**ВВЕДЕНИЕ**

В настоящее время микроконтроллеры имеются в каждом электронном устройстве. Они используются в бытовой технике, автоматизированных системах управления, автомобиле, системах безопасности и т.д.

Микроконтроллер (МК) – мини компьютер на одном кристалле, который может осуществлять вычисления и управления техническими объектами. Основное отличие микроконтроллера от компьютера в том, что МК предназначен для управления конкретным устройством (имеет узконаправленное предназначение), а персональный компьютер способен управлять и выполнять задачи многими разными устройствами, который сделает его более универсальным. МК работает по программе, которая заложена в память. Ведущими изготовителями являются производители, такие как TI (Texas Instruments), Microchip, Motorola, Atmel, Renesas и т.д. Из достаточно большого количества моделей можно выбирать соответствующий тип, зависимо от решаемой задачи. Для наблюдения за напряжением батареи и выдачи сигнала при повышение за пределы порогового значения достаточно МК с малым количеством входов и выходов. Если регулировать температуру в нескольких объектах и отображать данный процесс управления на цветном индикаторе или мониторе с сенсорным экраном, то в этом случае необходимо применять с несколькими выводами. Также отличаются они по количество разряда (бит), которые одновременно обрабатываются. На сегодняшний день применяются модули с разрядностью шины от 4 до 32 битов, но больше всего распространены с длиной слов от 8 до 16 битов.

Структурно МК состоит из отдельных функциональных блоков, выполняющие определенные функций и обеспечивающие работы системы. Особо важным элементом является микропроцессор. МП реализует обработку данных, оперирует над двоичными операндами с помощью арифметико-логическое устройство – АЛУ (англ. Arithmetic Logic Unit). Также, в составе МК имеются систем памяти и периферийные устройства. МК-ы из одного семейство классифицируется по показателям ОЗУ, ПЗУ и EEPROM памяти, также наличием либо отсутствием некоторых периферийных модулей.

Программная часть и программирование контроллеров требует особое внимания программиста, разработчика и тех, кто хочет заниматься с проектированием. Потому что результаты выполненных задач зависит от корректности листинга и правильности алгоритма. Для сопряжения аппаратной части с программной часть существуют специальные среды разработки. Типы среды разработок зависит от моделей контроллера.

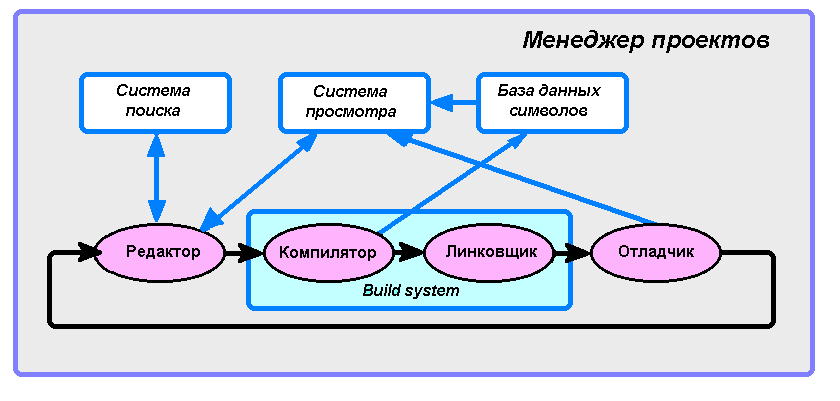
В данном учебнике рассматриваются контроллеры фирмы Freescale Semiconductor, семейства HCS12, а все примеры были запрограммированы для контроллера MC9S12CS128. Контроллеры других моделей данного семейства функционируют по такому же принципу и отличаются возможностями и системой команд. Указанные в этой книге примеры программировались в среде CodeWarrior с помощью отладочной платы APS12DT56. Кроме того для наглядных демонстрации результатов использовались специальная плата PBMCUSLK и комплекс NI ELVIS II+ .

Знакомство с

## Общая характеристика среды разработки Code Warrior IDE

Code Warrior (*в пер. – воитель с кодами*) – универсальная интегрированная среда разработки и отладки программного обеспечения для встраиваемых микропроцессорных систем. Управляющая оболочка *IDE CodeWarrior* – это стандартное *Windows* приложение, которое объединяет в себе менеджер проектов, редактор с интерактивной коррекцией ошибок, модуль управления компиляцией и созданием программы *(Build system)*, символьный отладчик. *IDE CodeWarrior* состоит из универсальных программных модулей (рис.4.1). Управляющая оболочка *IDE CodeWarrior* позволяет создавать исходные файлы и организовывать их в проект, определяющий законченное приложение. В дополнение к этому управляющая оболочка обеспечивает автоматическую компиляцию, ассемблирование и линковку/загрузку всех необходимых исходных файлов для получения готового абсолютного модуля программы для микроконтроллера. Управляющая оболочка *Code Warrior* имеет два типовых режима работы.

* Режим «построения» программы (*Build*) позволяет транслировать все файлы приложения и получать готовый запускаемый модуль программы.
* Режим отладки (*Debug*) обеспечивает отладку и испытание приложения с помощью встроенного отладчика или симулятора (рис. 4.2).



