, потому что при этом невозможно обеспечить выравнивание.

Советы для написания "прочного" кода

Под "прочностью" или "отказоустойчивостью" подразумевается свойство программного кода противостоять ошибочным действиям программиста и не "разваливаться" в результате допущенных ошибок.

- Дайте возможность компилятору сообщать все предупреждения и добейтесь, чтобы код компилировался без этих предупреждений. Любые предупреждения, даже в отладочных версиях кода, не допустимы.
- Пользуйтесь встроенной отладочной диагностикой. Проверяйте все допущения, которые вы имеете в виду при написании кода, даже если это довольно очевидно. Очень важно проверять, что указатели не равны NULL. Для проверки можно, например, использовать макрос ASSERT:

```
void SetMemory(int * pBuffer, int count)
{
 ASSERT(pBuffer!=NULL);
 while(count-- > 0)
  *pBuffer++=0;
}
```

- По возможности используйте более общие проверочные условия:

```
if(A==256) ☹              if(A>=256) ☺
 A=0;                          A=0;
```

- Обязательно перечитывайте то, что вы написали.

Заключение

Описанные методы были успешно использованы при создании многих проектов для платформ ЦОС инженерами Центра цифровой обработки сигналов фирмы SPIRIT Corp.

Приведенный список принципов является далеко не полным и постоянно обновляется и дополняется. Нашей целью было просто дать пример свода правил и рекомендаций. Мы надеемся, что программисты и менеджеры проектов найдут в этом для себя что-то полезное.

Предполагается в дальнейшем написать серию статей по этой тематике, если наш опыт найдет понимание в кругу разработчиков систем ЦОС.

Литература

1. "eXpressDSP Algorithm Standard. Rules and Guidelines" Фирма Texas Instruments, SPRU352, February 2000, http://dspvillage.ti.com/docs/algo_stand/tms320algohome.jhtml.
2. Рекомендация ITU-T G.723.1: "DUAL RATE SPEECH CODER FOR MULTIMEDIA COMMUNICATIONS TRANSMITTING AT 5.3 AND 6.3 kbit/s", 1996.

М.С. Куприянов, Б.Д. Матюшкин, Д.Б. Головкин

Реализация многофункционального обнаружителя гармонических сигналов на DSP

Постановка задачи

Задача обнаружения гармонического сигнала (или его отрезков) на фоне шума достаточно хорошо изучена [1, 2]. Вместе с тем технические условия реализации конкретного приложения, для которого разрабатывается обнаружитель, в том числе и требования реализации на цифровом сигнальном процессоре (DSP), существенно влияют на выбор алгоритма обработки и аспекты технической реализации.

В настоящей работе рассматривается реализация многофункционального обнаружителя гармонических сигналов, используемого при передаче служебной информации в канале связи на фоне речевого сигнала.

В реализованном проекте ставилась задача создать устройство, позволяющее распознавать гармонические тональные сигналы на фоне речевого сигнала в телефонном канале связи. Задание осложнялось высоким уровнем шумов с неопределенным законом распределения.

В проекте реализованы следующие режимы работы обнаружителя.

Режим 1: обнаружение 8 низкочастотных тональных сигналов при отношении сигнал/шум ≤ –20 Дб.

Режим 2: обнаружение 64 низкочастотных тональных сигналов на заданной сетке частот при отношении сигнал/шум ≤ –20 Дб.

Режим 3: генерация 64 тональных сигналов на заданной сетке частот. Этот режим согласовывался с режимом 2 при парном соединении устройств в режиме приемник – передатчик.

Режим 1

Выделение тональных гармонических сигналов может быть реализовано с использованием набора узкополосных фильтров [2], скользящего спектрального анализа [1] и некоторых других методов. Для данного технического задания проведенный сравнительный анализ по критерию "качество – требуемая производительность DSP" показал, что наиболее эффективным является применение скользящего спектрального анализа.

Структура алгоритма обработки (для одной частоты) приведена на *рис. 1*.

На *рис.1* используются следующие обозначения: x_n – отсчеты входного сигнала системы ЦОС, поступающие с кодека; z^{-N} – линия задержки на N тактов ($N = 1023$); z_i – точка на z-плоскости, в которой проводится спектральный анализ ($z_i = exp(j2\pi \hat{f}_i n)$, где $\hat{f}_i$ – нормированная частота, на которой проводится анализ ($i = 1, 2, ..., 8$); $-z_i^{-N}$ и z_i^{-1} – величины масштабных множителей (комплексные) для i-й частоты; $S_n(z_i)$– текущее мгновенное значение спектра на i-й частоте; $A^2_{n,i}$ – квадрат текущего значения амплитудного спектра на i-й частоте; $d_{n,i}$ – сигнал после порогового устройства на i-й частоте; $p_{n,i}$ – выходной сигнал системы на i-й частоте.

Входной сигнал представляет собой сумму полезного сигнала (тональных посылок) и помехи (речевого сигнала) с заданными параметрами.

Сигнал x_n обрабатывается в комплексном БИХ-фильтре, представляющем собой узкополосный фильтр. Величине $N = 1023$ соответствует полоса пропускания фильтра около 10 Гц. Выходной комплексный сигнал $S_n(z_i)$ фильтра представляет собой отсчеты текущих мгновенных значений спектра на данной частоте.

Вещественная и мнимая составляющие выходного сигнала $S_n(z_i)$ фильтра возводятся в квадрат и складываются, что приводит к формированию текущего значения $A^2_{n,i}$ квадрата амплитудного спектра сигнала на данной частоте.

Отсчеты сигнала $A^2_{n,i}$ на каждом такте сравниваются с пороговым значением $G_{пор}$. В результате формируется сигнал $d_{n,i}$ по алгоритму

$$d_{n,i} = \begin{cases} 1, \text{ при } A^2_{n,i} < G_{пор} \\ -1, \text{ при } A^2_{n,i} < G_{пор} \end{cases}, \quad n = 0,1,2,...$$

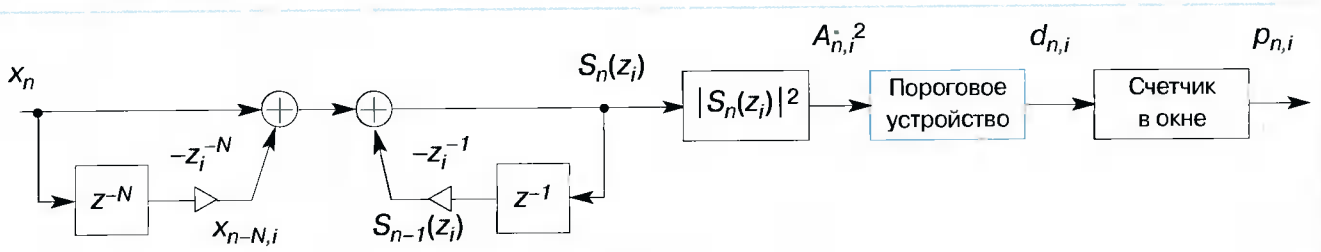


Рис. 1.

Счетчик в окне организует подсчет относительного количества превышений порогового уровня сигналом в окне из 4500 отсчетов, что соответствует длительности анализируемого сигнала, равной 563 мс.

К текущему состоянию счетчика добавляется сигнал (содержимое счетчика увеличивается, если оно не достигло максимального допустимого значения, которое равно числу отсчетов, укладывающихся на длительности интервала обнаружения, и уменьшается, если только содержимое счетчика не достигло нулевого значения).

Если относительное количество превышений порогового уровня составит 0.5 и более, формируется сигнал $p_{n,i}$= 1, что свидетельствует об обнаружении тональной посылки.

После фиксации факта обнаружения гармоники устанавливается счетчик паузы (смены состояния обнаружителя данной гармоники).

Если относительное количество превышений порогового уровня меньше, чем 0.5, и значение счетчика паузы (смены состояния обнаружителя для данной гармоники) равно нулю, то фиксируется факт отсутствия данной гармоники. Если же значение счетчика паузы больше нуля, то значение счетчика уменьшается и состояние обнаружителя не изменяется.

Наличие счетчика смены состояния обнаружителя позволяет регулировать длительность сигнала, фиксирующего факт обнаружения, а также исключить небольшие разрывы в сигнале обнаружения в моменты, когда речевая компонента подавляет гармонику. Вместе с тем выбор слишком большого значения паузы может привести к нежелательному эффекту увеличения длительности сигнала обнаружения при ложном срабатывании от сильных гармонических составляющих речевого сигнала.

Счетчик паузы (смены состояния обнаружителя) вводится для устранения провалов и увеличения длительности импульса обнаружения. Счетчик "откладывает" переход обнаружителя в состояние "гармоника не обнаружена" на некоторое время (0.05 с) после последней фиксации факта превышения гармоникой порога обнаружения в течение 0.25 с.

Необходимо отметить, что рассмотренная структура используется для обнаружения тональной посылки на одной частоте при допустимом отклонении частоты от номинального значения порядка 2 Гц (при заданных номинальных уровнях тональных сигналов и речевого сигнала). Поскольку допуски на отклонения частот от номинальных значений для ряда частот существенно превышают указанную величину, для обнаружения тональных сигналов на разных частотах использовалось разное количество структур, аналогичных представленной на *рис. 1*. Таким образом, общее количество фильтров равно 42.

Отметим, что выбор алгоритма обработки, разработка и отладка ассемблерной программы, а также анализ качественных характеристик обнаружения велись с использованием Стартового Комплекса Разработчика (СКР) систем цифровой обработки сигналов [3, 4].

Режим 2

При выборе алгоритма обработки сигнала для режима 2 использовался тот же критерий "качество – требуемая производительность DSP". Алгоритм с использованием скачущего БПФ для оценки амплитуд спектральных составляющих на заданных частотах оказался наиболее эффективным для выполнения поставленной задачи.

Структура алгоритма обработки (для одной частоты) приведена на *рис. 2*.

На *рис.2* используются следующие обозначения: x_n– отсчеты входного сигнала системы ЦОС, поступающие с кодека; $A^2_{n,i}$ – квадрат текущего значения амплитудного спектра на i-й частоте; $d_{n,i}$ – сигнал после порогового устройства на i-й частоте; $p_{n,i}$ – выходной сигнал системы на i-й частоте.

Входной сигнал x_n представляет собой сумму полезного сигнала (тональной посылки) и помехи (речевого сигнала) с заданными параметрами.

Входной сигнал x_n обрабатывается в гребенчатом фильтре. Для простоты реализации гребенчатый фильтр построен на основе однородного КИХ-фильтра путем интерполяции его импульсной характеристики. Задачей гребенчатого фильтра является предварительное выделение тональных сигналов и частичное подавление помехи (речевого сигнала).

Выходной сигнал гребенчатого фильтра подвергается скачущему БПФ (со скачком в 64 точки). На длительности тональной посылки выполняются 53 БПФ на 512 точек ($БПФ_{512}$). Каждое измерение позволяет получить оценку амплитуд компонент текущего спектра на заданных частотах (вещественная и мнимая составляющие выходного сигнала $БПФ_{512}$ для каждой из требуемых частот f_i, i = 1, 2, ..., 64 возводятся в квадрат и складываются, что приводит к формированию те-

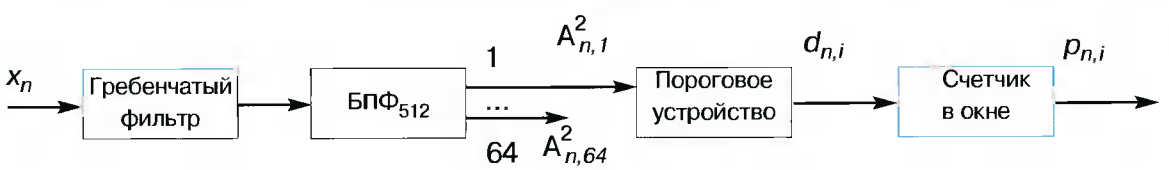


Рис. 2.

кущего значения квадрата амплитудного спектра сигнала $A^2_{n,i}$ на данной частоте).

Выходные сигналы $БПФ_{512}$ после каждого измерения сравниваются с пороговым значением $G_{пор}$. В результате формируется сигнал $d_{n,i}$ по алгоритму

$$d_{n,i} = \begin{cases} 1, \text{ при } A^2_{n,i} \geq G_{пор} \\ -1, \text{ при } A^2_{n,i} < G_{пор} \end{cases}, n = 0,1,2,...$$

Счетчик в окне организует подсчет количества превышений порогового уровня сигналом $A^2_{n,i}$. Если относительное количество превышений порогового уровня составит 0.5 и более, формируется сигнал $p_{n,i} = 1$, что свидетельствует об обнаружении тональной посылки на i-й частоте.

В алгоритме предусмотрены счетчики пауз смены состояний обнаружителей гармоник, как и в первом режиме.

На этапах анализа и проектирования использовался СКР [3, 4].

Режим 3

Алгоритм генерации тональных сигналов основан на использовании встроенной таблицы синусов. Чтобы не производить вычисления интерполированных значений синуса для промежуточных значений аргументов, не представленных в таблице, что требует дополнительных вычислительных затрат и может ухудшить качество воспроизведения тональных сигналов, используется таблица с количеством коэффициентов, кратным генерируемым частотам и частоте дискретизации.

Пример: при частоте дискретизации 8 кГц возьмем таблицу с полным периодом синуса на 256 значений. Значит, если последовательно выдавать значения из таблицы с частотой 8 кГц, то в информационном канале будет генерироваться синусоида с частотой 8000/256=31.25 Гц. Если выдавать значения с интервалом в N элементов, то на выходе будет частота N*31.25 Гц. Если N равно 16, то генерируемая синусоида будет иметь частоту 500 Гц.

Аппаратная реализация

При выборе процессора цифровой обработки сигналов обычно руководствуются следующими очевидными соображениями.

- Целесообразно выбирать процессор с некоторым запасом производительности относительно требуемой прогнозируемой производительности на начальном этапе.
- Процессор должен обладать необходимым набором периферийных модулей и интерфейсов к внешней памяти, кодеку и персональному компьютеру.
- Разработчику должны быть доступны средства разработки: кросс-средства и средства анализа системы ЦОС.
- Также должна быть доступна соответствующая документация: руководства пользователя, описание электротехнических характеристик и библиотека приложений.
- Цена, простота и скорость поставки должны удовлетворять потребности и возможности производителей.

В качестве ядра системы был выбран цифровой сигнальный процессор DSP56002 фирмы Motorola [5, 6]. Характеристики процессора отвечают всем перечисленным требованиям. Средства разработки: компиляторы ассемблера и Си, компоновщик и симулятор и описания к ним распространяются бесплатно. Для анализа системы ЦОС использовался Стартовый Комплекс Разработчика (СКР) – интегрированная среда разработки, включающая кросс-средства для данного процессора и подсистему анализа и моделирования алгоритмов ЦОС с автоматической кодогенерацией для процессоров DSP5600x/DSP5630x [3, 4]. Процессор имеет невысокую цену и продается рядом фирм – поставщиков микроэлектроники.

Краткая характеристика DSP56002 [5, 6]:

- Внутренняя частота – 66 МГц при низкой внешней частоте.
- Производительность – 33 MIPS.
- Разрядность памяти – 24 бита.
- Отдельные память программ и две памяти данных X и Y с параллельным доступом.
- Внутреннее ОЗУ – 512/256/256 слов (память программ/X/Y).
- Адресное пространство – 64K/64K/64K слов с поддержкой режим загрузки из внешнего 8-битного ПЗУ.
- Внутренняя таблица синусов – 256 значений полного периода (память Y).
- Режимы модульной и бит-реверсивной адресации.
- Поддержка быстрых прерываний.
- Встроенный аппаратный стек.
- Команды аппаратного цикла.
- Периферия, включающая порт к внешней памяти, последовательные синхронный и асинхронный интерфейсы и таймер.

На базе этого процессора было разработано устройство, реализующее распознавание и генерацию тональных сигналов. Обобщенная структурная схема устройства показана на *рис. 3*.

На *рис. 3* используются следующие обозначения: SSI – периферийный модуль последовательного синхронного интерфейса; SCI – периферийный модуль последовательного асинхронного интерфейса.

Во внешнем флеш-ПЗУ хранятся программа и данные. При запуске производится загрузка внутреннего ОЗУ программ из внешнего флеш-ПЗУ. Далее загруженная программа производит счи-

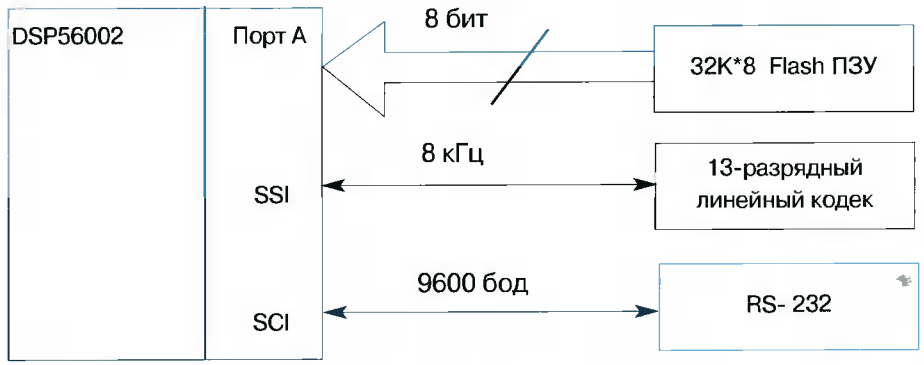


Рис 3.

тывание данных и подпрограмм, необходимых для выполнения того или иного режима.

Прием и передача аналоговых сигналов осуществляются через 13-битный линейный кодек, связь с которым осуществляются по последовательному синхронному интерфейсу через периферийный модуль процессора SSI со скоростью 8 тысяч посылок в секунду, что соответствует частоте дискретизации сигнала 8 кГц.

Связь устройства с персональным компьютером или другим управляющим устройством осуществляется по последовательному асинхронному интерфейсу RS-232 через периферийный модуль процессора SSI в 10-битном режиме без проверки четности на стандартной скорости 9600 бод.

Программная реализация

Разработанное программное обеспечение состоит из ряда частей: подпрограмма инициализации (отвечает за загрузку процессора и перевод его в нормальный режим работы); подпрограмма загрузки режимов из флеш-ПЗУ (осуществляет смену режимов по внешнему управлению); подпрограмма обмена информацией (обрабатывает управляющие команды и выдает результаты обнаружения по последовательному интерфейсу) и подпрограмма трех режимов работы.

Отладка и тестирование

Отладка программно-аппаратного обеспечения состояла из нескольких этапов.

На первом этапе производились анализ и тестирование на математических моделях алгоритмов цифровой обработки. Анализировалось качество работы алгоритмов при обнаружении тональных сигналов без помех и тональных сигналов с помехами в виде речевых фрагментов с усредненной энергией, соответствующей данной в задании. Анализ производился в интегрированной среде CKP [3, 4]. Звуковые фрагменты создавались в звуковом редакторе Sound Forge.

На втором этапе проводилось тестирование созданного программного обеспечения в программном симуляторе процессора в режиме обработки входного сигнала. Тестирование производилось на тех же фрагментах, что и на первом этапе.

На следующем этапе производилось тестирование на отладочной плате EVM56k фирмы Domain Technology. Тестовые звуковые фрагменты подавались на встроенный кодек. Информация передавалась в персональный компьютер по встроенному последовательному интерфейсу RS-232. Для отладки порогов обнаружения и времени обнаружения использовался генератор сигналов HP33120A фирмы Hewlett-Packard.

На заключительном этапе производилось тестирование изготовленного прототипа аналогичным способом, что и на предыдущем этапе. Тестовые испытания подтвердили аналитические расчеты.

Заключение

Разработанная в Центре микропроцессорных технологий (CMT) методика выполнения проектов реализации систем цифровой обработки сигналов, а также соответствующее программное обеспечение и отладочные средства, имеющиеся в CMT, позволяют выполнить аналогичные проекты в достаточно короткие сроки.

Полный цикл анализа и поиска алгоритмов ЦОС, разработки и отладки программного и аппаратного обеспечения с созданием рабочего образца многофункционального обнаружителя гармонических сигналов на DSP составил 3 месяца для коллектива из 4 человек.

Литература

1. Рабинер Л., Гоулд Б. Теория и применение цифровой обработки сигналов. М.: Мир, 1978, 848 с.: ил.
2. Гольденберг Л.М., Матюшкин Б.Д., Поляк М.Н. Цифровая обработка сигналов. Учебное пособие для вузов. М.: Радио и связь, 1990, 256 с.: ил.
3. Куприянов М.С., Матюшкин Б.Д. Цифровая обработка сигналов: процессоры, алгоритмы, средства проектирования. 2-е изд., перераб. и доп. СПб.: Политехника, 1999, 592 с.: ил.
4. Техническое обеспечение цифровой обработки сигналов. Справочник. Куприянов М.С и др. СПб.: Форт, 2000, 752 с.: ил.
5. DSP56000. Digital Signal Processor. Family Manual. – Motorola Inc.,1993.
6. DSP56002. Digital Signal Processor. User's Manual. – Motorola Inc.,1993.

