МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

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

И. А. Кудрявцев

Радиотехнические комплексы контроля полёта и управления микро/наноспутников

Электронный конспект лекций

САМАРА

2010

Автор: КУДРЯВЦЕВ Илья Александрович

Конспект лекций по дисциплине «Радиотехнические комплексы контроля полёта и управления микро/наноспутников» рассчитан на ознакомление с основами разработки электронных систем технологического назначения, применяемых в микро- и наноспутниках.

Электронный конспект лекций предназначен для магистрантов, обучающихся по магистерской программе «Космические информационные системы и наноспутники. Навигация и дистанционное зондирование» по направлению 010900.68 «Прикладные математика и физика».

Электронный конспект лекций разработан на межвузовской кафедре космических исследований.

© Самарский государственный
аэрокосмический университет, 2010

Радиотехнические комплексы контроля полета и управления микро и наноспутниками

Курс рассчитан на ознакомление с основами разработки электронных систем технологического назначения, применяемых в микро- и наноспутниках. Прежде всего необходимо отметить, что все спутники несут определенную функциональную нагрузку, например, спутники связи, навигационные спутники, спутники для проведения научных экспериментов и т.д. Диапазон применений практически не ограничен и расширяется с каждым годом вместе с совершенствованием спутниковых технологий. В этой ситуации микроспутники имеют практическое преимущество, выражающееся в относительно низкой стоимости производства, запуска (как попутный груз) и эксплуатации. Вместе с тем имеются и недостатки, в частности, малые габариты не позволяют обеспечить спутник значительной энергетикой, что ограничивает спектр применений, а запуск в виде попутной нагрузки сужает выбор орбит. Учитывая указанные особенности, разработчики микроспутников в настоящее время чаще всего ориентируют их на образовательные цели, проведение научных экспериментов и отработку технологических аспектов создания космической аппаратуры.

Для решения практических задач от спутника обычно требуется обеспечение следующих функций (и соответствующих систем):

1. Управляемая ориентация в пространстве;
2. Наличие телекоммуникационных средств для управления и передачи информации на Землю;
3. Наличие энергосистемы.

Эти функции возлагаются на технологические системы спутника и их обеспечение требует от разработчика решения целого ряда технических вопросов, в том числе и специфических для надежной эксплуатации оборудования в условиях вакуума и невесомости.

Ориентация в пространстве

Для решения многих задач необходима управляемая извне или автоматическая ориентация спутника по отношению к Земле и Солнцу. Не затрагивая вопросов создания спутников с автономными двигательными установками, а также системы пассивной ориентации, можно выделить два типа систем ориентации: на основе маховиков (reaction wheels) и магнитные системы с использованием магнитного поля Земли. Последний вариант обычно реализуется для относительно низкоорбитальных спутников, находящихся в зоне сколько-нибудь заметного магнитного поля Земли. С точки зрения разработчика радиоэлектронных управляющих комплексов реализация системы ориентации обоих типов может быть разбита на две отдельных задачи:

- Определение текущей ориентации объекта;
- Изменение текущей ориентации объекта.

Первая задача решается с помощью разнообразных датчиков, из которых наибольшее распространение нашли звездные датчики, датчики солнца,

датчики инфракрасного излучения Земли и датчики магнитного поля Земли. Как правило, при определении ориентации используют несколько датчиков, дублирующих и дополняющих друг друга. Основная идея всех методов определения ориентации – это регистрация положения спутника относительно опорных источников (Солнца, звезд, Земли). Указанная концепция открывает простор для поиска и других опорных источников, например, навигационных или связных спутников Земли, положение которых хорошо известно. В любом случае задача определения текущей ориентации сводится к обработке сигналов нескольких датчиков и решения специфической навигационной задачи на основе имеющейся априорной информации о положении опорных объектов.

Вторая задача решается путем изменения момента вращения маховиков или собственного магнитного поля аппарата путем активизации магнитных катушек. Как правило, магнитный вариант системы ориентации используется как вспомогательный или для поддержания ориентации, так как он не обеспечивает значительных вращательных моментов для спутника.

Телекоммуникационная система

Исключая из рассмотрения спутники, находящиеся на геостационарной орбите, можно сказать, что с фиксированной точки Земной поверхности каждый спутник виден достаточно короткое время (10-15 минут). За это время желательно передать на Землю максимум информации. Возможно увеличение пропускной способности такой телекоммуникационной системы за счет организации сети приемных станций, однако, такой вариант решения выходит за пределы полномочия разработчика, хотя и может быть заложен в расчет при предварительном проектировании спутника. Альтернативным методом является организация телекоммуникационной системы на базе спутниковой телефонной связи (системы Inmarsat, Globalstar и др.), обеспечивающей связь на большей части орбиты спутника. При разработке телекоммуникационной системы следует принимать во внимание следующие аспекты эксплуатационного характера:

- Ограниченные возможности энергосистемы спутника;
- Возможную потерю ориентации спутника;
- Ограничения на размер антенной системы;
- Ограничения на используемый диапазон частот.

Как правило, для микроспутника, чей энергоресурс определяется возможностями солнечных элементов, мощность передатчика будет ограничиваться единицами ватт, для наноспутника еще меньше. В связи с этим ограничением телекоммуникационные системы целесообразно строить по принципу «минимум на борту, максимум на Земле».

Потеря ориентации спутника может привести к невыгодному расположению антенны передатчика относительно приемника и ухудшить условия радиосвязи. Данное обстоятельство побуждает разработчика создавать антенные системы, способные работать при любом положении спутника.

Если спутник выполняет образовательные или отладочные функции, решение организационных проблем с государственными органами, контролирующими использование сетки частот (Россвязьнадзор, ГКРЧ) является нежелательным или даже невозможным. Таким образом, выбор частоты передачи/приема будет ограничен частотами радиолюбительского диапазона (144, 430, 2400МГц и выше). Вместе с конструктивными соображениями по расположению и развертыванию антенной системы, а также затуханием радиоволн, данное обстоятельство сужает варианты построения телеметрической системы. Также здесь играет роль желаемая ширина полосы частот, занимаемая сигналом и выбор элементной базы для построения приемников/передатчиков.

Энергосистема

Для построения энергосистемы микро- и наноспутников чаще всего применяются солнечные батареи и накопительные системы на базе аккумуляторов. Здесь выбор разработчика ограничен стоимостными параметрами и габаритами спутника. Как правило, реализуется следующая стратегия:

- Максимальная площадь фотоэлементов, обеспечивающая подзаряд аккумуляторов на солнечной части орбиты;
- Работа от аккумуляторов не теневой стороне орбиты.

Реализация этой концепции сталкивается с целым рядом проблем, начиная от ориентации спутника и заканчивая реализацией режимов жесткой экономии энергии. Для микроспутника обычно затруднительно реализовывать крупноразмерные системы фотоэлементов с механизмом развертывания, с другой стороны, часто не требуется постоянное функционирование всех систем, поэтому распространенным решением является жесткая установка фотоэлементов на всех поверхностях спутника и реализация дежурного режима, максимально сохраняющего заряд аккумулятора с активацией спутника только во время сеансов связи с Землей. От разработчика энергосистемы здесь требуется обеспечение максимальной эффективности преобразователей напряжения и адаптивная система отбора мощности от солнечных элементов.

Система управления

Полноценная эксплуатация спутника возможна при условии грамотной координации работы всех его подсистем, в связи с чем роль бортового компьютера становится решающей. К нему предъявляются разнообразные функциональные требования и требования надежности и устойчивой работы в сложных условиях эксплуатации. Разработка (или выбор) бортового вычислителя возможна двумя путями:

- Разработка уникальной программы управления под конкретный набор функциональных узлов спутника;
- Установка универсальной операционной системы, поддерживающей все функциональные узлы спутника.

Оба решения имеют свои достоинства и недостатки: например, операционная система для своего функционирования может требовать дополнительных ресурсов (и, значит увеличенного энергопотребления), однако существенно сокращаются сроки разработки и упрощается работа программиста. Без операционной системы усложняется реализация многих базовых функций (например, реализация TCP/IP протокола), однако, окончательное решение может оказаться лучшим по соотношению функциональность/потребляемая мощность. Необходимым элементом разработки бортового вычислителя является необходимость обеспечения восстановления после сбоев. Например, очень полезной является функция удаленной перезагрузки операционной системы и изменения программного обеспечения. Следует отметить, что реализация такой функциональности гораздо проще при использовании операционной системы.

АНАЛИЗ ВАРИАНТОВ ПОСТРОЕНИЯ ЗЕМНЫХ ДАТЧИКОВ ОРИЕНТАЦИИ

Для решения задачи астроориентации существуют типовые схемы: солнечно-звездная, солнечно-земная, и звездно-земная. [1] Звездные астроприборы построены либо на принципе получения изображения звездного неба и последующем сравнении его с имеющейся в памяти звездной картой, либо на слежении за одной, яркой звездой. В связи с тем, что освещенность от звезд очень мала, необходимо применять оптические системы с большим увеличением. Поэтому, солнечно-земная типовая схема астроориентации может иметь очевидные преимущества в смысле стоимости датчика и простоты его исполнения. В литературе [1] рассматривается вариант ориентации с помощью земных астроприборов с предположением, что Земля выглядит как квазиточечный источник, что непригодно для низкоорбитальных спутников.

Будем считать, что КА находится на относительно низкой орбите порядка 250-750км, откуда следует, что угол обзора Земли близок к 180°. Земля в данном случае не может быть принята за точечный источник, и ориентировать КА нужно в центр наблюдаемого светящегося круга. В свою очередь, Солнце видно системе не постоянно, время его восхода, заката и долгота «дня» на микроспутнике зависят от параметров орбиты.

Работа солнечно-земной системы ориентации начинается с вращения КА (или его подвижной части с астроприбором) вокруг некоторой оси, поиска Земли соответствующим датчиком, и последующей стабилизации вертикальной оси относительно направления на Землю. Затем КА закручивается вокруг этой оси и осуществляет поиск Солнца и стабилизацию второй оси относительно направления на него.

Земля, как тип астроориентира, может выступать двояко в зависимости от диапазона длин волн используемого излучения. Земные астроприборы подразделяются на приборы видимого диапазона и инфракрасного диапазона. В видимом диапазоне Земля, в зависимости от угла Солнце-КА-Земля, представляется в виде светлого круглого диска, либо серпа, либо вообще не видна. Это создает определенные ограничения в получении информации с прибора, работающего по Земле в видимом диапазоне. В ИК-диапазоне Земля представляет собой круглый излучающий диск, вне зависимости от каких-либо условий, поэтому земные астроприборы ИК-диапазона свободны от ограничений в возможности получения от них информации.

Приборы ориентации по Земле по функциональному назначению, возможностям и особенностям применения можно разделить на три типа:

- датчики наличия Земли (ДНЗ);
- построители местной вертикали (ПМВ);

– измерители угла отклонения.

Датчик наличия Земли определяет присутствие Земли в одном из двух угловых полей, неподвижно расположенных относительно КА и развернутых на 180º одно относительно другого.

ПМВ определяют отклонение оси прибора от направления на центр Земли в двух взаимно перпендикулярных плоскостях в пределах небольшой зоны углов отклонения и формируют выходной управляющий сигнал. Измерение производится путем сканирования углового поля прибор, а по круговой траектории и формированием выходных сигналов с помощью фазовых детекторов. При отсутствии рассогласования траектория сканирования совпадает с горизонтом Земли. Зона линейности выходных сигналов небольшая, как правило, 1-2° , погрешность возрастает при увеличении рассогласования. Серьезным недостатком приборов этого типа является большая методическая погрешность, вызванная неравномерностью излучения Земли.

Измерители угла отклонения работают в широком диапазоне (до десятков градусов), при этом точность прибора меняется мало. В них используется так называемый секущий принцип сканирования, при котором траектория сканирования лежит либо в плоскости, либо на конической поверхности, пересекающей окружность горизонта Земли.

В земных астроприборах используется длинноволновая часть ИК-диапазона (λ>7 мкм). Излучение Земли в этом диапазоне представляет собой почти полностью собственное излучение атмосферы и поверхности, влияние отраженного солнечного излучения в этом диапазоне пренебрежимо мало, т.к. оно сосредоточено на длинах волн менее 3 мкм. Энергетическая яркость наблюдаемых из космоса участков Земли представляет собой случайную функцию пространственных координат и времени. Она может меняться в несколько раз (до 3-4 раз) в зависимости от положения точки на поверхности Земли, времени и длины волны.

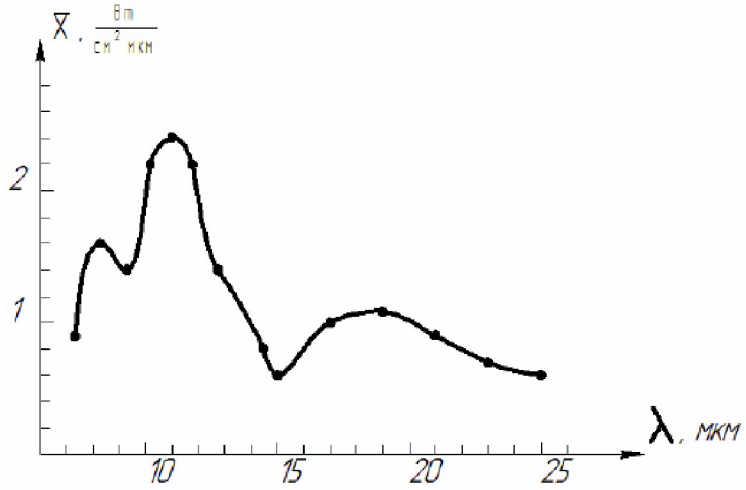


Рисунок 1 – Зависимость математического ожидания спектральной энергетической светимости Земли от длины волны

На основе экспериментальных исследований, проведенных с помощью установленной на спутниках аппаратуры (в частности на КА «Космос-45» и «Космос-65»), были получены вероятностные характеристики этой случайной функции (рисунок 1). Из рисунка можно видеть, что наименьшие флуктуации излучения Земли наблюдаются в диапазоне длин волн от 16 до 18 мкм. Это связано с тем, что в верхней атмосфере, на высоте около 40 км слой углекислого газа мало подвержен изменениям и поглощает излучение нижележащих слоев. Для наиболее точных приборов выбирается именно эта полоса.
Средняя спектральная светимость Земли в диапазоне длин волн 7-26 мкм на широтах от -65° до +65° составляет 2,06 мВт/(см^2 мкм). Максимум интенсивности ИК- излучения от Земли приходится примерно на 10 мкм.

Выбор датчика

Весь инфракрасный диапазон спектра разделен на 5 поддиапазонов:

- ближний ИК (БИК) – 0,76–1,1 мкм;
- коротковолновый ИК (КВИК) – 1,1–2,5 мкм;
- средневолновый ИК (СВИК) – 3,0–5,0 мкм;
- длинноволновый ИК (ДВИК) – 7,0–14,0 мкм;
- дальний ИК (ДИК) – 15,0–1000 мкм.

В настоящее время разработано и активно используется большое количество датчиков и тепловизоров средневолнового и длинноволнового ИК- поддиапазонов. Начинается активная разработка приборов ближнего и коротковолнового ИК-поддиапазонов, однако широкого распространения они пока не получили. В основном используются датчики и тепловизоры длинноволнового ИК-поддиапазона. Это связано с техническими (большинство тепловизоров СВИК охлаждаемые) и физическими (в ДВИК находится максимум излучения тел при комнатной температуре) причинами.

Важно отметить, что из всего многообразия полупроводниковых материалов в подобных системах могут использоваться лишь те, фотоэффект в которых наблюдается лишь в узком диапазоне длин волн - в противном случае не исключено вмешательство фотонов видимого света и "ближнего" ИК-излучения. Такие материалы называются узкозонными полупроводниками, а наибольшее применение в тепловидении нашли теллурид ртути и кадмия HgCdTe и антимонид индия InSb. Последний оказался более технологически пригодным, однако диапазон его чувствительности не позволяет эффективно улавливать инфракрасные лучи длинноволнового диапазона. Требования к химической чистоте и редкость самих компонентов лишь добавляют сложностей производителям и обуславливают высокую стоимость решений на базе фотоэффекта.

Пироэлектрические ИК-датчики, применяемые часто в так называемых «датчиках движения» систем безопасности, изменяют выходное сопротивление при изменении освещенности. Таким образом, для того, чтоб применить их в данном устройстве, пришлось бы устанавливать механическую или оптическую шторку, для модуляции излучения, а также усложнять схему устройства.

Сравнительно недавно были разработаны неохлаждаемые инфракрасные детекторы. Их можно разделить на следующие классы:

- микроболометры;
- ферроэлектрики;
- другие типы.

В свою очередь, микроболометры подразделяются на два подкласса: VOx -микроболометр на оксиде ванадия и alpha-Si - микроболометр на аморфном кремнии. Ферроэлектрики также делятся на два подкласса: Thick Film BST - толстопленочная технология и Thin Film PLZT - тонкопленочная технология. К другим неохлаждаемым датчикам можно отнести приемники на солях свинца и Poly-SiGe.

Таким образом, инфракрасные чувствительные элементы достаточно многообразны и основаны на различных технологиях, но наиболее популярными на рынке ИК-чувствительных неохлаждаемых детекторов можно считать ферроэлектрики и микроболометры. Наибольший интерес в рамках данной работы представляют особенности конструкции и работы микроболометров и разработки в этой области. Функционируя при температуре окружающей среды либо при незначительной температурной коррекции, осуществляемой аппаратным путем, неохлаждаемые микроболометрические матрицы на сегодняшней стадии развития технологии уступают своим охлаждаемым собратьям по показателям чувствительности. Принцип измерения, используемый в них, основан на регистрации изменений электрических свойств материала в зависимости от количества попадающей на поверхность датчика тепловой энергии. Изменения напряжения, сопротивления либо силы тока фиксируются относительно существующего волнового фона - поэтому термостабилизированные приборы имеют в целом более низкий уровень паразитных шумов.

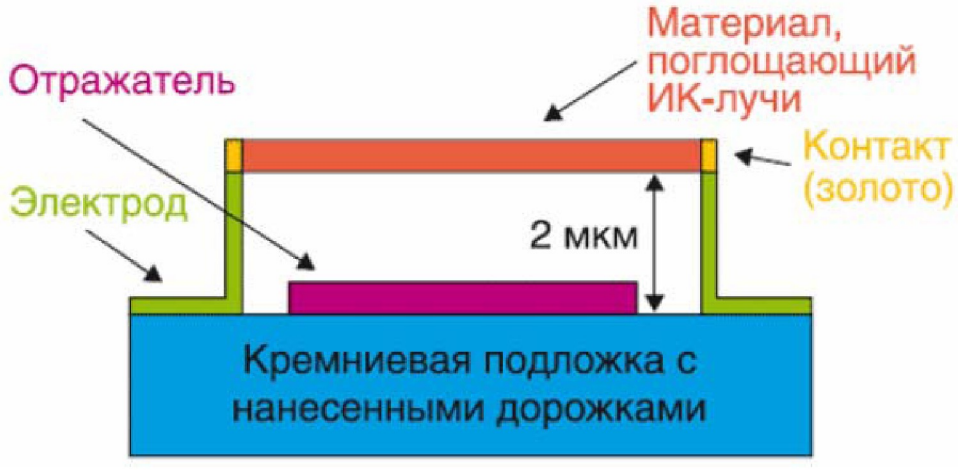


Рисунок 2 – Микроболометр

Чувствительность и уровень шума неохлаждаемых сенсоров ограничена характеристиками удельной теплопроводности и теплоемкости полупроводникового материала. Теплоемкость - предмет технического компромисса. Чтобы обеспечить приемлемую скорость реакции сенсора, она должна быть достаточно низкой. Однако при снижении удельной теплоемкости элемент становится более чувствительным к колебаниям температуры – это выливается в заметно возросшем уровне шума. Но помимо технических компромиссов, существуют еще и рыночные: при существующем уровне цен на охлаждаемые системы продажи их можно считать разовыми. Неохлаждаемые системы нашли весьма обширные области применения, и их производство стало приносить стабильные доходы. Особенно с того момента, как появились портативные тепловизионные приборы - сегодня минимизация довела их до размеров чуть больше бытовых видеокамер. [5]

В Государственном научном центре «НПО “Орион”» ведутся работы по созданию микроболометрических матриц, чувствительных в спектральном диапазоне 8—14 мкм.

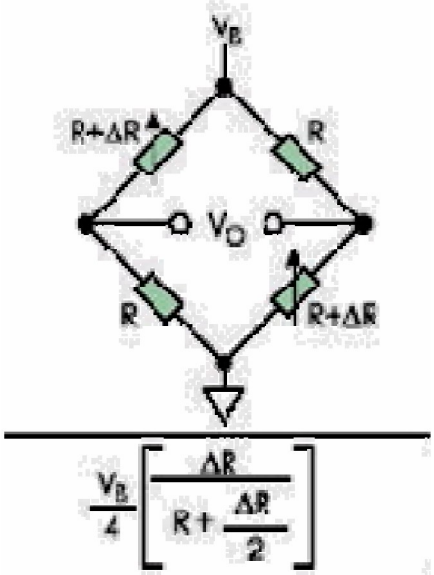


Рисунок 3 – Схема микроболометрического датчика

В качестве примера рассмотрим микроболометрический датчик МГ32А (рисунок 3), устроенный по принципу моста Уитстона. В него входят четыре одинаковых болометра, два из которых (например, левый нижний и правый верхний) изолированы от излучения и выполняют роль нагрузочных сопротивлений, а два других – служат приемниками излучения. Такая схема имеет высокую чувствительность и стабильность. Датчик питается от постоянного напряжения +5В. В него встроен источник постоянного стабильного тока. На дифференциальном выходе датчика образуется напряжение, пропорциональное освещенности. Чувствительность – 400000 В/Вт. Размер датчика 20х20 мкм, диапазон 7-16 мкм.

В ДВИК диапазоне применяется, в основном, преломляющая оптика из германия. Чистый германий обладает лучшими характеристиками пропускания в этом диапазоне. Но это очень дорогой материал. Так как для датчиков ориентации не нужно получать изображение –в качестве оптики можно использовать диафрагму. Устройство МГ-32 таково, что угол обзора каждого датчика – 17º.

Принцип определения угла отклонения МКА изображен на рисунке 1

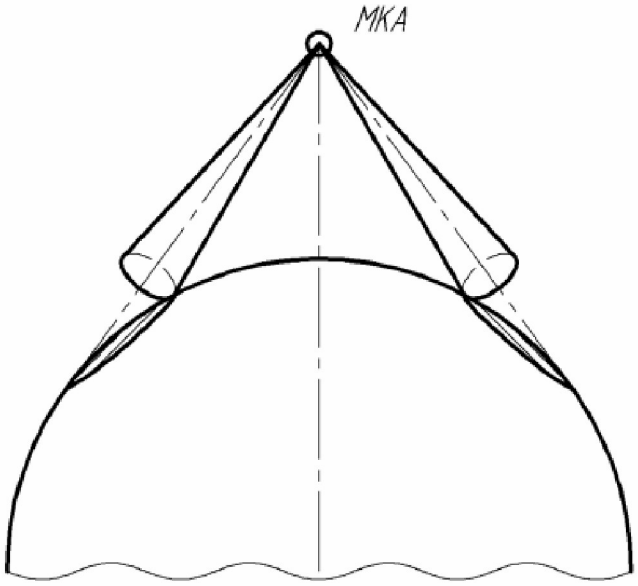


Рисунок 4 – Принцип определения угла отклонения МКА

Два ИК-датчика направлены на противоположные стороны видимого Земного горизонта. При наклоне МКА освещенность одного из датчиков уменьшается, а другого – увеличивается. Разница освещенностей датчиков пропорциональна углу отклонения. Выходное напряжение датчиков, пропорциональное освещенности, усиливается преобразователем и поступает в устройство управления. Один канал измерителя угла состоит из двух датчиков освещенности и двух преобразователей. Коды двух датчиков вычитаются в устройстве управления, и полученная разница будет пропорциональна углу отклонения МКА от вертикали.

Один такой канал позволяет стабилизировать вращение аппарата вокруг одной из двух осей – крена или тангажа. Поэтому для полной стабилизации вертикальной оси аппарата необходимо два канала, перпендикулярных друг другу. На рисунке 5 изображены поля обзора четырех датчиков. Здесь Д1 и Д2 – датчики канала крена. Д3 и Д4 – датчики канала тангажа, рисунок представляет собой вид на КА сверху.

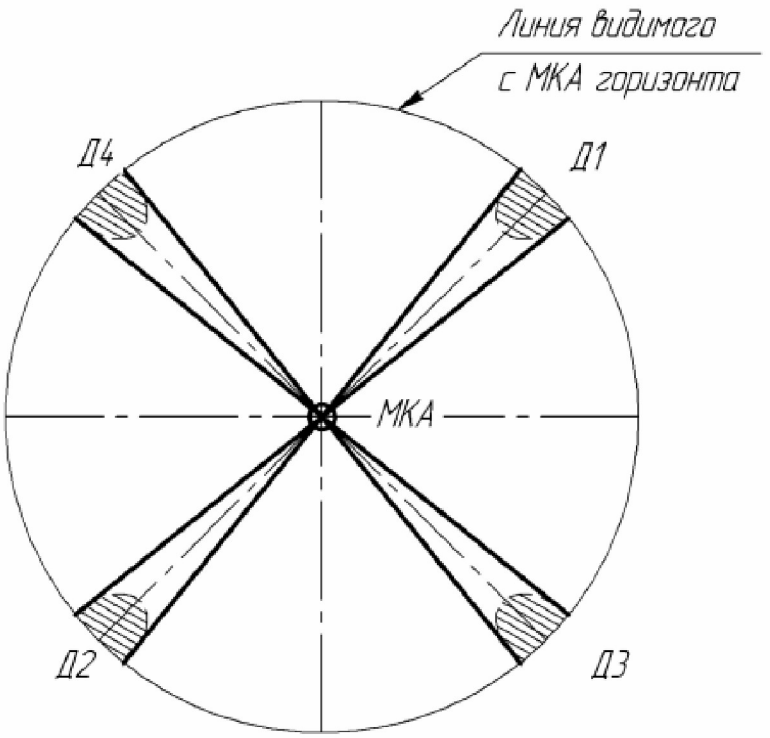


Рисунок 5 – МКА вид сверху

Датчики должны находиться под углом к вертикали, меньшим 90º, так чтобы в стабилизированном состоянии МКА оси их были направлены точно на линию видимого горизонта. Этот угол зависит от высоты орбиты. Такой прибор работает как построитель местной вертикали. Благодаря использованию длинноволнового ИК-диапазона, прибор выполняет свою задачу независимо от взаимного расположения МКА, Земли и Солнца. При наличии Солнца в поле зрения прибора можно использовать дополнительный канал ориентации по Солнцу, устроенный аналогично.

РАСЧЕТ ОСВЕЩЕННОСТИ ИНФРАКРАСНОГО ДАТЧИКА

В качестве естественных помех для приборов ориентации по Земле могут выступать Солнце и Луна. Звездное небо имеет пренебрежимо малую яркость, по сравнению с остальными источниками.

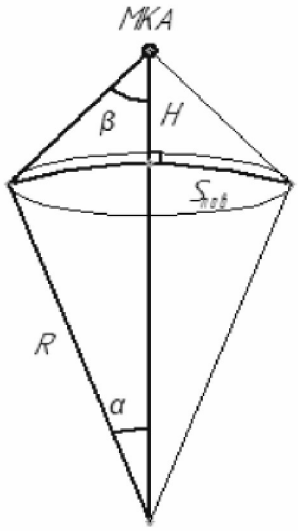


Рисунок 6 – Часть поверхности шара, видимая из точки, удаленной от нее на расстояние H

Рассмотрим шар радиуса R и точку наблюдения, удаленную на расстояние H от его поверхности. Необходимо найти угол, относительно вертикали, β, под которым виден горизонт шара из точки наблюдения.

$$\beta = \arcsin\left(\frac{R}{R+H}\right)$$

Площадь $S_{пов}$ поверхности шара с радиусом R, видимая с высоты H:

$$S_{пов} = 2 \cdot \pi \cdot R^2 \cdot \left(1 - \frac{R}{R+H}\right)$$

Угловой размер этой части поверхности – α также можно найти аналогично:

$$\alpha = \arccos\left(\frac{R}{R+H}\right)$$

Солнце в ИК-диапазоне излучает как черное тело с температурой 5770К. В диапазоне длин волн от 7 до 26 мкм содержится 0.14% всей излучаемой Солнцем энергии [1]. Мощность, сосредоточенную в диапазоне от 7 до 16 мкм можно найти из закона Планка:

$$U(\lambda, T) = \frac{2\pi \cdot h \cdot c^2}{\lambda^5} \cdot \frac{1}{e^{\frac{h \cdot c}{\lambda \cdot k \cdot T}} - 1}$$

Чтобы найти освещенность, создаваемую в точке приема, необходимо проинтегрировать вышеприведенное выражение по диапазону длин волн примененного датчика, умножить на площадь видимой по-

верхности и разделить на квадрат расстояния от датчика до Солнца. Так как Солнце интересует нас только в качестве помехи, за расстояние до Солнца можно приближено принять перигелий Земли, в этом случае освещенность будет максимальной.

$$P_C = \frac{1}{R_{оз}^2} \cdot S_{пов}\left(R_c, R_{оз}\right) \cdot \int_{\lambda_1}^{\lambda_2} U(\lambda, 5770K)\, d\lambda$$

Луна в ИК-диапазоне излучает как черное тело с температурой, соответствующей ее поверхности. Максимальная дневная температура поверхности Луны может достигать 373К, минимальная ночная температура – 120К.

Освещенность, создаваемая полной Луной, находится аналогично солнечной, за расстояние до датчика принимается перигей Луны:

$$P_{Л1} = \frac{1}{R_{ол}^2} \cdot S_{пов}\left(R_л, R_{ол}\right) \cdot \int_{\lambda_1}^{\lambda_2} U(\lambda, 373K)\, d\lambda$$

Реальная освещенность от лунного «серпа» также зависит от фазы луны.

При проектировании приборов ориентации по Земле и их испытаниях используются различные модели излучения Земли. Простейшая модель – это равнояркий диск, угловой размер которого соответствует угловому размеру Земли, наблюдаемому с заданной высоты. Энергетическая яркость диска равна средней энергетической яркости реальной Земли для выбранного спектрального диапазона.

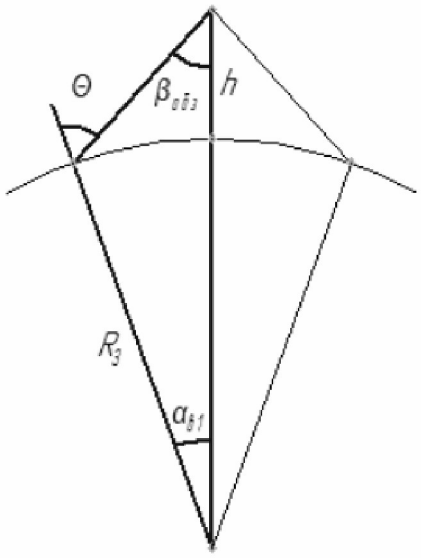


Рисунок 7 – Поверхность Земли, «видимая» датчиком, направленным вертикально вниз

Обозначим Θ - угол между нормалью к Земле и направлением на приемник (рисунок 7). По теореме синусов

$$\Theta(\beta) = \operatorname{asin}\left(\frac{R_з + h_o}{R_з}\cdot\sin(\beta)\right)$$

Видимый сектор земной поверхности:

$$\alpha_{в1} = \Theta\left(\beta_{обз}\right) - \beta_{обз}$$

Видимая поверхность Земли – это круг радиусом $\alpha_{B1}\cdot R_з$ с площадью

$$2\pi\cdot R_з^{\,2}\cdot\left(1-\cos\left(\alpha_{в1}\right)\right)$$

Расстояние от датчика до края такого "диска" равно:

$$R_з\cdot\frac{\sin\left(\Theta\left(\beta_{обз}\right)-\beta_{обз}\right)}{\sin\left(\beta_{обз}\right)}$$

Освещенность датчика, направленного вертикально на землю:

$$P_{Зн} = \left(\lambda_2-\lambda_1\right)\cdot U_з\cdot\frac{2\pi\cdot R_з^{\,2}\cdot\left(1-\cos\left(\alpha_{в1}\right)\right)}{h_o^{\,2}}$$

Освещенность, создаваемая Землей, в случае, когда датчик направлен на горизонт

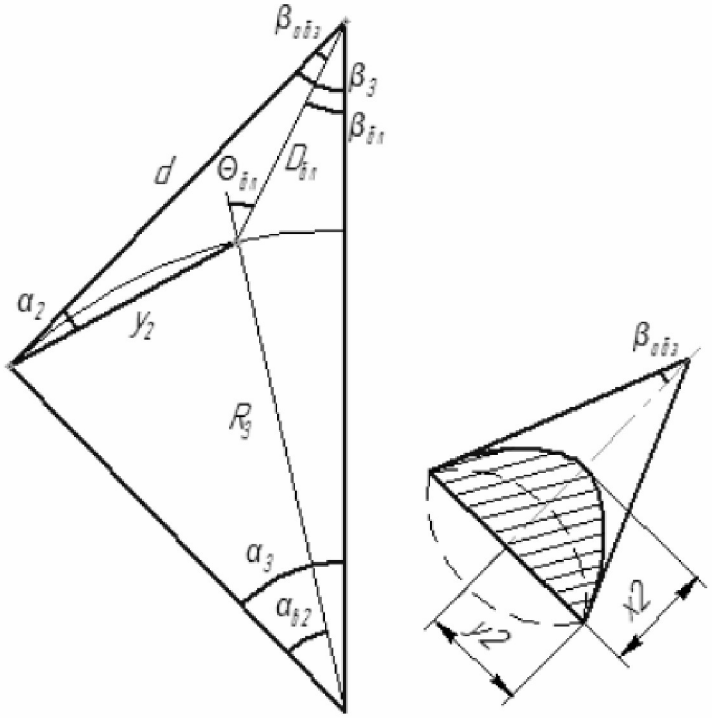


Рисунок 8 – Углы и расстояния, образующиеся при направлении датчика на видимый земной горизонт

Часть поверхности Земли, «видимая» датчиком, представляет собой пересечение сферы земного радиуса с круговым конусом, обра-

зованным вращением апертуры датчика вокруг оси, соединяющей МКА с точкой горизонта Земли.

Чтобы упростить задачу нахождения освещенности датчика, создаваемой такой площадкой, излучающую область поверхности Земли можно в первом приближении считать сечением конуса плоскостью. Такая область будет представлять собой параболу, схематично изображенную на рисунке 8. Можно показать, что освещенность можно рассчитать по формуле

$$P_{3г} = (\lambda_2 - \lambda_1) \cdot U_з \cdot \int_0^{\beta_{бл}} \left(\frac{\cos(\Theta(\beta_3 - \beta))}{D(\beta)^2} \right) \cdot 2 \cdot \left(a \cdot \sqrt{x_2 - x(\beta)} \right) \cdot dx(\beta)\, d\beta$$

Возрастающая степень интеграции ПЛИС, новые концепции проектирования (система на кристалле) накладывают свой отпечаток на способы описания проекта на ПЛИС. Языки описания аппаратуры (Hardware Description Language), являются формальной записью, которая может быть использована на всех этапах разработки цифровых электронных систем. Это возможно вследствие того, что язык легко воспринимается как машиной, так и человеком. Он может использоваться на этапах проектирования, верификации, синтеза и тестирования аппаратуры так же, как и для передачи данных о проекте, модификации и сопровождения. Существует несколько разновидностей этих языков: AHDL, VHDL, VerilogHDL, Abel и др. Известны также случаи использования стандартных языков программирования, например Си, для описания структуры БИС.

Одним из наиболее универсальных языков описания аппаратуры является VHDL, первый стандарт которого был разработан в 1983–1987 годах 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 операций языка, каждая из которых может управляться логическими сигналами. Потоковое описание соответствует стилю описания, используемому в языках регистровых передач.
- И, наконец, поведенческое описание (behavioral description), в котором логические преобразования описываются последовательными программными предложениями, которые похожи на имеющиеся в любом современном языке программирования высокого уровня. Все три стиля могут совместно использоваться в одной VHDL программе.

Структурное и потоковое описание используется для проектирования цифровых схем, поведенческое - в основном для моделирования.

Наиболее важными в языке VHDL являются такие понятия как объект проекта, интерфейс, порт, архитектура, сигнал, процесс, параллелизм и иерархия.