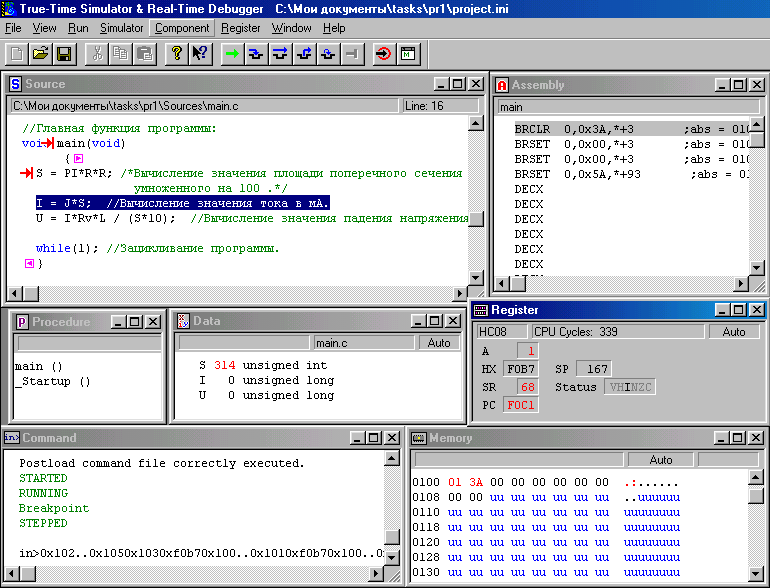
***Рис. 4.1. Структура интегрированной среды разработки CodeWarrior***

На основе универсальной среды *IDE CodeWarrior* компания *Freescale Semiconductor* выпускает программные средства, ориентированные на определенное семейство микропроцессоров или МК. Такие пакеты именуются CodeWarrior Development Studio или CodeWarrior Development Tools с указанием семейства МК, для которого они предназначены. В данном лабораторном практикуме Вы будете использовать пакет ***CodeWarrior Development Studio for HCS08.*** Для приобретения начальных навыков работы в среде *Code Warrior* самостоятельно проделайте на ПК те действия, которые описаны в разделах 4.2 и 4.3 данной главы, или воспользуйтесь англоязычным мультимедийным курсом *Code Warrior Leaning Edition*.

Очень интересным представляется учебный аспект интегрированной системы разработки *IDE CodeWarrior*. Причин тому несколько. Определяющей конечно является наличие полностью бесплатной академической версии с ограниченным объемом кода, который, однако, не оказывает существенного влияния на учебный процесс.

Возможность программирования как на языке ассемблера, так и на *С*, позволяет использовать CodeWarrior на различных стадиях обучения: начиная с курса основ микропроцессорной техники, который обязательно базируется на языке ассемблера, и заканчивая курсами по изучению алгоритмов работы МК в системах управления и обработки данных, в которых при надлежащем уровне обучения следует использовать языки программирования высокого уровня. Очень важным является тот факт, что уже на начальной стадии обучения студенту можно предложить профессиональный пакет разработки программного обеспечения для МК. Это позволит избежать непроизводительной траты времени на усвоение навыков работы с очередной учебной версией, которые зачастую можно забыть уже в следующем семестре. Состав и графический интерфейс современных средств разработки ПО для МК разных производителей в целом очень похожи, поэтому переход к какому-либо другому пакету в будущем уже не вызовет существенных затруднений.

Единый графический интерфейс CodeWarrior в сочетании с возможностью подключения аппаратных средств, выполненных на разных МК одного семейства, изменяют представление о составе технических средств учебной лаборатории. Теперь совсем не обязательно иметь так называемый фронтальный практикум, в котором все стенды одинаковы. Наоборот, можно показать разнообразие моделей, и при этом использовать фронтальный метод обучения, т.к. большая часть лабораторных работ будет одинаковой на всех стендах.



***Рис. 4.2. Графический интерфейс отладчика среды CodeWarrior***

Единая среда отладки, как в режиме полной программной симуляции, так и при работе с реальным контроллером, очень удобна не только ученику, но и опытному разработчику. Последний хорошо знает, что при работе с симулятором не удается “выловить” значительную часть ошибок только потому, что не хватает времени и терпения вручную перебирать все возможные комбинации входных воздействий. Аналогичной трудностью любого учебного процесса является невозможность “подключения” объекта управления в режиме симуляции. Авторы CodeWarrior преодолели эту проблему, предоставив в распоряжение пользователей виртуальные устройства ввода/вывода и объекты управления.

Программно-логический симулятор пакета CodeWarrior снабжен специальными утилитами внешних объектов, которые выбираются пользователем из уже готового перечня или создаются самим пользователем с использованием утилиты Visualisation Tool. Утилиты генерируют графический образ типовых устройств ввода и вывода МП системы, имитируют подключение этих устройств к портам МК, моделируют алгоритм работы каждого устройства.

## 4.2. Технология создания проекта в среде Code Warrior

В данном разделе рассматривается архитектура МК *MC9S08QG8*, программно-логическая модель центрального процессора *HCS08*, распределение адресного пространства МК *MC9S08QG8*. Изучается простейший формат записи программы на языке ассемблер, технология преобразования исходного текста программы в машинные коды для загрузки в МК, технология отладки разработанной программы с использованием программно-логической модели (симулятора) *IDE CodeWarrior* *Development Studio for HC(S)08.*

В ходе выполнения каждой лабораторной работы для создания некоторой программы управления необходимо выполнить стандартную последовательность действий:

* Ввод исходного текста программы на языке ассемблер с использованием редактора пакета интегрированной среды разработки Программного Обеспечения (ПО) для микропроцессорных систем *WinIDE CodeWarrior.*
* Ассемблирование исходного текста прикладной программы с целью получения машинного кода для загрузки его в память МК.
* Загрузка машинного кода созданной Вами программы в память МК.
* Ввод в память МК исходных данных для выполнения программы.
* Выполнение программы, анализ результатов работы программы.
* Корректировка допущенных ошибок.

