Лекция 4 Микроконтроллер

План курса «Встраиваемые микропроцессорные системы»:

Лекция 1: Введение. Язык программирования С

Лекция 2: Язык программирования С. Стандартная библиотека языка С

Лекция 3: Применение языка С для встраиваемых систем

Лекция 4: Микроконтроллер

Лекция 5: Этапы разработки микропроцессорных систем

Лекция 6: Разработка и отладка программ для встраиваемых систем

Лекция 7: Архитектура программного обеспечения для встраиваемых систем

Лекция 8: Периферийные модули: DMA, USB, Ethernet

Что внутри микроконтроллера

Центральный процессор (ядро) может занимать менее 10% от площади кристалла.

Остальную часть занимают:

- Постоянная память (Flash, EEPROM)
- Оперативная память (SRAM)
- Периферия (таймеры, интерфейсы и т.д.)
- Система тактирования (PLL)
- Регулятор напряжения и цепи управления питанием
- Внутренние шины
- Дополнительные цепи для тестирования на производстве

Архитектура центрального процессора

Архитектура с точки зрения разработчика процессора (микро-архитектура):

- Внутренние шины;
- Количество стадий конвейера;
- Время исполнения команд;
- и т.д.

Знание микро-архитектуры нужно разработчикам микроконтроллеров и разработчиками компиляторов.

Архитектура с точки зрения программиста:

- Набор команд (то, из чего состоят программы);
- Программно-логическая модель (набор регистров и карта памяти)

(то, что видят программы);

- Методы отладки (то, что видят отладчики);

Знание архитектуры нужно программистам на ассемблере и частично программистам на С.

Архитектура ARM

ARM не является производителем микросхем, а только разрабатывает дизайн микропроцессоров и лицензирует собственные технологии другим фирмам.

Лицензия может быть за готовое ядро или с правом внесения изменения в ядро.

Cortex-A (Application) — высокопроизводительные процессоры для мобильных вычислений, смартфонов, серверов и т. д. Эти процессоры работают на более высокой тактовой частоте (более 1 ГГц) и поддерживают блок управления памятью (MMU), который требуется для полнофункциональных ОС общего назначения, таких как Linux, Android, MS Windows и мобильные ОС.

Cortex-R (Real-Time) — высокопроизводительные процессоры для приложений реального времени, таких как контроллер жесткого диска или привод в электромобиле. Эти процессоры не имеют MMU и обычно имеют блок защиты памяти (MPU), кэш-память и другие функции памяти, предназначенные для промышленных приложений. Они могут работать на довольно высокой тактовой частоте (например, от 200 МГц до > 1 ГГц) и имеют очень низкую задержку отклика. Хотя эти процессоры не могут работать с полными версиями Linux или Windows, существует множество операционных систем реального времени (RTOS), которые можно использовать с этими процессорами.

Cortex-M (Microcontroller) — эти процессоры обычно разрабатываются так, чтобы иметь гораздо меньшую площадь кристалла и высокую энергоэффективность. Как правило, у них короткий конвейер и более низкая максимальная частота (хотя некоторые из этих процессоров работают на частоте более 200 МГц). В то же время семейство процессоров Cortex-M очень простое в использовании и они очень популярны на рынке микроконтроллеров.

Сравнение семейств Cortex-A, Cortex-R, Cortex-M

Cortex	A	R	M
Расшифровка	Application	Real-Time	Microcontroller
Частоты	> 1 ГГЦ	200 МГц – 1 ГГц	< 200 МГц
Производительность	Высокая	Средняя	Низкая
Время отклика	Среднее	Низкое	Среднее
Конвейер	Длинный	Средний	Короткий
Энергопотребление	80 мкВт/МГц	120 мкВт/МГц	8 мкВт/МГц
ОС	ОС общего назначения	ОСРВ или без ОС	ОСРВ или без ОС
Пример	Cortex-A7, Cortex-A55, Cortex-A78	Cortex-R7, Cortex-R52	Cortex-M0, Cortex-M3, Cortex-M7
Применение	Системы с высокой производительностью	Системы жесткого реального времени с высокими требованиями к надежности	Системы с низким энергопотреблением
Устройства	Смартфоны, ноутбуки, серверы, устройства с графическим интерфейсом	Электронные блоки управления, преобразователи частоты	Устройства управления, устройства с автономным питанием

Сравнение ядер Cortex-M

Cortex	M0	M0+	M1	M3	M4	M7	M23	M33
DMIPS/MHz	0.87	0.95	0.8	1.25	1.25	2.14	0.98	1.5
CoreMark/MHz	2.33	2.46	1.85	3.34	3.42	4.02	2.64	5.01
Архитектура	Принстонская	Принстонская	Принстонская	Гарвардская	Гарвардская	Гарвардская	Принстонская	Гарвардская
Стадии конвейера	3	2	3	3	3	6	2	3
Прерывания	1-32	1-32	1-32	1-240	1-240	1-240	1-240	1-480
Аппаратное деление	нет	нет	нет	да	да	да	да	да
ЦОС команды	нет	нет	нет	да	да	да	нет	да
Плавающая запятая	нет	нет	нет	нет	да	да	да	да
TrustZone	нет	нет	нет	нет	нет	нет	да	да
Блок защиты памяти (MPU)	нет	да	нет	да	да	да	да	да
Применение	Энергоэффективные МК	Энергоэффективные МК	Softcore для ПЛИС	МК общего назначение	МК общего назначение	Высокопроизводительные МК	Энергоэффективные МК	МК общего назначение

Архитектура ARM Cortex-M3

Все ARM Cortex-M являются 32 разрядным RISC (Reduced Instruction Set Computing) процессорами:

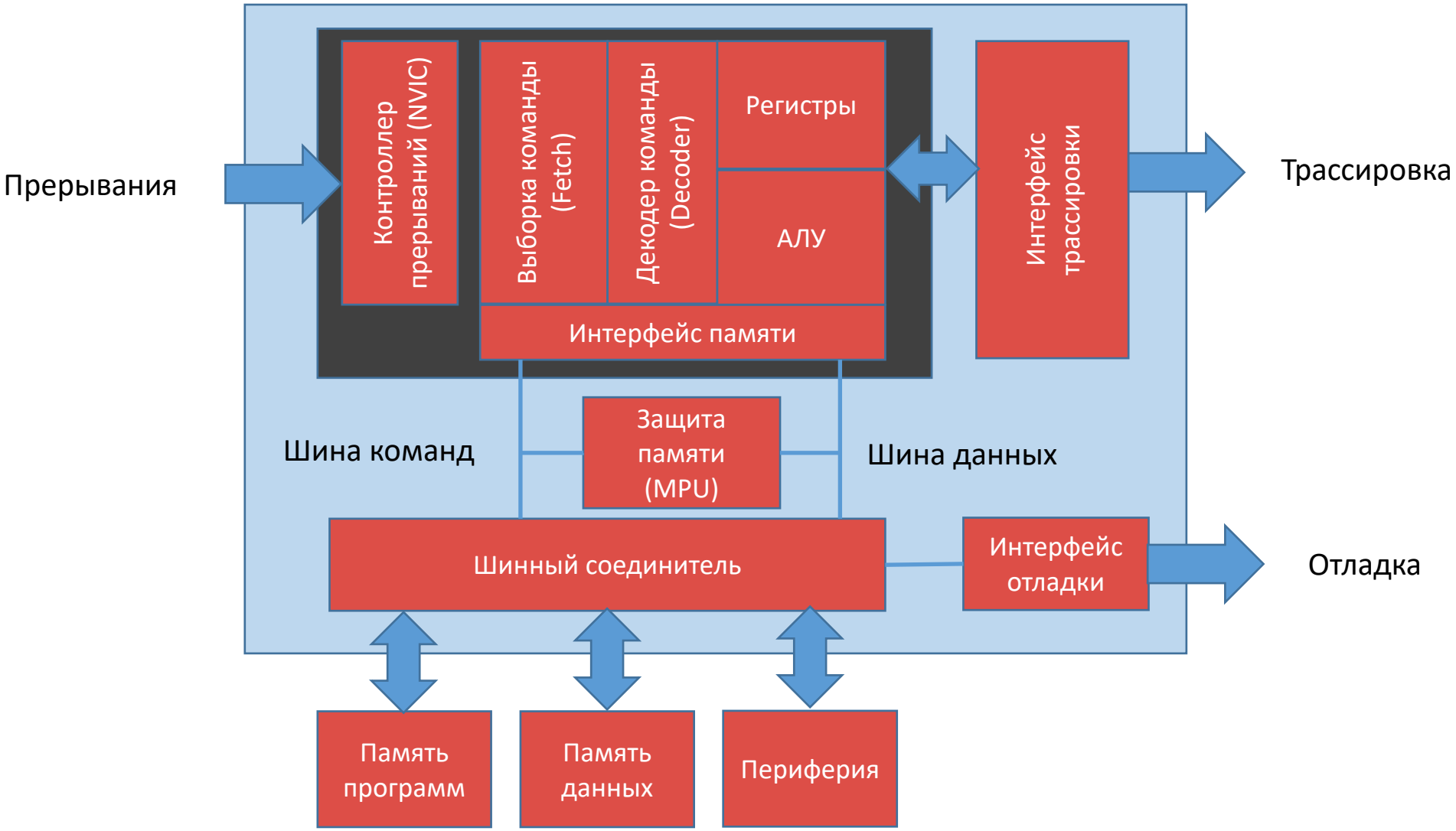
- 32 разрядные регистры;
- 32 разрядные внутренние магистрали;
- 32 разрядные шинные интерфейсы.

Микро-архитектура **ARM Cortex-M3,** в документах также называется **ARMv7-M.**

Особенности ARM Cortex-M3 (ARMv7-M):

- 3 ступенчатый конвейер (Fetch, Decode, Execute);
- набор команд Thumb-1, Thumb-2 (совместное использование 16-ти разрядных и 32-разрядных команд), аппаратное деление и команды с арифметики с насыщением;
- задержка выполнения прерывания (interrupt latency) – 12 тактов.

Структура ARM Cortex-M3

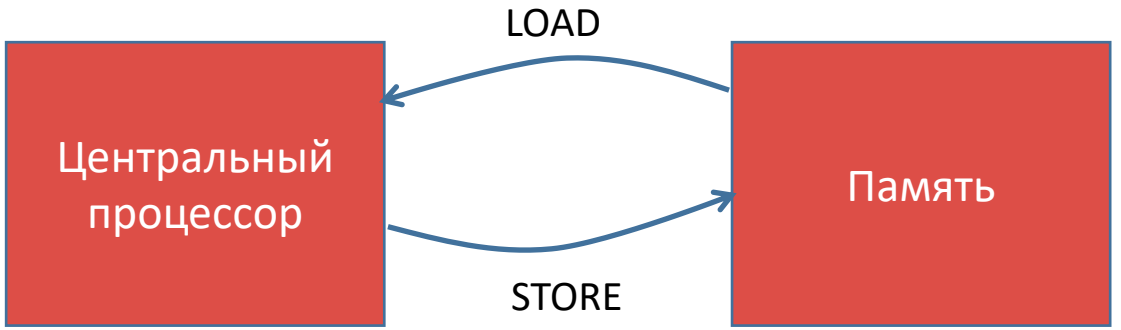


Архитектура Load-Store

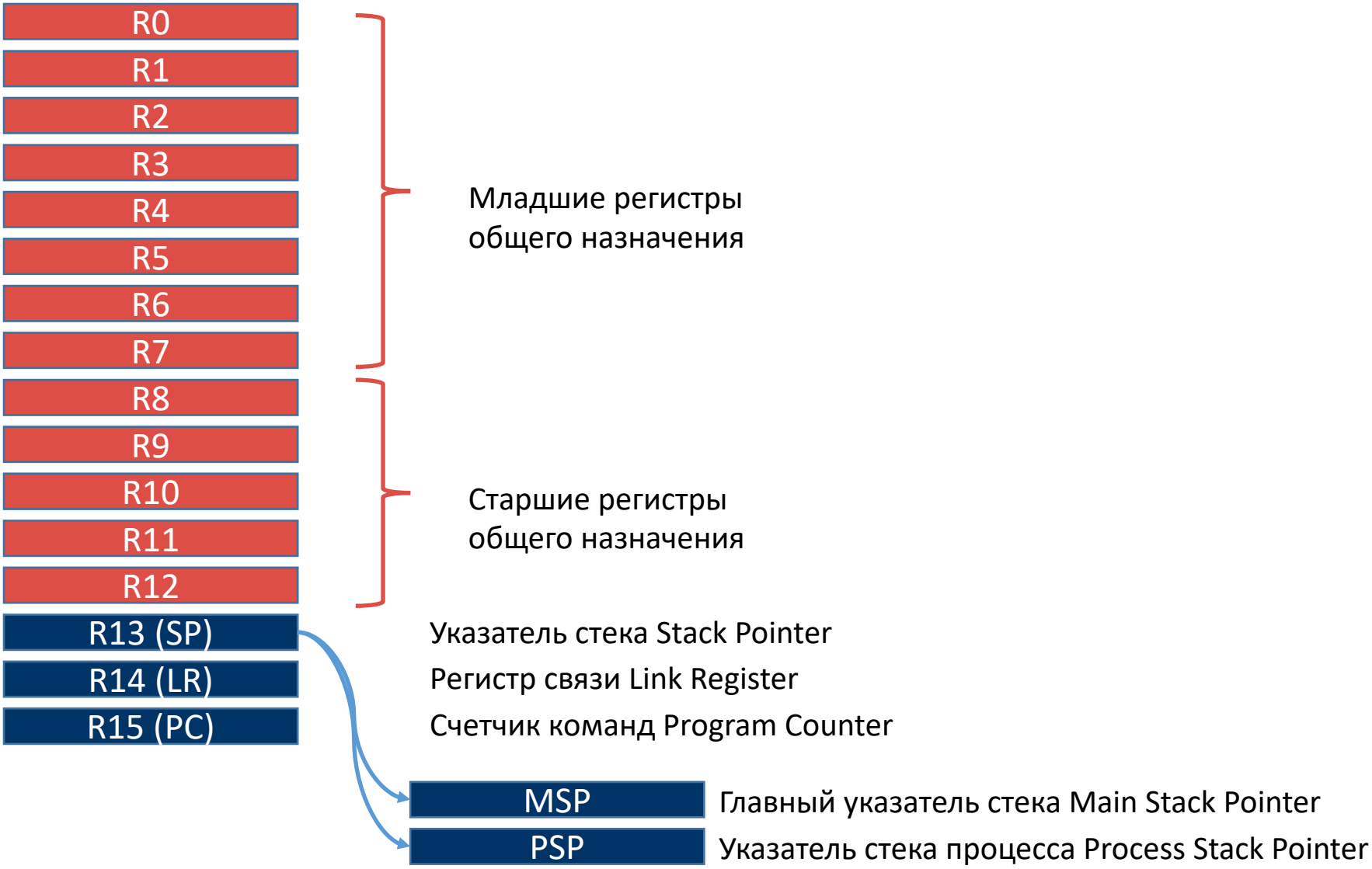
Некоторое количество регистров внутри центрального процессора требуется для выполнения операций. Если необходимо сделать какие-либо операции с данными, то данные необходимо вначале загрузить из памяти в регистр, провести обработку данных и сохранить результат в памяти (Load-Store Architecture).

