МИКРОСХЕМА ИНТЕГРАЛЬНАЯ
1891ВМ038

Руководство по эксплуатации
Часть 1

ТВГИ.431281.028РЭ

Литера « »

Перв. примен.	ТВГИ.431281.028

Настоящее руководство содержит основные сведения об устройстве и работе микросхемы интегральной 1891ВМ038 (микропроцессор Эльбрус-16С), которая является системой на кристалле и содержит 16 универсальных 64-разрядных процессорных ядер с архитектурой широкого командного слова Эльбрус, общий для процессорных ядер кэш третьего уровня, восемь контроллеров оперативной памяти с каналами обмена DDR4(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается), три контроллера межпроцессорных каналов для организации многопроцессорных систем, контроллер канала ввода-вывода для подключения контроллера периферийных интерфейсов КПИ-2, два набора контроллеров периферийных интерфейсов: высокоскоростных (PCI Express 3.0, 1G и 10G Ethernet, USB 3.0/2.0, SATA 3.0) и низкоскоростных и служебных интерфейсов (I2C, SPI, RS-232, GPIO, HDA и др.). В руководстве представлены также сведения об интерфейсах микросхемы, электрических параметрах входных и выходных сигналов, электропитании, потребляемой мощности, конструкции, даны рекомендации по использованию и условиям эксплуатации.

Руководство состоит из трех частей.

Часть 1 ТВГИ.431281.028.РЭ содержит описание устройства микропроцессора и его эксплуатационные характеристики.

Часть 2 ТВГИ.431281.028.РЭ1 содержит описание управляющих регистров, за исключением регистров процессорных ядер Core и контроллера периферийных интерфейсов EIOH.

Часть 3 ТВГИ.431281.028.РЭ2 содержит описание управляющих регистров контроллера периферийных интерфейсов EIOH.

Описание регистров процессорных ядер Core содержится в руководстве Система команд, часть 1 ТВГИ.431281.028.Д67 и часть 5 ТВГИ.431281.028.Д67.4.

Руководство предназначается для разработчиков и производителей вычислительных систем на основе микропроцессора Эльбрус-16С, разработчиков программного обеспечения для таких вычислительных систем и обслуживающего персонала.

При изучении данного руководства следует ознакомиться со следующими документами:

- таблица назначения выводов ТВГИ.431281.028ТБ;

- описание системы команд ТВГИ.431281.028Д67, ТВГИ.431281.028Д67.1, ТВГИ.431281.028Д67.2, ТВГИ.431281.028Д67.3, ТВГИ.431281.028Д67.4.

Содержание

Листов 327

Условные обозначения, термины и сокращения

Условный элемент	Описание
#	номер
глобалы	глобальные данные
квадро	формат данных 128 бит
линк	синоним канала обмена (межпроцессорный и ввода-вывода)
мкм	микрометр, единица длины, 10^{-6} метра
мс	миллисекунда, 10^{-3} секунды
МП	микропроцессор
нс	наносекунда, 10^{-9} секунды
ОП	оперативная память
ОО1	опытный образец 1 итерации
ОО2	опытный образец 2 итерации
ПЛИС	программируемая логическая интегральная схема
ПО	программное обеспечение
ресет	сброс
узел	синоним микропрцессора, когда он используется в качестве элемента в сети микропроцессоров (элемент в многопроцессорной системе)
AAU	Array Access Unit - устройство обращения к массивам
AC	Address Controller - адресный контроллер
ALC	Arithmetic Logic Channel - арифметико-логический канал (синоним ALU)
ALU	Arithmetic Logic Unit - арифметико-логическое устройство (синоним ALC)
APB	Array Prefetch Buffer - буфер предварительной подкачки массивов
CF	Chain File - файл стека связующей информации
CPU	Central Processor Unit - центральный процессор
CTPR	Control Transfer Preparation Register - регистр подготовки передачи управления
CU	Control Unit - устройство управления
DAM	Disambiguation Memory - устройство для мониторинга конфликтов

Условный элемент	Описание
	обращений в память по чтению и записи
DDR	Double Data Rate - передача данных с двойной скоростью
DW	Double Word – двойное слово (8 байтов)
hard_reset	аппаратный сброс
host	центральная часть устройства
IB	Instruction Buffer - буфер команд (устройство)
IO	Input Output - ввод - вывод
ITAG	Instruction Tag - память тегов кэша команд
ITLB	Instruction Table Look-Aside Buffer - кэш таблицы страниц для программного кода
I$	Instruction Cache - кэш команд (первого уровня)
L1D$	Level 1 Data Cache - кэш данных первого уровня
L2$	Level 2 Cache - кэш второго уровня
L3$	Level 3 Cache - кэш третьего уровня
MAU	Memory Access Unit - устройство обращения в память
MLT	Memory Locks Table - таблица блокировок памяти
MMX	Multi Media Extension - расширение системы команд для обработки мультимедийных данных
MMU	Memory Management Unit - устройство управления памятью
NA	Not Asign – не назначено, отсутствует
offset	смещение
PA	Phisical Address – физический адрес
PF	Predicate File - предикатный файл
PIO	Processor Input Output – обмен с вводом-выводом по командам от процессорного ядра
PLU	Predicate Logic Unit - устройство логических предикатов (синоним PU)
PT	Page Table - таблица страниц (соответствия страниц виртуальной и физической памяти)
PTE	Page Table Entry - строка таблицы страниц
PU	Predicate Unit - устройство предикатов (синоним PLU)

Условный элемент	Описание
RF	Register File - регистровый файл
SLT	Semaphore Locks Table - таблица блокировок семафора
soft_reset	программный сброс
src	source - источник
SRU	SubRoutine Unit - устройство подпрограмм
TLB	Table Look-Aside Buffer - буфер таблицы страниц
TLU	Table Look-Aside Unit - устройство обращения в таблицу страниц
TU	Trap Unit - устройство обработки прерываний
VA	Virtual Address – виптуальный адрес

1 Общие сведения

Микропроцессор Эльбрус-16С предназначен для использования в многопроцессорных системах с когерентной распределенной общей оперативной памятью, в которых каждый микропроцессор Эльбрус-16С имеет локальную секцию оперативной памяти, а доступ к секциям оперативной памяти других микропроцессоров Эльбрус-16С выполняется через межпроцессорные каналы. До четырех микропроцессоров Эльбрус-16С могут быть объединены в многопроцессорную систему простым соединением межпроцессорных каналов. В многопроцессорных системах с большим количеством микропроцессоров Эльбрус-16С следует использовать коммутаторы.

Встроенный набор контроллеров периферийных интерфейсов позволяет создавать вычислительные системы без внешнего контроллера периферийных интерфейсов КПИ-2. Однако применение контроллера периферийных интерфейсов КПИ-2 существенно увеличивает количество периферйных интерфейсов в вычислительной системе.

На рисунке 1.1 показана структурная схема 4-процессорной вычислительной системы на базе микропроцессоров Эльбрус-16С с контроллером периферийных интерфейсов КПИ-2 и двумя комплектами периферийных устройств, один из которых подключен непосредственно к микропроцессору Эльбрус-16С, а другой - к контроллеру периферийных интерфейсов КПИ-2.

Пространство оперативной памяти и пространство ввода-вывода в многопроцессорной системе распределены между микропроцессорами Эльбрус-16С. Расслоение памяти между микропроцессорами возможно 2 типов: по размеру страницы (4 Кбайт, 2 Мбайт или 1 Гбайт) и «крупноблочное» - пространство памяти, принадлежащей одному микропроцессору, является непрерывным. Тип расслоения задается программированием регистров конфигурации микропроцессора.

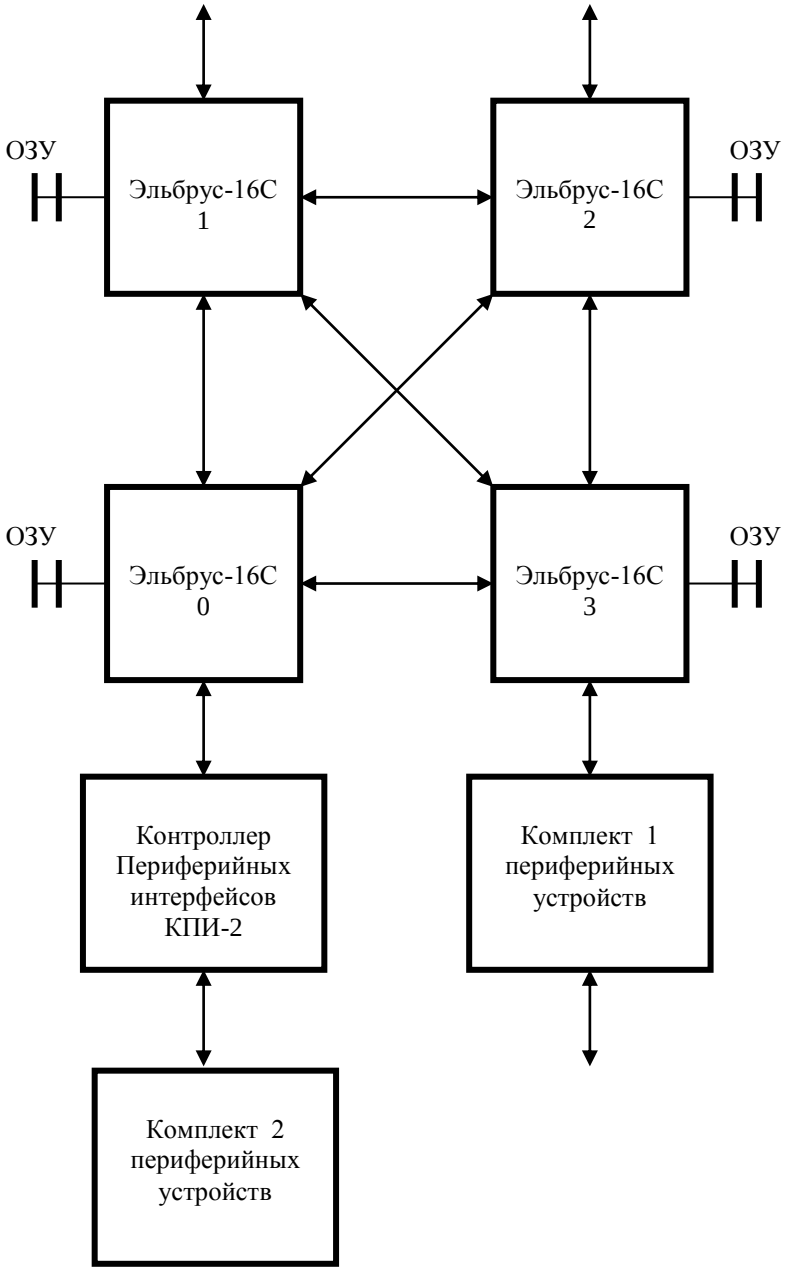


Рисунок 1.1 – 4-процессорная вычислительная система на базе микропроцессоров Эльбрус-16С

Оперативная память, подключенная к каждому микропроцессору, отображается на общее адресное пространство. Отсюда следует разделение памяти на "свою", т.е. подключенную к данному микропроцессору, и "чужую", обращение к которой требует пересылки запросов и данных по межпроцессорным каналам. Когерентность доступа нескольких микропроцессоров в общую память поддерживается посредством запросов проверки когерентности (когерентных запросов), рассылаемых в микропроцессоры системы в соответствии с информацией из кэша справочника, в котором представлены состояние каждого 64-байтового блока данных и указатель владельца самой "свежей" копии данных.

На рисунке 1.2 представлена структурная схема микропроцессора Эльбрус-16С. Микропроцессор Эльбрус-16С содержит функциональные блоки:

- коммутационную сеть OCN, обеспечивающую взаимодействие всех функциональных блоков микросхемы;
- 16 универсальных процессорных ядер Core 0 – 15;
- 16 банков L3 Bank 0 – 15 общей для процессорных ядер кэш-памяти третьего уровня L3 с локальным справочником LD;
- 4 устройства доступа к своей памяти HMU 0 - 3 с глобальными справочниками DIR;
- 8 контроллеров оперативной памяти MC 0 - 7 (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);
- устройство доступа к внешней памяти и каналам обмена XMU;
- система управления энергосбережением PCS;
- встроенный контроллер периферийных интерфейсов EIOH.

Коммутационная сеть OCN (On-Chip Network) реализует соединение между процессорными ядрами Core, кэш-памятью третьего уровня L3, устройствами доступа в оперативную память HMU и устройством доступа к внешней памяти и каналам обмена XMU. Коммутационная сеть OCN имеет распределенную структуру и обеспечивает высокую пропускную способность доступа процессорных

ядер к общей кэш-памяти третьего уровня L3, оперативной памяти и внешним устройствам, а также обменов между процессорными ядрами.

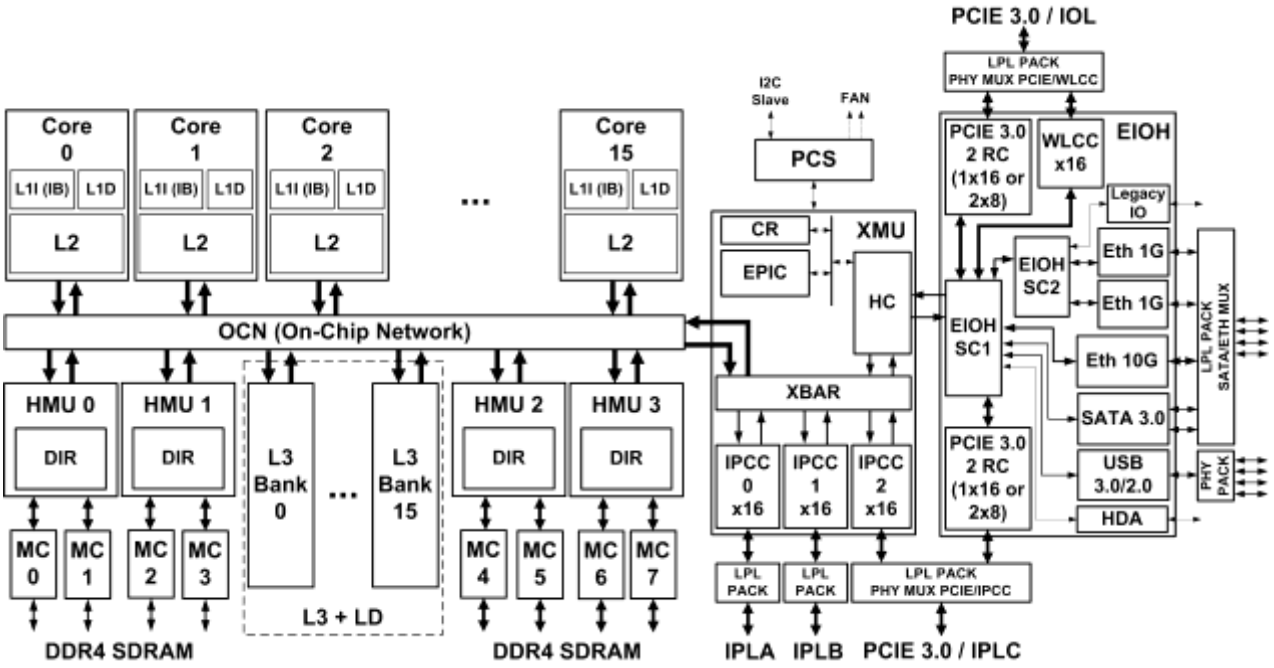


Рисунок 1.2 – Микропроцессор Эльбрус-16С

Процессорные ядра Core 0 – 15 являются высокопроизводительными универсальными процессорами с архитектурой широкого командного слова «Эльбрус», имеют параллельную внутреннюю организацию, позволяющую дешифрировать и выполнять каждый такт до 25 операций в скалярном режиме и до 73 операций для упакованных данных формата 32. В процессорном ядре Core реализована поддержка защищенных вычислений, двоичной компиляции кодов платформы Intel x86 и мультипрограммное выполнение на базе виртуальных машин. Каждое процессорное ядер Core имеет локальную кэш-память первого (L1I (IB) и L1D) и второго (L2) уровней.

Кэш-память третьего уровня L3, имеет объем 32 Мбайт, неинклюзивную организацию, распределенную структуру и адресное разделение на независимые

банки L3 Bank 0 - 15, причем любое процессорное ядро Core имеет доступ к каждому банку кэша. Локальный справочник LD (Local Directory) хранит информацию о владельцах актуальных копий кэш-строк, находящихся в кэшах процессорных ядер микросхемы. Это позволяет перенаправлять запросы владельцам актуальных копий кэш-строк.

Устройства доступа к своей оперативной памяти HMU 0 - 3 (Home Memory Unit) обслуживают запросы в оперативную память своих каналов памяти(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Каждое устройство HMU содержит глобальный справочник DIR (Directory), который хранит информацию обо всех кэш-строках (блоках), принадлежащих памяти своих каналов и находящихся в других микропроцессорах Эльбрус-16С многопроцессорной системы. Это позволяет перенаправлять запросы в память владельцам актуальных копий кэш-строк в другие микропроцессоры Эльбрус-16С.

Контроллеры оперативной памяти MC 0 – 7 (Memory Controller) управляют доступом к восьми секциям локальной памяти типа DDR4 SDRAM (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Каждая секция имеет размер до 128 Гбайт, а максимальный суммарный объем локальной оперативной памяти, подключенной к микропроцессору, составляет 1 Тбайт (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Контроллер MC реализует полудуплексный канал шириной восемь байт. Один канал обмена с памятью обеспечивает пиковую пропускную способность 19 Гбайт/с. Соответственно, пиковый темп обмена данными с памятью по восьми каналам составляет 152 Гбайт/с(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается).

Устройство доступа к внешней оперативной памяти и каналам обмена XMU (eXternal Memory Unit) включает в себя коммутатор XBAR, три контроллера

межпроцессорных каналов IPCC 0 – 2, хост-контроллер HC, блок управляющих регистров CR и контроллер процессорных прерываний EPIC.

Коммутатор XBAR обеспечивает подключение блоков устройства XMU к сети OCN. Кроме того, коммутатор XBAR поддерживает объединение двух или трех межпроцессорных каналов в один канал двойной или тройной ширины (мультилинк). Это позволяет повысить пропускную способность объединенного межпроцессорного канала в двухпроцесорных системах.

Контроллеры межпроцессорных каналов IPCC 0 – 2 (Inter Processor Channel Controller) обеспечивают доступ к оперативной памяти других микропроцессоров Эльбрус-16С в многопроцессорной системе. Это позволяет организовывать 2 – 4 процессорные системы с общей распределенной когерентной оперативной памятью. В каждом контроллере IPCC реализован дуплексный канал обмена шириной 16 линий с пиковой пропускной способностью 16 Гбайт/с. Пиковая пропускная способность трех межпроцессорных каналов составляет 48 Гбайт/с.

Хост-контроллер HC (Host-Controller) обеспечивает обмен с контроллером периферийных интерфейсов EIOH и отвечает за доступ к управляющим регистрам микропроцессора.

Блок управляющих регистров CR (Configuration Registers) содержит набор регистров состояния и управления всех устройств микропроцессора.

Контроллер процессорных прерываний EPIC (Elbrus Programmable Interrupt Controller) состоит из 16 отдельных контроллеров (по одному для каждого процессорного ядра Core). В их задачу входит прием прерываний для своего процессорного ядра от других процессорных ядер, контроллера EIOH и других микропроцессоров Эльбрус-16С, а также раздача прерываний от своего процессорного ядра в другие процессорные ядра и другие микропроцессоры Эльбрус-16С.

Система управления энергосбережением PCS (Power Control System) контролирует энергопотребление в зависимости от рабочей нагрузки путем изменения частоты синхронизации или ее отключения в неработающих устройствах. Изменение частоты синхронизации и отключение устройств используются

также в целях защиты микросхемы от перегрева при чрезмерной рабочей нагрузке в случаях превышения возможностей системы охлаждения.

Встроенный контроллер периферийных интерфейсов EIOH (Embedded Input Output Hub) содержит следующие контроллеры.

Контроллер канала ввода-вывода WLCC (Wide Link Channel Controller) обеспечивает доступ к контроллеру периферийных интерфейсов КПИ-2 (микросхема 1991ВГ2). В контроллере WLCC реализован дуплексный канал обмена шириной 16 линий с пиковой пропускной способностью 16 Гбайт/с.

Два блока двух контроллеров PCIE 3.0 RC – каждый из них содержит два контроллера PCI Express 1x16 и 1x8, причем контроллер 1x16 может работать также в режиме 1x8. Это позволяет использовать блок двух контроллеров PCIE 3.0 RC как один контроллер 1x16 или два контроллера 1x8. Пиковая пропускная способность блока двух контроллеров PCIE 3.0 RC составляет 32 Гбайт/с.

Мультиплексор PCIE/IPCC MUX позволяет подключить к одному физическому каналу, состоящему из 16 линий, контроллер межпроцессорного канала IPCC2 и/или блок двух контроллеров PCIE 3.0 RC. Возможные конфигурации:

- 1x16 PCI Express 3.0;
- 2x8 PCI Express 3.0;
- 1x16 IPCC0;
- 1x8 PCI Express 3.0 + 1x8 IPCC2;
- 2x4 PCI Express 3.0 + 1x8 IPCC2.

Мультиплексор PCIE/WLCC MUX позволяет подключить к одному физическому каналу, состоящему из 16 линий, контроллер канала ввода/вывода WLCC и/или блок двух контроллеров PCIE 3.0 RC. Возможные конфигурации:

- 1x16 PCI Express 3.0;
- 2x8 PCI Express 3.0;
- 1x16 WLCC;
- 1x8 PCI Express 3.0 + 1x8 WLCC;
- 2x4 PCI Express 3.0 + 1x8 WLCC.

Два контроллера Eth 1G являются контроллерами интерфейса Ethernet с режимами обмена 10/100/1000 или 25/250/2500 Мбит/с. Режим обмена задается программно подачей на контроллер частоты синхронизации 125 или 312,5 МГц соответственно.

Контроллер Eth 10G является контроллером интерфейса Ethernet с частотой обмена 10 Гбит/с.

Контроллер SATA 3.0 является 4-портовым контроллером интерфейса SATA 3.0 с частотой обмена 6 Гбит/с. Интерфейс используется для подключения накопителей на жестких магнитных и оптических дисках.

Мультиплексор SATA/ETH MUX позволяет подключать к 4четырем физическим одноразрядным каналам обмена контроллеры Eth 1G, Eth 10G и SATA 3.0. Возможные конфигурации:

- 4xSATA;
- 2xSATA +1xETH10G +1xETH1G;
- 2xSATA +1xETH10G +1xETH2.5G;
- 2xSATA+2xETH1G;
- 2xSATA+1xETH1G+1xETH2.5G;
- 2xSATA+2xETH2.5G;

Контроллер USB 3.0/2.0 имеет четыре порта интерфейса USB 3.0 с частотой обмена 5 Гбит/с и четыре порта интерфейса USB 2.0/1.1 с частотой обмена 480/12 Мбит/с, что позволяет реализовывать четыре универсальных порта USB 3.0/2.0/1.1 либо восемь портов (четыре USB 3.0 only + четыре USB 2.0/1.1). Интерфейс используется для подключения FLASH памяти, клавиатуры, графического манипулятора (мыши) и других устройств.

Контроллер HDA является аудиоконтроллером высокого разрешения.

Блок контроллеров низкоскоростных и служебных интерфейсов Legacy IO содержит набор контроллеров.

Контроллер I2C-SPI включает в себя контроллеры интерфейсов I2C/IPMB (пять каналов), SPI, контроллер прерываний ввода-вывода IOEPIC, системный и сторожевой (watchdog) таймеры, управление сбросом системы.

Два контроллера последовательного порта RS-232.

Контроллер SPMC - управлением питания и режимами энергосбережения на уровне вычислительного комплекса.

Контроллер GPIO/MPV - контроллер программируемых входов-выходов со встроенным функциональным блоком приема синхросигналов от систем реального времени или генераторов меток точного времени.

Микропроцессор имеет внешние интерфейсы:

- восемь каналов оперативной памяти DDR4 SDRAM (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);
- три межпроцессорных канала IPCC A, B, C x16;
- канал ввода-вывода WLCC x16;
- два канала PCI Express 1x16 или четыре канала PCI Express 1x8;
- два канала Ethernet 1G/2.5G и один канал Ethernet 10G;
- четыре порта SATA 3.0;
- четыре универсальных порта USB 3.0/2.0/1.1 (суммарно до восьми портов:

четыре порта USB 3.0 only + четыре порта USB 2.0/1.1);

- каналы интерфейсов I2C/IPMB (пять каналов), SPI, RS-232 (два канала), GPIO, HDA;
- канал I2C Slave для внешнего управления энергопотреблением и охлаждением;
- два канала управления вентиляторами FAN.

Примечание – Часть интерфейсов (контроллеров) делит одни и те же физические линии каналов обмена между собой (как описано выше), поэтому их одновременная работа не возможна.

Основные технические характеристики микропроцессора Эльбрус-16С представлены в таблице 1.1.

Таблица 1.1 - Основные технические характеристики микропроцессора

Эльбрус-16С

Наименование параметра	Значение параметра
Условное обозначение микросхем	1891ВМ03А8, 1891ВМ03В8
Основное функциональное назначение	Многоядерный микропроцессор для универсальной и специализированной обработки информации с производительностью терафлопного и петафлопного диапазонов
Количество процессорных ядер	16
Пиковая производительность 1891ВМ03А8*:	
- при выполнении операций с 64-разрядными числами, TFLOPS, не менее	0,75
- при выполнении операций с 32-разрядными числами, TFLOPS, не менее	1,5
- при выполнении операций с 64-разрядными числами, GOPS, не менее	320
- при выполнении операций с 32-разрядными числами, GOPS, не менее	640
- при выполнении операций с 16-разрядными числами, GOPS, не менее	1280
- при выполнении операций с 8-разрядными числами, GOPS, не менее	2560
Разрядность данных:	
─ целые	8, 16, 32, 64
─ с плавающей запятой	32, 64
Ёмкость кэш-памяти данных процессорного ядра, Кбайт	64
Ёмкость кэш- памяти команд процессорного ядра, Кбайт	128
Ёмкость кэш- памяти второго уровня процессорного ядра, Мбайт	1

Наименование параметра	Значение параметра
Ёмкость кэш- памяти третьего уровня, Мбайт	32
Контроллер оперативной памяти	8 (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)
- доступный объем памяти одного контроллера, Гбайт	128
- доступный объем локальной памяти микросхемы, Тбайт	1 (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)
- доступный объем памяти микросхемы, Тбайт	4 (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)
- пиковая пропускная способность одного контроллера, Гбайт/с	19
- суммарная пиковая пропускная способность восьми контроллеров, Гбайт/с	152
Контроллер межпроцессорных каналов	3
- пиковая пропускная способность одного контроллера, в одном направлении, Гбайт/с	8
- полная пиковая пропускная способность одного контроллера, Гбайт/с	16
- суммарная пиковая пропускная способность трех контроллеров, Гбайт/с	48
Контроллер канала ввода-вывода	1
- пиковая пропускная способность в одном направлении, Гбайт/с	8
- полная пиковая пропускная способность контроллера, Гбайт/с	16

Наименование параметра	Значение параметра
Контроллер PCI Express 1x16/2x8	2
- пиковая пропускная способность одного контроллера, Гбайт/с	32
- суммарная пиковая пропускная способность двух контроллеров, Гбайт/с	64
Контроллер Ethernet 1G	2
Контроллер Ethernet 10G	1
Контроллер SATA 3.0	4 порта
- частота обмена, Гбит/с	6
Контроллер USB 3.0/2.0	4+4 порта
- частота обмена USB 3.0, Гбит/с	5
- частота обмена USB 2.0, Мбит/с	480
Контроллер I2C/IPMB	5 каналов
Контроллер SPI	1
Контроллер HDA	1
Контроллер RS-232	2
Контроллер канала управления вентиляторами	2
Контроллер I2C Slave	1
Контроллер GPIO/MPV	16 линий
Контроллер EPIC	1
Контроллер IOEPIC	1
Системный таймер	1
Сторожевой (watchdog) таймер	1
Контроллер SPMC	1
Напряжения питания, В	0,8; 1,2; 1,5; 1,8; 3,3
Динамическая потребляемая мощность, Вт, не более	
- 1891ВМ03А8	170,0
- 1891ВМ03В8	163,0
Условное обозначение корпуса	4804 HFC BGA

Наименование параметра	Значение параметра
Количество элементов в схеме электрической	12000000000

_____

* Пиковая производительность микросхем 1891ВМ03В8 составляет 0,9 от пиковой производительности микросхем 1891ВМ03А8

2 Функциональные узлы микропроцессора

2.1 Процессорное ядро Core

2.1.1 Структурная схема

Процессорное (вычислительное) ядро имеет двухкластерную организацию. Каждый кластер содержит три арифметико-логических канала, локальный блок регистрового файла и локальный блок кэша данных первого уровня. Кластерная организация позволила уменьшить количество портов доступа в блоках регистрового файла и кэша данных, а также сократить задержки передачи сигналов между устройствами.

Система команд вычислительного ядра «Эльбрус» построена по принципам архитектуры широкого командного слова. Она однозначно определяет набор устройств вычислительного ядра, но не ограничивает их внутреннюю организацию и пропускную способность. Набор операций, предоставляемый системой команд, не имеет сколько-нибудь выраженной специализации, что позволяет использовать вычислительное ядро для создания универсальных микропроцессоров. Он включает скалярные целочисленные операции над данными форматов 4 и 8 байтов и скалярные вещественные операции над данными форматов 4 (single precision), 8 (double precision) и 10 (extended precision) байтов. Для массивных вычислений также предусмотрены SIMD-операции над 8- и 16-байтовыми «упаковками» из целочисленных данных форматов 1, 2, 4 и 8 байтов или вещественных данных форматов 4 и 8 байтов.

Для вычислительного процессорного ядра с архитектурой широкого командного слова компилятор выполняет планирование параллельного выполнения программы и составляет последовательность широких команд, каждая из которых состоит из различных наборов операций типа сложения, умножения, деления, логических операций, обращений в память по чтению и записи и пр. Все простые команды (далее по тексту операции), составляющие одну широкую команду,

дешифрируются и выдаются на выполнение одновременно. В дополнение к системе команд, для каждой версии процессорного ядра компилятор имеет информацию о структуре и временных характеристиках связей между исполнительными устройствами. Это позволяет в случае такой необходимости создавать код, оптимизированный для конкретной модели, но никак не отменяет возможности создавать универсальный код, достаточно эффективно исполняющийся на всех моделях.

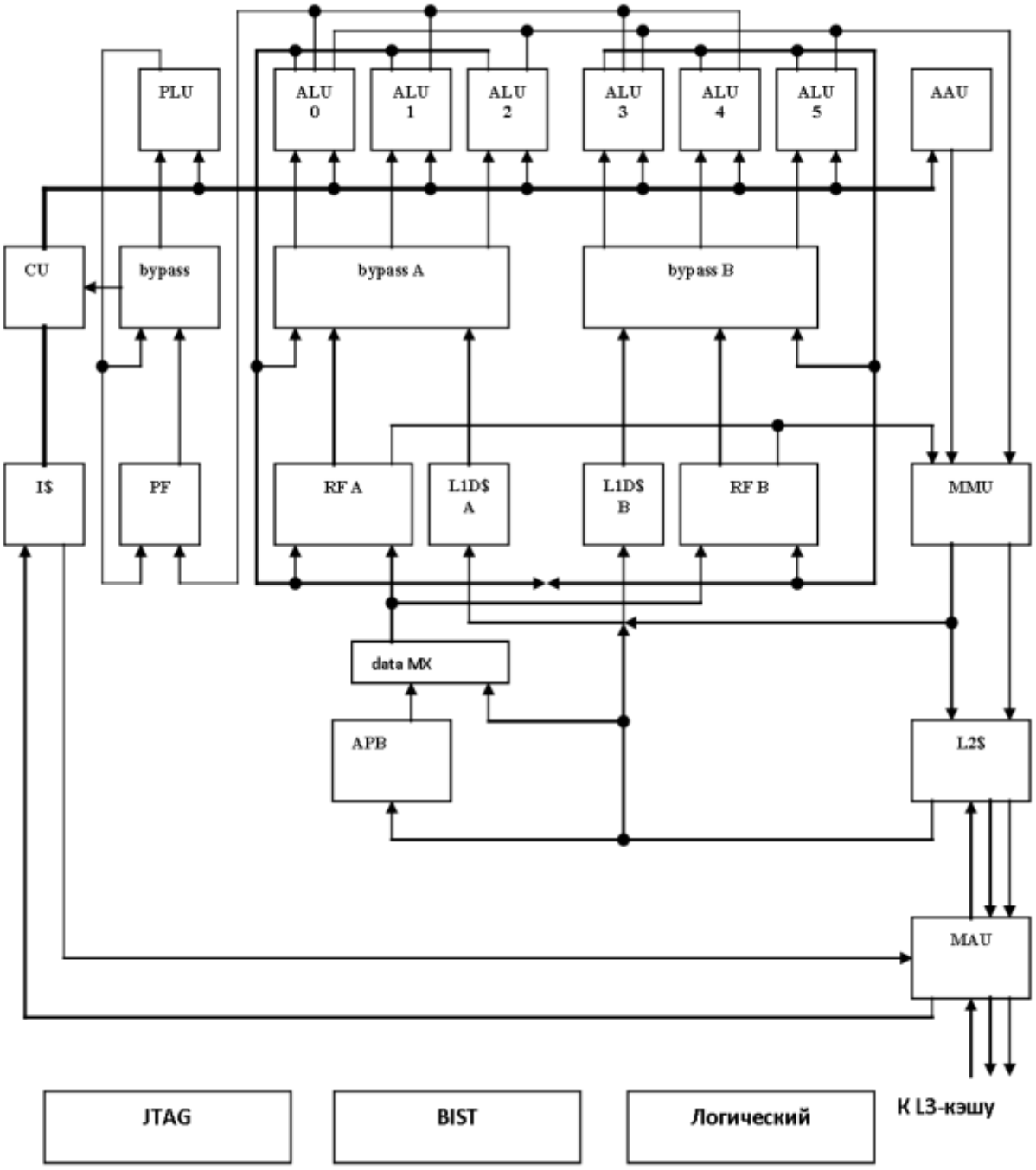
Параллельная структура процессорного ядра Эльбрус позволяет в различных сочетаниях дешифрировать в каждом такте:

- до 12 арифметико-логических операций;
- до четырех операций обращений в память (до четырех – по чтению, до двух – по записи);
- до трех операций обработки булевских значений;
- до четырех операций формирования 32-разрядных литералов (или до двух 64-разрядных);
- условную передачу управления;
- операцию приращение счетчика цикла;
- до двух групповых операций подкачки элементов массива;
- до четырех операций продвижения адресов массивов;
- до четырех операций считываний подкаченных элементов массива;
- до 6 операций вычислений условий, управляющих условным выполнением команд.

На рисунке 2.1 представлена структурная схема процессорного (вычислительного) ядра микропроцессора и секция кэша второго уровня L2$. Процессорное ядро содержит следующие устройства:

- буфер команд I$;
- устройство управления CU;
- устройство логических предикатов PLU;
- шесть арифметико-логических каналов ALC 0, 1, 2, 3, 4, 5;
- устройство обращения к массивам AAU;

- устройство байпаса логических предикатов bypass;



Устройства ALU 0, 1, 2, bypass A, RF A и L1$ A образуют кластер A.
Устройства ALU 3, 4, 5, bypass B, RF B и L1$ B образуют кластер B.

Рисунок 2.1 – Процессорное ядро Core

- два байпаса данных bypass A и bypass B;
- предикатный файл PF;
- два блока регистрового файла RF A и RF B;
- два блока кэша данных первого уровня L1D$ A и L1D$ B;
- устройство управления памятью MMU;
- коммутатор данных data MX;
- буфер предварительной подкачки APB;
- кэш второго уровня L2$;
- устройство обращения в память MAU;
- контроллер JTAG;
- контроллер BIST;
- распределенный логический анализатор.

Буфер команд IB предназначен для подкачки из памяти, хранения и выдачи в устройство управления программного кода. Буфер команд содержит кэш команд I$ объемом 128 Кбайт. Кэш команд является частично ассоциативным (4 колонки по 128 строк, блок данных в колонке - 256 байтов) и содержит широкие команды в упакованном формате, подобно тому, как они хранятся в памяти. Он имеет однопортовую организацию, и каждый такт может принимать 32 байта из памяти или выдавать до 256 байт программного кода для основной и 3 подготавливаемых для передачи управления ветвей программы.

Запрос на подкачку фрагмента кода 256 байт выдается в устройство памяти MAU, которое обеспечивает подкачку кода в I$ блоками по 32 байта.

Устройство управления CU выполняет выборку из кэша команд и распаковку широких команд из четырех программных потоков (по одной широкой команде из основного потока и трех, подготавливаемых для передачи управления). Каждая широкая команда имеет длину в упакованном виде до 64 байт (до 512 бит) и в распакованном виде 136 байт (1088 бит).

Устройство управления CU выдает на исполнение одну распакованную широкую команду из одного из четырех потоков (из основного потока или одного из трех подготовленных потоков при наличии в основном потоке операции передачи

управления с реализовавшимся условием перехода). Широкая команда в распакованном виде имеет позиционное представление в виде набора полей, в котором каждое отдельное поле содержит определенную операцию для выполнения в кластерах A и B, в предикатном файле PF и устройстве обращения к массивам AAU.

Суммарно распакованная широкая команда может содержать до 20 адресов считывания операндов, до 10 адресов записи результатов операций и до четырех адресов обращения в память.

Регистровый файл RF реализован как два отдельных блока RF A и RF B, по одному в каждом кластере. Оба блока регистрового файла содержат одни и те же данные. Каждый блок регистрового файла состоит из 256 регистров, каждый регистр имеет 136 разрядов. Регистровый файл RF является общим для операций над целыми числами и с плавающей запятой. Каждый блок регистрового файла имеет 10 физических портов чтения и 10 физических портов записи, что даёт суммарно 20 логических портов чтения и 10 логических портов записи. Данные записываются параллельно в оба блока регистрового файла, что обеспечивает их когерентность. Регистровый файл имеет конвейерную схему и использует одни и те же порты обращения для чтения и записи со сдвигом на полтакта. Большой размер регистрового файла позволяет хранить большое число промежуточных результатов и локальных переменных и, соответственно, значительно уменьшить число обращений к оперативной памяти.

При процедурных вызовах и возвратах требуется значительное время, чтобы сохранить и восстановить большое число регистров. По этой причине, в процессорном ядре реализован механизм регистрового окна изменяемого размера с регистровыми базами текущей процедуры, которые указывают на начало областей текущей процедуры в регистровом файле и предикатном файле. Реальный физический регистровый адрес вычисляется путем сложения адреса операнда из команды и значения регистровой базы текущей процедуры. При вызове процедуры или возврате из процедуры, окно перемещается за счет установки нового значения регистровой базы процедуры.

Предикатный файл PF хранит предикатные значения, выработанные операциями сравнения и логическими операциями над предикатами. С помощью предикатных значений задается условное выполнение операций и условия передачи управления для условных переходов.

Устройство логических предикатов PLU выполняет операции считывания логических предикатов из предикатного файла PF и направляет их в устройство управления для условного выполнения операций в арифметико-логических каналах ALC или операций передачи управления. Кроме того, устройство PLU выполняет логические операции формирования вторичных предикатов, используя в качестве операндов предикаты из предикатного файла PF. Результат таких операций может быть занесен в предикатный файл PF или использован для управления условным выполнением операций в каналах ALC или операции передачи управления. Необходимо заметить, что для архитектуры широкого командного слова наличие механизма условного выполнения отдельных операций в команде является критически важным для эффективного использования вычислительных ресурсов и, следовательно, достижения высокой производительности. Регистровый файл предикатов содержит 32 одноразрядных предикатных регистра, имеет 11 портов чтения и 7 портов записи, что позволяет бесконфликтным образом определять условия выполнения команд в устройствах ядра и формировать новые значения предикатов.

Числовые арифметико-логические устройства организованы в виде 6 арифметико-логических каналов. Проблемы физической реализации потребовали разделения устройств целочисленных и вещественных операций, а также разделения каналов на 2 кластера по 3 канала. Арифметико-логические каналы работают параллельно, и каждый выполняет определенный набор арифметических и логических команд. Четыре канала также содержат блоки вычисления адресов для операций обращения в память. Каждый арифметико-логический канал может получать операнды из регистрового файла, предикатного файла или через байпас, как результаты недавнего выполнения команд своего канала и других арифметико-логических каналов.

Устройство байпаса позволяет использовать результаты выполнения операций как операнды в последующих операциях до реальной записи этих результатов в регистровый файл. Каждый кластер имеет отдельный блок байпаса, который передает результаты своих арифметико-логических каналов на их входы и входы арифметико-логических каналов соседнего кластера. Кроме того, байпас также позволяет использовать результаты считывания из кэша данных L1D$ до их записи в регистровый файл.

Наличие двух кластеров в структуре ядра микропроцессора с локальными блоками регистрового файла RF и кэша данных L1D$, и комплектом арифметико-логических каналов в каждом позволило сократить количество портов доступа в блоках регистрового файла RF и кэша данных L1D$, и время передачи данных внутри каждого кластера.

Кэш-память данных первого уровня L1D$ имеет объем 64 Кбайт и частично ассоциативную организацию (4 колонки по 256 строк, размер блока данных –

32 байта). Она имеет 4 порта и может выполнять до 4 считываний или до

2 записей в память за 1 такт (два порта для считывания и записи и два порта только для считывания). В кэше L1D$ применена стратегия заведения write-through (заведение по чтению, сквозная запись).

Следующий уровень подсистемы памяти вычислительного ядра представлен кэшем второго уровня L2$, который является универсальным, то есть предназначенным для хранения, как данных, так и команд. Кэш L2$ имеет объем 1 Мбайт и организован в виде 4 банков с интерливингом по разрядам адреса, что позволяет обслуживать до 4 запросов в каждом такте, оставаясь при этом однопортовым. Каждый банк имеет частично ассоциативную организацию (4 колонки, размер блока данных — 64 байта).

Интерфейсным блоком вычислительного ядра является устройство обращения в память MAU. Оно преобразует внутреннее представление операций обращения в память в последовательность системных протокольных транзакций. Во время исполнения операций их атрибуты хранятся в специализированных буферах: для чтений — в буфере LDB объемом в 64 позиции, для записей — в

буфере STB объемом в 32 позиции. Ширина интерфейсных шин данных составляет 32 байта по чтению и 32 байта по записи.

Особенностью архитектуры «Эльбрус» является устройство AAU, предназначенное для предварительной подкачки элементов массивов в цикле. Оно может быть запрограммировано на предподкачку до 64 разных массивов. AAU имеет двухканальное устройство формирования адресов, которое поочередно посылает запросы предварительной подкачки указанных в программе массивов. Для временного хранения подкаченных данных предусмотрен буфер APB объемом 8 Кбайт, 2-канальный по записи и 4-канальный по чтению. Пространство APB программно разбивается на области по числу подкачиваемых массивов, и каждая область имитирует буфер FIFO ("первым пришел, первым вышел"). Предварительно загруженные данные временно хранятся в буфере APB, и затем пересылаются в регистровый файл по специальным командам, замещающим обычные операции чтения из памяти. AAU может работать асинхронно по отношению к основной программе, что позволяет демпфировать время и динамику обращений в память. Кроме того, AAU старается группировать запросы к элементам массива в более крупные блоки, что сокращает нагрузку на коммуникационные шины и сокращает время доступа.

Функционирует процессорное ядро следующим образом. Буфер команд IB вызывает программный код блоками по 256 байтов для основного и подготавливаемых потоков. Для основного потока обращение в память инициируется отсутствием необходимого блока в кэше команд I$, для подготавливаемого – операциями подготовки передачи управления из основного программного потока. Вызов программного кода осуществляется через устройство обращения в память MMU, которое проверяет наличие запрашиваемого блока в кэше второго уровня L2$ и, при его отсутствии там, обращается в устройство обращения в память MAU. Программный код поступает в IB блоками по 64 байта.

Буфер команд IB выдает каждый такт в устройство управления CU блок программного кода 256 байт, причем, это может быть программный код только

основного потока или смесь, состоящая фрагментов по 64 байта основного и подготавливаемых потоков.

Устройство управления CU выполняет распаковку широких команд всех четырех потоков и выбирает одну из них для дальнейшего выполнения. Основной поток выбирается при отсутствии операции передачи управления, один из подготавливаемых – по результатам операции передачи управления в предшествующей широкой команде. Затем CU преобразует относительные адреса операндов широкой команды в абсолютные адреса по регистровому файлу, проверяет условные признаки широкой команды и распределяет операции широкой команды в арифметико-логические каналы ALC 0 – 5, кэш данных L1D$, буфер устройство обращения к массивам AAU и устройство логических предикатов PLU. Устройство управления CU также проверяет условия выдачи команд, такие как прерывания и ситуации блокировки выдачи команд, а также наличие операндов в регистровом файле и байпасах. После устройства управления операции, составляющие широкую команду, выдаются в конвейеры различных устройств микропроцессора, где завершается их параллельное и уже независимое друг от друга выполнение, в общем случае в разное время.

Вещественные исполнительные устройства в каналах ALC начинают работу на такт позже, чем целые исполнительные устройства. Операнды для вещественных операций вначале поступают на регистры операндов целых исполнительных устройств и затем уже на регистры операндов вещественных устройств. Такая организация объясняется невозможностью в течение такта обеспечить передачу данных из регистрового файла сразу двум потребителям. Поэтому выбрана последовательная схема.

Процессорное ядро имеет сбалансированную структуру, которая обеспечивает непрерывный поток широких команд. Это свойство позволяет компилятору планировать потактное бесконфликтное выполнение программы в процессорном ядре, обеспечивающее максимальную загрузку его устройств. Такое расписание базируется на возможности определить на стадии компиляции время выполнения всех простых команд и спланировать их оптимальное распределение в последова-

тельности широких команд. В широких командах явным образом задается, в каком кластере и каком канале этого кластера выполнять ту или иную операцию.

Локальными нарушениями расписания выполнения могут быть только динамически возникающие события в программе, не поддающиеся анализу на стадии компиляции, такие как, например, промах в кэше данных L1$, обращение в оперативную память и динамические ситуации выполнения некоторых операций (например, вещественных операций, время выполнения которых зависит от операндов, вычисляемых при выполнении программы). Эти динамические ситуации могут привести к неготовности операндов для очередной широкой команды и, тем самым, приостановить выдачу команд из устройства управления. После завершения ожидания операндов расписание выполнения программы восстанавливается.

Арифметико-логические каналы ALC получают простые команды из устройства управления CU и операнды из регистрового файла RF или через байпас, выполняют заданные операции и заносят результаты в регистровый файл RF. Конвейер выполнения каждого арифметико-логического канала имеет пропускную способность одна операция за такт для большинства операций за исключением операции деления.

Виртуальный адрес операций обращения в память формируется в арифметико-логических каналах ALC 0, 1, 3, 5. Затем выполняются обращения в кэш первого уровня L1D$. Кэш первого уровня L1D$ использует виртуальную адресацию и стратегию заведения при считывании и записи насквозь. При попадании в кэш L1D$ считываемые значения поступают в регистровый файл RF, но прежде могут быть использованы в качестве операндов через байпас.

При отсутствии данных в кэше L1D$ запрос передается в устройство обращения в память MMU, где в кэше таблицы страниц TLB выполняется преобразование виртуального адреса в физический. Затем выполняется обращение в кэш второго уровня L2$ и, при отсутствии там, обращение в кэш-память следующего уровня L3$. Виртуальный адрес имеет длину 48 разрядов; длина физического адреса — 48 разрядов. Эта разрядность поддерживается всем трактом обращения в память.

При обращении в память по записи записываемое значение считывается непосредственно из регистрового файла RF в кэш L1D$ и далее сопровождает адрес до завершения операции. Кэш первого уровня L1D$ использует стратегии "заведение при считывании" и "запись насквозь". Поэтому записываемые данные заносятся в кэш L1D$ только при попадании в него и всегда передаются в кэш L2$. Кэш второго уровня L2$ использует стратегию "отложенная запись", то есть, при чтении и записи в кэше L2$ всегда резервируется блок данных, а запись в память выполняется лишь при замещении в нем блока. Аналогичная стратегия реализована и в кэше L3$.

Устройство обращения к массивам AAU выполняет предварительную подкачку элементов массивов из памяти во внутренний буфер устройства APB. Эта работа выполняется отдельным фрагментом программы, который загружается в устройство AAU из устройства управления CU перед началом циклического участка программы обработки массивов и далее асинхронным образом выполняет подкачку элементов массивов в свой внутренний буфер устройства APB. Основной программный поток передает подкаченные элементы массивов из внутреннего буфера устройства APB в регистровый файл RF для использования в операциях обработки.

Операции сравнения, выполняемые в арифметико-логических каналах ALC, формируют логические предикаты и записывают их предикатный файл PF. Устройство логических предикатов PLU формирует вторичные предикаты, используя в качестве операндов предикаты из предикатного файла PF. Устройство управления CU использует предикаты как условия для выполнения операций передачи управления, а также большого набора других операций. В их число входят, например, арифметико-логические операции, операции пересылки и операции обращения в память.

В состав процессорного ядра микропроцессора входят шесть арифметико-логических каналов ALC 0 - 5: по три в каждом кластере. Арифметико-логические каналы работают параллельно и выполняют почти одинаковый набор арифметических и логических операций. Каждый арифметико-логический канал

может получать операнды из ячеек регистрового файла или, как результаты недавнего выполнения операций, из других арифметико-логических каналов через байпас.

Набор операций, выполняемых в каждом канале ALC, приведен в таблице 2.1.

Таблица 2.1 - Набор операций в каналах ALC

Устройство	Операция	Каналы					
		0	1	2	3	4	5
Цел.+, логика, сдвиг	Двухоперанд. операции	x	x	x	x	x	x
	Трёхоперанд. операции		x			x	
	Compare	x	x		x	x	
load/store (адрес)	load/store	x		x	x		x
Преобразование адресов	Преобразование адресов	x	x		x	x	
	Запись и чтение рег. состояний	x					
Плавающее сложение	Add/Sub d/s/pack	x	x	x	x	x	x
	Add/Sub extended	x	x		x	x	
	Convert d/s/pack/extended	x	x		x	x	
	Compare d/s/extended	x	x		x	x	
Умножение	Int	x	x		x	x	
	Fp d/s /pack	x	x	x	x	x	x
	Fp Extended						
Fp-трёхоперанд. операции	Fp d/s/pack	x	x	x	x	x	x
Деление	Int						x
	Fp-divide s/d						x
	Remainder						
	Square root s/d						x

Устройство	Операция	Каналы					
		0	1	2	3	4	5
	Fp Extended						x
MMX	Shift		x			x	
	Multiply		x			x	
	Add/subtract	x			x		
	Compare	x	x		x	x	

В первой колонке перечислены исполнительные устройства для выполнения групп операций. Во второй колонке представлены группы операций, выполняемых устройствами, определенными в первой колонке. В третьей колонке показаны арифметико-логические каналы, выполняющие представленные операции. Трехоперандные комбинированные операции состоят из двух последовательно выполняемых операций, причем первая операция использует два операнда, а вторая - третий операнд и результат первой операции.

Каналы ALC имеют отдельные устройства для выполнения целочисленных операций и операций с плавающей запятой. Исключением являются целочисленные операции умножения и деления, для выполнения которых используются соответствующие блоки в исполнительном устройстве с плавающей запятой. Короткие целочисленные операции выполняют все каналы ALC. Скалярные обращения в память по считыванию выполняют каналы ALC 0, 2, 3 и 5, по записи – каналы ALC 2 и 5. Операции над адресными данными (дескрипторами, указателями и пр.) выполняют каналы ALC 0, 1, 3 и 4. Операции типа сложения с плавающей запятой выполняют все каналы ALC. Операции умножения выполняют каналы ALC 0, 1, 3 и 4, операцию деления - только канал ALC 5. Трехоперандные операции с плавающей запятой выполняют каналы все каналы ALC. Мультимедийные операции MMX типа умножения и сдвига выполняют каналы ALC 2 и 4, типа сложения - каналы ALC 0 и 3.

Устройство байпаса позволяет использовать результаты выполнения операций как операнды в последующих операциях до реальной записи этих результатов в регистровый файл. Каждый кластер имеет отдельный блок байпаса, который передает результаты своих арифметико-логических каналов на их входы

и входы арифметико-логических каналов соседнего кластера. Кроме того, байпас также позволяет использовать результаты считывания из кэша данных L1D$ до их записи в регистровый файл.

Процессорное ядро функционирует следующим образом. Устройство команд IU вызывает программный код фрагментами по 256 байтов для основного и подготавливаемых потоков. Для основного потока обращение в память инициируется отсутствием необходимого программного кода в кэше команд I$, для подготавливаемого потока – операциями подготовки передачи управления из основного программного потока. Вызов программного кода осуществляется через устройство обращения в память MMU, которое проверяет наличие запрашиваемого блока программного кода в кэше второго уровня L2$ и, при его отсутствии там, обращается в устройство обращения оперативную память MAU. Программный код поступает в кэш команд I$ квантами по 64 байт.

Устройство команд IB выдает каждый такт в устройство управления CU блок программного кода 256 байт, причем, это может быть программный код основного потока или смесь, состоящая фрагментов по 64 байт основного и подготавливаемых потоков.

Устройство управления CU выполняет распаковку широких команд всех четырех потоков параллельно и выбирает одну из них для дальнейшего выполнения. Основной поток выбирается при отсутствии передачи управления, один из подготавливаемых – по результатам операции передачи управления в предшествующей широкой команде. Затем CU преобразует относительные адреса операндов широкой команды в абсолютные адреса по регистровому файлу RF, проверяет условные признаки широкой команды и распределяет операции широкой команды в арифметико-логические каналы ALC 0 – 5, устройство обращения к массивам AAU и устройство логических предикатов PLU. Устройство управления CU также проверяет условия выдачи команд, такие как прерывания и ситуации блокировки выдачи команд, а также наличие операндов в регистровом файле и байпасах. Из устройства управления операции, составляющие широкую команду, выдаются в конвейеры различных устройств микропроцессора, где завершается

их параллельное и независимое друг от друга выполнение, в общем случае в разное время.

Исполнительные устройства с плавающей запятой в каналах ALC начинают работу на такт позже, чем целые исполнительные устройства. Операнды для операций с плавающей запятой поступают на регистры операндов целых исполнительных устройств и затем уже на регистры операндов устройств с плавающей запятой. Такая организация объясняется невозможностью в течение такта обеспечить передачу данных из регистрового файла сразу двум потребителям. Поэтому, выбрана последовательная схема.

Процессорное ядро Core имеет сбалансированную структуру, которая обеспечивает непрерывный поток широких команд. Это свойство позволяет компилятору планировать потактное бесконфликтное выполнение программы в процессорном ядре, обеспечивающее максимальную загрузку его устройств. Такое расписание базируется на возможности определять на стадии компиляции время выполнения всех простых команд (операций) и планировать их оптимальное распределение в последовательности широких команд. В широких командах явным образом задается, в каком кластере и в каком канале этого кластера выполнять ту или иную операцию.

Локальными нарушениями расписания выполнения могут быть только события, динамически возникающие при выполнении операций и не поддающиеся анализу на стадии компиляции, такие как, например, промах в кэше данных L1D$, обращение в оперативную память и динамические ситуации выполнения некоторых операций (например, операций с плавающей запятой, время выполнения которых зависит от операндов, вычисляемых при выполнении программы). Эти динамические ситуации могут привести к неготовности операндов для очередной широкой команды и, тем самым, приостановить выдачу команд из устройства управления. После завершения ожидания операндов расписание выполнения программы восстанавливается.

Арифметико-логические каналы ALC получают операции из устройства управления CU и операнды из регистрового файла RF или через байпас, выпол-

няют заданные операции и заносят результаты в регистровый файл RF. Конвейер выполнения каждого арифметико-логического канала имеет пропускную способность одна операция за такт для большинства операций за исключением операции деления.

Виртуальный адрес операций обращения в память формируется в арифметико-логических каналах ALC 0, 1, 3, 5. Затем выполняются обращения в кэш первого уровня L1D$. Кэш первого уровня L1D$ использует виртуальную адресацию и стратегию заведения при считывании и записи насквозь. При попадании в L1D$ считываемые значения поступают в регистровый файл, но прежде могут быть использованы в качестве операндов через байпас.

При отсутствии данных в L1D$ запрос передается в устройство управления памятью MMU, где в кэше таблицы страниц TLB выполняется преобразование виртуального адреса в физический. Затем выполняется обращение в кэш второго уровня L2$ и, при отсутствии там, обращение в кэш третьего уровня L3 кэш и контроллер системных обменов SIC.

При обращении в память по записи записываемое значение передается непосредственно из регистрового файла в L1D$ и далее сопровождает адрес до завершения операции. Кэш первого уровня L1D$ использует стратегии "заведение при считывании" и "запись насквозь". Поэтому, записываемые данные заносятся в L1D$ только при попадании в него и всегда передаются в L2$. Кэш второго уровня L2$ использует стратегию "отложенная запись" (Write-Back), то есть, при чтении и записи в L2$ всегда резервируется строка для блока данных 64 байта, а запись в память выполняется лишь при замещении в нем блока.

Устройство обращения к массивам AAU выполняет предварительную подкачку элементов массивов из памяти в буфер предподкачки APB. Эта работа выполняется отдельным фрагментом программы, который загружается в AAU из устройства управления CU перед началом циклического участка программы обработки массивов и далее асинхронным образом выполняет подкачку элементов массивов в буфер APB. Основной программный поток передает подкаченные

элементы массивов из буфера APB в регистровый файл RF для использования в операциях обработки.

Операции сравнения, выполняемые в арифметико-логических каналах ALC, формируют логические предикаты и записывают их в предикатный файл PF. Устройство логических предикатов PLU формирует вторичные предикаты, используя в качестве операндов предикаты из предикатного файла PF. Устройство управления CU использует предикаты как условия для выполнения операций передачи управления, а также большого набора других операций. В их число входят, например, арифметико-логические операции, операции пересылки и операции обращения в память.

2.1.2 Организация работы конвейера процессорного ядра

Конвейер процессорного ядра Core обеспечивает параллельное и бесконфликтное выполнение программы при оптимальном планировании ее выполнения на стадии компиляции.

Устройство буфера команд IB каждый такт может выдавать 256 байт программного кода в устройство управления CU.Устройство управления CU может дешифрировать каждый такт широкую команду длиной до 64 байт. Таким образом, пропускная способность буфера команд IB может обеспечить, в среднем, выдачу в устройство управления всех 4 ветвей программы, подготавливаемых для выполнения.

Устройство управления CU каждый такт может выдавать распакованную широкую команду из основного синхронного потока для выполнения в параллельных ветвях конвейера и одновременно загружать программу в устройство обращения к массивам AAU, которая затем будет выполняться в нем асинхронным образом по отношению к основному программному потоку. Распакованная широкая команда из основного программного потока содержит набор операций, имеющих в общем случае следующий формат:

opcode	src1	src2	src3	src4	rd	literal

opcode	- код операции;
src1, src2, src3, src4	- адреса операндов или короткие константы;
rd	- адрес результата;
literal	- константа из кода программы.

Количество полей адресов операндов может меняться в зависимости от типа операции:

src1	- одноместная операция;
src1, src2	- двухместная операция;
src1, src2, src3	- трехместная комбинированная операция или двухместная операция с одним из операндов формата квадро;
src1, src2, src4	- операция записи;
src1, src2, src3, src4	- операция записи с одним из операндов формата квадро, для считывания которого необходимо два порта регистрового файла.

Записываемое в память значение в операциях записи задается адресом операнда src4. Этот адрес подается на отдельный порт чтения в регистровом файле с задержкой в 1 такт, пока вычисляется адрес обращения в память в адресном сумматоре в исполнительном устройстве.

Каждый порт чтения в регистровом файле RF обеспечивает чтение операнда формата двойного слова dw и расширенного вещественного 80-разрядного формата EX. Операции в арифметико-логических каналах ALC, в зависимости от типа, используют 1, 2, 3 или 4 порта чтения в регистровом файле RF. Два порта чтения необходимо для простых двухместных операций, три - для комбинированных операций и операций записи, и четыре - для операций вычисления адресов, использующих операнды квадро формата и для чтения которых используются два

порта чтения в регистровом файле RF. Таким образом, в соответствии с распределением операций по каналам, приведенным в таблице 1.1, ALC 0, 1, 3 и 4 используют по три порта чтения в RF, а ALC 2 и 5 - по четыре порта. В сумме это составляет 20 портов чтения, имеющихся в регистровом файле RF, и обеспечивает бесконфликтное чтение операндов из RF.

Шесть каналов ALC используют 6 портов записи в регистровый файл RF. Из них 4 порта используются также для данных, поступающих из кэша данных L1D$. Еще 4 порта записи используются для записи данных, поступающих из кеша второго уровня L2$, оперативной памяти и буфера предварительной подкачки APB. В сумме это составляет 10 портов записи, имеющихся в регистровом файле RF.

Совмещение портов записи для каналов ALC и кэша данных L1D$ не вызывает конфликтов, так как операция считывания начинается как операция в одном их каналов ALC 0, 2, 3 или 5 и по времени выполнения совпадает с обычной короткой арифметической операцией, что и обеспечивает ее бесконфликтное прохождение через выделенный порт записи.

Совмещение портов записи для L2$, оперативной памяти и APB может приводить к конфликтам, которые разрешаются с использованием соответствующих буферов.

Канал ALC каждый такт может принимать для выполнения одну простую или одну комбинированную операцию. ALC содержит набор целых и вещественных исполнительных устройств с различными временами выполнения операций, которые могут иметь конфликты по выдаче своих результатов в назначенный порт записи в RF. Для устранения конфликтов между короткими операциями с длительностями выполнения в исполнительном устройстве 1 и 2 такта, времена выполнения однотактных операций приведены к двум тактам и дополнительные станции конвейера включены в систему байпаса, что позволяет использовать их результаты без этой дополнительной задержки.

Конфликт по использованию общего порта записи между короткими двухтактными операциями и более длинными (4 такта и больше) разрешен путем

предоставления коротким операциям большего приоритета при выдаче результатов в RF, а также введением буфера результатов для более длинных операций. Число буферных регистров достаточно, чтобы принять результаты всех операций, находящихся в длинном конвейере. Таким образом, короткие операции проходят конвейер без дополнительной задержки, но при этом количество длинных операций в канале не увеличивается. Они все могут разместить свои результаты в буфере результатов. Буфер результатов разгружается, когда в канал не поступают короткие операции.

Применение комбинированных операций, состоящих из двух отдельных операций, выполняемых последовательно, повышает производительность микропроцессора. Целые и вещественные комбинированные операции имеют различную организацию выполнения в каналах ALC. Целые комбинированные операции состоят из двух операций, для выполнения каждой из которых требуется 1 такт. В каналах ALC и имеются два исполнительных устройства целых коротких операций, включенных последовательно в два этажа для выполнения целых комбинированных операций. Это позволяет выполнить целую комбинированную операцию за 2 такта. Некомбинированные целые операции передают свои результаты в обход исполнительного устройства второго этажа.

Комбинированные вещественные операции выполняются двумя вещественными исполнительными устройствами, соединяемых последовательно на время выполнения таких операций, кроме комбинированной операции умножения и сложения. Для этой операции в каждом канале имеется устройство сложения второго этажа.

Обращения по считыванию из каналов ALC 0, 2, 3 и 5 в кэш данных L1D$ не порождает конфликтов. Если данные найдены в нем (hit), то они беспрепятственно поступают в соответствующий регистр RF по четырем портам записи, выделенным этим каналам. В случае промаха в L1D$ (miss) запрос передается DTLB для формирования физического адреса и далее в L2$ и MAU, если был промах в L2$.

Обращения по записи из каналов ALC 2 и 5 в кэш данных L1D$ требует двух тактов работы кэша. В первом такте выполняется поиск в кэше и, в случае hit, в следующем такте может быть произведена запись. Причем запись должна быть продублирована в обоих блоках L1D$ (в обоих кластерах). Для записи в порт для ALC 2 дублирующая запись выполняется аппаратно через входной порт для ALC 3, а для записи по каналу ALC 5 - через входной порт для ALC 0. Здесь используется метод отложенной записи, когда реально запись производится при благоприятных условиях. Такими условиями для L1D$ являются отсутствие входных запросов.

Запись данных в L1D$ производится также при выполнении операций считывания с промахом в L1D$. В этом случае запрашиваемые данные блоком в 32 байта поступают из L2$ или памяти.

Буфер таблицы страниц DTLB имеет 4-портовую организацию, работает параллельно с кэшем данных L1D$ и выполняет преобразования виртуальных адресов в физические. Он передает запросы с физическими адресами в кэш второго уровня L2$ в случае промаха в L1D$ и попадания (hit) в нем самом. В случае промаха в DTLB запрос обрабатывается устройстве таблицы страниц TLU, которое выполняет обращение в таблицу страниц PT и загружает в DTLB отсутствующую строку таблицы данных. Промах в DTLB заставляет устройство управления CU отменить в конвейере выполнение всех широких команд, начиная с широкой команды, следующей за вызвавшей промах в DTLB, и заново повторить их выполнение. Операция, вызвавшая промах в DTLB, дорабатывается устройством MMU после вызова в DTLB требуемой строки таблицы страниц PT.

Обращения в кэш второго уровня L2$ поступают в случае промаха в L1D$ операций считывания из памяти и, безусловно, при выполнении операций записи в память, так как L1D$ реализует стратегию "запись насквозь". Устройство L2$ имеет 4-канальную организацию, состоит из 4 банков, к которым, при отсутствии конфликтов по адресам, могут производиться параллельно 4 обращения.

Данные из L2$ поступают в кэш данных L1D$ и регистровый файл RF. Причем в L1D$ передается блок данных 32 слова, а регистровый файл RF - только запрашиваемое значение.

Устройство L2$ содержит общий входной буфер на 8 запросов для разрешения конфликтов обращений к банкам и по-банковые буфера на 7 запросов каждый для накопления запросов, поступающих из нижестоящих станций конвейера.

В случае промаха в L2$ запросы передаются в устройство обращения в память MAU. Для согласования работы с устройством обращения в память L2$ имеет выходные буфера на 3 запроса по считыванию из памяти и 3 запроса на запись в память. Устройство MAU выдает запросы в кэш третьего уровня L2$.

Каждый банк L2$ содержит буфер записи STB из 4 регистров по 64 байта. Буфер STB имеет ассоциативную организацию и реализует стратегию отложенной записи. Выполнение операции записи в L2$ имеет свои особенности. Так как L2$ поддерживает стратегию "заведение при записи", то, в случае промаха необходимо загрузить из памяти блок данных размером 64 байт, наложить на него записываемое значение и затем записать модифицированный блок в L2$. Заявка по записи находится в определенном регистре буфера STB и, пока не поступил блок данных из памяти, обслуживает последующие запросы по считыванию с совпадающими адресами и накладывает запросы по записи в данный блок данных.

Устройство обращения в память MAU содержит буфер на 32 заявки по считыванию из памяти LDB и буфер на 16 заявок по записи STB. Устройство MAU обеспечивает программный порядок выдачи запросов с совпадающими адресами.

Устройство обращения к массивам AAU предназначено для обращений к массивам в циклических участках программ. Устройство AAU, выполняя предварительно загруженную в него программу, может пересылать данные из массивов в памяти в свой внутренний буфер APB и затем по командам из основного программного потока пересылать элементы массивов из буфера в регистровый файл RF. Программа для вызова данных из массивов и дескрипторы, определяющие массивы, загружаются в AAU заблаговременно по отношению к их использова-

нию. Кроме того, обращения по чтению и записи к массивам могут быть выполнены из основного программного потока, используя хранящиеся в AAU дескрипторы массивов. Такой метод работы с массивами позволяет сократить программный код в циклах, совместить предварительную подкачку массивов из памяти с собственно вычислениями, используя ранее вызванные из памяти данные, увеличить эффективность использования регистров RF, так как позволяет не занимать их на время вызова данных из памяти.

Устройство AAU содержит два блока буфера предварительной подкачки массивов APB по 2 Кбайт каждый, два блока буфера команд асинхронной программы по 32 64-разрядных регистра, память 32 дескрипторов массивов, два канала асинхронной подкачки массивов, 4 канала синхронной пересылки элементов массивов из APB в RF, два канала синхронного обращения к массивам. Синхронные каналы получают команды из основного программного потока, асинхронные - из второй ветви программы, выдаваемой параллельно с основным программным потоком. Асинхронные каналы и синхронные каналы обращения к массивам работают независимо и обращаются в DTLB и далее в L2$ и память. Каналы синхронной пересылки принимают команды из основного потока и пересылают элементы массивов из APB в регистровый файл RF через те же порты записи, которые используются для записи данных из L2$ и памяти.

Устройство управления CU контролирует выполнение операций на первых трех станциях исполняющей части конвейера в арифметико-логических каналах ALC, кэше данных L1D$ и TLB, и, в случае возникновения конфликтной ситуации, организует повторное выполнение широких команд, начиная с вызвавшей конфликт. Организуя повторное выполнение, устройство управления фактически восстанавливает расписание выполнения программы, нарушенное конфликтной ситуацией.

С этой целью распакованное (дешифрированное) представление широких команд, выполняемых на первых трех исполнительных станциях конвейера, сохраняется в специальном буфере. Вместе с широкими командами в этом буфере сохраняется состояние ряда регистров процессорного ядра на момент дешифра-

ции широкой команды, которые могут модифицироваться на этих первых трех станциях конвейера. При возникновении конфликтной ситуации начальная часть конвейера (до уровня дешифрации) блокируется, основной конвейер на первых трех исполняющих станциях гасится и в него, после разрешения конфликтной ситуации, последовательно выдаются широкие команды из буфера.

Существует ряд ситуаций, которые требуют повторения выполнения широких команд. Устройство управления CU выполняет проверки, которые определяют возможность корректного завершения выполнения текущей широкой команды. К ним относятся проверка наличия готовых операндов для всех операций текущей широкой команды, возможность выдачи операций на выполнение, для которых темп выдачи раз в 2 такта и более, доступность регистров состояния процессорного ядра, модифицируемых предшествующими широкими командами, необходимость предварительно выполнить аппаратные операции, например, откачать или подкачать стековые регистры и так далее. Конфликтные ситуации, выявленные в ходе этих проверок, могут быть по причине неоптимального планирования выполнения программы на стадии компиляции или проявиться как результат вычислений. Другие конфликтные ситуации возникают только динамически в ходе выполнения широких команд. К ним относятся такие как промах в кэше данных L1D$ и кэше таблицы страниц DTLB.

Прерывания, возникающие при выполнении команд на первых станциях исполнительной части конвейера, могут обрабатываться как синхронные. При их возникновении устройство управления CU восстанавливает на стадии дешифрации широкую команду, вызвавшую прерывание, а вместе с ней и состояние всех управляющих регистров, включая адрес этой широкой команды. Далее управление передается на процедуру прерывания, которая фиксирует адрес команды, вызвавшей прерывание, с целью последующего возврата после отработки прерывания.

Существенным свойством всех этих ситуаций является то, что они требуют повторения выполнения вызвавшей их широкой команды, когда она уже покинула стадию дешифрации.

Возможность повторения широких команд, находящихся на первых станциях исполнительной части конвейера, позволила отказаться от буферов на этих станциях. Как уже было показано выше, буферы для согласования темпов выполнения операций в устройствах появляются только на последующих станциях конвейера. К ним можно отнести выходные буферы в арифметико-логических каналах ALC для хранения результатов длинных операций, буферы в кэше второго уровня L2$ и устройстве обращения в память MAU.

2.1.3 Обзор архитектуры широкого командного слова

В процессорном ядре микропроцессора реализована архитектура широкого командного слова, которая предоставляет компилятору явные средства для использования и управления работой параллельной аппаратуры микропроцессора. Компилятор анализирует внутренний программный параллелизм и генерирует параллельный код с учетом информационных и ресурсных зависимостей. В том случае, когда информационная зависимость определяется динамически, это явным образом отображается в коде программы.

Архитектура процессорного ядра позволяет в различных сочетаниях выполнять в одном такте 12 арифметико-логических операций, четыре обращения в память, три операции обработки булевских значений, условную передачу управления, приращение счетчика цикла, две групповые операции подкачки элементов массива в буфер предварительной подкачки с продвижением адресов, четыре считывания элементов массива из буфера предварительной подкачки в регистровый файл, вычисление 6 условий, управляющих условным выполнением команд. Ограничением является максимальный размер широкой команды - до 64 байт. В результате, в одной широкой команде может содержаться не более 25 операций с данными формата 64 и не более 61 операции с упакованными данными формата 32.

2.1.3.1 Явное выражение параллелизма в языке машины

В архитектуре широкого командного слова параллельное выполнение представлено явно в коде программы. Широкая команда содержит набор операций, которые одновременно дешифрируются и параллельно выполняются, каждая в своем отдельном конвейере с известным временем выполнения. Это дает возможность компилятору выполнять практически все планирование статически, во время трансляции, определяя все информационные зависимости и учет занятости ресурсов, управлять каждым тактом. Благодаря параллельному коду (представлению программы в виде последовательности широких команд, содержащих наборы параллельно выполняемых операций), компилятор может достаточно ясно и точно (с точностью до такта) представить результаты своего планирования для случая параллельного выполнения программы.

В процессе работы аппаратура с помощью широкой команды осуществляет только запуск исполнительных устройств. Кроме того, вся организация процессорного ядра выполнена таким образом, что вся дальнейшая после запуска устройств их работа предсказуема с точностью до такта процессора. Это делает возможным освободить аппаратуру от заботы о взаимных блокировках устройств и возложить ее на компилятор.

С точки зрения программы процессор состоит из исполнительных устройств: арифметико-логических устройств, адресных регистров с аппаратурой их продвижения и устройства запуска обращений в память как вне, так и внутри цикла. Все эти устройства работают на четырех видах регистровой памяти: регистровом файле, адресных регистрах, буфере предварительной подкачки массивов и регистре предикатов. Различные устройства, работающие на одном канале, могут иметь статически известные, но различающиеся времена исполнения.