Описанная последовательность действий именуется процессом разработки и отладки прикладной программы управления встраиваемой микропроцессорной системы. Далее описываются основные практические навыки управления программным пакетом “Интегрированная среда разработки *IDE CodeWarrior Development Studio for HC(S)08*” для МК модели *MC9S08QG8* семейства *HCS08.*

Чтобы начать работать с *Code Warrior* и приступить к созданию проекта, необходимо выполнить следующую последовательность операций.

Для этого запустить программу *CodeWarrior Development Studio for HC(S)08* и следовать по всплывающим диалоговым окнам для создания нового проекта.

Сначала появится окно рис. 4.3. В этом окне выбрать опцию *“Create New Project”,* если собираетесь создать новый проект. Если необходимо открыть свой предыдущий проект и на основе него произвести редактирование новой задачи, то следует выбрать опцию *“Load Previous Project”.*



***Рис.4.3. Первое окно для создания проекта***

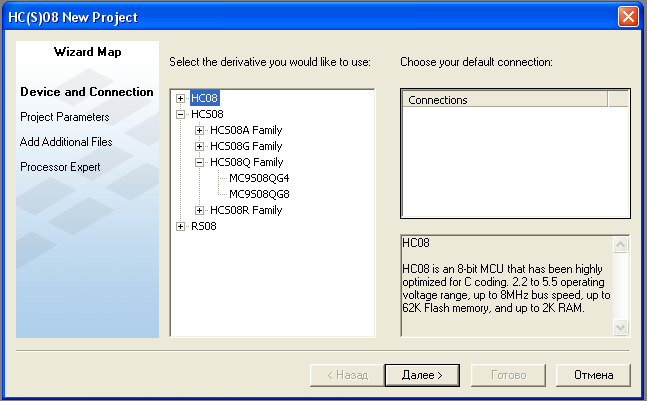
После этого откроется окно мастера создания новых проектов (рис. 4.4). В нем будет предложено выбрать модель МК, для которого будет создаваться прикладная программа, тип аппаратного устройства, на котором будет производиться отладка, и протокол связи с ним.

В левом окне следует выбрать сначала серию, а затем конкретную модель МК. В этом окне необходимо выбрать модель МК, для которой будете создавать проект - *MC9S08QG8*. Если среди предложенного списка МК *RS08, HC08, HCS08* нет необходимой модели, то она может быть добавлена с помощью мастера обновлений *CodeWarrior (Codewarrior updater tool).* В правом окне пользователь должен выбрать из предложенного списка тип аппаратного средства отладки. Возможные варианты выбора:

В диалоговом окне *“Connections”* этого же окна следует выбрать способ отладки проекта:

* ***Full Chip Simulation*:** полная программная симуляция МК выбранного типа. При этом способе отладки используется модель МК на компьютере, реальный МК не требуется. Этот способ отладки применяется для первичного тестирования алгоритмов, проверки времени выполнения алгоритмов без использования аппаратного устройства отладки;
* ***P&E Multilink/Cyclone Pro:*** *для* отладки приложения на аппаратном прототипе или на действующем устройсве. Для связи ПК с платой устройства используется *BDM* адаптер *P&E Multilink или Cyclone Pro.* Этот способ отладки позволяет тестировать программное обеспечение на действующем устройстве с микроконтроллером в реальном времени;

По умолчанию будет выбрана программная симуляция ***Full Chip Simulation***.



***Рис. 4.4. Второе окно для создания проекта***

Выполнив все установки в окне 2, нажмите кнопку “Далее”.Откроется окно 3, в котором предстоит выбрать используемый в проекте язык программирования, имя проекта и папку для файлов проекта (рис. 4.5).

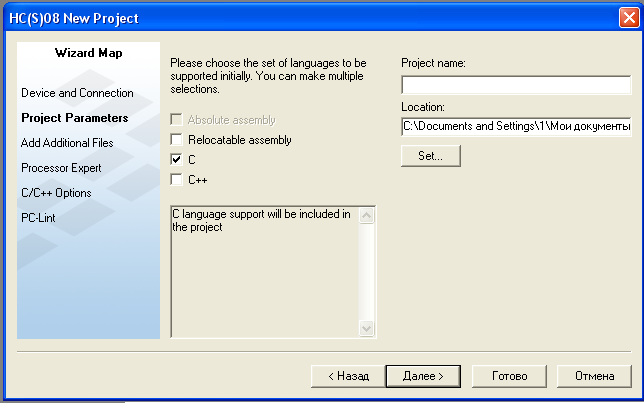
*IDE CodeWarrior* поддерживает несколько языков программирования: ассемблер*, C* и *C++.* Допускается возможность использовать их совместно в одном проекте. Язык *С* поддерживается всеми версиями *IDE* с ограничением максимального размера программы в 32 КБ в бесплатной специальной версии, которую можно загрузить с сайта компании *Freescale.* Язык *С++* поддерживается только профессиональной версией *CodeWarrior.*

Среди языков низкого уровня пользователь может выбрать абсолютный ассемблер, при программировании на котором переменные находятся в указанных программистом фиксированных ячейках памяти, или перемещаемый ассемблер, в котором используются специальные внутренние директивы, позволяющие линковщику выбирать адрес переменной. Перемещаемый ассемблер позволяет создать проекты, состоящие из нескольких файлов. Также он используется при комбинированном программировании на языке *C* и на ассемблере.