Ю.В. Гончаров

Интеграция цифровых сигнальных процессоров семейства TMS320C54xx в разрабатываемые устройства и системы

Широко известно и разрекламировано семейство ЦСП с фиксированной точкой С5000 фирмы TI. Сочетание 100MIPS производительности и $7 цены даже для младшего представителя семейства VC5402 в интеграции с мощными средствами отладки делает его очень привлекательным решением для применения в новых разработках. Мы не будем повторять еще раз описание ядра процессора и его возможностей, а остановимся на вопросах применения ЦСП семейства С5000. Ведь само мощное вычислительное ядро – это еще не все. Процессор надо установить на плату, обеспечить питание, обеспечить загрузку кода, выполнить интерфейс со всевозможной периферией и т.д. Именно аспектам интеграции ЦСП в разрабатываемую систему и посвящена эта статья.

Достигнув определенных и весьма высоких результатов в своем классе устройств, семейство ЦСП С54xx продолжает свое развитие. Несмотря на разработку процессорного ядра нового поколения и выпуск новой линейки ЦСП С5510 на его базе, TI продолжает совершенствовать процессоры на ядре С54xx. Причем, что интересно, расширение идет как в сторону выпуска более мощных устройств, так и в сторону выпуска более простых, чем выпускаемая младшая модель линейки С5402. В составе линейки ЦСП С54xx выпущены три новые модели процессоров: С5409A, С5410A и С5401.

Новые модели ЦСП С5409A и С5410A выпускаются по технологии 0.15 мкм и являются развитием популярных моделей семейства – процессоров 'С5409 и 'С5410. Новые ЦСП выпускаются с тактовой частотой 120 и 160 МГц и соответственно быстрее своих предшественников на 60%, но при этом за счет новой технологии изготовления имеют лучшее соотношение производительность/потребление. Новые ЦСП выпускаются в классических для С54xx корпусах TQFP-144 и MicroStar BGA размером 12x12 мм.

При разработке новых моделей ЦСП их старались сделать максимально совместимыми с популярными ЦСП С5409 и С5410, однако в них заложен ряд нововведений и усовершенствований. Для облегчения интеграции новых моделей в разрабатываемые устройства фирма TI выпустила специальные документы, описывающие отличия моделей ЦСП и последовательность перехода [1,2].

Еще одной интересной моделью является ЦСП С5401. Фактически С5401 – это половина от С5402 – в два раза меньшая частота (50 МГц) и в два раза меньше памяти, но при этом меньшее энергопотребление. В условиях жесткой конкуренции на рынке ЦСП все производители стремятся выпустить более мощный процессор и без серьезных причин снижать параметры выпускаемого изделия никто не будет. Особенно если это одна из лучших моделей в своем секторе рынка. Но как показала практика выпуска и применения ЦСП С54xx, заложенный в архитектуру потенциал даже младшей модели семейства достаточно велик для эффективной реализации большого класса задач. Причем он оказался настолько велик, что было принято решение об уменьшении произ-

Таблица 1. Основные характеристики ЦСП линейки С54xx

ЦСП	VC5401	VC5402	UC5402	VC5409	UC5409	VC5409A	VC5410	VC5410A	VC5416	VC5421	VC5441
Производительность, MIPS	50	100	80	80, 100	80	120, 160	100, 120	120, 160	120,160	200	532
ОЗУ с двойным доступом, 16-разр. слов	8K	16K	16K	32K	23K	32K	8K	8K	64K		
ОЗУ 16-разр. слов								56K	56K	64K	
ПЗУ 16-разр. слов	4K	4K	4K	16K	16K	16K	16K	16K	16K	4K	
ПДП	6 внутр.	6 внутр.	6 внутр.	6 внешн.	6 внешн.	6 внешн.	6 внешн.	6 внешн.	6 внешн.	12 внешн.	24
McBSP	2	2	2	3	3	3	3	3	3	6	12
HPI	HPI 8	HPI 8	HPI 8	HPI 8/16	HPI 8/16	HPI 8/16	HPI 8/16	HPI 8/16	HPI 8/16	HPI 16	HPI16
Питание ядра	1.8	1.8	1.8	1.8	1.8	1.5 – 120 1.6 – 160	2.5	1.5 – 120 1.6 – 160	1.5 – 120 1.6 – 160	1.8	1.8
Питание выводов	3.3	3.3	1.8 – 3.6	3.3	1.8 – 3.6	3.3	3.3	3.3	3.3	3.3	

водительности ЦСП в пользу снижения энергопотребления и, конечно же, стоимости.

Основные характеристики ЦСП семейства C54xx приведены в *табл. 1*.

Вернемся к теме нашей статьи – интеграции ЦСП семейства C5000, и последовательно рассмотрим все задачи, которые возникают при построении устройства на базе ЦСП C54xx и пути их решения.

Питание

Первый вопрос, естественно, – это вопрос подачи напряжения питания. ЦСП семейства C54xx имеют два напряжения питания – напряжение питания ядра и напряжение питания блоков ввода-вывода. Напряжение питания ядра составляет в зависимости от модели ЦСП от 1.5 до 1.9 В. Напряжение питания блоков ввода-вывода – 3.3 В. Фактически все внутренние узлы процессора питаются от источника напряжения ядра, а питание 3.3 В подается на блоки ввода-вывода для обеспечения электрической совместимости по внешним выводам со стандартными сигналами. Казалось бы, наличие второго напряжения питания для ядра процессора порождает целый ряд проблем, однако параллельно с выпуском ЦСП с двойным питанием фирмой TI предлагается целый ряд решений для обеспечения его установки.

Начнем с контроля напряжений питания. Для использования совместно с ЦСП и программируемой логикой с двойным напряжением питания TI выпускает целый ряд микросхем супервизоров как с возможностью контроля двух напряжений питания, например TPS3305/06, так и с возможностью контроля трех напряжений питания для случая, когда в системе используются два разных напряжения питания ядра, например ЦСП VC5402 и ПЛИС Virtex.

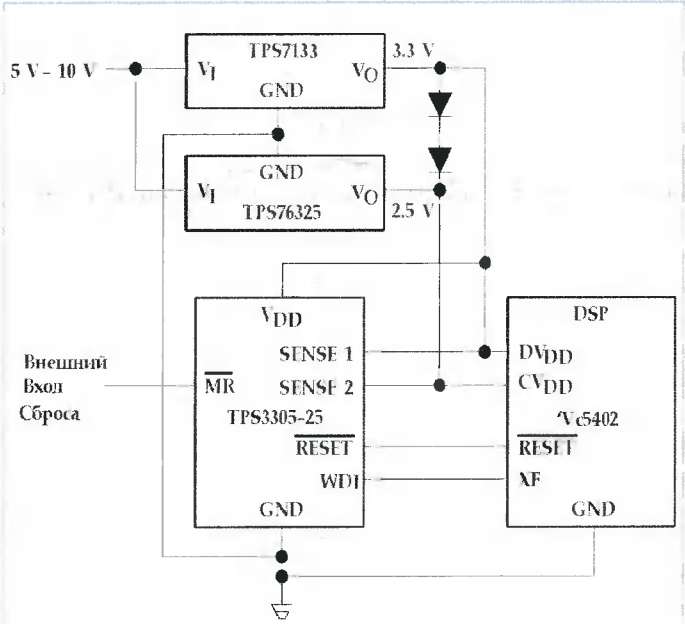


Рис. 1. Применение супервизора питания серии TPS3305 совместно с ЦСП C54xx

Супервизор питания TPS3305 представляет собой законченное решение для контроля напряжений питания и обеспечения начального сброса ЦСП и включает в себя две схемы слежения за напряжениями питания, схему сброса с задержкой 200 мс и Watchdog (сторожевой)-таймер. TPS3305 выпускается на фиксированные пары напряжений питания 3.3/2.5 В, 3.3/1.8 В и 3.3/5 В и не требует для работы внешних компонентов. При этом собственное потребление супервизора составляет менее 40 мкА. Микросхема выпускается в 8-выводном корпусе и имеет 2 входа контроля напряжений питания, вход Watchdog-таймера, выход RESET и вход внешнего принудительного сброса. Пример применения TPS3305 для контроля напряжения питания ЦСП VC54xx приведен на *рис. 1*.

Другим примером законченного решения для системы с двумя напряжениями питания может служить микросхема TPS3306, выпускающаяся на большее число пар напряжений – от 1.5 до 5 В и имеющая вместо Watchdog-таймера дополнительный компаратор напряжения.

Теперь перейдем к самим стабилизаторам питания. Возможны несколько вариантов. Первый из них – интеграция ЦСП с двойным питанием в 3 В систему. В данном случае требуется получить напряжение для питания ядра ЦСП, причем не обязательно включать его в основной источник питания устройства, достаточно установить локальный линейный стабилизатор. Для этой цели фирмой TI разработано специальное семейство регуляторов питания TPS76xxx. Это миниатюрные линейные регуляторы с низким падением напряжения, рассчитанные на ток до 150 mA, что вполне достаточно для питания ядра любого ЦСП линейки C54xx. Достаточно просто установить один такой дополнительный стабилизатор, подключить его к шине питания 3.3 В, и проблема выработки питающего напряжения для ядра ЦСП будет решена. При этом следует добавить, что линейка стабилизаторов TPS76xxx выпускается в корпусе SOT23, занимающем на печатной плате очень малое место.

Следующим вариантом является построение источника питания для системы с потреблением до 750 mA или интеграции небольшого 3.3 В ядра в состав 5 В системы. Для этого случая можно порекомендовать стабилизаторы питания серии TPS73HD3xx – двухканальные линейные стабилизаторы с током до 750 mA на канал. Отметим, что эти стабилизаторы выпускаются в специальном миниатюрном корпусе PowerPAD TSSOP-28.

Промежуточным решением для построения источников питания для тока от 0.5 до 3 А является комбинация схемы управления TL5001 и MOSFET TPS1101.

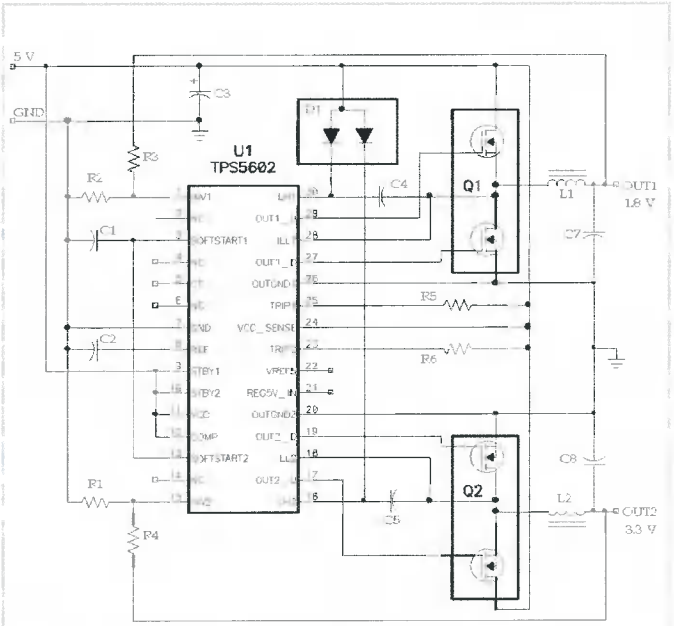


Рис. 2. Двухканальный источник питания на базе TPS5602

Далее идет вариант построения мощной системы с двойным питанием, включающей в себя один или несколько ЦСП, программируемую логику и периферию. Для этой цели предлагаются несколько решений, как комбинированных импульсных + линейных, так и импульсных стабилизаторов питания, например двухканальный импульсный стабилизатор TPS5602 (*рис.2*).

Подробнее о применении TPS5602 и аналогичных микросхем написано в [3].

Еще одним вариантом решения для построения источника питания является использование так называемых Pug-in Power Solutions – готовых источников питания серий 78xxx и PTxxx, которые представляют собой уже собранный модуль источника питания, устанавливаемый на плату. Встраиваемые модули питания выпускаются для различных выходных напряжений. Их полный список и таблица параметров доступны на сервере http://power.ti.com.

Отметим, что для всех предлагаемых решений характерен ряд особенностей. Во-первых, как и сами ЦСП C54xx, разные модели которых с разными параметрами выпускаются в совместимых корпусах, так и для микросхем источников и супервизоров питания выпускается линейка устройств в одинаковых корпусах, но рассчитанных на разные выходные напряжения, ряд которых совпадает с требуемыми для ЦСП линейки C54xx. Во-вторых, поскольку сами ЦСП C54xx рассчитаны на применение в малопотребляющих и малогабаритных мобильных устройствах, то и рекомендованные для них источники питания, также рассчитаны на мобильное применение и имеют ряд общих свойств:

- максимальная интеграция и минимальное число внешних элементов;
- малое собственное потребление;
- миниатюрные корпуса;
- малое падение напряжения (для линейных источников).

Описание популярных источников и супервизоров питания от TI доступно на сервере ЗАО SCAN по адресу http://www.texas.ru/ti/powsup/powsup.html или на сервере TI http://power.ti.com. Отметим также, что на сервере www.ti.com для каждого из ЦСП дается регулярно обновляемый список микросхем источников питания.

Тактовые сигналы

Следующий вопрос после подачи напряжения питания – подача тактового сигнала. ЦСП семейства работают на частотах до 160 МГц и имеют гибкий программируемый блок генерации тактовых сигналов, позволяющий работать как с внешним источником тактовой частоты, так и с подключаемым непосредственно к ЦСП кварцевым резонатором.

В состав блока генерации тактовых сигналов ЦСП C54xx входит программируемый умножитель частоты. Его использование позволяет решить проблему выработки и подачи на ЦСП высокочастотного тактового сигнала, а также позволяет легко синхронизировать работу ЦСП с имеющимися тактовыми сигналами. Программируемый умножитель частоты может работать в двух режимах:

- режим PLL: тактовый сигнал CLKIN умножается на один из 31 программируемого множителя от 0.25 до 15 с использованием блока умножителя частоты PLL;
- режим DIV: тактовый сигнал CLKIN делится на 2 или 4.

При старте процессора режим работы тактового генератора определяется состоянием трех внешних выводов CLKMD1 – CLKMD3. С их помощью можно задать основные режимы работы, такие, как поделить или умножить входной тактовый сигнал на 2 или 4. Конкретный список режимов, задаваемых с помощью входов CLKMD, отличается для различных моделей ЦСП и приведен в документации на конкретную модель процессора.

Очень эффективной поддержкой тактового генератора ЦСП C54xx является возможность программного задания режима работы и коэффициента умножения частоты с изменением этих параметров во время работы ЦСП. Возможность перепрограммирования частоты, на которой работает ЦСП, позволяет подобрать для каждого конкретного режима работы устройства свою тактовую частоту и соответственно производительность и энергопотребление ЦСП. Отметим, что программируемому блоку PLL требуется некоторое время на изменение режима работы, для отслеживания которого в состав тактового генератора входит специальный таймер.

Интерфейсы ЦСП семейства C5000

Сигнальные процессоры семейства C5000 имеют три типа интерфейсов (*рис.3*):

- интерфейс внешней памяти EMIF;
- интерфейс хост-порта HPI;
- 2 – 3 многоканальных буферизованных последовательных порта McBSP.

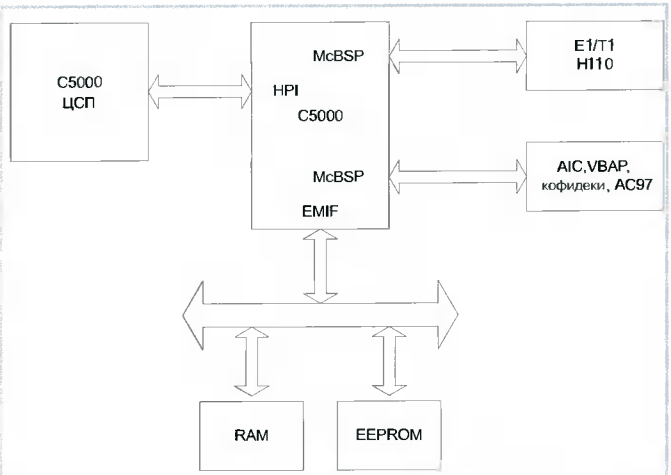


Рис. 3. Интерфейсы ЦСП семейства C5000

Интерфейс внешней памяти является продолжением шины адреса и шины данных процессора. К нему могут подключаться внешние ОЗУ, ПЗУ и порты периферийных устройств. Подключенные устройства могут адресоваться как память команд, память данных и порты ввода-вывода. Для определения типа обращения к внешним устройствам используются три сигнала выбора – PS (память команд), DS (память данных) и IS (порты ввода-вывода). Отметим, что большинство моделей ЦСП C54xx имеют возможность расширенной адресации памяти команд до 8 Мслов со специальными командами дальнего перехода и возврата.

Интерфейс хост-порта HPI

Интерфейс HPI – это интерфейс между ЦСП и управляющим хост-контроллером, где ЦСП подключается как периферийное устройство и управляется хост-контроллером. Хост-контроллер управляет интерфейсом через выделенные регистры адреса и данных, к которым ЦСП не имеет доступа, и регистр управления HPIC, который совместно используется хост-контроллером и ЦСП (*рис.4*).

Использование интерфейса хост-порта позволяет организовать управляющий и интерфейсный каналы загрузки программы в процессор практически без использования внутренних программных ресурсов.

Интерфейс HPI производит обмен 16-разрядными данными через экономичный внешний 8-разрядный интерфейс с автоматическим преобразованием разрядности данных. При передаче данных через HPI его внутренняя логика выполняет доступ к внутренним ОЗУ ЦСП.

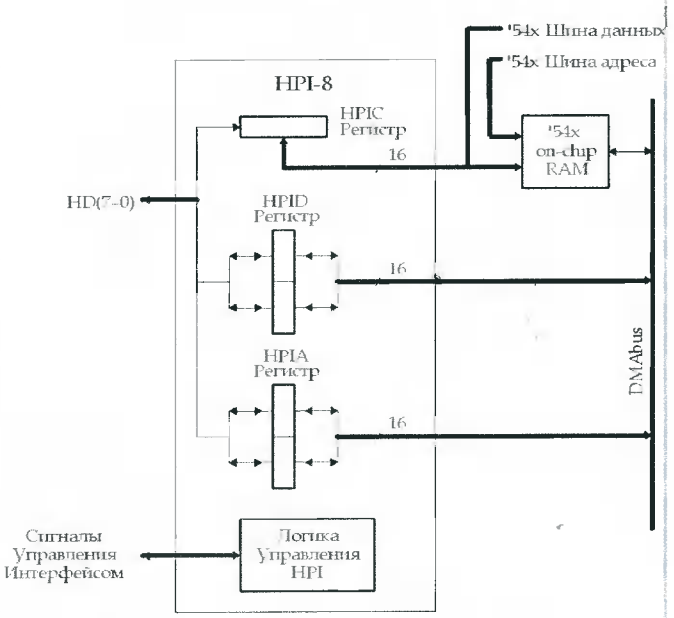


Рис. 4. Структура интерфейса хост-порта (HPI)

Внешний интерфейс HPI включает в себя 8 разрядов данных и сигналы управления. В большинстве случаев подключение интерфейса HPI к хост-контроллеру не требует никакой дополнительной логики (*рис.5*).

8 разрядов данных (HD0-HD7) служат для обмена данными. Поскольку ЦСП имеет 16-разрядную структуру, то каждый обмен должен содержать два последовательных байта, при этом какой байт передается (первый или второй), определяется выделенным выводом HBIL. Два контрольных входа определяют, какой из внутренних регистров будет доступен и тип доступа. Этими битами выбираются либо регистр управления, либо регистр адреса HPIA, либо регистр данных HPID.

В общем виде обмен по HPI производится следующим образом: после установки режима работы

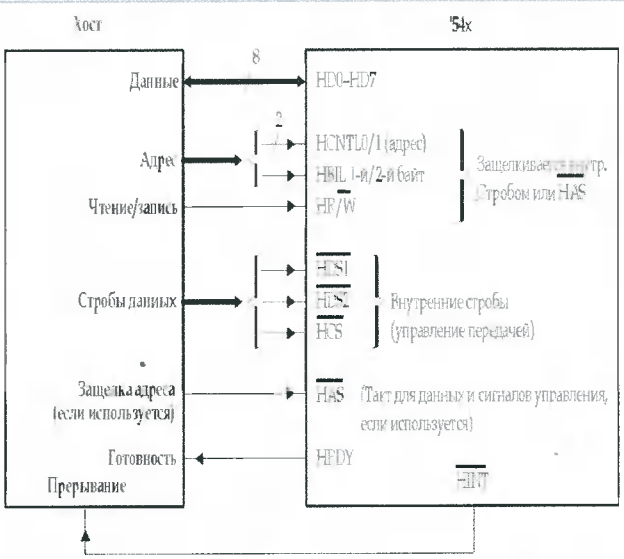


Рис. 5. Подключение контроллера к HPI

в регистре HPIC в регистр адреса, который является указателем на область памяти, доступную хост-порту, записывается адрес. Затем производится операция чтения или записи двух байтов через регистр HPID, которую внутренняя логика интерфейса преобразует к операциям чтения или записи слова из ОЗУ-процессора. При этом адресный регистр HPIA имеет возможность установки режима автоинкремента, в котором его содержимое изменяется на единицу после или перед каждой операцией обмена, что позволяет производить последовательный скоростной обмен массивами данных.

