1. СТРУКТУРА И АРХИТЕКТУРА МК: ОБЩИЕ ВОПРОСЫ

1.1. Общие положения. Определения структуры и архитектуры МК. Основные элементы архитектуры

Под **структурой** МК понимается собственно структурная схема реализации МК в целом, а также его функциональных узлов и блоков.

Несмотря на большое разнообразие выпускаемых в настоящее время моделей МК, основными компонентами структуры всех МК являются:

- центральный процессор (ЦП);
- память;
- периферийные устройства (ПУ) (порты ввода-вывода, таймеры и т. п.), называемые так потому, что структурно они не входят в состав ЦП, однако конструктивно располагаются на том же кристалле и в том же корпусе, что и ЦП.

Общий принцип работы МК состоит в осуществляемых ЦП обработке данных и управлении функциональными блоками МК, в соответствии с хранимой в памяти последовательностью команд. Конфигурация и управление блоками МК осуществляется посредством кодов, загружаемых в **программно-доступные регистры** данных блоков; их обобщенное название – **регистры специальных функций (РСФ)**. При этом регистры ЦП, служащие для промежуточного хранения данных и результатов обработки, называют **регистрами общего назначения (РОН)**.

Типовые примеры структуры МК представлены далее, в подразделе 1.2.

В свою очередь, под **архитектурой** МК понимается его модель с точки зрения разработчика электронных средств на его базе [3]. Основными элементами архитектуры МК являются [3]:

- режимы работы ЦП;
- типы, форматы и разрядность обрабатываемых данных;
- организация памяти, т. е. состав, назначение и объем ее модулей и/или областей, форматы адресов их содержимого и порядок доступа к нему;
- **регистровая модель ЦП**, т. е. состав, форматы и назначение его программно-доступных регистров;
- состав системы команд, под которой здесь и в дальнейшем будет пониматься **система команд машинного языка** (т. е. двоичных кодов, воспринимаемых ЦП без промежуточной

трансляции) и соответствующих им команд **языка ассемблера МК** (см. пункт 2.6.1), форматы команд и количество тактов синхрогенератора МК, необходимых для их выполнения;

- процедуры обслуживания запросов на прерывания;

- состав и назначение периферийных (по отношению к ЦП) устройств МК, их **регистровые модели** (состав, форматы и назначение программно-доступных регистров), процедуры задания реализуемых ими функций и их взаимодействия с ЦП;

- протоколы и процедуры программирования памяти МК, т. е. загрузки в нее программного обеспечения.

По принципу организации памяти архитектуру МК относят к одному из следующих базовых классов [3]:

- **Гарвардская архитектура**, характеризующаяся физическим разделением памяти программ и памяти данных, т. е. хранением программ и данных в физически различных блоках памяти; в частности, по Гарвардской архитектуре строятся МК семейства AVR (см. рис. 1.1);

- **фон Неймановская архитектура**, для которой характерно хранение программ и данных в физически одном и том же блоке памяти.

Фон Неймановская архитектура характеризуется меньшими (при прочих равных условиях) аппаратурными затратами на реализацию МК, чем Гарвардская, что было критично в недавнем прошлом, но не столь существенно в настоящем. С другой стороны, Гарвардская архитектура позволяет совмещать во времени обращение к памяти команд и к памяти данных за счет их физического разделения, что способствует повышению производительности МК (см. пункт 2.6.4). Поэтому в настоящее время для большинства МК общего назначения характерна Гарвардская архитектура.

С другой стороны, по составу системы команд различают [3]:

- ***CISC*-архитектуру** (от английского словосочетания *Complex Instruction Set Computer*, в дословном переводе – «компьютер со сложным набором команд», более корректный с технической точки зрения перевод – «компьютер с расширенным набором команд»);

- ***RISC*-архитектуру** (от английского словосочетания *Reduced Instruction Set Computer*, в переводе – «компьютер с сокращенным набором команд»).

Система команд МК с *CISC*-архитектурой (*CISC*-МК), как правило, включает в себя практически все команды, необходимость

в которых может возникнуть при решении задач, на которые ориентирован соответствующий МК. В свою очередь, система команд *RISC*-МК включает в себя только набор наиболее часто используемых простых инструкций (например, сложения, вычитания, пересылок «регистр - регистр» и «регистр - память» и т. п.). Более сложные процедуры реализуются в виде подпрограмм, написанных с использованием указанного простого набора команд.

Основными преимуществами *RISC*-архитектуры по сравнению с *CISC*-архитектурой являются следующие:

- упрощение аппаратных средств ядра МК (в первую очередь – декодера команд) и, как следствие, высвобождение места на кристалле под дополнительные РОН и периферийные устройства;

- возможность выполнения большинства команд за один такт МК, ввиду их простоты, и, как следствие, реального быстродействия МК в целом.

Данные преимущества весьма важны с точки зрения типовых областей практического применения МК. Поэтому большинство современных МК (особенно относительно несложных) реализуются на основе *RISC*-архитектуры.

Таким образом, для большинства современных МК характерна **Гарвардская *RISC*-архитектура.** Поэтому при дальнейшем изложении материала в качестве типовых примеров будут использоваться, в основном, МК именно с данным вариантом архитектуры, к которым относятся МК, структурные схемы которых приведены в подразделе 1.2.

Структура и архитектура МК тесно связаны между собой. Поэтому при дальнейшем изложении материала данного пособия будем использовать понятие **структурно-архитектурного решения МК**, т. е. сочетания его структуры и архитектуры. Аналогично, будет использоваться понятие структурно-архитектурных решений функциональных блоков МК, которое включает в себя:

- структурную схему блока;

- **регистровая модель блока**, т. е. состав, форматы и назначение его программно-доступных регистров;

- режимы работы блока;

- алгоритмы / протоколы его конфигурации и управления им в процессе работы и т. п.

Например, структурно-архитектурное решение встроенного АЦП МК включает в себя: структурную схему блока АЦП; режимы

его работы; состав, форматы и назначение его программно-доступных регистров; протоколы его конфигурации под каждый из возможных режимов работы, запуска процесса аналого-цифрового преобразования и считывания его результатов и т. п.

Введем тесно связанное с понятием архитектуры МК понятие **семейства** МК – группы серийно выпускаемых моделей БИС МК с одинаковыми базовыми элементами архитектуры, а именно:

- типом и архитектурой ЦП (см. раздел 2);
- типами, форматами и разрядностью обрабатываемых данных;
- организацией памяти (кроме объема ее модулей и/или областей);
- системой команд;
- процедурами обслуживания запросов на прерывания;
- процедурами программирования ПУ и их взаимодействия с ЦП.

С другой стороны, БИС МК одного и того же семейства отличаются между собой, в основном:

- объемом модулей памяти программ и данных;
- количеством и разрядностью портов ввода / вывода;
- количеством и составом ПУ (АЦП, таймеров, блоков стандартного цифрового интерфейса и т. п.).

В настоящее время различными компаниями мира (в т. ч. предприятиями электронной промышленности РФ) выпускается широкая номенклатура семейств и моделей БИС МК общего назначения. Исторически и организационно сложилось, что в различных странах популярны МК различных семейств. В РФ и в большинстве европейских стран – это семейства *AVR* и *ARM Cortex-Mx*. Поэтому дальнейшее изложение материалов данного пособия базируется, в первую очередь, на использовании структурно-архитектурных решений МК перечисленных семейств в качестве типовых примеров (при необходимости – с привлечением показательных примеров данных решений МК других семейств). Следует, однако, отметить, что структура и архитектура МК одинакового класса сложности практически всех семейств отличаются значительным сходством. Поэтому для читателя, освоившего материалы данного пособия, не представит серьезной проблемы применение для решения поставленных перед ним задач,

при необходимости, МК других семейств, структурно-архитектурные решения которых в пособии не отражены. См. также выводы по разделу 1.

1.2 Типовые структурные схемы, состав и назначение подсистем и функциональных блоков МК

Как указано ранее, МК общего назначения представляет собой БИС относительно простого однокристального компьютера, содержащую процессор, модули памяти и набор периферийных устройств, ориентированный на решение задач управления техническими объектами определенного уровня сложности. От него, в свою очередь, зависит класс / категория МК и состав его функциональных узлов и блоков.

Рассмотрим типовые структурные схемы основных классов МК общего назначения:

- «*cost-sensitive*» (бюджетных, невысокого уровня сложности);
- «*mainstream*» (базового, среднего уровня сложности);
- «*high performance*» (высокого уровня сложности).