Объект проекта (entity) представляет собой описание компонента проекта, имеющего чётко заданные входы и выходы и выполняющие чётко определённую функцию.
Объект проекта может представлять всю проектируемую систему, или некоторую подсистему, При этом в описании объекта проекта можно использовать компоненты, которые, в свою очередь, могут быть описаны как самостоятельные объекты проекта более низкого уровня. 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 и состоящую 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 а**рхитектурное тело (architecture body)**.

В языке VHDL предусмотрен механизм пакетов для часто используемых описаний, констант, типов, сигналов. Эти описания помещаются в **Объявление 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, и поэтому оно может быть независимо проанализировано моделирующей программой. Такое описание 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 записаны в один файл, который называется **файлом проекта (design file).**

Каждый проанализированный модуль проекта помещается компилятором Active VHDL в **Библиотеку проекта (design library)** и становится **Библиотечным модулем (library unit)**. Такой подход 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.

Объекты данных (data object) являются хранилищами для значений определённого типа. Следует заметить, что VHDL - сильно типизированный язык, и он допускает операции только с данными одного типа. Для преобразования типов служат ряд специально разработанных функций, которые обычно группируются в пакеты. Описание на VHDL содержат объявления, которые создают объекты данных всего лишь четырёх классов: константы, переменные, сигналы и файлы.

Алфавит VHDL

При работе с языком VHDL следует принимать во внимание следующие особенности синтаксиса:

1. Программы на VHDL не различают прописные и заглавные буквы, т.е нет разницы между верхним и нижним регистром;
2. Идентификатор должен начинаться с буквы и состоять из букв и цифр;
3. Допускается применение знака подчеркивания, но не двух подряд и не последним символом;
4. VHDL содержит набор ключевых слов, которые не могут быть идентификаторами;
5. Комментарии начинаются двойным тире и продолжаются до конца строки.

Литералы могут быть следующих типов:

1. Числовые (целые и вещественные, допускается символ E, указывающий на порядок);
2. Символьные (символ между апострофами, например 'A');
3. Строка символов (последовательность символов в двойных кавычках, располагающаяся в одной строке);
4. Строка бит (последовательность цифр и символов A,B,C,D,E,F в двойных кавычках. Строка бит может быть двоичной, восьмеричной или шестнадцатеричной, на что указывает соответственно символ B, O, X перед кавычками. Например X"AB12". Допускается применение подчеркивания в таких константах, но оно не является значимым.

Типы

1. Целый (integer) в диапазоне от $-(2^{31}-1)$ до $2^{31}-1$;
2. Логический (boolean принимает два значения true и false);
3. Битовый (BIT принимает значения '0' и '1');
4. STD_LOGIC и STD_ULOGIC - типы, определенные стандартом IEEE 1164;
5. Перечислимый (ENUMERATED), используемый для задания пользовательских типов;
6. Символьный (CHARACTER);
7. Физический (время).

Допускается использование массивов (ARRAY) следующих предопределенных типов:

1. BIT_VECTOR;
2. STD_LOGIC_VECTOR и STD_ULOGIC_VECTOR;
3. STRING.

Указанные массивы представляют собой одномерные массивы соответствующих типов. Направление и границы диапазонов должны быть указаны при объявлении объектов данных типов. Для объявления направления и границ диапазона используются ключевые слова to и downto.

bit_vector (0 to 7); -- возрастающий -- диапазон

1	0	1	1	1	1	0	0
0	1	2	3	4	5	6	7

число **61** (десятичное)

bit_vector (7 downto 0); -- убывающий -- диапазон

1	0	1	1	1	1	0	0
7	6	5	4	3	2	1	0

число **188** (десятичное)

Например, BIT_VECTOR(7 downto 0), STD_LOGIC_VECTOR(1 to 8). Также пользователь может объявить собственные массивы, пользуясь декларацией вида:

TYPE X IS ARRAY(1 to 2) OF REAL;

Аналогично объявляются перечислимые типы, состоящие из конечного множества значений: TYPE DIGIT IS (0,1,2,3,4,5,6,7,8,9);

Допускается объявление объектов целого типа с ограниченным множеством значений: TYPE BYTE_INT 0 TO 255;

Константы

В языке VHDL допускается объявление констант следующим образом:

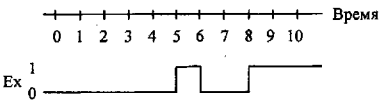
CONSTANT <name> **:** <type> **:=** initial value;

Применение констант улучшает читаемость и повышает гибкость VHDL - описаний.

Атрибуты

Атрибутами называются значения, связанные с именованными объектами языка VHDL. Для целей разработки цифровой аппаратуры нам потребуется только атрибут EVENT.

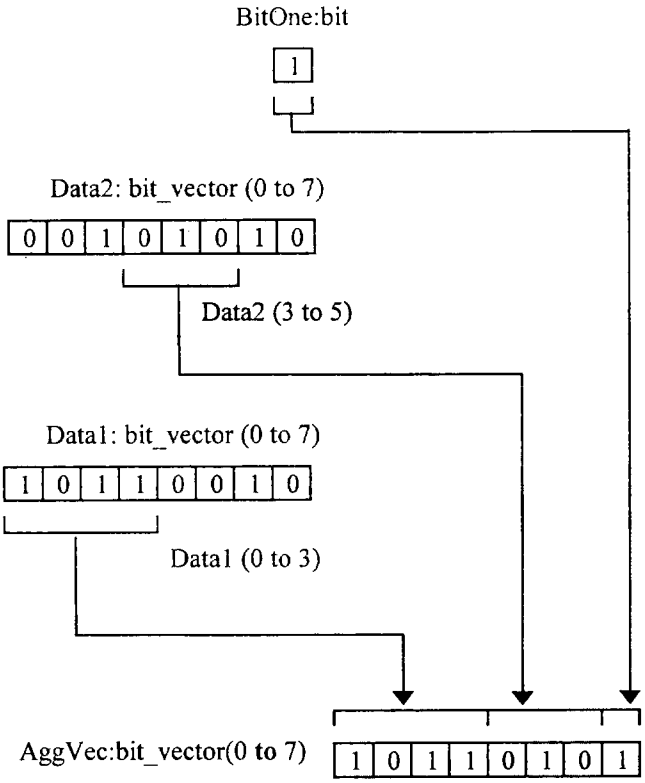
Атрибут	Время								
	t=0	t=1	t=2	t=3	t=4	t=5	t=6	t=7	t=8
Ex'transaction	false	false	true	false	false	true	true	false	True
Ex'event	false	false	false	false	false	true	true	false	True
Ex'last_value	0	0	0	0	0	0	1	0	0
Сигнал ex	0	0	0	0	0	1	0	0	1



Ex<= '0' after 2s;
'1' after 5s;
'0' after 6s;
'1' after 8s;

Операторы

Язык VHDL предполагает выполнение операций над предусмотренными типами данных слева направо и с учетом приоритета и скобок. Допускаются: возведение в степень, деление, умножение, вычисление остатка и модуля, сложение и вычитание. Логические операции: И, ИЛИ, НЕ, исключающее ИЛИ, И-НЕ, ИЛИ-НЕ, исключающее ИЛИ-НЕ. Операторы сравнения: <=,<,>=,>,=,/= (не равно). Объекты типа строка символов могут сцепляться вместе с помощью операции конкатенации (&).



AggVec<=(Data1(0 to 3)&Data (3 to 5)&BitOne);

Операторы сложения и вычитания выполняются над данными одного типа, логические операции - над данными типа BIT, STD_LOGIC и соответствующими векторами.

Различают **последовательные и параллельные операторы** языка. Последовательные операторы выполняются один за другим и применяются в описании процессов, процедур и функций. Параллельные операторы выполняются одновременно, и порядок их выполнения не связан с местом расположения в тексте программы.

Составные части VHDL программы

VHDL программа объекта проекта состоит следующих основных частей:

- описание интерфейса объекта проекта **(ENTITY)**, включающее (необязательно) **PORT** (список входных и выходных сигналов) и GENERIC (список настраиваемых констант, например, задержек, переключения и разрядностей данных);
- архитектура объекта проекта **ARCHITECTURE** , включающая объявление переменных и дополнительных (внутренних) сигналов проекта и операторную часть, описывающую объект на структурном или поведенческом уровне

Важно придерживаться определенного стиля написания программы на VHDL. Правильное расположение составных частей программы в процессе ее написания поможет избежать ряда ошибок как при ее создании, так и облегчит процесс сопровождения программного продукта в течении его жизни.

Интерфейсная спецификация начинается с ключевого слова **Entity** и описывает входные и выходные порты объекта. Другими словами, интерфейсная декларация (entity declaration) описывает интерфейс между объектом и окружением, в котором находится объект.

Синтаксис данной декларации следующий:

entity <identifier_name> **is**

port (<identifier> **:** [<mode>] <type_indication> ;
<identifier> **:** [<mode>] <type_indication>);
end [<identifier_name>] ;

В простейшем случае возможно отсутствие декларации **port**.

После ключевого слова **port** перечисляются сигналы, связывающие объект с внешним миром. Если объект имеет несколько однотипных сигналов, их идентификаторы перечисляются через запятую. После двоеточия следует указание направления сигнала, которое может принимать одно из следующих значений:

<mode> = **in, out, inout, buffer, linkage**

in = входной;

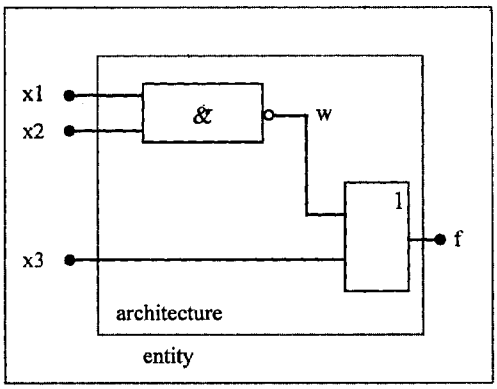
out = выходной;

inout = двунаправленный;

buffer = двунаправленный буферный;

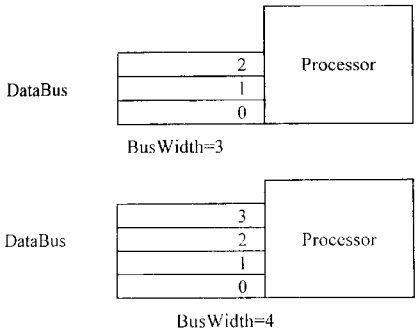
linkage = связной (не используется при проектировании).

Пример:




```
entity ANDOR is
        port (x1, x2, x3 :      in bit;
              f      :      out bit);
end ANDOR;
```

Синтаксис языка VHDL допускает настраиваемое описание интерфейса объекта с помощью ключевого слова **generic.** Его использование позволяет легко изменять настраиваемый параметр объекта, не внося других изменений в VHDL-код.




```
Entity Processor is
generic(BusWidth: Integer:=3);
port(DataBus: inout_bit_vector(BusWidth-1 downto 0);
end Processor
```

Архитектурная спецификация начинается с ключевого слова **Architecture** и описывает функциональное назначение проектируемого объекта. Другими словами, архитектурная декларация определяет тело компоненты проектируемого объекта, выполняемые функции которой зависят от значений входных и выходных сигналов или других параметров, задаваемых в интерфейсной декларации. Имя Архитектурной спецификации может быть любым, однако сама декларация привязана к имени Интерфейсной спецификации. Синтаксис данной декларации следующий:

architecture <architecture_name> **of** <entity_identifier> **is**
[<architecture_declarative_part>]
begin
<architecture_statement_part>
end [<architecture_name>];

Поскольку интерфейсному описанию одного проектируемого объекта может соответствовать несколько архитектурных деклараций с разными идентификаторами, то мы можем описывать устройство с разной степенью детализации.

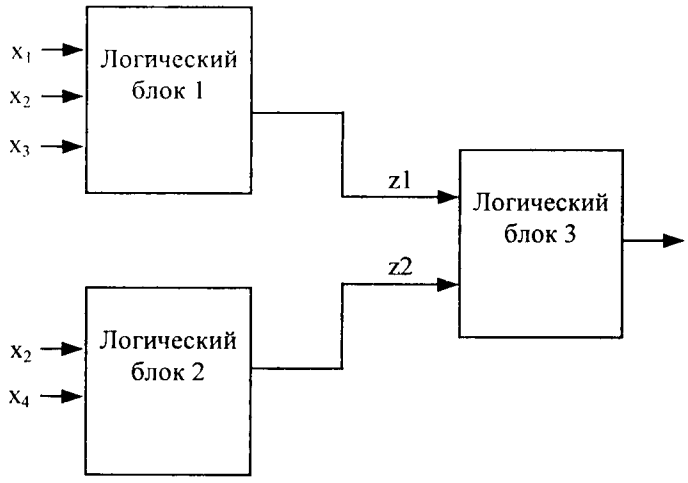
В декларативной части архитектуры могут содержаться объявления типов, сигналов, констант, подпрограмм и компонентов.

Описание архитектуры может быть выполнено в соответствии с одним из трех типов VHDL - описания: структурным, потоковым и поведенческим.

```
architecture RTL1 of ANDOR is
begin
  f <= (x1 and x2) or x3;
end RTL1;

architecture RTL2 of ANDOR is
  signal w : bit;
begin
  w <= x1 and x2;
  p1 : process (w, x3)
  begin
    f <= w or x3;
  end process p1;
end RTL2;
```

Сигналы и процессы

Понятие "сигнал" соответствует физическому проводнику на схеме устройства. Сигналу может быть присвоено значение с помощью оператора <=. Логические сигналы могут передаваться и обрабатываться параллельно, для чего в VHDL предусмотрен механизм процессов. Каждый процесс представляет собой блок моделируемой схемы, причем все процессы выполняются параллельно.



Интерфейс объекта, описываемый в декларации entity, использует сигналы, поступающие на вход устройства и формируемые им. Сигналы, не перечисленные в entity, должны быть объявлены в декларативной части архитектуры.

Синтаксис процесса имеет следующий вид:

<label:> **process** [<(sensitivity list)>]
<[process_declarations]>;
begin
<sequential operators>;
end process [<label>];

Процесс может иметь список чувствительности, в котором перечислены сигналы, активизирующие процесс. Каждый сигнал должен иметь только один источник (драйвер), в противном случае потребуется **специальная функция разрешения**. Каждый процесс может находиться в одном из трех состояний:

- выполняющийся (когда выполняется системой моделирования);
- активный (ожидающий выполнения);
- приостановленный (когда не является активным или выполняющимся).

Моделирование работы устройства посредством механизма процессов осуществляется следующим образом:

Параллельные процессы, которые необходимо выполнить (активные) формируют очередь, из которой ЭВМ выбирает один процесс и выполняет его, после чего он считается приостановленным. Затем выполняется следующий активный процесс, приостанавливается, и так далее до тех пор, пока в очереди остаются активные процессы. После этого начинается следующий цикл моделирования. Приостановленный процесс становится активным при изменении хотя бы одного сигнала из списка чувствительности. Если у процесса отсутствует список чувствительности, то он выполняется всегда. Альтернативой списку чувствительности является оператор **wait.** Наличие сигнала в списке чувствительности эквивалентно оператору **wait on** в конце процесса. Запрещается совместное использование wait и списка чувствительности.

Переменные

Переменные могут быть объявлены в области операторов процессов и подпрограмм, область их видимости будет ограничена телом того объекта, внутри которого она декларирована (в отличие от сигналов, которые не могут быть объявлены внутри процесса или подпрограммы). Значение переменным может быть присвоено в любой момент с помощью оператора := (изменение сигналов происходит только после окончания действия процесса), понятие времени не ассоциируется с понятием переменной. Переменные могут быть аргументами подпрограмм (так же, как и сигналы).

Условный оператор

Условный оператор является последовательным и имеет следующий синтаксис:

If condition **then**
Sequence of sequential statements
[**elsif** condition **then**
Sequence of sequential statements]
[else
Sequence of sequential statements]
End if;

Выражение condition должно иметь тип boolean. В операторе if может быть одна или несколько (или не одной) частей elsif, и только одна (или не одной) часть else.

```
D_FF_RST : process (D, CLK, RST)
  begin
    if RST = 1
      then Q <= '0';
        elsif rising_edge(CLK)
          then
            Q <= D;
    end if;
  end process D_FF_RST;
```

Оператор case

Оператор case также является последовательным оператором и имеет следующий синтаксис:

Case expression **is**
When choice=> Sequence of sequential statements
[**when** choice=> Sequence of sequential statements]
[**when others**=> Sequence of sequential statements]
end case;

Выражение **expression** должно иметь дискретный тип или тип одноразмерного массива символов, состоящих из строк или строк битов. Выражение **choice** должно иметь такой же тип, как и expression. Все значения выбора должны быть покрыты. Выбор **others** должен покрывать значения, не покрытые предыдущими альтернативами. Допускается указание множественного выбора (с разделением символом "|"), а также указание диапазона (с применением **to** или **downto**).

Оператор цикла

Оператор цикла имеет следующий синтаксис:

[label:] [**while** condition | **for** identifier **in** discrete_range]
loop Sequence of sequential statements
end loop [label];

В случае использования в условии цикла конструкции while, сначала вычисляется условие condition. При значении TRUE выполняется последовательность операторов, в противном случае цикл завершается.

При необходимости пропустить выполнение оператора цикла при каком-либо условии используется конструкция next со следующим синтаксисом:

Next when condition

При этом происходит переход к концу цикла и выполнение следующей итерации.

```
signal DataBus : bit_vector(3 downto 0);
signal Ones : integer;

CountOnes: process(DataBus)
  variable NumOfOnes : integer := 0;
  begin
    for Cntr in 3 downto 0 loop
      next when DataBus(Cntr)='0';
      NumOfOnes := NumOfOnes + 1;
    end loop;
    Ones <= NumOfOnes;
  end process CountOnes;
```

При необходимости прервать выполнение цикла используется оператор exit:

Exit label **when** condition

Параллельные операторы

Параллельные операторы определяют параллельное поведение схем. К параллельным относится оператор процесса и оператор параллельного вызова подпрограммы, который представляет собой оператор процесса с последовательным вызовом подпрограммы.

Параллельный оператор условного назначения сигнала

Синтаксис следующий:

Signal <= expression **when** condition **else**
[Signal <= expression **when** condition **else** ..]

```
q <=  3 WHEN high = '1' ELSE      -- when high
      2 WHEN mid  = '1' ELSE      -- when mid but not high
      1 WHEN low  = '1' ELSE      -- when low but not mid or high
      0;                          -- when not low, mid, or high
```

Данный оператор эквивалентен оператору if в процессе.

Параллельный оператор выборочного назначения сигнала

Синтаксис:

with selection **select** signal <= expression **when** choice, expression **when** choice…

Аналогично оператору case допускаются множественные выборы и применение ключевого слова others.

```
WITH s SELECT       -- creates a 4-to-1 multiplexer
      output <=     d0 WHEN 0,
                    d1 WHEN 1,
                    d2 WHEN 2,
                    d3 WHEN 3;
```

Оператор конкретизации компонента

Синтаксис:
Label : component name
Port_map_description;

Метка является обязательной, описание Port_map_description может быть двух типов: с позиционным сопоставлением и с ключевым сопоставлением.

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY meth IS
      PORT( data, clock, clearn, presetn   : IN STD_LOGIC;
            q_out                          : OUT STD_LOGIC;
            a, b, c, gn                    : IN STD_LOGIC;
            d                              : IN STD_LOGIC_VECTOR(7 DOWNTO 0);
            y, wn                          : OUT STD_LOGIC);
END meth;

ARCHITECTURE a OF meth IS
BEGIN
dff1 : DFF PORT MAP (d =>data, q => q_out, clk => clock, clrn => clearn, prn => presetn);
mux: MUXX PORT MAP (c, b, a, d, gn, y, wn);
END a;
```

В данном случае компонент DFF присоединен с ключевым соответствием (явно указано соответствие сигналов), а выводы компонента MUXX присоединены по порядку: первый вывод к c, второй - к b и т.д.

Для выходных неиспользуемых портов необходимо указать ключевое слово open.

```
P1: add1 port map(b1=>'0', b2=>'1', c1=>open);
```

Параллельный оператор генерации

Синтаксис:

label: **for** parameter_specification
generate [**if** condition **generate**]
parallel statements
end generate [label];

Параметр генерации - константа дискретного типа в определенном диапазоне (параметром не может быть объявленная переменная или сигнал).

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY meth IS
        PORT  (data, clock, clearn, presetn   : IN STD_LOGIC;
                q_out                         : OUT STD_LOGIC);
END meth;

ARCHITECTURE a OF meth IS
COMPONENT DRFF
  PORT (d,clk,          clrn,prn : IN STD_LOGIC;
           q   : OUT STD_LOGIC);
END COMPONENT;
signal TS: STD_LOGIC_VECTOR(8 downto 0);
BEGIN
TS(8)<=data;
q_out<=TS(0);
G0:     for i in 7 downto 0 generate
Allbit:  DRFF PORT MAP (d=>TS(i+1),q=>TS(i),clk=>clock,clrn=>clearn,prn=>presetn);
End generate;
END a;

entity DRFF is
PORT (d,clk,clrn,prn : IN STD_LOGIC;
           q   : OUT STD_LOGIC);
END DRFF;
Architecture ADRFF of DRFF is
begin
process(clk,clrn,prn)
        begin
        if clrn='1' then q<='0';
        elsif prn='1' then q<='1';
        elsif clk'event and clk='1' then q<=d;
        end if;
        end process;
end ADRFF;
```

Функции преобразования

В VHDL предусмотрены функции приведения типов: CONV_INTEGER(x), CONV_SIGNED(x,size), CONV_UNSIGNED(x,size), CONV_STD_LOGIC_VECTOR(x,size). Данные функции доступны при инициализации арифметической библиотеки (USE ieee_std_logic_arith.all).

Цифровые сигнальные процессоры

ЦСП – цифровой сигнальный процессор (DSP – Digital Signal Processor). ЦСП предназначены для выполнения специфических задач цифровой обработки сигналов (ЦОС). В основном ЦОС сводится к дискретизации и квантованию аналогового сигнала, выполнению определенных алгоритмов обработки полученных цифровых данных и обратному преобразованию в аналоговую форму, пригодную для восприятия человеком. ЦОС имеет ряд преимуществ перед чисто аналоговой техникой преобразования сигналов и позволяет достичь результатов, невозможных при традиционной аналоговой обработке. Алгоритмы ЦОС имеют ряд общих особенностей, учитывая которые, возможно создать специализированные процессоры, которые обеспечивают более высокую эффективность по сравнению с универсальными при решении задач ЦОС.

Основные отличия ЦСП от универсальных процессоров с архитектурой ОШ:

- архитектура;
- система макрокоманд.

В основу ЦСП положены следующие принципы:

1. Использование Гарвардской архитектуры;
2. Сокращение длительности командного цикла;
3. Применение внутренней конвейеризации;
4. Применение аппаратного умножителя;
5. Включение в систему команд специальных команд для цифровой обработки сигналов (ЦОС) и использование специальных методов адресации.

ЦСП применяется в следующих областях:

- цифровая фильтрация;
- кодирование и декодирование информации;
- распознавание звука и речи;
- обработка изображений;
- спектральный анализ;
- цифровая звукотехника;
- медицина (томографы);
- измерительная техника;
- системы управления;

Классическая **гарвардская архитектура** предполагает хранение данных и команд в разных ЗУ. В ЦСП применяется супергарвардская архитектура, имеющая несколько шин адреса и данных для доступа к памяти данных и памяти команд с возможностью одновременного доступа к нескольким областям памяти. Обычно выделяются изолированные памяти для хранения первого и второго операнда и память для хранения программ. Преимущества такой организации в следующем: несколько разделенных шин данных и адреса позволяют совмещать по времени выборку данных и использование команд.

Короткий командный цикл необходим для работы в реальном масштабе времени и высокой производительности процессора. Следует отметить, что реальный масштаб времени предполагает, что данные на выходе должны появляться в том же темпе, что и поступающие на вход (с частотой дискретизации), откуда следует, что алгоритм обработки отдельной выборки должен завершаться за время, меньшее интервала дискретизации. С развитием технологии производства СБИС и уменьшением проектных норм (более тонкие соединительные линии между активными элементами и меньшая площадь, на которой реализован транзистор) снижается длительность командного цикла процессоров и повышается тактовая частота, однако этот процесс, как считают в настоящее время многие эксперты, достиг почти максимума своих возможностей. Дальнейшее повышение тактовой частоты и уменьшение проектных норм сталкивается с серьезными физическими ограничениями и вряд ли приведет к революционным качественным изменениям по сравнению с достигнутыми результатами. Повышение производительности в настоящее время связывают с применением многоядерных структур, алгоритмов распараллеливания обработки, конвейеризацией и т.п.

Конвейерный режим используется для сокращения времени командного цикла. Обычно в ЦСП используется двух- или трехкаскадный конвейер. Он позволяет одновременно выполнять 2 – 3 команды на разных стадиях выполнения.

Стадиями обработки в трехкаскадном конвейере являются: выборка, декодирование и исполнение. Очевидно, что наибольшая эффективность будет достигаться при линейном выполнении программы с минимальным количеством ветвлений. Наиболее просто реализуется конвейер при фиксированной длине команды.

Таблица: Стадии выполнения команд трехкаскадного конвейера

Цикл	Состояние внутреннего конвейера		
	выборка	декодирование	исполнение
инициализация			
предварительный 1	команда 1		
предварительный 2	команда 2	команда 1	
1	команда 3	команда 2	команда 1
2	команда 4	команда 3	команда 2
3	команда 5	команда 4	команда 3
…	…	…	…

Различают внутреннюю и внешнюю конвейеризации. При внутренней обработка команд осуществляется внутри ЦСП. Конвейеризация на внутреннем уровне позволяет реализовать только внутренний командный цикл. Внешняя конвейеризация, по сути, есть способ распараллеливания вычислений между несколькими отдельными процессорами. Такая конвейеризация применяется чаще всего в систолических алгоритмах и вычислителях. Систолические вычислители предоставляют возможность повышения производительности не за счет внутренних архитектурных особенностей процессоров, а за счет использования принципов параллельной и конвейерной обработки данных на физическом уровне. Они реализуются с помощью процессорной платы со многими процессорами. Примерами реализации систолических архитектур являются матричные операции, фильтрация, преобразования Фурье и др. Соответствующие систолические ВА позволяют заменить программы, подпрограммы или функции аппаратной реализацией алгоритма и позволяют сократить время выполнения программы на 1 – 3 порядка.

Аппаратный умножитель применяется в ЦСП для сокращения длительности времени исполнения одной из основных операций ЦОС – операции умножения. Например, при цифровой фильтрации необходимо перемножение матрицы коэффициентов на вектор с компонентами сигнала для разных моментов времени. В алгоритмах быстрого преобразования Фурье применяется операция умножения с накоплением (Y=Y+A*B). В ЦСП операция умножения выполняется за один командный цикл благодаря специальному аппаратному умножителю.

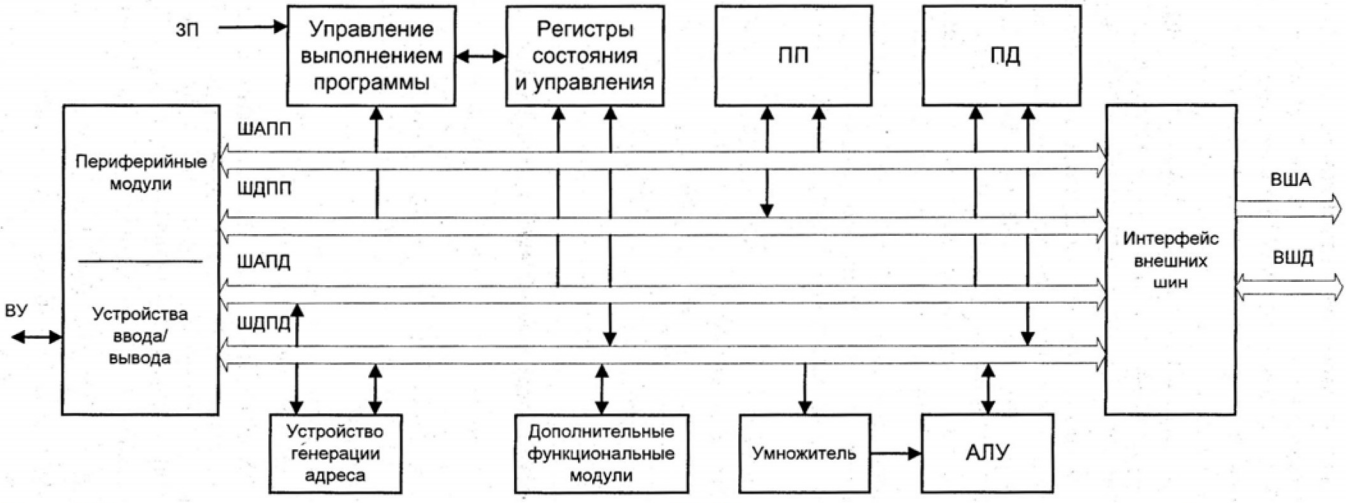
Специальные команды для ЦОС вводятся в систему команд ЦСП для ее оптимизации и для выполнения базовых задач и алгоритмов ЦОС в масштабе реального времени. Как правило, специальные команды выполняют за один цикл операцию, для выполнения которой универсальному процессору понадобилось бы несколько циклов. В ряде случаев ЦСП используют комбинированные команды, представляющие собой комбинацию нескольких элементарных операций, выполняемых параллельно. Ряд ЦСП имеют в своем составе специальные периферийные модули, для управления которыми в системе команд предусматриваются специальные инструкции.

Историческая справка

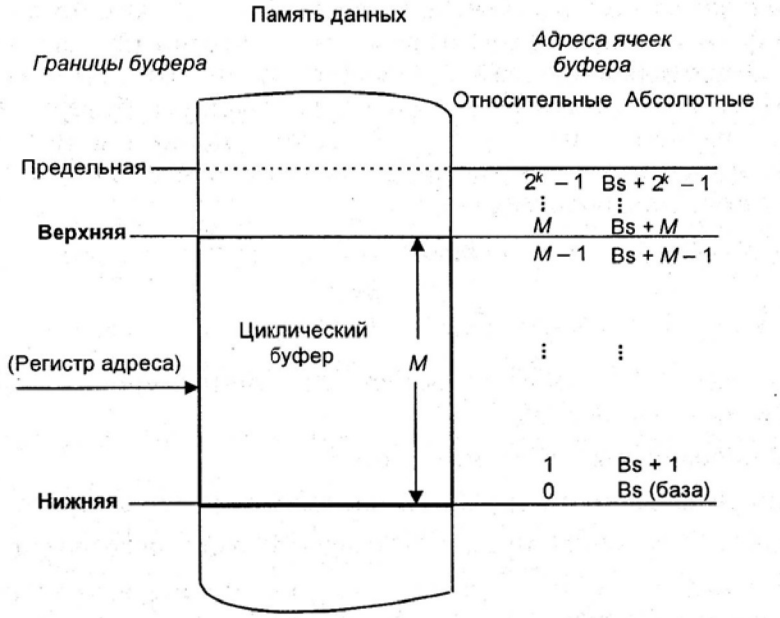
В 1982 году фирма Texas Instruments выпустила ЦСП TMS 32010. Т.к. он был первым, то он стал своеобразным промышленным стандартом в области ЦСП, т.е. другие фирмы в качестве стандарта стали использовать некоторые его архитектурные решения. Он был 16-разрядным, выполнял 5 млн. операций типа “×” и “+” в секунду. В составе TMS 32010 имелось встроенное ОЗУ, емкостью 144 или 256 слов, встроенное ПЗУ, емкостью 2048 слов, арифметико-логического устройства (АЛУ) с аккумулятором разрядностью 32 бита, встроенный умножитель 16×16 бит (операнды) с 32-разрядным результатом.

Архитектура ЦСП

С учетом вышесказанного типичный ЦСП имеет гарвардскую (супергарвардскую) архитектуру с несколькими шинами адреса и данных, высокопроизводительные АЛУ, аппаратный умножитель и сдвигатель, набор периферийных модулей.



Для эффективного управления шинами в состав ЦСП включаются специальные устройства генерации адреса (УГА), функцией которых является манипуляция адресами и поддержка **специальных режимов адресации.** К специальным режимам адресации относятся циклическая и бит-реверсивная адресация. **Циклическая адресация** обеспечивает автоматическую поддержку замкнутых циркулярных буферов с произвольным шагом (инкрементом или декрементом).



Принцип циклической адресации описывается выражениями:

$$EA_{i+1} = (EA_i - Bs + N) \bmod M + Bs \text{ , если } EA_i - Bs + N \geq 0$$

$$EA_{i+1} = (EA_i - Bs + N) \bmod M + Bs + M \text{ , если } EA_i - Bs + N < 0 \text{ ,}$$

где N - величина инкремента/декремента

Циклическая адресация в ЦСП реализуется автоматически средствами УГА без затраты дополнительных циклов на вычисление необходимого адреса.

Бит-реверсивная адресация применяется при реализации алгоритмов быстрого преобразования Фурье (БПФ), при котором операнды выбираются в нестандартном порядке. Прямая реализация бит-реверсии программным образом требует довольно много манипуляций, поэтому этот вид адресации реализуется либо специальными командами либо аппаратным способом с помощью УГА, что значительно более выгодно с точки зрения производительности процессора.

Порядок следования *n* исходных отсчетов *x* (*n*)

Прямой		Бит-реверсивный	
ятичная система	Двоичная система	Двоичная система	Десятичная система
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Как правило, ЦСП имеют в своем составе мощный контроллер ПДП, который позволяет осуществлять независимые от ядра пересылки массивов данных между блоками памяти и периферийными устройствами. Обычно реализуется ПДП двух видов: память-память и память-периферийные устройства. Иногда используются пересылки типа периферийное устройство - периферийное устройство, хотя для реализации такой функции использование ЦСП вряд ли целесообразно.

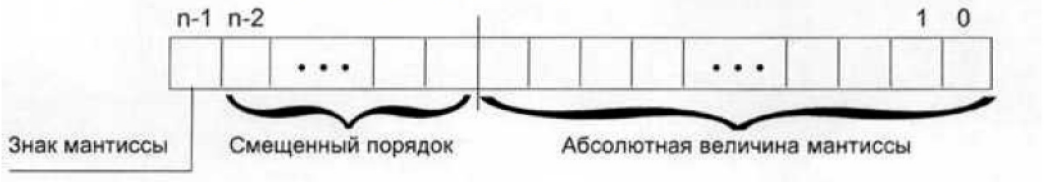
Классификация ЦСП

В настоящее время ЦСП выпускаются многими фирмами, наиболее известными следует признать Texas Instruments, Analog Devices, Freescale. Все многообразие выпускаемых ЦСП можно классифицировать следующим образом:

- ЦСП со стандартной архитектурой
- ЦСП с улучшенной стандартной архитектурой
- VLIW ЦСП
- Суперскалярные ЦСП
- Гибридные ЦСП

ЦСП стандартной и улучшенной стандартной архитектуры в основном соответствуют вышеописанным принципам и отличаются разрядностью, шинной архитектурой, наличием и объемом памяти и т.п. Повышение производительности в процессорах улучшенной стандартной архитектуры достигается, в основном увеличением количества одновременно выполняемых операций. Суперскалярные и VLIW процессоры были рассмотрены ранее, гибридные ЦСП представляют собой скорее микроконтроллеры, чем процессоры, предназначены для выполнения относительно узких функций (управление электроприводом, телекоммуникационные задачи), однако имеют все черты ЦСП.

Принято также делить ЦСП на процессоры с фиксированной и плавающей точкой. Процессоры с фиксированной точкой оперируют целочисленными значениями или дробными с фиксированным форматом (1.15 или 1.31), ЦСП с плавающей точкой поддерживают несколько международных стандартных форматов (наиболее популярен стандарт IEEE754), различающихся по точности, но имеющих одинаковую структуру следующего вида:



Здесь порядок n-разрядного нормализованного числа задается в так называемой смещенной форме: если для задания порядка выделено k разрядов, то к истинному значению порядка, представленного в дополнительном коде, прибавляют смещение, равное $(2^{k-1} - 1)$. Например, порядок, принимающий значения в диапазоне от —128 до +127, представляется смещенным порядком, значения которого меняются от 0 до 255.

Использование смещенной формы позволяет производить операции над порядками, как над беззнаковыми числами, что упрощает операции сравнения, сложения и вычитания порядков, а также упрощает операцию сравнения самих нормализованных чисел.