Хост-порт может передавать один байт каждые 5 тактов процессора, таким образом для 100 МГц процессора скорость обмена по HPI-8 может достигать 20 Мбайт/с. При этом для организации интерфейса используются всего 12 – 14 выводов, включая выводы адресации и управления.

Отметим, что при использовании выводов выбора HCS возможно параллельное подключение нескольких хост-портов, что дает возможность легкого подключения нескольких ЦСП к одному управляющему контроллеру.

Хост-контроллер и ЦСП могут обмениваться прерываниями. Записью в HPIC хост-контроллер вызывает прерывание в ЦСП. Для генерации прерывания от ЦСП к хост-контроллеру используется выделенный выход HINT. Хост может показывать реакцию на прерывание путем сброса HINT записью в HPIC.

Некоторые модели ЦСП семейства C5000 могут иметь еще и 16-битный HPI-интерфейс (HPI-16). В этом интерфейсе в качестве шины данных может использоваться внешняя 16-разрядная шина данных процессора. Интерфейс HPI-16 может иметь два режима – мультиплексный и не мультиплексный. В мультиплексном режиме, так

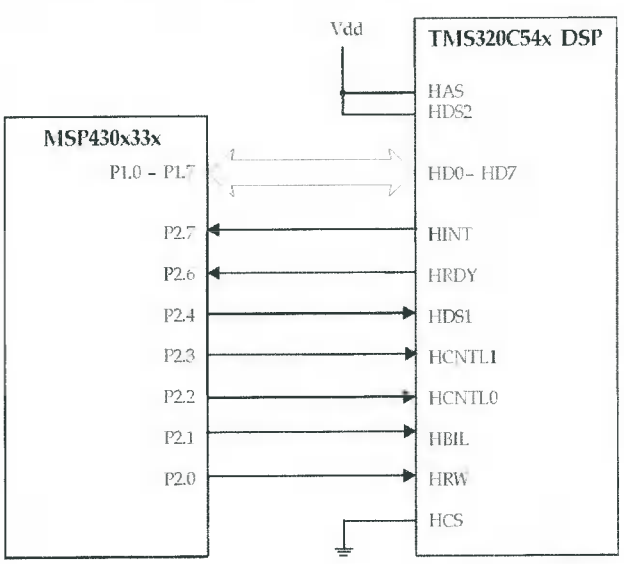


Рис. 6. Подключение HPI-интерфейса к микроконтроллеру MSP-430

же, как и в HPI-8, адрес задается специальным адресным регистром, а в не мультиплексном режиме адрес передается непосредственно на контроллер ПДП.

В качестве примера использования HPI покажем его подключение к контроллеру типа MSP430 (*рис.6*).

Старт процессора

Следующим вопросом, который мы рассмотрим, будет вопрос о начальном старте процессора. ЦСП семейства C5000 имеют достаточно большую скорость работы, поэтому желательно размещать исполняемую программу во внутренней памяти.

В масочном ПЗУ процессора содержится специальная программа начального загрузчика – bootloader. При старте процессора он может начать либо исполнение программы из внешней 16-разрядной памяти, либо запустить программу загрузчика. Метод старта определяется состоянием входа MP/MC процессора при опускании сигнала RESET.

Программа начального загрузчика осуществляет загрузку программы из указанного источника во внутреннее ОЗУ процессора и передачу управления загруженной программе. Начальная загрузка программы возможна из следующих источников:

- 8-разрядное ПЗУ;
- 16-разрядное ПЗУ;
- параллельный порт;
- хост-порт (HPI);
- последовательный порт;
- мягкий старт.

После запуска программа начального загрузчика устанавливает статусные регистры процессора в следующее состояние: прерывания запрещаются, все внутреннее ОЗУ отображается как в память данных, так и в память команд, для доступа ко всему внешнему ОЗУ устанавливается 7 тактов ожидания.

После этого по состоянию входа INT2 определяется, не используется ли режим загрузки из хост-порта.

Режим загрузки из хост-порта

Переход на режим загрузки из хост-порта определяется по отклику окружения процессора на выход прерывания HINT. Хост-процессор должен откликнуться на переход этого сигнала в низкое состояние переводом входа прерывания IT2 в низкое состояние. Наличие этого отклика и определяет переход начального загрузчика в режим загрузки из хост-порта. Простейший метод установки этого режима – соединение выхода HiT со входом IT2. При отсутствии такого соединения окружение процессора должно обеспечить корректное прерывание по входу IT2 в тече-

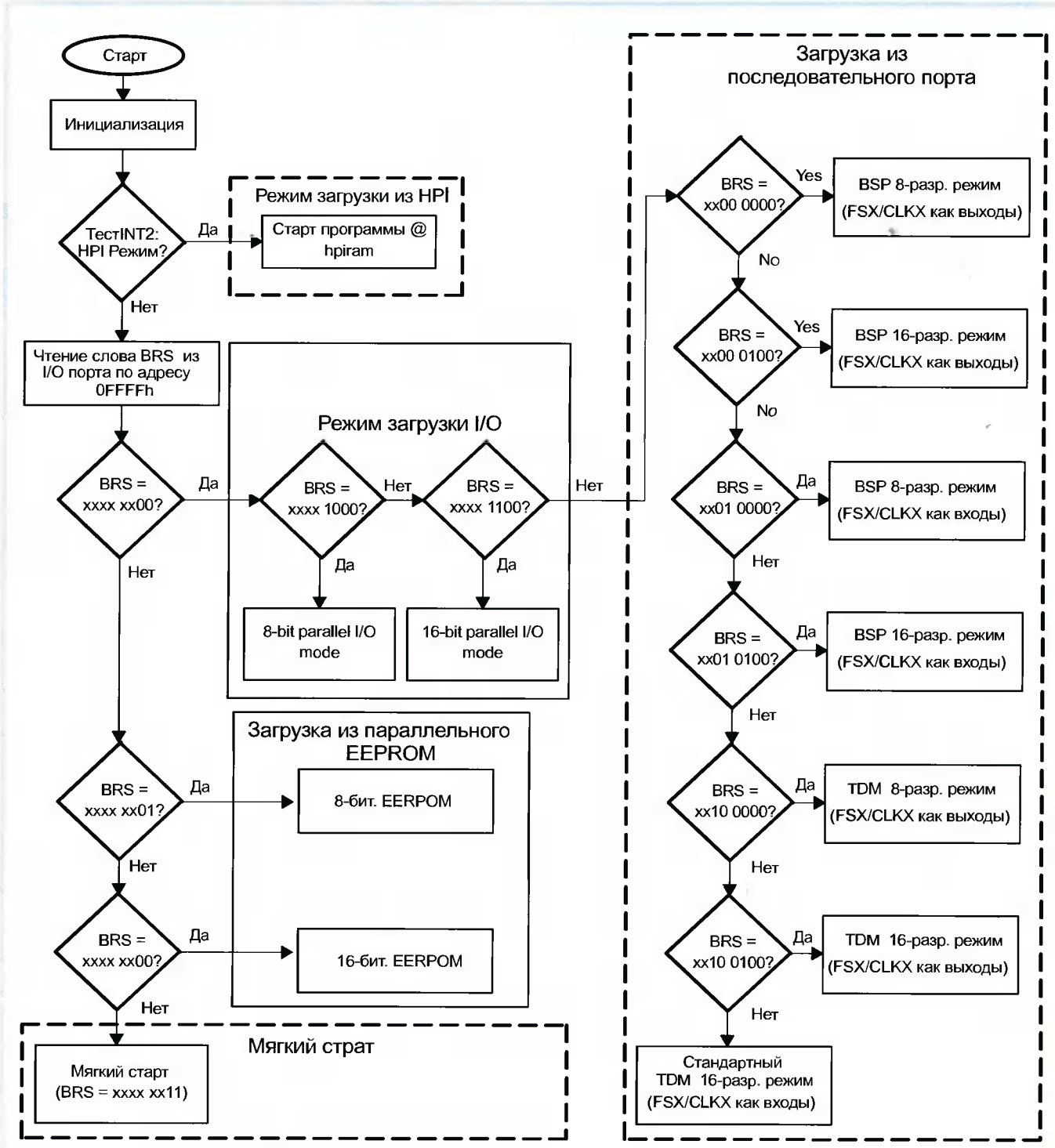


Рис. 7. Алгоритм определения источника начальной загрузки

ние 30 тактов после выхода снятия сигнала RESET.

После перехода в режим загрузки из хост-порта управляющий контроллер должен загрузить исполняемый код во внутреннюю память ЦСП. Во время загрузки из хост-порта выход прерывания HIT остается в низком состоянии. После окончания процесса загрузки хост-контроллер сигнализирует начальному загрузчику об окончании процесса загрузки сбросом выхода прерывания HIT с записью в регистр HPIC.

Затем производится чтение порта ввода-вывода по адресу 0FFFFh (опусканием строба S). Младшие 8 разрядов прочитанного значения определяют режим загрузки. Разряды последовательно декодируются от младшего к старшему в соответствии с алгоритмом, показанным на *рис.7*.

Режим загрузки из параллельного ПЗУ

Рассмотрим простейший вариант – на внешнюю шину памяти подключается одно 8-разряд-

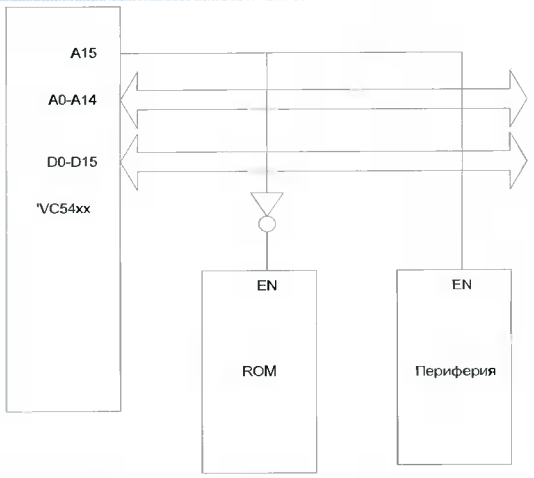


Рис. 8. Простейший вариант загрузки из 8-разрядного параллельного ПЗУ

ное загрузочное ПЗУ размером до 64 кбайт. При этом на это ПЗУ подключаются младшие адресные разряды и 8 младших разрядов шины данных. На вход разрешения выхода EN подключается сигнал чтения-записи R/W. Если больше к внешней шине ничего не подключено, то никаких дополнительных аппаратных средств не требуется. При запуске процессора начальный загрузчик выставляет на адресные шины адрес 0FFFFh и выполняет операцию чтения. При этом считывается содержимое старшей ячейки ПЗУ вне зависимости от ее размера. Как уже было сказано выше, младшие разряды считанного кода определяют режим загрузки, то есть туда должно быть записано значение xxxx xxx01b. Получив это значение, программа начального загрузчика переходит в режим загрузки из 8-разрядного ПЗУ, что и требовалось.

В случае наличия каких-либо еще устройств ввода-вывода простейшим вариантом будет разделение загрузочного ПЗУ и периферийных устройств путем дешифрации старшего разряда адреса, как показано на *рис. 8*. При этом остается только 32 К на загрузочное ПЗУ и 32 К на порты ввода-вывода, но в большинстве случаев этого более чем достаточно.

Многоканальный буферизованный последовательный порт McBSP

ЦСП C54xx имеют 2 или 3 (модель VC5441-12) многоканальных буферизованных последовательных порта (McBSP). Эти порты являются основными интерфейсами ЦСП для скоростного потокового ввода-вывода.

Порт McBSP является синхронным программируемым портом с широкими возможностями программирования схем тактирования и структуры данных. Порт может работать на тактовой частоте до $^1/_4$ частоты ЦСП, таким образом для 160 МГц ЦСП тактовая частота порта 40 МГц. При этом максимальная скорость обмена составляет практически 40 Мбит/с. Если перевести эти цифры на практическое применение, то можно рассчитать, что для 24-разрядного ЦАП или АЦП, подключенных к этому порту, частота дискретизации может составлять более 1.5 МГц. Как еще один пример, сравним 40 Мбит/с со скоростью потока E1, которая составляет 2 Мбит/с. Как видно, скорости последовательного порта вполне хватает для тех областей, в которых применяется ЦСП C54xx. Но максимальная скорость – это важный, но еще не определяющий фактор. Посмотрим на остальные возможности McBSP.

McBSP – это программируемое устройство, состоящее из независимых каналов приема и передачи. При этом поддерживаются независимые тактовые частоты и схемы синхронизации для приема и передачи. Порт поддерживает возможность обмена 8-, 12-,16-, 20-, 24- и 32-разрядными данными, при этом возможна передача первым младшего или старшего бита.

Интересной особенностью McBSP, показывающей его ориентацию на телекоммуникационное применение, является наличие встроенного в него блока u-Law и A-Law компандирования.

В названии порта имеется слово многоканальный, однако при рассмотрении схемы мы видим три линии для приемника (такт, данные и синхронизация) и три линии для передатчика. В чем же заключается многоканальность? Возможность эта заключается в том, что схемы тактирования порта имеют возможность “разбирать” часто применямые в телекоммуникационных системах мультифреймовые структуры с временным разделением, такие, как канал E1, и осуществлять прием и передачу только указанных фреймов. При этом поддерживается разделение до 128 каналов для приема и передачи.

Сочетание всех перечисленных функциональных возможностей позволяет осуществлять прямое подключение McBSP к целому ряду стандартных интерфейсов:

- T1/E1-фреймеры (*рис.9*);
- MVIP- и ST-BUS-совместимые устройства, такие, как:
  - MVIP-фреймеры
  - H.100-фреймеры
  - SCSA-фреймеры;

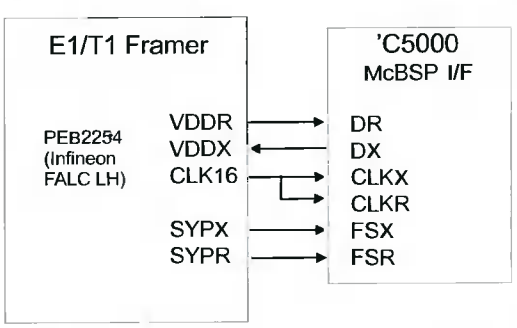


Рис. 9. Подключение McBSP к фреймеру E1

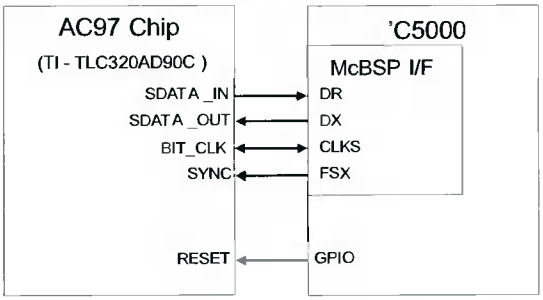


Рис. 10. Подключение McBSP к кодеку AC97

- IOM-2-совместимые устройства;
- AC97-совместимые устройства (*рис.10*);
- IIS-совместимые устройства.

А также к SPI-интерфейсу.

Наличие скоростного обмена и гибких схем тактирования полагает наличие продуманной и гибкой схемы обмена данными с памятью ЦСП, желательно с минимальным участием в этом процессе самого ядра, поскольку в противном случае все ресурсы ЦСП будут уходить на обмен, а не на обработку. Причем здесь уже плохо подходит стандартная для последовательных портов схема с двойной буферизацией и обработкой прерываний.

Для решения проблемы передачи скоростного потока данных через последовательный порт в McBSP введена такая возможность, как автобуферизация.

При этом, как показано на *рис.11,* в памяти ЦСП выделяется буфер для приема и буфер для передачи, границы которых задаются в специальном модуле автобуферизации. После чего схема последовательного порта без участия вычислительного ядра ЦСП производит обмен данными между этими областями памяти и последовательным портом. При этом прерывания от последовательного порта генерируются по достижении границы буфера. Такая схема обмена позволяет производить скоростной обмен данными с минимальным участием в этом процессе вычислительного ядра.

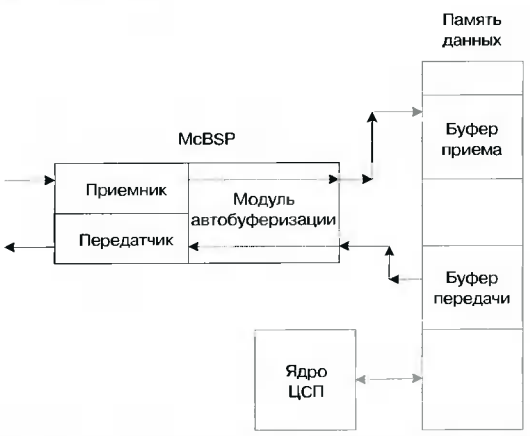


Рис. 11. Автобуферизация последовательного порта

Стыковка системы на базе ЦСП и персонального компьютера

Существует множество задач, которые требуют стыковки системы на базе ЦСП и персонального компьютера. С расширением области применения ЦСП и увеличением их мощности круг этих задач только расширяется. Кроме того, увеличиваются и требования к скорости обмена между ЦСП и PC. Если раньше вполне удовлетворительными решениями были порт RS-232 для внешнего устройства и шина ISA для внутреннего, то теперь ситуация радикально изменилась: ISA отмирает как таковая, а RS-232 уже не проходит по скорости обмена. В реальных условиях для подключения ЦСП-системы к PC используется три вида интерфейсов: параллельный порт и USB для внешних устройств и PCI для плат, устанавливаемых внутрь компьютера. Нельзя не отметить и набирающий силу интерфейс IEEE1394, но пока он не стал стандартом де-факто, имеющимся в каждом настольном компьютере.

В качестве интерфейса со стороны ЦСП для интеграции, как правило, используется 8- или 16-битный HPI, как интерфейс с наиболее простым протоколом подключения, в то же время обладающий достаточной пропускной способностью и встроенным механизмом арбитража. Гораздо реже, когда требуется обеспечить предельно высокую пропускную способность, используется подключение непосредственно к интерфейсу внешней памяти. Синхронные последовательные порты, как было показано выше, как правило, заняты кодеками или устройствами аналогово-цифрового преобразования, хотя их высокая пропускная способность позволяет реализовать такой вариант, особенно в случае внешнего устройства.

Подключение ЦСП к шине PCI

Интерфейс PCI является скоростным, очень широко распространенным и фактически единственным интерфейсом для устанавливаемых в PC дополнительных модулей.

Существует несколько вариантов стыковки ЦСП с PCI-шиной: с использованием специализированного PCI-to-DSP моста, с использованием микросхемы PCI-контроллера общего назначения и с использованием PCI-ядра, размещенного в FPGA. Каждый из этих вариантов имеет свои преимущества и недостатки.

Начнем с варианта использования специализированного моста. Поскольку подключение ЦСП к шине PCI является достаточно широко распространенной и массовой задачей, то вполне естественным ходом производителей как ЦСП, так и интерфейсных PCI-контроллеров является выпуск мостов между ЦСП и PCI. В качестве примера такого специализированного моста приведем микросхему PCI2040, выпускаемую фирмой TI. Это полностью законченное решение для подключения к шине PCI до 4 ЦСП через интерфейс HPI-8 или HPI-16.

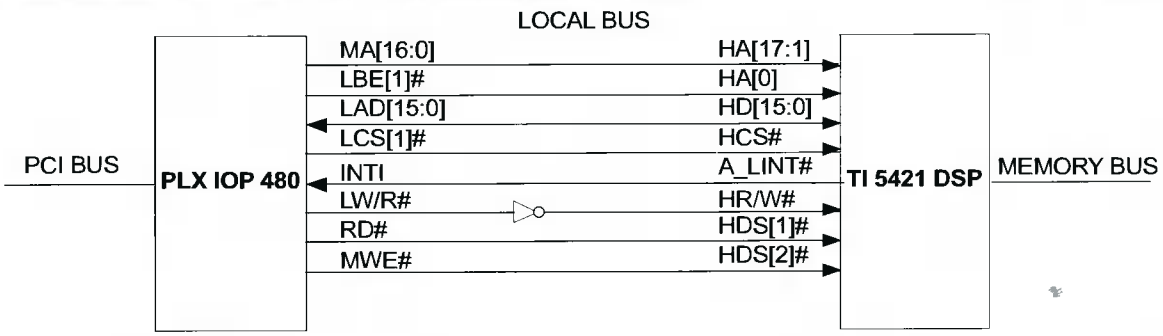


Рис. 12. Стыковка PCI-интерфейса PLX IOP 480 и TMS320VC5409

Мост PCI2040 содержит все необходимые компоненты для работы с шиной PCI, с одной стороны, и для прямого подключения до 4 ЦСП от TI, с другой стороны. Кроме того, PCI2040 имеет отдельную 16-разрядную шину общего назначения (GPB), которая может использоваться, например, для подключения JTAG-контроллера. Мост выпускается в корпусе 144-LQPF или 144-PBGA и имеет питание 3.3 В, но при этом сохраняет совместимость по выводам с 5 В PCI-интерфейсом. Для загрузки ID-системы и ID-производителя имеется интерфейс EEPROM.