К классу «*cost-sensitive*» относят относительно несложные, недорогие МК, выпускаемые в корпусах с числом выводов («пинов») от 6-и до 20-и. Типовой объем их встроенной (резидентной) памяти команд – порядка единиц килобайт, памяти данных – порядка сотен байт. Для них характерен ограниченный состав периферийных устройств (один – два порта ввода / вывода, один – два таймера, у ряда моделей – один блок стандартного цифрового интерфейса, один блок АЦП или один – два аналоговых компаратора). МК данного класса предназначены для «интеллектуального» управления сравнительно простыми техническими объектами (в первую очередь – бытовой техникой). Следует отметить, что ввиду современной тенденции к миниатюризации электронных устройств, с одной стороны, и постоянного снижения стоимости МК – с другой, даже относительно простые электронные устройства, (например, управляемые кодом генераторы импульсов) в настоящее время рациональнее строить на МК класса «*cost-sensitive*», а не на ИС малой или средней степени интеграции.

В качестве типовых примеров МК класса «*cost-sensitive*» могут служить:

- МК *PIC16F505* [4], один из простейших МК данного класса (и МК в целом), являющийся 8-битовым *PIC*-МК младшего подсемейства; его структурная схема представлена на рис. 1.1, а пояснения к ней – в табл. 1.1;

- КР1878ВЕ1 [5], принадлежащий к семейству ТЕСЕЙ (Российская разработка), по архитектуре близкому к семейству *PIC*; его структурная схема представлена на рис. 1.2, а пояснения к ней – также в табл. 1.1.

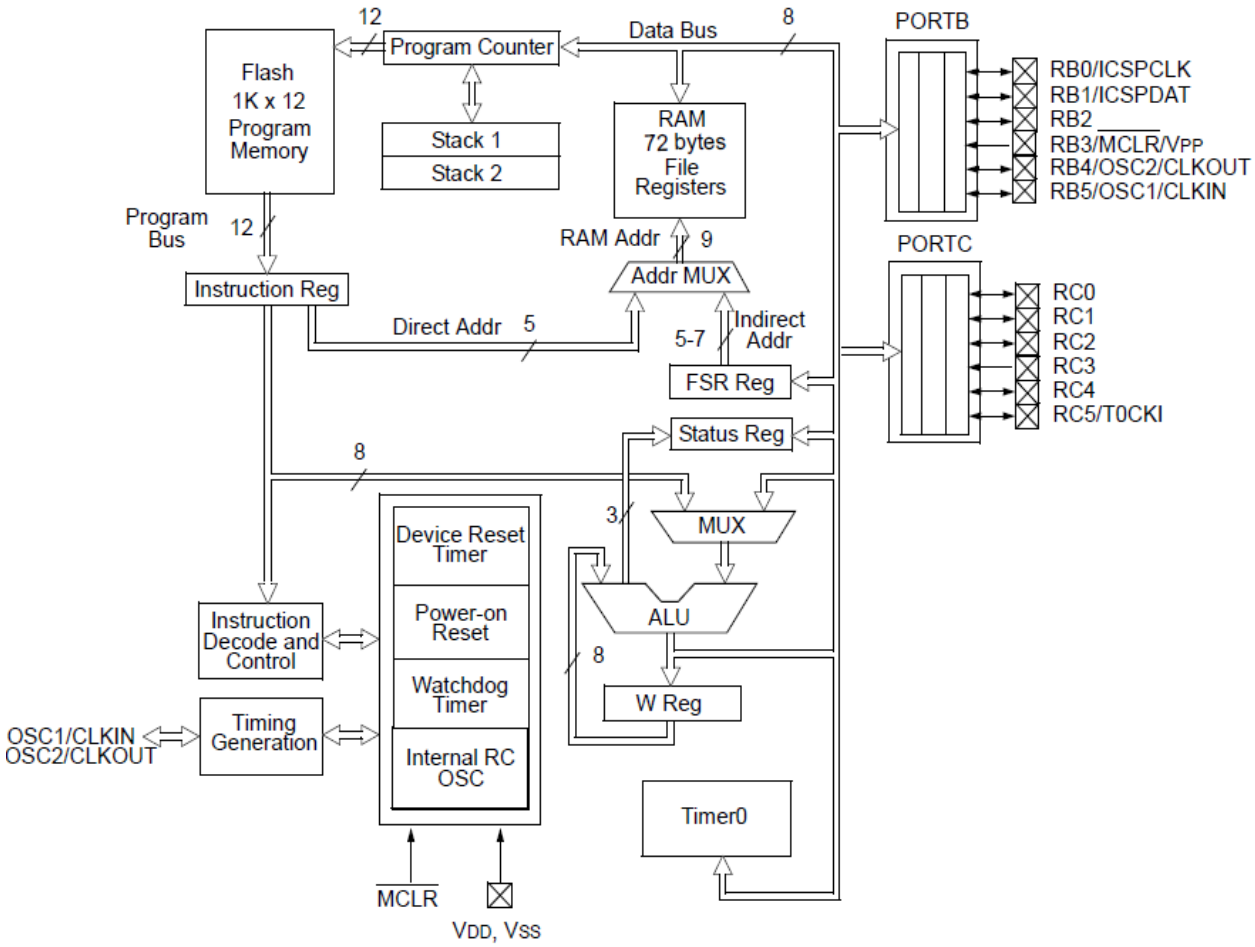


Рис. 1.1. Структурная схема МК *PIC16F505* [4] (пояснения – в табл. 1.1)

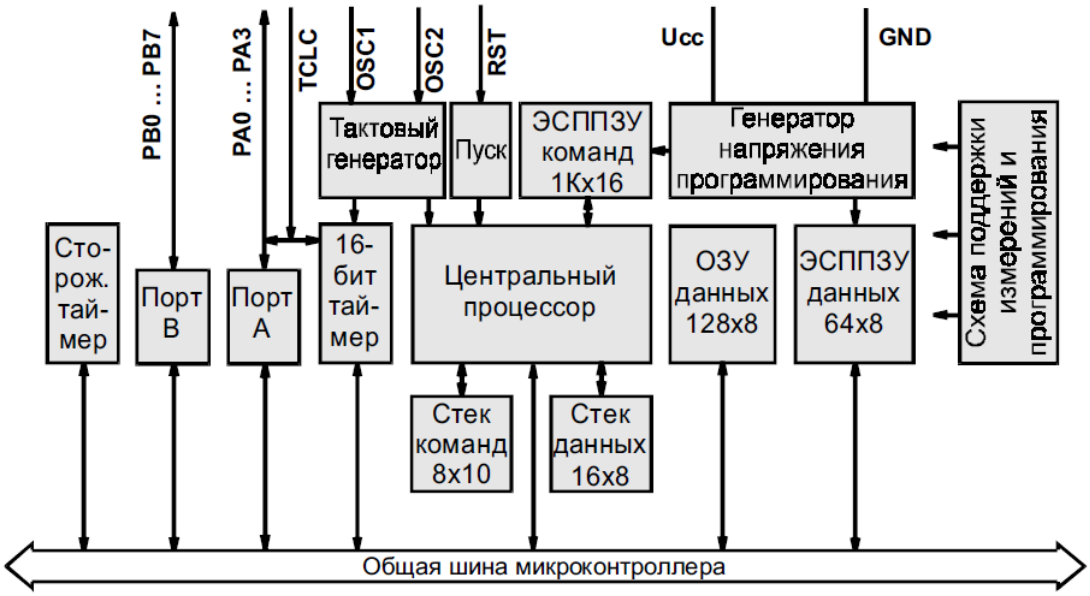
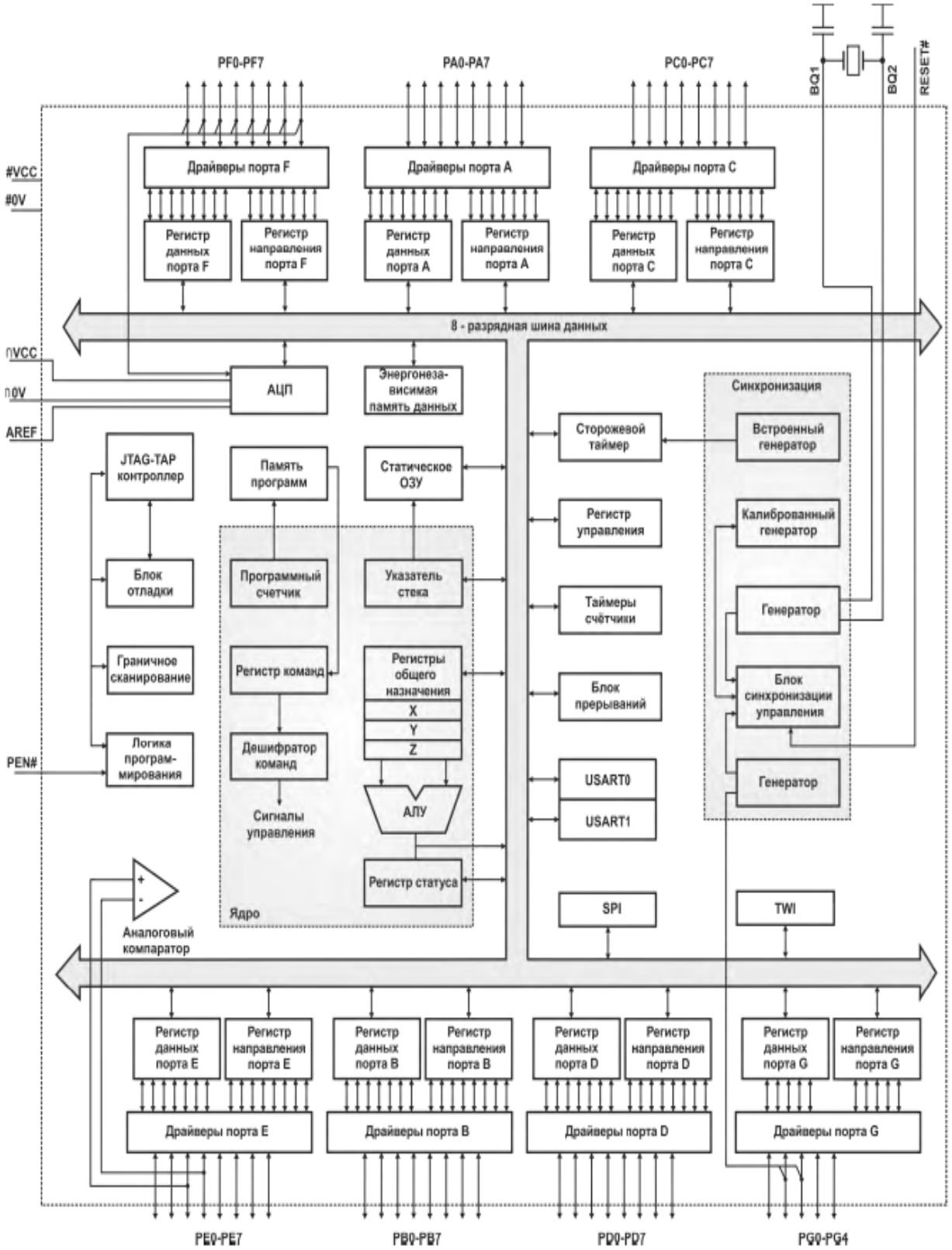