Чем больше разрядов отводится под запись мантиссы, тем выше точность представления числа. Чем больше разрядов занимает порядок, тем шире диапазон от наименьшего отличного от нуля числа до наибольшего числа, представимого в машине при заданном формате.

Стандартные форматы представления вещественных чисел:

1) одинарный — 32-разрядное нормализованное число со знаком, 8-разрядным смещенным порядком и 24-разрядной мантиссой (старший бит мантиссы, всегда равный 1, не хранится в памяти, и размер поля, выделенного для хранения мантиссы, составляет только 23 разряда).

2) двойной — 64-разрядное нормализованное число со знаком, 11-разрядным смещенным порядком и 53-разрядной мантиссой (старший бит мантиссы не хранится, размер поля, выделенного для хранения мантиссы, составляет 52 разряда).

3) расширенный — 80-разрядное число со знаком, 15-разрядным смещенным порядком и 64-разрядной мантиссой. Позволяет хранить ненормализованные числа.

Следует отметить, что вещественный формат с m-разрядной мантиссой позволяет абсолютно точно представлять m-разрядные целые числа, т. е. любое двоичное целое число, содержащее не более m разрядов, может быть без искажений преобразовано в вещественный формат.

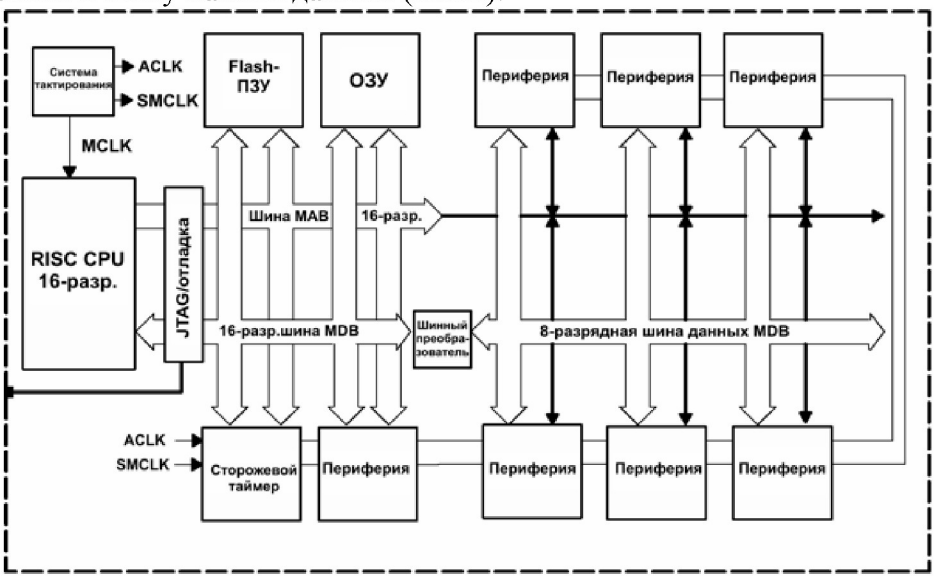
ЦСП с плавающей точкой обычно сложнее и дороже процессоров с фиксированной точкой, но обладают рядом преимуществ:

- Высокая точность;
- Практически неограниченный диапазон значений;
- Улучшенная совместимость с языком Си;
- Отсутствие необходимости в масштабировании.

Под масштабированием здесь понимается часто используемый в целочисленной арифметике прием, когда один из операндов масштабируют, т.е. специально умножают на масштабный множитель, чтобы добиться необходимой точности при операции деления.

Архитектура

Микроконтроллеры семейства MSP430 содержат 16-разрядное RISC CPU, периферийные модули и гибкую систему тактирования, соединенные через фон-Неймановскую общую адресную шину (MAB) памяти и шину памяти данных (MDB).

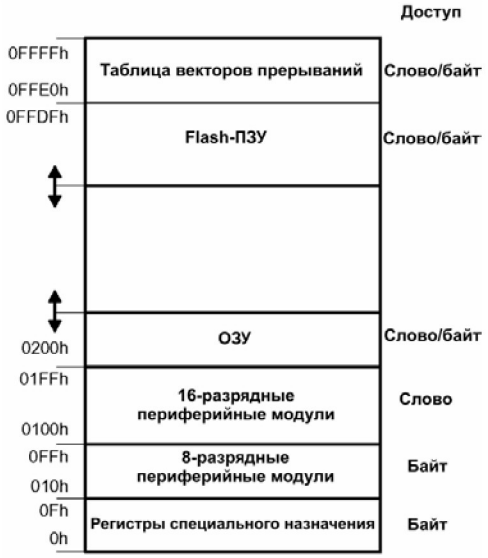


Система тактирования разработана специально для использования в приложениях с питанием от батарей. Вспомогательная низкочастотная система тактирования (ACLK) работает непосредственно от обычного 32 кГц часового кристалла. Модуль ACLK может использоваться в качестве фоновой системы реального времени с функцией самостоятельного «пробуждения». Интегрированный высокоскоростной осциллятор с цифровым управлением (DCO) может быть источником основного тактирования (MCLK) для ЦПУ и высокоскоростных периферийных устройств. Модуль DCO становится активным и стабильным менее чем через 6 мкС после запуска.

Семейство MSP430 имеет фон-Неймановскую архитектуру с единым адресным пространством для регистров специального назначения (SFR), периферии, ОЗУ и Flash-памяти программ.

Организация памяти

Байты расположены в четных или нечетных адресах. Слова располагаются только в четных адресах. При работе с командами-словами должны использоваться только четные адреса. Младший байт слова всегда расположен по четному адресу. Старший байт – в следующем нечетном адресе. Например, если слово данных расположено по адресу xxx4h, то младший байт слова данных будет иметь адрес xxx4h, а старший байт слова адрес xxx5h.



Flash-память программ

Начальный адрес Flash-памяти зависит от объема имеющейся памяти и различается для разных устройств. Конечный адрес Flash-памяти всегда 0FFFFh. Flash - память может использоваться как для программного кода, так и для данных. Байты или слова таблиц данных могут со-

храняться и использоваться непосредственно в Flash-памяти, что исключает необходимость копировать эти таблицы в ОЗУ перед дальнейшим использованием.

Таблица векторов прерываний занимает верхние 16 слов адресного пространства Flash-памяти, при этом вектор прерывания с наивысшим приоритетом находится в самом верхнем адресном слове Flash-памяти (0FFFEh).

ОЗУ начинается с адреса 0200h. Конечный адрес ОЗУ зависит от объема представленной памяти и различается для каждого конкретного устройства. ОЗУ может использоваться как для программного кода, так и для данных.

Периферийные модули

Периферийные модули отображены в адресном пространстве. Адреса с 0100 до 01FFh зарезервированы для 16-разрядных периферийных модулей. Они будут доступны с помощью команд-слов. Если используются однобайтные команды, допустимы только четные адреса, при этом старший байт результата всегда будет содержать «0».

Адресное пространство с 010h по 0FFh зарезервировано для 8-разрядных периферийных модулей. Эти модули доступны с помощью однобайтных команд. Чтение байтов модулей с помощью команд-слов приведет к появлению в старшем байте непредсказуемого содержимого. Если в байт модуля будут записываться данные в виде слова, то в регистре периферийного модуля сохранится только младший байт этого слова, старший будет проигнорирован.

Некоторые функции периферии конфигурируются в SFRs. Регистры специального назначения расположены в низших 16-ти байтах адресного пространства и организованы в виде байтов. Обращение к регистрам SFRs производится только с использованием однобайтных команд. Назначение отдельных битов регистров SFRs описано в техническом руководстве на каждое конкретное устройство.

Прерывания

Прерывания делятся на три типа:

- Системное (системный сброс)
- Немаскируемое (NMI)
- Маскируемое

Системный сброс

Источниками сброса могут быть сигналы сброса при включении (POR) и очистки при включении (PUC). Различные события и исходные условия определяют, какой именно из этих сигналов будет сгенерирован.

Сигнал POR сбрасывает устройство. Он может быть сгенерирован в следующих двух случаях:

- включение устройства;
- появление сигнала низкого уровня на выводе RST/NMI, когда он сконфигурирован как вход сигнала «сброса».

Сигнал PUC генерируется всегда при появлении сигнала POR, сигнал POR не генерируется сигналом PUC. Следующие события приводят к появлению сигнала PUC:

- сигнал POR;
- срабатывание «сторожевого» таймера (только если сторожевой таймер активирован);
- произошло нарушение ключа безопасности «сторожевого» таймера;
- произошло нарушение ключа безопасности Flash-памяти.

Исходное состояние устройства после системного сброса

После снятия сигнала POR, MSP430 переходит в следующее состояние:

- Вывод RST/NMI конфигурируется как вход «сброса»
- Выводы ввода/вывода переключаются в режим ввода в соответствии с описанием в разделе «Цифровые входы/выходы»
- Другие периферийные модули и регистры инициализируются так, как описано в соответствующих разделах этого руководства
- Регистр статуса (SR) сбрасывается

- Сторожевой таймер активизируется в сторожевом режиме
- В программный счетчик загружается адрес, содержащийся в вектора сброса (0FFFEh). ЦПУ начинает выполнять команды с этого адреса.

Программная инициализация

После системного сброса пользовательское программное обеспечение должно инициализировать MSP430 в соответствии с требованиями конкретного приложения. Необходимо выполнить следующие действия:

- Инициализировать указатель стека SP (как правило, указывается вершина ОЗУ)
- Инициализировать сторожевой таймер в зависимости от требований приложения
- Сконфигурировать периферийные модули в зависимости от требований приложения

Немаскируемые прерывания

Немаскируемые прерывания NMI не маскируются общим битом разрешения прерываний (GIE), но могут управляться индивидуальными битами включения прерывания (ACCVIE, NMIIE, OFIE). Когда происходит немаскируемое прерывание NMI, все биты разрешения NMI-прерываний автоматически сбрасываются. Выполнение программы продолжается с адреса, содержащегося в векторе немаскируемого прерывания (0FFFCh). Программное обеспечение пользователя должно установить необходимые биты NMI-прерывания, чтобы оно было разрешено вновь. Немаскируемое прерывание NMI может быть вызвано тремя событиями:

- Появление фронта сигнала на выводе RST/NMI
- Появление неисправности осциллятора
- Нарушение доступа к флэш-памяти

Вывод Reset/NMI

При включении микроконтроллера вывод RST/NMI конфигурируется как вывод сброса. Его функциональное назначение определяется в регистре управления сторожевым таймером WDTCTL. Если вывод RST/NMI запрограммирован на функцию сброса, ЦПУ будет находиться в состоянии сброса до тех пор, пока на этом выводе присутствует сигнал низкого уровня. После смены уровня на этом входе на лог.«1», ЦПУ начинает выполнять программу с команды, адрес которой хранится в векторе сброса (0FFFEh).

Если вывод RST/NMI сконфигурирован программой пользователя как вход вызова немаскируемого прерывания, фронт сигнала, выбранного битом NMIES вызовет NMI-прерывание, если установлен бит NMIIE. Также будет установлен флаг NMIFG

Неисправность осциллятора

Сигнал неисправности осциллятора позволяет предотвратить ошибки, связанные с неправильным функционированием осциллятора. Установкой бита OFIE можно разрешить генерацию NMI-прерывания при неисправности осциллятора. С помощью флага OFIFG процедура обработки NMI-прерывания может проверить, было ли NMI-прерывание вызвано неисправностью осциллятора. Сигнал неисправности осциллятора может быть вызван PUC-сигналом, поскольку он переводит осциллятор LFXT1 из режима HF в режим LF. Сигнал PUC также отключает осциллятор XT2.

Нарушение доступа к Flash-памяти

Флаг ACCVIFG устанавливается, когда происходит нарушение доступа к Flash-памяти. Генерация NMI-прерывания при нарушении доступа к Flash-памяти происходит при установленном бите ACCVIE. Проверкой флага ACCVIFG в процедуре обработки NMI-прерывания можно определить, было ли вызвано прерывание нарушением доступа к Flash-памяти.

Маскируемые прерывания

Маскируемые прерывания вызываются периферийными устройствами, имеющими возможность вызова прерываний, включая ситуацию переполнения сторожевого таймера в активном режиме. С помощью индивидуальных битов разрешения прерывания можно отключать источники прерываний как по отдельности, так и все сразу с использованием общего бита разрешения всех прерываний (GIE) в регистре статуса (SR).

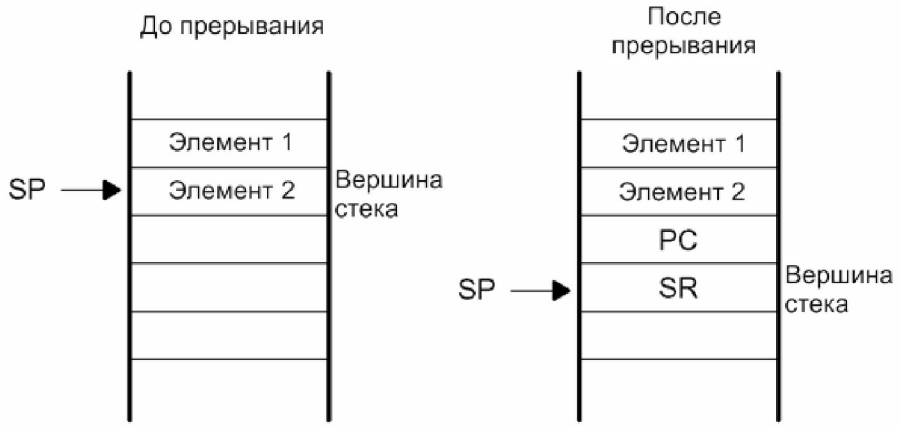
Обработка прерывания

Если периферийное устройство запросило прерывание и включены биты общего разрешения прерываний GIE и индивидуальный бит разрешения прерывания от этого устройства, будет вызвана процедура обработки прерывания. Для вызова немаскируемого (NMI) прерывания достаточно установки только индивидуального бита разрешения прерывания.

Время задержки вызова прерывания составляет 6 машинных циклов, с момента приема запроса на прерывание и до начала выполнения первой команды процедуры обработки прерывания. Логика обработки запроса прерывания имеет следующую последовательность:

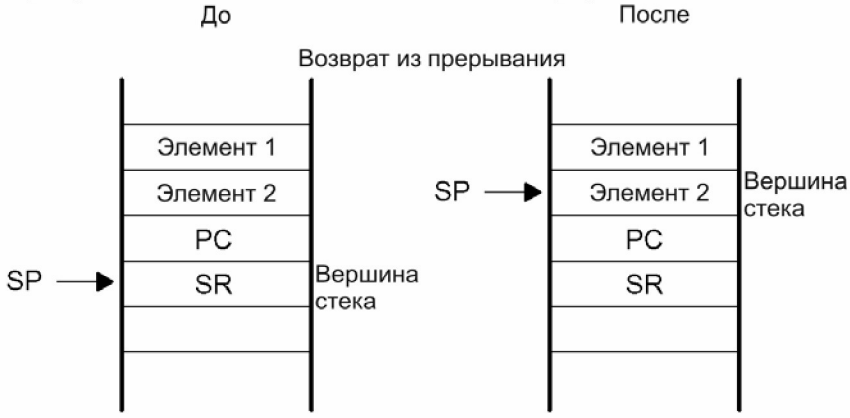
1. Любая текущая команда выполняется до конца;
2. Содержимое программного счетчика PC, указывающего на следующую команду, помещается в стек;
3. Содержимое регистра статуса SR помещается в стек;
4. Если поступило несколько прерываний во время выполнения последней команды, обрабатывается прерывание с наивысшим приоритетом, остальные ожидают обслуживания;

Автоматически сбрасывается флаг одного источника прерывания. Флаги запроса остальных прерываний остаются установленными в ожидании обслуживания программным обеспечением. Регистр SR очищается, за исключением бита SCG0, остающегося неизменным. Процессор переходит из режима пониженного потребления в активный режим; Содержимое вектора прерывания загружается в PC и начинается выполнение процедуры обработки прерывания с загруженного адреса.



Возврат из прерывания

Подпрограмма обработки прерывания заканчивается командой RETI (возврат из подпрограммы обработки прерывания) Для возврата из прерывания необходимо 5 машинных циклов. Восстанавливается из стека содержимое регистра SR. Становятся актуальными все предыдущие установки GIE, CPUOFF и пр., в не зависимости от установок, использовавшихся в процедуре обработке прерывания. Восстанавливается из стека содержимое программного счетчика PC и начинается выполнение программы с того места, где она была прервана



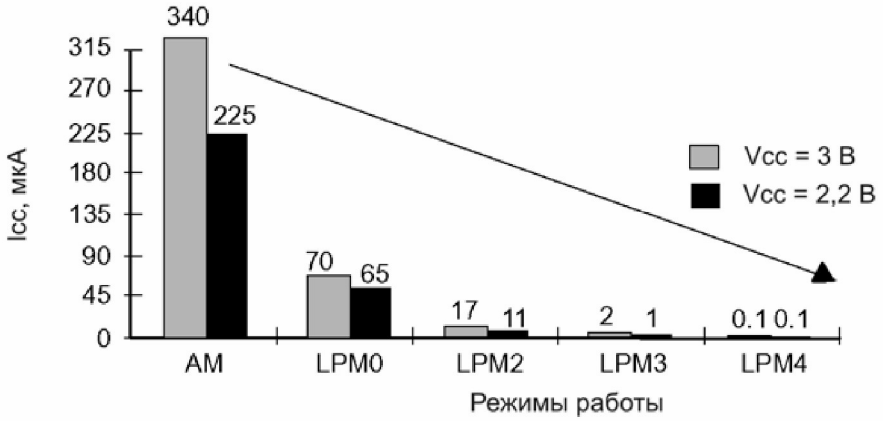
Допускаются вложенные прерывания, если бит GIE установлен во время выполнения процедуры обработки прерывания.

Векторы прерываний

Векторы прерываний и стартовые адреса расположены в адресном диапазоне с 0FFFFh по 0FFE0h. Вектор программируется пользователем с помощью указания 16-разрядного стартового адреса соответствующей процедуры обработки прерывания.

Режимы работы

Семейство MSP430 разработано для приложений с ультранизким потреблением мощности и имеет различные режимы работы.



Режимы работы учитывают три различные потребности:

- ультранизкое потребление
- скорость и пропускную способность
- минимизацию потребления тока конкретной периферией

Режимы низкого энергопотребления 0-4 конфигурируются с помощью битов CPUOFF, OSCOFF, SCG0 и SCG1 в регистре статуса. Преимущество включения битов управления режимом CPUOFF, OSCOFF, SCG0 и SCG1 в состав регистра статуса SR состоит в том, что текущий режим работы может быть сохранен, путем помещения содержимого SR в стек во время работы процедуры обработки прерывания. Выполняемая программа возвращается к предыдущему режиму работы, если сохраненное содержимое регистра SR не было изменено процедурой обработки прерывания. Выполнение программы может продолжиться в другом рабочем режиме, если процедура обработки прерывания изменит значение регистра SR в стеке. Обращение к битам управления режимом и стеку может производиться с помощью любой команды.

При изменении любого бита управления режимом, выбранный режим работы активизируется немедленно. При отключении любой системы тактирования, блокируются также периферийные устройства, работающие от этой системы. Периферийные устройства также могут отключаться с помощью соответствующих им индивидуальных управляющих регистров. Состояние всех выводов портов ввода/вывода и ячеек ОЗУ остается неизменным. «Пробуждение» возможно через все разрешенные прерывания.

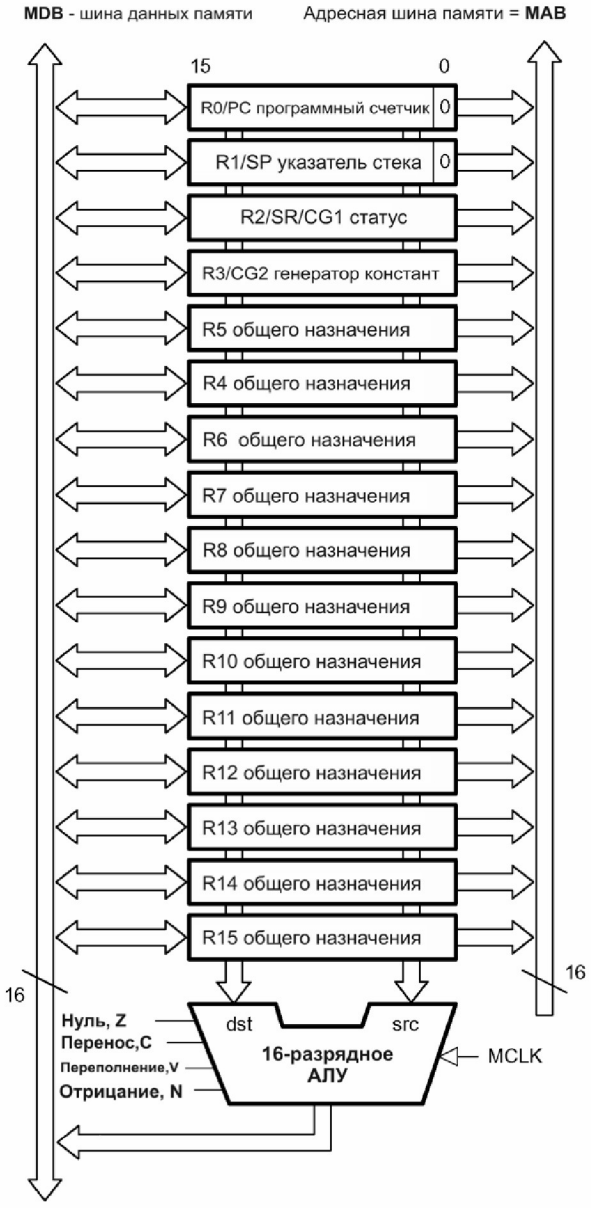
SCG1	SCG0	OSCOFF	CPUOFF	Режим	Состояние ЦПУ и систем тактирования
0	0	0	0	Активный	ЦПУ и все системы тактирования активны
0	0	0	1	LPM0	ЦПУ и MCLK отключены; SMCLK и ACLK активны
0	1	0	1	LPM1	ЦПУ, MCLK и DCO-осцил. отключены; DC генератор отключен, если DCO не используется для MCLK или SMCLK в активном режиме; SMCLK и ACLK активны
1	0	0	1	LPM2	ЦПУ, MCLK, SMCLK и DCO-осцил. отключены; DC генератор остается включенным; ACLK активно
1	1	0	1	LPM3	ЦПУ, MCLK, SMCLK и DCO-осцил. отключены; DC отключен; ACLK активно
1	1	1	1	LPM4	ЦПУ и все системы тактирования отключены

Вход и выход из режимов пониженного энергопотребления

Появление прерывания выводит микроконтроллер семейства MSP430 из любого режима пониженного энергопотребления. Программный поток выглядит так:

- Вход в процедуру обработки прерывания:
  - Содержимое регистров PC и SR сохраняется в стеке;
  - Биты CPUOFF, SCG1 и OSCOFF автоматически сбрасываются;
- Параметры для возвращения из процедуры обработки прерывания:
  - Исходное содержимое регистра SR восстанавливается из стека, что приводит к возобновлению работы устройства в предыдущем режиме;
  - Биты регистра SR, сохраненного в стеке, могут быть модифицированы процедурой обработки прерывания, что приведет к переходу в другой рабочий режим после выполнения команды RETI.

ЦПУ



ЦПУ обладает RISC-архитектурой с 27 командами и 7 режимами адресации;

- Ортогональная архитектура, при которой каждая команда пригодна для каждого режима адресации;
- Полный доступ ко всем регистрам, включая программный счетчик, регистры статуса и указатель стека;
- Однотактные регистровые операции;
- Большой 16-разрядный регистровый файл, уменьшающий количество обращений к памяти;

- 16-разрядная адресная шина, обеспечивающая прямой доступ и ветвление во всем диапазоне памяти;
- 16-разрядная шина данных, позволяющая напрямую манипулировать параметрами шириной в слово;
- Генератор констант немедленно предоставляет шесть используемых наиболее часто значений, уменьшая размер кода;
- Прямой обмен между ячейками памяти без промежуточной записи в регистр;
- Команды и адресация в форматах «слово» и «байт».

ЦПУ включает шестнадцать 16-разрядных регистров. Регистры R0, R1, R2 и R3 имеют специальное назначение. Регистры с R4 по R15 являются рабочими регистрами общего назначения.

Программный счетчик (PC)

16-разрядный программный счетчик (PC/R0) указывает на следующую команду, которая будет выполняться. Каждая команда состоит из четного числа байтов (два, четыре или шесть), поэтому PC инкрементируется соответственно. Команды доступа в адресном пространстве 64 кБайт выполняются к границам слов, поэтому PC выравнивается к четным адресам. Программный счетчик PC может быть адресован всеми командами и во всех адресных режимах.

Указатель стека (SP)

Указатель стека (SP/R1) используется ЦПУ для хранения адресов возврата из подпрограмм и прерываний. Стек основан на предекрементной постинкрементной схеме. Кроме того, указатель стека SP может использоваться со всеми командами и во всех адресных режимах. Указатель стека SP инициализируется в ОЗУ пользователем и выравнивается к четным адресам.

Регистр статуса (SR)

Регистр статуса (SR/R2), используемый как регистр источника или получателя, может адресоваться в регистровом режиме только с помощью команд-слов. Прочие комбинации режимов адресации используются для поддержки генератора констант.

<table>
<tr><th>Бит</th><th colspan="2">Описание</th></tr>
<tr><td rowspan="3">V</td><td colspan="2">Бит переполнения. Этот бит устанавливается, если результат арифметической операции имеет переполнение в области знаковых переменных.</td></tr>
<tr><td>ADD(.B),ADDC(.B)</td><td>Устанавливается, когда:
Положительный + Положительный = Отрицательный
Отрицательный + Отрицательный = Положительный
в противном случае сбрасывается</td></tr>
<tr><td>SUB(.B),
SUBC(.B),
CMP(.B)</td><td>Устанавливается, когда:
Положительный – Отрицательный = Отрицательный
Отрицательный – Положительный = Позитивный
в противном случае сбрасывается</td></tr>
<tr><td>SCG1</td><td colspan="2">Системный тактовый генератор 1. Когда этот бит установлен, SMCLK выключен.</td></tr>
<tr><td>SCG0</td><td colspan="2">Системный тактовый генератор 0. Когда этот бит установлен, генератор DCO выключен, если DCOCLK не используется для MCLK или SMCLK.</td></tr>
<tr><td>OSCOFF</td><td colspan="2">Выключение осциллятора. Когда этот бит установлен, осциллятор LFXT1, использующий кристалл, выключен, если LFXT1CLK не используется для MCLK или SMCLK.</td></tr>
<tr><td>CPUOFF</td><td colspan="2">Выключение ЦПУ. Когда этот бит установлен, ЦПУ выключено.</td></tr>
<tr><td>GIE</td><td colspan="2">Общий бит разрешения прерываний. Когда этот бит установлен, маскируемые прерывания разрешены. Когда сброшен, все маскируемые прерывания запрещены.</td></tr>
<tr><td>N</td><td colspan="2">Бит отрицательного результата. Этот бит устанавливается, когда результат операции с байтом или словом отрицательный и сбрасывается, когда результат не отрицательный. Операции со словами: N устанавливается по значению бита 15 результата Операции с байтами: N устанавливается по значению бита 7 результата</td></tr>
<tr><td>Z</td><td colspan="2">Бит нуля. Этот бит устанавливается, когда результат операции с байтом или словом равен «0» и очищается, если результат не равен «0».</td></tr>
<tr><td>C</td><td colspan="2">Бит переноса. Этот бит устанавливается, когда результат операции с байтом или словом имеет перенос и очищается, когда переноса нет.</td></tr>
</table>

Регистры генератора констант CG1 и CG2

Шесть обычно используемых констант генерируются с помощью регистров R2 и R3 генератора констант, что исключает необходимость использования дополнительного 16-разрядного слова в программном коде. Константы выбираются путем изменения режима адресации (As) регистра-источника.

Регистр	As	Константа	Комментарий
R2	00	- - - - -	Регистровый режим
R2	01	(0)	Режим абсолютной (безусловной) адресации
R2	10	00004h	+4, побитовая обработка
R2	11	00008h	+8, побитовая обработка
R3	00	00000h	0, обработка по словам
R3	01	00001h	+1
R3	10	00002h	+2, побитовая обработка
R3	11	0FFFFh	-1, обработка по словам

Генератор констант обладает следующими преимуществами:

- Не требуются особые команды
- Код не содержит дополнительного слова для шести констант
- Не требуется код (команда) доступа к памяти для получения константы

Ассемблер автоматически использует генератор констант, если одна из шести констант используется как непосредственный исходный операнд. При использовании регистров R2 и R3 в режиме генерации констант, адресация к ним не может быть явной – они действуют только как регистры-источники.

Генератор команд – расширенная система команд

Набор RISC-команд семейства MSP430 состоит только из 27 команд. Однако, генератор констант позволяет поддерживать MSP430-ассемблеру 24 дополнительные эмулированные команды. К примеру, команда с одним операндом:

CLR dst

эмулируется командой с двумя операндами такой же длины:

MOV R3,dst

где #0 замещается ассемблером, а R3 используется в режиме As=00

Команда INC dst замещается командой ADD 0(R3),dst

Регистры общего назначения R4-R15

Двенадцать регистров с R4 по R15 являются регистрами общего назначения. Все эти регистры могут быть использованы в качестве регистров данных, указателей адресов или индексных значений и доступны с помощью команд работы с байтами или словами, как показано на рис.3.7.

Режимы адресации

Семь режимов адресации для операнда источника и четыре режима адресации для операнда назначения могут адресовать полное адресное пространство без исключений. В таблице 3.3 приводится конфигурация битов для режимов As (источник) и Ad (назначение).

Таблица 3.3 Режимы адресации операндов источника/получателя

Режим адресации	Синтаксис	Описание
Регистровый режим	Rn	Содержимое регистра является операндом
Индексный режим	X(Rn)	Значение (Rn+X) указывает на операнд. X сохранен в следующем слове
Символьный режим	ADDR	Значение (PC+X) указывает на операнд. X сохранен в следующем слове. Использован индексный режим X(PC)
Абсолютный (безусловный) режим	&ADDR	Слово, следующее за командой, содержит абсолютный адрес. X сохранен в следующем слове. Использован индексный режим X(SR)
Косвенный регистровый режим	@Rn	Содержимое Rn использовано как указатель на операнд
Косвенный автоинкремент	@Rn+	Содержимое Rn использовано как указатель на операнд. Содержи-

		мое Rn впоследствии увеличивается на 1 для байтовых команд и на 2 для команд-слов.
Прямой (непосредственный) режим	#N	Слово, следующее за командой, содержит непосредственную константу N. Использован косвенный автоинкрементный режим @PC+

Семь упомянутых способов адресации подробно рассматриваются в следующих разделах. В большинстве примеров показаны схожие режимы адресации для источника и получателя, но в команде возможны любые правильные комбинации способов адресации источника и получателя.

Контроллер флэш-памяти

Флэш-память в MSP430 адресуется побитно, побайтно или пословно и может перепрограммироваться. Модуль флэш-памяти имеет интегрированный контроллер, управляющий процессом стирания и программирования. Контроллер имеет три регистра, тактовый генератор и генератор напряжения для обеспечения напряжений стирания и программирования.

Флэш-память в MSP430 обладает следующими возможностями:

- внутренний генератор напряжения для программирования;
- программирование битов, байтов или слов;
- работа при ультранизком потреблении мощности;
- стирание сегмента или массовое (полное) стирание.

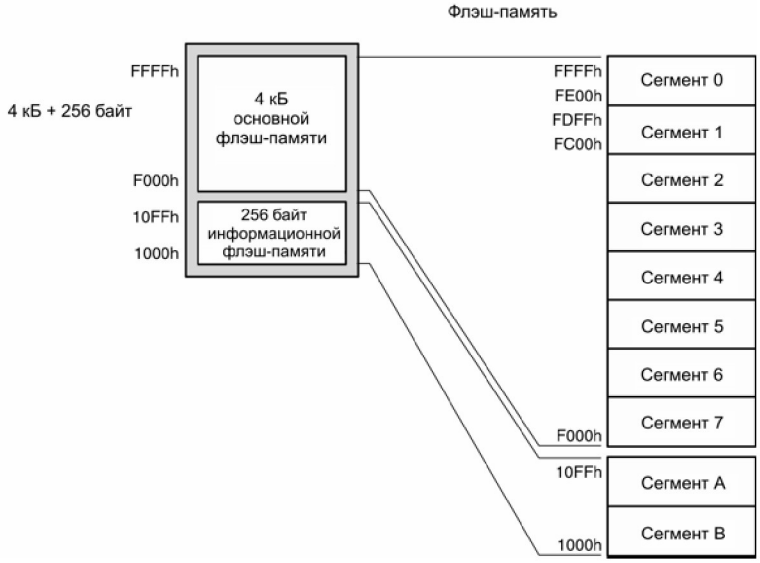
Сегментация флэш-памяти

Флэш-память в MSP430 разбита на сегменты. В неё может быть записан один бит, байт или слово, но сегмент – это минимальный размер флэш-памяти, который можно стереть. Три режима стирания позволяют стереть один сегмент, стереть все главные сегменты или стереть все сегменты (основные и информационные сегменты).

Флэш-память разделена на основной и информационный разделы памяти. Нет никаких различий в работе основного и информационного разделов памяти. Программный код или данные могут быть расположены в любом разделе. Различие между этими двумя разделами заключается в разных размерах сегмента и различных физических адресах.

Информационная память имеет два 128-байтных сегмента (в устройствах MSP430x1101 есть только один сегмент). Основная память имеет два или более 512-байтных сегмента. См. справочное руководство конкретного устройства для выяснения точной карты памяти.

На рисунке показана сегментация памяти на основе примера 4 кБ флэш-памяти, имеющей восемь основных сегментов и оба информационных сегмента.



Функционирование флэш-памяти

Режим по умолчанию для флэш-памяти – режим чтения. В этом режиме флэш-память не может быть стерта или записана, тактовый генератор и генератор напряжения выключены – память работает подобно ПЗУ.

Флэш-память MSP430 поддерживает внутрисистемное программирование (ISP) и не нуждается в использовании дополнительного внешнего напряжения. ЦПУ может программировать собственную флэш-память. Чтение или запись флэш-памяти во время программирования или стирания за-

прещены. Если требуется выполнение программы ЦПУ в течении записи или стирания, исполняемый код должен быть помещен в ОЗУ. Любое обновление флэш может инициироваться из флэш-памяти или ОЗУ.

Программирование устройств с флэш-памятью

Имеется три способа программирования флэш-устройств MSP430. Все способы поддерживают внутрисистемное программирование (ISP):

- Программирование через JTAG1
- Программирование через самозагрузчик
- Программирование через пользовательское решение

Программирование флэш-памяти через самозагрузчик (BSL)

Каждое MSP430 устройство с флэш-памятью содержит самозагрузчик BSL. Он позволяет пользователю читать или программировать флэш-память или ОЗУ с помощью последовательного интерфейса UART3. Доступ к флэш-памяти MSP430 через BSL защищен 256-разрядным паролем, определяемым пользователем.

Программирование флэш-памяти через пользовательское решение

Способность ЦПУ в MSP430 записывать собственную флэш-память позволяет реализовать внутрисистемное программирование внешними пользовательскими решениями. Пользователь может выбрать, каким образом данные будут поступать в MSP430 с использованием любого имеющегося доступного способа (UART, SPI и пр.). Разработанное пользователем программное обеспечение может получать данные и программировать флэш-память. Так как этот тип решения разработан пользователем, его можно настроить таким образом, чтобы наиболее полно удовлетворялись потребности в программировании, стирании и обновлении флэш-памяти.

Аппаратный умножитель является периферийным устройством и не является частью ЦПУ MSP430. Это означает, что его действия не пересекаются с действиями ЦПУ. Регистры умножителя – это периферийные регистры, которые загружаются и читаются командами ЦПУ.

Аппаратный умножитель поддерживает операции умножения без знака, умножения со знаком, умножения без знака с накоплением и умножение со знаком и накоплением. Тип операции выбирается адресом, в который записан первый операнд. Аппаратный умножитель имеет два 16-разрядных регистра OP1 и OP2 и три регистра результата RESLO, RESHI и SUMEXT. В регистре RESLO содержится младшее слово результата, в RESHI - старшее слово результата, а в регистре SUMEXT находится информация о результате. Результат может быть прочитан следующей командой после записи в OP2, кроме случая, когда используется косвенный режим адресации.