Использование специализированного моста PCI2040 наиболее выгодно с экономической точки зрения из-за низкой стоимости самой микросхемы PCI-моста (менее $10) и практического отсутствия дополнительных компонентов.

Следующим вариантом является подключение ЦСП к микросхеме PCI-интерфейса общего назначения. Но тут уже разработчик должен сам подбирать соответствие интерфейса PCI-моста интерфейсу ЦСП и, как правило, добавлять дополнительную внешнюю логику для их стыковки. Хотя, как показано на примере, приведенном на *рис. 12*, гибкость и простота HPI-интерфейса позволяет решить эту проблему минимальными средствами.

Отметим, что такой вариант не всегда выгоден с экономической точки зрения. Обусловлено это тем, что микросхемы PCI-мостов общего назначения сами по себе достаточно дороги, как и все интерфейсные микросхемы общего назначения, не ориентированные на конкретную задачу и имеющие излишнюю функциональность, за которую надо платить.

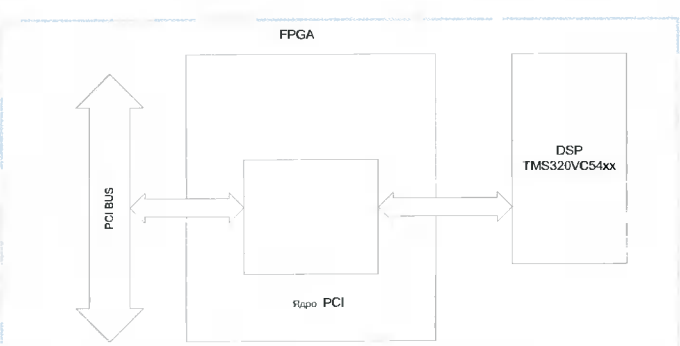


Рис. 13. Подключение ЦСП к шине PCI с использованием ядра PCI, размещенного в ПЛИС

Очень интересным вариантом как с экономической точки зрения, так и с точки зрения функциональности является использование для стыковки ЦСП и PCI-интерфейсного PCI-модуля, размещаемого в программируемой логической микросхеме (*рис. 13*). Увеличение емкости и уменьшение стоимости ПЛИС делают это решение уже заведомо дешевле PCI-контроллера общего назначения и, несмотря на то что ПЛИС, необходимая для размещения ядра PCI-контроллера, пока еще дороже специализированного моста PCI2040, такой вариант решения имеет ряд очень интересных технических возможностей.

Начнем с того, что, как видно из предыдущих примеров, для стыковки в общем случае используется интерфейс HPI либо потому, что на него рассчитан мост, либо потому, что к нему можно пристыковаться минимальными затратами. При использовании же ПЛИС возможна стыковка между ЦСП и PCI любым из имеющихся интерфейсов, в том числе последовательным портом с любой требуемой промежуточной логикой. Далее, хотя ПЛИС и дороже приведенного в примере специализированного моста, но в тот же корпус ПЛИС, где находится ядро PCI, на оставшееся место можно разместить еще и массу дополнительных компонентов устройства, которые могут быть легко сконфигурированы под конкретные нужды. При этом разрабатываемое устройство имеет минимальное количество внешних компонентов: при текущей низкой стоимости программируемой логики все может быть упаковано в один недорогой чип. Так, например, ПЛИС Xilinx серии Spartan-II емкостью 100 тыс. вентилей стоит около $20, а ядро PCI занимает в ней менее 10%.

Единственным возражением против такого подхода служит то, что само ядро PCI для ПЛИС надо либо разработать самому, либо купить готовое. Однако в настоящий момент сам модуль интерфейса PCI отечественной разработки фирмы SET стоит около $300.

Литература

1. SPRU684A.PDF Application Report: Migrating from TMS320VC5409 to TMS320VC5409A
2. PRU685A.PDF Application Report: Migrating from TMS320VC5410 to TMS320VC5410A
3. SLVA078A.PDF Application Report: Using the TPS5602 for Powering DSPs

С.В. Витязев

Стратегия компании Analog Devices в области DSP: обзор новых средств проектирования систем цифровой обработки сигналов

Введение

Компания Analog Devices Inc. (ADI) является одной из ведущих на мировом рынке производителей и поставщиков прецизионных и быстродействующих аналоговых интегральных схем, а также цифровых сигнальных процессоров (DSP). Российским специалистам хорошо известны сигнальные процессоры семейства SHARC и ADSP-21xx. Наряду с этими компонентами компания ADI предоставляет потребителям целую гамму специализированных DSP со встроенными аппаратными и программными средствами для таких применений, как xDSL-модемы, контроллеры электродвигателей, устройства удаленного абонентского доступа и распознавания речи. Отличительной особенностью поставок DSP является наличие для них полного набора надежных и удобных в использовании инструментальных средств поддержки разработки и оптимизации систем цифровой обработки сигналов (ЦОС) на базе процессоров, производимых компанией ADI. Это позволяет значительно ускорить процесс создания изделий и сократить время выхода продукции на рынок.

Проявляя постоянный повышенный интерес к инструментальным средствам поддержки проектирования и отладки систем ЦОС, компания ADI приобрела в феврале 1999 года двух ведущих поставщиков средств разработки цифровых систем обработки сигналов: фирму White Mountain DSP Inc. (США) и фирму Edinburgh Portable Compilers (Шотландия). До этого "слияния" указанные фирмы обеспечивали компанию ADI и ее клиентов самыми современными системами эмулирования и компилирования. Фирма White Mountain DSP была ведущим мировым производителем эмуляторов и программных средств отладки систем ЦОС. Фирма Edinburgh Portable Compilers специализировалась на создании компиляторов для сложных встраиваемых систем, а также на разработке программных компонентов генерации кодов микропроцессоров фирмы Intel и других производителей. Компиляторы и средства отладки – это главные инструменты, которые нужны потребителям для разработки систем на базе DSP, а предприятиям-партнерам – для создания средств поддержки разработок. Приобретение этих фирм значительно увеличило потенциальные возможности компании ADI благодаря объединению собственного опыта и достижений в разработке архитектур DSP с опытом фирм White Mountain DSP и Edinburgh Portable Compilers в области создания средств разработки и отладки систем ЦОС. Поэтому включение их в свой состав позволило ADI взять на себя роль лидера в разработке новых перспективных средств проектирования систем ЦОС.

При создании инструментальных средств поддержки, поставляемых компанией ADI, главное внимание было сконцентрировано на обеспечении инженеров-разработчиков простыми в использовании и надежными в работе инструментами проектирования, отладки и оптимизации устройств на базе DSP в целях максимального ускорения процесса создания конечных изделий и выпуска их на рынок. Набор инструментов включает в себя интегрированную среду разработки VisualDSP, средства генерации кода, программные симуляторы, оценочные платы и аппаратные эмуляторы. В *табл. 1* приведены краткие данные по всем инструментальным средствам разработки систем ЦОС, которые предлагает компания ADI.

К числу новейших разработок инструментальных средств проектирования и отладки систем ЦОС относятся:

VisualDSP++ – интегрированная среда разработки (IDE), которая пополнила семейство программных средств для сигнальных процессоров ADI. Новая среда IDE создана на базе программного пакета VisualDSP, в который введена поддержка языка C++, что позволило значительно расширить число программистов, эффективно реализующих алгоритмы ЦОС в реальном времени на сигнальных процессорах компании ADI.

ADSP-21160 EZ-KIT Lite™ – простой в использовании аппаратно-программный комплект начального освоения, который позволяет легко изучить все возможности ADSP-21160 – наиболее производительного процессора в семействе SHARC, а также начать разработки систем на базе этого процессора. Комплект состоит из автономной оценочной (макетной) платы, которая может быть использована как самостоятельное устройство, и специальной версии программного пакета VisualDSP с несколько ограниченными функциональными возможностями.

ADSP-2189M EZ-KIT Lite™ – аппаратно-программный комплект начального освоения, который состоит из автономной оценочной платы с процессором ADSP-2189M и инструментального пакета, обеспечивающих генерацию кода и возможность отладки программ. Этот продукт способствует популяризации и продвижению на рынок высокопроизводительных процессоров с архитектурой ADSP-218x, а также эффективной сре-

Таблица 1

Процессор	Аппаратно-программный комплект отладки	Эмулятор	Пакет программ разработки
ADSP-2101	ADDS-21XX-EZLITE	ADDS-2101-EZ-ICE	ADDS-21XX-PC-1
ADSP-2104	ADDS-21XX-EZLITE	ADDS-2101-EZ-ICE	ADDS-21XX-PC-1
ADSP-2105	ADDS-21XX-EZLITE	ADDS-2101-EZ-ICE	ADDS-21XX-PC-1
ADSP-2115	ADDS-21XX-EZLITE	ADDS-2101-EZ-ICE	ADDS-21XX-PC-1
ADSP-216x			ADDS-21XX-PC-1
ADSP-217x	ADDS-2171-EZ-LAB ADDS-21XX-EZLITE	ADDS-2171-EZ-ICE	ADDS-21XX-PC-1
ADSP-2181	ADSP-2189M-EZLITE	ADDS-218X-ICE-2.5	VDSP-21XX-PC-FULL
ADSP-2183	ADSP-2189M-EZLITE	ADDS-218X-ICE-2.5V	VDSP-21XX-PC-FULL
ADSP-2184	ADSP-2189M-EZLITE	ADDS-218X-ICE-2.5V	VDSP-21XX-PC-FULL
ADSP-2185	ADSP-2189M-EZLITE	ADDS-218X-ICE-2.5V	VDSP-21XX-PC-FULL
ADSP-2186	ADSP-2189M-EZLITE	ADDS-218X-ICE-2.5V	VDSP-21XX-PC-FULL
ADSP-2187	ADSP-2189M-EZLITE	ADDS-218X-ICE-2.5V	VDSP-21XX-PC-FULL
ADSP-2189M	ADDS-2189M-EZLITE	ADDS-218X-ICE-2.5V	VDSP-21XX-PC-FULL
ADSP-2192-12 DSP	ADSP-2192-12EZLITE	ADDS-APEX-ICE ADDS-TREK-ICE ADDS-SUMMIT-ICE ADDS-MTN-ICE	VDSP-21XX-PC-FULL
ADSP-21020	ADDS-21020-EZ-LAB	ADDS-21020-EZ-ICE	ADDS-21020-SW-PC
ADSP-21060	ADSP-2106X-EZLITE	ADDS-APEX-ICE ADDS-TREK-ICE ADDS-SUMMIT-ICE ADDS-MTN-ICE	VDSP-SHARC-PC-FULL
ADSP-21061	ADSP-2106X-EZLITE	ADDS-APEX-ICE ADDS-TREK-ICE ADDS-SUMMIT-ICE ADDS-MTN-ICE	VDSP-SHARC-PC-FULL
ADSP-21062	ADSP-2106X-EZLITE	ADDS-APEX-ICE ADDS-TREK-ICE ADDS-SUMMIT-ICE ADDS-MTN-ICE	VDSP-SHARC-PC-FULL
ADSP-21065L	ADDS-21065L-EZLITE	ADDS-APEX-ICE ADDS-TREK-ICE ADDS-SUMMIT-ICE ADDS-MTN-ICE	VDSP-SHARC-PC-FULL
ADSP-21160M	ADDS-21160M-EZLITE	ADDS-APEX-ICE ADDS-TREK-ICE ADDS-SUMMIT-ICE ADDS-MTN-ICE	VDSP-SHARC-PC-FULL
ADSP-TS001M	ADDS-TS001-EZLITE	ADDS-MTN-ICE ADDS-SUMMIT-ICE ADDS-APEX-ICE ADDS-TREK-ICE	VDSP-TS-PC FULL

ды программирования устройств цифровой обработки сигналов – пакета VisualDSP. Имея ADSP-2189M EZ-KIT Lite, пользователь может изучить и аппаратные, и программные средства DSP, практически испытать и оценить возможности реального воплощения различных устройств на базе процессоров семейства ADSP-218x.

Apex-ICE™ – это первый промышленный эмулятор для DSP, подключаемый к персональному компьютеру через USB-порт. Вместе с тем это первое мобильное средство отладки сигнальных процессоров Analog Devices, так как USB-интерфейс обеспечивает простой и эффективный способ подключения как к настольным, так и к портативным компьютерам.

Trek-ICE™ – это первый промышленный продукт в классе сетевых эмуляторов для DSP. Он подключается через Т-разъем к локальной сети Ethernet-10/100 и обеспечивает возможность дистанционной отладки устройства пользователя с удаленного персонального компьютера или рабочей станции Sun. Эмулятор Trek-ICE работает на ПК под управлением ОС Windows®, Windows NT и на рабочей станции Sun под управлением

ОС Solaris 2.x. Он обеспечивает режим неразрывной эмуляции всех типов сигнальных процессоров Analog Devices с интерфейсом JTAG, используя для связи протокол TCP/IP.

Компания Analog Devices стремится к непрерывному расширению поставок инструментальных средств для инженеров – разработчиков систем на базе DSP и продвижению их на рынок по всему миру.

Ниже обсуждаются:

- основные программные и аппаратные средства проектирования и отладки систем ЦОС, предоставляемые компанией ADI;
- отличительные особенности и возможности новых средств разработки;
- ключевые технологии будущего, требуемые для создания эффективных средств проектирования и отладки систем ЦОС на базе процессоров нового поколения.

Статья подготовлена по материалам журнала Analog Devices' DSP Connection, v. 4, 2000.

Интегрированная программная среда VisualDSP

VisualDSP – высокоэффективная программная среда, предназначенная для разработок в области цифровой обработки сигналов, является в некотором смысле системой управления проектированием систем ЦОС различного назначения. Она содержит в своем составе интегрированную оболочку, включающую редактор и построитель системы, которые работают совместно с высокоэффективным отладчиком и дают возможность контроля разработки проекта от начала до конца в рамках единого интерфейса. Среда VisualDSP служит для разработчика графическим интерфейсом (*рис. 1*), отображающим весь процесс проектирования и отладки систем ЦОС с использованием программы-симулятора, аппаратно-программных средств макетирования и оценивания EZ-KIT Lite и эмуляторов DSP.

Интегрированная среда VisualDSP обладает следующими характерными свойствами:

Единство интерфейса – единый интерфейс поддерживается полностью для всех применяемых средств.

Простое редактирование – средствами оболочки создаются и редактируются исходные файлы программ и выполняется построение системы ЦОС (при этом поддерживаются стандартные для современных текстовых редакторов приемы редактирования).

Эффективное управление проектированием – параметры проекта могут быть легко заданы и изменены в процессе построения и отладки системы. Опции для компилятора, ассемблера и редактора связей являются легко устанавливаемыми и модифицируемыми для каждого конкретного проекта или файла.

Широкие возможности отладчика – отладчик полностью поддерживает языки С и ассемблер, предоставляет многофункциональную панель инструментов стандартного вида, обладает следующими, ставшими уже стандартом, свойствами:

1. Избирательный мультипроцессорный режим – среда VisualDSP предоставляет возможность выбора интересующих разработчика реги-

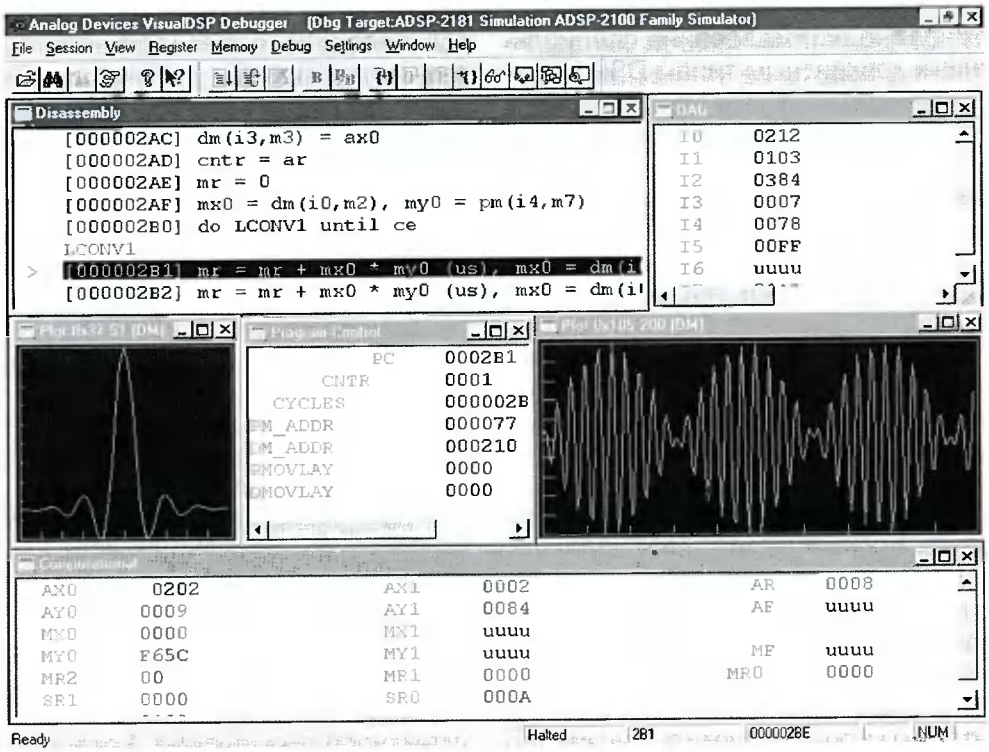


Рис. 1. Графический интерфейс отладчика программной среды VisualDSP

стров или области памяти каждого из процессоров многопроцессорной системы, отметив их как подлежащие наблюдению и внесению изменений между прогонами программы целиком, с остановками или в пошаговом режиме выполнения при трассировке. Это позволяет избежать возможной "неразберихи" на экране при работе отладчика в мультипроцессорном режиме работы.

2. Интерфейс командного режима тестирования – этот инструмент облегчает и ускоряет набор повторяющихся последовательностей операций отладки. Он подобен языку С по своему интерфейсу и синтаксису, хорошо документирован и предоставляет возможность эффективной подготовки программ, в полной мере тестирующих систему ЦОС.

Среда VisualDSP претерпела ряд значительных усовершенствований, направленных на уменьшение времени, затрачиваемого на проектирование и отладку систем ЦОС и ускорение выхода на рынок готовой продукции. Новая версия среды VisualDSP – интегрированная программная оболочка VisualDSP++, обеспечивает расширенные возможности разработки программ для всех семейств DSP, производимых компанией ADI, включая SHARC, TigerSHARC, ADSP-218x и ADSP219x. Включенный в VisualDSP компилятор языка C++ был разработан специально для потребителей DSP и поддерживает предложенный стандарт языка для встраиваемых устройств (Embedded C++), который определяет подмножество полного языка C++ по стандарту ISO/IEC. Имея компилятор C++, разработчики получают значительный выигрыш в сроках выпуска продукции на рынок, что связано с возможностью эффективной обработки различных типов данных, а также с возможностью использования специальных, свойственных конкретным типам DSP, инструкций цифровой обработки без детального представления об архитектуре этих DSP.

Компиляторы DSP и симуляторы

Переходя к описанию следующего блока программных средств поддержки проектирования систем ЦОС, отметим прежде всего, что на ранних стадиях разработки системы широко используются программы-симуляторы, моделирующие архитектуру и систему команд сигнального процессора в отлаживающей среде. Сейчас доступны симуляторы любых сигнальных процессоров компании ADI. Симулятор – это чисто программный инструмент, позволяющий разработчику проанализировать работу процессора, не имея самого процессора. Они повсеместно используются в системах ЦОС по двум причинам. Во-первых, написание алгоритма цифровой обработки сигналов сводится к сложной математической задаче, решая которую можно допустить множество ошибок. Симулятор дает разработчику возможность реализовать алгоритм на программной модели процессора и проанализировать качество алгоритма при его пошаговом исполнении. Во-вторых, появляется возможность провести первый анализ работы системы без существенных затрат на приобретение какого-либо аппаратного обеспечения, что особенно важно для разработчиков, мало знакомых с ЦОС. Все симуляторы, предоставляемые компанией ADI, обеспечивают высокую точность моделирования эффекта конвейерного выполнения команд и взаимодействия между центральным процессором, памятью и периферией.

Перед использованием симулятора разработчик должен написать программу (или программы) на языке С или языке ассемблер DSP. Наличие С-компилятора процессора следует выделить особо. Это связано с тем, что наряду с использованием ассемблера, непосредственно предназначенного для программирования конкретного DSP, часто возникает необходимость иметь эффективный компилятор с языка С, поскольку именно на С написаны многие стандартные алгоритмы обработки речи и данных. Кроме того, желание уменьшить сроки разработки систем "требует" от разработчиков работать с языком С на всем интервале проектирования и лишь для окончательной высокоточной настройки, если потребуется, использовать ассемблер. Таким образом, возможность эффективно использовать С-компилятор является важным критерием сегодняшних программных средств проектирования систем на DSP. Эффективность определяется как отношение числа циклов оптимального алгоритма (программы), написанного "вручную" на ассемблере, к их числу при использовании компилятора языка С.