Рис. 1.2. Структурная схема МК КР1878ВЕ1 [5] (пояснения – в табл. 1.1)

К классу «*cost-sensitive*» также относятся МК подсемейства *ATtiny* семейства *AVR* [6], базовые структурно-архитектурные решения которых существенно не отличаются от таковых МК подсемейства *ATmega* того же семейства, относящихся к классу «*mainstream*», и отличающихся от МК *ATtiny*, в основном, объемами памяти и расширенным составом периферийных устройств. На примере подсемейства *ATmega* в данном и последующих разделах будут рассматриваться типовая структура и архитектура МК класса «*mainstream*». Поэтому отдельное рассмотрение структурно-архитектурных решений МК подсемейства *ATtiny* представляется излишним. При необходимости, с ними можно ознакомиться, например, по источнику [7].

К классу «*mainstream*» относят МК среднего (базового) уровня сложности, выпускаемые в корпусах с числом выводов порядка нескольких десятков. Они характеризуются объемом встроенной памяти команд порядка нескольких десятков килобайт, памяти данных – порядка единиц килобайт. Типовой состав их ПУ – несколько портов ввода / вывода, несколько многофункциональных таймеров, один или два АЦП, аналоговые компараторы, несколько блоков стандартного цифрового интерфейса. Типовыми МК класса «*mainstream*» являются МК подсемейства *ATmega*, реализованные по расширенной архитектуре *AVR* [6], например, *ATmega128* и его

отечественный аналог 1887ВЕ7Т [8]. Его структурная схема приведена на рис. 1.3, а пояснения к ней – в табл. 1.1.



#VCC, ∩VCC – напряжения питание цифровой и аналоговой части
#0V, ∩0V – общая шина цифровой и аналоговой части

Рис. 1.3. Структурная схема МК 1887ВЕ7Т [8]
(пояснения – в табл. 1.1)

К классу «*high performance*» относят МК относительно высокой (по сравнению с предыдущими классами) сложности, выпускаемые в корпусах с числом выводов от нескольких десятков до нескольких сотен. Объем их встроенной памяти команд – от сотен килобайт до единиц мегабайт, памяти данных – от десятков до сотен килобайт. МК данного класса характеризуются развитой системой ПУ, включающей в себя от 5 – 6 до 10 – 15 портов ввода / вывода, два – три многоканальных АЦП, один – два ЦАП, от 3 – 4 до 10 – 15 многофункциональных таймеров, разнообразные блоки стандартного цифрового интерфейса (*USART, SPI, I2C, CAN, USB, Ethernet* и др.).

Типовыми МК класса «*high performance*» являются МК на базе процессоров семейства *ARM Cortex-Mx* [9]. В данном семействе процессоров, в свою очередь, выделяется несколько подсемейств (*ARM Cortex-M0, ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4* и т. д. [9]), отличающихся между собой сложностью архитектуры и производительностью, при общих базовых структурно-архитектурных решениях. МК на основе процессоров семейства *ARM Cortex-Mx* выпускаются рядом компаний, в том числе АО «ПКК Миландр» (Россия). Широко распространенными и популярными у разработчиков являются МК *STM32xxx* производства компании *ST Microelectronics* [9].

В дальнейшем, вместо словосочетаний «МК на основе процессора семейства *ARM Cortex-Mx*» или, например, «МК на основе процессора подсемейства *ARM Cortex-M3*», для краткости, будем употреблять словосочетания «**МК семейства *ARM Cortex-Mx***» или «**МК подсемейства *ARM Cortex-M3***».

На рис. 1.4 представлена структурная схема МК модельного ряда *STM32F103x* [10] производства компании *ST Microelectronics*, относящихся к подсемейству *ARM Cortex-M3* (характеризуемому средней сложностью в пределах семейства), а на рис. 1.5 – МК К1986ВЕ92F1I того же подсемейства производства АО «ПКК Миландр» [11].

Пояснения к структурным схемам, приведенным на рис. 1.1 – 1.5, представлены в табл. 1.1. В ней перечислен типовой состав подсистем МК общего назначения, приведены краткие описания выполняемых ими функций, а также указаны блоки МК, относящиеся к соответствующим подсистемам.