Регистры операндов

Регистр OP1 первого операнда имеет четыре адреса, показанные в таблице 7.1, используемые при выборе режима умножения. Запись первого операнда по желаемому адресу позволит выбрать тип операции умножения, но не приведет к началу выполнения какой-либо операции. Запись второго операнда в регистр OP2 второго операнда инициирует операцию умножения. Запись в OP2 стартует выбранную операцию над значениями, сохраненными в OP1 и OP2. Результат записывается в три регистра результата RESLO, RESHI и SUMEXT.

Повторение операций умножения может выполняться без перезагрузки OP1, если значение в OP1 используется для последовательных операций. Нет необходимости перезаписывать значение в OP1 для выполнения операций.

Адрес OP1	Имя регистра	Операция
0130h	MPY	Умножение без знака
0132h	MPYS	Умножение со знаком
0134h	MAC	Умножение без знака с накоплением
0136h	MACS	Умножение со знаком и накоплением

Регистры результата

Младший регистр результата RESLO содержит младшие 16 разрядов вычисленного результата. Содержимое старшего регистра результата RESHI зависит от операции умножения. Различные варианты содержимого RESHI приведены в таблице 7.2.

Таблица 7-2. Возможные варианты содержимого регистра RESHI

Режим	Содержимое RESHI
MPY	Старшие 16 разрядов результата
MPYS	В старшем бите MSB регистра находится знак результата. Оставшиеся биты содержат старшие 15 разрядов результата. Используется представление результата с дополнением до двух.
MAC	Старшие 16 разрядов результата
MACS	Старшие 16 разрядов результата. Используется представление результата с дополнением до двух.

Содержимое регистра расширенного суммирования SUMEXT зависит от выполненной операции умножения.

Режим	SUMEXT	
MPY	SUMEXT всегда содержит 0000h	
MPYS	SUMEXT	содержит расширенный знак результата
	0000h	результат был положительный
	0FFFFh	результат был отрицательный
MAC	SUMEXT	содержит перенос результата
	0000h	результат не содержит переноса
	0FFFFh	результат имеет перенос
MACS	SUMEXT	содержит расширенный знак результата
	0000h	результат был положительный
	0FFFFh	результат был отрицательный

Потеря значащих разрядов и переполнение в режиме MACS

Умножитель не может автоматически определить потерю значащих разрядов или переполнение в режиме MACS. Диапазон аккумулятора для положительных чисел равен 0 – 7FFF FFFFh, а для отрицательных чисел 0FFF FFFh – 8000 0000h. Переполнение происходит, когда результат суммирования двух отрицательных чисел выходит за диапазон для положительного числа. Потеря значащих разрядов происходит, когда результат сложения двух положительных чисел выходит за диапазон для отрицательного числа. В обоих случаях регистр SUMEXT содержит правильный знак результата: 0FFFFh при переполнении и 0000h при потере значащих разрядов. Программное обеспечение пользователя должно определить и соответствующим образом обработать эти состояния.

Цифровые входы/выходы

Устройства MSP430 имеют до 6 портов цифровых входов/выходов от P1 до P6. Каждый порт имеет 8 выводов входа/выхода. Каждый вывод индивидуально конфигурируется как вход или выход и каждая линия ввода/вывода может быть индивидуально считана или записана.

Порты P1 и P2 имеют возможность вызывать прерывание. Для каждой линии ввода/вывода портов P1 и P2 можно индивидуально разрешить прерывания и сконфигурировать их так, чтобы прерывание происходило по фронту или спаду входного сигнала. Все линии ввода/вывода порта P1 являются источником одного вектора прерывания, а все линии ввода/вывода порта P2 – источник другого вектора прерывания.

Регистры ввода PnIN

Каждый бит в каждом регистре PnIN отражает величину входного сигнала на соответствующей ножке ввода/вывода, когда она сконфигурирована на функцию ввода/вывода.

Регистры вывода PnOUT

Каждый бит в каждом регистре PnOUT содержит значение, которое будет выведено на соответствующую ножку ввода/вывода, сконфигурированную на функцию ввода/вывода и имеющую направление на вывод.

Регистры направления PnDIR

Каждый бит в каждом регистре PnDIR позволяет выбрать направление соответствующей ножки ввода/вывода, независимо от выбранной для этой ножки функции. Биты PnDIR для ножек

ввода/вывода, выбранные для других функций модуля должны быть установлены так, как это требуется для другой функции.

Бит = 0: Ножка порта переключается на ввод;

Бит = 1: Ножка порта переключается на вывод.

Регистры выбора функции PnSEL

Ножки порта часто мультиплексированы с другими функциями периферийных модулей. См. справочное руководство по конкретному устройству для выяснения возможных функций вывода. Каждый бит PnSEL определяет, как будет использована ножка – в качестве порта ввода/вывода или в качестве функции периферийного модуля.

Бит = 0: Для ножки выбирается функция ввода/вывода

Бит = 1: Для ножки выбирается функция периферийного модуля

Установка PnSEL=1 автоматически не определяет направление движения информации для ножки. Некоторые функции периферийных модулей требуют конфигурирования битов PnDIR для выбора направления, необходимого для правильной работы этой функции.

Прерывания P1 и P2

Каждый вывод портов P1 и P2 имеет возможность вызова прерывания, конфигурируемую регистрами PnIFG, PnIE и PnIES. Все выводы P1 – источник одного вектора прерывания, а все выводы P2 – источник другого одиночного вектора прерывания. Определить источник прерывания - P1 или P2 можно путем проверки регистра PnIFG.

Каждый бит PnIFG – это флаг прерывания соответствующей ножки ввода/вывода, устанавливаемый, когда происходит перепад выбранного входного сигнала на ножке. Все флаги прерывания PnIFG запрашивают прерывание, когда установлен их соответствующий бит PnIE и установлен бит GIE. Каждый флаг PnIFG должен быть сброшен программно. Программное обеспечение также может устанавливать каждый флаг PnIFG, обеспечивая возможность генерации программно-инициированного прерывания.

Бит = 0: Прерывание не ожидается

Бит = 1: Прерывание ожидается

Прерывания вызывают только перепады уровней, а не статические уровни. Если любой флаг PnIFG оказывается установленным во время выполнения процедуры обработки прерывания Px или устанавливается после команды RETI выполняемой процедуры обработки прерывания Px, установка флага PnIFGx генерирует другое прерывание. Таким образом, гарантируется, что каждый перепад уровня будет учтен.

Регистры выбора фронта прерывания P1IES, P2IES

Каждый бит PnIES позволяет выбрать, по какому фронту сигнала будет происходить прерывание для соответствующей ножки ввода/вывода.

Бит = 0: Флаг PnIFG устанавливается при изменении уровня сигнала с низкого на высокий;

Бит = 1: Флаг PnIFG устанавливается при изменении уровня сигнала с высокого на низкий.

Разрешение прерываний P1IE, P2IE

Каждый бит PnIE разрешает прерывание от соответствующего флага прерываний регистра PnIFG.

Бит = 0: Прерывание запрещено

Бит = 1: Прерывание разрешено

Конфигурирование неиспользуемых выводов порта

Неиспользуемые ножки ввода/вывода должны быть сконфигурированы на функцию ввода/вывода, в направлении вывода и оставаться неподключенными на печатной плате для уменьшения потребляемой мощности.

Таймеры

Таймеры A,B – это 16-разрядный таймеры/счетчики с тремя регистрами захвата/сравнения.

Таймер А обладает следующими возможностями:

- Асинхронный 16-разрядный таймер/счетчик с четырьмя режимами работы;
- Выбираемый и конфигурируемый источник тактирования;
- Три конфигурируемых регистра захвата/сравнения;
- Конфигурируемые выходы с возможностью ШИМ;
- Асинхронная фиксация (защелка) входа и выхода;

Таймер B обладает следующими возможностями:

- Асинхронный 16-разрядный таймер/счетчик с четырьмя режимами работы и четырьмя настраиваемыми длительностями
- Выбираемые и конфигурируемые источники тактирования
- Три или семь конфигурируемых регистров захвата/сравнения
- Конфигурируемые выходы с возможностью ШИМ
- Защелки сравнения с двойной буферизацией и синхронизируемой загрузкой

АЦП12

Модуль АЦП12 обеспечивает быстрые 12-разрядные аналого-цифровые преобразования. Модуль имеет 12-разрядное ядро SAR, схему выборки, опорный генератор и буфер преобразования и управления объемом 16 слов. Буфер преобразования и управления позволяет получать и сохранять до 16 независимых выборок АЦП без вмешательства ЦПУ. АЦП12 обладает следующими возможностями:

- Максимальная скорость преобразования свыше 200 ksps
- Монотонный 12-разрядный преобразователь без кодов ошибок
- Выборка и хранение с программируемыми периодами выборки, определяемыми программным обеспечением или таймерами
- Преобразование инициируется программным обеспечением, таймером A или таймером B
- Программно выбираемый интегрированный генератор опорного напряжения (1,5 В или 2,5 В)
- Программно выбираемый внутренний или внешний опорный источник
- Восемь индивидуально конфигурируемых внешних входных каналов
- Каналы преобразования для внутреннего температурного датчика, AVCC и внешних опорных источников
- Независимые опорные источники, задаваемые путем выбора канала, для обоих положительных и отрицательных опорных источников
- Выбираемый источник тактирования преобразований
- Одноканальный, повторный одноканальный, последовательный и повторно-последовательный режимы преобразования
- Ядро АЦП и опорное напряжение могут выключаться раздельно
- Регистр вектора прерываний для быстрого декодирования 18 прерываний АЦП
- 16 регистров хранения результата.

12-разрядное ядро АЦП

Ядро АЦП преобразует аналоговый входной сигнал в 12-разрядное цифровое представление и сохраняет результат в памяти преобразований. Ядро использует два программируемых/выбираемых уровня напряжения (VR+ и VR-) для задания верхнего и нижнего пределов преобразования. На цифровом выходе (NADC) представлена полная шкала (0FFFh), когда входной сигнал равен или выше VR+, и ноль, когда входной сигнал равен или ниже VR-. Входной канал и опорные уровни напряжения (VR+ и VR-) задаются в памяти управления преобразованиями. Формула преобразования для результата АЦП NADC выглядит следующим образом:

$$N_{ADC} = 4095 \times \frac{Vin - V_{R-}}{V_{R+} - V_{R-}}$$

Выбор тактирования преобразования

ADC12CLK используется как для тактирования преобразования, так и для генерации периода выборки, когда выбран импульсный режим выборки. Возможно использование следующих источников ADC12CLK: SMCLK, MCLK, ACLK и внутреннего осциллятора ADC12OSC.

ADC12OSC, генерируемый внутренне, лежит в диапазоне 5 МГц, но варьируется в зависимости от конкретного устройства, напряжения питания и температуры.

Пользователь должен гарантировать, что выбранный источник тактирования для ADC12CLK останется активным до конца преобразования. Если тактовые сигналы будут сняты во время преобразования, операция не будет завершена и любой результат будет неверным.

Входы АЦП12 и мультиплексор

Восемь внешних и четыре внутренних аналоговых сигнала выбираются как канал для преобразования аналоговым входным мультиплексором. Входной мультиплексор имеет тип break-before-make (разрыв перед включением), что уменьшает инжекцию шумов от канала к каналу, возникающую при переключении каналов. Входной мультиплексор также является Т-переключателем, минимизирующим взаимосвязь между каналами. Невыбранные каналы изолированы от АЦП, а промежуточный узел подключен к аналоговой земле (AVSS), поэтому паразитная емкость заземляется, что помогает устранять перекрестные помехи.

АЦП12 использует метод перераспределения заряда. Когда входы внутренне переключаются, переключение может привести к переходным процессам на входном сигнале. Эти переходные процессы затухают и устанавливаются до появления ошибочного преобразования.

Выбор аналогового порта

Входы АЦП12 мультиплексированы с ножками порта P6, имеющими цифровые КМОП ячейки. Когда аналоговые сигналы прикладываются к цифровым КМОП-схемам, может течь паразитный ток от VCC к GND. Этот паразитный ток появляется, если величина входного напряжения находится около переходного уровня ячейки. Отключение буфера ножки порта устраняет протекание паразитного тока и вследствие этого уменьшает общий потребляемый ток. Биты P6SELx дают возможность отключать входные буферы ножки порта.

Генератор опорного напряжения

Модуль АЦП12 содержит встроенный генератор опорного напряжения с двумя выбираемыми уровнями напряжения: 1,5 В и 2,5 В. Любое из этих опорных напряжений может быть использовано внутренне или внешне на выводе VREF+. Установкой REFON=1 включается внутренний опорный источник. Когда REF2_5V=1, внутреннее опорное напряжение равно 2,5 В, при REF2_5V=0 опорное напряжение равно 1,5 В. Если генератор опорного напряжения не используется, он может быть выключен для уменьшения потребления энергии.

Для правильной работы внутреннего генератора опорного напряжения необходимо использовать емкость временного хранения энергии, подключенную между VREF+ и AVSS. Рекомендуется в качестве такой емкости использовать комбинацию из включенных параллельно конденсаторов на 10 мкФ и 0,1 мкФ. После включения в течение максимум 17 мс необходимо дать возможность генератору опорного напряжения зарядить конденсаторы хранения энергии. Если внутренний опорный генератор не используется при преобразованиях, конденсаторы не требуются

Синхронизация выборки и преобразования

Аналого-цифровое преобразование инициируется по нарастающему фронту входного сигнала выборки SHI. Источник для SHI выбирается с помощью битов SHSx и может быть таким:

- Бит ADC12SC
- Модуль вывода 1 таймера A
- Модуль вывода 0 таймера B
- Модуль вывода 1 таймера B

Полярность источника сигнала SHI может быть инвертирована битом ISSH. Сигнал SAMPCON управляет периодом выборки и началом преобразования. Когда SAMPCON имеет высокий уровень, выборка активна. Переход сигнала SAMPCON с высокого уровня на низкий стартует аналого-цифровое преобразование, которому необходимо 13 циклов ADC12CLK. Два различных метода выборки-синхронизации задаются управляющим битом SHP, расширенный режим выборки и импульсный режим.

Расширенный режим выборки

Расширенный режим выборки выбирается, когда SHP=0. Сигнал SHI напрямую управляет SAMPCON и определяет длительность периода выборки tsample. Когда SAMPCON имеет высокий уровень, выборка активна. Переход сигнала SAMPCON с высокого уровня на низкий стартует преобразование после синхронизации с ADC12CLK.

Импульсный режим выборки

Импульсный режим выборки выбирается, когда SHP=1. Сигнал SHI используется для запуска таймера выборки. Биты SHT0x и SHT1x в ADC12CTL0 управляют интервалом таймера выборки, который задает период tsampe выборки SAMPCON. Таймер выборки оставляет высокий

уровень SAMPCON после синхронизации с ADC12CLK для запрограммированного интервала tsampe. Общее время выборки равно tsampe плюс tsync.

Биты SHTx устанавливают время выборки в 4 раза больше чем ADC12CLK. SHT0x устанавливает время выборки для ADC12MCTL0-7, а SHT1x устанавливает время выборки для ADC12MCTL8-15.

Память преобразований

Результаты преобразований сохраняются в 16-ти регистрах памяти преобразований ADC12MEMx. Каждый регистр ADC12MEMx конфигурируется соответствующим управляющим регистром ADC12MCTLx. Биты SREFx устанавливают опорное напряжение, а биты INCHx задают входной канал. Бит EOS определяет конец последовательности, когда используется последовательный режим преобразования. Следующие друг за другом преобразования последовательно сохраняются в регистрах с ADC12MEM15 по ADC12MEM0, когда бит EOS в ADC12MCTL15 не установлен.

Режимы преобразований АЦП12

АЦП12 имеет четыре режима работы, выбираемые битами CONSEQx.

CONSEQx	Режим	Операция
00	Одноканальный с одиночным преобразованием	Выполняется одно преобразование в одном канале.
01	Последовательность каналов	Выполняются однократные преобразования последовательности каналов.
10	Повторяющийся одноканальный	Выполняется повторяющееся преобразование в одном канале.
11	Повторяющаяся последовательность каналов	Выполняются повторяющиеся преобразования последовательности каналов.

Одноканальный режим с одиночным преобразованием

В одном канале однократно выполняется выборка и преобразование. Результат АЦП записывается в регистр ADC12MEMx, определенный битами CSTARTADDx. Если преобразования запускаются ADC12SC, поочередные преобразования могут быть запущены битом ADC12SC. Когда используется другой источник запуска, ENC должен переключаться между каждым преобразованием.

Режим последовательности каналов

В режиме последовательности каналов однократно выполняется выборка и преобразование. Результат АЦП записывается в память преобразований, начиная с ADCMEMx, определенным битами CSTARTADDx. Последовательность останавливается после измерения в канале с установленным битом EOS. Если последовательность запускает ADC12SC, поочередные последовательности могут запускаться битом ADC12SC. Когда используется другой источник запуска, ENC должен переключаться между каждой последовательностью

Повторяющийся одноканальный режим

В одном канале непрерывно выполняются выборка и преобразование. Результат АЦП записывается в ADC12MEMx, определенный битами CSTARTADDx. Необходимо считывать результат после завершения преобразования, потому что используется только один регистр памяти ADC12MEMx, перезаписываемый с каждым новым преобразованием.

Режим повторяющейся последовательности каналов

Непрерывно выполняются выборка и преобразование последовательности каналов. Результат АЦП записывается в память преобразований, начиная с ADC12MEMx, определенного битами CSTARTADDx. Последовательность останавливается после измерения в канале с установленным битом EOS и стартует снова по следующему сигналу запуска.

Останов преобразований

Прекращение активности АЦП12 зависит от режима работы. Рекомендуются следующие способы останова активного преобразования или последовательности преобразований:

- Сброс ENC в одноканальном режиме одиночного преобразования немедленно останавливает преобразование, при этом результат оказывается непредсказуемым. Для получения правильного результата необходимо опрашивать бит занятости до сброса перед очисткой ENC.

- Сброс ENC во время повторяющегося одноканального преобразования останавливает преобразователь в конце текущего преобразования.
- Сброс ENC во время последовательного или повторно-последовательного режимов останавливает преобразователь в конце последовательности.
- Любой режим преобразования может быть немедленно остановлен установкой CONSEQx=0 и сбросом бита ENC. Данные преобразования будут ненадежны

Использование АЦП12 с контроллером DMA

Устройства MSP430 с интегрированным контроллером DMA могут автоматически перемещать данные из любого регистра ADC12MEMx в другое место. DMA-переносы выполняются без вмешательства ЦПУ и независимо от любого выбранного режима пониженного энергопотребления. Контроллер DMA увеличивает пропускную способность модуля АЦП12 и расширяет круг приложений с пониженным энергопотреблением, позволяющих ЦПУ оставаться выключенным во время переноса данных.

Использование интегрированного температурного датчика

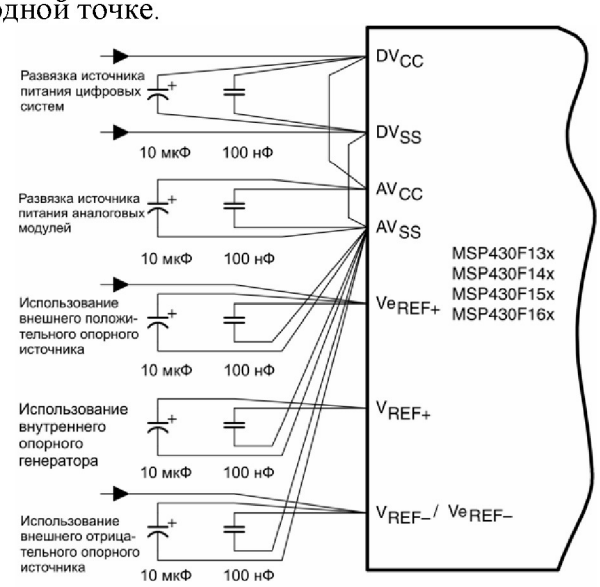
При использовании имеющегося на кристалле температурного датчика пользователь выбирает аналоговый входной канал INCHx=1010. Любая другая конфигурация рассматривается как выбор внешнего канала, включая выбор опорного источника, выбор памяти преобразований и т.д. Если используется температурный датчик, период выборки должен быть больше 30 мкс. Ошибка смещения температурного датчика может быть большой и может потребоваться калибровка для большинства приложений. При выборе температурного датчика автоматически запускается расположенный на кристалле опорный генератор в качестве источника напряжения для температурного датчика. Однако это не включает выход VREF+ и не влияет на выбор опорного источника для преобразования. Процедура выбора источника для преобразования информации с температурного датчика подобна процедуре выбора для любого другого канала.

Заземление АЦП12 и рассмотрение влияния помех

Как в любом АЦП с высоким разрешением, для устранения нежелательных паразитных эффектов и шумов, а также предотвращения возникновения паразитных контуров с замыканием на землю, необходимы особая разводка печатной платы и особые методы заземления.

Паразитные общие петли формируются, когда ток возврата от АЦП проходит совместно с токами других аналоговых и цифровых схем. Если не принимать специальных мер, этот ток может генерировать нежелательные напряжения смещения, которые могут прибавляться или вычитаться из опорного или входного напряжений аналого-цифрового преобразователя.

В дополнение к заземлению, пульсации и шумовые выбросы на линиях источника питания, вызванные переключениями цифровых схем или переключениями в источнике питания могут повредить результат преобразования. Для получения высокой точности рекомендуется создавать разработки, свободные от шумов, что достигается разделением аналоговых и цифровых контуров земли с соединением их в одной точке.



Система команд MSP430

Система команд семейства MSP430 содержит 24 основные команды и 27 эмулируемых.

Можно выделить команды с одним операндом, двумя операндами, и команды перехода. Все команды с операндами могут оперировать байтами или словами, при этом байтовые команды имеют окончание ".b", а команды, работающие со словами - ".w". По умолчанию предполагаются команды, работающие со словами.

MSP430 поддерживает семь режимов адресации источника и четыре режима адресации приемника, представленные ниже.

Режим адресации	Синтаксис	Тип	Описание
Регистровая	Rn	src/dst	Операнд - содержимое регистра
Индексная	X(Rn)	src/dst	X+Rn указывает на операнд, хранящийся в памяти
Относительная	ADDR	src/dst	На операнд указывает PC+X
Абсолютная	&ADDR	src/dst	Абсолютный адрес операнда содержится в команде
Косвенная регистровая	@Rn	src	Rn - указатель на операнд (содержит адрес операнда)
Косвенная автоинкрементная	@Rn+	src	Rn - указатель на операнд. После операции содержимое Rn увеличивается на 1, если команда байтовая и на 2, если команда оперирует со словами
Непосредственная	#N	src	Операнд содержится в команде

Обозначения, используемые при описании команд.

src - операнд - источник;
dst - операнд - приемник;
label - адресная метка.

Таблица 2 Мнемоники команд

Мнемоника		Описание		V	N	Z	C
ADC(.B)*	dst	Сложение C с dst	dst + C → dst	*	*	*	*
ADD(.B)	src,dst	Сложение src с dst	dst + src → dst	*	*	*	*
ADDC(.B)	src,dst	Сложение src с C и dst	dst + src +C→ dst	*	*	*	*
AND(.B)	src,dst	Логическое "И" src и dst	dst AND src→ dst	0	*	*	*
BIC(.B)	src,dst	Сброс бита	not src AND dst →dst	-	-	-	-
BIS(.B)	src,dst	Установка бита	not src AND dst →dst	-	-	-	-
BIT(.B)	src,dst	Проверка бита	src AND dst	0	*	*	*
BR*	dst	Безусловный переход	dst→PC	-	-	-	-
CALL*	dst	Вызов подпрограммы	PC+2→Stack, dst→PC	-	-	-	-
CLR(.B)*	dst	Сброс приемника	0→dst	-	-	-	-
CLRC*		Сброс C	0→C	-	-	-	0
CLRN*		Сброс N	0→N	-	0	-	-
CLRZ*		Сброс Z	0→Z	-	-	0	-
CMP(.B)	src,dst	Сравнение src и dst	dst-src	*	*	*	*
DADC(.B)*	dst	Дес. сложение C и dst	dst+C→dst	*	*	*	*
DADD(.B)	src,dst	Дес. сложение src и dst	src+dst+C→dst	*	*	*	*
DEC(.B)*	dst	Декремент приемника	dst-1→dst	*	*	*	*
DECD(.B)*	dst	Двойной декремент	dst-2→dst	*	*	*	*
DINT*		Запрет прерываний	0→GIE	-	-	-	-
EINT*		Разрешение прерываний	1→GIE	-	-	-	-

INC(.B)*	dst	Инкремент приемника	dst+1→dst	*	*	*	*
INCD(.B)*	dst	Двойной инкремент	dst+2→dst	*	*	*	*
INV(.B)*	dst	Инверсия приемника	not dst→dst	*	*	*	*
JC/JHE	label	Переход, если C=1		-	-	-	-
JEQ/JZ	label	Переход, если Z=1		-	-	-	-
JGE	label	Переход, если N^V=0		-	-	-	-
JL	label	Переход, если N^V=1		-	-	-	-
JMP	label	Безусловный переход	PC+2*offset→PC	-	-	-	-
JN	label	Переход, если N=1		-	-	-	-
JNC/JLO	label	Переход, если C=0		-	-	-	-
JNE/JNZ	label	Переход, если Z=0		-	-	-	-
MOV(.B)	src,dst	Пересылка	src→dst	-	-	-	-
NOP*		Нет операции		-	-	-	-
POP(.B)*	dst	Выталкивание из стека	@SP→dst, SP+2→SP	-	-	-	-
PUSH(.B)	src	Занесение в стек	SP-2→SP, src→@SP	-	-	-	-
RET*		Выход из подпрограммы	@SP→PC,SP+2→SP	-	-	-	-
RETI		Выход из прерывания	@SP→PC,SP+2→SP	-	-	-	-
RLA(.B)	dst	Арифм. сдвиг влево		*	*	*	*
RLC(.B)	dst	Цикл. сдвиг влево		*	*	*	*
RRA(.B)	dst	Арифм. сдвиг вправо		0	*	*	*
RRC(.B)	dst	Цикл. сдвиг вправо		*	*	*	*
SBC(.B)*	dst	Вычитание $\overline{C}$ из dst	dst+0FFFFh+C→dst	*	*	*	*
SETC*		Установка C	1→C	-	-	-	1
SETN*		Установка N	1→N	-	1	-	-
SETZ*		Установка Z	1→Z	-	-	1	-
SUB(.B)	src,dst	Вычитание src и C из dst	dst+not src+1→dst	*	*	*	*
SUBC(.B)	src,dst	Вычитание src и $\overline{C}$ из dst	dst+not src+C→dst	*	*	*	*
SWPB	dst	Перестановка байт		-	-	-	-
SXT	dst	Знаковое расширение	Bit7→Bit8…Bit15	0	*	*	*
TST(.B)*	dst	Проверка dst	dst+0FFFFh+1	0	*	*	1
XOR(.B)	src,dst	Искл. ИЛИ src и dst	src XOR dst→dst	*	*	*	*

* Эмулируемая команда

Условные переходы обеспечивают ветвление программы относительно программного счетчика PC и не оказывают влияния на биты статуса. Возможный диапазон переходов с помощью команды перехода составляет от -511 до +512 слов относительно текущего значения PC. 10-разряное смещение программного счетчика обрабатывается как 10-разрядное значение со знаком: удваивается и складывается с содержимым программного счетчика:

PCnew=PCold + 2 + PCoffset x 2

где: PCnew – новое содержимое программного счетчика;
PCold – исходное содержимое программного счетчика;
PCoffset – 10-разрядная величина смещения программного счетчика.

Интерфейс JTAG

Блок интерфейса JTAG позволяет управлять микроконтроллером извне, что необходимо в целях программирования и отладки. Контроллер может работать в режиме пропуска внутренних сигналов (bypass mode) и в режиме внешнего управления, позволяющем программировать память контроллера и отлаживать программы. После отладки программного обеспечения можно в целях запрещения внешнего доступа к памяти контроллера пережечь перемычку, соединяющую контроллер с внешними выводами. После этого работа с JTAG интерфейсом будет недоступна.

Pin	Direction	Usage
TMS	IN	Signal to control the JTAG state machine
TCK	IN	JTAG clock input
TDI	IN	JTAG data input / TCLK input
TDO	OUT	JTAG data output
TEST	IN	Enable JTAG pins (20- and 28-pin devices only)

Данные сигналы используются для работы интерфейса JTAG. Реализация JTAG для MSP430 позволяет:

- Блокировать и инициировать работу процессора;
- Имитировать системный сброс;
- Загружать счетчик команд процессора;
- Программировать и читать память контроллера;
- Программировать и читать биты конфигурации контроллера;
- Передавать команды процессору в отладочных целях;

Пример программы на ассемблере

```
;*******************************************************************************
Иллюстрирует работу с АЦП в режиме повторяющаяся последовательность каналов.
Источником опорного напряжения служит AVCC, сигналы подключены к каналам A0 - A3, результаты сохраняются в ADC12MEM0 - 3. После каждого преобразования данные перемещаются в память (канал A0 - по адресу 0x200-0x20E, A1 - 0x210-0x21E и т.д.
;*******************************************************************************
#include        "msp430x14x.h" // Standard Equations

A0result    EQU    0200h            ; Channel A0 results
A1result    EQU    0210H            ; Channel A1 results
A2result    EQU    0220H            ; Channel A2 results
A3result    EQU    0230H            ; Channel A3 results
                          ;
;-------------------------------------------------------------------------
main       ORG    01100h            ; Program Start
;-------------------------------------------------------------------------
RESET          mov    #0A00h,SP            ; Initialize stackpointer
StopWDT        mov    #WDTPW+WDTHOLD,&WDTCTL  ; Stop watchdog
               bis.b  #BIT0+BIT1+BIT2+BIT3,&P6SEL ; Enable A/D inputs
SetupADC12     mov    #SHT0_8+MSC+ADC12ON,&ADC12CTL0
                          ; Turn on ADC12, use int. osc.
                          ; extend sampling time so won't
                          ; get overflow
                          ; Set MSC so conversions triggered
                          ; automatically
               mov    #SHP+CONSEQ_3,&ADC12CTL1
                          ; Use sampling timer, set mode
               mov    #BIT3,&ADC12IE        ; Enable ADC12IFG.3 for ADC12MEM3
               mov.b  #INCH_0,ADC12MCTL0     ; A0 goes to MEM0
               mov.b  #INCH_1,ADC12MCTL1     ; A1 goes to MEM1
               mov.b  #INCH_2,ADC12MCTL2     ; A2 goes to MEM2
               mov.b  #EOS+INCH_3,ADC12MCTL3  ; A3 goes to MEM3, end of sequence
                          ;
               clr    R5                ; Clear pointer
               eint                     ; Enable interrupts
```

```
                        ;
Mainloop    bis    #ENC,&ADC12CTL0       ; Enable conversions
            bis    #ADC12SC,&ADC12CTL0   ; Start conversions
            bis    #CPUOFF,SR            ; Hold in LPM0
;-------------------------------------------------------------------------
ADC12ISR    ; Interrupt Service Routine for ADC12
;-------------------------------------------------------------------------
            mov    &ADC12MEM0,A0result(R5) ; Move results to RAM
            mov    &ADC12MEM1,A1result(R5) ; Move results to RAM
            mov    &ADC12MEM2,A2result(R5) ; Move results to RAM
            mov    &ADC12MEM3,A3result(R5) ; Move results to RAM, IFG reset
            incd   R5                ; Increment results table pointer
            and    #0Eh,R5           ; Modulo pointer
            reti
;-------------------------------------------------------------------------
;           Interrupt Vectors
;-------------------------------------------------------------------------
            ORG    0FFFEh            ; MSP430 RESET Vector
            DW     RESET             ;
            ORG    0FFEEh            ; ADC12 Interrupt Vector
            DW     ADC12ISR          ;
            END
```

Эта же программа на языке С

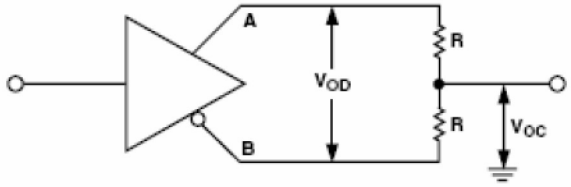
```
#include        "msp430x14x.h" // Standard Equations
#define  Num_of_Results  8

static unsigned int A0results[Num_of_Results];  // These need to be global in
static unsigned int A1results[Num_of_Results];  // this example. Otherwise, the
static unsigned int A2results[Num_of_Results];  // compiler removes them because
static unsigned int A3results[Num_of_Results];  // they are not used

void main(void)
{
  WDTCTL = WDTPW+WDTHOLD;                 // Stop watchdog timer
  P6SEL = 0x0F;                           // Enable A/D channel inputs
  ADC12CTL0 = ADC12ON+MSC+SHT0_8;         // Turn on ADC12, extend sampling time
                                          // to avoid overflow of results
  ADC12CTL1 = SHP+CONSEQ_3;               // Use sampling timer, repeated sequence
  ADC12MCTL0 = INCH_0;                    // ref+=AVcc, channel = A0
  ADC12MCTL1 = INCH_1;                    // ref+=AVcc, channel = A1
  ADC12MCTL2 = INCH_2;                    // ref+=AVcc, channel = A2
  ADC12MCTL3 = INCH_3+EOS;                // ref+=AVcc, channel = A3, end seq.
  ADC12IE = 0x08;                         // Enable ADC12IFG.3
  ADC12CTL0 |= ENC;                       // Enable conversions
  _EINT();                                // Enable interrupts
  ADC12CTL0 |= ADC12SC;                   // Start conversion
  _BIS_SR(LPM0_bits);                     // Enter and stay in LPM0
}

#pragma vector=ADC_VECTOR
__interrupt void ADC12ISR (void)
{
  static unsigned int index = 0;
```

```
  A0results[index] = ADC12MEM0;        // Move A0 results, IFG is cleared
  A1results[index] = ADC12MEM1;        // Move A1 results, IFG is cleared
  A2results[index] = ADC12MEM2;        // Move A2 results, IFG is cleared
  A3results[index] = ADC12MEM3;        // Move A3 results, IFG is cleared
  index = (index+1)%Num_of_Results;    // Increment results index, modulo
}
```

Интерфейс RS-485

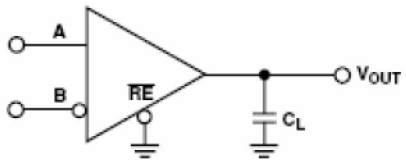
Интерфейс RS-485 – один из наиболее простых и распространенных интерфейсов, применяемых для последовательной передачи данных в промышленных системах управления и сбора данных. Интерфейс RS-485 основан на стандарте EIA RS-422/RS-485.

Сигналы интерфейса RS-485 передаются дифференциальными перепадами напряжения величиной (0,2…8)В, что обеспечивает высокую помехоустойчивость и общую длину линии связи до 1 км (и более с использованием специальных устройств – повторителей). Кроме того, интерфейс RS-485 позволяет создавать сети путем параллельного подключения многих устройств к одной физической линии (так называемая “мультиплексная шина”). Для формирования сигналов RS-485 используются специальные микросхемы – драйверы, выпускаемые многими фирмами (например, ADM485 фирмы Analog Devices, MAX485 – фирмы Maxim Integrated Products и др.)

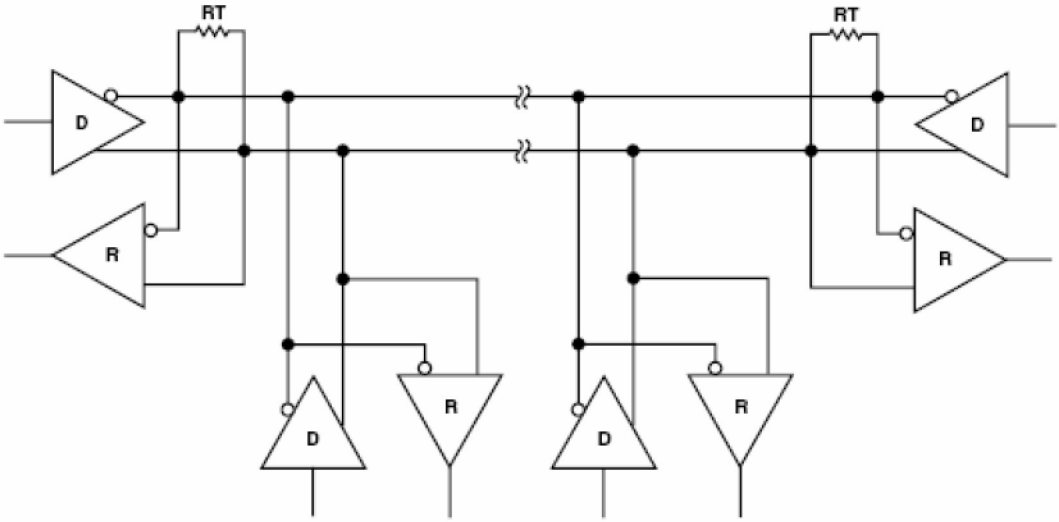


Формирователь уровней RS-485 (передатчик) представляет собой усилитель с дифференциальным выходом, характеризующийся дифференциальным и синфазным напряжением. Величина дифференциального напряжения зависит от нагрузки и обычно составляет от двух до пяти вольт, величина синфазного напряжения, как правило, около трех вольт.

