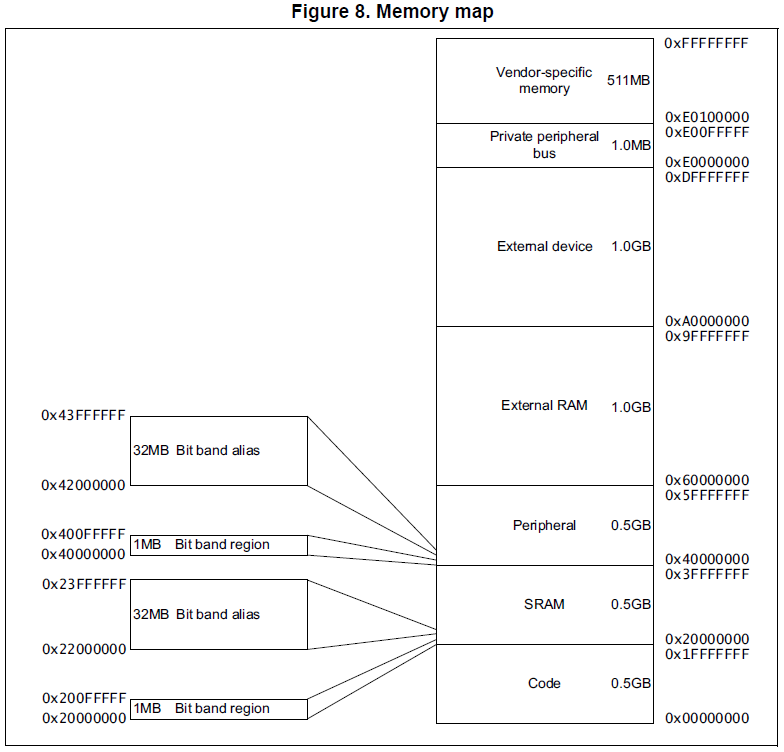
2.2 Модель памяти.

Этот раздел описывает карту памяти процессора, поведение доступов к памяти и особенности bit-banding’а. Процессор имеет фиксированную карту памяти, которая предоставляет до 4 Гб адресуемой памяти.



Регионы для SRAM и периферии включают регионы bit-band. Bit-banding предоставляет атомарные операции с битовыми данными. Процессор резервирует регионы области памяти закрытой периферийной шины (Private Peripheral Bus – PPB) для периферийных регистров ядра.

2.2.1 Регионы памяти, типы и атрибуты.

Карта памяти и программирования MPU разбивает карту памяти на регионы. Каждый регион имеет определённый тип памяти, и некоторые регионы имеют дополнительно атрибуты памяти. Тип памяти и атрибуты определяют поведение доступов к региону. Существуют следующие типы памяти:

Normal – процессор может переупорядочить транзакции для эффективности, или выполнять спекулятивные чтения.

Device – процессор сохраняет порядок транзакции относительно других транзакций к Device-памяти (т.е. к устройству) или Strongly-ordered памяти.

Strongly-ordered – процессор сохраняет порядок транзакций относительно всех остальных транзакций.

Различные требования упорядоченности для Device- или Strongly-ordered памяти означают, что система памяти может буферизовать запись в Device- память, но не должна буферизовать запись в Strongly-ordered память.

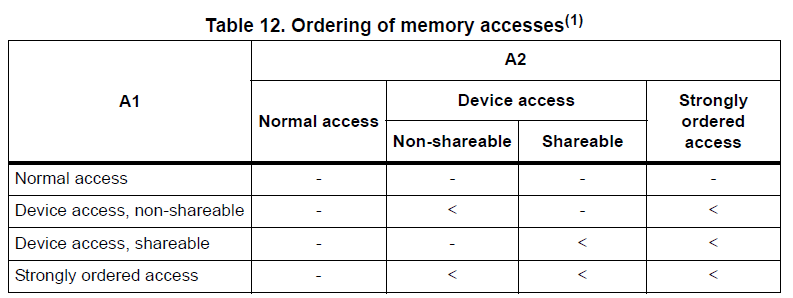
Дополнительные атрибуты памяти включают:

Execute Never (XN) – означает, что предотвращен доступ процессора к инструкциям. Любая попытка выборки инструкции из XN региона приведет к выбросу исключения memory management fault exception.