Новые версии компиляторов DSP компании ADI являются более эффективными, поскольку реализуют методы оптимизации как общего назначения, так и ориентированные на архитектуру сигнального процессора. Назначение компилятора в том, чтобы непосредственно работать с С-кодами, описывающими алгоритм, а не с кодами, которые были декодированы для согласования с архитектурой процессора. Например, компилятор DSP для процессора ADSP-2106x (SHARC) переводит операции матричного умножения, входящие во внутренний цикл КИХ-фильтра, в единственную мультифункциональную команду, исполняемую с нулевыми потерями на организацию циклической обработки.

Компилятор DSP поддерживает многочисленные расширения, чтобы дать программисту больший доступ к особенностям архитектуры процессора. Например, под управлением пользователя компилятор процессора ADSP-21160 реализует преобразование одного цикла в два цикла, которые будут выполняться в режиме SIMD (одна команда – много данных), предусмотренном архитектурой процессора. Кроме того, компилятор анонсированного недавно процессора TigerSHARC ADSP-TS001 включает расширения

для работы с 16-битными массивами данных, чтобы оптимизировать выполнение векторных операций.

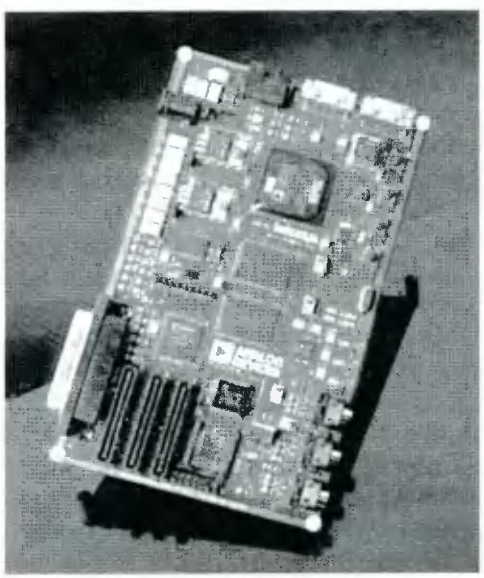
Оценочные (макетные) модули EZ-KIT-Lite

ADSP-21160 EZ-KIT Lite – оценочный модуль для процессоров ADSP-2116xM SHARC компании Analog Devices

Модуль ADSP-21160 EZ-KIT Lite позволяет легко оценить возможности процессора ADSP-21160M SHARC с использованием мощной программной среды для ряда общих целей при разработке высокопроизводительных цифровых систем той или иной области применения. Модуль включает оценочную плату для ПК (разработанную специально для ADI компанией Spectrum Signal Processing: www.spectrumsignal.com/EZ-KIT) и программную отлаживающую среду, обеспечивающую удобство работы с оценочной платой через интерфейс персонального компьютера. Имея EZ-KIT Lite, пользователь получает возможность изучить программные и аппаратные средства разработки систем ЦОС на базе процессора ADSP-21160M и наметить области применения.

EZ-KIT Lite имеет полный набор программных инструментов проектирования, включающий компилятор C, ассемблер, редактор связей (линковщик) и последнюю версию программы VisualDSP++. Все программные продукты ограничены использованием только с EZ-KIT Lite.

С модулем EZ-KIT Lite поставляются демонстрационные программы, которые включают известные алгоритмы ЦОС, такие, как дискретное преобразование Фурье и алгоритмы фильтрации. Дополнительно поставляются программы, демонстрирующие использование аппаратных особенностей процессора ADSP-21160M. Они касаются прерываний и кодека платы.



Характеристики:

- процессор ADSP-21160 SHARC;
- 64 на 64K SBSRAM;
- программы для оценки из пакета VisualDSP++;
- JTAG-интерфейс процессора системы;
- AD1881 16-битный стереокодек AC'97 Soundmax;
- сертификат CE.

Требования к системе:

- процессор Pentium 166 МГц и выше;
- минимум 32 Мбайта ОЗУ;
- Windows 95, Windows 98 или Windows NT 4.0;
- один EPP-параллельный порт.

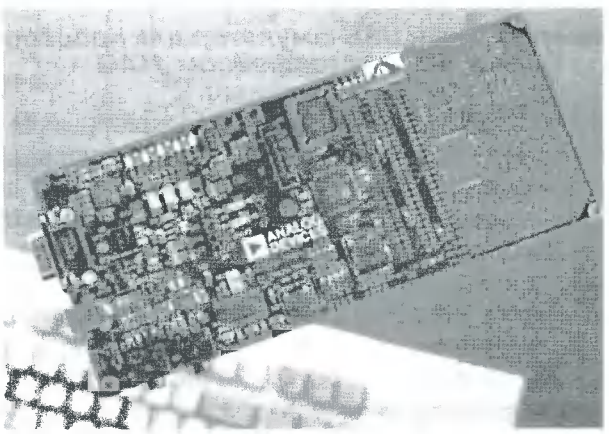
ADSP-2189M EZ-KIT Lite – оценочный модуль для процессоров семейства ADSP-218x компании Analog Devices

Модуль ADSP-2189M EZ-KIT Lite включает автономную оценочную плату на базе процессора ADSP-2189M с полнофункциональной генерацией кодов и программу-отладчик. Модуль делает удобной оценку архитектуры процессора, а также его окружения для семейства ADSP-2189M. Оценочный модуль EZ-KIT Lite имеет полный набор программных инструментов проектирования, включающий компилятор C, ассемблер, редактор связей (линковщик) и последнюю версию программы VisualDSP++. Все программные продукты ограничены использованием только с EZ-KIT Lite.

С модулем EZ-KIT Lite поставляются демонстрационные программы, которые включают известные алгоритмы ЦОС, такие, как свертка и вычисления Фибоначчи. Также поставляются программы, демонстрирующие использование аппаратных особенностей процессора ADSP-2189M. Они касаются прерываний, переполнений, таймеров и кодека платы.

Характеристики:

- процессор ADSP-2189M 75MIPS;
- поддержка процессоров семейства ADSP218x;
- программы для оценки из пакета VisualDSP++;
- расширенный разъем;
- поддержка по выбору 2.5 или 3.3 Вход/Выход;



- порт интерфейса RS-232;
- кодек AD 73322;
- сертификат CE.

Требования к системе:

- процессор Pentium 166 МГц и выше;
- минимум 32 Мбайта PC-AT памяти;
- Windows 9x, Windows 2000 или Windows NT 4.0;
- один RS-232-совместимый параллельный порт.

Эмуляторы

Эмулятор DSP, поддерживаемый мощной средой проектирования, подобной VisualDSP, предоставляет разработчику доступ к любым процессам, происходящим в реальном отлаживаемом процессоре. Эмуляторы, используемые компанией ADI, были разработаны фирмой White Mountain. Они обеспечивают доступ к процессору и его окружению через JTAG-порт. Такой порт характерен для всех современных процессоров Analog Devices. Он дает возможность наблюдения в реальном времени всей внутренней работы процессора.

Эмуляторы различаются по своим формам и размерам. Особенно важным является различие в интерфейсе с отлаживающим ПК. Он может осуществляться через шину стандарта ISA, характерную для ПК старого образца, шину PCI, USB-шину и Ethernet – это набор стандартных предложений компании ADI.

Новым направлением является разработка схемы "клиент – сервер" для передачи по сети информации, используемой для отладки и эмуляции. В этом случае разработчик может производить отладку системы, находясь от нее на большом расстоянии. Локальные сети делают удобной отладку распределенных систем, а также обеспечивают доступ к системе из различных пунктов. В частности, все более популярный эмулятор Trek-ICE обеспечивает связь через Ethernet-сеть. Его подключают к локальной сети, и отлаживаемая система становится доступной из любой точки этой сети.

Apex-ICE™

Универсальный USB-эмулятор для процессоров ADI с интерфейсом JTAG

Apex-ICE™ – это первый промышленный эмулятор для DSP, подключаемый к персональному компьютеру через USB-порт. Вместе с тем это первое мобильное средство отладки сигнальных процессоров Analog Devices с интерфейсом JTAG, так как возможность работы через порт USB обеспечивает простой и эффективный способ подключения как к настольным, так и к портативным компьютерам. В качестве программного обеспечения эмулятора используется интегрированная среда VisualDSP++™ под управлением ОС Windows 98 и Windows 2000.

Характеристики:

- миниатюрный корпус;
- длина кабеля от хост-компьютера до эмулятора 5 м;
- работает под управлением ОС Windows 98 и Windows 2000;
- поддерживает ADSP-2106x, ADSP-2116x и все новые процессоры ADI с интерфейсом JTAG;
- сертифицирован по CE.

Требования к системе:

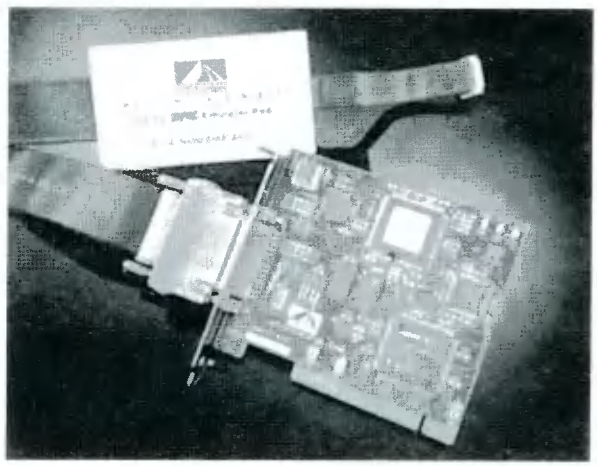
- процессор Pentium 166 МГц или выше;
- операционная система Windows 98 или Windows 2000;
- минимум 32 Мбайт оперативной памяти для PC-AT;
- один полноскоростной порт USB.

Trek-ICE™

Сетевой эмулятор для DSP Analog Devices с интерфейсом JTAG

Trek-ICE™ – это эмулятор для работы в локальной сети Ethernet, обеспечивающий возможность дистанционной отладки устройств на базе цифровых сигнальных процессоров компании ADI с интерфейсом JTAG. Через стандартный для сети Ethernet 10/100BASE-T разъем Trek-ICE подключается к кабелю, который служит для связи хост-компьютера системы удаленной эмуляции (PC или рабочей станции Sun) с отлаживаемым устройством на базе DSP.

Trek-ICE размещается в малогабаритном корпусе, который похож на типовой корпус mini-tower персональных компьютеров. Все разъемы для подключения к пользовательской локальной сети и отлаживаемому устройству выведены на заднюю панель. Trek-ICE был создан для работы в локальной сети 10/100BASE-T, которая имеется на большинстве предприятий. Для подключения к сети применяется разъем телефонного типа RJ-45. На сетевом уровне Trek-ICE использует протокол TCP/IP, который стал общепринятым для большинства локальных сетей, а также Internet.



Trek-ICE предназначен для осуществления дистанционной (удаленной) отладки. Многие крупные компании организуют работу в режиме распределенного доступа, используя для связи между расположенными в разных местах отделениями арендуемые и корпоративные закрытые сети, покрывающие расстояния в сотни и, возможно, даже в тысячи миль. Для таких потребителей использование Trek-ICE может быть особенно эффективным, так как позволяет проводить дистанционную отладку систем на базе DSP. Например, потребители, имеющие удаленные телекоммуникационные узлы с DSP и выделенные каналы подключения к сети, могут применить Trek-ICE для загрузки новых версий программного обеспечения или для выполнения системной диагностики. В качестве программного обеспечения можно использовать операционную среду VisualDSP++™.

Характеристики:

- сетевой эмулятор в корпусе mini-tower, подключаемый к сети Ethernet 10/100BASE-T;
- прочный и гибкий эмуляционный кабель длиной 5 футов с переходным устройством обеспечивают высокую скорость передачи данных, согласуются с 3В-/5В-схемами;
- поддерживаются процессоры ADSP-2106x, ADSP2116x и все JTAG DSP;
- программные драйверы для Windows 9x и Windows NT;
- сертифицирован по CE.

Требования к системе:

- процессор Pentium® 166 МГц и выше или рабочая станция Sun SPARC;
- минимум 32 Мбайт ОЗУ;
- операционные системы Windows® 9x, Windows 2000, Windows NT 4.0 SP3 и выше, Solaris 2.x;
- Windows NT 4.0 SP3 DHCP- сервер для обеспечения IP-адреса.

Summit-ICE™

Эмулятор на шину PCI для DSP с интерфейсом JTAG

Summit-ICE™ – это первый промышленный эмулятор, подключаемый к шине PCI персональных компьютеров, который был создан для поддержки разработок на базе DSP Analog Devices, имеющих интерфейс JTAG. Переходное устройство на кабель эмулятора имеет защиту входных цепей и допускает работу с уровнями сигналов 3В и 5В. Это максимально снижает влияние эмулятора, подключенного к JTAG-порту DSP, на работу отлаживаемого изделия. Благодаря тому, что интерфейс PCI стал фактически стандартным для всех современных компьютеров, выбор эмулятора Summit-ICE™ для отладки устройств на базе DSP Analog Devices будет вполне естественным. В качестве программного обеспечения при работе с эмулятором используется интегрированная среда VisualDSP++™ под управлением одной из операционных систем: Windows® 9x, Windows 2000, Windows NT 4.0.

Характеристики:

- интерфейс PCI;
- прочный гибкий кабель длиной 5 футов, переходное устройство, допускающее работу с уровнями сигналов 3В и 5В;
- установка в режиме Plug and Play;
- поддерживаются разработки на базе DSP семейств ADSP-2106x, ADSP-2116x и всех новых DSP Analog Devices с интерфейсом JTAG
- имеет сертификат CE.

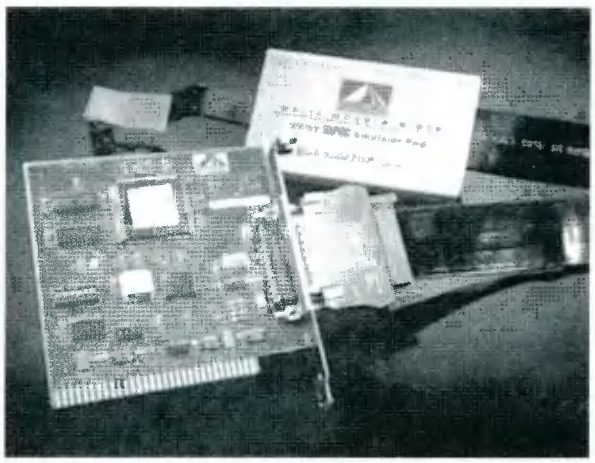
Требования к системе:

- процессор класса Pentium® 166 МГц или выше;
- минимальный объем оперативной памяти 32 Мбайт;
- операционная система Windows 9x, Windows 2000 или Windows NT (4.0 и выше);
- один 32-разрядный слот PCI.

Mountain-ICE™

Эмулятор на шину ISA для DSP с интерфейсом JTAG

Mountain-ICE™ – это первый промышленный эмулятор, основанный на периферийном сканировании через интерфейс JTAG, разработан специально для поддержки разработок на базе множества DSP, которые производит Analog Devices.



Специальный переходный модуль, связывающий основной соединительный кабель со шлейфом на разъем JTAG, допускает подачу сигналов с уровнями 3В и 5В. Это позволяет подключать эмулятор в процессе отладки к любым платам с DSP Analog Devices, имеющим порт JTAG, не нарушая их работу. На плате устройства, вставляемой в слот ISA, смонтирован специальный модуль в индустриальном стандарте ICEPAC™, который и выполняет основные функции эмуляции. Этим достигается повышенная надежность эмулятора, необходимая профессиональным разработчикам систем на базе DSP. В качестве программного обеспечения при работе с эмулятором используется интегрированная среда VisualDSP++™ под управлением одной из операционных систем: Windows® 9x, Windows 2000, Windows NT.

Характеристики:

- интерфейс ISA;
- прочный гибкий кабель длиной 5 футов, переходное устройство, допускающее работу с уровнями сигналов 3В и 5В;
- конструкция на базе встраиваемых модулей по технологии ICEPAC имеет повышенную надежность;
- поддержка разработок на базе DSP семейств ADSP-2106x и всех новых DSP с интерфейсом JTAG;
- наличие сертификата CE.

Требования к системе:

- процессор класса Pentium® 166 МГц или выше;
- минимальный объем оперативной памяти 32 Мбайт;
- операционная система Windows 9x, Windows 2000 или Windows NT;
- один 8-разрядный слот ISA.

Эмулятор с последовательным интерфейсом для семейства ADSP-218x

ADSP-218x EZ-ICE® – это эмулятор для отладки систем на базе 16-разрядных DSP Analog Devices из семейства ADSP-218x, имеющий последовательный интерфейс RS-232 для подключения к компьютеру через COM-порт. Он обеспечивает наиболее эффективную эмуляцию, предоставляя в распоряжение разработчика настраиваемую среду, которая позволяет наблюдать текущее состояние устройства пользователя, производить его отладку и тестирование. ADSP-218x EZ-ICE подключается непосредственно к эмулируемому процессору через специальный интерфейс. В качестве программного обеспечения при работе с эмулятором используется интегрированная среда VisualDSP++™.

Характеристики:

- поддержка всех DSP Analog Devices из семейства ADSP-218x;
- поддержка высокоскоростного режима обмена через последовательный порт RS-232;
- наличие защитного корпуса с кнопкой сброса;
- светодиодная индикация установленных уровней сигналов интерфейса ввода/вывода;
- поддержка DSP с напряжениями питания 2,5 В, 3,3 В и 5В.

Требования к системе:

- процессор класса Pentium® 166 МГц или выше;
- минимальный объем оперативной памяти 32 Мбайт;
- операционная система Windows 9x, Windows 2000 или Windows NT 4.0;
- один последовательный порт RS-232.

Разработки будущего

Новые разработки средств отладки, предлагаемые компанией ADI, будут ориентированы на повышение эффективности программирования, что является необходимым в современных жестких к срокам выполнения заказов условиях рынка. Ниже приведены области, в которых намечено проводить наиболее активные исследования и разработки.

Компиляторы языка С для DSP – роль этого инструмента в наборе средств проектирования будет все более значительной. Необходимость ускорения процесса разработки требует, чтобы базовые коды большинства DSP были взяты из уже готовых коммерческих алгоритмов или из ра-

нее разработанных версий проекта. Такие приемы, как анализ всех уже имеющихся программ, будут использоваться для ускорения разработки.

Использование корпоративных и расширенных локальных сетей – это должно стать стандартным инструментом разработчиков. Чтобы повысить производительность и в особенности гибкость программирования, отладочные средства ЦОС должны быть также естественно связанными с сетевыми технологиями, как и другие программы и устройства, которыми мы пользуемся ежедневно.

Использование языков высокого уровня – несомненно требует присоединения к набору уже имеющихся инструментов. Такие пакеты, как Matlab, должны быть включены в интегрированную среду проектирования.

Встраивание средств разработки – существующие сегодня встраиваемые среды разработки должны поддерживать проектирование систем ЦОС. Тогда разработчики получат возможность быстро выйти на широкий рынок технологий цифровой обработки сигналов.

Заключение

Цифровая обработка сигналов стремительно развивается и находит все более широкое применение. Ключевую роль в ускорении процесса разработки систем ЦОС для более быстрой их реализации на рынке играют отладочные средства. В статье был дан краткий обзор основных инструментальных средств, предоставляемых компанией ADI, рассмотрены достижения на сегодняшний день и тенденции будущего.

За дополнительной информацией обращайтесь на Internet-сервер Analog Devices по адресу: http://www.analog.com/dsp/tools/, а также на сервер АО "AUTEX Ltd." по адресу: http://www.autex.ru.

А.А. Грибачев

TMS320C641x: новые рекорды Texas Instruments в области сверхмощных цифровых сигнальных процессоров

Введение

Совсем недавно в серии статей, посвященных концепции трех платформ цифровых сигнальных процессоров (ЦСП) фирмы Texas Instruments, мы рассмотрели платформу TMS320C6000 (см. "Цифровая обработка сигналов", №1, 1999), оптимизированную для решения задач многоканальной многофункциональной высокоскоростной обработки. Первое поколение процессоров этой платформы TMS320C62xx/67xx, базируясь на архитектуре RISC-процессора с очень длинным командным словом (VLIW), названной авторами VelociTI™, с момента своего появления установило очень высокие рекорды производительности ЦСП: рабочая частота 200 МГц, 1600 MIPS, 400 млн. умножений с накоплением в секунду для процессоров с фиксированной запятой и 1 GFLOPS для процессоров с плавающей запятой. Дальнейшее совершенствование процессоров этого поколения позволило в 1,5 раза улучшить эти показатели: до 300 МГц, 2400 MIPS и 600 млн. умножений с накоплением соответственно. Но это было лишь эволюционное развитие. Намерения фирмы совершить следующий революционный шаг стали ясны в феврале 2000 г., когда была обнародована архитектура ядра процессоров второго поколения TMS320C64x, получившая название VelociTI.2™ и позволяющая, по мнению авторов, достигнуть рабочей частоты 1 ГГц и 10-кратного улучшения производительности по сравнению с процессорами TMS320C62x. Подтверждением реальности этих заявок стало объявление в феврале 2001 г. трех процессоров: TMS320C6414, TMS320C6415 и TMS320C6416, – первых процессоров с архитектурой VelociTI.2™. Знакомству с ними и посвящена эта статья.