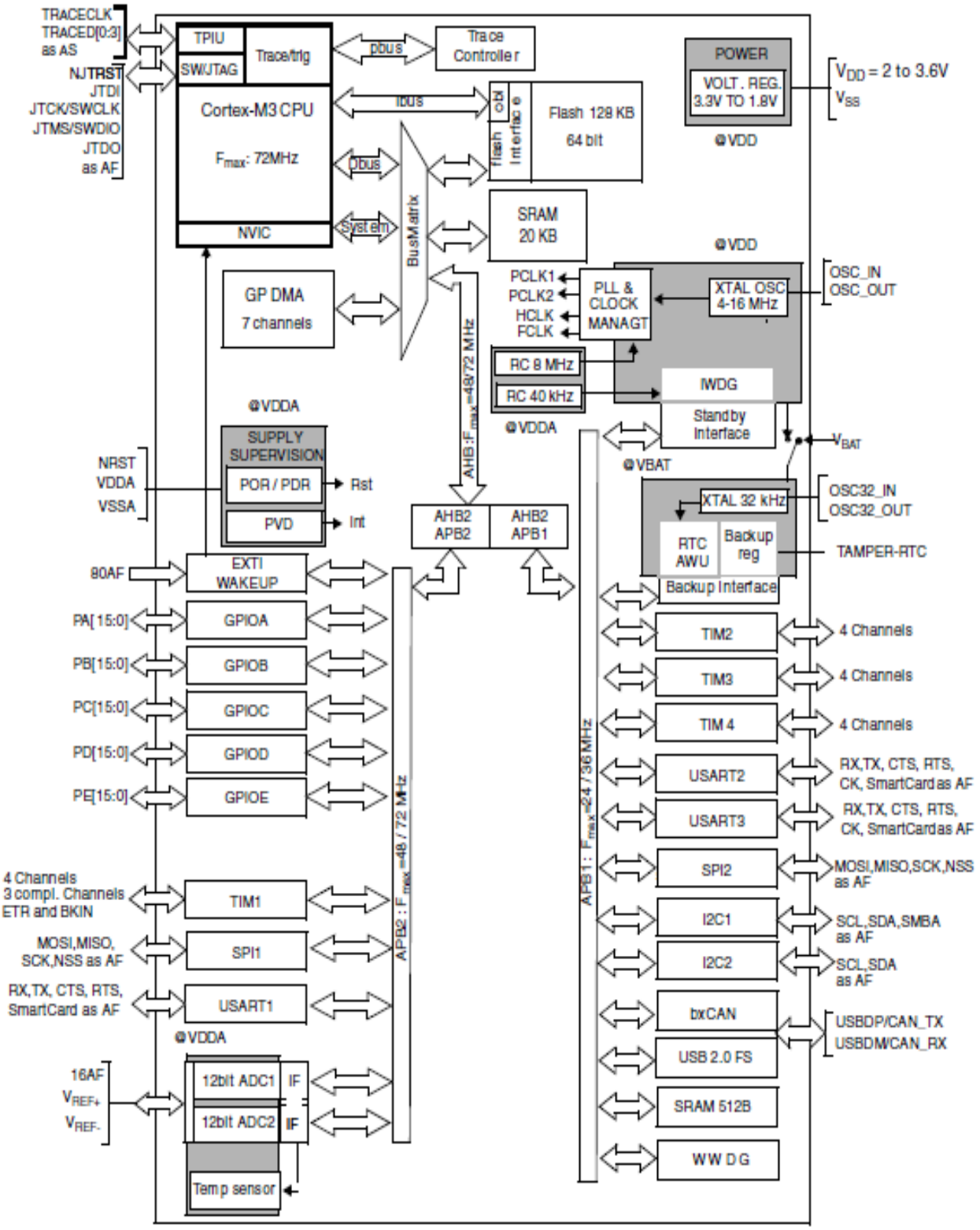


Рис. 1.4. Структурная схема МК модельного ряда *STM32F103x* [10] (пояснения – в табл. 1.1)

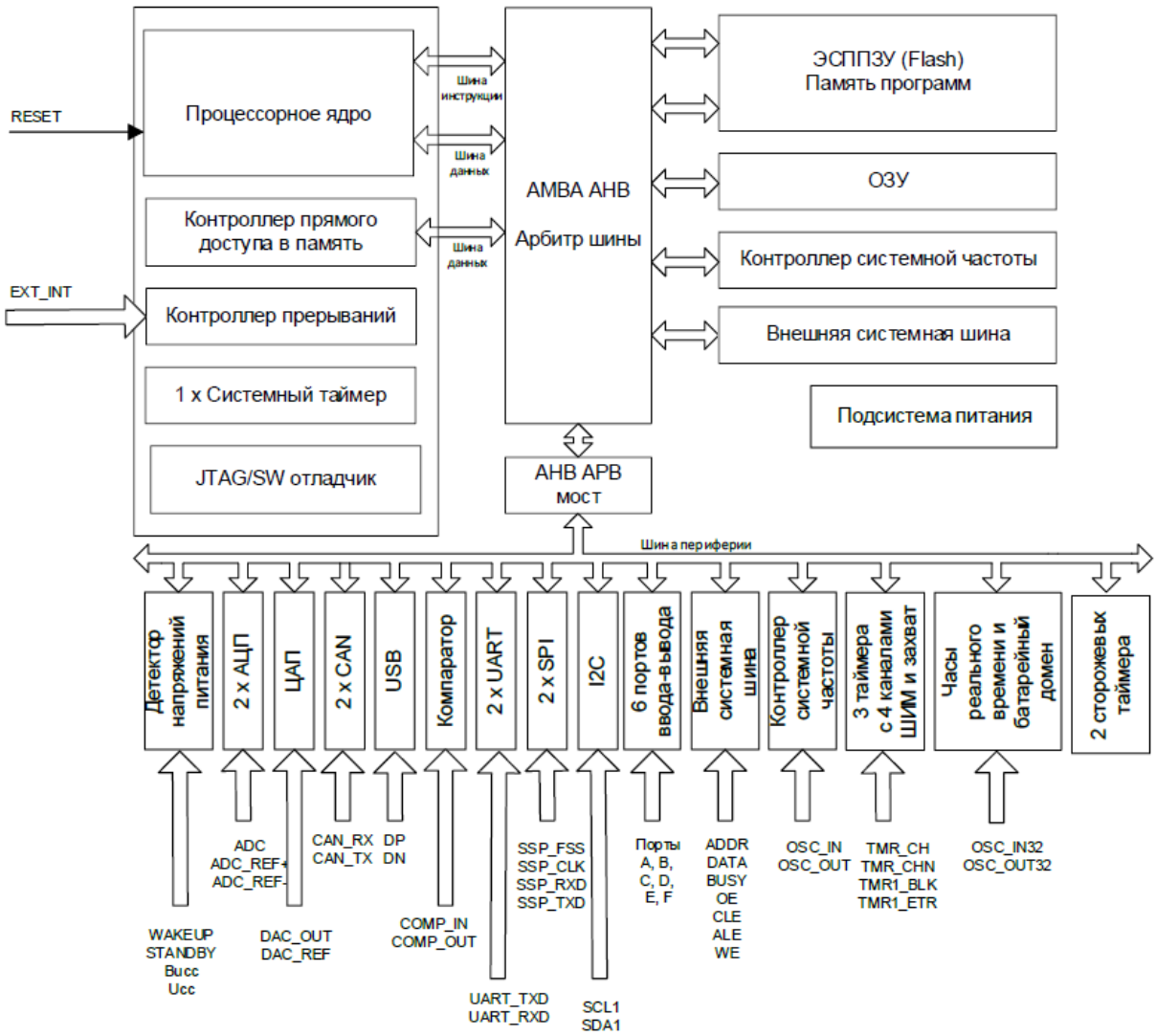


Рис. 1.5. Структурная схема МК К1986ВЕ92F1I [11] (пояснения – в табл. 1.1)

Из рис. 1.1 – 1.5 и табл. 1.1 можно сделать следующие **базовые выводы.** В структуру современных МК всех классов входят:

- центральный процессор (ЦП);
- встроенная энергонезависимая (т. е. характеризуемая отсутствием потерь хранимого в ней кода при выключении питания) память программ;
- встроенная энергозависимая память данных;
- блоки синхронизации и сброса;
- блоки программирования (т. е. загрузки) модулей энергонезависимой памяти МК;
- минимум один порт ввода / вывода;
- минимум один таймер общего назначения.

Таблица 1.1

Состав и назначение основных подсистем типовых МК общего назначения различных классов

Подсистема МК [1)]	**Назначение**	**Функциональные блоки, относящиеся к данной подсистеме, в МК:**				
		PIC16F505 **(рис. 1.1)**	**КР1878ВЕ1 (рис. 1.2)**	***ATmega128 /*** **1887ВЕ7Т (рис. 1.3)**	***STM32F10xx*** **(рис. 1.4)**	**К1986ВЕ92F1I (рис. 1.5)**
1	2	3	4	5	6	7
Центральный процессор, ЦП, ядро (*Central Processing Unit, Core*)	Обработка данных и управление другими функциональными блоками МК, в соответствии с последовательностью команд, хранимой в памяти программ	*Instruction Reg; Program Counter; Instruction Decode and Control; ALU; W (Working) Reg; Status Reg*	Центральный процессор	Ядро	*Cortex-M3 CPU*	Процессорное ядро
Энергонезависимая память программ (*Flash Program Memory*)	Хранение программ обработки данных и управления МК	*Flash Program Memory*	ЭСППЗУ команд	Память программ	*Flash*	ЭСППЗУ (*Flash*) Память программ
Энергонезависимая память данных (*EEPROM*)	Хранение данных, потеря которых при выключении питания недопустима (например, констант или уставок)	Отсутствует	ЭСППЗУ данных	Энергонеза-висимая память данных	*Flash*[2)]	ЭСППЗУ (*Flash*)[2)]

Продолжение таблицы 1.1

1	2	3	4	5	6	7
Энергозависимая память данных (*RAM*)	Хранение данных в процессе работы МК	*RAM*	ОЗУ данных	Статическое ОЗУ	*SRAM*	ОЗУ
Стек (*Stack*)	Хранение адресов возврата из подпрограмм, в т. ч. подпрограмм обслуживания прерываний	*Stack 1* *Stack 2*	Стек команд, стек данных	Формируется программно в ОЗУ		
Резервная память данных (*Backup Memory*)	Сохранение данных при сбоях по питанию или в энергосберегающем режиме	Отсутствует			*Backup reg*	Батарейный домен
Память периферийных устройств (*I/O Registers, Special Function Registers*)	Хранение кодов конфигурации ПУ, а также данных, назначение которых определяются типом ПУ (например, результатов преобразования АЦП, передаваемых или принятых данных блоков интерфейса и т. п.)	Регистры или / и модуля памяти, входящие в состав периферийных устройств; на рис. 1.1 – 1.5 не показаны; будут рассмотрены в разделах, посвященных архитектуре соответствующих классов периферийных устройств. Формально память периферийных устройств является частью памяти данных				