2.2.2 Системное упорядочивание памяти доступов к ней.

Для большинства обращений к памяти, вызванных явными инструкциями доступа к памяти, система памяти не гарантирует, что порядок, в котором выполняются реальные доступы к памяти, полностью соответствует программному порядку инструкций, но это не повлияет на поведение последовательности инструкций. Обычно, если корректное выполнение программы зависит от завершения двух доступов к памяти согласно их программному порядку, программа должна вставить инструкцию барьера памяти между инструкциями доступа к памяти.

Однако, система памяти гарантирует некоторую упорядоченность доступов к устройству и памяти типа Strongly-ordered. Для двух инструкций доступа к памяти A1 и A2, если A1 происходит перед A2 в порядке программы, то порядок доступов к памяти, вызванных этими инструкциями будет следующий:



1. – означает, что система памяти не гарантирует упорядоченность доступов.

< означает, что доступы происходят согласно их порядку в программе, т.е. A1 всегда происходит перед A2.

Прим. Автора – т.е. если первая инструкция обращается к региону типа «Normal», и вторая к региону типа «Normal», то порядок доступа может не соблюдаться. А чтобы он соблюдался – необходима инструкция барьера. А если обе инструкции обращаются к памяти Strongly-ordered access, то порядок доступа будет обеспечен.

2.2.3 Поведение доступов к памяти.

Поведение доступов в каждом регионе на карте памяти следующее:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Диапазон адресов | Регион памяти | Тип памяти | XN | Описание |
| 0x00000000-  0x1FFFFFFF | Code | Normal | - | Исполняемый регион для программного кода. Здесь также возможно размещение данных |
| 0x20000000-  0x3FFFFFFF | SRAM | Normal | - | Исполняемый регион для данных, здесь также возможно размещение кода. Этот регион включает bit-band и bit-band alias области. |
| 0x40000000-  0x5FFFFFFF | Периферия | Device | XN | Этот регион включает bit-band и bit-band alias области. |
| 0x60000000-  0x9FFFFFFF | Внешняя RAM | Normal | - | Исполняемый регион для данных. |
|  |  |  |  |  |
| 0xA0000000-  0xDFFFFFFF | Внешнее устройство | Device | XN | Память внешних устройств. |
| 0xED000000-  0xED0FFFFF | Закрытая периферийная шина | Strongly ordered | XN | Этот регион включает NVIC, Системный таймер и блок управления системой. |
| 0xED100000-  0xFFFFFFFF | Размеченная на карте периферия | Device | XN | Этот регион включает всю стандартную периферию STM32. |

Регионы Code, SRAM и внешний RAM могут хранить программы. Однако, рекомендуется, чтобы программы всегда использовали регион Code, потому что процессор имеет раздельные шины, которые позволяют выполнять выборки инструкций и доступ к данным одновременно. MPU может переопределить стандартное поведение доступа к памяти, описанное в этом разделе.

Предвыборка инструкции и предсказание ветвлений.

Процессор Cortex-M4:

* Выполняет упреждающую выборку инструкции.
* Выполняет теоретические предвыборки адресов ветвления.

2.2.4 Программное упорядочивание доступов к памяти.

Порядок инструкций в программном потоке не всегда гарантирует соответствующий порядок транзакций памяти. Это происходит потому что:

* Процессор может изменить порядок для некоторых доступов к памяти для улучшения эффективности, но это не влияет на поведение последовательности инструкций.
* Процессор имеет несколько шин интерфейсов.
* Память или устройства на карте памяти (на разметке) имеют различные wait states.
* Некоторые доступы к памяти являются буферизируемыми, или мнимыми.

Если порядок доступов к памяти критичен, то программа должна включить инструкции барьера памяти для принудительного сохранения порядка транзакций. Процессор предоставляет следующие инструкции барьера памяти:

DMB – Data Memory Barrier (DMB) инструкция, гарантирует, что невыполненные транзакции с памятью завершатся перед последующими транзакциями.

DSB – Data Synchronization Barrier (DSB) инструкция гарантирует, что невыполненные транзакции с памятью завершатся перед последующим исполнением инструкции.

ISB – Instruction Synchronization Barrier (ISB) – гарантирует, что эффект от всех завершенных транзакций с памятью будет распознан всеми последующими инструкциями.