С учетом конвейерной работы всех устройств, компилятор должен следить, чтобы отсутствовали конфликты по ресурсам. В частности, например, две разные по времени операции, запущенные в одном канале не должны заканчиваться в одном и том же такте и т.д. Ошибки компилятора в планировании не нарушают

корректности, но приводят к снижению эффективности. Результаты выполнения с выходов устройств на их входы поступают:

- через регистровый файл;
- через байпасы с выходов на входы непосредственно.

Последний вариант самый быстрый. Это свойство в сочетании с универсальным набором устройств в кластерах позволяет компилятору размещать большинство критических путей на этом наиболее быстродействующем пути.

2.1.3.2 Реализация целочисленных и вещественных операций

Процессорное ядро микропроцессора имеет шесть каналов для выполнения арифметико-логических операций. Каждый канал содержит набор исполнительных устройств для выполнения целочисленных и вещественных операций. Регистровый файл является общим для хранения результатов целых и вещественных операций. В связи с этим, каждый канал имеет три выделенных порта чтения операндов и один порт записи результатов операций. Каждый такт канал может принимать одну целочисленную или вещественную операцию. Организация конвейера канала обеспечивает его постоянную готовность к приему очередной операции. Исключение составляет операция деления, для которой имеются ограничения на темп обработки.

2.13.3 Ускорение выполнения вычислений

Архитектура широкого командного слова, благодаря параллельной работе аппаратуры и глубокому анализу программы и тщательному планированию ее выполнения транслятором на стадии компиляции, позволяет значительно повысить скорость счета как скалярных, так и векторных задач. Для скалярных вычислений основным фактором, ограничивающим скорость счета таких задач, является информационная зависимость между различными операциями. Для векторных вычислений, как правило, ограничивающим факторам является не

столько информационная зависимость, сколько общий объем вычислений и недостаток вычислительных ресурсов (в основном исполнительных устройств).

Спекулятивный и условный режимы выполнения команд, а также выполнение передач управления с использованием предварительной подкачки кода в направлении передачи управления способствуют ускорению выполнения скалярного кода.

Спекулятивный режим выполнения операций позволяет компилятору перемещать выше по коду отдельные операции и фрагменты программы и выполнять их параллельно на фоне текущих вычислений, что сокращает общее время выполнения программы. Например, спекулятивное чтение из памяти скрывает задержку доступа в память на фоне выполнения текущих вычислений. Спекулятивный режим выполнения арифметических операций позволяет производить параллельно вычисления сразу по нескольким альтернативным ветвям программы на фоне вычисления условия ветвления. Спекулятивный режим не вызывает прерываний, а результаты выполнения спекулятивных операций могут использоваться после вычисления условий их выполнения.

Условный режим выполнения операций часто заменяет условную передачу управления. Условный режим выполнения операций позволяет управлять выполнением операции по заданному и ранее вычисленному условию (аналогично использованию условия в операции условной передачи управления). В частности, так может быть выбрано для продолжения вычислений одна из спекулятивно выполненных ветвей программы.

Предварительная подкачка кода в направлении ветвления (на фоне выполнения основной ветви) скрывает задержку по доступу к коду при передачах управления и, тем самым, позволяет выполнить передачу управления без остановки конвейера выполнения, когда уже известно условие ветвления.

Файл предикатов дает возможность параллельного вычисления нескольких условий. Это важно как для скалярных вычислений, обеспечивая условное выполнение операций, так и для работы в циклах, содержащих условные ветви.

Выполнение циклов методом программного конвейера является действенным методом ускорения векторных вычислений. Последовательные итерации цикла выполняются с некоторым наложением друг на друга. Шаг, с каким итерации накладываются, определяет темп выполнения итераций, который может быть существенно выше строго последовательного выполнения итераций. В каждом шаге программного конвейера выполняются операции из разных наложенных итераций, и их количество может быть большим. Для их выполнения может потребоваться одна или несколько широких команд. Как отмечалось выше, широкая команда "вмещает" в себя до 25 различных операций с 64-разрядными данными и до 61 операции с 32-разрядными данными (за счет использования операций с 32-разрядными данными, упакованными в формате 128), что эффективно используется при конвейерном выполнении циклов.

Обращения к массивам в памяти является важным моментом в векторных вычислениях. Здесь играют роль большое и переменное время доступа и необходимость на длительное время резервировать ячейки регистрового файла под вызываемые элементы массивов, что снижает эффективность использования регистрового файла. В микропроцессоре введен буфер предварительной подкачки, в который данные из массивов вызываются отдельной ветвью программы, выполняемой параллельно на фоне вычислений. Имеются специальные средства управления, позволяющие подстраивать опережение вызова элементов массивов по отношению к их использованию в вычислениях.

Вызванные элементы массивов хранятся в буфере подкачки и передаются в регистровый файл из основной ветви программы непосредственно перед использованием. В регистровом файле определена область для загрузки элементов массивов в цикле с базой, определяющей начало данных для текущей итерации цикла. Это позволяет обращаться к данным различных итераций цикла путем соответствующей индексации, поскольку в широкой команде могут присутствовать операции, относящиеся этим различным итерациям.

2.1.3.4 Организация сверхоперативной памяти

Сверхоперативная память имеет многоуровневую организацию. Она включает в себя регистровый файл RF, буфер предварительной подкачки массивов APB, кэш команд первого уровня I$, кэш данных первого уровня L1D$, кэши второго и третьего уровней L2$ и L3$ соответственно, общие для команд и данных.

Регистровый файл RF состоит из 256 128-разрядных регистров. Время доступа - 1 такт. 20 портов чтения и 10 портов записи. Регистровый файл RF предназначен для хранения глобальных данных (32 регистра) и областей данных последовательно запущенных процедур (верхней части стека процедур – 224 регистра). Глобальные данные и область данных текущей процедуры доступны для выполнения вычислений в текущей процедуре. Текущей процедуре может быть выделено окно размером до 64 регистров для локальных данных процедуры и окно размером до 128 регистров с продвигаемой в циклах базой для хранения подкачиваемых элементов массивов. Переполнение регистрового файла вызывает откачку его части в продолжение стека процедур в памяти, опустошение - подкачку из продолжения стека процедур в памяти.

Регистровый файл RF обеспечивает в каждом такте для каждого из шести арифметических каналов чтение трех аргументов, чтение двух аргументов для записи в память, запись шести результатов (по одному из каждого арифметического канала) и запись четырех чисел, считанных из памяти.

Буфер предварительной подкачки массивов APB, объем 4 Кбайт, состоит из двух блоков по 2 Кбайт, время доступа 4 такта. Имеет два канала подкачки массивов буфер и 4 канала пересылки элементов массивов из буфера APB в регистровый файл RF. Можно определить до 64 массивов подкачки. Подкачка порциями по 8, 16 или 32 байта, чтение в формате элементов массивов. Для каждого из подкачиваемых массивов в APB организуется FIFO буфер. Канал подкачки массивов наполняет такой FIFO буфер, а канал пересылки передает данные из FIFO буфера в регистровый файл RF (опустошает FIFO буфер).

2-канальный по чтению и однопортовый по записи кэш команд первого уровня I$. Объем 128 Кбайт, время доступа 3 такта. Чтение программного кода блоками по 64 байта. Одновременно может быть считан код для основной ветви и в направлении подготавливаемой передачи управления (в пределах 256 байт в сумме). Подкачка программного кода блоками по 256 байт (8 запросов в память по 32 байта). Обращение по виртуальному и физическому адресам.

4-портовый кэш данных первого уровня L1D$, объем 64 Кбайт, время доступа

3 такта, частичная ассоциативность 4 колонки, размер блока данных 32 байта, заведение при чтении, запись в память насквозь (write-through). Обращение по виртуальному и физическому адресу.

4-канальный кэш второго уровня L2$, объем 1 Мбайт, время доступа
9 тактов, сет-ассоциативность 4 колонки, размер блока данных 64 байта, заведение при чтении и записи, запись в память при замещении (write-back). Обращение по физическому адресу.

16-канальный кэш третьего уровня L3$, общий для всех 16 процессорных ядер. Объем 32 Мбайт, состоит из 16 банков по 2 Мбайт, каждый банк
сет-ассоциативный и состоит из 16 колонок по 2 К строк, блок данных 64 байт.

Все четыре кэша поддерживают согласованное (когерентное) состояние в многопроцессорной системе.

2.1.3.4 Виртуальная память

В микропроцессоре реализована общая для команд и данных виртуальная память объемом 256 Тбайт (48 разрядов виртуального адреса). Физическая память также имеет объем 256 Тбайт (48 разрядов физического адреса). Поддерживаются три размера страниц - 4 Кбайт, 2 Мбайт и 1 Гбайт.

Архитектура предоставляет два различных виртуальных пространства - первичное виртуальное пространство и вторичное виртуальное пространство.

Первичное (primary) пространство - это основное рабочее пространство архитектуры Эльбрус. В нем разворачиваются коды и данные ОС и запускаемых ею пользователей. Вторичное виртуальное пространство, главным образом, предназначено для эмуляционного исполнения приложений, скомпилированных для иныхх архитектур. Существенным отличием от первичного пространства является применение альтернативного алгоритма трансляции виртуальных адресов в физические, что предполагает использование независимой таблицы страниц.

Тип адресуемого виртуального пространства определяется кодом операции обращения в память или разметкой адресных переменных. Для каждого пользователя организуется свое адресное пространство, маркируемое целым числом - идентификатором процесса. Аппаратно поддержан 12-разрядный идентификатор процесса (pid), который дополняет виртуальный адрес в разнообразных кэширующих структурах, что позволяет одновременно хранить в них записи, относящиеся к разным пользователям.

Таблица страниц PT устанавливает соответствие виртуальных и физических адресов страниц. В первичном виртуальном пространстве использован "линейный" вариант организации таблицы страниц с загрузкой ее в виртуальную память. 4 уровня таблицы страниц загружены в ту же виртуальную память задачи, которую они же сами описывают. Трансляция виртуальных адресов таблицы выполняется в точности по тому же правилу, что и любого другого адреса задачи. Адресация к элементам таблицы осуществляется через указатель, индексируемый виртуальным номером страницы. Формат строки таблицы страниц PTE для всех уровней - 64 разряда. Таким образом, последний 3 уровень таблицы, описывающий исходное 48-разрядное виртуальное пространство, занимает 2^{27} 4 Кбайт страниц. 2-й уровень таблицы, содержащий PTE, описывающие собственно этот последний уровень таблицы, занимает уже 2^{18} 4 Кбайт страниц. Следующий 1 уровень – 2^{9} 4 Кбайт страниц. Наконец последний 0 уровень занимает одну 4 Кбайт страницу и адресация к нему осуществляется через указатель по физическому адресу.

Во вторичном виртуальном пространстве реализованы таблицы страниц для различных сочетаний 32- и 48-разрядных виртуальных и физических адресов.

2.1.3.5 Организация работы виртуальной машины

Понятие "виртуальная машина" обозначает режим исполнения программ (включая и системное ПО), в котором истинные аппаратные ресурсы изолированы от них дополнительным программным слоем - гипервизором. Для исполняемого системного ПО эта изоляция может быть невидимой, и оно считает, что распоряжается непосредственно аппаратными ресурсами (оперативной памятью, устройствами ввода-вывода и т.д).

Однако, в действительности оно распоряжается виртуальными ресурсами, предоставляемыми ему гипервизором. Гипервизор формирует и запускает виртуальную машину (также используетя термин "гость") с заказанной конфигурацией и далее отображает ее виртуальные ресурсы на реальные аппаратные. Такой подход позволяет запустить несколько независимых виртуальных машин - одновременно или в режиме разделения времени.

Архитектура определяет набор средств, обеспечивающих изоляцию виртуальной машины от аппаратных ресурсов.

1 Дополнительный режим исполнения, отличающий гостя от гипервизора. Этот режим дополняет привычные привилегированный/непривилегированный режимы и ортогонален им, то есть программные компоненты гипервизора и гостя могут исполняться как в непривилегированном, так и в привилегированном режиме.

2 Операция запуска гостя/виртуальной машины - GLAUNCH. Операция автоматически сохраняет оперативную часть контекста гипервизора в стеках и теневых регистрах, загружает управляющие регистры значениями из гостевого контекста, подкачивает гостевые стеки из памяти и передает управление на гостя. Одна и та же операция служит как для первоначального старта гостя/виртуальной

машины, так и для возврата в гостя/виртуальной машины после перехвата (см. ниже).

3 Изоляция гостя от аппаратных ресурсов реализуется так называемым механизмом "перехвата" избранных гостевых событий, то есть автоматической передачи управления гипервизору в случае попытки гостя выполнить некое действие. Этот механизм похож на вход в прерывание, но отличается полным переключением контекста и режима исполнения. Среди перехватываемых событий прежде всего нужно назвать обращение к физической оперативной памяти, к контроллеру прерываний и к устройствам ввода-вывода. Также бывает необходимо перехватывать обращения к некоторым управляющим регистрам - главным образом задающим режимы работы аппаратуры. Наконец, для имитации непрерывности и монотонности астрономического времени необходимо корректировать значения, считываемые гостем с регистров, имитирующих часы или подсчитывающих некоторые события.

Механизм перехвата обеспечивает автоматическое полное переключение контекста гостя на контекст гипервизора и соответствующую смену режима исполнения.

4 Для создания у гостя полной иллюзии монопольного владения оборудованием, перехват, после обработки гипервизором завершается возвратом в гостя. При этом часто необходимо подменить некоторые элементы его контекста, например, значение, записанное гостем в управляющий регистр.

5 В аппаратуру введен механизм двухуровневой трансляции виртуальных адресов гостя: сначала гостевой виртуальный адрес VA транслируется в гостевой физический адрес GPA, а затем GPA транслируется в физический адрес супервизора PA. Первая трансляция осуществляется типовым образом по таблице страниц PT, которую формирует и поддерживает гостевое СПО. Вторая трансляция осуществляется по таблице страниц, которую формирует и поддерживает гипервизор.

Второй уровень трансляции выполняется "невидимо" для гостя, поддерживая стратегию изоляции гостя от реальных аппаратных ресурсов.

2.1.3.6 Защищенные вычисления

Архитектура микропроцессора включает аппаратные средства для организации защищенных вычислений. Использует метод "контекстной защиты", заключающийся в том, что каждый активный агент (это может быть пользователь, программа, процедура и т.п.) имеет перечень известных ему объектов со спецификацией прав доступа. Физически это выражается в использовании набора некоторых служебных слов - дескрипторов, каждый из которых описывает местоположение своего объекта, а также содержит необходимые признаки. Доступ к объектам возможен только посредством дескрипторов.

Дескрипторы являются системной информацией и не могут быть модифицированы агентом иначе, как в сторону сокращения прав. Равным образом, агент не может расширить список (свой контекст) иначе, как получив доступ к объекту в качестве параметра, либо запросив новый объект у системы. Агент может передавать свои дескрипторы другому агенту (возможно, с сокращением прав доступа), тем самым, образуя нужную конфигурацию доступности. Для этого в архитектуре адресные данные отличаются от числовых данных, то есть типизированы.

В архитектуре используется динамическая типизация или тегирование. Она реализуется добавлением к информационным разрядам дополнительного поля - тега, содержащего тип. Минимальный нечисловой тип занимает 4 байта, поэтому тег добавляется к 4-байтовому слову. Слово в памяти, на регистрах и в шинах сопровождается 2-разрядным тегом. В архитектуре определено три базовых формата нечисловых данных: одинарное слово, двойное слово, квадро слово, причем все слова, образующие составные форматы должны иметь одинаковый тип.

Таким образом, тег слова кодирует следующее:

0 - слово содержит числовую информацию, либо само по себе, либо являясь фрагментом составного формата;

1 - слово содержит нечисловую информацию формата одинарное слово;

2 - слово содержит фрагмент нечисловой информации формата двойное слово;

3 - слово содержит фрагмент нечисловой информации формата квадро.

Ни длина, ни тип числовых данных архитектурно не различимы: слово числового типа может содержать как переменную типа char, так и фрагмент переменной типа longint или longdouble. Семантическое наполнение числовой переменной отслеживается компилятором и проявляется, когда она становится операндом какой-либо операции.

В отличие от числовых данных, нечисловые данные строго типизируются. Прежде всего, гарантируется целостность составных форматов (двойное слово и квадро), что предполагает, что:

- все фрагменты должны иметь тип, соответствующий формату;
- при любых манипуляциях фрагменты должны сохранять свой порядок.

Второе качество реализуется выровненным расположением данных, как в памяти, так и в оперативных регистрах процессора. Так, младшее слово адресной переменной формата двойное слово всегда помещается по адресу, кратному 8-ми байтам, а старшее - следом за ним, смежным образом. Пересылки отдельных фрагментов приводят к разрушению их типов (превращению в неспециализированные данные). Операции, требующие специализированных операндов (например, адресных данных), строго контролируют их тип и целостность.

В рамках каждой из форматных групп конкретный тип специализированных данных определяется значением выделенной группы информационных разрядов - внутреннего тега.

2.1.3.7 Стеки

Данные, доступ к которым организован по методу стека, подразделяются на три группы по своим функциональным характеристикам и уровню доступности для пользователя:

- параметры, локальные данные и промежуточные значения процедуры, размещенные в оперативных регистрах;

- параметры и локальные данные процедуры, размещенные в памяти;

- "связующая информация", описывающая предыдущую (запустившую) процедуру в стеке процедур.

Для каждой из групп выделяется отдельный стек (перечислены в том же порядке):

- стек процедур;

- стек пользователя;

- стек связующей информации.

Стек процедур предназначен для данных, вынесенных на оперативные регистры. Это могут быть параметры, локальные данные и промежуточные значения. Каждая процедура работает только в своем окне, которое может пересекаться с предыдущим окном областью параметров (она же является областью возвращаемых значений). Верхняя часть стека (окно текущей процедуры - обязательно, данные предыдущих процедур - возможно) расположена на регистрах, оставшаяся часть продолжается в памяти.

Обращение за данными (для пользователя) возможно только в текущее окно, всегда расположенное на оперативных регистрах. Регистры адресуются с помощью индекса относительно начала окна. Индекс может быть как абсолютный, так и вычисляемый относительно дополнительной базы, предназначенной для обработки циклов.

Стек пользователя предназначен для данных, которые пользователь считает нужным разместить в памяти. На регистр процессора загружен дескриптор, описывающий свободную память под стек пользователя. Введена специальная операция, которая резервирует память требуемого размера (фрейм) и возвращает в качестве результата дескриптор на зарезервированную для текущей процедуры область, а также уменьшает свободную часть стека пользователя. Освобождение памяти выполняется автоматически при возврате из процедуры.

Поскольку в стеке виртуальная память переиспользуется, встает проблема защиты данных. Она имеет два аспекта: 1) переиспользование памяти (выделение ранее освобожденного пространства), 2) "зависшие" указатели. Первая проблема решается автоматической чисткой переиспользуемой памяти. Принцип решения второй проблемы следующий. Указатели на текущий фрейм процедуры можно сохранять только в текущем фрейме, либо передавать в качестве параметра в вызываемую процедуру (передавать вверх по стеку). Соответственно, указатель не может быть ни записан в глобалы, ни передан в качестве возвращаемого значения, ни записан в глубину стека. Это касается как стека пользователя, так и стека процедур.

Стек связующей информации предназначен для размещения информации о предыдущей (вызвавшей) процедуре и используемой при возврате. При защищенном программировании пользователь не имеет возможности изменять эту информацию, поэтому для нее выделен специальный стек, доступный только операционной системе и аппаратуре. Стек связующей информации устроен так же, как стек процедур, то есть верхняя часть его размещена на специальных регистрах, а продолжение - в памяти. Область памяти, зарезервированная под стек связующей информации, описывается указателем стека связующей информации, загруженным на регистр процессора. Указатель стека связующей информации состоит из дескриптора и индекса. Дескриптор описывает всю зарезервированную память, а индекс указывает на верхнюю границу той части стека, которая расположена в памяти. Регистровая часть стека описывается указателем аппаратной вершины стека связующей информации, также загруженным на регистр процессора. Перекачка данных между памятью и регистрами осуществляется автоматически.

2.1.3.8 Блокировка и мониторинг фрагментов памяти

Фрагменты памяти длиной байт, полуслово, одинарное слово, двойное слово или квадро слово могут быть заблокированы, чтобы регистрировать неожиданные

обращения к этому фрагменту, как от своего микропроцессора, так и от других микропроцессоров многопроцессорной системы (термин "свой" означает процессор, который заблокировал фрагмент). Операция является конфликтующей с блокировкой, если адресуемый фрагмент пересекается с заблокированным фрагментом. Блокировки памяти реализуются посредством ассоциативных таблиц DAM, SLT, MLT и MPT, в которых каждая строка соответствует защищаемому фрагменту памяти.

Таблица DAM используется для проверки корректности перестановок операций считывания по коду выше предшествующих операций записи. Операция считывания со специальным признаком блокировки "lock" выполняет обращение в память, а также заносит в DAM физический адрес и номер регистра результата (устанавливает блокировку). Последующие операции записи сравнивают свои физические адреса с адресами, хранимыми в DAM и, в случае сравнения, сбрасывают признак блокировки. Повторная контролирующая операция считывания с таким же регистром результата и специальным признаком проверки выполняет ассоциативный поиск по заданному номеру регистра результата и анализирует состояние блокировки. В случае сохранности блокировки операция считывания не выполняется, так как заблокированный фрагмент не был модифицирован со времени установки блокировки. В случае сброса блокировки контролирующая операция считывания выполняется повторно, так как после установки блокировки фрагмент был модифицирован и требуется его повторное считывание. Это же условие сброса блокировки может использоваться и для передачи управления на некий компенсирующий код, если по алгоритму программы некорректный результат считывания был использован в вычислениях и требуется провести коррекцию вычислений с использованием уже модифицированного значения.

Таблица SLT используется при программировании атомарных модификаций семафора. Модификация семафора заключается в считывании из памяти, изменении его состояния и обратной записи в память. Таблица SLT блокирует обращения к ячейке семафора в памяти от других микропроцессоров на период времени между двумя этими операциями. Операция считывания с признаком

блокировки "lock wait" выполняет чтение значения (семафора) из памяти, заносит физический адрес в строку таблицы SLT и устанавливает признак блокировки (захватывает адрес семафора). Признак блокировки, установленный в одном процессоре первым, запрещает последующим операциям считывания с признаком блокировки "lock wait" в других процессорах устанавливать блокировку в своих таблицах SLT в своих процессорах.

Операция записи с совпадающим физическим адресом в случае наличия блокировки выполняет запись модифицированного семафора в память и сбрасывает блокировку. Операция записи при отсутствии блокировки не выполняется, так как предшествующая попытка захватить адрес семафора была неудачной. Такая операция записи устанавливает предикат "lock_cond", по которому может быть выполнена передача управления на повторную попытку захватить семафор (выполнить чтение семафора с признаком блокировки "wait").

Таблица MLT позволяет заблокировать обращение по записи или считыванию/записи в заданный фрагмент памяти. Это свойство используется для некоторых оптимизаций компилятора. Например, в системе битовой компиляции, когда определенные переменные передаются на регистры процессора в целях ускорения вычислений и, поэтому, необходимо запретить доступ в память к фрагментам, где они ранее находились. Нарушение блокировки вызывает выработку особой ситуации "exc_mem_lock".

Кэш таблицы защиты памяти MPT (Memory Protection Table) содержит 8 64-разрядных слов для хранения флагов доступа к 4 Кбайт областям памяти при обращении во вторичное виртуальное пространство. Каждое слово хранит флаги доступа по записи MPT_WR и считыванию MPT_RD для 32 4 Кбайт областей памяти.

2.1.3.9 Средства поддержки совместимости с платформой x86

Архитектура определяет средства поддержки выполнения программного обеспечения платформы x86 и x86-64, перетранслированного в коды Эльбрус. К

ним относятся:

- совместимое представление данных, реализация набора операций, позволяющего эффективно имитировать операции платформы x86 и x86-64, включая моделирование стека для вещественных операций;
- два пространства виртуальной памяти;
- совместимый формат таблиц страниц;
- поддержка сегментной адресации;
- поддержка совместимости на уровне управления доступом в память (соблюдение порядка выполнения операций записи и разрешение невыровненных считываний).

2.1.3.10 Поддержка согласованного состояния памяти

Четырехуровневая организация памяти (кэши команд и данных первого уровня, кэш второго уровня, кэш третьего уровня и оперативная память), а также многопроцессорный режим работы с общей оперативной памятью требуют специальной организации для обеспечения согласованного (когерентного) состояния памяти. Кэш первого уровня использует стратегию заведение при считывании и запись насквозь в кэш второго уровня. Это обеспечивает согласованное состояние кэшей всех уровней. Кэш второго уровня использует стратегию заведение при считывании и записи, и запись в память только в случае замещения. То есть, операции считывания и записи при промахах в кэше второго уровня всегда вызывают данные из памяти в кэш второго уровня, где и выполняется их модификация. Кэш третьего уровня имеет неинкюзивную организацию по данным.

Когерентность доступа микропроцессоров в общую память поддерживается посредством запросов проверки когерентности (когерентных запросов), рассылаемых в микропроцессоры системы в соответствии с информацией из кэша справочника, в котором представлены состояние каждого 64-байтового используемого блока данных и указатель владельца самой "свежей" копии данных.

2.1.3.11 Организация данных и принципы доступа к ним

Архитектура микропроцессора позволяет образовывать два независимых виртуальных пространства: первичное (эльбрусовское) и вторичное. Обращения в то или иное пространство определяется кодом операции. В первичном пространстве могут размещаться данные двух типов: незащищенные и защищенные. Незащищенные данные адресуются произвольно, посредством абсолютного виртуального адреса, который представляется целым числом.

Для обращения к защищенным данным используется относительная адресация, посредством специальных адресных данных - дескрипторов и индексов. Данные обоих типов могут быть перемешаны в одном виртуальном пространстве, однако во избежание незащищенных обращений к защищенным данным используется постраничная защита. Страницы в таблице станиц помечаются как содержащие либо защищенные, либо незащищенные данные. Механизмы выделения памяти различаются для защищенных и незащищенных программ и оперируют каждый только своими страницами. При обращении к данным контролируется, что тип адреса соответствует типу страницы.

Область памяти представляется как связная последовательность байтов и описывается двумя параметрами: адрес первого байта (как правило, имеет название "база" или его сокращения) и количество байтов (как правило, имеет название "размер" или его сокращения). Байты нумеруются от нуля.

2.1.3.12 Организация программного кода и доступ к нему

Различаются защищенные и незащищенные коды.

Защищенные коды организованы в виде модулей компиляции (compilation unit), представляющих собой смежную область памяти, и содержащих одну или несколько статически слинкованных процедур со своими константами. Модуль занимает целое количество страниц и описывается дескриптором модуля компиляции CUD (Compilation Unit Descriptor).

Все модули задачи пронумерованы. Номер присваивается модулю на все время исполнения задачи и называется индексом модуля компиляции (Compilation Unit Index - CUI). Дескриптор модуля размещен в таблице модулей компиляции (Compilation Unit Table - CUT), в строку с номером, равным CUI. CUT описывается дескриптором CUTD, загруженным на регистр процессора.

При передаче управления модулю, нужная строка CUT считывается, и дескриптор помещаются на регистр процессора для текущего использования. CUI, присвоенный модулю, записан во всех строках таблицы страниц всех страниц кода, принадлежащих модулю. Таким образом, передача управления процедуре модуля предполагает следующую цепочку действий:

```
target address -> page table -> PTE -> code
                                  |
                                  +-> CUI -+
                                        |
                                        +-> CUT -> CUD -> регистр процессора
                                        |
                                    CUTD -+
```

Поскольку форматы строки таблицы страниц PTE для защищенного и незащищенного кода не отличаются, то для определенности 0-й CUI зарезервирован для незащищенных страниц. Защищенные же модули (страницы) нумеруются, начиная с единицы.

Незащищенные коды размещаются в произвольном месте виртуальной памяти за вычетом областей, отведенных под защищенные коды и данные. Обращение к незащищенным кодам осуществляется с помощью целого числа, значение которого воспринимается как адрес.

2.1.3.13 Представление данных

С точки зрения разрядности данные могут иметь следующие форматы:

Формат	Наименование	Значение
8 бит	байт	целое
2 байта	полуслово	целое
4 байта	одинарное слово	целое, вещественное
8 байтов	двойное слово	целое, вещественное, адресная информация
10 байтов	расширенное	вещественное
16 байтов	квадро слово	адресная информация, упакованные данные

2.1.3.14 Структура широкой команды

Архитектура определяет два типа команд и, соответственно, программ:

- синхронные, или "просто" команды (программы);

- асинхронные команды (программы), предназначенные для предварительной подкачки элементов массивов в циклах.

Асинхронная программа является узко специализированной и представляет собой несвязанные фрагменты кода, каждый из которых логически связан с некоторым циклом основной программы и предназначен только для циклического исполнения. Очередной фрагмент асинхронной программы запускается и останавливается из основной программы. Программы разного типа образуют разные командные потоки и предназначены для исполнения на разном оборудовании параллельно и асинхронно по отношению друг к другу. Программы разного типа

могут содержаться в одном программном сегменте в произвольном порядке, но программист должен гарантировать, что командные потоки не пересекаются.

Команда имеет длину от 8 до 64 байтов и выровнена до двойного слова. Она состоит из слогов и полуслогов - нетегированных слов (4 байта) и полуслов (2 байта). Каждый слог, один или в комбинации с другими (полу) слогами, кодирует отдельную операцию.

Последовательность полуслогов упаковывается в слова. Перечень типов (полу) слогов представлен в таблице 2.2.

Таблица 2.2 - Перечень типов слогов и их возможное количество в составе широкой команды

Обозначение	Наименование	Количество в команде
HS	Слог Заголовок	1
SS	Слог коротких операций	1
ALS	Слоги арифметико-логических каналов	до 6
ALES	Полуслоги расширения для арифметико-логических каналов	до 6
CS	Слоги команд управления	до 2
CDS	Слоги условного исполнения	до 3
PLS	Слоги каналов обработки логических предикатов	до 3
LTS	Слоги литералов	до 4
AAS	Полуслоги каналов обращения к элементам массивов	до 6

Слог Заголовок HS присутствует обязательно, прочие слоги - по необходимости. Определен следующий порядок размещения слогов в широкой команде: HS, ALS0, ALS1, ALS2, ALS3, ALS4, ALS5, CS0, CS1, SS, ALES0,

ALES1, ALES3, ALES4, AAS0, AAS1, AAS2, AAS3, AAS4, AAS5, LTS3, LTS2, LTS1, LTS0, PLS2, PLS1, PLS0, CDS2, CDS1, CDS0.

Слог Заголовок HS является обязательной составной частью любой широкой команды, поскольку он содержит информацию о длине и структуре широкой команды, используемую при ее распаковке и дешифрации.

Слог SS предназначен для операций, кодировка которых занимает всего несколько разрядов, например, передача управления или продвижение вращающейся базы регистрового файла.

Каждый слог типа ALS целиком содержит операцию одноименного арифметико-логического канала. Арифметико-логические каналы выполняют арифметические и логические операций, формируют адрес обращение в память, выполняют запись и считыванием регистров процессора. Формат операции традиционный. Слог содержит код операции, адреса операндов и результата, как правило, в регистровом файле. В качестве адреса назначения могут быть указаны также некоторые регистры процессора. Слог содержит специальный разряд, определяющий режим неспекулятивного или спекулятивного выполнения операции. В полях операндов может содержаться или кодироваться литеральное значение.

Полуслоги ALES являются расширением соответствующих слогов ALS и содержат либо просто расширение кода обычной операции, либо код для второго этажа комбинированной двухэтажной операции.

В слогах CS кодируются операции, выполняющиеся в рамках устройства управления или специальные операции. В типовом случае слог CS0 содержит операции подготовки передачи управления со статическим смещением, а слог CS1 - операции быстрой литеральной загрузки некоторых специальных регистров, а также операции синхронизации.

Слоги CDS содержат указания на то, какие из операций в обрабатывающих каналах выполняются условно и под управлением каких предикатов.

Слоги PLS содержат операции обработки логических величин - предикатов, считываемых из файла предикатов. Результат также помещается в файл предикатов.

Слоги LTS содержат литеральные значения, которые могут использоваться арифметико-логическими каналами в качестве операндов. Каждый слог может содержать либо 32-разрядное знаковое значение, либо любую из половин 64-разрядного значения.

Полуслоги AAS кодируют операции, которые пересылают элементы массивов, предварительно подкаченные в буфер, в регистровый файл для обработки.

Кодировка операции может занимать часть слога, слог или состоять из нескольких слогов. Так слог ALS определяет операцию для арифметико-логического канала над операндами из регистрового файла или с использованием короткого литерала в качестве операнда. Добавление слога (слогов) LTS позволяет задать литералы длиной 2 байта, 4 байта или 8 байтов. Добавление полуслога ALES позволяет определить комбинированную (составную) операцию или расширить код операции. Добавление слога CDS позволяет задать условный режим выполнения операции, представленной в слоге ALS, и предикат, определяющий возможность выполнения заданной операции.

Операции скалярного обращения в память кодируются также в слогах ALS. Добавление слога CS определяет режимы выполнения обращения в память: без поиска в кэше, без трансляции виртуального адреса в физический адрес, спекулятивный режим выполнения, специальные операции в кэше и таблице страниц типа откачки и чистки, обращения к регистрам подсистемы памяти и т.д.

Слог коротких операций SS, наоборот, позволяет задать сразу несколько коротких операций. В их число входят операция передачи управления (задаются номер регистра подготовки передачи управления и адрес предиката для условной передачи управления), операции продвижения текущих баз во вращающихся областях стека процедур и предикатном файле, а также операции начала и конца выполнения асинхронной программы в устройстве обращения к массивам AAU.

Операция пересылки элементов массивов из буфера предварительной подкачки APB в регистровый файл RF всегда занимает полтора полуслога AAS. Целый слог определяет формат элемента массива и область в APB, а полуслог - адрес назначения в регистровом файле RF.

2.1.3.15 Структура асинхронной команды

Асинхронная команда имеет фиксированную длину - 8 байтов и выровнена по границе двойного слова. Аналогично синхронной команде, асинхронная команда состоит из слогов словной длины (4 байта), которые кодируют операции. Различаются два типа слогов, которые должны присутствовать в команде в следующем порядке:

Название	Обозначение
Слог подкачки массива	APS0
Слог литерала для подкачки массива	APLS0
Слог подкачки массива	APS1
Слог литерала для подкачки массива	APLS1

Асинхронная команда может содержать до двух операций подкачки (для левого и правого каналов предварительной подкачки), а также операцию циклического перехода. Слоги APS0 и APLS0 задают операцию предварительной подкачки для левого канала, а APS1 и APLS1 - для правого. В слоге APS задается дескриптор массива и размер кванта подкачки, в слоге APLS - константа для индексации адреса обращения в память (постоянное смещение). Операция циклического перехода передает управление на начало асинхронной программы.

2.1.3.16 Операции широкого командного слова

Целочисленные и вещественные арифметические операции, логические операции и операции сдвига используют, в зависимости от типа операции один, два или три операнда и формируют результат в виде числового значения, набора флагов или предиката (операции сравнения). В качестве операндов могут быть числовые значения форматов слово или двойное слово из регистрового файла или константы, представленные в коде операции. Константы могут кодироваться в поле адреса операнда или как ссылка на дополнительные один или два слога литерала для задания констант форматов 32 и 64 соответственно. Результаты в виде числового значения или флагов помещаются в регистровый файл, в виде предикатов – предикатный файл.

Определены также вещественные арифметические операции с расширенным вещественным 80-разрядным форматом.

Целочисленные и вещественные операции могут использоваться в комбинированных трехадресных операциях. Комбинированная операция состоит из двух последовательно выполняемых операций. Первая операция выполняет заданную операцию с операндами 1 и 2, вторая – с результатом первой операции и третьим операндом. Результат второй операции заносится по адресу назначения в регистровый файл.

Операции над упакованными значениями используют операнды и формируют результат в формате двойное слово. В упакованном формате могут быть представлены байты и полуслова (только для операций над целыми значениями), одинарные слова и двойные слова. Операции над упакованными значениями определены для целых и вещественных данных.

Операции над предикатами позволяют считать предикаты из предикатного файла, выполнить функцию "логическое И" над двумя предикатами или их инверсиями (что реализует наиболее употребительный набор бинарных логических функций), записать вычисленный предикат в предикатный файл или направить его на управление условным выполнением арифметических операций или операций передачи управления.

Операции обращения в память выполняют считывание и запись значений в

память. Определены форматы значений байт, полуслово, одинарное слово, двойное слово и квадро слово.

Обращения в память сопровождаются спецификатором, задаваемым явно или формируемым по умолчанию и определяющим режим обращения в память. Спецификатор по умолчанию задает обращение в память с размещением данных в кэшах процессора. Спецификатор, явно определенный в дополнительном слоге, позволяет задать обращения в память: минуя кэши процессорного ядра, без трансляции виртуального адреса в физический, с проверкой блокировки фрагментов памяти в таблицах DAM, SLT и MLT, в область ввода-вывода.

Адрес обращения в память формируется как результат индексации дескриптора. Операнд 1 задает дескриптор – это значение формата квадро, операнд 2 задает индекс – это значение формата одинарное слово. Операнд 3 в операциях записи – записываемое в память значение.

Обращения в память к глобалам и коду выполняется с использованием аппаратных регистров, хранящих соответствующие дескрипторы. В этих случаях операнды 1 и 2 – индексы формата одинарное слово.

В операциях обращения в память к массиву адреса дескриптора и индекса задаются в устройстве обращения к массивам AAU, а литеральное смещение задается в дополнительном литеральном слоге.

В операциях обращения в память в незащищенное пространство дескриптор и индекс – это операнды формата двойное слово.

Операции преобразования адресных объектов выполняют индексацию заданного дескриптора и помещают результат в регистровый файл.

Операции с тегом выполняют считывание и установку внешнего тега для заданного значения.

Операции обращения к управляющим регистрам выполняют чтение и запись заданных управляющих регистров.

Операции подготовки передачи управления формируют значение указателя команды для предстоящей передачи управления и выполняют подкачку кода в заданный регистр подготовки передачи управления.

Операции передачи управления выполняют переключение дешифрации команд на заданное подготовленное направление и формируют новое регистровое окно в стеке процедур и стеке связующей информации (для процедурных передач управления).

Операции поддержки наложений цикла выполняют установку и продвижение управляющих регистров выполнения цикла.

Операции над кэшами позволяют очистить заданный кэш или его строку, а также прочитать значение заданной строки кэша.

2.1.3.17 Аппаратные команды

В определенных динамических ситуациях процессорное ядро выполняет некоторые действия, не описанные в программном коде (откачка / подкачка регистрового файла, вход в прерывание). В этих случаях формируются соответствующие аппаратные команды.

2.1.3.18 Прерывания

Прерывания могут быть программными и аппаратными. Программное прерывание инициируется специальной операцией и подобно вызову процедуры. Аппаратное прерывание инициируется аппаратурой и может принадлежать к одной из следующих категорий:

- точное; команда, которая вызвала прерывание, не произвела видимого эффекта и может быть повторно выполнена после возврата из обработчика прерывания;

- отложенное; команда, которая вызвала прерывание, произвела некоторый видимый эффект и не может быть повторно выполнена после возврата из обработчика прерывания. Однако, сохранена информация о конкретной операции, которая вызвала прерывание. Используя эту информацию, причина прерывания может быть устранена;

- неточное; команда, которая вызвала прерывание, произвела некоторый видимый эффект. Отсутствует информация (аргументы, окружение, т.п.) о конкретной операции, которая вызвала прерывание, поэтому она не может быть повторена или смоделирована программными средствами;

- асинхронное; прерывание, которое случается асинхронно по отношению к командному потоку, например, внешнее прерывание.

2.1.4 Конвейер выполнения команд

Широкое командное слово содержит набор параллельно выполняемых различных операций. В связи с этим конвейер выполнения широких команд имеет сложную структуру, позволяющую выполнять параллельно различные составляющие операции разной длительности.

Начальная часть конвейера является общей для всех операций широкой команды и включает 8 регистровых станций (рисунок 2.2). После их прохождения каждая из операций широкой команды поступает в свою параллельную ветвь для продолжения и завершения выполнения.

Такт	1	2	3	4	5	6	7	8
Станция конвейера	**L/C**	**A**	**F0**	**F1**	**S**	**D**	**B**	**R**

Рисунок 2.2 – Начальная часть конвейера

Станция конвейера	Описание
L/C	Регистры указателей команд для прямой (Linear) ветви Flat_IP 0, 1 и для передач управления (control trancfer) CTPR 1,2,3
A	Associative search in I$. Обращение в память тегов команд ITAG и формирование адреса строки по памяти данных буфера команд IDATA. Обращение в память тегов ITLB и формирование адреса по памяти физических адресов

Станция конвейера	Описание
F0	Fetch 0. Регистры адресов четной и нечетной строк в памяти команд IDATA, регистр адреса памяти тегов ITLB. Формирование адресов обращения в каждый банк памяти команд IDATA. Обращение в память физических адресов ITLB и формирование физического адреса блока программного кода
F1	Fetch 1. Входные регистры в банках памяти команд IDATA. Выборка строки до 256 байтов из памяти команд IDATA
S	Scattering. 4 регистра распаковки S. Распаковка широких команд основного и подготавливаемых потоков на составляющие операции
D	Decode. 4 регистра декодирования D. Выборка широкой команды из 4 потоков для выполнения. Дешифрация операций, вычисление адресов операндов
B	Branch. Регистр B. Передача адресов операндов на регистры адресов стека операндов. Проверка условий передачи управления и коммутация широких команд на стадии D
R	Read. Регистр адреса регистрового файла RF. Чтение операндов из RF

Конвейер выполнения операции загрузки Load для случая чтения из кэша первого уровня L1$ представлен на рисунке 2.3.

Такт	1-8	9	10	11	12	13
Станция конвейера	**L-R**	**E0**	**M1**	**M2**	**M3**	**M4**

Рисунок 2.3 – Временная диаграмма операции загрузки из кэша первого уровня

Станция конвейера	Описание

L-R	Начальная часть конвейера
E0	Входной регистр АУ. Вычисление адреса
M1	Memory 1. Входной регистр адреса L1D$. Обращение в память тегов и данных L1D$
M2	Memory 2. Выходной регистр данных L1D$. Hit в L1D$.
M3	Memory 3. Входной регистр RF. Запись данных в RF
M4	Memory 4. Данные в RF.Считывание данных из RF

Конвейер выполнения операции загрузки Load для случая чтения из кэша второго уровня L2$ представлен на рисунке 2.4.

Такт	1-8	9	10	11	12	13	14	15	16	17
Станция конвейера	**L-R**	**E0**	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**	**M8**

Рисунок 2.4 – Временная диаграмма операции загрузки из кэша второго уровня

Станция конвейера	Описание
L-R	Начальная часть конвейера
E0	Входной регистр АУ. Вычисление адреса
M1	Memory 1. Входные регистры адреса L1D$ и DTLB. Обращение в память тегов и данных кэша первого уровня. Обращение в память тегов и строк TLB, формирование физического адреса
M2	Miss в L1D$. Выходной регистр физического адреса ADDR_OUT TLB. Передача физического адреса по байпасу на входной регистр арбитра L2$
M3	Выходной регистр арбитра L2$. Обращение в память тегов и данных L2$
M4	Выходной регистр L2$, hit в L2$
M5	Регистр сборки с данными из буфера записи STB
M6	Регистр сборки с данными из MAU
M7	Регистр сборки с данными из APB (входной регистр RF). Запись в RF
M8	Данные в RF. Считывание данных из RF

Временная диаграмма операции записи Store представлена на рисунке 2.5.

Такт	1-8	9	10	11	12	13	14	15	16	17
Станция конвейера	**L-R**	**E0**	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**	**M8**

Рисунок 2.5 - Временная диаграмма операции записи

Станция конвейера	Описание
L-R	Начальная часть конвейера
E0	Входные регистры целочисленного АУ. Вычисление адреса и его передача в L1D$ и TLB. Чтение записываемого значения из стека операндов
M1	Адрес на входных регистрах L1D$ и TLB. Записываемые данные на буферном регистре st_align. Обращение память тегов кэша первого уровня и TLB
M2	Hit в L1D$. Записываемые данные на нижнем регистре входного буфера DBF в L1D$ Физический адрес на выходном регистре ADDR_OUT в TLB. Записываемые данные на выходном регистре DATA в TLB
M3	Записываемые данные на верхнем регистре входного буфера DBF в L1D$. Запись значения в память данных L1D$ Физический адрес на выходном регистре арбитра L2$. Записываемые данные в буфере арбитра L2$. Передача адреса и значения в буфер записи
M4	Hit в L2$. Адрес записи и данные в буферном регистре STB
M5	Адрес записи и данные в STB
M6	Записываемое значение на выходном регистре STB
M7	Записываемое значение на входном регистре памяти данных L2$. Запись значения в память данных L2$
M8	Записываемое значение в L2$

Временная диаграмма выполнения целочисленной операции представлена на рисунке 2.6.

Такт	1-8	9	10	11	12	13
Станция конвейера	**L-R**	**E0**	**E1**	**E2**	**E3**	**E4**

Рисунок 2.6 – Временная диаграмма целочисленной операции

Станция конвейера	Описание
L-R	Начальная часть конвейера
E0	Входной регистр целочисленного АУ первого уровня. Выполнение операции в исполнительном устройстве первого уровня
E1	Входной регистр целочисленного АУ второго уровня. Выполнение операции в исполнительном устройстве второго уровня
E2	Выходной регистр целочисленного АУ
E3	Входной регистр RF. Запись в RF
E4	Данные в RF. Считывание данных из RF

Временная диаграмма выполнения операции вещественного сложения представлена на рисунке 2.7.

Такт	1-8	9	10	11	12	13	14	15
Станция конвейера	**L-R**	**E0**	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**

Рисунок 2.7 – Временная диаграмма вещественной операции сложения

Станция конвейера	Описание
L-R	Начальная часть конвейера
E0	Входные регистры целочисленного АУ. Передача операндов на входные регистры вещественного АУ
E1	Входные регистры вещественного АУ. Выравнивание порядков
E2	Входные регистры сумматора. Выполнение сложения
E3	Выходной регистр сумматора. Выбор результата без нормализации или с нормализацией
E4	Выходной буфер вещественного АУ.

Станция конвейера	Описание
	Сборка результатов целого и вещественного АУ
E5	Входной регистр RF. Запись в RF
E6	Данные в RF. Считывание данных из RF

2.2 Кэш-память третьего уровня L3

2.2.1 Общее описание

Кэш-память третьего уровня (L3 кэш) предназначена для хранения команд и данных и является общей для всех процессорных ядер одного процессора, то есть каждое ядро имеет доступ ко всему объему L3 кэша. L3 кэш уменьшает поток обращений в память и задержку считывания данных, отвечает за поддержку межъядерной когерентности данных и участвует в поддержке когерентности в многопроцессорной системе.

Основные характеристики.

Объем –32 MB.

Размер блока данных (кэш-строки) – 64 B.

Расслоение на 16 банков (секций), работающих параллельно, объем одного банка – 2 MB; по умолчанию расслоение по 6, 7, 8 и 9 разрядам физического адреса с возможностью настройки.

Общее количество строк – 32 MB/ 64 B = 512K строк.

Физический адрес 48 разрядов (см. рисунок 2.2.1): [5:0] – номер байта, [9:6] – номер банка (по умолчанию), [10]–номер саб-банка, [20:11] – индекс, [47:21] – тэг.

Shared – кэш-память третьего уровня общая для всех процессорных ядер.

Каждый банк сет-ассоциативный по 16 колонок (16-way set associative cache).

Non-inclusive data, inclusive tags–относительно кэш-памяти верхнего уровня (кэш-памяти ядер) L3 кэш неинклюзивен по данным, но инклюзивен по тэгам.

Неблокирующий кэш (multiple hit on miss, miss on miss).

Точка сериализации с адресной блокировкой.

Политика записи при промахе (Write Miss Policy) – заведение при промахе (WriteAllocate).

Политика записи при попадании (Write Hit Policy) – отложенная запись (WriteBack).

Политика вытеснения (RPP, Replacement & Promotion Policy) – NRU (NotRecently Used)/ RRIP (custom Re-Reference Interval Prediction).

Протокол когерентности – блокирующий, расширенный набор основных состояний MOESI (Modified-Owned-Exclusive-Shared-Invalid) с дополнительными основными состояниями R (Reserved), B (Bypass),

D (Deleted).

DCA (Direct Cache Access) –прямой доступ к кэшу L3 по DMA.

QoS:

Way Partitioning с 32 идентификаторами политики разделения.

Независимые политики для кода и для данных.

RAS:

- память тэгов и состояний защищена кодом SECDED (Single Error Correction, Double Error Detection), память RPP не защищена (сбой ячейки этой памяти не приводит к нарушению корректной работы процессора);

- память данных защищена кодом SECDED (Single Error Correction, Double Error Detection), реализовано кодирование каждого двойного слова (8b на 68b) с перемешиванием битов 4 закодированных двойных слов для снижения вероятности возникновения неисправимой множественной ошибки.

Demand & Patrol Scrubbing.

Bad Line Disabling.

S-NUCA (Static Non Uniform Cache Access) – статическое распределение адресов по банкам, полное время доступа при попадании 40-70 тактов, зависит от взаимного расположения запросчика и адресуемого банка.

Пропускная способность по данным: 16X (32 байта/такт по чтению и 32 байта/такт по записи), возможно одновременное выполнение чтения и записи.

Частота работы – clk_uncore.

Тэг	Индекс банка		Номер банка	Номер байта
	Индекс саб-банка	Номер саб-банка		
47:21	20:11	10	9:6	5:0

Рисунок 2.2.1 - Разбиение физического адреса в L3 кэше (по умолчанию)

Размещение L3 кэша в структуре микропроцессора представлено на рисунке 2.2.2.

Обозначения на рисунке 2.2.2:

Core – процессорное ядро, реализующее архитектуру «Эльбрус»;

L1I(IB) – кэш-память команд первого уровня, входит в состав процессорного ядра;

L1D – кэш-память данных первого уровня, входит в состав процессорного ядра;

DAM/MLT – кэширующие устройства, входят в состав процессорного ядра;

L2 – кэш-память второго уровня (L2 кэш), входит в состав процессорного ядра;

MAU – Memory Access Unit, устройство доступа в память, входит в состав процессорного ядра;

OCN – On-Chip Network, сеть-на-кристалле;

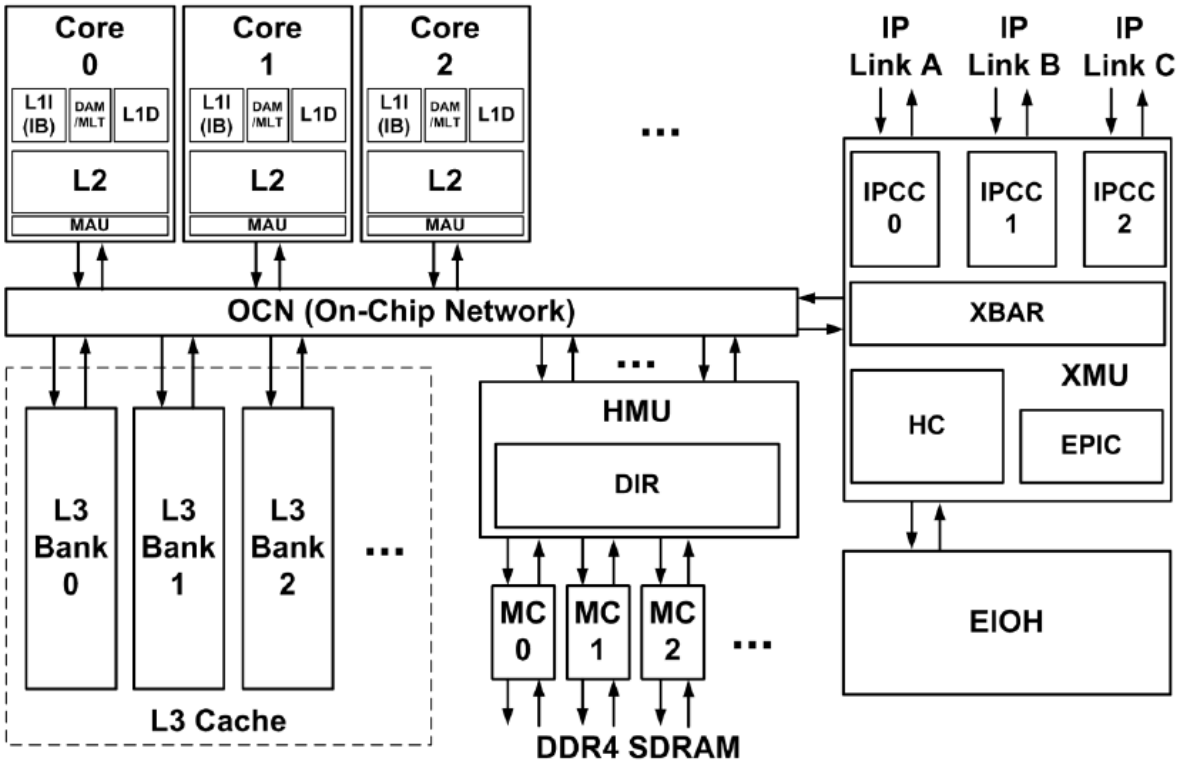


Рисунок 2.2.2 – Размещение L3 кэша в структуре микропроцессора

L3 Cache – общая кэш-память третьего уровня (L3 кэш) с интегрированным локальным справочником LD (Local Directory), имеет распределенную топологию;

L3 Bank – банк кэш-памяти третьего уровня, подключается к сети-на-кристалле;

HMU – Home Memory Unit,устройство доступа к оперативной памяти с интегрированным глобальным справочником DIR;

MC – Memory Controller, контроллер доступа к оперативной памяти, одноканальный;

XMU – eXternal Memory Unit, устройство доступа к внешней памяти;

IPCC – контроллер межпроцессорного линка;

XBAR – XMU Cross Bar, коммутатор, обеспечивающий подключение контроллеров межпроцессорных линков и ввода-вывода к сети OCN и поддерживающий агрегирование 2 или 3 межпроцессорных линков в один линк двойной или тройной ширины (мультилинк);

НС – Host-Controller;

EPIC – Elbrus Programmable Interrupt Controller, контроллер прерываний;

EIOH – Embedded IO-Hub, встроенный контроллер со всеми периферийными контроллерами.

Структурная схема банка L3 кэша представлена на рисунке 2.2.3.

Банк L3 кэша состоит из трех блоков - блока ENGINE со всей управляющей логикой и двух массивов памяти данных DA0 и DA1. Блок ENGINE подключается к сети-на-кристалле OCN и логически разделен на три части: блок управления банком CTRLи два саб-банка SBNK0 и SBNK1.

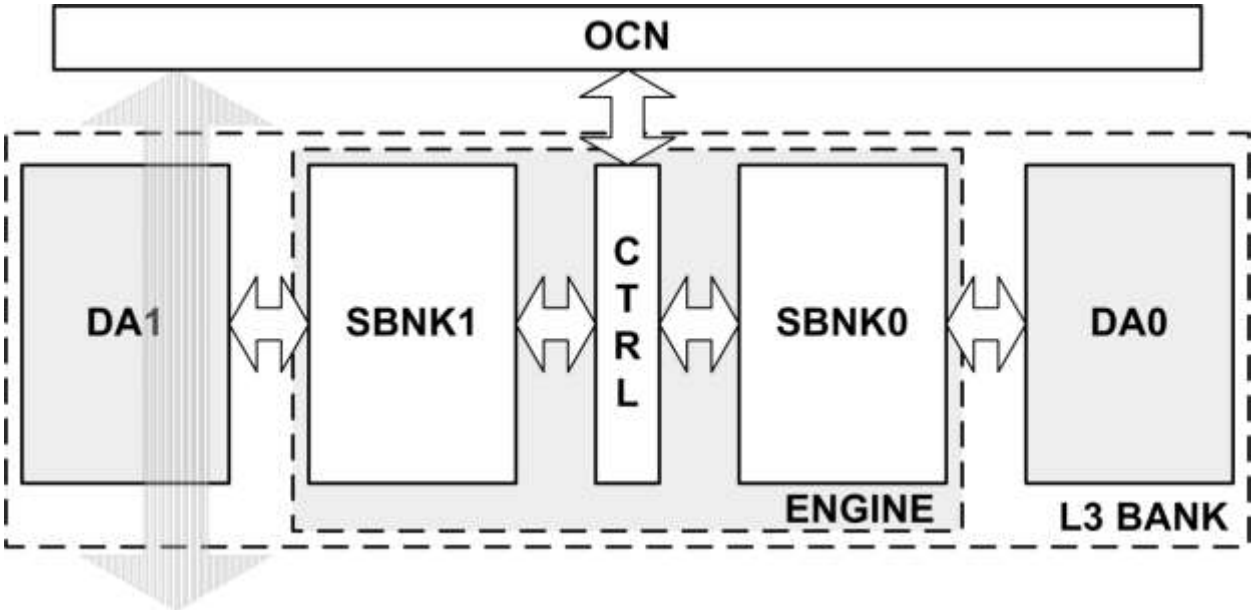


Рисунок 2.2.3 – Банк L3 кэша

Блок управления банком CTRL (Control) содержит интерфейсные модули банка для подключения к сети-на-кристалле OCN, различные служебные средства управления, тестирования, контроля и отладки банка L3 кэша.

В каждом банке L3 реализованы два независимых саб-банка с адресным разделением по младшему биту индекса. Каждый саб-банк SBNK (Sub-Bank) содержит память локального справочника, конвейеры доступа к памяти локального справочника и памяти данных L3 кэша, схемы сериализации и буферизующие структуры.

Массивы данных DA (DataArray) также имеют адресное разделение по младшему биту индекса и подключаются к соответствующим саб-банкам блока ENGINE: DA0 подключен к SBNK0, DA1 подключен к SBNK1.

2.2.2 Принципы работы

Подсистема памяти процессора с использованием общего кэша третьего уровня основывается на следующих принципах.

1 Кэш-память третьего уровня имеет распределенную структуру с адресным расслоением на независимые банки. Каждый банк имеет собственный интерфейс подключения к сети-на-кристалле OCN. Вычисление адреса назначения и маршрутизация пакетов в сети выполняется в сетевом адаптере OCN вне банка L3 кэша. Взаимодействие с другими устройствами процессора выполняется в соответствии с протоколом ESP.

2 Ввиду распределенной структуры время доступа к кэш-памяти третьего уровня неодинаково и зависит от взаимного расположения запросчика и адресуемого банка. При этом распределение адресов по банкам статическое, любая кэш-строка может находиться только в одном определенном банке (S-NUCA).

3 L3кэш является общим для всех ядер и DMA устройств (в режиме DCA) своего процессора, и любое ядро имеет доступ ко всему объему каждого банка. Общая кэш-памяти третьего уровня динамически распределяет свой объем между процессорными ядрами в зависимости от их потребностей и повышает эффективность и скорость работы с общими данными. Поддерживается QoS на основе Way Partitioningc программным управлением для изоляции задач и повышения общей эффективности использования L3 кэша.

4 В каждом банке используется схема с раздельными конвейерами тэгов и данных и схемы маскирования синхросигнала для неактивных блоков данных и регистров, позволяющие значительно снизить динамическую мощность L3 кэша ввиду его высокой ассоциативности.

5 Кэш-память третьего уровня обеспечивает корректную сериализацию первичных запросов от процессорных ядер своего процессора (сериализация первого уровня). L3 кэш поддерживает режим DCA, когда DMA запросы отправляются напрямую в L3 кэш своего процессора и сериализуются наравне с запросами от процессорных ядер. Запросы по одному адресу с точностью до кэш-строки (64 В) выстраиваются в очередь и блокируются при совпадении по адресу с еще обрабатываемыми запросами, которые пришли раньше. В любой момент времени для каждого адреса возможна обработка только одного первичного запроса для обеспечения атомарности транзакции (исключением является режим групповой обработки запросов). В L3 кэш отправляются только когерентные запросы по чтению и когерентные кэшируемые запросы по записи, остальные запросы от процессорных ядер и DMA отправляются напрямую в устройства сериализации второго уровня. Поддерживается режим отправки в L3 всех запросов независимо от их типа, в этом режиме некогерентные запросы и некэшируемые запросы по записи пересылаются в память без обработки в L3 кэше.

6 Блоки HMU и XMU отвечают за сериализацию второго уровня. Они имеют независимые интерфейсы подключения к сети-на-кристалле OCN и могут иметь адресное расслоение на независимые банки.

7 HMU отвечает за доступ к оперативной памяти своего процессора и содержит в своем составе глобальный справочник и DMA справочники для уменьшения времени доступа и когерентного трафика в многопроцессорной системе. Глобальный справочник хранит информацию о совладельцах кэш-строк с точностью до номера процессора и не содержит информацию о ядрах-совладельцах. Первичные запросы по одному адресу с точностью до кэш-строки 64 В выстраиваются в очередь и блокируются при совпадении по адресу с еще обрабатываемыми запросами, которые пришли раньше. В любой момент времени для каждого адреса возможна обработка только одного первичного запроса для обеспечения атомарности транзакции. Принимает некогерентные запросы и когерентные некэшируемые запросы по записи от процессорных ядер всех процессоров системы и по DMA, когерентные запросы по чтению и когерентные

кэшируемые запросы по записи от L3 кэшей всех процессоров системы и DMA (в режиме без DCA).

8 XMU отвечает за доступ к пространству ввода-вывода своего процессора и обработку DMA транзакций от контроллеров своего процессора. Принимает некогерентные запросы и когерентные некэшируемые запросы по записи от процессорных ядер всех процессоров системы, когерентные запросы по чтению и когерентные кэшируемые запросы по записи от L3 кэшей всех процессоров системы. Не поддерживает когерентность ввода-вывода и не рассылает снуп-запросов. Первичные запросы не блокируются при совпадении по адресу с еще обрабатываемыми запросами, которые пришли раньше.

9 Общая кэш-память третьего уровня является не инклюзивной по отношению к кэш-памяти верхних уровней, что обеспечивает гибкость и эффективность использования ее объема. При этом сохранено свойство инклюзивного локального справочника за счет расширения памяти тэгов общего кэша (для части кэш-строк хранятся только тэги и информация справочника, сами данные этих строк в L3 не хранятся). Такая схема сочетает преимуществами инклюзивной и не инклюзивной организаций общей кэш-памяти. Строки, находящиеся в L1 и L2 кэшах и DAM/MLT всех процессорных ядер, обязательно присутствуют в локальном справочнике, вытеснение строки из локального справочника вызывает вытеснение этой строки из всех кэшей верхних уровней (Back-Invalidate).

10 Локальный справочник выполняет функции снуп-фильтра для снуп-запросов в процессорные ядра и снижает когерентный трафик. При попадании в локальный справочник для запрашиваемой кэш-строки считывается вектор ядер-совладельцев, в кэш-памяти которых может присутствовать данная кэш-строка и которым необходимо разослать снуп-запросы. При промахе в локальный справочник гарантируется отсутствие запрашиваемой кэш-строки в кэш-памяти всех ядер процессора, следовательно, нет необходимости рассылать снуп-запросы.

11 Для каждой операции доступа в память от процессорного ядра заводится транзакция в MAU (в LDB для операций чтения, в STB для операций записи).

MAU отвечает в том числе за реализацию барьерных операций, с помощью которых реализуется нужная модель консистентности памяти.

12 MAU просматривается снуп-запросами вместе с кэш-памятью процессорного ядра. Снуп-запросы из L3кэша передаются в кэши верхнего уровня (L1I (IB), L1D, DAM/MLT, L2) через MAU. Одновременно с выполнением когерентных действий в кэшах верхнего уровня происходит поиск заявок Write-Back из L2кэша с совпадающим адресом в MAU. Результаты поиска строк в кэшах ядра и в MAU объединяются, и выдается обобщенный когерентный ответ из процессорного ядра. Если в MAU была найдена требуемая заявка, то ее данные передаются в L3 кэш как когерентный ответ с данными. В случае снуп-запроса на получение эксклюзивности (Read-Invalidate, Invalidate) найденная транзакция помечается признаком отмены, а соответствующий запрос Write-Back будет отменен по результатам просмотра локального справочника в L3кэше.

13 Рассылкой когерентных снуп-запросов при доступе в оперативную память занимается адресный контроллер блока HMU в соответствии с состоянием строки в глобальной директории. Сбор когерентных ответов от других процессоров и DMA кэша выполняет сам L3кэш.

14 Одновременно в L3 кэше может обрабатываться не более одного снуп-запроса по любому адресу. Это обеспечивается блокировкой всех запросов с совпадающим адресом в адресном контроллере Home-процессора на время обработки первого пришедшего.

Таким образом, подсистема памяти процессора имеет два уровня сериализации: сериализация первичных запросов от ядер и DMA устройств своего процессора в L3кэше (локальных протокол когерентности)и сериализация первичных запросов устройствах доступа в память HMU/XMU (глобальный протокол когерентности). Наличие более одной точки сериализации запросов значительно усложняет подсистему памяти и требует разработки корректного протокола их взаимодействия для предотвращения взаимных блокировок и соблюдения атомарности выполнения транзакций. Например, первичный запрос от процессорного ядра может заблокировать некоторый адрес в L3кэше и вызвать

отправку первичного запроса в HMU. К этому времени в адресном контроллере HMU этот же адрес блокируется первичным запросом из другого процессора и формируется когерентный снуп-запрос в L3кэш. Ни один из запросов обработаться не может, так как ожидает окончания обработки другого, что приводит к зависанию процессора. Для исключения таких ситуаций принимаются следующие принципы взаимодействия точек сериализации.

1 Когерентный снуп-запрос имеет больший приоритет, чем первичный запрос. Если некоторый запрос заблокировал адрес в адресном контроллере устройства доступа в память Home-процессора и для его обработки был отправлен когерентный снуп-запрос в L3 кэш, то такой снуп-запрос не может быть заблокирован в L3кэше первичным запросом, требующим обращения в тоже Home-устройство и пришедшим раньше. Первичный запрос вначале должен заблокировать адрес в адресном контроллере Home-устройства, только после чего блокировать адрес в L3кэше и приступать к обработке.

2 Если первичный запрос не требует обращения к Home-устройству, то он может сразу блокировать адрес в L3кэше и начинать обрабатываться. Следующие когерентные снуп-запросы по этому адресу будут блокироваться в L3 кэше и ожидать окончания исполнения первичного.

3 Если когерентным снуп-запросом заблокирован некоторый адрес в L3 кэше, то следующие первичные запросы с таким же адресом блокируются на время обработки этого снуп-запроса. Другие когерентные снуп-запросы по тому же адресу за время обработки первого прийти не могут, так как в адресном контроллере Home-устройства по каждому адресу может выполняться только одна транзакция в любой момент времени. Если первичным запросом заблокирован некоторый адрес в L3кэше, что возможно только если он уже заблокировал адрес в Home-устройстве или он не требует обращения к последнему, то последующий когерентный снуп-запрос с таким же адресом блокируются на время обработки этого первичного запроса.

4 Если первичным запросом заблокирован некоторый адрес в L3кэше, то последующие первичные запросы с таким же адресом блокируются на время обработки первого.

Вытеснения из L3 кэша и локального справочника вызываются только необходимостью завести новую строку в L3кэш/локальный справочник при промахе первичного запроса. В общем случае такие вытеснения требуют как локальных когерентных действий (например, вытеснение строк с таким же адресом из всех кэшей верхних уровней), так и глобальных когерентных действий (обращение в устройство доступа в память Home-процессора для записи модифицированных относительно памяти строк данных). Обработка вытеснений из L3 кэша отличается от обработки первичных запросов и снуп-запросов и основывается на следующих правилах.

1 Запрос на вытеснение при промахе формируется только в том случае, если есть строка с требуемым индексом, адрес которой не заблокирован никаким запросом, то есть все запросы с таким же адресом, пришедшие до момента формирования запроса на вытеснение, уже обработаны. Если все строки с индексом входного первичного запроса в текущий момент заблокированы, то вытеснения не происходит и этот первичный запрос переходит в ожидание. Исполнение первичного запроса не начинается, пока не разблокируется какая-нибудь строка этого сета и не будет сформирован запрос на вытеснение.

2 Как только запрос на вытеснение сформирован, он сразу начинает выполнять необходимые локальные действия в соответствии с исходным состоянием вытесняемой кэш-строки. В первую очередь выполняются действия, связанные с освобождением ячейки в памяти кэша (чтение данных из памяти L3 кэша), чтобы запрос, породивший вытеснение, мог раньше завершиться.

3 Если для некоторой строки кэша был сформирован запрос на вытеснение, то последующие первичные запросы и когерентный снуп-запрос к этой строке блокируются на время выполнения локальных действий.

4 После выполнения локальных действий (чтение данных из памяти L3 кэша, удаление копий строки из кэш-памяти ядер в случае вытеснения из локаль-

ного справочника, отправка первичного запроса на откачку модифицированной строки в память или обновление состояния глобального справочника) могут потребоваться глобальные действия (передача модифицированных данных в память). Во время выполнения глобальных действий первичные запросы и когерентный снуп-запрос к этой строке не блокируются.

5 После завершения глобальных действий (передача данных в Home-устройство) транзакция по вытеснению кэш-строки завершается.

2.3 Коммутационная сеть OCN

Назначение коммутационной сети OCN (On-Chip Network) - передача пакетов между абонентами микропроцессора – процессорными ядрами, банками кэш-памяти третьего уровня, контроллерми доступа к оперативной памяти HMU, контроллером доступа к периферийным устройствам XMU.

Сеть OCN состоит из 16 коммутаторов Tile для подключения банков кэша L3 и процессорного ядра CORE (рисунок 2.3.1) и двух коммутаторов XMU com для подключения устройства XMU (рисунок 2.3.2). Коммутаторы обеспечивают своим абонентам доступ к сети и транслируют пакеты сообщений, адресованные к абонентам в других узлах сети. Структурная схема сети OCN представлена на рисунке 2.3.3.

Коммутатор Tile имеет 6 портов. Четыре порта North (X), South (Y), Upper (Z--) и Down (Z++) используются для образования сети. К коммутатору Tile могут подключаться до двух абонентов. К портам Port 0 и Port 1 подключены банк кэша L3 и процессорное ядро Core соответственно.

Коммутатор XMU com также имеет 6 портов. Два порта Upper (Z--), два порта Down (Z++) и один North используюся для образования сети. К порту Port 0 коммутатора XMU com 0 подключен один абонент устройство XMU. Port 0 не используется в коммутаторе XMU com 1.

Соединения портов North (X) и South (Y) коммутаторов Tile образуют четыре кольца из узлов в плоскости X-Y. Соединения портов Upper (Z--) и Down

(Z++) позволяют сформировать трехмерную структуру сети OCN, состоящую из 4 колец узлов (рисунок 2.3.3). Между двумя верхними и двумя нижними кольцами узлов размещаются коммутаторы XMU com 0 и XMU com 1, соединенные между собой через порт North (X). К верхнему и нижнему кольцам узлов подключены 4 устройства доступа к оперативной памяти HMU (на рисунке 2.3.3 не показаны).