Область применения

Давно уже канули в Лету те времена, когда после появления нового сверхмощного процессора создавалось впечатление, что его рекордные показатели долго не будут востребованы: в современной вычислительной технике вообще и в области цифровой обработки сигналов в частности нет и не может быть "слишком высокой" вычислительной мощности. И тем не менее всегда интересно знать, под какие именно применения изначально создавался новый рекордсмен. Texas Instruments считает основной сферой приложений для новых процессоров базовые станции мобильной связи третьего поколения (3G). Базовые станции 3G позволяют пользователям сотовых телефонов иметь прямой выход в Интернет, непосредственно работать с электронным почтовым ящиком (E-mail), участвовать в видеоконференциях и могут предоставлять еще множество других качественно новых услуг. Внедренные и разрабатываемые стандарты 3G-связи, такие, как WCDMA (Япония), UMTS (Европа), CDMA2000 (США), предъявляют очень высокие требования к вычислительной мощности и гибкости базовых станций. В настоящее время подавляющее большинство мировых производителей 3G-станций используют процессор TMS320C6203 – самый мощный из процессоров с архитектурой VelociTI™. Очевидно, что в процессе усовершенствования своих станций для достижения еще большей плотности обработки каналов при одновременном снижении потребляемой мощности и стоимости в расчете на один канал эти и новые производители 3G-станций будут переходить на новый процессор TMS320C6416 для того, чтобы в полной мере реализовать функции, закладываемые в стандарты 3G-связи.

Процессорное ядро

Рассмотрим объединенную блок-диаграмму процессоров TMS320C641x, представленную на *рис. 1*. Ядро процессоров C64x претерпело следующие основные улучшения по сравнению с ядром процессоров C62x:

- **Регистровые файлы**

– Количество 32-разрядных регистров общего назначения увеличено в 2 раза с 32 до 64.
– Увеличено число "условных" регистров, т.е. регистров, по значению которых выполняются условные операции.
– Регистровые файлы процессоров C62x поддерживают работу с упакованными 16-, 32- и 40-разрядными данными. В процессорах C64x они поддерживают также работу с 8- и 64-разрядными данными.

- **Организация потоков данных**

– Каждое устройство типа .D может теперь считывать и записывать двойное слово (64 разряда) за одну инструкцию.
– Устройства .D могут использовать перекрестные ссылки для передачи данных, как и остальные устройства типа .L, .M и .S.
– Один и тот же регистр может использоваться в качестве операнда несколькими функциональными устройствами по перекрестным ссылкам в одном исполняемом командном пакете.

• **Улучшенный механизм упаковки/распаковки инструкций**

В архитектуре VelociTI инструкции хранятся в упакованном виде. Специальный механизм в каждом такте выбирает пакет выборки длиной в восемь инструкций. Из них формируется исполняемый пакет длиной от 1 до 8 инструкций, которые исполняются одновременно. При этом запрещается переход исполняемого пакета через границу пакета выборки. Это требование реализуется компилятором путем вставки в программный код "пустышек" – операций NOP. В архитектуре VelociTI.2 улучшенный механизм распаковки инструкций снимает все ограничения на расположение границ исполняемого пакета. В результате отпадает необходимость вставлять "пустышки" и сокращается объем программного кода.

• **Обработка упакованных данных**

– В систему команд введены инструкции непосредственной обработки упакованных данных, а именно счетверенные команды обработки 8-разрядных операндов и сдвоенные команды обработки 16-разрядных операндов, упакованных в 32-разрядные слова.
– Для простоты работы с упакованными данными введены команды упаковки и распаковки.

• **Дополнительные свойства функциональных устройств**

– Каждое устройство .M может теперь дополнительно выполнять два умножения 16x16 разрядов или четыре умножения 8x8 разрядов.
– Устройства .D могут обращаться к любому слову или двойному слову, начинающемуся с любого байта в памяти.
– Устройства .L могут выполнять дополнительно байтные сдвиги, а устройства .M – двунаправленные сдвиги в дополнение к возможности выполнять сдвиги на устройствах .S. Двунаправленные сдвиги широко используются при реализации вокодерных алгоритмов.
– Устройства .L могут выполнять четыре 8-разрядных вычитания по модулю, что с успехом используется в алгоритмах оценки движения.
– В систему команд введены некоторые специальные инструкции, реализуемые устройствами .M, необходимые для обработки кодов, корректирующих ошибки, а также для обработки изображений и реализации алгоритмов шифрования.

• **Дополнительные улучшения**

– Устройства .D могут теперь выполнять 32-разрядные логические операции в дополнение к устройствам .S и .L.
– Устройства .D могут непосредственно считывать и записывать данные с двойной длиной слова. Этого свойства не было у процессоров C62x, но было у процессоров C67x.
– В дополнение к возможности загрузки 16-разрядных констант через устройства .S добавлена возможность загрузки 5-разрядных констант через устройства .L и .D.

Напряжение питания ядра новых процессоров снижено до 1.2 В, что позволило сохранить низкое энергопотребление при возрастании вычислительной мощности.

Рабочая частота первых процессоров C641x составляет 400 – 600 МГц. При этом встроенная схема ФАПЧ позволяет выполнять умножение входной частоты синхронизации с набором коэффициентов x1, x6 и x12.

Внутренняя память

Как это ни парадоксально, но структура внутренней памяти процессоров C641x аналогична памяти самых "облегченных" процессоров предыдущего поколения: C6211 и C6711. Память организована в двухуровневый кэш: кэш-память первого уровня L1P и L1D для программы и данных соответственно, объемом по 16 KB каждая (что в четыре раза больше соответствующей памяти в процессорах C6211/6711), и объединенная кэш-память второго уровня L2 объемом 1024 KB (в 16 раз больше). Кроме того, память L2 может быть сконфигурирована как прямо адресуемое ОЗУ или как комбинация ОЗУ и кэш-памяти.

Периферийные устройства

Контроллер прямого доступа к памяти (EDMA). В процессорах C641x использован усовершенствованный контроллер ПДП, аналогичный применяемому в процессорах C6211. Однако количество обслуживаемых независимых каналов увеличено с 16 до 64.

Интерфейс внешней памяти (EMIF). Процессоры C641x имеют два интерфейса внешней памяти: 64-разрядный EMIFA и 16-разрядный EMIFB. Кроме того, расширен список типов синхронной памяти, с которыми интерфейс работает без дополнительной логики.

Таймеры. Общее число 32-разрядных таймеров увеличено на 1 и доведено до 3.

Многоканальные буферизованные последовательные порты (McBSP). Добавлен еще один McBSP, и их общее количество теперь равно 3.

Интерфейс хост-процессора (HPI). Реализован улучшенный 32-разрядный HPI, который может работать как в 32-, так и в 16-разрядном режиме.

Интерфейс PCI. В процессорах C6416 и C6415 реализован интерфейс шины PCI (32 разряда, 33 МГц), улучшенный по сравнению с интерфейсом, реализованным в процессорах C6205.

Интерфейс UTOPIA. Этот интерфейс реализован только в процессорах C6416 и C6415. Устройство реализует Уровень 2 интерфейса UTOPIA в режиме ведомого контроллера ATM. Доступ к этому интерфейсу имеют как центральный процессор, так и контроллер EDMA.

Сопроцессоры VCP/TCP. Процессоры C6416 имеют два встроенных высокопроизводительных сопроцессора: VCP (Viterbi Decoder Coprocessor) и TCP (Turbo Decoder Coprocessor), существенно ускоряющих вычисления на телекоммуникационных задачах.

Различия между процессорами C6414, C6415 и C6416

Наиболее функционально полным процессором является C6416. В его состав входят все узлы, представленные на *рис.1*.

В процессоре C6415 реализовано все, кроме сопроцессоров VCP/TCP.

Процессор C6414 – самый "облегченный". В его состав не вошли сопроцессоры VCP/TCP, а также интерфейсы PCI и UTOPIA.

Совместимость по ножкам

Процессоры C6414, C6415 и C6416 совместимы "ножка-в-ножку" при условии использования одинакового набора периферийных устройств.

Программная совместимость

Одним из важнейших свойств платформы C6000 с момента ее появления является полная программная совместимость всех входящих в нее ЦСП на уровне объектных кодов. Это позволяет начинать разработку систем на процессорах C641x уже сейчас, не дожидаясь появления "кремния". Компилятор C поддерживает все процессоры платформы C6000. Обновленная версия симулятора позволяет с точностью до такта трассировать выполнение программы ядром процессоров C641x. Кроме того, симулятор позволяет моделировать работу кэш-памяти. В качестве аппаратных отладочных средств можно использовать серийно выпускаемый C6711 Starter Kit. Многие периферийные устройства процессоров C641x являются усовершенствованными вариантами соответствующих устройств C6711, поэтому многие режимы работы периферии могут быть отлажены в реальном времени уже сейчас. Конечно, это не дает 100% гарантии отладки системы на базе C641x, но может существенно сократить время разработки после появления процессоров C641x "в кремнии".

Информационная поддержка

Огромный объем информационных материалов, подробно описывающих процессоры C641x и аппаратные и программные средства для работы с ними, свободно доступен через сайт фирмы-производителя:

http://www.ti.com/sc/docs/dsps/products/c6000/index.htm, а также через официального представителя Texas Instruments в России:

http://www.scanti.ru.

Там же можно получить бесплатную 30-дневную версию полного программного отладочного пакета C6000 Code Composer Studio, информационные компакт-диски и необходимые консультации.

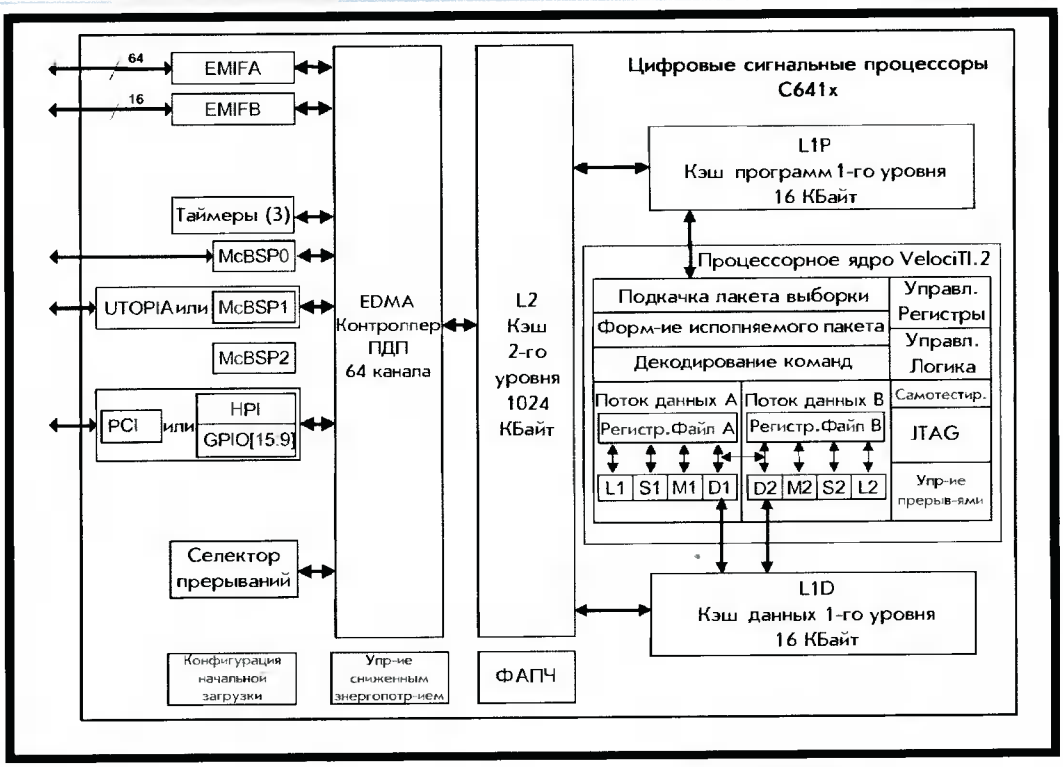


Рис.1. Блок-диаграмма процессоров TMS320C641x

В.А. Кашкаров, С.В. Мушкаев

Организация параллельных вычислений в алгоритмах БПФ на процессоре NM6403

Изучаются возможности параллельных вычислений в алгоритмах быстрого преобразования Фурье (БПФ). Исследуется схема вычислений преобразования Фурье по основанию 16. Производится сравнительный анализ со стандартным методом вычисления по основанию 2. На примере БПФ-256 описываются принципы построения параллельных вычислений на базе процессора NeuroMatrix(r) NM6403. Приводятся оценки точности и производительности БПФ процедур для процессора NM6403.

Введение

Значительная часть задач анализа временных рядов связана с преобразованием Фурье и методами его эффективного вычисления. В этих задачах преобразование Фурье играет важную роль как необходимый промежуточный шаг в определении плотностей спектра мощности, кросс-спектральных плотностей, передаточных функций, сверток, корреляционных функций, а также в задачах интерполяции.

На практике наиболее широкое распространение получили алгоритмы БПФ по основанию 2 [1], где каждый функциональный узел выполняет базовую операцию – двухвходовую "бабочку". Эти алгоритмы ориентированы прежде всего на сведение к минимуму числа операций умножения. Но с появлением векторных процессоров этот критерий становится несущественным. Напротив, число одновременно выполняемых умножений главным образом определяет производительность процессора. Поэтому возникает вопрос о распараллеливании вычислений и реализации алгоритмов БПФ с более высокими основаниями и их возможными комбинациями

Последовательность вычислений любого БПФ можно описать в виде графа, узлы которого выполняют фактически обычное дискретное преобразование, но с меньшей размерностью входных векторов (меньшим основанием). В зависимости от выбора основания меняется как общее число арифметических операций, так и количество слоев графа (*рис. 1*).

В алгоритмах БПФ по основанию 2 количество таких слоев максимально (*табл. 1*), поэтому при поэтапном поступлении результатов вычислений от слоя к слою происходит большее накопление ошибок округления, нежели в алгоритмах с более высокими основаниями. Чем выше размерность вектора входных данных, тем больше количество слоев и в следствие этого – значительнее ошибка. Накопление ошибок особенно критично в случаях, когда вычисления проводятся в целочисленной арифметике (с фиксированной точкой) или при недостаточно широкой разрядности данных. Следует также отметить, что в этом случае для предотвращения переполнения промежуточные результаты после каждого или после группы этапов умножения (слоев графа) необходимо дополнительно нормализовать, применяя операцию сдвига вправо (*рис. 1*). Однако нормализация помимо сдвига, как правило, включает в себя процедуру округления, что также вносит дополнительные вычислительные затраты.

Возможным компромиссным решением может выступать подход, основанный на увеличении основания в алгоритмах БПФ. Ниже рассматривается вариант БПФ-256 по основанию 16. Выбор такого основания, с одной стороны, дает возможность для организации параллельных вычислений, а с другой – снижает количество слоев графа до двух.

Таблица 1. Вычислительная сложность БПФ

	Прямое вычисление ДПФ (основание N)			Вычисление БПФ по основанию 2			Вычисление БПФ с комбинированными основаниями 2,16,32			
	Число компл. умнож.	Число компл. сложен.	Кол-во слоев графа	Число компл. умнож.	Число компл. слож.	Кол-во слоев графа	Число компл. умнож.	Число компл. слож.	Комбинация оснований	Кол-во слоев графа
N	N^2	N^2-N		$(N/2)\log_2 N$	$N\log_2 N$					
256	65536	65280	1	1024	2048	8	8192	7680	16 – 16	2
512	262144	261632	1	2304	4608	9	16384	16384	2 – 16 – 16	3
1024	1048576	1047552	1	5120	10240	10	49152	49152	32 – 32	2
2048	4194304	4194304	1	11264	22528	11	131072	131072	2 – 32 – 32	3

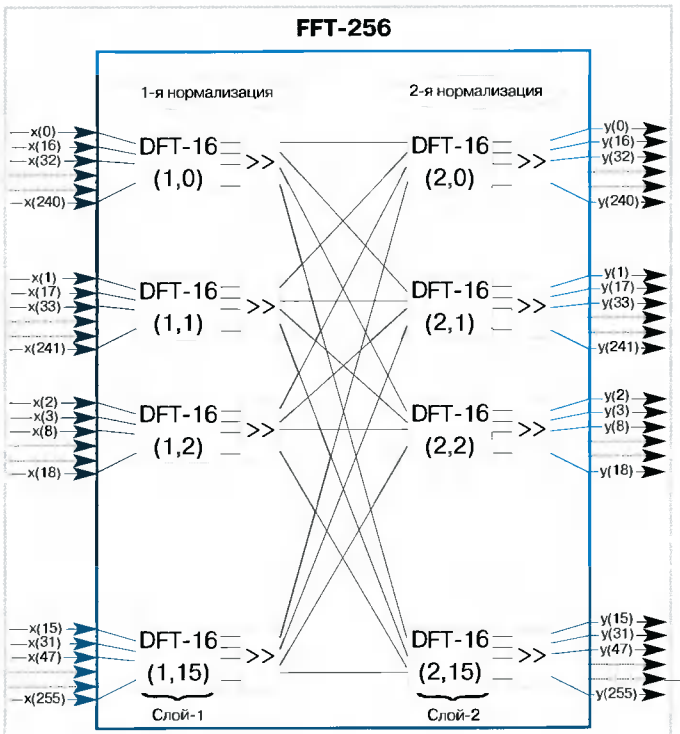


Рис.1. Обобщенный граф вычисления БПФ-256 по основанию 16

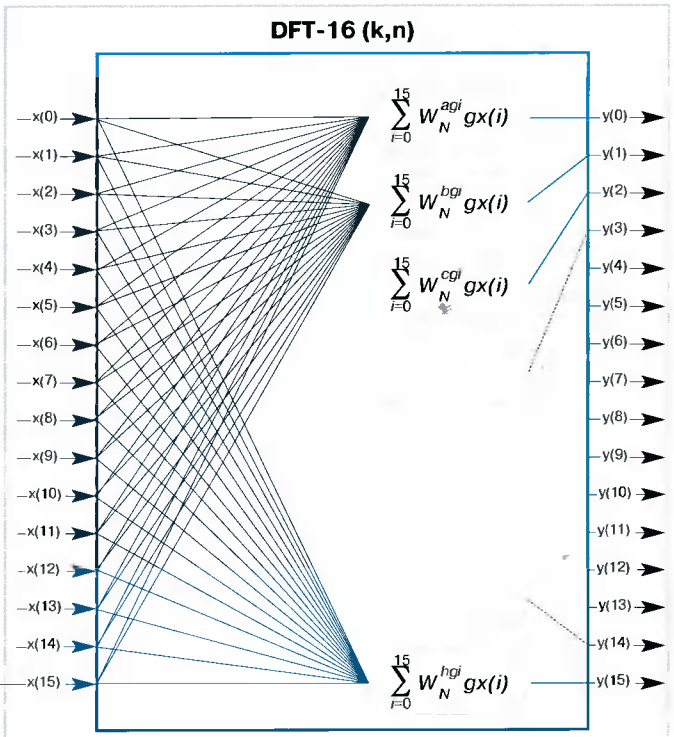


Рис.2. Развернутая схема блока 16-точечного дискретного преобразования Фурье

Дискретное 256-точечное преобразование Фурье определяется формулой

$$Y(k) = \sum_{n=0}^{255} W_{256}^{k\cdot n} \cdot X(n)\ ;\ k = 0..255,$$

где $W_{256}^{k} = \exp(-\frac{2\pi \cdot i \cdot k}{256})$.

Данная формула после тождественных преобразований принимает вид, являющийся опорным для построения БПФ-256 по основанию 16

$$Y(k) = \sum_{n=0}^{15} W_{256}^{k\cdot n} \cdot \sum_{i=0}^{15} W_{256}^{16k\cdot i} \cdot X(16\cdot i+n);\ k = 0..255.$$

Конечный граф вычисления БПФ-256 по основанию 16 строится из этой формулы. Структура такого графа показана на *рис.1*.

Граф состоит из двух слоев по 16 блоков. Каждый блок графа имеет 16 комплексных входов и выходов. Как показано на *рис.2*, каждый блок графа представляет собой 16-точечное дискретное преобразование Фурье и отличается от остальных блоков только комплексными коэффициентами W. Таким образом, распараллеливание алгоритма БПФ фактически сводится к реализации эффективного вычисления ДПФ-16, т.е. к нахождению 16 скалярных произведений различных векторов [W] с одним вектором [x], что эквивалентно умножению матрицы коэффициентов преобразования Фурье – [W] размерностью 16x16 на входной вектор [x].