Примеры использования инструкции-барьера:

* Векторная таблица. Если программа изменяет точку входа в векторную таблицу, и затем разрешает соответствующее исключение, используйте DMB инструкцию между операциями. Это гарантирует, что если произойдет выброс исключения сразу после того, как оно было разрешено, процессор использует новый вектор исключения. Прим. Автора – т.е. при изм. Адреса векторной таблицы мы исп. Память normal, разрешая прерывание, мы исп. Память device. И может получиться так, что разрешение прерывания произойдёт прежде, чем будет перенесена векторная таблица. Поэтому здесь нужен барьер.
* Само-изменяющийся код. Используйте ISB инструкцию немедленно после изменения кода в программе. Это гарантирует, что последующее исполнение программы будет по новым инструкциям.
* Переключение разметки памяти. Если система содержит механизм переключения карты памяти (разметки памяти), используйте DSB инструкцию после переключения карты в программе. Это гарантирует, что последующие инструкции будут использовать обновленную разметку.
* Динамическое изменение приоритета исключения. Когда изменяется приоритет исключения и при этом исключение находится в состоянии ожидания или активно, используйте DSB инструкции после изменения. Это гарантирует, что изменение приоритета вступит в силу после завершения инструкции DSB.
* Использование семафора в много-мастерных системах. Если система содержит на шине более одного мастера, например, если другой процессор присутствует в системе, каждый процессор должен использовать инструкцию DMB после любой инструкции семафора. Это гарантирует, что другие шинные мастеры видят транзакции с памятью в том порядке, в каком они были исполнены.

Доступы к памяти типа Strongly-ordered, например, от блока управления системой, не требуют использования DMB инструкций.

2.2.5 Bit-banding.

Bit-band регион проецирует каждое **слово из региона доступа к битам bit-band alias** на единственный **бит в регионе хранения битов bit-band**. Регионы хранения битов bit-band занимают нижний 1 Мбит в SRAM и в периферийном регионе памяти. На разметке памяти имеется два 32 мб-ых региона доступа к битам, которые проецируются на два 1 Мбит-ых региона хранения битов.

* Доступы к 32 Мбит-му региону доступа к битам в SRAM проецируются в 1Мбит-ый регион хранения битов в SRAM, как показано в таблице 14.
* Доступы к 32 Мбит-му периферийному региону доступа к битам проецируются в 1Мбит-ый периферийный регион хранения битов, как показано в таблице 15.

Таблица 14. Bit-banding регионы SRAM.

|  |  |  |
| --- | --- | --- |
| Диапазон адресов | Регион памяти | Доступы к инструкциям и данным |
| 0x20000000-0x200FFFFF | Регион хранения битов в SRAM | Прямой доступ к этой области памяти ведет себя как доступ к памяти  SRAM, но эта область доступна также побитно, с помощью ссылок из  области bit-band alias |
| 0x22000000-0x23FFFFFF | Регион доступа к битам в SRAM | Доступ к данным в этой области переназначается на область bit band.  Операция записи данных выполняется в стиле "чтение-изменение-запись".  При выборке инструкции операция переназначения не производится. |

Таблица 15. Bit-banding регионы памяти периферии.

|  |  |  |
| --- | --- | --- |
| Диапазон адресов | Регион памяти | Доступы к инструкциям и данным |
| 0x40000000-  0x400FFFFF | Периферийный регион хранения битов | Прямой доступ к этой области памяти ведет себя как доступ к памяти  периферии, но эта область доступна также побитно, с помощью ссылок из  области bit-band alias |
| 0x42000000-  0x43FFFFFF | Периферийный регион доступа к битам. | Доступ к данным в этой области переназначается на область bit band.  Операция записи данных выполняется в стиле "чтение-изменение-запись".  При выборке инструкции операция переназначения не производится. |

Примечание: доступ к слову в SRAM или периферийному региону доступа к битам bit-band alias проецируется на единственный бит в SRAM или периферийном регионе хранения битов bit-band.

Битовый доступ может использовать посылки, размерами байт, полуслово или слово. Размер битовой посылки соответствует размеру посылки инструкции, выполняющей битовый доступ.