Как правило, процессоры с архитектурой RISC (такие как ARM, MIPS, RISC-V) вообще не имеют команд для манипуляции данным прямо в памяти, только через загрузку в регистры центрального процессора.

Банк регистров центрального процессора содержит шестнадцать 32-х разрядных регистров. Из них 13 регистров – это регистры общего назначения (РОН), остальные – специальные регистры.

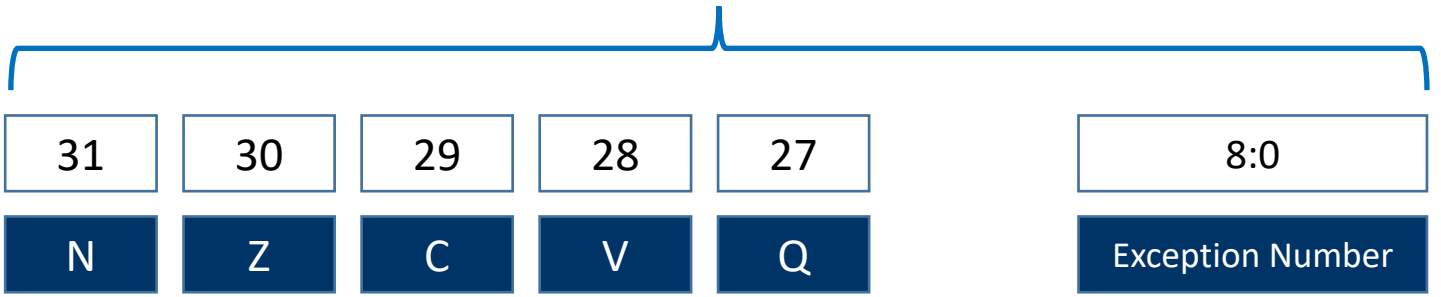


Программно-логическая модель ARM Cortex-M3



Программно-логическая модель ARM Cortex-M3

PRIMASK	Регистр – разрешения прерывания
FAULTMASK	Регистр разрешения исключения
BASEPRI	Регистр маски прерываний
CONTROL	Регистр управления процессором
xPSR	Регистр состояние программы



N – отрицательное число
Z – ноль
C – перенос при сложении беззнаковых чисел
V – переполнение при сложении чисел со знаком
Q – насыщение

Exception Number – номер текущего исключения/ прерывания

Команды ARM Cortex-M3

Команды перемещения данных:

```
MOV  R4, R0
```

Команды загрузки/выгрузки:

```
LDR  R0, [R1, #0x8]
STR  R0, [SP]
PUSH {R0-R5}
POP  {R1}
```

Арифметические, логические команды и команды сдвига:

```
ADD  R0, R0, R1
AND  R4, R4, R3
ASRS R2, R1, #2
REV  R0, R1
```

Команды перехода:

```
BL   func
B    loop
BEQ m1
BNE m2
```

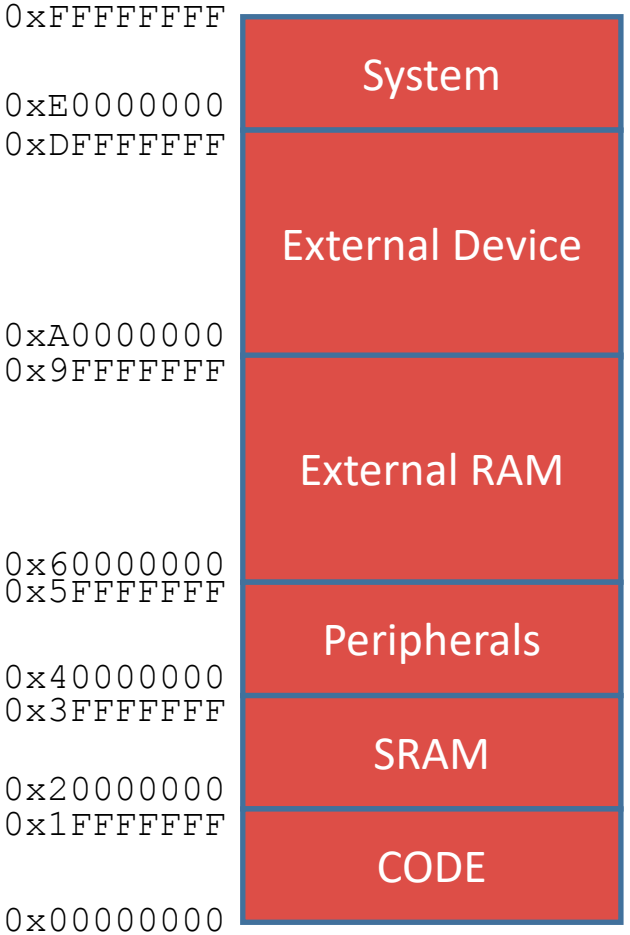
Специальные команды:

```
SEV
ISB
DSB
SVC #10
```

Карта распределения памяти ARM Cortex-M3

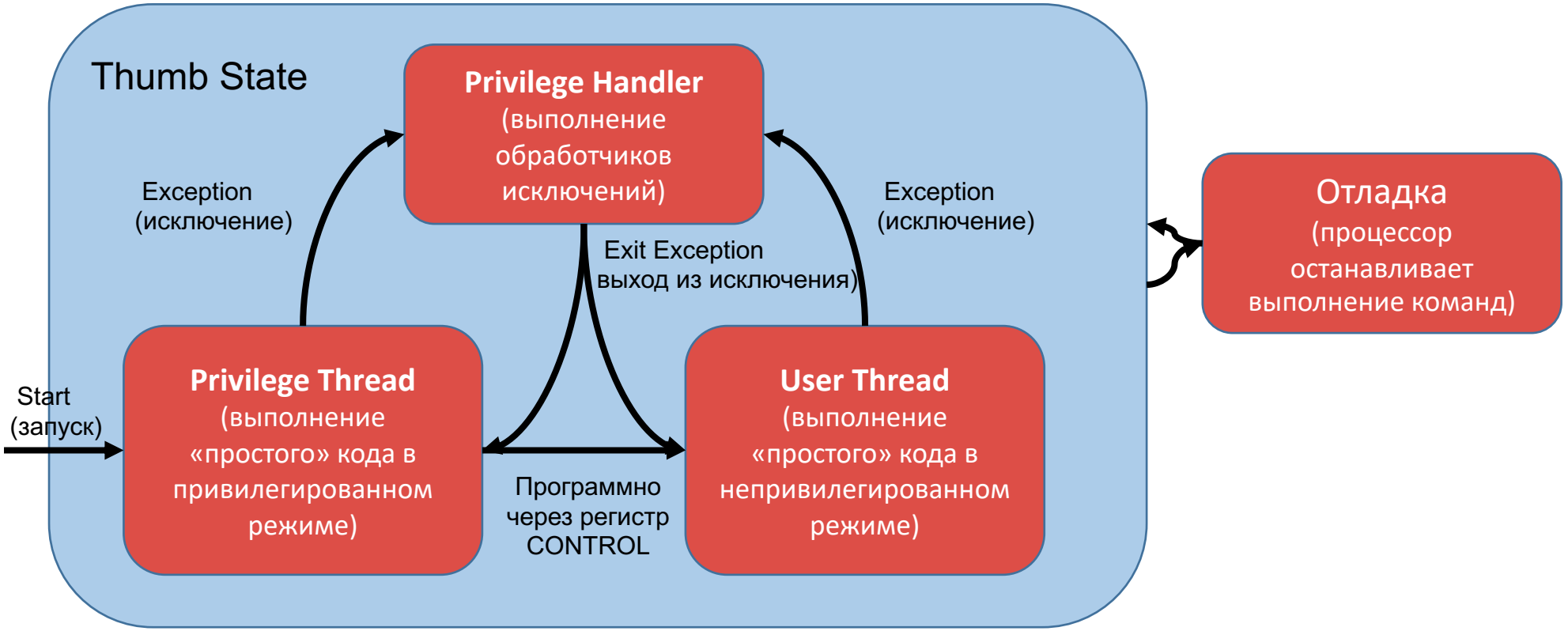
Особенности распределения памяти:

- 4 GB адресное пространство;
- Поддерживает Big-Endian и Little Endian;
- Память разделенная на области.



0.5GB	Область для системной периферии (NVIC) и отладки
1.0GB	Область для внешней периферии
1.0GB	Область для внешней памяти
0.5GB	Область для периферии
0.5GB	Память данных
0.5GB	Память программ и таблица векторов прерываний

Состояния ARM Cortex-M3



	Привилегированный режим	Непривилегированный режим
Выполняются код исключения (Handler)	**Подпрограммы обработчики прерываний**	-
Выполняется основной код (Thread)	**Код операционной системы**	**Код задачи/процесса**

Контроллер прерываний NVIC

Вложенный векторный контроллер прерываний (NVIC) является встроенной частью процессора Cortex–M3. Управляющие регистры отображены на память. Помимо регистров управления и логики управления обработкой прерываний, блок NVIC также содержит регистры управления для таймера SYSTICK и средства управления отладкой.

NVIC поддерживает от 1 до 240 входов внешних прерываний (запрос на прерывание IRQ). Точное количество поддерживаемых прерываний определяется производителями. Кроме того, NVIC также имеет вход немаскируемого прерывания (NMI). Фактическая функция NMI также определяется производителем чипа.

Исключения и прерывания ARM Cortex-M3

№ исключения	Название	Приоритет	Описание
0	-	-	Нет активного исключения
1	Reset	-3 (макс.)	Сброс
2	NMI	-2	Немаскируемое прерывание (внешняя линия)
3	Hard Fault	-1	Все аппаратные аварии
4	MemManage Fault	Программ.	Обращение к адресам под защитой MPU
5	Bus Fault	Программ.	Ошибка обращения к шине
6	Usage Fault	Программ.	Ошибка исполнения команды
7-10	Reserved	-	
11	SVCall	Программ.	Вызов супервизора (программное прерывание)
12	Debug Monitor	Программ.	Монитор отладки (breakpoint, watchpoint и т.д.)
13	Reserved	-	
14	PendSV	Программ.	Отложенная обработка супервизора
15	SYSTICK	Программ.	Системный таймер
16	IRQ #1	Программ.	Прерывание №1 от периферии
17	IRQ #2	Программ.	Прерывание №2 от периферии
…	…	…	…
255	IRQ #239	Программ.	Прерывание №239 от периферии

Сброс и последовательность запуска ARM Cortex-M3

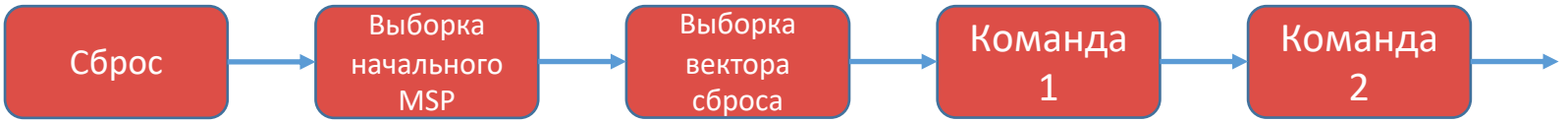
Виды сброса (Reset):

- Power on Reset – сброс по питанию, сбрасывает все в микроконтроллере – процессор, периферию и модуль отладки
- System Reset – сбрасывает только процессор, периферию, но не модуль отладки
- Processor Reset – сбрасывает только процессор

Startup файл:

```
__Vectors DCD __initial_sp  ; Вершина стека
          DCD Reset_Handler ; Вектор сброса
...
```

Последовательность сброса:



Чтение слова по адресу 0x00000000 и копирование в SP

Чтение слова по адресу 0x00000004 и копирование в PC

Чтение первой команды

Двоичный интерфейс приложений

Для того, чтобы подпрограммы и программы понимали друг друга разработан двоичный интерфейс приложений (ABI – Application Binary Interface).

В этом интерфейсе описано то, как использовать регистры и стек между вызовами, как осуществлять вызовы супервизора операционной системы и так далее.

ABI для ARM:

https://github.com/ARM-software/abi-aa

Разработчики компиляторов в обязательном порядке соблюдают ABI. В противном случае программы собранные в разных компиляторах не понимали друг друга.

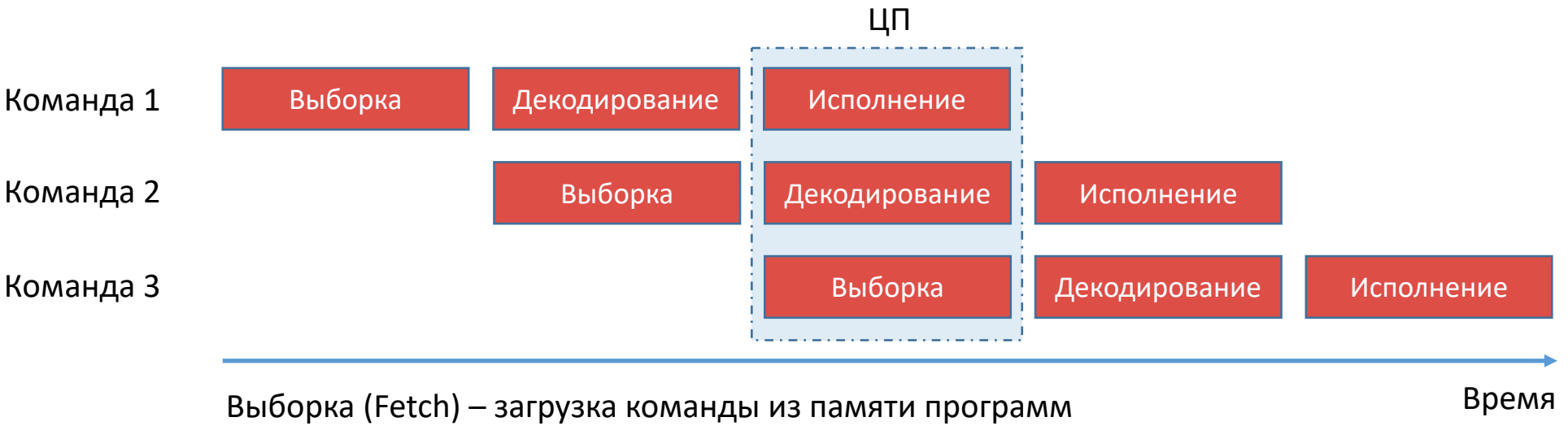
Если требуется написать подпрограмму на ассемблере с последующим вызовом из языка C, то также необходимо соблюдать требования ABI.