Продолжение таблицы 1.1

1	2	3	4	5	6	7
Подсистема питания (*Power, Power Management, Supply Supervision*)	Формирование и контроль напряжений питания МК	На рис. 1.1 – 1.3 в явном виде не показана			*POWER; SUPPLY SUPERVISION*	Подсистема питания
Подсистема синхронизации (*Oscillators and Clocks*)	Генерация сигналов тактирования и их распределение между узлами и блоками МК с целью обеспечения их синхронной работы	*Timing Generation; Internal RC OSC*	Тактовый генератор	Блок синхрониза-ции	*RC 8 MHz; XTAL OSC 4 – 16 MHz; RC 40 kHz; XTAL 32 kHz; PLL & CLOCK MANAGT*	Контроллер системной частоты
Подсистема сброса (*RESET*)	Установка всех функциональных узлов и блоков МК в некоторое определенное состояние при включении питания, после сбоев по питанию или при перезагрузке МК	*Device Reset Timer; Power-on Reset; Watchdog Timer*	Пуск + сторожевой таймер	Блок синхрониза-ции и управления, сторожевой таймер	*POR / PDR (Power-On Reset / Power-Down Reset); IWDG (Independent Watchdog): WWDG (Window Watchdog)*	*RESET*, детектор напряжения питания, сторожевые таймеры

Продолжение таблицы 1.1

1	2	3	4	5	6	7
Подсистема внутренних шин (магистралей) (*Bus System*)	Обмен информацией (адресами, командами, данными) между функциональными блоками МК	*Program Bus; Data Bus; Control Bus*[3)]	Общая шина МК, шины ЦП[3)], шина сигналов управления[3)]	Шина данных, шины ядра[3)], шина сигналов управления[3)]	См. рис. 1.7	См. рис. 1.6
Подсистема программирования энергонезависимой памяти (*Non-Volatile Memory Programming*)	Загрузка программного кода в память команд; загрузка энергонезависимой памяти данных (при ее наличии)	*PORT B* (послед-я загрузка кода)	Генератор напряжения программ-я, схема поддержки программ-я, порт *B* (загрузка кода)	Порты *B* и *D* (параллель-я загрузка кода), блок *SPI* (послед-я загрузка кода), контроллер *JTAG-TAP* + логика программ-я	Отладчик *SW/JTAG* (*Serial Wire / JTAG*)	Отладчик *JTAG/SW*, блок *UART2* (загрузка кода)
				Возможно самопрограммирование (см. раздел 12)		
Подсистема тестирования и отладки ПО (*Debug Module*)	Проверка корректности работы (в том числе пошаговая) и отладка ПО, загруженного в память команд	Отсутствует		Контроллер *JTAG-TAP (Trace Access Port)* + блок отладки	Отладчик *SW/JTAG* + *TPIU (Trace Port Interface Unit)* + *Trace Controller*	Отладчик *JTAG/SW*

Продолжение таблицы 1.1

1	2	3	4	5	6	7
Подсистема обслуживания прерываний (*Interrupt Controller*)	Управление обслуживанием запросов на прерывания от периферийных устройств (ПУ) или ядра МК, требующих приостановки выполнения основной программы и выполнения подпрограммы обслуживания запроса	Отсутствует	На рис. 1.1 не показана, входит в состав ЦП	Блок прерываний	*NVIC (Nested Vectored Interrupt Controller)*	Контроллер прерываний
Подсистема управления прямым доступом к памяти (ПДП) (*DMA Controller*)	Управление обменом данными между памятью и ПУ, или между ПУ, без участия ЦП в процедуре обмена (включая обслуживание запросов на обмен)	Отсутствует			*GP (General Purpose) DMA*	Контроллер прямого доступа к памяти

Продолжение таблицы 1.1

1	2	3	4	5	6	7
Порты ввода / вывода (*I/O Ports*)	Взаимодействие МК с внешними по отношению к нему устройствами[4)]	*PORT B, PORT C*	Порты *A* и *B*	Порты *A* - *G*	*GPIOA* - *GPIOE*	Порты *A* - *F*
Подсистема аналого-цифрового интерфейса (*Analog Front-End*)	Преобразование в цифровой код входных аналоговых сигналов системы контроля и управления на базе МК (например, сигналов датчиков)	Отсутствует		АЦП	*ADC1, ADC2*	АЦП1, АЦП2
	Сравнение с уставками входных аналоговых сигналов системы на базе МК и формирование цифровых сигналов оповещения или управления по результатам сравнения			Аналоговый компаратор	*Analog Watchdog* (входит в состав *ADC1* и *ADC2*)	Аналоговый компаратор

Продолжение таблицы 1.1

1	2	3	4	5	6	7
	Преобразование цифровых кодов в аналоговые сигналы управления, оповещения или уставок[5)]			Собственно ЦАП отсутствуют[5)]		ЦАП
Таймеры общего назначения (*General-Purpose Timers*)	Формирование интервалов времени между событиями (например, включением и выключением исполнительных устройств); генерация импульсов с программно-управляемыми частотно-временными параметрами (в том числе ШИМ- и ЧИМ-сигналов); цифровое измерение частоты, периода и длительности сигнала	*Timer 0*	16-битовый таймер	Таймеры / счетчики (0-й, 1-й, 2-й и 3-й)	*TIM1 – TIM4; RTC (Real-Time Clock)*	Таймеры 1-й, 2-й и 3-й, часы реального времени

Продолжение таблицы 1.1

1	2	3	4	5	6	7
Подсистема стандартного цифрового интерфейса (*Digital Communication Interface Modules*)	Обмен цифровыми данными между МК и внешними по отношению к нему устройствами в соответствии с некоторым общепринятым стандартом / протоколом (*UART, USART, SPI, I2C, CAN, USB* и др.)	Отсутствует[6)]		Блоки *USART, SPI, TWI (*синоним *I2C)*	Блоки *USART, SPI, I2C, CAN, USB*	Блоки *UART, SPI, I2C, CAN, USB*
Подсистема интерфейса с внешней памятью (*External Memory Interface*)	Управление обменом информацией с внешними по отношению к МК модулями памяти и/или периферийными устройствами	Отсутствует[7)]		Интерфейс внешней памяти (на рис. 1.3 явно не показан)	Отсутствует[7)] См. также[8)]	Внешняя системная шина

Окончание таблицы 1.1

1	2	3	4	5	6	7
Примечания. [1)] Описание структуры и архитектуры данных подсистем и относящихся к ним функциональных блоков МК представлено в соответствующих разделах. [2)] В качестве энергонезависимой памяти данных МК семейства *ARM Cortex-Mx* обычно используется некоторая не занятая под программный код область ЭСППЗУ программ, выделяемая разработчиком ПО МК. [3)] На рис. 1.1 – 1.3 в явном виде не показаны, см. текст далее. [4)] Основная функция каждого из выводов портов – цифровой вход / выход общего назначения. Кроме нее, каждый из выводов может выполнять одну или несколько закрепленных за ним (согласно *datasheet*) **альтернативных** функций – входов или выходов каких-либо функциональных блоков МК. Настройка каждого из выводов порта на выполнение основной функции (в том числе на ввод или на вывод) или какой-либо из альтернативных осуществляется пользовательским ПО МК. Подробности – см. раздел 6. В частности, в качестве изображенных на рис. 1.1 – 1.5 входов / выходов периферийных устройств реально служат выводы портов. [5)] Преобразование цифровых кодов в аналоговые постоянные или низкочастотные (до единиц кГц) сигналы может быть реализовано на базе таймеров, способом широтно-импульсной модуляции (см. раздел 9). [6)] Обмен данными по соответствующим протоколам может быть организован программно, с использованием портов ввода / вывода в качестве приемопередатчиков. [7)] Данный интерфейс может быть реализован программно, с использованием портов ввода / вывода для передачи адресов и приема / передачи данных. [8)] В структуру некоторых МК модельного ряда *STM32F10xxx* входит блок *FSMC* (*Flexible Static Memory Controller*). См., например, рис. 1.7.						

