УТВЕРЖДЕН
ЮКСУ.431281.029Д4-УД

МИКРОСХЕМА ИНТЕГРАЛЬНАЯ

1890ВМ3Т

Указания по применению
ЮКСУ.431281.029 Д4

2005

Указания по применению ЮКСУ.431281.029Д4 распространяются на микросхему интегральную 1890ВМ3Т ЮКСУ.431281.029.

Указания по применению разработаны и утверждены на период действия конструкторской документации по литере «$О_1$».

Содержание

Настоящие «Указания по применению» распространяются на микросхему интегральную 1890ВМ3Т и содержат описание назначения, технических характеристик, устройства и принципа работы, параметры и указания по применению.

Перечень принятых обозначений и сокращений

AdEL - Неправильный адрес при загрузке
AdES - Неправильный адрес при записи
Bp - Команда BREAK
CpU - Недоступный сопроцессор
DBE - Ошибка шины при чтении данных
FPE - Исключение от FPU
IBE - Ошибка шины при выборке команд
Int - Прерывание
MCheck -TLB Shutdown
Mod - Запись в защищённую страницу
Ov - Целочисленное переполнение
RI - Нереализованная команда
Sys - Команда SYSCALL
TLBL - Непопадание в TLB при загрузке
TLBS - Непопадание в TLB при записи
Tr - Команды Trap

1 Назначение и область применения

Однокристальный микропроцессор 1890ВМ3Т предназначен для применения в различных вычислительных системах с широким диапазоном параметров «производительность – цена» и является дальнейшим развитием микросхемы 1890ВМ2Т. Микропроцессор 1890ВМ3Т может быть применён как в быстродействующих системах со сложным системным контроллером и динамической памятью, так и в дешёвых системах с невысокой производительностью и простым устройством памяти.

Однокристальный микропроцессор 1890ВМ3Т является 64-х разрядным RISC-процессором с системой команд MIPS64 Release 1 (с некоторыми отличиями, описанными ниже) и предназначен для эксплуатации в диапазоне температур от минус 60 до плюс 85 ^{o}C (температура корпуса).

реализован в виде КМОП СБИС с проектными нормами 0,35 микрон.

1890ВМ3Т содержит большое количество функциональных блоков, таких как встроенные кэш-памяти команд и данных, сопроцессор вещественной арифметики и буфер ассоциативной трансляции виртуальных адресов (TLB), что позволяет уменьшить количество аппаратуры в проектируемой системе.

1890ВМ3Т может работать с операционными системами реального времени, а также с Unix-подобными системами, такими как Linux.

Однокристальный микропроцессор 1890ВМ3Т предназначен для применения в ЭВМ специального назначения.

2 Технические характеристики

Микропроцессор 1890ВМ3Т - 64-х разрядный однокристальный микропроцессор, прототипом которого является микропроцессор IDT79RC64475 фирмы IDT.

Краткая характеристика 1890ВМ3Т:

— разрядность процессора – 64 бит;

— архитектура процессора и система команд – MIPS64 Release 1* (с некоторыми отличиями **);

— наличие сопроцессора вещественной арифметики;

— 6-ступенчатый конвейер с буфером предвыборки команд и возможностью одновременной выборки и запуска на выполнения двух команд – целочисленной и вещественной (Dual Issue);

— наличие контроллера шины;

— трансляция 32-разрядных и 64-разрядных виртуальных адресов в 36-разрядные физические с помощью буфера трансляции TLB (48 входов, 96 страниц). Возможность настройки размера страниц;

— наличие кэш-памятей команд и данных по 16 кБ (4-way). Возможность блокировки кэшей на уровне строки. Механизмы записи в кэш данных Write-through и Write-back;

— наличие трёх режимов привилегий – User, Supervisor, Kernel;

— 64-разрядная системная шина SysAD. Разрядность физического адреса – 36 бит;

— наличие EJTAG и режима Debug.

В данном документе излагается архитектура процессора 1890ВМ3Т с точки зрения программного обеспечения. Приведённая информация носит уточняющий характер. Для глубокого изучения архитектуры MIPS64 Release 1 рекомендуются следующие документы, доступные на сайте http://www.mips.com:

- MIPS64 Architecture For Programmers. Vol. I: Introduction to the MIPS64 Architecture (Rev. 1.00);
- MIPS64 Architecture For Programmers. Vol. II: The MIPS64 Instruction Set (Rev. 1.00);
- MIPS64 Architecture For Programmers. Vol. III: The MIPS64 Privileged Resource Architecture (Rev. 1.00).

---

* - Архитектура MIPS64 Release 1 является надмножеством архитектуры MIPS32 Release 1.

** - Отличия заключаются в том, что в 1890ВМ3Т не реализована команда RSQRT (см. главу 3.2), и в сопроцессоре вещественной арифметики не делается различий между «тихими» и «сигнализирующими» не-числами (см. главу 3.3).

На рисунке 2.1 представлена структурная схема 1890ВМ3Т. На рисунке 2.2 и рисунке 2.3 показаны конвейеры целочисленной части и сопроцессора вещественной арифметики FPU.

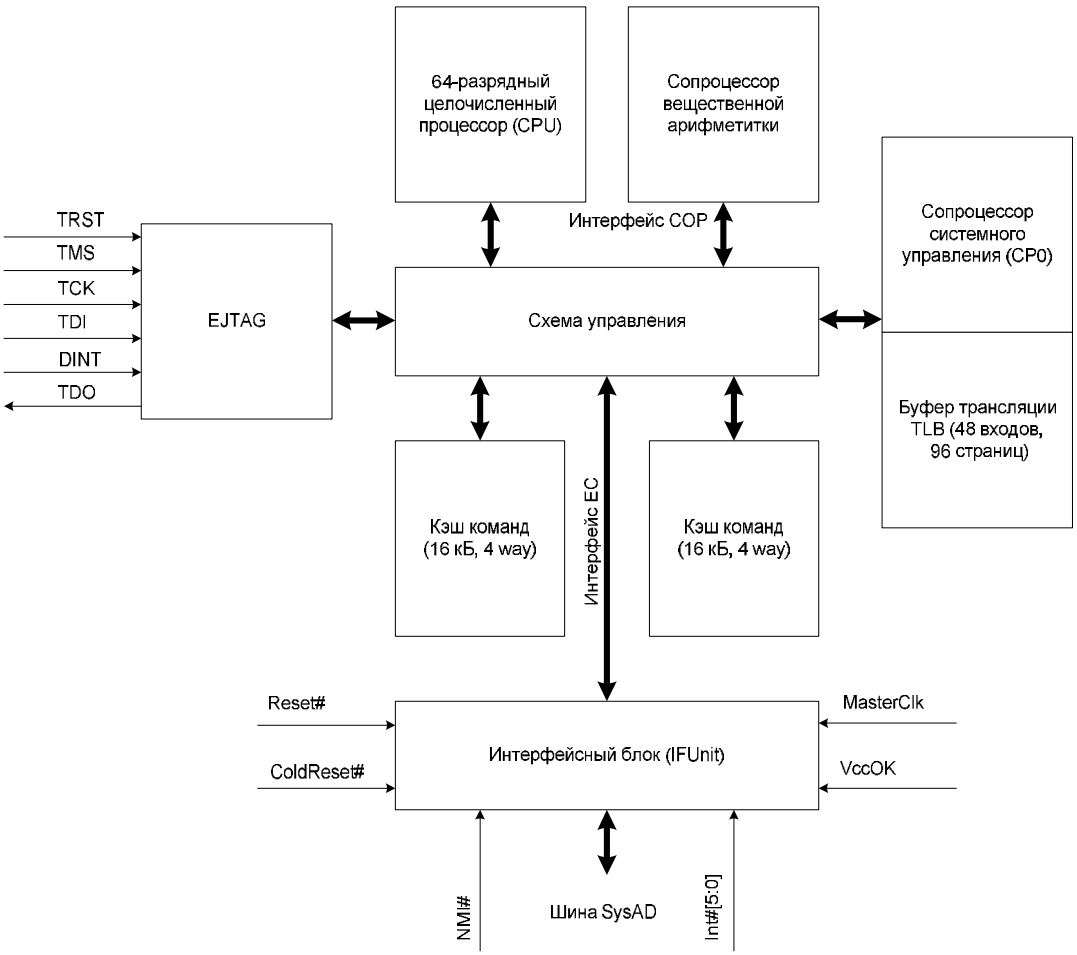


Рисунок 2.1 – Структурная схема однокристального микропроцессора 1890ВМ3Т

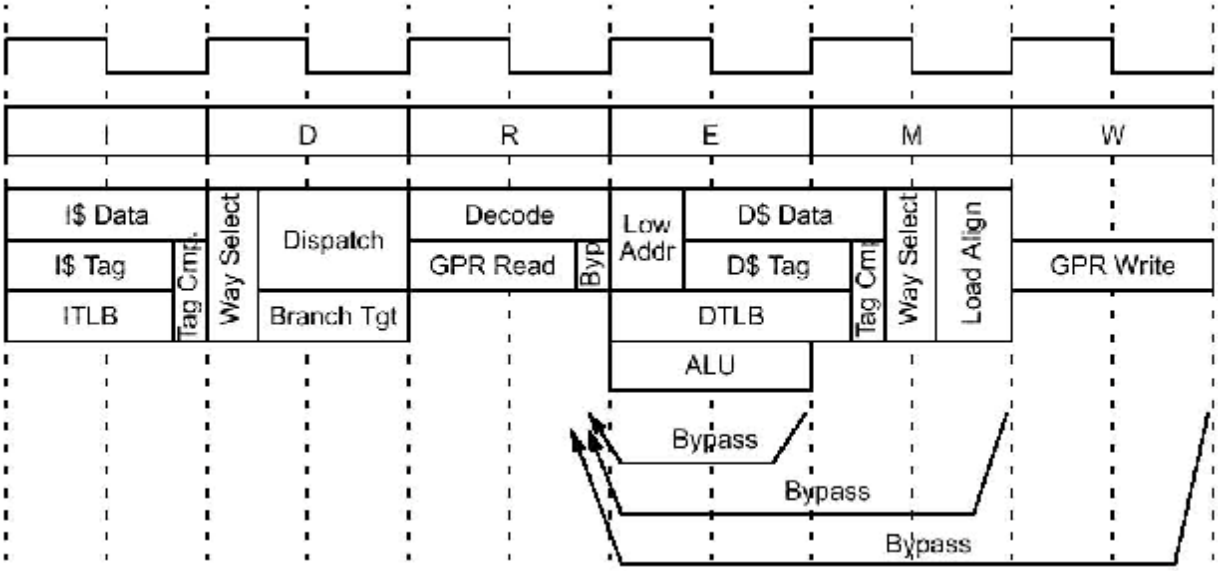


Рисунок 2.2 – Целочисленный (CPU) конвейер

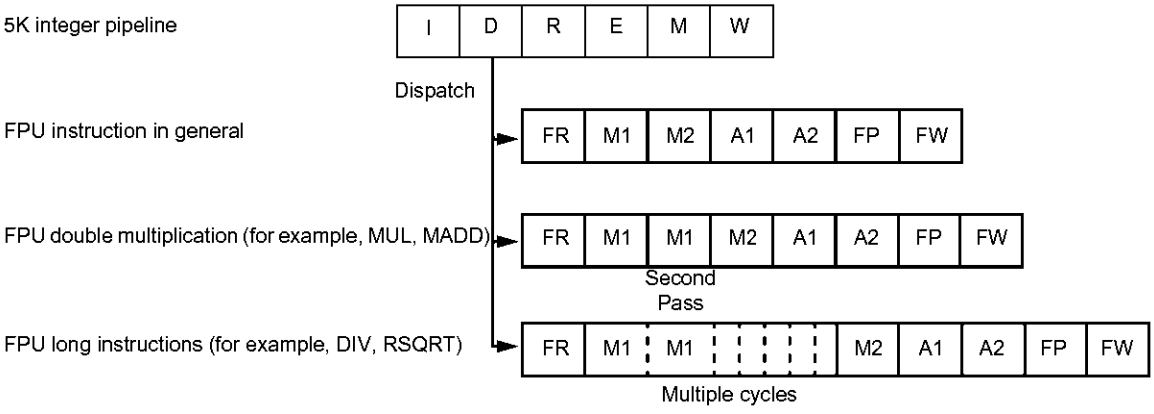


Рисунок 2.3 – Конвейер FPU

3 Описание и принцип работы микросхемы

В настоящее время на основании проведенных патентных исследований микросхемы 1890ВМ3Т (по рубрикам G06F 9/00, 9/06, 9/40, 9/42, 13/00,13/42) сделан вывод в отношении патентов выданных в России на ноябрь 2004 года: исследуемая СБИС 1890ВМ3Т является патентно-чистой.

3.1 Общая программная модель

3.1.1 Форматы данных целочисленной части

B – байт (8 бит)
H – полуслово (16 бит)
W – слово (32 бит)
D – двойное слово (64 бит)

3.1.2 Форматы данных FPU

S – вещественное слово одинарной точности (32 бит)
D – вещественное слово двойной точности (64 бит)
W – целое слово (32 бит)
L – целое двойное слово (64 бит)
Формат L поддерживается только при 32-регистровом режиме FPU (бит FR=1 в регистре Status сопроцессора CP0). (См. ниже).
Формат PS (Paired Single – 32+32 бит) не реализован.

3.1.3 Сопроцессоры (CP0-CP3)

CP0 – сопроцессор системного управления.
CP1 – сопроцессор вещественной арифметики FPU.
CP2 – отсутствует.
CP3 – отсутствует. Часть кодировки команд CP3 используется для CP1.

3.1.4 Регистры целочисленной части (CPU)

Регистры общего назначения – R0-R31. Регистр R0 всегда равен 0.
Регистры результата целочисленного умножения/деления – HI, LO.
Регистр счётчика команд (программно недоступный) – PC.

Все регистры имеют разрядность 64 бит. При выполнении некоторых 32-разрядных операций (сложение, умножение, сдвиг и т.д.) в качестве операндов используются младшие 32 бита регистров. Результат пишется в младшую часть, а на старшую расширяется знак (31-ый бит). Это обеспечивает совместимость 32-разрядных приложений с 64-разрядным режимом процессора. (Описание режимов см. ниже). (Подробное описание команд см. в указанных документах MIPS).

3.1.5 Регистры вещественной арифметики (FPU)

Регистры сопроцессора CP1 делятся на регистры данных (FPR) и регистры управления (FCR). 1890ВМ3Т может работать в двух режимах FPU – 32-регистровом и 16-регистровом. Данные режимы задаются битом FR в регистре Status сопроцессора CP0 (см. ниже).

При FR=0 включается 16-регистровый режим, совместимый с 32-разрядными процессорами MIPS. При данном режиме FPU содержит 16 64-разрядных регистров, каждый из которых может хранить одно значение типа D, S, W (значение L в данном режиме не поддерживается). Обмен данными между этими регистрами и CPU осуществляется 32-разрядными словами. Чётные номера используются для передачи младшей части регистра и для идентификации всего регистра при операциях FPU. Нечётные номера обозначают старшую часть при передаче, а при операциях FPU не используются.

При FR=1 включается 32-регистровый режим. При данном режиме FPU содержит 32 64-разрядных регистра. Регистры адресуются всеми номерами (и чётными, и нечётными). Для обмена 32-разрядными данными (S, W) используются команды MTC1, LWC1 и т. д. Для обмена 64-разрядными данными используются только 64-разрядные команды DMTC1, LDC1 и т. д. Команды обмена старшими 32-разрядными словами MTHC1, MFHC1 относятся к архитектуре MIPS64 Release 2 и в 1890ВМ3Т реализованы не будут.

Описание регистров управления FCR см. ниже в разделе, посвященном FPU.

3.1.6 Режимы адресации Big- и Little-Endian

Процессор 1890ВМ3Т может работать в одном из двух режимов адресации – Big-Endian или Little-Endian. Режим задаётся аппаратно при инициализации процессора. В режиме User (режимы привилегий описаны ниже) возможно переключение на обратный Endian режим (бит RE в регистре Status CP0).

В режиме Big-Endian старшему байту в слове (или старшему слову в двойном слове) соответствует младший адрес (например 0). А младшему байту в слове соответствует старший адрес (например, 3 для байта в слове, 4 для слова в двойном слове).

В режиме Little-Endian – наоборот. Младшим адресам соответствуют младшие байты и слова.

Режим Endian влияет на выполнение команд загрузки/сохранения байтов, полуслов, слов (если используется 64-разрядная внешняя шина или на процессоре выполняются 64-разрядные приложения), невыровненных слов.

Так как внешняя шина процессора может работать в режиме 64 бит (т. е. процессор может читать по 2 команды за шинный такт), то Endian режим нужно учитывать при хранении команд во внешней памяти. В режиме Big-Endian первая (по исполнению) команда должна храниться в старшем слове двойного слова, вторая – в младшем. В Little-Endian – наоборот. Первая команда – в младшем слове, вторая – в старшем.

3.2 Обзор системы команд

В нижеследующих таблицах приведены списки команд, поддерживаемых в 1890ВМ3Т. В графе «архитектура»: MIPS-I – данные команды поддерживаются в процессорах 1В812 и 1890ВМ2Т, MIPS-32 – новые 32-разрядные команды, MIPS-64 – команды, которые поддерживаются только в 64-разрядном режиме 1890ВМ3Т.

В процессоре 1890ВМ3Т реализованы некоторые элементы суперскалярности. Если пара команд представляет собой целочисленную команду и команду вещественной арифметики, то данные команды могут исполнятся одновременно (2 команды за такт). Такой механизм называется Limited Dual-Issue. В графе DualIssue: Int – команда целочисленная, Fp – команда вещественная, NonDU – для пары, содержащей эту команду, Dual Issue не работает.

Подробное описание системы команд микропроцессора с архитектурой MIPS-III приведено на сайте http://www.mips.com (MIPS64 Architecture For Programmers. Vol. II: The MIPS64 Instruction Set (Rev. 1.00)).

Нужно также сказать несколько слов о слотах задержки загрузки (Load Delay Slots) и о некоторых других программных ограничениях, которые налагала архитектура MIPS-I. В архитектуре MIPS-64 данные ограничения снимаются. Действуют следующие положения:

1) За командой загрузки (LB, LW, LD и т. д.) может сразу следовать команда, использующая результат загрузки.
2) Команды загрузки невыровненного слова (LWL/LWR, LDL/LDR), загружающие один регистр, могут следовать в паре – непосредственно одна за другой.
3) За командой передачи слова в FPU (MTC1, DMTC1, LDC1 и т. д.) может сразу следовать команда FPU, использующая это слово.
4) За командой передачи слова из FPU (MFC1, DMFC1) может сразу следовать команда CPU, использующая это слово.
5) За командой сравнения FPU C.cond.fmt могут сразу следовать команды BC1T/F, BC1T/FL, MOVT/F и т. д.
6) Для команд CP0 действуют правила CP0 Hazards (см. раздел, посвященный CP0).

Хотя программно Load Delay Slot отсутствует, всё равно рекомендуется вставлять после команд загрузки одну полезную команду для улучшения производительности, иначе аппаратно возникнет останов на 1 такт.

Таблица 3.1 - Команды арифметические целочисленные

Мнемоника	Описание	Архитектура	Dual Issue
ADD	Сложение со знаком (32 бит)	MIPS-I	Int
ADDI	Сложение непосредственное со знаком (32 бит)	MIPS-I	Int
ADDIU	Сложение непосредственное без знака (32 бит)	MIPS-I	Int
ADDU	Сложение без знака (32 бит)	MIPS-I	Int
CLO	Подсчёт старших единиц в слове (32 бит)	MIPS-32	Int
CLZ	Подсчёт старших нулей в слове (32 бит)	MIPS-32	Int
DADD	Сложение со знаком (64 бит)	MIPS-64	Int
DADDI	Сложение непосредственное со знаком (64 бит)	MIPS-64	Int
DADDIU	Сложение непосредственное без знака (64 бит)	MIPS-64	Int
DADDU	Сложение без знака (64 бит)	MIPS-64	Int
DCLO	Подсчёт старших единиц в дв. слове (64 бит)	MIPS-64	Int
DCLZ	Подсчёт старших нулей в дв. слове (64 бит)	MIPS-64	Int
DDIV	Деление (со знаком) (64 бит)	MIPS-64	Int
DDIVU	Деление (без знака) (64 бит)	MIPS-64	Int
DIV	Деление (со знаком) (32 бит)	MIPS-I	Int
DIVU	Деление (без знака) (32 бит)	MIPS-I	Int
DMULT	Умножение (со знаком) (64 бит)	MIPS-64	Int
DMULTU	Умножение (без знака) (64 бит)	MIPS-64	Int
DSUB	Вычитание со знаком (64 бит)	MIPS-64	Int
DSUBU	Вычитание без знака (64 бит)	MIPS-64	Int
MADD	Умножение с накоплением (32 бит)	MIPS-32	Int
MADDU	Умножение с накоплением (без Ov) (32 бит)	MIPS-32	Int
MSUB	Умножение с вычитанием (32 бит)	MIPS-32	Int
MSUBU	Умножение с вычитанием (без Ov) (32 бит)	MIPS-32	Int
MUL	Умножение (со знаком, результат в GPR) (32 бит)	MIPS-32	Int
MULT	Умножение (со знаком) (32 бит)	MIPS-I	Int
MULTU	Умножение (без знака) (32 бит)	MIPS-I	Int
SLT	Установить 1, если меньше	MIPS-I	Int
SLTI	Установить 1, если меньше (сравнение с непосредственным операндом)	MIPS-I	Int
SLTIU	Установить 1, если меньше (беззнаковое сравнение с непосредственным операндом)	MIPS-I	Int
SLTU	Установить 1, если меньше (беззнаковое сравнение)	MIPS-I	Int
SUB	Вычитание со знаком (32 бит)	MIPS-I	Int
SUBU	Вычитание без знака (32 бит)	MIPS-I	Int

Таблица 3.2 - Команды логики

Мнемоника	Описание	Архитектура	Dual Issue
AND	«И» (64 бит)	MIPS-I	Int
ANDI	«И» непосредственное (64 бит)	MIPS-I	Int
OR	«ИЛИ» (64 бит)	MIPS-I	Int
ORI	«ИЛИ» непосредственное (64 бит)	MIPS-I	Int
XOR	Исключающее «ИЛИ» (64 бит)	MIPS-I	Int
XORI	Исключающее «ИЛИ» непосредственное (64 бит)	MIPS-I	Int
NOR	«НЕ-ИЛИ» (64 бит)	MIPS-I	Int
LUI	Загрузка старшего полуслова непосредственным операндом (32 бит)	MIPS-I	Int

Таблица 3.3 - Команды перемещения

Мнемоника	Описание	Архитектура	Dual Issue
MFHI	Перемещение из регистра HI	MIPS-I	Int
MFLO	Перемещение из регистра LO	MIPS-I	Int
MTHI	Перемещение в регистр HI	MIPS-I	Int
MTLO	Перемещение в регистр LO	MIPS-I	Int
MOVN	Перемещение, если не ноль	MIPS-32	Int
MOVZ	Перемещение, если ноль	MIPS-32	Int
MOVT*	Перемещение, если условие в FPU – 1	MIPS-32	NonDU
MOVF*	Перемещение, если условие в FPU – 0	MIPS-32	NonDU

* - данные команды вызывают исключение, если бит CU1=0 в регистре Status CP0.

Таблица 3.4 - Команды сдвига

Мнемоника	Описание	Архитектура	Dual Issue
SLL	Сдвиг влево логический (32 бит)	MIPS-I	Int
SRL	Сдвиг вправо логический (32 бит)	MIPS-I	Int
SRA	Сдвиг вправо арифметический (32 бит)	MIPS-I	Int
SLLV	Сдвиг влево логический регистровый (32 бит)	MIPS-I	Int
SRLV	Сдвиг вправо логический регистровый (32 бит)	MIPS-I	Int
SRAV	Сдвиг вправо арифметический регистровый (32 бит)	MIPS-I	Int
DSLL	Сдвиг влево логический (64 бит) (от 0 до 31)	MIPS-64	Int
DSRL	Сдвиг вправо логический (64 бит) (от 0 до 31)	MIPS-64	Int
DSRA	Сдвиг вправо арифметический (64 бит) (от 0 до 31)	MIPS-64	Int
DSLLV	Сдвиг влево логический регистровый (64 бит)	MIPS-64	Int
DSRLV	Сдвиг вправо логический регистровый (64 бит)	MIPS-64	Int
DSRAV	Сдвиг вправо арифметический регистровый (64 бит)	MIPS-64	Int
DSLL32	Сдвиг влево логический (64 бит) (от 32 до 63)	MIPS-64	Int
DSRL32	Сдвиг вправо логический (64 бит) (от 32 до 63)	MIPS-64	Int
DSRA32	Сдвиг вправо арифметический (64 бит) (от 32 до 63)	MIPS-64	Int

Таблица 3.5 - Команды переходов

Мнемоника	Описание	Архитектура	Dual Issue
J	Безусловный переход (непосредственный)	MIPS-I	NonDU
JAL	Безусловный переход с возвратом (непосредственный)	MIPS-I	NonDU
JR	Безусловный переход (регистровый)	MIPS-I	NonDU
JALR	Безусловный переход с возвратом (регистровый)	MIPS-I	NonDU
BEQ	Переход, если равно	MIPS-I	NonDU
BNE	Переход, если неравно	MIPS-I	NonDU
BLEZ	Переход, если меньше или равно 0	MIPS-I	NonDU
BGTZ	Переход, если больше 0	MIPS-I	NonDU
BLTZ	Переход, если меньше 0	MIPS-I	NonDU
BGEZ	Переход, если больше или равно 0	MIPS-I	NonDU
BLTZAL	Переход, если меньше 0 с возвратом	MIPS-I	NonDU
BGEZAL	Переход, если больше или равно 0 с возвратом	MIPS-I	NonDU
BEQL	Переход, если равно (Likely)	MIPS-32	NonDU
BNEL	Переход, если неравно (Likely)	MIPS-32	NonDU
BLEZL	Переход, если меньше или равно 0 (Likely)	MIPS-32	NonDU
BGTZL	Переход, если больше 0 (Likely)	MIPS-32	NonDU
BLTZL	Переход, если меньше 0 (Likely)	MIPS-32	NonDU
BGEZL	Переход, если больше или равно 0 (Likely)	MIPS-32	NonDU
BLTZALL	Переход, если меньше 0 с возвратом (Likely)	MIPS-32	NonDU
BGEZALL	Переход, если больше или равно 0 с возвратом (Likely)	MIPS-32	NonDU

Примечание - MIPS не рекомендует использование команд условных переходов «Likely», так как они не будут поддерживаться в будущих архитектурах MIPS.

Таблица 3.6 - Команды пустых операций

Мнемоника	Описание	Архитектура	Dual Issue
NOP	Нет операции	MIPS-I	Int
SSNOP	Нет операции (1 команда за такт)	MIPS-32	NonDU

Таблица 3.7 - Команды загрузки/сохранения

Мнемоника	Описание	Архитектура	Dual Issue
LB	Загрузить байт (со знаком)	MIPS-I	Int
LBU	Загрузить байт (без знака)	MIPS-I	Int
LH	Загрузить полуслово (16 бит) (со знаком)	MIPS-I	Int
LHU	Загрузить полуслово (16 бит) (без знака)	MIPS-I	Int
LW	Загрузить слово (32 бит) (со знаком)	MIPS-I	Int
LWL	Загрузить старшую часть 32-битного слова	MIPS-I	Int
LWR	Загрузить младшую часть 32-битного слова	MIPS-I	Int
SB	Сохранить байт	MIPS-I	Int
SH	Сохранить полуслово (16 бит)	MIPS-I	Int
SW	Сохранить слово (32 бит)	MIPS-I	Int
SWL	Сохранить старшую часть 32-битного слова	MIPS-I	Int
SWR	Сохранить младшую часть 32-битного слова	MIPS-I	Int
LD	Загрузить двойное слово (64 бит)	MIPS-64	Int
LDL	Загрузить старшую часть 64-битного слова	MIPS-64	Int
LDR	Загрузить младшую часть 64-битного слова	MIPS-64	Int
LL*	Загрузить слово (32 бит) атомарно	MIPS-32	Int
LLD*	Загрузить двойное слово (64 бит) атомарно	MIPS-64	Int
LWU	Загрузить слово (32 бит) (без знака)	MIPS-64	Int
SC*	Сохранить слово (32 бит) атомарно	MIPS-32	Int
SCD*	Сохранить двойное слово (64 бит) атомарно	MIPS-64	Int
SD	Сохранить двойное слово (64 бит)	MIPS-64	Int
SDL	Сохранить старшую часть 64-битного слова	MIPS-64	Int
SDR	Сохранить младшую часть 64-битного слова	MIPS-64	Int
PREF**	Предвыборка	MIPS-32	Int
PREFX**	Предвыборка индексированная	MIPS-64	Int
SYNC***	Синхронизировать операции загрузки/сохранения	MIPS-32	NonDU

* Для команд атомарной загрузки/сохранения: LLBit сбрасывается только по командам ERET, SC, SCD.

** При попытке выполнения команд PREF и PREFX исключения Reserved Instruction не возникает, но их выполнение равносильно NOP.

*** Команда SYNC останавливает конвейер до тех пор, пока не завершатся все предыдущие команды, и не очистится буфер записи.