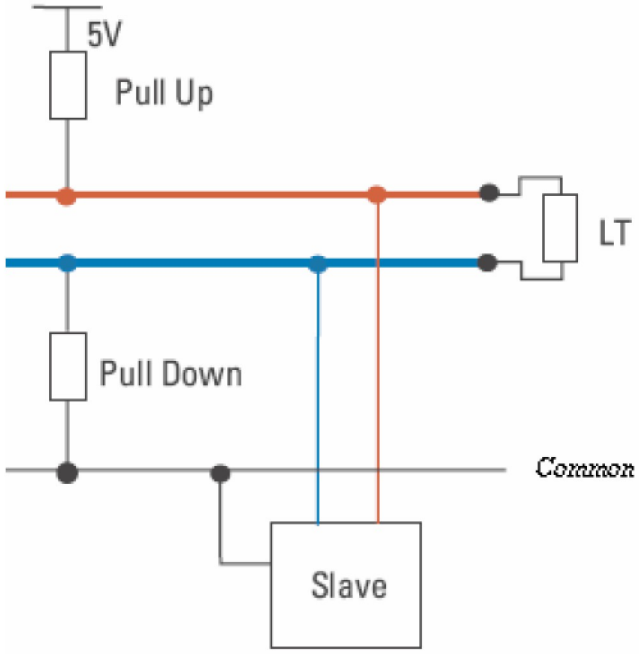
Приемник представляет собой усилитель с дифференциальным входом и однофазным выходом, формирующий сигнал логического уровня.



Линия RS-485 обычно состоит из двух сигнальных проводов (A и B) и общего провода. Наличие общего провода, строго говоря, не является обязательным, однако физические приемники должны иметь близкие потенциалы общего провода, для чего и необходим третий провод. При необходимости может быть применена гальваническая развязка с помощью оптронов.



Для работы с несколькими источниками и приемниками сигнала, как правило, используют конфигурацию «общая шина». Для согласования на концах линии резисторы (терминаторы) с сопротивлением, равным волновому сопротивлению кабеля. Для уменьшения помех, наводимых на линию внешними магнитными полями в качестве кабеля обычно применяют витую пару, при необходимости возможно дополнительное внешнее экранирование.



В такой конфигурации одно устройство является передатчиком, остальные приемниками. Для управления направлением передачи в микросхемах драйверах предусматриваются специальные управляющие сигналы. Если все передатчики выключены и к линии подключены только приемники, уровни сигнала могут быть неопределенными, приводя к генерации помеховых сигналов на выходах приемников. Для исключения этого явления могут предусматриваться резисторные делители, фиксирующие разность потенциалов между линиями на уровне не менее 0.2В.

При организации передачи/приема данных в сетях с несколькими передатчиками/приемниками необходимо обеспечить арбитраж и адресацию узлов. Обычно сети на основе RS-485 выполняются с использование модулей UART микроконтроллеров. Для организации адресации в этом случае могут быть применены различные методы, но наибольшую популярность имеют системы с использованием девятого бита (в восьмибитных посылках) и протокола Modbus.

Организация сети с использованием девятого бита (бита четности)

Большинство UART микроконтроллеров поддерживают девятибитную передачу, в которой помимо восьми бит данных передается девятый (бит четности). Основное назначение этого бита – контроль правильности передачи байта (четности/нечетности числа единичных битов в посылке), однако часто его используют для указания приемнику на тип передаваемой информации (адрес/данные). В этом случае каждому узлу присваивается уникальный адрес и передатчик, желающий передать информацию одному из приемников, вначале передает адрес этого узла в посылке с установленным (сброшенным) битом четности. Узел, принявший адресную посылку, совпадающую с собственным адресом, принимает далее байты со сброшенным (установленным) битом четности, указывающим, что передаваемая информация – это данные, а не адрес. Многие микроконтроллеры имеют встроенный аппаратный механизм анализа девятого бита, позволяющий генерировать прерывание при совпадении принятого адреса со своим. Простота такой системы обуславливает привлекательность в простых системах сбора данных, однако, пропускная способность несколько снижается за счет передачи бита четности. Как правило, такие протоколы строятся по схеме с одним ведущим узлом (мастером) и остальными узлами в пассивной роли (помощников). Наличие нескольких мастеров требует реализации процедуры арбитража, усложняющей протокол.

Протокол Modbus

Протокол Modbus может быть реализован при различных типах физической организации сети (не обязательно на базе RS-485), однако, его реализация на базе RS-485 является одной из наиболее популярных в промышленных системах. Modbus предполагает наличие в сети до 247

узлов. Существует достаточно много «диалектов» Modbus, однако наиболее популярны три разновидности: для передачи по проводным, беспроводным или оптическим каналам – Modbus RTU и Modbus ASCII, а также для передачи по сетям Ethernet поверх TCP/IP – Modbus TCP. Передача обычно ведется стандартными восьмибитными посылками, объединенными в пакеты. Не углубляясь в вопросы применимости тех или иных стандартов, рассмотрим одну из основных реализаций Modbus-RTU, которая, впрочем, также не является особенно жесткой и может несколько варьироваться в различных ситуациях.

Каждому злу присваивается адрес от 1 до 247, адрес 0 соответствует широковещательной передаче, адреса в диапазоне 248…255 зарезервированы. Информация передается пакетами, состоящими из отдельных посылок с восемью информационными битами, старт-битом и 1-2 стоп битами. Возможно использование бита контроля четности для контроля передачи отдельных посылок. Пакет имеет следующий формат:

старт	адрес	функция	данные	CRC	конец
T1-T2-T3-T4	8 бит	8 бит	n x бит	16 бит	T1-T2-T3-T4

Поля «старт» и «конец» представляют собой поля «тишины», необходимые для разделения отдельных пакетов. Основная их функция – реализация арбитража в системах с несколькими ведущими узлами, имеющими право независимо начинать передачу. Поле адреса содержит адрес устройства. Поле функции содержит функциональный код – число в диапазоне от 1 до 255, описывающий смысловую нагрузку пакета. В большинстве реализаций узлы представляются программисту в виде набора регистров, в которые можно что-то записывать и откуда можно что-то считывать. В такой модели используются функциональные коды 03/04 – «читать регистр», код 06 – «записать данные в отдельный регистр», код 10 – «записать группу регистров». Большинство кодов уже стандартизовано и используется в стандартных реализациях Modbus. Устройство, которое обращается к другому устройству, посылает пакет с адресом и функциональным кодом, получая в ответ пакет с адресом «ответчика», тем же функциональным кодом и данными. Разработчикам не рекомендуется использовать стандартные общепринятые функциональные коды для своих целей, для этого резервируются специальные пространства 65-72 и 100-110. Пример такого обмена пакетами при чтении группы регистров приведен ниже.

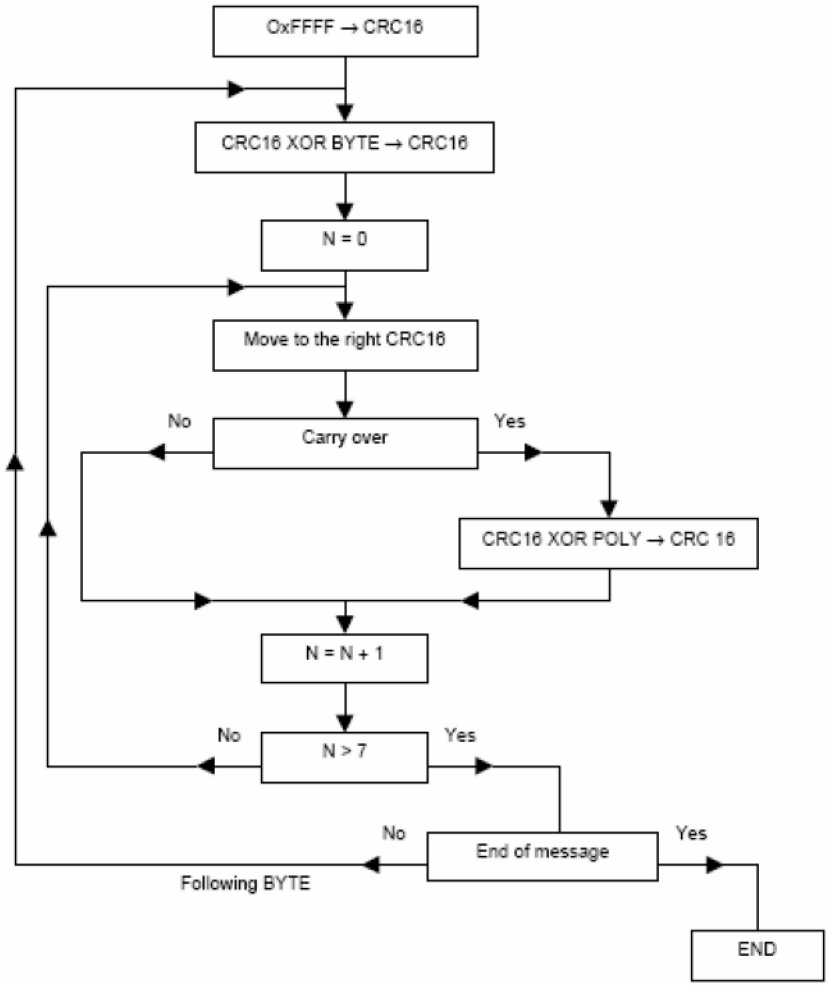
Запрос

Имя поля	Значение (Hex)
Адрес подчиненного	11
Функция	03
Начальный адрес ст.	00
Начальный адрес мл.	6B
Кол-во регистров ст.	00
Кол-во регистров мл.	03
Контрольная сумма	--

Ответ

Имя поля	Значение (Hex)
Адрес подчиненного	11
Функция	03
Счетчик байт	06
Данные (регистр 40108) ст.	02
Данные (регистр 40108) мл.	2B
Данные (регистр 40109) ст.	00
Данные (регистр 40109) мл.	00
Данные (регистр 40110) ст.	00
Данные (регистр 40110) мл.	64
Контрольная сумма	--

Поле CRC (Cyclic Redundancy Code или Циклический Избыточный Код) содержит специальным образом вычисленную контрольную сумму и предназначено для контроля правильности передачи/приема пакета. Каждое устройство, принимающее пакет, вычисляет CRC по стандартному алгоритму и сравнивает получившееся значение с полем CRC. При несовпадении фиксируется состояние ошибки. Полином CRC16, используемый в реализациях протокола Modbus позволяет эффективно обнаруживать ошибки в пакете данных. Вычисление контрольной суммы производится побитовой обработкой сообщения или несколько более быстрой табличной обработкой. При реализации алгоритма на микроконтроллерах разработчику приходится делать выбор между скоростью вычислений и объемом памяти, необходимым для реализации алгоритма.



Обработка ошибок

При передаче данных возможны ошибки, различают два типа ошибок: ошибки связи и оперативные или программные ошибки. При возникновении ошибок связи (искажение битов или нарушение в длине пакета) устройство, получившее такое сообщение, игнорирует его. Такая ситуация будет иметь место и при отсутствии с сети устройства с нужным адресом или передаче неправильного адреса. В этих случаях обычно передатчик повторяет сообщение, оставшееся без ответа. Сообщение, принятое правильно (нужной длины и с правильной CRC) без ответа оставаться не должно. Если же сообщение принято без ошибок, но некорректно по содержанию, устройство, к которому оно обращено, отвечает сообщением об ошибке. Сообщение об ошибке содержит в поле функционального кода значение исходного функционального кода с установленным в единицу старшим битом. Например, если запрос имел функциональный код 01, ответный код ошибки будет иметь значение 81h. Далее следует код исключительной ситуации из нижеприведенной таблицы и контрольная сумма. Приведенные коды исключительных ситуаций достаточны в большинстве случаев, но иногда разработчики расширяют этот список для особых ситуаций.

Код	Название	Смысл
01	ILLEGAL FUNCTION	Функция в принятом сообщении не поддерживается данным узлом.
02	ILLEGAL DATA ADDRESS	Адрес, указанный в поле данных, является недопустимым для данного узла.
03	ILLEGAL DATA VALUE	Значения данных недопустимы для данного узла.
04	FAILURE IN ASSOCIATED DEVICE	Узел не может ответить на запрос или произошла авария.

Код	Название	Смысл
05	ACKNOWLEDGE	Узел принял запрос и начал выполнять долговременную операцию программирования. Для определения момента завершения операции используется запрос типа POLL PROGRAM COMPLETE. Если этот запрос был послан до завершения операции программирования, то узел ответит сообщением REJECTED MESSAGE.
06	BUSY, REJECTED MESSAGE	Сообщение было принято без ошибок, но узел в данный момент выполняет долговременную операцию программирования. Запрос необходимо ретранслировать позднее.
07	NAK-NEGATIVE ACKNOWLEDGMENT	Функция программирования не может быть выполнена.

Интерфейс CAN

CAN (Controller Area Network) - это последовательный протокол связи с эффективной поддержкой распределения контроля в реальном времени и очень высоким уровнем безопасности. Основное назначение: организация передачи информации в сложных условиях, таких как среды с высоким уровнем различного рода помех. Этот протокол передачи применяется в автомобильной электронике, машинных устройствах управления, датчиках при передаче информации со скоростями до 1 Мбит/сек.

Протокол CAN можно разделить на следующие уровни:

- объектный уровень
- канальный уровень
- физический уровень

Объектный и канальный уровни включают весь сервис и функции передачи данных определяемых ISO/OSI моделью. Область объектного уровня включает:

- Поиск сообщений для передачи.
- Фильтрация сообщений, полученных от канального уровня
- Обеспечение связи между прикладным уровнем и аппаратными средствами.

Объектный уровень можно реализовывать различными способами.

Область канального уровня главным образом - протокол передачи, т.е. управление кадрами, выполнение арбитража, проверка и сигнализация ошибок, типизация ошибок. Внутри канального уровня решается, является ли шина свободной для начала новой передачи. Все что находится внутри канального уровня, не имеет ни какой свободы к модификации.

Область физического уровня - фактическая передача бит между различными узлами. Внутри одной сети физический уровень должен быть одинаков для всех узлов. Физический уровень можно реализовать различными способами.

На базе CAN стандартизованы протоколы ISO11898 (для высокоскоростных приложений), ISO11519 (для низкоскоростных приложений) и J1939 (для грузовиков и автобусов). Все эти протоколы используют в качестве физического носителя дифференциальную электрическую линию передачи. CAN относится к семейству протоколов CSMA/CD Carrier Sense Multiple Access with Collision Detection (прослушивание несущей с разрешением коллизий). Это означает, что передатчик может начинать передачу, только если линия свободна в течение определенного интервала времени, а если два и более узлов начинают передачу одновременно, то они должны обнаружить конфликт и предпринять соответствующие действия. Протокол CAN предусматривает недеструктивный арбитраж, что означает, что обнаружение коллизии не ведет к повреждению передаваемого сообщения.

Основные особенности

Сообщения

Информация по шине посылается в фиксированном формате сообщений различной, но фиксированной длины. Когда шина свободна, любой узел может начать передачу нового сообщения.

Уровни шины

Шина может принимать одно из дополняющих друг друга значений: "dominant" и "recessive". В случае одновременной подачи "dominant" бита и "recessive" бита, возникающее в результате значение шины будет "dominant". Для электрического носителя обычно "recessive" соответствует "1", а "dominant" - "0".

Синхронизация

CAN использует для синхронизации "bit stuffing" (вставка бита), при которой за пятью последовательными битами одной полярности вставляется бит противоположной полярности. Этот бит используется приемниками сообщений для синхронизации, но игнорируется, как носитель информации.

Информационная маршрутизация

В CAN нет никакой информации относительно конфигурации сети (например, адреса узла). Это имеет несколько важных следствий:

Гибкость системы:

Узел может быть добавлен в CAN - сеть, без каких либо изменений в программном или аппаратном обеспечении, какого - либо узла в сети.

Маршрутизация сообщений:

Содержание сообщения определяется идентификатором. Идентификатор не указывает адреса сообщения, а описывает значение данных так, чтобы все узлы сети были способны решить фильтрацией сообщений, нужны им эти данные или нет.

Передача группе:

Как следует из фильтрации сообщений, любое число узлов может одновременно получать и реагировать на одно и тоже сообщение.

Скорость передачи информации

Скорость передачи информации в CAN - сети может быть различной для каждой сети. Однако в каждой конкретной сети скорость передачи информации фиксирована.

Приоритеты

Идентификатор и RTR - бит определяют статический приоритет сообщения в течение доступа к шине.

Удаленный запрос данных

Посылая кадр удаленного запроса данных, узел может потребовать данные от другого узла. Кадр данных и кадр удаленного запроса данных должны иметь одинаковый идентификатор.

Multimaster

Когда шина свободна, любой узел может начать передачу сообщения. Доступ к шине получает узел, передающий кадр с наивысшим приоритетом.

Арбитраж

Когда шина свободна, любой узел может начать передачу сообщения. Если два или больше узла начинают передавать сообщения в одно и тоже время, конфликт при доступе к шине будет решен поразрядным арбитражем используя идентификатор и RTR - бит. Механизм арбитража гарантирует, что ни время, ни информация не будут потеряны. Если кадр данных и кадр удаленного запроса данных начинают передаваться в одно время, то кадр данных имеет более высокий приоритет, чем кадр удаленного запроса данных. В течение арбитража каждый передатчик сравнивает уровень переданного бита с уровнем, считываемым с шины. Если эти уровни одинаковы, узел может продолжать посылать данные дальше. Если был послан уровень лог. '1' (recessive), а с шины считан уровень лог. '0' (dominant), то узел теряет право дальнейшей передачи данных и должен прекратить посылку данных на шину.

Соединения

Линия связи по протоколу CAN - это шина, к которой может быть подключён ряд узлов. Количество узлов не имеет никакого теоретического предела. Фактически количество узлов будет ограничено временами задержек и/или электрической нагрузкой на линии шины. Способ, которым выполнена шина, не установлен в данной спецификации. Например, это может быть одиночный провод (+земля), два дифференциальных провода, оптическое стекловолокно.

Подтверждение

Все приёмники проверяют непротиворечивость принимаемого сообщения и подтверждают непротиворечивое сообщение.

Обработка ошибок

Узлы в сети, объединенной интерфейсом CAN, могут обнаруживать ошибки и реагировать на них несколькими способами. При кратковременных сбоях работоспособность восстанавливается автоматически, при серьезных ошибках узел может отключиться от сети, таким образом, изолируя источник ошибки. CAN определяет пять видов возможных ошибок и три состояния, в которых могут находиться узлы, в зависимости от серьезности и количества ошибок.

Типы ошибок

Ошибка CRC

Если значение, указанное в поле CRC не совпадает со значением, вычисленным узлом, то генерируется кадр ошибки. Если хотя бы одно устройство обнаружило такую ошибку, то сообщение повторяется через определенный интервал времени.

Ошибка подтверждения

После передачи сообщения передатчик проверяет значение бита в поле подтверждения. Если значение этого бита - "recessive", то это значит, что ни один узел не подтвердил получения сообщения. При этом генерируется кадр ошибки, и сообщение повторяется через определенный интервал времени.

Ошибка кадра

При получении доминантного бита в некоторых определенных служебных полях сообщения (конец кадра, межкадровый интервал, разделители CRC и поля подтверждения), констатируется состояние ошибки, и сообщение повторяется через определенный интервал времени.

Битовая ошибка

Фиксируется передатчиком при несовпадении передаваемого бита и состояния шины. Если при передаче поля арбитража или поля подтверждения фиксируется доминантный уровень, то состояние ошибки не фиксируется (нормальное состояние при арбитраже или подтверждении приема), в других случаях генерируется кадр ошибки и сообщение поторяется.

Ошибка синхронизации

Если при передаче данных приемник обнаруживает шесть и более последовательных бит одной полярности, то фиксируется состояние ошибки и сообщение повторяется.

Состояния узлов

Активный узел (ERROR ACTIVE)

Узел в этом состоянии активно работает на шине, обменивается сообщениями, генерирует кадры ошибок с активным флагом, вынуждая другие узлы также отвечать на них. Это состояние является нормальным состоянием узла и требует, чтобы значения счетчиков ошибок передачи и приема не превышали 128.

Пассивный узел (ERROR PASSIVE)

Узел становится пассивным, когда значения счетчиков превышают 128. Такому узлу не разрешается передавать активный флаг ошибки, он должен передавать пассивный флаг ошибки. Когда такой узел является единственным передатчиком на шине, то его флаг нарушает синхронизацию и другие устройства передают ответные кадры ошибок, если он является приемником или имеются другие передатчики, то его флаг не влияет на состояние шины.

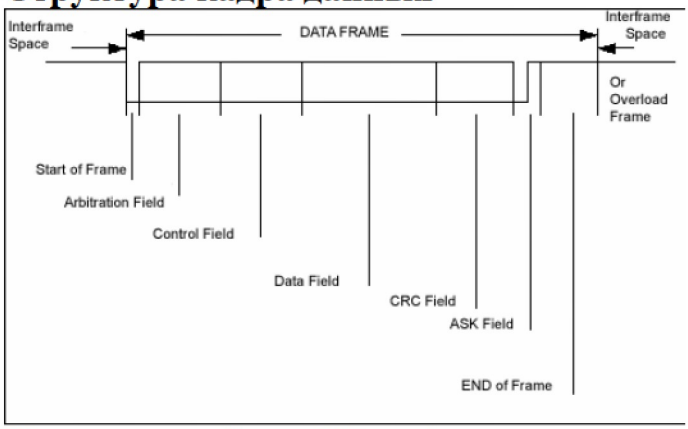
Отключенный узел

Узел переходит в это состояние, когда значение счетчика ошибок передачи превышает 255. В этом состоянии узел не може передавать и принимать сообщения и генерировать какие-либо кадры ошибок. Протокол CAN предусматривает специальную последовательность восстановления шины, когда такой узел возвращается в активное состояние после устранения проблемы.

Формат сообщений

При передаче информации с помощью протокола CAN используется четыре типа кадров. Кадр данных содержит данные, передаваемые передатчиком приёмнику (ам). Кадр удаленного запроса данных передается на шину для запроса передачи кадра данных с тем же самым идентификатором. Кадр ошибки передаётся при обнаружении ошибки на шине. Кадр перегрузки используется для обеспечения дополнительной задержки между предшествующим и последующим кадрами данных или кадрами удаленного запроса данных. Кадры данных и кадры удаленного запроса данных отделяются от предшествующих кадров межкадровым пространством.

Структура кадра данных



Кадр данных (DATA FRAME)

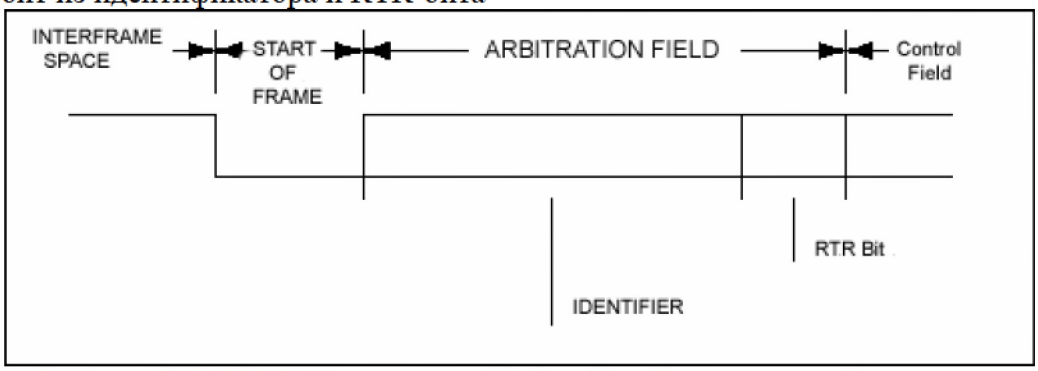
Кадр данных состоит из 7 различных полей: "Начало кадра", "поле арбитража", "поле контроля", "поле данных", "поле CRC", "поле подтверждения", "конец кадра".

Начало кадра (Start of Frame)

Отмечает начало кадра данных или кадра удаленного запроса данных. Состоит из бита с лог. '0'. Узлу разрешено начинать передачу только при свободной шине (см. простой шины). Все узлы должны быть синхронизированы по началу фронта, вызванного полем "начало кадра" (см. аппаратная синхронизация) узла, начавшего работу первым.

Поле арбитража (Arbitration Field)

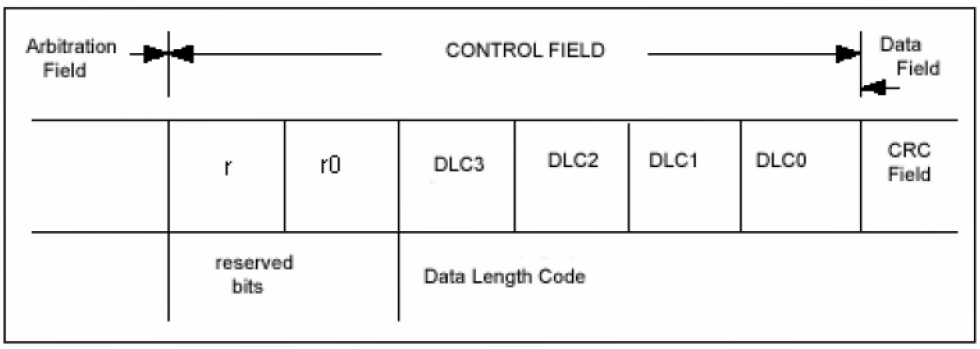
Состоит из идентификатора и RTR-бита



Идентификатор (Identifier) имеет длину 11 бит. Эти биты должны быть переданы в порядке от ID10 до ID4. Самый старший бит ID0. 7 старших битов не должны быть все битами с лог '1'. Расширенный формат предполагает 29 бит в этом поле (базовый идентификатор и расширение). RTR-бит (Бит запроса передачи) в кадре данных - "0", в кадре удаленного запроса данных - "1". Для совместимости расширенного и стандартного форматов 12-й бит идентификатора расширенного формата является рецессивным.

Поле контроля (CONTROL FIELD)

Включает 6 бит. Это - код длины данных (4бита) и 2 бита зарезервированные под будущие расширения. Зарезервированные биты должны быть "0".



Код длины данных (DLC)

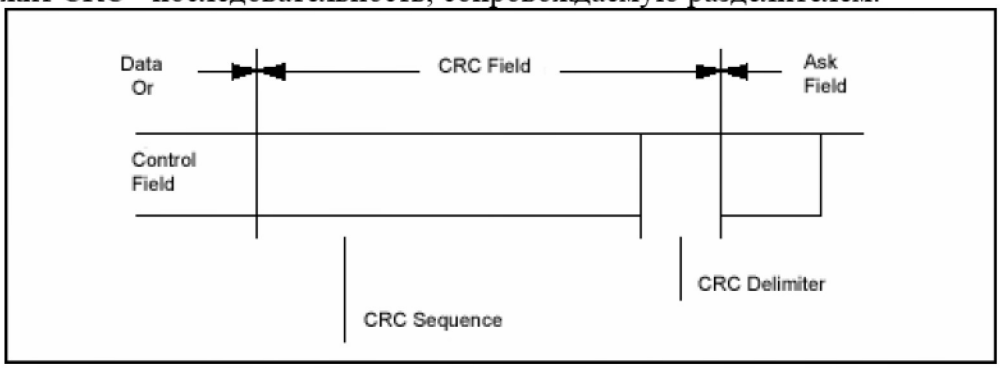
Показывает количество байт в поле данных. Код длины данных имеет размер 4 бита и передаётся внутри контрольного поля. Допустимые значения: 0.....8. Другие значения использоваться не могут.

Поле данных (DATA FIELD)

Включает данные, передаваемые внутри кадра данных. Оно может содержать от 0 до 8 байт, каждый из которых содержит 8 бит.

CRC поле (CRC FIELD)

Содержит CRC - последовательность, сопровождаемую разделителем.



CRC-последовательность (CRC Sequence):

Для вычисления CRC полинома, полином, коэффициенты которого задаются потоком, состоящим из значений, бит полей: "начало кадра ", "поле арбитража", "поле контроля", "поле данных" (если имеется) (самые младшие 15 коэффициентов полинома =0), должен быть разделён на полином следующего вида:

x^15+x^14+x^10+x^8+x^7+x^4+x^3+1

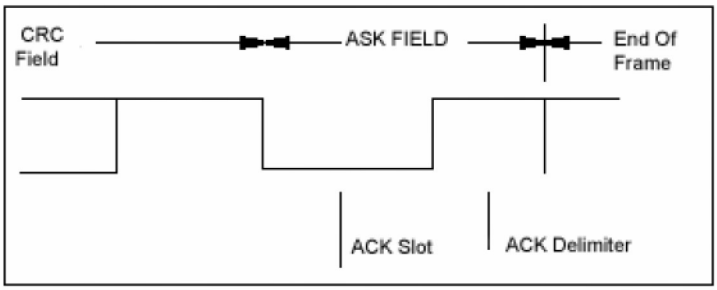
Остаток этого полиномиального деления есть CRC-последовательность, передаваемая по шине.

CRC-разделитель (CRC Delimiter):

CRC-последовательность сопровождается CRC-разделителем, который всегда равен лог. "1".

Поле подтверждения (ACK FIELD)

Длина 2 бита. Содержит область подтверждения (1 бит) и разделитель подтверждения (1 бит). В поле подтверждения передающий узел посылает два бита с лог. "1". Приемник, получивший правильное сообщение, информирует об этом передатчик, посылая бит с лог. "0" (т.е. перезаписывая бит в области подтверждения с лог. "1" на бит с лог. "0").



Область подтверждения (ACK Slot)

Все узлы, получившие соответствующую CRC-последовательность, сообщают об этом внутри области подтверждения перезаписью бита с лог. "1" на бит с лог. "0".

Разделитель подтверждения (ACK Delimiter)

Второй бит области подтверждения должен быть - лог. "1". Следовательно, область подтверждения окружена битами с лог. "1" (CRC-разделитель и разделитель подтверждения).

Конец кадра (END OF FRAME)

Каждый кадр данных и кадр удаленного запроса данных разграничены последовательностью флагов, состоящей из семи битов с лог. "1".

Структура кадра удаленного запроса данных (REMOTE FRAME)

Узел может инициализировать передачу кадра данных другим узлом, посылая кадр удаленного запроса данных. Этот кадр состоит из 6 полей: "Начало кадра", "поле арбитража", "поле контроля", "поле CRC", "поле подтверждения", "конец кадра".

В отличие от кадра данных , RTR бит = "1". Здесь нет поля данных, зависящего от значения "кода длины данных", внутри этого поля может быть записано любое из допустимых значений (0…….8). Полярность RTR бита показывает, является ли передаваемый кадр кадром данных или кадром удаленного запроса данных

Структура кадра ошибки (ERROR FRAME)

Состоит из двух различных полей. Первое поле является суперпозицией флагов ошибки различных узлов, второе поле - поле разделителя ошибки. Флаги ошибки могут быть двух типов: активный флаг ошибки (ACTIVE ERROR FLAG) и пассивный кадр ошибки (PASSIVE ERROR FLAG). Активный флаг формируется активным узлом (см. ниже) и состоит из шести доминантных битов (нарушающих синхронизацию или структуру полей подтверждения или конца кадра). Пассивный флаг формируется пассивным узлом и состоит из шести рецессивных бит. После передачи флага ошибки устройство посылает рецессивные биты до тех пор, пока не обнаружит их на шине, после чего передает еще семь рецессивных битов.

Структура кадра перегрузки

Кадр перегрузки состоит из двух флаговых битов и разделителя. Кадр передается в двух случаях:

– При необходимости задержки следующего кадра данных или кадра запроса данных по внутренним обстоятельствам;
– При обнаружении доминантного бита во время межкадрового интервала.

В первом случае передача должна начинаться во время передачи первого бита межкадрового интервала, во втором случае - после обнаружения доминантного бита. Флаг состоит из шести доминантных битов, разделитель - из восьми рецессивных.

Межкадровый интервал

Разделяет последовательные кадры данных и запроса данных и состоит из трех рецессивных бит собственно интервала и далее из неограниченного числа рецессивных бит свидетельствующих о незанятой шине. Узлам шины не разрешается начинать передачу во время межкадрового интервала, за исключением передачи кадра перегрузки. Узлы, находящиеся в состоянии пассивной ошибки после передачи собственного кадра данных должны передавать дополнительные восемь рецессивных бит после трех бит общего интервала.

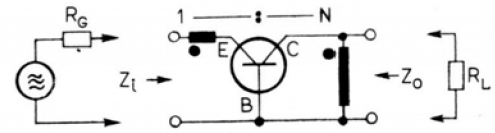
1 Высокочастотная схемотехника

При разработке ВЧ устройств много внимания уделяется вопросу согласования, имеющему большое значение для достижения эффективной передачи сигналов. Стандартом считаются устройства, входное и выходное сопротивление которых соответствует волновому сопротивлению 50 Ом. Такая стандартизация позволяет облегчить сопряжение различных устройств.

Параметрами, характеризующими степень согласования, являются коэффициент стоячей волны (КСВ) и обратный ему коэффициент бегущей волны (КБВ). Идеальным является единичное значение КСВ и КБВ.

1.1 Широкополосные усилители малых сигналов

Наряду с традиционными схемами с ОЭ, ОБ распространение получили усилители с индуктивными связями (ООС). Различают три основных типа: A, B, C согласно /1/.

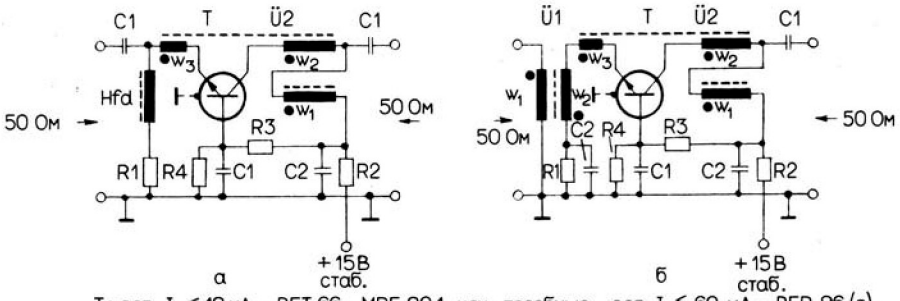


$$Z_i = \frac{N+1}{N^2} \cdot R_L + \frac{R_B}{\beta}$$

$$Z_o = \frac{N^2}{\beta \cdot (N+1)} \cdot (\beta \cdot R_G + R_B)$$

$$G_U = \frac{N}{1 + \frac{N^2}{\beta \cdot (N+1)} \cdot \frac{R_B}{R_L}}$$

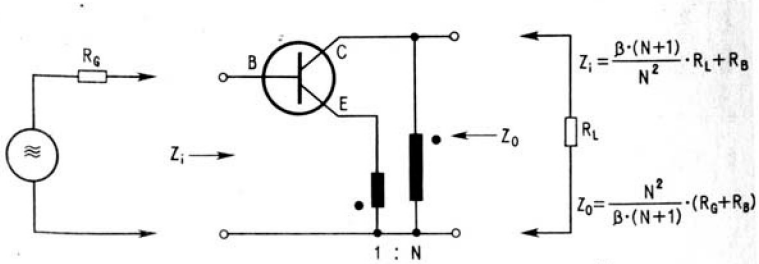
$R_B = R_{BB} + R_{BE}$; R_{BB} – диффузное сопротивление базы,
R_{BE} – внутреннее сопротивление база–эмиттер;
β – коэффициент усиления тока в схеме с общим эмиттером,
Транзистор n–p–n или p–n–p



T: для $I_C \leq 18$ мА – BFT 66, MRF 904 или подобные, для $I_C \leq 60$ мА – BFR 96 (s), MRF 965 или подобные.
Остальные компоненты описаны в тексте. Строго соблюдайте полярность при подключении обмоток трансформатора!