На начальном этапе изучения Вы будете использовать абсолютный ассемблер. Для того чтобы выбрать опцию *Absolute Assembly*, следует сначала отменить установленный по умолчанию выбор языка *C*.

После выбора одного или нескольких языков программирования для создаваемого проекта, необходимо указать имя проекта и адрес файлов проекта в диске

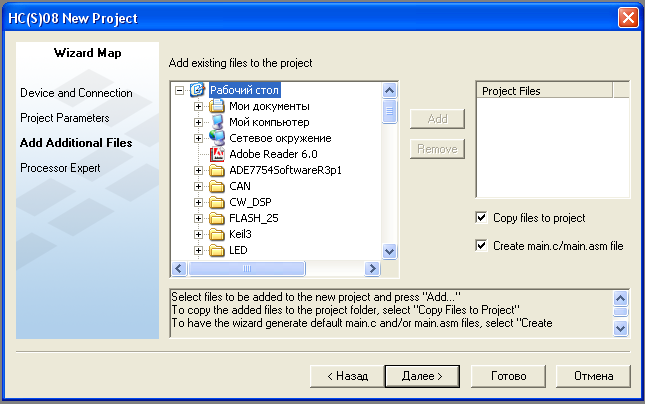
В левой верхней строке следует определить имя нового проекта, адрес местоположения файлов можно выбрать, нажав на кнопку *“Set”,* так и прописать прямой адрес к папке проекта, в которой желаете его расположить.



***Рис.4.5. Третье окно для создания проекта***

Выполнив все установки в окне 3, нажмите кнопку “Далее”. Мастер откроет окно 4 (рис. 4.6). Оно позволяет совершить два типа действий:

* ***Copy files to project –*** опция копирует все выбранные файлы в папку с создаваемым проектом.
* ***Create main.c/main.asm –*** опция автоматически создает файлы main.c или main.asm в папке с исходным кодом проекта.

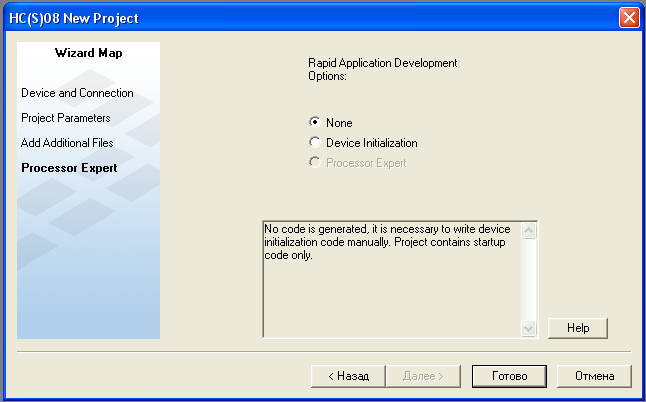
****

***Рис. 4.6. Четвертое окно для создания проекта***

Во время первого сеанса работы просто нажмите кнопку “Далее”. Вы падете в окно 5 мастера проектов (рис. 4.7). В этом окне происходит выбор приложения для ускорения создания программного обеспечения. В этом окне будет предложено три опции:

* ***None –*** не использовать дополнительных инструментов;
* ***Device initialization*** – использовать программный инструмент Device initialization. Он позволяет автоматически генерировать программный код начальной установки периферийных модулей, чем упрощает начальную настройку МК: С помощью интуитивно понятного графического интерфейса пользователь выбирает желаемые режимы работы периферийных модулей, а Device initialization генерирует программный код на ассемблере или на С, в котором записаны операции присвоения символьным именам регистров спецфункций подходящих численных значений;
* ***Processor Expert –*** использовать программный инструмент Processor Expert. Этот инструмент, так же, как Device initialization, позволяет автоматически генерировать программный код инициализации периферии, но дополнительно содержит в себе набор готовых типовых функций обслуживания периферийных модулей в процессе работы. В профессиональной версии Codewarrior, инструмент Processor Expert может генерировать код для некоторых алгоритмов.

Поскольку в данном лабораторном практикуме необходимо научиться настраивать МК и его периферию, то нет необходимости для проведения ассемблирования данной программы использовать специальные автоматизированные генераторы текста программ. Поэтому выберем опцию *None* и нажмем на кнопку “Далее”.



***Рис. 4.7. Пятое окно для создания проекта***

Окно 5 является завершающим в последовательности выбора опций для создаваемого проекта. Далее необходимо нажать на кнопку “Готово”, после этого *IDE* сгенерирует новый проект. Для последующей работы с ним перейдите к разделу 4.3.

**Работа с портами ввода/вывода микроконтроллера MC9S12C128**

*Цель работы:* Целью данной лабораторной работы является получение практических навыков в работе с портами ввода/вывода микроконтроллера.