Таблица 3.8 - Команды вызова специальных исключений

Мнемоника	Описание	Архитектура	Dual Issue
SYSCALL	Системный вызов	MIPS-I	NonDU
BREAK	Точка останова	MIPS-I	NonDU
TGE	Исключение, если больше или равно	MIPS-32	NonDU
TGEU	Исключение, если больше или равно (сравнение без знака)	MIPS-32	NonDU
TLT	Исключение, если меньше	MIPS-32	NonDU
TLTU	Исключение, если меньше (сравнение без знака)	MIPS-32	NonDU
TEQ	Исключение, если равно	MIPS-32	NonDU
TNE	Исключение, если неравно	MIPS-32	NonDU
TGEI	Исключение, если больше или равно (сравнение с непосредственным операндом)	MIPS-32	NonDU
TGEUI	Исключение, если больше или равно (сравнение без знака с непосредственным операндом)	MIPS-32	NonDU
TLTI	Исключение, если меньше (сравнение с непосредственным операндом)	MIPS-32	NonDU
TLTUI	Исключение, если меньше (сравнение без знака с непосредственным операндом)	MIPS-32	NonDU
TEQI	Исключение, если равно (сравнение с непосредственным операндом)	MIPS-32	NonDU
TNEI	Исключение, если неравно (сравнение с непосредственным операндом)	MIPS-32	NonDU

Таблица 3.9 - Команды сопроцессора CP0

Мнемоника	Описание	Архитектура	Dual Issue
MTC0	Запись в регистр сопроцессора CP0 (32 бит)	MIPS-I	NonDU
MFC0	Чтение из регистра сопроцессора CP0 (32 бит)	MIPS-I	NonDU
DMTC0	Запись в регистр сопроцессора CP0 (64 бит)	MIPS-64	NonDU
DMFC0	Чтение из регистра сопроцессора CP0 (64 бит)	MIPS-64	NonDU
TLBR	Чтение строки TLB	MIPS-I	NonDU
TLBWI	Запись строки TLB по Index	MIPS-I	NonDU
TLBWR	Запись строки TLB по Random	MIPS-I	NonDU
TLBP	Поиск в TLB	MIPS-I	NonDU
CACHE*	Специальная операция кэш-памяти	MIPS-32	Int
ERET***	Возврат из исключения	MIPS-32	NonDU
WAIT	Режим останова	MIPS-32	NonDU
DERET**	Возврат из исключения EJTAG	EJTAG	NonDU
SDBBP**	Генерация исключения EJTAG	EJTAG	NonDU

Примечание - Команды сопроцессора CP0 (кроме команды SDBBP) являются привилегированными и управляются битом CU0 регистра Status CP0 (см. ниже).

* Команда CACHE и архитектура кэш-памятей подробно описаны ниже.

** Командам и механизмам EJTAG будет посвящен отдельный документ.

*** Для возврата из исключения используется новая команда ERET. Старая команда RFE не поддерживается.

Далее перечислены команды, относящиеся к FPU. Доступность этих команд определяется битом CU1 регистра Status CP0 (см. ниже).

Таблица 3.10 - Арифметические команды FPU

Мнемоника	Описание	Архитектура	Dual Issue
ADD.(S,D)	Вещественное сложение	MIPS-I	Fp
SUB.(S,D)	Вещественное вычитание	MIPS-I	Fp
MUL.(S,D)	Вещественное умножение	MIPS-I	Fp
DIV.(S,D)	Вещественное деление	MIPS-I	Fp
ABS.(S,D)	Взятие абсолютной величины вещественного числа	MIPS-I	Fp
NEG.(S,D)	Смена знака вещественного числа	MIPS-I	Fp
SQRT.(S,D)	Извлечение квадратного корня	MIPS-32	Fp
MADD.(S,D)	Умножение со сложением	MIPS-64	Fp
MSUB.(S,D)	Умножение с вычитанием	MIPS-64	Fp
NMADD.(S,D)	Умножение со сложением и со сменой знака	MIPS-64	Fp
NMSUB.(S,D)	Умножение с вычитанием и со сменой знака	MIPS-64	Fp
RECIP.(S,D)	Взятие обратной величины	MIPS-64	Fp

Примечание - В 1890ВМ3Т не реализована команда RSQRT, что является отступлением от архитектуры MIPS64.

Таблица 3.11 - Команды загрузки, сохранения и перемещения данных в FPU

Мнемоника	Описание	Архитек-тура	Dual Issue
MTC1	Запись в регистр FPU (32 бит)	MIPS-I	Int
MFC1	Чтение из регистра FPU (32 бит)	MIPS-I	Int
DMTC1	Запись в регистр FPU (64 бит)	MIPS-64	Int
DMFC1	Чтение из регистра FPU (64 бит)	MIPS-64	Int
LWC1	Загрузка в регистр FPU из памяти (32 бит)	MIPS-I	Int
SWC1	Сохранение регистра FPU в память (32 бит)	MIPS-I	Int
LDC1	Загрузка в регистр FPU из памяти (64 бит)	MIPS-64	Int
SDC1	Сохранение регистра FPU в память (64 бит)	MIPS-64	Int
LUXC1	Загрузка в регистр FPU из памяти (2-х регистровая адресация) (адрес выравнивается по двойному слову) (32 бит)	MIPS-64	Int
SUXC1	Сохранение регистра FPU в память (2-х регистровая адресация) (адрес выравнивается по двойному слову) (32 бит)	MIPS-64	Int
LWXC1	Загрузка в регистр FPU из памяти (2-х регистровая адресация) (32 бит)	MIPS-64	Int
SWXC1	Сохранение регистра FPU в память (2-х регистровая адресация) (32 бит)	MIPS-64	Int
LDXC1	Загрузка в регистр FPU из памяти (2-х регистровая адресация) (64 бит)	MIPS-64	Int
SDXC1	Сохранение регистра FPU в память (2-х регистровая адресация) (64 бит)	MIPS-64	Int
CTC1	Запись в управляющий регистр FPU	MIPS-I	NonDU
CFC1	Чтение из управляющего регистра FPU (32 бит)	MIPS-I	NonDU
MOV.(S,D)	Перемещение вещественных чисел	MIPS-I	Fp
MOVT.(S,D)	Перемещение вещественных чисел, если сравнение истинно	MIPS-32	Fp
MOVF.(S,D)	Перемещение вещественных чисел, если сравнение ложно	MIPS-32	Fp
MOVN.(S,D)	Перемещение вещественных чисел, если не ноль	MIPS-32	NonDU
MOVZ.(S,D)	Перемещение вещественных чисел, если ноль	MIPS-32	NonDU

Таблица 3.12 - Команды преобразования в FPU

Мнемоника	Описание	Архитектура	Dual Issue
CVT.S.(D,W,L)	Перевод в число с плавающей точкой одинарной точности	MIPS-I, MIPS-64 (для L)	Fp
CVT.D.(S,W,L)	Перевод в число с плавающей точкой двойной точности	MIPS-I, MIPS-64 (для L)	Fp
CVT.W.(S,D)	Перевод в целое число со знаком (32 бит)	MIPS-I	Fp
CVT.L.(S,D)	Перевод в целое число со знаком (64 бит)	MIPS-64	Fp
ROUND.W.(S,D)	Перевод в целое число со знаком (32 бит) (RM=0)	MIPS-32	Fp
ROUND.L.(S,D)	Перевод в целое число со знаком (64 бит) (RM=0)	MIPS-64	Fp
TRUNC.W.(S,D)	Перевод в целое число со знаком (32 бит) (RM=1)	MIPS-32	Fp
TRUNC.L.(S,D)	Перевод в целое число со знаком (64 бит) (RM=1)	MIPS-64	Fp
CEIL.W.(S,D)	Перевод в целое число со знаком (32 бит) (RM=2)	MIPS-32	Fp
CEIL.L.(S,D)	Перевод в целое число со знаком (64 бит) (RM=2)	MIPS-64	Fp
FLOOR.W.(S,D)	Перевод в целое число со знаком (32 бит) (RM=3)	MIPS-32	Fp
FLOOR.L.(S,D)	Перевод в целое число со знаком (64 бит) (RM=3)	MIPS-64	Fp

Примечание - В 1890ВМ3Т не реализован формат данных Paired Single (PS). Соответственно, не реализованы команды, использующие данный формат.

Таблица 3.13 - Команды сравнения и ветвления в FPU

Мнемоника	Описание	Архитектура	Dual Issue
C.cond.(S,D)	Сравнение вещественных чисел	MIPS-I	Fp
BC1T	Переход, если результат сравнения истинен	MIPS-I	NonDU
BC1F	Переход, если результат сравнения ложен	MIPS-I	NonDU
BC1TL*	Переход, если результат сравнения истинен (Likely)	MIPS-32	NonDU
BC1FL*	Переход, если результат сравнения ложен (Likely)	MIPS-32	NonDU

* MIPS не рекомендует использование команд перехода «Likely», так как эти команды будут исключены из архитектуры в будущем.

3.2.1 Времена исполнения команд

В таблице 3.14 и таблице 3.15 приведены времена исполнения команд в тактах. Данные значения являются предварительными и могут в будущем уточняться.

Колонка Latency показывает, через сколько тактов результат команды станет доступным. Если следующая команда требует результат раньше, чем он доступен, то возникает аппаратная задержка, которая прозрачна с программной точки зрения.

Колонка Repeat показывает, через сколько тактов может начать выполняться одноимённая команда. В случае команд перехода программно должен соблюдаться Branch Delay Slot (одна команда между переходами), иначе результат непредсказуем. На остальные команды программных ограничений не накладывается – если не выполняется требование Repeat, то возникает аппаратная задержка.

Таблица 3.14 - Время исполнения целочисленных команд

Команда	Latency	Repeat
LD, LW, LWU, LL, LLD	2	1
LB(U), LH(U), LWL/R, LDL/R	3	1
MULT(U), DMULT(U), MADD(U), MSUB(U), MUL (Операнды 32*16)	3	3
MULT(U), DMULT(U), MADD(U), MSUB(U), MUL (Операнды 32*32)	4	4
DMULT(U) Операнды 64*64 или 32*64	11	11
DDIV(U) Операнды 64/x	69	69
DDIV(U) Операнды 56/x	61	61
DDIV(U) Операнды 48/x	53	53
DDIV(U) Операнды 40/x	45	45
DDIV(U), DIV(U) Операнды 32/x	37	37
DDIV(U), DIV(U) Операнды 24/x	29	29
DDIV(U), DIV(U) Операнды 16/x	21	21
DDIV(U), DIV(U) Операнды 8/x	13	13
Branch	2	2
J, JAL	2	2
JR, JALR	3	2
Другие целочисленные команды	1	1

Нужно отметить, что одновременно может исполняться только одна команда деления/умножения (исполнение сразу нескольких команд не имеет смысла – результат пишется в HI/LO). То же относится и к команде MUL. Причём выполнение команды MUL согласно архитектуре MIPS64 не гарантирует сохранность содержимого регистров HI/LO.

Таблица 3.15 - Время исполнения команд FPU

Команда	Latency	Repeat
MFC1, DMFC1	2	1
(D)MTC1, LD/WC1, LD/WXC1, LUXC1	3	1
ABS, NEG, MOV, MOV(T/F/N/Z)	4	1
ADD.(S,D), SUB.(S.D)	4	1
CEIL, FLOOR, TRUNC, ROUND	4	1
C.cond.fmt	4	1
CVT.S.(D,W,L)	4	1
CVT.D.S	4	1
CVT.D.(W,L)	4	1
CVT.W.(S,D), CVT.L.(S,D)	4	1
DIV.S	17	14
DIV.D	32	29
MADD.S, MSUB.S, NMADD.S, NMSUB.S	4	1
MADD.D, MSUB.D, NMADD.D, NMSUB.D	5	2*
MUL.S	4	1
MUL.D	5	2*
RECIP.S	17	14
RECIP.D	32	29
SQRT.S	17	14
SQRT.D	32	29

* Любая арифметическая команда (кроме команд DIV.fmt, RECIP.fmt, SQRT.fmt), следующая сразу после команды умножения двойной точности, будет задержана на 1 такт.

Для обеспечения высокой производительности компилятором должны учитываться времена выполнения команд умножения/деления и команд FPU.

Между вышеперечисленными командами и командами, требующими результат выполнения, по возможности должны вставляться другие полезные команды.

В 1890ВМ3Т реализован механизм неблокирующего кэша данных (Nonblocking Cache – hit under 1 miss). Этот механизм позволяет не останавливать конвейер в случае одиночного непопадания в кэш, если результат команды загрузки не требуется немедленно. Для повышения производительности компилятору при заведомом непопадании в кэш следует также между командой загрузки и командой, требующей результат, вставлять по возможности другие полезные команды.

3.3 Обзор сопроцессора вещественной арифметики FPU

Сопроцессор вещественной арифметики 1890ВМ3Т в целом удовлетворяет архитектуре MIPS64. Небольшое отступление заключается в отсутствии различия между «тихими» и «сигнализирующими» не-числами в 1890ВМ3Т. Любое не-число, используемое в качестве операнда вычислительной команды FPU, вызывает исключение V. (Данное утверждение не совсем относится к командам сравнения. Исключение V вызывается только сигнализирующими сравнениями при использовании не-числа в качестве операнда.)

Данная глава освещает только регистры управления FPU и особенности возникновения исключений FPU. Регистровый файл общего назначения описан выше. Подробную информацию можно найти в стандарте IEEE 754, в документах MIPS и в руководстве по ОМП 1В812.

В таблице 3.16 приведены регистры управления FPU. Обмен данными с ними осуществляется командами CTC1 и CFC1.

Таблица 3.16 - Регистры управления FPU

Номер	Название	Описание
0	FIR	Floating-Point Implementation register. Содержит идентификационную информацию FPU
25	FCCR	Floating-Point Condition Codes register. Дублирование поля FCSR. Коды условий.
26	FEXR	Floating-Point Exceptions register. Дублирование поля FCSR. Текущие исключения.
28	FENR	Floating-Point Enables register. Дублирование поля FCSR. Разрешения исключений.
31	FCSR	Floating-Point Control and Status register. Регистр состояния.

3.3.1 Регистр идентификации FPU (FIR)

Данный регистр доступен только на чтение.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												3D	PS	D	S	ProcessorID								Revision							
0												0	0	1	1	0x81								0x01							

Рисунок 3.1 - Регистр FIR

Таблица 3.17 - Поля регистра FIR

Поле	Описание
3D	0. В 1890ВМ3Т отсутствует MIPS-3D расширение.
PS	0. Не реализован формат Paired Single.
D	1. Поддерживается формат двойной точности.
S	1. Поддерживается формат одинарной точности.
ProcessorID	ID процессора. Для 1890ВМ3Т – 0x81.
Revision	Ревизия процессора. Для 1890ВМ3Т – 0x01 .

3.3.2 Регистр состояния FPU (FCSR)

Данный регистр доступен по чтению и записи.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FCC							FS	FCC	0					Cause						Enables					Flags					RM	
7	6	5	4	3	2	1		0						E	V	Z	O	U	I	V	Z	O	U	I	V	Z	O	U	I		

Рисунок 3.2 - Регистр FCSR

Таблица 3.18 - Поля регистра FCSR

Поле	Описание
FCC	FPU Condition Codes. Биты, устанавливаемые командой C.cond.fmt и используемые командами перехода и условного перемещения
FS	Flush to Zero. Если FS=1 и Enables[U]=0, то в случае антипереполнения (Underflow) исключения (E) не происходит, и результат округляется до 0 в соответствии с RM (см. ниже).
Cause	Биты, показывающие, какие исключения вызвала последняя команда FPU. (См. ниже). Запись в это поле (в случае установленных соответствующих Enables) вызывает исключение FPU.
Enables	Биты, разрешающие исключения FPU. Если соответствующий бит сброшен, то в случае возникновения арифметического исключения – исключения FPU не происходит, выдаётся соответствующий результат (см. ниже), и устанавливается соответствующий бит в Flags.
Flags	Аккумулирующие биты. Сбрасываются в 0 только программно. Устанавливаются в 1 в случае возникновения исключения при соответствующем сброшенном Enable.
RM	Rounding mode. Режим округления. 00 – RN – округление в сторону ближайшего 01 – RZ – округление в сторону нуля 10 – RP – округление в положительную сторону 11 – RM – округление в отрицательную сторону

Таблица 3.19 - Исключения FPU

E	Нереализованная операция
V	Неправильная операция
Z	Деление на ноль
O	Переполнение
U	Антипереполнение
I	Неточный результат

Условия возникновения исключений описаны ниже.

3.3.3 Регистр кодов условий (FCCR)

Данный регистр обеспечивает альтернативный доступ к FCSR. Доступен на чтение/запись.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0																								FCC							
																								7	6	5	4	3	2	1	0

Рисунок 3.3 - Регистр FCCR

3.3.4 Регистр исключений (FEXR)

Данный регистр обеспечивает альтернативный доступ к FCSR. Доступен на чтение/запись.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														Cause						0					Flags					0	
														E	V	Z	O	U	I						V	Z	O	U	I		

Рисунок 3.4 - Регистр FEXR

3.3.5 Регистр разрешений (FENR)

Данный регистр обеспечивает альтернативный доступ к FCSR. Доступен на чтение/запись.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																				Enables					0				FS	RM	
																				V	Z	O	U	I							

Рисунок 3.5 - Регистр FENR

3.3.6 Исключение Z

Исключение Z возникает при выполнении команд RECIP, DIV, когда делитель равен 0, а делимое не равно 0.

Если Enables[Z]=1, возникает исключение FPU (Cause[Z]=1).

Если Enables[Z]=0, то устанавливается Flags[Z]=1, Cause[Z]=1, и в качестве результата выдаётся бесконечность с соответствующим знаком.

3.3.7 Исключение I

Исключение I возникает при потере точности результата при выполнении вычислительных команд и команд конвертации, а также при возникновении исключений O и U.

Если Enables[I]=1, возникает исключение FPU (Cause[I]=1). Нужно отметить, что без лишней надобности не следует устанавливать этот бит, так как это повлечёт значительное ухудшение производительности FPU. (В этом случае отключается конвейерный режим работы FPU, т.к. необходимо обеспечить точность исключения I, а возможность возникновения этого исключения нельзя предсказать на ранних стадиях).

Если Enables[I]=0, то устанавливается Flags[I]=1, Cause[I]=1, и выдаётся округлённый результат в соответствии с режимом округления.

3.3.8 Исключение O

Исключение O возникает при выполнении вычислительной команды, если полученный результат превышает границу формата. (Не относится к командам конвертации – они в случае переполнения вызывают V). При данном исключении устанавливаются биты Cause O и I.

Если Enables[O]=1 или Enables[I]=1, то вырабатывается исключение FPU (Cause[O,I]=1).

Если Enables[O]=0 и Enables[I]=0, то устанавливаются Flags[O,I] и Cause[O,I], и вырабатывается результат, зависящий от режима округления (таблица 3.20).

Таблица 3.20 - Результат переполнения

RM	Результат
00	Бесконечность с соответствующим знаком
01	Самое большое (по модулю) конечное число с соответствующим знаком
10	Если отрицательное переполнение – самое большое (по модулю) конечное отрицательное число. Если положительное – +∞.
11	Если положительное переполнение – самое большое (по модулю) конечное положительное число. Если отрицательное – –∞.

3.3.9 Исключение U

Исключение U возникает при выполнении вычислительной команды, если полученный результат настолько мал, что он денормализован, либо его округление даст 0. При данном исключении устанавливаются биты Cause U и I.

Если Enables[U]=1, то устанавливаются Cause[U,I]=1, и вырабатывается исключение FPU.

Если Enables[U]=0 и бит в FCSR FS=0, то устанавливается Cause[E]=1, и вырабатывается исключение FPU.

Если Enables[U]=0, бит в FCSR FS=1 и Enables[I]=1, то устанавливается Cause[E]=1, и вырабатывается исключение FPU.

Если Enables[U]=0, бит в FCSR FS=1 и Enables[I]=0, то устанавливаются Flags[U,I]=1, Cause[U,I]=1, и вырабатывается результат, зависящий от режима округления (таблица 3.21).

Таблица 3.21 - Результат при FS=1

RM	Результат
00	Ноль с соответствующим знаком
01	Ноль с соответствующим знаком
10	Если отрицательное антипереполнение – –0. Если положительное – самое маленькое (по модулю) конечное положительное число.
11	Если положительное антипереполнение – +0. Если отрицательное – самое маленькое (по модулю) конечное отрицательное число.

3.3.10 Исключение V

Исключение неправильной операции возникает при следующих ситуациях:

- Сложение или вычитание, при котором невозможно предсказать результат. Например, (+∞)-(+∞) или (-∞)+(+∞).
- Умножение, при котором невозможно предсказать результат. Например, 0*∞.
- Деление 0/0 или ∞/∞ с любым знаком.
- Преобразование числа с плавающей запятой в формат числа с фиксированной запятой с переполнением, или когда операндом является бесконечная величина или не-число, и оно не может быть представлено в заданном формате.
- Сигнализирующая операция сравнения, если операнды неупорядочены, то есть хотя бы один из них является не-числом.
- Любая арифметическая операция над не-числами. Нужно отметить, что команды MOV не являются арифметической командой, а команды ABS и NEG рассматриваются в качестве арифметических команд.
- Извлечение квадратного корня из отрицательного числа.

Если Enables[V]=1, то устанавливается бит Cause V, и вырабатывается исключение FPU.

Если Enables[V]=0, то устанавливается бит Cause E, и вырабатывается исключение FPU.

3.3.11 Исключение E

Исключение E немаскируемое. Оно всегда вызывает исключение FPU. Исключение нереализованной операции возникает в следующих случаях:

- вышеперечисленные случаи (замаскированные исключения V или (U при FS=0));
- в качестве операнда команды используется денормализованное число.

3.3.12 Замечание по командам MADD, MSUB, NMADD, NMSUB

В 1890ВМ3Т команды MADD, MSUB, NMADD, NMSUB выполняются в соответствии со стандартом IEEE 754. То есть сначала выполняется умножение, промежуточный результат округляется (в соответствии с RM), далее выполняется сложение (или вычитание), результат округляется, далее для команд NMADD, NMSUB выполняется смена знака.

3.4 Режимы привилегий и разрядности в 1890ВМ3Т

В микропроцессоре 1890ВМ3Т существуют следующие режимы привилегий:
- режим Debug;
- режим Kernel;
- режим Supervisor;
- режим User.

Также независимо от этих режимов существуют режимы разрядности процессора – 64 и 32. (См. ниже).

3.4.1 Режим Debug

Режим Debug является отладочным режимом для механизма EJTAG. Механизм EJTAG подробно описан в отдельном документе.

Процессор находится в режиме Debug, если установлен в 1 бит DM в регистре Debug сопроцессора CP0. Данный режим имеет такие же привилегии, как и режим Kernel.

3.4.2 Режим Kernel

Процессор находится в режиме Kernel, если он не находится в режиме Debug, и выполняется хотя бы одно из следующих условий:
- бит EXL=1 в регистре Status CP0;
- бит ERL=1 в регистре Status CP0;
- поле KSU=00 в регистре Status CP0.

В данном режиме доступны все сегменты памяти (kernel, supervisor, user). Доступны регистры и команды сопроцессора CP0 (независимо от бита CU0 регистра Status CP0). Процессор работает в начальном Endian режиме независимо от бита RE регистра Status CP0.

При начальном пуске и при возникновении исключения процессор переходит в режим Kernel.

3.4.3 Режим Supervisor

Процессор находится в режиме Supervisor, если он не находится в режимах Debug и Kernel, и если поле KSU=01 регистра Status CP0.

В данном режиме доступны сегменты памяти user и supervisor. Если бит CU0=0 в регистре Status, то команды CP0 недоступны. Процессор работает в начальном Endian режиме независимо от бита RE регистра Status CP0.

3.4.4 Режим User

Процессор находится в режиме User, если он не находится в режимах Debug и Kernel, и если поле KSU=10 регистра Status CP0.

В данном режиме доступны только сегменты памяти user. Если бит CU0=0 в регистре Status, то команды CP0 недоступны. Endian режим определяется начальным режимом и битом RE регистра Status CP0.

3.4.5 Режимы разрядности процессора

В регистре Status CP0 есть биты KX, SX, UX, определяющие текущий режим разрядности процессора соответственно для режимов Kernel, Supervisor, User. Здесь важно не путать два независимых режима разрядности – режим исполнения команд и режим адресации.

Режим исполнения команд (64- или 32-разрядный) определяется текущим режимом привилегий (биты KSU, EXL, ERL) и (в случае режима User) битами UX и PX. В 64-разрядном режиме доступны все команды. В 32-разрядном – только команды, помеченные как MIPS-32 и MIPS-I в вышеприведённых таблицах команд. Попытка исполнить команду MIPS-64 приведёт к исключению Reserved Instruction или E (Unimplemented operation – для команд FPU). На само исполнение команд режим не влияет. 32-разрядные команды знаково расширяют результат до 64 разрядов. Тем самым обеспечивается 32/64-разрядная совместимость. Процессор находится в 32-разрядном режиме исполнения команд, только когда он находится в режиме User, и оба бита UX и PX сброшены. Иначе процессор находится в 64-разрядном режиме исполнения команд. В режимах Kernel и Supervisor доступны все команды.

Режим адресации независимо от режима привилегий определяется сегментом памяти, в который производится текущее обращение, и соответствующим битом KX, SX, UX. То есть, если (даже из режима Kernel) производится обращение в сегмент user, то будет действовать адресация, определяемая битом UX (а не KX).

3.5 Организация памяти

Виртуальная память 1890ВМ3Т поделена на сегменты. Сегменты различаются по доступности (kernel, supervisor, user), по способу отображения (прямое, через TLB), по политике кэшируемости (некэшируемый, кэшируемый (3 политики)). На рисунке 3.6 показано распределение виртуального адресного пространства.

Физический адрес в 1890ВМ3Т имеет разрядность 36 бит.

Существует два вида адресации – 64- и 32-разрядная. Последняя является подмножеством 64-разрядной адресации. Если из 32-разрядного режима производится попытка обращения в сегмент 64-разрядного режима (бит 31 не расширен на 63:32), то возникает исключение Address Error. Это же исключение возникает при нарушении прав привилегий.

3.5.1 Буфер трансляции адресов TLB

В 1890ВМ3Т трансляция виртуальных адресов в физические происходит двумя способами – прямое отображение и трансляция через TLB.

Прямое отображение адресов происходит путём обнуления старших битов виртуального адреса (см. описание сегментов).

1890ВМ3Т содержит буфер трансляции TLB, состоящий из 48 строк, в котором производится сопоставление виртуальной страницы с физической. Строка

TLB (рисунок 3.7) по сути имеет аналогичные поля, как в регистрах CP0 EntryHi, EntryLo0, EntryLo1, PageMask. (Описание CP0 см. ниже).

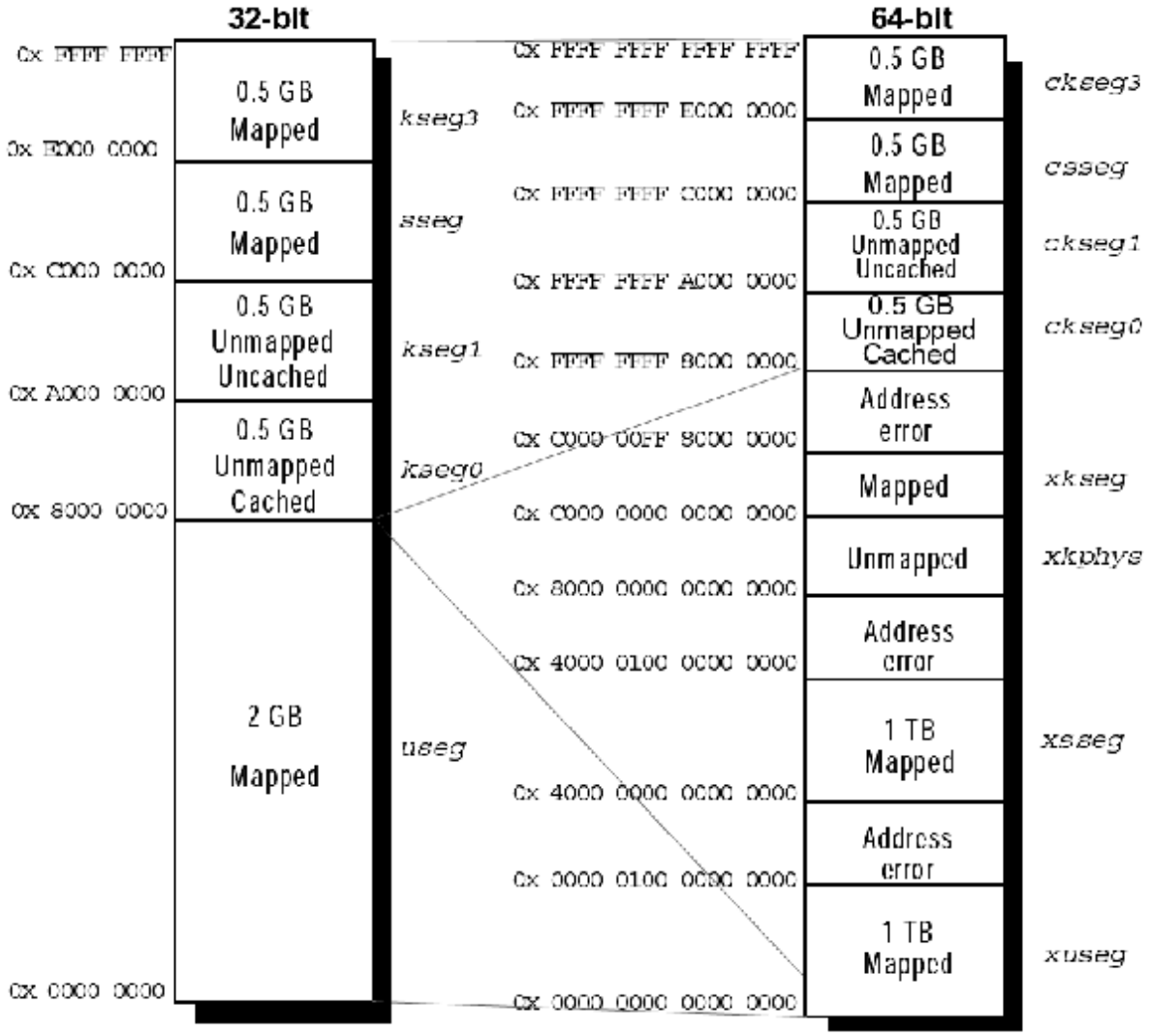


Рисунок 3.6 - Виртуальное адресное пространство 1890ВМ3Т

Mask	VPN2	G	ASID	PFN0	C0	D0	V0	PFN1	C1	D1	V1
12	29	1	8	24	3	1	1	24	3	1	1

Рисунок 3.7 - Строка TLB

Mask – маска страницы. Страницы могут быть разных размеров – от 4 кБ до 16 МБ. (См. PageMask ниже).

