СОДЕРЖАНИЕ

1 Обзор литературы

Прежде чем писать обзор литературы, необходимо четко сформулировать тему и цель курсового проекта. Назначение обзора, в первую очередь, заключается в описании того, что было сделано по заданной теме к моменту проведения курсового проектирования и, соответственно в определении диагностических методов исследований или методов терапевтического воздействия, а также путей их технической реализации. Более того, в обзоре литературы по возможности должна быть обоснована необходимость проведения технической разработки. То есть нужно показать, что разработка данного медицинского аппарата актуальна и перспективна, поскольку до настоящего момента не проводилась или проводилась в недостаточном объеме.

1.1 Разработка плана обзора литературы

Исходя из темы и цели курсового проекта, необходимо составить план литературного обзора. Составление плана позволяет четко представить себе, какую именно литературу нужно собрать. Благодаря этому впоследствии удобно будет работать над каждым пунктом обзора отдельно. План должен быть изложен максимально подробно. В соответствии с планом в дальнейшем можно разбить обзор на параграфы, что, во-первых, упростит его восприятие, во-вторых, позволит создать некую внутреннюю логику обзора, в которой отразится его идея. Наличие плана значительно улучшит впечатление от работы.

Прежде чем составлять план обзора, помимо выработки основной идеи необходимо хотя бы в общих чертах представлять, какая литература имеется в наличии, то есть необходимо учесть реальность выполнения плана. Зачастую этому способствует чтение других обзоров по изучаемой теме. Нередко в процессе работы план может видоизменяться в результате добавления новых частей или удаления тех, что не несут смысловой нагрузки или не выполнены из-за отсутствия нужной литературы. Однако все время необходимо помнить, что план должен отражать основную идею литературного обзора.

План обзора литературы обсуждается с руководителем курсового проектирования. Это позволяет не только выработать наилучший вариант плана, но и значительно уменьшить объем исправлений в процессе сдачи курсового проекта. Если руководитель хочет внести в план дополнительный пункт, литературу по которому Вы не можете найти, необходимо уточнить, к каким источникам следует для этого обратиться.

В общем виде план обзора будет включать три основных пункта:

1 Общая характеристика заболевания и патологических процессов, протекающих в организме человека для лечения или диагностики которых производится разработка прибора или аппарата.

2 Конкретные медицинские методики (последовательность действий, режимы и параметры стимулирующих воздействий, методики диагностики и контроля), используемые при указанном заболевании.

3 Технические характеристики и режимы работы уже выпускаемых приборов и аппаратов для диагностики или лечения указанного заболевания. По возможности следует рассмотреть структурные и электрические принципиальные схемы, интерфейсы взаимодействия с пользователем, сравнить преимущества и недостатки существующих устройств.

В конце обзора делается заключение, в котором указывается методика, которая будет реализована в приборе, а также основные технические характеристики или режимы работы прибора. На основании этой информации производится анализ технического задания.

1.2 Сбор литературы

Сбор литературы – это важнейший этап выполнения литературного обзора, который равен по своей значимости половине всей работы. Однако в отличие от написания обзора, которое требует значительной затраты времени, основная работа по сбору литературы, как правило, выполняется в короткие сроки. Особенно это актуально для иногородних студентов, не имеющих возможности долго находиться в Минске. Приведенный ниже алгоритм действий, поможет сократить время выполнения этой части работы без потери качества ее исполнения.

Поиск литературы в электронном каталоге.

Сформулируйте термины, по которым Вы хотите найти литературу. Помните, что при формулировании слишком узких понятий Вы рискуете не найти достаточного количества литературы. Это часто случается при поиске информации о конкретных схемах устройств или малоизвестных методах лечения заболеваний. В этом случае необходимо попытаться найти информацию о группе, к которой относится это устройство или метод. Укажите все известные синонимы интересующего Вас термина.

Например, необходимо собрать литературу по теме «Многоканальная программируемая электростимуляция мышц». Для поиска информации можно использовать Интернет поисковики Google.com, yandex.ru, rambler.ru и т.д. Поиск в Интернете будет включать следующий примерный перечень терминов:

- «электростимуляция»,
- «электростимуляция мышц»,
- «многоканальная электростимуляция»,
- «методики многоканальной электростимуляции»,
- «аппарат электростимуляции»,
- «аппарат для многоканальной электростимуляции» и т.д. так как при поиске, как правило, дополнительно используются термины из найденных источников.

Кроме того, информацию по заданной теме можно найти в Национальной библиотеке Беларуси (г. Минск, просп. Независимости, 116), Республиканской научной медицинской библиотеке (г. Минск, ул. Фабрициуса, 28), Республиканской научно-технической библиотеке, (г. Минск, пр-т Победителей, 7) и т.д.

Реальность такова, что почти все обзоры литературы создаются с использованием чужих литературных обзоров по заданной теме. Однако далеко не всегда находятся обзоры, в точности соответствующие теме, поэтому приходится использовать несколько исходных литературных обзоров. Кроме того, отдельные вопросы в этих обзорах могут быть не освещены вообще. Наконец, с момента написания предшествующих обзоров до момента выполнения Вами обзора литературы проходит значительное время, за которое выпускается новая литература, которая также может быть использована при создании обзора. Писать обзор литературы исключительно путем синтеза предшествующих обзоров – это mauvais ton («плохой тон», франц.). Чем больше в работе самостоятельно добытых данных, тем выше ее качество.

Чем больше исходных литературных обзоров будет использовано, тем полнее будет изложен материал и тем легче будет добиться того, чтобы литературный обзор не походил на предшествовавшие ему обзоры. Однако отбирать необходимо только те обзоры, в которых действительно содержится полезная информация. Кроме того, предпочтение нужно отдавать обзорам, содержащим более свежие ссылки. Особое внимание следует обращать на то, что новые литературные обзоры пишутся с использованием старых и значительная доля информации в них может дублироваться.

1.3 Написание обзора

Написание – наиболее сложный и длительный этап подготовки обзора.

Ниже изложен алгоритм действий, позволяющий выполнить эту работу с минимальными затратами времени.

Создайте файл Word, в котором будет в дальнейшем создаваться обзор. В этом файле напишите крупным шрифтом заголовки и подзаголовки, соответствующие плану обзора.

Скопируйте все отсканированные Вами обзоры в отдельные файлы и внимательно перечитайте их, разбейте на части таким образом, чтобы в каждой из них содержалась отдельная идея. Желательно, чтобы они соответствовали плану Вашего обзора. При этом весь ненужный материал нужно удалить, а материал, который может хоть как-нибудь пригодиться, необходимо сохранить и озаглавить.

Если ссылки на источники в тексте обзоров представлены в виде цифр, необходимо либо поменять их на фамилии, либо, выделив весь текст, изменить его цвет (кнопки «цвет шрифта») так, чтобы каждому обзору соответствовал свой цвет. Это делается для того, чтобы в дальнейшем можно было составить список литературы и расставить ссылки по ходу текста. Если ссылки представлены в виде «фамилия+год», проверьте, чтобы по ним можно было в дальнейшем найти источники в списке литературы.

Скопируйте из получившихся файлов литературу в файл с обзором и рассортируйте ее в соответствии с планом Вашего обзора. Скопируйте также дополнительный материал, собранный Вами из журналов и сборников, и распре-

делите по тексту обзора. Не забудьте, что каждый абзац должен содержать ссылку на источник.

В дальнейшем работа ведется отдельно с каждым параграфом обзора. Необходимо разбить каждый параграф на максимальное количество пунктов. При этом нужно следить, чтобы текст был связным и логичным как внутри параграфов, так и между ними и соответствовал общей идейной линии обзора. Для этого необходимо изменять формулировки, использовавшиеся во взятых Вами за основу обзорах. Кроме того, это необходимо сделать для того, чтобы Ваш обзор не создавал такого впечатления, что он списан с чужих обзоров. Также следует учитывать, что разные обзоры пишутся в разных стилях, тогда как ваш итоговый обзор должен быть написан в одном.

1.4 Проблемы, возникающие при выполнении обзора

При поиске как отечественной, так и зарубежной литературы обращайте внимание на используемые Вами поисковые слова: проблемы с поиском литературы могут быть обусловлены употреблением неправильных терминов.

По целому ряду направлений российскими разработчиками сделано гораздо меньше исследований, чем иностранными. Поэтому, если Вы не нашли отечественной литературы по теме, возможно, Вам удастся обнаружить иностранную литературу.

Если Вам все-таки не удалось обнаружить литературу по Вашей теме, необходимо скорректировать план обора исходя из реальных возможностей. При этом сложившаяся ситуация должна быть своевременно обсуждена с руководителем.

2 Анализ технического задания

В пункте «анализ технического задания» производится постановка задачи, выполняемой в курсовом проекте, выбор методов решения поставленной задачи. Содержание этого раздела должно быть достаточным для принятия конкретных технических решений и разработки структурной схемы.

Анализ технического задания (ТЗ) на проведение курсового проектирования (КП) включает следующие разделы:

1 Вводные данные.

1.1 Наименование КП.

1.2 Срок выполнения КП.

2 Цель выполнения КП.

3 Технические требования к изделию.

3.1 Состав изделия. Необходимо указать, из каких составных частей будет состоять разрабатываемый прибор. Данный пункт можно выполнить с разной глубиной детализации. Например для терапевтического прибора можно указать комплектность поставки: блок электростимулятора, система электродов, блок питания (при внешнем блоке питания). Можно сделать более детальный анализ

составных частей, рассмотрев отдельные составные части блока электростимуляции. Например в данной ситуации могут быть добавлены: блок управления, блок генерации стимулирующего сигнала, блок усиления, блок индикации, клавиатура.

3.2 Назначение прибора. Необходимо четко сформулировать, для чего будет применяться прибор. Назначение устройства непосредственно влияет на его технические особенности. При анализе этого пункта необходимо указать диапазоны значений рабочих характеристик устройства, учесть условия эксплуатации, обосновать эксплуатационно-технические требования к проектируемому устройству. К таким требованиям относятся:

– диапазон рабочих частот;

– характер передаваемой информации;

– мощность передающих устройств;

– чувствительность приемных устройств.

3.3 Функции прибора. В данном разделе необходимо рассмотреть последовательность действий, выполняемых прибором. Например для диагностического прибора можно выделить следующие функции:

– выполнение пользовательских настроек (выбор количества каналов регистрации данных, выбор длительности регистрируемых данных, выбор алгоритмов обработки данных и т.д.);

– выполнение диагностической процедуры, оцифровка и сохранение данных в памяти прибора;

– расчет выбранных диагностических показателей;

– отображение диагностических показателей;

– вывод информации на внешние носители или на персональный компьютер.

3.4 Метрологические характеристики (точность измерений, погрешности задания амплитуд стимулирующих сигналов и т.д.)

3.5 Требования к электропитанию (стационарный прибор – питание от сети 220 В; переносной – возможность питания от батарей; бортовой или автомобильный прибор – питание от аккумулятора автомобиля 12 – 13 В).

4 Этапы выполнения КП.

Все требования технического задания тщательно обосновываются (желательны ссылки на соответствующие ГОСТы, ТУ и другие нормативные документы, а также стандарты ISO, IEC, ETSI, CENELEC, EBU, рекомендации и отчеты ITU-R, ITU-T). При необходимости производится разработка дополнительных требований, которые в ряде случаев подтверждаются расчетами и ссылками на литературные источники. Приводятся характеристики входных и выходных сигналов, рассматривается возможность использования унифицированных узлов и блоков, применения интегральных схем и т.д.

3 Разработка структурной и функциональной схем устройства

3.1 Общие сведения

Электрическая структурная схема отражает основные функциональные части изделия (элементы, устройства, функциональные группы), их назначение и связи. Все функциональные части на схеме изображают в виде прямоугольников или условных графических обозначений (УГО). Если функциональных частей много, вместо наименований, типов и обозначений допускается проставлять порядковые номера справа от изображения или над ним, как правило, сверху вниз в направлении слева направо с их расшифровкой в таблице, помещаемой на схеме. На схеме даются поясняющие надписи, диаграммы, таблицы, указания параметров в характерных точках (величины токов, напряжений, формы и величины импульсов) (рисунок 3.1), математические зависимости и т.п.

На функциональной схеме изображают функциональные части изделия (элементы устройства и функциональные группы) и связи между ними с разъяснением последовательности процессов, протекающих в отдельных функциональных цепях изделия или в изделии в целом.

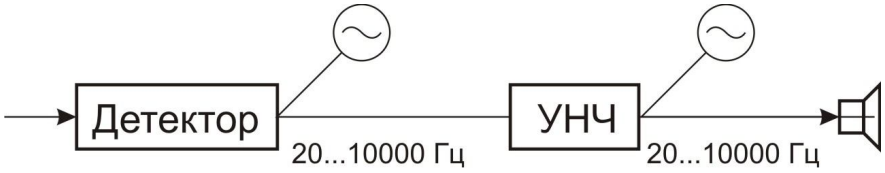


Рисунок 3.1 – Пример расположения параметров в характерных точках

Функциональные части схемы принято изображать в виде условных обозначений либо прямоугольников с указанием:

— позиционных обозначений функциональных групп, устройств, элементов, присвоенных им на принципиальной схеме, и их наименований;

— типов;

— обозначений документов, на основании которых функциональные части применены;

— технических характеристик функциональных частей;

— поясняющих надписей, диаграмм, таблиц, параметров в характерных точках.

Эти сведения приводятся выборочно в объеме, необходимом для наиболее полного и. наглядного представления о последовательности процессов, иллюстрируемых схемой. Наименования, типы и обозначения рекомендуется вписывать в прямоугольники.

3.2 Алгоритм создания схемы структурной

Разработка схемы структурной ведется в соответствии с пунктами 3.1 – 3.2. На основании выделенных составных частей и функций прибора определяются его структурные блоки.

Например, рассмотрим функции миостимулятора и выделим структурные блоки для их реализации:

№	Структурный блок прибора	Функция прибора
1	Блок управления	Управление работой аппарата
2	Клавиатура аппарата	Ввод параметров стимулирующих токов
3	Дисплей аппарата	Отображение текущего режима
4	Блок управления на основе микроконтроллера, или специальный блок генератора	Генерация стимулирующего сигнала
5	Блок преобразования уровня	Преобразование сигнала
6	Усилитель	Усиление сигнала
7	Электроды	Подача стимулирующего тока на пациента
8	Блок контроля амплитуды	Контроль амплитуды стимулирующего тока

Далее необходимо проанализировать направление связей в структурной схеме прибора, проанализировать уровни и частотные характеристики сигналов и начертить схему (рисунок 3.2)

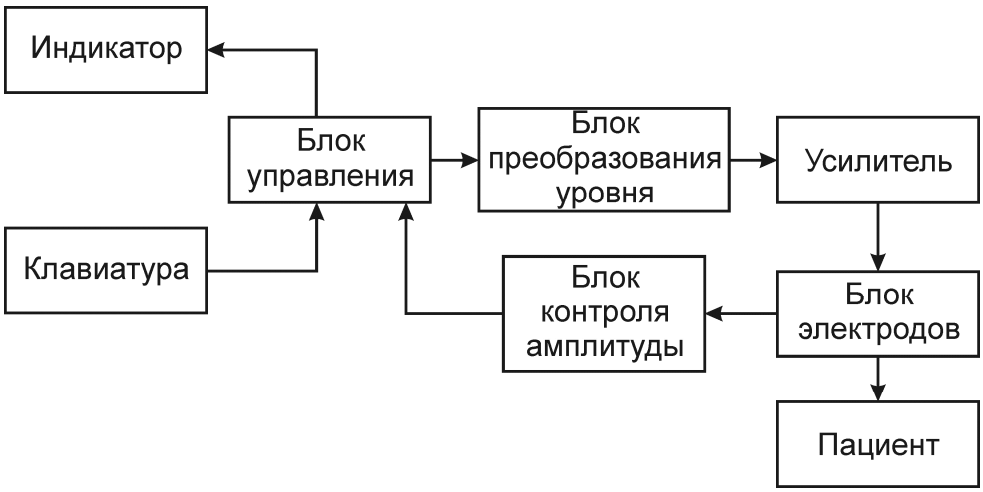


Рисунок 3.2 – Структурная схема

Дополнительно на схему можно нанести надписи, поясняющие работу схемы, в виде технических характеристик функциональных частей, диаграмм, таблиц, параметров сигналов в характерных точках и другой дополнительной информации, которая может помочь при защите курсового проекта.

4 Разработка схемы электрической принципиальной

4.1 Общие положения

Схема электрическая принципиальная – схема, определяющая полный состав элементов и связей между ними и, как правило, дающая детальное представление о принципах работы изделия. Данная схема отражает взаимные связи между отдельными электрическими компонентами, устройствами, аппаратами или приборами с учетом принципа действия и последовательности работы отдельных ее элементов. Схема электрическая принципиальная разрабатывается на основе технического задания, структурной или функциональной схемы устройства следующим образом:

1 Определяются отдельные функциональные узлы устройств (блок питания, генератор сигнала, усилитель сигнала, АЦП, ЦАП, датчик, устройство индикации, контроля и т.д.) и технические требования к ним;

2 Для каждого функционального узла выбирается электрическая схема и производится расчет последней в соответствии с техническими требованиями (напряжение питания, потребляемая мощность, форма и вид сигнала, частота сигнала, нагрузочная способность и т.д.);

3 Осуществляется согласование электрических характеристик (по уровню питания, амплитуде и частоте сигнала, входному и выходному сопротивлению и т.д.) рассчитанных схем функциональных узлов устройства;

4 В соответствии с параметрами полученной электрической схемы устройства выбирается подходящая элементная база (аналоговые и цифровые компоненты, устройства, элементы индикации, коммутации и т.д.);

5 В соответствии со стандартами ЕСКД оформляется графический материал схемы электрической принципиальной и перечень элементов.

4.2 Примеры разработки электрических схем функциональных узлов

4.2.1 Программируемый генератор синусоидального сигнала с цифровым потенциометром

При необходимости разработки генератора синусоидального сигнала часто встает вопрос о программном управлении частотой сигнала посредством цифрового интерфейса. Для этого могут использоваться схемы на основе цифровых потенциометров (ЦП), положение подвижного контакта которых определяется цифровым кодом, записанным в специализированные регистры последнего. Чаще всего используются три типа шин управления ЦП: SPI, I2C и UDC.

Цифровые потенциометры являются довольно универсальными устройствами и могут применяться в фильтрах и генераторах. В таких случаях разработчик должен всегда иметь в виду, что цифровые потенциометры работают в ограниченном частотном диапазоне и этот диапазон зависит от установленной величины сопротивления, поэтому необходимо внимательно сверяться с данными, опубликованными в техническом описании. Если требования по частоте

удовлетворяются, то такой потенциометр может быть с успехом использован для реализации генератора с установкой частоты посредством цифрового интерфейса.

На рисунке 4.1 показана схема генератора на основе моста Вина с цепью стабилизации амплитуды и с ЦП в цепи частотнозависимой обратной связи. Для аналоговых генераторов гармонических колебаний важной проблемой является автоматическая стабилизация амплитуды выходного напряжения. Если в схеме не предусмотрены устройства автоматической стабилизации, устойчивая работа генератора окажется невозможной. В этом случае после возникновения колебаний амплитуда выходного напряжения начнет постоянно увеличиваться, и это приведет к тому, что активный элемент генератора (например, операционный усилитель) войдет в режим насыщения. В результате напряжение на выходе будет отличаться от гармонического.

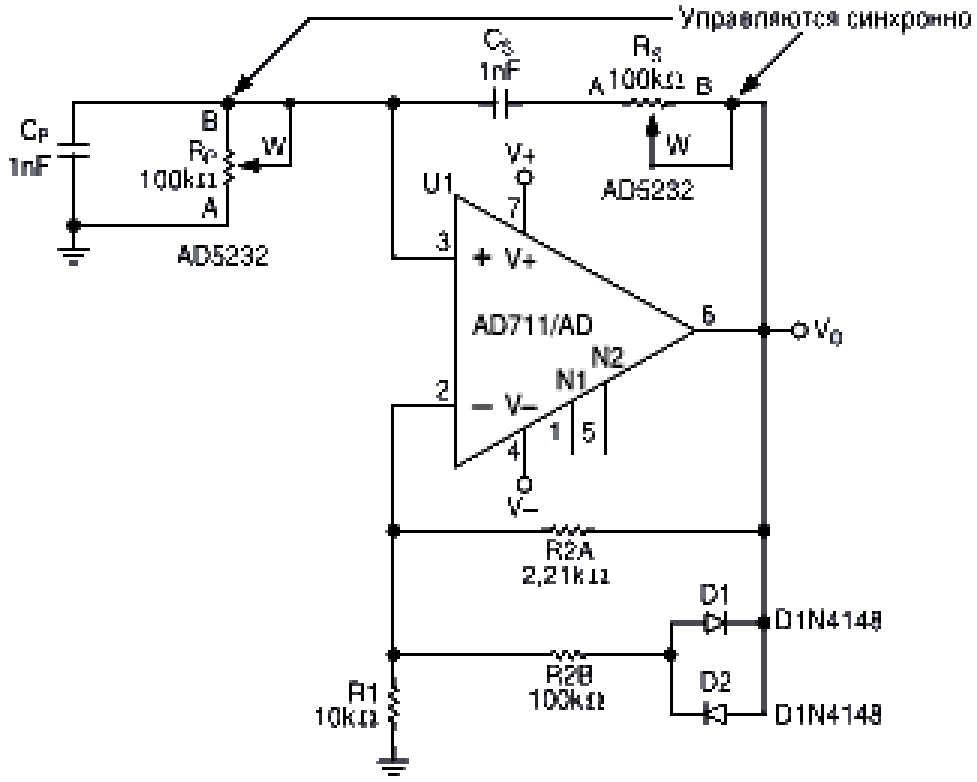


Рисунок 4.1 – Пример программируемого генератора с цифровым потенциометром

В представленной схеме самовозбуждение генератора (т.е. выполнение условия баланса амплитуд операционного усилителя) возможно только при выполнении следующего условия: операционный усилитель по неинвертирующему входу должен иметь коэффициент усиления K=3 на частоте резонанса моста Вина. Элементы R1 и R2 предназначены для получения требуемого коэффици-

ента усиления усилительного звена. Амплитуда стабилизируется за счет диодов D1 и D2, которые открываются при росте амплитуды выходного сигнала.

Частота генерируемого сигнала равна:

$$f = \frac{1}{2\pi * R * C},$$

Где $R=R_p=R_s$, $C=C_p=C_s$.

Сопротивление цифрового потенциометра, в регистр которого записана величина D, составляет

$$R_{AW} = \frac{2^N - D}{2^N} * R_{AB} + 50,$$

где R_{AB} – полное сопротивление ЦП, то есть сопротивление между выводами A и B, N – разрядность регистра ЦП, R_{AW} – сопротивление введенной части потенциометра между выводами A и W, 50 Ом – сопротивление подвижного контакта, то есть приблизительное значение сопротивления открытого транзисторного ключа ЦП.

Одна из возможных проблем, связанных с использованием цифровых потенциометров, заключается в проникновении шумов цифровых линий управления в аналоговые цепи. Такое может произойти при неудачной трассировке печатной платы, если дорожки цифровых сигналов расположены слишком близко к аналоговым. Поскольку обмен данными по цифровым шинам происходит, как правило, весьма активно, чувствительные аналоговые цепи будут испытывать постоянное воздействие шумов.

На рисунке 4.2 представлена схема, в которой ЦП используется для изменения коэффициента усиления операционного усилителя и в которой демонстрирует простое решение указанной проблемы. В данном примере потенциометр управляется по последовательному интерфейсу SPI, но не меньшее распространение имеет интерфейс I^2C. Идущие от контроллера сигналы SCLK и SDO шины SPI пропускают через логические элементы «2ИЛИ», благодаря которым цифровой потенциометр «видит» сигналы SCLK и SDO только при наличии низкого уровня на входе разрешения потенциометра (CS), т.е. тогда, когда выполняется перенастройка потенциометра.

Такое решение позволяет оградить аналоговые цепи от воздействия высокочастотных сигналов шины управления в течение всего времени, пока не требуется перенастройка цифрового потенциометра.

Таким образом, шумы перестают быть проблемой для схем, которые не нуждаются в частом перепрограммировании параметров, например, для схем начальной инициализации.

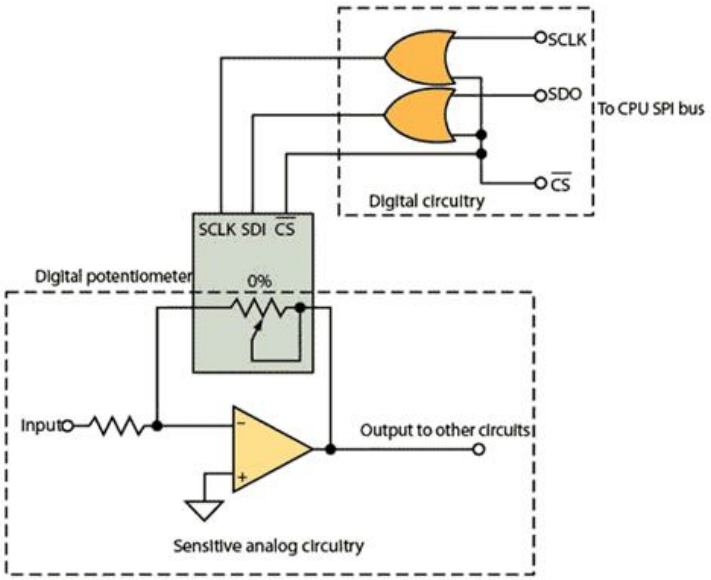


Рисунок 4.2 – Пример организации защиты от проникновения шумов цифровых линий управления в аналоговые цепи

4.2.2 Усилитель биоэлектрических потенциалов

Качество электронной системы съема, усиления и регистрации биопотенциалов во многом определяется параметрами входной цепи, образованной электродами отведений и резистивными суммирующими схемами, а также параметрами входных каскадов усиления.