*Необходимые оборудования:*

1. Плата APS12DT56 – представляет собой отладочную плату, совместившая функцию программатора. На плате расположены микроконтроллер, кварцевый резонатор, потенциометр, переключатели, управляющие кнопки, светодиоды, перемычки и интерфейсы. Для программирования микроконтроллера необходимо соединить плату с компьютером через USB-интерфейс.

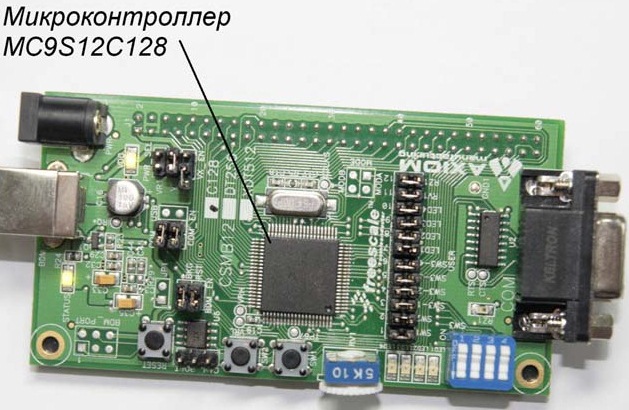


Рис.1 – Отладочная плата APS12DT56

2. Платформа NI ELVIS II+

3. Платформа PBMCUSLK

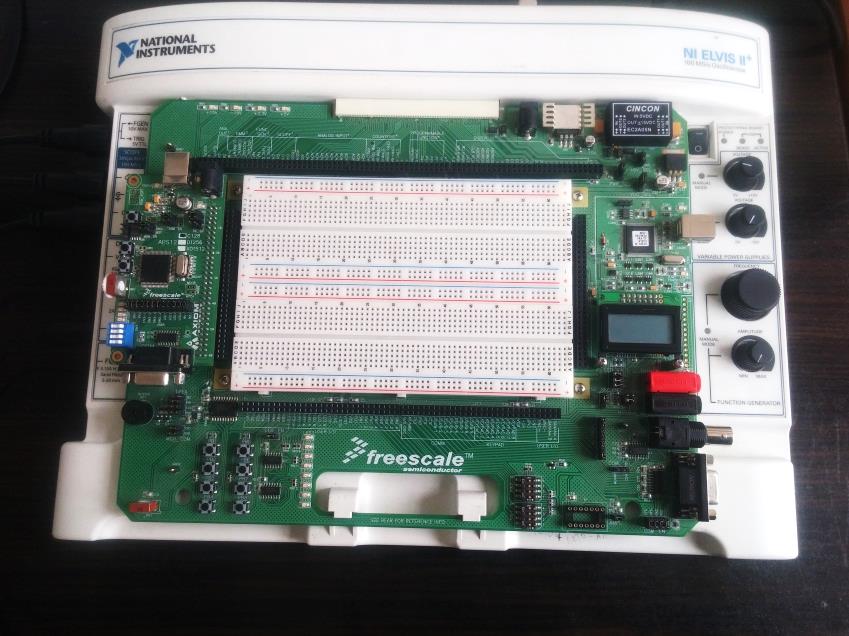
4. Персональный компьютер

Рис.2 – Платформа NI ELVIS II+ c платой PBMCUSLK

**1.2 Микроконтроллер MC9S12C128**

В данном лабораторном практикуме вами будет детально изучаться однокристальный 16-разрядный микроконтроллер семейства HCS12: MC9S12C128.

Основные технические характеристики этого МК:

* **16-разрядное процессорное ядро HCS12.**
* **Напряжение питания 2.97..5.5 В**. В лабораторном макете составляет 5 В.
* **Система тактирования.** Тактирование микроконтроллера возможно

как от внешних источников, так и от внутренних. В лабораторном стенде используется элемент, относящийся к первому типу: кварцевый резонатор. Частота тактового сигнала, который он генерирует, составляет 20 МГц. При этом частота внутренней шины микроконтроллера BUS f уменьшается вдвое и равна 10 МГц. Максимальное её значение для этой модели микроконтроллера составляет 20 МГц;

* **Резидентная память программ (ПЗУ).** Её объем равен 128K = 131072 ячеек (128

КБ). Память выполнена по технологии flash с эмуляцией EEPROM, число циклов

записи/стирания составляет не менее 100000.

* **Резидентная память данных (ОЗУ).** Объём составляет 4 КБ.
* **Общее число выводов микроконтроллера – 40.** Тип корпуса – QFP (Quad Flat

Package).

На кристалле микроконтроллера имеются следующие периферийные модули:

* **Порты ввода/вывода.** Всего доступно 9 портов, более подробное их описание

приведено в табл. 2.1.;

* **Модуль таймера TIM** с 16-разрядным счетчиком временной базы и восемью

каналами IC/OC/PWM.

* **Встроенный аналого-цифровой преобразователь ATD.** Число каналов

оцифровки равно 8, разрядность – 10 бит.