Для усилителей этого типа большое значение имеет линейность и устойчивость, достигающиеся с помощью отрицательной обратной связи. Схема склонна к потере устойчивости при рассогласовании на выходе.

Усилители класса B и C (с резистивной ОС) с большей стабильностью строятся последующим схемам:



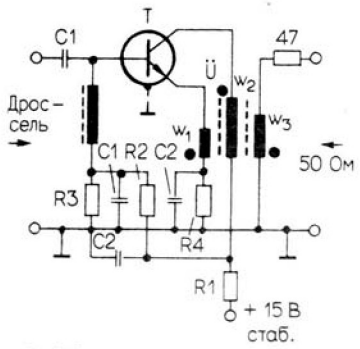
$$Z_i = \frac{\beta \cdot (N+1)}{N^2} \cdot R_L + R_B$$

$$Z_o = \frac{N^2}{\beta \cdot (N+1)} \cdot (R_G + R_B)$$

$$G_U = \frac{N}{1 + \frac{N^2}{\beta \cdot (N+1)} \cdot \frac{R_B}{R_L}}$$

$R_B = R_{BB} + R_{BE}$; R_{BB} – диффузное сопротивление базы
R_{BE} – внутреннее сопротивление база–эмиттер
β – коэффициент усиления тока

f 1 – 150 МГц F = 5,1 дБ (⩽ 50 МГц)
IP_{i3} + 8,5 дБм G_p = 20 дБ
KP_i – 1,7 дБм $A_x \approx$ 3 дБ



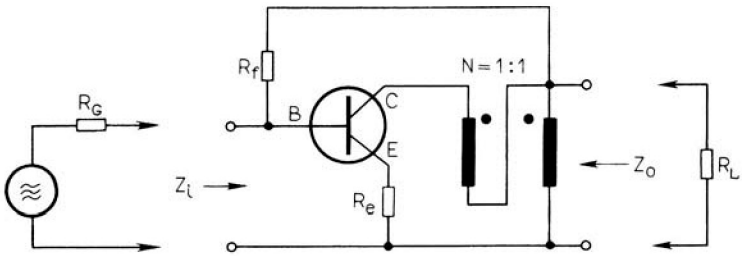
T – BFR 96 (s), MRF 965 или подобные

D – 1 N 4148 или подобный

Ü – 2×6 витков, провод ⌀0,25 мм, медный с лаковой изоляцией, намотка бифилярная, виток к витку на тороидальном сердечнике FT – 23 – 72

L – B 82114 – R – A3 (Сименс) или аналогичный (широкополосный дроссель с Q ⩽ 1)

T: для $I_C \leqslant 18$ мА – BFT 66, MRF 904 или подобные, для $I_C \leqslant 60$ мА – BFR 96 (s), MRF 965 или подобные. Остальные компоненты описаны в тексте. Строго соблюдайте полярность при подключении обмоток трансформатора Ü!



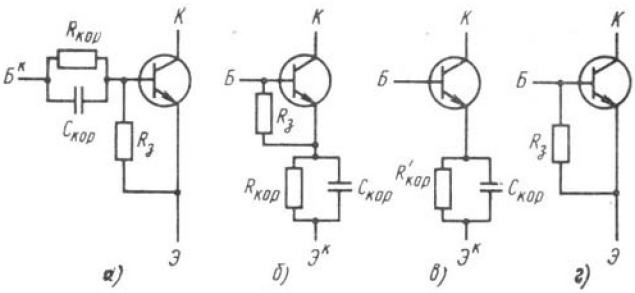
Условия: $R_G = R_L = 50$ Ом; $R_e \cdot R_f = 2500$ Ом, N = 1:1;
Результаты: $Z_i \approx Z_o \approx 50$ Ом, $G_u \approx \sqrt{\frac{R_f}{R_e}}$;
Транзисторы n – p – n или p – n – p

1.2 Генераторы

Усилители - генераторы с независимым возбуждением строятся по схеме усилителей с общей базой или эмиттером, при этом работая в режиме отсечки. Принципы работы рассматривались в курсе ОРЭС. Основные тонкости - выбор режима, согласование, обеспечение требуемой полосы и фильтрация гармоник.

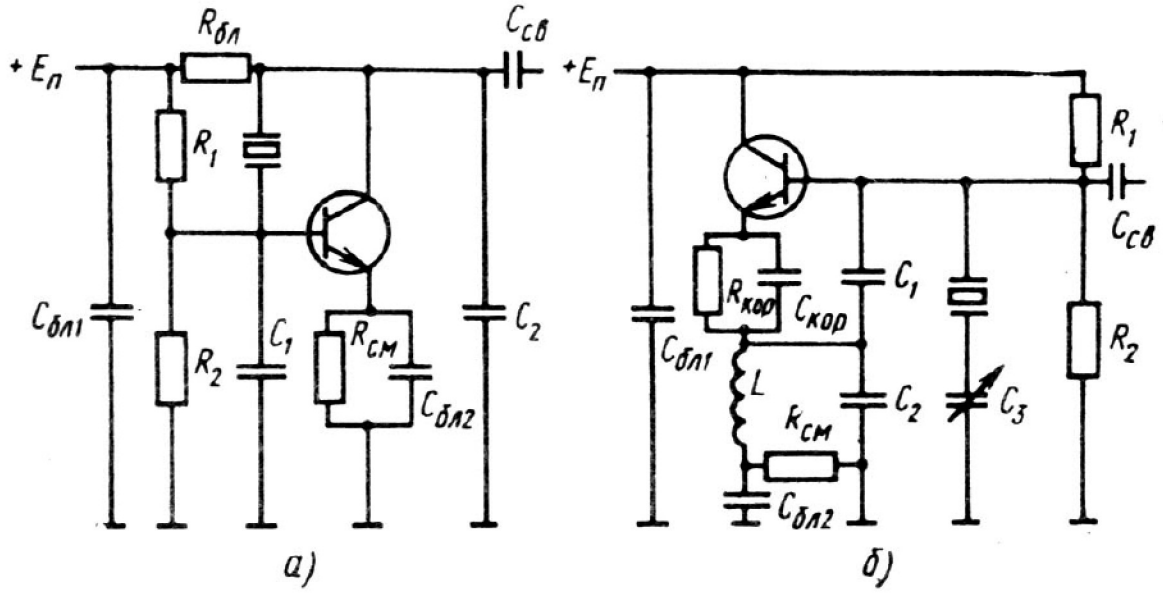
Основные схемы автогенераторов - емкостная и индуктивная трехточки изучались в курсе ОРЭС, поэтому о них мы говорить не будем, остановимся на некоторых схемотехнических особенностях генераторных схем.

Для стабилизации параметров транзисторов применяются схемы ВЧ эмиттерной и базовой коррекции



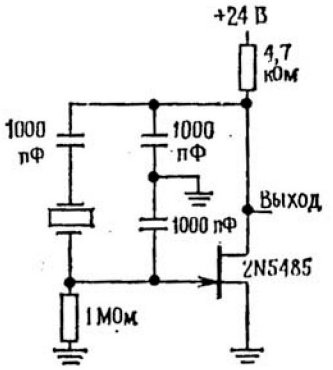
Схемы автогенераторов проектируют таким образом, чтобы обеспечивалась максимальная стабильность частоты. Для этого необходимо, прежде всего обеспечить максимальную добротность контура (с помощью частичного включения).

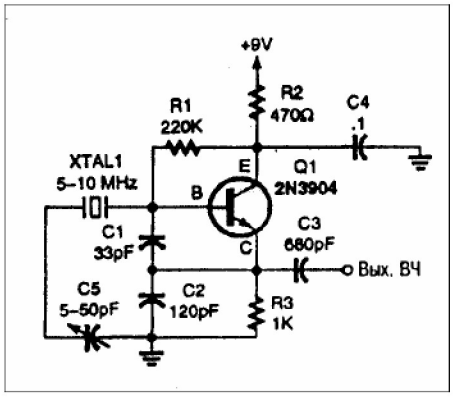
Лучших результатов добиваются с помощью кварцевой стабилизации частоты, при этом кварцевые резонаторы используют на основной гармонике, либо на нечетных гармониках. Кварцевый резонатор включают либо в качестве индуктивности, либо в качестве полосового фильтра. Также реализуются и мультивибраторы на логических элементах, в микропроцессорах такие элементы встраиваются внутрь микропроцессоров.



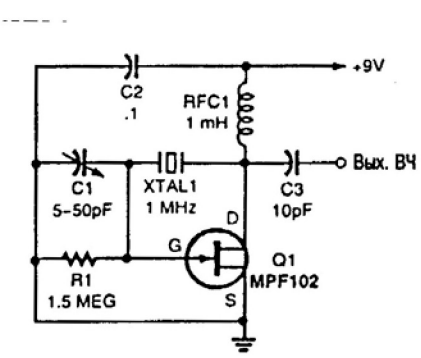
1.3 Схемы кварцевых автогенераторов

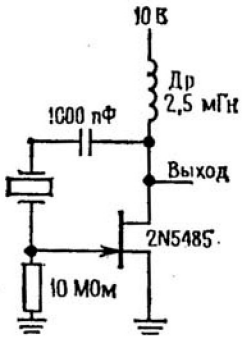
В основном получили распространение схемы с кварцем в цепи ОС и с кварцем в контуре (обычно в качестве индуктивности). Последний вариант в зарубежной литературе называется схемой Колпитца.





Незначительно отличается от схемы Колпитца схема Пирса, где кварцевый резонатор устанавливается между базой и коллектором (стоком и затвором) транзистора, а необходимый сдвиг фаз получается за счет свойств кварца. Однако в этих схемах затруднено возбуждение колебаний на гармониках.





Первый вариант в зарубежной литературе получил название схемы Батлера. В этой схеме статическая емкость кварца компенсируется с помощью дополнительной индуктивности $L_{ком}$. Генерируемая частота в таких схемах близка к частоте последовательного резо-

нанса кварца.

Низкочастотные гармонические генераторы принято реализовывать на базе RC-генераторов на базе ОУ, или с помощью технологии прямого синтеза.

1.4 Умножители частоты

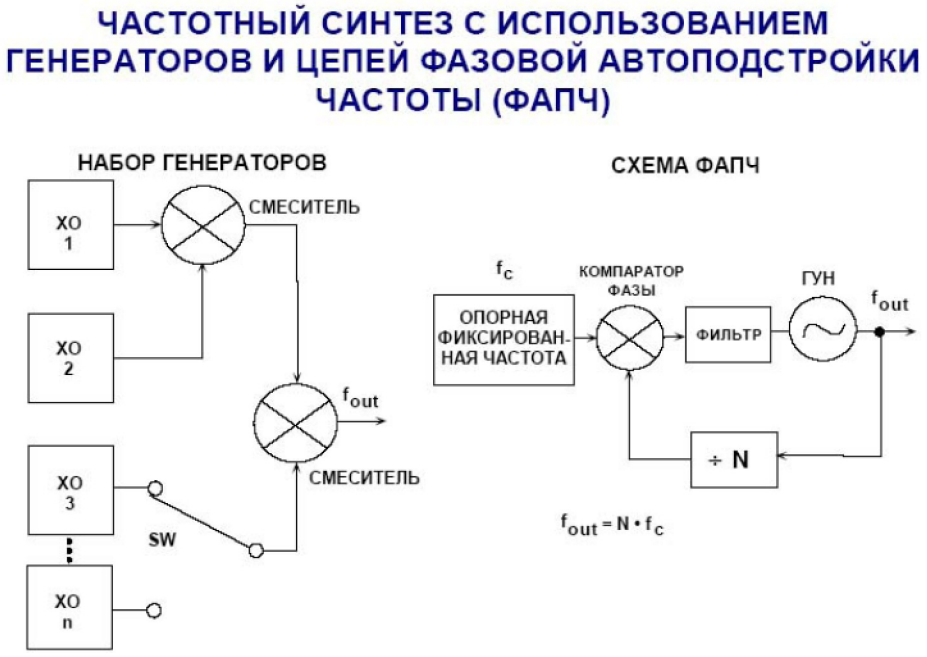
Высокостабильные генераторы на базе кварцевых резонаторов не могут быть сделаны высокочастотными из-за физических особенностей кварцевых резонаторов (20 Мгц на основной гармонике и порядка 100-150 на кратных гармониках). Повышения частоты добиваются с помощью умножителей.

Различают умножители на базе нелинейных сопротивлений или нелинейных емкостей. Умножители на базе нелинейных сопротивлений выполняются аналогично усилителям мощности с отсечкой с помощью соответствующей настройки колебательных систем и выбора угла отсечки. Недостаток этих умножителей - низкий КПД и кратность не выше 3.

Умножители на варакторах имеют более высокий КПД, но не могут усиливать сигнал, обеспечивая при этом более высокую кратность умножения.

1.5 DDS

Цифровые методы эффективны и для построения синтезаторов частоты. Классические методы предполагают суммирование сигналов группы генераторов либо применение ФАПЧ. При проектировании ФАПЧ часто приходится идти на компромиссы между фазовым шумом, скоростью подстройки, разрешающей способностью по частоте и т.д.

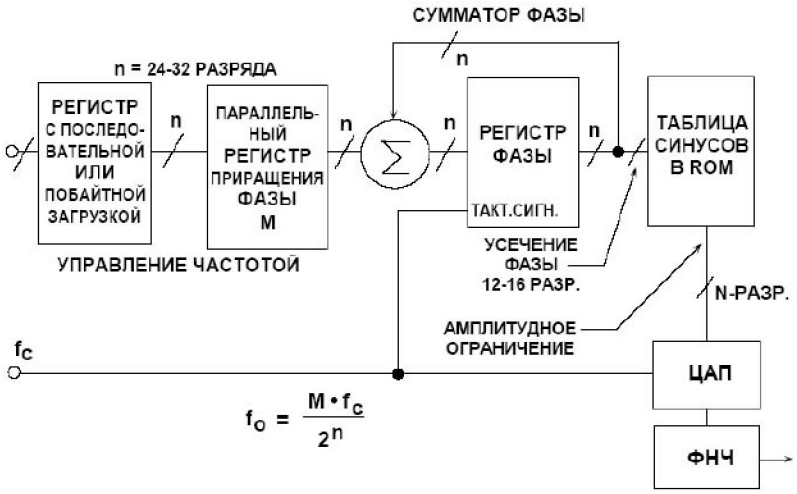


Метод генерации набора частот от источника опорной частоты, реализуемый в цифровой форме, развился в так называемый метод прямого цифрового синтеза (DDS).

В этой упрощенной модели, стабильный генератор тактового сигнала управляет программируемым ПЗУ (PROM), который хранит один или большее целое число циклов синусоидального сигнала (или другого сигнала произвольной формы). По мере того, как адресный счетчик проходит через каждую ячейку памяти, соответствующая цифровая амплитуда сигнала из каждой ячейки подается на ЦАП, который, в свою очередь, воспроизводит аналоговый выходной сигнал. Спектральная чистота

конечного аналогового выходного сигнала определяется, прежде всего, ЦАП. Фазовый шум является, в основном, шумом задающего генератора. Основная проблема этой простой DDS-системы состоит в том, что выходная частота может быть изменена только путем изменения частоты задающего генератора или посредством перепрограммирования ПЗУ, что делает систему весьма негибкой. На практике DDS-система осуществляет эту основную функцию намного более гибким и эффективным способом, используя цифровую схему, называемую генератором с цифровым управлением.

ГИБКАЯ СИСТЕМА ПРЯМОГО ЦИФРОВОГО СИНТЕЗА (DDS)



Сердцем системы является сумматор фазы, чье содержимое обновляется однократно за каждый тактовый цикл. Каждый раз при обновлении сумматора фазы цифровое число M, сохраненное в регистре приращения фазы, добавляется к числу в сумматоре фазы. Предположим, что число в delta-регистре равно 00...01 и что начальное содержимое сумматора фазы равно 00...00. Сумматор фазы обновляется значением 00...01 каждый тактовый цикл. Если сумматор является 32-разрядным, для возврата сумматора фазы в состояние 00...00 требуется 2^{32} тактовых цикла (более 4 миллиардов), после чего цикл повторяется. Усеченное значение выходного сигнала сумматора фазы служит адресом для таблицы задания синуса (или косинуса). Каждый адрес в таблице соответствует точке синусоидального сигнала с фазой от 0° до 360°. Таблица поиска содержит информацию, соответствующую цифровой амплитуде для одного полного цикла синусоидального сигнала (в действительности, требуются только данные для 90°, потому что данные о квадранте содержатся в двух старших значащих разрядах). Таким образом, таблица отображает фазу синусоидального сигнала сумматора фазы в виде значения цифровой амплитуды, которое, в свою очередь, подается на ЦАП.

Разрешающая способность системы по частоте равна $fc/2^n$. Для n=32 разрешающая способность больше, чем один к четырем миллиардам! В реальной DDS-системе не все разряды от сумматора фазы используются для выбора значения из таблицы, оставляются только первые 12-16 старших значащих разрядов (MSB), тогда как младшие разряды игнорируются. Это уменьшает размер таблицы и не ухудшает разрешающую способность по частоте. Усечение разрядности фазы только добавляет незначительное, но приемлемое количество фазового шума к окончательному выходному сигналу; тогда как большая часть выходных искажений возникает непосредственно в ЦАП. Описанная выше базовая DDS-система представляет чрез-

вычайно гибкое решение с весьма высокой разрешающей способностью. Частота может быть мгновенно изменена без искажения фазы простым изменением содержимого М-регистра. Реальные DDS-системы сначала требуют выполнения последовательной или параллельной загрузки нового значения частоты во внутренний буферный регистр, который предшествует М- регистру с параллельным выходом. Это делается для минимизации числа выводов в микросхеме счетчика. Число тактовых циклов, требуемых для загрузки регистра приращения фазы, определяет максимальную скорость, с которой можно менять выходную частоту. Современные ИМС генераторов прямого синтеза позволяют работу со скоростью до 300 MSPS и обеспечивают дополнительные возможности (ЧМ, ФМ).

МЕТОДЫ МОДУЛЯЦИИ В ЦИФРОВЫХ СИСТЕМАХ БЕСПРОВОДНОЙ СВЯЗИ

Для организации современных каналов связи используют стандартные виды модуляции (АМ, ЧМ, ФМ) с учетом специфики передаваемой информации, а также современных требований к радиопередающему и радиоприемному трактам.

В настоящее время все большая часть информации, передаваемой по разнообразным каналам связи, существует в цифровом виде. Это означает, что передаче подлежит не непрерывный (аналоговый) модулирующий сигнал, а последовательность целых чисел n_0, n_1, n_2,…, которые могут принимать значения из некоторого фиксированного конечного множества. Эти числа, называемые *символами* (symbol), поступают от источника информации с периодом T, а частота, соответствующая этому периоду, называется *символьной скоростью* (symbol rate): $f_T = 1/T$.

Замечание. Часто используемым на практике вариантом является *двоичная* (binary) последовательность символов, когда каждое из чисел n_i может принимать одно из двух значений — 0 или 1.

Рассмотрим подробнее применяемые схемы:

Типичный подход при осуществлении передачи дискретной последовательности символов состоит в следующем. Каждому из возможных значений символа сопоставляется некоторый набор параметров несущего колебания. Эти параметры поддерживаются постоянными в течение интервала T, то есть до прихода следующего символа. Фактически это означает преобразование последовательности чисел $\{n_k\}$ в ступенчатый сигнал $s_n(t)$ с использованием кусочно-постоянной интерполяции:

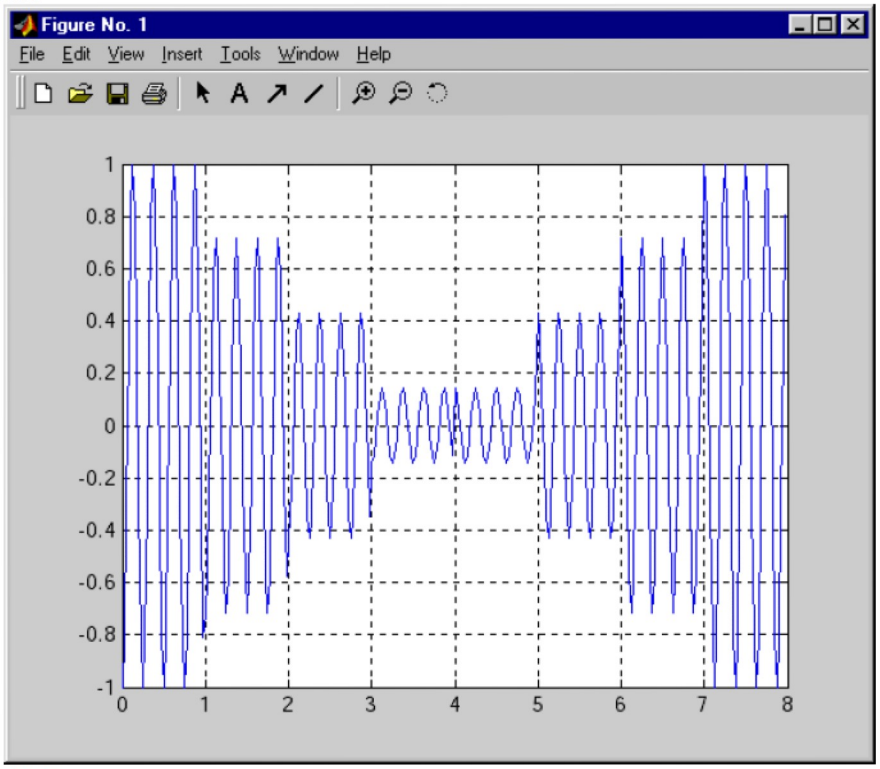
$$s_n(t) = f(n_k),\ kT <= t < (k+1)T.$$

Здесь f — некоторая функция преобразования. Полученный сигнал $s_n(t)$ далее используется в качестве модулирующего сигнала обычным способом.

Такой способ модуляции, когда параметры несущего колебания меняются скачкообразно, называется *манипуляцией* (keying). В зависимости от того, какие именно параметры изменяются, различают амплитудную (АМн), фазовую (ФМн), частотную (ЧМн) и квадратурную (КАМн) манипуляцию.

Амплитудная манипуляция

Амплитудная манипуляция (АМн; английский термин — amplitude shift keying, ASK), при которой скачкообразно меняется *амплитуда* несущего колебания, является частным случаем квадратурной манипуляции.



Демодуляция АМн-сигнала может выполняться теми же методами, что и в случае квадратурной манипуляции (путем умножения на несущее колебание). Однако наличие всего лишь двух

возможных значений начальной фазы несущей, отличающихся друг от друга на 180° , делает возможной реализацию автоматической подстройки начальной фазы с помощью петли ФАПЧ.

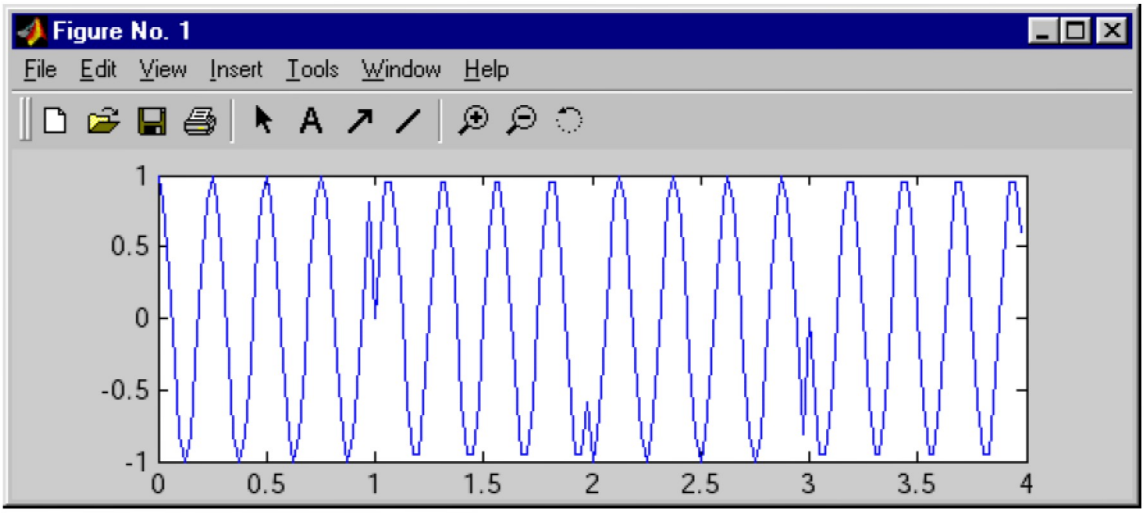
На графике хорошо виден бросок фазы в середине сигнала. Кроме того, можно заметить, что фазы посылок в первой и второй половинах сигнала, имеющих одинаковые амплитуды, отличаются на 180° .

Фазовая манипуляция

Фазовая манипуляция (ФМн; английский термин — phase shift keying, PSK), при которой скачкообразно меняется фаза несущего колебания, тоже является частным случаем квадратурной манипуляции.

На практике фазовая манипуляция используется при небольшом числе возможных значений начальной фазы — как правило, 2, 4 или 8. Кроме того, при приеме сигнала сложно измерить *абсолютное* значение начальной фазы; значительно проще определить *относительный* фазовый сдвиг между двумя соседними символами. Поэтому обычно используется *фазоразностная* манипуляция (синонимы — дифференциальная фазовая манипуляция, относительная фазовая манипуляция; английский термин — differential phase shift keying, DPSK).

Демодуляция фазовой манипуляции может выполняться тем же методом, что и в случае квадратурной манипуляции (путем умножения на несущее колебание). Применительно к ФМн данный метод демодуляции часто называется *корреляционным*.



На графике видны скачки фазы на 90° , происходящие при переходе от одного символа к другому.

Частотная манипуляция

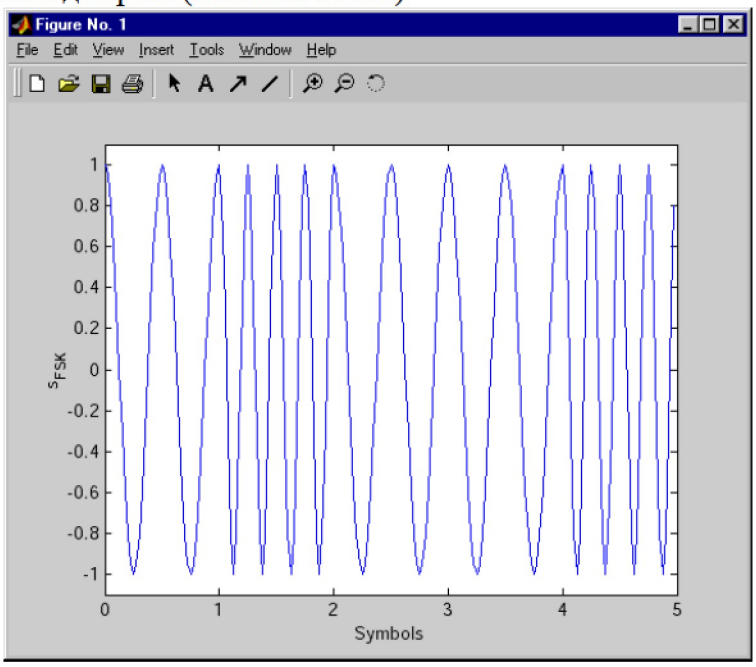
При частотной манипуляции (ЧМн; английский термин — frequency shift keying, FSK) каждому возможному значению передаваемого символа сопоставляется своя частота. В течение каждого символьного интервала передается гармоническое колебание с частотой, соответствующей текущему символу. При этом возможны различные варианты, различающиеся выбором начальной фазы отдельных синусоидальных посылок.

Первый способ — когда все посылки, соответствующие одному закону передаваемого символа, имеют одинаковую начальную фазу, то есть являются идентичными. При этом можно заранее сформировать наборы отсчетов для всех возможных дискретных символов. Тогда осуществление частотной манипуляции сводится к последовательной передаче заранее рассчитанных последовательностей отсчетов, соответствующих поступающим символам. Однако если используемые частоты манипуляции не кратны символьной скорости, сформированный таким образом ЧМн-сигнал будет содержать разрывы (скачки) на стыках символов. Вследствие этого спектр сигнала будет иметь всплески на частотах, кратных символьной скорости.

Второй способ — непрерывная генерация колебаний всех необходимых частот и осуществление переключения между этими сигналами в соответствии с поступающими символами. Данный метод также не гарантирует отсутствие скачков на стыках символов, но вследствие того, что начальные фазы посылок меняются от символа к символу, скачки возникают не на всех стыках, и их

величина оказывается различной. В результате возникающие из-за скачков всплески спектра в данном случае выражены слабее.

Наконец, третий способ — когда поступающие для передачи символы управляют скоростью линейного нарастания текущей фазы, а частотно-манипулированный сигнал формируется путем вычисления косинуса этой текущей фазы. При этом фазовая функция, а значит, и сам ЧМн-сигнал оказываются непрерывными (не имеющими скачков). Данный способ сложнее в реализации, но он дает наиболее компактный спектр сигнала. ЧМн-сигнал, полученный таким образом, называется *частотно-манипулированным сигналом с непрерывной фазовой функцией* (continuous phase frequency shift keying — CPFSK). На рисунке изображен сигнал частотной манипуляции, с частотами, отличающимися в два раза (посылка 01001)



На графике хорошо заметно двукратное изменение частоты сигнала при смене значения передаваемого бита. В данном примере на длительности символа укладывается два периода колебания при передаче нулевого бита и четыре периода — при передаче единичного бита.

Демодуляция ЧМн-сигнала

Прием ЧМн-сигнала, как правило, осуществляется *корреляционным* методом. При этом корреляционный прием может быть *когерентным* или *некогерентным*. Когерентный метод может использоваться, если известны начальные фазы посылок. Сущность его состоит в вычислении *взаимной корреляции* между принимаемым сигналом и колебаниями-образцами (опорными сигналами), представляющими собой гармонические колебания с используемыми для манипуляции частотами. Взаимная корреляция сигнала с k-м опорным сигналом для n-го по времени символа рассчитывается следующим образом:

.

Здесь $s(t)$ — ЧМн-сигнал, ω_k — частота манипуляции, соответствующая символу, равному k, φ_{0k} — начальная фаза посылки, T — длительность передачи символа. Использованные пределы интегрирования задают обработку n-го (по счету) символа.

После расчета взаимных корреляций $z_k(n)$ для всех k они сравниваются друг с другом в поисках максимального значения. Значение k, соответствующее максимальному $z_k(n)$, принимается в качестве демодулированного символа.

Замечание. При цифровой реализации демодуляции ЧМн-сигнала вместо *интегрирования*, разумеется, используется *суммирование* дискретных отсчетов подынтегрального выражения.

Если начальные фазы передаваемых посылок неизвестны (на практике так чаще всего и бывает), приходится использовать *некогерентный* (или *квадратурный*) корреляционный прием. В этом случае опорными сигналами служат не вещественные гармонические колебания, а комплексные экспоненты, а у результата интегрирования вычисляется модуль:

Из-за игнорирования фазовой информации помехоустойчивость некогерентного метода несколько ниже, чем у когерентного.

Минимальная частотная манипуляция

Для повышения помехоустойчивости ЧМн желательно, чтобы посылки, соответствующие разным символам, были *некоррелированы*, то есть имели нулевую взаимную корреляцию. Считая начальные фазы посылок нулевыми, ЧМн-сигналы для символов 0 и 1 можно записать так:

$$s_0(t) = A \cos \omega_0 t,\ 0 <= t <= T,$$
$$s_1(t) = A \cos \omega_1 t,\ 0 <= t <= T.$$

Их взаимная корреляция при нулевом сдвиге по времени равна

.

Если $(\omega_1 + \omega_0)T >> 1$, то первое слагаемое значительно меньше второго и им можно пренебречь:

.

Это значение равно нулю при $(\omega_1 - \omega_0)T = \pi n$, где n — целое число, не равное нулю. Таким образом, минимальное значение расстояния между соседними частотами манипуляции, при котором посылки, соответствующие разным символам, оказываются некоррелированными, составляет половину символьной скорости:

,

, где f_T — символьная скорость.

Двухпозиционная (двоичная) ЧМн, частоты которой выбраны согласно приведенной формуле, получила название *минимальной частотной манипуляции* (МЧМн, английский термин — minimum shift keying, MSK).

Квадратурная модуляция

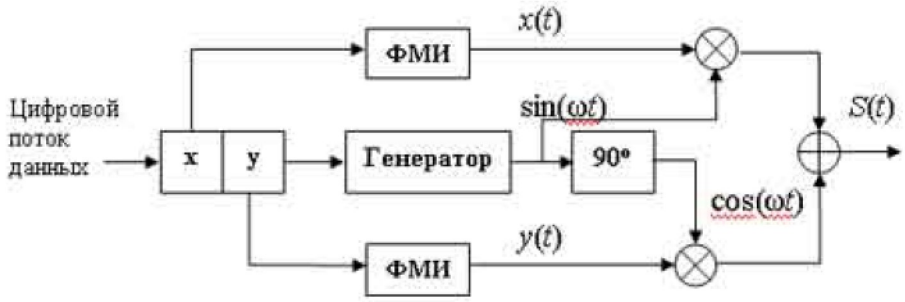


Схема квадратурного модулятора

Исходная последовательность двоичных символов длительностью T при помощи регистра сдвига разделяется на нечетные y и четные x импульсы, которые поступают на входы формирователей манипулирующих импульсов (ФМИ) соответственно квадратурного ($\cos\omega t$) и синфазного ($\sin\omega t$) каналов. На выходах ФМИ образуются последовательности биполярных импульсов $x(t)$ и $y(t)$ с амплитудой $\pm U_m$ и длительностью $2T$, которые поступают на входы канальных перемножителей, где они независимо друг от друга модулируют по амплитуде два одинаковых несущих колебания, сдвинутых по фазе на 90°, т.е. находящихся в квадратуре. В результате, на их выходах формируются двухфазные (0, π) ФМ колебания. После суммирования они образуют сигнал ФМ-4 или квадратурный ФМ-сигнал (Quadrature Phase Shift Keying – QPSK).

Поскольку в каждом канале осуществляется амплитудная манипуляция, этот вид модуляции называют еще ***квадратурной амплитудной манипуляцией*** (QASK – Quadrature Amplitude Shift Keying) или просто квадратурной амплитудной модуляцией (КАМ).

При *квадратурной манипуляции* (КАМн; английский термин — quadrature amplitude shift keying, QASK) каждому из возможных значений дискретного символа C_k ставится в соответствие *пара* величин — амплитуды синфазной и квадратурной составляющих либо, что эквивалентно, амплитуда и начальная фаза несущего колебания:

$$C_k \rightarrow (a_k, b_k),\ s(t) = a_k \cos \omega_0 t + b_k \sin \omega_0 t,\ kT < t < (k+1)T$$

или

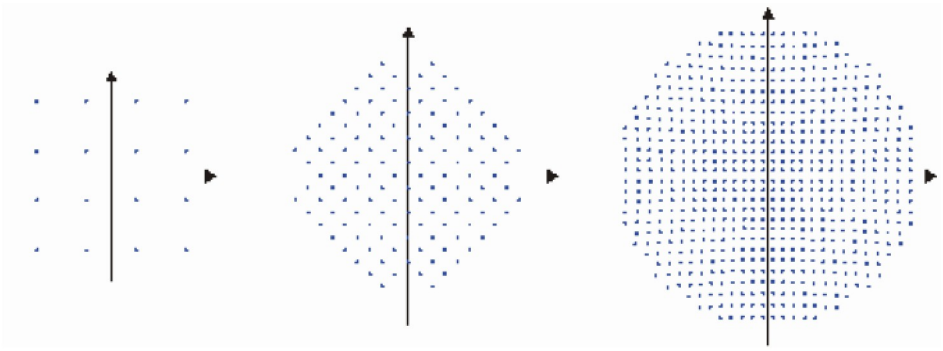
$$C_k \rightarrow (A_k, \varphi_k),\ s(t) = A_k \cos(\omega_0 t + \varphi_k),\ kT < t < (k+1)T.$$

Параметры аналогового колебания, сопоставленные дискретному символу C_k, удобно представлять в виде комплексного числа в алгебраической ($a_k + jb_k$) или экспоненциальной ($A_k \exp(j\varphi_k)$) форме. Совокупность этих комплексных чисел для всех возможных значений дискретного символа называется сигнальным *созвездием* (constellation).