VPN2 – старшая часть виртуального адреса (для 4 кБ страниц – 64 бит – VAddr[63:62, 39:13], 32 бит – VAddr[31:13]). VAddr[12] (при страницах 4 кБ) используется для выбора конкретной физической страницы – PFN0 или PFN1. (Для страниц 16 кБ, 64 кБ… используются соответственно VAddr[14], VAddr[16]…).

G – бит глобальной страницы (для обеих физических страниц). При выполнении команд записи в TLB (TLBWI, TLBWR) этот бит получается путём логического «И» битов G регистров EntryLo0 и EntryLo1.

ASID – номер процесса (участвует при сравнении, если G=0).

PFN0 – чётная физическая страница. (Для 4 кБ страниц – PAddr[35:12]).

C0 – политика кэширования для чётной страницы. Описание кэш-памяти см. ниже.

Таблица 3.22 - Значения полей C0 и C1

C0	Политика
000	Write-thru, no write allocate
001	Write-thru, write allocate
010	Uncached
011	Write-back, write allocate
1xx	Резерв

D0 – бит Dirty для чётной страницы. Если 0 – страница защищена от записи.

V0 – бит действительности для чётной страницы. Если 0, то при обращении в данную страницу будет выработано исключение Mod.

PFN1, C1, D1, V1 - то же для нечётной страницы.

Подробно трансляция виртуального адреса через TLB описана в вышеуказанных документах MIPS.

3.5.2 Сегменты памяти

3.5.2.1 Сегмент useg

Доступен из всех режимов. Преобразование виртуального адреса в физический производится через TLB. Политика кэширования определяется через TLB. В 64-разрядном режиме адресации является подмножеством сегмента xuseg.

3.5.2.2 Сегмент xuseg

Доступен из Kernel, Supervisor, User при UX=1. Преобразование виртуального адреса в физический производится через TLB. Политика кэширования определяется через TLB.

3.5.2.3 Сегмент xsseg

Доступен из Kernel, Supervisor при SX=1. Преобразование виртуального адреса в физический производится через TLB. Политика кэширования определяется через TLB.

3.5.2.4 Сегмент xkphys

Доступен из Kernel при KX=1. Физический адрес получается из младших 36 разрядов виртуального адреса. Политика кэширования определяется из битов

61:59 виртуального адреса (см. таблицу 3.22). Биты VAddr[58:36] должны быть в нуле, иначе исключение Address Error.

3.5.2.5 Сегмент xkseg

Доступен из Kernel при KX=1. Преобразование виртуального адреса в физический производится через TLB. Политика кэширования определяется через TLB.

3.5.2.6 Сегмент kseg0/ckseg0

Доступен из Kernel при любом KX. Физический адрес получается из VAddr[28:0]. Политика кэширования задаётся полем K0 регистра Config0 CP0.

3.5.2.7 Сегмент kseg1/ckseg1

Доступен из Kernel при любом KX. Физический адрес получается из VAddr[28:0]. Обращение в данный сегмент является некэшируемым.

3.5.2.8 Сегмент sseg/csseg

Доступен из Kernel, Supervisor при любом SX. Преобразование виртуального адреса в физический производится через TLB. Политика кэширования определяется через TLB.

3.5.2.9 Сегмент kseg3/ckseg3

Доступен из Kernel при любом KX. Преобразование виртуального адреса в физический производится через TLB. Политика кэширования определяется через TLB.

3.6 Исключения

Исключительные ситуации подробно описаны в документе MIPS64 Architecture For Programmers. Vol. III: The MIPS64 Privileged Resource Architecture. В 1890ВМ3Т поддерживаются следующие исключения:

- Reset – холодный старт.
- SoftReset – тёплый старт.
- NMI – немаскируемое прерывание.
- EJTAG Debug – исключение EJTAG (механизму EJTAG будет посвящен отдельный документ).
- TLBRefill, XTLBRefill – непопадание в TLB.
- Int – прерывание.
- Mod – попытка записи в защищённую страницу.
- TLBL/S – повторное непопадание в TLB или страница недействительна.
- AdEL/S – неправильный адрес.
- I/DBE – ошибка шины.

- Sys – системный вызов.
- Bp – точка останова.
- RI – нереализованная инструкция.
- CpU – недоступный сопроцессор.
- Ov – целочисленное арифметическое переполнение.
- Tr – исключение Trap команд.
- FPE – исключение от FPU (E, V, Z, O, U, I).
- MCheck – аппаратная ошибка.

1890ВМ3Т имеет некоторые особенности:

- Никогда не возникает исключение CacheErr, так как не реализован механизм контроля чётности в кэшах.
- Не реализованы исключения C2E, MDMX, WATCH, так как не реализована соответствующая аппаратура.
- Исключение MCheck возникает только, если обнаружено множественное совпадение в TLB (при этом также устанавливается бит TS в регистре Status).
- Исключение DBE (Data Bus Error) реализовано как неточное. То есть при возникновении данного исключения регистр EPC может не содержать точный адрес команды загрузки, его вызвавшей. Это обусловлено тем, что при непопадании в кэш данных при загрузке начинает работать механизм Instruction Scheduling, и команда загрузки не блокирует конвейер. По регистру EPC в данном случае можно определить только примерную область возникновения DBE.

3.7 Регистры сопроцессора управления CP0

Регистры CP0 всегда доступны из режима Kernel. Они доступны из других режимов, если бит CU0=1 в регистре Status. Обмен с регистрами осуществляется через команды MTC0, MFC0, DMTC0, DMFC0. Для записи в 64-разрядный регистр также может использоваться команда MTC0. (См. описание команд в «MIPS64 Architecture For Programmers. Vol. II: The MIPS64 Instruction Set»). Для адресации некоторых регистров с одинаковым номером используется поле select в кодировке команд MTC0, MFC0, DMTC0, DMFC0.

Таблица 3.23 - Регистры CP0

Номер регистра	Название	Описание
0	Index	Используется для адресации строки TLB
1	Random	Используется для адресации строки TLB (случайное значение)
2	EntryLo0	Младшая часть строки TLB для чётных VPN
3	EntryLo1	Младшая часть строки TLB для нечётных VPN
4	Context	Указатель на запись в таблице страниц (для 32-разрядных адресов)
5	PageMask	Маска страницы TLB
6	Wired	Определяет защищённую область TLB
8	BadVAddr	Ошибочный виртуальный адрес при исключении
9	Count	Счётчик таймера
10	EntryHi	Старшая часть строки TLB
11	Compare	Значение для сравнение с таймером
12	Status	Регистр состояния
13	Cause	Регистр причины исключения
14	EPC	Адрес возврата из исключения
15	PrId	Идентификационный номер процессора
16, sel 0	Config0	Первый регистр конфигурации
16, sel 1	Config1	Второй регистр конфигурации
20	XContext	Указатель на запись в таблице страниц (для 64-разрядных адресов)
23	Debug	Регистр для EJTAG
24	DEPC	Регистр для EJTAG
25, sel 0-3	PerfCnt	Счётчики производительности
26	ErrCtl	Диагностический регистр
28, sel 0	TagLo	Младшая часть тэга кэша
28, sel 1	DataLo	Регистр данных кэша
29, sel 0	TagHi	Старшая часть тэга кэша
29, sel 1	DataHi	Старшая часть данных кэша
30	ErrorEPC	Состояние счётчика команд на момент исключения NMI
31	DESAVE	Регистр для EJTAG

При записи в регистры CP0 следует помнить о проблеме «CP0 Hazards». Эффект от записи в регистр наступает через 2-3 команды после записи. Конкретные значения приведены в таблице 3.24.

Таблица 3.24 - CP0 Hazards

Producer	→	Consumer	Hazard On	"Typical" Spacing (Cycles)
TLBWR, TLBWI	→	TLBP, TLBR	TLB entry	3
		Load/store using new TLB entry	TLB entry	3
		Instruction fetch using new TLB entry	TLB entry	5
MTC0 Status[CU]	→	Coprocessor instruction needs CU set	Status[CU]	4
MTC0 Status	→	ERET	Status	3
MTC0 Status[IE]	→	Interrupted Instruction	Status[IE]	3
TLBR	→	MFC0 EntryHi MFC0 PageMask	EntryHi, PageMask	3
MTC0 EntryLo0 MTC0 EntryLo1 MTC0 Entry Hi MTC0 PageMask MTC0 Index	→	TLBP TLBR TLBWI TLBWR	EntryLo0 EntryLo1 EntryHi PageMask Index	2
TLBP	→	MFC0 Index	Index	2
MTC0 EPC	→	ERET	EPC	2

Отдельно нужно отметить, что нельзя изменять политику кэширования сегмента kseg0 (поле K0 регистра Config0), находясь в сегменте kseg0. Команда MTC0, изменяющая K0 Config0, должна вызываться из другого сегмента. А перед переходом в сегмент kseg0 полезно выполнить команду SYNC.

3.7.1 Регистр Index (0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	0																									Index					
1	26																									6					

Рисунок 3.8 - Регистр Index

Таблица 3.25 - Поля регистра Index

Поле	Описание	Доступ	Значение после Reset
P	Устанавливается и сбрасывается командой TLBP.	R	?
Index	Используется в командах TLBP, TLBWI, TLBR. Может иметь значения от 0 до 47. Адресует строку TLB.	R/W	?

3.7.2 Регистр Random (1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0																										Random					
																										6					

Рисунок 3.9 - Регистр Random

Таблица 3.26 - Поля регистра Random

Поле	Описание	Доступ	Значение после Reset
Random	Используется в команде TLBWR. Может иметь значения от 0 до 47. Адресует строку TLB.	R	47

Уменьшается на 1 при выполнении команды TLBWR. Номер строки формируется путём преобразования значения поля Random с помощью неприводимого полинома. Если Random достигает значения Wired, то при выполнении следующей команды TLBWR в Random загружается 47. (См. Wired). При записи в регистр Wired в Random также записывается 47.

3.7.3 Регистры EntryLo0 (2) и EntryLo1(3)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		PFN																								C			D	V	G
		24																								3			1	1	1

Рисунок 3.10 - Формат регистров EntryLo

Таблица 3.27 - Поля регистров EntryLo

Поле	Описание	Доступ	Значение после Reset
PFN	Старшая часть физического адреса (для 4 кБ страниц – PAddr[35:12]). Маскируется полем Mask регистра PageMask.	R/W	?
C	Политика кэширования. 000 – Write-thru, no write allocate 001 – Write-thru, write allocate 010 – Uncached 011 – Write-back, write allocate 1xx – Резерв См. описания кэш-памяти ниже.	R/W	?
D	Бит Dirty.	R/W	?
V	Бит Valid.	R/W	?
G	Бит Global. При выполнении команд записи в TLB (TLBWI, TLBWR) бит G в строке TLB получается путём логического «И» битов G регистров EntryLo0 и EntryLo1.	R/W	?

Используются в командах TLBWI, TLBWR, TLBR.

3.7.4 Регистр Context (4)

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
PTEBase (cont.)																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTEBase									BadVPN2																			0			
41									19																						

Рисунок 3.11 - Регистр Context

Таблица 3.28 - Поля регистра Context

Поле	Описание	Доступ	Значение после Reset
PTEBase	Старшая часть адреса таблицы страниц. Назначение этого поля определяется операционной системой.	R/W	?
BadVPN2	Смещение в таблице страниц. Это поле представляет из себя отражения битов [31:13] регистра BadVAddr.	R	?

Данный регистр может использоваться операционной системой для обслуживания исключений непопадания в TLB при 32-разрядной адресации.

3.7.5 Регистр PageMask (5)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							Mask												0												
							12																								

Рисунок 3.12 - Регистр PageMask

Таблица 3.29 - Поля регистра PageMask

Поле	Описание	Доступ	Значение после Reset
Mask	Маска, определяющая размер страницы TLB: 000000000000 – 4 кБ 000000000011 – 16 кБ 000000001111 – 64 кБ 000000111111 – 256 кБ 000011111111 – 1 МБ 001111111111 – 4 МБ 111111111111 – 16 МБ Другие значения недопустимы.	R/W	?

Используется в командах TLBWI, TLBWR, TLBR.

3.7.6 Регистр Wired (6)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0																										Wired					
																										6					

Рисунок 3.13 - Регистр Wired

Таблица 3.30- Поля регистра Wired

Поле	Описание	Доступ	Значение после Reset
Wired	Определяет нижнее граничное значение номера строки TLB, в которую может быть запись по команде TLBWR	R/W	0

3.7.7 Регистр BadVAddr (8)

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
BadVAddr (cont.)																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BadVAddr																															
64																															

Рисунок 3.14 - Регистр BadVAddr

Таблица 3.31 - Поля регистра BadVAddr

Поле	Описание	Доступ	Значение после Reset
BadVAddr	Содержит виртуальный адрес, который вызвал последнее исключение Adress Error, TLBL/S, (X)TLBRefill, Mod.	R	?

3.7.8 Регистр Count (9)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Count																															
32																															

Рисунок 3.15 - Регистр Count

Таблица 3.32 - Поля регистра Count

Поле	Описание	Доступ	Значение после Reset
Count	Каждый второй такт увеличивается на 1. Когда достигает значения регистра Compare, и если разрешено прерывание от таймера (установлен бит TE в регистре Config0), то в регистре Cause устанавливается в 1 поле IP[7], и (если разрешены прерывания) возникает исключение Int. Запись в регистр Compare снимает сигнал прерывания от таймера.	R/W	?

3.7.9 Регистр EntryHi (10)

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
R		0																						VPN2(cont.)							
2																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VPN2																			0					ASID							
27																								8							

Рисунок 3.16 - Регистр EntryHi

Таблица 3.33 - Поля регистра EntryHi

Поле	Описание	Доступ	Значение после Reset
R	Старшие биты виртуального адреса (VAddr[63:62])	R/W	?
VPN2	Биты виртуального адреса – VAddr[39:13]. Маскируется полем Mask регистра PageMask.	R/W	?
ASID	Номер текущего процесса.	R/W	?

Используется командами TLBWI, TLBWR, TLBR, TLBP. ASID используется для любой трансляции адреса через TLB. Поля R и VPN2 устанавливаются битами ошибочного адреса при исключениях TLB.

3.7.10 Регистр Compare (11)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Compare																															
32																															

Рисунок 3.17 - Регистр Compare

Таблица 3.34 - Поля регистра Compare

Поле	Описание	Доступ	Значение после Reset
Compare	Значение для сравнения с регистром Count. (См. Count). Запись в Compare снимает запрос на прерывание.	R/W	?

3.7.11 Регистр Status (12)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CU3..CU0				RP	FR	RE	MX	PX	BEV	TS	SR	NMI	0		TIP	IM7..IM0								KX	SX	UX	KSU		ERL	EXL	IE
4				1	1	1	1	1	1	1	1	1			1	8								1	1	1	2		1	1	1

Рисунок 3.18 - Регистр Status

Таблица 3.35 - Поля регистра Status

Поле	Описание	Доступ	Значение после Reset
CU3..CU0	Биты, разрешающие доступ к сопроцессорам. Биты CU2 и CU3 доступны только на чтение и равны 0. При попытке доступа к неразрешённому сопроцессору возникает исключение CpU. В режиме Kernel сопроцессор CP0 всегда доступен, независимо от CU0.	R/W	?
RP	Установка бита в 1 вводит процессор в «спящий» режим, предварительно выполнив все предыдущие команды и очистив буфер записи. Процессор может быть «разбужен» только активным сигналом внешнего прерывания (или перезапуском), после чего бит аппаратно сбрасывается. Установка бита аналогична выполнению команды WAIT.	R/W	0
FR	Задаёт режим регистрового файла FPU. FR=0 – 16 64-разрядных регистров. FR=1 – 32 64-разрядных регистра.	R/W	?
RE	Reverse Endian. Если RE=1, то в режиме User Endian режим переключается на обратный.	R/W	?
MX	Всегда 0. MDMX не реализован.	R	0
PX	В режиме User разрешает выполнение 64-разрядных команд независимо от бита UX.	R/W	?
BEV	Определяет размещение векторов исключений. См. документацию MIPS.	R/W	1
TS	TLB Shutdown. Устанавливается в 1 при обнаружении множественного совпадения в TLB. При этом возникает исключение MCheck. Обработчик должен очистить TLB и сбросить этот бит.	R/W	0
SR	Устанавливается в 1 после исключения SoftReset. Устанавливается в 0 после Reset или NMI.	R	1-SoftReset 0-Reset
NMI	Устанавливается в 1 после исключения NMI. Устанавливается в 0 после Reset или SoftReset. Обработчик NMI должен записывать 0.	R/W	0
TIP	Timer Int Pending. Диагностический бит. Выставляется, когда Count=Compare (независимо от маски прерывания). Сбрасывается записью в Compare.	R	0
IM7..IM0	Маска прерываний. IM7..IM2 – внешние. (IM7 служит также маской прерывания от таймера или от PerfCounter). IM1-IM0 – программные. 0 – прерывание запрещено. 1 – прерывание разрешено. Запрещение прерываний битами ERL, EXL, IE является более приоритетным.	R/W	?
KX	Задаёт режим разрядности адресации Kernel сегментов. (См. о режимах выше). 0 – 32 бит. 1 – 64 бит.	R/W	?

Продолжение таблицы 3.35

Поле	Описание	Доступ	Значение после Reset
SX	Задаёт режим разрядности адресации Supervisor сегментов. (См. о режимах выше). 0 – 32 бит. 1 – 64 бит.	R/W	?
UX	Задаёт режим разрядности адресации User сегментов. (См. о режимах выше). 0 – 32 бит. 1 – 64 бит.	R/W	?
KSU	Задаёт текущий режим привилегий (если EXL=ERL=0). 00 – Kernel 01 – Supervisor 10 – User 11 – резерв	R/W	?
ERL	Устанавливается при особых исключениях (Reset, SoftReset, NMI). При ERL=1 запрещены прерывания, и процессор находится в режиме Kernel. (Подробнее см. документы MIPS).	R/W	1
EXL	Устанавливается при обычных исключениях. При EXL=1 запрещены прерывания, и процессор находится в режиме Kernel. (Подробнее см. документы MIPS).	R/W	?
IE	Общее разрешение прерываний (при EXL=ERL=0). 1 – разрешения прерываний определяется полем IM. 0 – прерывания запрещены.	R/W	?

3.7.12 Регистр Cause (13)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BD	0	CE		0				IV	0							IP7..IP0								0	ExcCode					0	
1		2						1	8															5							

Рисунок 3.19 - Регистр Cause

Таблица 3.36 - Поля регистра Cause

<table>
<tr><th>Поле</th><th>Описание</th><th>Доступ</th><th>Значение после Reset</th></tr>
<tr><td>BD</td><td>Если BD=1, то команда, вызвавшая исключение, находилась в слоте задержки перехода, и регистр EPC содержит адрес команды перехода (PC-4).</td><td>R</td><td>?</td></tr>
<tr><td>CE</td><td>При возникновении исключения CpU, данное поле содержит номер сопроцессора, к которому была попытка обращения.</td><td>R</td><td>?</td></tr>
<tr><td>IV</td><td>0 – Для обработки исключений типа «прерывание» используется общий вектор.
1 – Используется специальный вектор (смещение 0x200). (См. документацию MIPS).</td><td>R/W</td><td>?</td></tr>
<tr><td>IP7..IP2</td><td>Биты отражают состояния внешних запросов на аппаратное прерывания. Если соответствующие прерывания разрешены, то происходит исключение. Обработчик должен «сообщить» внешним устройствам, что прерывание обслужено, чтобы те сняли запросы. Иначе исключение произойдёт снова по возвращению из обработчика.
Если разрешены прерывания от таймера Count (бит TE в регистре Config0) или прерывания от счётчиков производительности PerfCount (биты IE в соответствующих PerfCount), то IP7 также используется для этих прерываний.</td><td>R</td><td>?</td></tr>
<tr><td>IP1..IP0</td><td>Программные прерывания. Запись 1 в данные биты (если прерывания не замаскированы) вызывает исключение. Обработчик должен программно записать нули в эти биты для снятия запросов на прерывания.</td><td>R/W</td><td>?</td></tr>
<tr><td>ExcCode</td><td>Определяет вид исключения (при обычных иск лючениях).
<table>
<tr><td>0</td><td>Int</td><td>Прерывание</td></tr>
<tr><td>1</td><td>Mod</td><td>Запись в защищённую страницу</td></tr>
<tr><td>2</td><td>TLBL</td><td>Непопадание в TLB при загрузке</td></tr>
<tr><td>3</td><td>TLBS</td><td>Непопадание в TLB при записи</td></tr>
<tr><td>4</td><td>AdEL</td><td>Неправильный адрес при загрузке</td></tr>
<tr><td>5</td><td>AdES</td><td>Неправильный адрес при записи</td></tr>
<tr><td>6</td><td>IBE</td><td>Ошибка шины при выборке команд</td></tr>
<tr><td>7</td><td>DBE</td><td>Ошибка шины при чтении данных</td></tr>
<tr><td>8</td><td>Sys</td><td>Команда SYSCALL</td></tr>
<tr><td>9</td><td>Bp</td><td>Команда BREAK</td></tr>
<tr><td>10</td><td>RI</td><td>Нереализованная команда</td></tr>
<tr><td>11</td><td>CpU</td><td>Недоступный сопроцессор</td></tr>
<tr><td>12</td><td>Ov</td><td>Целочисленное переполнение</td></tr>
<tr><td>13</td><td>Tr</td><td>Команды Trap</td></tr>
<tr><td>15</td><td>FPE</td><td>Исключение от FPU</td></tr>
<tr><td>24</td><td>MCheck</td><td>TLB Shutdown</td></tr>
</table></td><td>R</td><td>?</td></tr>
</table>

Регистр Cause прописывается при возникновении исключения. В нём содержится информация о конкретном исключении. Подробнее см. документацию MIPS.

3.7.13 Регистр EPC (14)

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
EPC (cont.)																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPC																															
64																															

Рисунок 3.20 - Регистр EPC

Таблица 3.37 - Поля регистра EPC

Поле	Описание	Доступ	Значение после Reset
EPC	Адрес команды, вызвавшей исключение. Прописывается при возникновении обычного исключения (первого уровня). Если BD=1 в регистре Cause, то EPC указывает на команду перехода, предыдущую команде, вызвавшей исключение.	R/W	?

3.7.14 Регистр PrID (15)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Company Options								Company ID								Processor ID								Revision							
0x12								0x34								0x81								0x01							
8								8								8								8							

Рисунок 3.21 - Регистр PrID

Таблица 3.38 - Поля регистра PrID

Поле	Описание	Доступ	Значение после Reset
Company Options	Поле специфической информации компании.	R	0x12
Company ID	Идентификатор компании.	R	0x34
Processor ID	Идентификатор процессора. (Аналогичен процессорному ядру MIPS 5K).	R	0x81
Revision	Номер ревизии процессора.	R	0x01

MIPS не рекомендует использовать данный регистр для определения возможностей и конфигурации процессора. Всю необходимую информацию можно получить из регистров Config0 и Config1.

3.7.15 Регистр Config0 (16, select 0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M	Impl															BE	AT		AR			MT			0			VI	K0		
	TE		WP		BW	ADP		0			ISD	0		TR	DID	BM															

Рисунок 3.22 - Регистр Config0

Таблица 3.39 - Поля регистра Config0

Поле	Описание	Доступ	Значение после Reset
M	Указывает, что есть регистр Config1.	R	1
Impl	Набор полей, специфичных для 1890ВМ3Т.	-	-
TE	Timer Interrupt Enable. Если TE=1, то IP7 в регистре Cause используется для прерывания от таймера (регистры Compare, Count).	R/W	0
WP	Write Pattern. Определяет, как будут подаваться данные на внешней шине при записи блока слов. (См. описание внешнего интерфейса). 00 – DDDD (WWWWWWWW) 01 – DxDxDxDx (WxWxWxWxWxWxWxWx) 10 – DxxDxxDxxDxx (Wxx*8) 11 – Dxxx*4 (Wxxx*8) (по умолчанию)	R/W	3
BW	Bus Width. Определяет ширину внешней шины. 0 – 64 бит. 1 – 32 бит. Данный режим не имеет никакой связи с режимом адресации и исполнения команд (32/64).	R	Задаётся аппаратно при Reset
ADP	Address/Data Pattern. Определяет количество шинных тактов между выдачей адреса и выдачей данных на шине при записи. 00 – AD 01 – AxD 10 – AxxD 11 – AxxxD (по умолчанию)	R/W	3
ISD	Instruction Scheduling Disable. Выключает механизм Instruction Scheduling. 0 – Enabled. 1 – Disabled.	R/W	0
BBL	Включение улучшенного механизма предсказания. 0-выкл. 1-вкл. Желательно установить в 1.	R/W	0
TR	TLB Reset. Запись 1 очищает TLB (начальное состояние после Reset). Для продолжения работы с TLB нужно записать 0.	R/W	0
DID	Dual Issue Disable. Выключает механизм Dual Issue. 0 – Enabled. 1 – Disabled.	R/W	0
BM	Burst Mode. Определяет очерёдность подачи данных на шине. 1890ВМ3Т поддерживает только Sub-block ordering. (См. описание внешнего интерфейса). 0 – Sequential ordering. 1 – Sub-block ordering.	R	1
BE	Big-Endian. Определяет основной Endian режим работы процессора. 0 – Little-endian. 1 – Big-endian.	R	Задаётся аппаратно при Reset

Продолжение таблицы 3.39

<table>
<tr><th>Поле</th><th>Описание</th><th>Доступ</th><th>Значение после Reset</th></tr>
<tr><td>AT</td><td>Определяет архитектуру процессора
0 – MIPS32
1 – MIPS64 с 32-разрядной адресацией
2 – MIPS64 (для 1890ВМ3Т)
3 – резерв.</td><td>R</td><td>2</td></tr>
<tr><td>AR</td><td>Определяет ревизию архитектуры (Release).
0 – Release 1 (для 1890ВМ3Т)
1 – Release 2
2-7 – резерв.</td><td>R</td><td>0</td></tr>
<tr><td>MT</td><td>Определяет организацию памяти.
0 – нет
1 – обычная TLB (для 1890ВМ3Т)
2 – BAT (см. документы MIPS)
3 – фиксированное отображение
4-7 – резерв.</td><td>R</td><td>1</td></tr>
<tr><td>VI</td><td>Определяет тип кэша команд.
0 – кэш физических адресов
1 – кэш виртуальных адресов.</td><td>R</td><td>0</td></tr>
<tr><td>K0</td><td>Определяет политику кэширования для сегмента kseg0/ckseg0.
<table>
<tr><td>000</td><td>Write-thru, no write allocate</td></tr>
<tr><td>001</td><td>Write-thru, write allocate</td></tr>
<tr><td>010</td><td>Uncached</td></tr>
<tr><td>011</td><td>Write-back, write allocate</td></tr>
<tr><td>1xx</td><td>Резерв</td></tr>
</table></td><td>R/W</td><td>2</td></tr>
</table>

3.7.16 Регистр Config1 (16, select 1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M	TLBSize						IS			IL			IA			DS			DL			DA			C2	MD	PC	WR	CA	EP	FP
1	6						3			3			3			3			3			3			1	1	1	1	1	1	1

Рисунок 3.23 - Регистр Config1

Таблица 3.40 а - Поля регистра Config1

Поле	Описание	Доступ	Значение после Reset
M	Указывает, что отсутствуют регистры Config2 и Config3.	R	0
TLBSize	Определяет размерность TLB (кол-во строк – 1).	R	47
IS	Определяет кол-во строк в кэше команд. 0 – 64 1 – 128 (для 1890ВМ3Т) 2 – 256 3 – 512 4 – 1024 5 – 2048 6 – 4096 7 – резерв	R	1
IL	Определяет размер строки кэша команд 0 – нет кэша 1 – 4 байта 2 – 8 байт 3 – 16 байт 4 – 32 байта (для 1890ВМ3Т) 5 – 64 байта 6 – 128 байт 7 – резерв	R	4
IA	Определяет ассоциативность кэша команд 0 – кэш прямого отображения 1 – 2-множественный 2 – 3-множественный 3 - 4-множественный (для 1890ВМ3Т) 4 - 5-множественный 5 - 6-множественный 6 - 7-множественный 7 - 8-множественный	R	3
DS	Определяет кол-во строк в кэше данных 1 – 128 (для 1890ВМ3Т)	R	1
DL	Определяет размер строки кэша данных 4 – 32 байта (для 1890ВМ3Т)	R	4
DA	Определяет ассоциативность кэша данных 3 - 4-множественный (для 1890ВМ3Т)	R	3
C2	Определяет присутствие сопроцессора CP2. В 1890ВМ3Т CP2 отсутствует.	R	0
MD	Определяет наличие MDMX. В 1890ВМ3Т MDMX отсутствует.	R	0

Продолжение таблицы 3.40 а

Поле	Описание	Доступ	Значение после Reset
PC	Определяет наличие счётчиков производительности (PerfCnt). В 1890ВМ3Т есть счётчики производительности.	R	1
WR	Определяет наличие Watch регистров. В 1890ВМ3Т Watch регистры отсутствуют.	R	0
CA	MIPS16e Code compression. Отсутствует в 1890ВМ3Т	R	0
EP	Наличие EJTAG. Присутствует в 1890ВМ3Т. Информацию по EJTAG см. в отдельном документе.	R	1
FP	Наличие FPU. Присутствует в 1890ВМ3Т.	R	1

3.7.17 Регистр XContext (20)

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
PTEBase																															R
31																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BadVPN2																											0			
2	27																														

Рисунок 3.24 - Регистр XContext

Таблица 3.40 б - Поля регистра XContext

Поле	Описание	Доступ	Значение после Reset
PTEBase	Старшая часть адреса таблицы страниц. Назначение этого поля определяется операционной системой.	R/W	?
R	Старшие биты неправильного виртуального адреса BadVAddr[63:62].	R	?
BadVPN2	Смещение в таблице страниц. Это поле представляет из себя отражения битов [39:13] регистра BadVAddr.	R	?