Свойства входной цепи определяют степень подавления синфазных помех, вызванных наводками от силовой и осветительной сети и электрических установок. Неодинаковое для разных электродов изменение во времени сопротивления кожа–электрод совместно с нестабильностями электрических параметров электродов приводит к изменению коэффициента передачи входной цепи по постоянному току и искажению АЧХ системы электрод–вход усилителя биопотенциалов. При этом может происходить преобразование синфазной помехи в разностную, которая, складываясь с полезным сигналом, имеет аддитивный характер.

На входные цепи усилителей биоэлектрических сигналов наряду с воздействием синфазных и разностных помех от физических источников (сети питания, электросиловых приборов) воздействуют синфазные и разностные помехи биологического происхождения (артефакты). Снижение влияния синфазных биологических и физических помех достигается применением усилителей с достаточно большим коэффициентом подавления этих помех. Устранение влияния противофазных физических помех и наводок достигается уменьшением площади замкнутого контура, образованного проводами отведений и применением методов экранирования.

Проблема подавления противофазных помех биологического происхождения, вызванных работой других органов и систем биообъекта, кроме интересу-

ющих исследователя, более сложна и требует применения систем фильтрации и специальных методик регистрации сигналов.

В качестве усилителей биоэлектрических сигналов широко используются усилители постоянного тока с непосредственными связями. Входные каскады усилителей выполняются по симметричным дифференциальным схемам, обеспечивающим высокий уровень подавления синфазных помех.

Для уменьшения влияния синфазных помех применяются специальные компенсационные схемы. Ввиду сравнительно малого сопротивления тканей биообъекта величина синфазной помехи на всей его поверхности практически одинакова. Напряжение синфазной помехи снимается с одного или нескольких активных электродов и подается на инвертирующий усилитель, выход которого подключается через индифферентный электрод между биообъектом и общим проводом.

На рисунке 4.3 показана типовая схема усилителя биопотенциалов со схемой компенсации синфазной помехи на базе инструментального усилителя AD620 (схема рекомендована производителем Analog Devices).

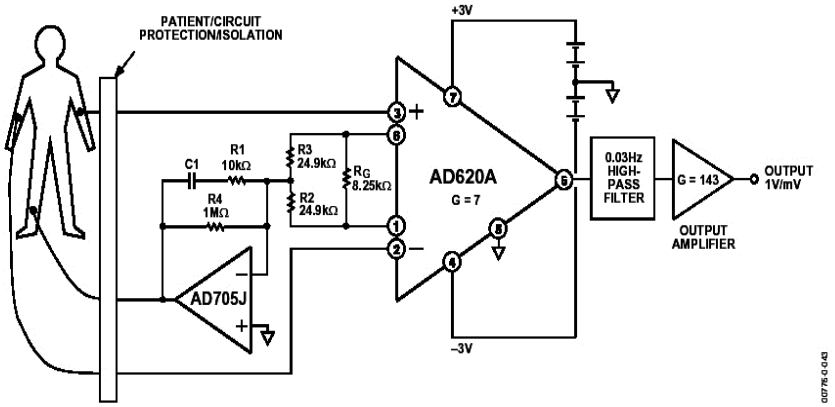


Figure 41. A Medical ECG Monitor Circuit

Рисунок 4.3 – Усилитель биопотенциалов со схемой компенсации синфазной помехи

В качестве инструментального усилителя применен AD620 – недорогой усилитель с высокой точностью и превосходными характеристиками на постоянном токе: коэффициент ослабления синфазного сигнала CMR >> 100 dB на частотах вплоть до 1 кГц, смещение на входе не более 50 мкВ, малый входной ток (1 нА макс.) и низкое напряжение шума (0,28 мкВ в полосе 0,1...10 Гц). Для AD620 требуется единственный внешний резистор R_G, задающий коэффициент усиления (при условии, что R_2 и R_3 равны 24,9 кОм):

$$G = \frac{49{,}4}{R_G} + 1 \ (кОм).$$

В схеме обратной связи, предназначенной для компенсации синфазного сигнала, может быть применен операционный усилитель типа OP97 — малопотребляющий прецизионный ОУ с чрезвычайно высоким коэффициентом ослабления синфазного сигнала (мин. значение 114 дБ). Эта схема подает на пациента напряжение, компенсирующее синфазную составляющую сигнала с целью устранить влияние синфазного сигнала.

Далее в усилительный тракт необходимо включить ФВЧ для устранения постоянной составляющей в спектре полезного сигнала и схему основного усиления на базе операционного усилителя.

4.2.3 Аналоговые фильтры на основе модифицированной схемы Салле-на-Кея

Распространенной задачей при разработке электронных систем является задача выделения или подавления сигнала заданного частотного диапазона из сигнала более широкого спектрального состава. С этой целью используются фильтры, которые делятся на аналоговые и цифровые. Наиболее распространенными типами аналоговых фильтров являются: Чебышева, Баттервофа и Бесселя (также называемый фильтр Томпсона). Каждый из них оптимизирован по различным параметрам. Сложность каждого фильтра определяется выбранным числом полюсов. Чем больше полюсов в фильтре, тем сложнее электроника и лучше характеристики.

На рисунке 4.4 показано типичное построение аналогового фильтра, выполненного по модернизированной схеме Саллена-Кея. На схеме представлен двухполюсный низкочастотный (low-pass) фильтр, который может быть сконфигурирован под любой базовый тип фильтра. В таблице 4.1 представлена необходимая информация для выбора коэффициентов k_1 и k_2 , далее выбираются параметры соответствующих резисторов и емкостей (R_1 и C) и рассчитываются параметры резисторов R и R_f для частоты отсечки f_c в Гц:

$$R = \frac{k_1}{c * f_C}, \qquad R_f = R_1 k_2,$$

где k_1 и k_2 – коэффициенты (выбираются из таблицы 4.1), R_1 и C – номиналы емкостей и резисторов (задаются произвольно), f_c – частота отсечки фильтра.

Например, для создания 1кГц, 2-полюсного фильтра Баттервофа из таблицы 4.1 выбираем коэффициенты $k_1 = 0{,}1592$ и $k_2 = 0{,}586$. Произвольно выбираем $R_1 = 10$ К и $C = 0{,}01$ μF (обычные величины для схем с операционным усилителем), R и R_f вычисляются как величины, равные 15,95 К и 5,86 К соответственно. Округляем их до ближайшего значения 1 % стандартных резисторов и получаем $R = 15{,}8$ К и $R_f = 5{,}90$ К. Все компоненты должны иметь 1 % точность или выше.

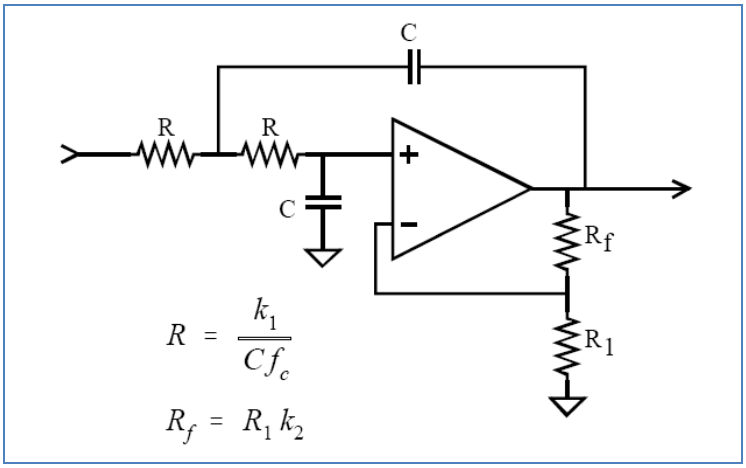


Рисунок 4.4 – Схема активного фильтра на основе модифицированной схемы Саллена-Кея

На рисунке 4.4 показана схема 2-полюсного низкочастотного фильтра. Фильтры с большим числом полюсов получаются последовательным соединением каскадов. Коэффициенты k_1 и k_2 определяются из таблицы 4.1, произвольно выбираются R_1 и C (10 К и 0,01 μF), а затем вычисляются R и R_f по формулам на рисунке 4.4, где f_c – частота отсечки в Гц.

Таблица 4.1 – Коэффициенты для создания фильтров

# poles		Bessel		Butterworth		Chebyshev	
		k_1	k_2	k_1	k_2	k_1	k_2
2	stage 1	0.1251	0.268	0.1592	0.586	0.1293	0.842
4	stage 1	0.1111	0.084	0.1592	0.152	0.2666	0.582
	stage 2	0.0991	0.759	0.1592	1.235	0.1544	1.660
6	stage 1	0.0990	0.040	0.1592	0.068	0.4019	0.537
	stage 2	0.0941	0.364	0.1592	0.586	0.2072	1.448
	stage 3	0.0834	1.023	0.1592	1.483	0.1574	1.846
8	stage 1	0.0894	0.024	0.1592	0.038	0.5359	0.522
	stage 2	0.0867	0.213	0.1592	0.337	0.2657	1.379
	stage 3	0.0814	0.593	0.1592	0.889	0.1848	1.711
	stage 4	0.0726	1.184	0.1592	1.610	0.1582	1.913

Четырех, -шести, -восьмиполюсные фильтры формируются последовательным включением двух, трех и четырех каскадов соответственно. Например, на рисунке 4.5 показана схема 6-полюсного фильтра Бесселя, выполненного последовательным включением 3-х каскадов. Каждый каскад имеет различные коэффициенты k_1 и k_2 согласно таблице 4.1, и, соответственно, различные конденсаторы и емкости. Если необходим высокочастотный фильтр, нужно поменять местами R и C, не изменяя R_f и R_1.

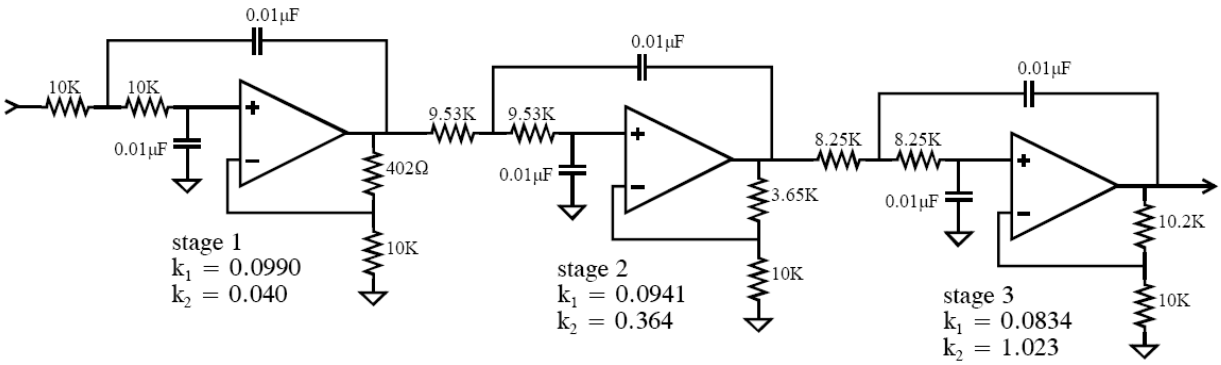


Рисунок 4.5 – Каскадное выполнение 6-полюсного фильтра Бесселя из 3 схем Саллена-Кея (низкочастотный фильтр с частотой отсечки 1 кГц)

4.3 Выбор элементной базы

В процессе разработки технического устройства необходимо учитывать не только технические, но и экономические показатели будущего прибора: в конструируемом устройстве необходимо сочетание минимальных затрат на производство прибора (а в них входит и стоимость элементной базы), максимальных показателей производительности (функциональности) и качества. Для этого подходить к проектированию следует так, как к разработке прибора, который будет в дальнейшем изготовлен.

Выбор элементной базы проводится на основе схемы электрической принципиальной с учетом изложенных в техническом задании условий и требований. Эксплуатационная надежность элементной базы в основном определяется правильным выбором типа элементов при проектировании и при использовании в режимах, которые не превышают предельно допустимых.

Для правильного выбора типа электронных компонентов необходимо на основе требований по установке, в частности климатических, механических и др. влияний, проанализировать условия работы каждого элемента и определить:

– эксплуатационные факторы (интервал рабочих температур, относительную влажность окружающей среды, атмосферное давление, механические нагрузки и др.);

– значения параметров и их разрешенные изменения в процессе эксплуатации (номинальное значение, допуск, сопротивление изоляции, шумы, вид функциональной характеристики и др.);

– разрешенные режимы и рабочие электрические нагрузки (мощность, напряжение, частота, параметры импульсного режима и др.);

– показатели надежности, долговечности и срока сохранения.

Критерием выбора в устройстве элементной базы является соответствие технологических и эксплуатационных характеристик электронных компонентов заданным условиями работы и эксплуатации.

Основными параметрами при выборе элементной базы являются:

1) технические параметры:

– номинальное значение параметров электронных компонентов согласно принципиальной электрической схеме прибора;

– допустимые отклонения величины электронных компонентов от их номинального значения;

– допустимое рабочее напряжение электронных компонентов;

– допустимая мощность рассеивания электронных компонентов;

– диапазон рабочих частот электронных компонентов;

– коэффициент электрической нагрузки электронных компонентов.

2) эксплуатационные параметры:

– диапазон рабочих температур;

– относительная влажность воздуха;

– атмосферное давление;

– вибрационные нагрузки;

– другие показатели.

Дополнительными критериями при выборе элементной базы являются:

– унификация электронных компонентов;

– масса и габариты электронных компонентов;

– наименьшая стоимость.

Выбор элементной базы по вышеназванным критериям позволяет обеспечить надежную работу изделия.

Следует отметить, что электронная промышленность является одной из самых динамичных отраслей в мировой экономики. Поэтому информация о новых выпускаемых компонентах появляется буквально каждую неделю. Поэтому разработчик должен отслеживать такие изменения, для того чтобы не пропустить важную новую информацию. Традиционными источниками информации разработчиков являются технические справочники, научно-технические журналы, документация фирм-производителей. В последние годы документация стала появляться не только в виде книг, но и на компакт-дисках. Перечисленные источники обладают как достоинствами, так и недостатками. Справочники общего характера обычно предоставляют лишь краткую информацию о компонентах. Этих сведений обычно недостаточно для применения, и требуется дополнительная документация. Журналы обычно содержат краткие обзоры конкретных типов электронных компонентов, в том числе и новых. Эту информацию следует рассматривать как сигнальную. Кроме того, в журналах часто приводятся подробные описания конкретных электронных приборов с рекомендациями по применению. Документация фирм-производителей является наиболее полной и достоверной. Документация делится в основном на сигнальную информацию (Short Form), технические данные (Data Sheet) и рекомендации по применению (Application Note). Часто имеются дополнительные издания с примерами применения. Основным недостатком всех перечисленных типов информации является отсутствие сведений о последних разработках. Таким образом, эта литература должна регулярно обновляться. В связи с вышесказанным, для разработчика электронного оборудования необходим такой источник информации, который сочетал бы в себе возможность поиска и сравнения различ-

ных электронных компонентов (справочник), при необходимости получение подробных сведений о конкретных электронных компонентах (документация фирм-производителей) и кроме этого обеспечивал бы возможность получения сведений о любых появляющихся новых компонентах. Единственным источником технической информации, удовлетворяющим вышеперечисленным требованиям, является мировая компьютерная сеть Internet. В настоящее время практически все производители электронных компонентов имеют электронные базы данных компонентов, которые регулярно обновляются по мере поступления на рынок новых компонентов. Имеется также информация о таких изделиях, которые только планируются к выпуску или находятся на этапе тестирования. Кроме этого такие базы данных имеют средства быстрого поиска по названию или типу компонента. При необходимости можно воспользоваться службой технической поддержки фирм.

4.4 Оформление схемы электрической принципиальной

Принципиальная схема является наиболее полной электрической схемой изделия, на которой изображают все электрические элементы и устройства, необходимые для осуществления и контроля в изделии заданных электрических процессов, все связи между ними, а также элементы подключения (разъемы, зажимы), которыми заканчиваются входные и выходные цепи. На схеме могут быть изображены соединительные и монтажные элементы, устанавливаемые в изделии по конструктивным соображениям.

Электрические элементы на схеме изображают условными графическими обозначениями, начертание и размеры которых установлены в стандартах ЕСКД [СТП01–2010]. Элементы, используемые в изделии частично, допускается изображать не полностью, а только используемые части.

Двоичные логические элементы на схеме изображают в виде УГО, построенных по правилам, установленным ГОСТ 2.743–82. Номера контактов устройств указываются над линиями связи рядом с соответствующими УГО логических элементов.

Схемы выполняют для изделий, находящихся в отключенном положении, в соответствии с рисунком 4.6. В технически обоснованных случаях допускается отдельные элементы схемы изображать в выбранном рабочем положении с указанием поля режима этих элементов.

В состав схемы кроме изображения входят надписи, характеризующие входные и выходные цепи, позиционные обозначения элементов и перечень элементов.

Условные графические обозначения элементов и устройств выполняют совмещенным или разнесенным способом. При совмещенном способе составные части элементов или устройств изображают на схеме так, как они расположены в изделии, т. е. в непосредственной близости друг к другу. При разнесенном способе условные графические обозначения составных частей элементов располагают в разных местах схемы с учетом порядка прохождения по ним тока (т.

е. последовательно), так, чтобы отдельные цепи были изображены наиболее наглядно. Разнесенным способом можно вычерчивать как отдельные элементы или устройства (например, обмотки и контакты группы реле, контакты штепсельных разъемов и др.), так и всю схему.

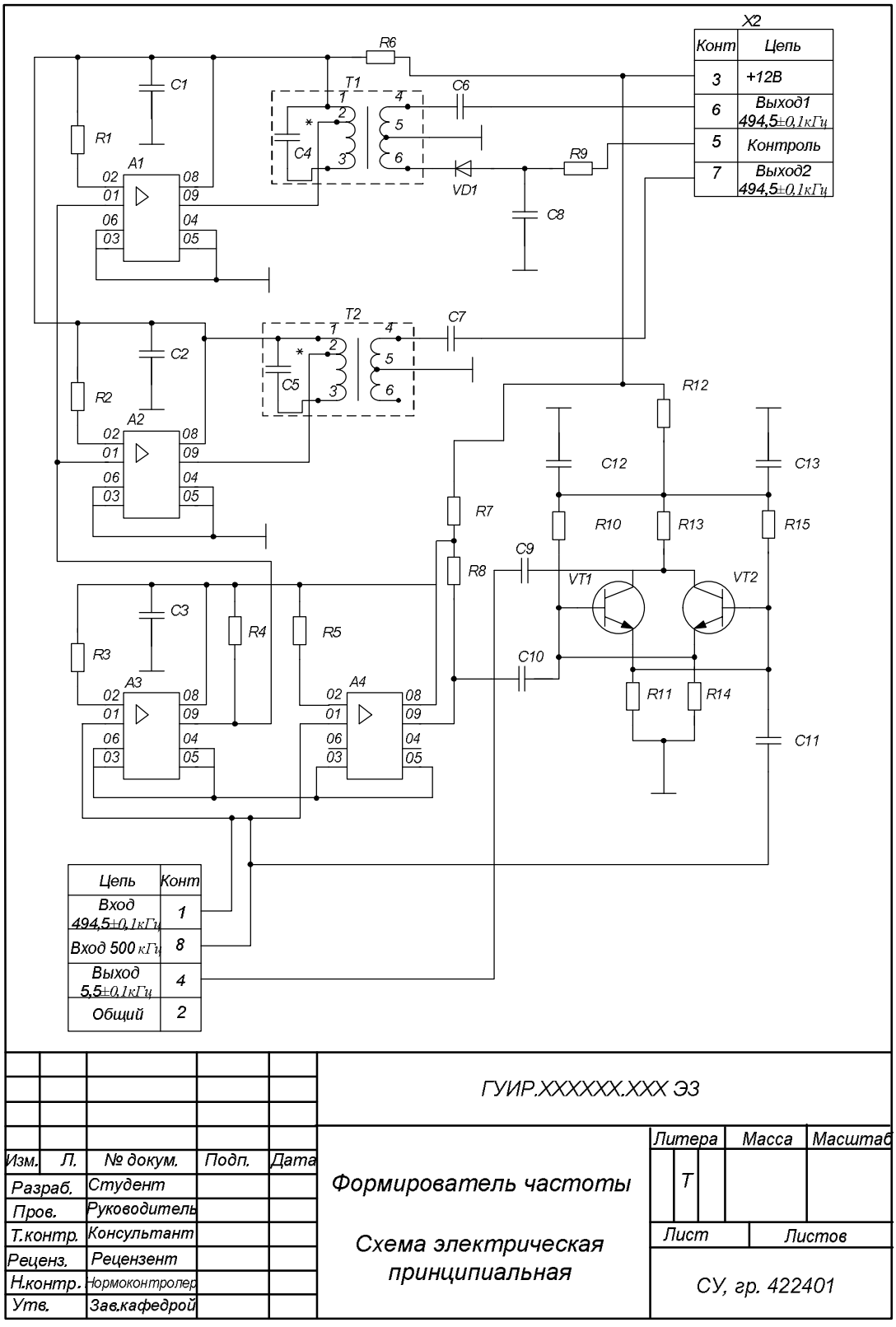


Рисунок 4.6 – Пример оформления схемы Э3 на листе графического материала

Позиционные обозначения элементов. Всем изображенным на схеме элементам и устройствам присваиваются условные буквенно-цифровые позиционные обозначения в соответствии с ГОСТ 2.710–81.

Позиционные обозначения элементов (устройств) в пределах изделия. Порядковые номера элементам (устройствам) начиная с единицы присваивают в пределах группы элементов (устройств) с одинаковым буквенным позиционным обозначением одной группы или одного типа в соответствии с последовательностью их расположения на схеме сверху вниз в направлении слева направо, например, *R*1, *R*2,…,*C*1, *C*2 (см. рисунок 4.6). Буквы и цифры позиционного обозначения выполняют чертежным шрифтом одного размера.

Последовательность присвоения порядковых номеров может быть нарушена в зависимости от размещения элементов изделия, направления прохождения сигналов или функциональной последовательности процесса, а также при внесении в схему изменений.

Позиционные обозначения проставляют на схеме рядом с условными графическими обозначениями элементов и устройств с правой стороны или над ними.

На схеме изделия, в состав которого входят устройства, позиционные обозначения элементам присваивают в пределах каждого устройства, а при наличии нескольких одинаковых устройств – в пределах этих устройств по правилам, изложенным выше.

$XA(\longrightarrow)$

Конт.	Цепь	Адрес
1	Δf = 10...100 Гц	= *A*1 – *X*1:5
2	$U_{вых}$ = 2 В; $R_н$ = 600 Ом	= *A*5 – *X*4:7
3	$I_{вых}$ = 100 мА; $R_н$ = 10 Ом	= *A*1 – *X*1:6
4	$f_{гти}$ = 1 МГц	= *A*3 – *X*1:20

Примечание – «= *A*5 – *X*4: 7» означает, что выходной контакт изделия должен быть соединён с седьмым контактом четвёртого соединителя устройства *A*5.

Рисунок 4.7 – Пример оформления таблицы с характеристиками выходных цепей и адресата внешних соединений

Если в состав изделия входят функциональные группы, то вначале присваивают позиционные обозначения элементам, не входящим в функциональные

группы, а затем элементам, входящим в функциональные группы. Для одинаковых функциональных групп позиционные обозначения элементов, присвоенные одной из них, повторяют во всех последующих группах.

Обозначения устройства указывают сверху или справа от изображения. При разнесенном способе изображения позиционные обозначения проставляют около каждой составной части.

На принципиальной электрической схеме изображают клеммы, разъемы и другие элементы и указывают характеристики входных и выходных цепей изделия (величину напряжения, силу тока, частоту и т. д.), а также указывают адреса внешних соединений (рисунок 4.7).

Рекомендуется взамен условных графических обозначений соединительных элементов помещать таблицы с характеристиками входных и выходных цепей изделия и адресами их внешних подключений. Над таблицей допускается указывать графическое или буквенное обозначение гнезда или штыря.

4.4.1 Условные буквенно-цифровые обозначения в электрических схемах

1 Элементы (устройства, функциональные группы), входящие в изделие, на схеме должны иметь буквенные или буквенно-цифровые обозначения.

Буквенно-цифровые обозначения предназначены для записи в сокращенной форме сведений об элементах, устройствах и функциональных группах в документации на изделие или для нанесения непосредственно на изделие.

Типы условных буквенно-цифровых обозначений и правила их построения устанавливает ГОСТ 2.710–81.

2 Буквенные коды видов элементов приведены в таблице 4.2. Элементы разбиваются по видам на группы, имеющие обозначения из одной буквы. Для уточнения вида элементов применяют двухбуквенные и многобуквенные коды. При применении двухбуквенных и многобуквенных кодов первая буква должна соответствовать группе видов, к которой принадлежит элемент. Дополнительные обозначения должны быть пояснены на поле схемы.