Организация параллельных вычислений ДПФ-16 на процессоре NM6403

Эффективное распараллеливание вычислений ДПФ-16 достигается за счет аппаратной поддержки операции векторно-матричного умножения на процессоре NeuroMatrix(r) NM6403. Все арифметические вычисления, относящиеся непосредственно к вычислению

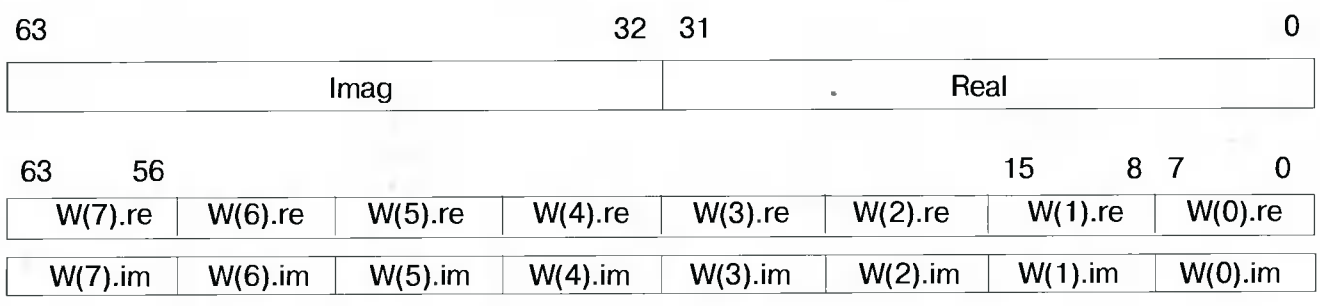


Рис. 3. Формат хранения входных данных и коэффициентов преобразования

ДПФ-16, производятся на векторном сопроцессоре. Так как векторный сопроцессор позволяет оперировать данными переменной разрядности, то для хранения входных данных и результатов вычислений удобно отводить по 32 разряда на мнимую и действительную часть, а для хранения комплексных коэффициентов W – по 8 разрядов на действительную и мнимую часть. Таким образом, в одном 64-разрядном слове может содержаться одно комплексное число. Мнимые и действительные части коэффициентов W хранятся также в упакованном виде, но в разных 64-разрядных словах. Все коэффициенты W вычисляются заранее и поэтому хранятся внутри массива в порядке, удобном для последующих вычислений (*рис.3*).

Вследствие такого представления данных векторный умножитель работает в двух конфигурациях (*рис.4* и *5*).

По приведенным двум вариантам разбиения матрицы векторного умножителя производится полный процесс скалярного умножения двух комплексных векторов. Первая схема выполняет 16 умножений с накоплением за такт и служит для нахождения сумм попарных произведений мнимых и действительных частей, вторая выполняет 4 умножения с накоплением за такт, но фактически служит только для окончательного сложения полученных частичных сумм. Полная схема умножения двух комплексных векторов длиной 16 элементов отображена на *рис.6*. Так как за один раз в матрицу весовых коэффициентов можно загрузить только 8 элементов вектора [x], загрузка всего вектора [x] происходит в два этапа.

Весь процесс вычисления скалярного произведения

$$y(k) = \sum_{i=0}^{15} W_{256}^{a \cdot i} \cdot X(i)$$

состоит из трех этапов.

1. В матрицу весовых коэффициентов загружаются 8 комплексных чисел x(0)..x(7).
   На вход умножителя X поочередно подаются сначала вектор из 8 действительных частей комплексных коэффициентов W(0)..W(7) (здесь $W(i)=W_{256}^{a \cdot i}$), а затем вектор из 8 мнимых частей. Умножение производится согласно схеме на *рис.4*.
2. Далее с выхода умножителя результат произведения в виде двух 64-разрядных слов непосредственно поступает на суммирующий Y-вход умножителя. При этом в матрицу

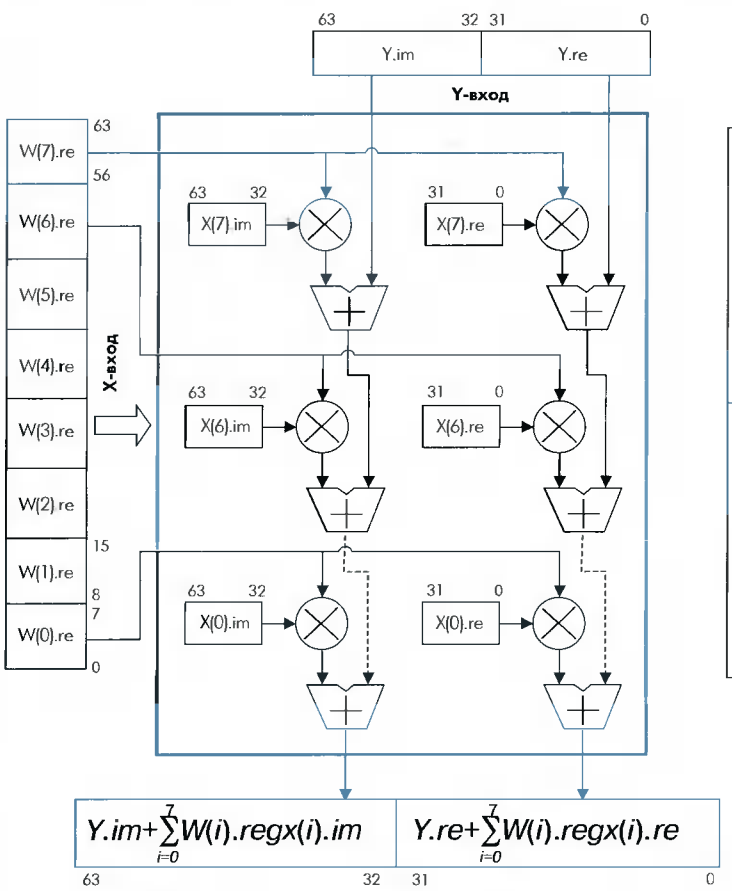


Рис.4. Эквивалентная схема умножителя векторного сопроцессора NM6403 при разбиении матрицы весовых коэффициентов – (2×32 разр)/(8×8 разр)

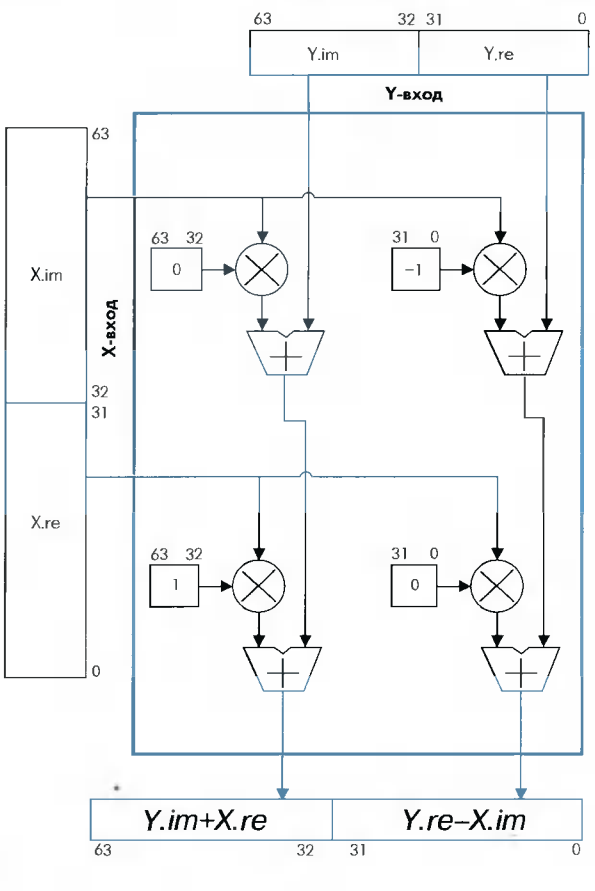


Рис.5. Эквивалентная схема умножителя векторного сопроцессора NM6403 при разбиении матрицы весовых коэффициентов – (2×32 разр)/(2×32 разр)

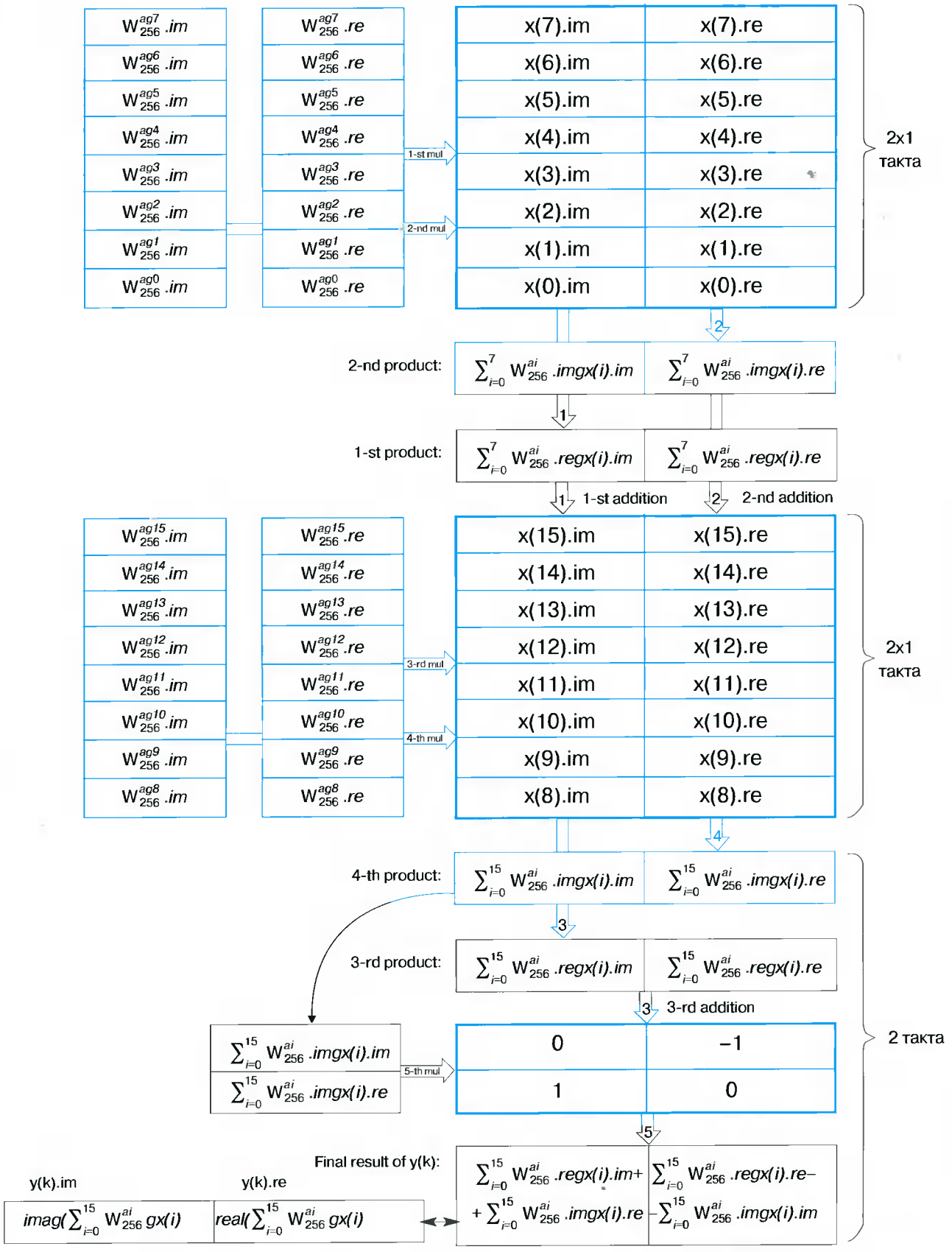


Рис.6. Последовательность вычислений скалярного произведения $y(k) = \sum_{i=0}^{15} W_{256}^{a \cdot i} \cdot X(i)$

весовых коэффициентов загружаются числа x(8)..x(15), а на вход X умножителя аналогично поступают и умножаются новые коэффициенты W(8)..W(15).

3. Для получения окончательного результата – y(k) суммы в левых и правых частях двух последних результатов ("3-rd product" и "4-th poduct") необходимо перекрестно сложить (с учетом знака "-"). Для этого, как показано на *рис.6*, в матрицу весовых коэффициентов загружаются числа 0,1 и –1, а сами суммы подаются на вход X и Y и далее, работая по схеме на *рис.5*, векторный умножитель выдает конечный результат y(k).

Для наглядности на *рис.6* проиллюстрирован процесс скалярного умножения только двух векторов. В действительности загрузка входных данных осуществляется пакетами по 32 64-разрядных слова, что позволяет максимально эффективно использовать векторный сопроцессор. В результате с учетом времени передачи данных каждый шаг умножения (*рис. 4, 5*) практически занимает один процессорный такт, это достигается за счет одновременного использования двух шин данных – подкачка входных данных x(i) по одной шине совмещается с загрузкой коэффициентов W(i) или выгрузкой результатов умножения y(i) по другой. Таким образом, реально вся процедура скалярного умножения двух комплексных 16-мерных векторов в среднем по всему БПФ-256 составляет 7 процессорных тактов.

Производительность и точность вычислений

Точность вычислений определяется количеством двоичных разрядов, отводимых для представления коэффициентов W. Имеются два способа представления значений косинусов и синусов в 8-разрядной сетке:

1. W =round (64.0*cos(x) – (условно 6 бит на единицу).
2. W =round (127.0*cos(x) – (условно 7 бит на единицу).

Первый способ имеет 65 градаций косинуса в диапазоне 0..1, в то время как второй способ – 128 и, следовательно, обладает более высокой точностью. Однако выполняя в ходе вычислений только операции целочисленного умножения и сложения, в конце необходимо провести нормализацию результатов, т.е. каждый элемент выходного массива требуется поделить на соответствующий масштабирующий коэффициент. Для первого случая он равен 64^2, а для второго – 127^2. В обоих случаях деление целесообразно заменить сдвигом вправо, но в отличие от первого способа замена деления на 127^2 сдвигом на 14 бит вправо привносит небольшую систематическую ошибку. Нормализацию можно проводить либо один раз в конце умножений, либо дважды – после каждого этапа умножения (*рис.1*) (промежуточная нормализация служит для предотвращения переполнения в ходе вычислений). Количество необходимых этапов нормализации определится в зависимости от диапазона входных данных (см. *табл. 3*).

Для оценки точности произвольный сигнал обрабатывался прямым и обратным преобразованием Фурье, после чего исходный сигнал **x** сравнивался с восстановленным сигналом **x'**. В частности, находилось математическое ожидание **M** и среднее квадратическое отклонение σ относительных ошибок δ.

$$\delta(i) = \frac{x'(i).re - x(i).re}{x(i).re}, \quad i=0..255.$$

Как видно из приведенной таблицы, 7-разрядное представление коэффициентов для БПФ по основанию 2 вносит значительную систематическую ошибку и делает такой алгоритм менее эффективным. Алгоритмы БПФ с основаниям 16 и 32 позволяют более эффективно использовать имеющуюся разрядную сетку под коэффициенты W и обладают более высокой точностью за счет уменьшения в 4 – 5 раз числа слоев графа, что также снижает дополнительные затраты на нормализацию и округление. При этом операция округления на процессоре NM6403 выполняется с помощью векторного регистра **vr** [2] совместно с основными вычислениями, не приводя к дополнительным затратам времени. Дополнительно имеется возможность проведения оптимизации за счет совмещения процедуры нормализации (арифметического сдвига вправо) с последним этапом вычисления скалярного произведения (*рис. 6*).

Таблица 2. Сравнительная характеристика точности восстановленного сигнала после прямого и обратного БПФ с разными основаниями

Преобразование Фурье	Систематическая ошибка-M		СКО – σ	
	6 разр/1.0	7 разр/1.0	6 разр/1.0	7 разр /1.0
FFT-256 по основанию 2	-1%	-12%	2.0%	0.9%
FFT-256 по основанию 16	-0.4%	-3%	1.2%	0.6%

Знак "–" означает, что восстановленный сигнал ослабляется на M процентов по сравнению с исходным.

Таблица 3. Производительность функций прямого и обратного БПФ на процессоре NM6403

Кол-во комплекс-отсчетов	Без нормализации			С одной нормализацией			С двумя нормализациями		
	Так-тов	Время, мс	Диапазон входных данных	Так-тов	Время, мс	Диапазон входных данных	Так-тов	Время, мс	Диапазон входных данных
256	3662	0.092	±512 (7 bit) ±2047(6 bit)	3994	0.1	±512 (7 bit) ±2048 (6 bit)	4300	0.11	$\pm 2^{20}$ (7 bit) $\pm(2^{21}-1)$ (6 bit)
512	8180	0.2	±256 (7 bit) ±1023(6 bit)	8766	0.22	±256 (7 bit) ±1023 (6 bit)	9350	0.24	$\pm 2^{19}$ (7 bit) $\pm(2^{20}-1)$ (6 bit)
1024	18900	0.47	±128 (7 bit) ±511 (6 bit)	20041	0.5	±128 (7 bit) ±511 (6 bit)	21200	0.53	$\pm 2^{18}$ (7 bit) $\pm(2^{19}-1)$ (6 bit)
2048	47624	1.2	±64 (7 bit) ±255 (6 bit)	49800	1.25	±64 (7 bit) ±255 (6 bit)	52000	1.3	$\pm 2^{17}$ (7 bit) $\pm(2^{18}-1)$ (6 bit)

NM6403 cycle time=25 ns (40 MHz)

Общая характеристика функций БПФ

Входные и выходные данные – целые 32-разрядные комплексные числа, формат хранения показаны на *рис.3*.

Диапазон входных данных указан в *табл. 3*.

Разрядность коэффициентов преобразования – 8 двоичных единиц (два варианта представления: 6 и 7 разрядов на единицу).

Работа с данными – арифметика с фиксированной точкой.

Выходные данные расположены в правильном порядке.

Литература

1. Отнес Р., Эноксен Л. Прикладной анализ временных рядов. М.: Мир,1982.
2. Нейропроцессор NM6403. Введение в архитектуру. М.: НТЦ "Модуль", 1998. http://www.module.ru/files/nm6403ao11b-r.pdf

В.Б. Стешенко

Проектирование аппаратуры цифровой обработки сигналов на ПЛИС с использованием языка описания аппаратуры VHDL

Возрастающая степень интеграции ПЛИС, новые концепции проектирования [1] накладывают свой отпечаток на способы описания проекта на ПЛИС. На смену графическим редакторам, применяющим библиотеку стандартных логических функций, пришли языки описания аппаратуры HDL (Hardware Description Language), которые могут быть эффективно использованы на всех этапах разработки цифровых электронных систем. Существуют несколько разновидностей этих языков: AHDL, VHDL, VerilogHDL, Abel и др. Ряд языков описания аппаратуры (AHDL, Abel) предназначены для описания систем на ПЛИС, другие появились изначально как средство моделирования цифровых систем, а затем стали инструментом описания цифровых систем. В статье дается краткое описание основных понятий и отличительных особенностей языка VHDL. Приведены примеры описания и моделирования цифрового автомата для преобразования параллельного кода в последовательный и цифрового КИХ-фильтра.

Введение в описание языка VHDL

Одним из наиболее универсальных языков описания аппаратуры является VHDL [2], первый стандарт которого был разработан с 1983 по 1987 год при спонсорстве минобороны США. На этом языке возможно как поведенческое, так структурное и потоковое описание цифровых схем.

VHDL поддерживает три различных стиля для описания аппаратных архитектур.

Первый из них – структурное описание (structural description), в котором архитектура представляется в виде иерархии связанных компонентов. Второй – потоковое описание (data-flow description), в котором архитектура представляется в виде множества параллельных регистровых операций, управляемых вентильными сигналами. Потоковое описание соответствует стилю описания, используемому в языках регистровых передач. И наконец, поведенческое описание (behavioral description), в котором преобразование описывается последовательными программными предложениями, похожими на имеющиеся в любом современном языке программирования высокого уровня. Все три стиля могут совместно использоваться в одной архитектуре.

Структурное и потоковое описание используется в основном для проектирования цифровых схем, поведенческое – только для моделирования, так как содержит конструкции, которые невозможно реализовать в виде схемы. Наиболее важными в языке VHDL являются понятия параллелизма и иерархии.

ОБЪЕКТ ПРОЕКТА (entity) представляет собой описание компоненты проекта, имеющей четко заданные входы и выходы и выполняющей четко определенную функцию. Обьект проекта может представлять всю проектируемую систему, некоторую подсистему, устройство, узел, стойку, плату, кристалл, макроячейку, логический элемент и т.п. В описании объекта проекта можно использовать компоненты, которые, в свою очередь, могут быть описаны как самостоятельные объекты проекта более низкого уровня. Таким образом, каждый компонент объекта проекта может быть связан с объектом проекта более низкого уровня. В результате такой декомпозиции пользователь строит иерархию объектов проекта (представляющих весь проект в целом), которая состоит из нескольких уровней абстракций. Такая совокупность объектов проекта называется ИЕРАРХИЕЙ ПРОЕКТА (design hierarchy).