Двоичный интерфейс приложений ARM

R0	Аргумент / возвращаемое значение / временный регистр 1
R1	Аргумент / возвращаемое значение / временный регистр 2
R2	Аргумент / временный регистр 3
R3	Аргумент / временный регистр 4
R4	Регистр для переменных 1
R5	Регистр для переменных 2
R6	Регистр для переменных 3
R7	Регистр для переменных 4
R8	Регистр для переменных 5
R9	Регистр для переменных 6
R10	Регистр для переменных 7
R11	Регистр для переменных 8
R12	Временный регистр
R13 (SP)	Указатель стека Stack Pointer
R14 (LR)	Регистр возврата Link Register
R15 (PC)	Счетчик команд Program Counter

```
возвр-значение function (арг1, арг2, арг3, арг4)
{
   врем-переменная;
}
```

Конвейер ARM Cortex-M3



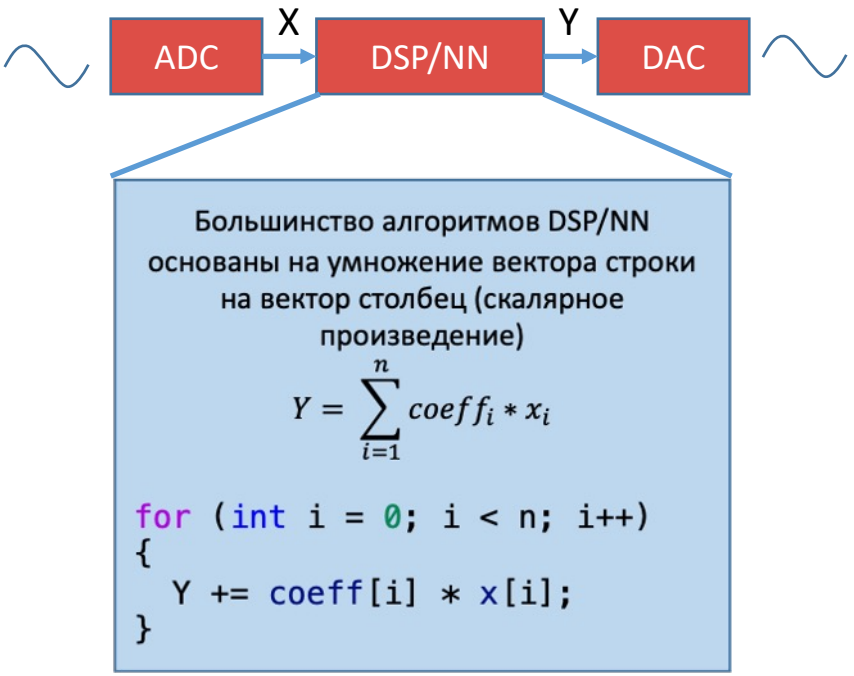
Выборка (Fetch) – загрузка команды из памяти программ
Декодирование (Decode) – декодирование команды
Исполнение (Execute) – чтение данных из регистров, передача в АЛУ, запись в регистры

Исходный код:

```
LDR  R0, [R1]
LDR  R2, [R3]
ADDS R0, R0, R2
ANDS R0, R0, R2
STR  R0, [R3]
STR  R2, [R1]
```

Выборка	Декодирование	Исполнение
LDR R0, [R1]	-	-
LDR R2, [R3]	LDR R0, [R1]	-
ADDS R0, R0, R2	LDR R2, [R3]	LDR R0, [R1]
ANDS R0, R0, R2	ADDS R0, R0, R2	LDR R2, [R3]
STR R0, [R3]	ANDS R0, R0, R2	ADDS R0, R0, R2
STR R2, [R1]	STR R0, [R3]	ANDS R0, R0, R2
-	STR R2, [R1]	STR R0, [R3]

Конвейер ARM Cortex-M3



Команды перехода приводят к перезагрузке конвейера (особенно когда где нет блока предсказания ветвлений), а в других системах к промахам кэша.

Компилятор (с помощью включенной оптимизации или специальной директивы) или программист должны раскрутить цикл (loop unrolling) для уменьшения количества переходов. Это приводит к увеличению быстродействия, но и к увеличению размера программы.

```
for (int i = 0; i < n; i += 4)
{
  Y += coeff[i + 0] * x[i + 0];
  Y += coeff[i + 1] * x[i + 1];
  Y += coeff[i + 2] * x[i + 2];
  Y += coeff[i + 3] * x[i + 3];
}
```

Оптимальная функция для Cortex-M3

Для написания оптимальное функции на языке С следует понимать архитектуру процессора.

В оптимальной функции:

- До четырех 32-х разрядных аргументов (остальные через стек)
- До одного 32-х разрядного или одного 64-х разрядного возвращаемого значения
- До 8 – 12 32-х разрядных локальных переменных (остальные на стеке)
- Оптимальный тип данных: `uint32_t` или `int32_t`

```
int32_t perfect_func(int32_t a, int32_t b, int32_t c, int32_t d)
{
  int32_t e, f, g, h;
  uint32_t i, j, k, l;
  ...
  return e;
}
```

Ограничение на количество локальных переменных и аргументов также соотносится с рекомендацией писать функции, имеющие не более 4 – 7 переменных. Человек способен в уме удерживать одновременно не более 4 – 7 элементов, что позволяет «отлаживать и запускать программу в уме».

Другие особенности Cortex-M3

Типы данных float/double медленные в Cortex-M3. В процессоре отсутствует аппаратная поддержка чисел с плавающей запятой (нет FPU), только программная эмуляция.

Критичные по скорости вычисления циклы должны учитывать наличие и глубину конвейера (3 стадии) центрального процессора. Например, циклы преобразований данных.

Программы для многоядерных систем должны учитывать наличие конвейера при работе с общей памятью. Один процессор может прочитать память до того как полностью исполнилась команда другого процессора, которая обращалась к той же памяти. Существуют команды для синхронизации памяти и команд.

Центральный процессор имеет модуль защиты памяти (MPU), который позволяет защитить критически важные участки кода и данных.

Центральный процессор имеет некоторую поддержку операционных систем: команды вызова супервизора (`SVC`), два стека (MSP, PSP), режимы работы процессора (Thread/Process, Privilege/User).

Другие популярные архитектуры микроконтроллеров

Наимено-вание	S08	MCS51	AVR	PIC24	MSP430	S12	ARM Cortex-M3	RISC-V (RV32I)	MIPS32
Разрядность	8 бит			16 бит			32 бит		
Набор команд	CISC	CISC	RISC	RISC	RISC	CISC	RISC	RISC	RISC
Архитектура	Принстонская	Гарвардская	Гарвардская	Гарвардская	Принстонская	Принстонская	Гарвардская	Гарвардская	Гарвардская
Регистры, шт.	2	2	32	16	16	4	16	32	32
Шина адреса, бит	16	2 x 16	16	24	16	16	32	32	32
Произво-дители	NXP	Silicon Labs, Microchip, Nuvoton	Microchip	Microchip	Texas Instruments	NXP	ST, NXP, Texas Instuments, GigaDevice	GigaDevice, WCH, Espressif, Micron, SiFive	Microchip

Заключение

1. Архитектура ARM Cortex-M
   - Современная 32-х разрядная RISC архитектура;
   - Большинство производителей микроконтроллеров имеют изделия с ядром Cortex-M.