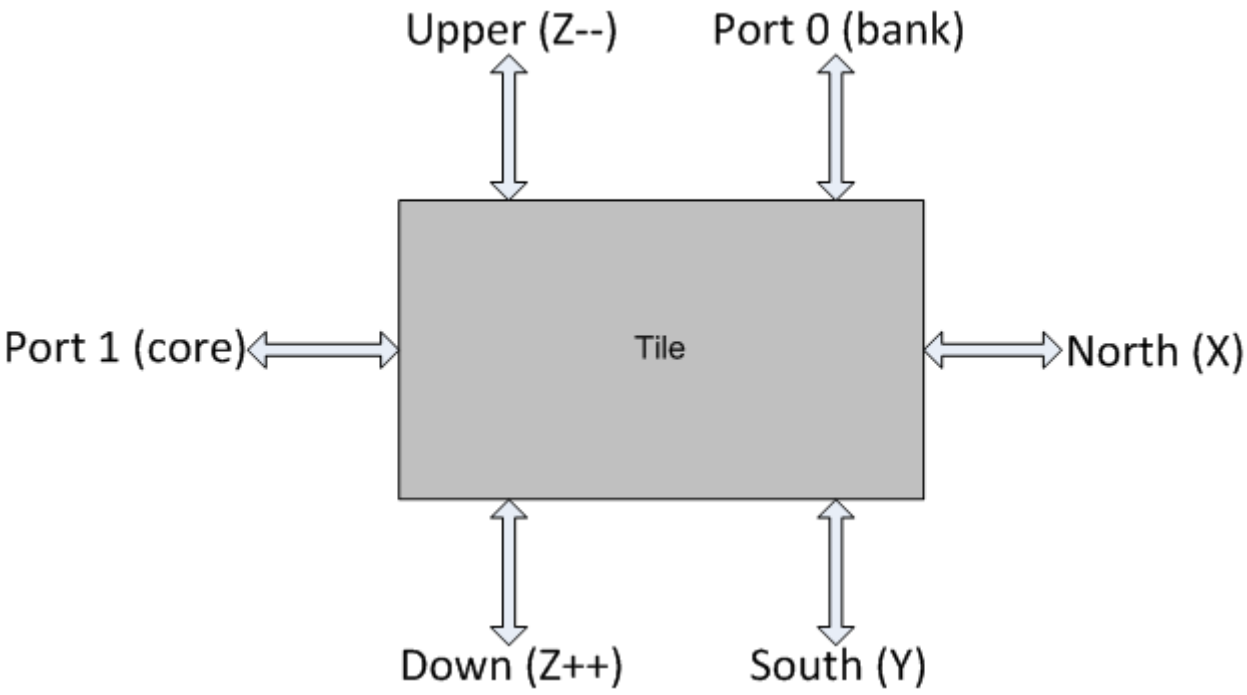


Рисунок 2.3.1 – Коммутатор Tile для подключения банка кэша L3 и процессорного ядра CORE

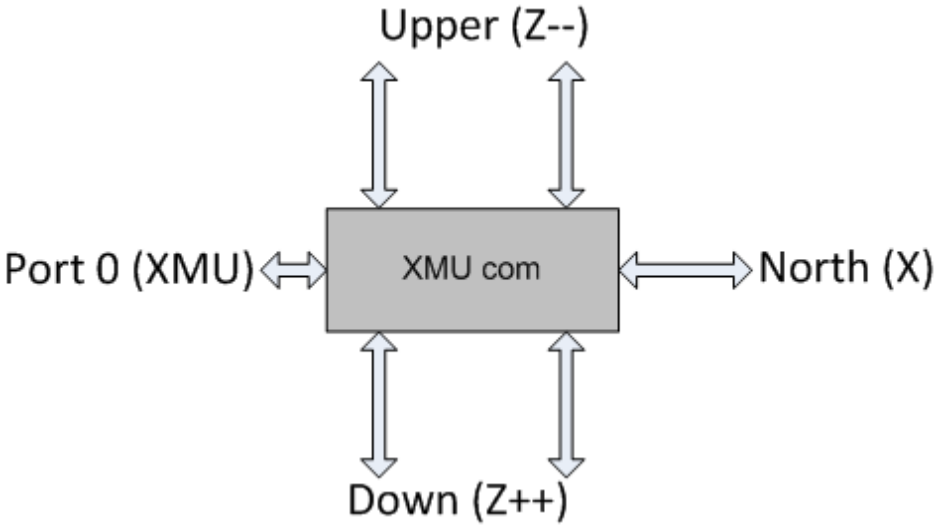


Рисунок 2.3.2 – Коммутатор XMU com для подключения устройства XMU

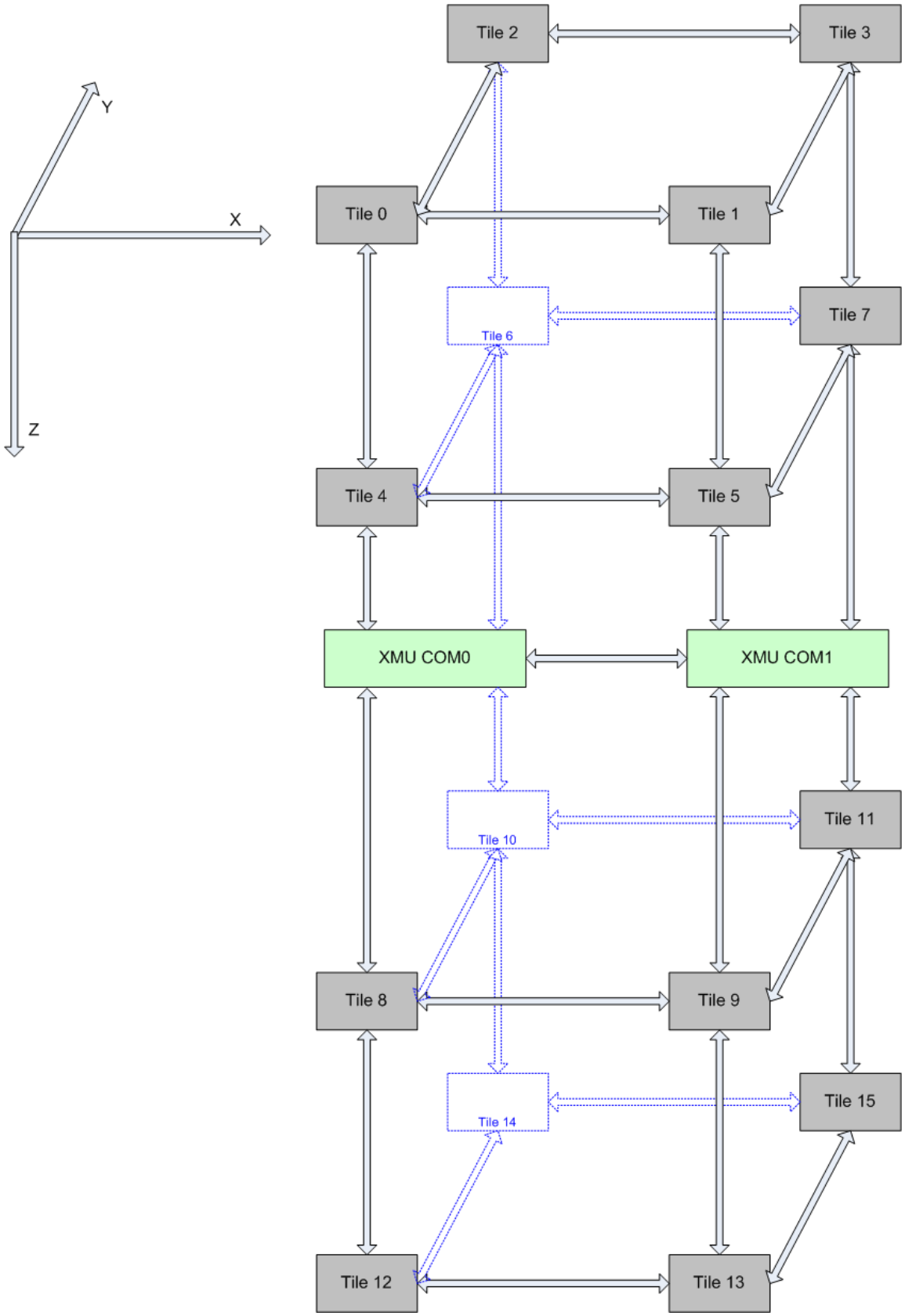


Рисунок 2.3.3 – Коммутационная сеть OCN

2.4 Устройство доступа к оперативной памяти HMU

Подсистема памяти микропроцессора содержит 4 устройства HMU (Home Memory Unit – устройство доступа в свою память) – по одному на каждые два канала оперативной памяти(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Основным функциональным назначением HMU является обеспечение когерентного доступа в оперативную память от различных абонентов.

Основные функции HMU:

- HMU отвечает за сериализацию всех запросов в свою память;

- для сокращения когерентного трафика и уменьшения времени доступа в память в HMU находится глобальный справочник (директория), который отслеживает обращения других процессоров к своей памяти;

- для поддержки межпроцессорной когерентности HMU занимается рассылкой когерентных запросов в соответствии с состоянием глобального справочника;

- HMU собирает короткие когерентные ответы и когерентные ответы с данными для операций записи в память. Для операций по чтению из памяти когерентные ответы собирает сам запросчик (DMA или L3 кэш).

2.4.1 Внешние интерфейсы HMU

Внешние интерфейсы и основные модули HMU представлены на структурной схеме HMU (рисунок 2.4.1).

Описание внешних интерфейсов HMU:

(1) – IRQ (Initial Request) – первичные запросы от L3 кэша, процессорных ядер или DMA (от своего или другого процессора);

(2) – SRQ (Snoop Request) – когерентные запросы от HMU в L3 кэш,

процессорные ядра или DMA (своего или другого процессора);

(3) – HAK (Home Acknowledgment) – короткий ответ от HMU запросчику (в L3 кэш, DMA). При записях – запрос за данными DRQ/DRC (Data Request/ Data Request & Complete), при чтениях – сообщение с количеством ожидаемых ответов, которые должен собрать запросчик;

(4) – ACK (Acknowledgment) – короткий когерентный ответ без данных от L3 кэша, процессорных ядер, DMA (своего или другого процессора);

(5, 6) – RLS (Release) – сообщение о завершении чтения или записи. При записях HMU отправляет сообщение WRLS запросчику. При чтениях сообщение RRLS отправляется в HMU запросчиком;

(7) – RDAT (Read Data) – считанные данные из оперативной памяти запросчику;

(8) – WDAT (Write Data), CDAT (Coherent Data) – данные записи в оперативную память, а также когерентные ответы с данными для записи в память. Данные записи приходят в ответ на запрос за данными DRQ/DRC, когерентные ответы с данными – в ответ на когерентные запросы SRQ;

(9, 10) – RQ0, RQ1 – 2 порта запросов в контроллер памяти MC (интерфейс функционирует на частоте контроллера памяти);

(11) – MSG – интерфейс обмена сообщениями с контроллером памяти (отмена запросов, резервирование ячеек для данных записи). Интерфейс работает на частоте контроллера памяти.

(12) – RDT – интерфейс приема данных чтения из контроллера памяти (интерфейс функционирует на частоте контроллера памяти, ширина интерфейса 32 байта);

(13) – WDT – интерфейс выдачи данных в MC для записи в оперативную память памяти (интерфейс функционирует на частоте контроллера памяти, ширина интерфейса 32 байта).

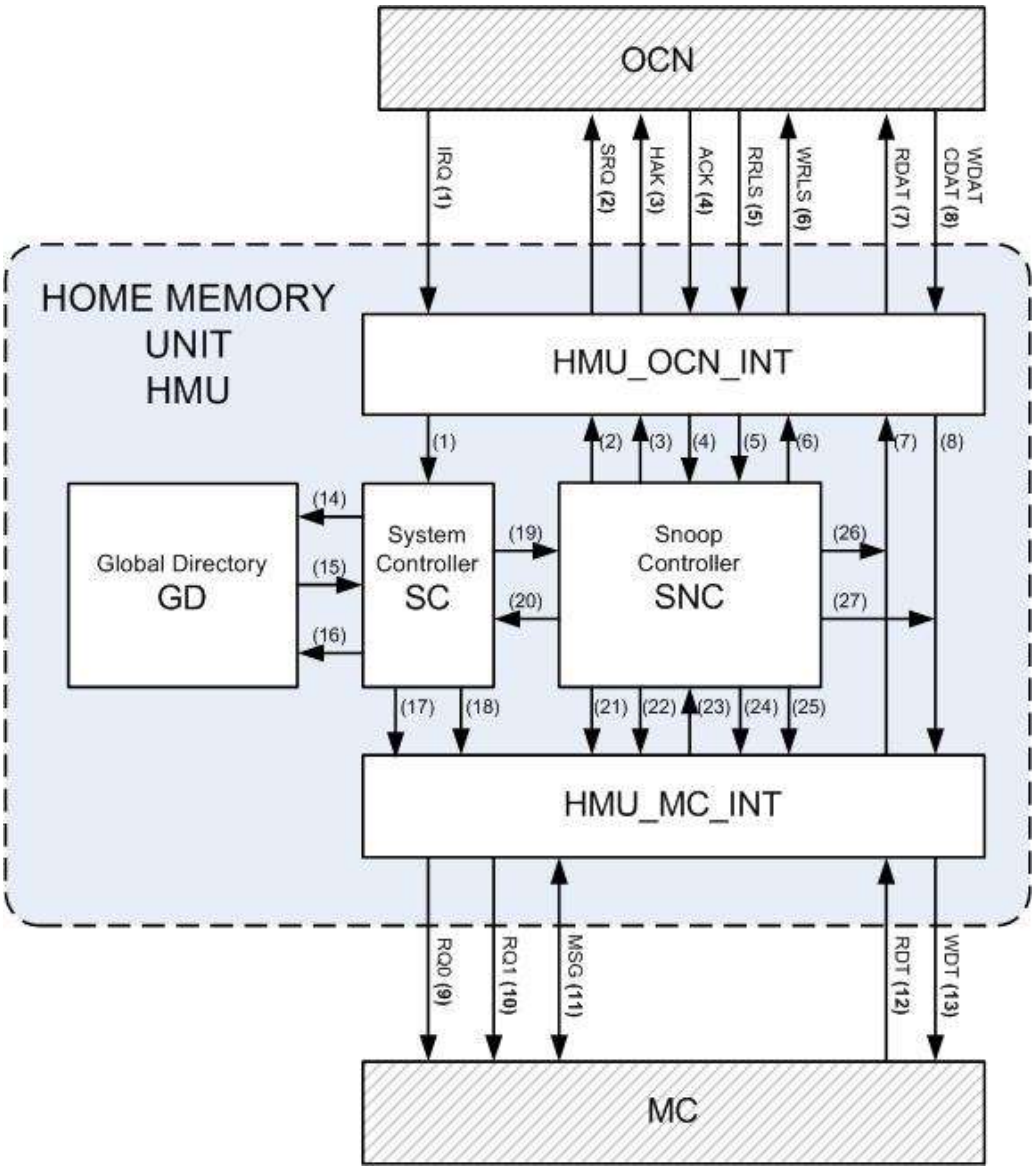


Рисунок 2.4.1 - Устройство HMU

2.4.2 Основные модули HMU

Описание основных модулей HMU:

HMU_OCN_INT (HMU OCN Intreface) – интерфейсный модуль между устройствами HMU и OCN;

HMU_MC_INT (HMU MC Interface) – интерфейсный модуль между HMU и контроллером памяти MC. В данном модуле происходит пересинхронизация интерфейса на частоту контроллера памяти. Модуль отвечает за выдачу номеров свободных ячеек для приема данных записи;

SC (System Controller) – системный контроллер. Устройство, отвечающее за сериализацию запросов в память и управляющее работой глобального справочника. Системный контроллер содержит конвейер, в котором определяется возможность выполнения запроса (проверяются адресные коллизии и наличие ресурсов для выполнения запроса). Также в конвейере системного контроллера обеспечивается чтение актуальной информации из справочника и запись новой информации в справочник. Если ничего не мешает выполнению запроса, запрос регистрируется в буфере запросов SC (Release Buffer, RLB) и вместе со считанной информацией из справочника передается на дальнейшее исполнение в контроллер когерентных запросов SNC. В буфере RLB (64 ячейки) хранятся все запросы в оперативную память, выполняющиеся в данный момент в системе. Также системный контроллер занимается отправкой команды чтения в контроллер памяти байпасом;

GD (Global Directory) – глобальный справочник. Справочник представляет собой множественно-ассоциативную память, в которой хранится информация о кэш-строках своей оперативной памяти, которые находятся в L3 кэшах других процессоров. Первичный запрос отправляется в справочник одновременно с выдачей запроса в конвейер SC, считанная информация из справочника (вектор-направление рассылки снуп-запросов) выдается на последней стадии конвейера SC. Справочник не является полным, поэтому в случае отсутствия места для размещения информации о новой кэш-строке, справочник обеспечивает вытеснение одной из своих записей. В устройстве GD также располагается дополнительный справочник для DMA-запросов. DMA-справочник представляет собой копию DMA-кэшей всех процессоров (DMA-кэш расположен в Хост-контроллере HC, XMU). DMA-справочник является полным, он отслеживает все кэш-строки, которые находятся в DMA-кешах;

SNC (Snoop Controller) – контроллер когерентных запросов. Устройство, осуществляющее контроль выполнения всех операций для запросов. Контроллер когерентных запросов рассылает все необходимые сообщения для выполнения запроса: когерентные запросы, сообщения HAK и RLS, запросы чтения и записи в

оперативную память. Устройство SNC собирает короткие когерентные ответы и отправляет сообщения о завершении операции в буфер запросов SC.

2.4.3 Принципы работы HMU

Для обращения в пространство памяти абонент (банк L3 кэша своего или другого процессора, устройство ввода-вывода DMA своего или другого процессора, процессорное ядро) отправляет первичный запрос IRQ (1) в HMU в соответствии с адресом. Каждый запрос имеет уникальную метку Label, которая однозначно идентифицирует запросчика. Запрос попадает в системный контроллер SC (System Controller), буферизуется и отправляется в конвейер SC и глобальный справочник (14). При прохождении запросом конвейера, производится адресная проверка запроса. Если предыдущий запрос с таким же адресом не завершился, запросу будет отказано в исполнении. Также, при прохождении запросом конвейера проверяется наличие ресурсов для выполнения данного запроса (наличие свободных ячеек в контроллере памяти, в очередях когерентных запросов и т.д). Если запрос прошел проверку на адресные коллизии и есть все необходимые ресурсы для его выполнения, запрос записывается в Release Buffer (RLB) системного контроллера, где хранится до своего завершения. Параллельно с работой конвейера системного контроллера справочник читает из своей памяти информацию о наличии кэш-строки с данным адресом в L3 кэшах других процессоров (15).

Прошедший конвейер запрос и считанная информация из справочника (19) передаются в контроллер когерентных запросов (Snoop Controller, SNC). Отправляется запрос в контроллер памяти (21, 22). Запрос чтения может быть отправлен байпасом в конвейере системного контроллера (17) и отменен в соответствии с информацией справочника (18). При отправке запроса записи в контроллер памяти резервируется ячейка для приема данных записи (и когерентных ответов с данными) в контроллере памяти (23). На основе информации из справочника рассылаются когерентные запросы SRQ (2).

Запросчику отправляется сообщение HAK (3). Для чтений сообщение HAK содержит информацию о количестве ожидаемых когерентных ответов, которые должен собрать запросчик. Для записей сообщение HAK – это запрос за данными записи. В случае, когда нет необходимости дожидаться сбора когерентных ответов, сообщение HAK – это запрос за данными записи и сообщение о завершении операции записи. В ответ на данное сообщение запросчик выдает данные записи в HMU.

Для запросов по чтению когерентные ответы собирает запросчик. Данные поступают запросчику либо из оперативной памяти (12, 7), либо от владельца кэш-строки (L3 или DMA кэша своего или чужого процессора). Информация о запросчике не передается в запросах в память, а сохраняется в контроллере когерентных запросов SNC и восстанавливается по номеру ячейки RLB (26). После сбора всех ответов, запросчик отправляет сообщение RRLS о завершении чтения в HMU (5).

Для запросов по записи когерентные ответы собираются в HMU. Короткие когерентные ответы собираются в контроллере когерентных запросов (4). Когерентные ответы с данными CDAT (8) передаются в HMU_MC_INT. Номер ячейки в контроллере памяти и номер контроллера памяти восстанавливается из контроллера когерентных запросов SNC по номеру ячейки RLB (27). После сбора когерентных ответов в контроллер памяти отправляется сообщение об отмене/подтверждении запроса записи (25). Данные записи WDAT (8) передаются в HMU_MC_INT и не требуют восстановления номера ячейки контроллера памяти. После сбора всех когерентных ответов, не дожидаясь данных записи (для когерентных непочтовых записей необходимо дожидаться данных записи), HMU отправляет запросчику сообщение о завершении операции записи (6).

После завершения обработки запроса SNC отправляет сообщение о завершении выполнения запроса в RLB Системного контроллера (20). Системный контроллер после получения сообщения о завершении выполнения запроса отправляет сообщение в Справочник (16), по которому Справочник записывает новое состояние в свою память.

2.4.4 Особенности выполнения атомарных операций

Атомарность выполнения операций типа RMW обеспечивается устройством HMU. Каждая такая операция разбивается на две команды в оперативную память: 1 – запись когерентной подложки, 2 – чтение подложки из памяти и модификация. После сбора когерентных ответов, в память записывается когерентная подложка (либо запись подложки отменяется). После получения данных записи, данные записи накладываются на подложку, считанную из памяти. Немодифицированные данные могут быть выданы запросчику. Наложение данных записи на подложку или модификация подложки осуществляется в буфере данных контроллера памяти.

2.4.5 Описание HMU_OCN_INT

HMU_OCN_INT - интерфейсный модуль между устройствами HMU и модулем Network adapter сети на кристалле OCN (рисунок 2.4.2).

Основные функции HMU_OCN_INT:

- буферизация первичных запросов IRQ, распаковка и отправка в SC (1);
- прием данных чтения RDT из оперативной памяти (3) (заголовок данных формируется в SNC (2)), склеивание половин кэш-строк и отправка их в ocn_HMU_na. Подсчет кредитов, доступно X кредитов. Данные выдаются по 2 каналам, привязанным к контроллерам памяти. Распределение данных в нужный канал сети на кристалле происходит в модуле ocn_HMU_na;
- буферизация данных записи в оперативную память WDT (CDT). Отправка заголовка данных в SNC для восстановления номера контроллера памяти и номера ячейки в буфере записи контроллера памяти для когерентных ответов с данными (5). Отправка заголовка данных записи в SNC, для завершения запроса записи (wnp) (5). Арбитраж и отправка данных в нужный контроллер памяти (4);
- подсчет кредитов для сообщений HAK (6). Доступно X кредитов;

- подсчет кредитов для когерентных запросов SRQ (7). Доступно X кредитов;

- подсчет кредитов для сообщений о завершении записи RLS (8). Доступно X кредитов;

- распаковка коротких когерентных ответов ACK и отправка их в SNC (9);

- распаковка сообщений о завершении чтения RLS и отправка их в SNC (10).

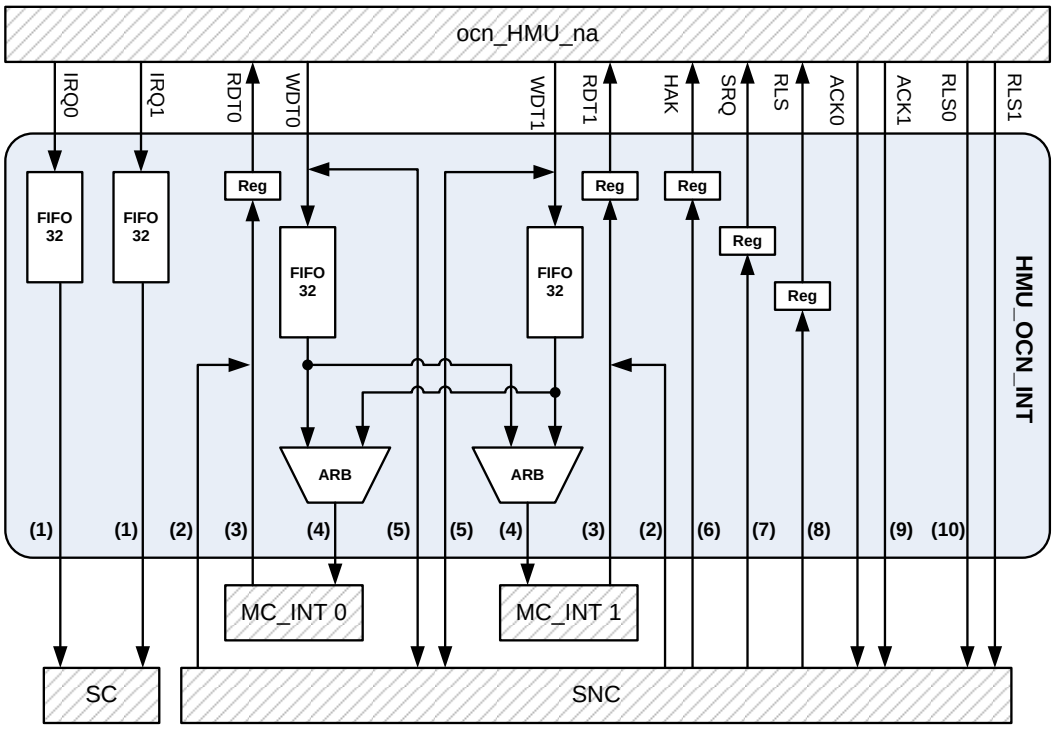


Рисунок 2.4.2 - HMU_OCN_INT

2.4.6 Системный контроллер SC

2.4.6.1 Назначение системного контроллера

Системный контроллер – устройство, отвечающее за сериализацию запросов в память и управляющее работой справочника.

Основные функции SC:

- буферизация и арбитраж запросов;

- поддержание очередности выполнения запросов с одинаковыми адресами;
- чтение информации о кэш-строке из справочника;
- обновление информации о кэш-строке и запись ее в память справочника;
- отправка команды чтения в контроллер памяти байпасом;
- отправка принятого на исполнение запроса в контроллер когерентных запросов SNC.

2.4.6.2 Структура системного контроллера

Системный контроллер состоит из следующих блоков (рисунок 2.4.3):

- очередь первичных запросов (Initial Request Queues, IRQ): предназначена для буферизации и арбитража входных запросов. Состоит из 8 очередей: распределение запросов по очередям осуществляется исходя из [9:7] разрядов адреса (после вырезания бит адреса, отвечающих за интерливинг между HMU). Содержит дополнительную очередь для отложенных запросов;
- конвейер запросов (Pipe): определяет возможность выполнения запроса и управляет работой справочника. В микропроцессоре справочник не содержит свой конвейер, вся управляющая логика перенесена в конвейер системного контроллера. Количество стадий: 4 + output register;
- буфер запросов ожидающих завершения (Release Buffer, RLB): хранит сведения о выполняющихся в настоящий момент запросах. Глубина RLB: 64;
- буфер отложенных запросов (Delayed Buffer, DLB): хранит запросы, не прошедшие конвейер из-за адресной коллизии с уже исполняющимися запросами;
- анализатор состояний протокола когерентности (Coherence Analyzer, CA): расшифровывает состояние, считанное из памяти справочника, формирует новое состояние, формирует вектор рассылки снуп-запросов.

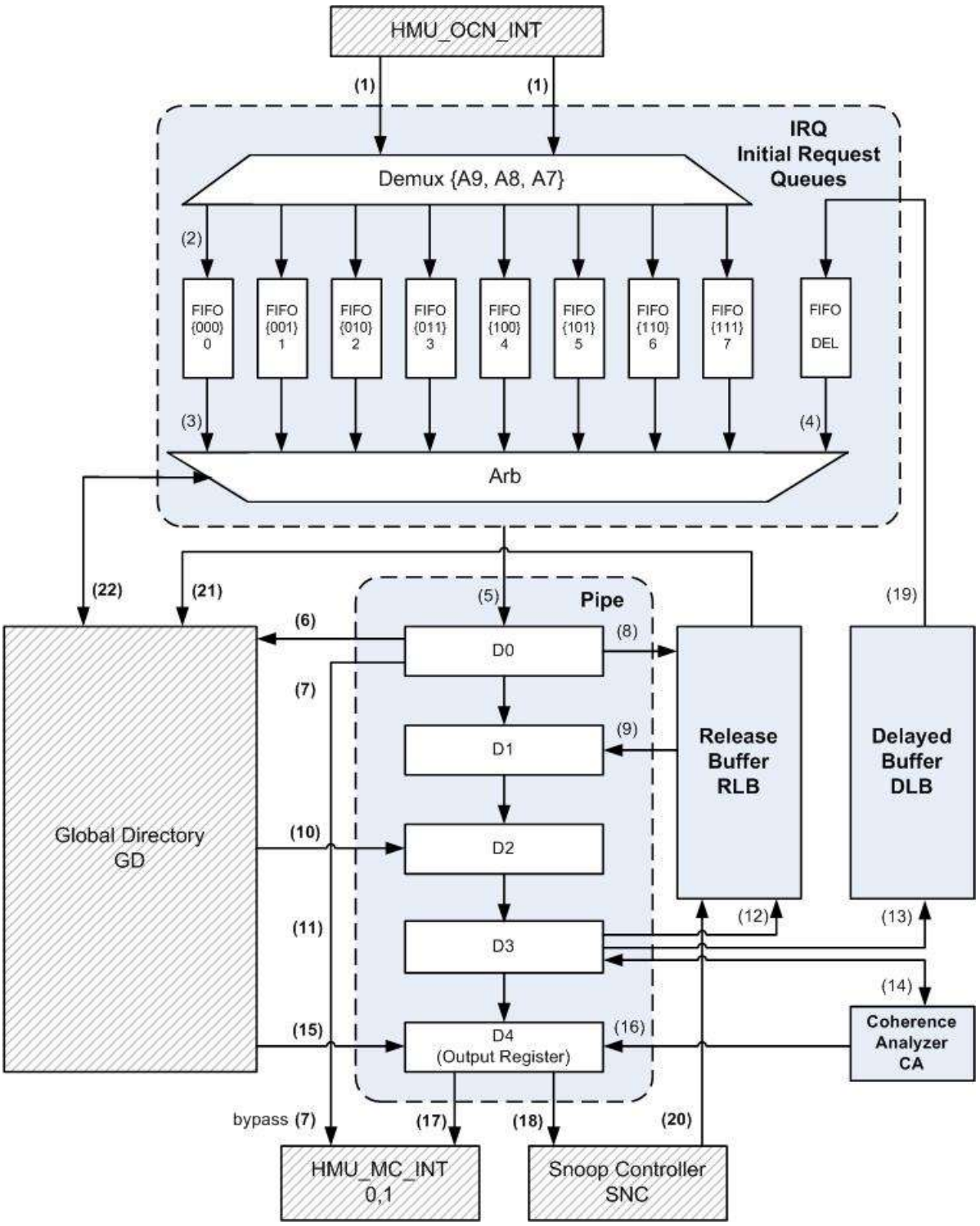


Рисунок 2.4.3 – Системный контроллер

2.4.6.3 Функциональное описание системного контроллера

Запросы в оперативную память поступают в очереди первичных запросов IRQ по 2 интерфейсам (1). Перед записью запроса в одну из 8 очередей, из адреса вырезаются биты, отвечающие за интерливинг между HMU (по умолчанию вырезаются 9 и 6 разряды адреса). Далее запрос записывается в одну из 8 очередей, исходя из [9:7] бит преобразованного адреса (2). Из очередей запросы поступают в весовой арбитр (3), который работает по принципу lru (least recently used). У очереди отложенных запросов абсолютный приоритет (4). Если включен справочник, арбитр не выдаст грант запросу, индекс ([19:7] разряды преобразованного адреса) которого совпадает с индексом запроса, находящегося на стадиях конвейера SC d0-d3.

Исключение делается только для запросов в соседние кэш-строки ([6] разряд адреса у запросов разный). Если справочник отключен, арбитр не выдаст грант запросу, адрес которого (с точность до кэш-строки) совпадает с адресом запроса на d0-d3 стадиях конвейера. Выбранный арбитром запрос передается в конвейер системного контроллера Pipe (5).

На стадии d0 конвейера SC отправляется запрос на чтение состояния справочника (6). Параллельно запрос отправляется в буфер запросов RLB (8), где производится адресная проверка запроса. Если запрос является чтением из памяти, производится преобразование адреса запроса в адрес для контроллера памяти (по умолчанию вырезается [7] бит исходного адреса). Если контроллер памяти свободен, отправляется команда чтения в контроллер памяти байпасом (7).

Если в буфере запросов RLB уже имеется запрос с таким же адресом (9), то запрос не может быть принят на исполнение. Такой запрос отправляется в буфер отложенных запросов DLB (13), где будет храниться до завершения предыдущего запроса с таким же адресом. Запросы с одинаковыми адресами могут образовывать в DLB связный список, в котором каждый запрос ожидает

выполнения предыдущего. После завершения запроса в RLB, головной запрос из DLB будет повторно отправлен в очереди первичных запросов IRQ (19).

Результат чтения справочника поступает в конвейер системного контроллера на стадии d2 (10). На стадии d3 конвейера SC считанная информация из справочника передается в анализатор состояний протокола когерентности CA (14). Анализатор состояний CA расшифровывает состояние строки, считанное из памяти справочника, формирует новое состояние (14) в соответствии с протоколом когерентности и формирует вектор рассылки снуп-запросов (16). На стадии d3 принимается решение о выполнении запроса. Если запрос был принят на исполнение, новое состояние строки записывается в справочник (11).

Отсутствие в справочнике (директории) записи с соответствующим адресом означает, что данная строка не содержится ни в кэшах других процессоров, ни в DMA-кэше какого-либо процессора. Если первичный запрос требует заведения новой записи в справочнике, в то время как соответствующий адресу запроса сет справочника полностью заполнен, одна из имеющихся в справочнике записей должна быть удалена. Информация об удаляемой из справочника записи поступает в SC на стадии d4 (15). Если запрос, вызвавший вытеснение из справочника был принят на исполнение, SC формирует от 1 до 2 запросов на вытеснение строки (flush) из кэшей GD WB. Запросы на вытеснение принимаются безусловно на исполнение.

Конвейер SC на стадиях d0-d3 проверяет возможность выполнения запроса. Причинами отказа в выполнении запроса могут быть (подробнее в таблице 2.4.1):

- нехватка ресурсов**:** заполнение очередей запросов в SNC, заполнение буфера запросов RLB, буфера отложенных запросов DLB;

- адресная коллизия**:** в буфере запросов RLB уже имеется запрос с таким же адресом, буфер отложенных запросов DLB либо заполнен, либо отключен;

- запрос порождает вытеснение из директории, но ни один из элементов директории нельзя использовать в качестве жертвы;

- запрос порождает вытеснение из директории, и на предыдущих стадиях конвейера есть другие запросы. В этом случае запросы с предыдущих стадий

конвейера будут возвращены в очереди запросов IRQ, а на их место будут вставлены запросы на вытеснение строки из кэшей (flush);

- при считывании информации из директории возникла ошибка ECC (стадия d3). В этом случае запрос на стадии d2 возвращается в очереди запросов IRQ;

- запросу на стадии d3 отказано в исполнении. Всем зависимым от него запросам, находящимся на стадиях d0-d2 будет отказано в исполнении (зависимые запросы: запросы из одной очереди IRQ, в некоторых случаях запросы в соседние кэш-строки).

В случае выполнения какого-либо из вышеперечисленных условий, запрос не может быть выполнен. Запрос отправляется обратно в очереди запросов IRQ.

Если ничего не мешает выполнению запроса, запрос записывается в буфер запросов RLB (12), где будет храниться до своего завершения. Информация о принятом на исполнение запросе (информация из первичного запроса, вектор рассылки когерентных запросов из директории и номер ячейки RLB npos, куда был записан запрос) передается в контроллер когерентности SNC (18). Если запрос является запросом записи, производится преобразование адреса запроса в адрес для контроллера памяти (по умолчанию вырезается [7] разряд исходного адреса). Команда на запись выдается в контроллер памяти из SNC через буферизующую очередь.

Если на стадии d0 была отправлена команда чтения контроллеру памяти, на стадии d4 отправляется отмена/подтверждение этой команды (17). Если запросу отказано в исполнении, либо чтение справочника показало, что актуальной строки нет в оперативной памяти, то команда чтения контроллеру памяти аннулируется.

Если команда чтения не была отправлена в контроллер памяти байпасом, запрос был принят на исполнение, и директория показала необходимость чтения памяти, команда чтения передается вместе с запросом в контроллер когерентности SNC (18), где буферизуется и выдается в контроллер памяти.

В отличие от предыдущих проектов директория реализована на однопортовой памяти, для которой запись и чтение в один такт невозможны. Для увеличения пропускной способности, директория разбита на 8 независимых по

чтению и записи банков. Директория может выполнить одно чтение за такт по адресу из одного банка, и одну запись за такт по адресу из другого банка. Для увеличения пропускной способности директории по чтению, в директории реализован буфер записи состояний Write Buffer WRB. Размер буфера записи соответствует размеру RLB системного контроллера. При чтении памяти директории, буфер записи состояний также просматривается. Информация из буфера записи считается наиболее актуальной. При записи состояния в директорию, состояние записывается в буфер записи. Если нет обращений по чтению к нужному из 8 банков, буфер записи переносит состояние в память директории. После завершения обработки запроса в RLB, состояние из буфера записи гарантированно записывается в память директории. Запрет одновременного доступа к одному банку директории по чтению и записи контролируется в SC.

После выполнения всех необходимых действий (отправка команды в контроллер памяти, рассылка когерентных запросов, сбор когерентных ответов и тд) контроллер когерентности SNC отправляет в SC сообщение о завершении запроса (20). Буфер запросов RLB освобождает ячейку и отправляет в директорию сообщение (21), по которому директория записывает состояние из буфера записи состояний в память директории. При записи состояния в память директории, справочник высылает сообщение, в какой банк в следующем такте будет произведена запись (22). Интерливинг между банками директории выполняется по тем же битам адреса, что и распределение запросов по очередям в IRQ. Арбитр не выдаст грант очереди, запрос из которой в следующем такте может обратиться по чтению к этому банку директории.

2.4.7 Контроллер когерентных запросов SNC

2.4.7.1 Назначение SNC

SNC (Snoop Controller) – контроллер когерентных запросов, осуществляющий контроль выполнения всех операций для запросов. Контроллер

SNC рассылает все необходимые сообщения для выполнения запроса: когерентные запросы, сообщения HAK и RLS, запросы чтения и записи в оперативную память. Устройство SNC собирает короткие когерентные ответы и отправляет сообщения о завершении операции в буфер запросов SC.

Основные функции SNC:

- буферизация принятых на исполнение запросов;
- буферизация и отправка запросов на чтение и запись в контроллер памяти;
- рассылка когерентных запросов SRQ;
- рассылка сообщения HAK;
- рассылка сообщений о завершении записи WRLS;
- прием сообщения о завершении чтения RRLS;
- сбор коротких когерентных ответов;
- восстановление номера контроллера памяти и номера ячейки в буфере записи контроллера памяти для когерентных ответов с данными;
- восстановление метки запроса и метки первичного источника запроса для данных чтения;
- отправка в контроллер памяти сообщения о сборе когерентных ответов для освобождения ячейки в буфере записи контроллера памяти;
- отправка сообщения о завершении операции в RLB системного контроллера.

2.4.7.2 Структура SNC

Контроллер когерентных запросов состоит из следующих блоков (рисунок 2.4.4):

- очередь первичных запросов (Initial Request Queues, IRQ). Модуль предназначен для буферизации запросов принятых на исполнение. Состоит из трех очередей: очередь для запросов по чтению, очередь для запросов по записи в 0-й канал контроллера памяти, очередь для запросов по записи в 1-й канал контроллера памяти;

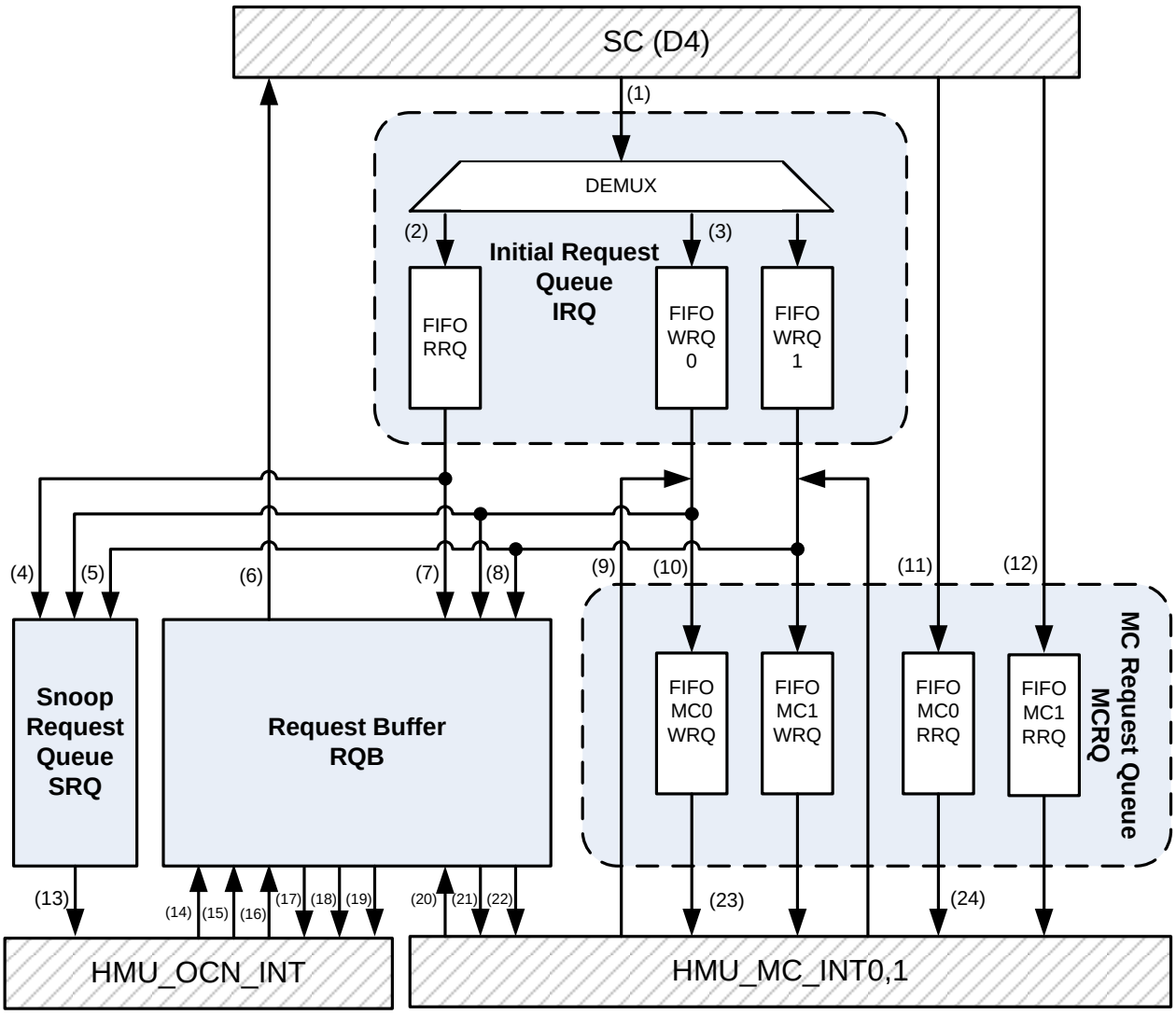


Рисунок 2.4.4 – Контроллер когерентных запросов SNC

- очередь запросов в контроллер памяти (Memory Controller Request Queue, MCRQ). Предназначен для буферизации запросов в контроллер памяти. Состоит из 4 очередей: 2 очереди для запросов чтения в 0-й и 1-й канал оперативной памяти, 2 очереди для запросов записи в 0-й и 1-й канал оперативной памяти;

- очередь когерентных запросов (Snoop Request Queue, SRQ). Предназначен для буферизации когерентных запросов с трех направлений (три очереди IRQ);

- буфер запросов (Request Buffer, RQB): хранит сведения о выполняющихся в настоящий момент запросах и контролирует их выполнение. Собирает короткие когерентные ответы, формирует необходимые сообщения запросчику (HAK,

WRLS) и контроллеру памяти, принимает и формирует сообщения об окончании операции. Отправляет сообщение в RLB о завершении операции.

2.4.7.3 Функциональное описание SNC

Принятый на исполнение запрос вместе с вектором рассылки когерентных запросов поступает с последней стадии конвейера в IRQ контроллера когерентных запросов (1).

Запрос записывается исходя из кода операции в одну из трех очередей (очередь для запросов по чтению FIFO RRQ (2) и две очереди для запросов по записи FIFO WRQ (3) по одной на каждый канал оперативной памяти). Очередь для запросов по чтению является однопортовой по чтению и однопортовой по записи. Очередь для запросов по записи является однопортовой по чтению и двухпортовой по записи. Два порта записи нужны для обработки операций типа RMW (запись по маске, атомарные операции). Такая операция разбивается на две операции – чтение когерентной подложки (induced/internal flush) и сама операция записи. На выходе из очередей FIFO WRQ, перед отправкой когерентных запросов и регистрацией запроса в буфере RQB, необходимо зарезервировать ячейку (dpos) в буфере данных контроллера памяти (для данных записи и/или когерентной подложки) (9). После резервирования ячейки dpos формируется команда записи в контроллер памяти (23), которая буферизуется в MCRQ (10), рассылаются когерентные запросы (13) через буферизующую очередь SRQ (5) и запрос регистрируется в RQB (8). С выхода очереди чтения FIFO RRQ запросы сразу передаются в SRQ (4) и регистрируются в RQB (7). Для буферизации когерентных запросов используется однопортовая по чтению и трехпортовая по записи очередь SRQ. Для операций по записи когерентные ответы собирает Home-устройство, поэтому когерентные запросы рассылаются с меткой HMU. Для операций по чтению когерентные ответы собирает запросчик, поэтому когерентные запросы рассылаются с меткой из первичного запроса (label – метка запроса, slabel – метка первичного источника запроса).

Если команда чтения не была отправлена в контроллер памяти байпасом, команда чтения передается вместе с первичным запросом в контроллер когерентности по выделенным интерфейсам (11, 12), где буферизуется и выдается в контроллер памяти (24).

Из очередей IRQ запрос попадает в буфер RQB (7,8). Глубина буфер запросов RQB соответствует глубине буфера RLB системного контроллера. Запрос записывается в ячейку с тем же номером npos, что и в RLB. После регистрации запроса в буфере запросов RQB для завершения запроса необходимо выполнить ряд действий (таблица 2.4.2).

Таблица 2.4.2 - События, необходимые для завершения запроса

Запрос	MC RQ	DAT	ACK	MC ACK	HAK	RLS
RI/I	+	RDAT	-	-	+	+
	+	RDAT	+	RACK	+	+
R/Rnc	+	RDAT	-	-	+	+
	+	RDAT	+	RACK	-	+
Rns	+	RDAT	-	-	-	-
	+	RDAT	+	RACK	-	-
Flush	+	CDAT	+	WACK	-	+
	+	CDAT	+	WACK	-	+
WB	+	WDAT*	-	-	+	-
	+	WDAT*	-	-	+	-
Wns	+	WDAT*	-	-	+	-
	+	WDAT*	-	-	+	-
W64np	+	WDAT	+	-	+	+
	+	WDAT	+	-	+	+
W32mnp, W32np	++	CDAT WDAT	+	WACK	+	+
	++	CDAT WDAT	+	WACK	+	+
Atomic	++					
	++					

Запрос	MC RQ	DAT	ACK	MC ACK	HAK	RLS
DirWb	+	CDAT	+	WACK	-	-
	+	CDAT	+	WACK	-	-

_____

***** Запрос может завершиться, не дожидаясь этого события

Для операций по чтению необходимо:

- отправить команду чтения в контроллер памяти MCRQ(24) (при необходимости чтения памяти);

- отправить сообщение HAK запросчику с меткой первичного запроса и меткой Home-устройства (17);

- дождаться выдачи данных чтения RDAT для восстановления метки первичного запроса (20, 19) и дождаться сообщения о завершении операции чтения от запросчика RLS (14). Выдача данных чтения из оперативной памяти отменяется, если в одном из коротких ответов был признак данных (DATC, DATD).

Для операций по записи необходимо:

- отправить одну или две команды записи в контроллер памяти MCRQ (23);

- отправить сообщение HAK запросчику с меткой запросчика и контроллера памяти (17) для получения данных записи (для flush не формируется).

Необходимо собрать все когерентные ответы ACK (15) (для операций Wns и WB когерентные запросы не рассылаются), после чего сформировать сообщение MC WACK с отменой/подтверждением записи (22). Также необходимо дождаться когерентного ответа с данными CDAT для восстановления номера контроллера памяти и номера ячейки dpos (16,19). Для операций Wnp необходимо дождаться данных записи WDAT (16). После выполнения всех этих действий запросчику отправляется сообщение об окончании операции записи RLS (18).

Выбор запроса, для которого необходимо выполнить определенное действие, происходит в одном из трех весовых арбитров в RQB. Арбитры выдают грант запросу с наибольшим возрастом. Первый арбитр выбирает запрос, ожидающий

отправки сообщения HAK. Если необходимо отправить сообщение MC RACK, оно отправляется одновременно с выдачей сообщения HAK. Второй арбитр выбирает запрос, ожидающий отправки сообщения RLS. Одновременно с сообщением RLS при необходимости отправляется сообщение MC WACK. Третий арбитр выбирает запрос, для которого выполнены все действия и он ожидает своего завершения в RLB. После отправки сообщения об окончании выполнения операции в RLB системного контроллера (6), запрос считается завершенным.

2.4.8 Глобальный справочник

2.4.8.1 Характеристики глобального справочника

Тип памяти – сет-ассоциативная.

Объем – 4 x 640 Кбайт (4 x 752 Кбайт с учётом ECC-битов).

Общее количество строк – 8192 строки.

Длина строки – 80 байт (94 байт с ECC-битами).

Расслоение на 8 секций, к которым возможен параллельный доступ по чтению и записи, расслоение по 3 младшим разрядам индекса.

Размер секции – 80 Кбайт (94 Кбайт).

Каждая секция сет-ассоциативная по 16 колонок (16-way set associative cache).

Общее количество банок памяти – 128 (8 секций по 16 банок в секции).

Размер одной банки памяти – 1024 строки по 47 бит.

Размер индекса – 13 разрядов.

Размер тэга – 28 разрядов.

Алгоритм замещения – NRU-FIFO.

Когерентный протокол глобального справочника – MOSI.

Темп поступления запросов – один за такт.

Время обработки запросов – 4 такта.

Размер элемента глобального справочника – 47 бит (7 ECC бит, 28 бит тэга, 12 бит - состояние двух кэш-строк).

Частота работы – синхронно с процессором.

2.4.8.2 Организация глобального справочника

Организация глобального справочника представлена на рисунке 2.4.5.

Глобальный справочник представляет собой сет-ассоциативную память, в которой лежит информация о кэш-строках, принадлежащих данной памяти и использующихся в других микропроцессорах. В состав микропроцессора входят 4 экземпляра глобального справочника, по одному в каждом HMU.

Кэш память разбита на 8 секций по 16 колонок - 128 банок памяти. Размер каждой банки 1024 строки на 47 бит. 7 бит отводятся под ECC биты, остальные 40 бит содержат информацию о кэш-строке. Таким образом, всего в одном экземпляре глобального справочника содержатся 16 колонок (way) по 8192 строк размером 47 бит. Если считать размер элемента глобального справочника равным 40 битам (без дополнительных ECC-битов), можно считать, что один экземпляр справочника занимает 640 Кайт памяти. Если же сложить вместе 16 колонок для одной строки, то получится строка размером 80 байт.

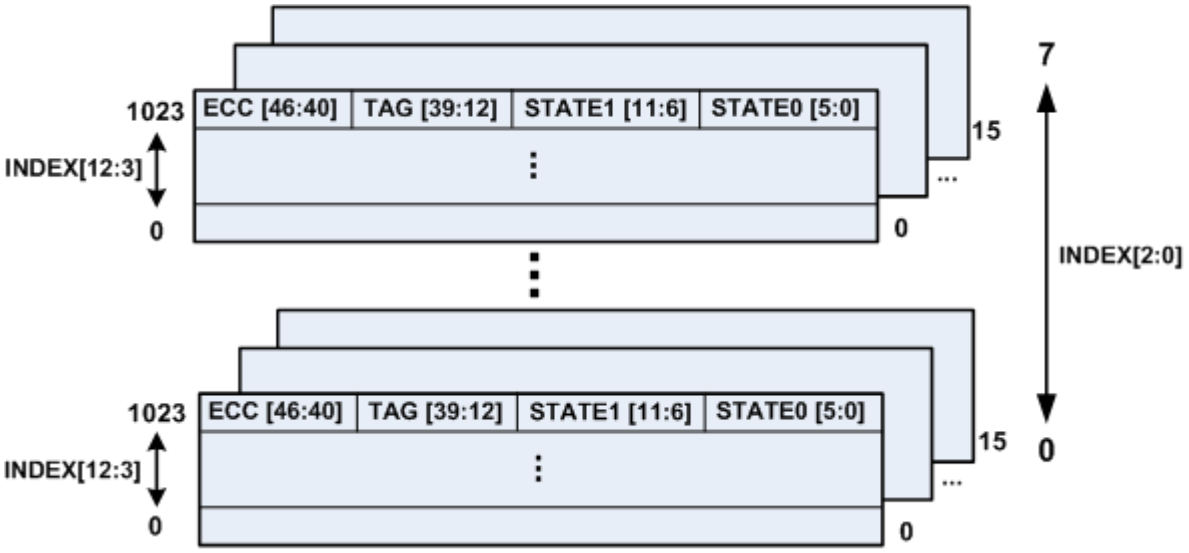


Рисунок 2.4.5 - Организация глобального справочника

2.4.8.3 Общие принципы работы

В глобальный справочник попадают только строки, принадлежащие памяти данного микропроцессора, которыми пользуются один или несколько других микропроцессоров. При этом в глобальном справочнике не хранится информация о нахождении данной строки в кэше своего микропроцессора, данная информация находится в L3-кэше процессора.

При поступлении запроса в глобальный справочник, информация о кэш-строке заводится в том случае, если какой-либо чужой процессор начал работать с памятью данного процессора. Если процессор завершает работу с данной кэш-строкой, информация о ней вычёркивается из глобального справочника и в том случае, если процессор был хозяином данных, передаётся им в оперативную память. Если при записи нового состояния в глобальный справочник необходимо вычеркнуть из него старую строку, необходимо выполнить операцию Read-Invalidate и вытеснить данные из кэшей всех процессоров, которые использовали данную строку. В том случае, если при этом один из процессоров являлся хозяином данных, эти данные должны быть перемещены в память. Таким образом, можно говорить, что между записями в глобальный справочник и чужими строками в кэш-памяти процессоров ставится некое соответствие. Если записи о данной кэш-строке нет в глобальном справочнике, то она не находится в кэше ни одного из чужих процессоров, и если не указан хозяин данных для кэш-строки, то самое актуальное её состояние хранится в оперативной памяти. В том случае, если для данной кэш-строки есть ссылка на хозяина данных, то можно считать, что наиболее актуальные данные хранятся в процессоре, который является хозяином данной строки. И, наконец, в случае пометки о shared-копии данных в каком-либо процессоре, нельзя с точностью сказать, есть ли там данные, или строка уже вычеркнута, однако для корректности когерентного протокола считается, что данные там есть.

Глобальный справочник не является полным, он располагается

исключительно в специализированном кэше глобального справочника внутри процессора, не имеет продолжения в оперативной памяти и не устанавливает взаимно-однозначного соответствия между кэш-строками данного процессора, находящимися в других процессорах, и глобальным справочником этого процессора. Поэтому для работы данного глобального справочника необходимо присутствие в записи как информации о кэш-строке, так и тэга кэш-строки. Принцип работы и эффективность глобального справочника основаны на том, что глобальный справочник перекрывает объём кэш-памяти процессоров и является достаточно большим для того, чтобы считать взаимные конфликты и вытеснения записей о различных кэш-строках редким событием.

Глобальный справочник реализован на однопортовой памяти, для которой запись и чтение в один такт невозможны. Для увеличения пропускной способности, глобальный справочник разбит на 8 независимых по чтению и записи секций. Глобальный справочник может выполнить одно чтение за такт по адресу из одной секции, и одну запись за такт по адресу из другой секции глобального справочника. Для увеличения пропускной способности глобального справочника по чтению, в глобальном справочнике реализован буфер записи состояний Write Buffer (WRB). Размер буфера записи соответствует размеру Release Buffer системного контроллера SC. На стадии d0 конвейера системного контроллера SC в глобальный справочник поступает запрос на чтение. На стадии d3 конвейера системного контроллера глобальный справочник выдает считанную информацию в конвейер SC, причем буфер записи также просматривается. Информация из буфера записи считается наиболее актуальной. На стадии d4 системный контроллер выполняет запись новой информации в глобальный справочник. Новое состояние записывается в буфер записи глобального справочника. Если нет обращений по чтению к нужной из 8 секции, буфер записи переносит состояние в память глобального справочника. После завершения обработки запроса в RLB, состояния из буфера записи гарантированно записывается в память глобального справочника. Отсутствие одновременного доступа к одной секции глобального справочника по чтению и записи

контролируется в SC.

Глобальный справочник не поддерживает переходных состояний. Все конфликты, требующие нахождения кэш-строки в переходном состоянии, решаются в модуле Release Buffer системного контроллера SC. Блокировки, предотвращающие повторную обработку запроса по занятому адресу, предотвращение вытеснения заданного адреса, предотвращение одновременного нахождения одного и того же адреса на разных стадиях конвейера осуществляются в системном контроллере SC.

2.4.8.4 Глобальный справочник для DMA-запросов

Для корректной и быстрой обработки DMA-записей, которые также являются когерентными запросами, но разосланными не от процессорных ядер, а от DMA-контроллеров, находящихся в Host Controller, протокол с поддержкой кэшей DMA-записей реализован внутри Host Controller и специализированных справочников для DMA-записей.

Основной проблемой при реализации работы DMA-записей является сохранение порядка следования записей (DMA-ordering) при конвейерном исполнении этих записей для поддержания высокой пропускной способности DMA-записей (в сравнении с атомарным выполнением DMA-записей, которые исполняются крайне медленно). При этом, если контролирование порядка следования записей осуществляется не в Host Controller, то возможны взаимные блокировки записей, осуществляющихся из разных DMA-устройств, что лишает возможности конвейеризовать записи, делая необходимой их атомарную обработку.

Решением данной проблемы является контролирование порядка записей внутри того устройства, которое их рассылает, то есть, внутри Host Controller. Для этого внутри каждого Host Controller содержится полностью ассоциативная кэш-память на 32 строки, в которую с помощью операции Read Invalidate и Request-For-Owning (Invalidate) подкачиваются и хранятся подложки для DMA-

записей. При этом запрос Read Invalidate высылается тогда, когда Host Controller будет писать 32 байта или менее. Если же необходимо записать кэш-строку целиком, то есть 64 байта, то выполняется запрос Request-For-Owning (Invalidate). Сами записи также выполняются в Host Controller, после чего данные отправляются в память с помощью операции Write Back.

Для того, чтобы процессор обладал информацией обо всех кэш-строках, которые забрал для записи Host Controller, внутри глобального справочника располагается дополнительный справочник DMA-записей. По сути, этот справочник является полной копии кэша DMA-записей, находящегося в

Host Controller, в которой хранятся адреса кэш-строк, которые забрал себе каждый Host Controller, и состояния этих кэш-строк в нём. Глобальный справочник поддерживает расширенный протокол MOI состоящий из следующих состояний:

I – Invalid – строки в кэше DMA-записей нет;

M – Modified – в кэше DMA-записей находятся наиболее свежие данные, которых нет в памяти, либо подложка, на которую будет произведена запись
32 байт или менее; строка из этого состояния может перейти в Owned, если кто-нибудь запросит эту строку по чтению (Read);

O – Owned – строка в кэше DMA-записей была запрошена по чтению и теперь для того, чтобы выполнить запись в эту строку Host Controller должен повторно выполнить Read Invalidate;

M64 – Modified64 – в кэше DMA-записей находятся наиболее свежие данные, которых нет в памяти, либо, если запись 64 байт не успела осуществиться, в кэше нет данных по этому адресу; строка из этого состояния может перейти в Owned64, если кто-нибудь запросит эту строку по чтению (Read);

O64 – Owned64 – строка в кэше DMA-записей была запрошена по чтению и теперь для того, чтобы выполнить запись 64 байт в эту строку Host Controller должен повторно выполнить Invalidate.

Соответственно, перечень возможных событий определяется следующим списком:

cRI – coherent Read-Invalidate, внешний когерентный запрос на чтение и владение кэш-строкой (для записи); сюда относятся все процессорные запросы, которые приводят к рассылке coherent Read-Invalidate, а также запросы Read-Invalidate и Invalidate от других Host Controller;

cR – coherent Read, внешний когерентный запрос на чтение кэш-строки; сюда относятся все процессорные запросы по чтению;

RI – Read-Invalidate, запрос на чтение и владение кэш-строкой (от того же Host Controller);

I – Invalidate, запрос на владение кэш-строкой (от того же Host Controller);

WB – WriteBack.

Запросы из Host Controller для глобального справочника являются глобальными, то есть, внешними. Поэтому каждый такой запрос снупирует всех владельцев кэш-строки, находящихся за межпроцессорными линками, а также в обязательном порядке Home-L3. Память опрашивается только в том случае, если Host Controller посылает запрос Read Invalidate, и глобальный справочник не нашел строки в Modified-состоянии ни в кэш-памяти какого-либо процессора, ни в Host Controller какого-либо процессора. В случае запроса Invalidate от Host Controller запрос в память не высылается.

В случае если внешний запрос находит в DMA-директории строку в состоянии, отличном от Invalid, когерентный запрос всегда отправляется в Host Controller. Работа по направлениям, соответствующим отключенным межпроцессорным линкам, не проводится. Однако если тот находился в состоянии Modified64 или Owned64, то Host Controller может не ответить данными, если он их ещё не успел записать. Таким образом, если строка находится в DMA-директории в состоянии Modified64 или Owned64, выполняется обязательный запрос на чтение строки в оперативную память, чтобы запросчик гарантировано получил данные в ответ на запрос.

Схема переходов когерентных состояний для справочника DMA-запросов выглядит следующим образом (рисунок 2.4.6):

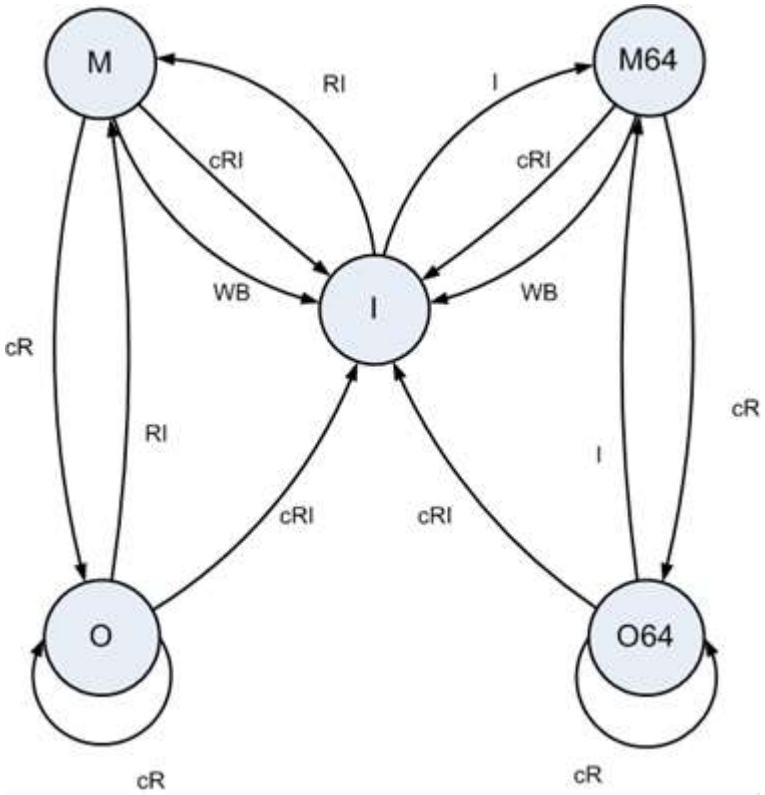


Рисунок 2.4.6 - Схема переходов когерентных состояний

2.5 Контроллер памяти MC

В микропроцессоре имеется восемь независимых контроллеров памяти DDR4 SDRAM(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Разделение адресного пространства микропроцессора между контроллерами производится по заданным адресным разрядам. Номера адресных разрядов задаются на этапе инициализации системы и хранятся в регистре конфигурации микропроцессора.

2.5.1 Технические характеристики

Общие технические характеристики:

Соответствие стандарту ”Jedec Standard. DDR4 SDRAM Specification” (JESD79- 4A).

Максимальный поддерживаемый объем памяти - 128 Гбайт.

DDR4 SDRAM интерфейс ECC & TAGs: 72 бита = 64 Data-бит +

+4 Tag-бита + 4 ECC-бита noECC & noTAGs: 64 бита = 64 Data-бит.

Поддержка модулей памяти Registered (RDIMM), Unbuffered (EUDIMM), Load Reduced (LRDIMM).

Поддержка 3D Stacked чипов памяти (3DS) до глубины 8H.

Поддержка чипов памяти: 512 Mb x 4, 1 Gb x 4, 2 Gb x 4, 4 Gb x 4 256 Mb x 8, 512 Mb x 8, 1 Gb x 8, 2 Gb x 8 128 Mb x 16, 256 Mb x 16, 512 Mb x 16, 1 Gb x 16.

Поддержка длины пакета 32 и 64 байта (BL=4 и BL=8).

Поддержка до 64 одновременно открытых страниц (2 слота X 2 физ. банка X 16 лог. банков).

Поддержка режима 2T (режим расширения фазы команды/адреса на интерфейсе DDR4 SDRAM до 2-х тактов).

Возможность распределения адресов (интерливинг) по физическим банкам.

Оптимизация выполнения последовательности запросов путём планирования очерёдности обращения в память (изменение порядка следования операций).

Функциональные характеристики:

Частота функционирования ядра (mc_clk) - до 800 МГц.

Частота функционирования внешнего интерфейса DDR4 (ddr_clk) - до 3200 Мбит/с.

Размер буфера запросов: 48.

Темп приёма запросов: 2 запроса/такт.

Задержка формирования команды на внешнем интерфейсе: 2 такта ядра +

+ 2 такта DDR4 интерфейса.

Задержка по считыванию данных: 2 такта ядра + 2 такта DDR4 интерфейса.

Интерфейсы контроллера памяти представлены на рисунке 2.5.1.

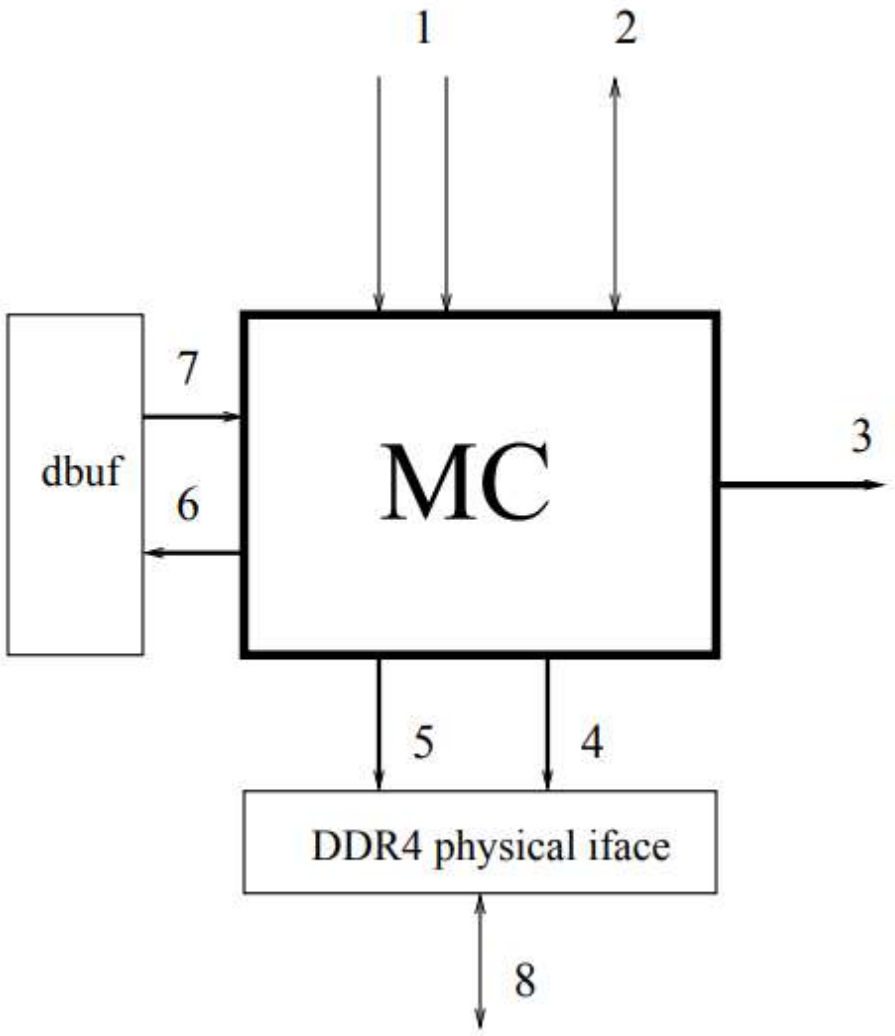


Рисунок 2.5.1 - Внешние интерфейсы контроллера памяти

1 - два порта приёма запросов в память от системы. Интерфейс каждого порта разделён на две группы. Первая группа синхронизируется системным синхросигналом и предназначена для передачи большей части параметров запроса (адрес, код операции, атрибуты, номер ячейки в буфере контроллера памяти). По второй группе передаётся значимость запроса, пересинхронизированная на частоту ядра контроллера (далее, просто контроллера).

2 - интерфейс обмена сообщениями с системным коммутатором. Прием сообщений производится по синхросигналу MC (mc_clk).

3 - выдача параметров считанных данных в систему.

4 - интерфейс взаимодействия с физическим уровнем DDR4: шина команд.

5 - интерфейс взаимодействия с физическим уровнем DDR4: шина данных

(чтение и запись).

6 - выдача номера регистра с данными, запись которых производится в память, в буфер данных.

7 - получение данных для записи.

Структурная схема контроллера памяти представлена рисунке 2.5.2.

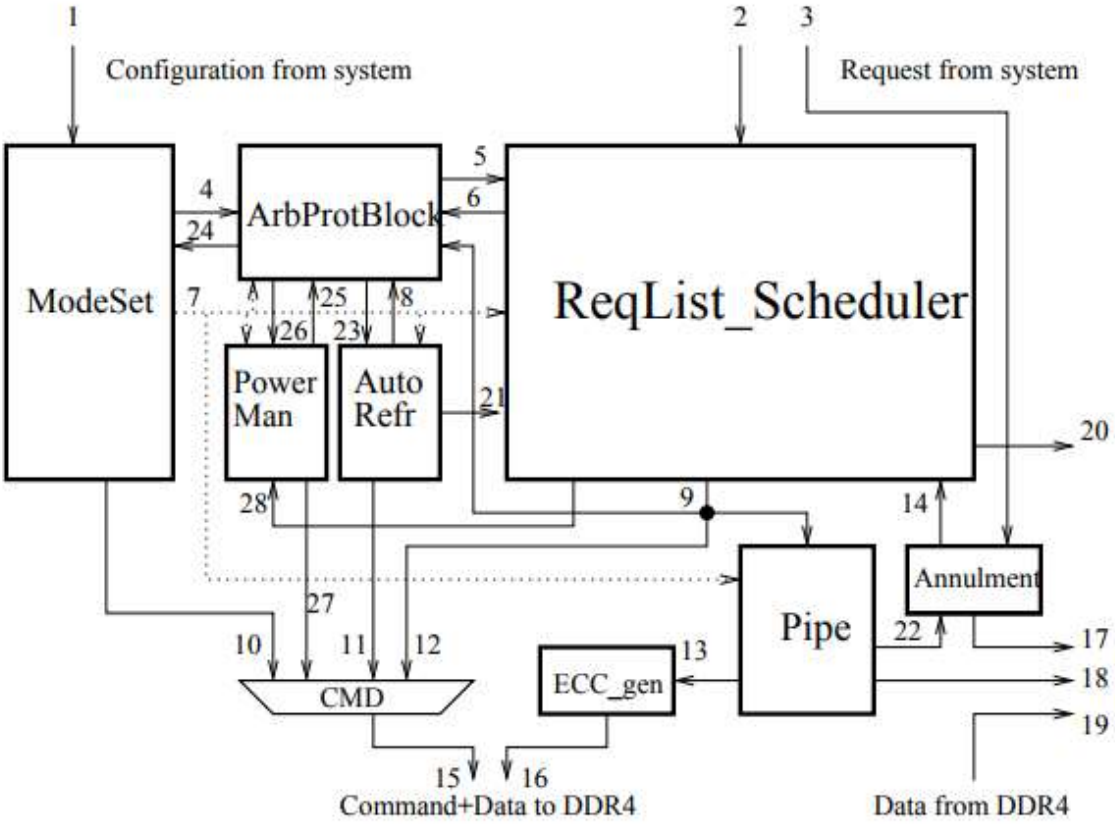


Рисунок 2.5.2 - Контроллер памяти МС

Контроллер памяти МС имеет в своем составе следующие модули.

RequestList_Scheduler - реестр заявок, объединённый с планировщиком. Представляет собой буфер из 26 ячеек и набор фильтров. Каждая ячейка хранит в себе следующие параметры заявки:

VAL - значимость. При помещении заявки в ячейку данный признак устанавливается в активное состояние, при завершении обработки заявки признак

сбрасывается. Регистр работает по mc_clk;

ADDR_ar, ADDR_ac, ADDR_lb, ADDR_bg ADDR_pb - составляющие адреса запроса в терминах интерфейса DDR4. Регистры работают по req_clk;

COP - тип операции. Регистр работает по req_clk;

COPext - расширение кода операции. Регистр работает по req_clk;

DRDY - признак готовности данных (для операций записи). Регистр работает по mc_clk;

STAGE - шкала состояния процесса обработки заявки. Данная информация используется для принятия решения о дальнейших действиях по обработке заявки. Регистр работает по mc_clk;

AGE - возраст заявки. Данный параметр нужен для соблюдения корректной последовательности выполнения заявок с совпадающими адресами доступа (перемежение внутри пар rd→wr, wr→wr, wr→rd с одинаковыми адресами доступа запрещено). Регистр работает по mc_clk;

DST - локальный идентификатор запросчика. Регистр работает по req_clk;

ATTR - системные параметры запроса (номер регистра процессора, идентификатор запросчика и т.д). Данный параметр возвращается в систему вместе со считанными из памяти данными без изменения, имеет параметризуемый размер. Регистр работает по req_clk;

DCELL - номер ячейки буфера данных чипсета, хранящего данные для записи. Регистр работает по req_clk;

SIZE - размер запроса. Параметр используется для терминации burst-посылки (оптимизация) и определения достаточного места для приёма данных по чтению в коммутаторе данных. Регистр работает по req_clk;

SEALED - признак ”залипания” заявки чтения (заявка находится в реестре в течение продолжительного периода из-за того, что она не ”вписывается” в стратегию оптимизации планировщика в данный период).

Приём запросов из системы производится по интерфейсу 2, организованному в виде двух независимых портов. Каждый порт разделён на две группы сигналов: параметры запроса, синхронизованные по системному

синхроимпульсу (req_clk) и значимость запроса (синхронизация mc_clk).

Обе группы сигналов сопровождаются номером ячейки по буферу контроллера памяти. При одновременном поступлении запросов по двум портам более старым полагается запрос, пришедший с порта 1. Слежение за занятостью входного буфера контроллера возложено на систему. Все заявки, принятые от системы, передаются в блок планирования выполнения запросов. Поскольку чипсет, передающий запросы в МС, допускает bypass, имеется механизм отмены запросов по чтению (3). Из блока аннулирования заявок чтения Annulment по линии 14 приходит номер ячейки, содержимое которой подлежит аннулированию. В зависимости от статуса выполнения запроса в данной ячейке либо выполняется ее сброс и сообщение в систему номера освободившейся ячейки (20), либо сигнал аннулирования игнорируется.

Scheduler - планировщик. Осуществляет оптимизацию очерёдности выполнения заявок с целью увеличения эффективной пропускной способности канала памяти. При переупорядочивании учитываются возможные зависимости по адресам между заявками, попадания и промахи в открытые строки модулей памяти, тип последней выполненной операции, наличие ”забытых” заявок (заявки чтения, которые находятся в очереди в режиме ожидания по причине нарушения ими стратегии оптимизации планировщика при попытке их выполнения в течение продолжительного времени) и т.п. В планировщике запрос может быть разбит на несколько субопераций, каждая из которых является зависимой или независимой с точки зрения перемежения с субоперациями других запросов. Например, операция записи с маской разбивается на интерфейсе DDR4 на 2 операции: чтение подложки и запись модифицированных данных.