Данный регистр может использоваться операционной системой для обслуживания исключений непопадания в TLB при 64-разрядной адресации.

3.7.18 Регистр Debug (23)

Данный регистр используется механизмом EJTAG. Описание EJTAG находится в отдельном документе.

3.7.19 Регистр DEPC (24)

Данный регистр используется механизмом EJTAG. Описание EJTAG находится в отдельном документе.

3.7.20 Регистры PerfCnt (25, select 0-3)

В 1890ВМ3Т реализовано 2 счётчика производительности. Счётчики производительности независимо друг от друга могут считать разные системные события, такие как количество тактов, количество выполненных инструкций, количество непопаданий в кэш и т. д. Также в случае переполнения счётчики могут вызывать прерывание (поле IP7 в регистре Cause).

Для каждого счётчика отводится пара регистров – регистр управления счётчиком и регистр счётчика.

Таблица 3.41 - Назначение select для PerfCnt

Поле select	Регистр
0	Регистр управления счётчиком 0
1	Счётчик 0
2	Регистр управления счётчиком 1
3	Счётчик 1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M	0																						Event				IE	U	S	K	EXL
1																							4				1	1	1	1	1

Рисунок 3.25 - Формат регистров управления PerfCnt

Таблица 3.42 - Поля регистров управления

Поле	Описание	Доступ	Значение после Reset
M	Показывает, существует ли счётчик со следующим номером	R	1 для счётчика 0 0 для счётчика 1
Event	Тип подсчитываемых событий (см. таблицу 3.44)	R/W	?
IE	Interrupt Enable. Если 1, то выдаётся запрос на прерывание (IP7), если бит 31 регистра счётчика равен 1.	R/W	0
U	Если 1, то в режиме User включён подсчёт событий	R/W	?
S	Если 1, то в режиме Supervisor включён подсчёт событий	R/W	?
K	Если 1, то в режиме Kernel включён подсчёт событий	R/W	?
EXL	Если 1, то при EXL=1 включён подсчёт событий	R/W	?

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Event Count																															
32																															

Рисунок 3.26 - Формат счётных регистров PerfCnt

Таблица 3.43 - Поля счётных регистров

Поле	Описание	Доступ	Значение после Reset
Event Count	Счётчик событий. Если бит 31 равен 1, и IE=1, то выставляется запрос на прерывание IP7.	R/W	?

Таблица 3.44 - Типы подсчитываемых событий (поля Event)

Поле Event	Счётчики 0/1
0	Такты
1	Выполненные команды
2	Команды загрузки/sync/cache
3	Команды сохранения
4	Команды ветвления
5	Команды FPU
6	Исключения (X)TLBMiss
7	Непопадание в ITLB
8	Непопадание в DTLB
9	Непопадание в кэш команд
10	Непопадание в кэш данных
11	Неправильно предсказанный переход
12	Dual Issue. 2 команды за такт
13	Резерв
14	Резерв
15	Резерв

3.7.21 Регистр ErrCtl (26)

Данный регистр является специфическим для 1890ВМ3Т и содержит биты, необходимые только для диагностики.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0																									VG	?			DMH	IMH	WST
																									1				1	1	1

Рисунок 3.27 - Регистр ErrCtl

Таблица 3.45 - Поля регистра ErrCtl

Поле	Описание	Доступ	Значение после Reset
VG	Global Valid. После выполнения команды TLBR сюда записывается скрытый бит валидности соответствующей строки TLB.	R/W	?
DMH	Data Cache Multi-hit. Устанавливается в 1, если произошло множественное попадание в кэш данных. Не должно возникать при нормальной работе.	R/W	?
IMH	Instruction Cache Multi-hit. То же для кэша команд.	R/W	?
WST	Way Select Test. Запись в WST 1 включает особый режим диагностики, при котором команда CACHE будет выполнять специальные операции. (См. ниже).	R/W	0

3.7.22 Регистр TagLo (28, select 0)

Регистр TagLo используется для команды CACHE (см. ниже). В зависимости от конкретной операции команды CACHE регистр может иметь разный формат.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						L	V	Tag																							
						1	1	24																							

Рисунок 3.28 - Формат регистра TagLo при обычных операциях CACHE

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						?																				WS					
																										6					

Рисунок 3.29 а - Формат регистра TagLo при специальных операциях CACHE для кэша команд

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						?								Dirty				?								WS					
														4												6					

Рисунок 3.29 б - Формат регистра при специальных операциях CACHE для кэша данных

Таблица 3.46 - Поля регистра

Поле	Описание	Доступ	Значение после Reset
L	Locking. Если L=1, строка кэша замкнута (locked), т.е. есть гарантия, что это строка никогда не будет заменена на другую.	R/W	?
V	Valid. Если V=1, то строка содержит действительные данные.	R/W	?
Tag	Тэг строки кэша. Представляет из себя биты [35:12] физического адреса.	R/W	?
WS	Строка памяти Way Select (для диагностических целей).	R/W	?
Dirty	Биты Dirty[3:0] для строк четырёх множеств кэша (данных/команд) (для диагностических целей). Используются при политике кэша Write-back.	R/W	?
?	При выполнении специальных операций команды CACHE данные биты игнорируются (при записи в память) и могут содержать непредсказуемые значения (при чтении из памяти)	R/W	?

3.7.23 Регистр DataLo (28, select 1)

Регистр DataLo используется для команды CACHE (см. ниже).

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
Data (cont.)																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
64																															

Рисунок 3.30 - Регистр DataLo

Таблица 3.47 - Поля регистра DataLo

Поле	Описание	Доступ	Значение после Reset
Data	Двойное слово из кэш-памяти.	R/W	?

3.7.24 Регистр TagHi (29, select 0)

Данный регистр всегда содержит 0.

3.7.25 Регистр DataHi (29, select 1)

Регистр DataHi используется для команды CACHE (см. ниже).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DataHi																															
32																															

Рисунок 3.31 - Регистр DataHi

Таблица 3.48 - Поля регистра DataHi

Поле	Описание	Доступ	Значение после Reset
DataHi	Старшая часть двойного слова из кэш-памяти. Регистр DataHi является отображением старших битов регистра DataLo[63:32]. Данный регистр реализован для совместимости с 32-разрядным режимом, в котором недоступна команда DMFC0.	R/W	?

3.7.26 Регистр ErrorEPC (30)

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
ErrorEPC (cont.)																															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ErrorEPC																															
64																															

Рисунок 3.32 - Регистр ErrorEPC

Таблица 3.49 - Поля регистра ErrorEPC

Поле	Описание	Доступ	Значение после Reset
ErrorEPC	Адрес команды, вызвавшей исключение NMI. Прописывается при возникновении специального исключения (второго уровня). В отличие от случая с регистром EPC, бит BD в регистре Cause не устанавливается. В случае, если исключение NMI произошло во время исполнения команды, находящейся в слоте задержки перехода (Branch Delay Slot), то в ErrorEPC записывается адрес команды перехода.	R/W	?

3.7.27 Регистр DESAVE (31)

Данный регистр используется механизмом EJTAG. Описание EJTAG находится в отдельном документе.

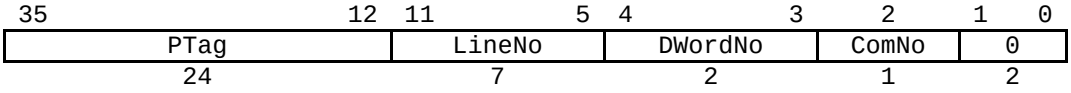
3.8 Кэш команд и кэш данных

3.8.1 Кэш-память команд

Кэш-память команд (рисунок 3.34) представляет собой 4-множественно ассоциативный кэш размером 16 кБ.

Кэш содержит 4 множества тегов-данных (0, 1, 2, 3) по 128 строк. Каждая строка каждого множества состоит из тега и 4-х двойных слов разрядностью 64 бита (8 команд). Тэг содержит старшую часть 36-разрядного физического адреса (биты [35:12]) и биты управления. Формат тэга соответствует формату регистра TagLo (см. выше) для обычных операций команды CACHE.

Таким образом, биты физического адреса команды имеют следующие назначения.



PTag	старшая часть адреса, сравниваемая с полем PTag в кэше
LineNo	адресация конкретной строки в кэше
DWordNo	адресация конкретного двойного слова в строке
ComNo	адресация конкретной команды в двойном слове (см. влияние Big/Little-Endian)
0	адрес команды должен быть выровнен по 4

Рисунок 3.33 - Формат физического адреса команды применительно к кэшу

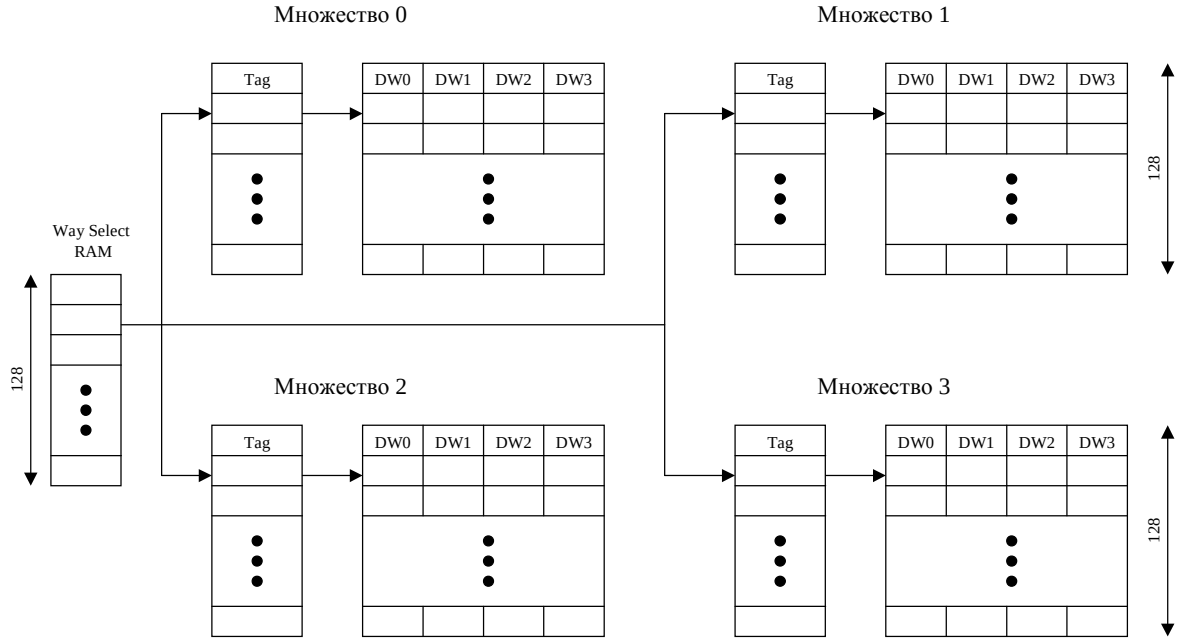


Рисунок 3.34 - Архитектура кэша команд

3.8.2 Работа кэша команд

При обращении процессора за очередной командой в кэшируемую область памяти происходит следующее. В кэше команд выбираются строки из всех множеств (0, 1, 2, 3), адресуемые полем LineNo физического адреса требуемой команды. Далее анализируются биты V всех четырёх строк. В тех строках, в которых V=1, происходит сравнение полей PTag строк с полем PTag физического адреса требуемой команды. Если имеет место совпадение (при нормальной работе кэша не должно быть множественных совпадений, иначе устанавливается бит IMH/DMH в регистре ErrCtl), то из конкретной строки выбирается команда, адресуемая полями DWordNo и ComNo (см. влияние Big/Little-Endian). Также при совпадении обновляется соответствующая строка в памяти Way Select (см. ниже). Если совпадения нет, или все биты V=0, то начинается процесс заполнения строки кэша из памяти.

При заполнении кэша должна быть выбрана одна строка из четырёх множеств (0, 1, 2 или 3). В данную строку будет загружено 4 двойных слова из памяти, в тег будет прописано значение поля PTag физического адреса, и бит V будет установлен в 1. Выбор строки из четырёх множеств осуществляется с помощью механизма Way Select (см. ниже). Строки, в тэгах которых установлен бит L, считаются замкнутыми (Locked) и выбору не подлежат. Таким образом в 1890ВМ3Т реализован механизм построчного Cache Locking.

3.8.3 Механизм Way Select

В кэшах команд и данных процессора реализованы дополнительные памяти WaySelect. Она используется механизмом WaySelect для выбора конкретной строки из четырёх множеств при замещении строки в кэше. Механизм абсолютно прозрачен программно. Доступ к этой памяти осуществляется только в

диагностических целях. Начальному загрузчику нужно только обнулить памяти WaySelect для кэшей команд и данных.

Память WaySelect представляет из себя 128 строк по 6 бит. Каждая строка WaySelect соответствует строке кэш-памяти (всех 4-х множеств). 6-разрядное слово из памяти WaySelect определяет очерёдность выбора строки конкретного множества. При попадании в кэш происходит соответствующее обновление строки памяти WS. В таблице 3.50 показаны возможные значения строки Way Select. При обнаружении запрещённого значения устанавливается бит WSE в регистре ErrCtl.

Таблица 3.50 - Допустимые значения Way Select

Selection Order	WS[5:0]	Selection Order	WS[5:0]
0123	000000	2013	100010
0132	000001	2031	110010
0213	000010	2103	100110
0231	010010	2130	101110
0312	010001	2301	111010
0321	010011	2310	111110
1023	000100	3012	011001
1032	000101	3021	011011
1203	100100	3102	011101
1230	101100	3120	111101
1302	001101	3201	111011
1320	101101	3210	111111

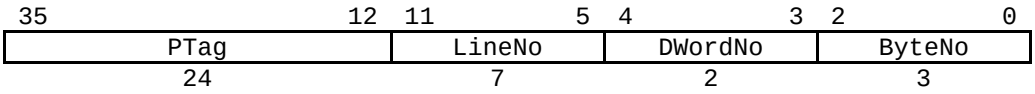
3.8.4 Влияние Big/Little-Endian на кэш команд

Режимы Big- и Little-Endian не оказывают влияния на кэш команд. Так как загрузка кэша происходит двойными словами, и в одном двойном слове содержатся две команды, то может возникнуть вопрос, какая команда по исполнению первая (младший адрес), какая вторая (старший адрес). В 1890ВМ3Т строго определён порядок хранения команд в кэше команд независимо от режима. В старшем слове хранится команда с младшим адресом (первая), в младшем – со старшим адресом (вторая). «Переворот» команд происходит во время заполнения кэша команд. В случае Big-Endian «переворота» не происходит, в Little-Endian – команды меняются местами.

3.8.5 Кэш данных

Кэш данных имеет сходные характеристики с кэшем команд. Отличие заключается в наличии битов Dirty в тегах. При замещении действительной строки (V=1), у которой Dirty=1, осуществляется её выгрузка во внешнюю память перед заполнением данной строки новыми данными. Бит Dirty устанавливается, если в

кэш данных осуществляется запись, и на данную область памяти действует политика кэшированя Write-Back (см. ниже).



PTag	старшая часть адреса, сравниваемая с полем PTag в кэше
LineNo	адресация конкретной строки в кэше
DWordNo	адресация конкретного двойного слова в строке
ByteNo	адресация конкретного байта в двойном слове (с учётом Endian режима)

Рисунок 3.35 - Формат адреса данных по отношению к КЭШу

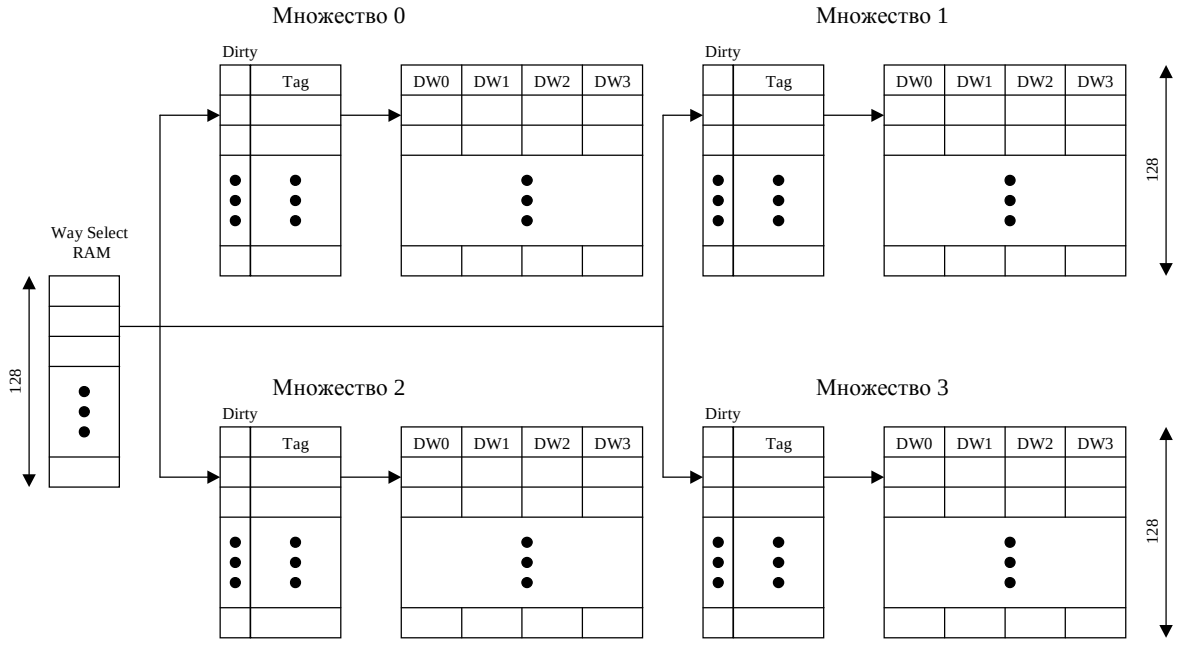


Рисунок 3.36 - Архитектура кэша данных

3.8.6 Политики кэширования для кэша данных

Политики кэширования определяют особенности функционирования кэша данных и порядок загрузки/выгрузки данных из кэша во внешнюю память. В 1890ВМ3Т реализовано 3 политики кэширования.

3.8.7 Write-thru with write allocate

При данной политике действуют следующие правила:
1) При непопадании в кэш при чтении строка загружается в кэш.
2) При попадании в кэш при записи строка в кэше модифицируется, и одновременно записываемое слово перемещается в буфер записи для последующей записи во внешнюю память.
3) При непопадании в кэш при записи слово помещается в буфер записи, далее буфер записи очищается, и загружается строка в кэш.

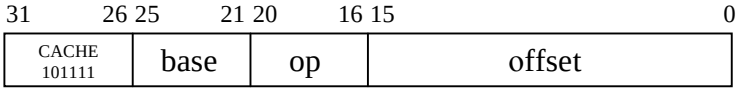
3.8.8 Write-thru with no write allocate

При данной политике действуют следующие правила:
1) При непопадании в кэш при чтении строка загружается в кэш.
2) При попадании в кэш при записи строка в кэше модифицируется, и одновременно записываемое слово перемещается в буфер записи для последующей записи во внешнюю память.
3) При непопадании в кэш при записи слово помещается в буфер записи. Чтение строки в кэш из внешней памяти не инициируется.

3.8.9 Write-back (with write allocate)

При данной политике действуют следующие правила:
1) При непопадании в кэш при чтении строка загружается в кэш.
2) При попадании в кэш при записи строка в кэше модифицируется, и устанавливается бит Dirty. Если при последующих операциях строка будет подлежать замене, то она будет выгружена во внешнюю память.
3) При непопадании в кэш при записи строка загружается в кэш. Слово записывается в кэш, и устанавливается бит Dirty.

3.8.10 Команда CACHE



Формат:
CACHE op,offset(base)

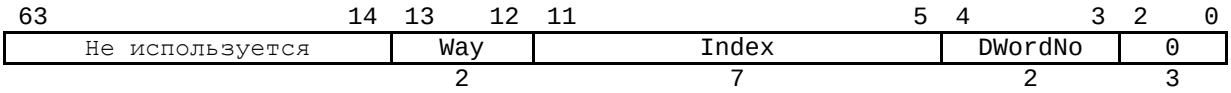
Рисунок 3.37 - Формат команды CACHE

Содержимое поля offset расширяется с учётом знака и складывается с содержимым регистра base. Выполняется специальная операция над кэшем команд или данных. Операция может интерпретировать полученный операнд либо как виртуальный адрес (который транслируется в физический), либо как индекс в кэш памяти. При трансляции виртуального адреса могут возникнуть соответствующие

исключения. При выполнении операции с адресом некэшируемой области результат команды не определен.

Команда относится к командам CP0. При её попытке выполнения в режиме User или Supervisor при сброшенном бите CU[0] в регистре Status произойдёт исключение Coprocessor Unusable.

Если операнд команды трактуется как индекс, то он имеет следующий формат (рисунок 3.38).



Way	номер множества (0, 1, 2, 3)
Index	номер строки кэша
DWordNo	адресация конкретного двойного слова в строке
0	адрес двойного слова должен быть выровнен по 8

Рисунок 3.38 - Формат индекса

Поле op команды CACHE определяет конкретную операцию. Биты [17:16] определяют тип кэш-памяти, над которой совершается операция. Биты [20:18] – тип операции. Тип операции зависит от бита WST в регистре ErrCtl. При WST=0 команда CACHE выполняет стандартные операции, предусмотренные архитектурой MIPS64. При WST=1 команда CACHE выполняет специальные диагностические операции.

Таблица 3.51 - Кодировка типа кэш-памяти (биты [17:16])

Биты [17:16]	Обозначение	Тип кэш-памяти
00	I	Кэш команд
01	D	Кэш данных
10	-	
11	-	

Таблица 3.52 - Кодировка типа операции для обычного режима (WST=0)

Биты [20:18]	Кэш	Название операции	Тип операнда
000	I	Index Invalidate	Индекс
	D	Index Write-Back Invalidate	
001	I	Index Load Tag/Data	Индекс
	D		
010	I	Index Store Tag	Индекс
	D		
011	-	NOP	-
100	I	Hit Invalidate	Адрес
	D		
101	I	Fill	Адрес
	D	Hit Write-Back Invalidate	Адрес
110	D	Hit Write-Back	Адрес
	I	NOP	-
111	I	Fetch And Lock	Адрес
	D		

Таблица 3.53 - Кодировка типа операции для диагностического режима (WST=1)

Биты [20:18]	Кэш	Название операции	Тип операнда
001	I	Index Load WS & BrRAM	Индекс
	D	Index Load WS & Dirty	
010	I	Index Store WS & BrRAM	Индекс
	D	Index Store WS & Dirty	
011	I	Index Store Tag/Data	Индекс
	D		

Index Invalidate (I-Cache). Очищаются биты V и L в строке определённого множества кэша команд, адресуемой индексом. Остальные поля тэгов и данные не изменяются.

Index Write-Back Invalidate (D-Cache). Очищаются биты V и L в строке определённого множества кэша данных, адресуемой индексом. Остальные поля тэгов и данные не изменяются. Если в строке были одновременно установлены биты V и Dirty, то перед очисткой строка кэша выгружается во внешнюю память. Адрес выгрузки формируется из содержимого тэга.

Index Load Tag/Data. Из указанной строки указанного множества читается тэг и управляющие биты в регистр TagLo (обычный формат – см. выше). В регистр DataLo (и DataHi) читается указанное двойное слово из строки кэша.

Index Store Tag. Из регистра TagLo (обычный формат – см. выше) записывается тэг и управляющие биты в указанную строку кэша указанного множества. Слово данных не записывается. Для записи слова нужно выполнить диагностическую операцию Index Store Tag/Data при WST=1.

Hit Invalidate. Если происходит попадание в кэш по указанному адресу, то очищаются биты V и L выбранной строки. Остальные данные кэша не изменяются. В случае кэша данных запись в память строки с установленным битом Dirty не производится.

Fill (I-Cache). Если происходит непопадание в кэш команд по указанному адресу, то происходит заполнение строки кэша команд данными из внешней памяти. Множество определяется по механизму Way Select.

Hit Write-Back Invalidate (D-Cache). Если происходит попадание в кэш данных по указанному адресу, то очищаются биты V и L выбранной строки (одной). Остальные данные кэша не изменяются. Если в тэге выбранной строки был установлен бит Dirty, то перед очисткой строка кэша выгружается во внешнюю память. Адрес выгрузки формируется из содержимого тэга.

Hit Write-Back (D-Cache). Если происходит попадание в кэш данных по указанному адресу, и в тэге выбранной строки установлен бит Dirty, то строка выгружается во внешнюю память. Адрес выгрузки формируется из содержимого тэга. Бит Dirty очищается. Остальные данные кэша не изменяются.

Fetch and Lock. Если происходит непопадание в кэш, то из памяти читается строка и записывается в кэш (по механизму Way Select). Устанавливается бит L. Если (для кэша данных) для старой заменяемой строки был установлен бит Dirty, то старая строка предварительно выгружается. В случае попадания в кэш просто устанавливается бит L для данной строки.

Index Load WS & BrRAM (I-Cache). В регистр TagLo (специальный формат для кэша команд – см. выше) загружается слово из памяти Way Select (адресуется

полем Index индекса – биты [11:5]) и слово из памяти Branch Prediction RAM (адресуется битами [12:3] индекса). Данная операция используется только для диагностики.

Index Load WS & Dirty (D-Cache). В регистр TagLo (специальный формат для кэша данных – см. выше) загружается слово из памяти Way Select и четыре бита Dirty из строк четырёх множеств (обе памяти адресуются полем Index индекса – биты [11:5]). Данная операция используется только для диагностики.

Index *Store WS & BrRAM (I-Cache).* Из регистра TagLo (специальный формат для кэша команд – см. выше) загружается слово в память Way Select (адресуется полем Index индекса – биты [11:5]) и слово в память Branch Prediction RAM (адресуется битами [12:3] индекса). Данная операция используется только для диагностики.

Index Store WS & Dirty (D-Cache). Из регистра TagLo (специальный формат для кэша данных – см. выше) загружается слово в память Way Select и четыре бита Dirty в строки четырёх множеств (обе памяти адресуются полем Index индекса – биты [11:5]). Данная операция используется только для диагностики.

Index Store Tag/Data. Операция аналогична Index Store Tag, но при этом записываются ещё и данные из регистров DataLo/DataHi в кэш.

3.9 Необходимая программная инициализация

Для нормальной работы процессора программному обеспечению необходимо выполнить следующие действия по инициализации:

а) Очистить кэш команд и кэш данных (сбросить биты V).

б) Очистить памяти Way Select для кэшей команд и данных (записать 0).

в) TLB в программной инициализации не нуждается.

г) Инициализировать надлежащим образом регистры Config0, Status, ErrCtl, Cause, Wired, регистры EJTAG.

д) Инициализировать регистр управления FPU.

3.10 Отличия 1890ВМ3Т от прототипа IDT79RC64475

3.10.1 Система команд

Микропроцессор 1890ВМ3Т поддерживает систему команд MIPS64 Release 1, которая является надмножеством системы команд MIPS-III, реализованной в прототипе. То есть 1890ВМ3Т поддерживает все команды прототипа, а также имеет дополнительные команды MIPS64. Обзор команд приведён в подразделе 3.2. Подробное описание команд содержится в документе MIPS64 Architecture For Programmers. Vol. II: The MIPS64 Instruction Set (rev 1.0).