Таблица 4.2 – Буквенные коды видов элементов

1-я буква кода (обязательная)	Группа видов элементов	Примеры видов элементов	Двухбуквенный код
1	2	3	4
A	Устройства (общее обозначение)	Усилители, приборы регулирования, телеуправления, лазеры, мазеры	

Продолжение таблицы 4.2

1	2	3	*4*
B	Преобразователи неэлектрических величин в электрические (кроме генераторов и источников питания) или наоборот, аналоговые или многоразрядные преобразователи или датчики для указания или измерения	Громкоговоритель Магнитострикционный элемент Детектор ионизирующих излучений Сельсин–приемник Телефон (капсюль) Сельсин–датчик Тепловой датчик Фотоэлемент Микрофон Датчик давления Пьезоэлемент Датчик частоты вращения (тахогенератор) Звукосниматель Датчик скорости	*BA* *BB* *BD* *BE* *BF* *BC* *BK* *BL* *BM* *BP* *BQ* *BR* *BS* *BV*
C	Конденсаторы		
D	Схемы интегральные, микросборки	Схема интегральная аналоговая Схема интегральная цифровая, логический элемент Устройства хранения информации Устройства задержки	*DA* *DD* *DS* *DT*
E	Элементы разные (осветительные устройства, нагревательные элементы)	Нагревательный элемент Лампа осветительная Пиропатрон	*EK* *EL* *ET*
F	Разрядники, предохранители, устройства защитные	Дискретный элемент защиты по току мгновенного действия Дискретный элемент защиты по току инерционного действия Предохранитель плавкий Дискретный элемент защиты по напряжению, разрядник	*FA* *FP* *FU* *FV*
G	Генераторы, источники питания, кварцевые осцилляторы	Батарея	*GB*
H	Устройства индикационные и сигнальные	Прибор звуковой сигнализации Индикатор символьный Прибор световой сигнализации	*HA* *HG* *HL*
K	Реле, контакторы, пускатели	Реле токовое Реле указательное Реле электротепловое Контактор, магнитный пускатель Реле времени Реле напряжения	*KA* *KH* *KK* *KM* *KT* *KV*

Продолжение таблицы 4.2

1	2	3	4
L	Катушки индуктивности, дроссели	Дроссель люминесцентного освещения	*LL*
M	Двигатели постоянного и переменного тока		
P	Приборы, измерительное оборудование *Примечание* – Сочетание PE применять не допускается	Амперметр Счетчик импульсов Частотомер Счетчик активной энергии Счетчик реактивной энергии Омметр Регистрирующий прибор Часы, измеритель времени действия Вольтметр Ваттметр	*PA* *PC* *PF* *PI* *PK* *PR* *PS* *PT* *PV* *PW*
Q	Выключатели и разъединители в силовых цепях (энергоснабжение, питание оборудования и т. д.)	Выключатель автоматический Короткозамыкатель Разъединитель	*QF* *QK* *QS*
R	Резисторы	Терморезистор Потенциометр Шунт измерительный Варистор	*RK* *RP* *RS* *RU*
S	Устройства коммутационные в цепях управления, сигнализации и измерительных цепях *Примечание* – Обозначение *SF* применяют для аппаратов, не имеющих контактов силовых цепей	Выключатель или переключатель Выключатель кнопочный Выключатель автоматический Выключатели, срабатывающие от следующих воздействий: уровня давления положения (путевой) частоты вращения температуры	*SA* *SB* *SF* *SL* *SP* *SQ* *SR* *SK*
T	Трансформаторы, автотрансформаторы	Трансформатор тока Электромагнитный стабилизатор Трансформатор напряжения	*TA* *TS* *TV*
U	Устройства связи Преобразователи одного вида электрических величин в другой вид	Модулятор Демодулятор Дискриминатор Преобразователь частотный, инвертор, генератор частоты, выпрямитель	*UB* *UR* *UI* *UZ*

Продолжение таблицы 4.2

1	2	3	4
V	Приборы электровакуумные и полупроводниковые	Диод, стабилитрон Прибор электровакуумный Транзистор Тиристор	*VD* *VL* *VT* *VS*
W	Линии и элементы СВЧ	Ответвитель Короткозамыкатель Вентиль	*WE* *WK* *WS*
	Антенны	Трансформатор, неоднородность, фазовращатель Аттенюатор Антенна	*WT* *WU* *WA*
X	Соединения контактные	Токосъемник, контакт скользящий Штырь Гнездо Соединение разборное Соединитель высокочастотный	*XA* *XP* *XS* *XT* *XW*
Y	Устройства механические с электромагнитным приводом	Электромагнит Тормоз с электромагнитным приводом Муфта с электромагнитным приводом Электромагнитный патрон или плита	*YA* *YB* *YC* *YH*
Z	Устройства оконечные, фильтры	Ограничитель Фильтр кварцевый	*ZL* *ZQ*

Буквенные коды применяются для обозначения элементов на структурных, функциональных и принципиальных схемах.

5 Разработка алгоритма работы микроконтроллера

5.1 Понятие алгоритма и его свойства

Проектирование устройств на базе микроконтроллеров предусматривает разработку алгоритма их работы как обязательного этапа, предшествующего аппаратно-программной реализации проектируемой системы.

В общем случае разработка алгоритма решает задачу создания точного набора инструкций, описывающих порядок действий исполнителя для достижения результата решения задачи за конечное время. Применительно к системам с микроконтроллерами разработка алгоритма состоит в создании набора инструкций и порядка их следования, при выполнении которого с помощью микроконтроллера достигаются цели функционирования проектируемой системы. При этом «порядок» следования инструкций в отличие от «последовательности» предполагает возможность параллельности в работе.

Алгоритм считается правильным, если он отвечает требованиям задачи. Алгоритм содержит ошибки, если для некоторых исходных данных он даёт неправильные результаты, сбои или не дает результатов вообще.

Основными характеристиками алгоритмов являются:

– Дискретность алгоритма: деление алгоритма на отдельные шаги (действия).

– Элементарность шагов: закон получения последующей системы величин из предыдущей должен быть простым и локальным. Какой шаг (действие) можно считать элементарным, определяется особенностями исполнителя алгоритма.

– Массовость алгоритма: начальная система величин может выбираться из некоторого множества. Данное свойство означает, что один алгоритм, т.е. один и тот же порядок действий, может применяться для решения некоторого класса задач.

– Детерминированность: совокупность промежуточных величин на любом шаге алгоритма однозначно определяется системой величин, имевшихся на предыдущем шаге. Данное свойство означает, что результат выполнения алгоритма определяется только входными данными и последовательностью действий алгоритма, а при применении алгоритма к одним и тем же исходным данным должен получаться всегда один и тот же результат.

– Результативность: выполнение алгоритма должно обязательно приводить к его завершению. Однако существуют примеры формально бесконечных алгоритмов, применяемых на практике. Так, алгоритм работы системы сбора метеорологических данных состоит в непрерывном повторении последовательности действий («измерить температуру воздуха», «определить атмосферное давление»), выполняемых с определенной частотой (через минуту, час) во все время всего существования данной системы.

5.2 Представление алгоритма в виде блок-схем. Правила оформления

Наиболее удобным способом записи алгоритма является запись на языке блок-схем – набора символов (блоков различной формы), краткого пояснительного текста и соединяющих линий (линий потока данных или потока управления). Каждый элемент является шагом алгоритма. Основное достоинство такой формы представления – наглядность: блок-схема позволяет охватить весь алгоритм сразу, отследить различные варианты его выполнения. Однако в блок-схеме, как правило, отсутствует подробное описание конкретных действий – их существование лишь обозначено.

Правила выполнения блок-схем определяются следующими документами:

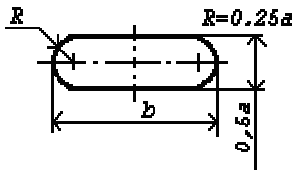
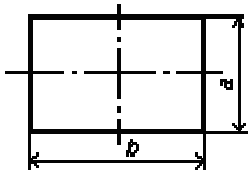
– ГОСТ 19.002-80. Схемы алгоритмов и программ. Правила выполнения.

– ГОСТ 19.003-80. Схемы алгоритмов и программ. Обозначения условные графические.

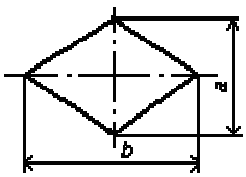
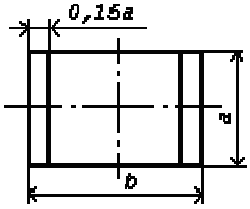
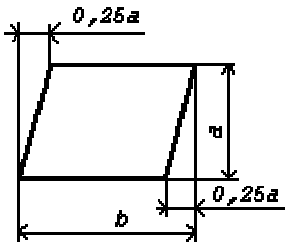
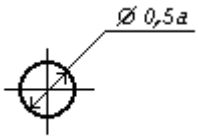
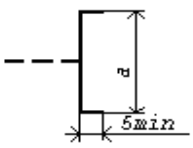
– ГОСТ 19.701-90. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.

Обозначение, размеры обязательных символов и отображаемые ими функции в алгоритме и программе должны соответствовать указанным в таблице 5.1.

Таблица 5.1 – Основные символы блок-схемы алгоритма

Обозначение символа на схеме	Размеры*	Функция
1	2	3
Терминатор (пуск-остановка)		
Действие		Начало, конец, прерывание процесса обработки данных или выполнения программы
Процесс		
Действие		Выполнение операций, в результате которых изменяется значение, форма представления или расположение данных

Продолжение таблицы 5.1

1	2	3
Решение		
		Выбор направления выполнения алгоритма в зависимости от некоторых переменных условий. Вход в элемент обозначается линией, входящей в верхнюю вершину элемента, выходы – линиями, выходящими из оставшихся вершин и сопровождающимися соответствующими значениями условий**
Предопределенный процесс		
		Использование ранее созданных и отдельно описанных алгоритмов или программ (например вызов процедуры или функции; внутри фигуры записывается имя вызываемой процедуры или функции)
Данные (ввод–вывод)		
		Преобразование данных в форму, пригодную для обработки (ввод) или отображения результатов обработки (вывод)
Соединитель		
		Указание связи между прерванными линиями потока (например разделение блок-схемы, не помещающейся на листе). Соответствующие соединительные символы должны иметь одно уникальное обозначение
Комментарий		
		Связь между элементом схемы и пояснением. Комментарий помещается со стороны квадратной скобки и охватывается ей по всей высоте. Пунктирная линия идет к описываемому элементу либо группе элементов (при этом группа выделяется замкнутой пунктирной линией)

* Рекомендуется размер *a* выбирать из ряда 10, 15, 20 мм, а также увеличивать размер *a* на число, кратное 5. Размер *b* определяется как 1,5*a* (ГОСТ 19.003-80. Схемы алгоритмов и программ. Обозначения условные графические).

** Если выходов больше трех, то их следует показывать одной линией, выходящей из вершины (чаще нижней) элемента, которая затем разветвляется в соответствующее число линий.

Правила применения символов:

1 Символ предназначен для графической идентификации функции, которую он отображает, независимо от текста внутри этого символа.

2 Символы в схеме должны быть расположены равномерно. Следует придерживаться разумной длины соединений и минимального числа длинных линий.

3 Большинство символов задумано так, чтобы дать возможность включения текста внутри символа. Формы символов, установленные настоящим стандартом, должны служить руководством для фактически используемых символов. Не должны изменяться углы и другие параметры, влияющие на соответствующую форму символов. Символы должны быть, по возможности, одного размера.

4 Символы могут быть вычерчены в любой ориентации, но, по возможности, предпочтительной является горизонтальная ориентация. Зеркальное изображение формы символа обозначает одну и ту же функцию, но не является предпочтительным.

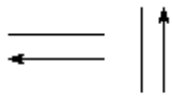
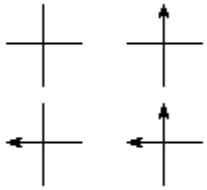
5 Минимальное количество текста, необходимого для понимания функции данного символа, следует помещать внутри данного символа. Текст для чтения должен записываться слева направо и сверху вниз независимо от направления потока.

6 Если объем текста, помещаемого внутри символа, превышает его размеры, следует использовать символ комментария.

7 Если использование символов комментария может запутать или разрушить ход схемы, текст следует помещать на отдельном листе и давать перекрестную ссылку на символ.

Обозначение соединений между символами и отображаемые ими функции в алгоритме и программе должны соответствовать указанным в таблице 5.2.

Таблица 5.2 – Обозначение соединений между символами блок-схемы алгоритма

Наименование	Обозначение на схеме	Функция
1	2	3
Линии потока		Указание направления линии потока: – допускается без стрелки, если линия направлена слева направо и сверху вниз; – со стрелкой в остальных случаях
Излом линии под углом 90°		Изменение направление потока
Пересечение линий потока		Пересечение двух несвязанных потоков
Слияние линий потока		Слияние линий потока, каждая из которых направлена к одному и тому же символу на схеме

Правила выполнения соединений:

1 Потоки данных или потоки управления в схемах показываются линиями. Направление потока слева направо и сверху вниз считается стандартным. В случае когда необходимо внести большую ясность в схему (например при соединениях), на линиях используются стрелки. Если поток имеет направление, отличное от стандартного, стрелки должны указывать это направление.

2 Расстояния между параллельными линиями потока должно быть не менее 3 мм, между символами схемы – не менее 5 мм.

3 В схемах следует избегать пересечения линий. Пересекающиеся линии не имеют логической связи между собой, поэтому изменения направления в точках пересечения не допускаются.

4 Линии в схемах должны подходить к символу либо слева, либо сверху, а исходить либо справа, либо снизу. Линии должны быть направлены к центру символа.

5 При необходимости линии в схемах следует разрывать для избежания излишних пересечений или слишком длинных линий, а также если схема состоит из нескольких страниц. Соединитель в начале разрыва называется внешним соединителем, а соединитель в конце разрыва – внутренним соединителем. В случае размещения алгоритма на нескольких страницах ссылки с указанием номера новой страницы могут быть приведены в символе комментария для их соединителей.

5.3 Поток управления действиями алгоритма и способы его организации

Поскольку алгоритм определяет порядок обработки информации, он должен содержать, с одной стороны, действия по обработке, а с другой стороны, порядок их следования, называемый потоком управления.

Рассмотренные выше блоки, связанные с обработкой данных, делятся на простые и условные. Особенность простого действия в том, что оно имеет один вход и один выход в отличие от условного, обладающего двумя выходами в зависимости от того, истинным ли окажется условие.

Часть алгоритма, организованная как простое действие, т.е. имеющая один вход (выполнение начинается всегда с одного и того же действия) и один выход (т.е. после завершения данного блока всегда начинает выполняться одно и то же действие), называется функциональным блоком. Из этого следует, что любое простое действие является функциональным блоком, а условное – нет.

Выделяют три различных варианта организации потока управления действиями алгоритма в зависимости от выполнения следующих свойств:

1 Все блоки алгоритма выполняются.

2 Каждый из выполняемых блок выполняется только один раз.

Поток управления, в котором выполняются оба эти свойства, называется линейным: в нем несколько функциональных блоков выполняются последовательно. Линейному потоку на языке блок-схем соответствует структура, приведенная на рисунке 5.1.

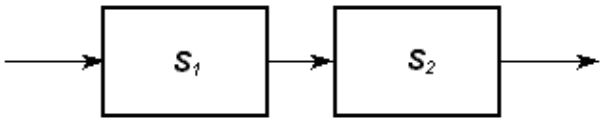


Рисунок 5.1 – Пример линейного потока управления алгоритма

Несколько блоков, связанных линейным потоком управления, могут быть объединены в один функциональный блок (рисунок 5.2).

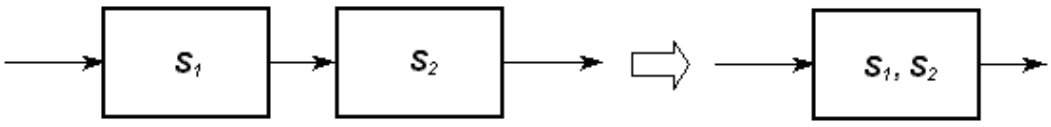


Рисунок. 5.2 – Пример объединения нескольких блоков, связанных линейным потоком управления, в один функциональный блок

Второй тип потока управления называется ветвлением: он организует выполнение одного из двух функциональных блоков в зависимости от значения проверяемого логического условия (рисунок 5.3). В этом типе выполняется свойство 2, но не выполняется свойство 1. Если структура содержит два функ-

циональных блока (S1 и S2), ветвление называется полным; возможно существование неполного ветвления – при этом один из блоков пуст (обычно S2).

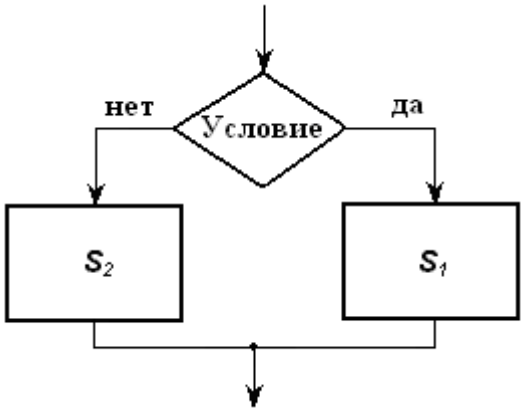


Рисунок 5.3 – Пример потока управления алгоритма
по типу ветвления

Третий тип потока управления называется циклическим: он организует многократное повторение функционального блока до тех пор, пока логическое условие его выполнения является истинным (рисунок 5.4). Для циклического потока выполняется свойство 1, но не выполняется свойство 2.

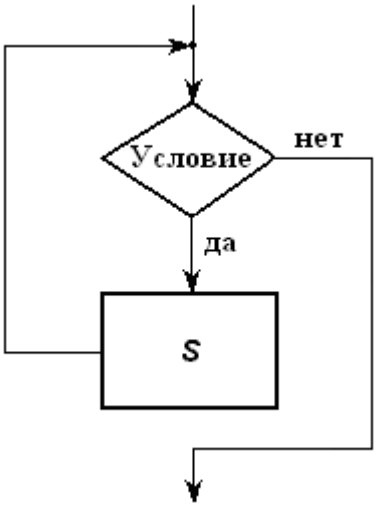


Рисунок 5.4 – Пример циклического потока
управления алгоритма

Процессы циклической структуры делятся на три подтипа:

1 Счетный цикл – циклический процесс, при котором количество повторений определено.

2 Итерационный цикл – циклический процесс, который завершается при достижении либо нарушении определенных условий.

3 Поисковый цикл – циклический процесс, который завершается по окончании другого процесса либо предусмотрено досрочное завершается при достижении определенных условий.

5.4 Концептуальный и функциональный алгоритмы

Схемы алгоритмов могут использоваться на различных уровнях детализации, причем число уровней зависит от размеров и сложности задачи обработки данных. Уровень детализации должен быть таким, чтобы различные части и взаимосвязь между ними были понятны в целом.

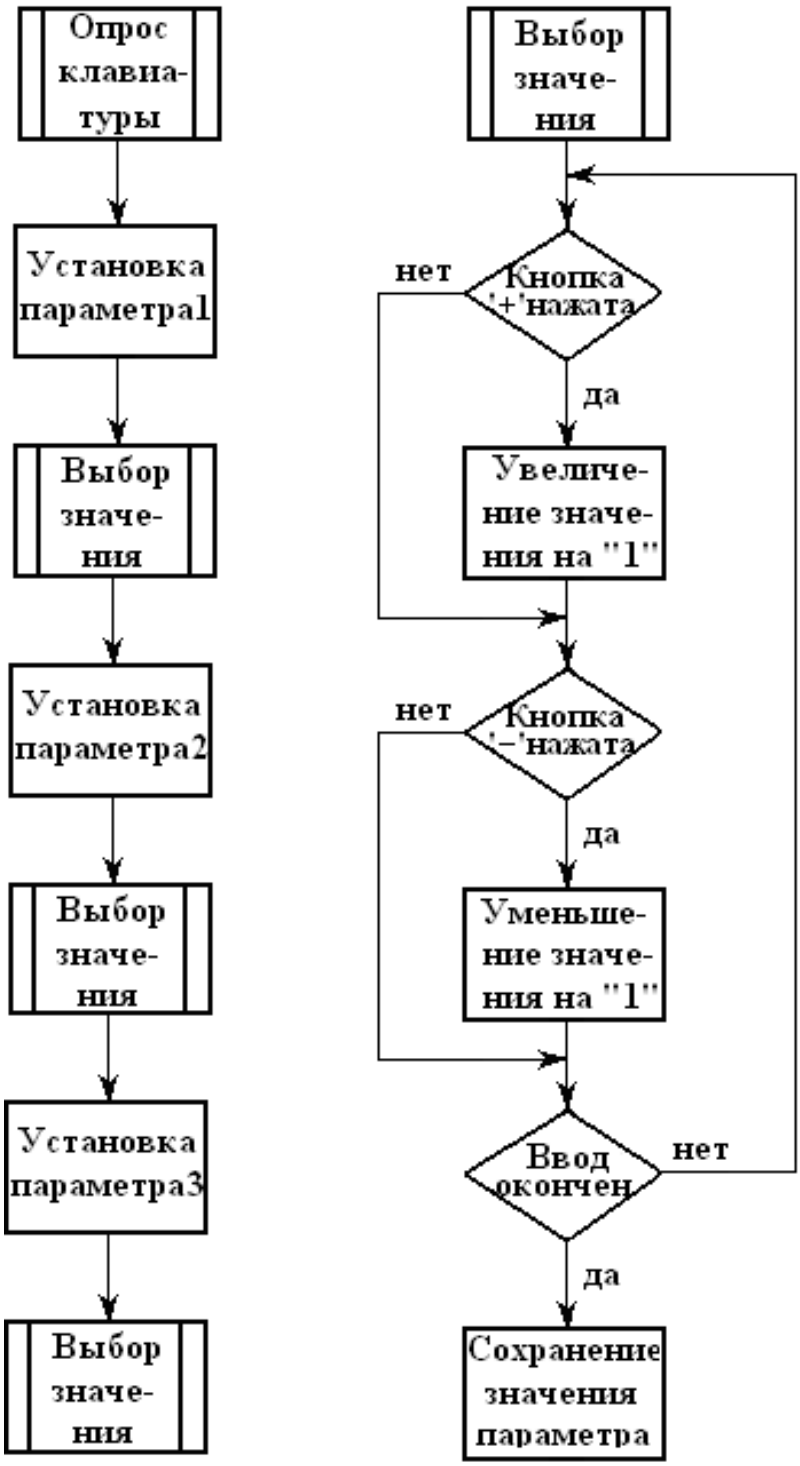
Обобщенный алгоритм работы проектируемого на базе микроконтроллеров устройства называется концептуальным. Такой алгоритм характеризуется низким уровнем детализации и укрупненно отражает принципиальные этапы работы устройства, соответствующие основным подзадачам разрабатываемой системы. Концептуальный алгоритм работы проектируемого устройства отличается понятностью и простотой восприятия, поскольку невелико число исходных структур, которыми он образован.

Так, например, концептуальный алгоритм работы микроконтроллера любого терапевтического аппарата можно представить в виде следующих функциональных блоков : инициализация, опрос клавиатуры, выбор режима работы, рабочая часть – заключенных между стандартными блоками «Начало» и «Конец» (рисунок 5.5). Блок «Инициализация» предусматривает выполнение микроконтроллером базового набора подготовительных операций после включения устройства. Блок «Опрос клавиатуры» решает задачу ввода пользователем требуемых параметров терапевтической процедуры (например, длительность процедуры, амплитуда и частота воздействия, непрерывный или прерывистый характер воздействия и т.д.). Блок «Выбор режима работы» выполняет выбор подпрограммы, обеспечивающей требуемый режим работы, а следующий далее блок «Рабочая часть» реализует работу подпрограммы требуемого режима в течение заданного промежутка времени.

Детальный алгоритм работы проектируемого на базе микроконтроллеров устройства называется функциональным. Такой алгоритм предусматривает организацию каждого программного модуля в виде стандартного функционального блока (одного из трех базовых структур), выполняющего только одну функцию. Модули обладают определенной автономностью, что позволяет вести поиск и устранение ошибок независимо от остальной части алгоритма (на последующих этапах программы), а также обеспечивает относительно простую модифицируемость как отдельного модуля, так и алгоритма (программы) в целом. Разработка функционального алгоритма особенно важна при создании сложных программ: модульный принцип позволяет разбить общую задачу на составные и относительно автономные части, каждая из которых может создаваться и отлаживаться независимо.

Рисунок 5.5 – Пример концептуального алгоритма работы микроконтроллера в терапевтическом устройстве

Примером фрагмента функционального алгоритма может служить подпрограмма опроса клавиатуры (рисунок 5.6, а), представляющая собой установку требуемого числа параметров с помощью многократного вызова вложенной подпрограммы выбора значения конкретного параметра (рисунок 5.6, б).



а б

а – подпрограмма опроса клавиатуры;
б – вложенная подпрограмма выбора значения конкретного параметра

Рисунок 5.6 – Фрагмент функционального алгоритма
работы микроконтроллера

6 Разработка программы для микроконтроллеров Cygnal

6.1 Краткий обзор микроконтроллера Cygnal семейства C8051F320

МК C8051F320 имеет встроенную схему сброса по включению питания, схему слежения за напряжением питания, регулятор напряжения, сторожевой таймер, тактовый генератор и представляет собой, таким образом, функционально законченную систему на кристалле. Имеется возможность внутрисхемного программирования Flash-памяти, что обеспечивает долговременное (энергонезависимое) хранение данных, а также позволяет осуществлять обновление программного обеспечения в готовых изделиях. Программа пользователя может полностью управлять всеми периферийными модулями, а также может индивидуально отключить любой из них с целью уменьшения энергопотребления. Встроенный двухпроводный Silicon Labs Development Interface (интерфейс C2) позволяет производить «неразрушающую» (не используются внутренние ресурсы) внутрисхемную отладку в режиме реального времени, используя МК, установленный в конечное изделие. Средства отладки обеспечивают проверку и модификацию памяти и регистров, расстановку точек остановок, пошаговое выполнение программы, а также поддерживают команды запуска и остановки. В процессе отладки с использованием интерфейса C2 все аналоговые и цифровые периферийные модули полностью сохраняют свою работоспособность. Два вывода интерфейса C2 могут использоваться для других пользовательских функций, что позволяет осуществлять внутрисистемную отладку, не занимая для этого отдельные выводы корпуса. Структурная схема МК C8051F320 изображена на рисунке 1.

Каждый МК предназначен для работы в промышленном температурном диапазоне (–40ºС…+85ºС) при напряжении питания 2,7 В…3,6 В. (Примечание – для взаимодействия по шине USB требуется напряжение 3.0 В…3.6 В). На порты ввода/вывода и вывод /RST могут быть поданы входные сигналы напряжением до 5В. МК C8051F320/1 выпускаются в 32-выводных корпусах типа LQFP и 28-выводных корпусах типа MLP.