Annulment - блок аннулирования заявок чтения. Номер ячейки и признак отмены для запроса по чтению передаются в блок аннулирования
по интерфейсу 3. Каждый запрос чтения сопровождается признаком отмены. Признак отмены приходит через несколько тактов или вместе с запросом чтения (величина данной задержки зависит от реализации чипсета), но не позже, чем в контроллер придут данные по чтению из модуля памяти. Запросы по чтению и

соответствующие признаки отмены приходят в МС в порядке очерёдности (обеспечивается чипсетом). Дескриптор отмены сохраняется во внутренней памяти блока Annulment. Если дескриптор содержит указание об отмене, по линии 14 в ReqList_Scheduler направляется номер ячейки, содержимое которой необходимо аннулировать. Сообщение об отмене заявки игнорируется в блоке ReqList_Scheduler, если оно пришло после начала выполнения заявки.

При получении данных чтения из памяти, помимо параметров запросов чтения с выхода конвейера Pipe, к ним присоединяется дескриптор отмен (линия 17), соответствующий номеру ячейки запроса (передающийся в Annulment из Pipe по линии 22). В случае отмены запроса считанные данные в систему не проходят.

Pipe - конвейер управляемой длины для передачи сигналов управления буфером данных по записи, шиной DDR4 SDRAM и выдачей считанных данных в систему. Т.к. между (выдачей команды записи / приёмом данных по чтению) и (выдачей / приёмом) соответствующих этой команде данных на шину DDR4 SDRAM необходимо выдерживать промежуток, определяемый временными параметрами и настройками контроллера и модуля памяти (CAS Latency, Additive Latency, Parity Latency, mT - режим расширения фазы адреса, RM - Registered или Unbuffered модуль памяти), необходимо осуществлять задержку сигналов управления (выдачей / приёмом) данных и управления шиной на соответствующую этим параметрам величину. Сигналы управления длиной конвейера приходят из блока ModeSet (линия 7). Конвейер организован в виде последовательно соединённых линий задержки двух типов: с постоянной и переменной задержками. Фиксированная задержка определяется, исходя из внутреннего устройства контроллера. Переменная задержка определяется в блоке ModeSet в зависимости от установленных модулей памяти. При записи задержанный сигнал управления выдачей данных передаётся по линии 13 в модуль генерации ECC ECC_gen. После выполнения операции Pipe формирует сообщение в чипсет о выполнении запроса освобождения ячейки буфера данных по записи. При адаптации контроллера памяти к физическому уровню DDR4

требуется подстройка глубины очереди параметров для операций чтения под задержку поступления считанных данных.

ECC Gen - генератор битов ECC. Для каждого (256 Data + 16 Tag = 272) - битового блока данных формируются 10 бит ECC. По линии 16 в записываемые данные с ECC передаются в физический уровень DDR4. Вспомогательные функции: генерация битов CRC при включённом режиме Write CRC(включение CRC при записи); инверсия байтов шины данных при включённом режиме Write DBI (инвертирование шины данных при записи).

ECC Check - блок обнаружения и коррекции ошибок ECC. Данный модуль вынесен за пределы контроллера памяти.

AutoRefr - блок планирования циклов регенерации памяти. По линии 7 в модуль передаются значение периода регенерации и признак расширения адресной фазы на шине DDR4 SDRAM до 2 тактов. С соответствующей периодичностью по линии 11 из модуля в арбитр поступает запрос, и после получения разрешения и при отсутствии блокировок (линия 23) на шине DDR4 SDRAM выполняется цикл ре- генерации. Блок планирования поддерживает режим упреждающей регенерации.

ModeSet - блок установки режимов контроллера памяти, программирования режимов модулей памяти и начальной инициализации памяти. При изменении значений параметров (линия 1), относящихся к модулям памяти, из модуля в арбитр поступает запрос (линия 10), и после получения разрешения и при отсутствии блокировок (линия 24) на шину DDR4 SDRAM посылается комплекс команд типа ”Mode Register Set” (команды программирования регистров управления и режима работы модулей памяти). После окончания выполнения команд на линиях режима и параметров работы контроллера памяти (линии 7) устанавливаются значения, соответствующие новому режиму работы модулей. При изменении значений параметров, относящихся только к режиму работы контроллера памяти (т.е. не требующих перепрограммирования регистров управления модулей памяти) и требующих остановки выдачи команд на шину DDR4 SDRAM, модуль также выдает запрос в арбитр (получение разрешения от

арбитра означает, что в данный момент никакие другие команды на шину не выдаются) и при получении разрешения выполняет установку новых значений на линиях параметров и управления контроллера памяти.

PowerMan - блок управления режимами энергосбережения DDR4 SDRAM. По линии 7 в модуль передаются параметры для выбранного режима энергосбережения DDR4 SDRAM. По линии 25 из модуля в арбитр поступает запрос, и после получения разрешения и при отсутствии блокировок (линия 26) DDR4 SDRAM переводится в заданный режим энергосбережения (линия 27) по истечении таймера перевода DDR4 SDRAM в режим энергосбережения при отсутствии заявок в Scheduler (линия 28).

ArbProtBlock - блок арбитража и протокольных блокировок. Подсистема арбитража определяет приоритеты запросчиков на выдачу команды на интерфейс DDR4 SDRAM. До завершения выполнения начальной инициализации модулей памяти подсистемой ModeSet арбитр заблокирован и никто из абонентов не может получить разрешения на доступ к шине DDR4 SDRAM. После завершения начальной инициализации арбитр начинает работу в нормальном режиме.

Приоритет обслуживания абонентов на доступ к шине DDR4:

1 AutoRefr ;

2 ModeSet;

3 PowerMan;

4 ReqList_Scheduler.

Подсистема протокольных блокировок обеспечивает соблюдение временных пауз между последовательно выдаваемыми на шину DDR4 SDRAM командами согласно параметрам работы модулей и контроллера памяти и требованиям стандарта DDR4 SDRAM. Параметры для работы данной подсистемы передаются в ArbProtBlock из ModeSet по линии 7.

ErrRec - блок исправления ошибок чётности адресно-командной шины интерфейса DDR, ошибок CRC на шине данных интерфейса DDR при записи и ошибок ECC на шине данных интерфейса DDR при чтении. Ошибки исправляются путём повтора серии запросов, включающей тот, на котором

произошла ошибка.

2.6 Устройство доступа к внешней памяти XMU

Связь процессорных ядер с пространством оперативной памяти других процессоров системы и с пространством ввода-вывода (совокупность данных пространств далее будет именоваться внешним) осуществляется посредством устройства XMU (eXternal Memory Unit). Доступ к пространству регистров и пространству ввода-вывода выполняется посредством входящего в состав XMU хост-контроллера HC. Также в состав XMU входят: три контроллера канала межпроцессорного обмена IPCC, программируемый контроллер прерываний (EPIC), блок конфигурационных регистров (CR), а также коммутатор XMU (XMU Xbar).

Структурная схема XMU представлена на рисунке. 2.6.1.

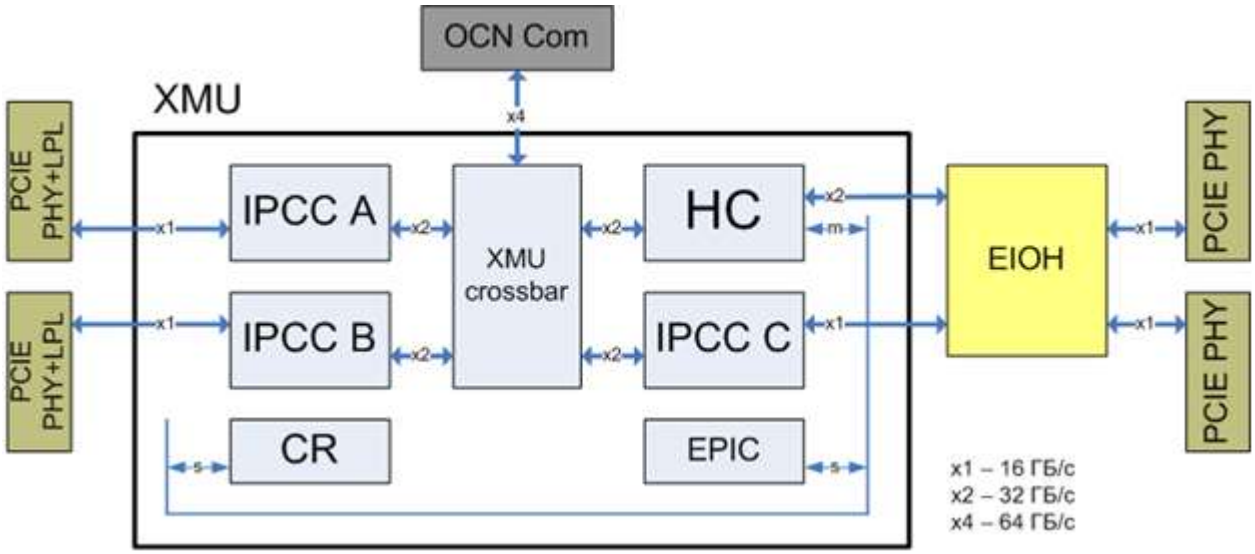


Рисунок 2.6.1 - Устройство XMU

Основными функциями XMU являются:

- пересылка пакетов между агентами, подключенными к OCN, и другими

процессорами;

- выполнение запросов от ядер/L3 всех процессоров 4-процессорного кластера в пространство ввода-вывода;

- выполнение DMA-запросов от устройств ввода-вывода, подключенных к процессору, в память всех процессоров кластера;

- пересылка сообщений между контроллером внешних прерываний IOEPIC и локальным контроллером прерываний EPIC;

- пересылка сообщений между локальным контроллером прерываний EPIC и EPIC других процессоров кластера.

Доступ к пространству регистров и пространству ввода-вывода выполняется посредством устройства Host Controller (HC), который взаимодействует с контроллером канала ввода-вывода WLCC, программируемым контроллером прерываний EPIC и блоком конфигурационных регистров CR.

В состав XMU входят следующие устройства:

HC – контроллер доступа в пространство ввода-вывода;

IPCC A, IPCC B, IPCC C – контроллеры межпроцессорных каналов;

CR – блок конфигурационных регистров процессорного узла;

EPIC – контроллер прерываний;

XMU Crossbar – коммутатор связей.

2.6.1 Контроллер доступа в пространство ввода-вывода HC

Host-контроллер (HC) обеспечивает взаимодействие процессорных ядер с каналом ввода-вывода, а также со встроенным программируемым контроллером прерываний EPIC и блоком конфигурационных регистров. Host-контроллер выполняет следующие функции:

- маршрутизация и обработка запросов от ядер в пространство ввода-вывода (IO-операции);

- осуществление функции DMA-канала для обеспечения доступа внешних устройств к оперативной памяти процессоров кластера (DMA-операции);

- маршрутизация сообщений о прерываниях между контроллером внешних прерываний IOEPIC и EPIC-ми кластера.

Host-контроллер взаимодействует со следующими устройствами:

- распределенный коммутатор XMU Xbar; интерфейсы: канал первичных запросов от процессора в пространство ввода-вывода и конфигурационные регистры, канал снуп-запросов, канал коротких сообщений о ходе выполнения процессорных запросов на запись, канал когерентных сообщений (ответов без данных), канал DMA-запросов, канал данных;

- встроенный контроллер периферийных интерфейсов EIOH; интерфейсы: двунаправленный канал запросов и ответов;

- программируемый контроллер прерываний EPIC; интерфейсы: интерфейс сообщений о прерывании и интерфейс доступа к регистрам EPIC;

- блок конфигурационных (или системных) регистров узла и блок конфигурационных регистров EPIC, взаимодействуют с HC посредством шины AXI.

2.6.1.1 Структура HC

Структурная схема HC представлена на рисунке 2.6.2.

В состав HC входят следующие устройства:

- IORE (IO Requests Executor) – устройство выполнения процессорных запросов (IO-запросов);

- URCE (Upward Requests and Completions Executor) – устройство обработки запросов от внешних устройств, ответов на процессорные запросы, а также снуп-запросов;

- IME (Interrupt Messages Executor) – устройство обработки сообщений EPIC, IOAPIC и MSI;

- Coherence Analyzer – устройство обработки snoop-запросов;

- IOMMU – устройство трансляции виртуальных адресов для запросов от внешних устройств;

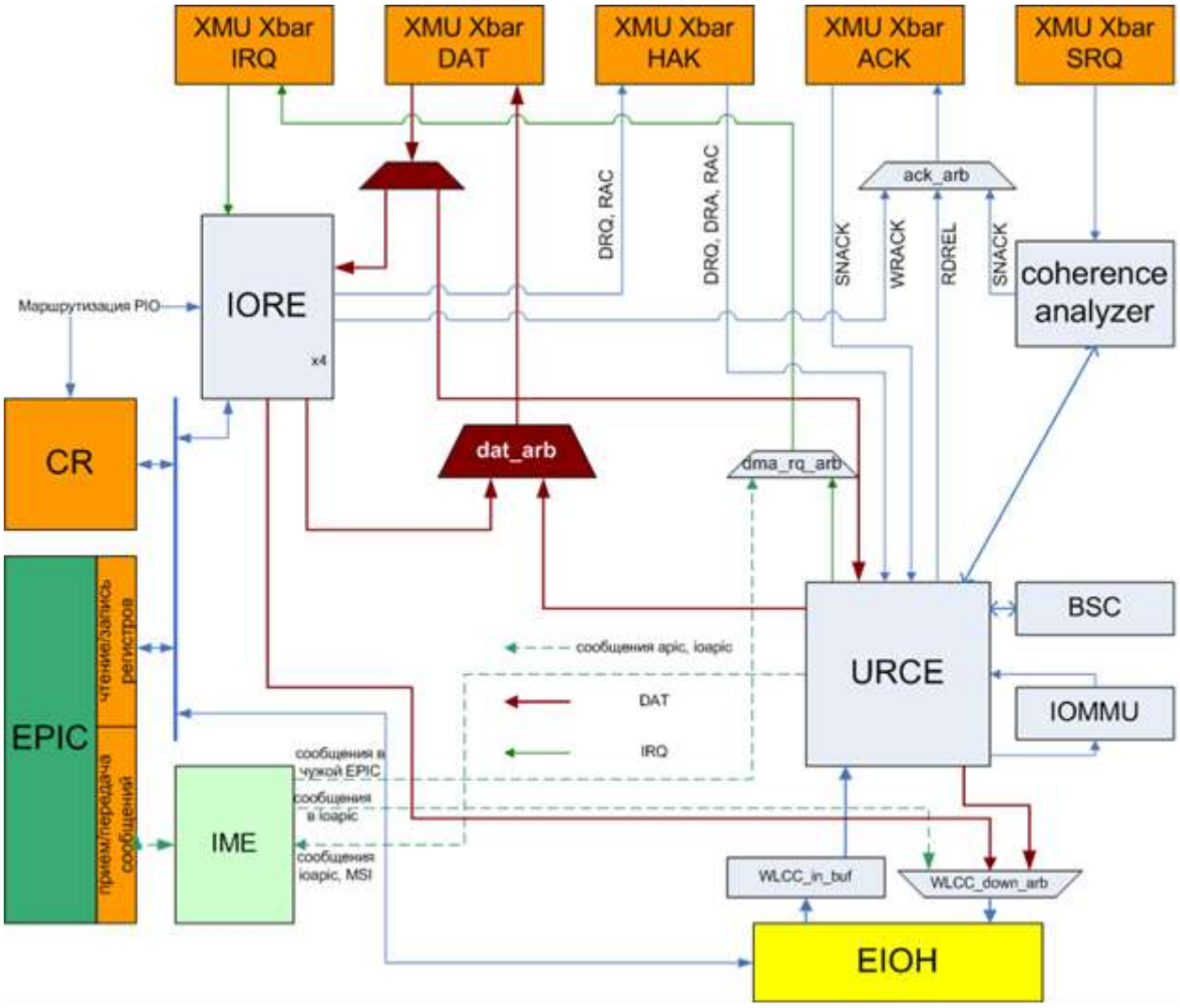


Рисунок 2.6.2 - Ност-контроллер НС

- BSC (Bit Scale Controller) – контроллер битовой шкалы, реализует механизмы защиты памяти по записи;
- WLCC_in_buf – приемный буфер входных транзакций из WLCC.

2.6.1.2 Обработчик запросов и ответов из IO-линка (URCE)

Структурная схема обработчика запросов и ответов из EIOH представлена на риснке 2.6.3.

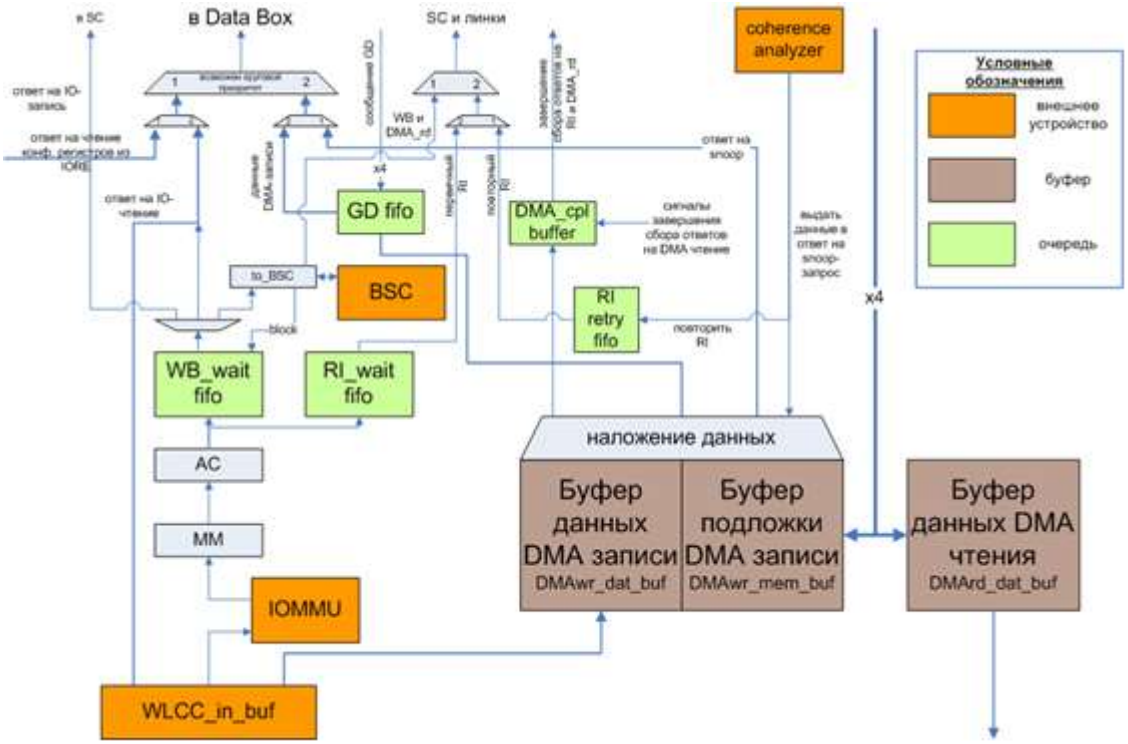


Рисунок 2.6.3 - Обработчик запросов и ответов из EIOH

Обработчик запросов и ответов из EIOH (Upward Requests and Completions Executor, URCE) предназначен для приема всех запросов и ответов из одного виртуального канала EIOH. Основной задачей URCE является контроль порядка выполнения операций.

В состав URCE входят:

- буфер данных DMA-записи;
- буфер подложки DMA-записи;
- буфер данных DMA-чтения;
- блок вычисления адресных конфликтов (Address Conflicts, AC);
- блок защиты памяти маскированием (Memory Masking, MM);
- очередь первичных запросов RI(I) RI_wait fifo и очередь повторных запросов RI RI retry fifo;
- очередь отправки запросов/ответов WB_wait fifo;
- очередь сообщений о завершении RI(I) и DMA read (DMA_cpl buffer);
- очередь сообщений «дай данные» GD fifo.

2.6.1.3 Обработчик запросов в пространство ввода-вывода (IORE)

Структурная схема обработчика запросов в пространство ввода-вывода представлена на рисунке 2.6.4.

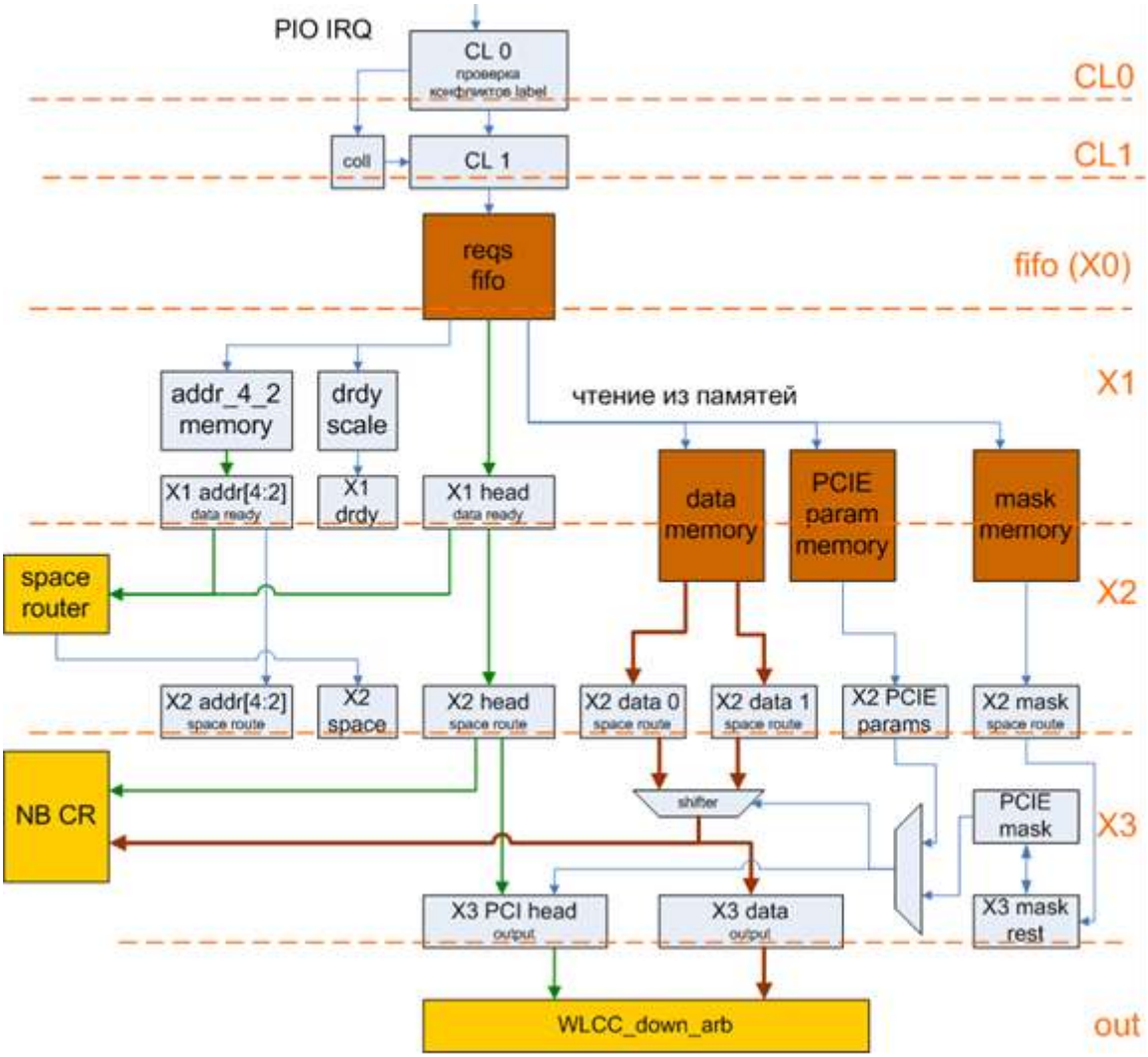


Рисунок 2.6.4 - Обработчик запросов в пространство ввода-вывода

В состав хост-контроллера НС входит обработчик запросов в пространство ввода-вывода (Input/Output Requests Executor, IORE). Основной задачей этого блока является прием и исполнение всех процессорных запросов в пространство

ввода-вывода и обращений к системным регистрам, регистрам WLCC, регистрам IOMMU, регистрам HC и регистрам EPIC.

В состав IORE входят:

- входная очередь запросов (reqs fifo);
- буферы данных, масок и параметров PIO-записей (data memory, PCIE param memory, mask memory);
- буфер параметров исполняемых запросов по записи Write_buf;
- буфер параметров исполняемых запросов по чтению Read_buf;
- очередь свободных идентификаторов запросов по чтению Read_nreg_fifo;
- очередь свободных идентификаторов запросов по записи Write_nreg_fifo;
- очередь коротких сообщений о ходе процессорных записей Write_cpl_fifo.

2.6.1.4 Устройство IOMMU

Структурная схема IOMMU представлена рисунке 2.6.5.

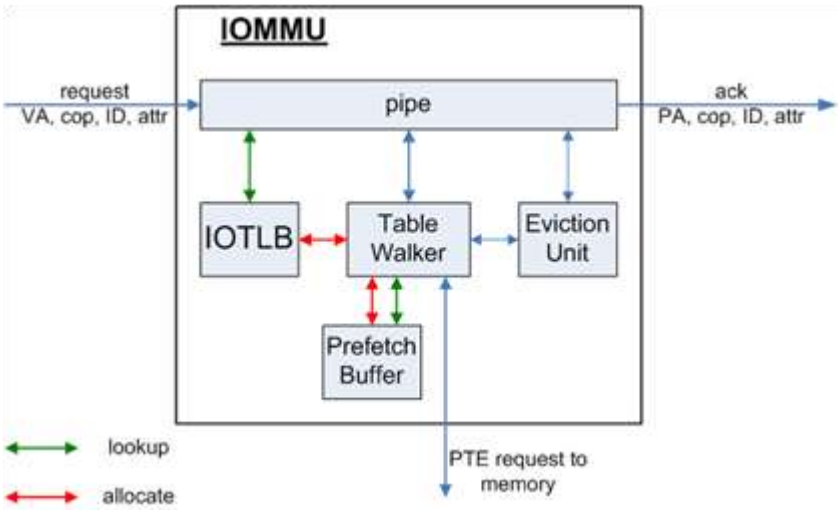


Рисунок 2.6.5 - Устройство IOMMU

Устройство IOMMU выполняет трансляцию виртуального адреса (VA) DMA-запроса в физический (PA). Трансляция выполняется в соответствие с таб-

лицей трансляции (Таблица страниц, Page Table, PT), которая расположена в оперативной памяти.

Виртуальные адреса транслируются в физические постранично, единственный поддерживаемый размер страницы – 4 Кбайт. Каждой виртуальной странице соответствует 4-байтный элемент таблицы трансляции (Page Table Element, PTE).

2.6.1.5 Обработчик сообщений о прерываниях IME

Интерфейсы обработчика сообщений и прерываний IME представлены на рисунке 2.6.6.

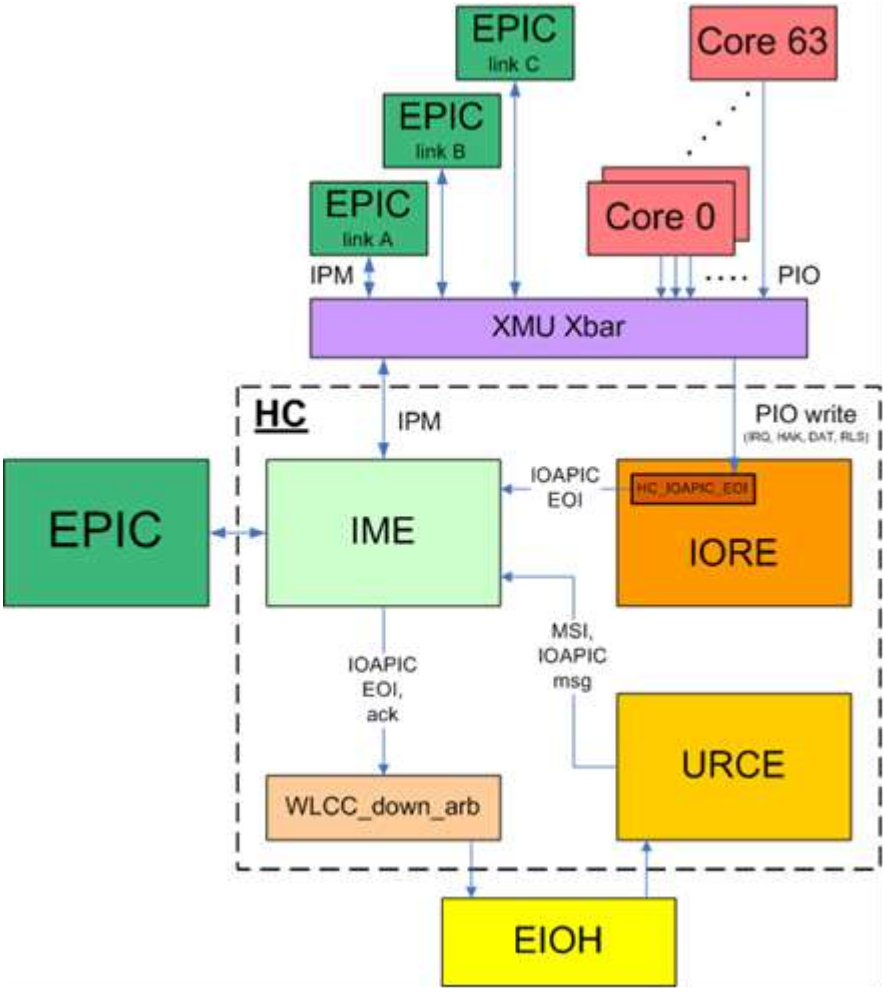


Рисунок 2.6.6 - Интерфейсы IME

IME (Interrupt Messages Executor) – устройство обработки сообщений о прерываниях. Устройство предназначено для маршрутизации и пересылки сообщений между EPIC-ми системы, внешними устройствами, поддерживающими MSI, и IOAPIC-ми внешнего КПИ. Сообщения о прерываниях в NUMA-системе могу передаваться между следующими абонентами:

- EPIC своего процессора;
- EPIC другого процессора;
- IOAPIC, расположенный в КПИ, подключенного к своему процессору;
- внешнее устройство, поддерживающее MSI.

2.6.1.5.1 Типы сообщений

Сообщение MSI поступает из EIOH как запрос по записи 4 байт по 56-разрядному адресу в область MSI, которая задается регистром RT_MSI. Сообщение поступает в WLCC_in_buf, затем в URCE проходит весь тракт обработки пакетов и из очереди WB_wait выдается в IME. IME после преобразований выдает его в EPIC.

Сообщение от IOAPIC обрабатывается аналогично MSI, но требует других преобразований. После получения сообщения от IOAPIC IME отправляет в EIOH подтверждение получения прерывания (сообщение IOAPIC ack).

Сообщения IPM (межпроцессорное прерывание) передаются между EPIC и EPIC-ми других процессоров. При поступлении сообщения от EPIC оно упаковывается в IME и отправляется в процессор-получатель по каналу IPM. Номер процессора-получателя определяется по одному из полей сообщения. При получении сообщения IPM из другого процессора IME распаковывает его и передает в EPIC.

Сообщение IOAPIC ack (о получении прерывания от IOAPIC) отправляется из IME в EIOH после получения сообщения от IOAPIC.

Сообщение EOI (End Of Interrupt в IOAPIC) требуется для снятия прерывания в IOAPIC. При поступлении в НС PIO-запроса по записи регистра HC_IOAPIC_EOI из IORE в IME выдаются данные записи по этому запросу. После этого IME отправляет сообщение EOI в EIOH.

2.6.2 Контроллер прерываний EPIC

Контроллер прерываний EPIC (Elbrus Programmable Interrupt Controller) в составе XMU представлен на рисунке 2.6.7.

Elbrus Programmable Interrupt Controller - программируемое устройство, предназначенное для фиксирования, хранения и доставки внешних и межпроцессорных прерываний ядрам вычислительной системы.

Контроллер может использоваться в многопроцессорных системах, содержащих до 1024 ядер, и поддерживает 10-разрядные вектора для маскируемых прерываний (доступные вектора: 1-1024, нулевой вектор запрещен).

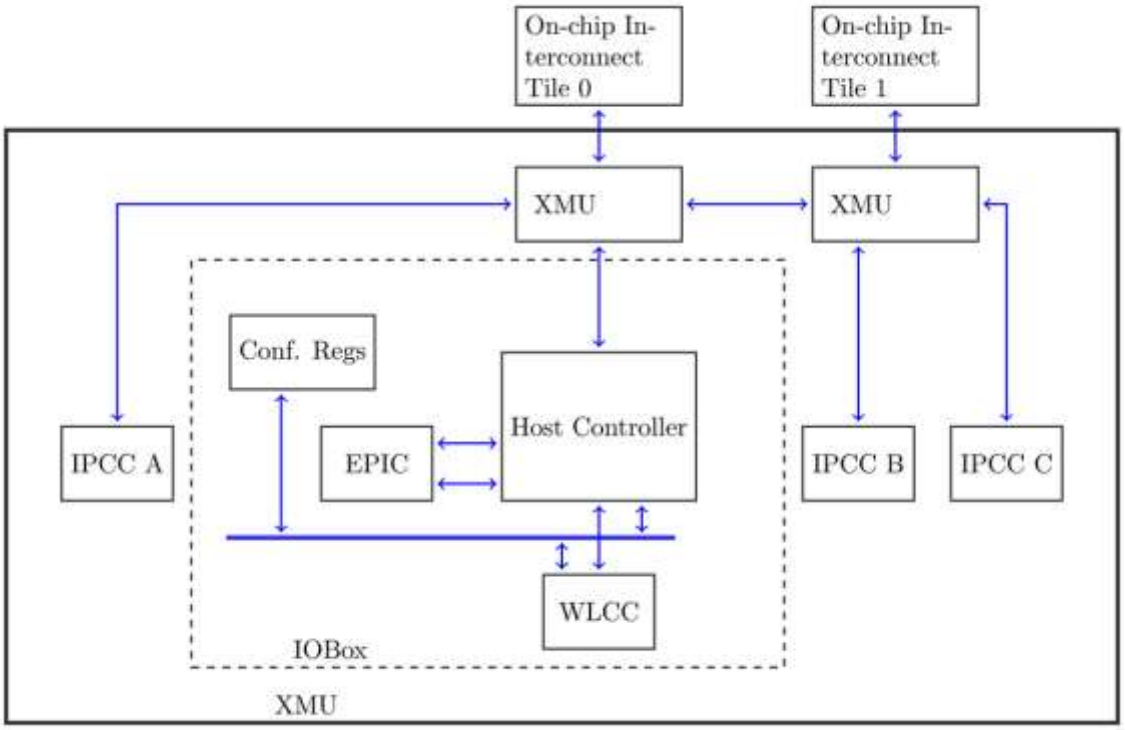


Рисунок 2.6.7 - Контроллер прерываний EPIC в составе XMU

Контроллер EPIC состоит из следующих элементов:

PREPIC (Processor Elbrus Programmable Interrupt Controller) - один на процессор (кристалл). Расположен в модуле epic;

CEPIC (Core Elbrus Programmable Interrupt Controller) - по одному на каждое ядро микропроцессора. Расположены в модуле epic;

IOEPIC (Input Output Elbrus Programmable Interrupt Controller) - может быть расположен вне кристалла процессора (часть чипсета "Повозка") или быть интегрированным в кристалл процессора.

Управление контроллером осуществляется через набор программно доступных регистров в выделенной области пространства физических адресов.

Элементы системы обработки прерываний (CEPIC, PREPIC, IOEPIC) обмениваются сообщениями (как в рамках одного кристалла, так и в 4-процессорной системе). Сообщения формируются аппаратно.

Контроллер реализует аппаратную поддержку виртуализации системы прерываний.

2.6.2.1 Программно-доступные регистры EPIC

Каждому ядру соответствует набор регистров cepic_*.

Набор регистров prepic_* - один на процессор.

Для аппаратной поддержки виртуализации:

- введены гипер-привилегированные регистры, доступные только гипервизору;
- в каждом CEPIC реализован дополнительный набор регистров.

Разделение между запросами хоста и гостя происходит по признаку host

(1 - обращения хоста, 0 - обращение гостя) в запросе.

Хост может обращаться к гостевому набору регистров через выделенную область адресов.

Область, отведенная под EPIC: (0xfee0_0000 - 0xfee0_ffff), адреса в таблице указаны относительно начала области (0xfee0_0000).

Области:

(0x0000 - 0x0fff) - адреса регистров CEPIC;

(0x1000 - 0x1fff) - адреса гипер-привилегированных регистров;

(0x2000 - 0x2fff) - адреса регистров PREPIC.

2.6.2.2 Структура EPIC

Структура EPIC представлена на рисунке 2.6.8

EPIC содержит:

- Core EPIC (CEPIC) для каждого ядра на кристалле;
- Processor EPIC (PREPIC).

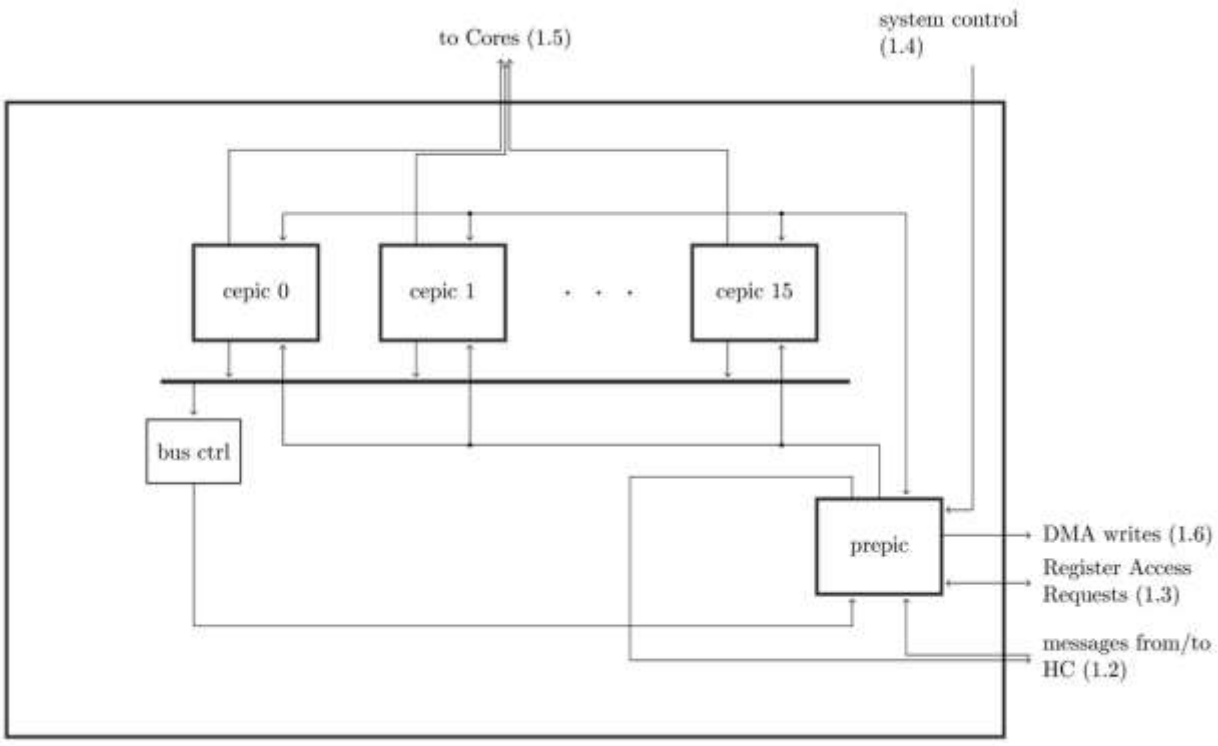


Рисунок 2.6.8 – Контроллер прерываний EPIC

2.6.2.3 Функции EPIC

Функции EPIC:

- принимает, обрабатывает и выдает ответы на запросы к регистрам системы прерываний;

- формирует, фиксирует, хранит и доставляет прерывания ядрам вычислительной системы;

- реализует аппаратную поддержку виртуализации системы прерываний.

2.6.2.3.1 Нумерация CEPIC

PREPIC идентифицируется номером процессора (PREPIC_ID.prepicn).

CEPIC идентифицируется номером процессора и номером ядра (CEPIC_ID.cepicn). Различают топологический номер ядра (соответствует расположению процессорных ядер на кристалле) и физический (аппаратура нумерует включенные ядра по порядку в процессе инициализации).

Топологический номер ядра используется при доставке запросов.

Физический номер ядра:

- содержится в CEPIC_ID;

- используется при доставке сообщений.

2.6.2.3.2 Запросы к регистрам EPIC

Запросы от ядер к регистрам EPIC приходят строго по одному. Запросы буферизуются в хост-контроллере НС. Перед отправкой в EPIC очередного запроса хост-контроллер ожидает ответа на предыдущий.

Запросы от ядер к некоторым регистрам (CEPIC_PMIRR, CEPIC_ICR, CEPIC_DAT) могут быть 4-байтными и 8-байтными (к остальным регистрам - только 4-байтные).

Опкоды обращений к регистрам EPIC:

W32MNSNP - запись (4 или 8 байт - определяется маской);

R4NS, R8NS – чтения.

Для обращения к регистрам CEPIC_PMIRR, CEPIC_PNMIRR помимо обычной записи предполагается запись "по-или". Запись "по-или" производится обычной записью в дополнительно выделенный для каждого из указанных регистров адрес.

Доставка запрос конкретному CEPIC’у производится по физическому номеру ядра.

2.6.2.3.3 Прерывания

2.6.2.3.3.1 Источники прерываний

Прерывания в процессоре могут формироваться:

- устройствами CEPIC, посредством регистров:
  - CEPIC_ICR (межпроцессорные прерывания);
  - CEPIC_EPIC_INT (служебные прерывания);
  - CEPIC_TIMER, CEPIC_NM_TIMER (прерывания по таймерам);
- устройством PREPIC, посредством регистров:
  - PREPIC_LINP (локальные системные прерывания);
  - PREPIC_EPIC_INT (служебные прерывания);
  - внешними устройствами (внешние прерывания).

2.6.2.3.3.2 Внешние и межпроцессорные прерывания

Внешние и межпроцессорные прерывания:

- межпроцессорные прерывания - прерывания от одного процессорного ядра другому. Межпроцессорные прерывания формируются CEPIC’ом при записи в регистры CEPIC_ICR. Межпроцессорные прерывания могут быть как маскируемыми, так и немаскируемыми;

- внешние прерывания формируются внешними устройствами и проходят через IOEPIC или IOAPIC. Данные из IOEPIC/IOAPIC (вектор прерывания и идентификатор ядра, которому предназначено прерывание) попадают в хост-контроллер, и упаковываются в сообщение формата EPIC. Сформированное сообщение направляется в соответствующий EPIC. Внешние прерывания могут быть только маскируемыми.

2.6.2.3.3.3 Маскируемые и немаскируемые прерывания

EPIC поддерживает два типа прерываний: маскируемые и немаскируемые. Каждый из двух типов имеет свой вход в аппаратуру ядра (mi и nmi соответственно) и поддерживается своим набором регистров.

Маскируемым прерываниям соответствуют 1023 вектора, разделенные на 4 группы приоритета. Маскируемые прерывания предназначены для обслуживания системных и пользовательских программ: организации внешних обменов, взаимодействия различных процессов, таймеров и т.д. Маскируемые прерывания передаются на регистр CEPIC_CIR (если он свободен) устройства CEPIC, принадлежащего выбранному для обработки прерывания ядру. Если этот регистр занят, делается запись о прерывании на регистр отложенных прерываний CEPIC_PMIRR. Отложенные прерывания обслуживаются по мере освобождения ресурсов.

Немаскируемые прерывания являются более приоритетными, чем маскируемые, и предназначены для:

– управления системой (startup, init, nmi, smi);

– поддержки двоичной компиляции (nm_special, int_violat).

Для немаскируемых прерываний в каждом устройстве CEPIC имеется регистр CEPIC_PNMIRR с полным набором битов для всех типов немаскируемых прерываний. Если нужный бит занят необработанным предыдущим прерыванием, признак нового на него накладывается.

2.6.2.3.3.4 Доставка прерываний в процессорное ядро

Обработка немаскируемого прерывания начинается с чтения регистра CEPIC_PNMIRR. Результат чтения - бит-вектор значимостей прерываний, которые имеют место. Чтение регистра включает блокировку выхода немаскируемых прерываний в процессорное ядро. После завершения обработки прерывания обработчик должен сделать запись в CEPIC_PNMIRR. Единица в разряде данных при записи погасит соответствующий разряд CEPIC_PNMIRR. Запись в регистр снимает блокировку входа немаскируемых прерываний в процессорное ядро. Если на регистре немаскируемых прерываний остались значимости прерываний, вход nmi процессорного ядра вновь будет выставлен в "1".

Обработка маскируемого прерывания начинается с чтения регистра CEPIC_VECT_INTA (выполнение цикла INTA). При этом программа обработчика прерывания получает вектор текущего прерывания и текущий приоритет процессорного ядра. Данная информация должна сохраняться программой-обработчиком до конца обработки прерывания.

Чтение регистра CEPIC_VECT_INTA освобождает регистр CEPIC_CIR (туда может поместиться новое прерывание).

После завершения процедуры обработки прерывания (перед возвратом на прерванную программу) выполняется запись в регистр CEPIC_EOI (выполняется команда End of Interrupt). Данные в запросе на запись должны содержать значение приоритета процессорного ядра (прерванной программы), которое используется для восстановления значения регистра CEPIC_CPR.

Если обработанное прерывание было уровневым, программа должна выдать дублирующую запись end_of_interrup в IOEPIC, где расположен источник прерывания.

2.6.2.3.3.5 Приоритеты прерываний

Определены 4 класса приоритетов маскируемых прерываний (биты 9:8 вектора прерывания), представлены в таблице 2.6.1.

Таблица 2.6.1 Приоритет прерываний

Приоритет	Диапазон векторов прерываний
4	1023 - 768
3	767 - 512
2	511 – 256
1	255 - 1

Priority = vector[9:8] + 1

Регистр CEPIC_CPR содержит текущий приоритет ядра (приоритет текущей задачи ядра).

Если на регистр CEPIC_CIR принимается маскируемое прерывание, приоритетный класс которого выше текущего приоритета ядра (CEPIC_CIR.vect[9:8] + 1) > CEPIC_CPR.cpr формируется сигнал прерывания в ядро (mi).

Текущий приоритет ядра может быть программно прописан при переключении задачи записью в регистр CEPIC_CPR (по умолчанию приоритет задачи равен нулю).

Если приходит прерывание, приоритет которого ниже или равен содержимому CEPIC_CPR, оно сохраняется на CEPIC_CIR и ждет понижения приоритета ядра.

1 Программное восстановление CEPIC_CPR.

В момент чтения регистра CEPIC_VECT_INTA, текущий приоритет ядра считывается вместе с вектором прерывания и должен сохраняться обработчиком прерывания до выхода из процедуры.

Регистру CEPIC_CPR при чтении регистра CEPIC_VECT_INTA аппаратно присваивается значение приоритета вектора прерывания, которое становится новым значением текущего приоритета ядра.

В момент выполнения записи в регистр CEPIC_EOI (команда End of interrupt) происходит восстановление приоритета ядра (данные запроса на запись в CEPIC_EOI.rcpr).

CEPIC_CIR всегда содержит наиболее приоритетное из ожидающих прерываний.

Если

- на регистре CEPIC_CIR содержится прерывание с некоторым приоритетом P0 и

- в CEPIC возникает новое маскируемое прерываний с приоритетом

P1 > P0 (прерывание может прийти в виде сообщения типа fixed; быть сгенерировано по таймеру; или ожидать на CEPIC_PMIRR),

то

- CEPIC_CIR примет значение, соответствующее более приоритетному прерыванию, менее приоритетное прерываний будет помещено на CEPIC_PMIRR,

за исключением случаев, когда

- новое прерывание, возникает одновременно с чтение регистра CEPIC_VECT_INTA.

2 Ложное прерывание.

Ложное прерывание выдается при чтении CEPIC_VECT_INTA, если mi не выставлен (к примеру, в ситуации, когда вследствие повышения процессором уровня приоритета ядра, CEPIC снимет сигнал прерывания (mi), а процессор уже выдаст запрос по чтению CEPIC_VECT_INTA. В этом случае CEPIC будет возвращать ложное прерывание (spurious interrupt), вектор которого хранится в регистре CEPIC_SVR.

2.6.2.3.3.6 Прерывания по таймерам

В каждом CEPIC 2 таймера: маскируемый и немаскируемый.

Счетчик таймера начинает работу при записи в CEPIC_TIMER_INIT. Счетчик CEPIC_TIMER_CUR при этом принимает записываемое значение. Вспомогательный счетчик (программно недоступен) устанавливается в соответствии со значением CEPIC_TIMER_DIV. Счетчики отсчитывают в сторону уменьшения значения. По достижению нуля, формируется прерывание (если оно не замаскировано).

Маскируемый таймер управляется набором регистров EPIC_TIMER_{LVTT, INIT, CUR, DIV}.

При формировании прерывания по этому таймеру статусный бит (12 бит CEPIC_TIMER_LVTT) устанавливается в 1 (если прерывание по таймеру не замаскировано соответствующим полем).

Соответствующий вектор затем должен быть принят либо на регистр текущего прерывания CEPIC_CIR, либо на регистр отложенных маскируемых прерываний CEPIC_PMIRR. При получении подтверждения о помещении прерывания на CEPIC_CIR или CEPIC_PMIRR статусный бит сбрасывается в ноль.

Немаскируемый таймер управляется набором регистров CEPIC_NM_TIMER_{LVTT, INIT, CUR, DIV}.

При формировании прерывание по этому таймеру всегда выставляется статусный бит (12 бит CEPIC_NM_TIMER_LVTT) (не предполагается маскирование на уровне таймера). Затем прерывание помещается на регистре CEPIC_PNMIRR. (Прерывание может быть замаскировано соответствующим битом регистра CEPIC_PNMIRR_MASK).

2.6.2.3.3.7 Прерывание физзащиты от неожиданных DMA обращений

При срабатывании физзащиты от неожиданных DMA обращений хост-контроллер HC выставляет сигнал `int_violat' (интерфейс 1.4) и ожидает

`int_violat_ack' - завершения доставки соответствующего прерывания всем CEPIC-ам в системе (включая CEPIC-и процессорных ядер других процессоров в многопроцессорной системе).

EPIC, получивший "1" на входе `int_violat' , гарантирует доставку соответствующего немаскируемого прерывания (поле `int_violat' регистра PNMIRR) во все процессорные ядра многопроцессорной системы.

Хост-контроллер HC, выдавший `int_violat', приостанавливает обработку DMA-обращений до тех пор, пока EPIC не выдаст подтверждение `int_violat_ack'.

2.6.2.3.4 Сообщения

Для передачи информации используются сообщения.

Сообщения передаются частями по 40 бит, каждое сообщение содержит адресную часть и одну часть с данными.

Исключение - гостевое сообщение о внешнем прерывании. В этом случае сообщение содержит дополнительную часть, содержащую базовый адрес для формирования DMA-записи

Протокол сообщений поддерживается аппаратными средствами и программно не доступен.

2.6.2.3.4.1 Типы сообщений

Определено несколько типов сообщений.

1 Сообщение о прерываниях (`msg_t == 0').

Сообщения о прерывании могут формироваться устройствами CEPIC, PREPIC, и хост-контроллером (из данных, полученных от IOEPIC/IOAPIC) или внешними устройствами, которые поддерживают MSI (Message Signalled Interrupts).

Устройства с поддержкой MSI при организации работы по прерываниям должны пользоваться стандартным для EPIC форматом сообщения о прерывании.

2 Управляющее сообщение DAT.

Управляющие сообщения DAT формируются устройствами CEPIC и обслуживают таблицу адресов назначений (Destination Address Table).

На управляющее сообщение DAT есть 2 типа квитанций:

- квитанция *datack* на управляющее сообщение DAT.

Формируется в PREPIC при чтении/записи/вычеркивании строки в DAT;

- квитанция *bgiack* на управляющее сообщение DAT.

При записи строки в DAT (во время постановки гостя на процессорное ядро) необходимо убедиться, что в аппаратном BGI не осталось сообщений о прерываниях, предназначенных данному гостю.

Когда все сообщения о прерываниях, помещенные в аппаратный BGI до записи строки в DAT, записаны в оперативную память, PREPIC формирует квитанцию *bgiack* в CEPIC, сформировавший управляющее сообщение DAT.

3 Внутренние сообщения: *retint* (returned interrupt), *clricr* (clear ICR).

Сообщения *retint* и *clricr* используются для взаимодействия CEPIC и PREPIC одного процессора, не предполагается отправка/получение этих сообщений на интерфейсе PREPIC <-> HC.

Сообщение *retint* формируется PREPIC'ом при получении от своего CEPIC гостевого сообщения о прерывании, доставка которого предполагается через перехват (смотрите CEPIC_EPIC_INT). А именно:

- широковещательное гостевое прерывание;

- одиночное гостевое прерывание, предназначенное отложенному гостевому ядру (промах по DAT), в случае если int_hv == 1.

Сообщение *clricr* формируется PREPIC'ом при получении от своего CEPIC сообщения о прерывании:

- хостовского сообщения о прерывании;

- гостевого сообщения о прерывании, предназначенного активному гостевому ядру (попадание по DAT);

- гостевого сообщения о прерывании, предназначенного отложенному гостевому ядру (промах по DAT), в случае если int_hv == 0.

2.6.2.3.4.2 Адресация сообщений

Каждому элементу системы EPIC (PREPIC, CEPIC) соответствует идентификатор. Идентификаторы используются при адресации сообщений EPIC.

Не предполагается изменение CEPIC_ID/PREPIC_ID в процессе работы.

PREPIC_ID (для хостовской ОС) совпадает (аппаратно подставляется) со значением поля RTLCFG.pn.

В регистрах и сообщениях, поля src/dst должны иметь следующую структуру:

- dst[9:8] - номер процессора;

- dst[7:0] - номер ядра.

Адресация при включенной аппаратной поддержке:

- гостевая ОС может иметь собственную (независимую от хостовской) нумерацию CEPIC_ID/PREPIC_ID.

Все сообщения в системе прерываний сопровождаются номером гостевой ОС (*gst_id* номером гостя).

- нулевой *gst_id* соответствует сообщениям о прерываниях для хостовской ОС.

- ненулевой *gst_id* соответствует сообщения о прерываниях для гостевой ОС.

Для доставки гостевых сообщений о прерываниях, гостевой идентификатор ядра преобразуется в хостовский.

Изменение гостевых CEPIC_ID/PREPIC_ID должно происходить до того, как будут сформированы какие-либо сообщения для данной гостевой ОС.

Процессорные ядра, расположенные в одном микропроцессоре, имеют одинаковый хостовский номер процессора (идентификатор PREPIC, PREPIC_ID.prepicn]). Гостевые ядра могут иметь любой PREPIC_ID.prepicn.

2.6.2.3.5 Аппаратная поддержка виртуализации системы прерываний

1 Виртуализация обращений к регистрам EPIC.

Для каждой гостевой ОС предусмотрены расположенные в оперативной памяти управляющие структуры, содержащие, помимо прочего, виртуальный EPIC.

Виртуальный EPIC отражает состояние системы прерываний с точки зрения гостевой ОС и содержит копии регистров CEPIC и PREPIC (кроме гипер-привилегированных).

Обращении гостевой ОС к регистрам EPIC, а также события, связанные с внешними прерываниями для гостевой ОС, перехватываются и обрабатываются гипервизором.

Аппаратная поддержка виртуализации системы прерываний позволяет обрабатывать обращения гостевого ядра к регистрам CEPIC без перехвата.

2 Виртуализация обращений к регистрам внешних устройств.

Для организации доступа гостя к регистрам внешних устройств используется соответствующая таблица устройств гостя, расположенная в его адресном пространстве (виртуальном с точки зрения гипервизора).

Таблица устройств гостя формируется при его "раскрутке" на основе содержимого таблицы внешних устройств гипервизора и содержит аналогичные физические базы регистров внешних устройств.

Каждое внешнее устройство, в том числе и подключенное через IOEPIC, может:

- управляться гипервизором;

- быть отдано под непосредственный контроль гостевой ОС ("проброс устройства").

Если конкретное внешнее устройство предоставлено в непосредственное распоряжение гостевой ОС, гостевой драйвер внешнего устройства может загружать DMA обмены напрямую своими драйверами, используя для организации прерывания гостевые адреса пересылаемых массивов, гостевые идентификаторы ядер и гостевые вектора прерываний.

В случае, если устройство не принадлежит гостевому ядру (оно отсутствует в его таблице внешних устройств), обращение к регистрам этого устройства перехватывается гипервизором, управление внешним обменом передается гипервизору.

При этом гипервизор может:

- либо сам организовать обмен и затем инжектировать прерывание о завершении обмена гостю;

- либо взять на себя только организацию очереди по обращению к данному устройству и в нужный момент внести изменение в IGT, временно передав устройство гостю, а затем вернуть ему управление, предоставив самому организовать обмен.

Аппаратно поддержаны:

- виртуализация обращений к регистрам CEPIC (обращение гостевой ОС к регистрам PREPIC аппаратно не поддержано);

- доставка прерываний гостевому ядру.

Аппаратная поддержка включается установкой поля `virt_en' в регистре PREPIC_CTRL.

2.6.2.3.5.1 Номер гостевой ОС

Все объекты, работающие с прерываниями, помечаются номером гостевой ОС, который позволяет идентифицировать принадлежность прерывания на всех этапах доставки:

- ядра содержат системный регистр номера гостя;

- внешние устройства, поддерживающие MSI или работающие через IOEPIC, содержат индивидуальный номер устройства, которому сопоставляется номер ОС.

Ненулевой номер гостя означает, что объект оборудования или сообщение о прерывании предназначено указанной гостевой ОС.

Нулевой номер гостя соответствует хостовской ОС.

2.6.2.3.5.2 Гостевой набор регистров CEPIC

Для работы с прерываниями гостевого ядра в каждом CEPIC предусмотрен дополнительный набор регистров, с которым гостевая ОС работает непосредственно. Это позволяет обрабатывать обращения гостя к регистрам CEPIC и доставлять прерывания гостевым ядрам без выхода в режим гипервизора.

Гостевые регистры:

- восстанавливается из управляющей структуры гостя при постановке на ядро;

- сохраняется в управляющей структуре гостя при переключении гостей.

В гостевом наборе регистров фигурируют гостевые вектора прерываний (независимые от векторов хостовских ядер) и гостевые идентификаторы процессоров и ядер.

2.6.2.3.5.3 Служебные прерывания

При работе EPIC в гостевом режиме могут возникать служебные прерывания. Такие прерывания формируются аппаратурой (посредством регистров CEPIC_EPIC_INT) и представляют собой маскируемые прерывания предназначенные гипервизору. Вектора для этих прерываний указываются программно.

Служебные прерывания обслуживают ситуации:

- доставка межпроцессорного прерывания отложенному (неактивному) гостевому ядру;

- сообщение об этом прерывании будет возвращено в CEPIC-источник прерывания. Информация о прерывании будет помещена на регистры CEPIC_EPIC_INT2, CEPIC_EPIC_INT3. Гипервизору будет выдано служебное прерывание с параметрами указанными в CEPIC_EPIC_INT.

Гипервизор, получив служебное прерывание, читает регистры CEPIC_EPIC_INT2, CEPIC_EPIC_INT3 и фиксирует прерывание в управляющей структуре гостя.

2.6.2.4 Контроллер IOEPIC

Контроллер IOEPIC предназначен для трансформации проводных прерываний от внешних устройств в сообщения о прерываниях.

Контроллер IOEPIC содержит входные буферы прерываний, набор регистров, порт по записи - чтению в эти регистры и схемы для формирования и передачи в шины сообщений о прерываниях.

2.6.2.4.1 Message Signaled Interrupts (MSI) и сообщения EPIC

Контроллер IOEPIC выдает прерывания формата MSI (запрос на запись, содержащий 4 части по 4 байта каждая). Указанные части формируются в соответствии со значением программно доступных регистров IOEPIC *ioepic_message_haddr*, *ioepic_message_laddr*, *ioepic_message_data*, *ioepic_int_rid*.

{*ioepic_message_haddr*, *ioepic_message_laddr*} составляют адрес, старшие разряды ([64:20]) которого используются при маршрутизации.

Младшие 20 разрядов `ioepic_message_laddr' заполняется программно в соответствии с адресной частью сообщения о прерывании формата EPIC (`atr' и `src' не заполняется):

- [19:16] – резерв;

- [15:06] - *dst*: идентификатор ядра, которому предназначено прерывание (хостовский или гостевой);

- [5] – резерв;

- [4:2] - *msgt*: тип сообщения EPIC (3'h0 - сообщение о прерывании);

- [1:0] – резерв.

ioepic_message_data заполняется программно в соответствии с данными в сообщении о прерывании формата EPIC:

- [31:20] – резерв;

- [19:16] – резерв;

- [15:13] - *dlvm*: режим доставки прерывания:

  - 2'h0, 2'h1: fixed;

  - 2'h2: smi;

  - 2'h3: nm_spec;

  - 2'h4: nmi;

  - 2'h5: init;

  - 2'h6: startup;

  - 2'h7: reserved;

- [12:10] – резерв;

- [9:0] - *vect*: вектор прерывания.

ioepic_inr_rid заполняется программно в соответствии со значением идентификатора устройства-источника прерывания (*rid*, requester ID, busnum, devnum, funcnum}) по которому, как результат неполной трансляции виртуального адреса, определяется номер гостя (*gst_id*, 12 бит) и физический базовый адрес виртуального EPIC соответствующего гостя (*gst_base*, 36 бит).

Хост-контроллер, получивший MSI от IOEPIC, формирует сообщение о прерывании (формата EPIC) в PREPIC своего процессора.

Разряды [29:28] адресной части формируемого сообщения (соответствующие положению поля src[9:8]) при этом аппаратно устанавливается в соответствии с номером процессора *pn0*.

epic_addr = {10'b0, pn0, 8'b0, ioepic_message_laddr[19:0]};

epic_data0 = { 8'b0, gst_id, ioepic_message_data[19:0]};

epic_data1 = { 4'b0, gst_base[35:0]}.

2.6.3 Коммутатор XMU Crossbar

Коммутатор XMU Crossbar (рисунок 2.6.9) состоит из 5 портов 3 типов:

1 IPCC A port (тип ipcc port);

2 IPCC B port (тип ipcc port);

3 IPCC C port (тип ipcc port);

4 hc port (тип hc port);

5 ocn port (тип ocn port).

Каждый тип порта является оберткой универсального порта x_port, в которой установленные необходимые для каждого типа параметры. Кроме того, порт содержит маршрутизатор пакетов внутри XMU Crossbar.

2.6.4 Универсальный порт коммутатора XMU

Универсальный порт коммутатора x_port (рисунок 2.6.10) состоит из 6 портов каналов передачи пакетов. Каналы соответствуют системным интерфейсам (с учетом направления передачи). Каждый порт канала является экземпляром универсального порта канала port.

2.6.5 Универсальный порт канала port

Универсальный порт канала port представлен на рисунке 2.6.11.

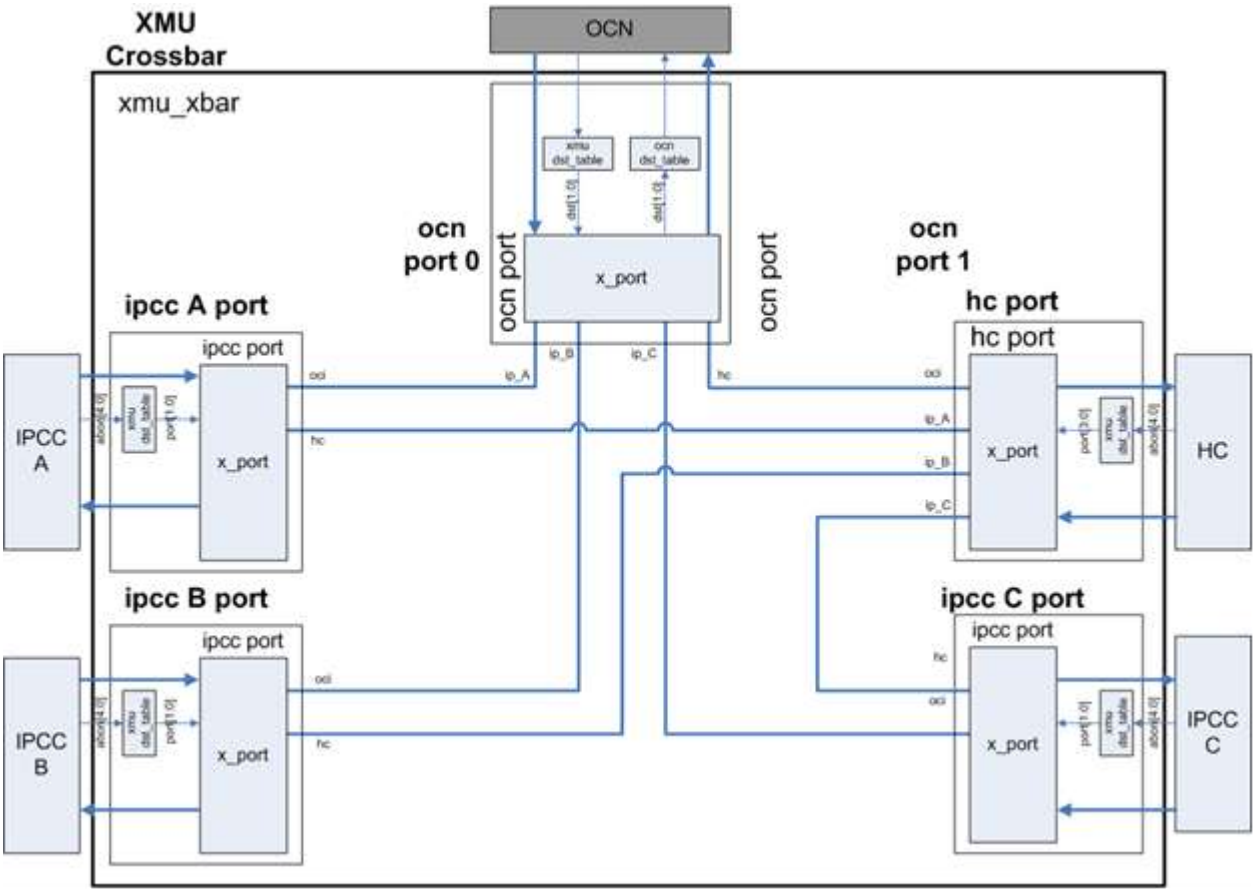


Рисунок 2.6.9 - Коммутатор XMU Crossbar

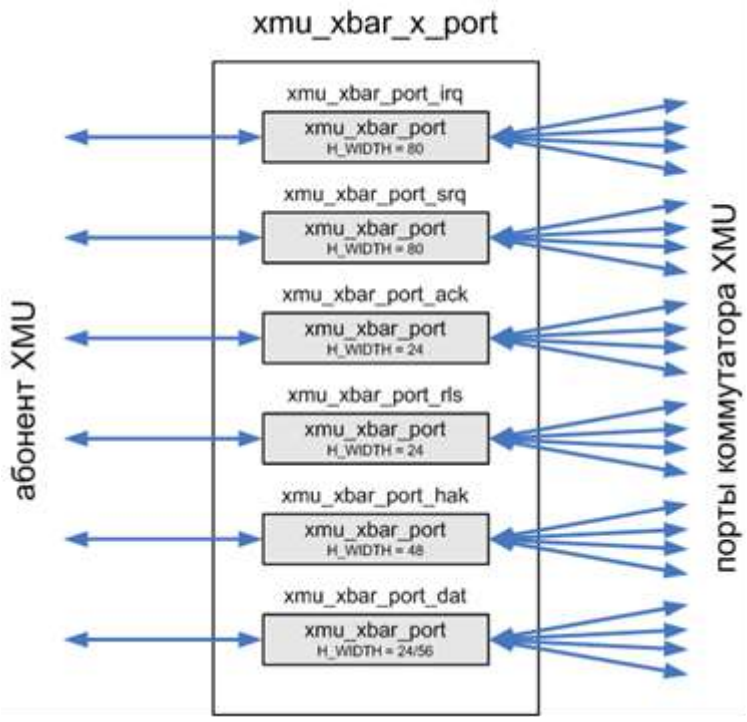


Рисунок 2.6.10 - Универсальный порт коммутатора XMU (x_port)

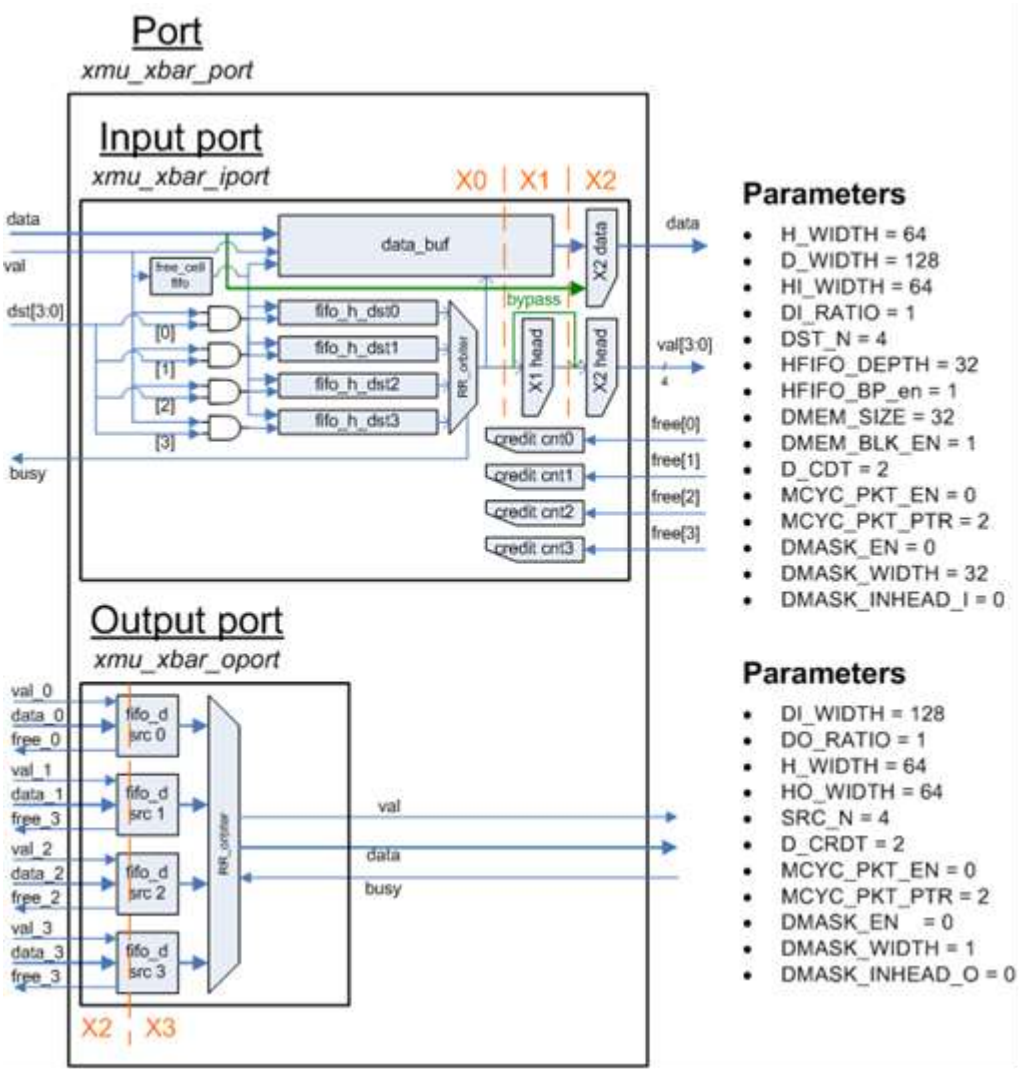


Рисунок 2.6.11 - Универсальный порт канала port

2.6.7 Контроллер межпроцессорного канала IPCC

Контроллер межпроцессорного канала IPCC (Inter-Processor Communication Controller) является контроллером высокоскоростного канала обмена.

Контроллер IPCC предназначен для организации многопроцессорных систем с общей памятью. Микропроцессор имеет в своем составе 3 контроллера IPCC (IPCCA, IPCCB, IPCCC). Это позволяет организовать 4-процессорную систему на общей памяти (рисунок 2.6.12).

Контроллер IPCC логически разделен на два уровня: контроллер канального уровня DLL (Data Link Layer) и контроллер управления физическим уровнем LPL (Logical Physical Layer).

Краткие характеристики:

- внешний линк - динамически масштабируемая (x4, x8, x16) последовательная шина;

- символьная (1 символ - 10-битовый вектор кода «*8/10*») пропускная способность одной линии 500 МегаСимволов/с для каждого из направлений;

- максимальная символьная пропускная способность линка (соответствует ширине линка - x16) 8 ГигаСимволов/с для каждого из направлений.

- максимальная байтовая пропускная способность канала обмена: 8 Гбайт/с для каждого из направлений;

- суммарная пропускная способность всех межпроцессорного каналов процессора составляет 24 Гбайт/с для каждого (прием/передача) из направлений.

2.6.7.1 Контроллер канального уровня DLL

Контроллер канального уровня DLL предназначен для преобразования потоков данных между контроллером физического уровня LPL и модулем системного интерфейса процессора, поддержания работоспособности внешнего линка, контроля целостности переданных и принятых с внешнего линка данных, учета ресурсов удаленного абонента и передачи информации удаленному абоненту об имеющихся локальных ресурсах для приема данных.

Структурная схема контролллера DLL представлена на рисунке 2.6.13.