Следующая формула показывает, как регион доступа к битам проецируется в регион хранения битов:

bit\_word\_offset = (byte\_offset x 32) + (bit\_number x 4)

bit\_word\_addr = bit\_band\_base + bit\_word\_offset

Где:

bit\_word\_offset – это позиция целевого бита в регионе хранения битов.

bit\_word\_addr – это адрес слова в регионе доступа к битам, которое проецируется на целевой бит.

bit\_band\_base – начальный адрес региона доступа к битам.

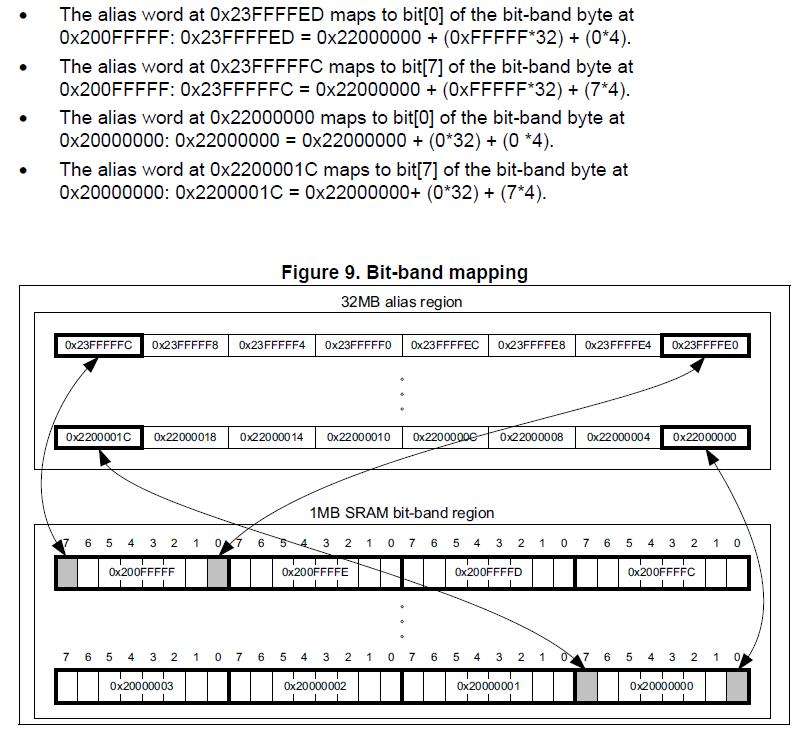
byte\_offset – количество байтов до целевого бита в регионе хранения битов, который содержит целевой бит.

bit\_number – это позиция целевого бита, от 0 до 7.

Прим. Автора: первый бит (0) из региона хранения битов адресуется первым словом (0) из области доступа к битам. 3-ий бит адресуется 3-им словом и т.д. Т.е. номер слова соответствует номеру бита. Вычислим адрес слова, адресующего 3ий бит:

bit\_word\_addr = bit\_band\_base + bit\_word\_offset = bit\_band\_base + (byte\_offset x 32) + (bit\_number x 4) = 0x22000000 + 0 + 3 \* 4 = 0x2200000C, что соответствует адресу третьего слова. Аналогично рассчитывается адрес слова, адресующего любой бит из региона хранения битов.

Еще примеры (в первом примере опечатка – на конце адреса слова, адресующего нулевой бит последнего байта из региона хранения битов, должен быть 0 а не D):



Прямой доступ к региону доступа к битам.

Запись в слово в регионе доступа к битам обновляет единственный бит в регионе хранения битов. Значение бита Bit[0], записанного в слово в регион доступа к битам, определяет значение, записанное в целевой бит в регион хранения битов. Запись значения с младшим битом равным 1 записывает в соответствующий бит из региона хранения битов 1, а если 0, то записывает туда 0. Остальные биты слова совершенно никак не влияют на целевой бит. Запись 0x01 – все равно что запись 0xFF. Запись 0x00 - все равно что запись 0x0E.

Чтение слова из региона доступа к битам: 0x00000000 индицирует, что целевой бит в регионе хранения битов установлен в 0. 0x00000001 индицирует, что целевой бит в регионе хранения битов установлен в 1.

Прямой доступ к региону хранения бит.