3.10.2 Регистр управления сопроцессора вещественной арифметики FCSR

Регистр FCSR (FCR31) совместим снизу вверх от прототипа к 1890ВМ3Т. В регистре FCSR 1890ВМ3Т введены дополнительные биты условий (C1-C7), предусмотренные архитектурой MIPS64. Также в 1890ВМ3Т к FCSR можно

обращаться по другим номерам (выборочное обращение к полям). См. подраздел 3.3.

3.10.3 Регистры сопроцессора системного управления CP0

1890ВМ3Т отличается от прототипа по регистрам CP0 в следующем:

а) Отличаются регистры Status.

б) Регистры Cause имеют совместимость снизу вверх (бит IV).

в) Отличаются регистры Config0. В 1890ВМ3Т введён регистр Config1.

г) В 1890ВМ3Т отсутствует регистр LLAddr.

д) В 1890ВМ3Т реализованы регистры EJTAG – регистры Debug, DEPC, DESAVE.

е) В 1890ВМ3Т введены регистры счётчиков производительности PerfCnt0-PerfCnt3.

ж) Отличаются регистры ErrCtl (ECC).

з) В 1890ВМ3Т отсутствует регистр CacheErr.

и) Отличаются регистры TagLo/TagHi. В 1890ВМ3Т введены регистры DataLo/DataHi.

Регистры CP0 процессора 1890ВМ3Т подробно описаны в подразделе 3.7.

3.10.4 Конвейеры целочисленной и вещественной арифметики

В отличие от 5-ступенчатого целочисленного конвейера, реализованного в прототипе, 1890ВМ3Т содержит 6-ступенчатый конвейер. Введена стадия Dispatch (между стадиями Instruction Fetch и Register Fetch). Данная стадия позволяет обеспечить более эффективный доступ к кэш-памяти команд и реализацию механизма Limited Dual-Issue (2 команды за такт).

Механизм Limited Dual-Issue позволяет исполнять одновременно до 2-х команд за такт при условии, что одна команда – целочисленная (или команда загрузки/сохранения), а другая команда – вещественная. При использовании оптимизирующего компилятора данный механизм позволяет увеличить производительность до 2 раз на задачах вещественной арифметики.

В 1890ВМ3Т также предполагается реализация механизма Instruction Scheduling, который позволяет не блокировать конвейер в случае непопадания в кэш, если результат команды загрузки не требуется немедленно. При использовании специального оптимизирующего компилятора этот механизм позволит увеличить производительность.

Времена исполнения команд (в тактах конвейера) перечислены в таблице 3.14 и таблице 3.15.

3.10.5 Архитектура кэш-памятей

Архитектура кэшей 1890ВМ3Т отличается от прототипа, хотя с программной точки зрения отличия прозрачны. Кэш команд и кэш данных 1890ВМ3Т также имеют объём по 16 кБ как и прототип, но в 1890ВМ3Т оба кэша 4-множественные. (В прототипе кэш-памяти 2-множественные).

Для выбора множества при замене строки в 1890ВМ3Т используется механизм Way Select, который использует специальные памяти Way Select RAM (для кэша команд и данных). Данные памяти должны быть очищены при инициализации. Также при инициализации должны быть сброшены все биты V в кэшах.

В 1890ВМ3Т реализована блокировка кэшей на уровне строки (а не множества, как в прототипе).

В 1890ВМ3Т не реализован контроль чётности в кэшах.

Отличается реализация команд CACHE и формат тэгов.

На кэш-памяти команд влияет режим Big/Little-Endian (см. в подразделе 3.8.4).

Архитектура кэшей 1890ВМ3Т и команда CACHE описываются в подразделе 3.8.

3.10.6 Исключения

Система исключений 1890ВМ3Т имеет небольшие отличия от прототипа, но при этом обеспечивается совместимость снизу вверх:

а) В 1890ВМ3Т имеется дополнительная возможность разместить обработчик внешних прерываний по отдельному вектору. Для этого в регистре Cause реализован специальный бит IV (Interrupt Vector).

б) В 1890ВМ3Т не реализовано исключение CacheError, так как не реализован механизм контроля чётности.

В 1890ВМ3Т дополнительно реализовано исключение MCheck, которое возникает при выполнении команд TLBWI/R, если происходит множественное совпадение в TLB (TLB Shutdown).

Исключения описаны в подразделе 3.6 и в документе MIPS64 Architecture For Programmers. Vol. III: The MIPS64 Privileged Resource Architecture (rev. 1.0).

3.10.7 Очистка TLB

Архитектура TLB процессора 1890ВМ3Т почти не отличается от прототипа. Отличие заключается в наличии специальных программно недоступных битов в каждой строке JTLB, которые сбрасываются по Reset и сигнализируют о том, что строка со сброшенным битом не участвует в сравнении. При заполнении TLB эти биты автоматически устанавливаются.

Из этого следует, что после Reset TLB не нуждается в очистке.

Если же потребуется дальнейшая программная очистка TLB, то при заполнении TLB неотображаемыми (unmapped) адресами нужно следить за их неповторением. Иначе в 1890ВМ3Т возникнет исключение MCheck.

3.10.8 Программная совместимость 1890ВМ3Т с прототипом

Пользовательские приложения (программы для режима User), исполняемые на прототипе совместимы снизу вверх с 1890ВМ3Т. То есть все команды, поддерживаемые прототипом, реализованы в 1890ВМ3Т. Также обеспечена совместимость снизу вверх по программной архитектуре целочисленных регистров

и регистров FPU. Однако пользовательские приложения для 1890ВМ3Т могут не работать на IDT79RC64475, так как архитектура MIPS-64 включает в себя больше непривилегированных команд, чем архитектура MIPS-III.

Привилегированные программы (загрузчики, обработчики исключений, менеджеры памяти и т. д.), предназначенные для прототипа, нуждаются в некоторой адаптации при переходе к 1890ВМ3Т. В адаптации, в основном, нуждаются программы инициализации (загрузчики), которые выполняют инициализацию регистров CP0, очистку кэшей и TLB и разные диагностические функции. Обработчики исключений и менеджеры памяти могут переноситься практически без изменений. (Исключение - очистка TLB. См. выше.)

Привилегированные программы, написанные специально для 1890ВМ3Т, могут оказаться совершенно несовместимыми с прототипом.

3.10.9 Внешний интерфейс

Подробное описание внешнего интерфейса 1890ВМ3Т приведено в подразделе 3.1.13.

Внешний интерфейс 1890ВМ3Т имеет следующие отличия от прототипа:

а) В 1890ВМ3Т не реализован механизм контроля чётности.

б) Инициализация при «холодном» старте в 1890ВМ3Т отличается от прототипа. Для задания режимов инициализации используются не сигналы ModeIn, ModeClock, а сигналы Int.

в) В 1890ВМ3Т не реализован Interrupt Register.

г) В 1890ВМ3Т реализован только механизм Write-reissue при цикле записи. Механизмы R4000-compatible и Pipline write не реализованы.

3.10.10 EJTAG

Механизм EJTAG в 1890ВМ3Т отличается от JTAG в IDT79RC64475. В 1890ВМ3Т предусмотрен особый режим привилегий (Debug) (см. подраздел 3.1.4.1) и несколько специальных регистров CP0 (Debug, DEPC, DESAVE), необходимых для EJTAG.

3.10.11 Отличия 1890ВМ3Т от программной архитектуры MIPS64

Стандарт MIPS64 предусматривает определённую свободу своей реализации в конкретных процессорах, и это не является отходом от стандарта.

Явные отличия 1890ВМ3Т от MIPS64 Release 1:

1) Отсутствует команда RSQRT.fmt (Обратная величина квадратного корня).

2) Не делается различий между «тихими» и «сигнализирующими» не-числами (Signaling and Quiet NaNs). Все не-числа ведут себя как «сигнализирующие».

3) Процессор 1890ВМ3Т не работает с денормализованными вещественными числами. Если в качестве операнда подаётся денормализованное число, вырабатывается исключение E (unimplemented operation).

4) Если исключение V (Invalid operation) в FPU замаскировано, то возникает исключение E (по стандарту должно возвращаться «тихое» не-число).

5) Не реализовано исключение CacheError.

Отличия от классического MIPS64, но не являющиеся отходом от стандарта:

1) Не реализован тип данных Paired Single (PS) в FPU, а также не реализованы все команды, работающие с таким типом данных.

2) Не реализованы некоторые регистры CP0 (WatchHi, WatchLo и т.д.), но согласно стандарту эти регистры не являются обязательными.

В документах MIPS описывается новая версия архитектуры MIPS64 Release 2, что является расширением MIPS64 Release 1. Release 2 содержит в себе новые команды и некоторые новые регистры сопроцессора CP0. 1890ВМ3Т эти команды и регистры не включает.

3.11 Программная совместимость 1890ВМ3Т с процессором 1890ВМ2Т

Пользовательские приложения для процессора 1890ВМ2Т совместимы (снизу вверх) с 1890ВМ3Т. Для обеспечения совместимости операционной системе необходимо включить режим 32-разрядной адресации для сегмента useg и режим 16-ти вещественных регистров (сбросить биты UX и FR в регистре Status).

Привилегированные программы процессоров несовместимы.

3.12 Описание внешнего интерфейса микропроцессора 1890ВМ3Т

3.12.1 Обзор интерфейса

В микропроцессоре 1890ВМ3Т в качестве внешнего интерфейса используется двунаправленная 64-разрядная шина SysAD. Обмен данными по шине осуществляется транзакциями. Существует 3 основных вида транзакций:

1) Чтение данных в процессор.
2) Запись данных во внешнее устройство.
3) Захват шины внешним устройством (external request).

Весь обмен по шине SysAD синхронизован по положительному фронту MasterClk. Все выходные сигналы идут с регистров, и все входные сигналы защёлкиваются в регистрах, что позволяет обеспечить хорошие временные характеристики шины.

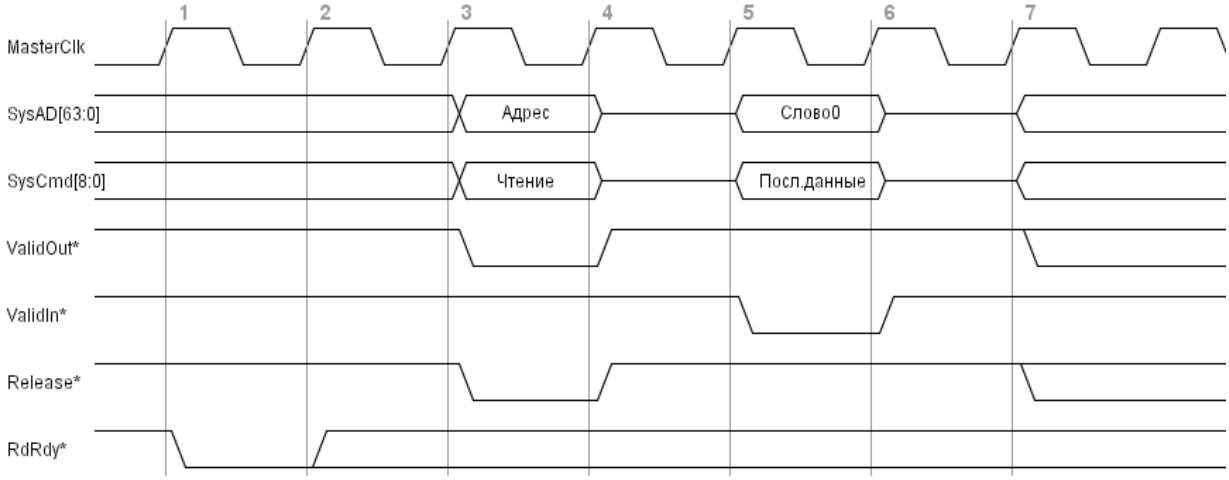


Рисунок 3.39 - Транзакция чтения одиночного слова

На рисунке 3.39 представлен пример транзакции чтения процессором одиночного слова без циклов ожидания. К началу такта 2 внешнее устройство выставило сигнал RdRdy*, сообщающий о готовности к чтению. Транзакция начинается на такт позже. В такте 3 на шину SysAD выставляется адрес требуемого слова, а на шину SysCmd – идентификатор транзакции «чтение одиночного слова». Сигнал ValidOut* показывает, что данные SysAD/SysCmd действительны. К началу такта 4 процессор выставляет сигнал Release*, сообщающий о том, что в такте 4 он освободит шины, и, начиная с такта 5, внешнее устройство может выдавать данные на шины. Данные, стробируемые сигналом ValidIn*, выдаются в такте 5 на шину SysAD. Идентификатор SysCmd сообщает процессору тип данных (слово прочитано успешно, слово последнее – завершение транзакции). В такте 6 внешнее устройство обязано освободить шину, так как в такте 7 процессор может начать новую транзакцию. Более подробно транзакции чтения описаны в подразделе 3.11.3.

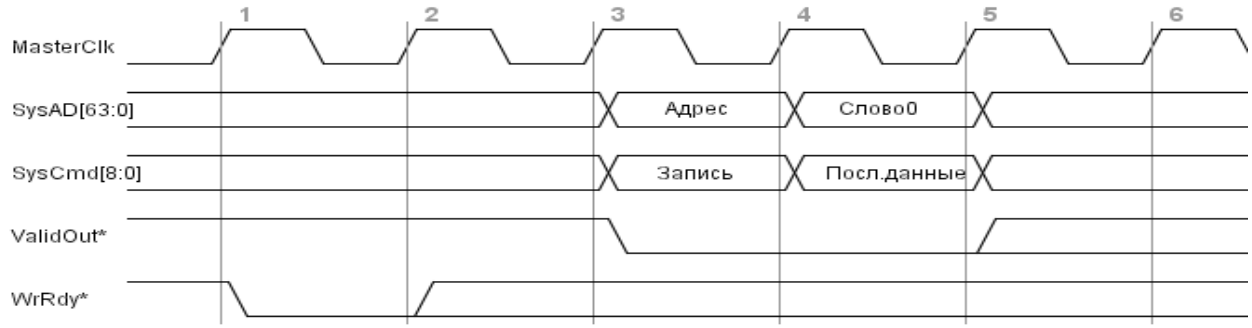


Рисунок 3.40 - Транзакция записи одиночного слова

На рисунке 3.40 приведён пример транзакции записи одиночного слова. К такту 2 внешнее устройство выставляет сигнал WrRdy*, сообщающий процессору о готовности устройства к записи. К такту 4 процессор выставляет адрес на SysAD и идентификатор записи на SysCmd. В следующем такте процессор передаёт данные. В такте 5 процессор может начать новую транзакцию. Подробнее транзакции записи описаны в подразделе 3.11.4.

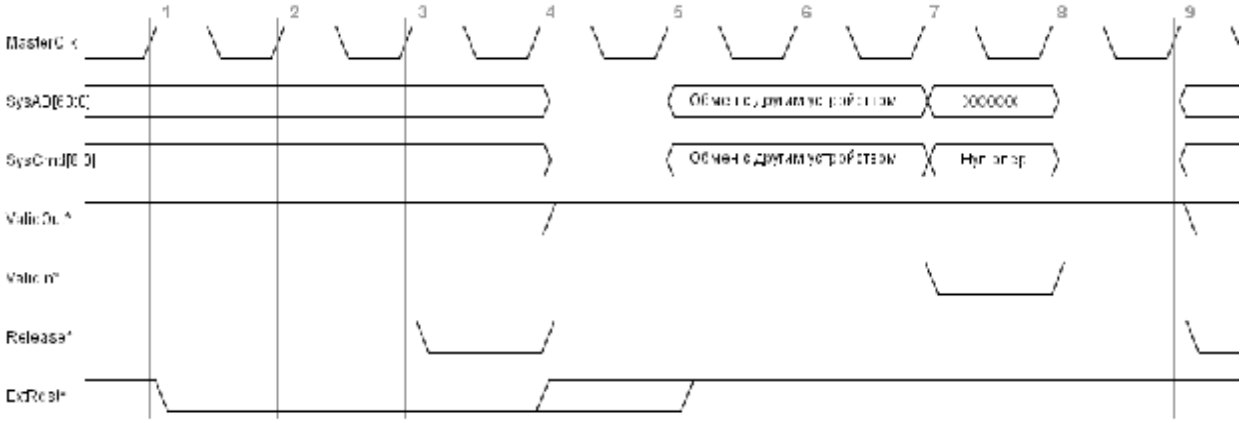


Рисунок 3.41 - Захват шины внешним устройством

На рисунке 3.41 приведён пример транзакции захвата шины. Внешнее устройство выставляет запрос на захват (сигнал ExtRqst*). Процессор заканчивает текущую транзакцию и разрешает захват, выставляя сигнал Release* (такт 3). В такте 4 процессор освобождает шины. Сигнал ExtRqst* должен быть снят не позже, чем к началу такта 6. Начиная с такта 5 внешние устройства могут использовать шины SysAD/SysCmd для своих нужд (сигнал ValidIn* должен оставаться неактивным). Для завершения захвата шины внешнее устройство выдаёт идентификатор нулевой операции (в такте 7). В такте 8 оно обязано освободить шину, и в такте 9 процессор может начать новую транзакцию. Подробнее захват шины описан в подразделе 3.11.5.

Шина SysAD может работать в одном из двух режимов – 64- и 32-разрядном. Режим задаётся при «холодном» перезапуске процессора (см. ниже).

3.12.2 Инициализация процессора

Можно выделить три вида инициализации процессора 1890ВМ3Т: начальная инициализация при включении питания, «холодный» перезапуск, «тёплый» перезапуск. Временные диаграммы инициализаций приведены на рисунках, где $t_3 = t_2 \geq 10$ периодов; $t_4 \geq 20$ периодов; $t_1 \geq 0$ периодов.

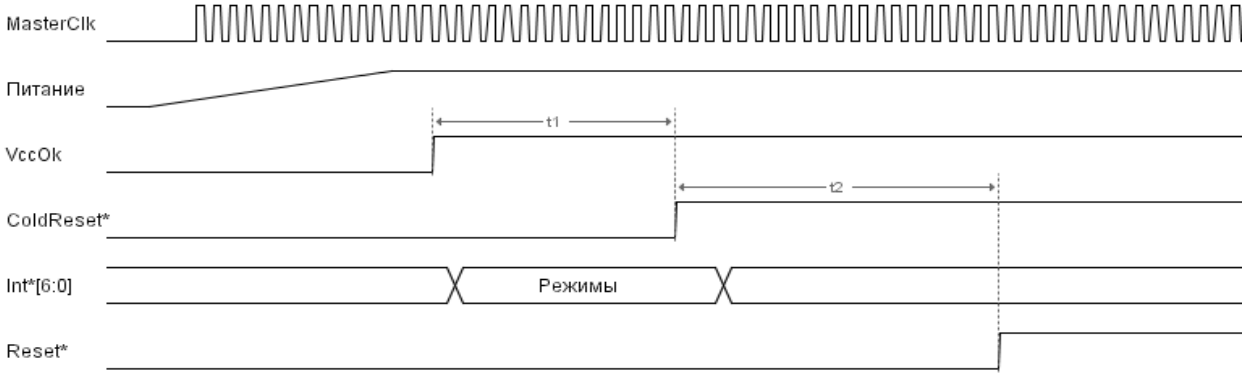


Рисунок 3.42 - Инициализация процессора при включении питания

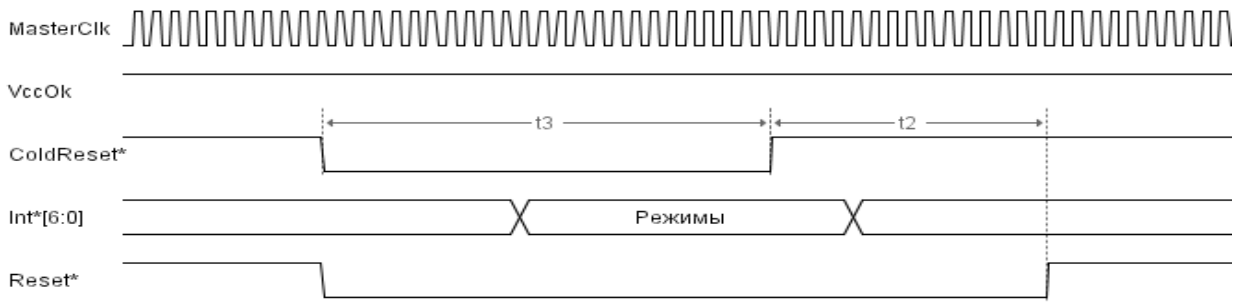


Рисунок 3.43 - «Холодный» перезапуск процессора

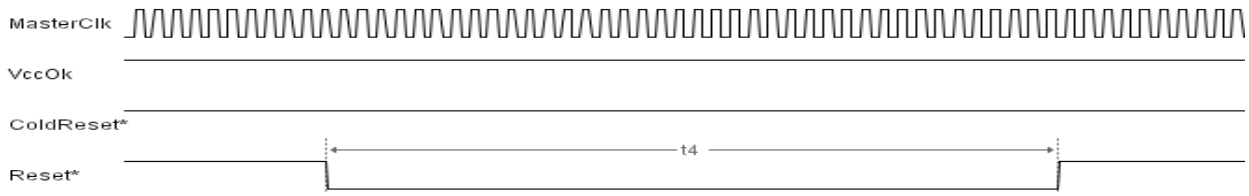


Рисунок 3.44 - «Тёплый» перезапуск процессора

При начальном запуске и «холодном» перезапуске процессора сигналами Int*[4:0] задаются начальные режимы работы процессора (таблица 3.54). При «тёплом» перезапуске сохраняются предыдущие режимы.

Таблица 3.54 - Кодировка начальных режимов работы

Сигнал	Режим
Int*[4]	Режим ширины шины SysAD. 0 – 64 бита, 1 – 32 бита
Int*[3]	Основной режим Endian. 0 – Little Endian, 1 – Big Endian
Int*[2:0]	Множитель внутренней частоты процессора (PClk). 0 – f(PClk)=f(MasterClk)*2 1 – f(PClk)=f(MasterClk)*3 2 – f(PClk)=f(MasterClk)*4 3 – f(PClk)=f(MasterClk)*5 4 – f(PClk)=f(MasterClk)*6 5 – f(PClk)=f(MasterClk)*7 6 – f(PClk)=f(MasterClk)*8 7 – f(PClk)=f(MasterClk) - тестовый режим (режим работы PLL без умножения частоты тактового сигнала и подстройки фазы)

Режим ширины шины определяет, сколько разрядов шины SysAD будет задействовано при передачи данных и адресов. В режиме 64 разрядов задействованы все разряды шины при передачи данных (обмен ведётся двойными словами) и младшие 36 разрядов при передачи адресов. В режиме 32 разрядов адреса «обрезаются» до младших 32 бит. Данные передаются 32-разрядными словами (подробнее см. конкретные транзакции в подразделах 3.11.3 и 3.11.4). Данный режим существует независимо от внутренних режимов процессора, и его не следует путать с режимами разрядности адресации (биты UX, SX, KX, PX регистра Status). Программно режим шины отображается в поле BW регистра Config0 CP0.

Режим Endian, заданный аппаратно, является основным для процессора и влияет на работу шины и исполнение некоторых команд. Программно основной режим Endian отображается в поле BE регистра Config0 CP0.

Для режима User может быть включён режим Endian, обратный установленному аппаратно, с помощью бита RE в регистре Status CP0, однако на работу шины SysAD влияет только основной режим. То есть, если в режиме Reverse Endian (основной режим, например, Big-Endian) происходит чтение байта с адресом 0 (читается младший байт, как в Little-endian), то на шине SysAD будет адресован байт 7 (младший байт в Big-Endian).

Множитель частоты настраивает внутреннюю PLL и определяет, во сколько раз внутренняя частота процессора будет больше частоты шины.

3.12.3 Интерфейс чтения

Процессор инициирует транзакцию чтения при непопадании в кэш команд/данных или при некэшированном чтении. Внешнее устройство, если оно не

готово, может предотвратить транзакцию чтения снятием сигнала RdRdy* за 2 такта до её возможного начала. В другом случае устройство должно быть готово (через 2 такта после активного RdRdy*) принять адрес и идентификатор чтения на шинах SysAD/SysCmd. Далее процессор переводит шины SysAD/SysCmd в третье состояние и ждёт от внешнего устройства прочитанные данные. В это время внешнее устройство может использовать шину для своих нужд, не выдавая процессору сигнал ValidIn* (см. подраздел 3.11.8).

Транзакции чтения бывают двух видов – чтение одиночного слова и чтение блока слов (burst). Чтения одиночных слов возникают при выполнении программы/чтения данных из некэшируемой области. Чтения блоков – при непопадании в кэш команд/данных.

3.12.3.1 Чтение одиночного слова

Транзакция чтения одиночного слова представлена на рисунке 3.39. Если включён режим 32-разрядной шины, то операцию чтения слова, превышающего разрядность 32, процессор разбивает на две транзакции чтения.

На рисунке 3.45 показан формат идентификатора (SysCmd[8:0]) при передачи адреса. Бит 8 указывает, что передаётся адрес (а не данные). Биты [4:3] указывают, что инициируется транзакция чтения одиночного слова. Биты [2:0] определяют разрядность читаемого слова (таблица 3.1). В режиме 32-разрядной шины это поле не может быть больше 3. В таблице 3.2 и таблице 3.3 показано, какие байты шины SysAD будут участвовать при передачи данных в зависимости от поля AccTy, основного режима Endian и младших битов адреса.

8	7	6	5	4	3	2	1	0
0	0			3		AccTy		

Рисунок 3.45 - Формат идентификатора адреса чтения

Таблица 3.55 - Кодировка поля AccTy в идентификаторе адреса

AccTy	Тип слова
0	Байт
1	Полуслово (2 байта)
2	Три-байт (3 байта)
3	Слово (4 байта)
4	Пента-байт (5 байт)
5	Сексти-байт (6 байт)
6	Септи-байт (7 байт)
7	Двойное слово (8 байт)

Таблица 3.56 - Используемые байты при передачи неполных слов (шина 64 бита)

SysCmd [2:0]	Адрес [2:0]	Используемые биты SysAD (основной режим Big-Endian)							
		63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
0 (байт)	0	•							
	1		•						
	2			•					
	3				•				
	4					•			
	5						•		
	6							•	
	7								•
1 (полуслово)	0	•	•						
	2			•	•				
	4					•	•		
	6							•	•
2 (три-байт)	0	•	•	•					
	1		•	•	•				
	4					•	•	•	
	5						•	•	•
3 (слово)	0	•	•	•	•				
	4					•	•	•	•
4 (5 байт)	0	•	•	•	•	•			
	3				•	•	•	•	•
5 (6 байт)	0	•	•	•	•	•	•		
	2			•	•	•	•	•	•
6 (7 байт)	0	•	•	•	•	•	•	•	
	1		•	•	•	•	•	•	•
7 (8 байт)	0	•	•	•	•	•	•	•	•
		7:0	15:8	23:16	31:24	39:32	47:40	55:48	63:56
		Используемые биты SysAD (основной режим Little-Endian)							

Таблица 3.57 - Используемые байты при передачи неполных слов (шина 32 бита)

SysCmd[2:0]	Адрес[1:0]	Используемые биты SysAD (режим Big-Endian)			
		31:24	23:16	15:8	7:0
0 (байт)	0	•			
	1		•		
	2			•	
	3				•
1 (полуслово)	0	•	•		
	2			•	•
2 (три-байт)	0	•	•	•	
	1		•	•	•
3 (слово)	0	•	•	•	•
		7:0	15:8	23:16	31:24
		Используемые биты SysAD (режим Little-Endian)			

На рисунке 3.46 представлен формат идентификатора данных, который должно возвращать внешнее устройство вместе с данными. После получения процессором данных и идентификатора транзакция считается завершённой, и через такт процессор захватывает шину.

8	7	6	5	4	3	2	1	0
1	0	0	BE	Игнорируются				

Рисунок 3.46 - Формат идентификатора данных при чтении одиночного слова

Бит 8 показывает, что по шине SysAD передаются данные (а не адрес). Бит 7 показывает, что передаётся последнее слово, и транзакция должна быть завершена. Бит 6 показывает, что данные являются ответом на запрос чтения. Бит 5 (Bus Error) показывает, произошла ли ошибка шины при чтении. При BE=1 данные на шине SysAD игнорируются, транзакция завершается, и в процессоре вырабатывается соответствующее исключение Bus Error (рисунок 3.49).

3.12.3.2 Чтение блока слов

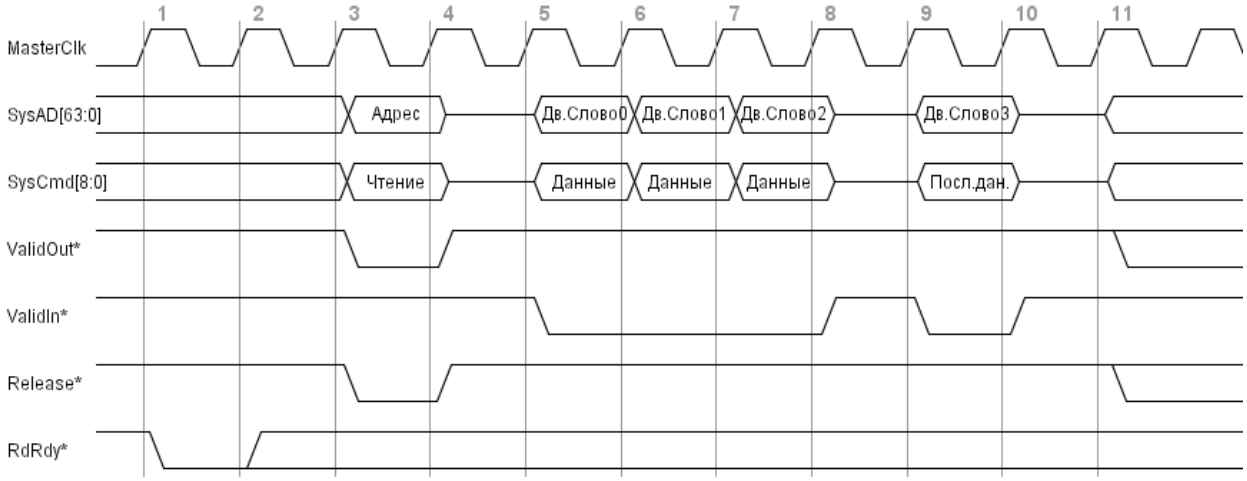


Рисунок 3.47 - Транзакция чтения блока слов (шина 64 бита)

На рисунке 3.47 приведён пример чтения блока слов для 64-разрядного режима шины. В такте 1 внешнее устройство устанавливает RdRdy*, разрешая транзакцию чтения (начиная с такта 3). В такте 3 процессор выставляет сигнал ValidOut*, сообщающий внешнему устройству, что на шины SysAD/SysCmd выставлены действительные значения. На SysAD выставляется адрес первого требуемого слова. На SysCmd – идентификатор чтения блока слов (см. ниже). В этом же такте процессор выставляет сигнал Release*, что означает, что в такте 4 шины будут освобождены, и с такта 5 внешнее устройство может их захватывать.

Далее процессор ожидает 4 двойных слова в 64-разрядном режиме шины или 8 32-разрядных слов в 32-разрядном режиме шины (рисунок 3.48), сопровождаемых соответствующими идентификаторами. Внешнее устройство может выдавать слова с любой задержкой. Скорость выдачи слов контролируется сигналом ValidIn*. Идентификатор, сопровождающий последнее слово, должен иметь признак Last. Нужно отметить, что в случае возникновения ошибки шины первый выдаваемый идентификатор должен содержать признак ошибки шины и признак последнего слова (рисунок 3.48). Данные на SysAD в этом случае игнорируются.

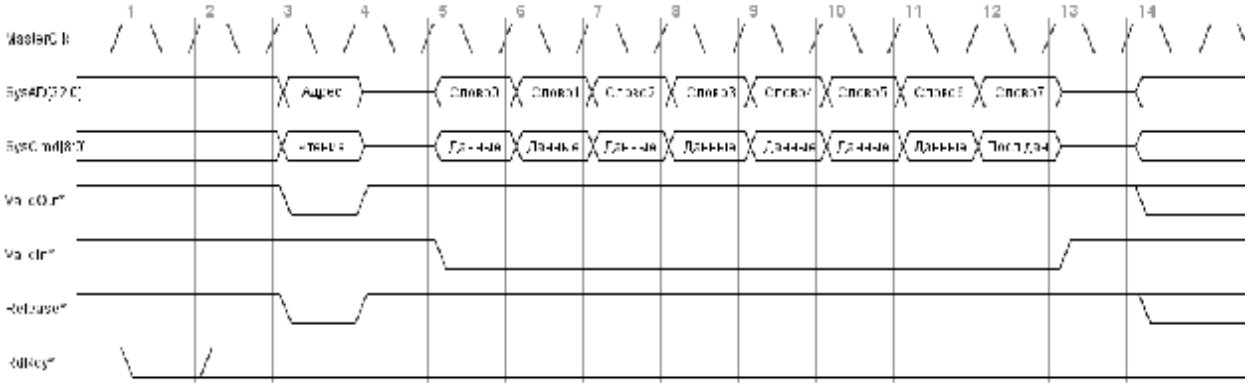


Рисунок 3.48 - Транзакция чтения блока слов (шина 32 бита)

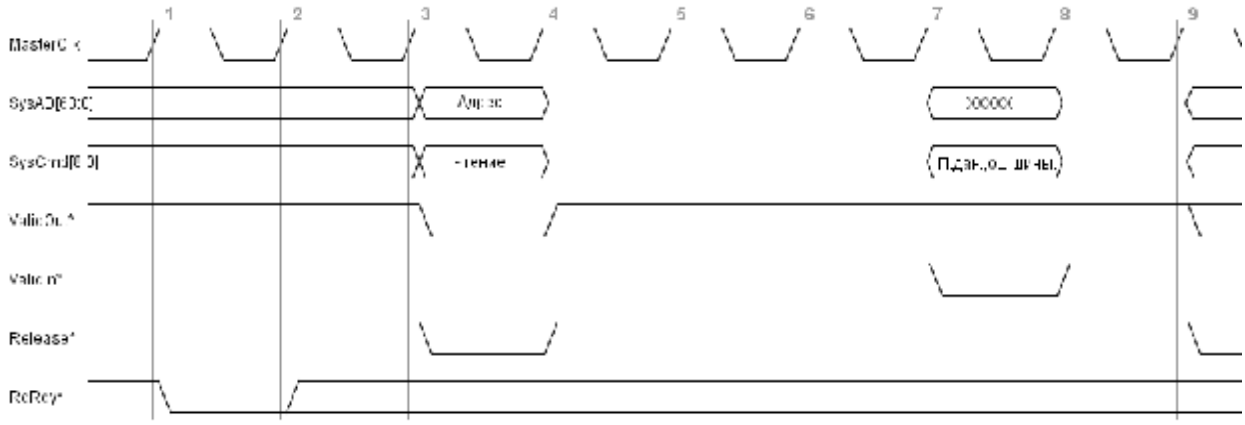


Рисунок 3.49 - Ошибка шины при чтении одиночного слова/блока слов

Младшие биты адреса, выставляемого процессором на SysAD при чтении блока слов, (SysAD[2:0] при 64-разрядном режиме шины или SysAD[1:0] при 32-разрядном) всегда равны 0. Биты SysAD[4:3] (SysAD[4:2] для 32-разрядного режима шины) указывают на двойное слово (слово для 32-разрядного режима шины), которое должно быть выдано на шину первым. Остальные слова должны выдаваться в последовательности Sub-block Order. Чтобы получить адрес очередного слова в порядке Sub-block, необходимо выполнить операцию XOR над адресом первого слова и порядковым номером текущего слова (нумерация с 0). То есть например, если первое требуемое двойное слово (режим 64 бит) имеет младший адрес SysAD[4:3]=1, то слова должны прийти в порядке 1, 0, 3, 2. Или если первое требуемое слово (режим 32 бит) имеет адрес SysAD[4:2]=5, то слова должны прийти в порядке 5, 4, 7, 6, 1, 0, 3, 2.

После выдачи последнего слова внешнее устройство должно освободить шины, так как через такт они будут заняты процессором (рисунок 3.47, такт 11).

8	7	6	5	4	3	2	1	0
0	0			2		0	1	

Рисунок 3.50 - Формат идентификатора адреса при чтении блока слов

На рисунке 3.50 представлен формат идентификатора, который выставляется процессором на шину SysCmd[8:0] в такте выдачи адреса. Бит 8 показывает, что это идентификатор адреса (а не данных). Биты [7:5] показывают, что процессор запрашивает чтение. Биты [4:3] сообщают что инициируется чтение блока слов (а не одиночного слова). Бит 2 не используется. Биты [1:0] сообщают, что ожидается 4 двойных слова (при 64-разрядной шине) или 8 слов (при 32-разрядной шине).

8	7	6	5	4	3	2	1	0
1	NLst	0	BE	Игнорируются				

Рисунок 3.51 - Формат идентификатора данных при чтении блока слов

На рисунке 3.51 представлен формат идентификатора, который должен выставляться внешним устройством на шину SysCmd[8:0] при передачи очередного слова данных. Бит 8 показывает, что по шине SysAD передаются данные (а не адрес). Бит 7 определяет, какое слово передаётся. Если он равен 0, то слово последнее. 1 – слово не последнее, и транзакция будет завершена. Бит 6 показывает, что данные являются ответом на запрос чтения. Бит 5 (Bus Error) показывает, произошла ли ошибка шины при чтении.

3.12.4 Интерфейс записи

Транзакция записи инициируется процессором, если буфер записи не пуст, и внешнее устройство выставило сигнал WrRdy*. В этом случае через 2 такта внешнее устройство должно быть готово принять адрес записи. После адреса процессор на шину SysAD выставляет данные. Задержка между адресом и данными задаётся полем ADP в регистре Config0 сопроцессора CP0. Она может составлять 0, 1, 2, 3 такта шины SysAD. После инициализации процессора она по умолчанию равна 3 тактам.

Существует два типа транзакций записи – запись одиночного слова и запись блока слов. Запись одиночного слова возникает после выполнения команды записи при некэшированном обращении или при политике кэширования Write-thru. Запись блока слов инициируется при выталкивании «грязной» строки кэша данных при политики Write-back.

Нужно отметить, что в шине SysAD/SysCmd не существует механизма информирования об ошибки шины (Bus Error) при записи. Внешнее устройство в этом случае должно сообщить процессору об ошибке другим способом – например, выставить запрос на прерывание.

3.12.5 Запись одиночного слова

Примеры транзакции записи одиночного слова представлены на рисунке 3.40 и на рисунке 3.52 (с задержкой адрес-данные в 2 такта).

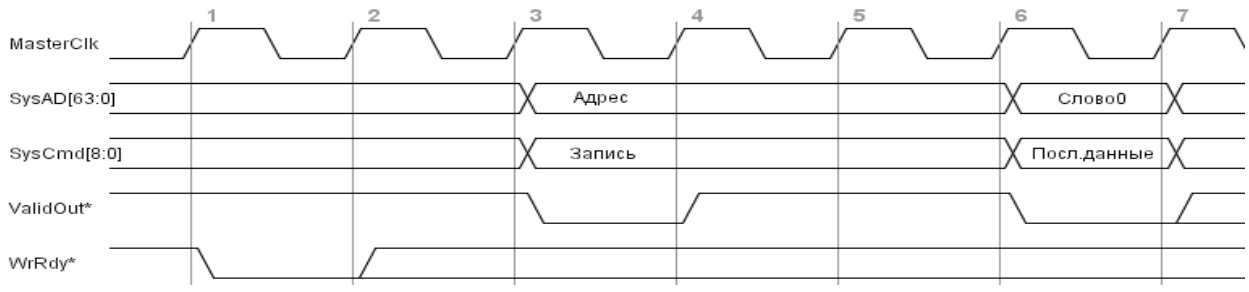


Рисунок 3.52 - Транзакция записи одиночного слова с задержкой данных

В режиме 32-разрядной шины при записи слова, разрядностью больше 32, процессор выполняет две независимых транзакции записи.

8	7	6	5	4	3	2	1	0
0	2			3		AccTy		

Рисунок 3.53- Формат идентификатора адреса при записи одиночного слова

На рисунке 3.53 представлен формат идентификатора, выставляемого процессором на шину SysCmd[8:0] в такте выдачи адреса. Бит 8 указывает, что шина SysAD содержит адрес. Биты [7:5] сообщают о начале транзакции записи. Биты [4:3] указывают, что будет записано одно слово. Биты [2:0] определяют формат записываемого слова (см. таблицу 3.1). В 32-разрядном режиме шины бит 2 всегда равен 0. Младшие биты адреса (SysAD[2:0] для 64-разрядной шины и SysAD[1:0] для 32-разрядной шины) определяют, какие байты в слове будут задействованы при записи (см. таблицу 3.2 и таблицу 3.3).

8	7	6	5	4	3	2	1	0
1	0	0						

Рисунок 3.54 - Формат идентификатора данных при записи одиночного слова

На рисунке 3.54 представлен формат идентификатора, выставляемого процессором на шину SysCmd[8:0] в такте выдачи данных. Бит 8 указывает, что шина SysAD содержит данные. Бит 7 сообщает внешнему устройству, что выдаваемое слово данных – последнее, и транзакция завершена. Остальные биты не используются.

3.12.6 Запись блока слов

На рисунке 3.55 показана транзакция записи блока слов по 64-разрядной шине. Если внешнее устройство установило сигнал WrRdy*, то оно должно быть в состоянии обработать транзакцию блочной записи, которая может инициироваться 2 такта спустя. Младшие биты передаваемого адреса (SysAD[4:0]) равны 0. После цикла адреса передаётся 4 двойных слова данных в режиме 64-разрядной шины или 8 слов в режиме 32-разрядной шины (рисунок 3.56). Слова передаются в естественном адресном порядке – от 0-го до 3-го (7-го). Последнее слово сопровождается идентификатором окончания, и транзакция заканчивается.

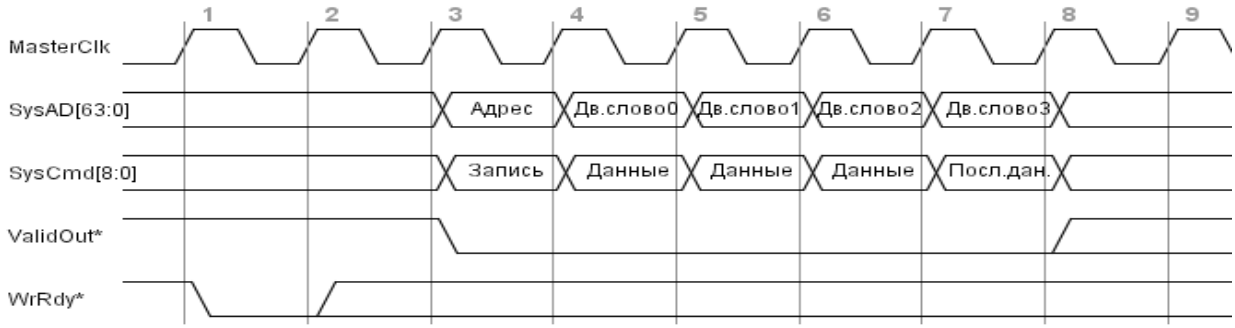


Рисунок 3.55 - Запись блока слов (шина 64 бита)

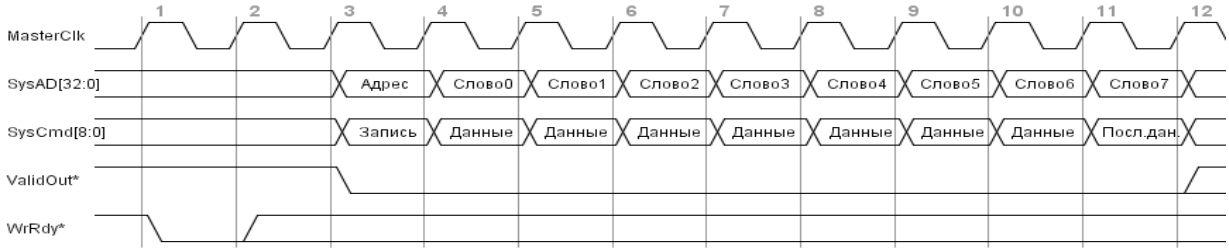


Рисунок 3.56 - Запись блока слов (шина 32 бита)

Для медленных внешних устройств в процессоре 1890ВМ3Т предусмотрена возможность вставки холостых циклов в транзакцию записи. При блочной записи помимо задержки «адрес-данные» (поле ADP в Config0 CP0) можно настроить задержку между словами данных. Она задаётся в поле WP регистра Config0 CP0 и может составлять 0, 1, 2, 3 такта. После инициализации процессора по умолчанию она составляет 3 такта. На рисунке 3.57 представлена транзакция записи при ADP=2 и WP=1.

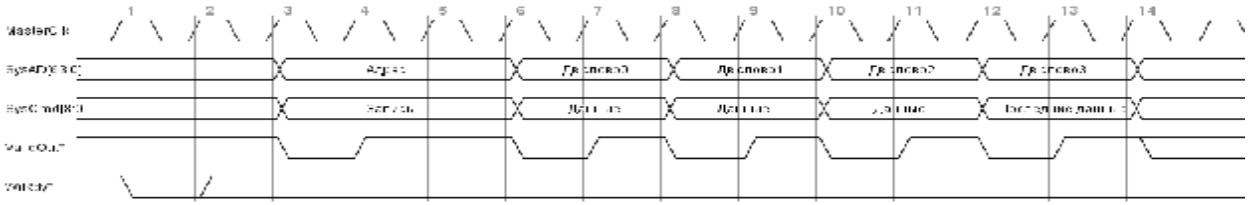


Рисунок 3.57 - Запись блока слов с холостыми циклами

8	7	6	5	4	3	2	1	0
0	2			2		0	1	

Рисунок 3.58 - Формат идентификатора адреса при записи блока слов

На рисунке 3.58 представлен формат идентификатора, выставляемого процессором на шину SysCmd[8:0] в такте выдачи адреса. Бит 8 означает, что по SysAD передаётся адрес. Биты [7:5] показывают, что инициируется транзакция записи. Биты [4:3] показывают, что запись блочная. Бит 2 не используется. Биты [1:0] сообщают, что блок состоит из 4-х двойных слов (в 64-разрядном режиме шины) или из 8 слов (в 32-разрядном режиме шины).

8	7	6	5	4	3	2	1	0
1	NLst	0						

Рисунок 3.59 - Формат идентификатора данных при записи блока слов

На рисунке 3.59 представлен формат идентификатора, выставляемого процессором на шину SysCmd[8:0] в тактах выдачи данных. Бит 8 означает, что по шине SysAD передаются данные. Бит 7 определяет, является ли передаваемое слово последним (1 – не последнее, 0 – последнее). Биты [6:0] не используются.

3.12.7 Подряд идущие транзакции записи

После завершения транзакции записи процессор может инициировать новую транзакцию записи в следующем такте, если 2 такта назад сигнал WrRdy* был активен (рисунок 3.60). Такой механизм позволяет сэкономить время при последовательных записях. Однако, может возникнуть проблема, если внешнее устройство не успевает обработать текущую запись. Дело в том, что внешнее устройство может снять сигнал WrRdy* не раньше, чем в такте 4 (рисунок 3.60). Но для того, чтобы предотвратить вторую транзакцию записи, WrRdy* должен быть снят в такте 3. Для решения этой проблемы в процессоре 1890ВМ3Т реализован механизм Write-reissue.

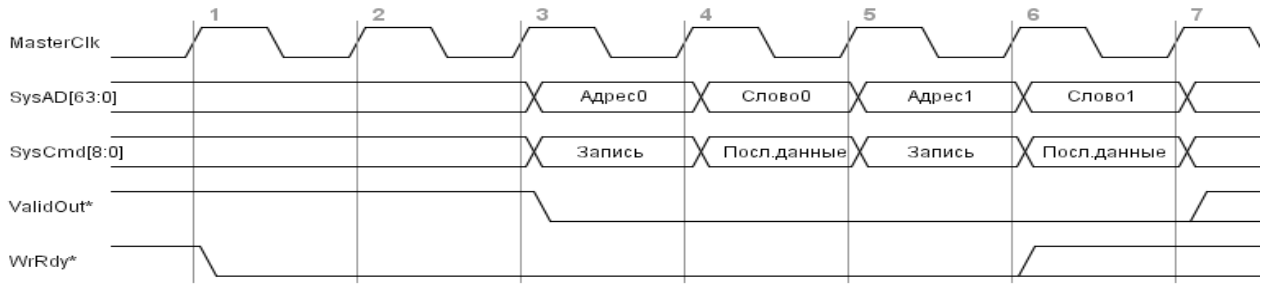


Рисунок 3.60 - Запись подряд двух одиночных слов

Если устройство не готово обработать вторую транзакцию записи «на лету» (рисунок 3.61), то оно должно снять сигнал WrRdy* не позже первого такта следующей транзакции (такт 4 или 5). При этом вторая транзакция по выше указанным причинам всё же инициируется, но она будет повторена после установки WrRdy* в активное значение (такты 8, 10 и 11). Такой механизм называется Write-reissue.

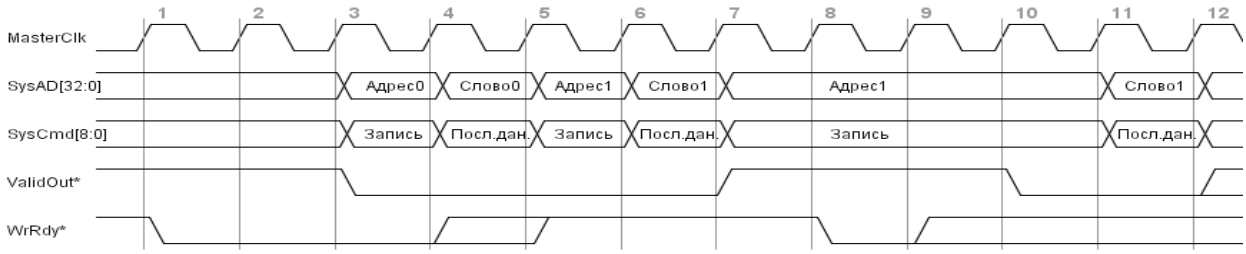


Рисунок 3.61 - Задержанная запись второго одиночного слова

Выше сказанное также относится к записи блока слов (рисунок 3.62). В этом случае транзакция будет прервана через 2 такта (такт 7 – снятие сигнала ValidOut*). В любом случае транзакция будет прервана через 2 такта после начала (например, в случае чтения одиночного слова с задержкой «адрес-данные»), и устройство не должно воспринимать «обрывок» прерванной транзакции.

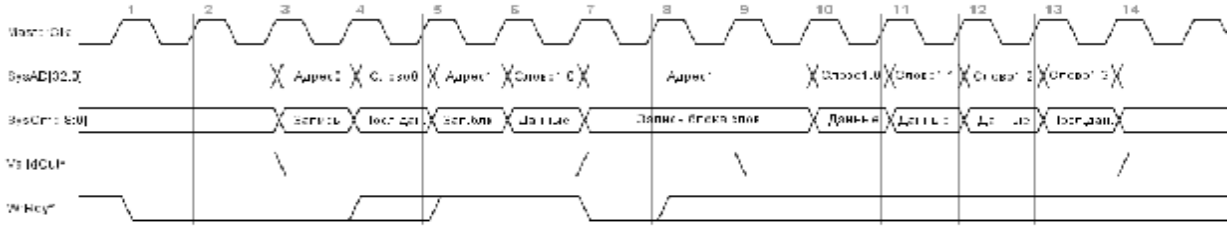


Рисунок 3.62 - Задержка записи блока слов

Нужно отметить, что прерванная транзакция считается завершённой в том смысле, что внешнее устройство после прерывания транзакции может, например, инициировать захват шины, который будет успешно обработан (процессор освободит шину для внешнего устройства, а после возврата шины и установки WrRdy* в активное значение повторит прерванную запись). Проблемы типа «deadlock» не возникнет.

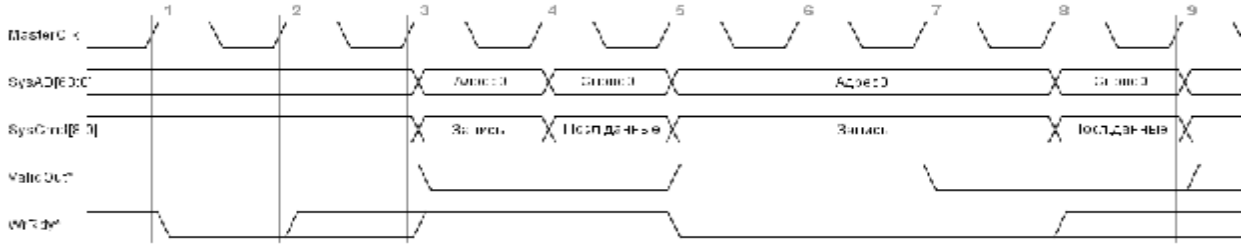


Рисунок 3.63 - Повторная запись первого слова

Для ясности необходимо также отметить, что механизм Write-reissue распространяется и на первую транзакцию записи (рисунок 3.63). Если в такте 3 сигнал WrRdy* будет не активен, то запись считается прерванной и будет повторена (такты 7 и 8). Для того, чтобы процессор считал запись законченной, WrRdy* должен быть ещё активным в цикле выдачи адреса.

3.12.8 Захват шины внешним устройством

Пример захвата шины (external request) показан на рисунке 3.41. Для завершения транзакции захвата шины внешнее устройство должно выставить идентификатор нулевой операции на шину SysCmd[8:0] (рисунок 3.68). Данные на шине SysAD игнорируются. В процессоре 1890ВМ3Т других операций при захвате шины не реализовано.

8	7	6	5	4	3	2	1	0
0	3			Игнорируются				

Рисунок 3.64 - Идентификатор нулевой операции

Нужно отметить, что в некоторых ситуациях внешнему устройству будет трудно отличить ответ процессора на захват шины от освобождения шины процессором при транзакции чтения. Например, если процессор инициирует транзакцию чтения в такте, когда внешнее устройство выставило сигнал ExtRqst*, то в этом же такте процессор выставит сигнал Release* и в следующем такте

освободит шину, ожидая прочитанных данных. Эта проблема разрешается в 1890ВМ3Т. Если процессор освободил шину, чтобы получить ответные данные чтения, то он также может воспринять идентификатор нулевой операции. В этом случае процессор заново инициирует транзакцию чтения (рисунок 3.69).

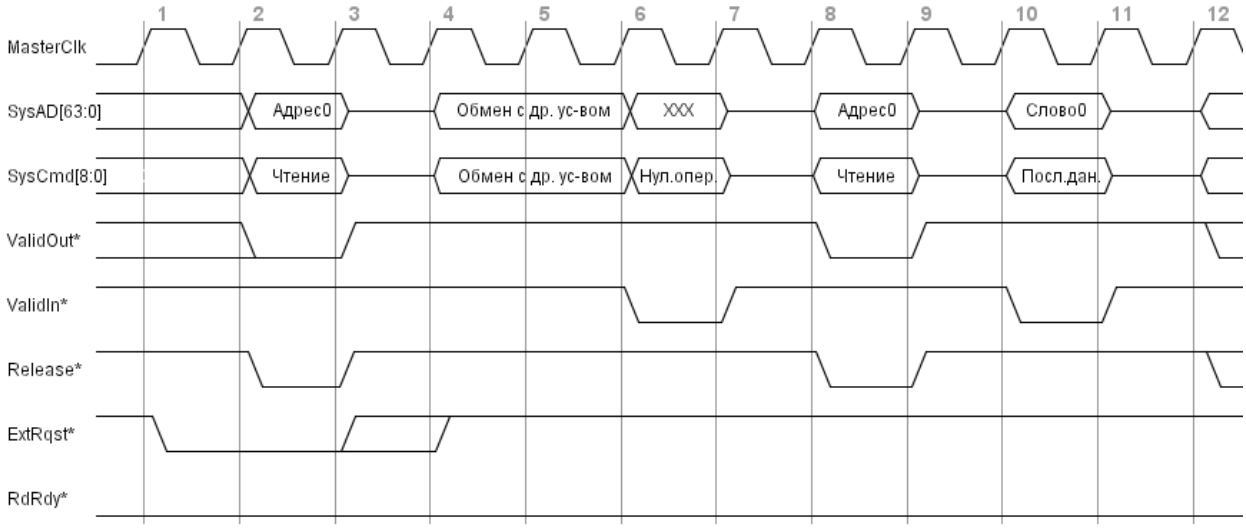


Рисунок 3.65 - Захват шины с отложенным чтением

Если внешнее устройство обработает сначала транзакцию чтения (рисунок 3.70), то процессор выдаст повторно сигнал Release* (такт 6), который будет означать ответ на запрос захвата шины. Сигнал ExtRqst* должен оставаться активным во время транзакции чтения (такты 2-6), но должен быть снят не позже, чем к началу такта 9.

Описанный способ откладывания транзакции чтения (возврат идентификатора нулевой операции) может быть использован в случаях, когда внешнее устройство не успело во время снять сигнал RdRdy*. Например, это может быть случай, когда транзакция чтения была инициирована сразу после транзакции записи, а запись по каким-либо причинам вызвала неготовность устройства к чтению (рисунок 3.67).