Контроллер DLL осуществляет:

- формирование и выдача в модуль LPL команды начальной инициализации линка после окончания системного сброса;

- прием от CPU и выдача в LPL команд на:

  - нниициализацию линка;

  - переинициализацию линка;

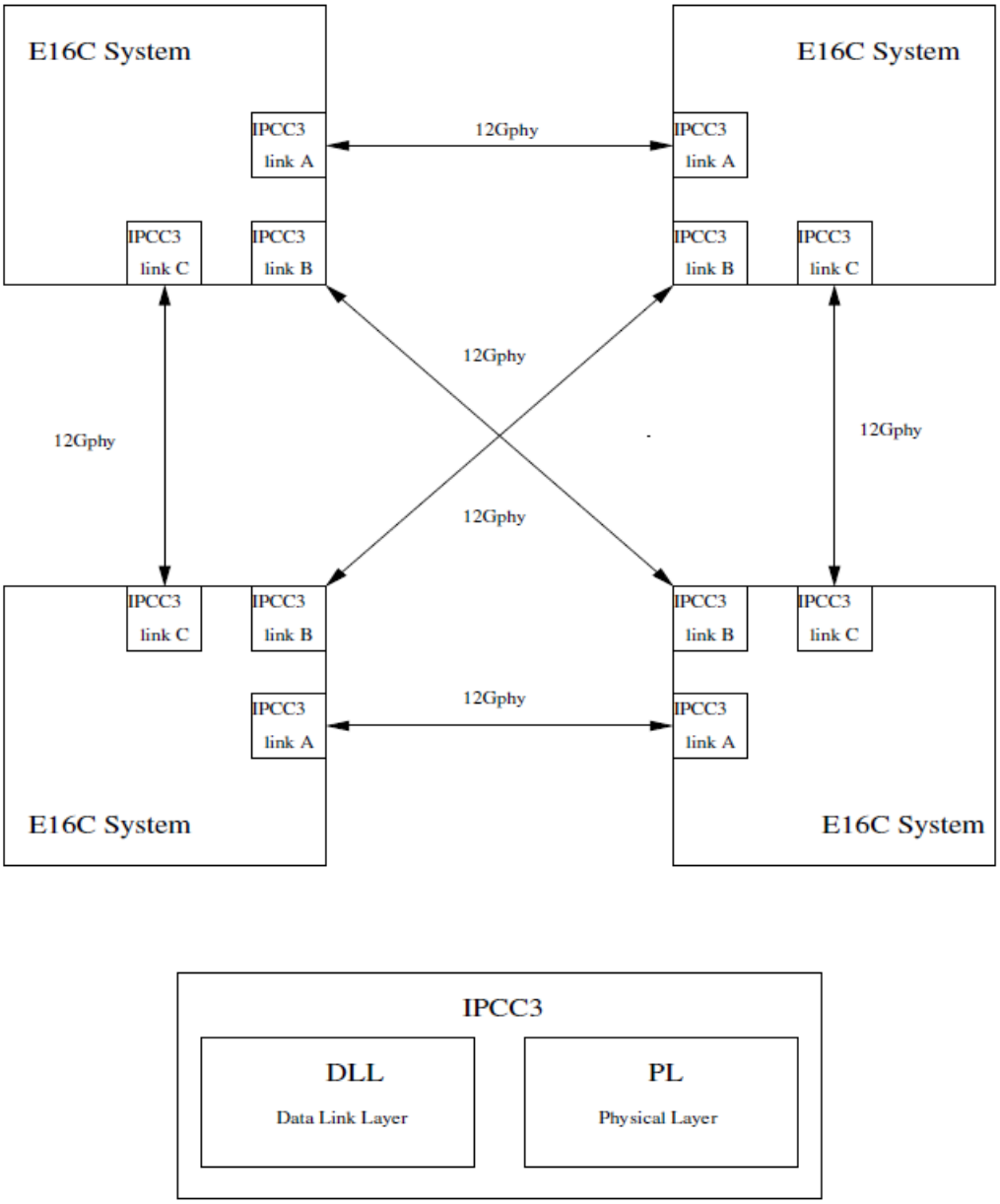


Рисунок 2.6.12 - Структурная схема 4-процессорной системы

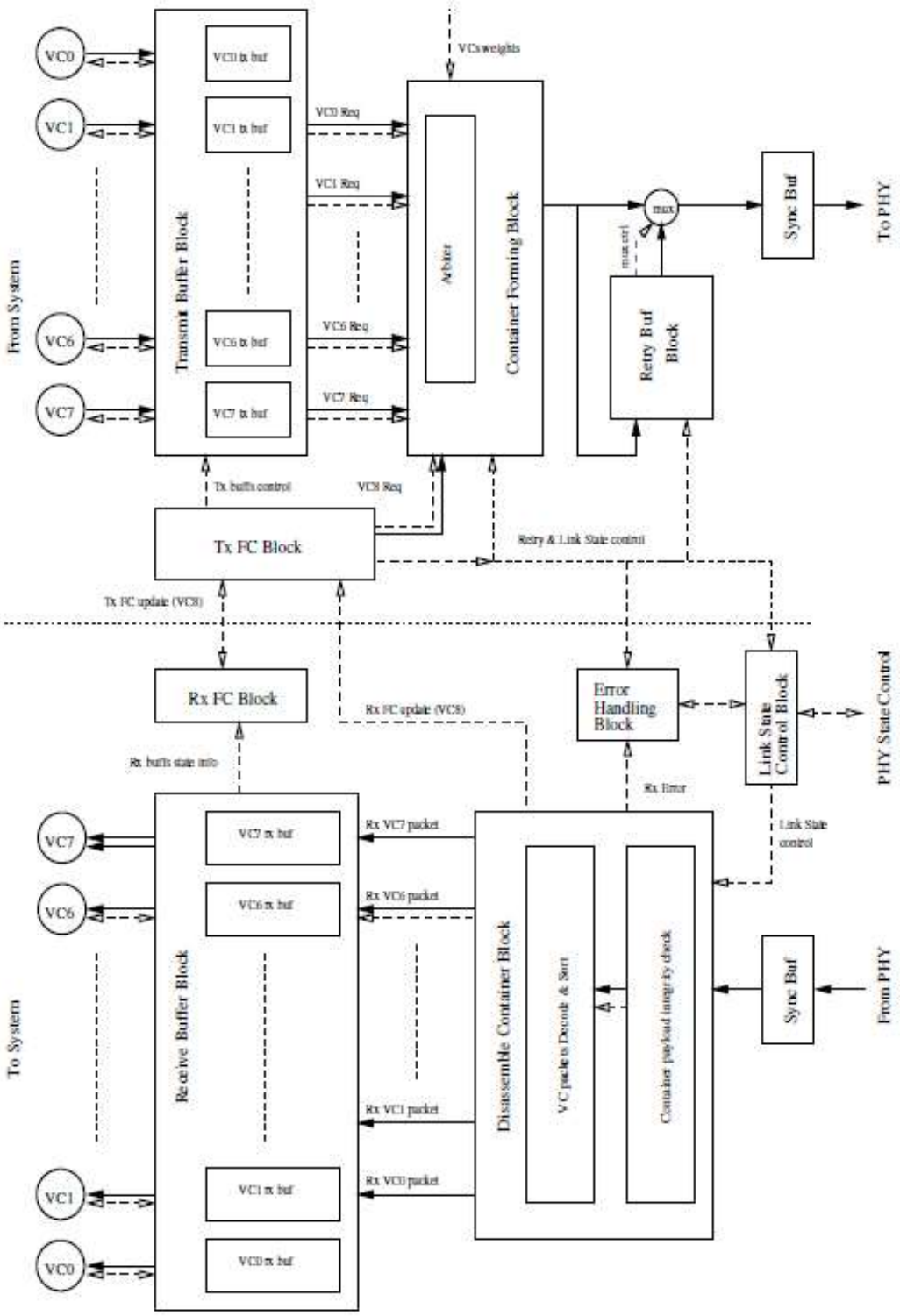


Рисунок 2.6.13 – Контроллер DLL

- переконфигурирование линка (изменение ширины и/или скорости обмена);

- перевод линка в одно из состояний: ”*Sleep*”, ”*Disable*”, ”*PowerOff*”;

- ведение учета локальных ресурсов и ресурсов удаленного абонента;

- формирование и передача (посредством модуля LPL) удаленному абоненту служебных пакетов управления ресурсами;

- прием (посредством модуля LPL) служебных пакетов управления ресурсами от удаленного абонента;

- преобразование принятых от CPU транзакций в пакеты для выдачи в модуль LPL;

- формирование номера пакета;

- формирование поля контрольной суммы *CRC* пакета;

- формирование маркеров начала и конца пакета;

- анализ наличия ресурсов удаленного абонента перед выдачей пакетов в LPL;

- повторная выдача группы пакетов в LPL для передачи на внешний линк в случаях получения запроса на повторную выдачу или отсутствия подтверждения приема указанной группы от удаленного абонента;

- формирование команды на восстановление или переинициализацию внешнего линка в случае обнаружения признаков нарушения его работы;

- передачу в регистры статуса CPU информации о результатах выполнения команд от CPU;

- передачу в регистры статуса CPU информации о состоянии внешнего линка;

- прием пакетов обмена от удаленного абонента (посредством модуля LPL), анализ целостности данных;

- формирование и выдачу в LPL служебных пакетов подтверждения корректного приема данных или служебных пакетов запроса на повторную передачу из-за нарушения целостности данных.

Контроллер IPCC_DLL содержит следующие устройства:

- буфер передатчика Transmit Buffer Block;
- блок упаковки контейнеров Container Forming Block;
- блок повтора передач Retry Buf Block;
- буфер приемника Receive Buffer Block;
- блок распаковки контейнеров Dissassemble Container Block.

Контроллер IPCC_DLL имеет внешние интерфейсы:

- системный канал приема/передачи транзакций;
- канал приема/передачи данных с контроллером LPL.

Системный канал приема/передачи транзакций подключен к фильтру пакетов межпроцессорных каналов IPCC_FLT. Системный канал имеет 7 виртуальных каналов VC 0 - 6, каждый из которых предназначен для передачи транзакций определенного типа:

vc0 — первичные запросы от хост-контроллера HC;

vc1 — первичные запросы от кэша L3$;

vc2 — запросы за данными;

vc3 — снуп запросы и сообщения APIC;

vc4 — данные для записи;

vc5 — считанные данные:

vc6 — короткие снуп-ответы и сообщения о завершении операций.

При выдаче транзакций контроллер IPCC_DLL формирует транспортные пакеты передачи (контейнеры), содержащие наборы разнотипных транзакций из разных виртуальных каналов, тем самым минимизируя время обслуживания этих транзакций. При приеме транзакций контроллер IPCC_DLL распаковывает контейнеры и распределяет транзакции по виртуальным подканалам.

2.6.7.2 Контроллер управления физическим уровнем LPL

Контроллер физического уровня LPL (Logical Physical Layer) предназначен для преобразования потоков данных между контроллером канального уровня DLL (Data Link Layer) и физической средой передачи (межпроцессорным каналом). Структурная схема контроллера физического уровня LPL представлене на рисунке 2.6.14.

Контроллер имеет в своем составе следующие интерфейсы:

- внешний канал - динамически масштабируемая (x4, x8, x16) последовательная шина (полный дуплекс 6 Гбит/с + 6 Гбит/с на линию);

- интерфейс с модулем DLL, тракт передачи - 128 бит, 1000 МГц, 16 Гбайт/с;

- интерфейс с модулем DLL, тракт приема - 128 бит, 600 МГц, 9,6 Гбайт/с.

Контроллер LPL состоит из двух блоков:

- электрический блок физического уровня - EPL (Electrical Physical Layer);

- логический блок физического уровня - LPL (Logical Physical Layer).

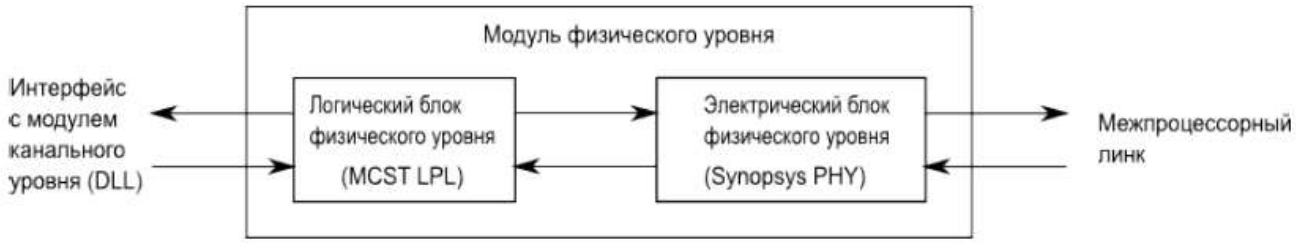


Рисунок 2.6.14 – Контроллер физического уровня LPL

Электрический блок физического уровня EPL осуществляет преобразование входящего битового потока межпроцессорного канала в символьный поток

(1 символ == 10 бит), направляемый в приемный канал модуля LPL, а также преобразование символьного потока из передающего канала модуля LPL в исходящий битовый поток межпроцессорного канала. Кроме того блок EPL формирует сигналы тактовой синхронизации для функционирования модуля LPL, а также осуществляет детектирование абонента на удаленном конце канала по команде от модуля LPL и передает результаты детектирования в LPL.

Логический блок физического уровня LPL контроллера межпроцессорного канала взаимодействует с одной стороны с блоком EPL, с другой стороны с контроллером канального уровня DLL.

Блок LPL осуществляет:

- начальную инициализацию блока EPL;

- исполнение служебных команд от системы;

- прием потока данных из DLL и его кодирование для выдачи в модуль EPL;

- прием потока данных из модуля EPL и его декодирование для передачи в DLL.

2.7 Встроенный контроллер периферийных интерфейсов EIOH

Струкутрная схема встроенного контроллера периферийных интерфейсов EIOH (Embedded IO Hub) представлена на рисунке 2.7.1.

2.7.1 Программная модель контроллера периферийных интерфейсов

С программной точки зрения контроллер периферийных интерфейсов представляет собой дерево PCI-устройств. Самым верхним в данной структуре является мост PCI-to-PCI (устройство номер N), подключенный к виртуальной шине номер 0. Номер N задается программно с помощью регистров <IOHUB_DevNum.N> и <IOHUB_DevNum.val> по следующему правилу.

Если <IOHUB_DevNum.val>==1, то N берется из <IOHUB_DevNum.N>.

Если <IOHUB_DevNum.val>==0, то мост PCI-to-PCI отвечает на все конфигурационные запросы с полем DevNum = k, где k может принимать любое значение от 0 до 31.

Табличное представление программной модели встроенного контроллера периферийных интерфейсов EIOH представлено в таблице 2.7.1.

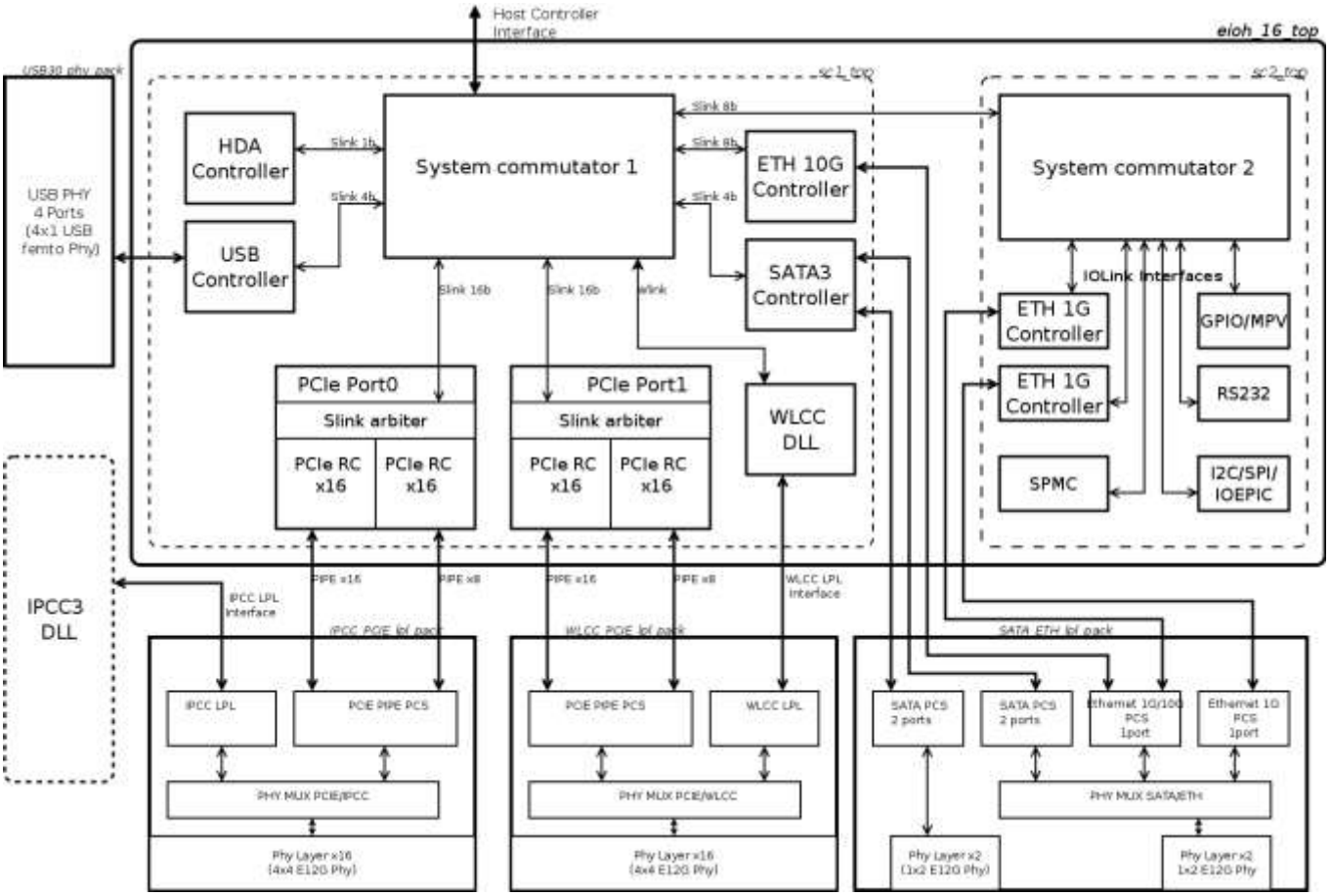


Рисунок 2.7.1 - Встроенный контроллер периферийных интерфейсов EIOH

Таблица 2.7.1 - Табличное представление программной модели встроенного контроллера периферийных интерфейсов EIOH

Bus:Device:Function	Описание функции
Bus 0: Dev N:Func 0	Мост PCI-to-PCI
Bus m: Dev 0:Func 0	USB 3.0
Bus m: Dev 1:Func 0	Ethernet1G_0
Bus m: Dev 1:Func 1	Ethernet1G_1
Bus m: Dev 2:Func 0	GPIO/MPV
Bus m: Dev 2:Func 1	I2C/SPI
Bus m: Dev 2:Func 2	Serial Port
Bus m: Dev 2:Func 3	HDA
Bus m: Dev 3:Func 0	SATA 3.0
Bus m: Dev 4:Func 0	Ethernet10G
Bus m: Dev 5:Func 0	PCIe 3.0 x16(x8)
Bus m: Dev 6:Func 0	PCIe 3.0 x8
Bus m: Dev 7:Func 0	PCIe 3.0 x16(x8)
Bus m: Dev 8:Func 0	PCIe 3.0 x8
Bus m: Dev 9:Func 0	SPMC
Bus m: Dev 10:Func 0	КПИ-2 (IOHUB2) *****

_____

***** В случае наличия в системе микросхемы КПИ-2 её корневой мост будет находиться по адресу
“ Bus m: Dev 10:Func 0”

Программная модель встроенного контроллера периферийных интерфейсов EIOH представлена на рисунке 2.7.2.

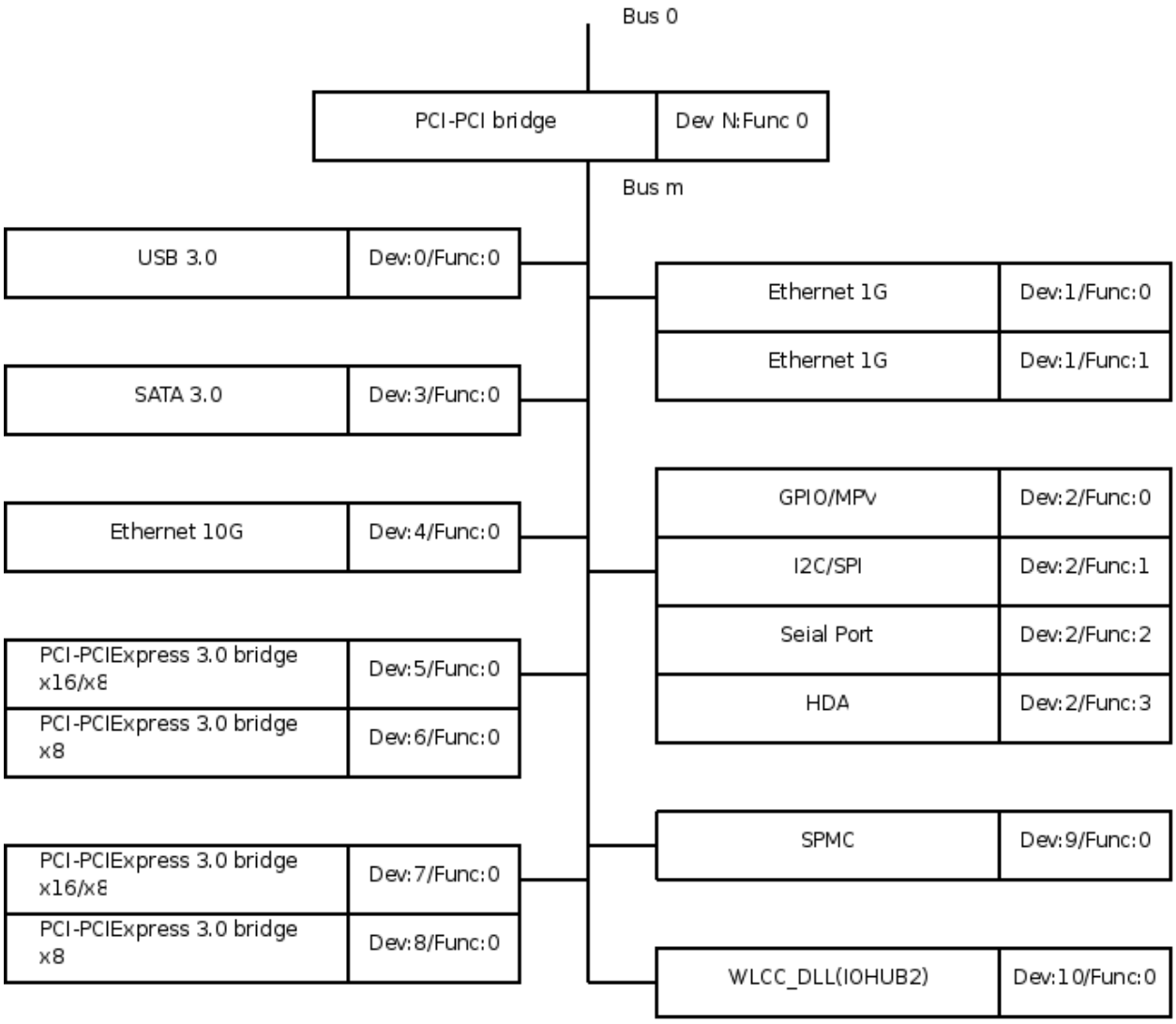


Рисунок 2.7.2 - Программная модель встроенного контроллера периферийных интерфейсов EIOH

2.7.2 Системный коммутатор

Все контроллеры в EIOH между собой связывает системный коммутатор. Он состоит из двух частей: System Commutator 1 и System Commutator 2.

2.7.2.1 Системный коммутатор первого уровня System Commutator 1

Системный коммутатор первого уровня System Commutator 1 объединяет контроллеры с Slink интерфейсом, системный коммутатор второго уровня System

Commutator 2 и контроллер канала ввода-вывода WLCC. System Commutator 1 подключен хост-контроллеру HC.

2.7.2.1.1 Межмодульный масштабируемый интерфейс Slink

При организации множества каналов обмена между однотипными модулями на одном кристалле, как правило, используется единый для всех модулей интерфейс, обладающий фиксированной максимальной пропускной способностью. Пропускная способность интерфейса выбирается такой, чтобы покрыть потребности самого требовательного из модулей. При этом для каналов, требования которых на пропускную способность ниже, используется тот же интерфейс и то же количество аппаратных ресурсов. В случае больших межмодульных расстояний и широкого диапазона требований на пропускную способность используемых каналов это может приводить к неоправданному избыточному использованию аппаратных ресурсов и усложнению трассировки проводников на кристалле, особенно в случае, когда потребности в пропускной способности самого требовательного из каналов велики. Одним из способов решения указанной проблемы является использование масштабируемого унифицированного интерфейса SLink (рисунок 2.7.3).

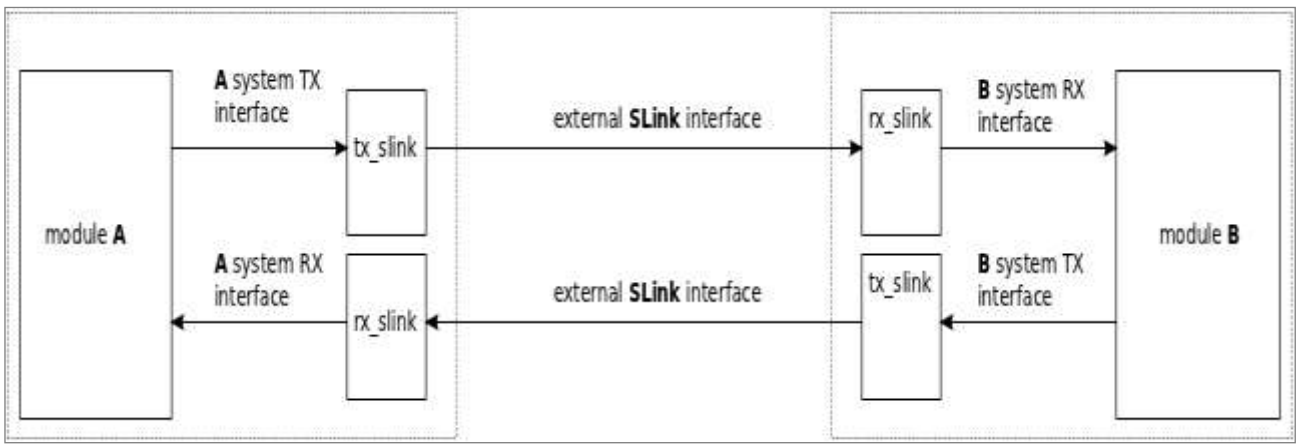


Рисунок 2.7.3 – Использование интерфейса SLink

Интерфейс SLink является коммутируемой (одни и те же линии используются для передачи как параметров (адрес, команда и т.п.) так и данных пакета) однонаправленной шиной. Ширина линии данных/параметров является задаваемым в соответствии с требуемой пропускной способностью параметром, а сам протокол обмена и набор управляющих сигналов является фиксированным. Параметр ширины линии данных/параметров задается в байтах и может быть задан с шагом в 1 байт в диапазоне от 1 байта и выше. Для организации двунаправленного обмена используются отдельные интерфейсы SLink для каждого из направлений. Таким образом, использование интерфейса SLink позволяет организовать межмодульные каналы обмена с пропускной способностью, сбалансированной с параметрами потоков обмена между указанными модулями в обоих направлениях. Для подключения контроллеров SLink к модулям используются унифицированные системные интерфейсы.

Для организации канала обмена в одном направлении между парой модулей пользователь делает оценку требуемой пропускной способности для соответствующего направления и выбирает в соответствии с указанной оценкой ширину линии данных/параметров внешнего интерфейса пары передатчик-приемник контроллера SLink. Кроме этого пользователь задает такие параметры системных интерфейсов контроллеров приёмника и передатчика как ширины линий заголовка и ширины линий данных и заносит их в файл конфигурации. После этого запускается wizard-скрипт с указанием сформированного файла конфигурации. Результатом работы скрипта являются два verilog-RTL файла: контроллер приёмника и контроллер передатчика. В случае необходимости аналогичная процедура выполняется для канала обратного направления обмена.

2.7.2.2 Системный коммутатор второго уровня System Commutator 2

Системный коммутатор второго уровня System Commutator 2 объединяет контроллеры с системным интерфейсом IOLink.

2.7.2.2.1 Межмодульный интерфейс IOLink

Интерфейс IOLink представляет собой коммутируемую (одни и те же линии используются для передачи как параметров (адрес, команда и т.п.) так и данных пакета) однонаправленную шину фиксированной ширины 4 байта. IO-link интерфейс предназначен для передачи запросов и ответов на запросы. Запросы и ответы передаются в виде пакетов, имеющих различный формат в зависимости от их параметров. Передаваемый пакет состоит либо только из заголовка длительностью от 2 до 4 тактов, либо из заголовка длительностью от 2 до 4 тактов и блока данных длительностью от 1 до 16 тактов. Для организации двунаправленного обмена используется 2 канала интерфейса.

2.7.3 Мультиконтроллер высокоскоростных линков PCIE/WLCC

Мультиконтроллер реализует линк шириной 16 линий, который может реализовывать различные интерфейсы, в зависимости, от настроек. Реализуемые интерфейсы — PCI-EXPRESS Gen 3 версии стандарта 3.1 и интерфейс собственной разработки WLCC. 16 линий могут работать как несколько линков, реализуя различные интерфейсы (в данном проекте PCI-EXPRESS и WLCC). Поддерживаемые режимы работы:

1) PCI-EXPRESS Gen3 x16;
2) PCI-EXPRESS Gen3 2x8;
3) PCI-EXPRESS Gen3 x8 + WLCC x8;
4) PCI-EXPRESS Gen3 2x4 + WLCCx8;
5) WLCC x16.

Режим работы задаётся внешними сигналами микропроцессора во время системного сброса.

Общая схема мультиконтроллера приведена на рисунке 2.7.4.

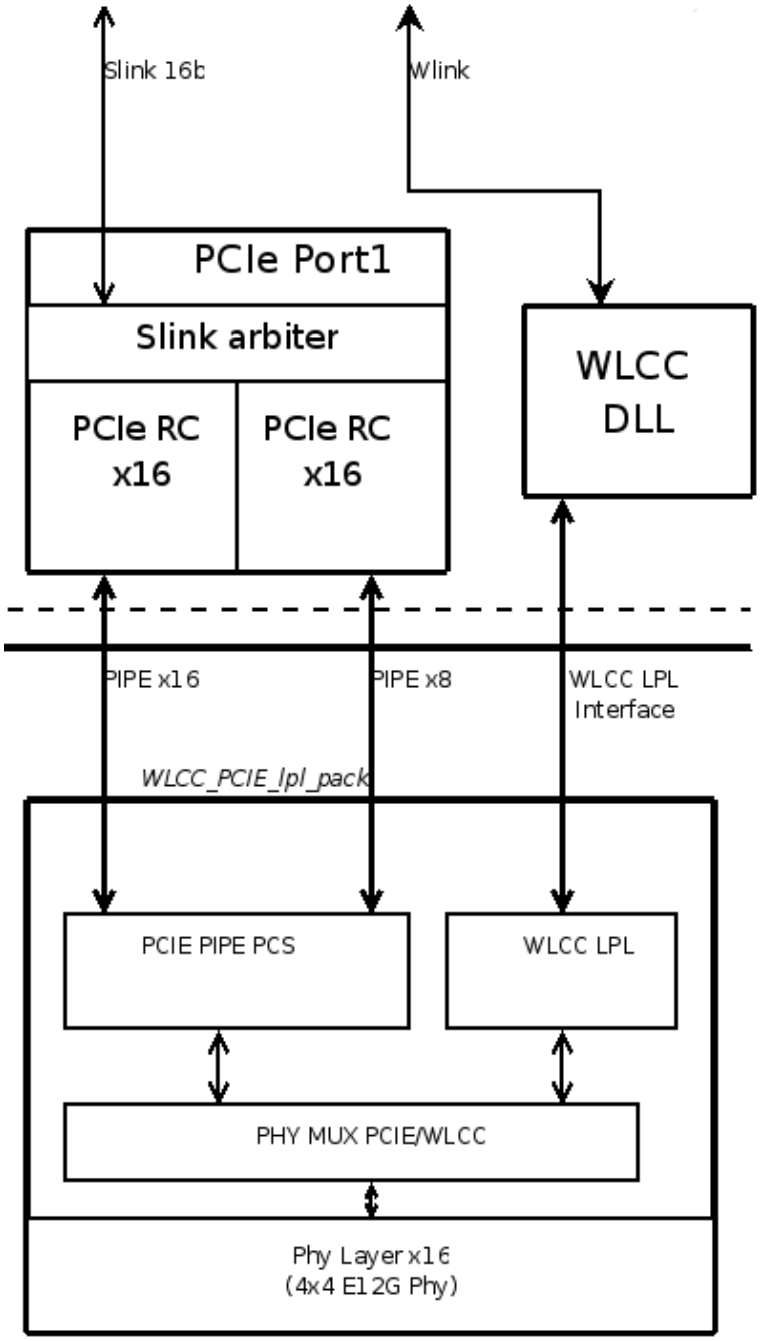


Рисунок 2.7.4 – Мультиконтроллер PCIE/WLCC

Upstream-интерфейсами контроллера являются Slink и Wlink. Slink — масштабируемый интерфейс, предназначенный передачи данных на большие расстояния внутри кристала. Wlink — интерфейс, предназначенный для реализации системного интерфейса. Наружу выходят 16 высокоскоростных линий способных передавать данные на частоте до 12 Гбит.

Для реализации стандарта PCI-EXPRESS Gen3 физический уровень настраивается на передачу данных со скоростью 8 Гбит/с и электрическими параметрами сигнала, соответствующие PCI-EXPRESS 3.1. В случае реализации интерфейса WLCC физический уровень настраивается в зависимости от подключаемого к интерфейсу устройства. К примеру, в случае подключения микросхемы КПИ-2 физический уровень должен быть настроен на передачу данных до 5 Гбит/с. Для будущих чипсетов возможно увеличение скорости передачи данных до 8 Гбит/с.

Мультиконтроллер включает в себя два контроллера интерфейса PCI-EXPRESS, каждый из которых виден в системе, как независимый PCI-to-PCI мост. Каждый контроллер PCIe RC x16 состоит из двух блоков, первый — покупной IP-блок фирмы Synopsys «DWC PCIe RC Controller», второй — блок сопряжения с интерфейсом slink, который разработан в АО «МЦСТ» и выходы которого подаются на блок Slink Arbiter, отвечающий за подключение двух контроллеров к одному slink - интерфейсу. Также используется покупной IP-блок «DWC Enterprise 12G PCIe 3.0 PCS» (PCIE PIPE PCS), который реализует управление физическим уровнем, а также расщепление физического уровня между контроллерами.

Для реализации WLCC — интерфейса используются:

WLCC_DLL — контроллер канального уровня (data link layer) интерфейса WLCC ,

WLCC_LPL — контроллер управления физическим уровнем (Logical Physical Layer) интерфейса WLCC.

Кроме того мультиконтроллер включает в себя мультиплексор интерфейсо, который реализует расщепление физического уровня между каналами PCI-EXPRESS и WLCC.

2.7.3.1 Контроллер PCI-EXPRESS

Краткие характеристики контроллера PCI-EXPRESS:

- интерфейс с системой – slink;

- поддержка двух режимов работы: x16 PCIe или 2x8 PCIe интерфейсов, в соответствии со спецификацией PCI Express 3.1;

- поддержка режимов x4, x8, x16 на интерфейсе PCIe;

- максимальная пропускная способность физического уровня PCIe Gen3 — 8 Гбит/с на линию (lane) в каждом направлении;

- два PIPE (максимум x16) интерфейса с элементами PHY;

- поддержка 32/64-битной адресации;

- поддержка механизма MSI (Message Signal Interrupt);

- максимальный размер передаваемых в пакете данных - 256 байт;

- поддержка до 64 незавершенных запросов;

- поддержка атомарных операций;

- поддержка TPH/Steering Tag.

Схема блока сопряжения Root Complex-a с интерфейсом Slink приведена на рисунке 2.7.5. Данный блок выполняет преобразование интерфейса RTRGT1 в Slink Tx и преобразование интерфейса Slink RX в XALI0/1 и DBI. Все вышеперечисленные интерфейсы являются пакетными. Блок реализует PCI-Express Ordering Rules и набор регистров для управления сбросом контроллера и переназначением прерываний. Краткое описание интерфейсов:

- RTRGT1 - по интерфейсу передаются пакеты запросов в направлении Upstream, а также ответы на запросы от процессора;

- XALI0 — передача пакетов непочтовых (Non-Posted) запросов от процессора;

- XALI1 — передача пакетов почтовых (Posted) запросов и пакетов ответов на DMA-обращения;

- DBI — интерфейс доступа к регистрам Root Complex-a;

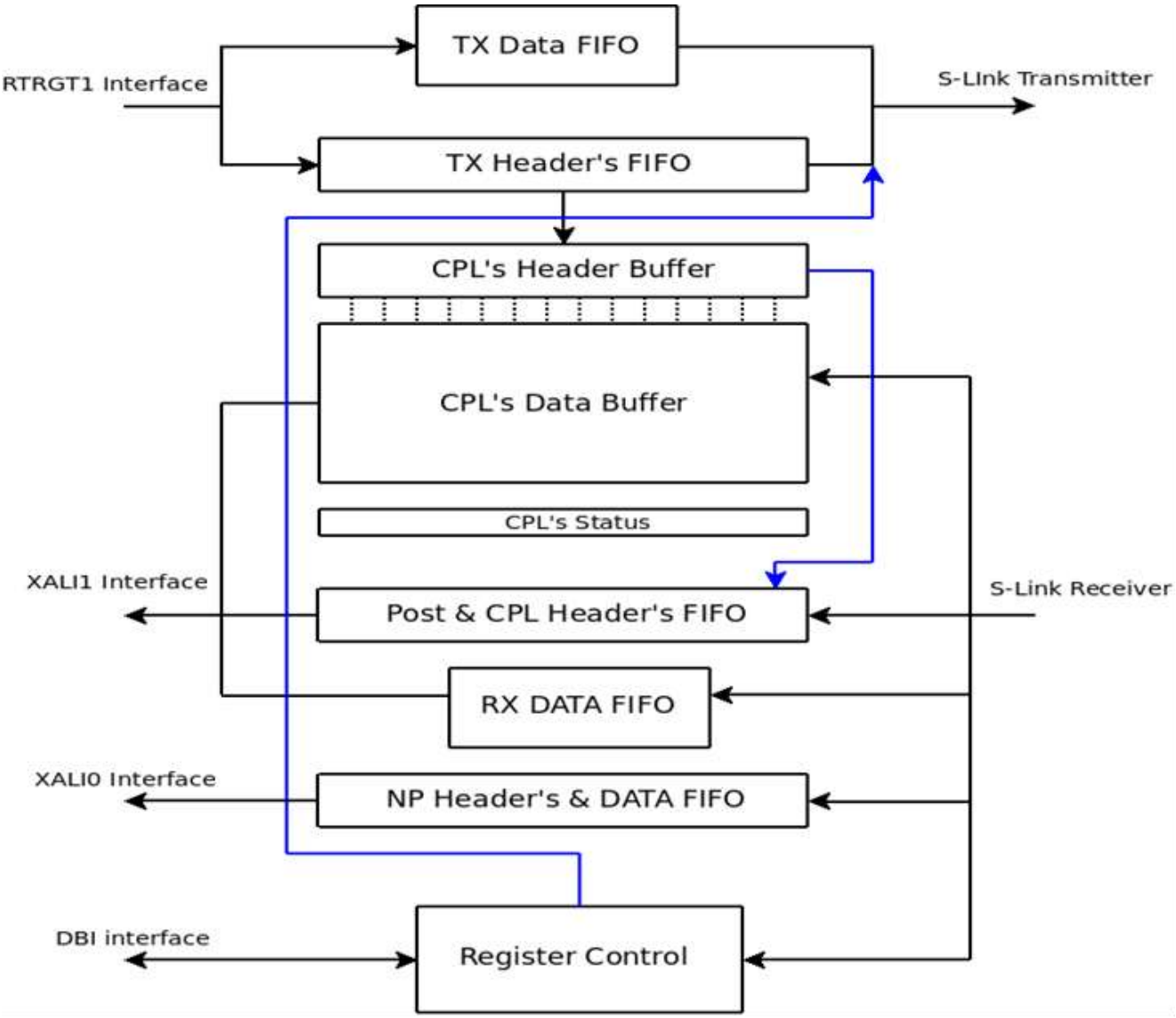


Рисунок 2.7.5 - Модуль DWC2Slink Bridge

Данный блок состоит из следующих модулей:

- Tx Header/Data FIFO — два буфера хранения и преобразования пакетов для направления Upstream, работающих по принципу FIFO. Причем в TX Header's Fifo хранятся заголовки, а в TX Data Fifo — данные пакетов. Также выполняют «нарезку» пакетов для процессора;

- CPL's Header/Data Buffer — два буфера хранения ответов от процессора. Также выполняет функцию сборки и упорядочивания ответов, что требует стандарт PCI-EXPRESS. В буфер CPL's Header записываются части заголовков запросов в upstream направлении;

- Post & CPL Header's FIFO и RX Data FIFO — два буфера хранения и преобразования пакетов для направления Downstream, работающих по принципу FIFO. Выходным интерфейсом данного буфера является интерфейс XALI1;

- NP Header's & DATA FIFO — буфер хранения и преобразования не почтовых запросов от процессора. Работает по принципу FIFO. Выходным интерфейсом данного буфера является интерфейс XALI0;

- Register Control — блок управления операционными регистрами. Реализует доступ к регистрам Root Complex-а через DBI интерфейс, а также занимается перехватом обращений, для получения контрольной информации.

2.7.3.2 Контроллер WLCC

Мультиконтроллер содержит контроллер WLCC для подключения контроллеров физического уровня WLCC_LPL и канального уровня WLCC_DLL.

Краткие характеристики:

- внешний линк - динамически масштабируемая (x4, x8, x12, x16) последовательная шина (full duplex - 8 Гбит/с + 8 Гбит/с per lane);

- интерфейс между WLCC_LPL и WLCC_DLL, тракт передачи - 128 бит, до 800 МГц, до 12.8 Гбайт/с (запас на будущее для работы с КПИ-2 500 МГц и 8 Гбайт/с);

- интерфейс между WLCC_LPL и WLCC_DLL, тракт приема - 128 бит, до 800 МГц, до 12,8 Гбайт/с.

Контроллер управления физическим уровнем WLCC_LPL взаимодействует с одной стороны с блоком физуровней (Synopsys E12G PHY), другой стороны с контроллером канального уровня WLCC_DLL.

WLCC_LPL осуществляет:

- начальную инициализацию блока физуровней (Synopsys E12G PHY);

- исполнение команд от системы на:

  - инициализацию линка;

  - переинициализацию линка;

- динамическое изменение ширины линка без остановки передачи данных;

- восстановление линка при нарушении его функционирования;

- перевод линка в одно из состояний: ”Sleep”, ”Disable”, ”PowerOff”;

- исполнение команд от удаленного абонента на:

- инициализацию линка;

- переинициализацию линка;

- динамическое изменение ширины линка без остановки передачи данных;

- восстановление линка при нарушении его функционирования;

- перевод линка в состояние ”Sleep”;

- передачу в контроллер WLCC_DLL информации о текущем состоянии линка;

- передачу в контроллер WLCC_DLL информации о результате выполнения команды от системы;

- прием потока данных из WLCC_DLL и его кодирование для выдачи на внешний линк посредством модуля Synopsys E12G PHY;

- прием потока данных с внешнего линка посредством модуля Synopsys E12G PHY и его декодирование для передачи в WLCC_DLL

Также контроллер включает в себя блок инициализации линка, который позволяет задать или поменять настройки для работы контроллера.

Для настройки параметров физического уровня используются 3 внешних вывода, задающие 8 рабочих точек. Подъем линка по включению питания осуществляется на наборе настроек задающихся внешними выводами. Также присутствует набор настроек размещенных в регистрах, которые расположены в модуле LPL контроллера WLCC. Данные регистры доступны через CR интерфейс и по системному сбросу устанавливаются в значения, соответствующие настройкам физического уровня, определяемым входными выводами. Данные регистры не сбрасывается «soft» reset-ом. ПО может изменить значения данного набора и выполнить процедуру soft reset-а, по окончанию которой линк будет поднят на

заданных настройках. Для оценки качества линка, добавлена группа регистров Result — в них содержится информация о качестве сигнала на приёмнике в линке, которую можно использовать для итерационного поиска оптимальных настроек при отладке.

Контроллер канального уровня WLCC_DLL с некоторыми функциями уровня транзакций в составе высокоскоростного канала ввода/вывода. Предназначен для обеспечения связи северного моста с контроллером периферийных интерфейсов КПИ-2 и выполняет следующие основные функции:

- обеспечение обмена с системным уровнем и с контроллером физического уровня в соответствии с протоколом;
- контроль целостности передаваемых данных;
- исправление ошибок, возникших при передаче;
- контроль состояния буферов.

Контроллер взаимодействует с системным уровнем посредством квантов информации, далее называемых транзакциями, обеспечивая сервис по надежному высокоскоростному обмену с системным уровнем внешнего абонента. Транзакции являются составляющими элементами системных операций, таких как чтение и запись, и передаются пакетами. Все операции включают отправку транзакции запроса (Request). Операции делятся на почтовые, то есть, не предполагающие транзакцию ответа (Completion), и на непочтовые, завершающиеся получением транзакции ответа (Completion).

Взаимодействие с контроллером физического уровня осуществляется посредством пакетов TLP. Передача пакета транзакции (TLP) на линке состоит из следующих фаз:

- Header - фаза заголовка;
- Payload - фаза данных транзакции (если есть);
- Tail - фаза завершения (или хвост) транзакции.

Для обеспечения контроля целостности данных при передаче по линку используется механизм нумерации TLP в последовательности и CRC-кодирование.

Протокол обмена по линку использует механизм группового подтверждения принятия TLPs. В случае сбоя на линке инициируется процесс повтора последовательности TLPs, начиная с соответствующего номера. Повтор не может возникнуть по причине нехватки ресурсов приемника: это гарантируется использованием кредитного механизма управления потоком Flow Control Mechanism. Для подтверждения приема, а также для передачи информации об освобождении ресурсов приемника, используются служебные пакеты DLLP.

Прием/передача пакетов транзакций по линку осуществляется в автоматическом режиме: то есть, для организации обмена не требуются обращения к внутренним регистрам. Доступ к внутренним регистрам контроллера используется только для задания режима работы, обработки прерываний и контроля состояния линка.

Контроллер поддерживает использование двух независимых виртуальных каналов. Под независимыми виртуальными каналами VC понимается способ организации однонаправленных потоков запросов и ответов между устройствами, находящимися в системных уровнях, исключающий взаимные конфликты по ресурсам системного уровня по приему (приемного буфера). Каждому виртуальному каналу на приемной стороне должен соответствовать выделенный для него ресурс буфера и кредитный механизм контроля его состояния (FC-механизм). Запросы, передающиеся внутри виртуального канала, сохраняют свой порядок, то есть, исключены обгоны одних запросов другими.

Ответы, передающиеся внутри виртуального канала имеют право обгонять другие ответы, а также все непочтовые запросы, и почтовые запросы от других источников. Ответ не может обгонять почтовый запрос от того же источника. Ответы на непочтовые запросы для всех VC имеют бесконечный ресурс, что исключает конфликты по ресурсам приемной стороны и, следовательно, не учитываются FC-механизмами.

Структурная схема контроллера WLCC_DLL представлена на рисунке 2.7.6.

Контроллер WLCC_DLL содержит следующие блоки:

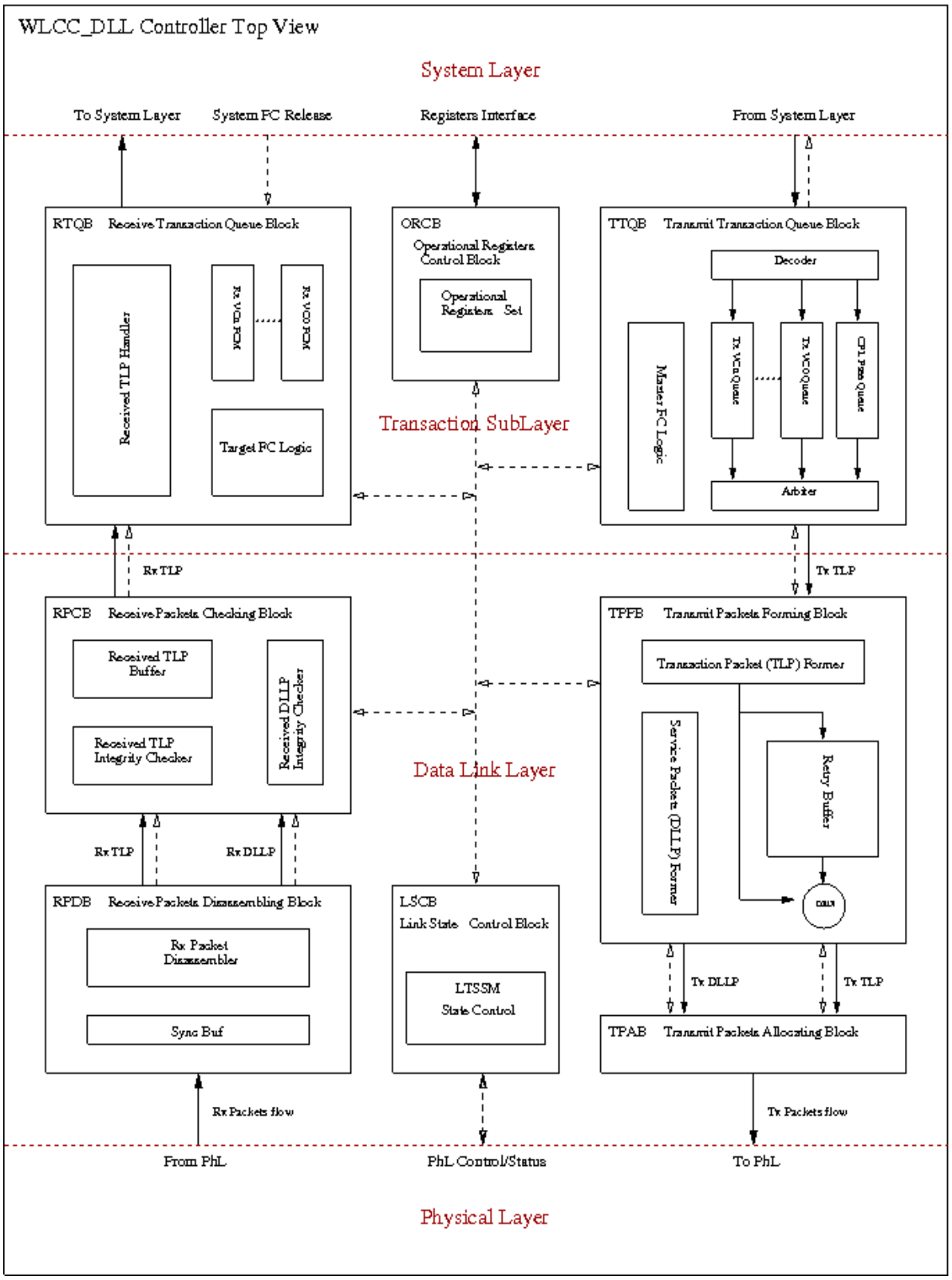


Рисунок 2.7.6 - Контроллер канального уровня WLCC_DLL

- TTQB (Transmit Transaction Queues Block) - Прием и распределение транзакций от системного уровня по VCs. Абитрация между VC. Обеспечение работы протокола управления FC-механизмами VCs по передаче (master part VC FC Management);

- TPFB (Transmit Packet Forming Block) - Формирование пакетов транзакций (TLP) и служебных --(DLLP) для передачи. Организация повторов последовательности TLPs;

- TPAB (Transmit Packet Allocating Block) - Подготовка потока передаваемых пакетов для оптимального использования пропускной способности линка. Выделение места под символы пакетной синхронизации;

- RPDB(Receive Packet Disassembling Block) - Прием и пересинхронизации потока принимаемых пакетов. Очистка от символов пакетной синхронизации. Разделение потока пакетов на TLP и DLLP потоки. Выделение в пакетах полей CRC;

- RPCB(Receive Packet Checking Block) - Разуплотнение потока TLPs, проверка целостности и генерация запросов на отправку подтверждений. Разуплотнение потока DLLPs, проверка целостности, декодирование и передача служебной информации в TPFB, TTQB, RTQB;

- RTQB (Receive Transaction Queue Block) - Анализ принятых транзакций и передача их в системный уровень. Обеспечение работы протокола управления FC-механизмами VCs по приему (target part VC FC Management);

- LSCB (Link State Control Block) - Управление состоянием физического уровня;

- ORCB (Operational Registers Control Block) - Обеспечение возможности управления режимами и контроля параметров работы линка посредством доступа к операционным регистрам контроллера.

Ниже приведены основные характеристики контроллера WLCC_DLL:

- полнодуплексный канал обмена. Системная частота работы WLCC_DLL — от 250 до 800 МГц. Ширина интерфейса по данным между уровнями - 16 байт;

- режим автоматического определения деградации линка и запуск процедуры подстройки параметров линка;

- набор операционных регистров для управления режимами работы контроллера. Отдельный интерфейс для доступа к ним;

- поддержка 2 независимых виртуальных каналов. Настраиваемая весовая схема арбитража виртуальных каналов при передаче;

- поддержка режимов 32- и 56-разрядной адресации при запросах в пространство памяти и операций скрытого канала;

- кредитный механизм контроля состояния буферов для каждого виртуального канала;

- учет кредитных ресурсов виртуального канала отдельно по заголовкам и по данным позволяет более рационально использовать буферы системного уровня;

- пакетная передача. Пакеты транзакций выровнены по границе 32 бит. Максимальный размер пакета - 80 байт (64 байт - данные), минимальный - 8 байт;

- низкие 'накладные расходы' при передаче пакета. Существенно меньше, чем в пакетах PCIe;

- сервисные пакеты для передачи служебной информации размером 8 байт;

- механизмы контроля целостности пакетов транзакций: 16-разрядный CRC-код, нумерация пакетов в последовательности;

- информирование системы прерыванием в случае возникновения сбоев или нештатных ситуаций, требующих реакции системы;

- преодоление сбоев на линке путем повтора последовательности пакетов;

- буфер повторов размером ~2 Кбайт.

2.7.3.3 Дополнительные контроллеры PCI-EXPRESS Gen3 в EIOH

В EIOH имеется второй блок контроллеров PCI-EXPRESS Gen 3, аналогичный описанному ранее. При этом блок физуровней делится (мультиплексируется) между этими контроллерами и контроллером межпроцессорного канала, точно также как это реализовано с контроллером WLCC (рисунок 2.7.7).

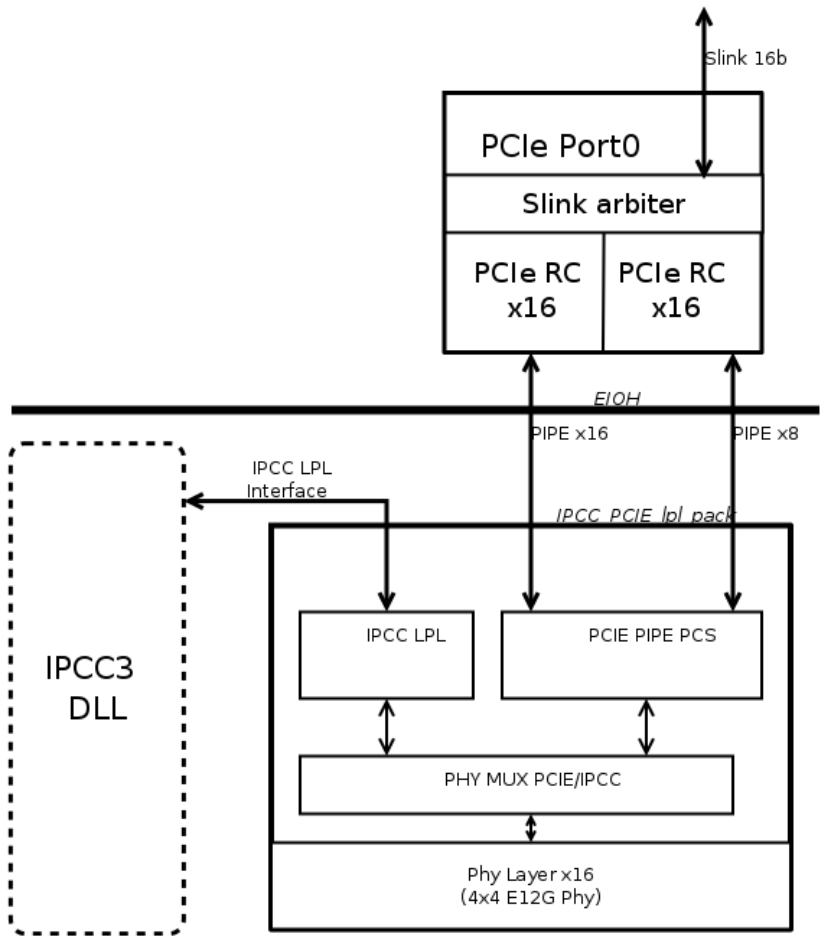


Рисунок 2.7.7– Порт PCIE/IPCC

Поддерживаемые режимы работы:

1) PCI-EXPRESS Gen3 1x16;

2) PCI-EXPRESS Gen3 2x8;

3) PCI-EXPRESS Gen3 1x8 + IPCC3 x8;

4) PCI-EXPRESS Gen3 2x4 + IPCC3x8;

5) IPCC3 x16.

Режим работы задаётся внешними сигналами микропроцессора во время системного сброса.

Для этого используются два контроллера стандарта PCI-EXPRESS Gen 3, точно такие же, как были описаны выше, такой же блок PCI-EXPRESS Gen 3 PCS и такой же набор блоков PHY.

2.7.4 Контроллер USB 3.0

Контроллер универсальной последовательной USB шины содержит 4 порта USB2.0 и 4 порта USB3.0.

Контроллер использует универсальный интерфейс XHCI (eXtensible Host Controller Interface), который поддерживает USB 1.x, USB 2.0, и USB 3.0 устройства.

Структурная схема хост-контроллера USB представлена на рисунке 2.7.8

Характеристики контроллера:

- поддержка до 63 USB устройств;
- поддержка 32/64-битной адресации;
- скорость каждого USB 2.0 порта – 480 Мбит/с (полудуплекс);
- скорость каждого USB 3.0 порта – 5 Гбит/с (в обоих направлениях);
- интерфейс с системой – Slink.

Контроллер способен взаимодействовать с оперативной памятью в режиме прямого доступа к памяти DMA.

Доступ к XHCI регистрам и внутренним регистрам хост-контроллера USB осуществляется через интерфейс AXI (Advanced eXtensible Interface). Регистры адресуются в пространстве памяти.

Для хранения пакетов необходимо подключить 3 памяти с произвольным доступом (RAM):

RAM_0 – кэш дескрипторов;

RAM_1 – пакеты на передачу;

RAM_2 – принятые пакеты.

Имеется возможность включить эти памяти в режим с коррекцией ошибок (ECC), исправляющий изменения одного бита в одном машинном слове и вызывающий прерывание при ошибках в более чем одном бите.

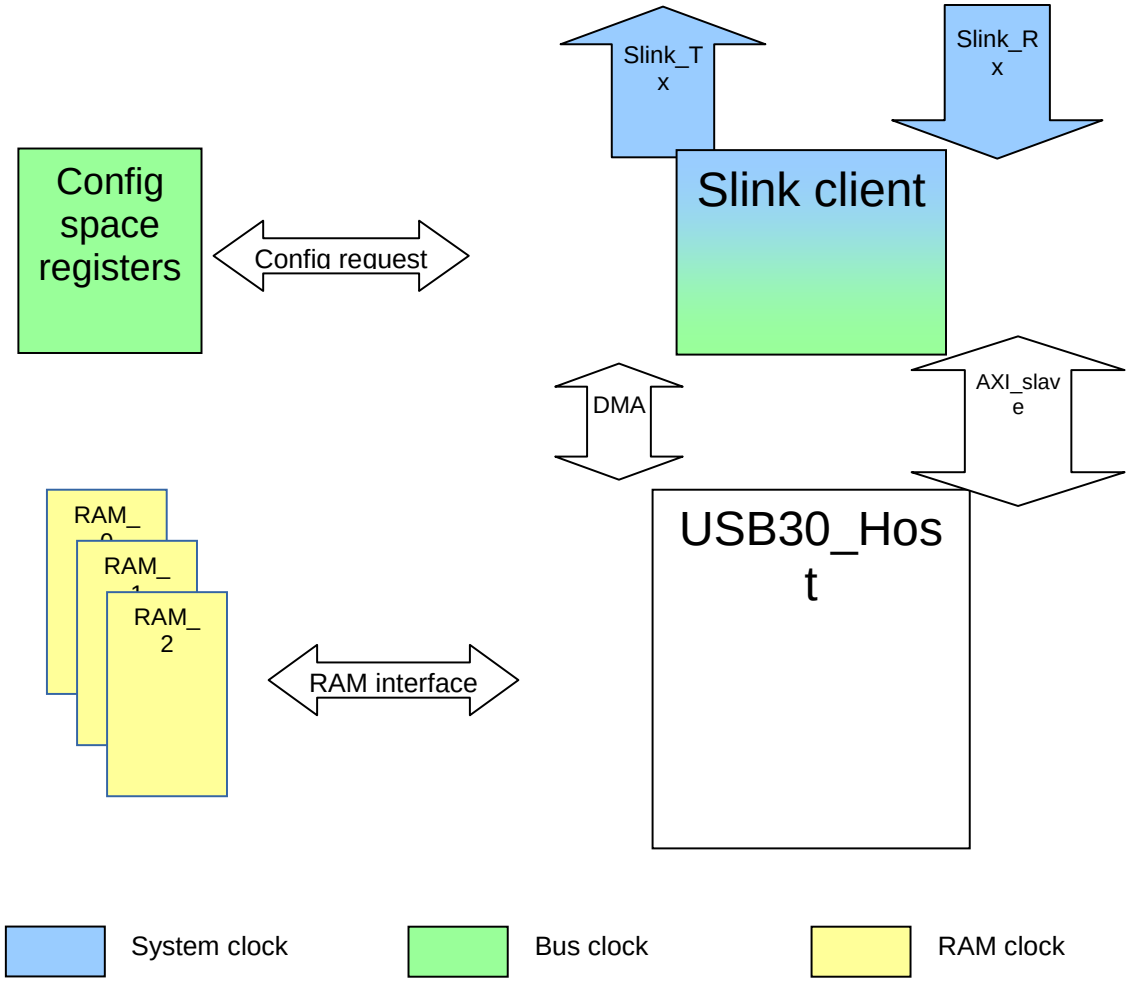


Рисунок 2.7.8 - Хост-контроллер USB

Связь с системным коммутатором происходит через интерфейс Slink.

Для преобразования DMA запросов в транзакции интерфейса Slink реализован Slink клиент, который осуществляет разбиение на транзакции DMA запроса и пересинхронизацию на системную частоту. Также клиент возвращает ответы на DMA запросы. Внеочередные ответы на транзакции хранятся в отдельной памяти.

Реализовано конфигурационное пространство PCI. Клиент осуществляет доступ к регистрам конфигурационного пространства через принятые запросы configuration requests type 0. Если запрос в пространство памяти, принятый из

системного коммутатора, попадает в регистр BAR0 конфигурационного пространства, происходит доступ к внутренним регистрам хост-контроллера USB.

2.7.5 Порт интерфейсов Sata и Ethernet

Порт интерфейсов Sata и Ethernet (рисунок 2.7.10) состоит из контроллера интерфейса SATA 3.0 на 4 порта, двух двухпортовых блоков подключения интерфейса SATA к мультистандартному физическому уровню PHY 12G x2, двух контроллеров интерфейса 1 Gb Ethernet, одного контроллера 10 Gb Ethernet, двух блоков подключения контроллеров интерфейса Ethernet к мультистандартному физическому уровню PHY 12G x2 (один блок подключает контроллер 1 Gb Ethernet к PHY, второй подключает к PHY контроллеры 1 Gb Ethernet и 10 Gb Ethernet, одновременно являясь мультиплексором этих интерфейсов), мультиплексора интерфейсов Sata и Ethernet (позволяет использовать одни и те же физические линии в зависимости от конфигурации либо для SATA, либо для Ethernet) и двух блоков физического уровня PHY 12G x2.

Поддерживаемые конфигурации:

- 4 SATA;
- 2 SATA + 2 ETHERNET 1G;
- 2 SATA + 1 ETHERNET 10G + 1 ETHERNET 1G;

Количество SATA портов задается внешним выводом во время сброса системы.

Конфигурация портов ETHERNET в том числе способность работать в режиме 2.5G для контроллера 1G ETHERNET настраивается в программном режиме во время конфигурации оборудования.

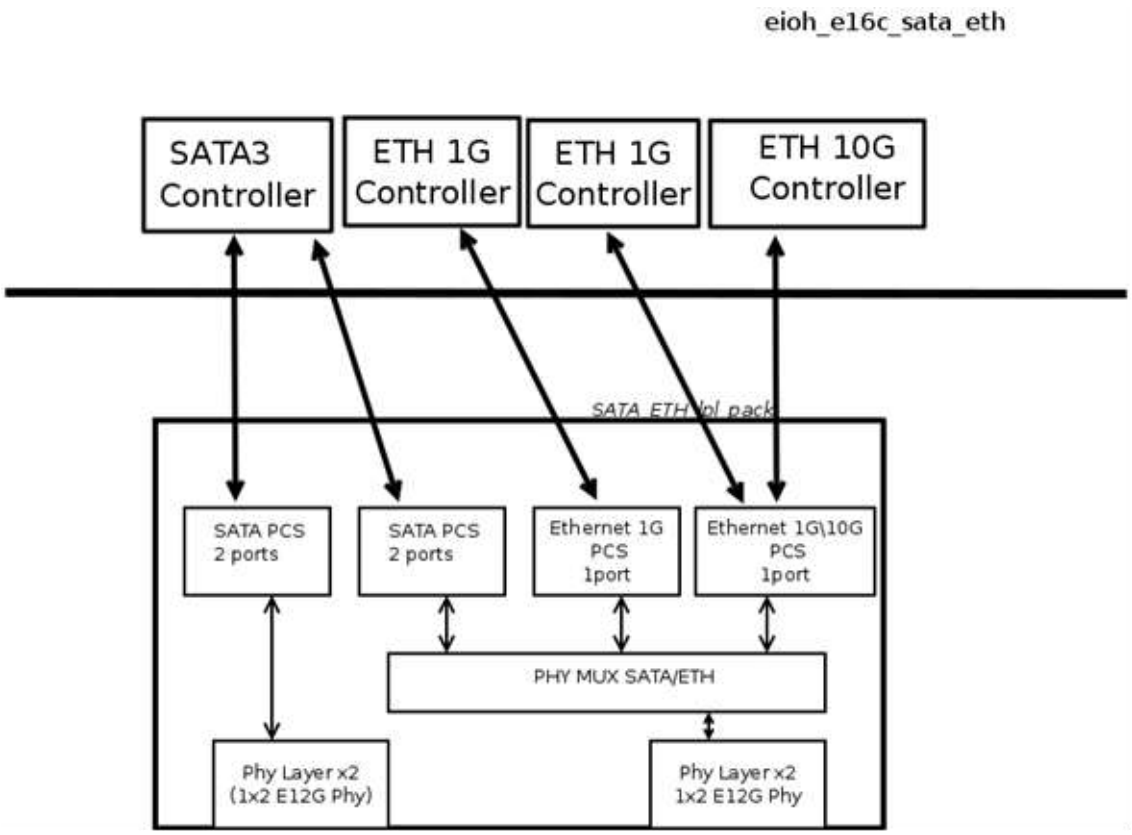


Рисунок 2.7.10 - Порт интерфейсов Sata и Ethernet

2.7.5.1 Контроллер SATA 3.0

Контроллер состоит из логической части и покупных IP-блоков физического уровня фирмы Synopsys.

Структурная схема логического уровня SATA контроллера представлена на рисунке 2.7.11.

Контроллер поддерживает 4 порта, каждый из которых способен работать в режимах SATA Gen3/2/1. Логический уровень контроллера реализует программную модель AHCI, а так же имеет возможность работать в режиме «legacy» (наследственный программный интерфейс IDE контроллера). В режиме AHCI контроллер поддерживает работу с 32 командными слотами для каждого порта, а так же поддерживает NCQ (zero buffer offset и non-zero buffer offset guaranted-in-order). Системным интерфейсом SATA контроллера является межмодульный масштабируемый интерфейс «Slink» шириной 4 байта. Системный интерфейс

реализуется с помощью Slink Client'a, через который осуществляется доступ к регистрам конфигурационного пространства, регистрам AHCI и legacy регистрам, а так же, с помощью внутреннего арбитра, клиент передает запросы чтения и записи системной памяти от DMA контроллеров портов.

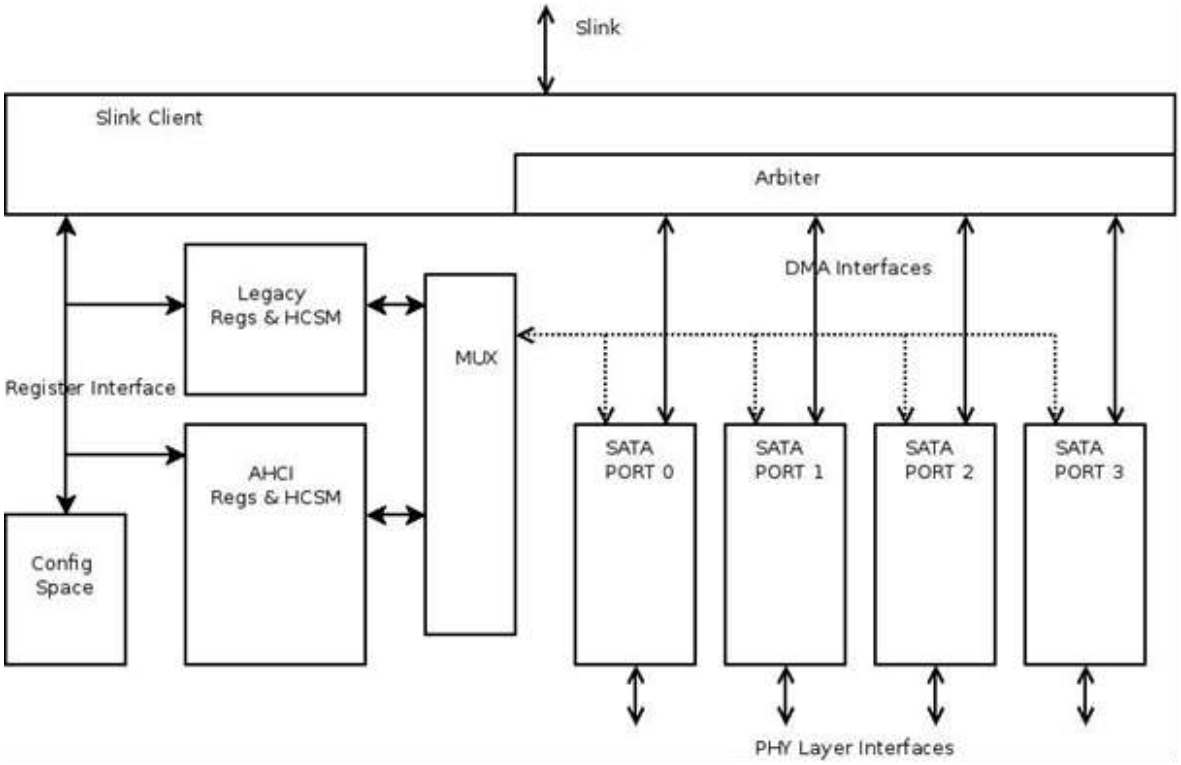


Рисунок 2.7.11 – Логический уровень SATA контроллера

Интерфейсы для связи с физическим уровнем для SATA контроллера разделены на две группы по два порта. Обе группы для связи с физическими уровнями используют модуль настройки и управления физическим уровнем (SATA PCS). Но при этом первая пара портов всегда подключена к физическому уровню. А вторая пара - через мультиплексор интерфейсов SATA и Ethernet, который может отключать эти порты SATA контроллера от физического уровня.

Краткие технические характеристики SATA 3.0 контроллера:

- 4 порта;

- программируемые регистры базовых адресов;

- буферы размером 1024 байт для передачи данных, для каждого порта, в каждом направлении;

- встроенный DMA-контроллер, для каждого порта;

- системный интерфейс Slink;

- интерфейс для связи с SATA phy уровнем первого, второго и третьего поколения SATA;

- поддержка режима AHCI;

- поддержка NCQ (zero buffer offset и non-zero buffer offset guaranted-in-order).

Контроллер соответствует спецификациям:

- Serial ATA specification, Revision 3.1;

- Programming Interface for Bus Master IDE Controller, Revision 1.0;

- Serial ATA Advanced Host Controller Interface (AHCI), Revision 1.3.

2.7.5.2 Контроллер Ethernet 1Gb

Контроллер EIOH содержит 2 сетевых контроллера Gigabit Ethernet**.** Структурная схема контроллера ETHERNET 1Gb представлена на рисунке 2.7.12.

Контроллер имеет интерфейс IOLINK с коммутатором, и интерфейс GMII с блоком ETHERNET_1G_PCS (покупной блок от фирмы Synopsys), который используется для связи с блоком физического уровня E12G также от фирмы Synopsys.