2. Понимание архитектуры процессора помогает писать оптимальный код на языке С.

Дополнительные слайды

Блок FPU процессора Cortex-M4

Блок обработки чисел с плавающей запятой FPU (Floating Point Unit) является опциональным в процессорах Cortex-M4 и выше.

Дробные числа в микропроцессорных системах могут быть представлены в нескольких форматах: числа с фиксированной запятой или числа с плавающей запятой по IEEE 754.

В процессорах без FPU происходит программная эмуляция операций с числами с плавающей запятой, которая может выполнятся сотни тактов. Например, для выражения `1.0/2.0` происходит вызов функции вида `_fdiv(1.0, 2.0)`.

Наличие блока FPU предполагает дополнительные регистры и команды, но при этом вычисление происходит за несколько тактов. Компилятор при наличии FPU формирует команды копирования операндов в регистры FPU и вызова соответствующих команд.

Например, выражение `1.0/2.0` будет исполнено одной командой `VDIV.F32 S0, S1, S0`

Блоки FPU бывают для чисел с одинарной точностью (float, 32 бита) и двойной точность (double, 64 бита).

В Cortex-M4 блок FPU с одинарной точностью имеет 16 регистров `S0-S15` для вычислений.

Шинные интерфейсы AMBA

Спецификация Advanced Microcontroller Bus Architecture (AMBA) определяет стандарт связи внутри кристалла для разработки систем на кристалле.

В спецификации AMBA определены несколько шин.

AXI (Advanced eXtensible Interface) – шинный интерфейс для высокопроизводительных и высокочастотных систем. Применяется для процессоров Cortex-A, Cortex-R, а также различных процессоров RISC-V.

AHB (Advanced High-performance Bus) – усовершенствованная высокопроизводительная шина.

AHB-Lite (Advanced High-performance Bus) – высокопроизводительная шина для выборки команд из памяти программ и работы с оперативной память центральным процессором. Применяется в микроконтроллерах на процессоре Cortex-M.

APB (Advanced Peripheral Bus) – шина периферийных модулей, имеет ограниченную пропускную способность (меньше разрядность и частота тактирования). Применяется в микроконтроллерах на процессоре Cortex-M.

Каждому устройству на шине требуется включать тактирование.

Архитектура RISC-V

RISC-V — расширяемая открытая и свободная система команд (ISA) и архитектура процессора.

Спецификации архитектуры доступны на сайте альянса RISC-V.

Различные компанию разрабатывают коммерческие процессоры по спецификациям RISC-V.

Существуют открытые реализации процессоров RISC-V, например Syntacore SCR1 или проект PULP.

Можно собрать свой собственный процессор на ПЛИС.

Программно-логическая модель и карта памяти RISC-V

Программно-логическая модель RISC-V отличается простотой, модульностью и гибкостью.

Регистры: RISC-V имеет небольшое фиксированное количество регистров общего назначения (32 или 16 для МК), которые можно использовать для манипулирования данными и их хранения. Эти регистры унифицированы, что облегчает компиляторам оптимизацию генерации кода.

Инструкции: RISC-V имеет несколько форматов (тип R, тип I, тип S, тип B, тип U и тип J), что обеспечивает широкий спектр операций и режимов адресации. Эта модульность обеспечивает эффективное кодирование инструкций и упрощает процесс декодирования.

Память: RISC-V поддерживает память как с побайтовой, так и с пословной адресацией.

Обработка исключений: RISC-V имеет механизм обработки исключений и прерываний, позволяющий процессору реагировать на такие события, как деления на ноль, системные вызовы или внешние прерывания.

Уровни привилегий: RISC-V имеет несколько уровней привилегий (пользовательский режим, режим супервизора, режим гипервизора и машинный режим), которые позволяют различным компонентам программного обеспечения выполняться на разных уровнях привилегий.

zero/x0	a6/x16
ra/x1	a7/x17
sp/x2	s2/x18
gp/x3	s3/x19
tp/x4	s4/x20
t0/x5	s5/x21
t1/x6	s6/x22
t2/x7	s7/x23
s0/fp/x8	s8/x24
s1/x9	s9/x25
a0/x10	s10/x26
a1/x11	s11/x27
a2/x12	t3/x28
a3/x13	t4/x29
a4/x14	t5/x30
a5/x15	t6/x31

Микроконтроллер 1986BE92QI. Структура

Миландр 1986BE92QI – 32-х разрядный МК общего назначения.