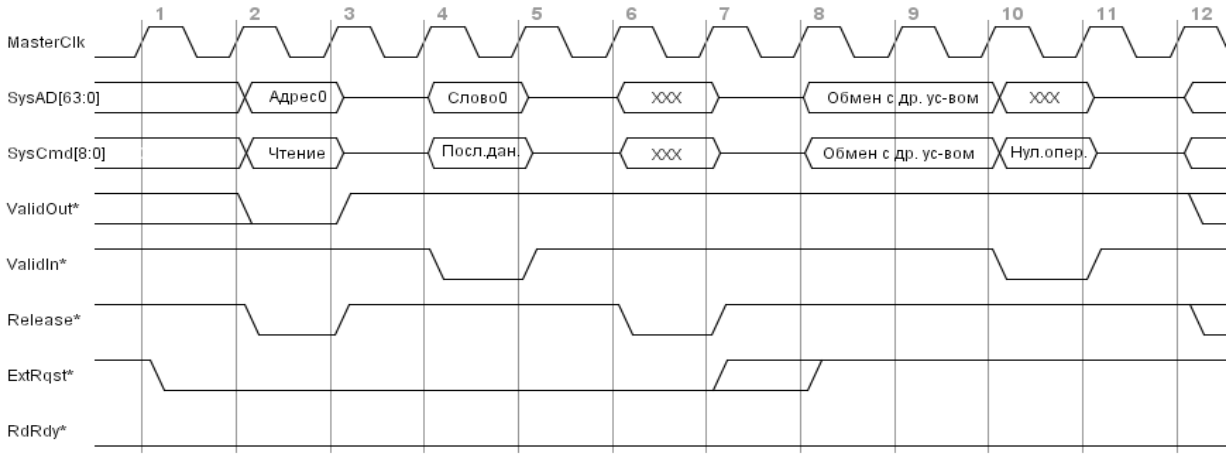


Рисунок 3.66 - Захват шины после чтения

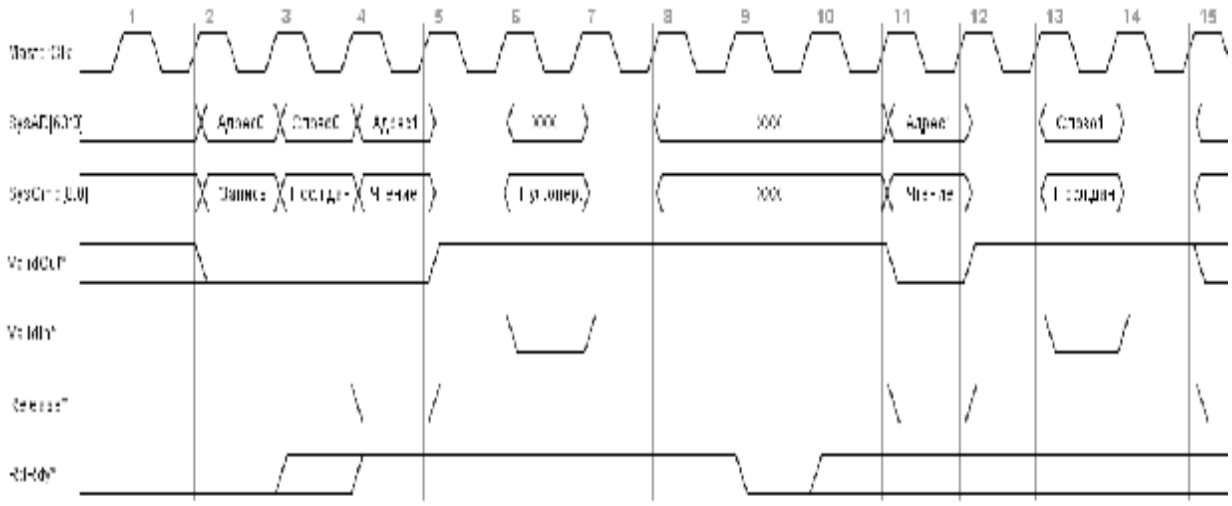


Рисунок 3.67 - Отложенное чтение

3.12.9 Внешние прерывания

Для запросов внешних прерываний используются сигналы Int*[5:0] и NMI. Все сигналы фиксируются по положительному фронту MasterClk. Активное значение сигналов является 0.

Сигналы Int*[5:0] вызывают маскируемые прерывания и программно отображаются в поле IP[7:2] регистра Cause CP0. Если разрешено хотя бы одно из выставленных прерываний (соответствующие разряды IM[7:2]=1, IE=1, EXL=ERL=0 в регистре Status CP0), то возникает исключение Int. Сигналы Int*[6:0] должны оставаться в стабильном состоянии, пока обработчик прерываний не запишет соответствующие данные в управляющие регистры устройств. В свою очередь обработчик прерываний после выполнения записи для снятия запросов Int* должен выполнить команду SYNC, чтобы успел очиститься буфер записи, и не возникло повторного исключения.

Прерывание от IP[7] может быть вызвано не только сигналом Int*[5], но также прерыванием от внутреннего таймера (регистры Count, Compare при TE=1 в Config0 CP0) и прерыванием от счётчиков производительности (регистры PerfCnt CP0). Данные прерывания объединяются по принципу «ИЛИ», то есть прерывание возникает, если хотя бы одно условие истинно.

Сигнал NMI* вызывает немаскируемое прерывание (исключение NMI). Сигнал может быть снят сразу после защёлкивания по MasterClk, либо может быть снят через определённое время, либо он может быть снят обработчиком прерывания при условии, что сигнал всё это время должен оставаться стабильным (иначе возникнет новое исключение NMI).

3.13 Корпус и назначение выводов микросхемы

Микросхема собрана в металлокерамический 240-ка выводной корпус 240CQFP (PB-FA3151) фирмы KYOCERA (Япония) с планарным расположением выводов с четырех сторон. Герметизацию микросхем проводят шовной контактной сваркой.

Габаритные размеры и расположение выводов микросхемы приведены на рисунке 3.68. Описание выводов микросхемы 1890ВМ3Т приведено в таблице 3.58.

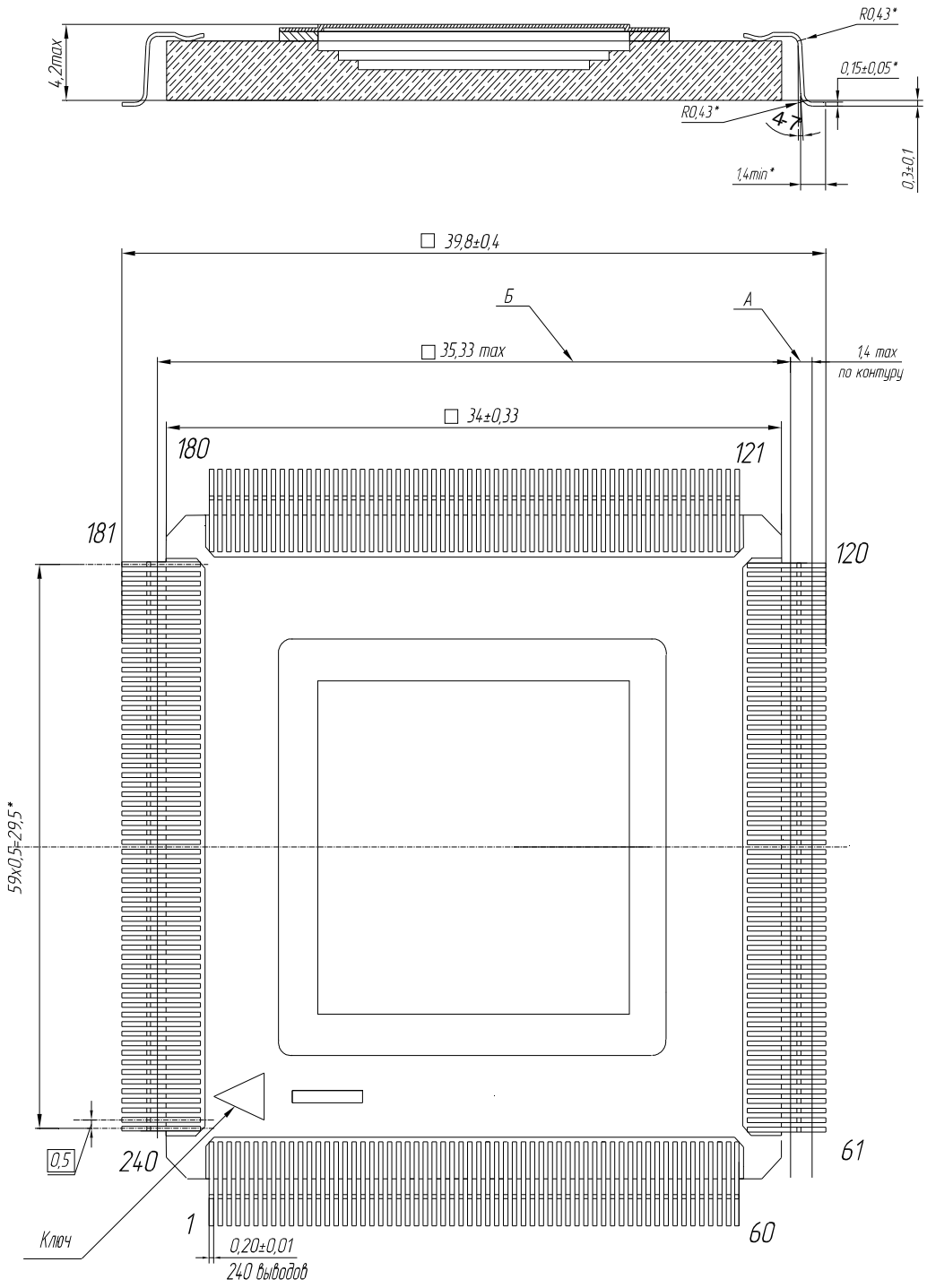


Рисунок 3.68 - Габаритные размеры и расположение выводов микросхемы

Таблица 3.58 - Назначение выводов микросхемы

Обозначение вывода	Тип вывода	Назначение вывода
Шина адреса/данных		
SysAD[63:0]	I/O	Разряды с 63-го по 0-й шины адреса-данных
Шина идентификатора адреса/данных		
SysCmd[8:0]	I/O	Шина идентификатора адреса/данных
Сигналы управления операциями записи/чтения		
RdRdy#	I	Сигнал готовности внешнего устройства к циклу чтения
WrRdy#	I	Сигнал готовности внешнего устройства к циклу записи
Шина внешних прерываний		
NMI#	I	Сигнал запроса внешнего немаскируемого прерывания
Int#[5:0]	I	Сигналы запросов внешних (маскируемых) прерываний
Контроллер JTAG		
TRST	I	Сброс логики тестирования
TMS	I	Выбор тестового режима
TCK	I	Тестовая частота
TDI	I	Вход тестовых данных
TDO	Z	Выход тестовых данных
DINT	I	Прерывание для отладки
Reset#	I	Сигнал «тёплого» перезапуска
ColdReset#	I	Сигнал «холодного» перезапуска
ValidIn#	I	Сигнал, показывающий, что внешнее устройство выставило корректные данные на шинах SysAD/SysCmd
ValidOut#	O	Сигнал, показывающий, что процессор выставил корректные данные на шинах SysAD/SysCmd
MasterClk	I	Входная частота шины процессора
U_{CC}OK	I	Сигнал, который должен стать активным после стабилизации питания и оставаться активным в течении всей работы процессора
ExtRqst#	I	Сигнал запроса от внешнего устройства к процессору освободить шины SysAD/SysCmd
Release#	O	Сигнал, сообщающий об освобождении процессором шин SysAD/SysCmd
GND		Общий
U_{CC}		Напряжение питания микросхемы (U_{CC}= U_{CC_output}= U_{CCR})
U_{CC_input}		Напряжение питания входов
U_{CC_output}		Напряжение питания выходных каскадов
U_{CC_core}		Напряжение питания внутренних логических элементов микросхемы
U_{CC_PLL}		Напряжение питания блока PLL

П р и м е ч а н и я

1 Сигналы за которыми следует символ “#” являются сигналами с низким активным уровнем.

2 В графе «Тип вывода» используются следующие обозначения:

I – вход,

O – выход,

I/O – комбинированный вывод (вход/выход),

Z – выход в состоянии "Выключено".

Табл ица 3.59 - Обозначение и номера выводов микросхемы

№	Обозначен.	№	Обозначен.	№	Обозначен.	№	Обознач.	№	Обозначен.
1	Резерв*	**49**	SysAD61	**97**	SysAD43	**145**	SysCmd4	**193**	Резерв*
2	Резерв*	**50**	SysAD60	**98**	SysAD42	**146**	U_{CC_input}	**194**	U_{CC_output}
3	Резерв*	**51**	SysAD59	**99**	SysAD41	**147**	GND	**195**	U_{CC_core}
4	Резерв*	**52**	U_{CC_output}	**100**	GND	**148**	GND	**196**	SysAD20
5	GND	**53**	U_{CC_core}	**101**	GND	**149**	GND	**197**	SysAD19
6	GND	**54**	U_{CC_input}	**102**	SysAD40	**150**	SysCmd3	**198**	SysAD18
7	U_{CC_input}	**55**	GND	**103**	Резерв*	**151**	SysCmd2	**199**	SysAD17
8	Int0#	**56**	GND	**104**	SysAD39	**152**	SysCmd1	**200**	GND
9	Int1#	**57**	Резерв*	**105**	SysAD38	**153**	SysCmd0	**201**	GND
10	Int2#	**58**	Резерв*	**106**	U_{CC_core}	**154**	Release#	**202**	SysAD16
11	Int3#	**59**	Резерв*	**107**	U_{CC_output}	**155**	U_{CC_core}	**203**	Резерв*
12	Int4#	**60**	Резерв*	**108**	SysAD37	**156**	U_{CC_output}	**204**	SysAD15
13	Int5#	**61**	Резерв*	**109**	SysAD36	**157**	ValidOut#	**205**	SysAD14
14	U_{CC_core}	**62**	Резерв*	**110**	SysAD35	**158**	ValidIn#	**206**	U_{CC_output}
15	U_{CC}OK	**63**	Резерв*	**111**	SysAD34	**159**	SysAD31	**207**	U_{CC_core}
16	ColdReset#	**64**	Резерв*	**112**	GND	**160**	GND	**208**	SysAD13
17	Reset#	**65**	GND	**113**	GND	**161**	GND	**209**	SysAD12
18	NMI#	**66**	GND	**114**	U_{CC_input}	**162**	SysAD30	**210**	SysAD11
19	MasterClk	**67**	U_{CC_input}	**115**	U_{CC_core}	**163**	SysAD29	**211**	SysAD10
20	Резерв*	**68**	U_{CC_core}	**116**	GND	**164**	SysAD28	**212**	GND
21	Резерв*	**69**	U_{CC_output}	**117**	Резерв*	**165**	SysAD27	**213**	GND
22	Резерв*	**70**	SysAD58	**118**	Резерв*	**166**	U_{CC_core}	**214**	GND
23	GND	**71**	SysAD57	**119**	Резерв*	**167**	U_{CC_output}	**215**	SysAD9
24	U_{CC_PLL}	**72**	SysAD56	**120**	Резерв*	**168**	SysAD26	**216**	U_{CC_input}
25	U_{CC_core}	**73**	Резерв*	**121**	Резерв*	**169**	SysAD25	**217**	SysAD8
26	U_{CC_output}	**74**	GND	**122**	Резерв*	**170**	SysAD24	**218**	Резерв*
27	GND	**75**	GND	**123**	Резерв*	**171**	Резерв*	**219**	SysAD7
28	GND	**76**	SysAD55	**124**	Резерв*	**172**	GND	**220**	U_{CC_output}
29	GND	**77**	SysAD54	**125**	GND	**173**	GND	**221**	U_{CC_core}
30	TCK	**78**	SysAD53	**126**	GND	**174**	U_{CC_input}	**222**	SysAD6
31	TMS	**79**	SysAD52	**127**	U_{CC_input}	**175**	U_{CC_core}	**223**	SysAD5
32	TDI	**80**	U_{CC_core}	**128**	U_{CC_core}	**176**	GND	**224**	SysAD4
33	TRST	**81**	U_{CC_output}	**129**	U_{CC_output}	**177**	Резерв*	**225**	SysAD3
34	DINT	**82**	SysAD51	**130**	SysAD33	**178**	Резерв*	**226**	GND
35	U_{CC_core}	**83**	SysAD50	**131**	SysAD32	**179**	Резерв*	**227**	GND
36	U_{CC_input}	**84**	SysAD49	**132**	Резерв*	**180**	Резерв*	**228**	SysAD2
37	GND	**85**	SysAD48	**133**	ExrRqst#	**181**	Резерв*	**229**	SysAD1
38	Резерв*	**86**	U_{CC_input}	**134**	GND	**182**	Резерв*	**230**	SysAD0
39	Резерв*	**87**	GND	**135**	GND	**183**	Резерв*	**231**	Резерв*
40	U_{CC_core}	**88**	GND	**136**	WrRdy#	**184**	Резерв*	**232**	U_{CC_output}
41	U_{CC_output}	**89**	GND	**137**	RdRdy#	**185**	GND	**233**	U_{CC_core}
42	Резерв*	**90**	Резерв*	**138**	Резерв*	**186**	U_{CC_core}	**234**	U_{CC_input}
43	TDO	**91**	SysAD47	**139**	SysCmd8	**187**	U_{CC_input}	**235**	GND
44	Резерв*	**92**	SysAD46	**140**	U_{CC_core}	**188**	GND	**236**	GND
45	SysAD63	**93**	SysAD45	**141**	U_{CC_output}	**189**	GND	**237**	Резерв*
46	GND	**94**	SysAD44	**142**	SysCmd7	**190**	SysAD23	**238**	Резерв*
47	GND	**95**	U_{CC_core}	**143**	SysCmd6	**191**	SysAD22	**239**	Резерв*
48	SysAD62	**96**	U_{CC_output}	**144**	SysCmd5	**192**	SysAD21	**240**	Резерв*

Примечания

1 Сигналы за которыми следует символ “#” являются сигналами с низким активным уровнем.

2 Резерв* - оставить не подключенным.

3.14 Электрические параметры 1890ВМ3Т

Таблица 3.60 - Электрические параметры микросхемы при приемке и поставке

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды, єС
		не менее	не более	
1	2	3	4	5
Выходное напряжение низкого уровня, В при I_{OL}= 4,0 мА, U_{CC}=3,1 В	U_{OL}	-	0,4	от -60 до +85**
Выходное напряжение высокого уровня, В при I_{OH}= -4,0 мА,U_{CC}=3,1 В	U_{OH}	2,4	-	от -60 до +85**
Ток потребления статический, мА при U_{CC}=3,5 В, U_{IH}=2,0 В(2,4В*), U_{IL}=0,8 В(0,4 В*)	I_{CC}	-	100	от -60 до +85**
Ток потребления динамический (при максимальной рабочей частоте), А при U_{CC}=3,5 В, U_{IH}=2,0 В(2,4 В*), U_{IL}=0,8 В(0,4 В*), f=120 МГц	I_{0CC}	-	1,5	от -60 до +85**
Ток утечки низкого уровня на входе, мкА при U_{CC}=3,5 В, U_{IL}=0,0 В	I_{ILL}	-10	-	от -60 до +85**
Ток утечки высокого уровня на входе, мкА при U_{CC}=3,5 В, U_{IH}=3,5 В	I_{ILH}	-	10	от -60 до +85**
Ток высокого уровня на входе, привязанном к общему уровню***, мкА при U_{CC}=3,5 В, U_{IH}=3,5 В	I_{IHPD}	2	400	от -60 до +85**
Выходной ток низкого уровня в состоянии «Выключено», мкА при U_{CC}=3,5 В, U_{OL}=0,0 В	I_{OZL}	-10	-	от -60 до +85**
Выходной ток высокого уровня в состоянии «Выключено», мкА при U_{CC}=3,5 В, U_{OH}=3,5 В	I_{OZH}	-	10	от -60 до +85**
Входная емкость, пФ	C_I	-	15	25 ± 10
Емкость входа/выхода, пФ	$C_{I/O}$	-	15	25 ± 10
Выходная емкость, пФ	C_O	-	15	25 ± 10

* Для входов Reset#, TCK, ColdReset#, MasterClk.

** Указана температура корпуса.

*** Для входов TRST, TMS, TCK, TDI, DINT.

Таблица 3.61 - Предельно допустимые и предельные электрические режимы эксплуатации в диапазоне температур среды

Наименование параметра режима, единица измерения	Буквенное обозначение	Норма			
		Предельно-допустимый режим		Предельный режим	
		не менее	не более	не менее	не более
Напряжение питания, В	U_{CC}	3,1	3,5	-0,3	4,3
Входное напряжение высокого уровня, В	U_{IH}	2,0	U_{cc}+0,2	-	U_{cc}+0,3
		2,4*	U_{cc}+0,2*		U_{cc}+0,3*
		2,0**	5,5**		5,7**
Входное напряжение низкого уровня, В	U_{IL}	-0,2 -0,2*	0,8 0,4*	-0,3	-
Напряжение, прикладываемое к выходу микросхемы в состоянии "Выключено", В	U_{OZ}	-0,2 -0,2***	U_{cc}+0,2 5,5***	-0,3	U_{cc}+0,3 5,7***
Выходной ток низкого уровня, мА	I_{OL}	-	4,0	-	5,2
Выходной ток высокого уровня, мА	I_{OH}	-4,0	-	-5,2	-
Рабочая частота, МГц	f	20	120	-	-
		0****	60****		
Емкость нагрузки каждого выхода, пФ	C_L	-	50	-	150
Длительность фронта и спада входных сигналов, нс	t_{LH}, t_{HL}	-	3	-	50

* Для входов Reset#, TCK, ColdReset#, MasterClk.

** Для входов TCK, TMS, TDI, TRST, DINT, Int#[5:0], U_{CC}OK, ColdReset#, Reset#, NMI#.

*** Для выхода TDO.

**** При работе без внутреннего умножения частоты.

3.15 Динамические параметры 1890ВМ3Т

На рисунках 3.69 и 3.70 представлены временные зависимости.

Динамические параметры, изображенные на рисунках, приведены в таблице 3.62.

Таблица 3.62 - Динамические параметры в диапазоне температур от минус 60 до плюс 85єС, Ucc =3,3В ± 5%, нагрузка 25 пФ (выходной сигнал фиксировался по уровню Ucc/2)

Наименование параметра	Сигналы	Направ-ление	Норма, нс		Примечание
			min	max	
t_{DS} – время предустановки к фронту LH MasterClk	SysAD[63:0]	Вход	3,0	-	Рисунок 3.69
	SysCmd[8:0]		3,0	-	
	RdRdy_		3,0	-	
	WrRdy_		3,0	-	
	ValidIn_		3,0	-	
	ExtRqst_		3,0	-	
t_{DH} – время удержания от фронта LH MasterClk	SysAD[63:0]	Вход	1,5	-	
	SysCmd[8:0]		1,0	-	
	RdRdy_		1,0	-	
	WrRdy_		1,0	-	
	ValidIn_		1,0	-	
	ExtRqst_		1,0	-	
t_{DO} – время установки корректных значений от фронта LH MasterClk	SysAD[63:0]	Выход	-	15	Рисунок 3.70
	SysCmd[8:0]		-	15	
	ValidOut_		-	12	
	Release_		-	12	
t_{DOH} – время удержания корректных значений от фронта LH MasterClk	SysAD[63:0]	Выход	3,0	-	

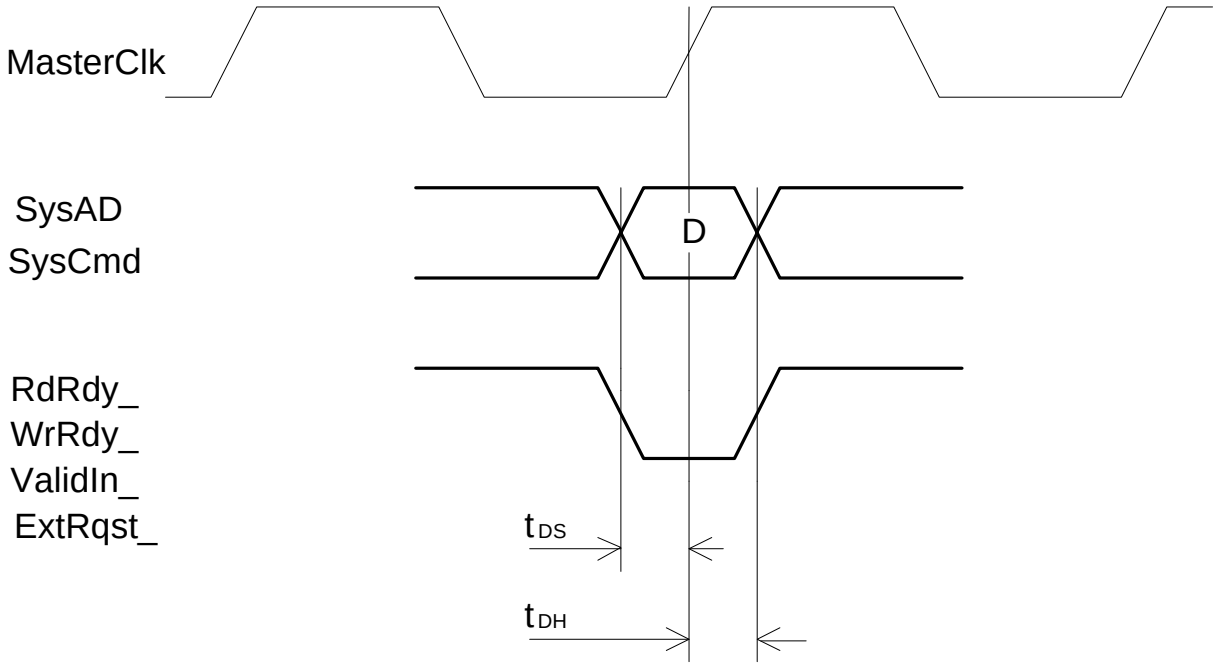


Рисунок 3.69 - Предустановка и удержание входных сигналов

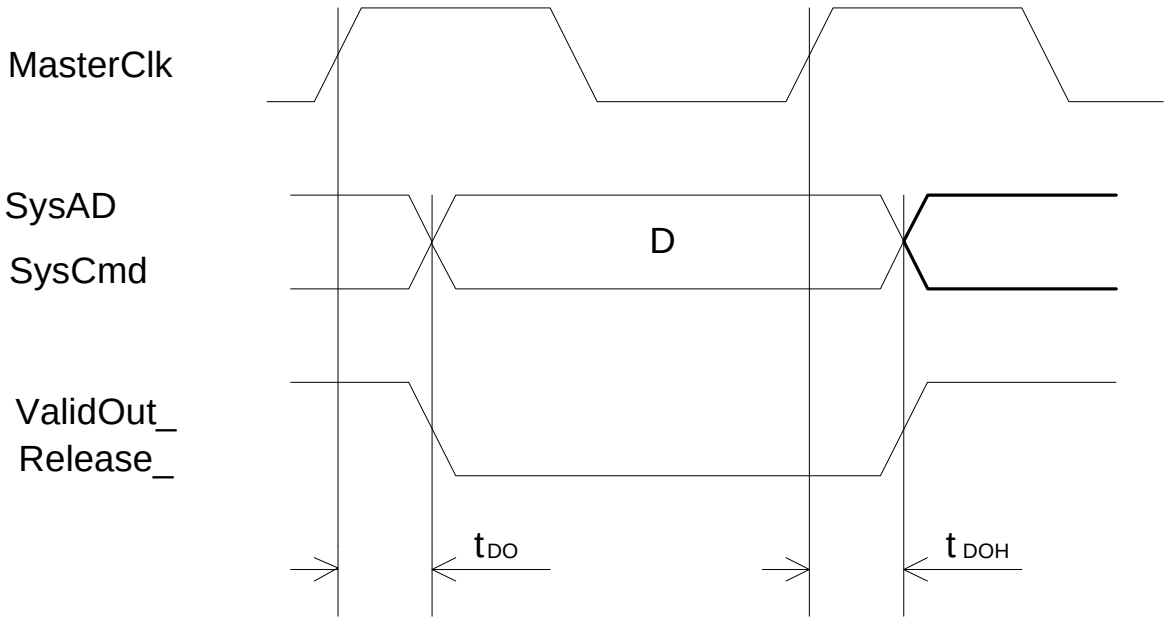


Рисунок 3.70 - Установка выходных сигналов

4 Указания по применению и монтажу

Указания по применению и эксплуатации микросхем - по ОСТ В 11 0998.

Устанавливать и извлекать микросхемы из контактирующих устройств, а также производить их замену необходимо только после снятия напряжения со всех выводов контактирующего устройства.

Режимы и условия монтажа в аппаратуре микросхем – по ОСТ 11 073.063.

Рекомендуется начинать пайку с выводов земли и питания. Пайку остальных выводов разрешается проводить в любой последовательности.

Порядок подачи на микросхему напряжения питания и входных сигналов:

общая точка, U_{CC}, входные напряжения.

При измерениях и эксплуатации микросхем должны быть приняты меры исключающие возможность накопления электростатических зарядов на выводах микросхемы в соответствии с ОСТ 11 073.062.

Для влагозащиты плат с микросхемами рекомендуется применять лак УР-231 ТУ6-21-14 или ЭП-730 ГОСТ 20824 в три слоя.

Габаритные размеры и расположение выводов микросхемы в корпусе приведены:

- на рисунке 3.68 для микросхемы в корпусе 240CQFP с формованными выводами (ЮКСУ.431281.029ТУ,А);
- на рисунке 3.69 для микросхемы в корпусе 240CQFP с неформованными выводами в технологической рамке (ЮКСУ.431281.029ТУ).

Типовая схема включения микросхемы 1890ВМ3Т приведена на рисунке 4.1.

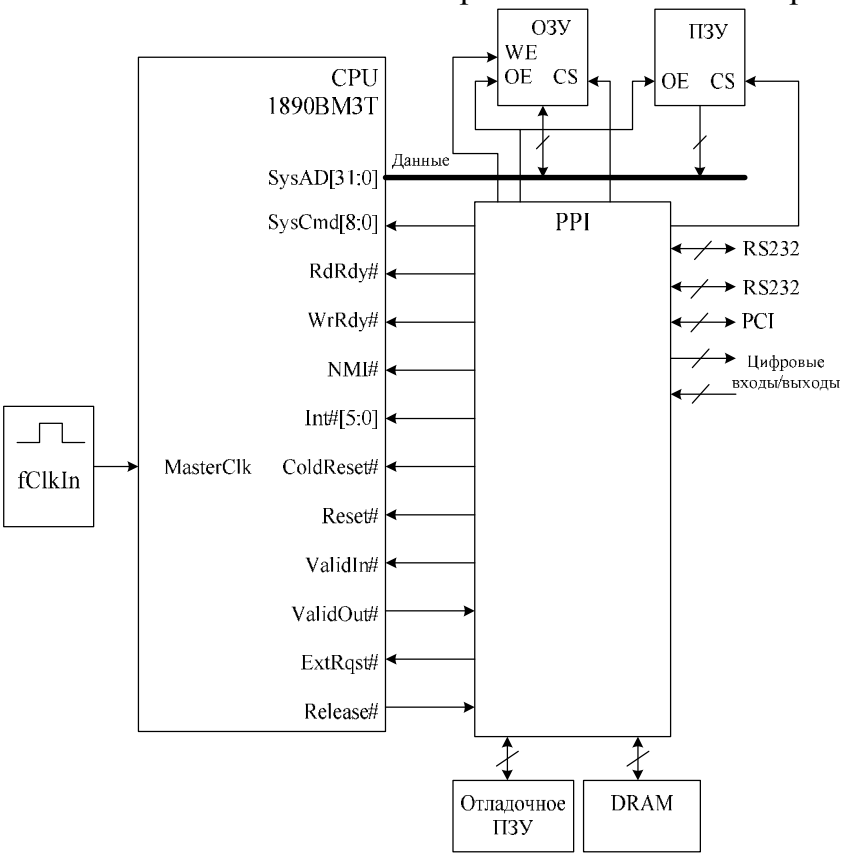


Рисунок 4.1 – Типовая схема включения 1890ВМ3Т

Для работы в режиме ширины шины 32 бита необходимо подключить неиспользуемые сигналы шины SysAD к общей точке через резистор 10 кОм и установить режим ширины шины SysAD 32 бита (Int# [4]=1). Кодировка начальных режимов работы приведена в таблице 3.54 настоящего документа.

5 Применение в режимах и условиях, не предусмотренных в ТУ

Если в процессе разработки РЭА к микросхемам предъявляются требования применения в режимах и условиях, отличных от установленных в ТУ на микросхемы, потребитель проводит испытания и тщательное исследование работы микросхем в таких режимах и условиях.

При положительных результатах испытаний применение микросхем в этих режимах и условиях согласовывается с потребителем по ОСТ 11 0492.

6 Требования по безопасности

При применении микросхем соблюдать требования безопасности, установленные «Правилами техники безопасности и производственной санитарии в электронной промышленности», изд. «Энергия», Москва, 1973, раздел «К», ГОСТ 12.2.007.0, ГОСТ 12.3.002, а также требования безопасности, установленные при работе с устройствами вычислительной техники и другой радиоэлектронной аппаратуры.

Приложение А

Зависимости основных электрических параметров микросхемы от режимов и условий эксплуатации

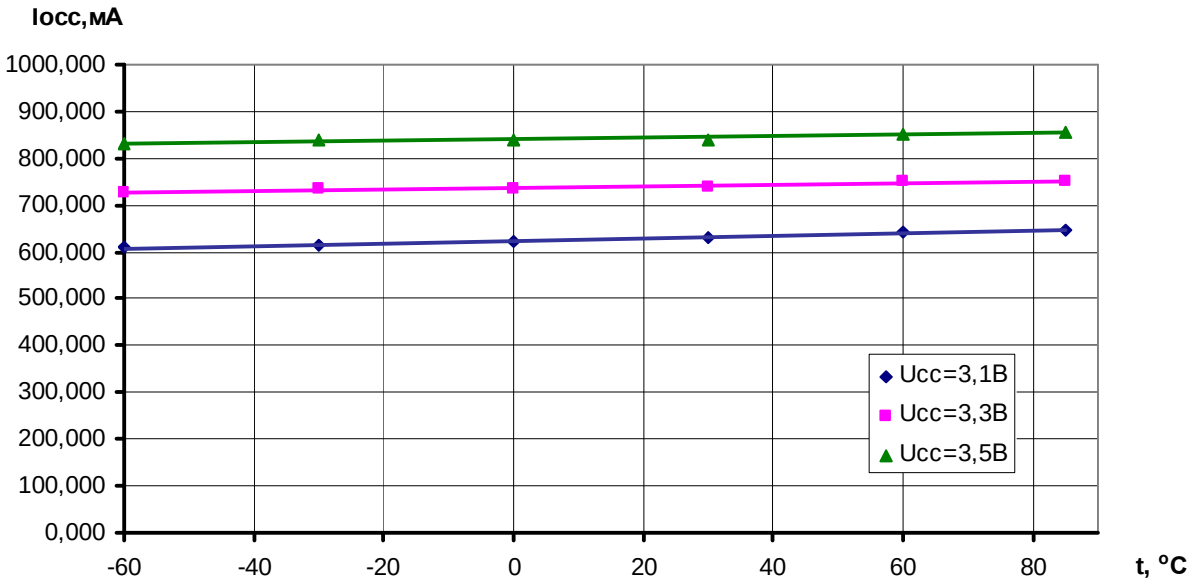


Рисунок А.1 - Зависимость динамического тока потребления (Iocc) от температуры (t) в диапазоне напряжений питания

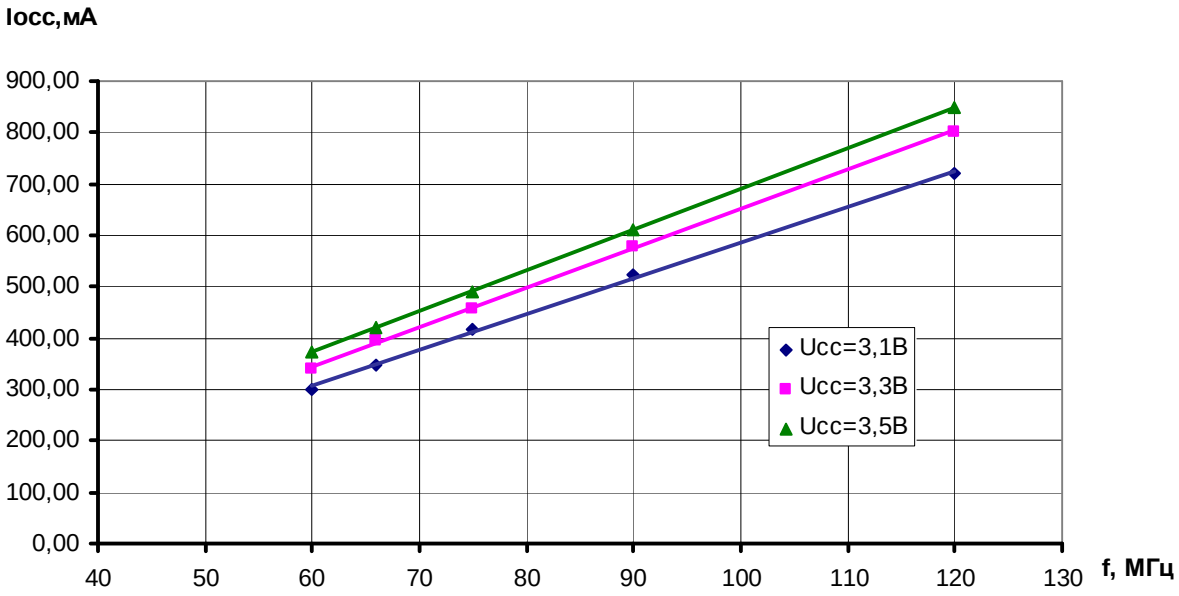


Рисунок А.2 - Зависимость динамического тока потребления (Iocc) от рабочей частоты (f) в диапазоне напряжений питания

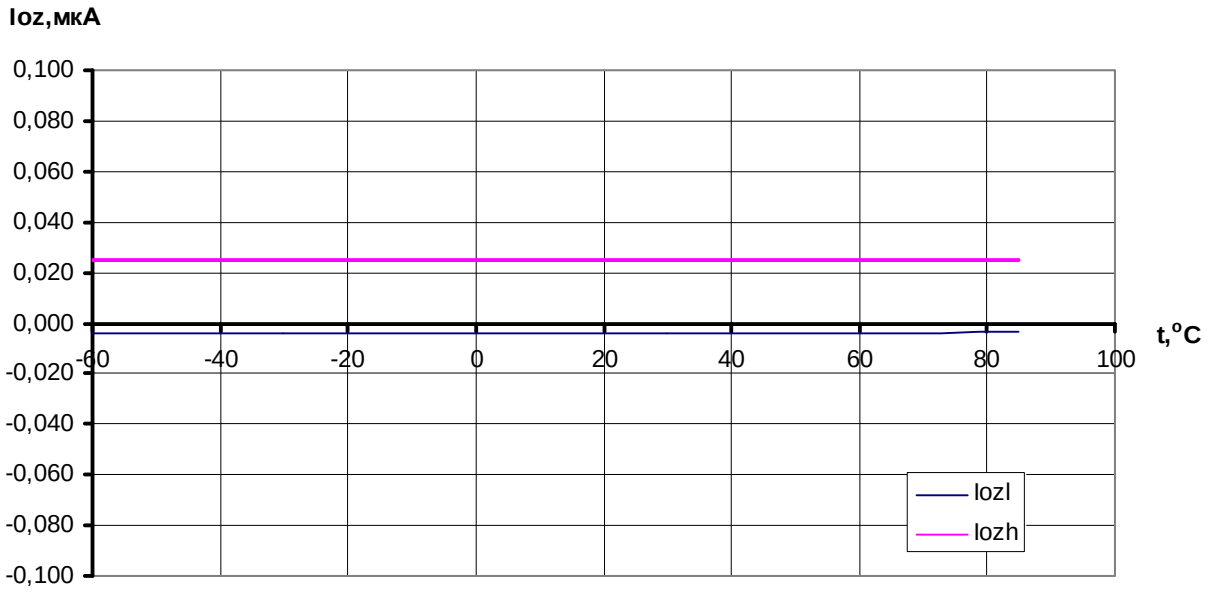


Рисунок А.3 - Зависимость тока утечек высокого и низкого уровня по выходам (Iozh и Iozl) от температуры (t)

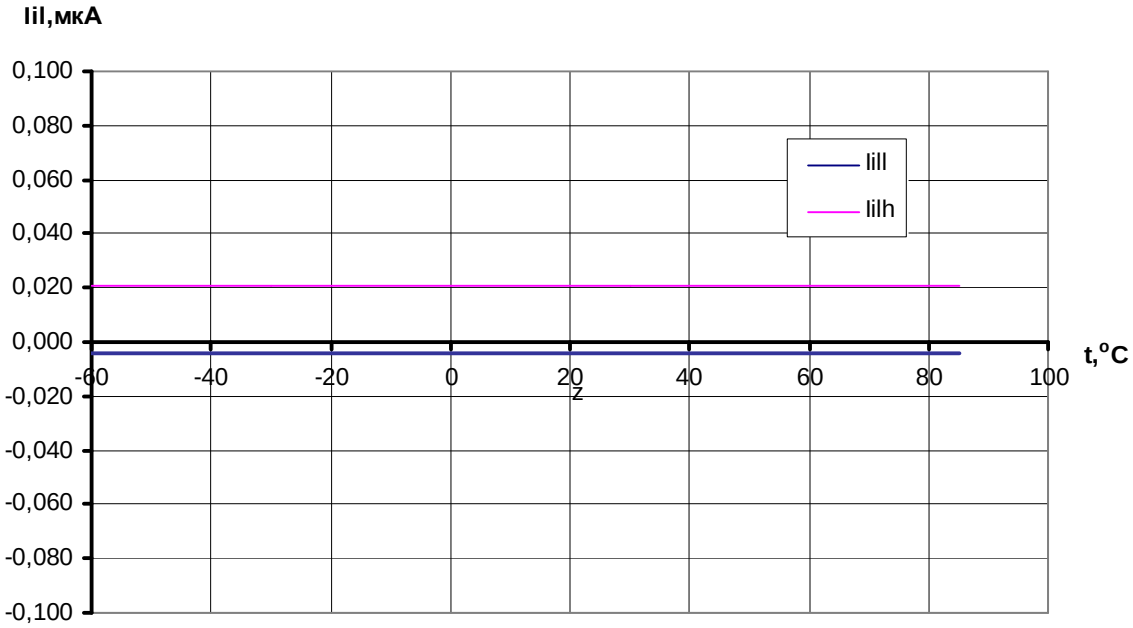


Рисунок А.4 - Зависимость тока утечек высокого и низкого уровня по входам (Iilh и Iill) от температуры (t)

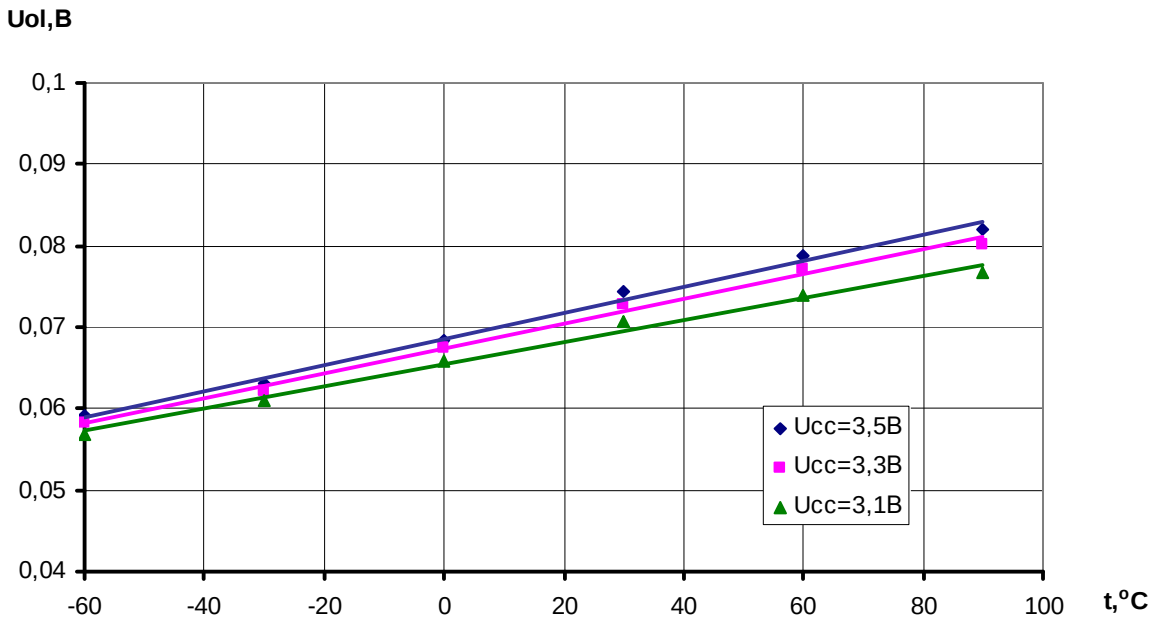


Рисунок А.5 - Зависимость выходных напряжений низкого логического уровня от температуры среды (при Io=4мА)

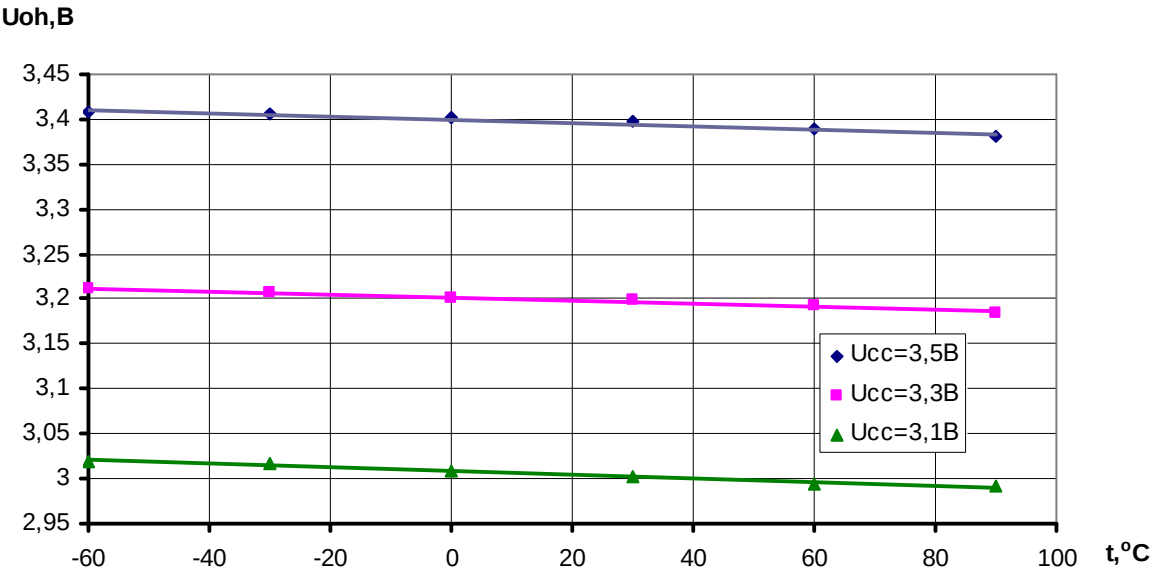


Рисунок А.6 - Зависимость выходных напряжений высокого логического уровня от температуры среды (при Io=4мА)

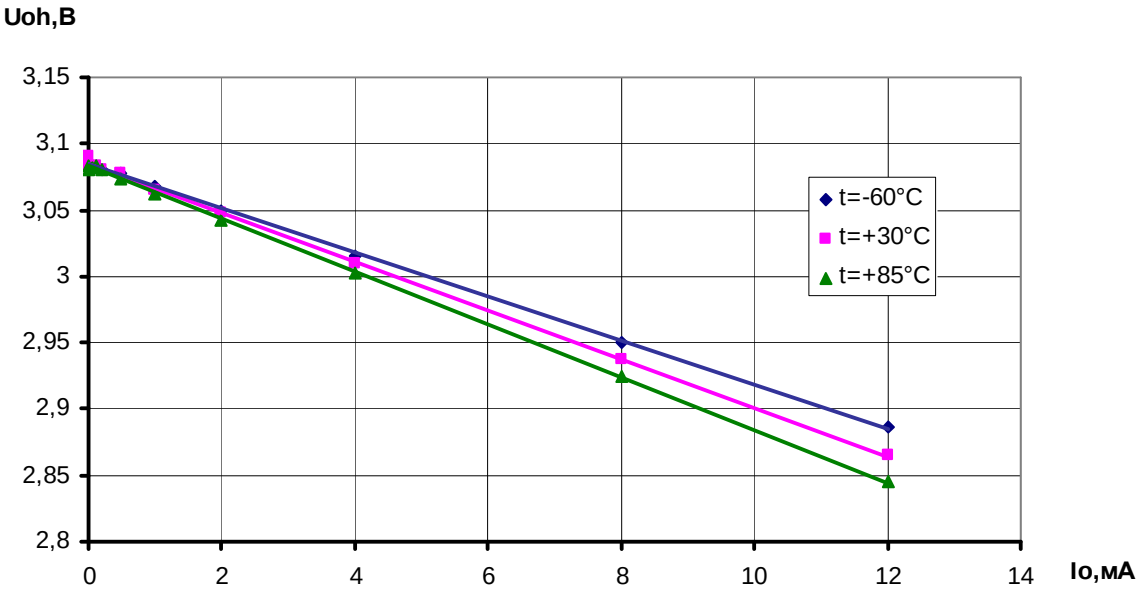


Рисунок А.7 - Зависимость выходного напряжения высокого логического уровня от выходного тока (при Ucc=3,1 В)

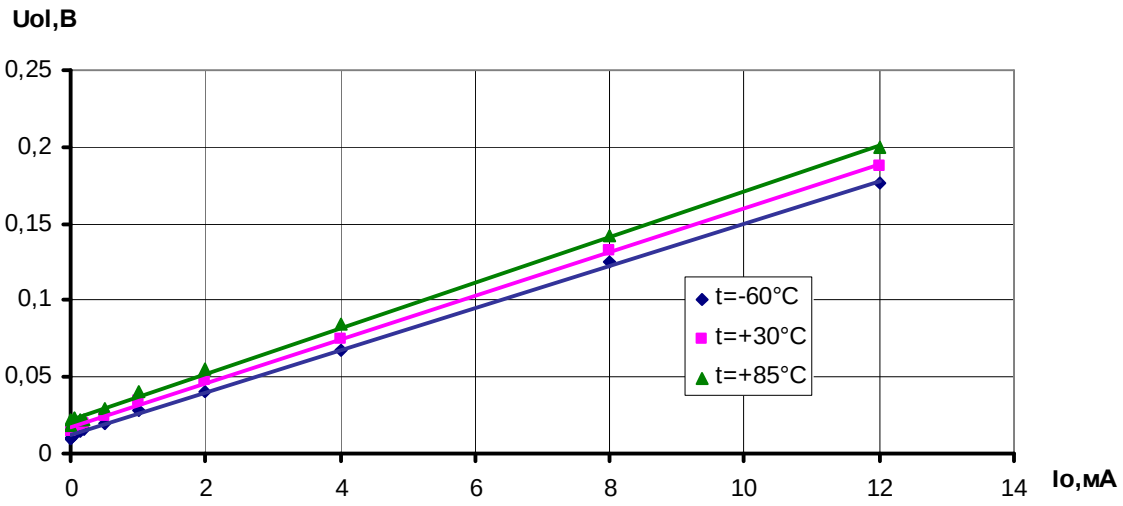


Рисунок А.8 - Зависимость выходного напряжения низкого логического уровня от выходного тока (при Ucc=3,1В)

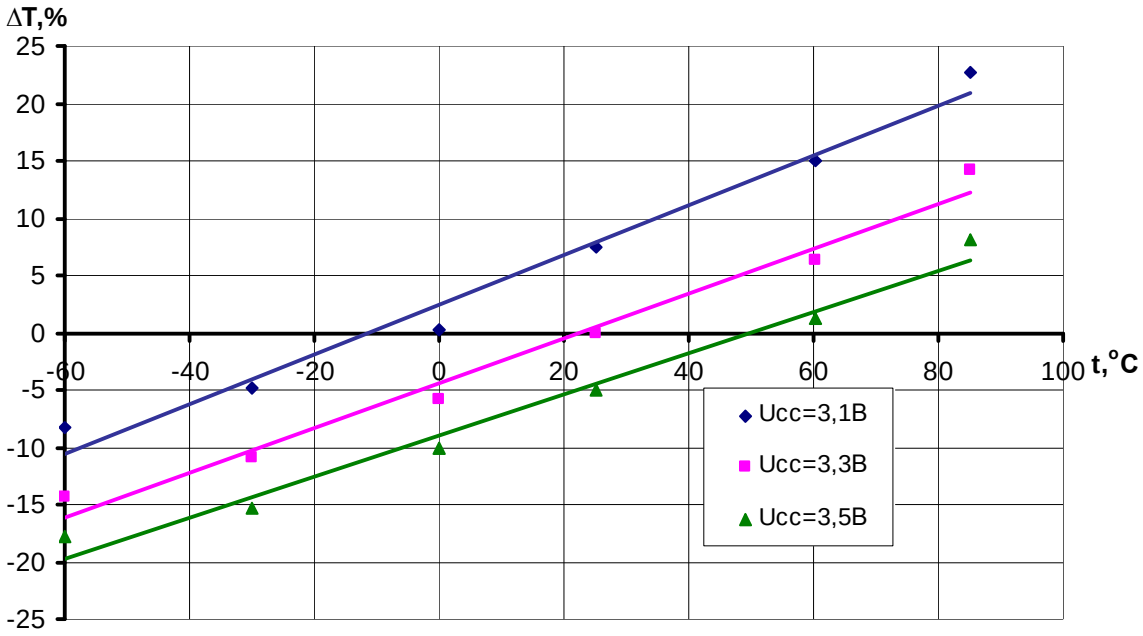


Рисунок А.9 - Зависимость временных задержек (ΔТ) от температуры (t) в диапазоне напряжений питания

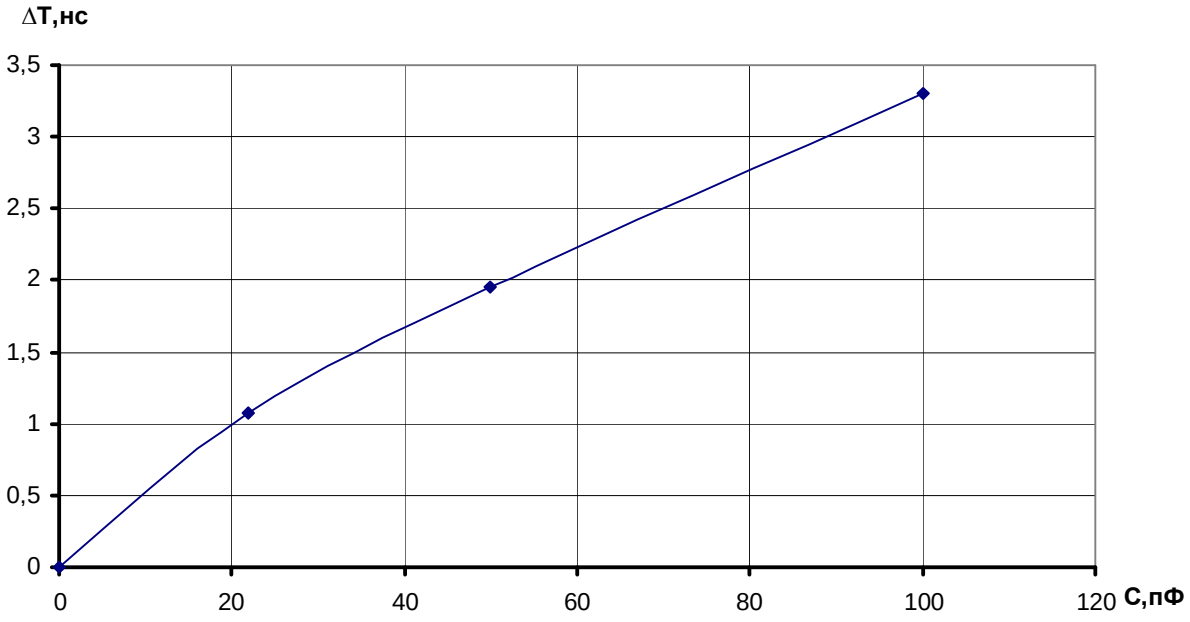


Рисунок А.10 - Зависимость изменения временных задержек (ΔТ) от ёмкости (С)

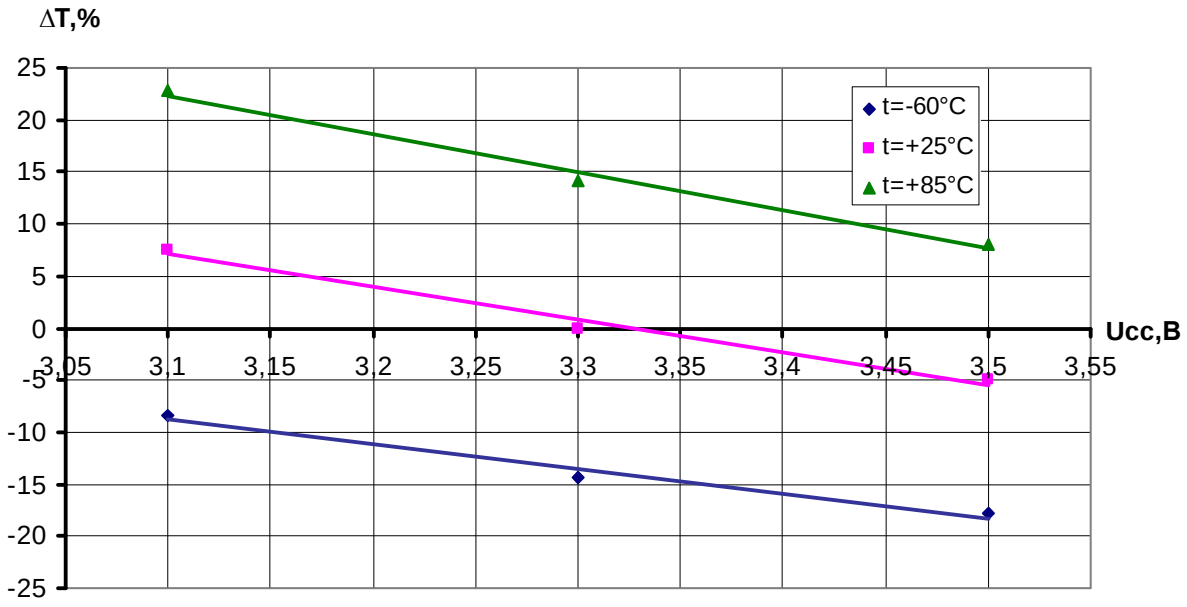


Рисунок А.11 - Зависимость изменения временных задержек (ΔT) от напряжения питания (Ucc) в диапазоне температур

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	Изъятых					
1	-	Все	-	-	-	ЮКСУ. 00.0503-05			
2	1о	-	-	-	-	ЮКСУ. 00.0504-05			
3	-	Все	-	-	-	ЮКСУ. 00.0529-05			
4	1о	-	-	-	-	ЮКСУ. 00.0530-05			