* **Блок широтно-импульсной модуляции PWM**
* **Последовательные интерфейсы:** синхронный SPI и асинхронный SCI, а также

модуль CAN со скоростью до 1 Мбит/с.

|  |  |  |
| --- | --- | --- |
| |  | | --- | |  | | Рис 1.3. Обобщенная структура МК семейства HCS12C128 | |
|  |

* 1. **Порты ввода/вывода**

Все МК семейства HCS12 имеют некоторое количество линий ввода/вывода данных. Линии объединены в 8 разрядные параллельные порты данных: Port A, Port B, Port E, За редким исключением, все линии ввода/вывода двунаправленные. Направление передачи линий ввода/вывода настраивается программно путем записи управляющего слова в регистр направления передачи соответствующего порта. Возможно изменение направления передачи в ходе выполнения программы посредством перепрограммирования этих регистров. Сигнал сброса устанавливает все двунаправленные линии в режим ввода. Следует особо подчеркнуть, что направление передачи каждой линии может быть выбрано разработчиком произвольно, независимо от других линий, принадлежащих к одному и тому же порту ввода/вывода. Исключение составляют лишь линии однонаправленной передачи, которые изначально специализированы на ввод или на вывод .

Часть линий ввода/вывода имеют так называемую альтернативную функцию, т.е. обеспечивают связь встроенных периферийных модулей МК с «внешним миром». Так линии порта PORT AD используются для подключения к встроенному АЦП измеряемых напряжений, линии порта PORT S служат входами и выходами контроллеров последовательного обмена. Если соответствующий периферийный модуль МК не используется, то его выводы можно задействовать как обычные линии ввода/вывода.

Если линии порта двунаправленные, то для его обслуживания такого порта предусмотрены два типа регистров:

PORTx — регистр данных порта x, где x — имя порта ввода/вывода;

DDRx — регистр направления передачи порта x.

Например, порт PORT A обслуживается регистрами PORTA и DDRA, а порт PORT B — регистрами PORTB и DDRB.

Если порт имеет схемотехнику с программно подключаемым «подтягивающим» резистором (R pullup), то для обслуживания такого порта предусмотрен дополнительный регистр входного сопротивления порта.

Ниже приведен фрагмент текста программы, которая конфигурирует PORT B для вывода данных, а затем записывает в порт число $62. Для того, чтобы все линии порта PORT B стали линиями вывода, необходимо записать в регистр направления передачи DDRB код $FF.

/\* МAIN PROGRAМ: \*/

/\*подключаемыефайлы\*/

#include <hidef.h>

#include"derivative.h"

void main{void) {

 unsigned char DDRB\_INIT = 0xFF;

 DDRB = DDRT\_INIT; //установить порт PORT B на вывод

 PORTB = 0x62;

}

* 1. **Спецификация портов ввода/вывода**

Подсистема параллельного ввода/вывода МК MC9S12C128состоит из 8 портов, причем линии многих портов обладают альтернативной функцией.

|  |
| --- |
| C:\Users\PHYSlC412_6\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Новый рисунок (3).bmp |
| 1.4. Цоколевка корпуса микроконтроллера **MC9S12C128** |

 PORT A. В однокристальном режиме работы — 8-разрядный порт ввода/вывода общего назначения. Направление передачи каждой линии порта определяется соответствующим битом регистра DDRA. В расширенном режиме работы на линиях порта формируются сигналы старшего байта мультиплексированной магистрали адрес/данные ADDR15–8/DATA15–8. В расширенном режиме с 8 разрядной шиной линии порта представляют собой мультиплексированную магистраль ADDR15–8/DATA7–0.

• PORT B. В однокристальном режиме работы — 8-разрядный порт ввода/вывода общего назначения. Направление передачи каждой линии порта определяется соответствующим битом регистра DDRB. В расширенном режиме работы на линиях порта формируются сигналы младшего байта мультиплексированной магистрали адрес/данные ADDR7–0/DATA7–0. В расширенном режиме с 8-разрядной шиной линии порта представляют собой немультиплексированную магистраль ADDR7-0.

• PORT E. 8-разрядный порт ввода/вывода общего назначения. Две линии порта PE0 и PE1 — однонаправленные и работают только на ввод. Остальные линии порта — двунаправленные, направление передачи линий PE2–PE7 определяется соответствующими битами регистра DDRE. Все линии порта имеют альтернативную функцию. Линии PE1 и PE0 могут использоваться как входы внешних прерываний  и . Остальные линии служат для задания режимов работы МК (MOD A и MOD B) и в качестве сигналов управления внешней шиной при работе МК в расширенном режиме.

• PORT AD. Однонаправленный 8-разрядный порт ввода. Все линии имеют альтернативную функцию. Если работа модуля аналого-цифрового преобразователя ATD разрешена, то линии порта используются для подключения измеряемых аналоговых сигналов.

• PORT T. Двунаправленный 8-разрядный порт ввода/вывода общего назначения. Направление передачи каждой линии порта определяется соответствующим битом регистра DDRT. Альтернативная функция линий порта PORT T — обслуживание модуля таймера. Если работа таймера разрешена, то линии используются в качестве входов входного захвата IC или выходов выходного сравнения OC.

• PORT S. Двунаправленный 8-разрядный порт ввода/вывода общего назначения. Направление передачи каждой линии порта определяется соответствующим битом регистра DDRS. Альтернативная функция линий порта PORT S — обслуживание модулей последовательного обмена SCI и SPI.