Краткие технические характеристики:

- поддерживаемые режимы:

1G - поддерживает передачу данных на скоростях 10/100/1000 Mbit в полнодуплексном и полудуплексном режимах;

2.5G - поддерживает передачу данных на скоростях 25/250/2500 Mbit в полнодуплексном и полудуплексном режимах;

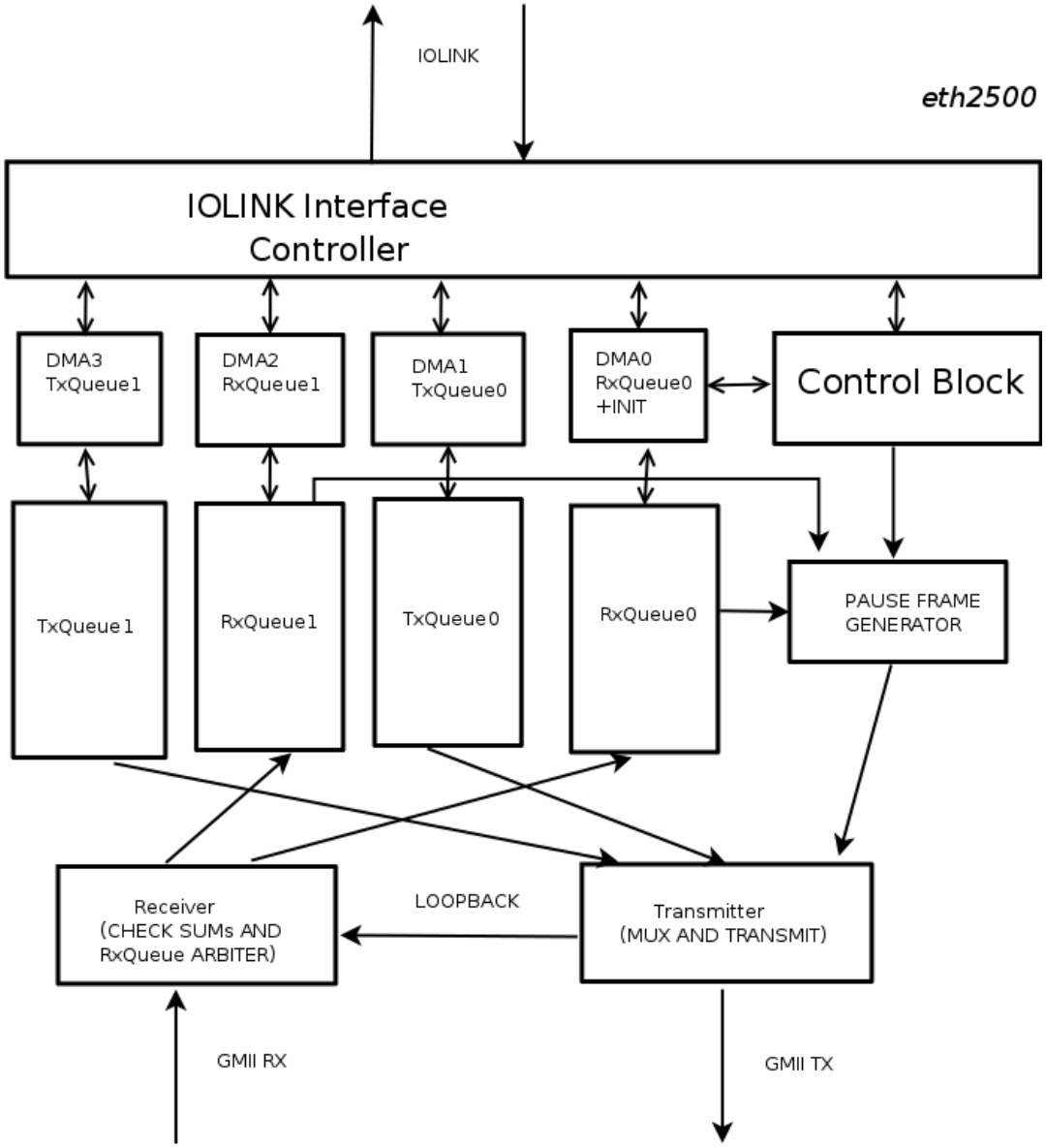


Рисунок 2.7.12 - Контроллер Ethernet 1Gb (ETH2500)

- количество очередей: 2 на передачу, 2 на приём — в каждой очереди может быть до 512 дескрипторов, у каждой очереди есть независимый dma — канал (канал прямого доступа в память);

- системный интерфейс Iolink (частота до 500 Мгц);

- внешний интерфейс (контроллера) - GMII (частота 125 Мгц для режима 1 Gbit и 312,5 МГц для режима 2.5 Gbit);

- поддержка управления потоком (pause frame);

- буферы для принимаемых и передаваемых пакетов в каждой очереди по 8 Кбайт;

- поддержка ieee-1588;

- автоматический подсчет контрольных сумм IPv4 пакетов (IPv4 header, TCP, UDP);

- автоматический подсчет CRC(FCS) для Ethernet фреймов;

- нтерфейс mdio для доступа к регистрам PCS блока и внешнего PHY;

- внешний интерфейс (микросхемы) - SGMII (1Gbit/2.5Gbit).

Контроллер не знает, в каком режиме работает - 1G или 2.5G, отличие состоит в частоте синхронизации 125 или 312,5 МГц. Разряд, управляющий этой функциональностью, находится в блоке ETHERNET_PCS_1G (или ETHERNET_PCS_1G/10G для случая, когда 10G и 1G контроллеры делят одну физическую линию). Этот блок подключен к контроллеру через mdio интерфейс. Также с помощью регистров этого блока программируется блок физического уровня E12G как для работы в режиме 1G , так и 2.5G.

2.7.5.3 Контроллер Ethernet 10Gb

2.7.5.3.1 Краткое описание внутреннего устройства контроллера

Контроллер Ethernet 10G состоит из следующих основных блоков (рисунок 2.7.13):

- контроллер S-Link – обеспечивает связь устройства с системной шиной S-Link, предоставляя следующие сервисы: PIO-доступ CPU в регистры устройства и DMA-доступа устройства в системную (процессорную) память;

- контроллер MSI-X – обеспечивает выдачу и обработку процессорных прерываний MSI-X, в т.ч. разделение прерываний от каждой очереди приёма и передачи на отдельный вектор MSI-X, что позволяет жёстко прикреплять очереди к процессорным ядрам;

- менеджер очередей – обеспечивает своевременную подкачку и отписывание дескрипторов, а также распределяет пропускную способность канала передачи между различными очередями;

- тракт передачи (PKT_GET+MAC_TX) – осуществляет по полученным дескрипторам чтение областей памяти с подготовленными к передаче пакетами,

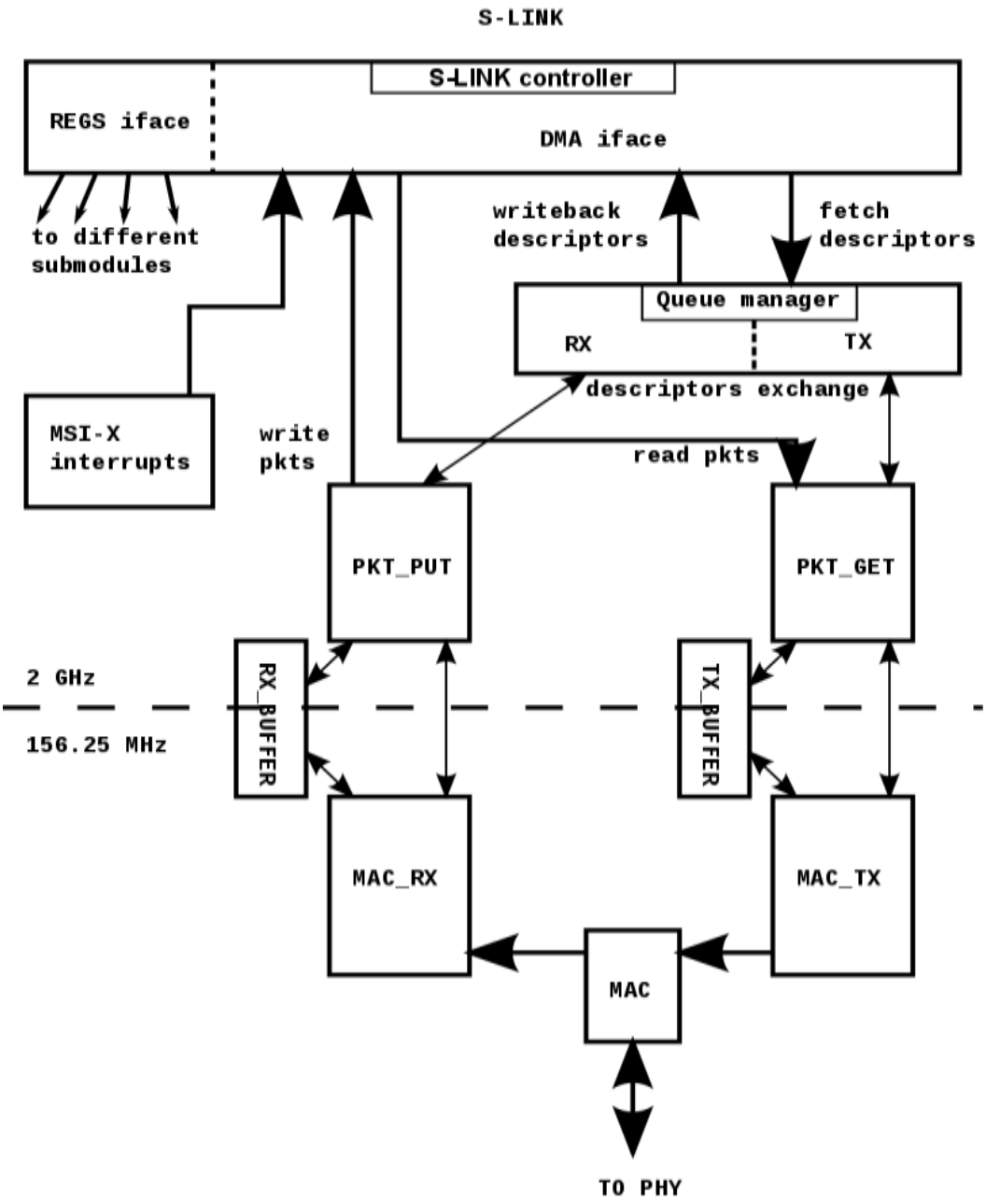


Рисунок 2.7.13 - Контроллер Ethernet 10 Gb

формирование всех необходимых заголовков, контрольных сумм и т.д., а также распределение пропускной способности канала между пакетами с различными приоритетами;

- тракт приёма (PKT_PUT+MAC_RX) – осуществляет приём входящих пакетов, обработку их по типам (broadcast/multicast/unicast, VLAN и т.д.), настраиваемое распределение между различными очередями приёма, проверку контрольных сумм, заполнение дескрипторов и запись пакетов в память;

- контроллер MAC – обеспечивает поддержку низкоуровневых протоколов готовности канала (local fault/remote fault/link interruption), реализует протоколы pause frames и priority pause frames, при необходимости ограничивает исходящий трафик заданными величинами packets-per-second или bytes-per-second.

2.7.5.3.1.1 Блок MAC

Обеспечивает функции:

- обработка ошибочных событий на интерфейсе XGMII/XAUI (IEEE 802.3-2012,46.3.4);

- пересинхронизация принимаемых данных на частоту передачи;

- внутренний LOOPBACK (без участия интерфейса XGMII/XAUI);

- проверка валидности входящих фреймов по критериям минимальной длины и верного FCS;

- формирование исходящих фреймов: расчёт и подстановка FCS, добавление preamble/postamble;

- обработка pause frames: поддержка priority pause frames, генерация исходящих pause frames в зависимости от заполненности буферов в MAC_RX, управление передачей в соответствии с входящими pause frames;

- ограничение исходящего потока данных по критериям max BPS (bits per second), max PPS (packets per second), min IPG (inter-packet gap).

2.7.5.3.1.2 Контроллер MSI-X

Контроллер MSI-X реализован в соответствии со стандартом PCI MSI-X.

Примечание - Контроллер не генерирует legacy и MSI прерывания.

2.7.5.3.1.3 Передающий тракт PKT_GET<->MAC_TX

Обеспечивает функции:

- обработка и декодирование поступающих из менеджера очередей дескрипторов на передачу;
- приоритеты (до 8 приоритетов на каждую очередь), определяющие степень использования пропускной способности памяти и исходящего ethernet-канала;
- разделение ресурса буфера передачи между пакетами с различными приоритетами;
- расчёт и подстановка в исходящие пакеты контрольных сумм (IPv4 header, TCP/UDP body);
- поддержка VLAN;
- максимальный размер фрейма – 16384 байт.

2.7.5.3.1.4 Принимающий тракт MAC_RX<->PKT_PUT

Обеспечивает функции:

- начальная фильтрация пакетов, такая как фильтрация по DSTMAC, VLAN, броадкастам, мультикастам;
- распределение приходящих пакетов по очередям дескрипторов на основе различных признаков (хешей различных частей пакета);
- подсчёт и проверка контрольных сумм: заголовки IPv4, пакеты TCP и UDP;
- поддержка VLAN;
- поддержка split descriptors (один пакет записывается в несколько буферов, описанных несколькими дескрипторами);
- максимальный размер фрейма – 16384 байт.

2.7.5.3.1.5 Блок обработки очередей дескрипторов Queue manager

Каждый принятый или переданный пакет описывается дескриптором, в котором содержится такая информация как положение пакета в памяти, тип пакета, наличие-отсутствие ошибок в нём и т.д.

При передаче или приёме дескрипторы группируются в т.н. очереди, группы подряд лежащих в памяти дескрипторов, работа с которыми происходит методом FIFO: процессор подготавливает дескрипторы в системной и сообщает устройству об их готовности, устройство их считывает и после обработки возвращает в системную память, сообщая процессору об их возврате.

Для дескрипторов передачи процессор подготавливает дескрипторы, указывающие на готовые к передачи пакеты; возврат означает что пакет, описываемый данным дескриптором, был перемещён из системной памяти в буфер передачи и будет отправлен в канал передачи.

Для дескрипторов приёма процессор подготавливает дескрипторы, описывающие буферы в памяти, пригодные для приёма пакетов. Возвращённый дескриптор обозначает, что принятый пакет (на который указывает этот дескриптор) был записан в память в указанное в дескрипторе место, кроме того в дескрипторе отражаются некоторые характеристики принятого пакета.

Существуют дескрипторы 2 типов - "короткие" (размером 16 байт) и "длинные" (размером 32 байта). Каждая очередь индивидуально настраивается на использование только коротких или только длинных дескрипторов.

Длинные и короткие дескрипторы имеют совпадающий формат в пределах первых 16 байт, таким образом, длинный дескриптор лишь расширяет функционал короткого. В длинной части дескриптора находятся следующие данные: теги inner VLAN и outer VLAN (для 802.1q), отметки времени (для IEEE 1588). Следовательно, длинные дескрипторы имеет смысл использовать лишь в случае потребности в соответствующем функционале.

Блок обработки очередей дескрипторов обеспечивает:

- поддержание очередей дескрипторов в виде колец дескрипторов, расположенных в системной памяти;

- каждая очередь содержит указатели головы (’head’) и хвоста (’tail’) внутри этого кольца. Указатель головы указывает на последний подготовленный процессором дескриптор, указатель хвоста – на последний обработанный (возвращённый) устройством дескриптор. Все дескрипторы в диапазоне от указателя хвоста до указателя головы считаются находящимися в обработке и неконсистентными;

- блок осуществляет предподкачку нескольких дескрипторов (до 32) во внутренние накристальные буфера памяти, откуда они могут быть выданы потребителям с минимальной задержкой;

- генерацию MSI-X прерываний (отдельный вектор прерывания на каждую очередь) по следующим событиям: опустошение очереди дескрипторов меньше заданного предела, наполнение очереди дескрипторов выше заданного предела либо исчерпание заданного таймаута с момента последнего возврата дескриптора устройством. Данные механизмы позволяют процессору своевременно снабжать очереди дескрипторами, а также обеспечивать разумный баланс между загруженностью прерываниями от очереди и задержкой обработки возвращённых дескрипторов;

- для всех очередей на передачу, блок обеспечивает разделение пропускной способности передающеготракта между приоритетами (максимально
8 приоритетов). Каждая очередь может произвольно иметь один из 8 приоритетов. Между каждой из равноприоритетных очередей разделение пропускной способности производится методом round-robin;

- блок снабжает тракт приёма дескрипторами из запрашиваемых очередей. Номер очереди вычисляется трактом приёма самостоятельно в зависимости от его настроек.

2.7.6 Legacy контроллеры в EIOH

Контроллер EIOH содержит следующий набор legacy контроллеров:

I2C-SPI – включает в себя контроллеры интерфейсов I2C, IPMB, SPI, контроллер прерываний IOEPIC, системный и Watchdog таймера, управление сбросом системы;

HDA - аудиоконтроллер высокого разрешения;

RS-232 - контроллер последовательного порта;

SPMC - контроллер управлением питания и режимами энергосбережения на уровне вычислительного комплекса;

GPIO/MPV - контроллер программируемых входов-выходов со встроенным функциональным блоком приема синхросигналов от систем реального времени или от генераторов меток точного времени.

2.7.6.1 Контроллер I2C-SPI и IOEPIC

Контроллер I2C-SPI обеспечивает работу с интерфейсами I2C, IPMB, SPI; дополнительно включает в себя IOEPIC — контроллер прерываний ввода-вывода с поддержкой виртуализации, системный и Watchdog таймеры, управление сбросом системы. Структурная схема I2C-SPI контроллера представлена на рисунке 2.7.14.

Состоит из следующих блоков.

Configuration Registers - конфигурационное пространство и пространство внутренних регистров Configuration Registers - обеспечивает доступ к операционным регистрам блоков I2C, IPMB, IOEPIC, SPI, System Timers и буферам чтения/записи.

System Arbiter/Decoder – арбитр/дешифратор отвечает за запросы/ответы, приходящие с IOLink. Формирует команды к IOLink по полученным прерываниям с контроллера.

IOEPIC - контроллер прерываний ввода-вывода с поддержкой виртуализации, обеспечивает управление и отправку прерываний через Arbiter/Decoder.

Reset Control - обеспечивает управление сбросом системы.

System и Watchdog таймеры обеспечивают работу по отправке прерываний через заданные промежутки времени и сброс системы.

I2C, IPMB, SPI - блоки отвечают за передачу сообщений по соответствующим интерфейсам.

Read Buffer и Write Buffer - буферы чтения/записи являются промежуточным звеном по передачи данных между Arbiter/Decoder и I2C, IPMB, SPI блоками.

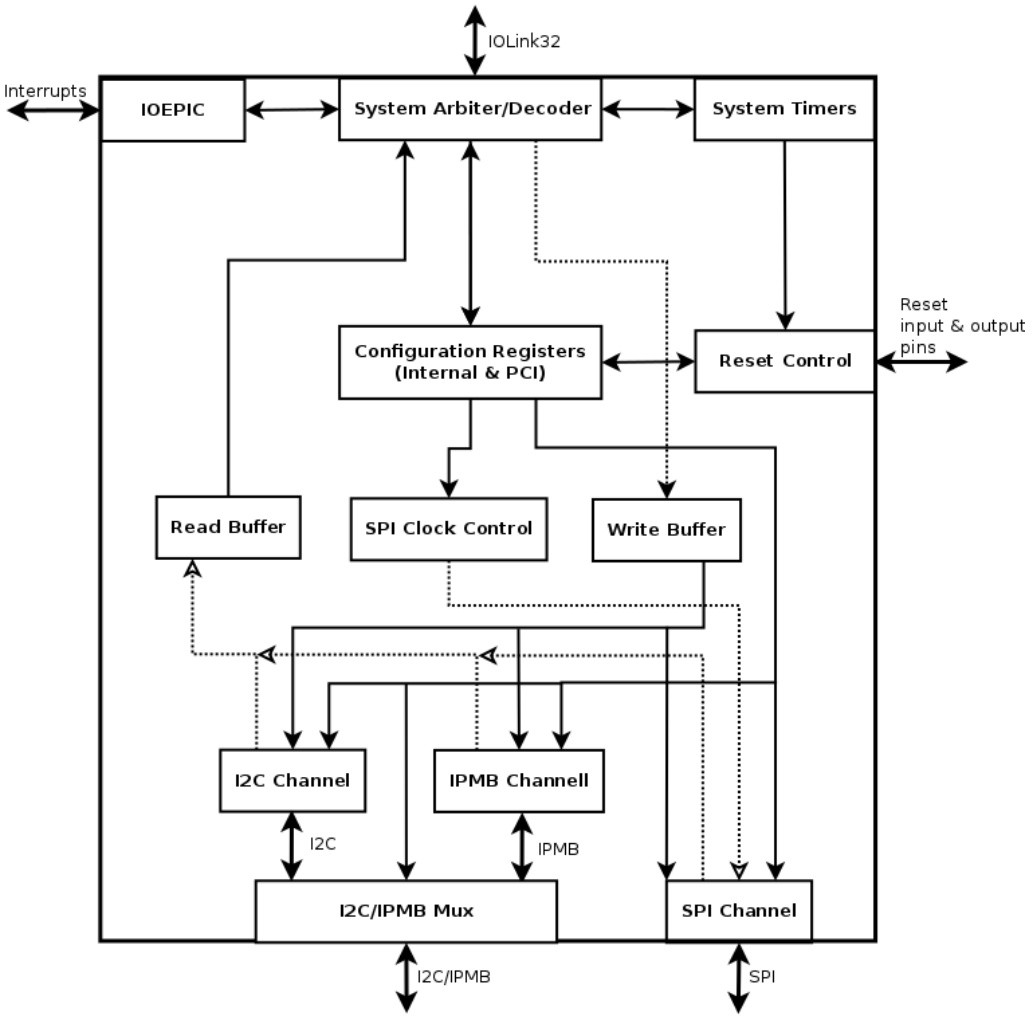


Рисунок 2.7.14 - Контроллер I2C-SPI

Технические характеристики.

Системная частота (iolink_clk) — до 500 МГц.

Опорная частота интерфейса SPI (spi_ref_clk) - 100 МГц.

Рабочая частота системного и Wathdog таймеров (tmr_clk) - 10 МГц.

SPI-Master:

- подключение до 4 устройств;
- поддержка SPI Mode 0,3;
- частоты SPI: 50, 25, 12,5, 6,25 МГц.

I2C/IPMB:

- подключение до 5 различных устройств I2C/IPMB;
- I2C Master;
- поддержка I2C Standard Mode, Fast Mode, Fast Mode Plus;
- поддержка 7- и 10-битной адресации (I2C);
- IPMB Master/Slave.

IOEPIC:

- поддержка до 64 прерываний;
- поддержка фронтовых и уровневых прерываний;
- отправка IOEPIC сообщений;
- программирование источника для поддержки виртуализации.

Таймеры и сброс:

- системный таймер;
- Watchdog таймер;
- управление сбросом системы.

Функционирование I2C-SPI.

Для доступа к конфигурационным регистрам используются конфигурационные обращения нулевого типа. Размер доступа - 1DW. При доступе к несуществующим конфигурационным регистрам, ошибка не возникает, запись не

имеет эффекта. Доступ к операционным регистрам осуществляется с помощью следующих базовых адресов, представленных в таблице 2.7.2

Таблица 2.7.2 - Базовые адреса контроллера I2C-SPI

Номер BAR	Назначение	Адрес
Base Address Register 0	I2C/SPI	10h
Base Address Register 1	Read/Write Buffers	14h
Base Address Register 2	System timers	18h
Base Address Register 3	IOEPIC	1Ch
Base Address Register 4	IPMB	20h

Для доступа к различным блокам I2C-SPI используется устройство, обеспечивающее управление по обмену данными через интерфейс IOLink - System Arbiter/Decoder. После получения команды через IOlink и ее дешифрации происходит чтение/запись данных во внутренние регистры блоков (контроллеров).

Для доступа к регистрам управления I2C, IPMB, SPI портов используется обращение в пространство памяти с 32-битным адресом. Регистры размещены в пространстве памяти с помощью BAR0 - I2C и SPI и BAR4 - IPMB. При обращении к несуществующим регистрам управления выдается ошибка. Размер доступа - 1 DW. В зависимости от настроек, может быть подключено пять I2C или IPMB устройств.

Для доступа к буферам записи и чтения используются обращения в пространство памяти. Размер доступа - 1, 2, 4, 8, 16 DW. Для обращения к буферу записи используются операции записи, буферу чтения - операции чтения.

Для обращения к System и Watchdog таймерам используются обращения в пространство памяти с 32-битным адресом. Регистры размещены в пространстве памяти с использованием BAR2. System timer может использоваться для жесткого, мягкого и системного сброса. Watchdog timer может работать в двух режимах. В первом режиме по достижению предела Watchdog таймера, генерируется сигнал сброса. Во втором случае, отправляется прерывание.

Для обработки внутренних и внешних прерываний используется контроллер IOEPIC. Регистры размещены в пространстве памяти с помощью BAR3. После получения прерываний, по алгоритму RoundRobin выбирается незамаскированное прерывание, проверяется Arbiter на выполнение задачи и, если Arbiter свободен, формируется команда, которая отправляется через IOlink. Сообщение состоит из 4 или 5 DW, в зависимости от размера адреса. В IOEPIC используется таблица прерываний, в которой могут храниться данные по 64 внутренним и внешним прерываниям. Прерывания могут быть как фронтовыми, так и уровневыми. Для блокировки прерывания используется маска. Используемые прерывания представлены в таблице 2.7.3.

Таблица 2.7.3 - Распределение прерываний IOEPIC контроллера

Номер входа IOEPIC	Источник прерывания
0	IPMB (внутри контроллера I2C-SPI)
1	SCI (power management, из SPMC контроллера)
2	System Timer (внутри контроллера I2C-SPI)
3	Ethernet0_tx0 – прерывание от передатчика нулевой очереди нулевого Ethernet контроллера
4	Ethernet0_tx1– прерывание от передатчика первой очереди нулевого Ethernet контроллера
5	Ethernet0_rx0 – прерывание от приёмника нулевой очереди нулевого Ethernet контроллера
6	Ethernet0_rx1 – прерывание от приёмника первой очереди нулевого Ethernet контроллера
7	Ethernet0_sys – системное прерывание от нулевого Ethernet контроллера
8	HDA (eioh)
9	Mpv_timers0
10	Mpv_timers1
11	Mpv_timers2

Номер входа IOEPIC	Источник прерывания
12	GPIO0
13	GPIO1
14	Serial Port
15	I2C/SPI (внутри контроллера I2C-SPI)
16	PCI IRQ A – legacy PCIE
17	PCI IRQ B – legacy PCIE
18	PCI IRQ C – legacy PCIE
19	PCI IRQ D – legacy PCIE
20	WD Timer (внутри контроллера I2C-SPI)
21	SATA
22	SERR (от всех устройств)
23	Ethernet1_tx0 – прерывание от передатчика нулевой очереди первого Ethernet контроллера
24	Ethernet1_tx1 – прерывание от передатчика первой очереди первого Ethernet контроллера
25	Ethernet1_rx0 – прерывание от приёмника нулевой очереди первого Ethernet контроллера
26	Ethernet1_rx1– прерывание от приёмника первой очереди первого Ethernet контроллера
27	Ethernet1_sys – системное прерывание от первого Ethernet контроллера
28	USB
29	WLCC
30 - 63	Резерв

2.7.6.2 Аудиоконтроллер высокого разрешения HDA

2.7.6.2.1 Технические характеристики

Основные технические характеристики аудиоконтроллера HDA (High Definition Audio) представлены в таблице 2.7.4.

Таблица 2.7.4 - Основные технические характеристики аудиоконтроллера HDA

Параметр	Значение
Скорость передачи данных, Мбит/с	48
Скорость приёма данных, Мбит/с	24
Число потоков воспроизведения	1
Число потоков записи	1
Число каналов в потоке	6
Поддерживаемые частоты дискретизации аудиопотока, кГц	от 6 до 192
Поддерживаемые разрядности квантов (замеров), бит	8, 16, 20, 24, 32

2.7.6.2.2 Структура контроллера

Аудиоконтроллер содержит пяти основных блоков: контроллер SLink, блок буферов (буфер команд, буфер ответов, буфер дескрипторов и буфер данных), блок операционных регистров, обработчик аудио потока, контроллер интерфейса с кодеком (High Definition Audio Interface).

Контроллер работает в двух режимах: «slave» и «DMA».

В режиме «slave» запросы записи/чтения регистров, приходящие от коммутатора EIOH, помещаются во внутренний буфер клиента системного интерфейса, затем по очереди выполняются. Выполнение следующего запроса невозможно до завершения текущего. Пакеты завершения непочтовых «slave» операций складываются во внутреннее FIFO “завершения slave операций/DMA запросов”.

В «DMA» режиме сформированный в блоке буферов пакет запроса поступает в клиент и (так же, как пакеты завершения «slave» операций) помещается в FIFO. При наличии доступа к системной шине клиент считывает очередной пакет из FIFO и выдаёт его на шину. Если в клиент одновременно поступают сигнал завершения непочтовой «slave» операции и «DMA» запрос, то пакет завершения «slave» операции будет записан в FIFO первым.

Буферы команд, ответов и дескрипторов являются кольцевыми и содержат соответствующие структуры данных. Буфер данных – линейный, содержит сэмплы аудиоданных. Структурная схема аудиоконтроллера представлена на рисунке 2.7.15.

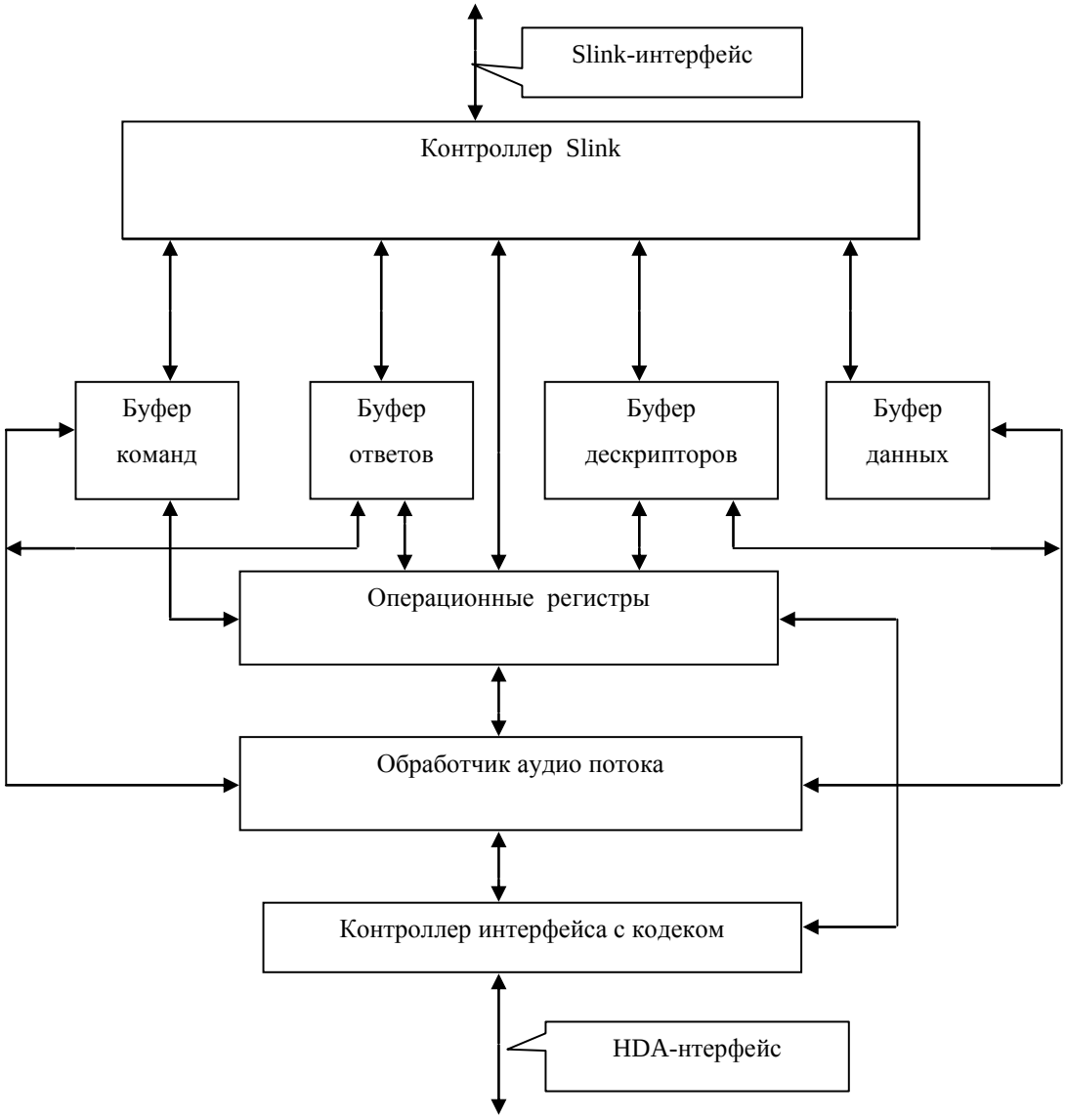


Рисунок 2.7.15 - Аудиоконтроллер HDA

2.7.6.3 Контроллер последовательного порта SP для реализации интерфейса RS-232

2.7.6.3.1 Структура SP

Контроллер последовательного порта SP обеспечивает подключение к интерфейсам RS-232/RS-485 через внешние микросхемы физического уровня (конкретный тип интерфейса определяется соответствующей микросхемой физического уровня).

Контроллер SP подключается к коммутатору EIOH через интерфейс Iolink. Он состоит из контроллера последовательных каналов SCC, а также, блока реализующего подключение к интерфейсу Iolink и содержащего регистры конфигурационного пространства. Структурная схема SP показана на рисунке 2.7.16.

Контроллер SP виден системе как функция 2 устройства 2 на внутренней шине EIOH. Взаимодействие с контроллерами осуществляется через операционные регистры. Для доступа к операционным регистрам контроллера последовательного канала в пространстве памяти выделены регистры.

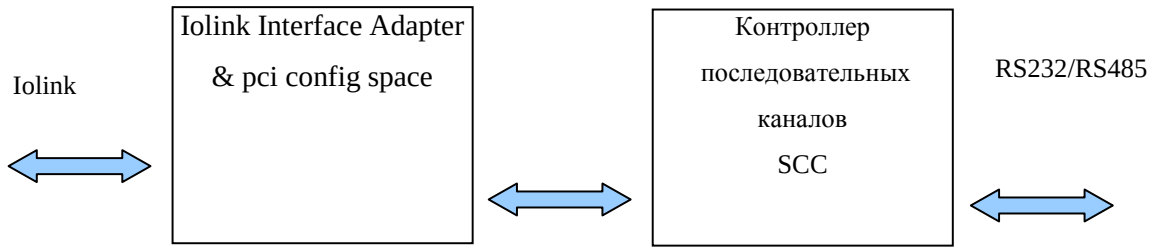


Рисунок 2.7.16 - Контроллер последовательного порта SP

2.7.6.3.2 Контроллер последовательных каналов SCC

Контроллер последовательных каналов SCC (Serial Channel Controller) содержит два полнодуплексных последовательных канала передачи. Характеристики контроллера представлены в таблице 2.7.5.

Таблица 2.7.5 - Характеристики контроллера последовательных каналов SCC

Параметр	Возможные значения	Значение по умолчанию	Регистр
Скорость, бит/с* ____ * при pclk = 4,9152 МГц	150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 76800, 115200	---	WR11 WR12 WR13 WR6 WR7
Кодировка	NRZ, NRZI, FM0, FM1	NRZ	WR10
Количество информационных битов на принимаемый символ	5, 6, 7, 8	8	WR3
Количество информационных битов на передаваемый символ	5, 6, 7, 8	8	WR5
Бит чётности	есть/нет, если есть, валиден 0 или 1	нет	WR4
Количество стоп-битов на символ	1; 1,5; 2	1	WR4
Источник сигнала синхронизации приёма	внешние сигналы - RTxC, TRxC, генератор частоты в бодах - BRG, схема цифровой АПЧ - DPPL	---	WR11
Источник сигнала синхронизации передачи	внешние сигналы - RTxC, TRxC, генератор частоты в бодах - BRG, схема цифровой АПЧ – DPPL	---	WR11

Структурная схема контроллера SCC приведена на рисунке 2.7.17

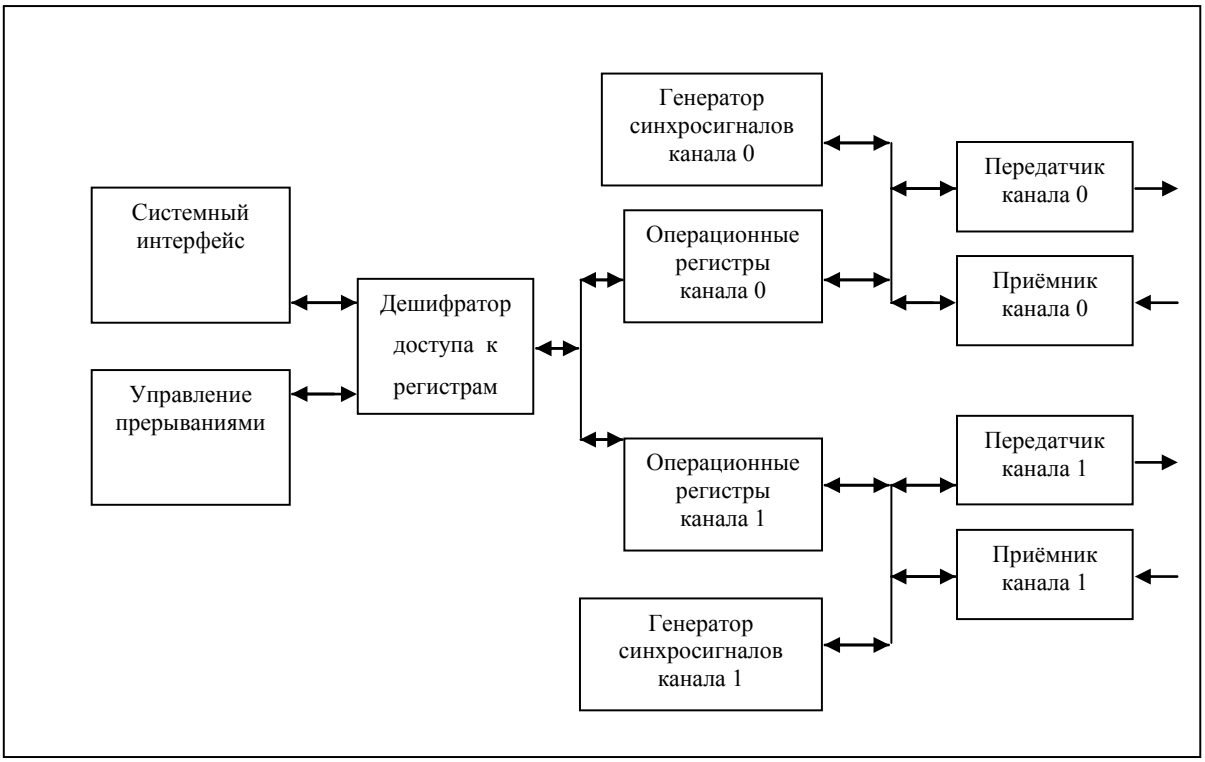


Рисунок 2.7.17 - Контроллер SCC

2.7.6.3.4 Контроллер управления питанием и энергосбережением SPMC

Аппаратная поддержка управления энергосбережением на уровне вычислительного комплекса реализована в контроллере SPMC, который помещен в домен неотключаемого питания. Управление энергосбережением производится программным путем из операционной системы путем записи/считывания программно-доступных регистров контроллера SPMC и обработки прерываний, обусловленных событиями энергосбережения, источником которых является SPMC.

Программно-доступные регистры SPMC позволяют реализовать следующие функции энергосбережения:

- отслеживание и обработка событий от кнопки питания;
- отслеживание и обработка событий от источника питания (переключение с питания от сети на питание от батареи);

- отслеживание и обработка событий от батареи (низкий уровень заряда батареи);

- отслеживание и обработка событий от таймера, отслеживающего состояния простоя (генерация прерывания при длительном состоянии простоя);

- перевод системы в состояние сна (пониженного энергопотребления) при длительном состоянии простоя (отсутствия задач);

- отслеживание и обработка событий от USB.

Контроллер SPMC поддерживает следующие состояния энергосбережения на уровне вычислительного комплекса:

S0 - active state;

S3- suspend-to-ram;

S4 – suspend-to-disk;

S5 – soft off.

За отключение питаний на плате отвечают выходные сигналы sleep_s3_, sleep_s4_, sleep_s5. Сигнал atn_sus может быть использован в качестве retention для режима S3 (suspend-to-ram).

Вход в состояние энергосбережения

Вход в состояние энергосбережения может осуществляться либо в автоматическом (без вмешательства ПО), либо программном режиме.

В автоматическом режиме возможен только переход в состояние s5 при нажатии и удержании кнопки «POWER» более 4 секунд. В остальных случаях используется программный переход, инициатором которого может быть

sci-прерывание.

Источники прерывания в s0 состоянии:

- PM таймер;

- нажатие кнопки «POWER»;

- изменение сигнала ac_power_psnt (подключение - отключение от сети переменного тока).

Последовательность перехода в состояние энергосбережения:

1 при переходе в состояния s3, s4, s5 сначала выставляется сигнал atn_sus на время, определяемое значением регистра ATNSUS_CNT, ноль в регистре означает, что этот сигнал не должен выставляться;

2 затем этот сигнал переходит в неактивное состояние с одновременным выставлением сигналов sleep_s3_, sleep_s4_, sleep_s5_ в значения соответствующие состоянию перехода:

s3 — sleep_s3_ → 0, sleep_s4_ → 1, sleep_s5_ → 1

s4 — sleep_s3_ → 0, sleep_s4_ → 0, sleep_s5_ → 1

s5 — sleep_s3_ → 0, sleep_s4_ → 0, sleep_s5_ → 0

Выход из состояния энергосбережения

Выход из состояний s3, s4, s5 энергосбережения невозможен, если ac_power_psnt ==0 и batlow_ == 0 (система работает от батареи, и её заряд недостаточен).

Инициаторами выхода из состояния энергосбережения могут быть следующие события:

- нажатие кнопки «POWER» (инициирует выход из состояний s3, s4, s5);

- получение запроса на выход из состояния энергосбережения wake_up_ (инициирует выход из состояний s3, s4).

Примечание - Минимальное время нахождения в состояниях s3, s4, s5 равно 1 секунде и достигается путем маскирования сигналов выхода из состояний на это время.

Последовательность выхода из состояния энергосбережения:

1 сигналы leep_s3_, sleep_s4_, sleep_s5_ переходят в неактивное состояние (→ 1);

2 ожидание установления всех системных питаний sys_pwrok == 1;

3 ожидание окончания ресетов;

4 выдача sci-прерывание.

2.7.6.5 Контроллер программируемых входов-выходов со встроенным функциональным блоком приема синхросигналов от систем реального времени или от генераторов меток точного времени GPIO/MPV

Контроллер ввода/вывода общего назначения позволяет программным способом управлять 16 линиями контроллера периферийных интерфейсов. Контроллер имеет следующие параметры:

- 16 линий ввода/вывода;
- системный интерфейс IOLink;
- настраиваемое направление линий (ввод или вывод);
- 2 группы прерываний (любую линию можно приписать к одному из двух прерываний);
- настраиваемый режим формирования прерываний (по фронту или уровню сигнала для каждой линии);
- программируемый уровень срабатывания прерывания для каждой линии.

Блок приема синхросигналов от систем реального времени или от генераторов меток точного времени с обеспечением микросекундной точности определения прихода этих сигналов относительно друг друга (МПВ) позволяет:

- принимать до трёх линий сигналов (2:0) внешних прерываний в соответствие с программно-задаваемой маской и полярностью;
- формировать сигналы прерывания системы, а также измерять время от получения сигнала прерывания до его восприятия системой (путём считывания значения счётчика текущего времени от момента получения прерывания);
- генерировать прерывания для каждой линии (2:0), выдавать генерируемый сигнал прерывания в систему через внешний вывод;
- выдавать в систему прерывания по трём линиям – по одной сигналы от каждой внешней линии;
- передавать в систему внешний сигнал прерывания для Ethernet контроллера напрямую для поддержки стандарта IEEE-1588.

Работой блока управляют с помощью операционных регистров, располагающихся в пространстве памяти.

2.8 Система синхронизации

Система синхронизации позволяет устанавливать оптимальную рабочую частоту устройств микропроцессора с учетом качества его изготовления, условий эксплуатации и текущей рабочей нагрузки. Большинство основных блоков микропроцессора имеют отдельные схемы формирования рабочих синхросигналов с установкой оптимальных рабочих частот блоков при аттестации микропроцессора после его изготовления и их оперативным изменением в целях поддержания температуры кристалла в заданном диапазоне и энергосбережения.

Микропроцессор имеет семь входных синхросигналов:

- четыре синхросигнала с частотой (100 ± 0,01) МГц используются в качестве опорного синхросигнала для получения рабочих частот почти всех основных блоков микропроцессора;

- синхросигнал с частотой (156,25 ± 0,015625) МГц используется в качестве опорного синхросигнала для одного блока E12G PHY x2 в том случае, если он работает в режиме Ethernet;

- синхросигнал с частотой (14,31818 МГц ± 0,001431818) МГц используется в качестве рабочего синхросигнала контроллера блока eFuse, регистров распределенной системы распространения и применения настроек из блока Efuse и таймеров SPMC-контроллера (System Power Management Controller), который управляет включением основных питаний процессора;

- синхросигнал контроллера JTAG может иметь частоту до 50 МГц.

Система синхронизации содержит следующие блоки:

- приемники входных опорных сигналов синхронизации LVDS (Low-Voltage Differential Signaling);

- умножители частоты входных опорных сигналов синхронизации UPLL (Ultra Phase-Locked Loop). Формируют основной сигнал синхронизации;

- делители частоты основного сигнала синхронизации UDIV (Ultra Divider) на фиксированные значения;

- блоки формирования синхросигналов БФС с программируемыми значениями делителей частоты основных синхросигналов.

Схемы формирования синхросигналов со входной частотой 100 МГц содержат цепочки блоков LVDS - UPLL - UDIV - БФС. Исходные настройки блоков UPLL, UDIV и БФС устанавливаются во время начальной загрузки микропроцессора.

Схемы формирования синхросигналов с входными частотами 156,25 и 14,318 МГц, а также для интерфейса JTAG состоят из одних входных блоков LVDS (для OO2 156,25 имеет тип сигнала HCSL), с выходов которых рабочие синхросигналы поступают сразу в свои контроллеры (Ethernet 10G, блока Efuse и JTAG).

Коэффициенты умножения в блоках UPLL зависят от назначения блоков и устанавливаются при начальной загрузке. Таким образом, почти все основные устройства микропроцессора могут иметь свою, отличную от других частоту синхронизации, заданную из соображений устойчивого функционирования при вариациях технологического процесса изготовления.

Блоки UPLL с частотой опорного синхросигнала 100 МГц формируют выходной синхросигнал с частотой 2 ГГц для большинства основных блоков микропроцессора. Исключение составляют блоки UPLL для контроллеров памяти MC, которые могут формировать выходной синхросигнал в диапазоне от 333 до 800 МГц, и блок UPLL для контроллеров USB, HDA и MPV, который формирует выходной синхросигнал с частотой 480 МГц.

Блоки UDIV имеют несколько вариантов исполнения со значениями деления частоты основного синхросигнала (1, 2, 3, 4 и 10) и формируют один или два синхросигнала с разной заданной частотой. Блоки UDIV для большинства основных блоков микропроцессора повторяют частоту своего входного синхросигнала 2 ГГц. Исключение составляют блоки UDIV:

- для контроллеров памяти MC, которые имеют входной синхросигнал в диапазоне от 333 до 800 МГц и формируют два выходных синхросигнала в этом же диапазоне;

- для устройства доступа к внешней памяти XMU, коммутаторов SC1 и SC2 контроллера периферийных интерфейсов EIOH, которые формируют два выходных синхросигнала с частотами 2 и 1 ГГц;

- для контроллеров USB, HDA и MPV, которые имеют входной синхросигнал с частотой 480 МГц и формируют два выходных синхросигнала с частотами 480 и 48 МГц.

Блоки БФС формируют четыре синхросигнала с раздельным управлением делителями частоты в диапазоне от 1 до 8 для каждого выходного синхросигнала. Делители частоты синхросигналов поступают в блоки БФС из устройства PCS (Power Control System). Блоки БФС используются для изменения частоты рабочих синхросигналов в большинстве основных блоков микропроцессора в целях энергосбережения и поддержания температуры кристалла в заданном диапазоне. Таким образом, частота синхронизации большинства основных блоков микропроцессора может быть адаптирована к рабочей нагрузке и температурному режиму эксплуатации.

3 Встроенные средства энергосбережения, контроля и восстановления работоспособности

3.1 Средства энергосбережения

В микропроцессоре реализованы средства снижения потребляемой мощности в случае постоянной или временной невостребованности аппаратных ресурсов. Эти средства включают:

- для процессорных ядер CORE: регулировка частоты и полное отключение синхронизации;
- для коммутационной сети OCN: регулировка частоты синхронизации;
- для кэша L3: схемы маскирования синхросигнала для неактивных блоков данных и регистров;
- для межпроцессорных каналов и канала ввода-вывода: регулировка частоты синхронизации, перевод в малопотребляющий режим и полное отключение синхронизации, аппаратное отключение синхронизации, если канал не подключен;
- для каналов обращения в оперативную память: регулировка частоты и полное отключение синхронизации, перевод в малопотребляющий режим.

3.2 Контроллер управления энергосбережением PCS

3.2.1 Общие сведения

Контроллер управления энергосбережением PCS (Power Control System) содержит средства управления энергопотреблением наиболее энергоемкой центральной части микропроцессора – процессорных ядер Core, коммутационной сети OCN и кэша L3. Эти средства используются в целях экономии энергопотребления при неполной рабочей нагрузке и для защиты от перегрева при чрезмерной рабочей нагрузке микропроцессора. В каждом процессорном ядре можно остано-

вить дещифрацию команд, снизить частоту синхронизации или отключить синхронизацию полностью. В коммутационной сети OCN и кэше L3 можно снизить частоту синхронизации.

Устройство PCS содержит также средства контроля и управления температурным режимом кристалла микросхемы с помощью встроенных термодатчиков и вениляторов охлаждения.

В состав устройства PCS входят:

- контроллер управления частотой синхронизации и общей политикой энергосбережения PMC (Power Manager Controller);

- блок датчиков PVT температуры, напряжения питания и качества изготовления (TS, VM, PD);

- контроллер датчиков PVT;

- два контроллера канала вентиляторов охлаждения FAN;

- блок системных событий;

- контроллер интерфейса I2C-slave, предоставляющий "внешнему system/power менеджеру" доступ к контроллерам FAN и PMC.

Контроллер PMC содержит регистры настройки частот системы синхронизации центральной части микропроцессора и реализует 8 режимов управления частотами синхронизации с разной степенью автоматизации отдельно для каждого устройства процессора.

Контроллер датчиков PVT:

- опрашивает датчики температуры TS, получая их показания во внутреннем представлении АЦП в режимах автоматического опроса показаний раз в 1,3 мс или однократного опроса;

- преобразует показания датчиков TS в градусы Цельсия;

- вычисляет максимум среди показаний всех термосенсоров;

- опрашивает датчики VM и PD под программным управлением.

Контроллер канала вентиляторов FAN обеспечивает:

- режим прямого программного управления (драйвер ОС или внешний менеджер задают темп вращения вентиляторов);

- режим автоматического управления по таблице с температурными порогами (драйвер ОС или внешний менеджер настраивают таблицу, а аппаратура изменяет скорость вращения вентиляторов в зависимости от текущей температуры).

Блок системных событий:

- регистрирует системные события;

- выдает прерывания при появлении интересующего события;

- по заданным событиям выполняет автоматическую аппаратную реакцию (снижение энергопотребления, перевод вентиляторов охлаждения на максимальные обороты и т.д.).

Контроллер интерфейса I2C-slave предоставляет внешний доступ к управлению частотой синхронизации и контроллерам каналов вентиляторов.

Краткие характеристики контроллера:

- рабочая частота 14,3 МГц;

- адрес устройства на шине - 7'b1001100;

- скорость передачи данных до 1 Мбит/с.

3.2.2 Структурная схема

Система управления энергосбережением PCS состоит из следующих модулей (рисунок 3.2.1):

- модуль PCS_TOP – включает в себя несколько других подмодулей;

– модуль PCS_CFG_PUB – коммутатор доступа к управляющим регистрам PMC и PVT контроллеров;

– модуль I2C DEV (SLAVE) – служебный блок для внешнего управления конфигурацией вентиляторов и регистрами PMC;

– модуль PCS_JTAG_REG – служебный блок для обращения к управляющим регистрам через JTAG;

– 2 модуля PWM Cotrloller – контроллеры вентиляторов охлаждения;

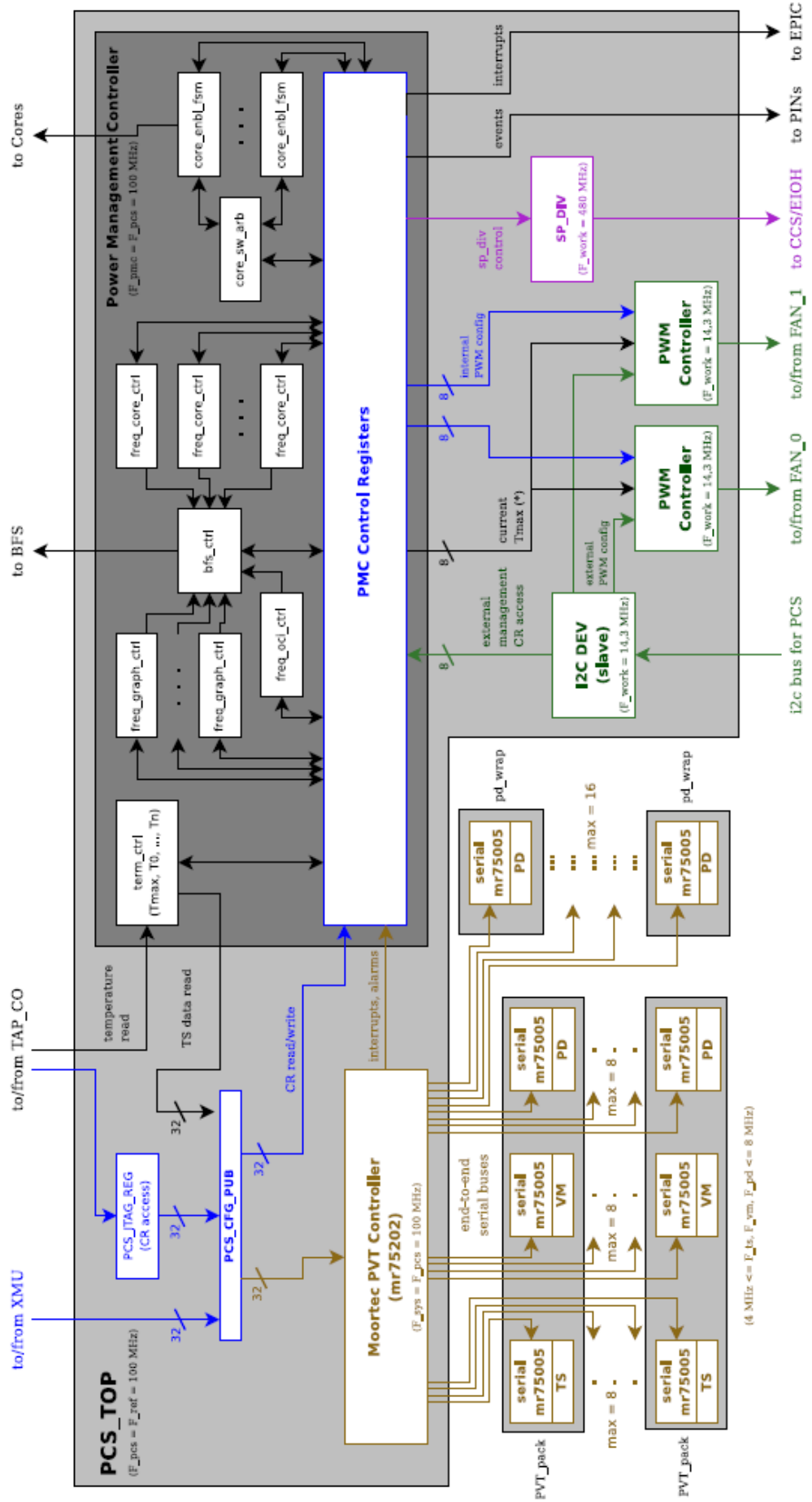


Рисунок 3.2.1 - Система управления энергосбережением PCS

– модуль SP_DIV – служебный блок для управления делителем частоты Serial Port (для нужд EIOH);

– модуль Power Management Controller – управление энергосбережением/энергопотреблением микропроцессора (PMC);

– модуль PVT Controller – контроллер датчиков PVT;

- 2 модуля ccs_clk_mux_scan – служебные мультиплексоры для подключения сигнала синхронизации JTAG TCK вместо рабочего синхросигнала;

- 8 модулей PVT_PACK – обертка для PVT-датчиков, включает несколько других подмодулей:

– 3 модуля serial mr75005 – адаптеры последовательного интерфейса для каждого PVT-датчика;

– модуль TS – термосенсор;

– модуль VM – монитор напряжения;

– модуль PD – детектор техпроцесса;

- 16 модулей pd_wrap – обертка для PD-датчиков, включает несколько других подмодулей:

– модуль serial mr75005 – адаптер последовательного интерфейса для каждого PD-датчика;

– модуль PD – детектор техпроцесса.

3.2.3 Контроллер и датчики PVT

Контроллер датчиков PVT подключен к 6 температурным сенсорам TS, 6 мониторам напряжения VM и 22 детекторам процесса PD. Контроллер PVT имеет параллельный интерфейс для обмена с системой и содержит адаптеры последовательной шины для подключения всех датчиков. Структурная схема контроллера PVT и его датчиков представлена на рисунке 3.2.2.

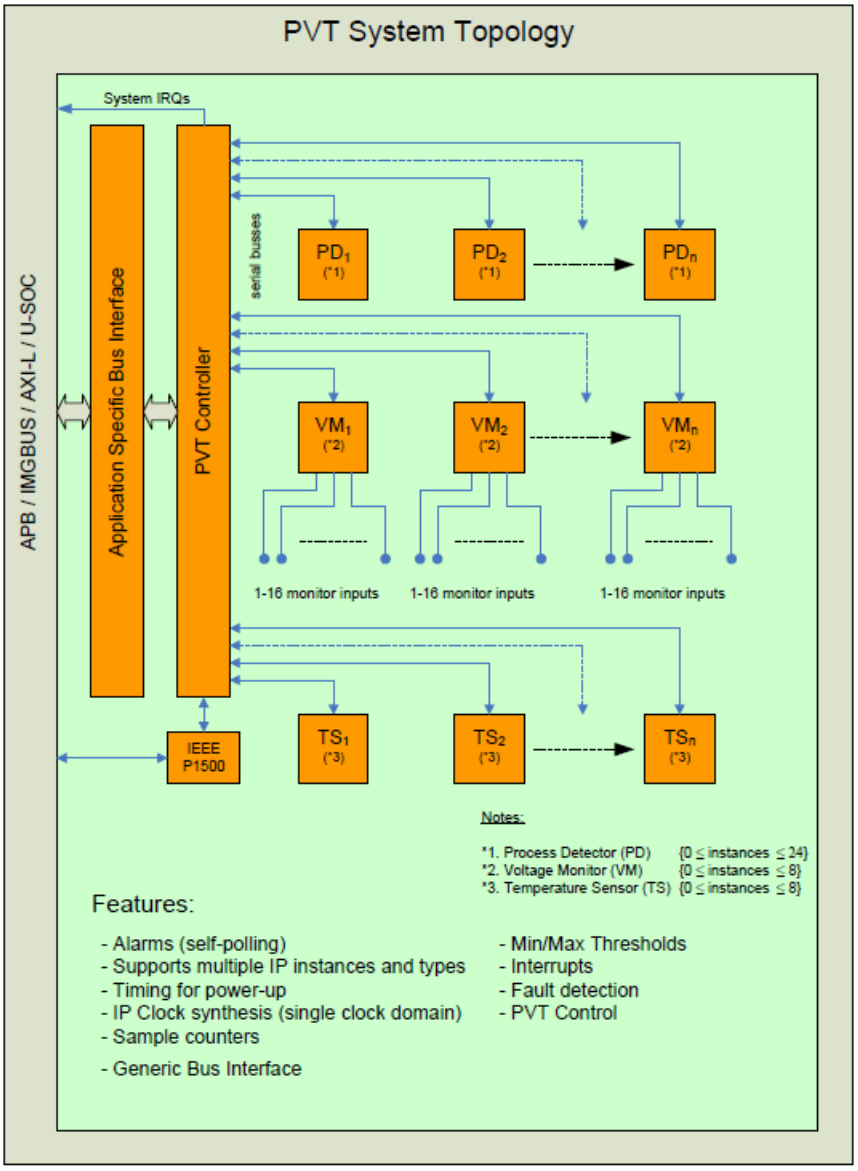


Рисунок 3.2.2 – Контроллер и датчики PVT

Температурный сенсор TS измеряет с высокой точностью температуру кристалла в заданной точке. Для увеличения точности измерений сенсор TS может быть откалиброван, причем для этого не требуется знать температуру кристалла. В типовое применение сенсора TS входит оптимизация частоты синхронизации, управление питанием, управление температурой и характеризация кристалла.

Основные характеристики:

- ±3,0 ºС точность без калибровки;
- ±1,0 ºС точность с калибровкой;

- 12-разрядный код результата измерения (10 или 8 разрядов при меньшей точности).

Монитор напряжения VM предназначен для измерения напряжений питания 0,8 В в ядре кристалла. Для увеличения точности монитор VM может быть откалиброван, причем для этого не требуется знать температуру кристалла. В типовое применение монитора VM входит измерение напряжения, измерение падения напряжения между различными точками измерений, управление питанием.

Основные характеристики:

- 10 точек измерения;
- ±1 % точность без калибровки;
- ±0,6 % точность с калибровкой;
- 14-разрядный код результата измерения.

Детектор процесса PD позволяет разработчикам интегральных схем обнаруживать изменения процесса, вызванные изменчивостью процесса изготовления, и отличия в улучшенных узлах цифровых КМОП устройств. Детектор PD измеряет частоту генерации сигнала в замкнутой цепи из набора инверторов.

Детектор PD может использоваться для непрерывной оптимизации управления напряжением питания и частотой, мониторинга вариаций производства, измерения задержки вентилей, анализа критических путей, анализа критических напряжений и мониторинга старения кристалла.

Основные характеристики:

- 5 встроенных кольцевых цепей задержки разной длины;
- возможность подключения внешней кольцевой цепи задержки;
- 12-разрядный код результата измерения.

3.3 Тестирование и разбраковка

Для отбраковки дефектных экземпляров микропроцессора предусмотрен режим скан-тестирования, в котором большинство триггеров микропроцессора логически объединяется в цепочку, что позволяет через канал JTAG загружать

произвольные тестовые последовательности, выполнять такт параллельного срабатывания и выдавать результат. Этот метод предназначен для проверки триггеров и комбинационной логики. Контроллеры канала JTAG размещены в каждом процессорном ядре и еще один общий контроллер канала JTAG предназначен для доступа к скан-цепям остальных блоков микропроцессора и контроллерам канала JTAG процессорных ядер.

Канал JTAG также используется для отладки микропроцессора. Через него управляется встроенная тестовая логика, позволяющая считывать внутреннее состояние процессорных ядер и внеядерных устройств, выполнять произвольные команды, обращаться в оперативную память и т.д.

3.4 Контроль работоспособности блоков кэш-памяти перед началом работы

Большинство блоков встроенной памяти процессорных ядер и внеядерных устройств снабжено схемами автоматической самопроверки BIST, которые запускаются либо аппаратно при включении процессора, либо командой через диагностический канал JTAG.

3.5 Исключение дефектных ячеек блоков кэш-памяти при сохранении общей работоспособности кэш-памяти

Для крупных блоков памяти, где вероятность дефекта повышена, предусмотрены средства исключения или замены дефектных ячеек на запасные — это позволяет повысить процент выхода годных микросхем.

Так в кэше L1 могут быть исключены 32-байтовые блоки с неисправными ячейками. После сигнала сброса выполняется BIST тестирование блоков памяти кэша и для неисправных блоков устанавливается бит запрета использования dontuse. Этот бит не позволяет механизму назначения использовать неисправный блок.

В кэше L2 предусмотрены дополнительные 2 разряда в каждом блоке 8 Кбайт, которыми можно заменить неисправные разряды. После сигнала сброса выполняется BIST тестирование памяти кэша и в неисправных блоках неисправные разряды автоматически заменяются на резервные.

В кэше L3 и кэше глобальной директории используются обе техники сокрытия дефектов изготовления. В кэше L3 предусмотрены дополнительные 2 разряда в каждом блоке 4 Кбайт массива данных и в каждом блоке 5 Кбайт массива адресных тегов, которыми можно заменить неисправные разряды. В кэше глобальнго справочника предусмотрены дополнительные 2 разряда в каждом блоке 4 Кбайт, которыми можно заменить неисправные разряды. После сигнала сброса выполняется BIST тестирование памяти кэша и неисправные разряды автоматически заменяются на резервные. Если дефекты невозможно исправить с помощью резервных разрядов, устанавливается блокировка использования для соответствующих 64-байтовых блоков.

3.6 Обнаружение и исправление в критических местах сбоев кэш-памяти и блоков оперативной памяти в процессе штатной работы

Помимо «постоянных» фабричных дефектов, блоки кэш-памяти и оперативной памяти могут быть подвержены временным сбоям, обусловленным воздействием реликтового излучения. Для обнаружения таких сбоев информация во всех блоках кэш-памятей и оперативной памяти сопровождается дополнительными контрольными разрядами.

В кэше L1 теги и каждый байт данных имеют дополнительные разряды для контроля по четности.

В кэше L2 теги и разряды механизма старения имеют дополнительные разряды для контроля по четности, а каждое 8-байтовое слово данных снабжено корректирующим кодом ECC, позволяющим корректировать одиночную ошибку и обнаруживать двойную.

В кэше L3 строки локального справочника (теги и разряды механизма старения), а также 32-байтовые блоки данных снабжены корректирующим кодом ECC, позволяющим корректировать одиночную ошибку и обнаруживать двойную.

Строки кэша глобального справочника также снабжены корректирующим кодом ECC, позволяющим корректировать одиночную ошибку и обнаруживать двойную.

Оперативная память снабжена средствами контроля корректности хранимой информации: каждые 32 байт данных сопровождаются дополнительными разрядами, содержащими код ECC, что позволяет корректировать одиночную ошибку и обнаруживать двойную.

Обнаружение ошибки при контроле по четности и двойной ошибки в коде ECC вызывает прерывание и обращение к операционной системе с целью устранения последствий ошибки.

3.7 Логические анализаторы

Для исследования функционирования микропроцессора в реальном времени предусмотрены встроенные логические анализаторы, к которым подключено большое количество внутренних сигналов различных устройств.

Логические анализаторы имеют встроенную память для записи трассы значений сигналов микропроцессора, подключенных к анализатору. Запись может осуществляться как в режиме реального времени, так и с паузами - это определяется статическими (программируемыми) режимами и динамическими состояниями анализатора. В логических анализаторах предусмотрены программируемые средства анализа входного потока и выработки событий, которые используются либо для изменения состояния анализатора, либо для выдачи вовне сигнала для остановки других анализаторов (включая себя) и микропроцессора. Логический анализатор программируется и запускается через процессорный канал JTAG.

4 Управляющие регистры

Содержание данного раздела оформлено документами ТВГИ.431281.028РЭ1 и ТВГИ.431281.028РЭ2.

5 Описание интерфейсов микропроцессора

Напряжение питания периферийных элементов типа SSTL_12	1,2 В
Напряжение питания периферийных элементов всех остальных типов, кроме USB	1,8 В
Напряжение питания периферийных элементов типа USB	3,3 В

Условные обозначения

Обозначение	Описание
in	вход
out	выход
inout	двунаправленный вход-выход
pullup	внутренняя утяжка на питание
pulldn	внутренняя утяжка на землю
analog	аналоговый сигнал
opendrain	выход с открытым коллектором
Rterm	внешняя (на корпусе) утяжка на землю

5.1 Системные сигналы

Название	Вход/ выход	Тип	Назначение
AC_POWER_PSNT	in	CMOS_18	Показывает, какого типа источник питания: 0 - аккумулятор; 1 - сеть переменного тока. Домен питания - "suspend"
ATE_MODE	in	CMOS_18 pulldn	Управление отладочным оборудованием: признак установки на тестирующий стенд и возможное отсутствие штатного входного синхросигнала. Активный уровень - высокий
ATN_SUS	out	CMOS_18	Сигнал о том, что скоро будет

Название	Вход/ выход	Тип	Назначение
			отключено питание; может быть использован для управления режимом retention. Активный уровень - низкий. Домен питания - "suspend"
BATLOW_ N	in	CMOS_18	Сигнал о низком уровне заряда аккумулятора. Активный уровень - низкий. Домен питания - "suspend"
CLK_REF_100M_BOT_P CLK_REF_100M_BOT_N	in	LVDS_18 Rterm	Опорный синхросигнал для получения рабочих частот процессорных ядер CORE0 – CORE7, а также каналов памяти DDR4 (MC и PHY) MC0 – MC3(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Подается парафазной линией. Рабочая частота 100 МГц. Для этого синхросигнала у процессора имеется встроенный (установленный на подложке корпуса) согласующий резистор с номиналом 100 Ом и мощностью рассеяния 0,063 Вт
CLK_REF_100M_EIOH_P CLK_REF_100M_EIOH_N	in	LVDS_18 Rterm	Опорный синхросигнал для получения рабочих частот EIOH и PHY PCIe, USB и SATA/ETH (кроме одного SATA/ETH, если для него выбрана конфигурация 2xETH). Подается парафазной линией. Рабочая частота 100 МГц. Для этого синхросигнала у процессора имеется встроенный установленный на подложке корпуса) согласующий резистор с номиналом 100 Ом и мощностью рассеяния 0,063 Вт
CLK_REF_100M_TEST_P CLK_REF_100M_TEST_N	in	LVDS_18 Rterm	Тестовый опорный синхросигнал. Подается парафазной линией.