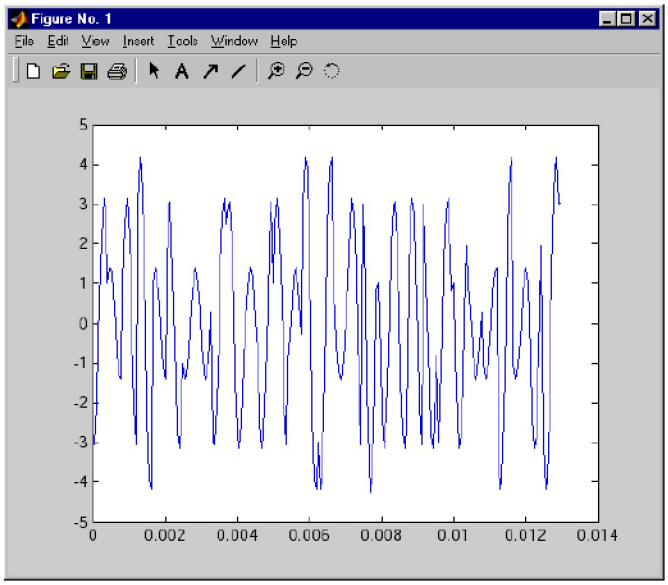
При представлении дискретного символа комплексным числом сигнал с квадратурной манипуляцией можно записать следующим образом:

$$, kT < t < (k+1)T.$$

На практике используются созвездия, содержащие от четырех до нескольких тысяч точек. Ниже показаны некоторые созвездия, используемые модемами, предназначенными для передачи данных по телефонным линиям.



Слева показано 16-точечное созвездие, используемое в протоколе V.32 при передаче данных со скоростью 9600 бит/с. Созвездие в центре имеет 128 точек, оно соответствует протоколу V.32bis и скорости передачи данных 14 400 бит/с. Наконец, созвездие, показанное справа, содержит 640 точек, оно используется модемами, работающими согласно протоколу V.34 при скорости передачи данных 28 800 бит/с.



Демодуляция КАМн-сигнала

Демодулируется сигнал с квадратурной манипуляцией так же, как и в случае аналоговой квадратурной модуляции — сигнал умножается на два несущих колебания, сдвинутых по фазе друг относительно друга на 90° , а результаты умножения пропускаются через ФНЧ. На выходе этих ФНЧ будут получены аналоговые сигналы синфазной и квадратурной составляющих. Далее эти сигналы дискретизируются с частотой, равной символьной скорости. Пары отсчетов синфазной и квадратурной составляющих образуют комплексное число, и ближайшая к этому числу точка используемого созвездия (а точнее — соответствующий этой точке информационный символ) выдается в качестве выходного результата.

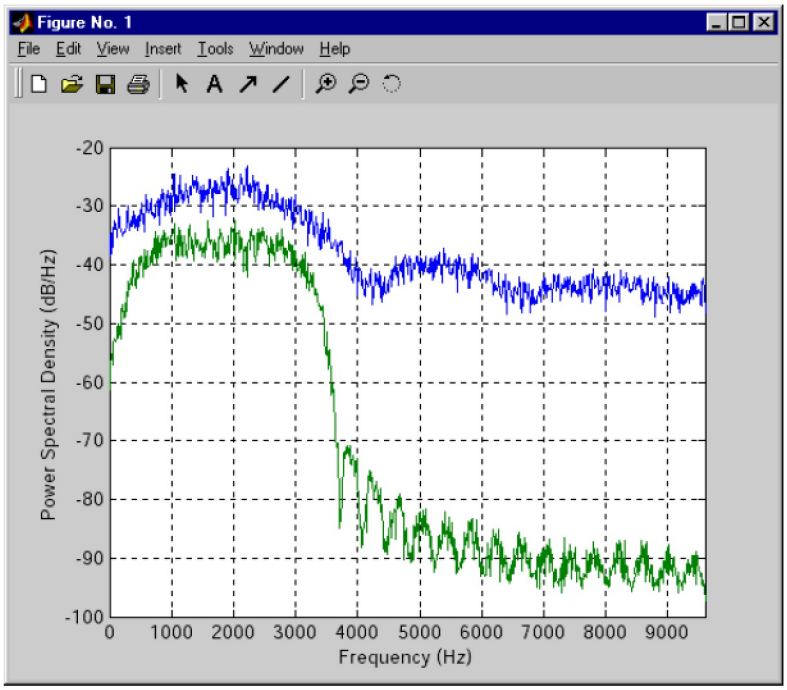
Формирование спектра

Если параметры модуляции аналогового сигнала поддерживаются постоянными в течение символьного такта и в начале нового такта изменяются скачкообразно, это приводит к появлению скачков и в сформированном сигнале. Как известно из теории преобразования Фурье, спектр сигнала, содержащего скачки, затухает с ростом частоты медленно — пропорционально $1/\omega$. Чтобы сделать спектр более компактным, необходимо обеспечить *гладкость* сигнала (то есть непрерывность сигнала и, возможно, некоторого количества его производных), а это, в свою очередь, означает гладкость модулирующей функции. Следовательно, вместо скачкообразного изменения параметров модуляции необходимо выполнить *интерполяцию* между точками созвездия, соответствующими последовательным символам.

Согласно теореме Котельникова, мы можем соединить отсчеты, следующие с символьной скоростью F_d, плавной функцией, занимающей полосу частот от нуля до $F_d/2$. В этом случае квадратурно-манипулированный сигнал будет занимать полосу частот шириной F_d. Однако медленное затухание функций $\sin(x)/x$, составляющих базис Котельникова, делает неудобной интерполяцию на их основе. Наибольшее распространение при интерполяции отсчетов для цифровой модуляции получил SQRT-вариант *фильтра с косинусоидальным сглаживанием АЧХ*.

Фильтр, используемый для интерполяции, определяет форму спектра КАМн-сигнала, поэтому его называют *формирующим фильтром* (shaping filter), а сам процесс интерполяции — *формированием спектра* (spectral shaping).

Скачкообразное изменение параметров модуляции можно рассматривать как использование формирующего фильтра с *прямоугольной* импульсной характеристикой, длительность которой равна символьному интервалу.



Из графиков видно, что при использовании формирующего фильтра (нижний график) спектр сигнала оказывается значительно компактнее, чем в случае, когда формирование спектра не производится (верхний график).

При приеме такого сигнала в качестве ФНЧ необходимо использовать такой же фильтр, как для формирования спектра. Последовательное использование двух SQRT-фильтров с косинусоидальным сглаживанием дает результирующую импульсную характеристику, соответствующую обычному фильтру с косинусоидальным сглаживанием, равную нулю в точках, сдвинутых на целое число символов относительно пика. Это позволяет при правильном выборе моментов взятия отсчетов устранить помехи от соседних символов (так называемую *межсимвольную интерференцию*, МСИ; английский термин — intersymbol interference, ISI).

При прохождении сигнала через канал связи, обладающий *частотной дисперсией*, то есть вносящий разную групповую задержку на разных частотах, символы оказываются “размазанными” во времени и “наползают” друг на друга. В этом случае устранить межсимвольную интерференцию полностью не удается. Чтобы минимизировать ее, используют *адаптивные фильтры*, параметры которых автоматически подстраиваются под характеристики обрабатываемого сигнала.

1 Помехоустойчивость

1.1 Виды помех

Можно выделить следующие виды помех, действующие в РЭА:

- Наносекундные (EFT - electric fast transients), возникающие, в основном, при срабатывании механических переключателей и реле;
- Микросекундные, связанные с работой реактивных элементов в цепях мощных нагрузок (surge);
- Помехи от электростатических разрядов, возникающие при касании аппаратуры "наэлектризованным" человеком (ESD - Electrostatic Discharge);
- Помехи от радиопередатчиков и естественных источников энергии (молнии).

Устойчивость РЭА ко всем видам помех оговаривается российскими и зарубежными стандартами электромагнитной совместимости. На основе стандартов проводятся испытания, имитирующие реальные помехи.

1.1.1 Наносекундные помехи (EFT)

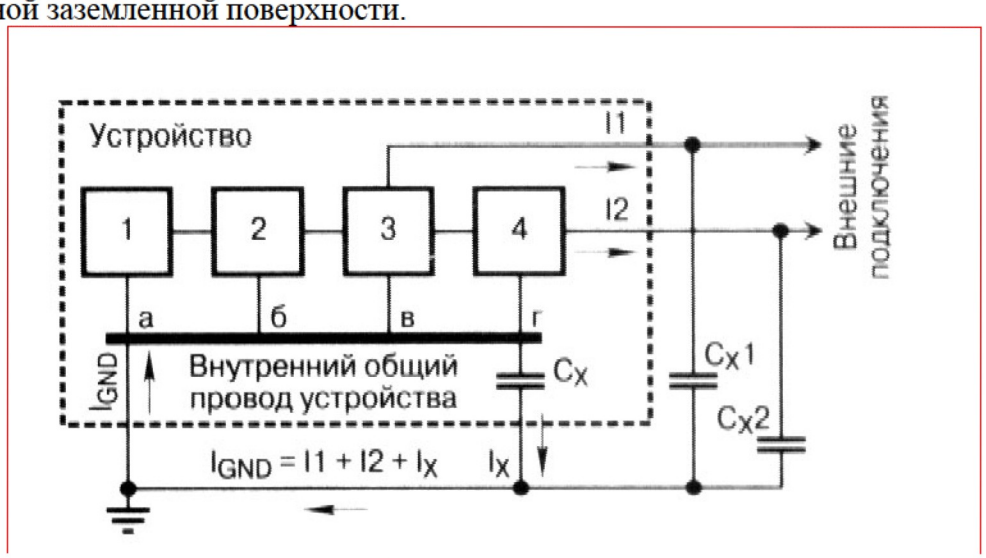
EFT обладают следующими особенностями:

- Широкий спектр (до единиц ГГц), т.е. можно считать ее дельта-функцией;
- Малая энергия, приводящая не к повреждению РЭА, а к обратимому сбою;

Следует отметить, что понятие "сбой" применимо только к устройствам с памятью, т.к. комбинационные устройства возвращаются в "правильное" состояние автоматически.

1.2 Испытания на устойчивость к EFT

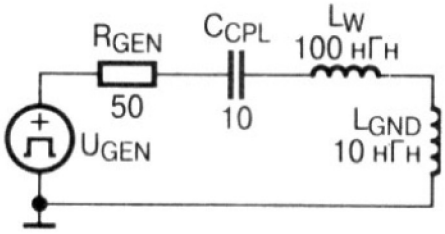
В соответствии со стандартом ГОСТ Р 51317.4.4-99, EFT помехи должны имитироваться треугольными импульсами длительностью 50нс по уровню 50% с длительностью фронта - 5нс. Импульсы должны генерироваться заземленным источником с внутренним сопротивлением 50 Ом. Амплитуда импульсов определяется степенью жесткости испытаний (всего 5 степеней), минимальная степень требует 0.5кВ, максимальная - 4кВ. В линии питания и заземления EFT инжектируются непосредственно (без развязки). При этом величина тока может достигать значительной величины. В сигнальные цепи импульсы инжектируются через "емкостные клещи", в которые по очереди закладываются все провода, подходящие к устройству. Величина связи составляет несколько пикофарад, амплитуда импульсов вдвое меньше, чем при испытании цепей питания и заземления. Устройство должно находиться на изолирующей подставке на расстоянии не менее 100 мм от сплошной заземленной поверхности.



1.3 Составляющие помехоустойчивости

1.3.1 Внутренний общий провод устройства

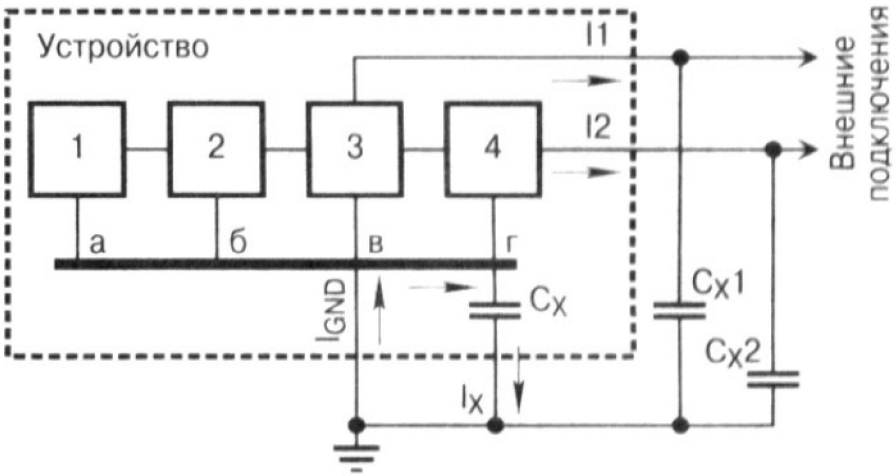
При прохождении помехи по внутреннему общему проводу создается разность напряжения между отдельными его точками. Например, если узлы 1, 2 построены по технологии ТТЛШ, достаточно разницы напряжения в 1В, чтобы вызвать сбой. Основную роль в создании падения напряжения играет индуктивность общего провода. Для примера, печатный проводник длиной 10мм и шириной 5мм имеет индуктивность около 10нГн.



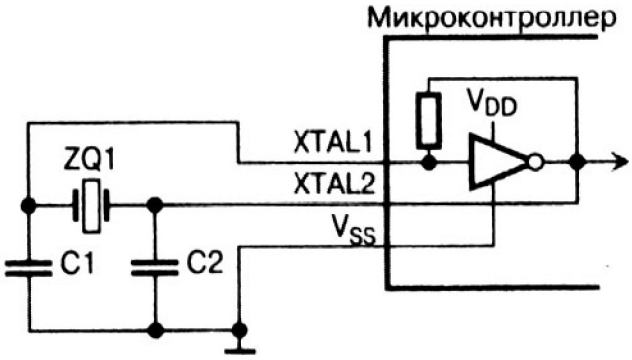
Эквивалентная схема, с помощью которой можно представить себе положение дел, состоит из сопротивления испытательного генератора, емкости C_{CPL}, состоящей из емкостей C_X, C_{X1}, C_{X2}, индуктивности проводов L_W (порядка 100нГн в типичном случае), подключенных к устройству, и индуктивности общего провода L_{GND}.

При таком положении дел от перекоса потенциалов не спасают широкие земляные провода и даже отдельные земляные слои МПП, хотя эти факторы, безусловно, улучшают ситуацию. Развязка внешних сигналов с помощью оптронов также улучшает ситуацию, но необходимо помнить о собственной емкости оптрона (порядка 0.5пФ), которая может свести на нет весь эффект.

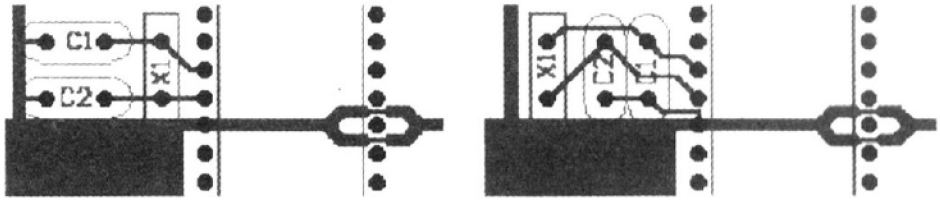
Радикального улучшения ситуации можно добиться с помощью перекомпоновки устройства и выбора правильной точки заземления. Например, на участке а-в токи помех не потекут, и, следовательно, не будет причины для сбоя. Таким образом, мы разделим общий провод на "чистую" часть и "грязную". В чистой части размещаем узлы, чувствительные к помехам, в грязной - нечувствительные. К сожалению, в реальности такой картины не получится, т.к. C_X не удастся сосредоточить в одном месте, в результате токи помех не будут полностью отсутствовать в чистой части общего провода.



Применительно к микроконтроллерам, причиной сбоя может быть тактовый генератор, выполняемый по схеме емкостной трехточки с кварцевым резонатором.



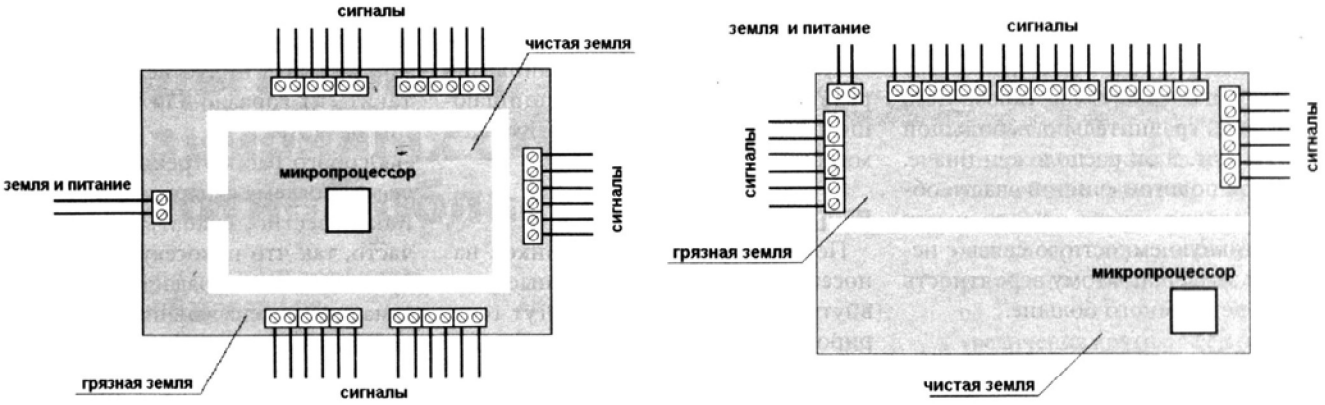
Перекос напряжения общего провода между C1 и V_{SS}, возникающий при действии EFT, может привести к формированию ложного короткого тактового импульса, проникающего внутрь микроконтроллера. Не всегда выручит здесь и сторожевой таймер, т.к. часто он тактируется от того же генератора.



На рисунке справа разводка обеспечит максимальную помехоустойчивость, т.к. ток помех не протекает по дорожке, соединяющей конденсаторы с выводом общего провода микроконтроллера.

Аналогичной проблемой является и цепь сброса микроконтроллера: решением может являться конденсатор, установленный таким образом, чтобы по проводу между ним и выводом общего провода микроконтроллера, не протекал ток помехи.

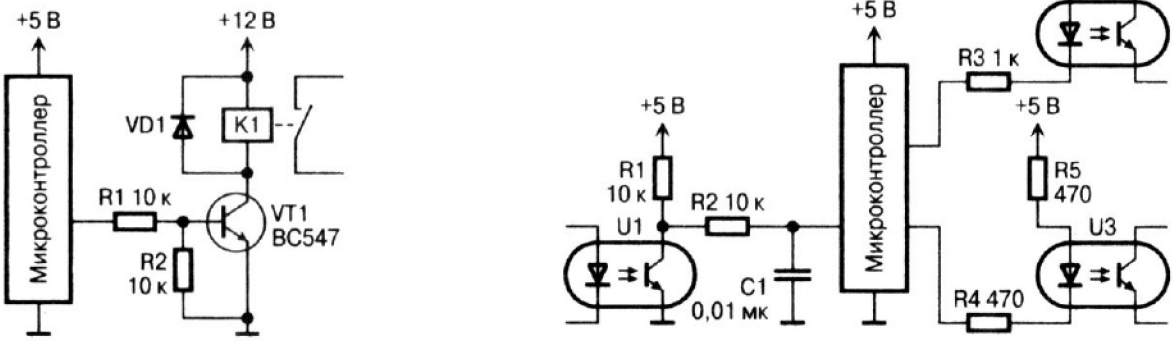
Обеспечить повышенную помехоустойчивость можно на этапе компоновки устройства. Например, в случае показанного ниже размещения, вероятность сбоя велика, так как микроконтроллер расположен в центре платы, т.е. максимально подвержен действию помех.



Не меняя компоновки устройства, можно улучшить помехоустойчивость за счет введения барьера, разделяющего плату на чистую и грязную части. Внешний контур предназначен для протекания токов помех, к этой части нельзя подключать помехочувствительные устройства. В сигнальные линии, проходящие между чистой и грязной частями, следует установить резисторы или дроссели в качестве барьеров.

1.3.2 Барьеры

Барьеры применяются для предотвращения проникновения помех из грязной части устройства в чистую. Применение барьеров позволяет в несколько раз повысить помехоустойчивость. Например, если микроконтроллер управляет мощной нагрузкой через реле, такая схема является потенциальным источником помехи.



Во-первых, необходимо подключать эмиттер транзистора к грязной части общего провода, во-вторых, резистор R1 является барьером для прохождения помехи внутрь микроконтроллера. При использовании полевого транзистора такой резистор вроде бы не нужен, однако его полезно установить именно, как барьер.

При использовании оптронов нужно учитывать их проходную емкость. Например, эмиттер транзистора входного оптрона нужно подключать к грязной части общего провода, а вот резистор R1 можно подключать как к чистому, так и к грязному питанию. Резистор R2 является барьером, конденсатор C1 не является обязательным, но может уменьшить ток помех, протекающий по выводу общего провода контроллера. Его нужно подключать к чистой части. Анод светодиода следует подключить к грязному питанию, при этом R3 будет служить барьером. При необходимости подключать оптрон к чистому питанию можно разделить резистор на две части, полезно также шунтировать светодиод конденсатором.

1.3.3 Емкостные связи

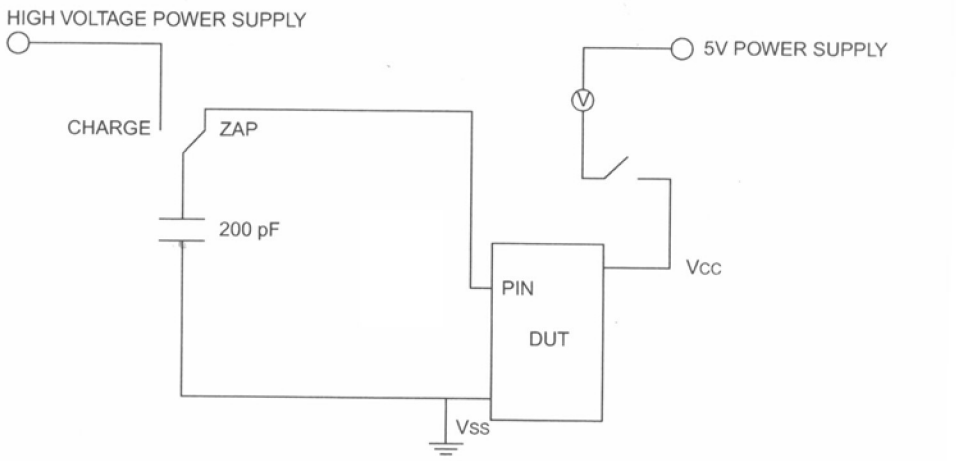
Разделение общего провода на чистую и грязную части не уменьшает емкости C_X, но соотношение площадей чистой и грязной частей соответствует отношению емкостей. Уменьшения C_X можно достичь также уменьшением площадей проводников и экранированием. Например, размещение микроконтроллера в центре платы не выгодно, т.к. значительная часть чистого общего провода будет образовывать значительную емкость с истинной землей.

1.3.4 Перекрестные связи

Источники EFT могут находиться и внутри устройства, например, внутри БИС, что связано с высокой скоростью переключения и перезарядом паразитных емкостей. Для уменьшения влияния этих помех необходимо устанавливать керамические конденсаторы вблизи выводов питания БИС. Аналогично, требуется устанавливать такой конденсатор и вблизи супервизоров питания, т.к. они могут являться проводниками EFT.

1.4 ESD

Электростатический разряд также необходимо учитывать при проектировании РЭА. Влияние ESD можно промоделировать разрядом высоковольтного конденсатора небольшой емкости.



Как правило, различают два вида опасности, исходящей от ESD: повреждение во время транспортировки микросхемы, платы и т.п. высоким напряжением и т.н. защелкивание, возникающее во включенном устройстве из-за воздействия ESD. Методы борьбы с первым типом воздействия очевидны: антистатическая упаковка, тщательное заземление, специальные меры при монтаже, ионизация воздуха.

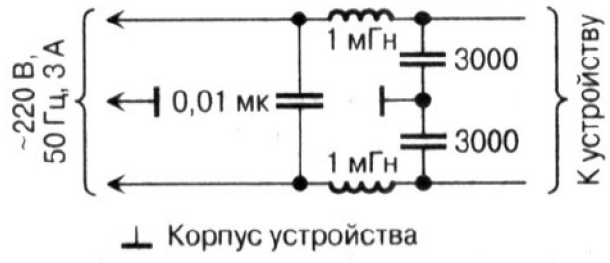
Защелкиванием называется особое состояние КМДП устройства, когда между выводами питания и земли образуется путь для протекания большого тока (КЗ или небольшое сопротивление). Это приводит к сильному нагреву устройства с возможными необратимыми последствиями. В отечественной литературе это явление иногда называют тиристорным эффектом. Для вывода устройства из состояния защелкивания можно только выключить питание. Для перехода в состояние защелкивания достаточно кратковременного воздействия импульса ESD. Основным видом защиты является установка керамического конденсатора возможно ближе к выводу питания БИС. В частности, фирма MICROCHIP рекомендует конденсатор емкостью 10000 пф. Характерно, что установка такого конденсатора между выводами питания и общего провода БИС повышает устойчивость к воздействию ESD и по другим выводам БИС.

1.5 Защита от сетевых помех

Источником помех может служить сеть питания устройства (220В, 50Гц), причем наличие помех и их интенсивность сильно зависят от условий эксплуатации РЭА. Для борьбы с влиянием этих помех можно рекомендовать несколько способов.

1.5.1 Применение сетевого фильтра

Фильтр, выполненный по схеме, приведенной ниже, позволяет подавлять кратковременно действующие помехи, проникающие из сети.

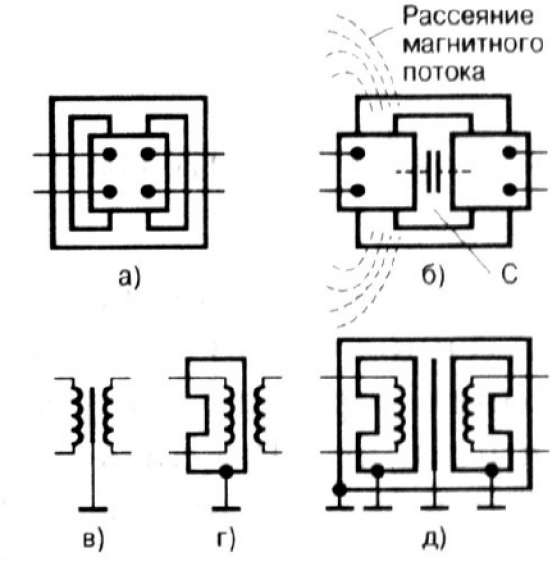


Необходимо использовать высокочастотные конденсаторы и дроссели без сердечников или с ВЧ сердечниками. В ряде случаев при действии эфирных ВЧ помех целесообразно экранировать провода питания (поместить в металлическую трубу).

1.5.2 Выбор сетевого трансформатора

При наличии емкости между обмотками трансформатора помехи могут проходить непосредственно во вторичную цепь. Рекомендуются следующие методы борьбы с такими помехами (в порядке возрастания эффективности):

- Применение трансформатора с обмотками, расположенными на разных катушках. Недостаток - снижается КПД за счет рассеяния магнитного поля;
- Применение заземленного экрана из медной фольги толщиной не менее 0.2мм между обмотками (не должен быть короткозамкнутый виток);
- Полное экранирование первичной обмотки заземленным экраном (не явл. КЗ витком);
- Индивидуальное экранирование обеих обмоток с разделительным экраном между ними. Весь трансформатор заключается в металлический корпус.



Разводка сетевых проводов внутри устройства должна выполняться отдельным жгутом, лучше экранированным проводом, подключенным к корпусу. Параллельно первичной обмотке устанавливается конденсатор емкостью около 0.1 мкф с последовательным токоограничивающим резистором сопротивлением 100Ом. Это позволяет замыкать энергию, накопленную в сердечнике, в момент размыкания сетевого выключателя. Необходимо помнить, что увеличение емкости фильтра выпрямителя приводит к значительным токам в момент включения устройства, что создает помехи соседним устройствам.

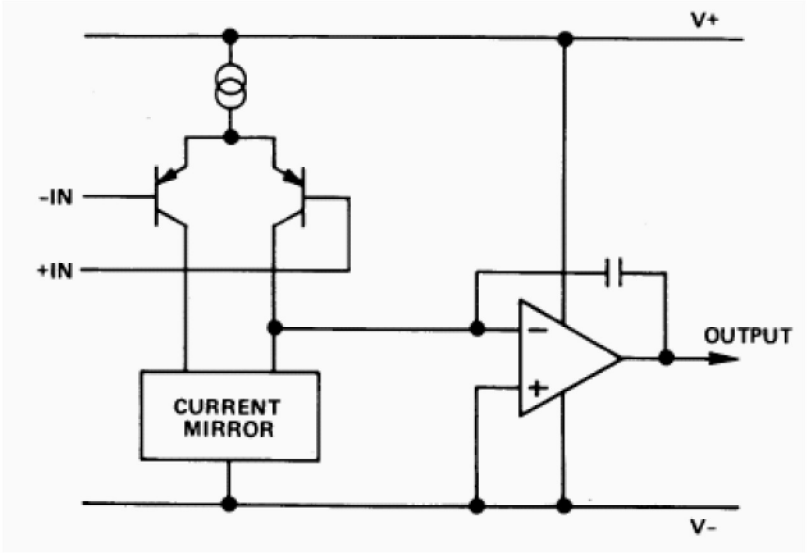
2 Печатные платы и помехоустойчивость

2.1 Разводка питания и общего провода

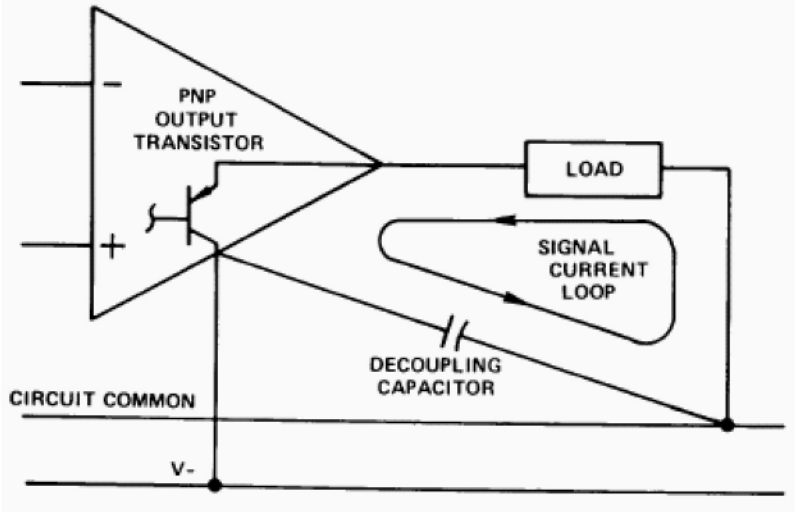
Расчет принципиальных схем исходит из предположения, что общий провод имеет потенциал, равный нулю во всех его точках. В реальности это не совсем так, и зависит от множества факторов, в том числе, и от разводки печатной платы. Неодинаковые потенциалы различных участков общего провода приводят к появлению эквивалентных источников помеховых сигналов в чувствительных цепях устройства. Необходимо максимально избегать возможного появления таких источников, для чего соблюдать ряд правил.

2.1.1 Правила для операционных усилителей

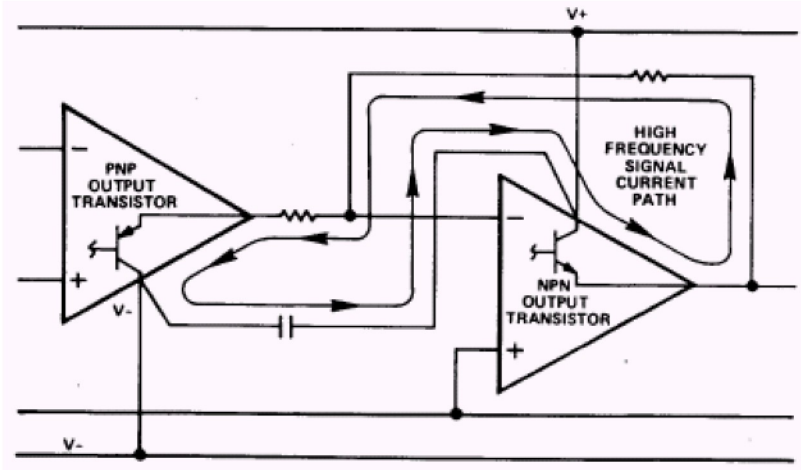
ОУ не имеют явного общего провода, однако, ток с выхода ОУ должен как-то возвращаться для образования замкнутого контура. Схемотехника современных ОУ такова, что выходной сигнал не может возвращаться через входные терминалы и должен возвращаться через отрицательный вывод питания (в случае однополярного питания он является общим проводом). Типичная схема современного ОУ включает интегратор, который сводит на нет коэффициент подавления влияния напряжения питания на высоких частотах.



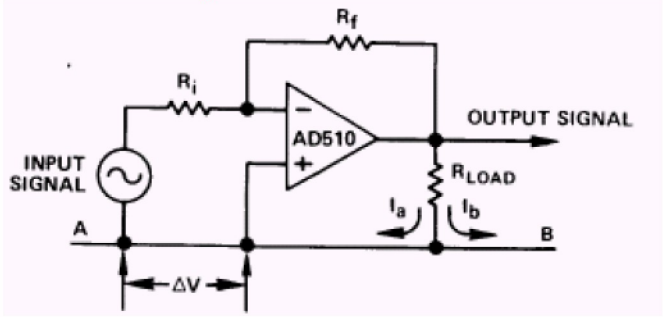
Очевидно, что напряжение ВЧ помехи, присутствующее на проводе отрицательного питания, будет передаваться на выход ОУ. Рекомендуемый метод борьбы с этой проблемой - развязка с помощью конденсатора, установленного **ВБЛИЗИ** вывода питания. Имеет значение и подключение второго вывода этого конденсатора. Рекомендуется подключать к общему выводу нагрузки.



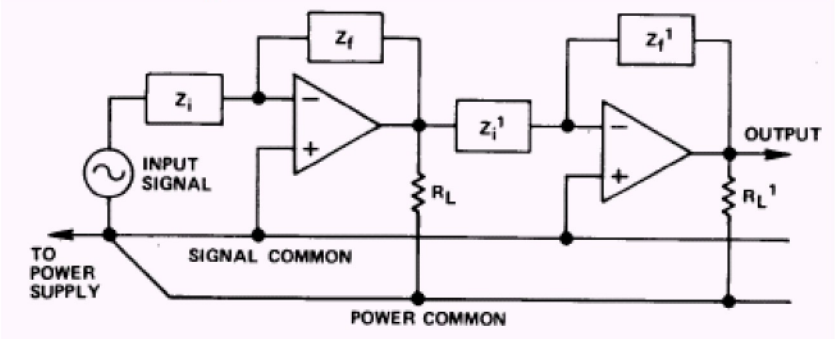
В более сложных случаях следует выбирать такой способ подключения развязывающего конденсатора, который замыкает петлю возвратного тока без участия общего провода (с возможно неоднородным потенциалом).



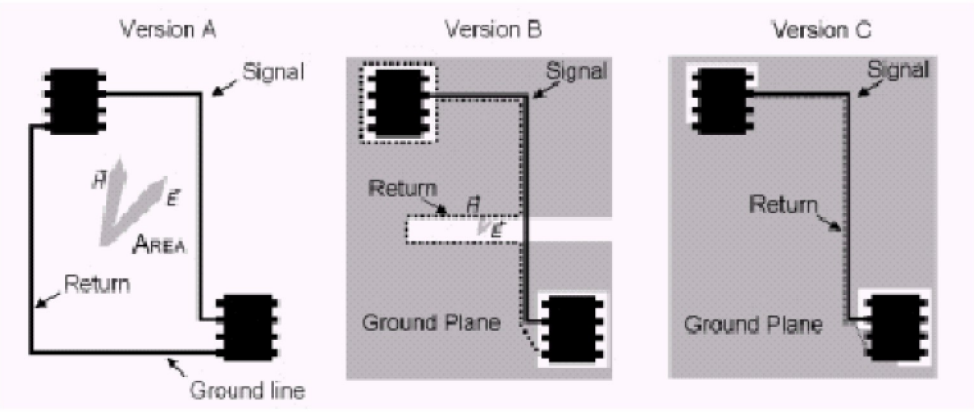
Похожие проблемы могут быть вызваны протеканием тока сигнала по одному проводу с силовой землей. Здесь возможен вклад помехи и положительная ОС через ненулевое сопротивление участка общего провода. Кроме этого наличие дополнительного сопротивления, через которое протекает ток сигнала, вызывает ошибки усиления.



Решением является организация раздельной силовой земли и сигнальной. Здесь исключается взаимное влияние больших выходных токов и малых сигнальных.



При использовании печатных плат с разводкой на одной или двух сторонах следует также располагать сигнальный и возвратный проводники таким образом, чтобы площадь образуемого контура была минимальной. Также следует избегать избыточной длины проводников. Одиночный проводник является антенной, начиная с 1/20 длины волны, максимум достигается при 1/4 длины волны.



Для петлевых антенн наилучшая эффективность приема достигается при равенстве плеча полуволне сигнала. При использовании многослойных (включая двухслойные) печатных плат рекомендуется выделять один слой для общего провода, что облегчает выполнение указанного правила.

Здесь на рис. А имеется петлевая антенна, на рис. В имеется нарушение слоя общего провода, приводящее к формированию приемной петли, рисунок С - оптимальный. Основными соображением при разводке таких плат являются:

- Постоянный ток течет, в основном, по пути наименьшего сопротивления, что соответствует кратчайшему расстоянию;
- Переменный ток (ВЧ) течет по пути наименьшей индуктивности, что соответствует минимальной площади контура. Это означает, что возвратный ток будет течь практически под сигнальным проводником уже на частоте свыше 1-2 МГц.

2.1.2 Особенности разводки ВЧ схем

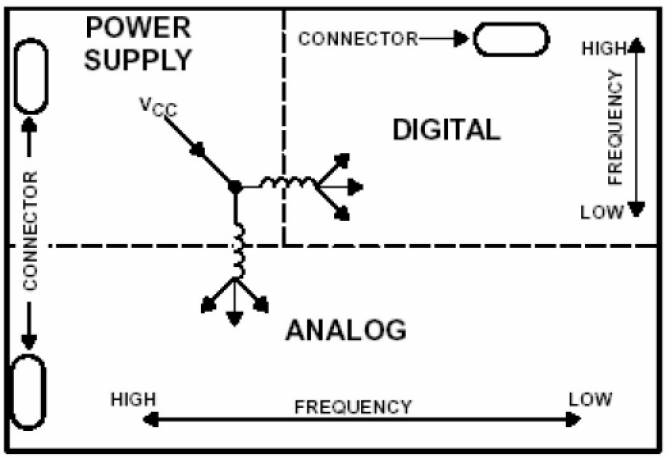
По сравнению с одно- и двухслойными печатными платами, многослойные обладают рядом преимуществ:

- Для питания и общего провода принято выделять отдельные слои, что облегчает разводку сигнальных и других цепей;
- Между слоями общего провода и питания имеется распределенная емкость, что уменьшает ВЧ шум;
- Улучшается помехоустойчивость к радиочастотным наводкам, благодаря близости заземленного слоя;

При выборе порядка расположения слоев принято отдавать предпочтение внутреннему расположению слоев общего провода и питания. Во-первых, это облегчает доступ к сигнальным слоям, во-вторых, уменьшает число переходов и вырезов во внешних слоях. При выборе структуры с более, чем 4-мя слоями, рекомендуется помещать сигнальные цепи между слоями общего провода и питания.

При разработке устройств, содержащих аналоговую и цифровую части рекомендуется организовывать отдельное заземление этих частей, что не всегда удобно в плане разводки. Исключением являются АЦП и ЦАП - их следует присоединять к аналоговой части общего провода, несмотря на наличие цифровой части.

Не рекомендуется располагать аналоговые и цифровые цепи со взаимным перекрытием, что может вызвать нежелательное взаимодействие через образующиеся емкости.



При проектировании многомодульных конструкций следует по возможности предусматривать разные контакты соединителей для разных цепей возврата, соединяя эти цепи в одном месте, возможно ближе к системной земле.

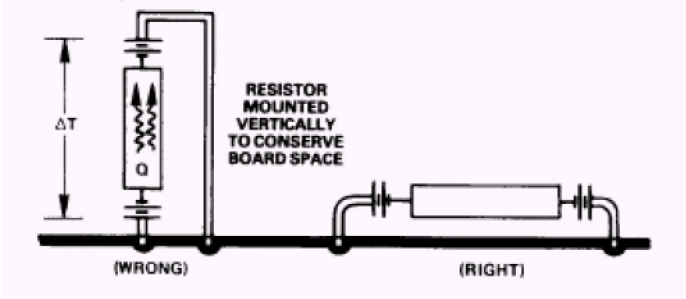
Рекомендуется по возможности располагать аналоговые цепи ближе к разъемам с целью уменьшения длины проводников.

2.2 Выбор пассивных компонентов

2.2.1 Резисторы

В основном, используются три типа резисторов: проволочные, угольно-композитные и пленочные. Следует помнить о наличии паразитной индуктивности и емкости. Считается, что пленочные резисторы сопротивлением менее 2К пригодны для ВЧ схем.

Следует учитывать также температурную зависимость сопротивления, например, угольные резисторы имеют на порядок более высокий ТКР, чем пленочные. Кроме ТКР, в точных схемах влияние могут иметь термоэлектрические эффекты, связанные с контактом разнородных материалов в местах крепления выводов. При этом неудачное расположение резистора может привести к тому что выводы будут иметь разную температуру, что приведет к появлению термоЭДС.



Высокоомные резисторы имеют нелинейную зависимость сопротивления от приложенного напряжения. Отказ резисторов также может иметь различные последствия. Например, пленочные угольные резисторы при отказе чаще всего разрывают цепь, а угольно-пленочные образуют короткое замыкание. Имеет место также старение резисторов со скоростью 50-75 ppm в год.

2.2.2 Конденсаторы

Конденсаторы также имеют паразитную индуктивность (особенно электролитические) и сопротивление, что должно быть учтено. Конденсаторы имеют неприятный эффект, называемый диэлектрической абсорбцией, который приводит к тому, что после заряда и последующего кратковременного разряда, на обкладках конденсатора появляется часть первичного напряжения. Этот эффект приводит к ошибкам УВХ и нелинейности различных преобразователей, возникают трудности с балансировкой интеграторов. Наименьшей абсорбцией обладают тефлоновые и полистиреновые конденсаторы (0.002-0.02%), наибольшей - электролитические (до 10%). Наличие утечки приводит к саморазряду конденсаторов. Наименьшей утечкой обладают также полимерные конденсаторы. Изменения температуры приводят к изменениям емкости, (до 10000ppm/°C у электролитических конденсаторов), кроме того температурный диапазон может быть ограничен. Например, электролитические распространены конденсаторы в двух исполнениях: 85°C и 105°C. Тефлоновые конденсаторы выдерживают температуру до 200°C.

В настоящее время от разработчика цифровых устройств требуется не только знание основ алгебры логики, принципов действия основных узлов цифровой техники и владение классическими принципами синтеза комбинационных и последовательностных автоматов, но и владение современной элементной базой, включая современные методы проектирования. Технологии разработки цифровой аппаратуры, преобладающие в XX веке, не могут удовлетворить требованиям XXI века по ряду причин, главные из которых:

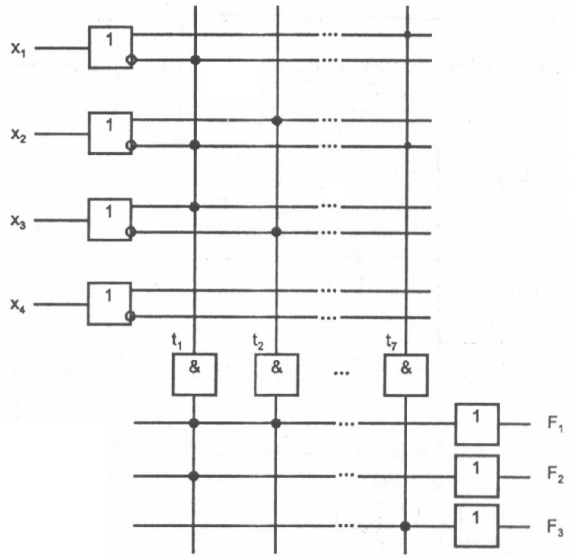
- Недостаточная гибкость, значительный интервал времени от начала разработки до выхода готовой продукции;
- Трудности с реализацией высокоскоростных интерфейсов, обусловленные физическими ограничениями печатных плат.

Решением этих проблем является применение программируемых логических интегральных схем (ПЛИС), представляющих собой набор конфигурируемых логических блоков, из которых разработчик может создать разнообразные структуры, не изменяя печатной платы устройства. Разработка цифрового устройства на базе ПЛИС сводится к разработке принципиальной схемы или текстового описания на специальном языке описания аппаратуры и оптимизации структуры под конкретные требования.

Рынок современных ПЛИС достаточно широк, можно перечислить такие фирмы, как XILINX, ALTERA, ACTEL, INTEL, LATTICE, ATMEL.

ПЛМ (PLA)

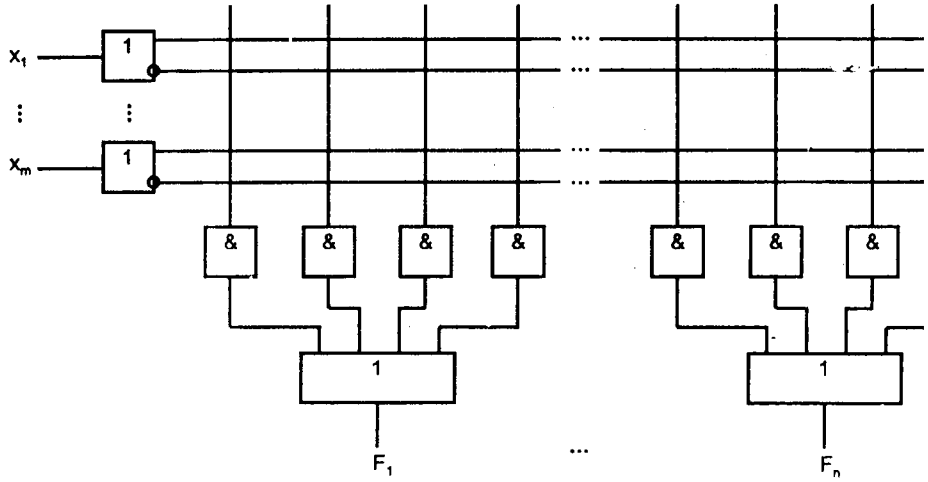
Развитие ПЛИС началось в 70-е годы с появления программируемых логических матриц (ПЛМ) (PLA). В состав ПЛМ входят программируемые матрицы И и ИЛИ, а также блоки входных и выходных буферов. В ПЛМ обычно входные сигналы подаются на программируемую матрицу И, на выходе которой формируется некоторое количество логических произведений (термов), состоящих из входных переменных в прямой либо инверсной форме. Полученные термы подаются на вход матрицы ИЛИ, которая формирует набор выходных функций. Отечественная промышленность выпускала ПЛМ К556РТ1,2



ПМЛ (PAL)

Практика использования таких ПЛМ показала, что часто матрица ИЛИ недоиспользуется, т.е. практические задачи не требуют использования любыми дизъюнкторами выходов любых конъюнкторов. Отказ от этой возможности привел к разработке программируемой матричной логики (ПМЛ). В ПМЛ выходы элементов И жестко распределены между элементами ИЛИ, что снижает функциональную гибкость, но изготовление и использование упрощаются. В структурах ПЛМ и ПМЛ также используются программируемые выходные буферы, позволяющие получить инверсию логической функции, а также схемы с

Z-состоянием на выходе. Отечественные ПМЛ представлены ИМС КР1556ХЛ8, ХП4, ХП6, ХП8. ИМС типа ХП обладают также элементами памяти - триггерами.



БМК

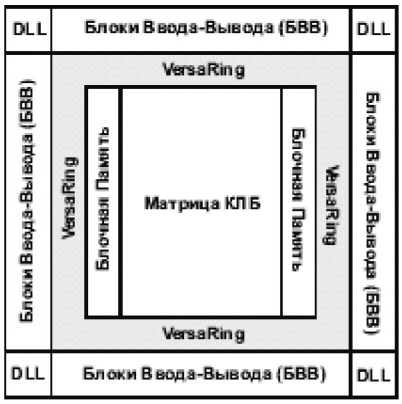
Кроме ПЛМ и ПМЛ к ПЛИС относятся также базовые матричные кристаллы (БМК). Различают БМК канального, бесканального и блочного типов. По канальной архитектуре строились БМК первого поколения, состоящие из набора базовых ячеек (БЯ), между которыми располагались свободные зоны для создания соединений (каналы). Первые БМК использовались в полузаказных БИС и программирование производилось масочным способом. Канальные БМК выпускались в основном на основе биполярной технологии, которая сама по себе ограничивает степень интеграции и их недостатком является необходимость резервирования значительной площади для каналов. Поиск путей повышения интеграции привел к бесканальным БМК. В кристаллах бесканальных БМК отсутствуют специальные каналы и соединения между отдельными БЯ могут выполняться в любом месте кристалла. Бесканальные БМК характерны для КМДП технологии и выпускаются в вариантах "море вентилей" и море "транзисторов". Блочные БМК содержат на кристалле несколько блоков-подматриц, каждый из которых представляет собой БМК меньшей размерности. Применение блочной архитектуры БМК облегчает создание комбинированных устройств, содержащих как блоки логической обработки, так и память. Отечественные БМК представлены ИМС К1520ХМ6 (канальная ЭСЛ), ТРАП-50 (канальная ТТЛШ), ТИТУЛ-30 (море вентилей КМДП).

FPGA

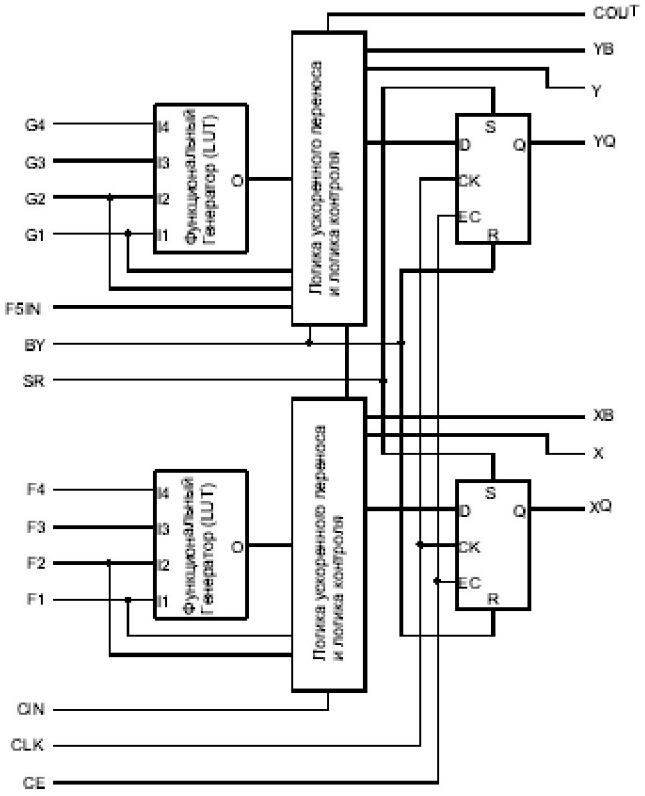
На основе рассмотренных ПЛМ, ПМЛ и БМК дальнейшее развитие ПЛИС двигалось по двум направлениям: CPLD (Complex Programmable Logic Devices) и FPGA (Field Programmable Gate Arrays). Стремление объединить достоинства обеих линий привело к слиянию архитектур, для которого название не устоялось (FLEX - название фирмы ALTERA).

FPGA похожи на канальные БМК, так как состоят из набора идентичных конфигурируемых логических блоков (ЛБ), между которыми проходят трассировочные каналы. Формирование выходных сигналов осуществляется блоками ввода-вывода, расположенными на периферии кристалла. Возможности FPGA определяются характером ЛБ и системой межсоединений. Различают мелкозернистые ЛБ (пары транзисторов, логические вентили) и крупнозернистые (мультиплексоры и специальные логические преобразователи). Естественно, использование крупнозернистых ЛБ упрощает систему межблочных соединений, но при этом наблюдается недоиспользование элементов ЛБ. Блоки ввода-вывода обеспечивают Z-состояние с подключением резистора утечки на землю или питание, формирование "пологих фронтов", а также триггеры для фиксирования входного и выходного сигналов. Система межсоединений состоит из проводящих сегментов и программируемых

элементов (ключей). Оптимизация количества сегментов, их длины и количества ключей приводит к иерархической системе межсоединений с несколькими типами межсоединений для передачи на различные расстояния. В качестве перемычек или "антиперемычек" используются электронные ключи, управляемые триггерами конфигурации, ЛИЗМОП - транзисторы и структуры ONO (оксид-нитрид-оксид). В соответствии с идеей иерархических связей различают связи общего назначения, длинные связи, прямые связи, линии тактирования.



Структура SPARTAN-II



Структура конфигурируемого логического блока SPARTAN-II

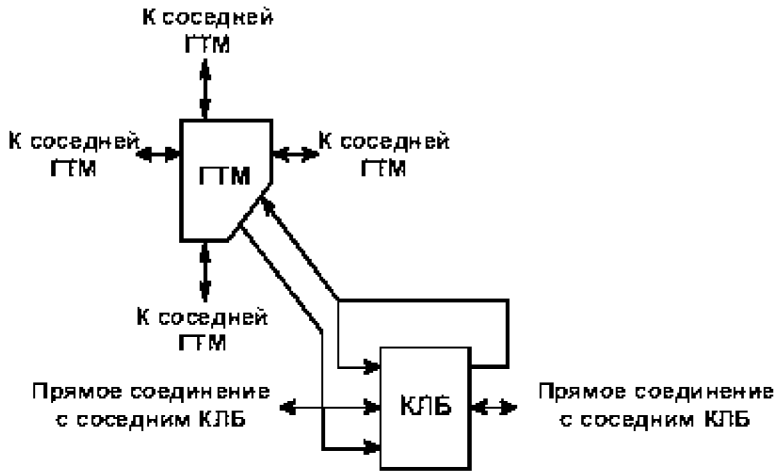
Функциональный генератор (Look-up Table) представляет собой 4-х входовых таблиц преобразования (Look-Up Table - LUT). Кроме использования в качестве функциональных генераторов, каждый LUT-элемент может быть также использован как синхронная память типа RAM размерностью 16x1 бит. Более того, из двух LUT-элементов в рамках одной секции можно реализовать синхронную RAM-память размерностью 16x2 бита или 32x1 бит, либо двухпортовую синхронную RAM-память размерностью 16x1 бит. В более поздних разработках идея сохранена, однако, увеличено количество и разрядность модулей и блоков.

Логические ячейки соединяются между собой с помощью локальных трассировочных ресурсов, трассировочных ресурсов общего назначения, трассировочных ресурсов блоков ввода-вывода, специальных трассировочных ресурсов и глобальных трассировочных ресурсов.

Локальные трассировочные ресурсы

Локальные трассировочные ресурсы позволяют реализовать три типа соединений:

1. Связи между таблицами преобразования (LUT), триггерами и главной трассировочной матрицей – ГТМ (GRM)
2. Внутренние обратные связи КЛБ (CLB), которые создают высокоскоростные связи с таблицами преобразования (LUT) в рамках одного КЛБ (CLB), и позволяют соединять их в виде цепочек с минимальными задержками распространения сигналов
3. Прямые трассы, которые создают высокоскоростные соединения с соседними по горизонтали КЛБ, избегая при этом больших задержек, присущих трассам главной трассировочной матрицы



Трассировочные ресурсы общего назначения

Большинство связей в кристаллах Spartan-II реализуются с помощью трассировочных ресурсов общего назначения и, следовательно, большая часть ресурсов межсоединений связана с этим типом трассировочной иерархии. Трассировочные ресурсы общего назначения расположены в виде горизонтальных и вертикальных трассировочных каналов и размещены в непосредственной близости от строк и столбцов матрицы, образованной блоками КЛБ. Ниже перечислены эти ресурсы:

• Примыкающая к каждому КЛБ главная трассировочная матрица – ГТМ (General Routing Matrix - GRM). ГТМ– это матрица переключателей, с помощью которых коммутируются горизонтальные и вертикальные трассы и посредством которых блоки КЛБ получают доступ к трассировочным ресурсам общего назначения

• ГТМ связана в каждом из четырех направлений с соседней ГТМ посредством 24-х трасс одинарной длины

• 96 буферизованных HEX-линий трассируют ГТМ сигналы к шести другим ГТМ в каждом из четырех направлений. HEX-линии организованны в виде зигзагообразных линий. HEX-линии могут подключаться к источникам сигнала только в своих конечных точках или серединных (три блока от источника). Одна третья часть HEX-линий является двунаправленными, в то время как остальные – однонаправленные.

• 12 длинных линий (Long lines) являются буферизированными, двунаправленными линиями, распространяющими сигналы в микросхеме быстро и эффективно. Вертикальные длинные линии имеют протяженность равную полной высоте кристалла, а горизонтальные длинные линии – полной ширине.

Трассировочные ресурсы для блоков ввода-вывода

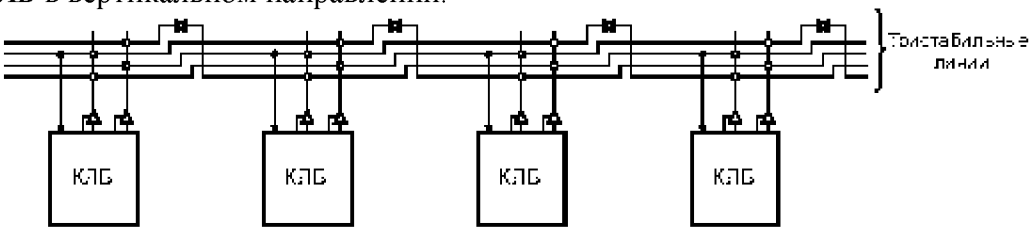
Кристалл Spartan-II имеет дополнительные трассировочные ресурсы, расположенные по периферии всей микросхемы. Эти трассировочные ресурсы формируют добавочный интерфейс между блоками КЛБ и блоками БВВ. Эти дополнительные ресурсы, называемые VersaRing, улучшают возможности закрепления сигналов за контактами и переназначения уже сделанного закрепления, если это требование накладывается расположением проводников на печатной плате. При этом сокращается время изготовления всего проекта, т. к. изготовление и проектирование печатной платы можно выполнять одновременно с проектированием FPGA.

Специальные трассировочные ресурсы

Некоторые классы сигналов требуют наличия специальных трассировочных ресурсов для максимизации быстродействия. В устройстве Spartan-II специальные трассировочные ресурсы создавались для двух классов сигналов:

• Горизонтальные трассировочные ресурсы создавались для реализации в микросхеме шин с тремя состояниями. Четыре разделенные линии шин реализованы для каждой строки КЛБ, позволяя организовывать сразу несколько шин в пределах одной строки. (см. рис. 7)

• Две специальные линии для распространения сигналов быстрого переноса к прилегающему КЛБ в вертикальном направлении.



Подключение буферов к горизонтальным линиям

Глобальные трассировочные ресурсы

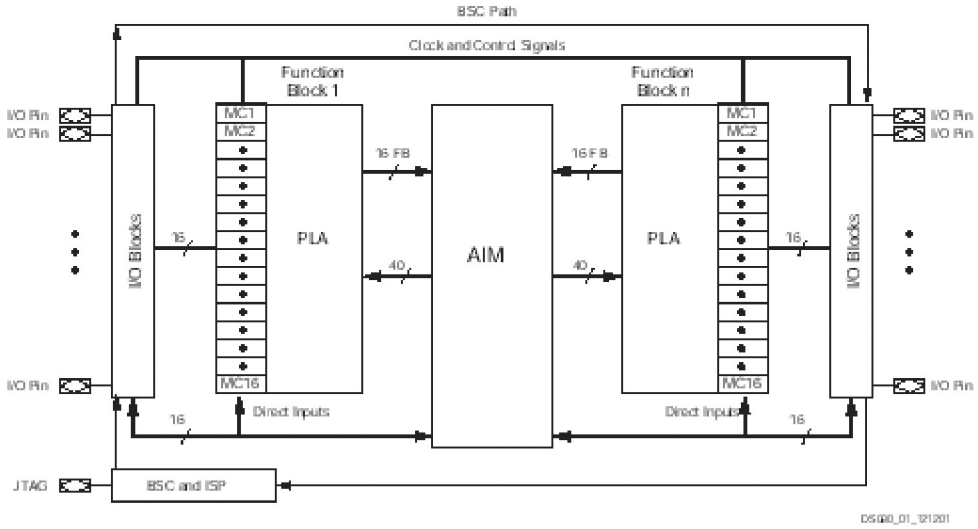
Глобальные трассировочные ресурсы распределяют тактовые сигналы и другие сигналы с большим коэффициентом разветвления по выходу на всем пространстве кристалла. Кристалл Spartan-II имеет два типа глобальных трассировочных ресурсов, называемых соответственно первичными и вторичными.

• Первичные глобальные трассировочные ресурсы представляют собой четыре специальные глобальные сети со специально выделенными входными контактами и связанными с ними глобальными буферами, спроектированными для распределения сигналов синхронизации с высоким коэффициентом разветвления и с минимальными разбегами фронтов. Каждая такая сеть может быть нагружена на входы синхронизации всех КЛБ, БВВ и BlockRAM – блоков микросхемы. Источниками сигналов для этих сетей могут быть только глобальные буферы. Всего имеется четыре глобальных буфера – по одному для каждой глобальной сети.

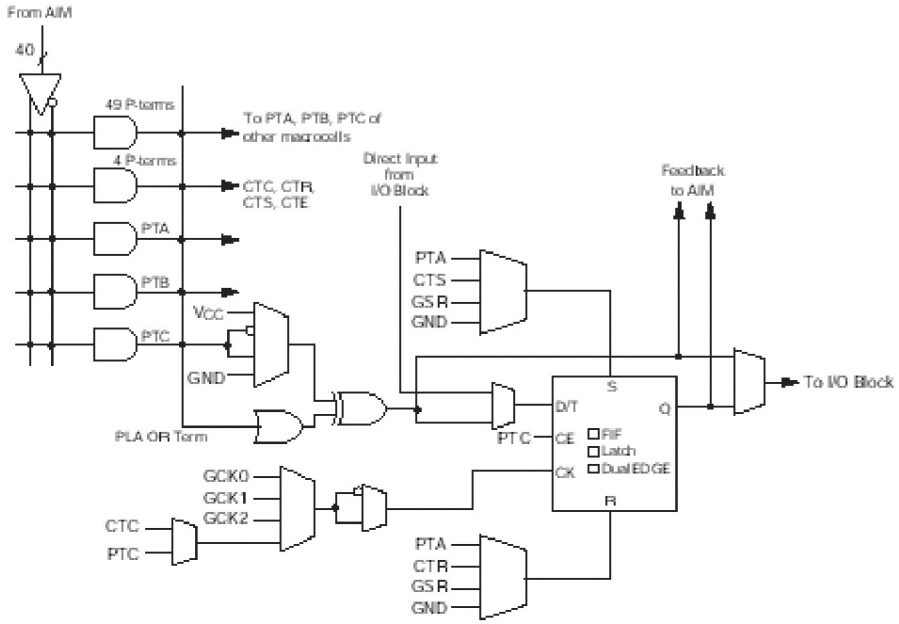
• Вторичные глобальные трассировочные ресурсы состоят из 24 магистральных линий, 12 вдоль верхней кромки кристалла и 12 вдоль нижней. По этим связям может быть распространено до 12-ти уникальных сигналов на колонку по 12 длинным линиям данной колонки. Вторичные ресурсы являются более гибкими, чем первичные, т.к. эти сигналы, в отличие от первичных, могут трассироваться не только до входов синхронизации.

CPLD

CPLD состоят из центральной коммутационной матрицы, множества функциональных блоков (макроячейки, макроэлементы) и блоков ввода-вывода на периферии кристалла. Функциональные блоки аналогичны ПЛМ, а система коммутации - непрерывная с идентичными связями, что позволяет довольно точно предсказывать задержку сигналов в процессе проектирования. Как правило, система коммутации позволяет соединять любой выход любого блока с любым входом любого блока (полная коммутируемость).



Структура CPLD CoolRunner



Структура макроячейки CPLD CoolRunner-II

Системы на кристалле

ПЛИС "Система на кристалле" предполагают наличие на кристалле специализированных областей (аппаратные ядра), реализующие определенные функции. Введение таких ядер снижает универсальность ПЛИС, но сокращает площадь кристалла при реализации сложных функций. В качестве аппаратных ядер используются блоки ОЗУ, умножители, ядра интерфейса JTAG, контроллеры PCI и др. Примеры - микросхемы FLEX фирмы ALTERA и VIRTEX фирмы XILINX. В настоящее время системы на кристалле (SoC) получили широкое распространение, причем на кристалле могут присутствовать и аналоговые и цифровые узлы различной степени интеграции, что позволяет практически на одном кристалле реализовать целое устройство, для которого ранее потребовалось бы несколько отдельных микросхем разного класса.

JTAG

Интерфейс JTAG - это совокупность средств и операций, позволяющая производить тестирование БИС без физического доступа к каждому ее выводу. Тестирование схемы по стандарту JTAG называется также периферийным сканированием. Для обеспечения периферийного сканирования в состав схемы должен входить набор ячеек периферийного

сканирования (BSC - Boundary Scan Cells) и схем управления их работой. Ячейки BSC располагаются между выводом устройства и внутренней схемой и работают в двух режимах - пропуска сигналов и тестирования. В первом режиме BSC просто пропускают сигнал, не влияя на него, во втором - объединяются в регистр, который может принимать и передавать данные. С помощью JTAG можно проверять функционирование самих микросхем, а также соединения между ними, для чего проверяемые трассы должны объединять ИМС с JTAG.

ПЛИС фирмы ALTERA

В настоящее время используются следующие семейства ПЛИС:

- MAX3000, MAXII – CPLD
- Семейства Cyclone, Stratix, Arria - FPGA

Stratix II — IV — мощные FPGA с числом вентилей до 800000, имеющие на кристалле встроенные множители, цифровые преимопередатчики.

Семейство Cyclone считается экономичным как с позиции энергопотребления, так и по цене, тем не менее имеющие до 150000 вентилей и способные решать широкий круг задач.

Семейство Arria – для телекоммуникационных применений.

Семейство MAX — экономичные CPLD,

ПЛИС фирмы XILINX

В настоящее время ПЛИС фирмы XILINX можно условно разделить на следующие семейства:

CPLD серий XC, CoolRunner XPLA3 и CoolRunner II – разнообразные конфигурации ;

FPGA Spartan-3,Spartan-5, Spartan-6 - FPGA с емкостью до 200000 логических ячеек

FPGA Virtex, Virtex-II, 4,6 - FPGA и системы на кристалле, включающие в себя ядра процессоров PowerPC, нацеленные на разработку систем цифровой обработки сигналов.

Список рекомендуемой литературы и интернет - источников

1. ARM Developer Suite. Assembler guide. Copyright 2000, 2001 ARM Limited
2. ARM7 TDMI Rev.3 Technical Reference Manual
3. Bird, B. M. and K. G. King. An Introduction To Power Electronics. John Wiley & Sons, 1983, pp. 250−281.
4. http://direct.xilinx.com/bvdocs/publications/DS054.pdf
5. http://direct.xilinx.com/bvdocs/publications/ds057.pdf
6. http://direct.xilinx.com/bvdocs/publications/ds090.pdf
7. http://direct.xilinx.com/bvdocs/publications/ds094.pdf
8. http://direct.xilinx.com/bvdocs/publications/ds099.pdf
9. http://direct.xilinx.com/bvdocs/userguides/ug130.pdf
10. http://www.actel.com
11. http://www.analog.com
12. http://www.digilentinc.com/Data/Products/S3BOARD/S3BOARD-rm.pdf
13. http://www.digilentinc.com/Data/Products/S3BOARD/S3BOARD-sch.pdf
14. http://www.irf.com
15. http://www.linear.com
16. http://www.national.com
17. http://www.nxp.com
18. http://www.ti.com
19. http://www.xilinx.com/support/sw_manuals/xilinx7/download/cgd.zip
20. http://www.xilinx.com/support/sw_manuals/xilinx7/download/dev.zip
21. http://www.xilinx.com/support/sw_manuals/xilinx7/download/hdl.zip
22. http://www.xilinx.com/support/sw_manuals/xilinx7/download/qst.zip
23. http://www.xilinx.com/support/sw_manuals/xilinx7/download/sim.zip
24. Spartan-3 Starter Kit Board User Guide. 2004, Xilinx
25. UM10139 Volume 1: LPC214X User Manual Rev.01 – 15 August 2005.
26. А. Солонина, Д. Улахович, Л. Яковлев Алгоритмы и процессоры ЦОС. - СПб.: БХВ-Петербург, 2002, 464 с
27. Айфичер Э.С., Джервис Б.У. Цифровая обработка сигналов: практический подход, 2-е издание, пер. с англ. – М.: Изд. Дом «Вильямс», 2004 – 992 с.
28. Безуглов Д.А., Калиенко И.В. Цифровые устройства и микропроцессоры. -Ростов-на-Дону,: Феникс, 2006, 468с. ISBN 5-222-08211-3
29. В.В. Соловьев Проектирование цифровых систем на основе ПЛИС. - М.:Горячая линия – Телеком, 2001 – 636с.
30. В.Ю. Зотов Проектирование цифровых устройств на основе ПЛИС фирмы XILINX в САПР WebPack ISE. - М.:Горячая линия – Телеком, 2003 – 624с.
31. Дикарев В.Н., Илюхин И.М. Оптические и оптико-электронные приборы угловой ориентации космических кораблей, Ч.1 и Ч.2. М.: Изд. МГТУ, 2000.

32. Зотов В.Ю. Проектирование встраиваемых микропроцессорных систем на основе ПЛИС фирмы XILINX. М.: Горячая линия – Телеком, 2006 - 520с.
33. И.Е. Тарасов Разработка цифровых устройств на основе ПЛИС XILINX с применением языка VHDL. - М.:Горячая линия – Телеком, 2005 – 252с.
34. Кловский Д.Д., Шилкин В.А. Теория электрической связи. Сб. задач и упражнений: Учеб. Пособие для вузов. – М.: Радио и связь, 1990 - 280с.
35. Ориентация и навигация подвижных объектов: современные информационные технологии. / под ред Б.С. Алёшина, К.К. Веремеенко, А.И. Черноморского. – М.: ФИЗМАТЛИТ, 2006 – 424с.
36. П.Н. Бибило Основы языка VHDL. – М.:СОЛОН-Р, 2000 – 200с.
37. Прокис Джон. Цифровая связь. Пер. с англ. /под ред. Д.Д. Кловского. М.: Радио и связь, 2000 – 800с.
38. Разработка программного обеспечения микроконтроллеров семейства MSP430: Методические указания для выполнения лабораторной работы / Самарский гос. аэрокосмический ун-т. Сост. И.А. Кудрявцев, Д.В. Корнилин. Самара 2010 20стр.
39. Разработка цифровых устройств на базе ПЛИС: Методические указания для выполнения лабораторных работ / Самарский гос. аэрокосмический ун-т. Сост. И.А. Кудрявцев. Самара 2007 28стр.
40. Сергиенко А.М. VHDL для проектирования вычислительных устройств. Киев: ЧП «Корнейчук», ООО «ТИД «ДС», 2003.
41. Скляр Бернард. Цифровая связь. Теоретические основы и практическое применение. Изд. 2-е, испр.: пер. с англ. – М.: Изд. Дом «Вильямс», 2003 - 1104 с.
42. В.В. Соловьев Проектирование цифровых систем на основе ПЛИС. - М.:Горячая линия – Телеком, 2001 – 636с
43. И.Е. Тарасов Разработка цифровых устройств на основе ПЛИС XILINX с применением языка VHDL. - М.:Горячая линия – Телеком, 2005 – 252с
44. Тревор Мартин Микроконтроллеры ARM7. Семейство LPC2000 компании Philips. Вводный курс. – М.:Издательский дом «Додэка-XXI», 2006. – 240с.
45. Федосеев, В.И., Колосов, М.П. Оптико-электронные приборы ориентации и навигации космических аппаратов. Москва : Логос, 2007. стр. 248.
46. Черемухин Г.С. Приборы ориентации на Солнце. М.: Техпромиздат, 1998.