Каждый объект проекта состоит, как минимум, из двух различных типов описаний: описания интерфейса и одного или более архитектурных тел. Интерфейс описывается в ОБЪЯВЛЕНИИ ОБЪЕКТА ПРОЕКТА (entity declaration) и определяет только входы и выходы объекта проекта. Для описания поведения объекта или его структуры служит АРХИТЕКТУРНОЕ ТЕЛО (architecture body). Чтобы задать, какие объекты проекта использованы для создания полного проекта, используется ОБЪЯВЛЕНИЕ КОНФИГУРАЦИИ (configuration declaration). В языке VHDL предусмотрен механизм пакетов для часто используемых описаний, констант, типов, сигналов. Эти описания помещаются в ОБЪЯВЛЕНИИ ПАКЕТА (package declaration). Если пользователь использует нестандартные операции или функции, их интерфейсы описываются в объявлении пакета, а тела содержатся в ТЕЛЕ ПАКЕТА (package body).

Таким образом, при описании цифровых схем на языке VHDL возможно использование пяти различных типов описаний: объявление объекта проекта, архитектурное тело, объявление конфигурации, объявление пакета и тело пакета. Каждое из описаний является самостоятельной конструкцией языка VHDL, может быть независимо проанализировано анализатором и поэтому получило название "МОДУЛЬ ПРОЕКТА" (design unit). Модули проекта, в свою очередь, можно разбить на две категории: ПЕРВИЧНЫЕ и ВТОРИЧНЫЕ. К первичным модулям относятся различного типа объявления. К вторичным – отдельно анализируемые тела первичных модулей. Один или несколько модулей проекта могут быть помещены в один файл, называемый ФАЙЛОМ

ПРОЕКТА (design file). Каждый проанализированный модуль проекта помещается в БИБЛИОТЕКУ ПРОЕКТА (design library) и становится БИБЛИОТЕЧНЫМ МОДУЛЕМ (library unit). Данная реализация позволяет создать любое число библиотек проекта. Каждая библиотека проекта в языке VHDL имеет логическое имя (идентификатор). Фактическое имя файла, содержащего эту библиотеку, может совпадать или не совпадать с логическим именем библиотеки проекта. Для ассоциирования логического имени библиотеки с соответствующим ей фактическим именем предусмотрен специальный механизм установки внешних ссылок.

Объекты данных (data object) являются хранилищами для значений определенного типа. Следует заметить, что все типы данных в VHDL конструируются из элементов, представляющих собой скалярные величины. Значения всех объектов в моделирующейся модели, взятые вместе, отражают текущее состояние моделирования. Описание на VHDL содержит объявления, которые создают объекты данных четырех классов: константы, переменные, сигналы и файлы.

Константы и переменные содержат одно значение данного типа. Значения переменных могут быть изменены назначением нового значения в предложении назначения переменной. Значение константы устанавливается до начала моделирования и не может после этого изменяться.

Сигнал имеет текущее значение подобно переменной. Он также имеет прошлую историю значений, на которые разработчик может пожелать сослаться, а также множество будущих значений, которые будут получены от формирователей сигналов. Новые значения для сигналов создаются предложениями назначения сигналов. Каждый объект в описании должен ассоциироваться с одним и только одним типом. Тип состоит из множества возможных значений и множества операций. Имеются операции двух видов. Некоторые операции являются предопределенными, это, к примеру, операторы '+', ' – ' для значений типа integer. Другие операции явно кодируются в VHDL: например, может быть написана функция подпрограмма Max, которая возвращает наибольший из двух целых аргументов. Тип объекта представляет информацию, которая окончательно определяется в момент записи модели. Эта информация способствует обнаружению несоответствий в тексте без обращения к моделированию. Например, легко обнаружить и отметить попытку назначения булевого значения (True или False) целой переменной. Новое значение, которое должно быть создано предложением назначения, определяется выражением в правой части. Выражения используются также и в других контекстах: например, как условие в предложении if. В состав выражения могут входить константы, переменные, сигналы, операторы и указатели функций. Когда имя объекта используется в выражении, то при расчете значения выражения учитывается его текущее значение.

Примеры проектирования

В качестве примера описания на языке VHDL цифровых схем рассмотрим проектирование цифрового автомата для преобразования параллельного кода в последовательный. Преобразователь кода представляет собой устройство, на вход которого подается n-битное число в параллельном коде 'd', сигнал загрузки 'load' и синхроимпульсы 'clk'. По сигналу загрузки происходит запись входного слова во внутренний регистр и последовательная выдача в течение n тактов этого входного слова в последовательном коде на выходе 'o' синхроимпульсами 'oclk'. После окончания преобразования на выходе 'e' появляется высокий уровень сигнала в течение одного такта. Такого рода преобразователи кода часто используются для управления синтезаторами частот.

Описание этого устройства на языке VHDL приведено ниже:

```
library ieee;
use ieee.std_logic_1164.all;

entity Serial is
    port (
          clk   : in STD_LOGIC;
          load  : in STD_LOGIC;
          reset: in STD_LOGIC;
          d     : in STD_LOGIC_vector (3 downto 0);
          oclk  : out STD_LOGIC;
          o     : out STD_LOGIC;
          e     : out STD_LOGIC
         );
end;

architecture behavioral of Serial is
    type t1 is range 0 to 4;
        signal s : STD_LOGIC_vector (2 downto 0);
    signal i : t1;

begin

process (clk)
        begin
        if reset = '1' then
                i <= 0;
        else
                if (clk'event and clk='1') then
                        if (i = 0 and load = '1') then
                            s(2 downto 0) <= d(3 downto 1);
                            o <= d(0);
                            i <= 4;
                        end if;
                        if (i > 1) then
                            o <= s(0);
                            s(1 downto 0) <= s(2 downto 1);
                            i <= i - 1;
                        end if;
                        if (i = 1) then
                            e <= '1';
                            i <= 0;
                        else
                            e <= '0';
                        end if;
                end if;
        end if;
        if i>0 then
                oclk <= not clk;
        else
```

```
                oclk <= '0';
        end if;

end process;

end behavioral;
```

По переднему фронту синхроимпульса 'clk' при высоком уровне на входе загрузки происходит загрузка трех старших бит входного слова d[3..1] во временный регистр s[2..0]. Младший бит входного слова d[0] подается на выход 'o'. На выходе 'oclk' появляются синхроимпульсы. На сигнале 'i' собран внутренний счетчик, выдающий сигнал окончания преобразования 'e'. При поступлении последующих синхроимпульсов происходит выдача на выход 'o' остальных бит входного слова, хранящихся в регистре s[2..0].

Моделирование этого устройства было проведено в системе проектирования OrCAD 9.0

Для тестирования схемы использовался тест:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity test_serial is end test_serial;
architecture testbench of test_serial is
component serial
  port (
     clk : in std_logic;
     load : in std_logic;
     reset : in std_logic;
     d : in std_logic_vector(3 downto 0);
     oclk : out std_logic;
     o : out std_logic;
     e : out std_logic
      );
end component;

signal  clk : std_logic;
signal   load : std_logic := '0';
signal   reset : std_logic;
signal   d : std_logic_vector(3 downto 0);
signal   oclk : std_logic;
signal   o : std_logic;
signal   e : std_logic;
begin
        process begin
                for i in 0 to 50 loop
                        clk <= '0'; wait for 5 ns;
                        clk <= '1'; wait for 5 ns;
                end loop;
        end process;
        process begin
                reset <= '1'; wait for 10 ns;
                reset <= '0' ;
                load <= '1';
                d <= "1010"; wait for 10 ns;
                load <= '0';
                d <= "0000"; wait for 500 ns;
        end process;
dut :  serial  port map (
     clk => clk,
     load => load,
     reset => reset,
     d => d,
     oclk => oclk,
     o => o,
     e => e
      );
end testbench;
```

Результаты моделирования представлены на *рис.1*.

В качестве примера описания устройства ЦОС на языке VHDL рассмотрим проектирование цифрового КИХ-фильтра [3 – 5].

Работа цифрового КИХ-фильтра описывается разностным уравнением

$$y_n = A_0 x_n + A_1 x_{n-1} + A_2 x_{n-2} + \ldots,$$

где y_n – реакция системы в момент времени n, x_n – входное воздействие,

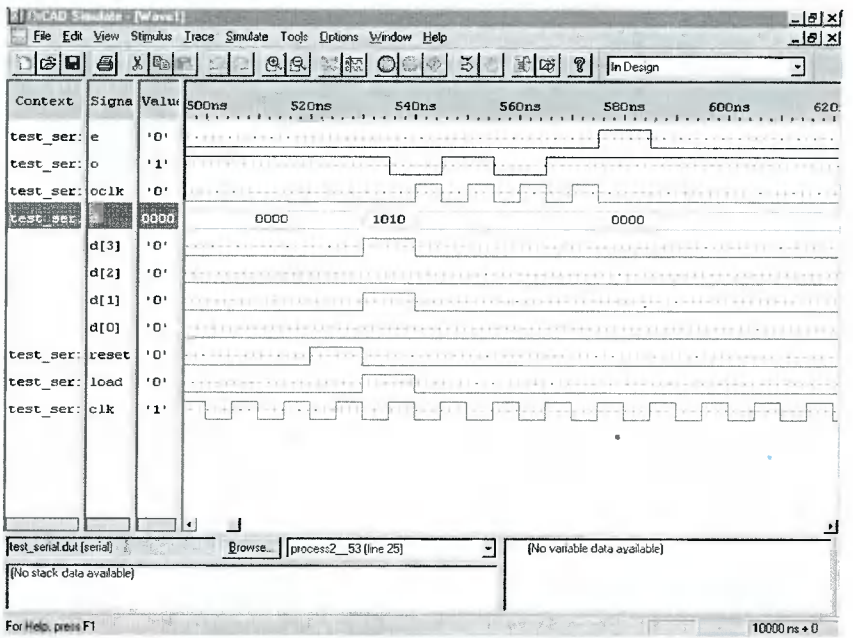


Рис. 1

A_i – Весовой коэффициент i-ой входной переменной.

На VHDL описание фильтра имеет вид

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity f is
port (
        din: in std_logic_vector(7 downto 0);
        sout: out std_logic_vector(15 downto 0);
        r: in std_logic;
        c: in std_logic
        );
end f;

architecture behavior of f is

constant h00 : std_logic_vector(7 downto 0) := "00000000";
constant h01 : std_logic_vector(7 downto 0) := "00000001";
constant h02 : std_logic_vector(7 downto 0) := "00000100";
constant h03 : std_logic_vector(7 downto 0) := "00001111";
constant h04 : std_logic_vector(7 downto 0) := "00100100";
constant h05 : std_logic_vector(7 downto 0) := "01000010";
constant h06 : std_logic_vector(7 downto 0) := "01100100";
constant h07 : std_logic_vector(7 downto 0) := "01111100";
constant h08 : std_logic_vector(7 downto 0) := "01111111";
constant h09 : std_logic_vector(7 downto 0) := "01101010";
constant h10 : std_logic_vector(7 downto 0) := "01000010";
constant h11 : std_logic_vector(7 downto 0) := "00010011";
constant h12 : std_logic_vector(7 downto 0) := "11101100";
constant h13 : std_logic_vector(7 downto 0) := "11010110";
constant h14 : std_logic_vector(7 downto 0) := "11010101";
constant h15 : std_logic_vector(7 downto 0) := "11100011";
constant h16 : std_logic_vector(7 downto 0) := "11110111";
constant h17 : std_logic_vector(7 downto 0) := "00001010";
constant h18 : std_logic_vector(7 downto 0) := "00010100";
constant h19 : std_logic_vector(7 downto 0) := "00010011";
constant h20 : std_logic_vector(7 downto 0) := "00001100";
constant h21 : std_logic_vector(7 downto 0) := "00000010";
constant h22 : std_logic_vector(7 downto 0) := "11111000";
constant h23 : std_logic_vector(7 downto 0) := "11110101";
signal x00, x01, x02, x03, x04, x05, x06, x07,
        x08, x09, x10, x11, x12, x13, x14, x15,
        x16, x17, x18, x19, x20, x21,
        x22, x23 : std_logic_vector(7 downto 0);
signal m00, m01, m02, m03, m04, m05, m06, m07,
        m08, m09, m10, m11, m12, m13, m14, m15,
        m16, m17, m18, m19, m20, m21,
        m22, m23 : std_logic_vector(15 downto 0);

begin
m00 <= (signed(x00)*signed(h00));
m01 <= (signed(x01)*signed(h01));
m02 <= (signed(x02)*signed(h02));
m03 <= (signed(x03)*signed(h03));
m04 <= (signed(x04)*signed(h04));
m05 <= (signed(x05)*signed(h05));
m06 <= (signed(x06)*signed(h06));
m07 <= (signed(x07)*signed(h07));
m08 <= (signed(x08)*signed(h08));
m09 <= (signed(x09)*signed(h09));
m10 <= (signed(x10)*signed(h10));
m11 <= (signed(x11)*signed(h11));
m12 <= (signed(x12)*signed(h12));
m13 <= (signed(x13)*signed(h13));
m14 <= (signed(x14)*signed(h14));
m15 <= (signed(x15)*signed(h15));
m16 <= (signed(x16)*signed(h16));
m17 <= (signed(x17)*signed(h17));
m18 <= (signed(x18)*signed(h18));
m19 <= (signed(x19)*signed(h19));
m20 <= (signed(x20)*signed(h20));
m21 <= (signed(x21)*signed(h21));
m22 <= (signed(x22)*signed(h22));
m23 <= (signed(x23)*signed(h23));

sout <= (signed(m00)+signed(m01)+signed(m02)+signed(m03)
        +signed(m04)+signed(m05)+signed(m06)+signed(m07)
        +signed(m08)+signed(m09)+signed(m10)+signed(m11)
        +signed(m12)+signed(m13)+signed(m14)+signed(m15)
        +signed(m16)+signed(m17)+signed(m18)+signed(m19)
        +signed(m20)+signed(m21)+signed(m22)+signed(m23));

process(c,r)
        begin
        if r='1' then
                x00 <= (others => '0');
                x01 <= (others => '0');
                x02 <= (others => '0');
                x03 <= (others => '0');
                x04 <= (others => '0');
                x05 <= (others => '0');
                x06 <= (others => '0');
                x07 <= (others => '0');
                x08 <= (others => '0');
                x09 <= (others => '0');
                x10 <= (others => '0');
                x11 <= (others => '0');
                x12 <= (others => '0');
                x13 <= (others => '0');
                x14 <= (others => '0');
                x15 <= (others => '0');
                x16 <= (others => '0');
                x17 <= (others => '0');
                x18 <= (others => '0');
                x19 <= (others => '0');
                x20 <= (others => '0');
                x21 <= (others => '0');
                x22 <= (others => '0');
                x23 <= (others => '0');
        elsif (c'event and c='1') then
                x00(7 downto 0) <= din(7 downto 0);
                x01(7 downto 0) <= x00(7 downto 0);
                x02(7 downto 0) <= x01(7 downto 0);
                x03(7 downto 0) <= x02(7 downto 0);
                x04(7 downto 0) <= x03(7 downto 0);
                x05(7 downto 0) <= x04(7 downto 0);
                x06(7 downto 0) <= x05(7 downto 0);
                x07(7 downto 0) <= x06(7 downto 0);
                x08(7 downto 0) <= x07(7 downto 0);
                x09(7 downto 0) <= x08(7 downto 0);
                x10(7 downto 0) <= x09(7 downto 0);
                x11(7 downto 0) <= x10(7 downto 0);
                x12(7 downto 0) <= x11(7 downto 0);
                x13(7 downto 0) <= x12(7 downto 0);
                x14(7 downto 0) <= x13(7 downto 0);
                x15(7 downto 0) <= x14(7 downto 0);
                x16(7 downto 0) <= x15(7 downto 0);
                x17(7 downto 0) <= x16(7 downto 0);
                x18(7 downto 0) <= x17(7 downto 0);
                x19(7 downto 0) <= x18(7 downto 0);
                x20(7 downto 0) <= x19(7 downto 0);
                x21(7 downto 0) <= x20(7 downto 0);
                x22(7 downto 0) <= x21(7 downto 0);
                x23(7 downto 0) <= x22(7 downto 0);
        end if;
        end process;
end behavior;
```

Входные данные считываются с входа din[7..0] в дополнительном коде по переднему фронту синхросигнала 'c'

На сигналах x0 ÷ x23 построен сдвиговый регистр, обеспечивающий задержку данных на 24 такта. Сигналы с регистров умножаются

на весовые коэффициенты h0 ÷ h23 и суммируются.

Для тестирования схемы использован тест:

```
- Test bench shell

library ieee;
use ieee.std_logic_1164.all;

entity test_f is end test_f;

architecture testbench of test_f is

component f
  port (
     din : in std_logic_vector(7 downto 0);
     sout : out std_logic_vector(15 downto 0);
     r : in std_logic;
     c : in std_logic
      );
end component;

signal   din : std_logic_vector(7 downto 0);
signal   sout : std_logic_vector(15 downto 0);
signal   r : std_logic;
signal   c : std_logic;

begin

        process begin
                for i in 0 to 50 loop
                        c <= '0'; wait for 5 ns;
                        c <= '1'; wait for 5 ns;
                end loop;
        end process;

        process begin
                r <= '1'; wait for 10 ns;
                r <= '0' ;
                din <= "00000001"; wait for 10 ns;
                din <= "00000000"; wait for 500 ns;
        end process;

dut :  f  port map (
     din => din,
     sout => sout,
     r => r,
     c => c
      );

end testbench;
```

Тест моделирует подачу на цифровой фильтр аналога δ-функции. На выходе фильтра – его импульсная характеристика.

Результаты моделирования представлены на *рис.2*.

Литература

1. Стешенко В.Б. Программируемые логические интегральные схемы: обзор архитектур и особенности применения в аппаратуре ЦОС. // Цифровая обработка сигналов, 2000, № 2, 2000, с. 39 – 48.
2. Дж.Р.Армстронг. Моделирование цифровых систем на языке VHDL. / Пер. с англ. М.: Мир, 1992. – 175 с.
3. Губанов Д.А., Стешенко В.Б. Методология реализации алгоритмов цифровой фильтрации на основе программируемых логических интегральных схем. // Сборник докладов 1-й Международной конференции "Цифровая обработка сигналов и ее применения" 30.06 – 3.07.1998, Москва, МЦНТИ, том 4, с. 9 – 19.
4. Стешенко В.Б. Особенности проектирования аппаратуры цифровой обработки сигналов на ПЛИС с использованием языков описания аппаратуры. // Сборник докладов 2-й Международной конференции "Цифровая обработка сигналов и ее применения" 21.09 – 24.09.1999, Москва, МЦНТИ, том 2, с. 307 – 314.
5. Губанов Д.А., Стешенко В.Б., Храпов В.Ю., Шипулин С.Н. Перспективы реализации алгоритмов цифровой фильтрации на основе ПЛИС фирмы ALTERA. // Chip News, 1997, № 9 – 10, с. 26 – 33.

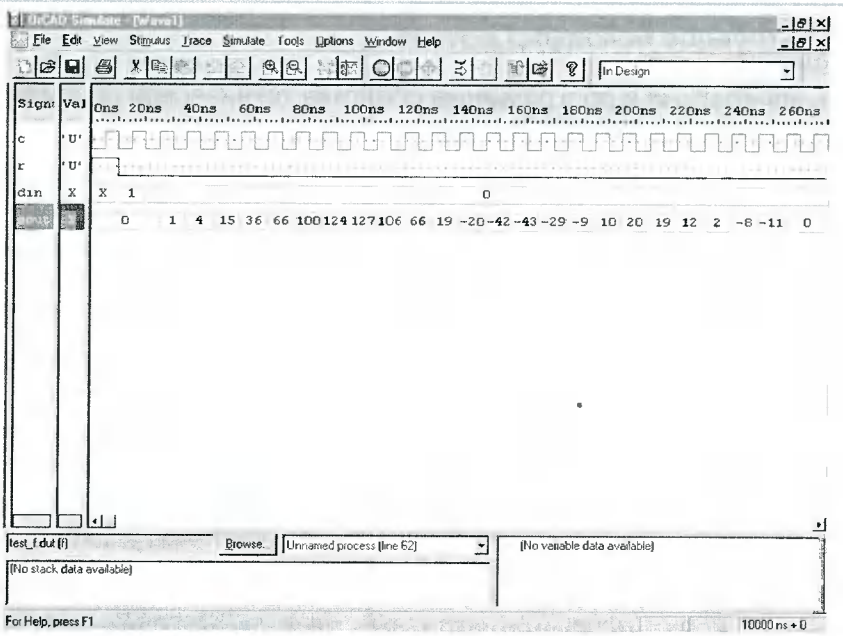


Рис.2