Название	Вход/ выход	Тип	Назначение
			Рабочая частота 100 МГц. Для этого синхросигнала у процессора имеется встроенный (установленный на подложке корпуса) согласующий резистор с номиналом 100 Ом и мощностью рассеяния 0,063 Вт. При нормальной работе процессора Этот опорный синхросигнал не используется. Этот опорный синхросигнал используется только при расширенном тестировании процессора на предприятии-изготовителе. На серийных системных платах допускается не поддерживать такое тестирование, и тогда эти контакты следует соединить с GND
CLK_REF_100M_TOP_P CLK_REF_100M_TOP_N	in	LVDS_18 Rterm	Опорный синхросигнал для получения рабочих частот процессорных ядер CORE8 – CORE15, а также каналов памяти DDR4 (MC и PHY) MC4 – MC7(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Подается парафазной линией. Рабочая частота 100 МГц. Для этого синхросигнала у процессора имеется встроенный (установленный на подложке корпуса) согласующий резистор с номиналом 100 Ом и мощностью рассеяния 0,063 Вт
CLK_TEST_OUT_P[1:0] CLK_TEST_OUT_N[1:0]	out	LVDS_18	Тестовые выходы для синхросигналов. Выдаются парафазными линиями. Эти тестовые выходы используются только при расширенном тестировании процессора на предприятии-изготовителе. На серийных системных платах допускается не поддерживать такое тестирование, и тогда эти контакты

Название	Вход/ выход	Тип	Назначение
			следует соединить с GND
CLK_TEST_VREF	in	analog	Опорное напряжение для тестовых выходов для синхросигналов clk_test_out_p, clk_test_out_n. Номинал 1,2 В, входной ток не более ±20 мкА. Это опорное напряжение используется только при расширенном тестировании процессора на предприятии-изготовителе. На серийных системных платах допускается не поддерживать такое тестирование, и тогда этот контакт следует соединить с GND
CPU_BSP	in	CMOS_18	Признак BSP-процессора
CPU_DSBL_SOFT_RST_N	in	CMOS_18 pullup	Запрет выработки программного ресета; может использоваться в случае наличия КПИ-2. Активный уровень - низкий
CPU_HRST_IN_ N	in	CMOS_18 pullup	Сигнал "жесткого" обнуления процессоров, кроме bsp. Активный уровень - низкий. Домен питания - "suspend"
CPU_HRST_OUT_ N	out	CMOS_18	Сигнал "жесткого" обнуления процессора. Выдается bsp процессором. Активный уровень - низкий. Домен питания - "suspend"
CPU_SRST_IN_ N	in	CMOS_18 pullup	Сигнал "мягкого" обнуления процессоров, кроме bsp. Активный уровень - низкий. Домен питания - "suspend"
CPU_SRST_OUT_ N	out	CMOS_18	Сигнал "мягкого" обнуления процессора. Активный уровень - низкий. Выдается bsp процессором. Домен питания - "suspend"

Название	Вход/ выход	Тип	Назначение
DBG_RST_DSBL	in	CMOS_18 pulldn	Управление отладочным оборудованием: запрет reset встроенных логических анализаторов. Активный уровень - высокий
DBG_STOP	in	CMOS_18 pulldn	Управление отладочным оборудованием: остановка процессора после reset. Активный уровень - высокий
DDR_PWROK[3:0]	in	CMOS_18	Признак нахождения DDRPHY в состоянии холодного сброса при включении питания/признак входа DDRPHY в режим энергосбережения S3. Соединяется на печатной плате с сигналом ATN_SUS. Активный уровень - низкий. Домен питания - "suspend"
EFUSE_MODE[1:0]	in	CMOS_18	Признак тестового режима для оборудования, связанного с eFUSE. Задает один из 4 режимов запуска процессора: 0 - полевой режим; 1 - обход efuse; 2 - проверка efuse; 3 - программирование efuse
FREQ_MODE[1:0]	in	CMOS_18 pulldn	Ограничение частот процессорных ядер и общей кэш-памяти третьего уровня (предназначено для машин, в которых требуется ограничить потребляемую процессором мощность): 0 (00) - ограничения частот нет; 1 (01) - самое слабое ограничение; 2 (10) - более сильное ограничение; 3 (11) - самое сильное ограничение
IPL_GEN2_ADAPT	in	CMOS_18	Выбор начальной скорости для

<table>
<tr><th>Название</th><th>Вход/ выход</th><th>Тип</th><th>Назначение</th></tr>
<tr><td></td><td></td><td></td><td>всех межпроцессорных линков:
0 – 2,5 Гбит/с;
1 – 5,0 Гбит/с;
(он же ipcc_initial_speed "speed jumper for ipcc channels")</td></tr>
<tr><td>IPL_MULTILINK</td><td>in</td><td>CMOS_18</td><td>Включение режима multilink. Требуется для двухпроцессорных машин с соединением процессоров двумя или тремя межпроцессорными линками</td></tr>
<tr><td>LIMIT_CORES[3:0]</td><td>in</td><td>CMOS_18 pulldn</td><td>Ограничение количества включенных процессорных ядер (для машин, в которых требуется ограничить потребляемую процессором мощность):
<table>
<tr><th>[3:0]</th><th>Количество включенных ядер</th></tr>
<tr><td>0 (0000)</td><td>16</td></tr>
<tr><td>1 (0001)</td><td>CORE0</td></tr>
<tr><td>2 (0010)</td><td>CORE0 и CORE1</td></tr>
<tr><td>...</td><td></td></tr>
<tr><td>14 (1110)</td><td>CORE0 - CORE13</td></tr>
<tr><td>15 (1111)</td><td>CORE0 – CORE14</td></tr>
</table>
*Для OO1 необходимо устанавливать значение большее или равное 7, либо 0</td></tr>
</table>

<table>
<tr><th>Название</th><th>Вход/ выход</th><th>Тип</th><th>Назначение</th></tr>
<tr><td>LIMIT_PHYS[1:0]</td><td>in</td><td>CMOS_18 pulldn</td><td>Ограничение конфигурации включенных межпроцессорных линков и каналов памяти DDR4 (предназначено для машин, в том числе таких, в которых предусмотрено использование только 4 каналов памяти DDR4(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)):
<table>
<tr><td>[1:0]</td><td>Ограничение конфигурации</td></tr>
<tr><td>0 (00)</td><td>ограничения нет</td></tr>
<tr><td>1 (01)</td><td>отключены линки IPLA, IPLB</td></tr>
<tr><td>2 (10)</td><td>отключены линки IPLA, IPLB и каналы MC0, MC2, MC4, MC6</td></tr>
<tr><td>3 (11)</td><td>отключены линки IPLA, IPLB и каналы MC1, MC3, MC5, MC7</td></tr>
</table></td></tr>
<tr><td>PWR_BTN_ N</td><td>in</td><td>CMOS_18</td><td>Сигнал о нажатии кнопки включения питания.
Активный уровень - низкий.
Домен питания - "suspend"</td></tr>
<tr><td>RFU[9:0]</td><td>inout</td><td>CMOS_18 pulldn</td><td>Резервные контакты.
Для OO1 не используются,
но могут быть задействованы по мере необходимости (в том числе в будущих ревизиях кристалла процессора) - скорее всего, как входы, но не исключено, что и как выходы
*RFU[0] (in, pullup) для OO2, контакт используется для определения версии substrate:
0 - кристалл в "старом" корпусе,
1 - кристалл в "новом" корпусе.</td></tr>
<tr><td>RT</td><td>in</td><td>CMOS_18 pulldn</td><td>Синхросигнал точного времени 1 Гц.
Используется для подстройки счетчика тактов опорной частоты</td></tr>
<tr><td>SLEEP_S3_ N</td><td>out</td><td>CMOS_18</td><td>Управление питанием в режиме энергосбере-</td></tr>
</table>

Название	Вход/ выход	Тип	Назначение
			жения - уровни S3 - S5. Активный уровень - низкий. Домен питания - "suspend"
SLEEP_S4_ N	out	CMOS_18	Управление питанием в режиме энергосбережения - уровни S4 - S5. Активный уровень - низкий. Домен питания - "suspend"
SLEEP_S5_ N	out	CMOS_18	Управление питанием в режиме энергосбережения - уровень S5. Активный уровень - низкий. Домен питания - "suspend"
SUS_CLK	in	CMOS_18	Синхросигнал логики домена "неотключаемого" ("suspend") питания; рабочая частота 14,31818 МГц. Домен питания - "suspend"
SUS_PWROK	in	CMOS_18	Сигнал о завершении включения на плате "неотключаемого" ("suspend") питания. Активный уровень - высокий. Домен питания - "suspend" *Для ОО1 при отключении штатного питания (sys_pwrok) кнопкой на плате должно также отключаться suspend-питание (не относится к перезагрузке); наличие suspend-питания должно отображаться на плате светодиодом
SYS_ETHREFCLK_P SYS_ETHREFCLK_N	in	LVDS_18 Rterm	Для ОО1. Опорный синхросигнал. Подается парафазной линией. Рабочая частота 156,25 МГц. Для этого синхросигнала у процессора имеется встроенный (установленный на подложке корпуса) согласующий резистор с номиналом

Название	Вход/ выход	Тип	Назначение
			100 Ом и мощностью рассеяния 0,063 Вт. Этот опорный синхросигнал используется только при условии, что на вход процессора SATAETH_CONFIG подается константа 1. Иначе допускается не подавать этот синхро-сигнал, и тогда эти контакты следует соединить с GND.
SYS_ETHREFCLK_P SYS_ETHREFCLK_N	in	HCSL noterm	Для OO2. Опорный синхросигнал для получения рабочих частот одного PHY SATA/ETH. Этому PHY принадлежат контакты SATAETH_{R,T}LANE_{P,N}[2], SATAETH_{R,T}LANE_{P,N}[3]). Подается парафазной линией. Рабочая частота должна быть 156.25 МГц,при условии, что выбран 2xETH режим(на вход процессора SATAETH_CONFIG подана константа 1). Рабочая частота должна быть 100 МГц, при условии, что выбран 4xSATA режим(на вход процессора SATAETH_CONFIG подана константа 0).
SYS_GPIO[15:0]	inout	CMOS_18 pullup	Системные сигналы GPIO
SYS_KPI2BOOT_ENA	in	CMOS_18 pulldn	Определяет направление считывания программа начальной загрузки (boot); 0 - доступ осуществляется через встроенный канал процессора; 1 - доступ осуществляется через микросхему КПИ-2, подключенную к процессору
SYS_PLTRST_ N	out	CMOS_18	Сигнал обнуления всех элементов вычислительной системы, за исключением процессоров. Выдается bsp процессором. Активный уровень - низкий. Домен питания - "suspend"
SYS_PWROK	in	CMOS_18	Сигнал о завершении включения на плате

<table>
<tr><th>Название</th><th>Вход/ выход</th><th>Тип</th><th>Назначение</th></tr>
<tr><td></td><td></td><td></td><td>штатного питания.
Активный уровень - высокий.
Домен питания - "suspend"</td></tr>
<tr><td>TCK</td><td>in</td><td>CMOS_18</td><td>Интерфейс JTAG. Сигнал синхронизации</td></tr>
<tr><td>TDI</td><td>in</td><td>CMOS_18 pullup</td><td>Интерфейс JTAG. Вход цепи сканирования</td></tr>
<tr><td>TDO</td><td>out</td><td>CMOS_18</td><td>Интерфейс JTAG. Выход цепи сканирования</td></tr>
<tr><td>TEST_AP_[1:0]_PWR_0V8_ CORE
TEST_AP_[1:0]_GND_0V8_CORE</td><td>out</td><td>analog</td><td>Тестовые выводы:
<table>
<tr><th>Разряд</th><th>Питание core</th></tr>
<tr><td>0</td><td>нижний левый угол кристалла (зона ядер), рядом с каналами памяти 0 и 1(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В теку-щей итерации МП не поддерживается)</td></tr>
<tr><td>1</td><td>нижний правый угол кристалла (зона ядер), рядом с каналами памяти 4 и 5(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В теку-щей итерации МП не поддерживается)</td></tr>
</table></td></tr>
<tr><td>TEST_AP_[3:0]_PWR_0V8_ MC
TEST_AP_[3:0]_GND_0V8_MC</td><td>out</td><td>analog</td><td>Тестовые выводы:
<table>
<tr><th>Разряд</th><th>Питание uncore в области каналов памяти</th></tr>
<tr><td>0</td><td>0 и 1</td></tr>
<tr><td>1</td><td>2 и 3</td></tr>
<tr><td>2</td><td>4 и 5</td></tr>
<tr><td>4</td><td>6 и 7</td></tr>
</table></td></tr>
</table>

Название	Вход/ выход	Тип	Назначение
			(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)
TMS	in	CMOS_18 pullup	Интерфейс JTAG. Включение режима JTAG. Примечание - Перед подключением JTAG-контроллера необходимо убедиться, что он работает в таком режиме, когда уровни сигналов не превышают 1,8 В
TRIG_IN	in	CMOS_18 pulldn	Сигнал остановки от внешнего логического анализатора. Активный уровень - высокий
TRIG_OUT	out	CMOS_18 pulldn	Сигнал остановки для внешнего логического анализатора. Активный уровень - высокий
WAKE_UP_N	in	CMOS_18	Сигнал о приходе события для выхода из энергосберегающего режима. Активный уровень - низкий. Домен питания - "suspend"

<table>
<tr><th>Название</th><th>Вход/ выход</th><th>Тип</th><th>Назначение</th></tr>
<tr><td>WLCC_SPEED_PRESETS[2:0]</td><td>in</td><td>CMOS_18 pulldn</td><td>Выбор настроек для канала WLCC. Одновременно выбирается битовая скорость канала BR (bitrate) и настройка коррекции АЧХ передатчиков DE (TX de-emphasis). При неудачном установлении соединения возможно автоматическое переключение на резервную скорость RBR (reserve BR), RBR = BR/2 для всех настроек:
<table>
<tr><th>Код [2:0]</th><th>BR, Гбит/с</th><th>RBR, Гбит/с</th><th>DE, дБ</th></tr>
<tr><td>0 (000)</td><td>5</td><td>2,5</td><td>3,7</td></tr>
<tr><td>1 (001)</td><td>6</td><td>3</td><td>3,7</td></tr>
<tr><td>2 (010)</td><td>5</td><td>2,5</td><td>6,0</td></tr>
<tr><td>3 (011)</td><td>6</td><td>3</td><td>6,0</td></tr>
<tr><td>4 (100)</td><td>2,5</td><td>1,25</td><td>0</td></tr>
<tr><td>5 (101)</td><td>4</td><td>2</td><td>3,7</td></tr>
<tr><td>6 (110)</td><td>4</td><td>2</td><td>6,0</td></tr>
<tr><td>7 (111)</td><td>8</td><td>4</td><td>3,7</td></tr>
</table>
</td></tr>
</table>

5.2 Каналы связи с оперативной памятью DDR4

Микропроцессор имеет 8 каналов памяти: 0, 1, 2, 3, 4, 5, 6, 7(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Они имеют одинаковые интерфейсы, и в таблице сигналов символ j представляет одну из этих цифр.

Название	Вход/ выход	Тип	Назначение
MCj_A[17:0]	out	SSTL_12	Адрес строки / Адрес столбца / команда
MCj_ACT_N	out	SSTL_12	Признак команды Activate

Название	Вход/ выход	Тип	Назначение
MCj_ALERT_N	inout	SSTL_12	Ошибка четности адресной шины или CRC данных, либо дополнительный тестовый вывод цифровых сигналов PHY. Указания по подключению: - при размещении микросхем памяти на материнской плате утягивающий резистор 50 Ом к верхнему уровню (pull-up) должен быть установлен в конце линии Alert_n после последней микросхемы SDRAM; - в случае использования модулей DIMM утягивающий резистор 50 Ом к верхнему уровню (pull-up) должен быть установлен на линии Alert_n в непосредственной близости к микропроцессору
MCj_BA[1:0]	out	SSTL_12	Адрес логического банка в группе банков
MCj_BG[1:0]	out	SSTL_12	Номер группы банков
MCj_CID[2:0]	out	SSTL_12	Номер логического ранка (стека) в модулях памяти типа 3DS. Примечание - К выводу CS2_n/C0 модуля памяти должен быть подведён вывод MCj_CID[0] процессора, к выводу CS3_n/C1 – вывод MCj_CID[1] процессора
MCj_CK_N[3:0]	out	SSTL_12	Синхросигналы. Передаются парафазными линиями
MCj_CKE[3:0]	out	SSTL_12	Clock enable. Высокий уровень активирует IO-буферы и синхроимпульс в модуле памяти. Сигнал используется при инициализации модулей памяти и их переводе в ждущее состояние
MCj_CS_N[3:0]	out	SSTL_12	Позиционный адрес физического банка (Chip select). Сигнал выбирает один из четырёх физических банков при отсылке команды в память. Активный уровень - низкий. Важно: к пину CS2_n/C0 модуля памяти должен быть подведён пин MCj_CID[0] процессора, к

Название	Вход/ выход	Тип	Назначение
			CS3_n/C1 - MCj_CID[1]
MCj_DQ[71:0]	inout	SSTL_12	Двунаправленная шина данных
MCj_DQS_P[17:0] MCj_DQS_N[17:0]	inout	SSTL_12	Дифференциальные линии строба данных. При записи передаётся из контроллера в модуль памяти со сдвигом pi/2. При чтении передаётся из памяти в контроллер без сдвига
MCj_MTEST	out	SSTL_12	Основной тестовый вывод цифровых сигналов PHY. Сигналы MCj_MTEST необходимо вывести на тестовые точки. Эти сигналы требуют согласования (100 Ом на GND) || (100 Ом на 1,2 В). Рядом с каждой тестовой точкой установить тестовую точку GND. При необходимости можно эти сигналы оставить не подключенными. Проводники, идущие к тестовым точкам, должны быть защищены от помех (экранированные GND). Не допускать прохождения над разрезами полигонов
MCj_ODT[3:0]	out	SSTL_12	Управление терминацией в модулях памяти. Активный уровень - высокий
MCj_PARITY	out	SSTL_12	Четность адресно-командной шины
MCj_RESET_N	out	SSTL_12	Сброс. Активный уровень - низкий
MCj_VREF_TEST	analog inout	SSTL_12	Тестовый вывод аналоговых сигналов PLL, либо опорное напряжение с MASTER_PHY. Нельзя использовать в качестве источника сигнала VREFCA для модулей памяти. Сигналы MCj_VREF_TEST необходимо вывести на тестовые точки. Рядом с каждой тестовой точкой установить тестовую точку GND. При необходимости можно эти сигналы оставить не подключенными. Проводники, идущие к тестовым точкам, должны быть

Название	Вход/ выход	Тип	Назначение
			защищены от помех (экранированные GND). Не допускать прохождения над разрезами полигонов
MCj_ZN	analog out	SSTL_12	Подключение внешнего калибровочного сопротивления 240 ± 1% Ом. Калибровочный резистор должен располагаться в непосредственной близости к микропроцессору. Проводники MCj_ZN идущие к калибровочным резисторам должны быть защищены от помех (экранированные GND). Не допускать прохождения над разрезами полигонов

5.3 Канал SPI

Название	Вход/ выход	Тип	Назначение
SPI_CS[3:0]_N	out	CMOS_18	Сигналы chip select
SPI_MISO	in	CMOS_18	Входные данные (master input, slave output)
SPI_MOSI	out	CMOS_18	Выходные данные (master output, slave input)
SPI_SCK	out	CMOS_18	Синхросигнал

5.4 Каналы I2C

Микропроцессор имеет 5 каналов I2C master: 0 - 4. Они имеют одинаковые интерфейсы, и в таблице сигналов символ j представляет одну из этих цифр.

Название	Вход/ выход	Тип	Назначение
I2C_SCLj	inout	CMOS_18 opendrain	Синхросигнал
I2C_SDAj	inout	CMOS_18 opendrain	Адрес/данные

5.5 Канал HDA

Название	Вход/ выход	Тип	Назначение
HDA_CLK	out	CMOS_18	Синхросигнал
HDA_RST_ N	out	CMOS_18	Сигнал начальной установки. Активный уровень - низкий
HDA_SDI	inout	CMOS_18	Линия приема ответов на команды и потока записи от кодека. Выходная линия в процессе присвоения кодекам адресов
HDA_SDO	out	CMOS_18	Линия передачи команд и потока воспроизведения в кодек
HDA_SYNC	out	CMOS_18	Маркер фреймов и потоков воспроизведения (Frame Synchronization). Активный уровень - высокий

5.6 Канал WLCC/PCIE0

Возможны следующие конфигурации:

1) PCIE0 x16

2) PCIE0 2x8

3) PCIE0 x8 + WLCC x8

4) PCIE0 2x4 + WLCCx8

5) WLCC x16

Назначения сигналов в конфигурациях:

1) PCIE0[15:0]: IOWL_PE_{R, T} LANE_{P, N}[15:0]

2) PCIE0[15:8] + PCIE0[7:0]: IOWL_PE_{R, T} LANE_{P, N}[15:8] + IOWL_PE_{R, T} LANE_{P, N}[7:0]

3) PCIE0[7:0] + WLCC[7:0]: IOWL_PE_{R, T} LANE_{P, N}[7:0] + IOWL_PE_{R, T} LANE_{P, N}[15:8]

4) PCIE0[7:4] + PCIE0[3:0] + WLCC[7:0]: IOWL_PE_{R, T} LANE_{P, N}[7:4] + IOWL_PE_{R, T} LANE_{P, N}[3:0] + IOWL_PE_{R, T} LANE_{P, N}[15:8]

5) WLCC[15:0]: IOWL_PE_{R, T} LANE_{P, N}[15:0]

Название	Вход/ выход	Тип	Назначение
IOWL_PE_RESREF_F	inout	analog	Подключение калибровочного сопротивления (200 ± 1%) Ом, 100-ppm/C
IOWL_PE_CONFIG[1:0]	in	CMOS_18	Конфигурация канала: IOWL_PE_CONFIG[0] - наличие WLCC: 0 - нет; 1 - есть; IOWL_PE_CONFIG[1] - ширина WLCC: 0 - x16; 1 - x8
IOWL_PE_PRE_DET[1:0]	in	CMOS_18	Сигналы Presence Detect, соответствующие подключенному PCIe устройству. Активный уровень - низкий
IOWL_PE_RLANE_P[15:0] IOWL_PE_RLANE_N[15:0]	in	PCIe	Входные данные. Передаются парафазной линией
IOWL_PE_TLANE_P[15:0]	out	PCIe	Выходные данные.

Название	Вход/ выход	Тип	Назначение
IOWL_PE_TLANE_N[15:0]			Передаются парафазной линией

5.7 Канал IP-linkA

Название	Вход/ выход	Тип	Назначение
IPLA_ADAPT_RST_IN	in	CMOS_18	Входной сигнал сброса для адаптации межпроцессорного линка A
IPLA_ADAPT_RST_OUT	out	CMOS_18	Выходной сигнал сброса для адаптации межпроцессорного линка A
IPLA_FLIP_EN	in	CMOS_18	Включение режима перекрутки межпроцессорного линка A (требуется для тех многопроцессорных машин, в которых "перекручен" линк A)
IPLA_RESREF_F	inout	analog	Подключение калибровочного сопротивления (200 ± 1%) Ом, 100-ppm/C
IPLA_RLANE_P[15:0] IPLA_RLANE_N[15:0]	in	PCIe	Входные данные. Передаются парафазной линией
IPLA_TLANE_P[15:0] IPLA_TLANE_N[15:0]	out	PCIe	Выходные данные. Передаются парафазной линией

5.8 Канал IP-linkB

Название	Вход/ выход	Тип	Назначение
IPLB_ADAPT_RST_IN	in	CMOS_18	Выходной сигнал сброса для адаптации межпроцессорного линка B
IPLB_ADAPT_RST_OUT	out	CMOS_18	Выходной сигнал сброса для адаптации межпроцессорного линка B
IPLB_RESREF_F	inout	analog	Подключение калибровочного

Название	Вход/ выход	Тип	Назначение
			сопротивления 200 ± 1% Ом, 100-ppm/ºС
IPLB_RLANE_P[15:0] IPLB_RLANE_N[15:0	in	PCIe	Входные данные. Передаются парафазной линией
IPLB_TLANE_P[15:0] IPLB_TLANE_N[15:0]	out	PCIe	Выходные данные. Передаются парафазной линией

5.9 Канал IP-linkC/PCIE1

Возможны следующие конфигурации:

1) IPCC x16

2) PCIE1 1x8 + IPCCx8

3) PCIE1 2x4 + IPCCx8

4) PCIE1 x16

Назначение сигналов в конфигурациях:

1) IPCC[15:0]: IPLC_PE_{R, T} LANE_{P, N}[15:0]

2) PCIE1[7:0] + IPCC[7:0]: IPLC_PE_{R, T} LANE_{P, N}[7:0] +

+ IPLC_PE_{R, T} LANE_{P, N}[15:8]

3) PCIE1[3:0] + PCIE [3:0] + IPCC[7:0]:

IPLC_PE_{R, T} LANE_{P, N}[7:4] + IPLC_PE_{R, T} LANE_{P, N}[3:0] +

+ IPLC_PE_{R, T} LANE_{P, N}[15:8]

4) PCIE1[15:0]: IPLC_PE_{R, T} LANE_{P, N}[15:0]

Название	Вход/ выход	Тип	Назначение
IPLC_PE_ADAPT_RST_IN	in	CMOS_18	Входной сигнал сброса для адаптации межпроцессорного линка С
IPLC_PE_ADAPT_RST_OUT	out	CMOS_18	Выходной сигнал сброса для адаптации межпроцессорного линка С

Название	Вход/ выход	Тип	Назначение
IPLC_PE_CONFIG[1:0]	in	CMOS_18	Конфигурация канала: IPLC_PE_CONFIG[0] - наличие IP-link2: 0 - нет; 1 - есть; IPLC_PE_CONFIG[1] - ширина IP-link2: 0 - x16; 1 - x8
IPLC_PE_PRE_DET[1:0]	in	CMOS_18	Сигналы Presence Detect, соответствующие подключенному PCIe устройству. Активный уровень - низкий
IPLC_PE_RESREF_F	inout	analog	Подключение на землю калибровочного сопротивления 200 ± 1% Ом, 100-ppm/ºC
IPLC_PE_RLANE_P[15:0] IPLC_PE_RLANE_N[15:0]	in	PCIe	Входные данные. Передаются парафазной линией
IPLC_PE_TLANE_P[15:0] IPLC_PE_TLANE_N[15:0]	out	PCIe	Выходные данные. Передаются парафазной линией

5.10 Каналы SATA/ETH

Возможны следующие конфигурации:

1) 4xSATA

2) 2xSATA +1xETH10G +1xETH1G

3) 2xSATA+2xETH1G

4) 2xSATA+1xETH1G+1xETH2.5G

5) 2xSATA +1x10GBASE-KR+1xSGMII2.5G

6) 2xSATA+2xETH2.5G

В конфигурации с 4x SATA назначение сигналов следующее:

sata0: SATA0_{R,T}}_{P,N};

sata1: SATA1_{R,T}}_{P,N};

sata2: SATA2_ETH0_{R,T}_{P,N}

sata3: SATA3_ETH1_{R,T}_{P,N}

Во всех остальных конфигурациях, с 2xSATA и 2xETH, назначение сигналов следующее:

sata0: SATA0_{R,T}}_{P,N};

sata1: SATA1_{R,T}}_{P,N};

eth10G 0 или eth1G 0: SATA2_ETH0_{R,T}_{P,N}

eth1G 1: SATA3_ETH1_{R, T}_{P,N}

Микропроцессор имеет 2 отдельных канала SATA: 0 - 1. Они имеют одинаковые интерфейсы, и в таблице сигналов символ “k” представляет одну из этих цифр.

Микропроцессор имеет 2 канала SATA: 2-3, разделяемые с каналами ETHERNET. Они имеют одинаковые интерфейсы, и в таблице сигналов символ “m” представляет одну из этих цифр.

Микропроцессор имеет 2 канала ETH: 0 - 1. Они имеют одинаковые интерфейсы (помимо разделяемых с SATA), и в таблице сигналов символ j представляет одну из этих цифр.

Название	Вход/ выход	Тип	Назначение
ETH0_RATE_SELECT	out	CMOS_18	ETH0 module rate select
ETHj_FDUP	in	CMOS_18	ETHj full duplex
ETHj_FETH	in	CMOS_18	ETHj fast ethernet
ETHj_GETH	in	CMOS_18	ETHj gigabit ethernet
ETHj_LINKUP_LED	out	CMOS_18	ETHj link statustransmitter
ETHj_LSTA	in	CMOS_18	ETHj link status
ETHj_MDC	out	CMOS_18	ETHj management clock
ETHj_MDIO	inout	CMOS_18 Внешний pullup	ETHj management data io
ETHj_MOD_ABS	in	CMOS_18	ETHj module absent
ETHj_PHYRST_ N	out	CMOS_18	ETHj phy reset

Название	Вход/ выход	Тип	Назначение
ETHj_RX_LOSS	in	CMOS_18	ETHj receiver loss
ETHj_RXACT_LED	out	CMOS_18	ETHj receiver activity LED
ETHj_TX_DISABLE	out	CMOS_18	ETHj transmitter disable
ETHj_TX_FAULT	in	CMOS_18	ETHj transmitter fault
ETHj_TXACT_LED	out	CMOS_18	ETHj transmitter activity LED
SATA_LED	out	CMOS_18	Контакт для подключения LED SATA. Активный уровень - высокий
SATAETH_CONFIG	in	CMOS_18	Конфигурация канала: 0 - 4x SATA; 1 - одна из оставшихся (задается программно)
SATAETH_RESREF	inout	analog	Подключение калибровочного сопротивления 200 ± 1% Ом, 100-ppm/ºC
SATAk_RX_P SATAk_RX_N	in	PCIe	Входные данные. Передаются парафазной линией
SATAk_TX_P SATAk_TX_N	out	PCIe	Выходные данные. Передаются парафазной линией
SATAm_ETHj_RX_P SATAm_ETHj_RX_N	in	PCIe	Входные данные. Передаются парафазной линией
SATAm_ETHj_TX_P SATAm_ETHj_TX_N	out	PCIe	Выходные данные. Передаются парафазной линией

5.11 Каналы RS-232

Микропроцессор имеет 2 канала RS-232: A, B. Они имеют одинаковые интерфейсы, и в таблице сигналов символ j представляет одну из этих букв.

Название	Вход/ выход	Тип	Назначение
SP_CTSj_N	in	CMOS_18 внешний pullup	Сигнал clear to send. Конечное устройство готово к приему. Активный уровень - низкий

Название	Вход/ выход	Тип	Назначение
SP_DCDj_ N	in	CMOS_18 внешний pullup	Сигнал data carrier detect. Обнаружено другое устройство на линии. Активный уровень - низкий
SP_DTRj_ N	out	CMOS_18	Сигнал data terminal ready. Запрос на прием данных. Активный уровень - низкий. (он же SP_RCQj_ "gp output")
SP_RTSj_ N	out	CMOS_18	Сигнал request to send. Запрос на выдачу данных. Активный уровень - низкий
SP_RXDj	in	CMOS_18 внешний pullup	Входные данные. Активный уровень - высокий
SP_SYNCj_ N	in	CMOS_18 внешний pullup	Синхросигнал
SP_TXDj	out	CMOS_18	Выходные данные. Активный уровень - высокий

5.12 Каналы USB

Микропроцессор имеет 4 порта USB 3.0 (0 – 3) и 4 порта USB 2.0 (0 – 3). Они имеют одинаковые интерфейсы, и в таблице сигналов символ j представляет одну из этих цифр.

Название	Вход/ выход	Тип	Назначение
USB_PORT_OVRCUR_[7:0]	in	CMOS_18 внешний pullup	Сигнал перегрузки по току порта usb. Разряды [3:0] соответствуют портам 3-0 USB 2.0. Разряды [7:4] соответствуют портам 3-0 USB 3.0.

Название	Вход/ выход	Тип	Назначение
			Активный уровень - низкий
USBj_D_P USBj_D_N	inout	USB2.0	Порт данных USB 2.0. Передаются парафазной линией
USBj_ID	in inout	USB2.0 analog	Признак соединителя mini-USB или контакт для аналогового тестирования
USBj_RESREF	inout	analog	Подключение калибровочного сопротивления (200 ± 1%) Ом, 100-ppm/C
USBj_RX_P USBj_RX_N	in	USB3.0	Входные данные USB 3.0. Передаются парафазной линией
USBj_TX_P USBj_TX_N	out	USB3.0	Выходные данные USB 3.0. Передаются парафазной линией
USBj_VBUS	inout	analog	Наличие питания 5,0 В. Подается через сопротивление 30 ± 1% кОм

5.13 Канал I2C Slave для чтения показаний температурных датчиков и управления и чтения показаний схемы управления вентиляторами

Используется внешним контроллером системного менеджмента (IPMI/BMC).

Название	Вход/ выход	Тип	Назначение
MNTR_SCL	inout	CMOS_18 opendrain	Синхросигнал
MNTR_SDA	inout	CMOS_18 opendrain	Адрес/данные

5.14 Сигналы предупреждения о нештатных ситуациях

Название	Вход/ выход	Тип	Назначение

Название	Вход/ выход	Тип	Назначение
CPU_FAULT_N	out	CMOS_18 opendrain	Аварийный сигнал, сообщающий о критических внутренних ошибках икропроцессора. Активный уровень - низкий
CPU_PWR_ALERT_N	in	CMOS_18	Сигнал предупреждения о нештатной ситуации от источников питания микропроцессора (за исключением источников питания 1,2 В для каналов памяти). Активный уровень - низкий
MACHINE_GEN_ALERT_N	in	CMOS_18	Сигнал предупреждения о нештатной ситуации от прочих компонентов машины (система охлаждения и т.д.). Активный уровень - низкий
MACHINE_PWR_ALERT_N	in	CMOS_18	Сигнал предупреждения о нештатной ситуации от AC/DC блока питания машины. Активный уровень - низкий
MC_0_3_DIMM_EVENT_N	in	CMOS_18	Сигнал предупреждения о нештатной ситуации от DIMM-модулей памяти каналов памяти MC0, MC1, MC2, MC3(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Активный уровень - низкий
MC_0_3_PWR_ALERT_N	in	CMOS_18	Сигнал предупреждения о нештатной ситуации от источника питания 1,2 В для каналов памяти MC0, MC1, MC2, MC3(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Активный уровень - низкий
MC_4_7_DIMM_EVENT_N	in	CMOS_18	Сигнал предупреждения о нештатной ситуации от DIMM-модулей памяти каналов

Название	Вход/ выход	Тип	Назначение
			памяти MC4, MC5, MC6, MC7(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Активный уровень - низкий
MC_4_7_PWR_ALERT_N	in	CMOS_18	Сигнал предупреждения о нештатной ситуации от источника питания 1,2 В для каналов памяти MC4, MC5, MC6, MC7(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Активный уровень - низкий
MC0_FAULT_N MC1_FAULT_N MC2_FAULT_N MC3_FAULT_N MC4_FAULT_N MC5_FAULT_N MC6_FAULT_N MC7_FAULT_N	out	CMOS_18 opendrain	Аварийные сигналы, сообщающие о критических ошибках внутри контроллеров и модулей памяти. Активный уровень - низкий (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)
MNTR_ALERTN_ N	out	CMOS_18 opendrain	Сигнал предупреждения о вынужденном снижении системной частоты из-за повышенной температуры микропроцессора. Активный уровень - низкий
MNTR_TTRIPN_ N	out	CMOS_18 opendrain	Аварийный сигнал, сообщающий о повышении температуры микропроцессора до предельно допустимой. Активный уровень - низкий

5.15 Каналы управления вентиляторами

Микропроцессор имеет 2 канала для управления вентиляторами: 0 - 1. Они имеют одинаковые интерфейсы, и в таблице сигналов символ j представляет одну из этих цифр.

Название	Вход/ выход	Тип	Назначение
FAN_ALERTN_[j]_N	out	CMOS_18 opendrain	Нештатная ситуация в FAN-контроллере или в вентиляторе. Активный уровень - низкий
FAN_PWM[j]	out	CMOS_18	ШИМ-сигнал управления скоростью вращения вентилятора. (Обычный push-pull без z-состояния)
FAN_TACH[j]	in	CMOS_18 pulldown	Сигнал обратной связи от вентилятора. Должен подключаться к вентилятору через level shifter

5.16 Калибровка и диагностика PVT-сенсоров

Название	Вход/ выход	Тип	Назначение
AN_IO_TS[1:0]	inout	CMOS_18	Сигнал калибровки и диагностики встроенных термосенсоров
AN_IO_VM[1:0]	inout	CMOS_18	Сигнал диагностики встроенных мониторов питания (вольтметров)

5.17 Цепи питания

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
PWR_0V8_CORE	Основная цепь питания МП	0,8	GND_CORE	U_{CC1}
PWR_0V8_BIO	Цепь питания низкоскоростных интерфейсов	0,8	GND_12G	U_{CC2}
PWR_A0V8_IOWL_PE_0	Цепь аналогового низковольтного питания PHY блока части интерфейса IOWLink/PCIe0 [0:3]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IOWL_PE_1	Цепь аналогового низковольтного питания PHY блока части интерфейса IOWLink/PCIe0 [4:7]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IOWL_PE_2	Цепь аналогового низковольтного питания PHY блока части интерфейса IOWLink/PCIe0 [8:11]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IOWL_PE_3	Цепь аналогового низковольтного питания PHY блока части интерфейса IOWLink/PCIe0 [12:15]	0,8	GND_12G	U_{CC2}

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
PWR_A0V8_IPLA_0	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_A [0:3]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLA_1	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_A [4:7]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLA_2	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_A [8:11]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLA_3	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_A [12:15]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLB_0	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_B [0:3]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLB_1	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_B [4:7]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLB_2	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_B [8:11]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLB_3	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_B [12:15]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLC_PE_0	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_C/PCIe1 [0:3]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLC_PE_1	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_C/PCIe1	0,8	GND_12G	U_{CC2}

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
	[4:7]			
PWR_A0V8_IPLC_PE_2	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_C/PCIe1 [8:11]	0,8	GND_12G	U_{CC2}
PWR_A0V8_IPLC_PE_3	Цепь аналогового низковольтного питания PHY блока части интерфейса IPLink_C/PCIe1 [12:15]	0,8	GND_12G	U_{CC2}
PWR_A0V8_SATA	Цепь аналогового питания физических уровней SATA	0,8	GND_12G	U_{CC2}
PWR_A0V8_SATA_ETH	Цепь аналогового низковольтного питания физических уровней SATA/Ethernet	0,8	GND_12G	U_{CC2}
PWR_A0V8_USB0	Цепь аналогового низковольтного питания физических уровней USB	0,8	GND_12G	U_{CC2}
PWR_A0V8_USB1	Цепь аналогового низковольтного питания физических уровней USB	0,8	GND_12G	U_{CC2}
PWR_A0V8_USB2	Цепь аналогового низковольтного питания физических уровней USB	0,8	GND_12G	U_{CC2}
PWR_A0V8_USB3	Цепь аналогового низковольтного питания физических уровней USB	0,8	GND_12G	U_{CC2}
PWR_1V2_MC_L	Цепь питания выходных каскадов каналов DDR4 0, 1, 2, 3, расположенных с левой стороны (делится на 2. Нечетные каналы заложены для будущего	1,2	GND_MC_L	U_{CC3}

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
	расширения функционала. В текущей итерации МП не поддерживается)			
PWR_1V2_MC_R	Цепь питания выходных каскадов каналов DDR4 4, 5, 6, 7, расположенных с правой стороны (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)	1,2	GND_MC_R	U_{CC4}
PWR_1V8_BIO	Цепь питания выходных каскадов низкоскоростных интерфейсов	1,8	GND_12G	U_{CC5}
PWR_A1V8_ IOWL_PE	Цепь аналогового высоковольтного питания PHY блока части интерфейса IOWLink/PCIe0 [0:15]	1,8	GND_12G	U_{CC5}
PWR_A1V8_ IPLA	Цепь аналогового высоковольтного питания PHY блока части интерфейса IPLink_A [0:15]	1,8	GND_12G	U_{CC5}
PWR_A1V8_ IPLB	Цепь аналогового высоковольтного питания PHY блока части интерфейса IPLink_B [0:15]	1,8	GND_12G	U_{CC5}
PWR_A1V8_ IPLC_PE	Цепь аналогового высоковольтного питания PHY блока части интерфейса IPLink_C/PCIe1 [0:15]	1,8	GND_12G	U_{CC5}
PWR_A1V8_MC03	Цепь аналогового питания	1,8	GND_MC_L	U_{CC5}

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
	блоков синхронизации каналов памяти 0, 1, 2, 3 (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)			
PWR_A1V8_MC47	Цепь аналогового питания блоков синхронизации каналов памяти 4, 5, 6, 7 (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)	1,8	GND_MC_R	U_{CC5}
PWR_A1V8_SATA_ETH	Цепь аналогового высоковольтного питания физических уровней SATA и SATA/Ethernet	1,8	GND_12G	U_{CC5}
PWR_A1V8_SYNC	Цепь аналогового высоковольтного питания блоков синхронизации	1,8	GND_SYNC	U_{CC5}
PWR_A1V8_SYNC_MC_L	Цепь аналогового высоковольтного питания блоков синхронизации	1,8	GND_SYNC_L	U_{CC5}
PWR_A1V8_SYNC_MC_R	Цепь аналогового высоковольтного питания блоков синхронизации	1,8	GND_SYNC_R	U_{CC5}
PWR_A3V3_USB	Цепь аналогового высоковольтного питания физических уровней USB	3,3	GND_12G	U_{CC6}
PWR_0V8_SUS	Цепь питания логики блока Suspend. Вместе с PWR_1V8_SUS включается	0,8	GND_12G	U_{CC7}

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
	первым и снимается последним. Источником питания является неотключаемое питание в соответствии со спецификацией ATX			
PWR_0V8_SUS_DDR01	Цепи питания логики блока DDR4_SUSPEND для каналов памяти 0 и 1 (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)	0,8	GND_MC_L	U_{CC7}
PWR_0V8_SUS_DDR45	Цепи питания логики блока DDR4_SUSPEND для каналов памяти 4 и 5 (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)	0,8	GND_MC_R	U_{CC7}
PWR_1V8_SUS	Цепь питания выходных каскадов блока Suspend. Вместе с PWR_0V8_SUS включается первым и снимается последним. Источником питания является неотключаемое питание в соответствии со спецификацией ATX	1,8	GND_12G	U_{CC8}
PWR_1V8_SUS_DDR01	Цепь питания выходных каскадов блока DDR4 SUSPEND для каналов памяти 0 и 1 (делится на 2. Нечетные каналы заложе-	1,8	GND_MC_L	U_{CC8}

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
	ны для будущего расширения функционала. В текущей итерации МП не поддерживается)			
PWR_1V8_SUS_DDR45	Цепь питания выходных каскадов блока DDR4 SUSPEND для каналов памяти 4 и 5 (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)	1,8	GND_MC_R	U_{CC8}
PWR_0V8_MC_L	Цепь питания логики каналов DDR4 0, 1, 2, 3, расположенных с левой стороны. Допускается подключение к группе U_{CC2}. Допускается подавать повышенное напряжение от 0,855 – 0,9 – 0,945 В (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)	0,8	GND_MC_L	U_{CC9}
PWR_0V8_MC_R	Цепь питания логики каналов DDR4 4, 5, 6, 7, расположенных справой стороны. Допускается подключение к группе U_{CC2}. Допускается подавать повышенное напряжение от 0,855 – 0,9 – 0,945 В (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей ите-	0,8	GND_MC_R	U_{CC10}

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
	рации МП не поддерживается)			
PWR_1V5_HDA	Цепь питания HDA. Допускает работу от номинала 1,8 В и объединение с группой U_{CC5}	1,5	GND_12G	U_{CC11}
PWR_A0V8_SYNC	Цепь аналогового низковольтного питания блоков синхронизации. Допускается подключение к группе U_{CC2}	0,8	GND_SYNC	U_{CC12}
PWR_A0V8_SYNC_MC_L	Цепь аналогового низковольтного питания блоков синхронизации. Допускается подключение к группе U_{CC2}	0,8	GND_SYNC_L	U_{CC12}
PWR_A0V8_SYNC_MC_R	Цепь аналогового низковольтного питания блоков синхронизации. Допускается подключение к группе U_{CC2}	0,8	GND_SYNC_R	U_{CC12}
PWR_1V8_EFUSE_PGM	Цепь питания, которое используется для программирования блока однократно программируемого ПЗУ на кристалле типа eFuse разработки фирмы TSMC. Для программирования необходимо подключить к питанию 1,8 В. Для штатной работы подключить к земле	1,8	GND_12G	U_{CC13}
GND_12G	Цепь земли высокоскоростных каналов и низкоскоростных интерфейсов. Должна быть подключена на общую землю МПП			GND

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
GND_CORE	Цепь земли основного домена питания. Должна быть подключена на общую землю МПП			GND
GND_MC_L	Цепь земли каналов DDR4 № 0, 1, 2, 3, расположенных с левой стороны. Должна быть подключена на общую землю МПП (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)			GND
GND_MC_R	Цепь земли каналов DDR4 4, 5, 6, 7, расположенных с правой стороны. Должна быть подключена на общую землю МПП (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)			GND
GND_SYNC	Цепь земли блоков синхронизации. Должна быть подключена на общую землю МПП			GND
GND_SYNC_L	Цепь земли блоков синхронизации. Должна быть подключена на общую землю МПП			GND
GND_SYNC_R	Цепь земли блоков синхронизации. Должна быть			GND

Наименование цепи питания	Описание	Номинальное значение, В	Опорная земля	Обозначение группы
	подключена на общую землю МПП			
Примечание – Все цепи аналогового питания должны подключаться через LC фильтр				

5.18 Режим переворота сигнальных выводов межпроцессорного канала «А»

Этот режим включается при подаче константы 1 на вывод IPLA_FLIP_EN.Под переворотом сигнальных выводов этого канала понимается определенное изменение функционального назначения этих выводов. При включении режима переворота, функциональное назначение каждого из этих выводов изменяется в соответствии с таблицей 5.18.1. В этой таблице, для каждого из 64 сигнальных выводов межпроцессорного канала «А» указывается его функциональное назначение в режиме переворота. Использование режима переворота позволяет упростить трассировку проводников в многопроцессорных системных платах, что повышает многие характеристики этих печатных плат. Также использование этого режима не оказывает побочного влияния на функционирование соответствующего канала.

Таблица 5.18.1 - Изменение функционального назначения выводов канала А

Название вывода	Назначение в режиме переворота	Название вывода	Назначение в режиме переворота
IPLA_O_P[0]	IPLA_O_N[15]	IPLA_O_N[0]	IPLA_O_P[15]
IPLA_O_P[1]	IPLA_O_N[14]	IPLA_O_N[1]	IPLA_O_P[14]
IPLA_O_P[2]	IPLA_O_N[13]	IPLA_O_N[2]	IPLA_O_P[13]

Название вывода	Назначение в режиме переворота	Название вывода	Назначение в режиме переворота
IPLA_O_P[3]	IPLA_O_N[12]	IPLA_O_N[3]	IPLA_O_P[12]
IPLA_O_P[4]	IPLA_O_N[11]	IPLA_O_N[4]	IPLA_O_P[11]
IPLA_O_P[5]	IPLA_O_N[10]	IPLA_O_N[5]	IPLA_O_P[10]
IPLA_O_P[6]	IPLA_O_N[9]	IPLA_O_N[6]	IPLA_O_P[9]
IPLA_O_P[7]	IPLA_O_N[8]	IPLA_O_N[7]	IPLA_O_P[8]
IPLA_O_P[8]	IPLA_O_N[7]	IPLA_O_N[8]	IPLA_O_P[7]
IPLA_O_P[9]	IPLA_O_N[6]	IPLA_O_N[9]	IPLA_O_P[6]
IPLA_O_P[10]	IPLA_O_N[5]	IPLA_O_N[10]	IPLA_O_P[5]
IPLA_O_P[11]	IPLA_O_N[4]	IPLA_O_N[11]	IPLA_O_P[4]
IPLA_O_P[12]	IPLA_O_N[3]	IPLA_O_N[12]	IPLA_O_P[3]
IPLA_O_P[13]	IPLA_O_N[2]	IPLA_O_N[13]	IPLA_O_P[2]
IPLA_O_P[14]	IPLA_O_N[1]	IPLA_O_N[14]	IPLA_O_P[1]
IPLA_O_P[15]	IPLA_O_N[0]	IPLA_O_N[15]	IPLA_O_P[0]
IPLA_I_P[0]	IPLA_I_N[15]	IPLA_I_N[0]	IPLA_I_P[15]
IPLA_I_P[1]	IPLA_I_N[14]	IPLA_I_N[1]	IPLA_I_P[14]
IPLA_I_P[2]	IPLA_I_N[13]	IPLA_I_N[2]	IPLA_I_P[13]
IPLA_I_P[3]	IPLA_I_N[12]	IPLA_I_N[3]	IPLA_I_P[12]
IPLA_I_P[4]	IPLA_I_N[11]	IPLA_I_N[4]	IPLA_I_P[11]
IPLA_I_P[5]	IPLA_I_N[10]	IPLA_I_N[5]	IPLA_I_P[10]
IPLA_I_P[6]	IPLA_I_N[9]	IPLA_I_N[6]	IPLA_I_P[9]
IPLA_I_P[7]	IPLA_I_N[8]	IPLA_I_N[7]	IPLA_I_P[8]
IPLA_I_P[8]	IPLA_I_N[7]	IPLA_I_N[8]	IPLA_I_P[7]
IPLA_I_P[9]	IPLA_I_N[6]	IPLA_I_N[9]	IPLA_I_P[6]

Название вывода	Назначение в режиме переворота	Название вывода	Назначение в режиме переворота
IPLA_I_P[10]	IPLA_I_N[5]	IPLA_I_N[10]	IPLA_I_P[5]
IPLA_I_P[11]	IPLA_I_N[4]	IPLA_I_N[11]	IPLA_I_P[4]
IPLA_I_P[12]	IPLA_I_N[3]	IPLA_I_N[12]	IPLA_I_P[3]
IPLA_I_P[13]	IPLA_I_N[2]	IPLA_I_N[13]	IPLA_I_P[2]
IPLA_I_P[14]	IPLA_I_N[1]	IPLA_I_N[14]	IPLA_I_P[1]
IPLA_I_P[15]	IPLA_I_N[0]	IPLA_I_N[15]	IPLA_I_P[0]

6 Распределение пространства физических адресов

6.1 Аппаратно различимые области в пространстве физических адресов

В пространстве физических адресов системы может быть выделено несколько семантически различных типов областей. В аппаратуре микропроцессора и межкластерных коммутаторов предусмотрены средства для определения принадлежности адреса запроса той или иной области и направления запроса в тот ресурс системы, который реализует данный адрес.

Аппаратно различаются следующие типы областей.

1 Область программы начальной старта - BOOT. Эта область является единственной в системе и располагается по фиксированным адресам (смотрите 6.2).

2 Системные и служебные регистры узлов - NBSRj (j - номер узла). Пространство для размещения этих областей начинается с фиксированного адреса (смотрите 6.2) и имеет переменный размер, определяемый количеством процессоров в системе. Адреса начала областей NBSRj кратны их размеру, поэтому одноименные адреса разных областей отличаются только значениями определенных разрядов.

3 Оперативная память. Вводятся понятия оперативной памяти верхнего диапазона и нижнего диапазона.

Верхний диапазон описывает всю установленную в системе оперативную память и располагается за совокупным пространством системных регистров вплоть до конца физической памяти. Этот диапазон заполняется непересекающимися областями MHIj (j – номер узла); каждая область описывает память, установленную в j-м узле. В общем случае соседние области MHIj могут лежать несмежно, а каждая из областей может быть не сплошной - это зависит от конфигурации установленных модулей памяти. Если память узла содержит "дыры", это должно учитываться системным ПО с помощью карты доступной памяти. Отоб-

ражение установленных модулей памяти узла в пространство физических адресов должно следовать определенным правилам, описанным в 6.3.1.

Нижний диапазон может располагаться от 0-го адреса до начала области PCI, но не выше 4 Гбайт. Он заполняется произвольным количеством непересекающихся областей MLOj (j - номер узла), каждая из которых должна целиком принадлежать одному узлу. Границы областей MLOj кратны 128 Мбайт. Поскольку вся установленная память изначально принадлежит верхнему диапазону, для наполнения областей MLOj реальной памятью в системе предусмотрены средства перемещения фрагментов памяти из верхнего диапазона в нижний; при этом "дыры", образующиеся в верхнем диапазоне, должны учитываться системным ПО с помощью карты доступной памяти. Переназначение памяти из верхнего диапазона в нижний выполняется по правилам, описанным в 6.3.2.

4 Области MEM на шинах PCI - PCIMj (j - номер узла). Они располагаются по динамически назначаемым непересекающимся адресам в пределах 4 Гбайт (смотрите 6.2). Границы областей кратны 128 Мбайт. Если к узлу не подключено периерийное оборудование, его область PCIMj имеет нулевой размер.

5 Области Prefetchable MEM на шинах PCI - PCIMPj (j - номер узла). Они располагаются по динамически назначаемым непересекающимся адресам (смотрите 6.2). Границы областей кратны 128 Мбайт. Если к узлу не подключено периферийное оборудование, его область PCIMPj имеет нулевой размер.

6 Области IO на шинах PCI - PCIIOj (j - номер узла). Они располагаются по динамически назначаемым непересекающимся адресам в определенном диапазоне (смотрите 6.2). Границы областей кратны 128 Мбайт; размеры равны 4 Кбайт.

7 Области конфигурации на шинах PCI - PCICFGj (j - номер узла). Пространство для размещения этих областей начинается с динамически назначаемого адреса и имеет переменный размер, определяемый количеством КПИ в системе. Адреса начала областей PCICFGj кратны их размеру, поэтому одноименные адреса разных областей отличаются только значениями определенных разрядов.

8 Программно доступные регистры IOAPIC - IOAPICREGj (j - номер узла). Они располагаются по динамически назначаемым непересекающимся адресам в

определенном диапазоне (смотрите 6.2) в пределах 4 Гбайт. Границы областей кратны 4 Кбайт; размеры равны 4 Кбайт.

9 Программно доступные регистры EPIC - EPICREG. Программные регистры EPIC конкретного ядра доступны только своему ядру, поэтому для всех EPIC системы выделяется одинаковый диапазон адресов - EPICREG. Он располагаются по динамически назначаемым адресам в определенном диапазоне (смотрите 6.2) в пределах 4 Гбайт. Границы областей EPICREG кратны 16 Кбайт; размеры равны 16 Кбайт.

10 Память видеокарты - VGAMEM. Эта область является единственной в системе и располагается по фиксированным адресам (смотрите 6.2). Эта область может пересекаться с областями MLOj - в этом случае пересекающиеся фрагменты MLOj теряются.

11 Служебные адреса EPIC, используемые для передачи сообщений из IOAPIC (КПИ-2) в EPIC.

12 Служебные адреса IOAPIC (КПИ-2), используемые для передачи сообщений из EPIC в IOAPIC (КПИ-2).

13 Область для передачи сообщений типа MSI из IOEPIC в CEPIC.

6.2 Карта физической памяти

Примечание - Многие из перечисленных в 6.1 областей могут быть размещены в произвольном месте физической памяти, поэтому приведенное ниже распределение является лишь примером.

Адреса	Наименование областей
00_0000_0000 00_xxFF_FFFF	Нижний диапазон оперативной памяти - MLO; верхняя граница диапазона MLO назначается динамически системным ПО и не превышает нижней границы областей PCIMj; границы областей MLOj кратны 128 Мбайт
00_000A_0000 00_000B_FFFF	Память видеокарты - VGAMEM; может пересекаться с диапазоном MLO - в этом случае пересекающиеся фрагменты MLO теряются
00_xx00 0000 00_FEBF_FFFF	Области MEM на шинах PCI - PCIMj; нижняя граница областей PCIMj назначается динамически системным ПО и не может быть меньше верхней

Адреса	Наименование областей
	границы диапазона MLO; границы областей PCIMj кратны 128 Мбайт
00_FEC0_0000 00_FED0_FFFF	Программно доступные регистры IOAPIC - IOAPICREGj; границы областей IOAPICREGj кратны 4 Кбайт; размеры равны 4 Кбайт.
00_FED1_0000 00_FFDF_FFFF	Резерв
00_FEE0_0000 00_FEF0_FFFF	Программно доступные регистры EPIC - EPICREG; границы областей EPICREG кратны 4 Кбайт; размеры равны 4 Кбайт.
00_FEE0_0000 00_FEE0_0FFF	Программно доступные регистры CEPIC; обращения в режиме (GM==0) попадают в комплект регистров хоста; обращения в режиме (GM==1) попадают в комплект регистров гостя
00_FEE0_1000 00_FEE0_1FFF	Адреса для обращения гипервизора (GM==0) к комплекту регистров CEPIC гостя;
00_FEE0_2000 00_FEE0_2FFF	Программно доступные регистры PREPIC - PREPICREG
00_FEE0_4000 00_FFFF_FFFF	Резерв
01_0000_0000 01_00FF_FFFF	Область программы начального старта - BOOT
01_0100_0000 01_0100_FFFF	Области IO на шинах PCI - PCIIOj; границы областей PCIIOj кратны 4 Кбайт
01_1000_0000 01_100F_FFFF	Программно доступные регистры узла #0 - NBSR0. Адреса начала и конца области j вычисляется как: Abgn = 0x01_1000_0000 + j*0x10_0000; Aend = Abgn + 1MB - 1;
----- 	-------
01_1xx0_0000 01_1xxF_FFFF	Программно доступные регистры узла #j - NSRj. Адреса начала и конца области j вычисляется как Abgn = 0x01_1000_0000 + j*0x10_0000; Aend = Abgn + 1 Мбайт - 1;
01_2000_0000 01_200F_FFFF	Область для передачи сообщений типа MSI из IOEPIC в CEPIC

Адреса	Наименование областей
01_2100_0000 01_21FF_FFFF	Область служебных адресов EPIC, используемых для передачи сообщений из IOAPIC (КПИ-2) в EPIC. Примечание - По умолчанию адресный регистр IOAPIC принимает нулевое значение, поэтому ПО обязано его инициализировать.
01_2200_0000 01_22FF_FFFF	Области служебных адресов IOAPIC, используемых для сообщений в с- Область служебных адресов IOAPIC, используемых для передачи сообщений из EPIC в IOAPIC (КПИ-2)
01_2300_0000 01_FFFF_FFFF	Резерв

02_0000_0000
0x_xFFF_FFFF
	- Области конфигурационных регистров PCI - PCICFG. Для каждого кластера выделяется смежная область, включающая подобласти для всех КПИ кластера. Начало области определяется регистром RT_PCICFGB.
	Адреса начала и конца области для КПИ #j вычисляется как:
	Abgn = RT_PCICFGB + j*0x1000_0000;
	Aend = Abgn + j*0x100_0000;
	Aend = Abgn + 256 Мбайт - 1

0x_x000 0000 yy_yyFF_FFFF	Области MEMP на шинах PCI - PCIMPj; нижняя граница областей PCIMj назначается динамически системным ПО; границы областей PCIMPj кратны 128 MB.
z0_0000_0000 FF_FFFF_FFFF	- Верхний диапазон оперативной памяти - MHI. Адрес начала диапазона зависит от размера диапазона PCIMP и назначается системным ПО; границы областей MHIj кратны 4 GB

6.3 Правила отображения оперативной памяти в пространство физических адресов

6.3.1 Мэппирование оперативной памяти верхнего диапазона

Подсистема оперативной памяти узла имеет K каналов, каждый из которых обслуживает по два разъема (m=0,1) для установки модулей памяти емкостью от 512 Мбайт до 128 Гбайт, причем в одном канале допускается установка разнотип-

ных модулей в произвольном сочетании (при условии, что суммарный объем памяти канала не превышает 64 Гбайт). Если в канале используются модули памяти разного объема, модуль большего объема нужно устанавливать в слот 0.

В подсистеме памяти предусмотрены управляющие регистры, позволяющие системному ПО задавать распределение (интерливинг) физических адресов по каналам. Поле регистра L3_L3IL.l3il_bit2 задает номер разряда физического адреса, значение которого определяет пару каналов:

0 - 0-й или 2-й;

1 - 1-й или 3-й.

Указанный разряд изымается из адреса, а более старшие разряды сдвигаются на его место. После этого поле регистра SIC_MIC.mcil задает номер разряда (модифицированного) физического адреса, значение которого определяет канал в паре:

0 - младший;

1 – старший.

В зависимости от соотношения настроек интерливинга и объема установленной памяти различают 4 адресных режима, описанных ниже; выбор режима определяет дополнительные ограничения и правила отображения установленной памяти. Однако, размер и расположение суммарного диапазона физических адресов вычисляется одинаково.

Пусть суммарная емкость памяти в k-м канале равна S_ch[k]. Пусть наибольшее из них равно s:

s = max(S_ch[k]), для всех k;

Пусть ближайшее число, равное или большее s и являющееся степенью двойки, равно S, а подходящая степень двойки равна N.

Тогда суммарный диапазон адресов, занимаемых узлом j, имеет размер

S_node[j] = K * S;

Область памяти узла размещается в пространстве физических адресов так, чтобы ее базовый адрес был кратен как минимум 4 Гбайт; если S_node[j] превышает 4 Гбайт, базовый адрес должен быть кратен S_node[j].

Если (S_ch[k] < S), системное ПО должно гарантировать неиспользование отсутствующих физических адресов.

6.3.1.1 Описание адресных режимов

1 Режим последовательного прохождения по каналам.

В этом режиме разряды, управляющие интерливингом, должны находиться за пределами разрядной сетки, необходимой для адресации установленной в канал памяти. В этом случае не возникает никаких дополнительных ограничений на конфигурацию устанавливаемых модулей памяти - конфигурации каналов не зависят друг от друга, а в каналы можно устанавливать разнотипные модули в произвольном сочетании или не устанавливать вообще. Системное ПО должно следовать следующим правилам отображения установленной памяти в пространство физических адресов.

Регистр L3_L3IL.l3il_bit2 должен кодировать разряд N); с учетом модификации физического адреса регистр SIC_MIC.mcil тоже должен кодировать разряд N. Каждому каналу последовательно отводится диапазон адресов равный S. Если (S_ch[k] < S), системное ПО должно гарантировать неиспользование отсутствующих физических адресов.

Профиль наличных адресов определяется последовательным рассмотрением памяти в каналах; порядок каналов следующий: 0, 1, 2, 3. В рамках канального диапазона адресов модули располагаются последовательно, сначала модуль слота 0, затем слота 1. При этом начальный адрес модуля должен быть кратен его объему.

2 Режим чередования адресов между парами каналов при последовательном расположении в паре.

Этот режим реализуется, когда номер адресного разряда, по которому выбирается пара каналов (регистр L3_L3IL.l3il_bit2), находится в пределах разрядной сетки, необходимой для адресации установленной в паре каналов памяти, а регистр SIC_MIC.mcil кодирует разряд N). В этом режиме на конфигурацию устанавливаемых модулей памяти накладываются следующие ограничения:

a) суммарный объем памяти в разных парах каналов должен совпадать;

b) если в одной паре каналов используются модули разной емкости,

конфигурация модулей в всех парах должна совпадать.

Если (S_ch[k] < S), системное ПО должно гарантировать неиспользование отсутствующих физических адресов. Профиль наличных адресов определяется последовательным рассмотрением памяти в одной (любой) из пар каналов (порядок каналов в паре: 0, 2; порядок слотов в канале: 0, 1). При этом объем найденных модулей нужно увеличивать вдвое; начальный адрес диапазона адресов модуля также должен быть выровнен на его удвоенный размер.

3 Режим чередования адресов между каналами в паре при последовательном расположении пар.

Этот режим реализуется, когда номер адресного разряда, по которому выбирается канал в паре (регистр SIC_MIC.mcil), находится в пределах разрядной сетки, необходимой для адресации установленной в канале памяти, а регистр L3_L3IL.l3il_bit2 кодирует разряд N+1). В этом режиме на конфигурацию устанавливаемых модулей памяти накладываются следующие ограничения:

a) суммарный объем памяти в разных каналах пары должен совпадать;

b) если в канале используются модули разной емкости, конфигурация модулей в обоих каналах пары должна совпадать;

Если (S_ch[k] < S), системное ПО должно гарантировать неиспользование отсутствующих физических адресов. Профиль наличных адресов определяется последовательным рассмотрением памяти в одном (любом) канале из каждой пары (порядок каналов: 0, 1; порядок слотов в канале: 0, 1). При этом объем найденных модулей нужно увеличивать вдвое; начальный адрес диапазона адресов модуля также должен быть выровнен на его удвоенный размер.

4 Режим полного чередования адресов.

Этот режим реализуется, когда и номер адресного разряда, по которому выбирается пара каналов (регистр L3_L3IL.l3il_bit2), и номер адресного разряда, по которому выбирается канал в паре (регистр SIC_MIC.mcil), меньше значения N) для данного объема памяти (смотрите п.1)). В этом режиме на конфигурацию устанавливаемых модулей памяти накладываются следующие ограничения:

a) суммарный объем памяти во всех каналах должен совпадать;

b) если в одном канале используются модули разной емкости, конфигурация модулей во всех каналах должна совпадать;

Если (S_ch[k] < S), системное ПО должно гарантировать неиспользование отсутствующих физических адресов. Профиль наличных адресов определяется последовательным рассмотрением памяти в одном (любом) канале (порядок слотов в канале: 0, 1). При этом объем найденных модулей нужно увеличивать вчетверо; начальный адрес диапазона адресов модуля также должен быть выровнен на его учетверенный размер.

6.3.2 Мэппирование оперативной памяти нижнего диапазона

Определено, что вся установленная в узле память изначально принадлежит верхнему диапазону, поэтому для наполнения областей MLOj реальной памятью в системе предусмотрены средства перемещения фрагментов памяти из верхнего диапазона в нижний. Перемещение выполняется по следующим правилам.

Пусть размер назначенной для узла j области верхнего диапазона равен S_node[j], и базовый адрес кратен как минимум 4 Гбайт (смотрите 6.3.1). Тогда разрешенными кандидатами на перемещение в нижний диапазон являются адреса, принадлежащие первому 4-гигабайтному участку. Порождаемый адрес нижнего диапазона (Ahi2lo) равен смещению (offset) адреса в верхнем диапазоне (Ahi) относительно его начала (Abgn):

Ahi = Abgn + offset;

Ahi2lo = offset;

Будучи перенесен в нижний диапазон, адрес Ahi должен быть изъят из карты памяти верхнего диапазона.

Примечания

1 Системное ПО должно учитывать, что при определенном расположении начала области MLOj адреса Ahi2lo могут "вырезать дыру" в верхнем диапазоне таким образом, что останется несмежный фрагмент MHIj. Например, пусть требуемый MLOj имеет размер 384 Мбайт (128 Мбайт*3) и начинается с адреса 128 Мбайт. Тогда в первых 4-х гигабайтах области MHIj будут "вырезаны" 1-й, 2-й и 3-й сегменты по 128 Мбайт, а 0-й сегмент останется в верхнем диапазоне отдельно от остальной памяти. Карта памяти узла становится "дырявой".

2 Системное ПО должно учитывать, что при размере MLOj, сравнимом с объемом памяти узла, не всякое расположение области MLOj корректно. Например, пусть в узле установлена память емкостью 1 Гбайт и она "прижата" к началу MHIj. Пусть требуемый MLOj имеет размер 0,5 Гбайт, но начинается с адреса 1 Гбайт. Тогда адреса-жертвы Ahi (и Ahi2lo) попадают в участок, который выходят за границы наличной памяти.

7 Эксплуатационные ограничения

7.1 Электрические параметры и режимы эксплуатации

Электрические параметры микросхем приведены в таблице 7.1.

Таблица 7.1.1 - Электрические параметры микросхем

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды, °С	Номер пункта примечания
		не менее	не более		
1 Выходное напряжение низкого уровня интерфейса управления и диагностики, В (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,2 В; U_{CC4} = 1,2 В; U_{CC5} = 1,7 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,7 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В; U_{CC12} = 0,8 В; I_{OL1} = 2,0 мА)	U_{OL1}	–	0,45	От минус 40 до плюс 90*	
2 Выходное напряжение высокого уровня интерфейса управления и диагностики, В (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,2 В; U_{CC4} = 1,2 В; U_{CC5} = 1,7 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,7 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В; U_{CC12} = 0,8 В; I_{OH1} = -2,0 мА)	U_{OH1}	1,30	–	От минус 40 до плюс 90*	
3 Ток утечки на входе интерфейса управления и диагностики, мкА, (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,2 В; U_{CC4} = 1,2 В; U_{CC5} = 1,9 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,9 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В; U_{CC12} = 0,8 В; и U_{IH1}=1,9 В; U_{IL1}=0 В)	I_{IL}	минус 10	10	От минус 40 до плюс 90*	1
		минус 300	300		2
4 Выходное напряжение низкого уровня интерфейса данных и адреса каналов памяти, В (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,14 В; U_{CC4} = 1,14 В; U_{CC5} = 1,8 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,8 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В; U_{CC12} = 0,8 В; I_{OL2} = 5,0 мА)	U_{OL2}	–	0,40	От минус 40 до плюс 90*	

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды, °С	Номер пункта примечания
		не менее	не более		
5 Выходное напряжение высокого уровня интерфейса данных и адреса каналов памяти, В (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,14 В; U_{CC4} = 1,14 В; U_{CC5} = 1,8 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,8 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В, U_{CC12} = 0,8 В; I_{OH2} = -5,0 мА)	U_{OH2}	0,74	–	От минус 40 до плюс 90*	
6 Ток утечки на входе-выходе в состоянии «Выключено» интерфейса каналов памяти, мкА (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,26 В; U_{CC4} = 1,26 В; U_{CC5} = 1,8 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,8 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В; U_{CC12} = 0,8 В; и U_{IH2} = 1,26 В; U_{IL2} = 0 В)	I_{ILZ}	минус 20	20	От минус 40 до плюс 90*	
7 Выходное напряжение низкого уровня интерфейса HDA, В (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,20 В; U_{CC4} = 1,20 В; U_{CC5} = 1,8 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,8 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,4 В; U_{CC12} = 0,8 В; I_{OL3} = 1,5 мА)	U_{OL3}	–	0,45	От минус 40 до плюс 90*	
8 Выходное напряжение высокого уровня интерфейса HDA, В (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,20 В; U_{CC4} = 1,20 В; U_{CC5} = 1,8 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,8 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,4 В; U_{CC12} = 0,8 В; I_{OH3} = 0,5 мА)	U_{OH3}	1,30	–	От минус 40 до плюс 90*	
9 Ток потребления статический от источников питания U_{CC1}, U_{CC2}, U_{CC7}, U_{CC9}, U_{CC10} и U_{CC12}, А (при U_{CC1}= 0,85 В; U_{CC2}= 0,85 В; U_{CC3} = 1,20 В; U_{CC4} = 1,20 В; U_{CC5} = 1,8 В; U_{CC6} = 3,3 В; U_{CC7} = 0,85 В; U_{CC8}= 1,8 В; U_{CC9}	I_{CC1}+ I_{CC2}+ +I_{CC7} + I_{CC9}+ +I_{CC10} + +I_{CC12}	–	12,0	От минус 40 до плюс 90*	

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды, °С	Номер пункта примечания
		не менее	не более		
= 0,85 В; U_{CC10} = 0,85 В; U_{CC11} = 1,5 В; U_{CC12} = 0,85 В)					
10 Ток потребления статический от источников питания U_{CC3} и U_{CC4}, А (при U_{CC1}= 0,80 В; U_{CC2}= 0,80 В; U_{CC3} = 1,26 В; U_{CC4} = 1,26 В; U_{CC5} = 1,8 В; U_{CC6} = 3,3 В; U_{CC7} = 0,80 В; U_{CC8}= 1,8 В; U_{CC9} = 0,80 В; U_{CC10} = 0,80 В; U_{CC11} = 1,5 В; U_{CC12} = 0,80 В)	I_{CC3}+ I_{CC4}	–	3,0	От минус 40 до плюс 90*	
11 Ток потребления статический от источника питания U_{CC5}, и U_{CC8}, А (при U_{CC1}= 0,80 В; U_{CC2}= 0,80 В; U_{CC3} = 1,20 В; U_{CC4} = 1,20 В; U_{CC5} = 1,9 В; U_{CC6} = 3,3 В; U_{CC7} = 0,80 В; U_{CC8}= 1,9 В; U_{CC9} = 0,80 В; U_{CC10} = 0,80 В; U_{CC11} = 1,5 В; U_{CC12} = 0,80 В)	I_{CC5}+ I_{CC8}	–	2,0	От минус 40 до плюс 90*	
12 Ток потребления статический от источника питания U_{CC6}, А (при U_{CC1}= 0,80 В; U_{CC2}= 0,80 В; U_{CC3} = 1,20 В; U_{CC4} = 1,20 В; U_{CC5} = 1,8 В; U_{CC6} = 3,6 В; U_{CC7} = 0,80 В; U_{CC8}= 1,8 В; U_{CC9} = 0,80 В; U_{CC10} = 0,80 В; U_{CC11} = 1,5 В; U_{CC12} = 0,80 В)	I_{CC6}	–	0,5	От минус 40 до плюс 90*	
13 Ток потребления статический от источника питания U_{CC11}, А (при U_{CC1}= 0,80 В; U_{CC2}= 0,80 В; U_{CC3} = 1,20 В; U_{CC4} = 1,20 В; U_{CC5} = 1,8 В; U_{CC6} = 3,3 В; U_{CC7} = 0,80 В; U_{CC8}= 1,8 В; U_{CC9} = 0,80 В; U_{CC10} = 0,80 В; U_{CC11} = 1,6 В; U_{CC12} = 0,80 В)	I_{CC11}	–	0,5	От минус 40 до плюс 90*	
14 Ток потребления при проверке коротких замыканий от источников питания U_{CC1}, U_{CC2}, U_{CC7}, U_{CC9}, U_{CC10} и U_{CC12}, мА (при нормальных климатических условиях, U_{CC1}= 0,3 В; U_{CC2}= 0,3 В; U_{CC3} = 0,3 В; U_{CC4} = 0,3 В; U_{CC5} = 0,3 В; U_{CC6}	I_{CCS1}+ I_{CCS2}+ +I_{CCS7}+ +I_{CCS9}+ + I_{CCS10}+ + I_{CCS12}	–	300	25	

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды, °С	Номер пункта примечания
		не менее	не более		
= 0,3 В; U_{CC7} = 0,3 В; U_{CC8} = 0,3 В; U_{CC9} = 0,3 В; U_{CC10} = 0,3 В; U_{CC11} = 0,3 В; U_{CC12} = 0,3 В)					
15 Ток потребления при проверке коротких замыканий от источников питания U_{CC3} и U_{CC4}, мА (при нормальных климатических условиях, U_{CC1}= 0,3 В; U_{CC2}= 0,3 В; U_{CC3} = 0,3 В; U_{CC4} = 0,3 В; U_{CC5} = 0,3 В; U_{CC6} = 0,3 В; U_{CC7} = 0,3 В; U_{CC8} = 0,3 В; U_{CC9} = 0,3 В; U_{CC10} = 0,3 В; U_{CC11} = 0,3 В; U_{CC12} = 0,3 В)	I_{CCS3} + I_{CCS4}	–	30	25	
16 Ток потребления при проверке коротких замыканий от источника питания U_{CC5}, и U_{CC8}, мА (при нормальных климатических условиях, U_{CC1}= 0,3 В; U_{CC2}= 0,3 В; U_{CC3} = 0,3 В; U_{CC4} = 0,3 В; U_{CC5} = 0,3 В; U_{CC6} = 0,3 В; U_{CC7} = 0,3 В; U_{CC8} = 0,3 В; U_{CC9} = 0,3 В; U_{CC10} = 0,3 В; U_{CC11} = 0,3 В; U_{CC12} = 0,3 В)	I_{CCS5} + I_{CCS8}	–	3	25	
17 Ток потребления при проверке коротких замыканий от источника питания U_{CC6}, мА (при нормальных климатических условиях, U_{CC1}= 0,3 В; U_{CC2}= 0,3 В; U_{CC3} = 0,3 В; U_{CC4} = 0,3 В; U_{CC5} = 0,3 В; U_{CC6} = 0,3 В; U_{CC7} = 0,3 В; U_{CC8} = 0,3 В; U_{CC9} = 0,3 В; U_{CC10} = 0,3 В; U_{CC11} = 0,3 В; U_{CC12} = 0,3 В)	I_{CCS6}	–	3	25	
18 Ток потребления при проверке коротких замыканий от источника питания U_{CC11}, мА (U_{CC1}= 0,3 В; U_{CC2}= 0,3 В; U_{CC3} = 0,3 В; U_{CC4} = 0,3 В; U_{CC5} = 0,3 В; U_{CC6} = 0,3 В; U_{CC7} = 0,3 В; U_{CC8} = 0,3 В; U_{CC9} = 0,3 В; U_{CC10} = 0,3 В; U_{CC11} = 0,3 В; U_{CC12} = 0,3 В)	I_{CCS11}	–	3	25	
19 Выходное дифференциальное напряжение интерфейса синхронизации и сигналов DQS каналов памяти, В	U_{OD1}	0,60	1,14	От минус 40 до	

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды, °С	Номер пункта примечания
		не менее	не более		
(при U_{CC1}= 0,80 В U_{CC2}= 0,80 В; U_{CC3} = 1,14 В; U_{CC4} = 1,14 В; U_{CC5} = 1,8 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,8 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В; U_{CC12} = 0,8 В; I_{O2} = 5,00 мА)				плюс 90*	
20 Выходное дифференциальное напряжение интерфейсов межпроцессорных каналов, канала ввода-вывода, PCIe, Ethernet и SATA В (U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,2 В; U_{CC4} = 1,2 В; U_{CC5} = 1,7 В; U_{CC6} = 3,3 В; U_{CC7} = 0,8 В; U_{CC8}= 1,7 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В; U_{CC12} = 0,8 В; R_{L1} =100 Ом)	U_{OD2}	0,40	0,80	От минус 40 до плюс 90*	
21 Выходное напряжение низкого уровня интерфейса USB (full speed режим), В (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,2 В; U_{CC4} = 1,2 В; U_{CC5} = 1,8 В; U_{CC6} = 3,0 В; U_{CC7} = 0,8 В; U_{CC8}= 1,8 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В; U_{CC12} = 0,8 В; R_{L2} = 1,5 кОм)	U_{OL4}		0,3	От минус 40 до плюс 90*	
22 Выходное напряжение высокого уровня интерфейса USB (full speed режим), В (при U_{CC1}= 0,8 В; U_{CC2}= 0,8 В; U_{CC3} = 1,2 В; U_{CC4} = 1,2 В; U_{CC5} = 1,8 В; U_{CC6} = 3,0 В; U_{CC7} = 0,8 В; U_{CC8}= 1,8 В; U_{CC9} = 0,8 В; U_{CC10} = 0,8 В; U_{CC11} = 1,5 В; U_{CC12} = 0,8 В; R_{L2} = 1,5 кОм)	U_{OH4}	2,80		От минус 40 до плюс 90*	

*Температура корпуса.