МК классов «*mainstream*» и «*high performance*», как правило, также содержат:

- энергонезависимые или питаемые от резервных источников модули памяти, предназначенные для хранения данных, потеря которых при выключении или сбоях питания, а также при переходах в энергосберегающий режим недопустима;
- контроллер прерываний;
- блоки аналого-цифрового и стандартного цифрового интерфейса, отсутствующие у большинства моделей МК класса «*cost-sensitive*»;
- контроллер / интерфейс внешней памяти;
- блоки тестирования (проверки правильности работы) и отладки (коррекции) загруженного программного кода.

Большинство моделей МК класса «*high performance*» также содержат:

- контроллеры прямого доступа к памяти;
- блоки ЦАП.

В целом, представленные на рис. 1.1 – 1.5 структурные схемы являются типовыми для большинства серийно выпускаемых в настоящее время МК общего назначения соответствующих классов. МК каждого из классов содержат набор функциональных блоков, которые потенциально могут потребоваться для решения прикладных задач в областях применения МК соответствующего класса или семейства. Конфигурация / настройка функциональных блоков МК под решение некоторой конкретной задачи осуществляется посредством ПО МК.

Интересно отметить, что, как следует из рис. 1.1 – 1.5 и из табл. 1.1, для возможности практического использования МК необходим, как минимум, следующий состав функциональных блоков / подсистем:

- ЦП;
- память программ;
- память данных;
- подсистема питания;
- подсистема синхронизации;
- подсистема сброса;

- порты ввода / вывода (ПВВ);
- подсистема программирования памяти.

Именно таков состав функциональных блоков / подсистем наиболее простых МК класса «*cost-sensitive*» (см., например, рис. 1.1 и 1.2), за исключением того, что в их структуре имеется еще, как минимум, один таймер.

Функциональные блоки / подсистемы МК, не входящие в состав вышеперечисленных, с формальной точки зрения, только позволяют реализовать функции контроля и управления техническими объектами с б**о**льшим быстродействием, меньшими затратами памяти и при минимуме внешних (по отношению к МК) узлов и блоков. Однако, на практике реализация, например, частотно-временных функций или стандартного цифрового интерфейса посредством только вышеперечисленных подсистем / функциональных блоков хотя и возможна, но весьма нерациональна. Поэтому для решения каждой конкретной задачи следует выбирать модель МК, в структуру которой входят все необходимые узлы и блоки, реализованные на аппаратном уровне. В свою очередь, модели МК, содержащие только вышеперечисленный минимальный состав функциональных блоков / подсистем, применимы только для решения простейших задач контроля и управления.

Типовые структурно-архитектурные решения подсистем и функциональных блоков МК, перечисленных в табл. 1.1, будут описаны в последующих разделах. В данном разделе необходимо подробнее остановиться только на подсистеме внутренних шин (магистралей) МК, выполняющих функцию физического и информационного **объединения** всех подсистем МК.

1.3 Подсистема внутренних шин МК

В МК класса «*cost-sensitive*» и в большинстве семейств класса «*mainstream*» обмен данными между всеми функциональными узлами и блоками МК осуществляется по одной общей шине данных МК (см. рис. 1.1 – 1.3). Кроме нее, в структуре МК указанных классов присутствуют не показанные на рис. 1.1 – 1.3 в явном виде

шины ЦП и шины управляющих сигналов. К первым из названных относятся:

- шина выборки команд из памяти программ;
- шина адреса памяти программ;
- шина адреса памяти данных.

Их назначение очевидно и в комментариях не нуждается.

Шина управляющих сигналов представляет собой набор линий, по которым от дешифратора команд, входящего в состав ЦП (см., например, рис. 1.3), передаются сигналы управления узлами и блоками МК.

Несколько сложнее организация подсистемы внутренних шин МК класса «*high performance*», что обусловлено следующими факторами:

- более развитым составом модулей памяти;
- более разнообразным составом периферийных устройств, существенно различающихся между собой по быстродействию;
- наличием подсистемы ПДП (*DMA*), для реализации которого необходима дополнительная шина обмена данными между памятью и периферийными устройствами.

Ввиду вышеперечисленных факторов, подсистема внутренних шин МК класса «*high performance*» организуется по принципу мультишинной архитектуры. Ее типовым примером является стандарт *AMBA* [12] (*Advanced Microcontroller Bus Architecture*, Усовершенствованная архитектура шины микроконтроллера), по которому, например, реализуется подсистема внутренних шин МК семейства *ARM Cortex-Mx* [9]. В рамках данного стандарта существует несколько спецификаций интерфейса [12]. Архитектура подсемейств от *ARM Cortex-M4* и ниже, в том числе *ARM Cortex-M3*, использует две из них (см. рис. 1.4 и 1.5):

- *AHB* (*Advanced High-performance Bus*, Усовершенствованная высокопроизводительная шина);
- *APB* (*Advanced Peripheral Bus,* Усовершенствованная шина периферийных устройств).

Соответственно, совокупность подсистем / функциональных блоков МК, связь с которыми осуществляется по спецификации *AHB*, называют *AHB*-доменом, а по спецификации *APB* – *APB*-доменом.

Обе данные спецификации характеризуются:

- организацией обмена данными по принципу «ведущий - ведомый» (в системе и *AHB* -, и *APB*-интерфейса имеется как минимум один абонент, за которым закреплен статус ведущего);

- передачей адреса и приемом / передачей данных в параллельном формате, в синхронном режиме;

- наличием физически разделенных магистралей адреса и данных, разрядностью 32 бита каждая (у последних версий *AHB*- и *APB*-спецификаций разрядность может быть больше).

Спецификация *AHB* применяется для обмена данными между наиболее скоростными подсистемами и функциональными блоками МК. Ее основными отличительными особенностями являются:

- возможностью наличия нескольких ведущих (что имеет место в МК семейства *ARM Cortex-Mx*);

- построение *AHB*-домена по топологии **коммутируемой матрицы**;

- возможность обмена данными в **пакетном** режиме.

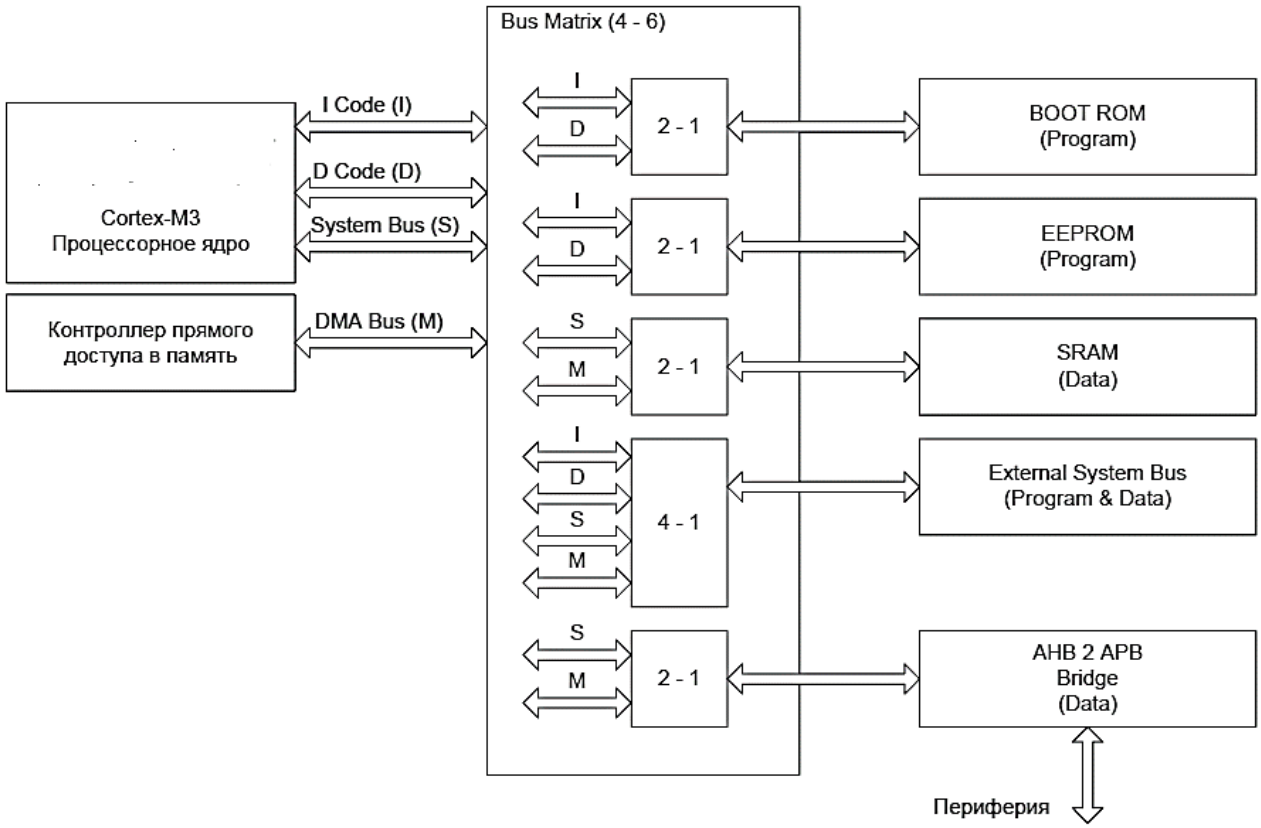
В свою очередь, спецификация APB используется для относительно низкоскоростного обмена данными с периферийными устройствами. Он характеризуется:

- наличием только одного ведущего;

- отсутствием режима пакетной передачи.

На рис. 1.6 представлена упрощенная структурная схема *AHB*-домена МК К1986ВЕ92F1I [11]. Процессорное ядро и контроллер прямого доступа к памяти являются ведущими устройствами, а функциональные блоки, расположенные в крайне правом ряду – ведомыми. Двунаправленными фигурными стрелками обозначены *AHB*-шины, содержащие предусмотренные спецификацией *AHB* линии адреса, данных, управляющих сигналов, синхронизации и т. д. По *AHB*-шинам, обозначенным на рис. 1.6 буквами *I*, *D*, *S* и *M*, осуществляется связь с ведущими устройствами; каждая из данных шин выделена под определенную функцию (см. подрисуночные подписи). *AHB*-шины ведомых устройств являются многофункциональными, т. е. могут подключаться к различным *AHB*-шинам ведущих устройств, в зависимости от выполняемых в конкретный момент операций. Переключение шин осуществляется

коммутатором матричного типа, обозначенным на рис. 1.6 как *Bus Matrix* и управляемым арбитром шин (на рис. 1.6 не показан, см. рис. 1.5). Структура коммутатора представлена на рис. 1.6 и в комментариях не нуждается.



I Code (I) – шина выборки команд
D Code (D) – шина выборки данных, расположенных в памяти программ
System Bus (S) – шина связи ЦП с памятью данных и с периферийными устройствами
DMA Bus (M) – шина ПДП
BOOT ROM – загрузочный сектор (см. далее рис. 2.26)
AHB 2 APB Bridge – мост между *AHB*- и *APB*-доменами

Рис. 1.6. Упрощенная структурная схема *AHB*-домена МК К1986ВЕ92F1I [11]

Мультишинная организация *AHB*-домена позволяет обеспечить максимальную общую производительность МК, в том числе выполнять параллельно во времени операции:

- выборки команд и выборки данных из памяти программ;
- обращения к памяти программ и к памяти данных;

- записи / чтения в режиме ПДП и обращения к памяти программ (заметим, что в МК ранних поколений при выполнении операций ПДП происходил захват внутренней шины МК контроллером ПДП, и выполнение основной программы фактически останавливалось);

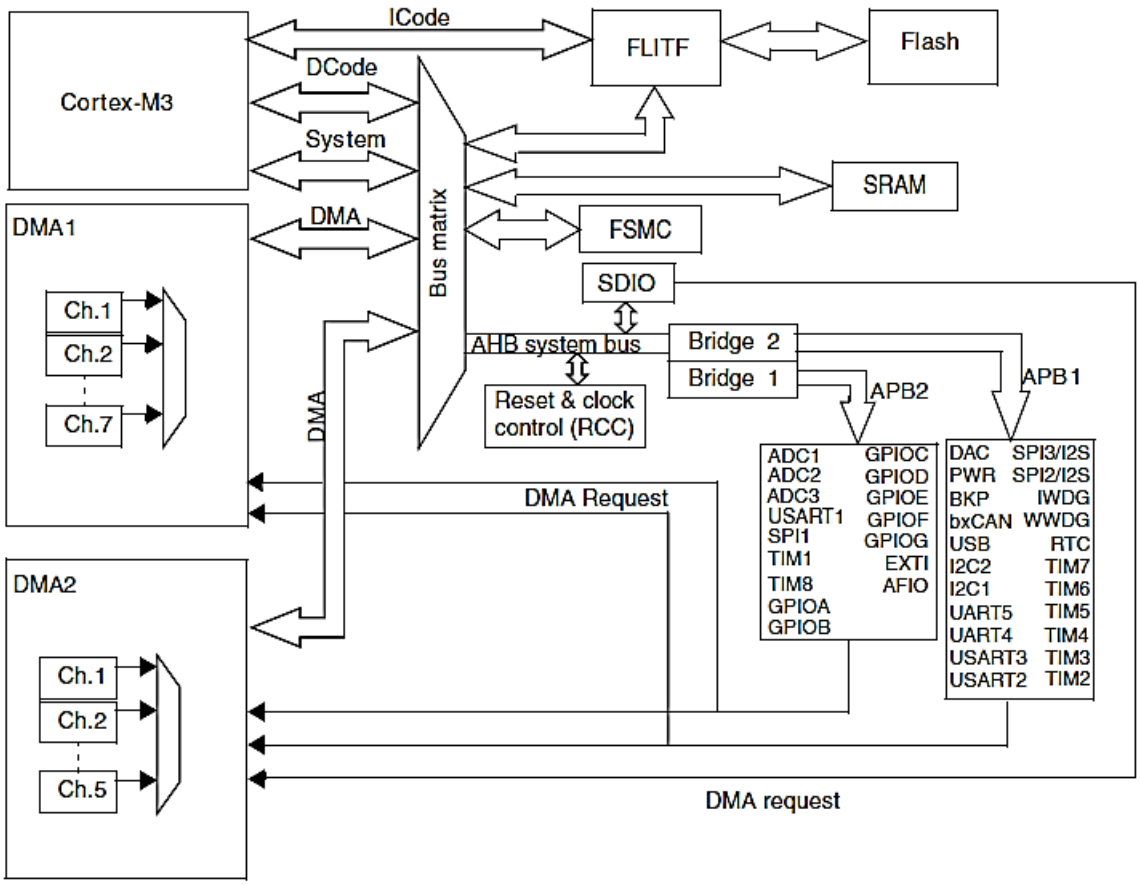
и т. п., при минимизации конфликтов между функциональными блоками / подсистемами из-за доступа к внутренним шинам МК

APB-домен формируется объединением одноименных линий *APB*-шин всех периферийных устройств, входящих в домен, в шину периферии (см. рис. 1.6). Она подключается к домену *AHB* через устройство, называемое **мостом** (*bridge*), обеспечивающее корректное информационное взаимодействие *AHB*- и *APB*-доменов. При этом в первом из них мост является одним из ведомых устройств, а во втором – ведущим.

Необходимо отметить, что во многих подсемействах / моделях МК семейства *ARM Cortex-Mx* домен *APB* разделяется на сегменты. В каждый из них группируются устройства с различными требованиями к быстродействию. В качестве примера на рис. 1.7 представлена схема организации внутренних шин МК модельного ряда *STM32F10xxx* [13], относящихся, как и К1986ВЕ92F1I, к подсемейству *ARM Cortex-M3*. В них домен *APB* разделен на два сегмента, *APB1* и *APB2*. При этом максимальная тактовая частота первого из них равна 36 МГц, а второго – 72 МГц (совпадает с максимальной частотой тактирования ЦП). Разделение домена *APB* на сегменты, в частности, позволяет выбирать частоты их тактирования независимо одну от другой, обеспечивая при этом рациональное сочетание производительности и энергопотребления (напрямую зависящего от частоты, на которой работает тот или иной блок, см. пункт 3.4.1). В ряде моделей МК семейства *ARM Cortex-Mx*, например, *STM32F405xx/07xx*, сегментирован также *AHB*-домен (см., например, [14]). Интересно отметить, что в МК модельного ряда *STM32F030xx* подсемейства *ARM Cortex-M0 AHB*-домен сегментирован, а домен *APB* – не сегментирован [15].

В целом, можно сказать, что внутренняя структура МК семейства *ARM Cortex-Mx* организована по принципу структурированной и сегментированной **микролокальной**

вычислительной сети. То же справедливо и для ряда других семейств МК класса «*high performance*».



FLITF – Flash Memory Interface
FSMC - Flexible static memory controller

Рис. 1.7. Схема организации внутренних шин МК модельного ряда *STM32F10xxx* [13]

К описанному выше принципу организации подсистемы внутренних шин МК семейства *ARM Cortex-Mx* потребуется неоднократно возвращаться в дальнейшем, при рассмотрении структурно-архитектурных решений подсистем и базовых функциональных блоков МК данного семейства (см., например, раздел 4).