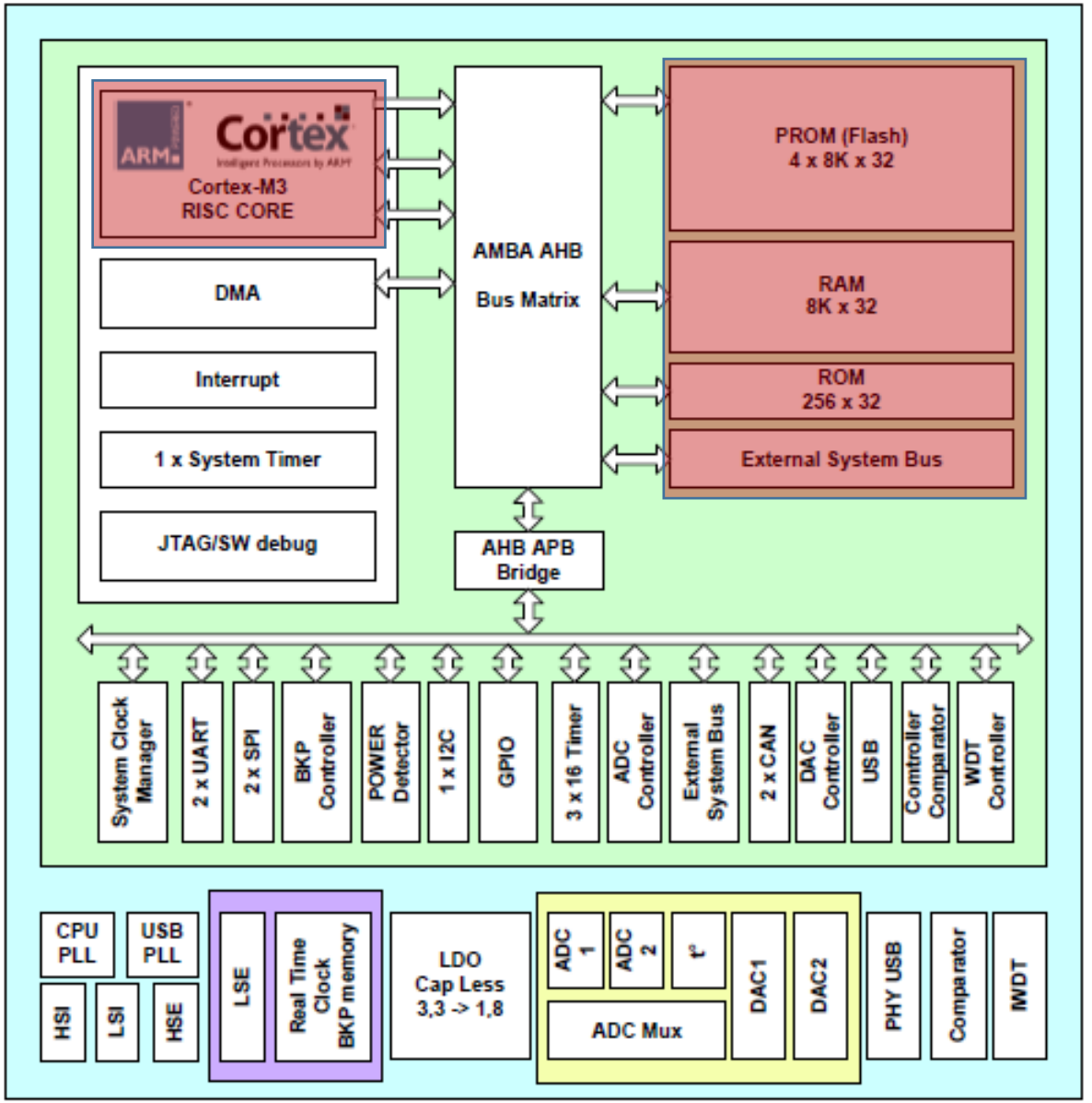
Сфера применения: системы управления

Ядро:

- ARM 32-битное RISC-ядро Cortex-M3, тактовая частота до 80 МГц;
- блок аппаратной защиты памяти MPU;

Память:

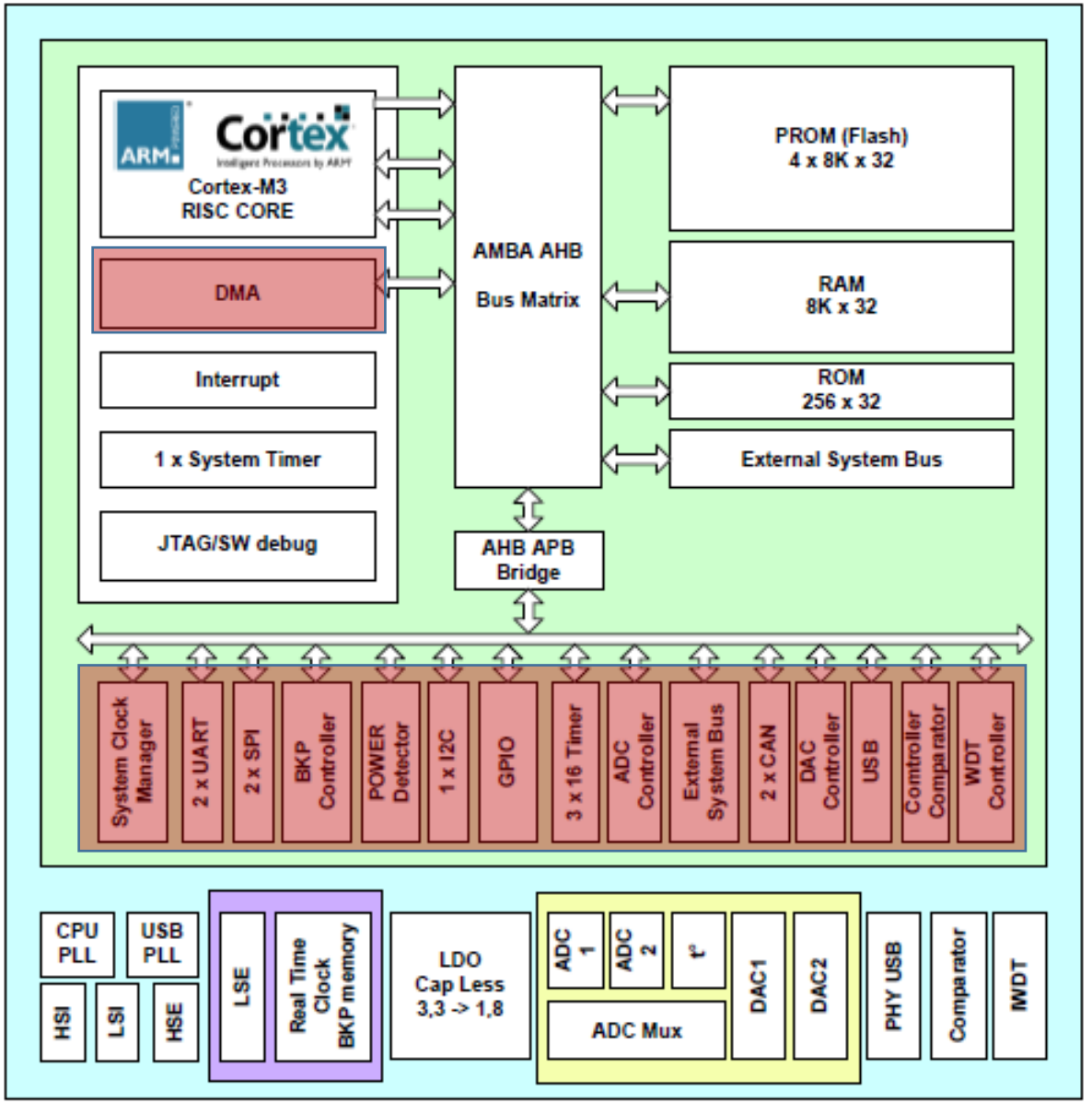
- встроенная энергонезависимая Flash-память программ - 128 Кбайт;
- встроенное ОЗУ - 32 Кбайт;
- контроллер внешней шины.



Микроконтроллер 1986BE92QI. Структура

Периферия:

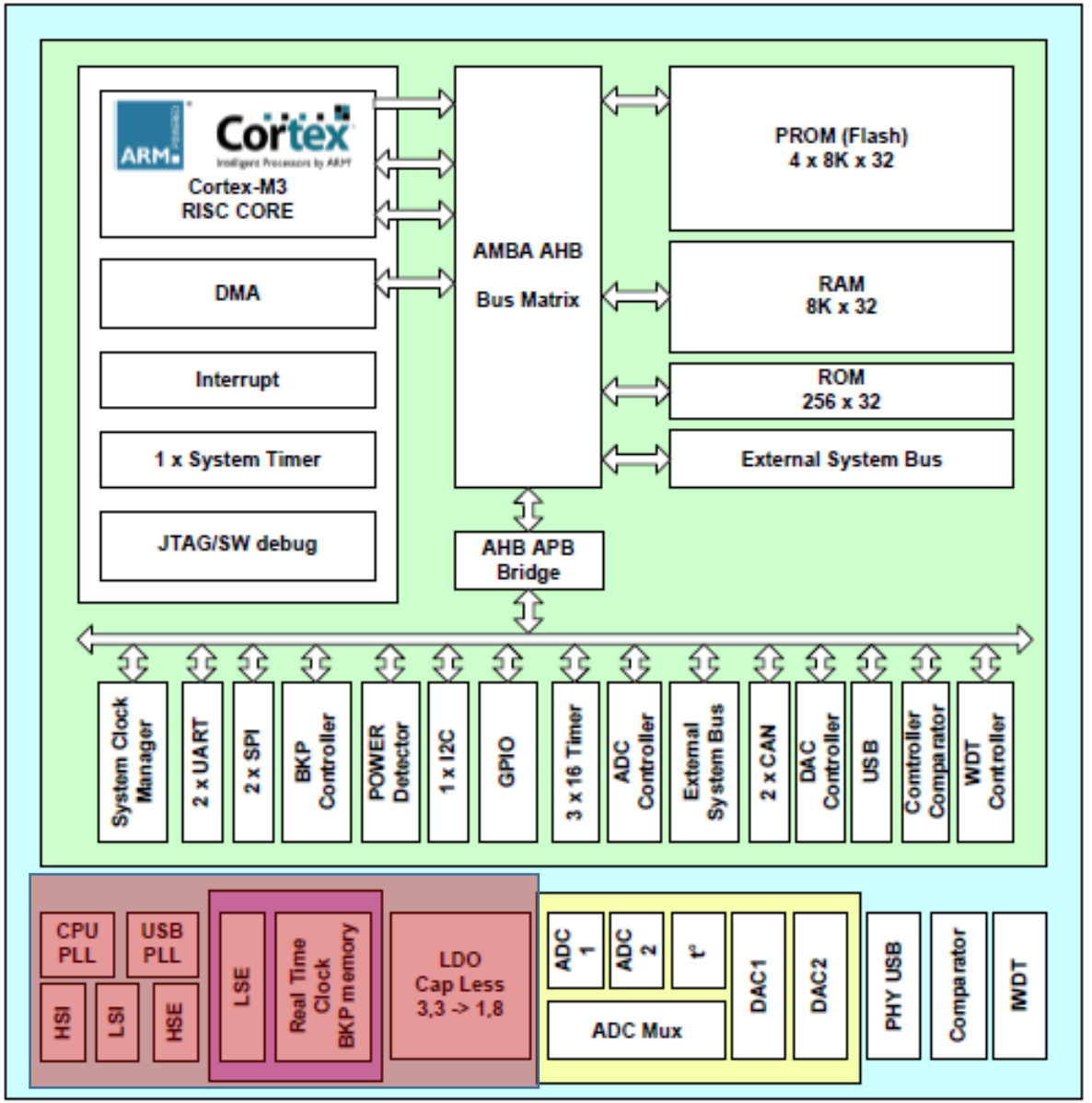
- контроллер DMA с функциями передачи Периферия-Память, Память-Память;
- контроллер USB интерфейса с функциями работы Device и Host;
- контроллеры интерфейсов UART, SPI, I2C, CAN;
- три 16-разрядных таймер-счетчика с функциями ШИМ и регистрации событий;
- до 96 пользовательских линий ввода-вывода.



Микроконтроллер 1986BE92QI. Структура

Питание и тактовые генераторы:

- внешнее питание 2,2 ÷ 3,6 В;
- встроенный регулируемый стабилизатор напряжения на 1,8 В для питания ядра;
- встроенные подстраиваемые RC генераторы 8 МГц и 40 кГц;
- внешние кварцевые резонаторы на 2 ÷ 16 МГц и 32,768 кГц;
- встроенный умножитель тактовой частоты PLL для ядра;
- встроенный умножитель тактовой частоты PLL для USB.



Микроконтроллер 1986BE92QI. Структура

Аналоговые модули:

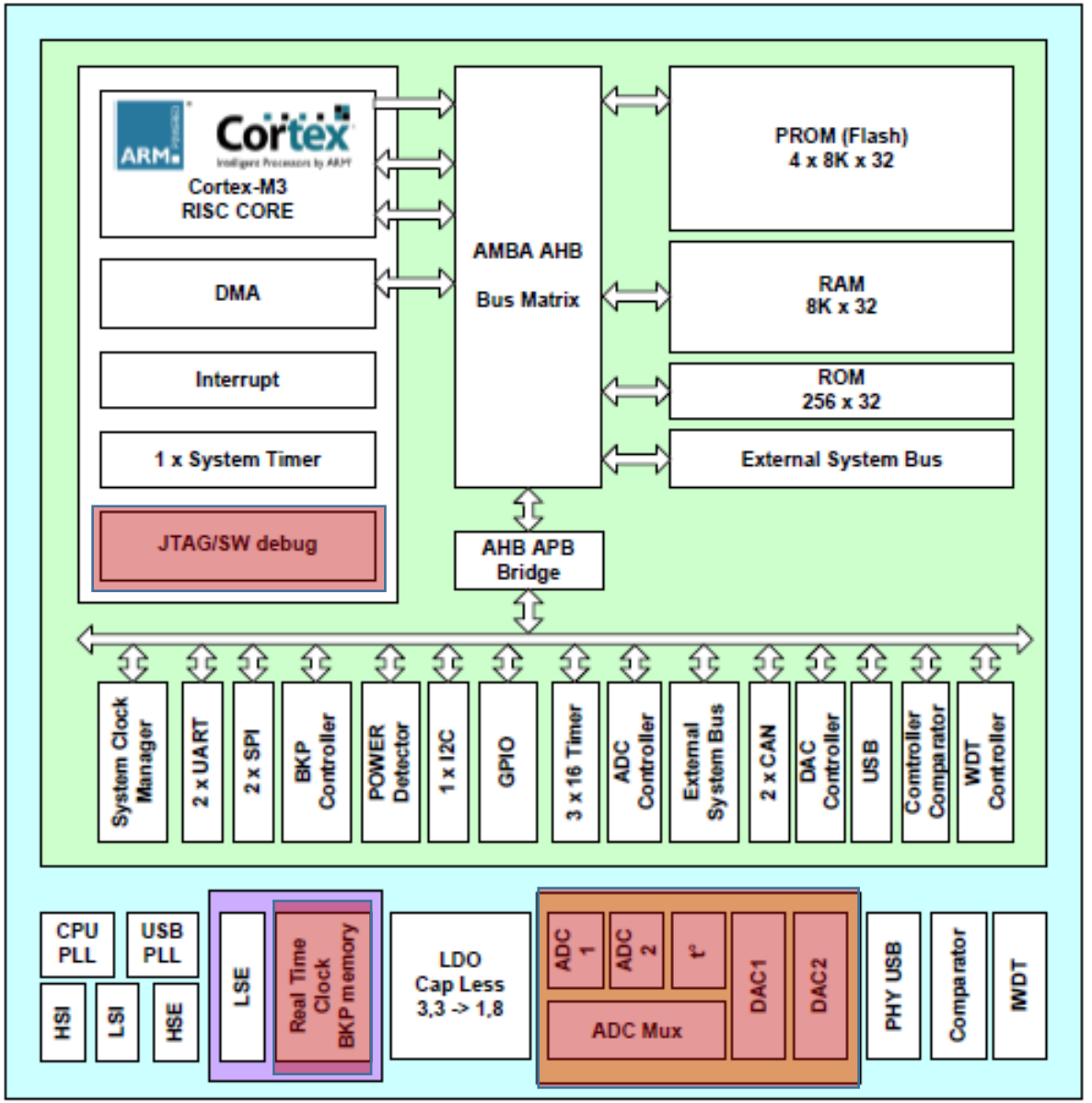
- два 12-разрядных АЦП (до 16 каналов);
- температурный датчик;
- двухканальный 12-разрядный ЦАП;
- встроенный компаратор.

Режим пониженного энергопотребления:

- режимы Sleep, Deep Sleep и Standby;
- батарейный домен с часами реального времени и регистрами аварийного сохранения.

Отладочные интерфейсы:

- последовательные интерфейсы SWD и JTAG.



Микроконтроллер STM32F072R8T6

Структура МК серий STM32F0x1, STM32F0x2, STM32F0x7