Примечания

1 В случаях, когда уровень входного сигнала U_{IL1} = 0 В или U_{IH1} = 1,90 В совпадает с уровнем подключения сопротивления на входе к 0 В или 1,90 В соответственно.

2 В случаях, когда уровень входного сигнала U_{IH1} = 1,90 В или U_{IL1} = 0 В не совпадает с уровнем подключения сопротивления на входе к 0 В или 1,90 В соответственно (см. ТВГИ.431281.028ТБ)

Номинальное значение напряжения питания микросхем:

U_{CC1} = 0,80 В для домена CORE;

U_{CC2} = 0,80 В для домена UNCORE и низковольтного питания PHY блоков периферии;

U_{CC3} = 1,20 В для IO ячеек каналов памяти MC0 - MC3(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);

U_{CC4} = 1,20 В для IO ячеек каналов памяти MC4 – MC7(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);

U_{CC5} = 1,80 В для интерфейсов управления и диагностики, LVDS и высоковольтного питания PHY блоков периферии (кроме USB);

U_{CC6} = 3,30 В для высоковольтного питания PHY блоков интерфейса USB;

U_{CC7} = 0,80 В для питания внутренних схем блока Suspend;

U_{CC8} = 1,80 В для питания выходных каскадов блока Suspend;

U_{CC9} = 0,80 В для питания внутренних схем каналов памяти MC0 - MC3(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);

U_{CC10} = 0,80 В для питания внутренних схем каналов памяти MC4 – MC7(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);

U_{CC11} = 1,50 В для интерфейса HDA;

U_{CC12} = 0,80 В для питания блоков синхронизации;

U_{CC13} = 1,80 В для питания при программировании блока ПЗУ eFuse.

Допустимые отклонения напряжений питания от номинальных должны быть не более:

±0,05 В для U_{CC1}, U_{CC2}, U_{CC7}, U_{CC9}, U_{CC10}, U_{CC12};

±0,06 В для U_{CC3} и U_{CC4};

±0,10 В для U_{CC5}, U_{CC8} и U_{CC13};

±0,30 В для U_{CC6};

±0,10 В для U_{CC11}.

Значения предельно допустимых и предельных режимов эксплуатации в диапазоне рабочих температур среды соответствуют нормам, приведенным в таблице 7.2.

Таблица 7.1.2 – Предельно допустимые и предельные режимы эксплуатации

Наименование параметра, режима, единица измерения	Буквенное обозначение параметра	Предельно допустимый режим		Предельный режим	
		не менее	не более	не менее	не более
1 Напряжение питания домена CORE, В	U_{CC1}	0,75	0,85	-0,2	1,1
2 Напряжение питания домена UNCORE, В	U_{CC2}	0,75	0,85	-0,2	1,1
3 Напряжение питания интерфейса каналов памяти, В	U_{CC3}, U_{CC4}	1,14	1,26	-0,3	1,5
4 Напряжение питания интерфейсов управления и диагностики, LVDS и PHY блоков периферии (кроме USB), В	U_{CC5}	1,7	1,9	-0,3	2,0
5 Напряжение питания PHY блоков интерфейса USB, В	U_{CC6}	3,0	3,6	-0,3	4,2
6 Напряжение питания внутренних схем Блока Suspend, В	U_{CC7}	0,75	0,85	-0,2	1,1
7 Напряжение питания выходных каскадов блока Suspend, В	U_{CC8}	1,7	1,9	-0,3	2,0
8 Напряжение питания внутренних схем каналов памяти, В	U_{CC9}, U_{CC10}	0,75	0,85	-0,2	1,1
9 Напряжение питания интерфейса HDA, В	U_{CC11}	1,4	1,6	-0,3	1,8
10 Напряжение питания внутренних схем блоков синхронизации, В	U_{CC12}	0,75	0,85	-0,2	1,1
11 Напряжение питания при программировании блока ПЗУ eFuse, В	U_{CC13}	1,7	1,9	-0,3	2,0
12 Входное напряжение высокого уровня интерфейса управления и диагностики, В	U_{IH1}	1,2	2,0	-0,3	U_{CC7} + 0,3
13 Входное напряжение низкого уровня интерфейса управления и диагностики, В	U_{IL1}	-0,3	0,6	-0,3	U_{CC7} + 0,3
14 Входное дифференциальное напряжение интерфейса LVDS, В	U_{ID1}	0,2	0,5	-0,3	U_{CC7} + 0,3
15 Входное дифференциальное напряжение интерфейса данных и адреса	U_{ID2}				

Наименование параметра, режима, единица измерения	Буквенное обозначение параметра	Предельно допустимый режим		Предельный режим	
		не менее	не более	не менее	не более
каналов памяти, В		0,28	–	–	–
16 Входное дифференциальное напряжение сигналов DQS интерфейса каналов памяти, В	U_{ID2}	0,36	1,20	-0,3	U_{CC7} + 0,3
17 Входное дифференциальное напряжение интерфейса межпроцессорных каналов, канала ввода-вывода, PCIe, Ethernet, SATA, В	U_{ID3}	0,15	0,60	-0,3	U_{CC7} + 0,3
18 Выходной ток интерфейса управления и диагностики, мА	I_{O1}	-8	8	–	–
19 Выходной ток интерфейса каналов памяти, мА	I_{O2}	- 10	20	–	–
20 Рабочая частота системного синхросигнала, МГц	f_C	99,98	100,02	–	–
21 Емкость нагрузки каждого выхода интерфейсов, кроме JTAG, пФ	C_{L1}	–	10	–	–
22 Емкость нагрузки каждого выхода интерфейса JTAG, пФ	C_{L2}	–	35	–	–

Таблица 7.1.3 - Предельно допустимые и предельные режимы источников питания

Наименование цепи питания микросхемы	Не менее, В	Номинал, В	Не более, В	Максимальный ток в цепи, мА	Максмальный ток по группе, А
1 PWR_0V8_CORE		0,8		130000	130,00
2 PWR_0V8_BIO	0,72	0,8	0,88		7,54
3 PWR_A0V8_IOWL_PE_0	0,744	0,8	0,935	435,43	
4 PWR_A0V8_IOWL_PE_1	0,744	0,8	0,935	435,43	
5 PWR_A0V8_IOWL_PE_2	0,744	0,8	0,935	435,43	
6 PWR_A0V8_IOWL_PE_3	0,744	0,8	0,935	435,43	
7 PWR_A0V8_IPLA_0	0,744	0,8	0,935	435,43	
8 PWR_A0V8_IPLA_1	0,744	0,8	0,935	435,43	
9 PWR_A0V8_IPLA_2	0,744	0,8	0,935	435,43	
10 PWR_A0V8_IPLA_3	0,744	0,8	0,935	435,43	
11 PWR_A0V8_IPLB_0	0,744	0,8	0,935	435,43	

Наименование цепи питания микросхемы	Не менее, В	Номинал, В	Не более, В	Максимальный ток в цепи, мА	Максмальный ток по группе, А
12 PWR_A0V8_IPLB_1	0,744	0,8	0,935	435,43	
13 PWR_A0V8_IPLB_2	0,744	0,8	0,935	435,43	
14 PWR_A0V8_IPLB_3	0,744	0,8	0,935	435,43	
15 PWR_A0V8_IPLC_PE_0	0,744	0,8	0,935	435,43	
16 PWR_A0V8_IPLC_PE_1	0,744	0,8	0,935	435,43	
17 PWR_A0V8_IPLC_PE_2	0,744	0,8	0,935	435,43	
18 PWR_A0V8_IPLC_PE_3	0,744	0,8	0,935	435,43	
19 PWR_A0V8_SATA	0,744	0,8	0,935	136,48	
20 PWR_A0V8_SATA_ETH	0,744	0,8	0,935	201,31	
21 PWR_A0V8_USB0	0,744	0,8	0,935	58,1	
22 PWR_A0V8_USB1	0,744	0,8	0,935	58,1	
23 PWR_A0V8_USB2	0,744	0,8	0,935	58,1	
24 PWR_A0V8_USB3	0,744	0,8	0,935	58,1	
25 PWR_1V2_MC_L	1,14	1,2	1,26	15600	15,60
26 PWR_1V2_MC_R	1,14	1,2	1,26	15600	15,60
27 PWR_1V8_BIO	1,62	1,8	1,98		2,80
28 PWR_A1V8_IOWL_PE	1,395	1,8	1,98	544,56	
29 PWR_A1V8_IPLA	1,395	1,8	1,98	544,56	
30 PWR_A1V8_IPLB	1,395	1,8	1,98	544,56	
31 PWR_A1V8_IPLC_PE	1,395	1,8	1,98	544,56	
32 PWR_A1V8_MC03	1,71	1,8	1,89	120	
33 PWR_A1V8_MC47	1,71	1,8	1,89	120	
34 PWR_A1V8_SATA_ETH	1,395	1,8	1,98	120,8	
35 PWR_A1V8_SYNC	1,7	1,8	1,9	112,75	
36 PWR_A1V8_SYNC_MC_L	1,7	1,8	1,9	80,25	
37 PWR_A1V8_SYNC_MC_R	1,7	1,8	1,9	64	
38 PWR_A3V3_USB	3,069	3,3	3,63	230,864	0,23
39 PWR_0V8_SUS	0,72	0,8	0,88		0,00
40 PWR_0V8_SUS_DDR01	0,72	0,8	0,88		
41 PWR_0V8_SUS_DDR45	0,72	0,8	0,88		
42 PWR_1V8_SUS	1,62	1,8	1,98		0,00
43 PWR_1V8_SUS_DDR01	1,62	1,8	1,98		
44 PWR_1V8_SUS_DDR45	1,62	1,8	1,98		
45 PWR_0V8_MC_L	0,76	0,8	0,84	8400	8,40
46 PWR_0V8_MC_R	0,76	0,8	0,84	8400	8,40
47 PWR_1V5_HDA		1,5	1,8		0,00
48 PWR_A0V8_SYNC	0,72	0,8	0,88	131,97	0,44
49 PWR_A0V8_SYNC_MC_L	0,72	0,8	0,88	181,83	
50 PWR_A0V8_SYNC_MC_R	0,72	0,8	0,88	123,27	
51 PWR_1V8_EFUSE_PGM	1,62	1,8	1,98	140	0,14

Таблица 7.1.4 – Подключение цепей земли микросхемы

Цепи земли микросхемы	Цепь земли МПП
GND_12G	GND
GND_CORE	GND
GND_MC_L	GND
GND_MC_R	GND
GND_SYNC	GND
GND_SYNC_L	GND
GND_SYNC_R	GND

7.2 Рекомендации по проектированию модулей на базе микропроцессора

7.2.1 Последовательность включения питания

Включение питания выполняется в 6 этапов. Распределение цепей питания по группам включения приведено в таблице 7.2.1. Номер группы включения определяет последовательность включения питания микропроцессора. Внутри 1, 2, 3 и 4 групп включения напряжения в цепях питания могут нарастать в любом порядке. Внутри 5 группы включения напряжения в цепях питания должны нарастать одновременно. Каждая последующая группа должна запускаться при достижении всех напряжений внутри предыдущей группы минимального порога в 90 % от номинальных значений. Группы 3 и 4 допускается объединять, если ждущий режим (S3) не применяется.

Таблица 7.2.1 – Распределение цепей питания по группам включения

Группа включения	Обозначение цепей питания	Наименование цепи питания	Примечание
1	U_{CC8}	PWR_1V8_SUS	Скорость нарастания напряжения не должна превышать 18 мВ/мкс. Должны включаться первыми и отключаться последними. Напряже-
		PWR_1V8_SUS_DDR01	
		PWR_1V8_SUS_DDR45	
2	U_{CC7}	PWR_0V8_SUS	

Группа включения	Обозначение цепей питания	Наименование цепи питания	Примечание
		PWR_0V8_SUS_DDR01	ния для этих цепей обычно формируются из напряжения +5 В дежурного питания, который подаётся из блока питания
		PWR_0V8_SUS_DDR45	
3	U_{CC3}	PWR_1V2_MC_L	Скорость нарастания напряжения не должна превышать 5 мВ/мкс
	U_{CC4}	PWR_1V2_MC_R	
4	U_{CC11}	PWR_1V5_HDA	Скорость нарастания напряжения не должна превышать 18 мВ/мкс
	U_{CC5}	PWR_1V8_BIO	
		PWR_A1V8_IOWL_PE	Скорость нарастания напряжения не должна превышать 180 мВ/мкс
		PWR_A1V8_IPLA	
		PWR_A1V8_IPLB	
		PWR_A1V8_IPLC_PE	
		PWR_A1V8_SATA_ETH	
		PWR_A1V8_MC03	Скорость нарастания напряжения не должна превышать 5 мВ/мкс
		PWR_A1V8_MC47	
		PWR_A1V8_SYNC	Скорость нарастания напряжения не должна превышать 18 мВ/мкс
		PWR_A1V8_SYNC_MC_L	
		PWR_A1V8_SYNC_MC_R	
	U_{CC6}	PWR_A3V3_USB	Скорость нарастания напряжения не должна превышать 100 мВ/мкс
5	U_{CC1}	PWR_0V8_CORE	Скорость нарастания напряжения во всех цепях питания должна быть одинаковой и не должна превышать 18 мВ/мкс. Время между включением питаний в группах 4 и 5 должно быть меньше десятков миллисекунд
	U_{CC2}	PWR_0V8_BIO	
		PWR_A0V8_IOWL_PE_0	
		PWR_A0V8_IOWL_PE_1	
		PWR_A0V8_IOWL_PE_2	
		PWR_A0V8_IOWL_PE_3	
		PWR_A0V8_IPLA_0	
		PWR_A0V8_IPLA_1	
		PWR_A0V8_IPLA_2	
		PWR_A0V8_IPLA_3	
		PWR_A0V8_IPLB_0	

Группа включения	Обозначение цепей питания	Наименование цепи питания	Примечание
		PWR_A0V8_IPLB_1	
		PWR_A0V8_IPLB_2	
		PWR_A0V8_IPLB_3	
		PWR_A0V8_IPLC_PE_0	
		PWR_A0V8_IPLC_PE_1	
		PWR_A0V8_IPLC_PE_2	
		PWR_A0V8_IPLC_PE_3	
		PWR_A0V8_SATA	
		PWR_A0V8_SATA_ETH	
		PWR_A0V8_USB0	
		PWR_A0V8_USB1	
		PWR_A0V8_USB2	
		PWR_A0V8_USB3	
	U_{CC9}	PWR_0V8_MC_L	
	U_{CC10}	PWR_0V8_MC_R	
	U_{CC12}	PWR_A0V8_SYNC	
		PWR_A0V8_SYNC_MC_L	
		PWR_A0V8_SYNC_MC_R	
6	U_{CC13}	PWR_1V8_EFUSE_PGM	Скорость нарастания напряжения не должна превышать 18 мВ/мкс

7.2.2 Последовательность выключения питания

Последовательность выключения питания обратна последовательности включения питания. Внутри 1, 2, 3 и 4 групп напряжения в цепях питания могут спадать в любом порядке. Внутри 5 группы напряжения в цепях питания должны спадать одновременно. Каждая последующая группа должна выключаться при достижении всех напряжений внутри предыдущей группы максимального порога в 10 % от номинальных значений.

7.2.3 Особенности включения/выключения PWR_1V8_EFUSE_PGM

Цепь питания PWR_1V8_EFUSE_PGM всегда должна включаться самой последней из всех цепей питания непосредственно перед программированием блока однократно программируемого ПЗУ на кристалле типа eFuse. Сразу после программирования это питание должно отключаться. Программирование блока eFuse происходит только в стенде тестирования и разбраковки микропроцессора, в серийных платах это питание должно быть подключено к GND.

7.2.4 Поддержка питания микропроцессора в состояниях энергосбережения

Поддержка питания микропроцессора и состояние сигналов завершения включения питания на его входах SUS_PWROK и SYS_PWROK в зависимости от состояния энергосбережения на уровне вычислительного комплекса приведена в таблице 7.2.2. Сигнал SUS_PWROK должен переключаться в высокий уровень только после того, как в норму придут напряжения из 1 и 2 групп включения и синхросигнал SUS_CLK. Сигнал SYS_PWROK должен переключаться в высокий уровень только после того, как в норму придут напряжения из 3, 4 и 5 групп включения.

Таблица 7.2.2 – Поддержка питания МП в зависимости от состояния энергосбережения

Состояние вычислительного комплекса	Поддержка питания	Состояние входов
Режим конфигурации блока eFuse	Все группы	SUS_PWROK = 1, SYS_PWROK = 1
Нормальная работа (S0)	Все группы, кроме группы 6	SUS_PWROK = 1, SYS_PWROK = 1
Ждущий режим (S3)	Группы 1, 2, 3	SUS_PWROK = 1, SYS_PWROK = 0

Состояние вычислительного комплекса	Поддержка питания	Состояние входов
Спящий режим (S4)	Группы 1, 2	SUS_PWROK = 1, SYS_PWROK = 0
Программное выключение (S5)	Группы 1, 2	SUS_PWROK = 1, SYS_PWROK = 0
Механическое выключение (G3)	Нет	SUS_PWROK = 0, SYS_PWROK = 0

7.2.5 Порядок включения и выключения синхросигналов

7.2.5.1 В схеме подключения МП всегда должен присутствовать источник синхросигнала SUS_CLK. Требования 7.2.5.1.1 – 7.2.5.1.5, связанные с синхросигналом SUS_CLK, должны выполняться безусловно.

7.2.5.1.1 Запрещается включать любое питание во всей схеме, если может быть нарушено хотя бы одно из требований 7.2.5.1.2 – 7.2.5.1.5. В частности, запрещается включать любое питание во всей схеме, если источник синхросигнала SUS_CLK отсутствует или неисправен. Нарушение этого запрета может привести к отказу МП.

7.2.5.1.2 При выключенном питании PWR_1V8_SUS, источник синхросигнала SUS_CLK должен быть выключен. Источник синхросигнала SUS_CLK считается выключенным только при выполнении одного из двух следующих условий. Источник синхросигнала SUS_CLK считается выключенным, если выключено его питание. Источник синхросигнала SUS_CLK также считается выключенным, если его выход находится в высокоимпедансном состоянии.

7.2.5.1.3 Включение питания PWR_1V8_SUS должно приводить к включению источника синхросигнала SUS_CLK. Источник синхросигнала SUS_CLK считается включенным только после того, как все параметры этого синхросигнала пришли в норму. Питание PWR_0V8_SUS должно включаться только после того, как включился источник синхросигнала SUS_CLK.

7.2.5.1.4 Пока питание PWR_1V8_SUS остается включенным, все параметры синхросигнала SUS_CLK должны оставаться в норме. Для этого необходимо, чтобы источник синхросигнала SUS_CLK оставался включенным, пока включено питание PWR_1V8_SUS.

7.2.5.1.5 Выключение питания PWR_1V8_SUS должно приводить к выключению источника синхросигнала SUS_CLK.

7.2.5.2 В схеме подключения МП всегда должен присутствовать источник синхросигнала CLK_REF_100M_TOP_P/N. Требования 7.2.5.2.1 – 7.2.5.2.5, связанные с синхросигналом CLK_REF_100M_TOP_P/N, должны выполняться безусловно.

7.2.5.2.1 Запрещается включать любое питание во всей схеме, если может быть нарушено хотя бы одно из требований 7.2.5.2.2 – 7.2.5.2.5. В частности, запрещается включать любое питание во всей схеме, если источник синхросигнала CLK_REF_100M_TOP_P/N отсутствует или неисправен. Нарушение этого запрета может привести к отказу МП.

7.2.5.2.2 При выключенном питании PWR_A1V8_SYNC_MC_R, источник синхросигнала CLK_REF_100M_TOP_P/N должен быть выключен. Источник синхросигнала CLK_REF_100M_TOP_P/N считается выключенным только при выполнении одного из двух следующих условий. Источник синхросигнала CLK_REF_100M_TOP_P/N считается выключенным, если выключено его питание. Источник синхросигнала CLK_REF_100M_TOP_P/N также считается выключенным, если его выход находится в высокоимпедансном состоянии.

7.2.5.2.3 Включение питания PWR_A1V8_SYNC_MC_R должно приводить к включению источника синхросигнала CLK_REF_100M_TOP_P/N. Источник синхросигнала CLK_REF_100M_TOP_P/N считается включенным только после того, как все параметры этого синхросигнала пришли в норму. Питание PWR_A0V8_SYNC_MC_R должно включаться только после того, как включился источник синхросигнала CLK_REF_100M_TOP_P/N.

7.2.5.2.4 Пока питание PWR_A1V8_SYNC_MC_R остается включенным, все параметры синхросигнала CLK_REF_100M_TOP_P/N должны оставаться в норме.

Для этого необходимо, чтобы источник синхросигнала CLK_REF_100M_TOP_P/N оставался включенным, пока включено питание PWR_A1V8_SYNC_MC_R.

7.2.5.2.5 Выключение питания PWR_A1V8_SYNC_MC_R должно приводить к выключению источника синхросигнала CLK_REF_100M_TOP_P/N.

7.2.5.3 В схеме подключения МП всегда должен присутствовать источник синхросигнала CLK_REF_100M_BOT_P/N. Требования 7.2.5.3.1 – 7.2.5.3.5, связанные с синхросигналом CLK_REF_100M_BOT_P/N, должны выполняться безусловно.

7.2.5.3.1 Запрещается включать любое питание во всей схеме, если может быть нарушено хотя бы одно из требований 7.2.5.3.2 – 7.2.5.3.5. В частности, запрещается включать любое питание во всей схеме, если источник синхросигнала CLK_REF_100M_BOT_P/N отсутствует или неисправен. Нарушение этого запрета может привести к отказу МП.

7.2.5.3.2 При выключенном питании PWR_A1V8_SYNC_MC_L, источник синхросигнала CLK_REF_100M_BOT_P/N должен быть выключен. Источник синхросигнала CLK_REF_100M_BOT_P/N считается выключенным только при выполнении одного из двух следующих условий. Источник синхросигнала CLK_REF_100M_BOT_P/N считается выключенным, если выключено его питание. Источник синхросигнала CLK_REF_100M_BOT_P/N также считается выключенным, если его выход находится в высокоимпедансном состоянии.

7.2.5.3.3 Включение питания PWR_A1V8_SYNC_MC_L должно приводить к включению источника синхросигнала CLK_REF_100M_BOT_P/N. Источник синхросигнала CLK_REF_100M_BOT_P/N считается включенным только после того, как все параметры этого синхросигнала пришли в норму. Питание PWR_A0V8_SYNC_MC_L должно включаться только после того, как включился источник синхросигнала CLK_REF_100M_BOT_P/N.

7.2.5.3.4 Пока питание PWR_A1V8_SYNC_MC_L остается включенным, все параметры синхросигнала CLK_REF_100M_BOT_P/N должны оставаться в норме. Для этого необходимо, чтобы источник синхросигнала

CLK_REF_100M_BOT_P/N оставался включенным, пока включено питание PWR_A1V8_SYNC_MC_L.

7.2.5.3.5 Выключение питания PWR_A1V8_SYNC_MC_L должно приводить к выключению источника синхросигнала CLK_REF_100M_BOT_P/N.

7.2.5.4 В схеме подключения МП всегда должен присутствовать источник синхросигнала CLK_REF_100M_EIOH_P/N. Требования 7.2.5.4.1 – 7.2.5.4.5, связанные с синхросигналом CLK_REF_100M_EIOH_P/N, должны выполняться безусловно.

7.2.5.4.1 Запрещается включать любое питание во всей схеме, если может быть нарушено хотя бы одно из требований 7.2.5.4.2 – 7.2.5.4.5. В частности, запрещается включать любое питание во всей схеме, если источник синхросигнала CLK_REF_100M_EIOH_P/N отсутствует или неисправен. Нарушение этого запрета может привести к отказу МП.

7.2.5.4.2 При выключенном питании PWR_A1V8_SYNC, источник синхросигнала CLK_REF_100M_EIOH_P/N должен быть выключен. Источник синхросигнала CLK_REF_100M_EIOH_P/N считается выключенным только при выполнении одного из двух следующих условий. Источник синхросигнала CLK_REF_100M_EIOH_P/N считается выключенным, если выключено его питание. Источник синхросигнала CLK_REF_100M_EIOH_P/N также считается выключенным, если его выход находится в высокоимпедансном состоянии.

7.2.5.4.3 Включение питания PWR_A1V8_SYNC должно приводить к включению источника синхросигнала CLK_REF_100M_EIOH_P/N. Источник синхросигнала CLK_REF_100M_EIOH_P/N считается включенным только после того, как все параметры этого синхросигнала пришли в норму. Питание PWR_A0V8_SYNC должно включаться только после того, как включился источник синхросигнала CLK_REF_100M_EIOH_P/N.

7.2.5.4.4 Пока питание PWR_A1V8_SYNC остается включенным, все параметры синхросигнала CLK_REF_100M_EIOH_P/N должны оставаться в норме. Для этого необходимо, чтобы источник синхросигнала

CLK_REF_100M_EIOH_P/N оставался включенным, пока включено питание PWR_A1V8_SYNC.

7.2.5.4.5 Выключение питания PWR_A1V8_SYNC должно приводить к выключению источника синхросигнала CLK_REF_100M_EIOH_P/N.

7.2.5.5 Синхросигнал SYS_ETHREFCLK_P/N требуется для работы МП во всех конфигурациях с каналами ETH. Этот синхросигнал используется тогда, когда вход SATAETH_CONFIG находится в состоянии лог. «1».

Если в схеме подключения МП не предусмотрена работа МП в конфигурации с каналами ETH, и вход SATAETH_CONFIG подключен к проводнику GND печатной платы, то источник синхросигнала SYS_ETHREFCLK_P/N может быть исключен из схемы подключения МП.

Если источник синхросигнала SYS_ETHREFCLK_P/N исключен из схемы подключения МП, то входы SYS_ETHREFCLK_P и SYS_ETHREFCLK_N должны быть подключены к тому проводнику GND печатной платы, к которому подключены выводы GND_SYNC. Запрещается включать любые питания МП, если источник синхросигнала SYS_ETHREFCLK_P/N исключен из схемы подключения МП, и хотя бы один из входов SYS_ETHREFCLK_P, SYS_ETHREFCLK_N не подключен к указанному проводнику GND.

Требования 7.2.5.5.1 – 7.2.5.5.5, связанные с синхросигналом SYS_ETHREFCLK_P/N, должны выполняться при условии, что источник синхросигнала SYS_ETHREFCLK_P/N не исключен из схемы подключения МП.

7.2.5.5.1 Запрещается включать любое питание во всей схеме, если может быть нарушено хотя бы одно из требований 7.2.5.5.2 – 7.2.5.5.5. В частности, запрещается включать любое питание во всей схеме, если источник синхросигнала SYS_ETHREFCLK_P/N отсутствует или неисправен. Нарушение этого запрета может привести к отказу МП.

7.2.5.5.2 При выключенном питании PWR_A1V8_SYNC, источник синхросигнала SYS_ETHREFCLK_P/N должен быть выключен. Источник синхросигнала SYS_ETHREFCLK_P/N считается выключенным только при выполнении одного из двух следующих условий. Источник синхросигнала SYS_ETHREFCLK_P/N

считается выключенным, если выключено его питание. Источник синхросигнала SYS_ETHREFCLK_P/N также считается выключенным, если его выход находится в высокоимпедансном состоянии.

7.2.5.5.3 Включение питания PWR_A1V8_SYNC должно приводить к включению источника синхросигнала SYS_ETHREFCLK_P/N. Источник синхросигнала SYS_ETHREFCLK_P/N считается включенным только после того, как все параметры этого синхросигнала пришли в норму. Питание PWR_A0V8_SYNC должно включаться только после того, как включился источник синхросигнала SYS_ETHREFCLK_P/N.

7.2.5.5.4 Пока питание PWR_A1V8_SYNC остается включенным, все параметры синхросигнала SYS_ETHREFCLK_P/N должны оставаться в норме. Для этого необходимо, чтобы источник синхросигнала SYS_ETHREFCLK_P/N оставался включенным, пока включено питание PWR_A1V8_SYNC.

7.2.5.5.5 Выключение питания PWR_A1V8_SYNC должно приводить к выключению источника синхросигнала SYS_ETHREFCLK_P/N.

7.2.5.6 Синхросигнал CLK_REF_100M_TEST_P/N используется только при расширенном тестировании МП на предприятии-производителе МП. Входы CLK_REF_100M_TEST_P и CLK_REF_100M_TEST_N должны быть подключены к тому проводнику GND печатной платы, к которому подключены выводы GND_SYNC. Запрещается включать любые питания МП, если хотя бы один из входов CLK_REF_100M_TEST_P, CLK_REF_100M_TEST_N не подключен к указанному проводнику GND.

7.2.5.7 Тестовые выходы для синхросигналов, CLK_TEST_OUT_P/N[1:0], используются только при расширенном тестировании МП на предприятии-производителе МП. Выходы CLK_TEST_OUT_P[1], CLK_TEST_OUT_N[1], CLK_TEST_OUT_P[0], CLK_TEST_OUT_N[0] должны быть подключены к тому проводнику GND печатной платы, к которому подключены выводы GND_SYNC. Запрещается включать любые питания МП, если хотя бы один из выходов CLK_TEST_OUT_P[1], CLK_TEST_OUT_N[1], CLK_TEST_OUT_P[0], CLK_TEST_OUT_N[0] не подключен к указанному проводнику GND.

7.2.5.8 Вход опорного напряжения CLK_TEST_VREF используется только при расширенном тестировании МП на предприятии-производителе МП. Вход CLK_TEST_VREF должен быть подключен к тому проводнику GND печатной платы, к которому подключены выводы GND_SYNC. Запрещается включать любые питания МП, если вход CLK_TEST_VREF не подключен к указанному проводнику GND.

7.2.6 Проектирование конфигураций памяти

Микропроцессор имеет восемь каналов памяти MC0 - MC7(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Каждый канал поддерживает работу с одно-, двух- и четырехранковыми модулями памяти ECC RDIMM, UDIMM, LRDIMM, 3DS в одно- и двухслотовом режиме.

Примечание – Двухслотовый режим поддерживается для всех типов модулей памяти, кроме четырехранковых.

Допускается установка модулей памяти в любые соединители любых каналов памяти микропроцессора. Конфигурации памяти микропроцессоров в многопроцессорных изделиях могут быть разными.

Выводы каналов памяти расположены симметрично по двум сторонам микропроцессора. Между каналами есть попарная микроархитектурная зависимость по частоте работы, что следует учитывать при реализации топологического проектирования изделий на основе микропроцессора Эльбрус-16С для обеспечения сбалансированной работы памяти.

Для каждой из четырех пар каналов используется отдельный блок формирования частоты, эти пары: MC0 и MC1, MC2 и MC3, MC4 и MC5, MC6 и MC7(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается). Поэтому для каналов внутри пар настройка частоты работы общая.

На рисунке 7.2.6.1 изображена схема расположения областей выводов каналов памяти кристалла и корпуса микропроцессора, где попарно объединенные по частоте каналы выделены одним цветом(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается).

Кроме того, топологический дизайн микропроцессора в части трассировки каналов памяти имеет ряд особенностей, которые могут быть использованы для получения оптимальных характеристик каналов памяти изделий на основе микропроцессора Эльбрус-16С.

На рисунке 7.2.6.2 изображена рекомендуемая схема подключения соединителей (слотов) для модулей памяти к каналам памяти микропроцессора.

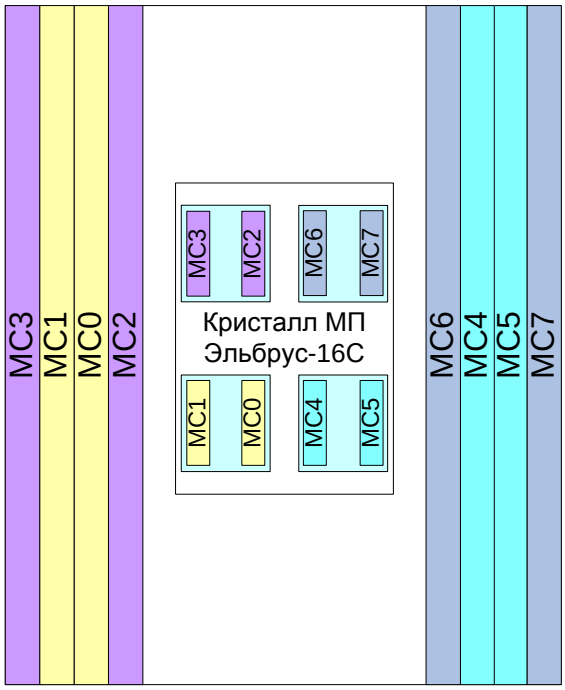


Рисунок 7.2.6.1 – Схема расположения областей выводов каналов памяти кристалла и корпуса микропроцессора (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)

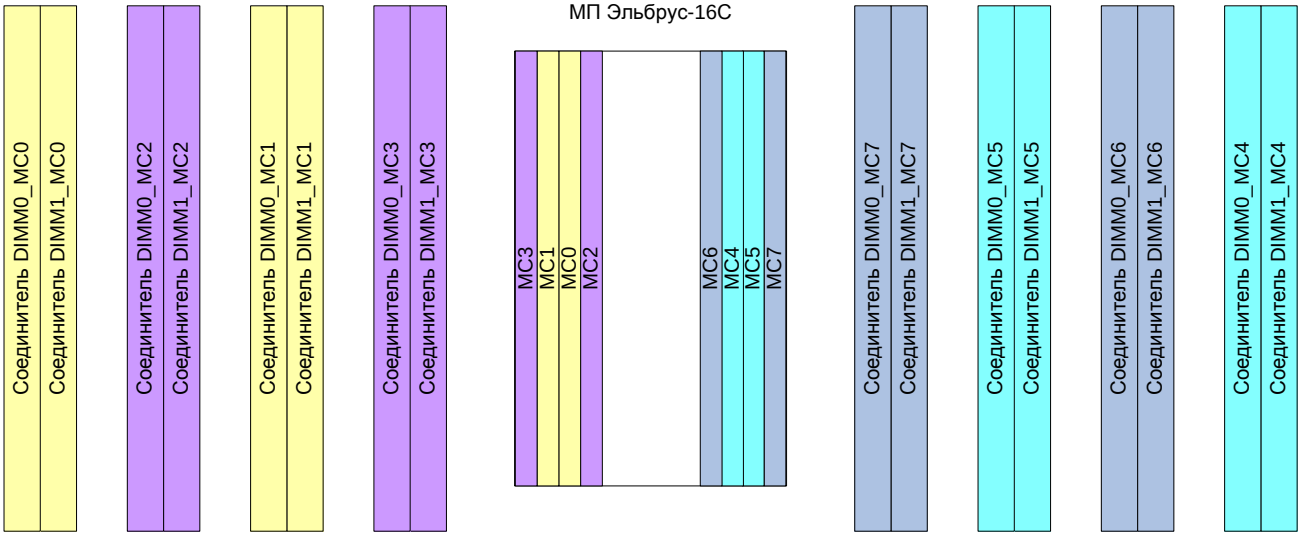


Рисунок 7.2.6.2 – Рекомендуемая схема подключения соединителей для модулей памяти к микропроцессору (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)

Рекомендуемые характеристики топологических связей каналов памяти печатной платы на базе микропроцессора приведены в таблице 7.2.6.1.

Таблица 7.2.6.1 – Рекомендуемые характеристики топологических связей каналов памяти печатной платы на базе микропроцессора Эльбрус-16С

Описание характеристики	Величина характеристики
Номинальное волновое сопротивление одиночного проводника, Zsingle, Ом	от 45 до 50
Номинальное волновое сопротивление дифференциальной пары проводников, Zdifferent, Ом	от 90 до 95
Ширина проводника, W, мкм, не менее	100
Зазор между проводниками сигналов DQ, не менее*	(2,5-3,0)·H**
Зазор между проводниками сигналов DQS/CLK и другими сигналами, не менее*	4·H

Описание характеристики	Величина характеристики
Максимальная длина связи, мм, не более	100
Допустимые отклонения длин проводников, мм, не более:	
DQ к DQ внутри одного байта (полубайта) канала памяти	1,5
DQ к DQS	1,0
DQS к CLK	12,0
AC (сигналы адресной шины) к CLK	2,0
Длина свободной части переходного отверстия (STUB) для каждого из двух отверстий, мм, не более	1,5

* Допускается уменьшение зазоров между проводниками в ограниченных по длине узких местах.

** Где H – высота диэлектрика до опорного полигона

Для обеспечения максимальной производительности оперативной памяти в конечном изделии рекомендуется проектировать сбалансированные конфигурации памяти, такие, в которых трассировка одного канала памяти ведется в двух слоях (минимизации помех и неоднородностей), а в части размещения модулей памяти применяются следующие правила:

− количество задействованных каналов памяти: один, два, четыре или восемь(делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);

− все задействованные каналы памяти должны иметь одинаковое количество установленных модулей памяти (один модуль на канал или два модуля на канал), и эти модули должны быть одинаковыми.

Примечание – Несбалансированные в части размещения модулей памяти конфигурации могут существенно уменьшить производительность конечного изделия.

Таблица 7.2.6.2 иллюстрирует применение правил проектирования сбалансированной в части производительности конфигурации.

DIMM0, DIMM1 – обозначения расположения установленных модулей памяти, причем DIMM0 устанавливается в соединитель, находящийся на большем

расстоянии от микропроцессора. Взаимное расположение соединителей для модулей памяти соответствует схеме, приведенной на рисунке 7.2.6.2.

Таблица 7.2.6.2 – Рекомендуемые конфигурации памяти на базе микропроцессора Эльбрус-16С (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)

<table>
<tr><td rowspan="2">Кол-во задействованных модулей памяти</td><td colspan="8">Обозначение канала памяти процессора</td></tr>
<tr><td>MC0</td><td>MC1</td><td>MC2</td><td>MC3</td><td>MC4</td><td>MC5</td><td>MC6</td><td>MC7</td></tr>
<tr><td>1</td><td colspan="8">DIMM0</td></tr>
<tr><td>2</td><td colspan="4">DIMM0</td><td colspan="4">DIMM0</td></tr>
<tr><td>4</td><td colspan="2">DIMM0</td><td colspan="2">DIMM0</td><td colspan="2">DIMM0</td><td colspan="2">DIMM0</td></tr>
<tr><td>8</td><td>DIMM0</td><td>DIMM0</td><td>DIMM0</td><td>DIMM0</td><td>DIMM0</td><td>DIMM0</td><td>DIMM0</td><td>DIMM0</td></tr>
<tr><td>16</td><td>DIMM0, DIMM1</td><td>DIMM0, DIMM1</td><td>DIMM0, DIMM1</td><td>DIMM0, DIMM1</td><td>DIMM0, DIMM1</td><td>DIMM0, DIMM1</td><td>DIMM0, DIMM1</td><td>DIMM0, DIMM1</td></tr>
</table>

Для обеспечения возможности подключения четырехранковых модулей памяти в однослотовом режиме рекомендуется использовать схемы, приведенные на рисунках 7.2.6.3 и 7.2.6.4.

Примечание – Схема подключения, приведенная на рисунке 7.2.6.3, может быть использована для подключения всех типов модулей памяти, за исключением 3DS.

Для обеспечения возможности подключения четырехранковых модулей памяти в двухслотовом режиме рекомендуется использовать схему, приведенную на рисунке 7.2.6.5.

При проектировании следует учитывать, что суммарная пиковая скорость работы памяти конечного изделия на основе микропроцессора зависит от примененных типов модулей памяти и оптимальности дизайна, а также от количества модулей памяти, подключенных к каналу (см. таблицу 7.2.6.3).

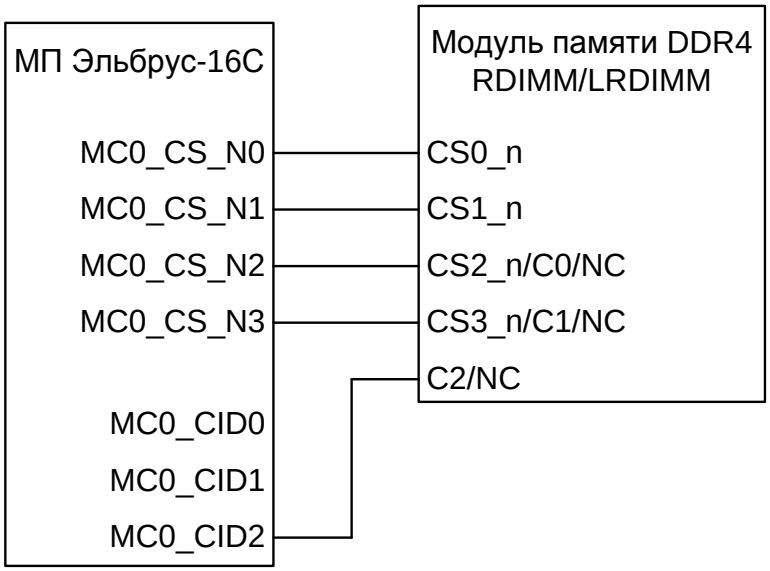


Рисунок 7.2.6.3 – Первая схема подключения четырехранковых модулей памяти к микропроцессору Эльбрус-16С в однослотовом режиме (за исключением модулей памяти 3DS)

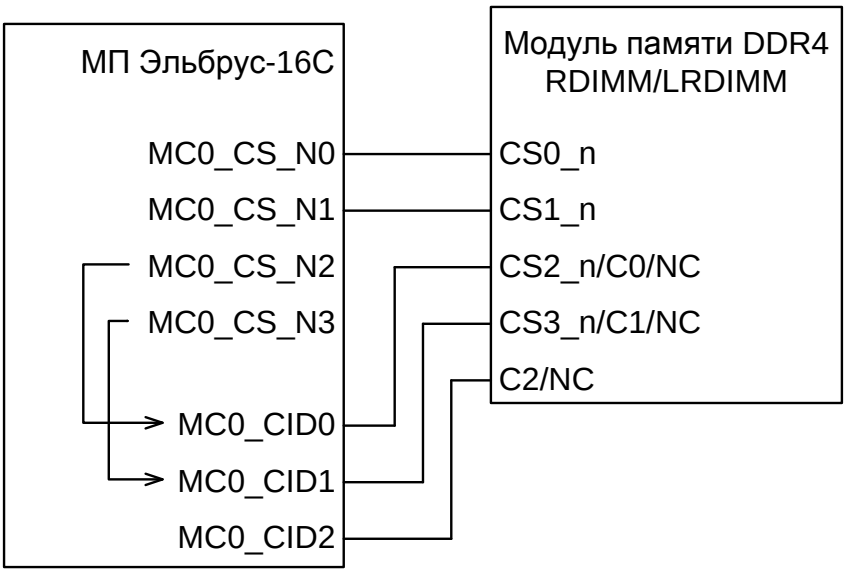


Рисунок 7.2.6.4 – Вторая схема подключения четырехранковых модулей памяти к микропроцессору Эльбрус-16С в однослотовом режиме

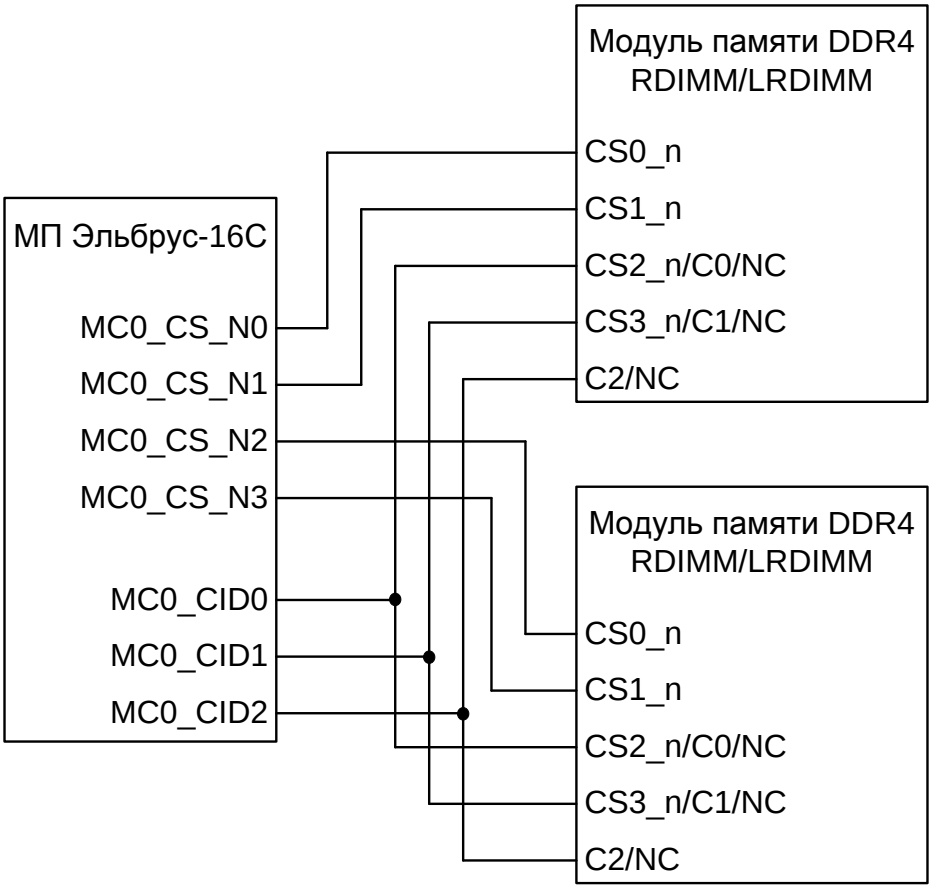


Рисунок 7.2.6.5 – Рекомендуемая схема подключения четырехранковых модулей памяти к микропроцессору Эльбрус-16С в двухслотовом режиме

Таблица 7.2.6.3– Суммарная пиковая скорость работы памяти в зависимости от типа модулей и конфигурации памяти (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается)

Тип модулей памяти	Количество ранков на модуль	Частота модуля памяти, МГц, не менее	Конфигурация памяти	Частота каналов памяти, МГц								Суммарная пиковая скорость работы памяти, Гбайт/с
				MC0	MC1	MC2	MC3	MC4	MC5	MC6	MC7	
RDIMM	1R		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый режим, два модуля памяти									
	2R		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый режим, два модуля памяти									
	4R		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый режим, два модуля памяти									
	3DS		Однослотовый режим, один									

Тип модулей памяти	Количество ранков на модуль	Частота модуля памяти, МГц, не менее	Конфигу-рация памяти	Частота каналов памяти, МГц								Суммарная пиковая скорость работы памяти, Гбайт/с
				MC0	MC1	MC2	MC3	MC4	MC5	MC6	MC7	
			модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый режим, два модуля памяти									
UDIMM	1R		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый режим, два модуля памяти									
	2R		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый режим, два модуля памяти									
	4R		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый									

Тип модулей памяти	Количество ранков на модуль	Частота модуля памяти, МГц, не менее	Конфигу-рация памяти	Частота каналов памяти, МГц								Суммарная пиковая скорость работы памяти, Гбайт/с
				MC0	MC1	MC2	MC3	MC4	MC5	MC6	MC7	
			режим, два модуля памяти									
LRDIMM	1R		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый режим, два модуля памяти									
	2R		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый режим, два модуля памяти									
	4R		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									
			Двухслотовый режим, два модуля памяти									
	3DS		Однослотовый режим, один модуль памяти									
			Двухслотовый режим, один модуль памяти									

Тип модулей памяти	Количество ранков на модуль	Частота модуля памяти, МГц, не менее	Конфигу-рация памяти	Частота каналов памяти, МГц								Суммарная пиковая скорость работы памяти, Гбайт/с
				МС0	МС1	МС2	МС3	МС4	МС5	МС6	МС7	
			Двухслотовый режим, два модуля памяти									

7.3 Стойкость к внешним воздействиям

7.3.1 Стойкость к воздействию механических факторов

Таблица 7.3.1- Стойкость к воздействию механических факторов

Параметры воздействующего фактора, единица измерения	Значение воздействующего фактора
1 Синусоидальная вибрация:	
- диапазон частот, Гц	1- 5000
- амплитуда ускорения, м/с^2 (g)	400 (40)
2 Удары одиночного действия в любом направлении	
- амплитуда пикового ударного ускорения, м/с^2 (g)	15000 (1500)
- длительность действия ударного ускорения, мс	0,1 – 2,0

7.3.2 Стойкость к воздействию климатических факторов

Таблица 7.3.2- Стойкость к воздействию климатических факторов

Параметры воздействующего фактора, единица измерения	Значение воздействующего фактора
1 Атмосферное пониженное рабочее давление, мм рт.ст	10^{-6}
2 Повышенное рабочее давление, ата	3
3 Повышенная температура:	

Параметры воздействующего фактора, единица измерения	Значение воздействующего фактора
- рабочая температура корпуса, ^{O}C	90
- предельная температура среды, ^{O}C	125
4 Пониженная температура среды:	
- рабочая, ^{O}C	минус 40
- предельная, ^{O}C	минус 60
5 Смена температур:	
-от пониженной предельной температуры среды, ^{O}C	минус 60
до повышенной предельной температуры среды, ^{O}C	125
6 Повышенная относительная влажность при 35 ^{O}C, %	98
7 Соляной (морской) туман	

7.3.3 Стойкость к воздействию специальных факторов

Микросхемы стойкие к воздействию специальных факторов 7.И, 7.С, 7К по ГОСТ В 20.39.414.2 со значениями характеристик:

- 7.$И_1$, соответствующими установленным для группы 4$У_С$;
- 7.$И_6$, соответствующими установленным для группы 0,5$У_С$;
- 7.$И_7$, соответствующими установленным для группы 1,5х2$У_С$;
- 7.$И_8$, соответствующими установленным для группы 0,02 х 1$У_С$;
- 7.$С_1$, соответствующими установленным для группы 4$У_С$;
- 7.$С_4$, соответствующими установленным для группы 0,1х1$У_С$;
- 7.$К_1$, соответствующими установленным для группы 1,5х1К;
- 7.$К_4$, соответствующими установленным для группы 0,09х1К.

Тиристорный эффект при указанных выше уровнях характеристики 7.$И_6$ отсутствует.

Допускается в процессе и непосредственно после воздействия специальных факторов 7.И с характеристикой 7.$И_6$ временная потеря работоспособности мик-

росхем. По истечении 2 мс от начала воздействия работоспособность восстанавливается.

7.4 Указания по применению и эксплуатации

Устанавливать и извлекать микросхемы из контактных приспособлений, а также производить их замену необходимо только после снятия напряжений со всех выводов микросхем.

При эксплуатации и испытаниях микросхем, в том числе в моменты включения, выключения, при переходных процессах, при изменении режимов работы должны быть приняты меры по исключению замыканий (даже кратковременных) цепей питания или нагрузочных элементов на выводах микросхем с другими выводами микросхем или другими элементами печатной платы.

После раскрытия упаковки микросхемы рекомендуется устанавливать в аппаратуру в течение 168 часов при следующих производственных условиях: температуре от плюс 20 ^{o}C до плюс 30 ^{o}C и относительной влажность не более 60 %.

Если при открытии пакета показания вложенного индикатора влажности превышает 20 % или не соблюдены рекомендации, представленные выше, то микросхемы перед установкой должны быть подвергнуты горячей сушке (термообработке) в течение 192 часов температуре при температуре (40 ± 5) ^{o}C и влажности менее 5 % в низкотемпературных контейнерах или в течение 24 часов при температуре (125 ± 5) ^{o}C в высокотемпературных контейнерах.

Если микросхемы не установлены в аппаратуру в течение 168 часов, то они должны быть подвергнуты горячей сушке и упакованы.

Монтаж (пайку) микросхемы и всех электрорадиоизделий, размещаемых совместно с микросхемой на одной печатной плате, производить в соответствии с типовым технологическим процессом монтажа, согласованным с АО «МЦСТ».

Микросхемы необходимо покрывать защитным лаком в случае использовании в условиях повышенной влажности.

Питание микросхемы должно осуществляться от стабилизированных источников напряжений питания с отклонением выходного напряжения в пределах ±1,5 %.

Микросхема после снятия с эксплуатации, подлежит утилизации в порядке и методами, устанавливаемыми в контракте на поставку, в соответствии с действующими нормативными документами.

7.5 Справочные данные

7.5.1 Гамма-процентная наработка T_γ при γ = 97,5 % в режимах и условиях эксплуатации, допускаемых ТУ, при температуре окружающей среды не более (65 + 5) °С, составляет 200000 ч.

Прогнозируемая зависимость показателей надежности от температуры кристалла приведена на рисунке 7.5.1.

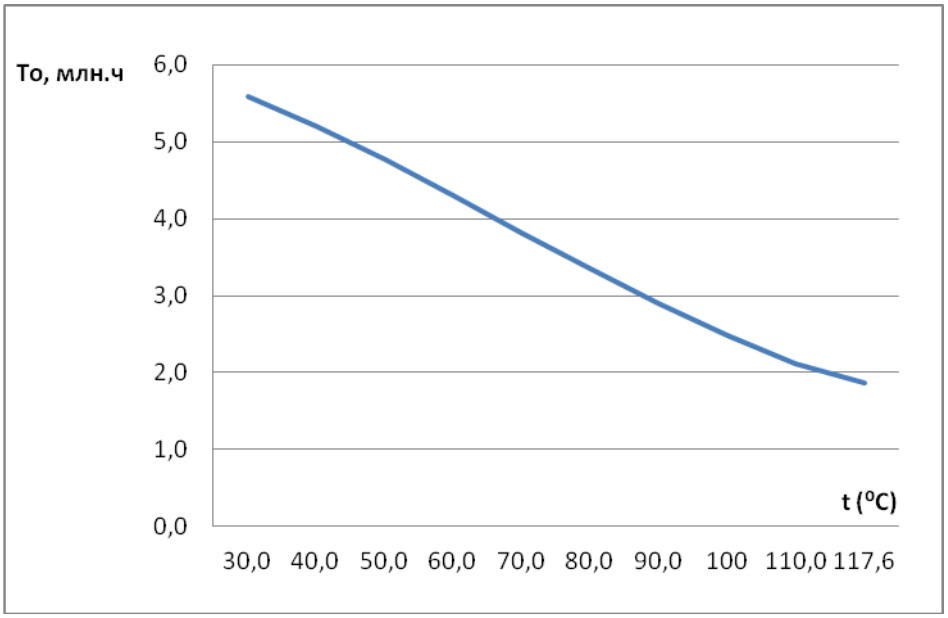


Рисунок 7.5.1 - Прогнозируемая зависимость показателей надежности от температуры кристалла

7.5.2 Зависимости основных электрических параметров микросхем от режимов и условий эксплуатации приведены ниже.

Рассеиваемая мощность не более 130 Вт.

Типовая рассеиваемая мощность, при температуре на кристалле 70 °С, не более:

- 1891ВМ03А8 – 80 - 90 Вт;
- 1891ВМ03В8 – 70 Вт.

Динамическая потребляемая мощность определяется, в основном, мощностью потребляемой ядром микросхемы. Потребляемая мощность ядра обратно пропорциональна квадрату изменения (уменьшения) напряжения питания ядра и составляет:

$$P_{U_x} = P_{\max} \times \left(\frac{U_x}{U_{\max}} \right)^2, \qquad (1)$$

где P_{max} - потребляемая мощность при номинальной частоте и максимальном напряжении питания U_{max} (с учетом допуска);

U_x – рабочее напряжение.

Потребляемая мощность ядра пропорциональна уменьшению рабочей частоты относительно номинальной:

$$P_{f\ раб} = 0{,}8P_{ном} \times \frac{f_{раб}}{f_{ном}} + 0{,}2P_{ном}, \qquad (2)$$

где $f_{ном}$ – номинальная частота, Гц;

$f_{раб}$ – рабочая (пониженная) частота, Гц;

$P_{ном}$ – мощность, потребляемая при номинальных частоте и напряжении питания.

Допустимое (номинальное) напряжение питания ядра может быть уменьшено до $0{,}9U_{ном}$ пропорционально уменьшению частоты синхронизации. Например, на частоте $0{,}9\,f_{ном}$ напряжение питания и потребляемая мощность могут иметь значения: $U_{ядра} = 0{,}9U_{ном}$ и $P_{потр} \approx 0{,}75P_{\max}$. Наиболее экономичным является одновременное снижение напряжения питания и частоты.

Микросхема может использоваться при пониженном значении частоты следования импульсов тактовых сигналов ядра микросхемы до 600 МГц.

Токи периферийных усилителей практически не зависят от температуры.

Статический ток ядра микросхемы имеет слабую зависимость от температуры - увеличивается в 5 раз при изменении температуры окружающей среды от минус 40 °С до 90 °С на крышке корпуса.

Динамический ток потребления микросхем изменяется линейно от изменения питающего напряжения и от изменения частоты.

Токи потребления внешних интерфейсов, использующих дифференциальные сигналы, не меняются при изменении частоты.

Задержки элементов ядра $t_{d\_эл.}$ и максимально-допустимая частота синхронизации f_C связаны с изменением питающего напряжения следующим образом:

$$t_{d\_эл.} = t_{d\_эл.\ ном.} \times (U_{ядра\ ном.} / U_{ядра}),$$
$$f_C = f_{C\ ном.} \times (U_{ядра} / U_{ядра\ ном.}),$$

где $t_{d\_эл.\ ном}$ - задержка элементов ядра при номинальном напряжении питания, с;

$f_{C\ ном.}$ - номинальное значение синхронизации, Гц;

$U_{ядра\ ном.}$- номинальное значение напряжения питания ядра, В.

Зная зависимости частоты синхронизации и мощности от напряжения питания ядра микросхемы можно оптимизировать режимы использования микропроцессора.

7.5.3 Предельная температура p-n-перехода кристалла – 150 ^{O}C.

7.5.4 Значения предельно-допустимого напряжения и предельно-допустимой энергии одиночных импульсов напряжения (ОИН), возникающих при воздействии электромагнитного излучения, при разных значениях длительности импульса приведены в таблице 7.5.1.

Таблица 7.5.1 - Показатели электрической прочности микросхем к воздействию одиночных импульсов напряжения, возникающих при воздействии электромагнитного излучения

Тип вывода	Длительность ОИН, мкс			Параметр
	0,1	1,0	10,0	
Вход	500	200	200	Предельно-допустимое напряжение ОИН, В
Выход	700	300	200	
Цепь питания	3250	1500	1200	

Тип вывода	Длительность ОИН, мкс			Параметр
	0,1	1,0	10,0	
Вход	0,023	0,57	6,2	Расчетная предельно-допустимая энергия ОИН, мДж
Выход	0,045	0,13	0,62	
Цепь питания	0,64	2,2	24	

7.5.5 Тепловые характеристики корпуса микросхемы представлены в таблице 7.5.2.

Таблица 7.5.2 - Тепловые характеристики корпуса микросхемы*

Наименование	Обозначение	Значение
Тепловое сопротивление кристалл – внешняя среда, ºС/Вт	θ_{JA}	0,32
Тепловое сопротивление кристалл – центр крышки корпуса, ºС/Вт	θ_{JT}	0,10
Тепловое сопротивление кристалл – крышка корпуса, ºС/Вт	θ_{JC}	0,10
Тепловое сопротивление кристалл – печатная плата, ºС/Вт	θ_{JB}	0,89

* - при использовании активного радиатора охлаждения типа FHC10040 фирмы Alphanovatech, USA.

Температура крышки корпуса микросхемы не должна превышать 90 ^{o}C.

7.5.6 Емкость входа, выхода и входа/выхода – 10 пФ.

7.5.7 Шарики выводов сформированы оловянно-свинцовым припоем, содержащим 63 % олова и 37 % свинца.

7.5.8 Размеры корпуса микросхемы 78,00 х 63,00 х 5,23 мм.

7.5.9 Количество выводов корпуса 4804.

7.5.10 Максимальный технологический размер элементов 16 нм. Размеры кристалла 25,30 х 24,42 мм (617,826 мм^2).

7.5.11 Условное графическое обозначение микросхем приведено на рисунке 7.5.3.

7.5.12 Микросхема не имеет собственных резонансных частот ниже 100 Гц.

7.5.13 Микросхема не содержит горючих материалов. Пожаробезопасный режим под электрической нагрузкой – не более 150 Вт потребляемой мощности в течение 30 минут в условиях отсутствия системы охлаждения.

7.5.14 Динамические токи потребления микросхемы составляют:

- I_{OCC1} (от источника питания домена CORE):

  — 1891ВМ03А8 – не более 130 А;

  — 1891ВМ03В8 – не более 117А;

- I_{OCC2} — не более 8 А (от источника питания домена UNCORE и низковольтного питания PHY блоков периферии);

- I_{OCC3} — не более 16 А (от источника питания IO ячеек каналов памяти MC0 – MC3) (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);

- I_{OCC4} — не более 16 А (от источника питания IO ячеек каналов памяти MC4 – MC7) (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);

- I_{OCC5} — не более 5 А (от источника питания интерфейсов управления и диагностики и высоковольтного питания PHY блоков периферии, кроме USB);

- I_{OCC6} — не более 0,3 А (от источника питания высоковольтного питания PHY блоков интерфейса USB);

- I_{OCC7} — не более 0.5 А (от источника питания внутренних схем блока Suspend);

- I_{OCC8} — не более 0,5 А (от источника питания выходных каскадов блока Suspend);

- I_{OCC9} — не более 9 А (от источника питания внутренних схем каналов памяти MC0 - MC3) (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);

- I_{OCC10} — не более 9 А (от источника питания внутренних схем каналов памяти MC4 – MC7) (делится на 2. Нечетные каналы заложены для будущего расширения функционала. В текущей итерации МП не поддерживается);

- I_{OCC11} — не более 0,5 А (от источника питания интерфейса HDA);

- I_{OCC12} ─ не более 0,5 А (от источника питания блоков синхронизации);

- I_{OCC13} ─ не более 0,2 А (от источника питания при программировании блока ПЗУ eFuse).

─ до 0,2 А (от источника питания при программировании блока ПЗУ eFuse).

7.5.15 Микросхема содержит цветные металлы:

- медь 7,58 г;

- алюминий 2,81 г;

- олово 0,23 г;

- свинец – 2,40 г.

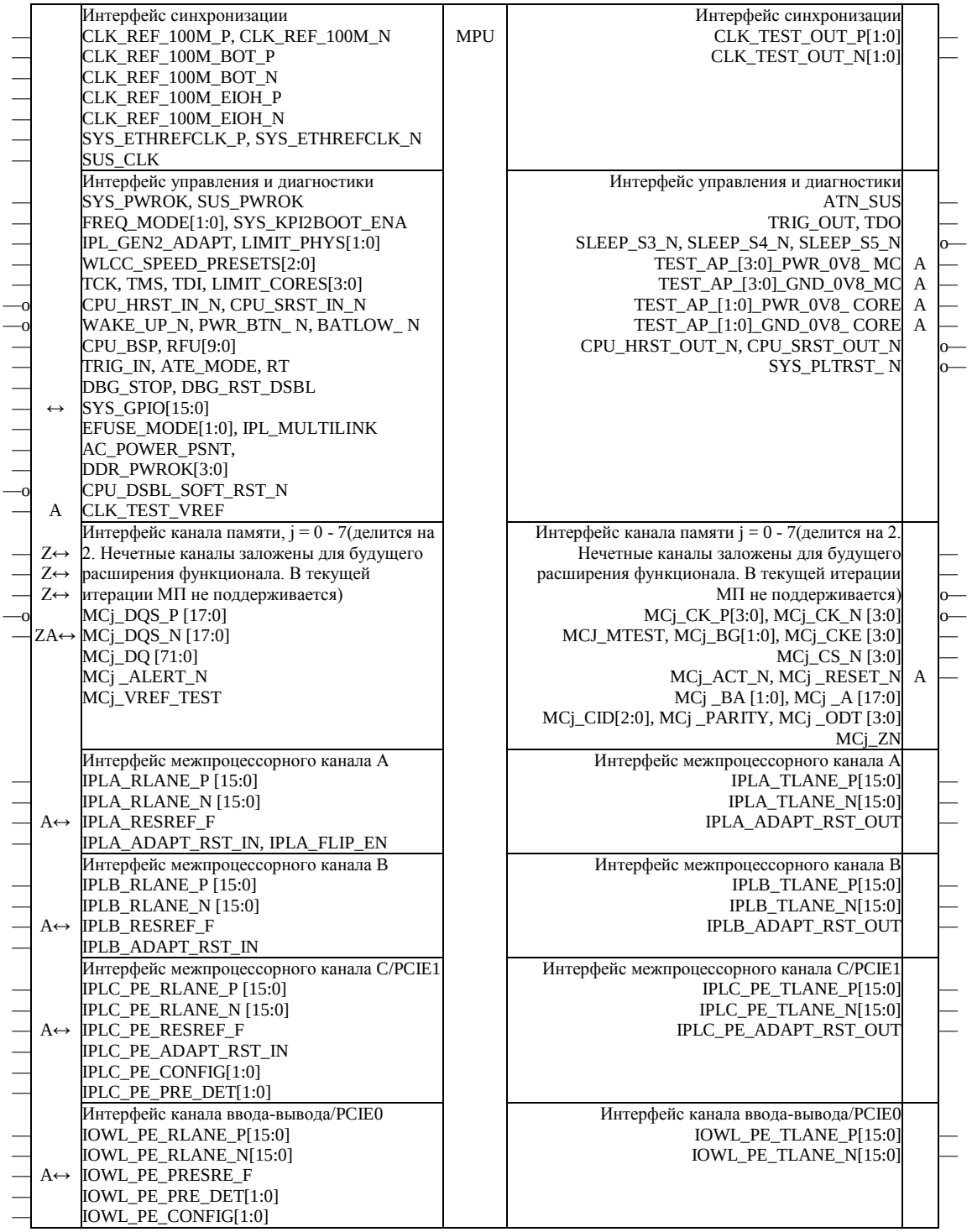


Рисунок 7.5.3 (Лист 1 из 3) – Условное графическое обозначение микросхем 1891ВМ03А8, 1891ВМ03В8

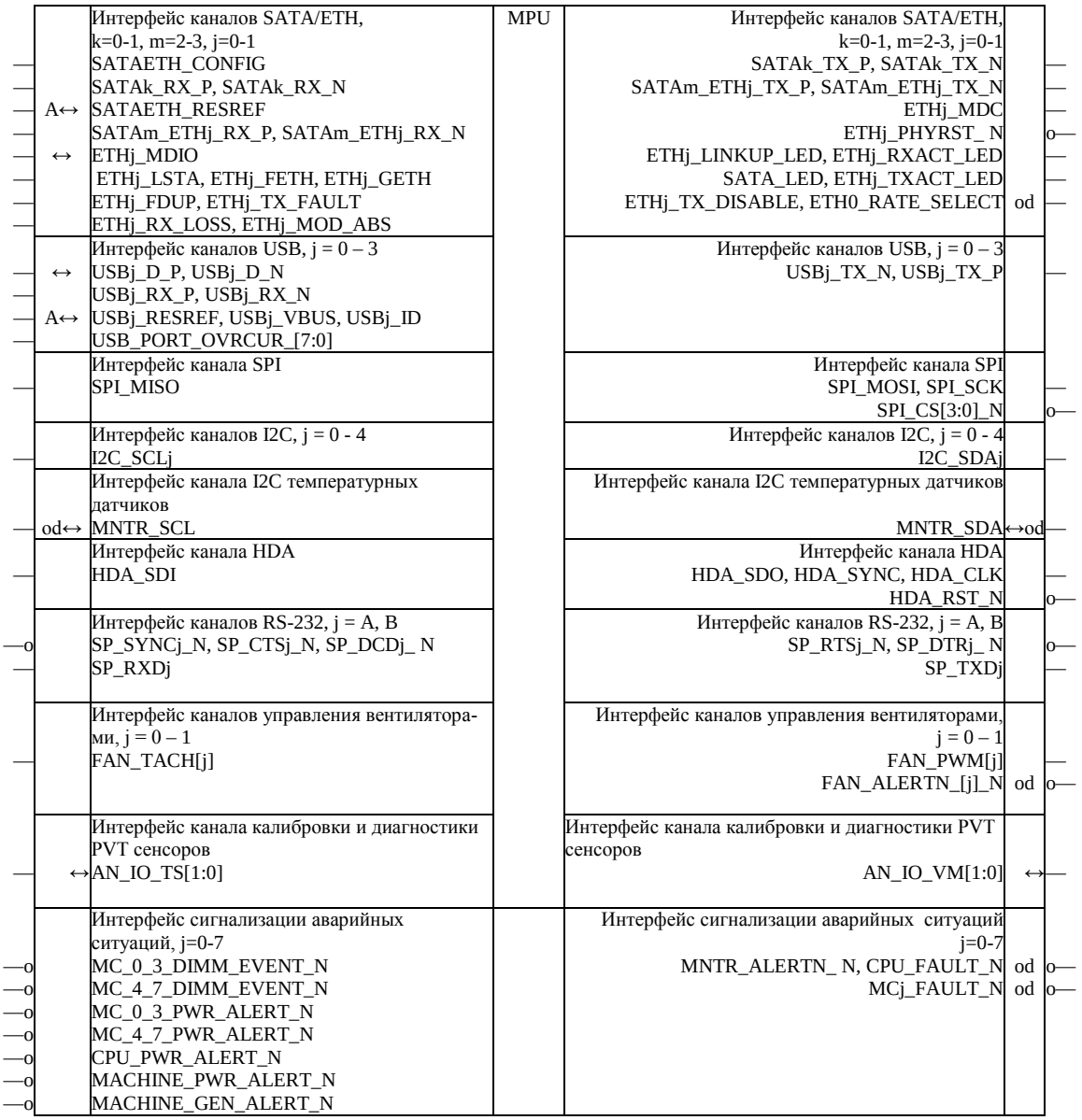


Рисунок 7.5.3 (Лист 2 из 3) – Условное графическое обозначение микросхем 1891ВМ03А8, 1891ВМ03В8

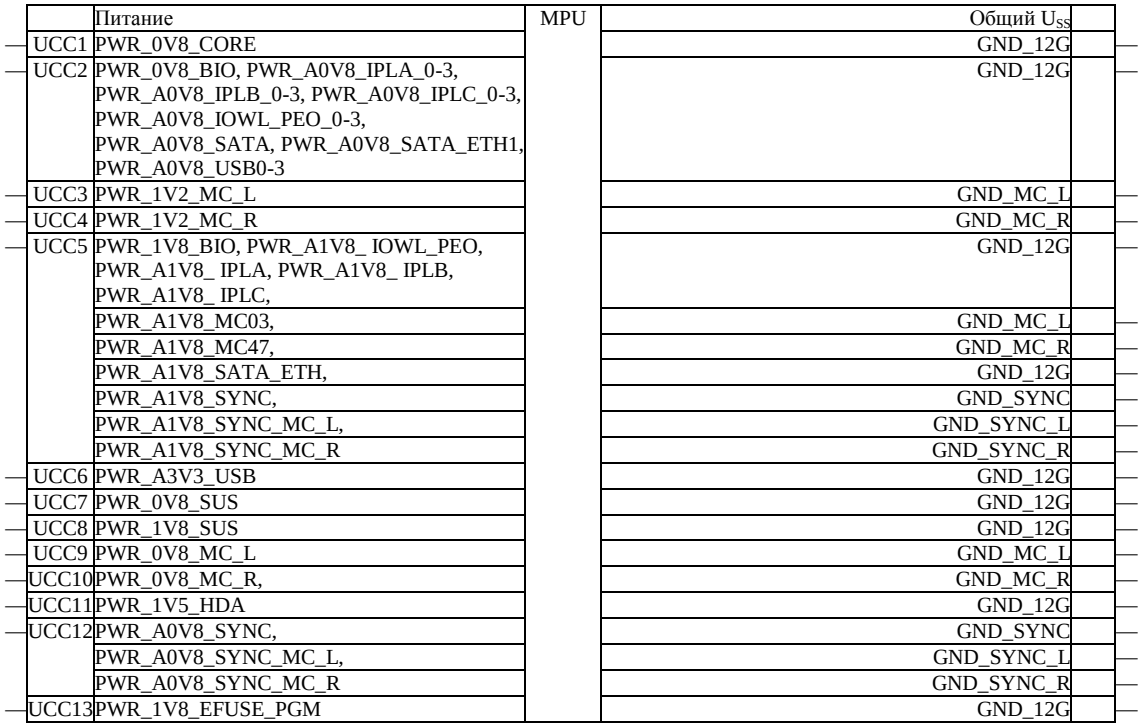


Рисунок 7.5.3 (Лист 3 из 3) – Условное графическое обозначение микросхем 1891ВМ03А8, 1891ВМ03В8

Примечание – Назначение сигналов и координаты выводов приведены в таблице назначения выводов ТВГИ.431281.028ТБ, где списки сигналов интерфейсов микросхем упорядочены в алфавитном порядке и по координатам выводов.

8 Хранение

8.1 Гамма-процентный срок сохраняемости Тсγ при γ = 99 % при хранении микросхем в упаковке изготовителя в условиях отапливаемых хранилищ, хранилищ с кондиционированием воздуха по ГОСТ В 9.003, а также вмонтированных в защищенную аппаратуру или находящихся в защищенном комплекте ЗИП – не мене 25 лет.

8.2 Значения Тсγ для всех климатических районах по ГОСТ В 9.003 (кроме районов с тропическим климатом) в условиях, отличных от указанных в 8.1, в зависимости от мест хранения равен значениям, указанным в таблице 8.1.

Таблица 8.1 – Значения гамма-процентного срока сохраняемости

Место хранения	Значения Тсγ, лет, при хранении	
	в упаковке изготовителя	в составе незащищенных аппаратуры и комплекта ЗИП
Неотапливаемое хранилище	16,5	16,5
Под навесом	12,5	12,5
На открытой площадке	хранение не допускается	12,5

9 Транспортирование

Транспортирование микросхем производится только в упаковке предприятия – изготовителя. При этом транспортирование допускается осуществлять транспортом любого вида на любые расстояния по правилам перевозок грузов, действующим на транспорте данного вида, согласно ГОСТ РВ 20.39.412.

10 Утилизация

Микросхемы не содержат драгоценных металлов и, после списания, подлежат передаче в установленном порядке в соответствующее специализированное предприятие по переработке.

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	Номер документа	Входящий номер сопроводительного документа и дата	Подпись	Дата
	измененных	замененных	новых	аннулированных					