1.4 Общие принципы разработки устройств на базе МК

1.4.1. Разработка некоторого устройства на базе МК сводится к выполнению следующих основных этапов:

- выбор модели МК, приемлемой для решения конкретной задачи по составу функциональных блоков, электрическим и частотно-временным параметрам, а также с организационно-экономической точки зрения;

- определение состава блоков МК, которые необходимо задействовать для решения поставленной задачи;

- выбор / разработка схемы подключения МК к объекту контроля и управления;

- разработка ПО МК, обеспечивающего конфигурирование, функционирование и взамодействие используемых блоков МК в соответствии с решаемой задачей;

- экспериментальная проверка и корректировка (при необходимости) разработанной схемы и ПО.

1.4.2. В несколько неформальном стиле изложим **главное положение**, которым необходимо руководствоваться при разработке ПО МК. Он (как и любое средство вычислительной техники, включая самые мощные суперкомпьтеры) подобен добросовестному, исполнительному, дисциплинированному, расторопному, но предельно тупому работнику. Поэтому для корректного выполнения им возлагаемых на него функций ему детально, четко и однозначно необходимо указать:

- режимы работы и **все** параметры конфигурации **всех** функциональных блоков, используемых при решении конкретной задачи;

- порядок действий в процессе ее выполнения.

1.4.3. Исходя из вышесказанного, ПО МК должно включать в себя:

- процедуры конфигурирования используемых функциональных узлов и блоков МК, которые могут оформляться как отдельные программные модули или включаться в основную программу;

- основной программный модуль (при необходимости – с подпрограммами, например, обслуживающими запросы на прерывания), реализующий алгоритм работы МК.

1.4.4. Режимы работы и параметры конфигурации функциональных блоков МК задаются посредством входящих в их состав программно-доступных регистров конфигурации. Если

возможны несколько режимов работы какого-либо функционального блока МК (например, порта ввода / вывода, см. раздел 6), в некоторых его регистрах **обязательно** имеются битовые поля, задающие режим работы. Естественно, если он не задан или задан некорректно, соответствующий блок не будет функционировать так, как требуется.

1.4.5. В свою очередь, порядок действий МК в процессе его работы задается кодом его основной программы и входящих в ее состав подпрограмм (например, подпрограмм обработки запросов на прерывания, см. раздел 7). При разработке данных кодов необходимо руководствоваться общепринятыми правилами разработки ПО. Некорректный алгоритм работы ПО, естественно, приведет к некорректному функционированию МК (даже в отсутствие формальных ошибок в коде программы). Основными специфическими особенностями, характерными для ПО МК, являются следующие:

- в качестве переменных может выступать содержимое регистров данных, входящих в состав функциональных блоков МК; поэтому для корректной работы ПО необходимо правильное указание их имен и учет их форматов (например, к непредсказуемым последствиям может привести присвоение значению переменной, объявленной как 8-битовая, содержимого некоторого 16-битового регистра данных);

- в качестве условий выполнения / не выполнения каких-либо действий могут выступать значения битов состояния функциональных блоков МК; в каждом из них имеется минимум один регистр состояния (статуса), значения битов которого отражают **все** события (см. подраздел 7.1), имевшие место в процессе работы блока;

- если условием генерации некоторого запроса на **прерывание** от какого-либо блока МК может служить одно из нескольких **событий** (см. подраздел 7.1), архитектура МК **всегда** позволяет проверить, какое именно, по значениям битов регистра (регистров) состояния соответствующего блока (см., например, рис. 5.2 и пояснения к нему).

Вышеперечисленные особенности в обязательном порядке должны учитываться при разработке ПО МК.

1.4.6. Для успешного выполнения перечисленных в пункте 1.4.1 этапов (с учетом пунктов 1.4.2 – 1.4.5) в первую очередь, необходима информация о следующих компонентах структуры и архитектуры МК:

- общих характеристиках архитектуры ЦП (семействе / подсемействе, разрядности, максимальной тактовой частоте и т. п.);
- составе, организации и объеме памяти;
- составе, структуре, функциональных возможностях и характеристиках периферийных устройств (портов, таймеров, блоков интерфейса и т. п.);
- составе и форматах программно-доступных регистров периферийных устройств, планируемых для применения, протоколах и алгоритмах их конфигурирования и обмена информацией с ними.

Вышеперечисленные компоненты структуры и архитектуры МК подлежат первоочередному изучению разработчиком в процессе проектирования. Им же будет уделяться основное внимание при дальнейшем изложении материала.

1.4.7. Следует также вкратце остановиться на источниках информации, необходимой разработчику электронных средств на базе МК. Полная информация по структуре и архитектуре МК классов «*cost-sensitive*» и «*mainstream*» обычно содержится в одном электронном документе. Отечественные производители называют данный документ руководством пользователя или техническим описанием, зарубежные – обычно «*Data Sheet*». Необходимая разработчику информация по структуре и архитектуре МК класса «*high performance*» зарубежных производителей обычно содержится в 3-х основных электронных документах:

- *Data Sheet*, включающем в себя структурную схему БИС МК, данные по назначению ее выводов (пинов), в том числе по закрепленным за ними альтернативным функциям, а также электрические, частотно-временные и механические параметры и характеристики БИС МК;

- *Reference Manual*, в котором содержатся необходимые разработчику сведения по архитектуре МК и его функциональных блоков (обычно – кроме архитектуры ЦП и системы команд);

- *Programming Manual*, содержащий описание архитектуры ЦП и системы команд.

При этом часто *Reference Manual* и *Programming Manual* бывают общими для ряда моделей БИС МК, принадлежащих к одному и тому же семейству.

Отечественные производители БИС МК часто объединяют всю необходимую разработчику информацию по структуре и архитектуре МК класса «*high performance*» в один электронный документ (см., например, [11]).

1.5 Выводы

1.5.1. Базовыми характеристиками каждой модели МК является ее структура (собственно структурная схема) и архитектура (модель МК с точки зрения разработчика электронных средств на его основе).

1.5.2. Серийно выпускаемые модели МК по степени сложности подразделяются на классы («*cost-sensitive*», «*mainstream*», «*high performance*»), по базовым компонентам архитектуры – на семейства (группы МК с одинаковой архитектурой ЦП, организацией памяти и системой команд). Типовые структурные схемы МК каждого из вышеперечисленных классов представлены на рис. 1.1 – 1.5, пояснения к ним – в табл. 1.1. МК каждого из классов содержат набор функциональных блоков, которые потенциально могут потребоваться для решения прикладных задач в областях применения МК соответствующего класса. Конфигурация / настройка функциональных блоков МК под решение некоторой конкретной задачи осуществляется посредством ПО МК.

1.5.3. Физическое и информационное объединение подсистем / функциональных блоков МК осуществляется посредством подсистемы внутренних шин. В относительно простых МК данная подсистема организуется по принципу общей шины (магистрали). В высокопроизводительных МК класса «*high performance*»,

отличающихся развитым составом модулей памяти и периферийных устройств, а также наличием режима ПДП, подсистема внутренних шин строится по принципу структурированной и сегментированной микролокальной вычислительной сети (см. рис. 1.6 и 1.7). Данный принцип позволяет обеспечить максимальную общую производительность МК, при минимизации конфликтов между функциональными блоками / подсистемами из-за доступа к внутренним шинам МК.

1.5.4. Основные этапы разработка устройства на базе МК перечислены в пункте 1.4.1; базовые принципы и особенности разработки ПО МК – в пунктах 1.4.2 – 1.4.5; основные элементы структуры и архитектуры МК, знание которых необходимо при разработке устройств на МК, и источники получения информации о данных элементах – в пунктах 1.4.6 и 1.4.7.

1.5.5. В целом, МК одинаковых классов сложности всех распространенных семейств характеризуются **значительно бóльшим сходством, чем различием** базовых структурно-архитектурных решений. Поэтому представляется целесообразным дальнейшее изложение материала данного пособия построить как описание типовых структурно-архитектурных решений функциональных узлов и блоков МК, основ их программирования и применения на типовых примерах МК классов «*cost-sensitive*», «*mainstream*» и «*high performance*», в частности, МК, структурные схемы которых приведены на рис. 1.1 – 1.5. Основное внимание при этом будет уделяться структурно-архитектурным решениям МК наиболее широко распространенных в настоящее время семейств – *AVR* и *ARM Cortex-Mx* (см., например, рис. 1.3 – 1.5), в первую очередь – компонентам их структуры и архитектуры, перечисленным в пункте 1.4.6.

Структурно-архитектурные решения МК других семейств, в основном, сходны с описанными в данном пособии и, при необходимости, могут быть изучены читателем самостоятельно.