6.1.1 Процессорное ядро CIP-51

По системе команд ядро полностью совместимо с ядром MCS-51. Для разработки программного обеспечения могут использоваться стандартные 803x/805x ассемблеры и компиляторы. Процессорное ядро CIP-51 использует конвейерную архитектуру, что существенно повышает скорость выполнения команд по сравнению со стандартной архитектурой 8051. В МК с архитектурой 8051 все команды, кроме MUL и DIV, исполняются за 12 или 24 системных тактовых цикла при максимальной тактовой частоте 12…24 МГц. При работе на тактовой частоте 25 МГц производительность ядра CIP-51 может достигать 25 MIPS.

Ядро имеет стандартную (8051) структуру адресного пространства *памяти программ и данных*. В состав памяти входит ОЗУ объемом 256 байт, старшие

128 байт которого имеют двойную конфигурацию. В режиме косвенной адресации осуществляется доступ к старшим 128 байтам ОЗУ общего назначения, а в режиме прямой адресации осуществляется доступ к 128 байтам адресного пространства регистров специального назначения (SFR). Младшие 128 байт ОЗУ доступны как для прямой, так и для косвенной адресации. Из них первые 32 байта адресуются как четыре банка регистров общего назначения, а следующие 16 байт адресуются побайтно или побитно.

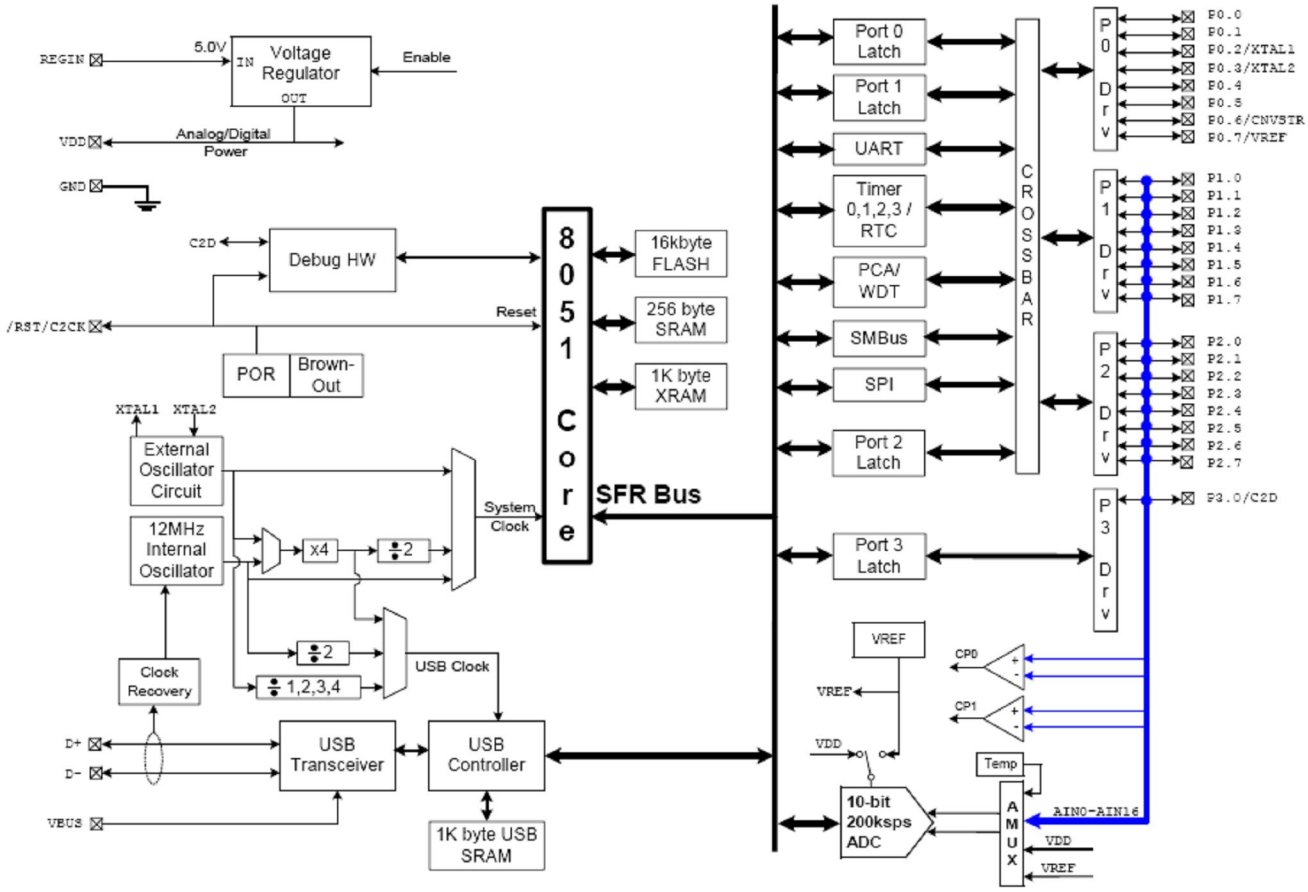


Рисунок 6.1 – Структурная схема C8051F320

Память программ МК состоит из 16 Кбайт Flash-памяти. Эта память может перепрограммироваться внутрисистемно секторами по 512 байт, не требуя при этом специального внешнего напряжения программирования. На рисунке 6.2 приведена карта распределения памяти МК.

6.1.2 Программируемые цифровые порты ввода/вывода и матрица соединений

МК C8051F320 имеет 25 выводов, которые составляют три 8-разрядных порта и один 1-разрядный порт. Порты функционируют в соответствии со стандартом 8051 с некоторыми дополнительными возможностями. Каждый вывод

порта можно настроить как аналоговый вход или как цифровой вход–выход. Выводы, настроенные как цифровые входы–выходы, можно кроме этого настроить как двухтактные цифровые выходы или выходы с открытым стоком. Также допускается глобальное отключение слаботоковых подтягивающих резисторов, что позволяет еще более снизить энергопотребление.

Рисунок 6.2 – Карта распределения памяти

Цифровая матрица позволяет соединять внутренние цифровые системные ресурсы с выводами портов ввода/вывода. При помощи регистров управления матрицей на выводы портов могут быть выведены сигналы от внутренних таймеров/счетчиков, от последовательных интерфейсов, аппаратные прерывания, выходы компараторов и другие цифровые сигналы. Это позволяет выбрать точную комбинацию связей между портами ввода/вывода общего назначения и цифровыми ресурсами, необходимую для каждого конкретного приложения.

6.1.3 Последовательные порты

В МК семейства C8051F320/1 встроены следующие последовательные интерфейсы:

– полнодуплексный УАПП с усовершенствованным генератором скорости передачи данных;

– усовершенствованный SPI;

– I2C/SMBus.

Каждый из этих интерфейсов реализован на аппаратном уровне и широко использует прерывания, требуя лишь незначительного вмешательства со стороны программы пользователя.

6.1.4 10-разрядный аналого-цифровой преобразователь

МК C8051F320/1 имеют встроенный 10-разрядный АЦП последовательного приближения с 17-канальным дифференциальным входным мультиплексором. При максимальной производительности 200 тыс. преобразований в секунду этот АЦП обеспечивает 10-битную точность преобразования с нелинейностью на уровне ±1МЗР. Система АЦП включает настраиваемый аналоговый мультиплексор, посредством которого осуществляется выбор как положительного, так и отрицательного входов АЦП. Порты 1–3 доступны как аналоговые входы; кроме этого, входными сигналами АЦП могут быть выходной сигнал встроенного датчика температуры и напряжение питания (VDD). Программа пользователя может отключать АЦП с целью уменьшения энергопотребления.

Для запуска преобразования могут использоваться: программная команда, переполнение таймеров 0, 1, 2 или 3, внешний сигнал запуска. Такая гибкость позволяет осуществлять запуск преобразований при возникновении определенных программных событий, периодически (при переполнениях таймера) или сигналом от внешних устройств. При завершении преобразования полученное 10-разрядное слово данных «защелкивается» в регистрах данных АЦП, устанавливается специальный бит состояния и генерируется прерывание (если оно разрешено).

АЦП может быть настроен таким образом, чтобы генерировать прерывание лишь при попадании или непопадании результата преобразования в заданный диапазон значений (окно). АЦП может непрерывно отслеживать сигнал в фоновом режиме, но не прерывать МК до тех пор, пока преобразованные данные находятся в пределах или вне пределов заданного окна.

6.1.5 Предельно допустимые параметры

1 Предельная рабочая температура: от –55ºС до +125ºС

2 Температура хранения от –65ºС до +150ºС

3 Напряжение на любом выводе порта ввода/вывода или на выводе /RST относительно GND от –0.3 В до 5.8 В

4 Напряжение на выводе VDD относительно GND от –0.3 В до 4.2 В

5 Максимальный суммарный ток по выводам VDD и GND 500 мА

6 Максимальный выходной втекающий ток по любому порту ввода/вывода или по выводу /RST 100 мА

Выход за указанные значения может привести к необратимым повреждениям микроконтроллера. Работа микроконтроллера в предельном режиме в течение длительного времени не предусмотрена. Длительная эксплуатация микроконтроллера в недопустимых условиях может повлиять на его надежность.

6.2 Система команд микроконтроллеров с ядром CIP-51

Система команд микроконтроллера представлена в таблице 6.1.

Таблица 6.1 – Система команд микроконтроллера

Мнемоника команды	Описание	Байты	Циклы
1	2	3	4
АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ			
ADD A,Rn	Сложение аккумулятора с регистром (n = 0…7)	1	1
ADD A,direct	Сложение аккумулятора с прямоадресуемым байтом	2	2
ADD A,@Ri	Сложение аккумулятора с косвенноадресуемым байтом ОЗУ	1	2
ADD A,#data	Сложение аккумулятора с константой	2	2
ADDC A,Rn	Сложение аккумулятора с регистром и переносом	1	1
ADDC A,direct	Сложение аккумулятора с прямоадресуемым байтом и переносом	2	2
ADDC A,@Ri	Сложение аккумулятора с косвенноадресуемым байтом ОЗУ и переносом	1	2
ADDC A,#data	Сложение аккумулятора с константой и переносом	2	2
SUBB A,Rn	Вычитание из аккумулятора регистра и заема	1	1
SUBB A,direct	Вычитание из аккумулятора прямоадресуемого байта и заема	2	2
SUBB A,@Ri	Вычитание из аккумулятора косвенноадресуемого байта ОЗУ и заема	1	2
SUBB A,#data	Вычитание из аккумулятора константы и заема	2	2
INC A	Инкремент аккумулятора	1	1
INC Rn	Инкремент регистра	1	1
INC direct	Инкремент прямоадресуемого байта	2	2
INC @Ri	Инкремент косвенноадресуемого байта ОЗУ	1	2
DEC A	Декремент аккумулятора	1	1
DEC Rn	Декремент регистра	1	1
DEC direct	Декремент прямоадресуемого байта	2	2
DEC @Ri	Декремент косвенноадресуемого байта ОЗУ	1	2
INC DPTR	Инкремент указателя данных	1	1
MUL AB	Умножение аккумулятора на регистр B	1	4
DIV AB	Деление аккумулятора на регистр B	1	8
DA A	Десятичная коррекция аккумулятора	1	1
ЛОГИЧЕСКИЕ ОПЕРАЦИИ			
ANL A,Rn	Логическое И аккумулятора и регистра	1	1

Продолжение таблицы 6.1

1	2	3	4
ANL A,direct	Логическое И аккумулятора и прямоадресуемого байта	2	2
ANL A,@Ri	Логическое И аккумулятора и косвенноадресуемого байта ОЗУ	1	2
ANL A,#data	Логическое И аккумулятора и константы	2	2
ANL direct,A	Логическое И прямоадресуемого байта и аккумулятора	2	2
ANL direct,#data	Логическое И прямоадресуемого байта и константы	3	3
ORL A,Rn	Логическое ИЛИ аккумулятора и регистра	1	1
ORL A,direct	Логическое ИЛИ аккумулятора и прямоадресуемого байта	2	2
ORL A,@Ri	Логическое ИЛИ аккумулятора и косвенноадресуемого байта ОЗУ	1	2
ORL A,#data	Логическое ИЛИ аккумулятора и константы	2	2
ORL direct,A	Логическое ИЛИ прямо-адресуемого байта и аккумулятора	2	2
ORL direct,#data	Логическое ИЛИ прямо-адресуемого байта и константы	3	3
XRL A,Rn	Исключающее ИЛИ аккумулятора и регистра	1	1
XRL A,direct	Исключающее ИЛИ аккумулятора и прямоадресуемого байта	2	2
XRL A,@Ri	Исключающее ИЛИ аккумулятора и косвенноадресуемого байта ОЗУ	1	2
XRL A,#data	Исключающее ИЛИ аккумулятора и константы	2	2
XRL direct,A	Исключающее ИЛИ прямоадресуемого байта и аккумулятора	2	2
XRL direct,#data	Исключающее ИЛИ прямоадресуемого байта и константы	3	3
CLR A	Сброс аккумулятора	1	1
CPL A	Инверсия аккумулятора	1	1
RL A	Сдвиг аккумулятора влево циклический	1	1
RLC A	Сдвиг аккумулятора влево через перенос	1	1
RR A	Сдвиг аккумулятора вправо циклический	1	1
RRC A	Сдвиг аккумулятора вправо через перенос	1	1
SWAP A	Обмен местами тетрад в аккумуляторе	1	1
КОМАНДЫ ПЕРЕДАЧИ ДАННЫХ			
MOV A,Rn	Пересылка в аккумулятор из регистра (n = 0…7)	1	1
MOV A,direct	Пересылка в аккумулятор прямоадресуемого байта	2	2
MOV A,@Ri	Пересылка в аккумулятор косвенноадресуемого байта ОЗУ	1	2
MOV A,#data	Загрузка в аккумулятор константы	2	2
MOV Rn,A	Пересылка в регистр из аккумулятора	1	1
MOV Rn,direct	Пересылка в регистр прямоадресуемого байта	2	2

Продолжение таблицы 6.1

1	2	3	4
MOV Rn,#data	Загрузка в регистр константы	2	2
MOV direct,A	Пересылка по прямому адресу аккумулятора	2	2
MOV direct,Rn	Пересылка по прямому адресу регистра	2	2
MOV direct,direct	Пересылка прямоадресуемого байта по прямому адресу	3	3
MOV direct,@Ri	Пересылка косвенноадресуемого байта ОЗУ по прямому адресу	2	2
MOV direct,#data	Пересылка по прямому адресу константы	3	3
MOV @Ri,A	Пересылка в косвенноадресуемую ячейку ОЗУ аккумулятора	1	2
MOV @Ri,direct	Пересылка в косвенноадресуемую ячейку ОЗУ прямо-адресуемого байта	2	2
MOV @Ri,#data	Пересылка в косвенноадресуемую ячейку ОЗУ константы	2	2
MOV DPTR,#data16	Загрузка указателя данных	3	3
MOVC A,@A+DPTR	Пересылка в аккумулятор байта из памяти программ	1	3
MOVC A,@A+PC	Пересылка в аккумулятор байта из памяти программ	1	3
MOVX A,@Ri	Пересылка в аккумулятор байта из внешней памяти данных	1	3
MOVX @Ri,A	Пересылка байта из аккумулятора во внешнюю память данных	1	3
MOVX A,@DPTR	Пересылка в аккумулятор из расширенной внешней памяти данных	1	3
MOVX @DPTR,A	Пересылка из аккумулятора в расширенную внешнюю память данных	1	3
PUSH direct	Загрузка в стек	2	2
POP direct	Извлечение из стека	2	2
XCH A,Rn	Обмен аккумулятора с регистром	1	1
XCH A,direct	Обмен аккумулятора с прямоадресуемым байтом	2	2
XCH A,@Ri	Обмен аккумулятора с косвенноадресуемым байтом ОЗУ	1	2
XCHD A,@Ri	Обмен младшей тетрады аккумулятора с младшей тетрадой косвенно-адресуемого байта ОЗУ	1	2
ОПЕРАЦИИ С БИТАМИ			
CLR C	Сброс переноса	1	1
CLR bit	Сброс бита	2	2
SETB C	Установка переноса	1	1
SETB bit	Установка бита	2	2
CPL C	Инверсия переноса	1	1
CPL bit	Инверсия бита	2	2
ANL C,bit	Логическое И бита и переноса	2	2
ANL C,/bit	Логическое И инверсии бита и переноса	2	2

Продолжение таблицы 6.1

1	2	3	4
ORL C,bit	Логическое ИЛИ бита и переноса	2	2
ORL C,/bit	Логическое ИЛИ инверсии бита и переноса	2	2
MOV C,bit	Пересылка бита в перенос	2	2
MOV bit,C	Пересылка переноса в бит	2	2
JC rel	Переход, если перенос равен единице	2	2/3
JNC rel	Переход, если перенос равен нулю	2	2/3
JB bit,rel	Переход, если бит равен единице	3	3/4
JNB bit,rel	Переход, если бит равен нулю	3	3/4
JBC bit,rel	Переход, если бит установлен, с последующим сбросом бита	3	3/4
ПРОГРАММНЫЕ ПЕРЕХОДЫ			
ACALL addr11	Абсолютный вызов подпрограммы в пределах страницы в 2 Кбайта	2	3
LCALL addr16	Длинный вызов подпрограммы	3	4
RET	Возврат из подпрограммы	1	5
RETI	Возврат из подпрограммы обработки прерывания	1	5
AJMP addr11	Абсолютный переход внутри страницы в 2 Кбайта	2	3
LJMP addr16	Длинный переход в полном объеме памяти программ	3	4
SJMP rel	Короткий относительный переход внутри страницы в 256 байт	2	3
JMP @A+DPTR	Косвенный относительный переход	1	3
JZ rel	Переход, если аккумулятор равен нулю	2	2/3
JNZ rel	Переход, если аккумулятор не равен нулю	2	2/3
CJNE A,direct,rel	Сравнение аккумулятора с прямо-адресуемым байтом и переход, если не равно	3	3/4
CJNE A,#data,rel	Сравнение аккумулятора с константой и переход, если не равно	3	3/4
CJNE Rn,#data,rel	Сравнение регистра с константой и переход, если не равно	3	3/4
CJNE @Ri,#data,rel	Сравнение косвенноадресуемого байта ОЗУ с константой и переход, если не равно	3	4/5
DJNZ Rn,rel	Декремент регистра и переход, если не нуль	2	2/3
DJNZ direct,rel	Декремент прямоадресуемого байта и переход, если не нуль	3	3/4
NOP	Холостая команда	1	1

Условные обозначения:

Rn - Регистр R0-R7 выбранного банка регистров.

@Ri – Ячейка ОЗУ данных, адресуемая косвенно через регистры R0-R1

rel – 8-битное смещение со знаком (в дополнительном коде) относительно первого байта следующей команды. Используется командой SJMP и всеми командами условных переходов.

Direct – 8-битный адрес ячейки внутреннего ОЗУ данных. Это может быть ячейка ОЗУ данных прямого доступа (0x00-0x7F) или регистр специального назначения SFR (0x80-0xFF).

#data – 8-битная константа

#data 16 – 16-битная константа

bit – Прямо-адресуемый бит ячейки ОЗУ данных или регистра специального назначения SFR.

addr 11 – 11-битный адрес перехода, используемый командами ACALL и AJMP. Переход должен осуществляться в пределах той 2-Кбайтной страницы памяти программ, в которой расположен первый байт следующей команды.

addr 16 – 16-битный адрес перехода, используемый командами LCALL и LJMP. Переход может осуществляться в пределах всего 64-Кбайтного пространства памяти программ.

Существует один неиспользуемый код операции (0xA5), который исполняется аналогично команде NOP.

Рассмотрим пример программы на языке Ассемблер для микроконтроллера C8051F320. При выполнении данной программы задействованы следующие периферийные модули микроконтроллера: Сторожевой Таймер (WDT), Таймер 0 (Timer0), тактовый генератор (Oscillator), модуль обработки прерываний (Interrupts), порты (P2.0 – кнопка, P2.2 – светодиод)

В общем виде код программы можно разбить на две части:

1 Декларативная часть, которая включает описание векторов прерываний, определение констант, названий переменных, пинов и т.д.

2 Исполняемый программный код.

Рассмотрим декларативную часть. Вначале подключаем файл с описанием регистров специальных функций.

```
$include (C8051F320.inc)      ; Include regsiter definition file.

;*************** определение векторов прерываний ***************

d_ResetVect     equ     000h       ; вектор сброса

d_Timer0Vect    equ     00bh       ; вектор прерывания от таймера 0

;*************** определение констант ********************

d_Timer0      equ     0bdch        ;  частота 2 Гц (деление на 48)

d_Timer1      equ     0fb1eh            ; частота 100 Гц (деление
на 48)

;************** определение имен переменных **************
```

```
CntTime    data     030h          ; счет времени между двумя пре-
рываниями INT0
;****************** определение имен выводов********************
pLED        bit     P2.2          ; вывод светодиода
pButton       bit     P2.0              ; вывод кнопки
;Исполняемый код программы может включать следующие составные части:
;1) Программные переходы по векторам прерываний
;2) Процедуры обработки прерываний
;3) Процедуры и функции
;4) Макросы
;5) Функция инициализации микроконтроллера
;6) Головная программа
;********* Программные переходы по векторам прерываний *********
    org d_ResetVect      ; переход если системный сброс
    ljmp FnMain          ; первая выполняемая команда
    org d_Timer0Vect     ; прерывание от таймера 0
    ljmp IntTimer0       ; переход на обработку прерывания от тай-
мера 0
;************* Процедуры обработки прерываний ****************
IntTimer0:
  push PSW               ; Сохранение в стек значений ACC и PSW
  push ACC
  setb RS0               ; устанавливаем банк регистров №1
  clr TR0                ; Перезагрузка Таймера 0
  mov TL0,#LOW(d_Timer0)
  mov TH0,#HIGH(d_Timer0)
  setb TR0
  ;*** Рабочий цикл прерывания***
   cpl pLED
  ;*** Рабочий цикл прерывания***
lb_iT0_end:
```

```
  pop ACC                ; выгрузка из стека значений ACC и PSW
  pop PSW
  reti
;**************** Процедуры и функции******************
;Вход R0 - время задержки в десятках милисекунд
TimeWeit:
lb_TmrW_TimeCount:
   mov TL1,#LOW(d_Timer1)     ;перезагрузка Таймера 1 .
   mov TH1,#HIGH(d_Timer1)
   clr TF1
   setb TR1
lb_TmrW_T1Work:
  nop
  jnb TF1,lb_TmrW_T1Work      ; Ожидание переполнения Таймера 1
  djnz R0,lb_TmrW_TimeCount   ; Счет переполнений Таймера 1
  ret
;***** Функции инициализации микроконтроллера ************
Init8051:
  lcall PCA_Init              ; инициализация WDT
  lcall Timer_Init            ; инициализация Таймера 0
  lcall Oscillator_Init  ; инициализация внутреннего генератора
  lcall Interrupts_Init       ; инициализация прерываний
  lcall Port_IO_Init
  ret
PCA_Init:
  anl PCA0MD,  #0BFh          ; отключаем WDT
  mov PCA0MD,  #000h
  ret
Timer_Init:
  mov TMOD,   #011h           ; Работаем с T0 режим 1; T1 режим 1
```

```
  mov CKCON,  #002h  ; Используем делитель системной частоты на 48
  mov TL0,#LOW(d_Timer0)  ; Начальная загрузка константы в Таймер 0
  mov TH0,#HIGH(d_Timer0)
  mov TL1,#LOW(d_Timer1) ; Начальная загрузка константы в Таймер 1
  mov TH1,#HIGH(d_Timer1)
  setb TR0
  ret
Oscillator_Init:
  mov OSCICN,  #082h     ; Используем внутренний тактовый генератор
  ret
Interrupts_Init:
  mov IE,    #082h       ; Разрешаем общие прерывания и прерывания
от Таймера 0
  ret
Port_IO_Init:
  mov P2MDOUT,  #004h    ; Включаем подтяжку
  mov XBR1, #040h        ; Включаем Crossbar
  ret
;***************** Головная программа *****************
FnMain:
  mov SP,#010h           ; Инициализируем стек
  lcall Init8051         ; Вызываем инициализацию микроконтроллера
lb_main_Loop:            ; Ожидаем нажатия кнопки
    nop
   jb pButton,lb_main_Loop
   mov R0,#30    ; Вызов задержки для устранения дребезга контактов
   lcall TimeWeit
   cpl TR0               ; Инверсия состояния Таймера 0
lb_main_WeitOffButt:     ; Ожидание отпускания кнопки
    nop
```

```
    jnb pButton,lb_main_WeitOffButt
    mov R0,#30    ; Вызов задержки для устранения дребезга контактов
    lcall TimeWeit
  sjmp lb_main_Loop      ; Переход на бесконечный цикл
end
```

В результате выполнения программы происходит мерцание светодиода с заданной частотой включения/выключения (определяется константой Таймера 0). При нажатии на кнопку мерцание прекращается. Повторное нажатие снова запускает мерцание светодиода.

6.3 Обработка прерываний в микроконтроллерах Cygnal семейства C8051F320

Процессорное ядро CIP-51 имеет развитую систему прерываний, поддерживающую в общей сложности 16 источников прерываний с двумя уровнями приоритета. Распределение источников прерываний между встроенными периферийными модулями и внешними входными выводами зависит от конкретного типа МК. Каждый источник прерываний имеет один или несколько связанных с ним флагов прерываний, размещенных в SFR. Когда периферийный модуль или внешний источник прерываний регистрирует событие, удовлетворяющее условию прерывания, соответствующий флаг прерывания устанавливается в 1.

Если прерывание от источника прерываний разрешено, то при установке флага прерывания генерируется запрос прерывания. Как только выполнение текущей команды завершится, будет сгенерирована команда LCALL перехода по предопределенному адресу, откуда начнется исполнение процедуры обслуживания прерывания (interrupt service routine – ISR). Каждая ISR должна заканчиваться командой RETI, которая возвращает управление прерванной программе и приводит к выполнению той команды, которая исполнилась бы, если бы запроса прерывания не было. Если прерывания не разрешены, флаг прерывания игнорируется и выполнение программы продолжается в нормальном режиме. (Флаг прерывания устанавливается в 1 независимо от того, разрешены прерывания или запрещены).

Прерывание от каждого источника прерываний может быть разрешено или запрещено с помощью соответствующих битов разрешения прерываний в регистрах SFR (IE-EIE2). Однако сначала прерывания необходимо разрешить глобально, установкой в 1 бита EA (IE.7), и только после этого состояние индивидуальных флагов разрешения прерываний будет иметь силу. Сброс в 0 бита EA запрещает прерывания от всех источников прерываний независимо от состояния индивидуальных флагов разрешения прерываний.

Некоторые флаги прерываний сбрасываются автоматически аппаратными средствами при переходе к процедуре ISR. Однако большинство флагов преры-

ваний не сбрасываются аппаратно и должны быть сброшены программно до возвращения из процедуры ISR. Если флаг прерывания остается установленным после завершения выполнения команды возврата из прерывания (RETI), то сразу же будет сгенерирован новый запрос прерывания и после завершения выполнения следующей команды произойдет повторный переход к процедуре ISR.

Данное семейство МК поддерживает 16 источников прерываний. Программа может симулировать прерывание установкой в 1 любого флага прерывания. Если прерывание для этого флага разрешено, будет сгенерирован запрос прерывания и произойдет переход по адресу процедуры ISR, связанной с этим флагом прерывания. Источники прерываний МК, соответствующие им адреса прерываний, уровень приоритета и биты управления перечислены в таблице 6.2.

6.3.1 Задержка обработки прерывания

Время реакции на прерывание зависит от состояния процессорного ядра в момент возникновения прерывания. Опрос флага прерывания и декодирование приоритета осуществляется каждый системный тактовый цикл. Поэтому наименее возможное время реакции на прерывание составляет 5 тактовых циклов: 1 цикл для определения прерывания и 4 цикла для выполнения команды LCALL перехода к процедуре ISR. Если в момент выполнения команды RETI появляется прерывание, то до выполнения команды LCALL перехода на процедуру обслуживания этого прерывания будет исполнена одна команда основной программы. Поэтому максимальное время реакции на прерывание (если в настоящий момент не обслуживается другое прерывание или новое прерывание имеет более высокий приоритет) будет тогда, когда выполняется команда RETI, а следом за ней должна выполняться команда DIV. В этом случае время реакции составляет 18 тактовых циклов: 1 цикл для определения прерывания, 5 циклов для выполнения команды RETI, 8 циклов для выполнения команды DIV и 4 цикла для выполнения команды LCALL перехода на процедуру ISR. Если выполняется процедура ISR для прерывания с равным или более высоким приоритетом, новое прерывание не будет обслужено до тех пор, пока не завершится текущая процедура ISR, включая команду RETI и следующую команду.

Таблица 6.2 – Источники прерываний

Источник прерывания	**Вектор прерывания**	**Приоритет**	**Флаг прерывания**	**Битовая адресация?**	**Аппаратный сброс?**	**Бит разрешения**	**Управление приоритетом**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Сброс	0x0000	max	Нет	N/A	N/A	Разрешен всегда	Всегда наивысший

Продолжение таблицы 6.2

1	2	3	4	5	6	7	8
Внешнее прерывание 0 (/INT0)	0x0003	0	IE0 (TCON.1)	Y	Y	EX0 (IE.0)	PX0 (IP.0)
Переполнение Таймера 0	0x000B	1	TF0 (TCON.5)	Y	Y	ET0 (IE.1)	PT0 (IP.1)
Внешнее прерывание 1 (/INT1)	0x0013	2	IE1 (TCON.3)	Y	Y	EX1 (IE.2)	PX1 (IP.2)
Переполнение Таймера 1	0x001B	3	TF1 (TCON.7)	Y	Y	ET1 (IE.3)	PT1 (IP.3)
Последовательный порт УАПП0	0x0023	4	RI0 (SCON0.0) TI0 (SCON0.1)	Y	N	ES0 (IE.4)	PS0 (IP.4)
Переполнение Таймера 2	0x002B	5	TF2H (TMR2CN.7) TF2L (TMR2CN.6)	Y	N	ET2 (IE.5)	PT2 (IP.5)
Модуль SPI0	0x0033	6	SPIF (SPI0CN.7) WCOL (SPI0CN.6) MODF (SPI0CN.5) RXOVRN (SPI0CN.4)	Y	N	ESPI0 (IE.6)	PSPI0 (IP.6)
Модуль SMBus0	0x003B	7	SI (SMB0CN.0)	Y	N	ESMB0 (EIE1.0)	PSMB0 (EIP1.0)
Детектор диапазона АЦП0	0x004B	9	ADWINT (ADC0CN.3)	Y	N	EWADC0 (EIE1.2)	PWADC0 (EIP1.2)
Завершение преобразования АЦП0	0x0053	10	ADC0INT (ADC0CN.5)	Y	N	EADC0 (EIE1.3)	PADC0 (EIP1.3)
Переполнение Таймера 3	0x0073	14	TF3H (TMR3CN.7) TF3L (TMR3CN.6)	N	N	ET3 (EIE1.7)	PT3 (EIP1.7)
Уровень VBUS	0x007B	15	N/A	N/A	N/A	EVBUS (EIE2.0)	PVBUS (EIP2.0)

Примеры работы с прерываниями будут рассмотрены при описании конкретных модулей микроконтроллера.

6.4 Порты ввода/вывода в микроконтроллерах Cygnal семейства C8051F320

Для доступа к аналоговым и цифровым ресурсам МК используются 17 внешних линий ввода/вывода, которые организованы как два 8-разрядных пор-

та и один 1-разрядный порт. Каждый из выводов портов можно определить как порт ввода/вывода общего назначения (GPIO) или аналоговый вход; выводы портов P0.0 – P1.7 можно назначить одному из внутренних цифровых модулей, как показано на рисунке 6.3. Разработчик системы сам определяет, какие цифровые ресурсы будут назначены внешним выводам, ограничиваясь только количеством доступных выводов. Такая гибкость при распределении ресурсов достигается благодаря использованию приоритетного декодера матрицы. Следует иметь ввиду, что состояние на внешнем выводе порта всегда можно прочитать из соответствующего регистра-защелки порта независимо от настройки матрицы.

Матрица назначает внешние порты ввода/вывода выбранным внутренним цифровым ресурсам, используя приоритетный декодер (см. рисунки 3 и 4). Для выбора внутренних цифровых ресурсов используются регистры XBR0 и XBR1.

Допустимое напряжение на внешних выводах портов составляет 5(см. схему ячейки порта на рисунке 6.4). С помощью регистров настройки выходов портов (PnMDOUT, где n = 0,1) можно настроить выходные драйверы портов ввода/вывода либо как обычные цифровые выходы, либо как выходы с открытым стоком.

6.4.1 Приоритетный декодер матрицы

Приоритетный декодер матрицы (см. рисунок 6.3) назначает приоритет каждой функции ввода/вывода начиная с выводов UART0. Если какой-либо цифровой ресурс выбран, то этому ресурсу назначается еще не назначенный внешний вывод с наименьшим приоритетом (кроме UART0, которому всегда назначаются выводы P0.4 и P0.5).

Если вывод порта назначен, то матрица пропускает этот вывод при назначении выводов следующему выбранному ресурсу. Кроме того, матрица будет пропускать выводы портов, если соответствующие им биты в регистрах PnSKIP установлены в 1. Регистры PnSKIP позволяют программе настроить матрицу таким образом, чтобы она пропускала выводы портов, которые используются в качестве аналоговых входов, портов ввода/вывода общего назначения или для реализации специальных функций.

Важное замечание относительно конфигурации матрицы: Если вывод порта закреплен за периферийным модулем без использования матрицы, то соответствующий ему бит в регистрах PnSKIP должен быть установлен в 1. Это касается P0.0, если используется VREF, P0.3 и/или P0.2, если включена схема возбуждения внешнего генератора, P0.6, если АЦП или ЦАП настроены на использование внешнего сигнала запуска преобразования (CNVSTR), а также любых выбранных входов АЦП или компаратора. Матрица пропускает выбранные выводы, как если бы они были уже назначены, и переходит к следующему не назначенному выводу. На рисунке 6.5 показаны приоритеты декодера матрицы без пропуска каких-либо выводов портов (P0SKIP, P1SKIP = 0x00); на рисунке

6.6 показаны приоритеты декодера матрицы с пропуском выводов XTAL1 (P0.2) и XTAL2 (P0.3) (P0SKIP = 0x0C).

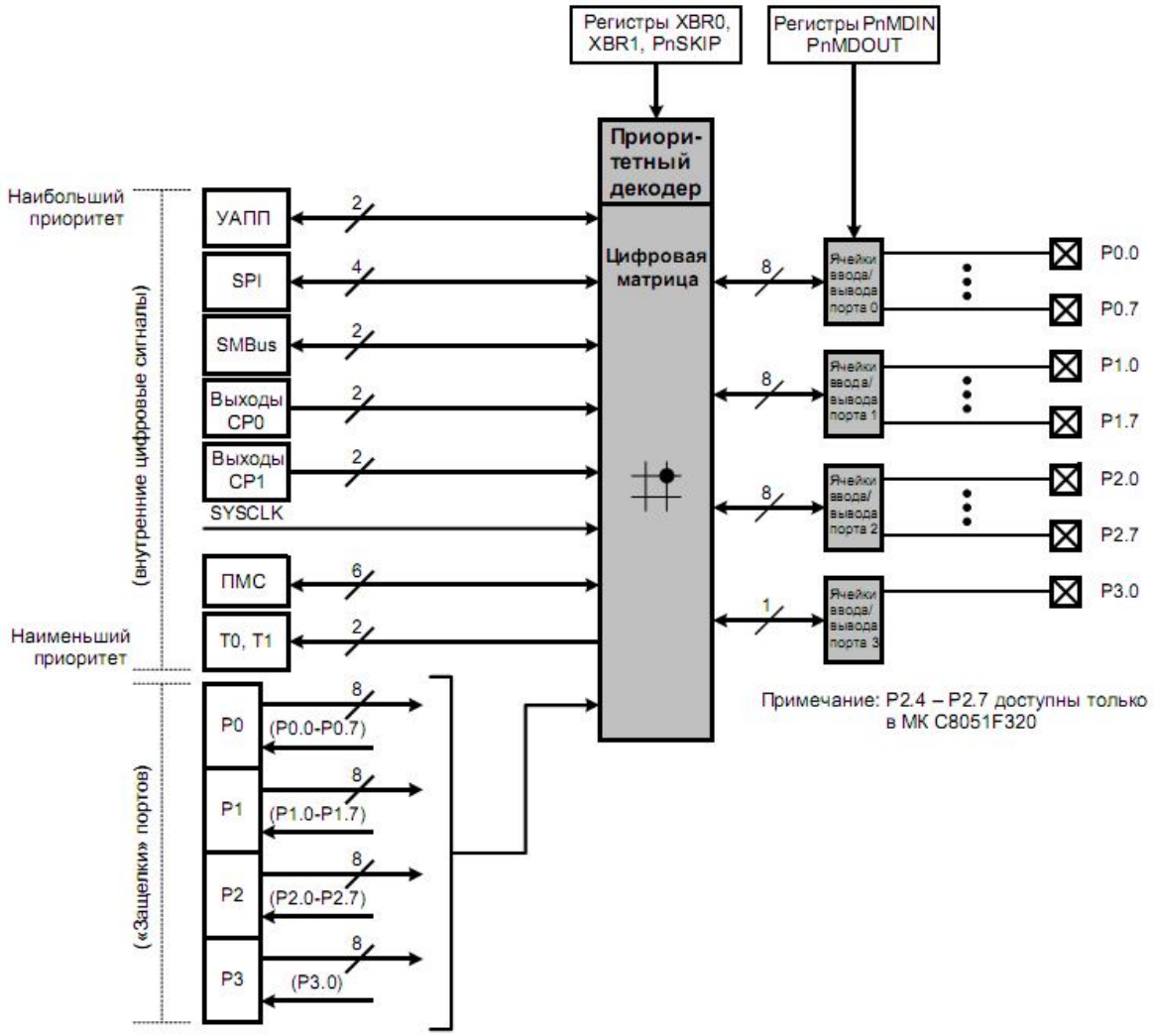


Рисунок 6.3 – Структурная схема коммутации ресурсов на внешние выводы

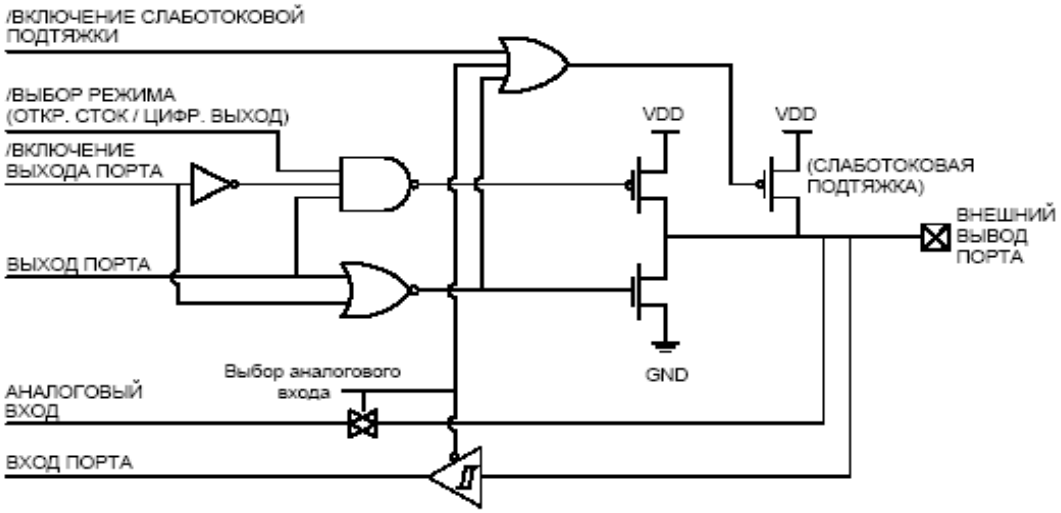


Рисунок 6.4 – Структурная схема ячейки порта ввода/вывода

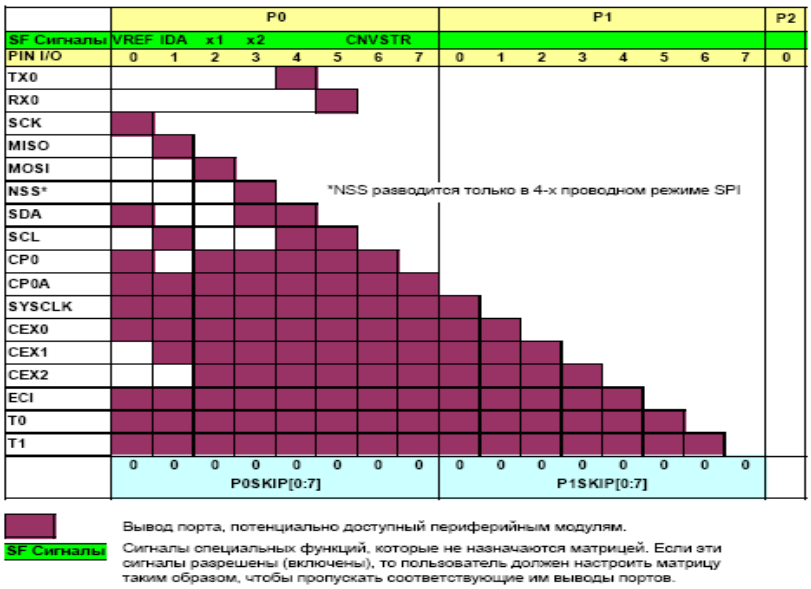


Рисунок 6.5 – Приоритетный декодер матрицы (нет пропускаемых выводов)

Регистры XBR0 и XBR1 используются для назначения цифровых ресурсов внешним портам ввода/вывода. Следует иметь ввиду, что если выбран SMBus, то матрица назначает оба вывода, связанные с модулем SMBus (SDA и SCL); если выбран UART0, то матрица назначает оба вывода, связанные с модулем UART0 (TX и RX). Назначение выводов UART0 фиксировано с целью обеспечения возможности самозагрузки: TX0 всегда назначается выводу P0.4; RX0 всегда назначается выводу P0.5. После назначения приоритетных функций следуют стандартные порты ввода/вывода.

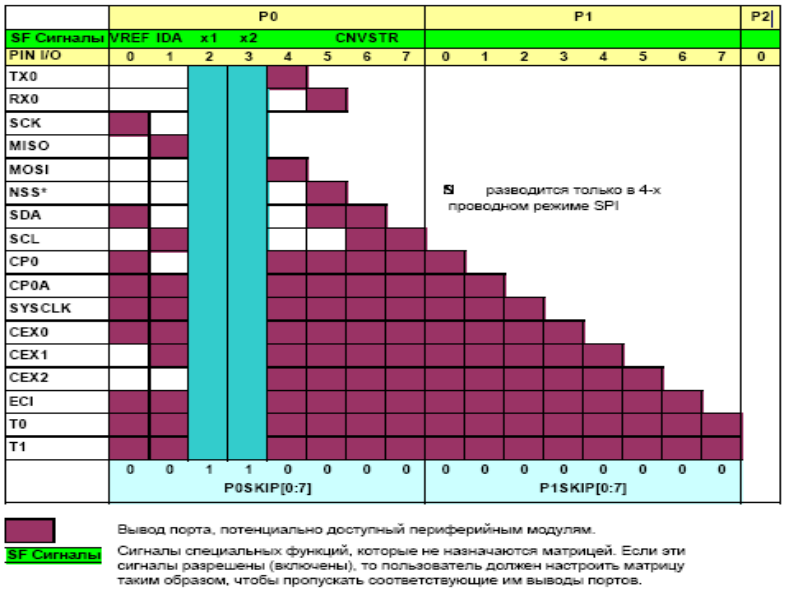


Рисунок 6.6 – Приоритетный декодер матрицы (пропускаются выводы подключения кварцевого резонатора)

Важное замечание: Модуль SPI может функционировать в 3- или 4-хпроводном режимах, в зависимости от состояния битов NSSMD1-NSSMD0 в регистре SPI0CN. В соответствии с режимом работы SPI сигнал NSS либо разводится, либо не разводится на внешний порт.

6.4.2 Порт ввода/вывода общего назначения

Выводы портов, которые не назначены матрицей и не используются аналоговыми периферийными модулями, можно использовать в качестве выводов ввода/вывода общего назначения. Порты 3-0 доступны с помощью соответствующих SFR-регистров как в побайтном, так и в побитном режимах адресации. При записи в порт значение, записываемое в SFR-регистр, «защелкивается»; это позволяет удерживать на каждом выводе порта выходное значение. При чтении логические уровни входных выводов портов возвращаются независимо от значений регистров XBRn (т.е. даже если вывод назначен матрицей другому сигналу, регистр порта все равно может прочитать логическое состояние на соответствующем входе). Исключением являются команды типа *чтение-модификация-запись*. При работе с SFR-регистром порта командами типа *чтение-модификация-запись* являются следующие команды: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ, а также MOV, CLR или SETB, если они адресуют отдельный бит в SFR-регистре порта. В случае использования этих команд считывается, модифицируется и записывается обратно значение регистра (а не вывода).

Инициализация порта ввода/вывода осуществляется следующим образом:

1 Выбрать тип входа (аналоговый или цифровой) для всех выводов порта, используя регистр настройки входов порта (PnMDIN).

2 Выбрать тип выхода (с открытым стоком или двухтактный цифровой) для всех выводов порта, используя регистр настройки выходов порта (PnMDOUT).

3 Выбрать все выводы, которые должны пропускаться матрицей при назначении выводов, используя регистры выбора пропускаемых выводов (PnSKIP).

4 Назначить выводы порта требуемым периферийным модулям (XBR0, XBR1).

5 Включить матрицу (XBARE = ‘1’).

Все выводы порта должны быть настроены как аналоговые или как цифровые входы. Любые выводы, используемые в качестве входов компаратора или АЦП, должны быть настроены как аналоговые входы. Если вывод настроен как аналоговый вход, то его слаботоковая подтяжка, цифровой драйвер и цифровой приемник отключаются. Это позволяет снизить энергопотребление и уменьшить уровень шумов на аналоговом входе.

Выводы, настроенные как цифровые входы, все равно могут использоваться аналоговыми периферийными модулями, однако это не рекомендуется. Чтобы настроить вывод порта как цифровой вход, следует сбросить в 0 соответствую-

щий бит в регистре PnMDOUT и установить в 1 соответствующий бит регистра-защелки порта (регистр Pn).

Кроме этого, все аналоговые входы необходимо настроить таким образом, чтобы они пропускались матрицей при назначении выводов (это достигается установкой в 1 соответствующих бит в регистрах PnSKIP).

Тип входа устанавливается с помощью соответствующих битов регистра PnMDIN (1 – цифровой вход, 0 – аналоговый вход). При сбросе все выводы настраиваются по умолчанию как цифровые входы. Параметры выходных драйверов выводов порта задаются с помощью регистров настройки выходов порта (PnMDOUT). Выходной драйвер каждого порта можно настроить либо как цифровой двухтактный выход, либо как выход с открытым стоком. Такая настройка не осуществляется автоматически, ее необходимо выполнить даже для цифровых ресурсов, выбранных в регистрах XBRn. Единственным исключением из этого правила являются выводы SMBus (SDA, SCL), которые настраиваются как выходы с открытым стоком независимо от значения PnMDOUT. Если бит WEAKPUD в регистре XBR1 сброшен в 0, то слаботоковая подтяжка отключается у всех выводов портов, настроенных как выходы с открытым стоком. Бит WEAKPUD не влияет на выводы, настроенные как цифровые двухтактные выходы. Более того, слаботоковая подтяжка отключается у выхода, на который выведен лог. «0», чтобы предотвратить нежелательное увеличение энергопотребления.

Для выбора цифровых ресурсов, требуемых для конкретного проекта, необходимо загрузить регистры XBR0 и XBR1 соответствующими значениями. Установка в 1 бита XBARE в регистре XBR1 включает матрицу. До включения матрицы внешние выводы остаются стандартными портами ввода/вывода (настроенными на вход) независимо от значений регистров XBRn. Зная значение регистров XBRn, можно определить разводку выводов, используя таблицу декодирования приоритетов.

Важное замечание: Чтобы использовать порты P0, P1 и P2.0 – P2.3 как стандартные порты ввода/вывода в режиме выходов, необходимо включить матрицу. Выходные драйверы этих портов отключаются при выключении матрицы. P2.4 – P2.7 и P3.0 всегда функционируют как стандартные порты ввода/вывода общего назначения.

6.4.3 Пример программы для работы с портами ввода-вывода

Программа написана на языке С. Микроконтроллер управляет кнопкой и светодиодом. При нажатии кнопки светодиод загорается, при отпускании – гаснет. Также рассмотрены функции инициализации портов ввода/вывода и тактового генератора.

```
#include <c8051f300.h>        // файл с описанием регистров спец.
функций

//-----------------------------------------------------------
```

```
// Описание выводов микроконтроллера (Pin Declarations)
sbit LED1 =  P0^2;           // LED1 ='1' значит светодиод горит
sbit SW1  =  P0^3;           // SW1 ='0' значит кнопка нажата
//---------------------------------------------------------------
// Описание функций (Function Prototypes)
void OSCILLATOR_Init (void);
void PORT_Init (void);
//---------------------------------------------------------------
// Главная функция программы
void main (void)
{
   PCA0MD &= ~0x40;          // выключаем WDT
   PORT_Init();              // инициализация портов ввода-вывода
   OSCILLATOR_Init ();       // инициализация тактового генератора
   while (1)
   {
      if (SW1 == 0)          // если кнопка нажата
      {
         LED1 = 1            // зажечь светодиод
      }
       else
      {
         LED1 = 0;           // иначе - выключить
      }
   }                         // конец цикла while(1)
}                            // конец главной функции main()
//---------------------------------------------------------------
// Функции инициализации
void OSCILLATOR_Init (void)
{
```

```
   OSCICN |= 0x03;             // Конфигурация тактового генератора
                               // на максимальную частоту (24.5 Mhz)
}
void PORT_Init (void)
{
   P0MDIN |= 0x0C;             // P0.2 и P03. цифровые
   P0MDOUT = 0x04;             // P0.2 включена подтяжка
   P0     |= 0x08;             // Установка в P0.3 значения '1'
   XBR2    = 0x40;             // Включение приоритетного декодера
матрицы
}
```

6.5 Таймер/счетчики в микроконтроллерах Cygnal семейства C8051F320

МК содержит четыре таймера/счетчика (Т/С): два из них представляют собой 16-разрядные Т/С, совместимые с Т/С стандартной архитектуры 8051, а другие два являются 16-разрядными Т/С с режимом автоперезагрузки и предназначены для использования совместно с АЦП, SMBus, USB (для измерения фреймов), а также в качестве Т/С общего назначения. Эти Т/С можно использовать для измерения временных интервалов, подсчета внешних событий, а также для генерации периодических запросов прерываний. Таймер 0 и Таймер 1 почти идентичны и имеют четыре основных режима работы. Таймер 2 и Таймер 3 могут работать (каждый) как один 16-разрядный таймер или как два 8-разрядных таймера, причем во всех случаях поддерживается режим автоперезагрузки. Рассмотрим подробно Таймер 0 и Таймер 1.

Режимы Таймера 0 и Таймера 1:

1 13-разрядный Т/С

2 16-разрядный Т/С

3 8-разрядный Т/С с автоперезагрузкой

4 Два 8-разрядных Т/С (только Таймер 0)

Таймеры 0 и 1 могут тактироваться от одного из пяти источников, выбор которых осуществляется с помощью битов выбора режима таймера (T1M – T0M) и битов выбора коэффициента деления тактовой частоты (SCA1 – SCA0). Биты выбора коэффициента деления тактовой частоты настраивают предварительный делитель тактовой частоты, сигнал с выхода которого может использоваться для тактирования Таймера 0 и/или Таймера 1.

В качестве сигнала тактирования Таймеров 0 и 1 можно выбрать либо сигнал с выхода предварительного делителя тактовой частоты, либо системный тактовый сигнал. Таймер 0 и Таймер 1 могут также функционировать как счетчики. В этом случае регистр таймера/счетчика инкрементируется под воздействием каждого перехода внешнего сигнала на выбранном входном выводе (T0

или T1) из состояния лог. «1» в состояние лог. «0». Могут подсчитываться импульсы с частотой до 1/4 системной тактовой частоты. Входной сигнал не обязательно должен быть периодическим, однако он должен удерживаться на заданном уровне как минимум в течение двух полных системных тактовых циклов, чтобы гарантировать его корректную выборку.

Каждый таймер реализован в виде 16-разрядного регистра, доступного как два отдельных байта: младший байт (TL0 или TL1) и старший байт (TH0 или TH1). Регистр управления T/C (TCON) используется для включения Таймера 0 и Таймера 1, а также для индикации их состояния. Прерывания от Таймера 0 можно включить установкой в 1 бита ET0 в регистре IE. Прерывания от Таймера 1 можно включить установкой в 1 бита ET1 в регистре IE. Оба таймера/счетчика работают в одном из четырех основных режимов, задаваемых битами выбора режима T1M1 – T0M0 в регистре режима T/C (TMOD). Каждый T/C может быть настроен независимо от другого.

Рассмотрим подробно Режим 0,1 и 2 (13-разрядный T/C, 16-разрядный T/C, 8-разрядный таймер/счетчик с автоперезагрузкой), который будет использован при выполнении лабораторной работы.

В *режиме 0* Таймеры 0 и 1 работают как 13-разрядный таймер/счетчик. Ниже приводится описание настройки и функционирования Таймера 0. Однако оба таймера идентичны, и Таймер 1 настраивается точно так же, как и Таймер 0.

Регистр TH0 содержит восемь старших бит 13-разрядного значения регистра T/C. Регистр TL0 содержит в разрядах TL0.4-TL0.0 пять младших бит 13-разрядного значения регистра T/C. Три старших бита регистра TL0 (TL0.7-TL0.5) не определены и должны маскироваться или игнорироваться при чтении регистра TL0. При инкрементировании 13-разрядного таймера и переполнении его из состояния 0x1FFF (все единицы) в состояние 0x0000 установится в 1 флаг переполнения таймера TF0 (TCON.5) и будет сгенерировано прерывание, если оно разрешено. Бит C/T0 (TMOD.2) выбирает источник сигнала тактирования T/C0. Если бит C/T0 установлен в 1, то инкрементирование регистра таймера осуществляется под воздействием перехода внешнего сигнала на выбранном входном выводе (T0) из состояния лог. «1» в состояние лог. «0». Если бит C/T0 сброшен в 0, то в качестве источника тактирования T/C0 будет использоваться сигнал, определяемый битом T0M (CKCON.3). Если бит T0M установлен в 1, то Таймер 0 тактируется системным тактовым сигналом. Если бит T0M сброшен в 0, то в качестве источника тактирования T/C0 будет использоваться сигнал, определяемый битами настройки предварительного делителя в регистре CKCON .

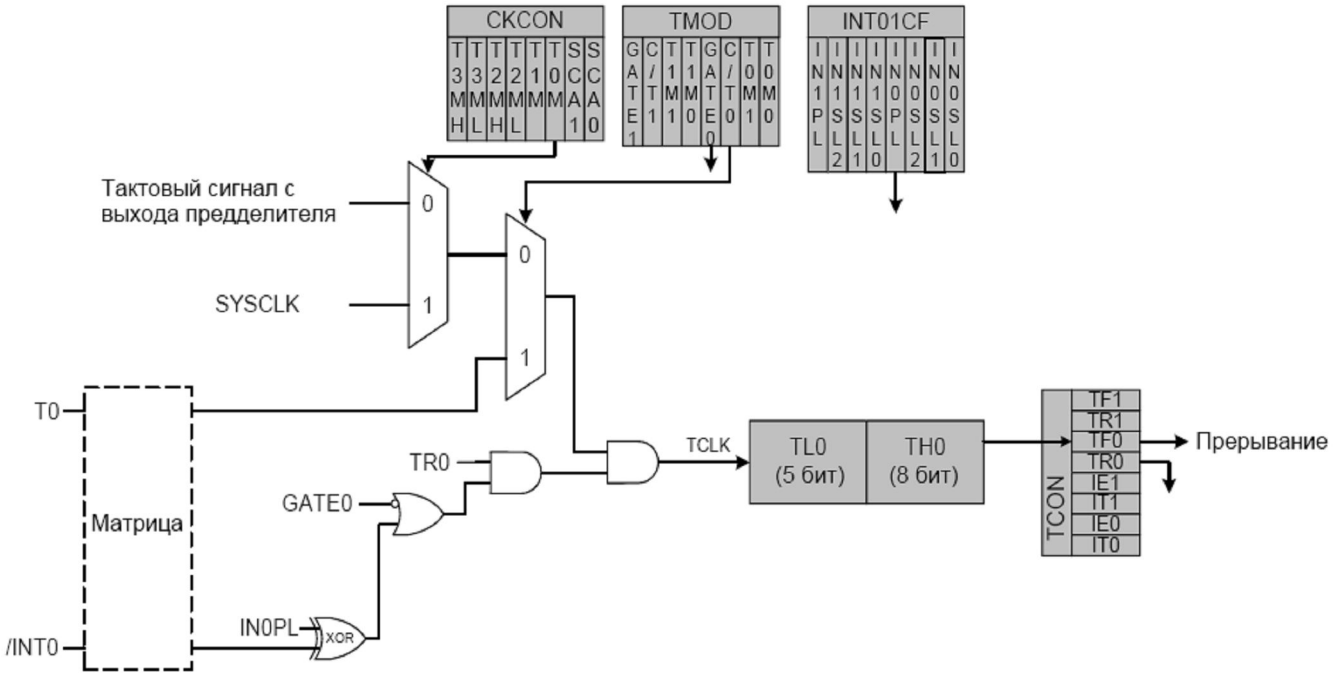


Рисунок 6.7 – Структурная схема Таймера 0 в режиме 0

Установка в 1 бита TR0 (TCON.4) включит таймер, если либо бит GATE0 (TMOD.3) равен нулю, либо на внешнем выводе /INT0 присутствует сигнал с активным логическим уровнем, который определяется битом IN0PL в регистре INT01CF. После установки в 1 бита GATE0 управление таймером передается внешнему сигналу /INT0, что позволяет легко осуществлять измерение ширины импульсов.

TR0	**GATE0**	**/INT0**	**Таймер/Счетчик**
0	X	X	Отключен
1	0	X	Включен
1	1	0	Отключен
1	1	1	Включен

X = не имеет значения

Установка TR0 не сбрасывает таймер. Регистры таймера следует инициализировать необходимыми значениями до включения таймера.

TL1 и TH1 образуют 13-разрядный регистр Таймера 1 точно так же, как описано выше для регистров TL0 и TH0. Для настройки Таймера 1 и управления им используются соответствующие биты регистров TCON и TMOD таким

же образом, как и для Таймера 0. Входной сигнал /INT1 используется совместно с Таймером 1;

Полярность /INT1 определяется битом IN1PL в регистре INT01CF.

Режим 1 аналогичен режиму 0 с тем лишь исключением, что регистры T/C используют все 16 бит. Таймеры/счетчики включаются и настраиваются в режиме 1 точно так же, как в режиме 0.

В режиме 2 Таймеры 0 и 1 настраиваются для работы в качестве 8-разрядных таймеров/счетчиков с автоматической перезагрузкой начального значения.

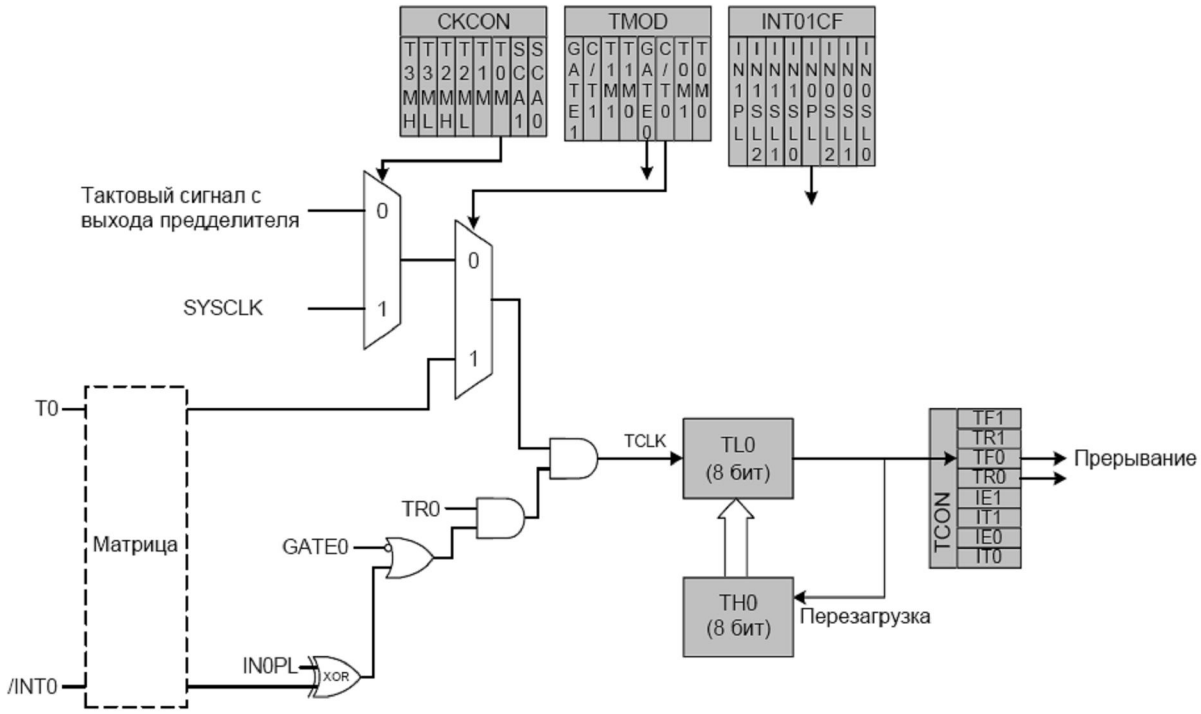


Рисунок 6.8 – Структурная схема Таймера 0 в режиме 2

Регистр TL0 содержит значение счетчика, а регистр TH0 содержит перезагружаемое значение. Когда счетчик в регистре TL0 переполняется (переходит из состояния 0xFF в состояние 0x00), флаг переполнения таймера TF0 (TCON.5) устанавливается в 1 и значение регистра TH0 загружается в регистр TL0. При установке флага TF0 будет сгенерировано прерывание, если оно разрешено. Перезагружаемое значение в регистре TH0 не изменяется. Чтобы первый отсчет был корректным, необходимо проинициализировать регистр TL0 требуемым значением до включения таймера. Таймер 1 в режиме 2 работает точно так же, как Таймер 0.

В режиме 2 оба T/C включаются и настраиваются точно так же, как и в режиме 0. Установка в 1 бита TR0 (TCON.4) включит таймер, если либо бит GATE0 (TMOD.3) равен нулю, либо на внешнем выводе /INT0 присутствует сигнал с активным логическим уровнем, который определяется битом IN0PL в регистре INT01CF.

Пример разработки программного кода для таймера 0:

```
#include <C8051F300.h>                  // SFR declarations
//-----------------------------------------------------------
// Определение констант
//-----------------------------------------------------------
#define SYSCLK 24500000/8     // SYSCLK в Герцах (24.5 MHz
                              // внутренний генератор / 8)
#define TIMER_PRESCALER  48   // выбор по значению регистра CKCON
#define LED_TOGGLE_RATE  50  // время для мерцания светодиода в
                    // миллисекундах. При LED_TOGGLE_RATE = 1,
                    //светодиод горит 1 мс. и не горит 1 мс.
// SYSCLK/TIMER_PRESCALER кол-во тактов за 1 секунду, т.о.
// SYSCLK/TIMER_PRESCALER/1000 -  кол-во тактов за 1 миллисекунду
#define TIMER_TICKS_PER_MS    SYSCLK/TIMER_PRESCALER/1000
// Важно: LED_TOGGLE_RATE*TIMER_TICKS_PER_MS не должен
// превышать 65535 (0xFFFF) для 16-битного таймера
// AUX 1 - AUX4 – промежуточные вычисления
#define AUX1     TIMER_TICKS_PER_MS*LED_TOGGLE_RATE
#define AUX2     -AUX1
#define AUX3     AUX2&0x00FF
#define AUX4     ((AUX2&0xFF00)>>8)
#define TIMER0_RELOAD_HIGH   AUX4  // Timer0 старший байт
#define TIMER0_RELOAD_LOW    AUX3  // Timer0 младший байт
sbit LED = P0^2;              // LED='1' значит светодиод горит
//-----------------------------------------------------------------
// Прототипы функций
//-----------------------------------------------------------------
void Port_Init (void);        // Инициализация портов ввода/вывода
void Timer0_Init (void);      // Инициализация Timer0
```

```
//-----------------------------------------------------------------
// Главная функция программы
//-----------------------------------------------------------------
void main (void)
{
   PCA0MD &= ~0x40;           // Остановить WDT
   Timer0_Init ();            // Инициализация Timer0
   Port_Init ();              // Инициализация портов ввода/вывода
   EA = 1;                    // Включить глобальные прерывания
   while (1);                 // Цикл программы
}
//-----------------------------------------------------------------
// Функции инициализации
//-----------------------------------------------------------------
void Port_Init (void)
{
   XBR2 = 0x40;                    // Enable crossbar
   P0MDOUT = 0x04;                 // Set LED to push-pull
}
void Timer0_Init(void)
{
   TH0 = TIMER0_RELOAD_HIGH;       // Init Timer0 High register
   TL0 = TIMER0_RELOAD_LOW ;       // Init Timer0 Low register
   TMOD = 0x01;                    // Timer0 in 16-bit mode
   CKCON = 0x02;                   // Timer0 uses a 1:48 prescaler
   ET0 = 1;                        // Timer0 interrupt enabled
   TCON = 0x10;                    // Timer0 ON
}
//-----------------------------------------------------------------
// Функции прерываний
```

```
//-----------------------------------------------------------
void Timer0_ISR (void) interrupt 1      // Timer0_ISR
{
   LED = ~LED;                        // Инвертировать светодиод
   TH0 = TIMER0_RELOAD_HIGH;          // Перезагрузка TH0
   TL0 = TIMER0_RELOAD_LOW;           // Перезагрузка TL0
}
```

6.6 10-разрядный АЦП (АЦП0).

Модуль АЦП0 МК C8051F320 состоит из двух 17-канальных аналоговых мультиплексоров (обозначаются вместе как AMUX0) и 10-разрядного АЦП последовательного приближения (максимальная производительность –200 тыс. преобразований в секунду) с интегрированным устройством выборки-хранения (УВХ) и программируемым детектором диапазона. AMUX0, режимы преобразования и детектор диапазона настраиваются программным путем при помощи регистров специального назначения (рисунок 6.9). АЦП0 функционирует как в однофазном, так и в дифференциальном режимах, и может быть настроен на измерение напряжения на выводах P1.0 – P3.0, выходного напряжения датчика температуры или напряжения VDD относительно P1.0 – P3.0, VREF или GND. Модуль АЦП0 включен только тогда, когда бит AD0EN регистра управления АЦП0 (ADC0CN) установлен в 1. Сброс этого бита в 0 переводит АЦП0 в режим отключения с пониженным энергопотреблением.

Аналоговый мультиплексор. AMUX0 осуществляет выбор положительного и отрицательного входов АЦП. В качестве положительного входа можно выбрать:

– P1.0 – P3.0;

– выходной сигнал встроенного датчика температуры;

– положительное напряжение питания (VDD).

В качестве отрицательного входа можно выбрать:

– P1.0 – P3.0;

– VREF;

– общий вывод питания GND.

Если в качестве отрицательного входа выбран GND, то АЦП0 функционирует в однофазном режиме; в остальных случаях АЦП0 функционирует в дифференциальном режиме. Входные каналы АЦП0 выбираются в регистрах AMX0P и AMX0N.

Аналоговый мультиплексор. AMUX0 осуществляет выбор положительного и отрицательного входов АЦП. В качестве положительного входа можно выбрать:

– P1.0 – P3.0;

– выходной сигнал встроенного датчика температуры;

– положительное напряжение питания (VDD).

В качестве отрицательного входа можно выбрать:

– P1.0 – P3.0;

– VREF;

– общий вывод питания GND.

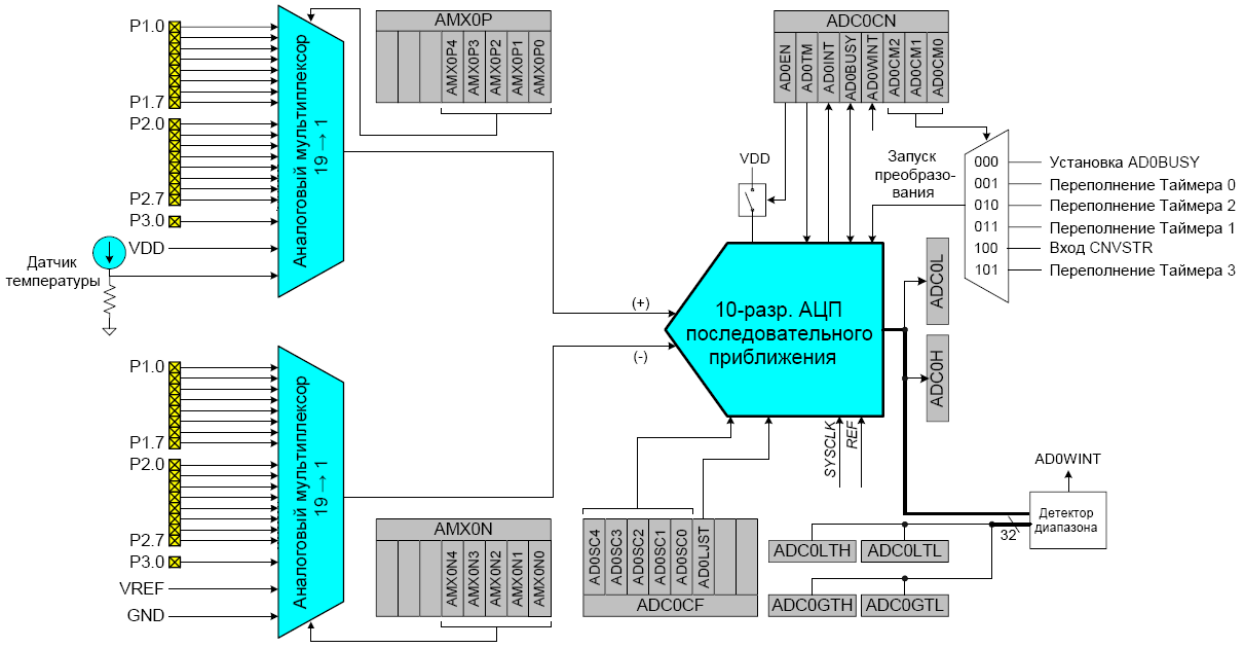


Рисунок 6.9 – Функциональная схема АЦП0

Если в качестве отрицательного входа выбран GND, то АЦП0 функционирует в однофазном режиме; в остальных случаях АЦП0 функционирует в дифференциальном режиме. Входные каналы АЦП0 выбираются в регистрах AMX0P и AMX0N.

Формат получаемого результата преобразования различен для однофазного и дифференциального режимов. По окончании каждого преобразования в регистры ADC0H и ADC0L записываются, соответственно, старший и младший байты результата преобразования. Данные могут быть выровнены вправо или влево в зависимости от значения бита AD0LJST (ADC0CN.0). В однофазном режиме результаты преобразований представлены в виде 10-разрядных целых чисел без знака. Диапазон измерения входных напряжений: 0 …VREF*1023/1024. В дифференциальном режиме результаты преобразований представлены в виде 10-разрядных чисел в дополнительном коде со знаком. Диапазон измерения входных напряжений: - VREF … VREF*511/512.

Важное замечание относительно настройки входов АЦП0: выводы порта, выбранные в качестве входов АЦП0, должны быть настроены как аналоговые входы и должны пропускаться матрицей при назначении выводов. Чтобы настроить вывод порта как аналоговый вход, следует сбросить в 0 соответствующий бит в регистре PnMDIN (для n=0,1,2,3). Чтобы заставить матрицу про-

пускать вывод порта при назначении выводов, следует установить в 1 соответствующий бит в регистре PnSKIP (для n=0,1,2). Более подробная информация о настройке порта ввода/вывода приведена в лабораторной работе №1.

Максимальная скорость преобразования АЦП0 – 200 тыс. преобразований в секунду. Частота дискретизации АЦП0 определяется частотой системного тактового сигнала, деленной на значение, задаваемое битами AD0SC регистра ADC0CF, т.е. SYSCLK/(AD0SC + 1) для $0 \leq$ AD0SC ≤ 31.

Запуск преобразования может быть осуществлен одним из шести способов в зависимости от состояния битов режима запуска преобразования АЦП0 (AD0CM2-0) в регистре ADC0CN. Преобразование может быть инициировано:

1) установкой в 1 бита AD0BUSY в регистре ADC0CN;

2) переполнением Таймера 0 (т.е. непрерывное по времени преобразование);

3) переполнением Таймера 2;

4) переполнением Таймера 1;

5) нарастающим фронтом внешнего входного сигнала CNVSTR (вывод P0.6);

6) переполнением Таймера 3.

Установка в 1 бита AD0BUSY позволяет осуществлять программное управление АЦП0, т.е. выполнять преобразования «по требованию». Бит AD0BUSY устанавливается в 1 во время преобразования и сбрасывается в 0 после окончания преобразования. При сбросе бита AD0BUSY инициируется прерывание (если оно разрешено) и устанавливается флаг прерывания от АЦП0 (AD0INT). Примечание: при определении окончания преобразования методом опроса следует использовать флаг прерывания от АЦП0 (AD0INT). Преобразованные данные доступны в регистрах старшего и младшего слова данных АЦП0, ADC0H и ADC0L соответственно, когда AD0INT=1. Следует иметь ввиду, что если запуск преобразования осуществляется переполнением Таймера 2 или Таймера 3, то используются переполнения младшего байта, когда Таймер 2/3 работает в 8-разрядном режиме, и переполнения старшего байта, когда Таймер 2/3 работает в 16-разрядном режиме. Важное замечание относительно использования CNVSTR: входной вывод CNVSTR функционирует также, как вывод порта P0.6. Если вход CNVSTR используется для запуска преобразования АЦП0, то вывод порта P0.6 должен пропускаться матрицей при назначении выводов. Чтобы заставить матрицу пропускать P0.6, следует установить в 1 бит 6 в регистре P0SKIP.

Время установления. Если конфигурация входов АЦП0 изменяется (т.е. изменяются настройки AMUX0), то после этого для обеспечения точности преобразования необходимо выдержать паузу длительностью не менее минимального времени установления сигнала. Время установления определяется сопротивлением AMUX0, емкостью накопительного конденсатора УВХ, сопротивлением внешнего источника сигнала и требуемой точностью преобразования. Следует отметить, что в энергосберегающем режиме выборки-хранения после запуска каждого преобразования выборка длится три периода сигнала дискретизации АЦП. Для большинства приложений эти три периода сигнала дискре-

тизации будут соответствовать требованиям, предъявляемым ко времени установления.

На рисунке 6.9 показаны эквивалентные схемы входов АЦП0 как для дифференциального, так и для однофазного режимов работы. Следует отметить, что эквивалентная постоянная времени для обоих схем одинакова. Требуемое время установления для заданной точности установления (settling accuracy – SA) можно приблизительно определить из уравнения t = ln(2n/SA) x RTOTALCSAMPLE. Если измеряется выходное напряжение датчика температуры или напряжение VDD относительно GND, то RTOTAL = RMUX, где SA – точность установления, задаваемая в долях МЗР (например 0.25 для установления в пределах ¼ МЗР); t – требуемое время установления в секундах RTOTAL – сумма сопротивления AMUX0 и сопротивления внешнего источника сигнала; n – разрешение АЦП в битах (10).

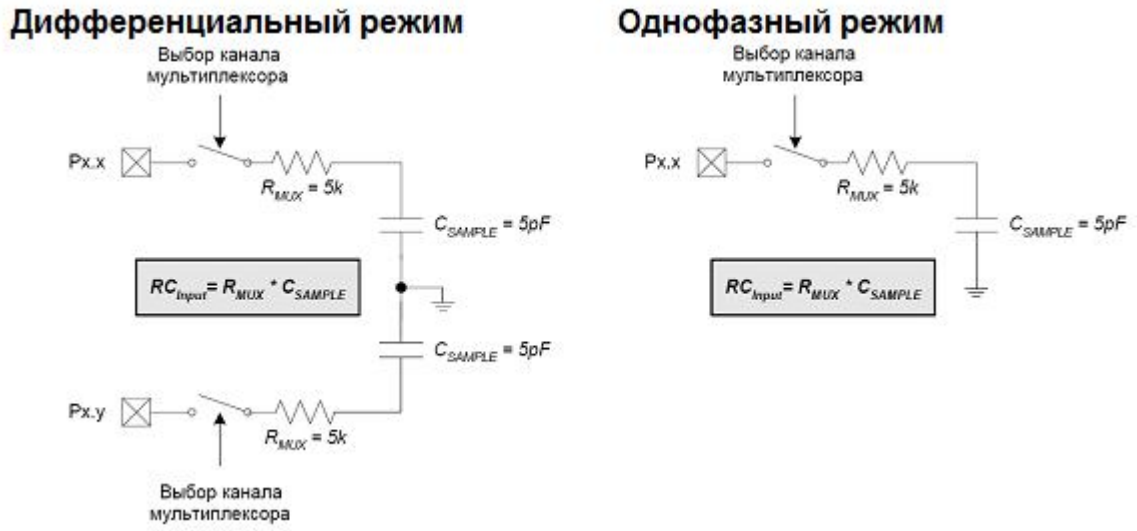


Рисунок 6.10 – Эквивалентные схемы входов АЦП0

Необходимые регистры АЦП перечислены в таблице 6.3 и подробно описаны в приложении.

Таблица 6.3 – Регистры АЦП

AMX0P	Регистр выбора положительного канала AMUX0
AMX0N	Регистр выбора отрицательного канала AMUX0
ADC0CF	Регистр конфигурации АЦП0
ADC0H	Регистр старшего байта слова данных АЦП0
ADC0L	Регистр младшего байта слова данных АЦП0
ADC0CN	Регистр управления АЦП0

Организация памяти. Организация памяти МК с ядром CIP-51 соответствует стандарту 8051. Имеется две отдельных области памяти, память программ и память данных, которые разделяют одно и то же адресное пространство, но доступ к ним осуществляется командами различного типа. Организация памяти CIP-51 показана на рисунке 6.11.

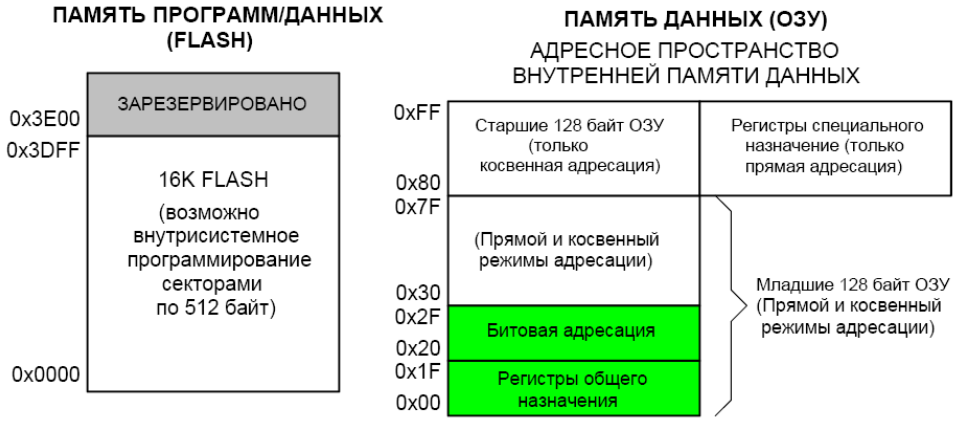


Рисунок 6.11 – Карта распределения памяти

Память программ. CIP-51 имеет адресное пространство памяти программ 64 Кбайт. В МК C8051F320/1 физически реализовано 16 Кбайт этой памяти программ, которая является внутрисистемной перепрограммируемой Flash-памятью, занимающей непрерывный блок адресов от 0x0000 до 0x3FFF. Адреса, превышающие 0x3FFF, зарезервированы. По умолчанию память программ настраивается только для чтения. Однако CIP-51 может записывать данные в память программ (с использованием команды MOVX), для чего необходимо установить в 1 бит разрешения записи памяти программ (PSCTL.0). Эта возможность позволяет CIP-51 обновлять программный код и использовать память программ для долговременного хранения данных.

Память данных. Физически реализовано 256 байт внутреннего ОЗУ, отображенного в пространстве памяти данных с адресами от 0x00 до 0xFF. Младшие 128 байт памяти данных используются для регистров общего назначения (РОН) и сверхоперативного ЗУ (СОЗУ). Для доступа к младшим 128 байтам памяти данных можно использовать либо прямую, либо косвенную адресацию. Ячейки с адресами от 0x00 до 0x1F разбиты на четыре банка РОН, каждый банк состоит из восьми однобайтовых регистров. Следующие 16 байт (0x20 – 0x2F) могут адресоваться побайтно или побитно как 128 бит, доступные в режиме прямой битовой адресации. Старшие 128 байт памяти данных доступны только в режиме косвенной адресации. Эта область памяти занимает то же самое адресное пространство, что и регистры специального назначения (Special Function Registers – SFR), но физически отделена от них. При обращении к ячейкам памяти с адресами 0x7F – 0xFF использующийся в команде режим адресации определяет, к чему осуществляется доступ: к страшим 128 байтам памяти данных или к SFR. Команды, которые используют режим прямой адресации, будут обращаться к SFR. Команды, использующие режим косвенной адресации, будут обращаться к старшим 128 байтам памяти данных.

Регистры общего назначения. Младшие 32 байта памяти данных (0x00 – 0x1F) разбиты на четыре банка регистров общего назначения. Каждый банк состоит из восьми однобайтовых регистров, обозначаемых R0-R7. В конкретный момент времени может быть активен лишь один банк, определяемый битами

RS0 (PSW.3) и RS1 (PSW.4) в слове состояния программы (program status word) PSW. Это позволяет осуществлять быстрое переключение контекста при вызове подпрограмм и процедур обработки прерываний. Режимы косвенной адресации используют регистры R0 и R1 в качестве индексных регистров.

Ячейки памяти с битовой адресацией. Кроме прямого (побайтного) доступа к памяти данных 16 ячеек этой памяти с адресами 0x20 – 0x2F доступны также, как 128 индивидуально адресуемых бит. Каждый бит имеет битовый адрес от 0x00 до 0x7F. Бит 0 байта 0x20 имеет битовый адрес 0x00, а бит 7 байта 0x20 имеет битовый адрес 0x07. Бит 7 байта 0x2F имеет битовый адрес 0x7F. Битовый доступ можно отличить от байтового доступа по типу используемой команды (операнды исходных данных и результата в первом случае являются битами, во втором – байтами).

Ассемблер MCS-51™ допускает альтернативную запись для режима битовой адресации в форме XX.B, где XX – адрес байта, а B – позиция бита внутри этого байта. Например, команда MOV C, 22h.3 присваивает значение бита 0x13 (бит 3 в ячейке с адресом 0x22) флагу переноса.

Стек. Программный стек может быть размещен в любом месте 256-байтной памяти данных. Область стека определяется с использованием указателя стека (Stack Pointer – SP, 0x81). SP будет указывать на последнюю использованную ячейку. Следующее значение, загружаемое в стек, размещается по адресу SP+1, и затем SP инкрементируется. При сбросе SP инициализируется значением 0x07. Поэтому первое значение, загружаемое в стек, размещается по адресу 0x08, которое также является первым регистром (R0) регистрового банка 1. Таким образом, если требуется использовать более одного банка регистров, SP следует инициализировать адресом ячейки ОЗУ, не используемой для хранения данных. Стек может иметь глубину до 256 байт.

Регистры специального назначения. Ячейки памяти данных с адресами от 0x80 до 0xFF, доступные в режиме прямой адресации, образуют регистры специального назначения (special function registers – SFR). SFR позволяют управлять ресурсами ядра CIP-51 и периферийными модулями, а также осуществлять обмен данными с ними.

Регистры SFR доступны в любое время, когда для доступа к ячейкам памяти с адресами от 0x80 до 0xFF используется режим прямой адресации. SFR с адресами, оканчивающимися на 0x0 или 0x8 (т.е. P0, TCON, SCON, IE, и т.д.), адресуются как побайтно, так и побитно. Все другие SFR адресуются только побайтно. Незанятые адреса в области SFR зарезервированы для дальнейшего использования. Обращение к ячейкам из этой области даст неопределенный результат и должно быть исключено.

6.7 Универсальный асинхронный приемо-передатчик

UART0 представляет собой асинхронный полнодуплексный последовательный порт, способный работать в режимах 1 и 3 стандартного (для архитектуры 8051) UART. Поддержка усовершенствованного режима генерации скорости

передачи данных позволяет использовать для генерации стандартных скоростей обмена различные источники тактирования. Буферизация принимаемых данных позволяет UART0 начать прием второго входящего байта данных до того, как программа закончит чтение предыдущего байта данных.

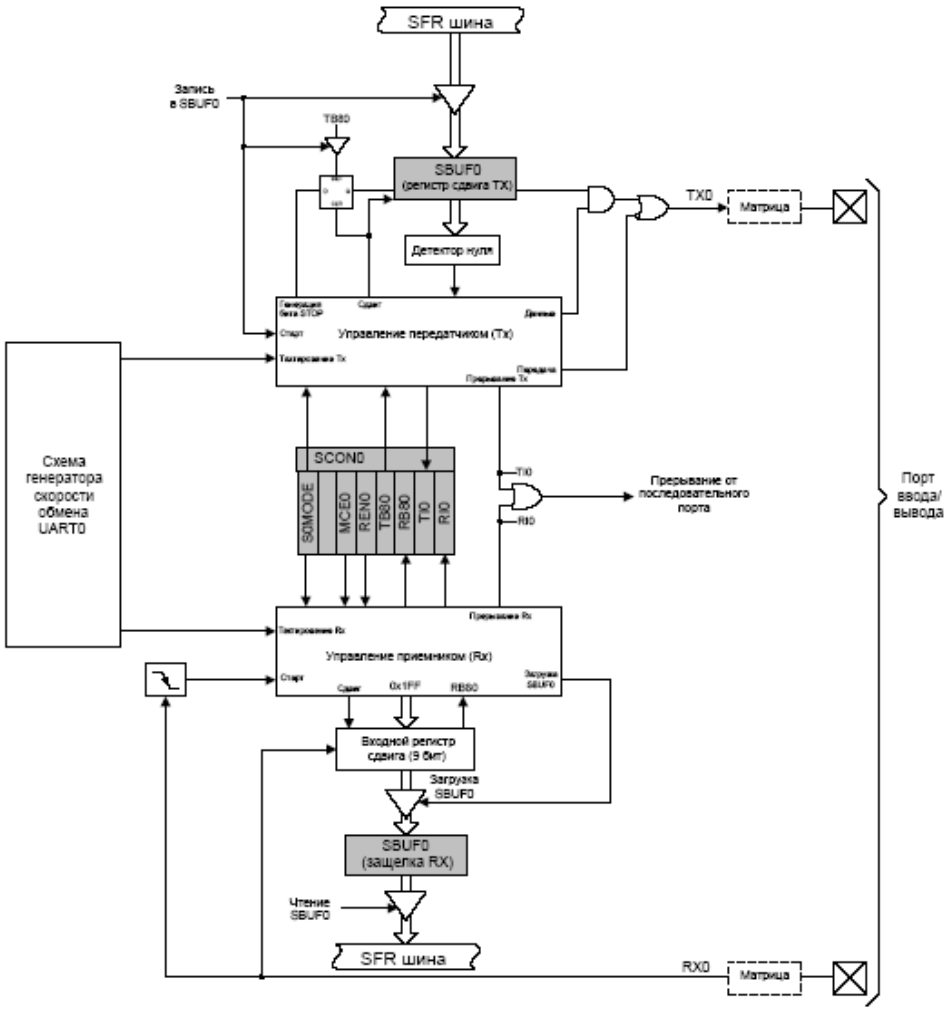


Рисунок 6.12 – Структурная схема UART0

С работой UART0 связаны следующие регистры специального назначения: регистр управления UART0 (SCON0) и буфер данных UART0 (SBUF0). Одна и та же ячейка памяти, адресуемая как SBUF0, обеспечивает доступ и к регистру передатчика, и к регистру приемника. Операции записи в SBUF0 всегда обращаются к регистру передатчика**.** Операции чтения из SBUF0 всегда обращаются к буферизованному регистру приемника; невозможно прочитать данные из регистра передатчика.

Если прерывания от модуля UART0 разрешены, то запрос прерывания генерируется каждый раз при завершении передачи байта данных (установка в 1 флага TI0 в регистре SCON0) или при получении байта данных (установка в 1 флага RI0 в регистре SCON0). Флаги прерываний от UART0 не сбрасываются аппаратно при переходе к процедуре обслуживания прерывания. Они должны сбрасываться программно. Это позволяет программе определить причину, вызвавшую прерывание от UART0 (завершение передачи или завершение приема).

6.7.1 Усовершенствованный режим генерации скорости передачи данных

Скорость передачи данных UART0 генерируется Таймером 1, работающим в 8-разрядном режиме с автоперезагрузкой. Частота передатчика (TX) определяется переполнением регистра TL1; частота приемника определяется переполнением регистра-копии регистра TL1 (обозначенного как «RX-Таймер» на рисунке 6.13), который недоступен из программы пользователя. Скорость передачи данных передатчика и приемника равна деленной на два частоте переполнения регистров TL1 и RX-Таймер соответственно. RX-Таймер работает тогда, когда включен Таймер 1, и использует то же самое значение перезагрузки (TH1). Однако перезагрузка регистра RX-Таймер происходит в тот момент, когда на выводе RX обнаруживается событие START. Это позволяет начать прием данных в любой момент при обнаружении события START независимо от состояния Таймера TX.

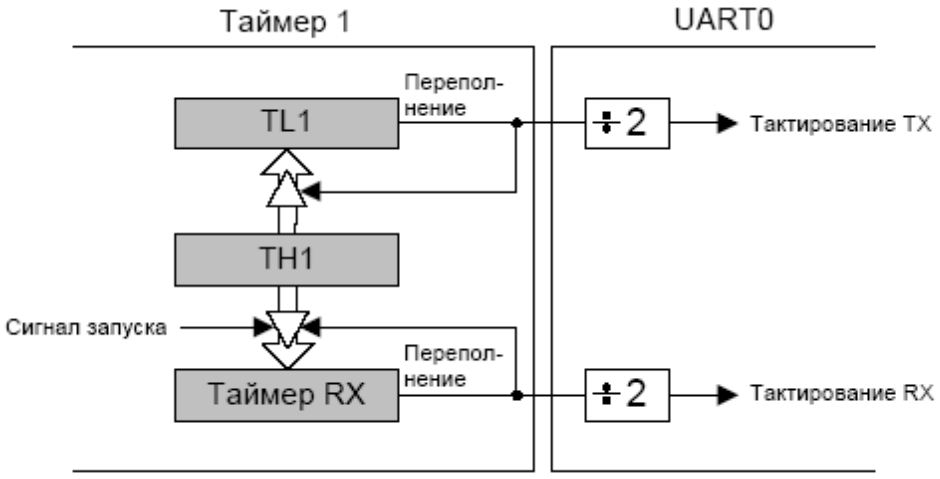


Рисунок 6.13 – Логика генератора скорости передачи данных UART0

Таймер 1 следует настроить для работы в режиме 2, т.е. как 8-разрядный таймер с автоперезагрузкой. Значение перезагрузки Таймера 1 следует установить таким образом, чтобы частота переполнений таймера была в два раза больше необходимой скорости передачи данных. Частота тактового сигнала Таймера 1 может быть одной из следующих:

1) SYSCLK;
2) SYSCLK/4;
3) SYSCLK/12;
4) SYSCLK/48;
5) частота внешнего генератора / 8;
6) частота внешнего сигнала на входе T1.

Скорость передачи данных UART0 определяется из равенства (6.1):

$$V_{UART0} = \frac{F_{T1}}{2}, \qquad (6.1)$$

где V_{UART0} – скорость передачи данных UART0; F_{T1} – частота переполнения Таймера 1.

Расчет частоты переполнения Таймера 1 F_{T1} производим по формуле

$$F_{T1} = \frac{CLK_{T1}}{256 - H_{T1}}, \qquad (6.2)$$

где CLK_{T1} – частота тактирования Таймера 1; H_{T1} – старший байт Таймера 1 (8-разрядное значение перезагрузки).

Следует отметить, что внутренний генератор может генерировать системный тактовый сигнал, в то время как выходной сигнал внешнего генератора подается на Таймер 1.

6.7.2 Режимы работы UART0

UART0 обеспечивает стандартный асинхронный полнодуплексный обмен данными. Режим работы UART0 (8-разрядный или 9-разрядный) выбирается при помощи бита S0MODE (SCON0.7). Типичные варианты использования UART приведены на рисунке 6.14.

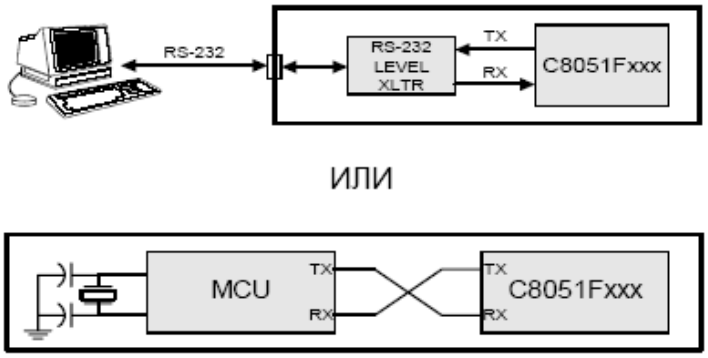


Рисунок 6.14 – Примеры использования UART

8-разрядный UART. В режиме 8-разрядного UART для передачи одного байта данных используются 10 бит: один стартовый бит, восемь бит данных (МЗР вперед) и один стоповый бит. Данные передаются МЗР вперед через внешний вывод TX0 и принимаются через внешний вывод RX0. При приеме в регистре SBUF0 сохраняются восемь бит данных, а бит RB80 (SCON0.2) принимает значение стопового бита.

Передача данных начинается, когда происходит запись байта данных в регистр SBUF0. Флаг прерывания от передатчика TI0 (SCON0.1) устанавливается в 1 в конце передачи (в начале передачи стопового бита). Прием данных может быть начат в любое время после установки в 1 флага включения приемника REN0 (SCON0.4). После приема стопового бита байт данных будет загружен в регистр приемника SBUF0, если соблюдаются следующие условия: RI0 должен быть равен лог. «0», и если MCE0 = 1, то стоповый бит должен быть равен лог. «1». В случае переполнения данных при приеме первые принятые 8 бит «за-

щелкиваются» в регистре приемника SBUF0, а следующие данные, вызвавшие переполнение, теряются.

Если эти условия соблюдаются, то восемь бит данных сохраняются в регистре SBUF0, стоповый бит сохраняется в бите RB80 и устанавливается в 1 флаг RI0. Если эти условия не соблюдаются, то SBUF0 и RB80 не будут загружаться и флаг RI0 не устанавливается. При установке флагов TI0 или RI0 будет сгенерировано прерывание, если оно разрешено.

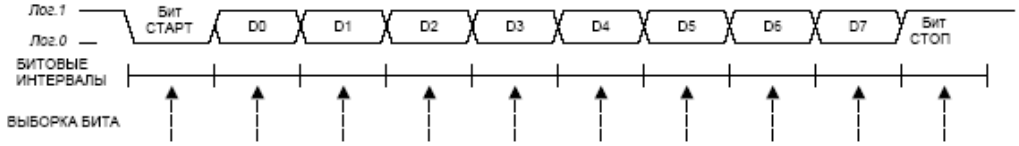


Рисунок 6.14 – Временные диаграммы в режиме 8-разрядного UART

6.7.3 9-разрядный UART.

В режиме 9-разрядного UART для передачи одного байта данных используются 11 бит: один стартовый бит, восемь бит данных (МЗР вперед), программируемый девятый бит данных и один стоповый бит. При передаче значение девятого бита данных определяется значением бита TB80 (SCON0.3), который устанавливается/сбрасывается программой пользователя. Значение девятого бита может либо соответствовать значению флага четности «P» регистра PSW (применяется для обнаружения ошибок), либо использоваться для организации связи с несколькими МК. При приеме значение девятого бита сохраняется в бите RB80 (SCON0.2), а стоповый бит игнорируется.

Передача данных начинается, когда происходит запись байта данных в регистр SBUF0. Флаг прерывания от передатчика TI0 (SCON0.1) устанавливается в 1 в конце передачи (в начале передачи стопового бита). Прием данных может быть начат в любое время после установки в 1 флага включения приемника REN0 (SCON0.4). После приема стопового бита байт данных будет загружен в регистр приемника SBUF0, если соблюдаются следующие условия: RI0 должен быть равен лог. «0», и если MCE0 = 1, то стоповый бит должен быть равен лог. «1» (когда MCE0 = 0, состояние девятого бита данных не имеет значения). Если эти условия соблюдаются, то восемь бит данных сохраняются в регистре SBUF0, девятый бит данных сохраняется в бите RB80 и устанавливается в 1 флаг RI0. Если эти условия не соблюдаются, то SBUF0 и RB80 не будут загружаться и флаг RI0 не будет устанавливаться. При установке флагов TI0 или RI0 будет сгенерировано прерывание от модуля UART0, если оно разрешено.

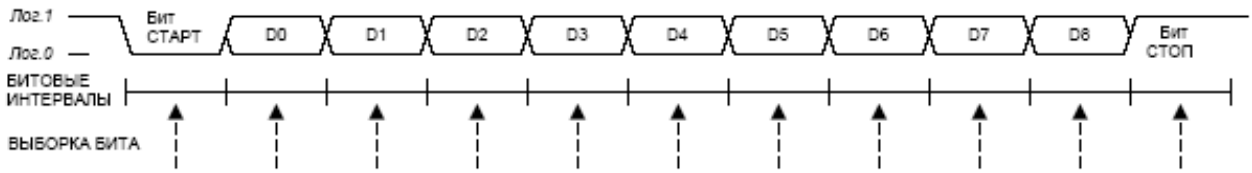


Рисунок 6.16 – Временные диаграммы в режиме 9-разрядного UART

Таблица 6.4 – Параметры настройки таймера для стандартных скоростей передачи данных при тактировании от внутреннего генератора(11,0592МГц)

	Частота: 11,0592 МГц						
	Требуемая скорость передачи данных (бит/сек)	**Погрешность установки скорости передачи данных**	**Коэффициент деления генератора**	**Частота сигнала тактирования**	**SCA1-SCA0 (выбор коэффициента предварительного деления)***	**T1M***	**Значение перезагрузки Таймера 1**
SYSCLK от внешнего генератора	230400	0,00%	48	SYSCLK	XX	1	0xE8
	115200	0,00%	96	SYSCLK	XX	1	0xD0
	57600	0,00%	192	SYSCLK	XX	1	0xA0
	28800	0,00%	384	SYSCLK	XX	1	0x40
	14400	0,00%	768	SYSCLK/12	00	0	0xE0
	9600	0,00%	1152	SYSCLK/12	00	0	0xD0
	2400	0,00%	4608	SYSCLK/12	00	0	0x40
	1200	0,00%	9216	SYSCLK/48	10	0	0xA0
SYSCLK от внутреннего генератора	230400	0,00%	48	EXTCLK/8	11	0	0xFD
	115200	0,00%	96	EXTCLK/8	11	0	0xFA
	57600	0,00%	192	EXTCLK/8	11	0	0xF4
	28800	0,00%	384	EXTCLK/8	11	0	0xE8
	14400	0,00%	768	EXTCLK/8	11	0	0xD0
	9600	0,00%	1152	EXTCLK/8	11	0	0xB8

X – Не имеет значения.

Пример листинга программы с использованием АЦП и модуля УАПП микроконтроллера:

```
#include <c8051f300.h>        // Описание регистров спец. функций
#include <stdio.h>            // Библиотека ввода/вывода
//-----------------------------------------------------------
// 16-bit SFR Definitions for 'F30x
//-----------------------------------------------------------
sfr16 TMR2RL   = 0xca;        // Timer2 перезагружаемое значение
sfr16 TMR2     = 0xcc;        // Timer2 счетчик
//-----------------------------------------------------------
// Определение констант
//-----------------------------------------------------------
```

```
#define SYSCLK    24500000    // SYSCLK системная частота в Герцах
#define BAUDRATE  115200      // скорость передачи в UART
sbit LED = P0^2;              // LED='1' - светодиод включен
//-----------------------------------------------------------------
// Прототипы функций
//-----------------------------------------------------------------
void SYSCLK_Init (void);
void PORT_Init (void);
void Timer2_Init(void);
void ADC0_Init(void);
void UART0_Init (void);
//-----------------------------------------------------------------
// Главная функция программы
//-----------------------------------------------------------------
void main (void)
{
   PCA0MD &= ~0x40;           // Выключение WDT
   SYSCLK_Init ();            // Инициализация системной частоты
   PORT_Init ();              //Инициализация ввода/вывода
   Timer2_Init();             // Инициализация Timer2
   UART0_Init();              // Инициализация UART0
   ADC0_Init();               // Инициализация ADC0
   EA = 1;                    // Включить глобальные прерывание
   while (1) {                // зацикливание программы
   }
}
//-----------------------------------------------------------------
// Initialization Subroutines
//-----------------------------------------------------------------
void SYSCLK_Init (void)
{
   OSCICN |= 0x07;            // внутренний генератор на 24.5MHz
   RSTSRC = 0x04;             // включить детектор генерации
}
void PORT_Init (void)
{
```

```
   XBR0    = 0x0c;       // пропустить выводы светодиода и кнопки
   XBR1    = 0x03;       // включить пины UART0: TX и RX
   XBR2    = 0x40;       // разрешить подтяжки выводов
   P0MDOUT |= 0x14;      // включить подтяжки на выводы TX0 and LED
   P0MDIN &= ~0x01;      // установить P0.1 как аналоговый вход
}
void Timer2_Init (void)
{
   TMR2CN  = 0x00;       // Остановить Timer2; очистить TF2;
// использовать для тактирования SYSCLK,
// режим 16-битн. счетчика с автоперезагрузкой
   CKCON |= 0x20;   // Timer2 использует для тактирования SYSCLK
   TMR2RL  = 65536 - (SYSCLK / 10000);  // перезагружаемое значе-
ние для 10мкс
   TMR2    = 0xffff;          // для мгновенной перезагрузки
   ET2 = 0;                   // выключить прерывания от Timer2
   TR2 = 1;                   // запустить Timer2
}
void ADC0_Init (void)
{
   ADC0CN = 0x02;   // ADC0 выключен, запуск преобразов. от TMR2
   REF0CN = 0x0A;        // Установить VDD как опорное напряжение
   AMX0SL = 0x81;        // ADC0 положительный вход = P0.1
                         // ADC0 отрицательный вход = GND
   ADC0CF = ((SYSCLK/3000000)-1)<<3; // частота тактирования АЦП =
=3MHz
   ADC0CF |= 0x01;       // усиление внутреннего усилителя = 1
   EIE1 |= 0x04;         // включить прерывания от ADC0
   AD0EN = 1;            // включить ADC0
}
//-------------------------------------------------------------
// UART0_Init
//-------------------------------------------------------------
void UART0_Init (void)
{
   SCON0 = 0x10;              // SCON0: 8-битный режим
                              // уровень бита STOP игнорируется
```

```
                                // очистить биты RI0 и TI0
   if (SYSCLK/BAUDRATE/2/256 < 1)
      {TH1 = -(SYSCLK/BAUDRATE/2);
      CKCON |= 0x10; }                  // T1M = 1; SCA1:0 = xx
    else if (SYSCLK/BAUDRATE/2/256 < 4)
      {TH1 = -(SYSCLK/BAUDRATE/2/4);
      CKCON |=  0x01;                   // T1M = 0; SCA1:0 = 01
      CKCON &= ~0x12; }
    else if (SYSCLK/BAUDRATE/2/256 < 12)
      {TH1 = -(SYSCLK/BAUDRATE/2/12);
      CKCON &= ~0x13; }                 // T1M = 0; SCA1:0 = 00
    else {
      TH1 = -(SYSCLK/BAUDRATE/2/48);
      CKCON |=  0x02;                   // T1M = 0; SCA1:0 = 10
      CKCON &= ~0x11;
   }

   TL1 = 0xff;           // установка для мгновенной перезагрузки
   TMOD |= 0x20;         // Timer 1 режим 8-битный с автопереза-
грузкой
   TMOD &= ~0xD0;
   TR1 = 1;              // запустить Timer1
   TI0 = 1;              // индикация TX0
}
//---------------------------------------------------------------
// Функция обработки прерывания
//---------------------------------------------------------------
void ADC0_ISR (void) interrupt 8
{
   static unsigned long accumulator = 0;     // Аккумулятор для
усреднения
   static unsigned int measurements = 2048; // Счетчик измерений
   unsigned long result=0;
   unsigned long mV;     // Амплитуда измерений в mV
   while(!AD0INT);       // Ожидание конца преобразования
   AD0INT = 0;           // Очистить флаг окончания преобразования
   accumulator += ADC0;
```

```
    measurements--;

    if(measurements == 0)
    {
       measurements = 2048;
       result = accumulator / 2048;
       accumulator=0;
       // The 8-bit ADC value is averaged across 2048 measurements.
       // The measured voltage applied to P1.4 is then:
       //                            Vref (mV)
       //   measurement (mV) =   --------------- * result (bits)
       //                         (2^8)-1 (bits)
       mV =  result * 3270 / 255;
       printf("P0.1 voltage: %ld mV\n",mV);
    }
}
```

7 Заключение курсового проекта

Заключение, наряду с введением, наиболее важная часть курсового проекта. В нем указывается итог всего проделанного исследования, выводы и предложения, перспективы развития того или иного вопроса. Удачно написанное заключение логично завершает курсовой проект, делает его цельным и законченным.

1. Заключение тесно связано с введением. Если во введении указывается цель и задачи курсового проекта, то в заключении указывается, удалось ли достичь указанной цели с помощью намеченных методов исследования.

2. Если в конце каждой главы или раздела вы делали краткий вывод, то составить заключение не составит особого труда. Просто соберите выводы воедино, а также добавьте перспективы развития исследуемой проблемы, практическое ее применение. Заключение должно быть построено в соответствии с логикой курсового проекта.

3. Заключение соединяет в себе результаты, полученные в ходе теоретического и практического проектирования и изучения поставленной темы. При этом необходимо озвучить новизну, которую вы достигли в ходе проектирования.

4. Объем заключения в курсовой работе не должен превышать 1-2 страниц. Выводы должны быть краткими и лаконичными, без излишних подробностей.

ПРИЛОЖЕНИЕ А

Описания регистров прерываний

Регистры SFR, используемые для разрешения/запрещения источников прерываний и установки их приоритетов, описаны ниже.

IE: Регистр разрешения прерываний.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значение при сбросе: 00000000 SFR Адрес: 0xA8
EA	ESPI0	ET2	ES0	ET1	EX1	ET0	EX0	
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	

Бит 7: **EA**: Бит разрешения всех прерываний.

Это бит глобально разрешает/запрещает все прерывания. Будучи сброшенным в 0, он перекрывает индивидуальные маски прерываний

0: Все источники прерываний запрещены.

1: Каждое прерывание разрешено/запрещено в соответствии с его индивидуальной маской.

Бит 6: **ESPI0**: Бит разрешения прерываний от модуля SPI0.

Этот бит устанавливает маскирование прерывания от модуля SPI0.

0: Все прерывания от модуля SPI0 запрещены.

1: Разрешены запросы прерываний, генерируемые при установке флага SPI0.

Бит 5: **ET2**: Бит разрешения прерывания от Таймера 2.

Этот бит устанавливает маскирование прерывания от Таймера 2.

0: Все прерывания от Таймера 2 запрещены.

1: Разрешены запросы прерываний, генерируемые при установке флагов TF2L или TF2H.

Бит 4: **ES0**: Бит разрешения прерываний от последовательного порта УАПП0.

0: Прерывания от УАПП0 запрещены.

1: Прерывания от УАПП0 разрешены.

Бит 3: **ET1**: Бит разрешения прерывания от Таймера 1.

Этот бит устанавливает маскирование прерывания от Таймера 1.

0: Все прерывания от Таймера 1 запрещены.

1: Разрешены запросы прерываний, генерируемые при установке флага TF1.

Бит 2: **EX1**: Бит разрешения внешнего прерывания 1.

Этот бит устанавливает маскирование внешнего прерывания 1.

0: Внешнее прерывание 1 запрещено.

1: Разрешены запросы прерываний, генерируемые сигналом на входе /INT1.

Бит 1: **ET0**: Бит разрешения прерывания от Таймера 0.

Этот бит устанавливает маскирование прерывания от Таймера 0.

0: Все прерывания от Таймера 0 запрещены.

1: Разрешены запросы прерываний, генерируемые при установке флага TF0.

Бит 0: **EX0**: Бит разрешения внешнего прерывания 0.

Этот бит устанавливает маскирование внешнего прерывания 0.

0: Внешнее прерывание 0 запрещено.

1: Разрешены запросы прерываний, генерируемые сигналом на входе /INT0.

IP: Регистр приоритетов прерываний.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значение при сбросе: 10000000 SFR Адрес: 0xB8
-	PSPI0	PT2	PS0	PT1	PX1	PT0	PX0	
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	

Бит 7: Не используется. Читается как 1b. Запись не оказывает никакого влияния.

Бит 6: **PSPI0**: Управление приоритетом прерывания от модуля SPI0.

Этот бит устанавливает приоритет прерывания от модуля SPI0.

0: Прерыванию от модуля SPI0 назначается низкий уровень приоритета.

1: Прерыванию от модуля SPI0 назначается высокий уровень приоритета.

Бит 5: **PT2**: Управление приоритетом прерывания от Таймера 2.

Этот бит устанавливает приоритет прерываний от Таймера 2.

0: Прерыванию от Таймера 2 назначается низкий уровень приоритета.

1: Прерываниям от Таймера 2 назначается высокий уровень приоритета.

Бит 4: **PS0**: Управление приоритетом прерывания от последовательного порта УАПП0.

Этот бит устанавливает приоритет прерываний от последовательного порта УАПП0.

0: Прерываниям от УАПП0 назначается низкий уровень приоритета.

1: Прерываниям от УАПП0 назначается высокий уровень приоритета.

Бит 3: **PT1**: Управление приоритетом прерывания от Таймера 1.

Этот бит устанавливает приоритет прерываний от Таймера 1.

0: Прерываниям от Таймера 1 назначается низкий уровень приоритета.

1: Прерываниям от Таймера 1 назначается высокий уровень приоритета.

Бит 2: **PX1**: Управление приоритетом внешнего прерывания 1.

Этот бит устанавливает приоритет внешнего прерывания 1.

0: Внешнему прерыванию 1 назначается низкий уровень приоритета.

1: Внешнему прерыванию 1 назначается высокий уровень приоритета.

Бит 1: **PT0**: Управление приоритетом прерывания от Таймера 0.

Этот бит устанавливает приоритет прерываний от Таймера 0.

0: Прерываниям от Таймера 0 назначается низкий уровень приоритета.

1: Прерываниям от Таймера 0 назначается высокий уровень приоритета.

Бит 0: **PX0**: Управление приоритетом внешнего прерывания 0.

Этот бит устанавливает приоритет внешнего прерывания 0.

0: Внешнему прерыванию 0 назначается низкий уровень приоритета.
1: Внешнему прерыванию 0 назначается высокий уровень приоритета.

Описания регистров портов ввода-вывода
XBR0: Регистр 0 матрицы портов ввода/вывода.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значение при сбросе: 00000000 SFR Адрес: 0xE1
CP1AE	CP1E	CP0AE	CP0E	SYSCKE	SMB0E	SPI0E	URT0E	
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	

Бит 7: **CP1AE**: Бит подключения асинхронного выхода Компаратора 1 (CP1)
0: Асинхронный выход CP1 не соединен с выводом порта.
1: Асинхронный выход CP1 соединен с выводом порта.
Бит 6: **CP1E**: Бит подключения выхода Компаратора 1 (CP1)
0: CP1 не соединен с выводом порта.
1: CP1 соединен с выводом порта.
Бит 5: **CP0AE**: Бит подключения асинхронного выхода Компаратора 0 (CP0)
0: Асинхронный выход CP0 не соединен с выводом порта.
1: Асинхронный выход CP0 соединен с выводом порта.
Бит 4: **CP0E**: Бит подключения выхода Компаратора 0 (CP0)
0: CP0 не соединен с выводом порта.
1: CP0соединен с выводом порта.
Бит 3: **SYSCKE**: Бит подключения выхода /SYSCLK
0: Выход /SYSCLK не соединен с выводом порта.
1: Выход /SYSCLK соединен с выводом порта.
Бит 2: **SMB0E**: Бит подключения входов/выходов модуля SMBus0
0: Входы/выходы модуля SMBus0 не соединены с выводами порта.
1: Входы/выходы модуля SMBus0 соединены с выводами порта.
Бит 1: **SPI0E**: Бит подключения входов/выходов модуля SPI0
0: Входы/выходы модуля SPI0 не соединены с выводами порта.
1: Входы/выходы модуля SPI0 соединены с выводами порта.
Бит 0: **URT0E**: Бит подключения входов/выходов УАПП0
0: Входы/выходы УАПП0 не соединены с выводами порта.
1: TX0 и RX0 соединены с выводами P0.4 и P0.5 соответственно.

XBR1: Регистр 1 матрицы портов ввода/вывода.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значение при сбросе: 00000000 SFR Адрес: 0xE2
WEAKPUD	XBARE	T1E	T0E	ECIE	PCA0ME			
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	

Бит 7: **WEAKPUD**: Бит отключения слаботоковых подтяжек портов ввода/вывода.

0: Слаботоковые подтяжки включены (кроме портов, чьи выводы настроены как аналоговые входы или двухтактные цифровые выходы)

1: Слаботоковые подтяжки отключены

Бит 6: **XBARE**: Бит включения матрицы.

0: Матрица отключена. Все драйверы портов отключены.

1: Матрица включена

Бит 5: **T1E**: Бит подключения T1.

0: T1 не соединен с выводом порта.

1: T1 соединен с выводом порта.

Бит 4: **T0E**: Бит подключения T0.

0: T0 не соединен с выводом порта.

1: T0 соединен с выводом порта.

Бит 3: **ECIE**: Бит подключения внешнего входа счетчика ПМС.

0: ECI не соединен с выводом порта.

1: ECI соединен с выводом порта.

Биты 2-0: **PCA0ME**: Биты подключения входов/выходов модуля ПМС

000: Все входы/выходы модуля ПМС не соединены с выводами порта.

001: CEX0 соединен с выводом порта.

010: CEX0, CEX1 соединены с двумя выводами порта.

011: CEX0, CEX1, CEX2 соединены с тремя выводами порта.

100: CEX0, CEX1, CEX2, CEX3 соединены с четырьмя выводами порта.

101: CEX0, CEX1, CEX2, CEX3, CEX4 соединены с пятью выводами порта.

110: Зарезервировано

111: Зарезервировано

P0MDIN: Регистр настройки входов Порта 0.

Биты 7-0: P0MDIN.[7:0]: Биты выбора режима входов Порта 0.

Если вывод настроен как аналоговый вход, то его слаботоковая подтяжка, цифровой драйвер и цифровой приемник отключаются.

0: Вывод P0.n настроен как аналоговый вход.

1: Вывод P0.n не настроен как аналоговый вход.

P0MDOUT: Регистр настройки выходов Порта 0.

Биты 7-0: P0MDOUT.[7:0]: Биты настройки выходного драйвера порта 0: игнорируются, если соответствующий бит в регистре P0MDIN сброшен в 0.

0: Соответствующий вывод P0.n настроен как выход с открытым стоком.

1: Соответствующий вывод P0.n настроен как цифровой двухтактный выход.

Примечание: Если сигналы SDA и SCL появляются на любом выводе порта, то каждый из этих выводов будет настроен как выход с открытым стоком независимо от значения регистра P0MDOUT.

P0SKIP: Регистр выбора выводов Порта 0, пропускаемых матрицей.

Биты 7-0: P0SKIP.[7:0]: Биты выбора выводов порта 0, пропускаемых матрицей при назначении выводов. Эти биты выбирают выводы порта, пропускаемые декодером матрицы. Выводы порта, используемые как аналоговые входы (для АЦП или компаратора) или используемые для специальных целей (вход VREF, схема внешнего генератора, вход CNVSTR) должны пропускаться матрицей.

0: Соответствующий вывод P0.n не пропускается матрицей при назначении выводов.

1: Соответствующий вывод P0.n пропускается матрицей при назначении выводов.

Описания регистров управления таймерами

TCON: Регистр управления Таймерами 0 и 1.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значение при сбросе: 00000000 SFR Адрес: 0x88
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	

Бит 7: **TF1**: Флаг переполнения Таймера 1.

Устанавливается аппаратно при переполнении Таймера 1. Сбрасывается аппаратно при переходе к процедуре обслуживания прерывания от Таймера 1, но может быть сброшен и программно.

0: Переполнения Таймера 1 не обнаружено.

1: Таймер 1 переполнился.

Бит 6: **TR1**: Управление запуском Таймера 1.

0: Таймер 1 отключен.

1: Таймер 1 включен.

Бит 5: **TF0**: Флаг переполнения Таймера 0.

Устанавливается аппаратно при переполнении Таймера 0. Сбрасывается аппаратно при переходе к процедуре обслуживания прерывания от Таймера 0, но может быть сброшен и программно.

0: Переполнения Таймера 0 не обнаружено.

1: Таймер 0 переполнился.

Бит 4: **TR0**: Управление запуском Таймера 0.

0: Таймер 0 отключен.

1: Таймер 0 включен.

Бит 3: **IE1**: Внешнее прерывание 1.

Этот флаг аппаратно устанавливается в 1 при обнаружении активного фронта/уровня (определяется битом IT1) внешнего сигнала. Может быть сброшен программно, но при переходе к процедуре обслуживания внешнего прерывания 1 сбрасывается аппаратно, если IT1=1. При IT1=0 этот флаг устанавливается в 1, если на внешнем выводе /INT1 присутствует сигнал с активным логическим уровнем, который определяется битом IN1PL в регистре INT01CF.

Бит 2: **IT1**: Выбор типа внешнего прерывания 1.

Этот бит определяет, какое событие будет вызывать внешнее прерывание 1: фронт или активный уровень внешнего сигнала /INT1 (активный уровень внешнего сигнала /INT1 определяется битом IN1PL в регистре INT01CF).

0: Внешнее прерывание 1 вызывается активным уровнем сигнала /INT1.

1: Внешнее прерывание 1 вызывается фронтом сигнала /INT1.

Бит 1: **IE0**: Внешнее прерывание 0.

Этот флаг аппаратно устанавливается в 1 при обнаружении активного фронта/уровня (определяется битом IT0) внешнего сигнала. Может быть сброшен программно, но при переходе к процедуре обслуживания внешнего прерывания 0 сбрасывается аппаратно, если IT0=1. При IT0=0 этот флаг устанавливается в 1, если на внешнем выводе /INT0 присутствует сигнал с активным логическим уровнем, который определяется битом IN0PL в регистре INT01CF.

Бит 0: **IT0**: Выбор типа внешнего прерывания 0.

Этот бит определяет, какое событие будет вызывать внешнее прерывание 0: фронт или активный уровень внешнего сигнала /INT0 (активный уровень внешнего сигнала /INT0 определяется битом IN0PL в регистре INT01CF).

0: Внешнее прерывание 0 вызывается активным уровнем сигнала /INT0.

1: Внешнее прерывание 0 вызывается фронтом сигнала /INT0.

TMOD: Регистр режима Таймеров 0 и 1.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значение при сбросе: 00000000 SFR Адрес: 0x89
GATE1	C/T1	T1M1	T1M0	GATE0	C/T0	T0M1	T0M0	
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	

Бит 7: **GATE1**: Управление блокировкой Таймера 1.

0: Таймер 1 включен, если TR1 = 1, независимо от логического уровня на входе /INT1.

1: Таймер 1 включен только тогда, когда TR1 = 1 и на входе /INT1 активный логический уровень, определяется битом IN1PL в регистре INT01CF.

Бит 6: **C/T1**: Выбор режима таймера или счетчика для T/C1.

0: T/C1 работает как таймер: Таймер 1 инкрементируется от внутреннего сигнала

тактирования, который задается битом T1M (CKCON.4).

1: T/C1 работает как счетчик: Таймер 1 инкрементируется под воздействием перехода из 1 в 0 внешнего входного сигнала (T1).

Биты 5-4: **T1M1-T1M0**: Выбор режима работы Таймера 1.

Эти биты определяют режим работы Таймера 1.

T1M1	T1M0	Режим
0	0	Режим 0: 13-разрядный таймер/счетчик
0	1	Режим 1: 16-разрядный таймер/счетчик
1	0	Режим 2: 8-разрядный таймер/счетчик с автоперезагрузкой
1	1	Режим 3: Таймер 1 неактивен

Бит 3: **GATE0**: Управление блокировкой Таймера 0.

0: Таймер 0 включен, если TR0 = 1, независимо от логического уровня на входе /INT0.

1: Таймер 0 включен только тогда, когда TR0 = 1 и на входе /INT0 активный логический уровень, определяется битом IN0PL в регистре INT01CF (см. рис.8.13).

Бит 2: **C/T0**: Выбор режима таймера или счетчика для T/C0.

0: T/C0 работает как таймер: Таймер 0 инкрементируется от внутреннего сигнала

тактирования, который задается битом T0M (CKCON.3).

1: T/C0 работает как счетчик: Таймер 0 инкрементируется под воздействием перехода из 1 в 0 внешнего входного сигнала (T0).

Биты 1-0: **T0M1-T0M0**: Выбор режима работы Таймера 0.

Эти биты определяют режим работы Таймера 0.

T0M1	T0M0	Режим
0	0	Режим 0: 13-разрядный таймер/счетчик
0	1	Режим 1: 16-разрядный таймер/счетчик
1	0	Режим 2: 8-разрядный таймер/счетчик с автоперезагрузкой
1	1	Режим 3: Два 8-разрядных таймера/счетчика

Описание регистров управления АЦП

ADC0CF: Регистр конфигурации АЦП0

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значение при сбросе: 11111000 SFR Адрес: 0xBC
AD0SC4	AD0SC3	AD0SC2	AD0SC1	AD0SC0	AD0LJST	-	-	
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	

Биты 7-3: **AD0SC4-0**: Биты установки периода сигнала дискретизации АЦП0

Частота сигнала дискретизации АЦП0 определяется частотой системного тактового сигнала в соответствии со следующим уравнением: AD0SC = (SYSCLK/CLKSAR0)– 1,

где AD0SC – 5-разрядное значение, задаваемое битами AD0SC4-0

CLKSAR0 – необходимая частота сигнала дискретизации АЦП0

Бит 2: **AD0LJST**: Бит выравнивания результата преобразования

0: Данные в регистровой паре ADC0H:ADC0L выровнены вправо.

1: Данные в регистровой паре ADC0H:ADC0L выровнены влево.

Биты 1-0: Не используются: читаются как 00b. Запись не оказывает никакого влияния.

ADC0CN: Регистр управления АЦП0

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значение
AD0EN	AD0TM	AD0INT	AD0BUSY	AD0WINT	AD0CM2	AD0CM1	AD0CM0	при сбросе: 00000000
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	

SFR Адрес: 0xE8

(доступен в битовом режиме адресации)

Бит 7: **AD0EN**: Бит включения АЦП0

0: АЦП0 отключен. АЦП0 находится в режиме пониженного энергопотребления.

1: АЦП0 включен. АЦП0 находится в активном режиме и готов к преобразованию данных.

Бит 6: **AD0TM**: Бит установки режима слежения (выборки-хранения) АЦП0

0: Нормальный режим: Когда АЦП0 включен, слежение осуществляется всегда, за исключением момента преобразования.

1: Энергосберегающий режим: Режим слежения определяется битами AD0CM2-0 (см. ниже).

Бит 5: **AD0INT**: Флаг прерывания от АЦП0 (устанавливается при завершении преобразования)

0: АЦП0 не закончил преобразование данных (с момента последнего обнуления этого флага)

1: АЦП0 закончил преобразование данных

Бит 4: **AD0BUSY**: Бит занятости АЦП0

Чтение:

0: Преобразование данных завершено или в данный момент преобразование не осуществляется.

При аппаратном обнулении этого бита флаг AD0INT устанавливается в 1.

1: Идет процесс преобразования данных.

Запись:

0: Не оказывает никакого влияния.

1: Инициирует запуск преобразования АЦП0, если биты AD0CM2-0 = 000b

Бит 3: **AD0WINT**: Флаг прерывания от детектора диапазона АЦП0.

0: Преобразованные данные не соответствуют заданному диапазону (с момента последнего обнуления этого флага).

1: Преобразованные данные соответствуют заданному диапазону.

Биты 2-0: **AD0CM1-0**: Биты выбора режима запуска преобразования АЦП0.

Если AD0TM = 0:

000: Запуск преобразования осуществляется установкой в 1 бита AD0BUSY.

001: Запуск преобразования осуществляется при переполнении Таймера 0.

010: Запуск преобразования осуществляется при переполнении Таймера 2.

011: Запуск преобразования осуществляется при переполнении Таймера 1.

100: Запуск преобразования осуществляется нарастающим фронтом внешнего сигнала CNVSTR.

101: Запуск преобразования осуществляется при переполнении Таймера 3.

11x: Зарезервировано.

Если AD0TM = 1:

000: слежение (выборка) начинается в момент установки в 1 бита AD0BUSY и длится 3 периода сигнала дискретизации АЦП0, затем начинается преобразование данных.

001: слежение (выборка) начинается при переполнении Таймера 0 и длится 3 периода сигнала дискретизации АЦП0, затем начинается преобразование данных.

010: слежение (выборка) начинается при переполнении Таймера 2 и длится 3 периода сигнала дискретизации АЦП0, затем начинается преобразование данных.

011: слежение (выборка) начинается при переполнении Таймера 1 и длится 3 периода сигнала дискретизации АЦП0, затем начинается преобразование данных.

100: слежение (выборка) происходит лишь при низком уровне сигнала на входе CNVSTR;

преобразование запускается нарастающим фронтом сигнала на входе CNVSTR.

101: слежение (выборка) начинается при переполнении Таймера 3 и длится 3 периода сигнала дискретизации АЦП0; затем начинается преобразование данных.

11x: Зарезервировано.

Регистры специального назначения (special function registers - SFR):
PSW: Слово состояния программы

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значение при сбросе: 00000000
CY	AC	F0	RS1	RS0	OV	F1	PARITY	SFR Адрес: 0xD0
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0	

(доступен в битовом режиме адресации)

Бит 7: **CY**: Флаг переноса.

Этот бит устанавливается, если в результате последней арифметической операции произошел перенос (сложение) или заем (вычитание). Он сбрасывается в 0 всеми другими арифметическими операциями.

Бит 6: **AC**: Флаг десятичного переноса.

Этот бит устанавливается, если в результате последней арифметической операции произошел перенос (сложение) в старший полубайт или заем (вычитание) из старшего

полубайта. Он сбрасывается в 0 всеми другими арифметическими операциями.

Бит 5: **F0**: Флаг пользователя 0.

Это доступный в битовом режиме адресации флаг общего назначения, предназначен-ный для использования под управлением программы.

Биты 4-3: **RS1-RS0**: Биты выбора банка регистров.

Эти биты определяют активный банк регистров.

Бит 2: **OV**: Флаг переполнения.

Этот бит устанавливается в 1 в следующих случаях:

– если в результате выполнения команды ADD, ADDC или SUBB произошло переполнение с изменением знака;

– если в результате выполнения команды MUL произошло переполнение (результат превышает значение 255);

– если при выполнении команды DIV произошло деление на ноль.

Бит OV сбрасывается в 0 командами ADD, ADDC, SUBB, MUL и DIV во всех других случаях.

Бит 1: **F1**: Флаг пользователя 1.

Это доступный в битовом режиме адресации флаг общего назначения, предназначенный для использования под управлением программы.

Бит 0: **PARITY**: Флаг четности.

Этот бит устанавливается в 1, если сумма восьми бит в аккумуляторе нечетная и сбрасывается, если сумма четная.