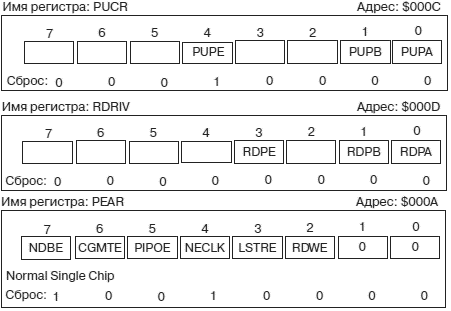
• PORT P. Двунаправленный 8-разрядный порт ввода/вывода общего назначения. Направление передачи каждой линии порта определяется соответствующим битом регистра DDRP. Четыре линии порта PORT P могут использоваться в качестве выходов модуля генератора ШИМ-сигнала (модуль PWM), если работа последнего программно разрешена.

* 1. **Регистры управления портами**

В МК семейства 68HC12/HCS12 каждый двунаправленный порт ввода/вывода общего назначения обслуживается двумя регистрами специальных функций. Это регистр данных порта и регистр направления передачи DDRx (вместо буквы «x» следует подставить буквенное обозначение порта). Если линии порта общего назначения настроены на ввод, то операция чтения регистра данных возвращает состояние выводов корпуса МК, с которыми связан порт. Если порт настроен на вывод, то операция записи в регистр данных устанавливает на выводах корпуса МК, связанных с портом, соответствующие логические уровни. Регистр DDRx определяет направления передачи каждой линии порта независимо от других линий этого же порта. Если какой либо бит регистра DDRx равен 0, то соответствующая линия настраивается на ввод, если 1 — то на вывод. Возможны решения, при которых часть линий одного и того же порта настроена на ввод, а часть на вывод. Например, при значении DDRx=10110010 линии D6, D3, D2 и D0 развернуты на ввод, а линии D7, D5, D4 и D1 — на вывод. В состоянии сброса МК все биты регистров направления передачи DDRx сбрасываются, поэтому сразу после включения питания все линии портов МК сконфигурированы как входы с высоким входным сопротивлением.

Часть портов ввода/вывода обслуживается дополнительными регистрами управления:

• PUCR (PullUpControlRegister) — регистр разрешения схемотехники подтягивающих резисторов. Формат регистра представлен на рис. 1. Если соответствующие биты регистра установлены, то в портах PORT A, PORT B и PORT E при конфигурировании какой либо линии порта на ввод автоматически подключается встроенный подтягивающий к напряжению питания резистор. Если же эта линия настраивается на вывод, то встроенный резистор автоматически отключается. В расширенных режимах работы МК, когда названные порты используются для формирования сигналов внешних магистралей адреса, данных и управления, встроенные резисторы также автоматически отключаются.



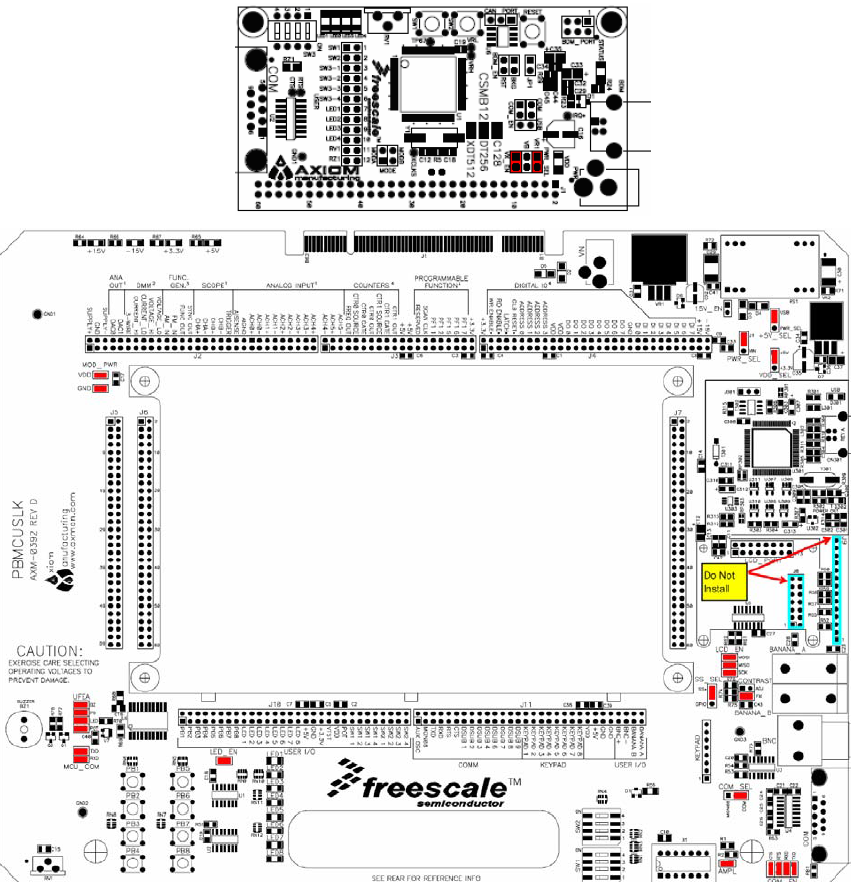
**Рис. 1.7.**Формат регистров PUCR, RDRIV, PEAR

• RDRIV (ReducedDRIVeRegister) — регистр выбора режима работы выходных каскадов с пониженными выходными токами. Формат регистра также представлен на рис. 1.7. Как следует из рис. 1.7, в МК MC9S12C128 этот режим также доступен только для портов PORT A, PORT B и PORT E. В МК иных моделей этой функцией могут обладать также и другие порты. Если функция для порта разрешена установкой разряда RDPx в 1, то при конфигурировании какой либо линии порта на вывод ее выходной ток снижается с 0,6 мА до 0,3 мА.

• PEAR (Port E AssignmentRegister) — регистр выбора назначения линий порта Port E. Отдельные биты этого регистра позволяют назначить линии альтернативную функцию или функцию линии ввода/вывода общего назначения.

# Порядок выполнения работы

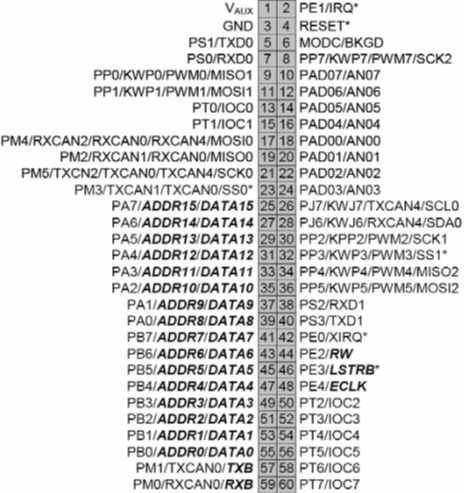
1. *Осторожно вставьте отладочную плату APS12C128 на платформу PBMCUSLK.*
2. *Соотвтственно с рисунком 3.1 ставим перемычки рабочей платы CSMB12С128 и PBMCUSLK.*

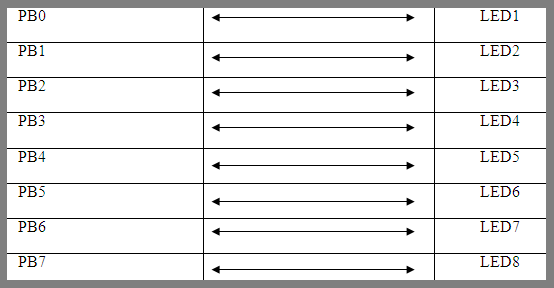
**

*Рис. 3.1. Конфигурация расположения перемычек на платах CSMB12С128 и PBMCUSL при их*

*совместном использовании с компьютером*

1. *Подсоединяем одладочную плату к компьютеру через порт USB*
2. *По ниже веденного схемы соедините порт В микроконтроллера к светодиодам платформы PBMCUSLK.*

**

**

1. *Проверте переключатели платы APS12C128 платы. Они дольжны быть в выключенном ввиде.*
2. *Включите компьютер и дайте ему загрузиться.*
3. *По этому ссылку запускаем CodeWarrior: . Пуск – Все программы – Freescaler CodeWarrior - CodeWarrior Development Studio for S12(X) – CodeWarrior IDE.*
4. *Создаем новый проект. В качестве языка программирования выбираем язык Си.*
5. *В окно текстового редактора рабочего среды CodeWarrior IDE, напишем ниже приведенный программный код.*

|  |
| --- |
| C:\Users\PHYSlC412_6\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Новый рисунок (20).bmp |

1. *С помощью команды “MAKE” проверяем написанный код программы.*
2. *Если проверка прошло успешно, нажимаем “DEBUG”.*
3. *В появившимся новом окне под названием True-Time Simulator & Real-Time Debugge , нажимаем “OK” и после прошивки программы нажимаем кнопку “Start/Continue (F5)”.*
4. *Закрываем окно True-Time Simulator & Real-Time Debugger, и анализируем полученный результат.*

**Выполните следующие задания:**

1. С помощью 4-х светодиодов на рабочей платформе CSMB12С128, реализуйте программу,чтобы светодиоды загорались по порядку, соответственно ниже приведенному рисунку.



1. Используя кнопку на рбочей платформе CSMB12С128 проделайте следующие работы:

С помощью порта В выведите двоичные коды цифр. При нажатии кнопки, к значении цифры должна прибавляться единица.

**Вопросы для проверки:**

1. Сколько портов ввода/вывода в MC9S12C128?
2. Какие альтернативные функции реализуют порты AD, T, S, P, Е?
3. Каково назначение регистра направления передачи порта?
4. В какое состояние устанавливается регистр направления передачи во время сброса МК?

**Литература**

1. [Иди Ф.](http://padabum.com/search.php?author=%D0%98%D0%B4%D0%B8%20%D0%A4.), Сетевой и межсетевой обмен данными с микроконтроллерами - Додэка-XXI,2007

# [Петров И.В.](http://padabum.com/search.php?author=%D0%9F%D0%B5%D1%82%D1%80%D0%BE%D0%B2%20%20%D0%98.%D0%92.), Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования - Солон-Пресс, 2004

# [Шпак Ю.А.](http://padabum.com/search.php?author=%D0%A8%D0%BF%D0%B0%D0%BA%20%D0%AE.%D0%90.), Программирование на языке С для AVR и PIC - МК-Пресс, 2-е издание, 2011

# [Голубцов М.С.](http://padabum.com/search.php?author=%D0%93%D0%BE%D0%BB%D1%83%D0%B1%D1%86%D0%BE%D0%B2%20%D0%9C.%D0%A1.), AVR - от простого к сложному - 2003

# С. Ф. Баррет., Д. Дж. Пак., Встраиваемые системы. Проектирование приложений на микроконтроллерах семейства 68HC12/HCS12 с применением языка С – 2006