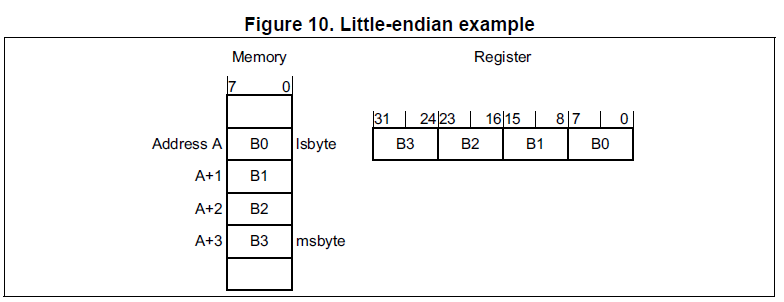
Обычное поведение – см. 2.2.3.

2.2.6 Байтовый порядок памяти.

Процессор видит память как линейный набор байтов, пронумерованных по порядку, начиная с нуля. Например, байты 0-3 хранят первое слово, байты 4-7 хранят второе слово и т.д.

Формат little-endian.

В формате little-endian, процессор сохраняет младший значащий байт слова по самому меньшему номеру байта, и старший значащий байт по самому большему номеру байта. Для примера – см. рисунок 10.



2.2.7 Примитивы синхронизации.

Набор инструкций Cortex-M4 включает пары примитивов синхронизации. Они обеспечивают не блокирующий механизм, который может использовать поток или процесс для получения монопольного доступа к ячейке памяти. Программа может использовать их для выполнения гарантированной последовательности обновления памяти типа чтение-модификация-запись, или для механизма семафора.

Пара примитивов синхронизации включает:

* Инструкция монопольной загрузки: используется для чтения значения ячейки памяти, запрашивая монопольный доступ к этой ячейке.
* Инструкция монопольного сохранения: используется для попытки записи в туже самую ячейку памяти, возвращая в регистр статусный бит. Если этот бит = 0, поток или процесс получил монопольный доступ к ячейке, и запись успешна. Если этот бит = 1, поток или процесс не получил монопольный доступ к ячейке, и никакой записи не выполнено.

Пары инструкций монопольной загрузки/сохранения следующие:

* Пословные инструкции LDREX и STREX.
* Полусловные инструкции LDREXH и STREXH.
* Байтовые инструкции LDREXB и STREXB.

Программа должна использовать инструкцию монопольной загрузки с соответствующей инструкцией монопольного сохранения.

Чтобы гарантированно выполнить операцию чтение-модификация-запись ячейки памяти, программа должна:

1. Использовать инструкцию монопольной загрузки для чтения значения ячейки.
2. Обновить значение, если требуется.
3. Использовать инструкцию монопольного сохранения для попытки записи нового значения обратно в память.
4. Проанализировать возвращенный статусный бит.

Программа может использовать примитивы синхронизации для реализации семафоров следующим образом:

1. Использовать инструкцию монопольной загрузки для чтения с адреса семафора для проверки, свободен ли семафор.
2. Если семафор свободен, использовать инструкцию монопольного сохранения для записи значения включенного семафора по адресу семафора.
3. Если возвращенный статусный бит с шага 2 индицирует, что операция выполнена успешно, то программа включила семафор. Однако, если нет, другой процесс мог включить семафор после шага 1.

Cortex-M4 включает монитор монопольного доступа, который отслеживает факт выполнения процессором инструкции монопольной загрузки. Если процессор – это часть многопроцессорной системы, система также глобально отслеживает ячейки памяти, адресуемые монопольным доступом другим процессором.

Процессор удаляет признак монопольного захвата если:

* Он выполнил инструкцию CLREX.
* Он выполнил инструкцию монопольного сохранения, независимо от того, была ли успешной запись.
* Произошло исключение. Это означает, что процессор может разрешить семафорные конфликты между различными потоками.

В многопроцессорных системах выполнение:

* CLREX инструкции удаляет только признак локального монопольного доступа для процессора.
* Инструкции монопольного сохранения, или исключения удаляет локальный признак монопольного доступа и глобальный признак для процессора.

2.2.8 Советы при программировании примитивов синхронизации.

ISO/IEC C не может напрямую генерировать инструкции монопольного доступа. CMSIS предоставляет внутренние функции для генерации этих инструкций:

