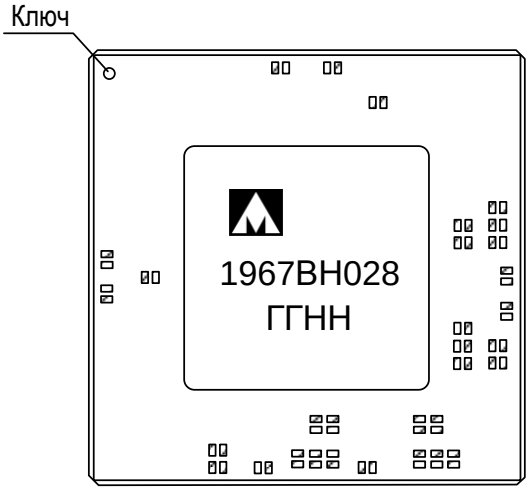
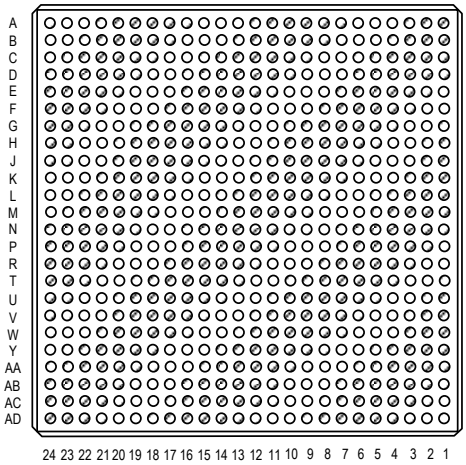
Сигнальный процессор со статической суперскалярной архитектурой 1967ВН028, К1967ВН028, К1967ВН028К, 1967ВН02Н4, К1967ВН02Н4



Лицевая сторона



Обратная сторона

ГГ – год выпуска
НН – неделя выпуска

Обозначение	Диапазон
1967ВН028	минус 60 –105 °С
К1967ВН028	минус 60 – 105 °С
К1967ВН028К	0 – 70 °С

Тип корпуса:

– 576-ти выводной металлокерамический корпус типа BGA МК8303.576-1;
– микросхемы 1967ВН02Н4 и К1967ВН02Н4 поставляются в бескорпусном исполнении.

Основные технические характеристики процессора

- Рабочая частота до 450 МГц, время цикла исполнения команды 2,2 нс;
- Пиковая производительность – 12 операций с плавающей точкой одинарной точности за такт;
- Встроенная оперативная память типа SRAM 24 Мбит;
- Корпус типа BGA повышенной термостойкости 25 х 25 мм (576 шариковых вывода);
- Два вычислительных блока, каждый из которых содержит АЛУ, умножитель, сдвигатель и коммуникационный блок (CLU);
- Два целочисленных АЛУ, обеспечивающих адресацию данных и содержащих буферы выравнивания данных (DAB).
- Интегрированная система ввода-вывода включает 14-канальный контроллер DMA, внешний 64/32-разрядный порт, 4 высокоскоростных двунаправленных LVDS порта передачи данных, контроллер SDRAM, пользовательские линии ввода-вывода и внешний флаг переполнения таймера для синхронизации системы;
- Два таймера;
- Порт доступа к интерфейсу JTAG (совместимому со стандартом 1149.1 IEEE) для эмуляции на кристалле;
- Формат данных: числа с плавающей точкой одинарной (32 бита) и двойной (64 бита) точности. Числа с фиксированной точкой: 8, 16, 32 и 64 бит.

Основные преимущества и области применения микросхемы

- Процессор обеспечивает высокопроизводительную суперскалярную цифровую обработку сигнала, оптимизированную для применения в телекоммуникациях или других областях, требующих мультипроцессорной системы цифровой обработки данных.
- Процессор особенно эффективен в алгоритмах цифровой обработки сигнала и системах ввода-вывода.
- Поддерживает на низком уровне дополнительные передачи через устройство прямого доступа к памяти между внутренней памятью, внешней памятью, отображаемыми в памяти периферийными устройствами, LVDS портами, главным процессором и другими (мультипроцессорными) устройствами цифровой обработки данных.
- Упрощает процесс программирования процессора цифровой обработки сигналов (DSP) за счет гибкого набора команд и использования языка высокого уровня в архитектуре DSP.
- Позволяет использовать изменяемую мультипроцессорную систему с низкими потерями пропускной способности.
- Обеспечивает встроенную арбитражную систему мультипроцессорной обработки данных без дополнительных устройств.

1 Структурная блок-схема микросхемы

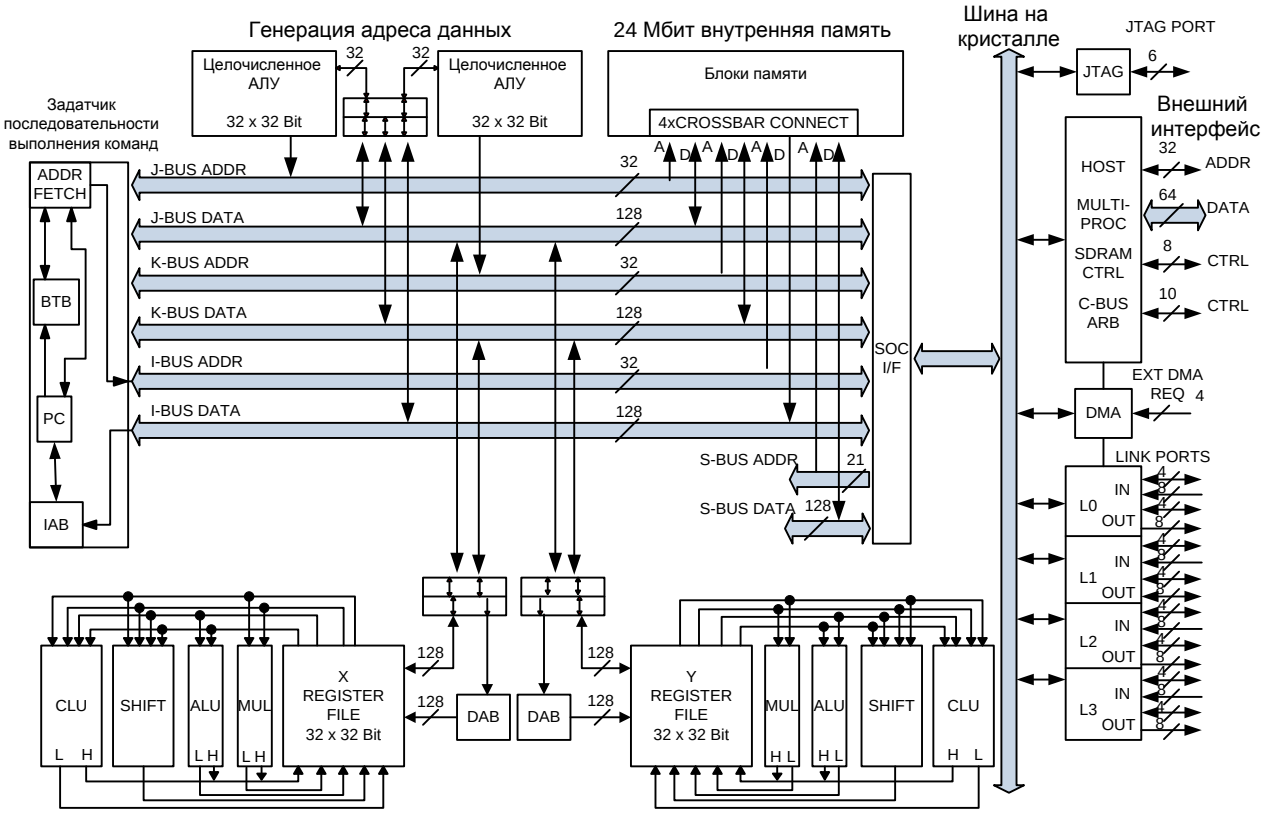


Рисунок 1 – Структурная блок-схема микросхемы

2 Условное графическое обозначение

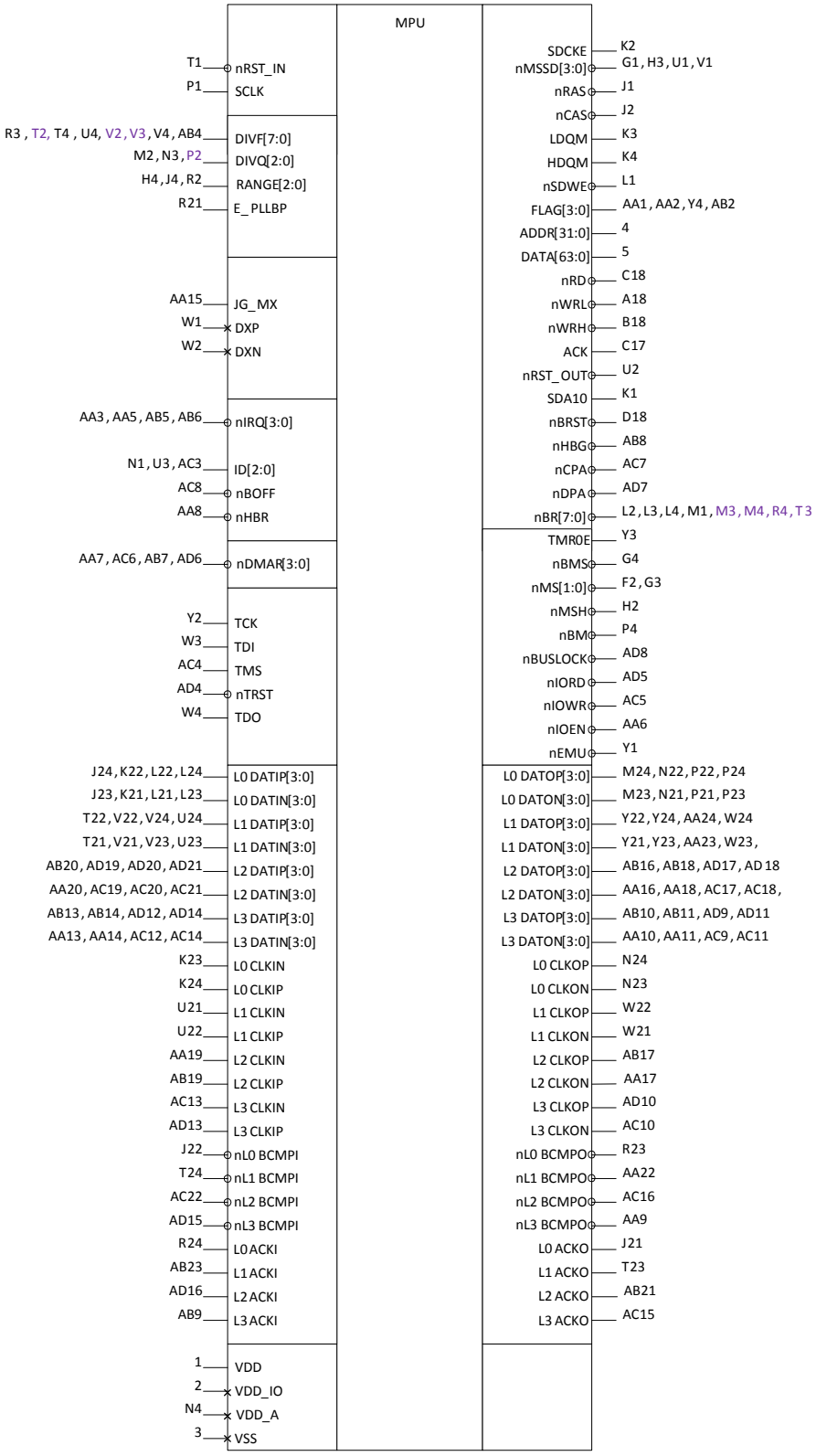


Рисунок 2 –Условное графическое обозначение

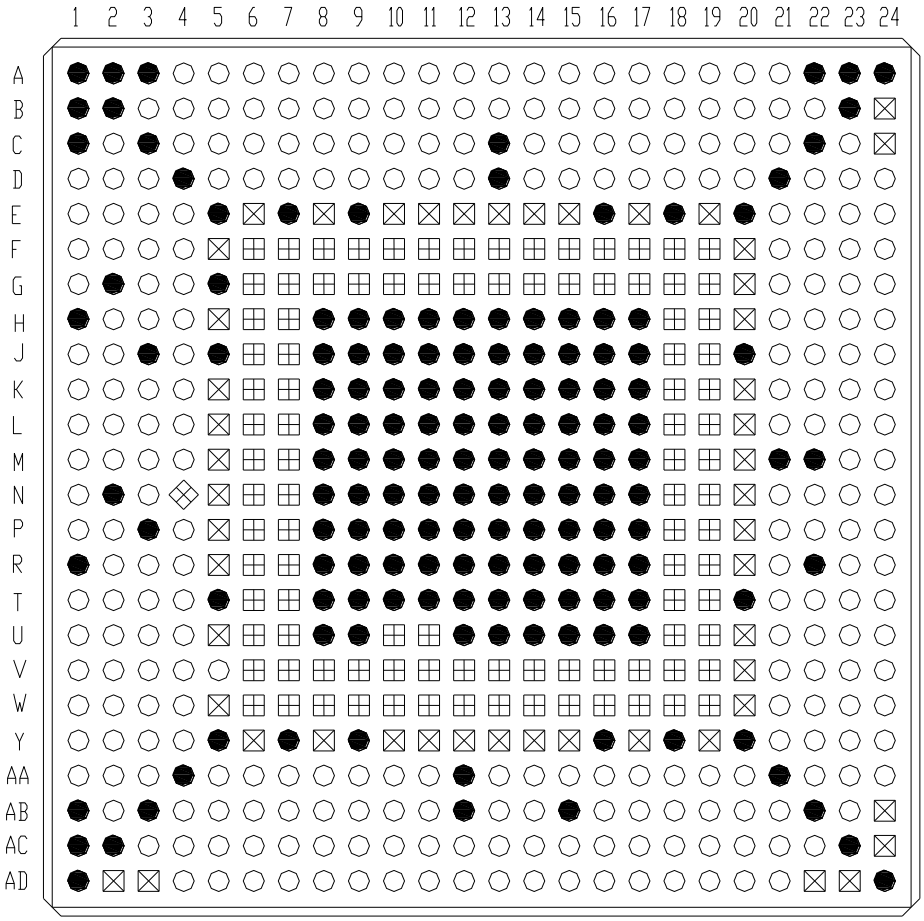
1 VDD – группа выводов: F6 – F19; G6 – G19, H6, H7, H18, H19, J6, J7, J18, J19, K6, K7, K18, K19, L6, L7, L18, L19, M6, M7, M18, M19, N6, N7, N18, N19, P6, P7, P18, P19, R6, R7, R18, R19, T6, T7, T18, T19, U6, U7, U10, U11, U18, U19, V6 – V19, W6 – W19;

2 VDD_IO – группа выводов: B24, C24, E6, E8, E10-E15, E17, E19, F5, F20, G20, H5, H20, K5, K20, L5, L20, M5, M20, N5, N20, P5, P20, R5, R20, U5, U20, V20, W5, W20, Y6, Y8, Y10-Y15, Y17, Y19, AB24, AC24, AD2, AD3, AD22, AD23;

3 VSS – группа выводов: A1 – A3, A22 – A24, B1, B2, B23, C1, C3, C13, C22, D4, D13, D21,E5, E7, E9, E16, E18, E20, G2, G5, H1, H8 – H17, J3, J5, J8 – J17, J20, K8 – K17, L8 – L17, M8 – M17, M21, M22, N2, N8 – N17, P3, P8-P17, R1, R8 – R17, R22, T5, T8 – T17, T20, U8, U9, U12 – U17, V5, Y5, Y7, Y9, Y16, Y18, Y20, AA4, AA12, AA21, AB1, AB3, AB12, AB15, AB22, AC1, AC2, AC23, AD1, AD24;

4 ADDR – группа выводов: A19 – A21, B19 – B22, C19 – C21, C23, D19, D20, D22 – D24, E21 – E24, F21 – F24, G21 – G24, H21 – H24;

5 DATA – группа выводов: A4 – A17, B3 – B17, C2, C4 – C12, C14 – C16, D1 – D3, D5 – D12, D14 – D17, E1 – E4, F1, F3, F4.



Обозначения:

○ выводные площадки сигналов;

⊞ VDD – питание ядра;

⊠ VDD_IO – питание ввода/вывода;

◈ VDD_A – питание аналоговых блоков;

● VSS – общий

Рисунок 3 – Схема расположения выводных площадок питания и сигнала «Общий»

3 Описание выводов

Таблица 1 – Описание выводов микросхемы в корпусе МК8303.576-1

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
A1	VSS	PWR	Общий
A2			
A3			
A4	DATA[49]	IO (pu_1)	Шина данных внешнего интерфейса
A5	DATA[43]		
A6	DATA[41]		
A7	DATA[37]		
A8	DATA[33]		
A9	DATA[29]		
A10	DATA[25]		
A11	DATA[23]		
A12	DATA[19]		
A13	DATA[15]		
A14	DATA[11]		
A15	DATA[9]		
A16	DATA[5]		
A17	DATA[1]		
A18	nWRL	IO (pu_2_id0)	Запись младшего слова внешней памяти (кроме SDRAM)
A19	ADDR[30]	IO (pu_1)	Шина адреса внешнего интерфейса
A20	ADDR[28]		
A21	ADDR[22]		
A22	VSS	PWR	Общий
A23			
A24			
B1			
B2			
B3	DATA[51]	IO (pu_1)	Шина данных внешнего интерфейса
B4	DATA[50]		
B5	DATA[44]		
B6	DATA[42]		
B7	DATA[38]		
B8	DATA[34]		
B9	DATA[30]		
B10	DATA[26]		
B11	DATA[24]		
B12	DATA[20]		
B13	DATA[16]		
B14	DATA[12]		
B15	DATA[10]		
B16	DATA[6]		
B17	DATA[2]		
B18	nWRH	IO (pu_2_id0)	Запись старшего слова внешней памяти (кроме SDRAM)

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
B19	ADDR[31]	IO (pu_1)	Шина адреса внешнего интерфейса
B20	ADDR[29]		
B21	ADDR[23]		
B22	ADDR[21]	IO (pu_1)	Шина адреса внешнего интерфейса
B23	VSS	PWR	Общий
B24	VDD_IO	PWR	Питание ввода/вывода
C1	VSS	PWR	Общий
C2	DATA[53]	IO (pu_1)	Шина данных внешнего интерфейса
C3	VSS	PWR	Общий
C4	DATA[52]	IO (pu_1)	Шина данных внешнего интерфейса
C5	DATA[47]		
C6	DATA[45]		
C7	DATA[39]		
C8	DATA[35]		
C9	DATA[31]		
C10	DATA[27]		
C11	DATA[21]		
C12	DATA[17]		
C13	VSS	PWR	Общий
C14	DATA[13]	IO (pu_1)	Шина данных внешнего интерфейса
C15	DATA[7]		
C16	DATA[3]		
C17	ACK	IO (pu_2, pu_4)	Подтверждение готовности обмена по внешней шине
C18	nRD	IO (pu_2_id0)	Чтение внешней памяти (кроме SDRAM)
C19	ADDR[26]	IO (pu_1)	Шина адреса внешнего интерфейса
C20	ADDR[24]		
C21	ADDR[20]		
C22	VSS	PWR	Общий
C23	ADDR[18]	IO (pu_1)	Шина адреса внешнего интерфейса
C24	VDD_IO	PWR	Питание ввода/вывода
D1	DATA[55]	IO (pu_1)	Шина данных внешнего интерфейса
D2	DATA[56]		
D3	DATA[54]		
D4	VSS	PWR	Общий
D5	DATA[48]	IO (pu_1)	Шина данных внешнего интерфейса
D6	DATA[46]		
D7	DATA[40]		
D8	DATA[36]		
D9	DATA[32]		
D10	DATA[28]		
D11	DATA[22]		
D12	DATA[18]		
D13	VSS	PWR	Общий
D14	DATA[14]	IO (pu_1)	Шина данных внешнего интерфейса
D15	DATA[8]		
D16	DATA[4]		
D17	DATA[0]		
D18	nBRST	IO (pu_2_id0)	Признак пакетного режима передачи
D19	ADDR[27]	IO (pu_1)	Шина адреса внешнего интерфейса
D20	ADDR[25]		

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
D21	VSS	PWR	Общий
D22	ADDR[19]	IO (pu_1)	Шина адреса внешнего интерфейса
D23	ADDR[17]		
D24	ADDR[16]		
E1	DATA[61]	IO (pu_1)	Шина данных внешнего интерфейса
E2	DATA[62]		
E3	DATA[57]	IO (pu_1)	Шина данных внешнего интерфейса
E4	DATA[58]		
E5	VSS	PWR	Общий
E6	VDD_IO	PWR	Питание ввода/вывода
E7	VSS	PWR	Общий
E8	VDD_IO	PWR	Питание ввода/вывода
E9	VSS	PWR	Общий
E10	VDD_IO	PWR	Питание ввода/вывода
E11			
E12			
E13			
E14			
E15			
E16	VSS	PWR	Общий
E17	VDD_IO	PWR	Питание ввода/вывода
E18	VSS	PWR	Общий
E19	VDD_IO	PWR	Питание ввода/вывода
E20	VSS	PWR	Общий
E21	ADDR[15]	IO (pu_1)	Шина адреса внешнего интерфейса
E22	ADDR[14]		
E23	ADDR[11]		
E24	ADDR[10]		
F1	DATA[63]	IO (pu_1)	Шина данных внешнего интерфейса
F2	nMS[1]	IO (pu_2_id0)	Выбор банка внешней памяти
F3	DATA[59]	IO (pu_1)	Шина данных внешнего интерфейса
F4	DATA[60]		
F5	VDD_IO	PWR	Питание ввода/вывода
F6	VDD	PWR	Питание ядра
F7			
F8			
F9			
F10			
F11			
F12			
F13			
F14			
F15			
F16			
F17			
F18			
F19			
F20	VDD_IO	PWR	Питание ввода/вывода
F21	ADDR[13]	IO (pu_1)	Шина адреса внешнего интерфейса

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
F22	ADDR[12]	IO (pu_1)	Шина адреса внешнего интерфейса
F23	ADDR[9]		
F24	ADDR[8]		
G1	nMSSD[1]	IO (pu_2_id0)	SDRAM. Выбор банка
G2	VSS	PWR	Общий
G3	nMS[0]	IO (pu_2_id0)	Выбор банка внешней памяти
G4	nBMS	IO (pu_2_id0, pd_2_id0)	Выбор загрузочного EPROM
G5	VSS	PWR	Общий
G6	VDD	PWR	Питание ядра
G7			
G8			
G9			
G10			
G11			
G12			
G13			
G14			
G15			
G16			
G17			
G18			
G19			
G20	VDD_IO	PWR	Питание ввода/вывода
G21	ADDR[7]	IO (pu_1)	Шина адреса внешнего интерфейса
G22	ADDR[6]		
G23	ADDR[5]		
G24	ADDR[4]		
H1	VSS	PWR	Общий
H2	nMSH	IO (pu_2_id0)	Выбор адресного пространства хоста
H3	nMSSD[3]	IO (pu_2_id0)	SDRAM. Выбор банка
H4	RANGE[0]	I	Диапазон входной частоты ГУН PLL
H5	VDD_IO	PWR	Питание ввода/вывода
H6	VDD	PWR	Питание ядра
H7			
H8	VSS	PWR	Общий
H9			
H10			
H11			
H12			
H13			
H14			
H15			
H16			
H17			
H18	VDD	PWR	Питание ядра
H19			
H20	VDD_IO	PWR	Питание ввода/вывода
H21	ADDR[3]	IO (pu_1)	Шина адреса внешнего интерфейса

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
H22	ADDR[2]	IO (pu_1)	Шина адреса внешнего интерфейса
H23	ADDR[1]		
H24	ADDR[0]		
J1	nRAS	IO (pu_2_id0)	SDRAM. Строб выбора ряда
J2	nCAS	IO (pu_2_id0)	SDRAM. Строб выбора колонки
J3	VSS	PWR	Общий
J4	RANGE[1]	I	Диапазон входной частоты ГУН PLL
J5	VSS	PWR	Общий
J6	VDD	PWR	Питание ядра
J7			
J8	VSS	PWR	Общий
J9			
J10			
J11	VSS	PWR	Общий
J12			
J13			
J14			
J15			
J16			
J17			
J18	VDD	PWR	Питание ядра
J19			
J20	VSS	PWR	Общий
J21	L0ACKO	O	Подтверждение готовности. Выход приемника
J22	nL0BCMPI	I (pd_1)	Завершение блока. Вход приемника
J23	L0DATIN[0]	I	Шина данных приемника. Инверсная
J24	L0DATIP[0]	I	Шина данных приемника. Прямая
K1	SDA10	IO (pu_2_id0)	SDRAM. 10-й бит адреса
K2	SDCKE	IO (pu_2, pd_2)	SDRAM. Синхронизация. Имеет встроенный резистор к 1
K3	LDQM	IO (pu_2_id0)	SDRAM. Маска младшего слова
K4	HDQM	IO (pu_2_id0)	SDRAM. Маска старшего слова
K5	VDD_IO	PWR	Питание ввода/вывода
K6	VDD	PWR	Питание ядра
K7			
K8	VSS	PWR	Общий
K9			
K10			
K11			
K12			
K13			
K14			
K15			
K16			
K17			
K18	VDD	PWR	Питание ядра
K19			
K20	VDD_IO	PWR	Питание ввода/вывода
K21	L0DATIN[1]	I	Шина данных приемника. Инверсная
K22	L0DATIP[1]	I	Шина данных приемника. Прямая
K23	L0CLKIN	I	Синхронизация приемника. Инверсная

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
K24	L0CLKIP	I	Синхронизация приемника. Прямая
L1	nSDWE	IO (pu_2_id0)	SDRAM. Строб записи
L2	nBR[0]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
L3	nBR[1]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
L4	nBR[2]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
L5	VDD_IO	PWR	Питание ввода/вывода
L6	VDD	PWR	Питание ядра
L7			
L8	VSS	PWR	Общий
L9			
L10			
L11			
L12	VSS	PWR	Общий
L13			
L14			
L15			
L16			
L17			
L18	VDD	PWR	Питание ядра
L19			
L20	VDD_IO	PWR	Питание ввода/вывода
L21	L0DATIN[3]	I	Шина данных приемника. Инверсная
L22	L0DATIP[3]	I	Шина данных приемника. Прямая
L23	L0DATIN[2]	I	Шина данных приемника. Инверсная
L24	L0DATIP[2]	I	Шина данных приемника. Прямая
M1	nBR[3]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
M2	DIVQ[0]	I	Постделитель ГУН PLL
M3	nBR[5]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
M4	nBR[6]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
M5	VDD_IO	PWR	Питание ввода/вывода
M6	VDD	PWR	Питание ядра
M7			
M8	VSS	PWR	Общий
M9			
M10			
M11			
M12			
M13			
M14			
M15			
M16			
M17			
M18	VDD	PWR	Питание ядра
M19			
M20	VDD_IO	PWR	Питание ввода/вывода

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
M21	VSS	PWR	Общий
M22			
M23	L0DATON[3]	O	Шина данных передатчика. Инверсная
M24	L0DATOP[3]	O	Шина данных передатчика. Прямая
N1	ID[0]	I (pd_2)	Номер процессора
N2	VSS	PWR	Общий
N3	DIVQ[2]	I	Постделитель ГУН PLL
N4	VDD_A	PWR	Питание аналоговых блоков
N5	VDD_IO	PWR	Питание ввода/вывода
N6	VDD	PWR	Питание ядра
N7			
N8	VSS	PWR	Общий
N9			
N10			
N11			
N12			
N13			
N14	VSS	PWR	Общий
N15			
N16			
N17			
N18	VDD	PWR	Питание ядра
N19			
N20	VDD_IO	PWR	Питание ввода/вывода
N21	L0DATON[2]	O	Шина данных передатчика. Инверсная
N22	L0DATOP[2]	O	Шина данных передатчика. Прямая
N23	L0CLKON	O	Синхронизация передатчика. Инверсная
N24	L0CLKOP	O	Синхронизация передатчика. Прямая
P1	SCLK	I	Синхронизация внешней шины
P2	DIVQ[1]	I	Постделитель ГУН PLL
P3	VSS	PWR	Общий
P4	nBM	IO	Арбитр. Признак мастера шины
P5	VDD_IO	PWR	Питание ввода/вывода
P6	VDD	PWR	Питание ядра
P7			
P8	VSS	PWR	Общий
P9			
P10			
P11			
P12			
P13			
P14			
P15			
P16			
P17			
P18	VDD	PWR	Питание ядра
P19			
P20	VDD_IO	PWR	Питание ввода/вывода
P21	L0DATON[1]	O	Шина данных передатчика. Инверсная

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
P22	L0DATOP[1]	O	Шина данных передатчика. Прямая
P23	L0DATON[0]	O	Шина данных передатчика. Инверсная
P24	L0DATOP[0]	O	Шина данных передатчика. Прямая
R1	VSS	PWR	Общий
R2	RANGE[2]	I	Диапазон входной частоты ГУН PLL
R3	DIVF[0]	I	Предделитель ГУН PLL
R4	nBR[7]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
R5	VDD_IO	PWR	Питание ввода/вывода
R6	VDD	PWR	Питание ядра
R7			
R8	VSS	PWR	Общий
R9			
R10			
R11			
R12			
R13			
R14			
R15			
R16	VSS	PWR	Общий
R17			
R18	VDD	PWR	Питание ядра
R19			
R20	VDD_IO	PWR	Питание ввода/вывода
R21	E_PLLBP	I (pd_1)	Обход дополнительного PLL
R22	VSS	PWR	Общий
R23	nL0BCMPO	IO (pu_1)	Завершение блока. Выход передатчика
R24	L0ACKI	I (pd_1)	Подтверждение готовности. Вход передатчика
T1	nRST_IN	I	Сброс. Активный низкий
T2	DIVF[1]	I	Предделитель ГУН PLL
T3	nBR[4]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
T4	DIVF[4]	I	Предделитель ГУН PLL
T5	VSS	PWR	Общий
T6	VDD	PWR	Питание ядра
T7			
T8	VSS	PWR	Общий
T9			
T10			
T11			
T12			
T13			
T14			
T15			
T16			
T17			
T18	VDD	PWR	Питание ядра
T19			
T20	VSS	PWR	Общий
T21	L1DATIN[0]	I	Шина данных приемника. Инверсная

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
T22	L1DATIP[0]	I	Шина данных приемника. Прямая
T23	L1ACKO	O	Подтверждение готовности. Выход приемника
T24	nL1BCMPI	I (pd_1)	Завершение блока. Вход приемника
U1	nMSSD[0]	IO (pu_2_id0)	SDRAM. Выбор банка
U2	nRST_OUT	O	Сброс. Активный низкий
U3	ID[2]	I (pd_2)	Номер процессора
U4	DIVF[5]	I	Предделитель ГУН PLL
U5	VDD_IO	PWR	Питание ввода/вывода
U6	VDD	PWR	Питание ядра
U7			
U8	VSS	PWR	Общий
U9			
U10	VDD	PWR	Питание ядра
U11			
U12	VSS	PWR	Общий
U13			
U14			
U15			
U16			
U17			
U18	VDD	PWR	Питание ядра
U19			
U20	VDD_IO	PWR	Питание ввода/вывода
U21	L1CLKIN	I	Синхронизация приемника. Инверсная
U22	L1CLKIP	I	Синхронизация приемника. Прямая
U23	L1DATIN[1]	I	Шина данных приемника. Инверсная
U24	L1DATIP[1]	I	Шина данных приемника. Прямая
V1	nMSSD[2]	IO (pu_2_id0)	SDRAM. Выбор банка
V2	DIVF[2]	I	Предделитель ГУН PLL
V3	DIVF[3]		
V4	DIVF[6]		
V5	VSS	PWR	Общий
V6	VDD	PWR	Питание ядра
V7			
V8			
V9	VDD	PWR	Питание ядра
V10			
V11			
V12			
V13			
V14			
V15			
V16			
V17			
V18			
V19			
V20	VDD_IO	PWR	Питание ввода/вывода
V21	L1DATIN[3]	I	Шина данных приемника. Инверсная
V22	L1DATIP[3]	I	Шина данных приемника. Прямая

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
V23	L1DATIN[2]	I	Шина данных приемника. Инверсная
V24	L1DATIP[2]	I	Шина данных приемника. Прямая
W1	DXP	I	Прямой вывод термодиода
W2	DXN	I	Инверсный вывод термодиода
W3	TDI	I (pu_1)	JTAG. Вход данных
W4	TDO	O	JTAG. Выход данных
W5	VDD_IO	PWR	Питание ввода/вывода
W6	VDD	PWR	Питание ядра
W7			
W8			
W9			
W10			
W11			
W12			
W13			
W14			
W15			
W16			
W17			
W18			
W19			
W20	VDD_IO	PWR	Питание ввода/вывода
W21	L1CLKON	O	Синхронизация передатчика. Инверсная
W22	L1CLKOP	O	Синхронизация передатчика. Прямая
W23	L1DATON[3]	O	Шина данных передатчика. Инверсная
W24	L1DATOP[3]	IO	Шина данных передатчика. Прямая
Y1	nEMU	O	Запрос к внешнему эмулятору
Y2	TCK	I	JTAG. Синхронизация
Y3	TMR0E	IO	Признак достижения таймером 0 нулевого значения счетчика/Выбор способа загрузки во время сброса
Y4	FLAG[3]	IO (pu_2)	Порт общего назначения
Y5	VSS	PWR	Общий
Y6	VDD_IO	PWR	Питание ввода/вывода
Y7	VSS	PWR	Общий
Y8	VDD_IO	PWR	Питание ввода/вывода
Y9	VSS	PWR	Общий
Y10	VDD_IO	PWR	Питание ввода/вывода
Y11			
Y12			
Y13			
Y14			
Y15			
Y16	VSS	PWR	Общий
Y17	VDD_IO	PWR	Питание ввода/вывода
Y18	VSS	PWR	Общий
Y19	VDD_IO	PWR	Питание ввода/вывода
Y20	VSS	PWR	Общий
Y21	L1DATON[1]	O	Шина данных передатчика. Инверсная
Y22	L1DATOP[1]	IO	Шина данных передатчика. Прямая
Y23	L1DATON[2]	O	Шина данных передатчика. Инверсная

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
Y24	L1DATOP[2]	IO	Шина данных передатчика. Прямая
AA1	FLAG[2]	IO (pu_2)	Порт общего назначения
AA2	FLAG[1]	IO (pu_2)	Порт общего назначения
AA3	nIRQ[3]	IO (pu_2)	Запросы прерываний
AA4	VSS	PWR	Общий
AA5	nIRQ[0]	IO (pu_2)	Запросы прерываний
AA6	nIOEN	IO (pu_2_id0)	Строб выбора внешнего устройства
AA7	nDMAR[0]	I	Запросы к каналам 3-0 ДМА на обслуживание внешних устройств
AA8	nHBR	I	Арбитр. Запрос шины от хоста
AA9	nL3BCMPO	IO	Завершение блока. Выход передатчика
AA10	L3DATON[1]	O	Шина данных передатчика. Инверсная
AA11	L3DATON[3]	O	Шина данных передатчика. Инверсная
AA12	VSS	PWR	Общий
AA13	L3DATIN[2]	I	Шина данных приемника. Инверсная
AA14	L3DATIN[1]	I	Шина данных приемника. Инверсная
AA15	JG_MX	I (pd_1)	Выбор модуля JTAG
AA16	L2DATON[0]	O	Шина данных передатчика. Инверсная
AA17	L2CLKON	O	Синхронизация передатчика. Инверсная
AA18	L2DATON[3]	O	Шина данных передатчика. Инверсная
AA19	L2CLKIN	I	Синхронизация приемника. Инверсная
AA20	L2DATIN[1]	I	Шина данных приемника. Инверсная
AA21	VSS	PWR	Общий
AA22	nL1BCMPO	IO	Завершение блока. Выход передатчика
AA23	L1DATON[0]	O	Шина данных передатчика. Инверсная
AA24	L1DATOP[0]	IO	Шина данных передатчика. Прямая
AB1	VSS	PWR	Общий
AB2	FLAG[0]	IO (pu_2)	Порт общего назначения
AB3	VSS	PWR	Общий
AB4	DIVF[7]	I	Предделитель ГУН PLL
AB5	nIRQ[2]	IO (pu_2)	Запросы прерываний
AB6	nIRQ[1]	IO (pu_2)	Запросы прерываний
AB7	nDMAR[1]	I	Запросы к каналам 3-0 ДМА на обслуживание внешних устройств
AB8	nHBG	IO (pu_2_id0)	Арбитр. Предоставление шины хосту
AB9	L3ACKI	I (pd_1)	Подтверждение готовности. Вход передатчика
AB10	L3DATOP[1]	O	Шина данных передатчика. Прямая
AB11	L3DATOP[3]	O	Шина данных передатчика. Прямая
AB12	VSS	PWR	Общий
AB13	L3DATIP[2]	I	Шина данных приемника. Прямая
AB14	L3DATIP[1]	I	Шина данных приемника. Прямая
AB15	VSS	PWR	Общий
AB16	L2DATOP[0]	O	Шина данных передатчика. Прямая
AB17	L2CLKOP	O	Синхронизация передатчика. Прямая
AB18	L2DATOP[3]	O	Шина данных передатчика. Прямая
AB19	L2CLKIP	I	Синхронизация приемника. Прямая
AB20	L2DATIP[1]	I	Шина данных приемника. Прямая
AB21	L2ACKO	O	Подтверждение готовности. Выход приемника
AB22	VSS	PWR	Общий

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
AB23	L1ACKI	I (pd_1)	Подтверждение готовности. Вход передатчика
AB24	VDD_IO	PWR	Питание ввода/вывода
AC1	VSS	PWR	Общий
AC2			
AC3	ID[1]	I (pd_2)	Номер процессора
AC4	TMS	I (pu_1)	JTAG. Выбор интерфейса
AC5	nIOWR	IO (pu_2_id0)	Строб записи внешнего устройства
AC6	nDMAR[2]	I	Запросы к каналам 3-0 ДМА на обслуживание внешних устройств
AC7	nCPA	IO (pu_3_id0)	Арбитр. Запрос высокоприоритетной транзакции ядра
AC8	nBOFF	I	Арбитр. Запрос прекращения цикла обмена от хоста
AC9	L3DATON[0]	O	Шина данных передатчика. Инверсная
AC10	L3CLKON	O	Синхронизация передатчика. Инверсная
AC11	L3DATON[2]	O	Шина данных передатчика. Инверсная
AC12	L3DATIN[3]	I	Шина данных приемника. Инверсная
AC13	L3CLKIN	I	Синхронизация приемника. Инверсная
AC14	L3DATIN[0]	I	Шина данных приемника. Инверсная
AC15	L3ACKO	O	Подтверждение готовности. Выход приемника
AC16	nL2BCMPO	IO	Завершение блока. Выход передатчика
AC17	L2DATON[1]	O	Шина данных передатчика. Инверсная
AC18	L2DATON[2]	O	Шина данных передатчика. Инверсная
AC19	L2DATIN[3]	I	Шина данных приемника. Инверсная
AC20	L2DATIN[2]	I	Шина данных приемника. Инверсная
AC21	L2DATIN[0]	I	Шина данных приемника. Инверсная
AC22	nL2BCMPI	I (pd_1)	Завершение блока. Вход приемника
AC23	VSS	PWR	Общий
AC24	VDD_IO	PWR	Питание ввода/вывода
AD1	VSS	PWR	Общий
AD2	VDD_IO	PWR	Питание ввода/вывода
AD3			
AD4	nTRST	I (pu_1)	JTAG. Сброс
AD5	nIORD	IO (pu_2_id0)	Строб чтения внешнего устройства
AD6	nDMAR[3]	I	Запросы к каналам 3-0 ДМА на обслуживание внешних устройств
AD7	nDPA	IO (pu_3_id0)	Арбитр. Запрос высокоприоритетной транзакции ДМА
AD8	nBUSLOCK	IO (pu_2_id0, pd_2_id0)	Арбитр. Признак блокировки шины
AD9	L3DATOP[0]	O	Шина данных передатчика. Прямая
AD10	L3CLKOP	O	Синхронизация передатчика. Прямая
AD11	L3DATOP[2]	O	Шина данных передатчика. Прямая
AD12	L3DATIP[3]	I	Шина данных приемника. Прямая
AD13	L3CLKIP	I	Синхронизация приемника. Прямая
AD14	L3DATIP[0]	I	Шина данных приемника. Прямая
AD15	nL3BCMPI	I (pd_1)	Завершение блока. Вход приемника
AD16	L2ACKI	I (pd_1)	Подтверждение готовности. Вход передатчика

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
AD17	L2DATOP[1]	O	Шина данных передатчика. Прямая
AD18	L2DATOP[2]		
AD19	L2DATIP[3]	I	Шина данных приемника. Прямая
AD20	L2DATIP[2]		
AD21	L2DATIP[0]		
AD22	VDD_IO	PWR	Питание ввода/вывода
AD23			
AD24	VSS	PWR	Общий

Примечания:

1 Обозначение типов выводов:
- PWR – вывод питания и «общий»;
- I – вход;
- O – выход;
- IO – вход/выход.

2 Номиналы подтягивающих резисторов:
- pd_1 – подтягивающий к земле резистор 40 кОм;
- pu_1 – подтягивающий к питанию резистор 40 кОм;
- pd_2 – подтягивающий к земле резистор 5 кОм;
- pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;
- pu_2 – подтягивающий к питанию резистор 5 кОм;
- pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;
- pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;
- pu_4 – подтягивающий к питанию резистор 50 Ом.

Таблица 2 – Назначение выводов по блокам микросхемы в корпусе МК8303.576-1

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
P1	SCLK	I	Синхронизация внешней шины
T1	nRST_IN	I	Сброс. Активный низкий
U2	nRST_OUT	O	
N1	ID[0]	I (pd_2)	Номер процессора
U3	ID[2]		
AC3	ID[1]		
R3	DIVF[0]	I	Предделитель ГУН PLL
T2	DIVF[1]		
V2	DIVF[2]		
V3	DIVF[3]		
T4	DIVF[4]		
U4	DIVF[5]		
V4	DIVF[6]		
AB4	DIVF[7]		
M2	DIVQ[0]	I	Постделитель ГУН PLL
P2	DIVQ[1]		
N3	DIVQ[2]		
H4	RANGE[0]	I	Диапазон входной частоты ГУН PLL
J4	RANGE[1]		
R2	RANGE[2]		

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
U1	nMSSD[0]	IO (pu_2_id0)	SDRAM. Выбор банка
G1	nMSSD[1]		
V1	nMSSD[2]		
H3	nMSSD[3]		
J1	nRAS	IO (pu_2_id0)	SDRAM. Строб выбора ряда
J2	nCAS	IO (pu_2_id0)	SDRAM. Строб выбора колонки
K3	LDQM	IO (pu_2_id0)	SDRAM. Маска младшего слова
K4	HDQM	IO (pu_2_id0)	SDRAM. Маска старшего слова
L1	nSDWE	IO (pu_2_id0)	SDRAM. Строб записи
K2	SDCKE	IO (pu_2, pd_2)	SDRAM. Синхронизация. Имеет встроенный резистор к 1
K1	SDA10	IO (pu_2_id0)	SDRAM. 10-й бит адреса
AD5	nIORD	IO (pu_2_id0)	Строб чтения внешнего устройства
AC5	nIOWR	IO (pu_2_id0)	Строб записи внешнего устройства
AA6	nIOEN	IO (pu_2_id0)	Строб выбора внешнего устройства
AA5	nIRQ[0]	IO (pu_2)	Запросы прерываний
AB6	nIRQ[1]		
AB5	nIRQ[2]		
AA3	nIRQ[3]		
AB2	FLAG[0]	IO (pu_2)	Порт общего назначения
AA2	FLAG[1]		
AA1	FLAG[2]		
Y4	FLAG[3]		
H24	ADDR[0]	IO (pu_1)	Шина адреса внешнего интерфейса
H23	ADDR[1]		
H22	ADDR[2]		
H21	ADDR[3]		
G24	ADDR[4]		
G23	ADDR[5]		
G22	ADDR[6]		
G21	ADDR[7]		
F24	ADDR[8]	IO (pu_1)	Шина адреса внешнего интерфейса
F23	ADDR[9]		
E24	ADDR[10]		
E23	ADDR[11]		
F22	ADDR[12]		
F21	ADDR[13]		
E22	ADDR[14]		
E21	ADDR[15]		
D24	ADDR[16]		
D23	ADDR[17]		
C23	ADDR[18]		
D22	ADDR[19]		
C21	ADDR[20]		
B22	ADDR[21]		
A21	ADDR[22]		
B21	ADDR[23]		
C20	ADDR[24]		
D20	ADDR[25]		
C19	ADDR[26]		
D19	ADDR[27]		
A20	ADDR[28]		
B20	ADDR[29]		

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
A19	ADDR[30]		
B19	ADDR[31]		
D17	DATA[0]	IO (pu_1)	Шина данных внешнего интерфейса
A17	DATA[1]		
B17	DATA[2]		
C16	DATA[3]		
D16	DATA[4]		
A16	DATA[5]		
B16	DATA[6]		
C15	DATA[7]		
D15	DATA[8]		
A15	DATA[9]		
B15	DATA[10]		
A14	DATA[11]		
B14	DATA[12]		
C14	DATA[13]		
D14	DATA[14]		
A13	DATA[15]		
B13	DATA[16]		
C12	DATA[17]		
D12	DATA[18]		
A12	DATA[19]		
B12	DATA[20]		
C11	DATA[21]		
D11	DATA[22]		
A11	DATA[23]		
B11	DATA[24]		
A10	DATA[25]		
B10	DATA[26]		
C10	DATA[27]		
D10	DATA[28]		
A9	DATA[29]	IO (pu_1)	Шина данных внешнего интерфейса
B9	DATA[30]		
C9	DATA[31]		
D9	DATA[32]		
A8	DATA[33]		
B8	DATA[34]		
C8	DATA[35]		
D8	DATA[36]		
A7	DATA[37]		
B7	DATA[38]		
C7	DATA[39]		
D7	DATA[40]		
A6	DATA[41]		
B6	DATA[42]		
A5	DATA[43]		
B5	DATA[44]		
C6	DATA[45]		
D6	DATA[46]		
C5	DATA[47]		
D5	DATA[48]		
A4	DATA[49]		
B4	DATA[50]		

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
B3	DATA[51]		
C4	DATA[52]		
C2	DATA[53]		
D3	DATA[54]		
D1	DATA[55]		
D2	DATA[56]		
E3	DATA[57]		
E4	DATA[58]		
F3	DATA[59]		
F4	DATA[60]		
E1	DATA[61]		
E2	DATA[62]		
F1	DATA[63]		
C18	nRD	IO (pu_2_id0)	Чтение внешней памяти (кроме SDRAM)
A18	nWRL	IO (pu_2_id0)	Запись младшего слова внешней памяти (кроме SDRAM)
B18	nWRH	IO (pu_2_id0)	Запись старшего слова внешней памяти (кроме SDRAM)
C17	ACK	IO (pu_2, pu_4)	Подтверждение готовности обмена по внешней шине
G4	nBMS	IO (pu_2_id0, pd_2_id0)	Выбор загрузочного EPROM
G3	nMS[0]	IO (pu_2_id0)	Выбор банка внешней памяти
F2	nMS[1]		
H2	nMSH	IO (pu_2_id0)	Выбор адресного пространства хоста
D18	nBRST	IO (pu_2_id0)	Признак пакетного режима передачи
L2	nBR[0]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
L3	nBR[1]		
L4	nBR[2]		
M1	nBR[3]		
T3	nBR[4]	IO	Арбитр. Запрос доступа к шине от процессоров кластера
M3	nBR[5]		
M4	nBR[6]		
R4	nBR[7]		
P4	nBM	IO	Арбитр. Признак мастера шины
AD8	nBUSLOCK	IO (pu_2_id0, pd_2_id0)	Арбитр. Признак блокировки шины
AA8	nHBR	I	Арбитр. Запрос шины от хоста
AB8	nHBG	IO (pu_2_id0)	Арбитр. Предоставление шины хосту
AC8	nBOFF	I	Арбитр. Запрос прекращения цикла обмена от хоста
AC7	nCPA	IO (pu_3_id0)	Арбитр. Запрос высокоприоритетной транзакции ядра
AD7	nDPA	IO (pu_3_id0)	Арбитр. Запрос высокоприоритетной транзакции ДМА
AA7	nDMAR[0]	I	Запросы к каналам 3-0 ДМА на обслуживание внешних устройств
AB7	nDMAR[1]		
AC6	nDMAR[2]		
AD6	nDMAR[3]		
Y3	TMR0E	IO	Признак достижения таймером 0 нулевого значения счетчика/Выбор способа загрузки во время сброса

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
Порт связи 0			
P24	L0DATOP[0]	O	Шина данных передатчика. Прямая
P22	L0DATOP[1]		
N22	L0DATOP[2]		
M24	L0DATOP[3]	O	Шина данных передатчика. Прямая
P23	L0DATON[0]	O	Шина данных передатчика. Инверсная
P21	L0DATON[1]		
N21	L0DATON[2]		
M23	L0DATON[3]		
J24	L0DATIP[0]	I	Шина данных приемника. Прямая
K22	L0DATIP[1]		
L24	L0DATIP[2]		
L22	L0DATIP[3]		
J23	L0DATIN[0]	I	Шина данных приемника. Инверсная
K21	L0DATIN[1]		
L23	L0DATIN[2]		
L21	L0DATIN[3]		
N24	L0CLKOP	O	Синхронизация передатчика. Прямая
N23	L0CLKON	O	Синхронизация передатчика. Инверсная
K24	L0CLKIP	I	Синхронизация приемника. Прямая
K23	L0CLKIN	I	Синхронизация приемника. Инверсная
J21	L0ACKO	O	Подтверждение готовности. Выход приемника
R24	L0ACKI	I (pd_1)	Подтверждение готовности. Вход передатчика
R23	nL0BCMPO	IO (pu_1)	Завершение блока. Выход передатчика
J22	nL0BCMPI	I (pd_1)	Завершение блока. Вход приемника
Порт связи 1			
AA24	L1DATOP[0]	IO	Шина данных передатчика. Прямая
Y22	L1DATOP[1]		
Y24	L1DATOP[2]		
W24	L1DATOP[3]		
AA23	L1DATON[0]	O	Шина данных передатчика. Инверсная
Y21	L1DATON[1]		
Y23	L1DATON[2]		
W23	L1DATON[3]		
T22	L1DATIP[0]	I	Шина данных приемника. Прямая
U24	L1DATIP[1]		
V24	L1DATIP[2]		
V22	L1DATIP[3]		
T21	L1DATIN[0]	I	Шина данных приемника. Инверсная
U23	L1DATIN[1]		
V23	L1DATIN[2]		
V21	L1DATIN[3]		
W22	L1CLKOP	O	Синхронизация передатчика. Прямая
W21	L1CLKON	O	Синхронизация передатчика. Инверсная
U22	L1CLKIP	I	Синхронизация приемника. Прямая
U21	L1CLKIN	I	Синхронизация приемника. Инверсная
T23	L1ACKO	O	Подтверждение готовности. Выход приемника

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
AB23	L1ACKI	I (pd_1)	Подтверждение готовности. Вход передатчика
AA22	nL1BCMPO	IO	Завершение блока. Выход передатчика
T24	nL1BCMPI	I (pd_1)	Завершение блока. Вход приемника
Порт связи 2			
AB16	L2DATOP[0]	O	Шина данных передатчика. Прямая
AD17	L2DATOP[1]		
AD18	L2DATOP[2]		
AB18	L2DATOP[3]		
AA16	L2DATON[0]	O	Шина данных передатчика. Инверсная
AC17	L2DATON[1]		
AC18	L2DATON[2]		
AA18	L2DATON[3]		
AD21	L2DATIP[0]	I	Шина данных приемника. Прямая
AB20	L2DATIP[1]		
AD20	L2DATIP[2]		
AD19	L2DATIP[3]		
AC21	L2DATIN[0]	I	Шина данных приемника. Инверсная
AA20	L2DATIN[1]		
AC20	L2DATIN[2]		
AC19	L2DATIN[3]		
AB17	L2CLKOP	O	Синхронизация передатчика. Прямая
AA17	L2CLKON	O	Синхронизация передатчика. Инверсная
AB19	L2CLKIP	I	Синхронизация приемника. Прямая
AA19	L2CLKIN	I	Синхронизация приемника. Инверсная
AB21	L2ACKO	O	Подтверждение готовности. Выход приемника
AD16	L2ACKI	I (pd_1)	Подтверждение готовности. Вход передатчика
AC16	nL2BCMPO	IO	Завершение блока. Выход передатчика
AC22	nL2BCMPI	I (pd_1)	Завершение блока. Вход приемника
Порт связи 3			
AD9	L3DATOP[0]	O	Шина данных передатчика. Прямая
AB10	L3DATOP[1]		
AD11	L3DATOP[2]		
AB11	L3DATOP[3]		
AC9	L3DATON[0]	O	Шина данных передатчика. Инверсная
AA10	L3DATON[1]		
AC11	L3DATON[2]		
AA11	L3DATON[3]		
AD14	L3DATIP[0]	I	Шина данных приемника. Прямая
AB14	L3DATIP[1]		
AB13	L3DATIP[2]		
AD12	L3DATIP[3]		
AC14	L3DATIN[0]	I	Шина данных приемника. Инверсная
AA14	L3DATIN[1]		
AA13	L3DATIN[2]		
AC12	L3DATIN[3]		
AD10	L3CLKOP	O	Синхронизация передатчика. Прямая
AC10	L3CLKON	O	Синхронизация передатчика. Инверсная

Номер вывода	Наименование сигнала	Тип вывода	Функциональное назначение вывода
AD13	L3CLKIP	I	Синхронизация приемника.Прямая
AC13	L3CLKIN	I	Синхронизация приемника.Инверсная
AC15	L3ACKO	O	Подтверждение готовности. Выход приемника
AB9	L3ACKI	I (pd_1)	Подтверждение готовности. Вход передатчика
AA9	nL3BCMPO	IO	Завершение блока. Выход передатчика
AD15	nL3BCMPI	I (pd_1)	Завершение блока. Вход приемника
W1	DXP	I	Прямой вывод термодиода
W2	DXN	I	Инверсный вывод термодиода
AD4	nTRST	I (pu_1)	JTAG. Сброс
Y2	TCK	I	JTAG. Синхронизация
AC4	TMS	I (pu_1)	JTAG. Выбор интерфейса
W3	TDI	I (pu_1)	JTAG. Вход данных
W4	TDO	O	JTAG. Выход данных
Y1	nEMU	O	Запрос к внешнему эмулятору
AA15	JG_MX	I (pd_1)	Выбор модуля JTAG
R21	E_PLLBP	I (pd_1)	Обход дополнительного PLL
Питание и «общий»			
Группа 1	VDD	PWR	Питание ядра
Группа 2	VDD_IO		Питание ввода/вывода
N4	VDD_A		Питание аналоговых блоков
Группа 3	VSS		Общий

Примечания:

1 Обозначение типов выводов:

PWR – вывод питания и «общий»;
I – вход;
O – выход;
IO – вход/выход.

2 Номиналы подтягивающих резисторов:

pd_1 – подтягивающий к земле резистор 40 кОм;
pu_1 – подтягивающий к питанию резистор 40 кОм;
pd_2 – подтягивающий к земле резистор 5 кОм;
pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;
pu_2 – подтягивающий к питанию резистор 5 кОм;
pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;
pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;
pu_4 – подтягивающий к питанию резистор 50 Ом.

3 Группы выводов:

Группа 1 (VDD): F6 – F19; G6 – G19, H6, H7, H18, H19, J6, J7, J18, J19, K6, K7, K18, K19, L6, L7, L18, L19, M6, M7, M18, M19, N6, N7, N18, N19, P6, P7, P18, P19, R6, R7, R18, R19, T6, T7, T18, T19, U6, U7, U10, U11, U18, U19, V6 – V19, W6 – W19;

Группа 2 (VDD_IO): B24, C24, E6, E8, E10 – E15, E17, E19, F5, F20, G20, H5, H20, K5, K20, L5, L20, M5, M20, N5, N20, P5, P20, R5, R20, U5, U20, V20, W5, W20, Y6, Y8, Y10-Y15, Y17, Y19, AB24, AC24, AD2, AD3, AD22, AD23;

Группа 3 (VSS): A1 – A3, A22 – A24, B1, B2, B23, C1, C3, C13, C22, D4, D13, D21, E5, E7, E9, E16, E18, E20, G2, G5, H1, H8-H17, J3, J5, J8 – J17, J20, K8 – K17, L8 – L17, M8 – M17, M21, M22, N2, N8 – N17, P3, P8 – P17, R1, R8 – R17, R22, T5, T8 – T17, T20, U8, U9, U12 – U17, V5, Y5, Y7, Y9, Y16, Y18, Y20, AA4, AA12, AA21, AB1, AB3, AB12, AB15, AB22, AC1, AC2, AC23, AD1, AD24

Таблица 3 – Функциональное назначение контактных шариков (КШ) кристалла, несущих логическую информацию (бескорпусное исполнение)

№ КШ X_Y	Функциональное назначение КШ	№ КШ X_Y	Функциональное назначение КШ	№ КШ X_Y	Функциональное назначение КШ	№ КШ X_Y	Функциональное назначение КШ
0_3	DATA[55]	1_8	MSSD[1]	2_13	nBR[1]	6_0	DATA[40]
0_4	DATA[57]	1_9	nMSH	2_14	RANGEFIL[2]	6_1	DATA[41]
0_5	DATA[60]	1_10	RAS	2_16	nBR[3]	6_28	nDMAR[3]
0_6	DATA[63]	1_11	SDCKE_up/down	2_17	nBM	6_29	nDMAR[0]
0_7	nMS[0]	1_12	SDA10	2_19	DIVR[2]	6_30	nDMAR[1]
0_8	RANGEFIL[0]	1_13	nBR[0]	2_21	DIVF[2]	7_0	DATA[37]
0_9	RANGEFIL[1]	1_14	DIVQ[1]	2_23	DIVF[6]	7_1	DATA[38]
0_10	HDQM	1_16	DIVQ[2]	2_25	nEMU	7_2	DATA[39]
0_11	SDCKE_up/down	1_17	DIVR[0]	2_26	FLAG[3]	7_28	nCPA_up
0_12	nBR[2]	1_18	DIVF[0]	2_27	nIRQ[3]	7_29	nCPA
0_13	SDWE	1_19	nRST_IN	2_28	ID[1]	7_30	nDPA_up
0_14	DIVQ[0]	1_20	DIVF[4]	2_29	DIVF[7]	8_0	DATA[35]
0_16	ID[0]	1_21	nRST_OUT	3_0	DATA[48]	8_1	DATA[36]
0_17	SCLK	1_22	MSSD[2]	3_1	DATA[49]	8_28	nDPA
0_18	DIVR[1]	1_23	DIVR[4]	3_2	DATA[50]	8_29	nHBR
0_19	DIVF[3]	1_24	DXN	3_28	TMS	8_30	nHBG
0_20	DIVF[1]	1_25	TDO	3_29	nTRST	9_0	DATA[32]
0_21	MSSD[0]	1_26	TMR0E	3_30	nIRQ[0]	9_1	DATA[33]
0_22	DIVF[5]	1_27	FLAG[1]	4_0	DATA[43]	9_2	DATA[34]
0_23	DIVR[3]	1_28	FLAG[0]	4_1	DATA[44]	9_28	nBOFF
0_24	DXP	2_1	DATA[52]	4_2	DATA[47]	9_29	nBUSLOCK_up/down
0_25	TDI	2_2	DATA[51]	4_28	nIRQ[2]	9_30	nBUSLOCK_up/down
0_26	TCK	2_3	DATA[54]	4_29	IOWR	10_0	DATA[30]
0_27	FLAG[2]	2_5	DATA[62]	4_30	IORD	10_1	DATA[31]
1_2	DATA[53]	2_6	DATA[59]	5_0	DATA[42]	10_28	nL3BCMPO
1_3	DATA[56]	2_7	nBMS_up/down	5_1	DATA[45]	10_29	L3DATON[0]
1_4	DATA[58]	2_9	MSSD[3]	5_2	DATA[46]	10_30	L3ACKI
1_5	DATA[61]	2_10	CAS	5_28	IOEN	11_0	DATA[27]
1_6	nMS[1]	2_11	LDQM	5_29	nIRQ[1]	11_1	DATA[28]

№ КШ X_Y	Функциональное назначение КШ	№ КШ X_Y	Функциональное назначение КШ	№ КШ X_Y	Функциональное назначение КШ	№ КШ X_Y	Функциональное назначение КШ
1_7	nBMS_up/down	2_12	VDD_IO	5_30	nDMAR[2]	11_2	DATA[29]
12_0	DATA[25]	18_30	L3CLKINN	24_29	L2CLKOP	31_1	ADDR[14]
12_1	DATA[26]	19_0	DATA[10]	24_30	L2DATON[2]	31_2	ADDR[11]
12_29	L3CLKON	19_1	DATA[7]	25_0	ADDR[30]	31_28	nL2BCMPI
12_30	L3DATOP[1]	19_2	DATA[8]	25_1	ADDR[31]	31_29	L2ACKO
13_0	DATA[24]	19_29	L3DATIN[1]	25_2	ADDR[26]	31_30	L2DATIN[0]
13_1	DATA[21]	19_30	L3DATIP[0]	25_29	L2DATON[3]	32_1	ADDR[10]
13_2	DATA[22]	20_0	DATA[5]	25_30	L2DATOP[2]	32_2	ADDR[13]
13_29	L3CLKOP	20_1	DATA[6]	26_0	ADDR[27]	32_3	ADDR[9]
13_30	L3DATON[2]	20_27	JG_MX	26_1	ADDR[28]	32_4	ADDR[6]
14_0	DATA[23]	20_28	L3ACKO	26_29	L2DATOP[3]	32_5	ADDR[3]
14_1	DATA[19]	20_29	nL3BCMPI	26_30	L2DATIP[3]	32_17	nL0BCMPO
14_29	L3DATON[3]	20_30	L3DATIN[0]	27_0	ADDR[29]	32_18	L1ACKO
14_30	L3DATOP[2]	21_0	DATA[3]	27_1	ADDR[24]	32_28	nL1BCMPO
15_0	DATA[20]	21_1	DATA[4]	27_2	ADDR[25]	32_29	L1ACKI
15_1	DATA[17]	21_2	DATA[1]	27_29	L2DATIP[2]	33_2	ADDR[12]
15_2	DATA[18]	21_28	nL2BCMPO	27_30	L2DATIN[3]	33_3	ADDR[8]
15_29	L3DATOP[3]	21_29	L2DATON[0]	28_0	ADDR[22]	33_4	ADDR[5]
15_30	L3DATIP[3]	21_30	L2ACKI	28_1	ADDR[23]	33_5	ADDR[2]
16_0	DATA[15]	22_0	DATA[2]	28_29	L2DATIN[2]	33_6	ADDR[0]
16_1	DATA[16]	22_1	ACK	29_0	ADDR[20]	33_7	nL0BCMPI
16_29	L3DATIP[2]	22_2	ACK_up	29_1	ADDR[21]	33_8	L0DATIN[1]
16_30	L3DATIN[3]	22_29	L2DATOP[0]	29_2	ADDR[18]	33_9	L0DATIP[1]
17_0	DATA[11]	22_30	L2DATON[1]	29_29	L2DATIP[1]	33_10	L0DATIN[2]
17_1	DATA[12]	23_0	DATA[0]	29_30	L2CLKIN	33_11	L0DATIP[2]
17_2	DATA[13]	23_1	nWRL	30_0	ADDR[19]	33_12	L0DATOP[3]
17_29	L3DATIN[2]	23_2	nWRH	30_1	ADDR[17]	33_13	L0DATON[3]
17_30	L3CLKINP	23_29	L2CLKON	30_2	ADDR[16]	33_14	L0CLKOP
18_0	DATA[14]	23_30	L2DATOP[1]	30_29	L2DATIN[1]	33_15	L0CLKON
18_1	DATA[9]	24_0	nRD	30_30	L2DATIP[0]	33_16	L0DATOP[0]
18_29	L3DATIP[1]	24_1	nBRST	31_0	ADDR[15]	33_17	L0DATON[0]

№ КШ X_Y	Функциональное назначение КШ	№ КШ X_Y	Функциональное назначение КШ	№ КШ X_Y	Функциональное назначение КШ	№ КШ X_Y	Функциональное назначение КШ
33_18	nL1BCMPI	33_27	L1DATOP[0]	34_10	L0CLKIP	34_19	L1DATIP[0]
33_19	L1DATIN[1]	33_28	L1DATON[0]	34_11	L0DATIN[3]	34_20	L1CLKIN
33_20	L1DATIP[1]	34_3	ADDR[7]	34_12	L0DATIP[3]	34_21	L1CLKIP
33_21	L1DATIN[2]	34_4	ADDR[4]	34_13	L0DATOP[2]	34_22	L1DATIN[3]
33_22	L1DATIP[2]	34_5	ADDR[1]	34_14	L0DATON[2]	34_23	L1DATIP[3]
33_23	L1DATOP[3]	34_6	L0ACKO	34_15	L0DATOP[1]	34_24	L1DATOP[2]
33_24	L1DATON[3]	34_7	L0DATIN[0]	34_16	L0DATON[1]	34_25	L1DATON[2]
33_25	L1CLKOP	34_8	L0DATIP[0]	34_17	L0ACKI	34_26	L1DATOP[1]
32_26	L1CLKON	34_9	L0CLKIN	34_18	L1DATIN[0]	34_27	L1DATON[1]

4 Указания по применению и эксплуатации

Микросхемы пригодны для монтажа в аппаратуре методом групповой пайки при условии соблюдения требуемого температурного профиля паяльной пасты и равномерном прогреве места монтажа микросхемы по всей его площади. Скорость нагрева и охлаждения компонентов монтажа не должна превышать 4 °С/с. Параметры профиля пайки оловянно-свинцовой паяльной пастой приведены в таблице 4.

Таблица 4 – Параметры профиля пайки оловянно-свинцовой паяльной пастой

Параметр	Оловянно-свинцовый припой
Температура солидуса припоя, °С	183
Температурный диапазон пайки припоем, °С	210–220
Минимальная пиковая температура пайки (в самой холодной точке платы), °С	205
Скорость нагрева компонента, °С/с	1-4
Скорость охлаждения компонента, °С/с	2-4
Температура предварительного нагрева, °С	100-180
Длительность предварительного нагрева, с	60-120
Время пребывания в расплавленном состоянии, с	60-90
Максимальная выдержка при пиковой температуре, с	20

Микросхемы пригодны для монтажа методом ручной пайки непосредственным присоединением с оплавлением шариков. Материал шариков: Sn63 / Pb37.

Крышка корпуса электрически изолирована от выводов микросхемы.

При ремонте аппаратуры и измерении электрических параметров микросхем замену микросхем необходимо проводить только при отключенных источниках питания.

Запрещается подведение каких–либо электрических сигналов (в том числе шин «Питание», «Общий») к выходам микросхем, не используемым согласно схеме электрической.

Типовая схема включения микросхем приведена на рисунке 80.

5 Описание функций выводов

Большинство входов процессора являются синхронными – связанными с соответствующим тактовым генератором, и лишь некоторые – асинхронными. Для этих асинхронных сигналов встроена схема синхронизации, которая предотвращает проблемы нестабильности.

Для синхронизации сигналов, в случае, когда в системе требуется их предсказуемое поведение, используйте данные, приведенные в разделе «Временные характеристики».

Некоторые порты процессора могут работать в двунаправленном режиме. Во время сброса процессор переводит все выходы в высокоомное состояние, и их фактическое значение определяется внутренними резистивными подтяжками к питанию или к земле. Внутренние резисторы этих портов (номинал резисторов может иметь разброс ± 30 %) поддерживают определенное значение при отсутствии активного драйвера на этом сигнале.

Таблица 5 – Описание выводов – тактовые сигналы и сброс

Сигнал	Тип сигнала	Описание
SCLK	I	Входной сигнал опорного генератора сигнального процессора для шины кластера. Частота ядра процессора задается пользователем с помощью выводов DIVF[7:0], DIVQ[2:0]. RANGE[2:0]
nRST_IN	I	Аппаратный сброс. Переводит сигнальный процессор в начальное состояние. Для получения подробной информации см. раздел «Сброс и запуск», рисунок 67.
nRST_OUT	O	Выход сигнала сброса. Обозначает завершение сброса сигнального процессора.

Примечания:
Обозначение типов выводов:
I – вход;
O – выход;
IO – вход/выход.

Таблица 6 – Конфигурация PLL

Вывод	Назначение	Ограничения
DIVF[7:0]	Постделитель частоты ядра PLL (двоичное число + 1 : 00000000 = ÷1)	Настройки должны поддерживать значения параметров рабочего режима PLL
DIVQ[2:0]	Оконечный делитель (2^(двоичное число)): 001 = ÷2, 010 = ÷4, 011 = ÷8, 100 =÷16, 101 =÷32	
RANGE[2:0]	Диапазон фильтра PLL 000 = не используется 001 = 10–16 МГц 010 = 16–25 МГц 011 = 25–40 МГц 100 = 40–65 МГц 101 = 65–100 МГц 110 = 100–160 МГц 111 = 160–200 МГц	Диапазон фильтра PLL должен быть выбран в соответствии с выходной частотой постделителя. Для уменьшения джиттера рекомендуется выбирать максимальный частотный диапазон

Таблица 7 – Описание выводов – управление шиной внешнего порта

Сигнал	Тип сигнала	Описание
ADDR31–0	IO (pu_1)	Адресная шина для доступа к памяти и периферийным устройствам. В мультипроцессорной системе процессор, который захватывает шину (мастер), формирует адреса для доступа к внутренней памяти, внутренним регистрам другого процессора и другим ресурсам.
DATA63–0	IO (pu_1)	Внешняя шина данных. Участники обмена записывают и считывают данные и команды по этой шине. В случае, если сипользуются не все биты шины, подключать к ним внешние задающие резисторы не обязательно.
nRD	IO (pu_2_id0)	Строб чтения памяти. nRD формируется мастером шины для доступа к другим устройствам, за исключением SDRAM. Если процессор ведомый, то nRD является входом и определяет цикл чтения внутренней памяти или внутреннего регистра. nRD изменяется синхронно с шиной ADDR.
nWRL	IO (pu_2_id0)	Строб записи младшего слова. nWRL формируется мастером шины для записи в другие устройства, за исключением SDRAM. В случае 64-битной внешней шины этот сигнал упрявляет записью по четным адресам. В случае 32-битной внешней шины сигнал активен при любой записи. Если процессор ведомый, то nWRL является входом и определяет цикл записи внутренней памяти или внутреннего регистра. nWRL изменяется синхронно с шиной ADDR.
nWRH	IO (pu_2_id0)	Строб записи старшего слова. nWRH формируется мастером шины для записи в другие устройства, за исключением SDRAM. В случае 64-битной внешней шины этот сигнал упрявляет записью по нечетным адресам. В случае 32-битной внешней шины это сигнал не используется. Если процессор ведомый, то nWRH является входом и определяет цикл записи внутренней памяти или внутреннего регистра. nWRH изменяется синхронно с шиной ADDR.
ACK	IO (pu_2, pu_4)	Сигнал удлинения цикла обмена по внешней шине (вставка циклов ожидания при неготовности ведомого устройства).
nBMS	IO (pu_2_id0, pd_2_id0)	Сигнал выбора кристалла внешней загрузочной EPROM- или флеш-памяти. Во время сброса процессор использует nBMS в качестве конфигурационного входа (EBOOT) для выбора режима запуска. В мультипроцессорной системе сигнал nBMS формирует процессор, захвативший внешнюю шину. Для получения подробной информации см. раздел «Сброс и запуск» и описание сигнала EBOOT (таблица 16).
nMS[1-0]	IO (pu_2_id0)	Сигналы выбора кристалла внешней памяти. MS0 или MS1 формируются при доступе к банкам 0 и 1 (соответственно) внешнего адресного пространства. Когда ADDR31:27 = 0b00110, формируется MS0. Когда ADDR31:27 = 0b00111, формируется MS1. В мультипроцессорной системе сигналы MS1–0 формируются процессором, захватившим внешнюю шину.
nMSH	IO (pu_2_id0)	Сигнал выбора кристалла внешнего хост устройства. nMSH формируется когда процессор обращается к адресному пространству хост устройства (ADDR31 = 0b1). nMSH изменяется синхронно с шиной ADDR. В мультипроцессорной системе nMSH формируются процессором, захватившим внешнюю шину.

nBRST	IO (pu_2_id0)	Сигнал блочного доступа. Этот сигнал формируется процессором, захватившим внешнюю шину, для доступа (чтения или записи) блока данных. Ведомый процессор при активном сигнале nBRST может игнорировать значения ADDR и самотоятельно вычислять нужный адрес при доступе к своей памяти.

Примечания:

1 Обозначение типов выводов:

I – вход;

O – выход;

IO – вход/выход.

2 Номиналы подтягивающих резисторов:

pd_1 – подтягивающий к земле резистор 40 кОм;

pu_1 – подтягивающий к питанию резистор 40 кОм;

pd_2 – подтягивающий к земле резистор 5 кОм;

pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;

pu_2 – подтягивающий к питанию резистор 5 кОм;

pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;

pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;

pu_4 – подтягивающий к питанию резистор 50 Ом.

Таблица 8 – Описание выводов арбитража внешнего порта

Сигнал	Тип сигнала	Описание
nBR[0-7]	IO	Сигнал запроса на захват внешней шины в мультипроцессорной системе. Используются процессорами для арбитража внешней шины. Каждый процессор управляет собственным сигналом nBRx (в соответствии со значениями входных сигналов ID2–0) и отслеживает остальные. В мультипроцессорных системах с сс количеством процессоров меньше четырех, необходимо установить высокий уровень на выводах nBRx (V_{DD_IO}).
ID[2-0]	I (pd_2)	Идентификатор процессора в мультипроцессорной системе. Эти выводы также задают тот сигнал nBRx, котрый необходимо активизировать при запросе на захват шины: 000 = BR0, 001 = BR1, 010 = BR2, 011 = BR3, 100 = BR4, 101 = BR5, 110 = BR6, 111 = BR7. ID2–0 должен быть стабильным во время работы системы и задаваться только во время сброса.
nBM	IO	Сигнал, формируемый процессором, который захватил внешнюю шину и используется только для отладки. При сбросе является конфигурационным входом. Более подробная информация приведена в таблице 16.
nBOFF	I	Сигнал принудительного освобождения шины. Ситуация блокировки может возникнуть, когда хост устройство и процессор пытаются одновременно обратиться друг к другу. Когда происходит блокировка, хост устройство может сформировать nBOFF для принудительного освобождения процессором внешней шины.
nBUSLOCK	IO (pu_2_id0, pd_2_id0)	Индикация захвата шины. При сбросе является конфигурационным входом. Более подробная информация приведена в таблице 16.

nHBR	I	Запрос на захват шины от хост устройства. Хост устройство должно сформировать nHBR для того, чтобы захватить внешнюю шину. Процессор, который в данный момент захватил шину, после завершения цикла обмена освобождает внешнюю шину и формирует сигнал nHBG.
nHBG	IO (pu_2_id0)	Сигнал передачи внешней шины хост устройству. Обработкой nHBR и формированием nHBG управляет мастер шины. Мастер также информирует внешнее хост устройство, что последний обмен завершен и шина свободна. При освобождении шины процессор переводит в высокоомные состояния выводы ADDR31–0, DATA63–0, nMSH, nMSSD3–0, MS1–0, nRD, nWRL, nWRH, nBMS, nBRST, nIORD, nIOWR, nIOEN, nRAS, nCAS, nSDWE, SDA10, SDCKE, LDQM и HDQM, и переводит SDRAM в режим автогенерации. Процессор продолжает удерживать nHBG, пока хост устройство не закончит обмен и не снимет nHBR.
nCPA	IO (pu_3_id0)	Синал приоритетного доступа ядра к шине. Сигнал формируется, когда процессор при выполнении кода обращается к внешней шине. Этот сигнал позволяет процессору захватить шину, по которой идет передача данных через DMA, и осуществить требуемый цикл обмена на внешней памяти. nCPA является выходом с открытым стоком, подключенным ко всем DSP в системе Если этот сигнал не используется в системе, то следует оставить его не подключенным (для процессоров с ID = 001 – 111 нужны внешние резисторы поддяжки к питанию).
nDPA	IO (pu_3_id0)	Синал приоритетного доступа к шине DMA. Сигнал формируется, когда требуется высокоприоритетный доступ к шине по каналу DMA. В этом случае процессор, который осуществляет обычный обмен по каналу DMA ,освобождает шину, а процессор с приоритетным доступом к шине по каналу DMA захватывает ее и осуществляет обмен. nDPA является выходом с открытым стоком, подключенным ко всем DSP в системе. Если этот сигнал не используется в системе, то следует оставить его не подключенным (для процессоров с ID = 001 – 111 нужны внешние резисторы поддяжки к питанию).

Примечания:

1 Обозначение типов выводов:

I – вход;

O – выход;

IO – вход/выход.

2 Номиналы подтягивающих резисторов:

pd_1 – подтягивающий к земле резистор 40 кОм;

pu_1 – подтягивающий к питанию резистор 40 кОм;

pd_2 – подтягивающий к земле резистор 5 кОм;

pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;

pu_2 – подтягивающий к питанию резистор 5 кОм;

pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;

pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;

pu_4 – подтягивающий к питанию резистор 50 Ом.

Таблица 9 – Описание выводов внешнего порта DMA/Flyby

Сигнал	Тип сигнала	Описание
nDMAR[3-0]	I	Выводы запроса на обмен по каналу DMA. В ответ на запрос nDMARx процессор осуществляет передачу DMA в соответствии с инициализированным каналом DMA. Процессор игнорирует запросы DMA к неинициализированных каналов.
nIOWR	IO (pu_2_id0)	Строб записи внешнего устройства. Когда DMA канал процессора инициирует транзакции чтения в режиме flyby, процессор формирует сигнал nIOWR в течение цикла обмена данных. Это позволяет устройству ввода-вывода захватывать данные вместо процессора.
nIORD	IO (pu_2_id0)	Строб чтения внешнего устройства. Когда DMA канал процессора инициирует транзакции записи режима flyby, процессор формирует сигнал nIORD в течение цикла обмена данных. Что совместно с nIOEN позволяет устройству ввода-вывода формировать данные вместо процессора.
nIOEN	IO (pu_2_id0)	Сигнал разрешения формирования данных внешним устройством. Разрешает выходным буферам внешнего устройства ввода-вывода операции прямой передачи данных между устройством и внешней памятью. Активный только в режиме flyby.

Примечания:

1 Обозначение типов выводов:
- I – вход;
- O – выход;
- IO – вход/выход.

2 Номиналы подтягивающих резисторов:
- pd_1 – подтягивающий к земле резистор 40 кОм;
- pu_1 – подтягивающий к питанию резистор 40 кОм;
- pd_2 – подтягивающий к земле резистор 5 кОм;
- pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;
- pu_2 – подтягивающий к питанию резистор 5 кОм;
- pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;
- pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;
- pu_4 – подтягивающий к питанию резистор 50 Ом.

Таблица 10 – Описание выводов контроллера внешнего порта SDRAM

Сигнал	Тип сигнала	Описание
nMSSD[3-0]	IO (pu_2_id0)	Выбор микросхемы памяти SDRAM. nMSSD0, nMSSD1, nMSSD2, или nMSSD3 формируются, когда процессор обращается к пространству памяти SDRAM (ADDR31:30 = 0b01, кроме зарезервированного пространства, представленного на рисунке 12). В мультипроцессорных системах процессор, который захватывает шину, формирует nMSSD3–0.
nRAS	IO (pu_2_id0)	Выбор адреса строки. При активном уровне nRAS указывает на то, что адрес строки является действительным при чтении или записи SDRAM. При других обменах с SDRAM он определяет тип выполняемой операции в соответствии со спецификацией SDRAM.

nCAS	IO (pu_2_id0)	Выбор адреса столбца. При активном уровне nCAS указывает на то, что адрес столбца является действительным в чтении или записи SDRAM. При других обменах с SDRAM он определяет тип выполняемой операции в соответствии со спецификацией SDRAM.
LDQM	IO (pu_2_id0)	Маска данных младшего слова SDRAM. При высоком уровне сигнала переводит DQ буферы SDRAM в высокоомные состояния. В транзакциях SDRAM LDQM является активным, когда активен nCAS и неактивным – во время операций чтения. В транзакциях записи LDQM является активным при доступе к нечетным адресам в 64-битном режиме внешней шины.
HDQM	IO (pu_2_id0)	Маска данных старшего слова SDRAM. При высоком уровне сигнала переводит DQ буферы SDRAM в высокоомные состояния. В транзакциях SDRAM HDQM является активным, когда активен nCAS и неактивным – во время операций чтения. В транзакциях записи HDQM является активным при доступе к четным адресам в 64-битном режиме внешней шине или при любом обмене в 32-битном режиме.
SDA10	IO (pu_2_id0)	Бит 10 шины адреса. Этот сигнал шины адреса участвует в операциях восстановления SDRAM-памяти при отсутствии явного доступа к ней.
SDCKE	IO (pu_2, pd_2)	Сигнал разрешения тактового сигнала SDRAM. Включает тактовый сигнал SDRAM для самовосстановления SDRAM или экономичного режима. У подчиненного контроллера в мультипроцессорной системе нет резистров к земле или к питанию. Мастер шины (или ID = 000 в однопроцессорной системе) через резистор к питанию подтягивает сигнал к 1 перед тем, как освободить шину для хост устройства, за исключением случая, когда SDRAM находится в режиме самовосстановления. В случае, когда SDRAM находится в режиме самосвосстановления, ведущий процессор через резистор подтяжки к земле поддятигавает этот сигнал к 0 перед тем, как освободить шину для хост устройства.
nSDWE	IO (pu_2_id0)	Сигнал разрешения записи в SDRAM. При активном сигнале nCAS (nCAS = 0), сигнал nSDWE определяет тип доступа (чтение/запись) к SDRAM. В случае неактивного значения nCAS функция nSDWE определяется спецификацией на SDRAM память.

Примечания:

1 Обозначение типов выводов:

I – вход;

O – выход;

IO – вход/выход.

2 Номиналы подтягивающих резисторов:

pd_1 – подтягивающий к земле резистор 40 кОм;

pu_1 – подтягивающий к питанию резистор 40 кОм;

pd_2 – подтягивающий к земле резистор 5 кОм;

pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;

pu_2 – подтягивающий к питанию резистор 5 кОм;

pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;

pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;

pu_4 – подтягивающий к питанию резистор 50 Ом.

Таблица 11 – Описание выводов порта JTAG

Сигнал	Тип сигнала	Описание
nEMU	O	Эмуляция. Подключается только к разъему аппаратного отладчика JTAG.
TCK	I	Сигнал синхронизации (JTAG)
TDI	I (pu_1)	Вход данных аппаратного отладчика JTAG
TDO	O	Выход данных аппаратного отладчика JTAG
TMS	I (pu_1)	Выбор режима аппаратного отладчика JTAG. Используется для управления переходами конечного автомата JTAG
nTRST	I (pu_1)	Сброс тестового режима (JTAG). Сбрасывает режим конечного автомата JTAG

Примечания:

1 Обозначение типов выводов:

I – вход;

O – выход;

IO – вход/выход.

2 Номиналы подтягивающих резисторов:

pd_1 – подтягивающий к земле резистор 40 кОм;

pu_1 – подтягивающий к питанию резистор 40 кОм;

pd_2 – подтягивающий к земле резистор 5 кОм;

pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;

pu_2 – подтягивающий к питанию резистор 5 кОм;

pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;

pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;

pu_4 – подтягивающий к питанию резистор 50 Ом.

Таблица 12 – Описание выводов – флаги, прерывания и таймер

Сигнал	Тип сигнала	Описание
FLAG[3-0]	IO (pu_2)	Сигналы FLAG. Двунаправленные порты предназначены для пользовательского ввода/вывода и могут использоваться в качестве условий программы. Каждый вывод может быть сконфигурирован индивидуально как вход или выход. После сброса или включения процессора порт FLAG3–0 сконфигурирован как вход.
nIRQ[3-0]	IO (pu_2)	Запрос на прерывание. Эти сигналы формируют прерывание работы процессора. Каждый из выводов nIRQ3–0 может быть независимо сконфигурирован для формирования прерывания по фронту сигнала или по уровню. В зависимости от выбранного способа загрузки, после сброса процессора прерывания от этих сигналов либо запрещены до тех пор, пока программа не изменит начальную конфигурацию, либо разрешены и могут использоваться для старта процессора из внешней памяти.
TMR0E	IO	Сигнал переполнения таймера 0. Каждый раз при переполнении таймера 0 на выходе формируется импульс. При сбросе процессора этот сигнал сконфигурирован как вход. Подробнее в таблице 16.

Примечания:

1 Обозначение типов выводов:

I – вход;

O – выход;
IO – вход/выход.
2 Номиналы подтягивающих резисторов:
pd_1 – подтягивающий к земле резистор 40 кОм;
pu_1 – подтягивающий к питанию резистор 40 кОм;
pd_2 – подтягивающий к земле резистор 5 кОм;
pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;
pu_2 – подтягивающий к питанию резистор 5 кОм;
pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;
pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;
pu_4 – подтягивающий к питанию резистор 50 Ом.

Таблица 13 – Описание выводов последовательного высокоскоростного порта

Сигнал	Тип сигнала	Описание
L0DATOP[3-0]	O	Положительный выход LVDS шины данных передатчика
L1DATOP[3-0]	IO	
L2DATOP[3-0]	O	
L3DATOP[3-0]		
LxDATON[3-0]	O	Отрицательный выход LVDS шины данных передатчика
LxCLKOP	O	Положительный выход LVDS сигнала синхронизации передатчика
LxCLKON	O	Отрицательный выход LVDS сигнала синхронизации передатчика
LxACKI	I (pd_1)	Сигнал готовности приемника. Используется для информирования передатчика, что приемник готов принять новые данные.
nL0BCMPO	IO (pu_1)	Сигнал завершения блочной передачи. В случае передачи данных через DMA этот сигнал информирует приемник, о том что передаваемый блок данных завершен. Резистор к питанию присутствует только на nL0BCMPO. При сбросе выводы nL1BCMPO, nL2BCMPO и nL3BCMPO являются конфигурационными входами. Подробнее приведено в таблице 16.
nL1-3BCMPO	IO	
LxDATIP[3-0]	I	Положительный вход LVDS шины данных передатчика
LxDATIN[3-0]	I	Отрицательный вход LVDS шины данных передатчика
LxCLKIP	I	Положительный вход LVDS сигнала синхронизации передатчика
LxCLKIN	I	Отрицательный вход LVDS сигнала синхронизации передатчика
LxACKO	O	Сигнал используется приемником для информирования передатчика о том, что он может принять новые данные
nLxBCMPI	I (pd_1)	Сигнал завершения передачи блока данных. Если идет передача через DMA, этот сигнал указывает приемнику, что блок передачи данных завершен

Примечания:
1 Обозначение типов выводов:
I – вход;
O – выход;

IO – вход/выход.
2 Номиналы подтягивающих резисторов:
pd_1 – подтягивающий к земле резистор 40 кОм;
pu_1 – подтягивающий к питанию резистор 40 кОм;
pd_2 – подтягивающий к земле резистор 5 кОм;
pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;
pu_2 – подтягивающий к питанию резистор 5 кОм;
pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;
pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;
pu_4 – подтягивающий к питанию резистор 50 Ом.

Таблица 14 – Описание выводов – конфигурационные сигналы и термодиод

Сигнал	Тип сигнала	Описание
E_PLLBP	I (pd_1)	Сигнал отключения компенсиции задержки синхросигнала внутри кристалла. Имеет внутреннюю подтяжку к земле.
JG_MX	I (pd_1)	Выбор внутреннего JTAG контроллера (0 – функциональный JTAG контроллер, 1 – тестовый JTAG контроллер). Имеет внутреннюю подтяжку к земле.
DXP	I	Анод внутреннего термодиода
DXN	I	Катод внутреннего термодиода

Примечания:
1 Обозначение типов выводов:
I – вход;
O – выход;
IO – вход/выход.
2 Номиналы подтягивающих резисторов:
pd_1 – подтягивающий к земле резистор 40 кОм;
pu_1 – подтягивающий к питанию резистор 40 кОм;
pd_2 – подтягивающий к земле резистор 5 кОм;
pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;
pu_2 – подтягивающий к питанию резистор 5 кОм;
pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0;
pu_3_id0 – подтягивающий к питанию резистор 240 Ом только для процессора с ID=0;
pu_4 – подтягивающий к питанию резистор 50 Ом.

Таблица 15 – Описание выводов – питание, земля

Сигнал	Тип сигнала	Описание
VDD	PWR	Выводы питания от цифровой логики
VDD_A	PWR	Выводы питания внутреннего PLL
VDD_IO	PWR	Выводы питания портов ввода/вывода
VSS	PWR	Вывод земли

Примечание – Обозначение типов выводов:
PWR – вывод питания и «общий».

6 Описание конфигурационных портов

Некоторые порты имеют альтернативную функцию при включении питания. Конфигурация определяется режимом работы процессора. Во время сброса процессор захватывает значения конфигурационных портов. Конфигурационные входы имеют встроенные резисторы подтяжки к питанию или к земле, с помощью которых задаются значения по умолчанию. Если конфигурационный порт не подключен через внешний резистор к земле или питанию или не задается внешней логикой, то во время сброса процессор захватывает значение по умолчанию. Если конфигурационные порты также являются входами внутренней логики, может потребоваться внешний резистор с меньшим значением сопротивления, чтобы надежно установить логический уровень, зависящий от токов утечки. Чтобы задать значение конфигурации, не совпадающее со значением по умолчанию, нужно подключить вход конфигурации к внешнему резистору достаточно малого сопротивления. Описание конфигурационных входов и их назначения приведены в таблице ниже.

Таблица 16 – Описание функций конфигурационных портов

Сигнал	Тип (при сбросе)	Вывод	Описание
EBOOT	IO (pd_2_id0)	nBMS	Загрузка из внешней EPROM-памяти. 0 = загрузка из EPROM сразу после сброса (по умолчанию) 1 = останов процессора после сброса и ожидание загрузки процессора внешним устройством через последовательный или параллельный порт
IRQEN	IO	nBM	Включение прерывания. 0 = выключение и установка прерываний nIRQ3–0 в режим срабатывания по фронту сигнала сразу после сброса (по умолчанию) 1 = включение и установка прерываний nIRQ3–0 в режим со срабатыванием по уровню сразу после сброса
LINK_DWIDTH	IO	TMR0E	Ширина шины данных LVDS порта. 0 = 1 бит 1 = 4 бит
SYS_REG_WE	IO (pd_2_id0)	nBUSLOCK	Разрешение записи SYSCON и SDRCON. 0 = возможность однократной записи после сброса (по умолчанию) 1 = возможность многократной записи
Примечания: 1 Обозначение типов выводов: IO – вход/выход. 2 Номиналы подтягивающих резисторов: pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0.			

При использовании конфигурации по умолчанию, внешние резисторы конфигурационных входов не требуются. Для применения других конфигураций рекомендуется подключать конфигурационные входы к V_{DD_IO} или к V_{SS} через резистор 500 Ом.

Все конфигурационные входы захватываются по завершению сброса. Каждый вывод защелкивает состояние вывода (состояние конфигурационного входа по нарастающему фронту nRST_IN). Вскоре после завершения сброса, эти выводы возвращаются к своим обычным функциям.

Конфигурационные входы могут иметь внутренний резистор к земле, резистор к питанию или не иметь резистора. Тип резистора, который подключается к входам/выводам, зависит от того, является ли активным сигнал nRST_IN (низкий уровень) или не является (высокий уровень).

В таблице 17 указаны резисторы, задействованные во время активного сброса и во время нормальной работы.

Таблица 17 – Внутренний резистор конфигурационных входов, активный сброс (nRST_IN = 0) / нормальная работа (nRST_IN = 1)

Вывод	nRST_IN = 0	nRST_IN = 1
nBMS	(pd_2_id0)	(pu_2_id0)
nBUSLOCK	(pd_2_id0)	(pu_2_id0)

Обозначения в таблице:

pd_2_id0 – подтягивающий к земле резистор 5 кОм только для процессора с ID=0;

pu_2_id0 – подтягивающий к питанию резистор 5 кОм только для процессора с ID=0.

7 Архитектура процессора

Процессор состоит их двух архитектурных частей:
- ядро процессора (рисунок 4), где исполняются команды;
- периферийные устройства (рисунок 5) для хранения данных и осуществления операций обмена с внешними устройствами.

В процессоре можно выделить следующие элементы:

- Два вычислительных модуля: X и Y, каждый из которых содержит умножитель, ALU, CLU, сдвиговое устройство и регистровый файл объемом в 32 слова.
- Два блока целочисленных ALU: J и K, каждый из которых содержит 32-битное целочисленное ALU, а также регистровый файл объемом в 32 слова
- Устройство управления (Sequencer), управляющее ходом исполнения программы, содержащее буфер выравнивания команд (instruction alignment buffer – IAB) и буфер целевых адресов перехода (branch target buffer – BTB).
- Три 128-битные шины, обеспечивающие возможность высокоскоростного обмена между внутренней памятью и другими компонентами ядра процессора (вычислительными блоками, блоками целочисленных ALU, устройством управления и SOC-интерфейсом).
- 128-битную шину, обеспечивающую возможность высокоскоростного обмена между внутренней памятью и периферийными устройствами внешнего ввода/вывода (DMA, внешним портом и портами линков).
- SOC-интерфейс, обеспечивающий связь между внутренними шинами ядра и шиной периферийных устройств.
- Интерфейс внешнего порта, включая хост-интерфейс, контроллер SDRAM, конвейерный интерфейс со статической организацией конвейера, четыре канала DMA, четыре порта LVDS-линков, каждый с двумя каналами DMA, и поддержку многопроцессорной работы.
- 24 Мбит внутренней памяти, организованной как шесть 4-Мбитных блоков, каждый из которых содержит 128К 32-битных слов.
- Средства поддержки отладки.
- Тестовый порт JTAG.

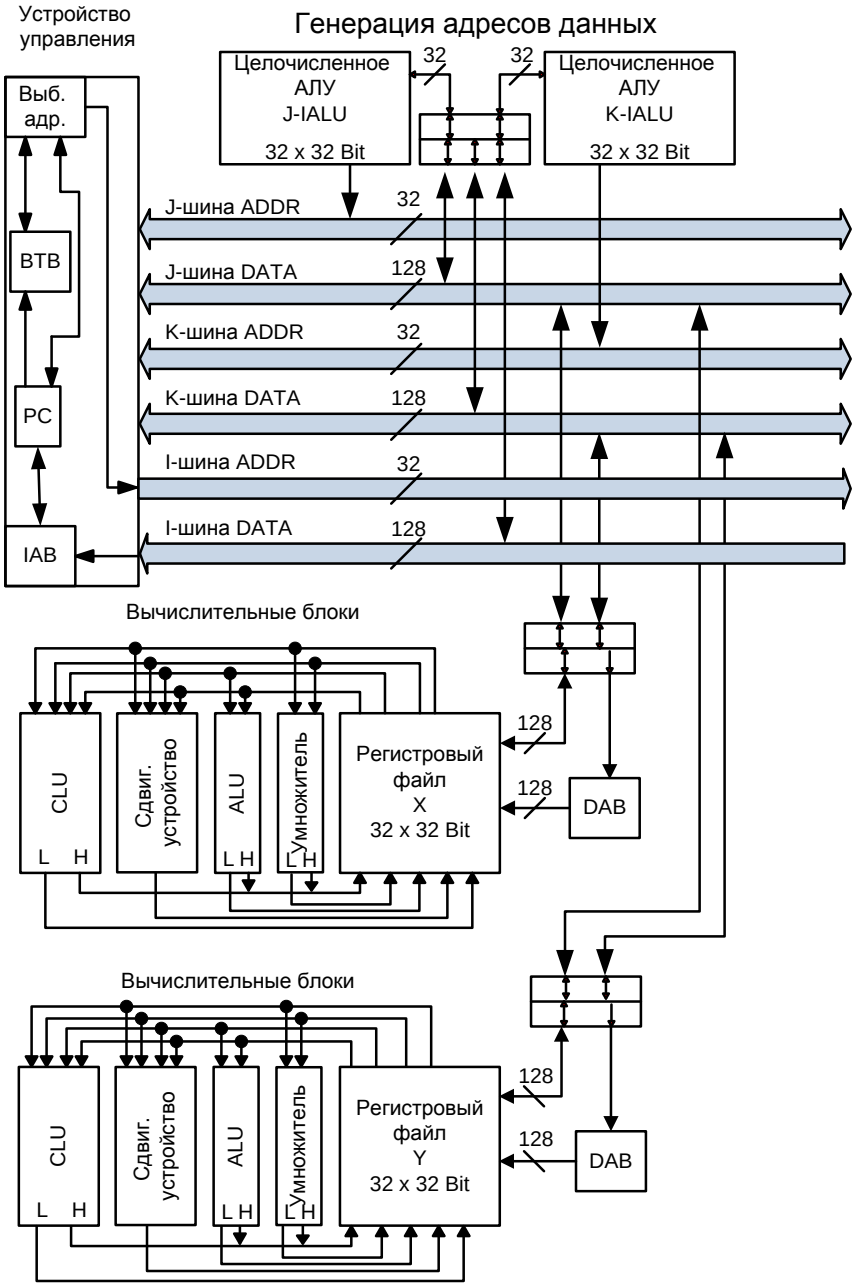


Рисунок 4 – Схема ядра процессора

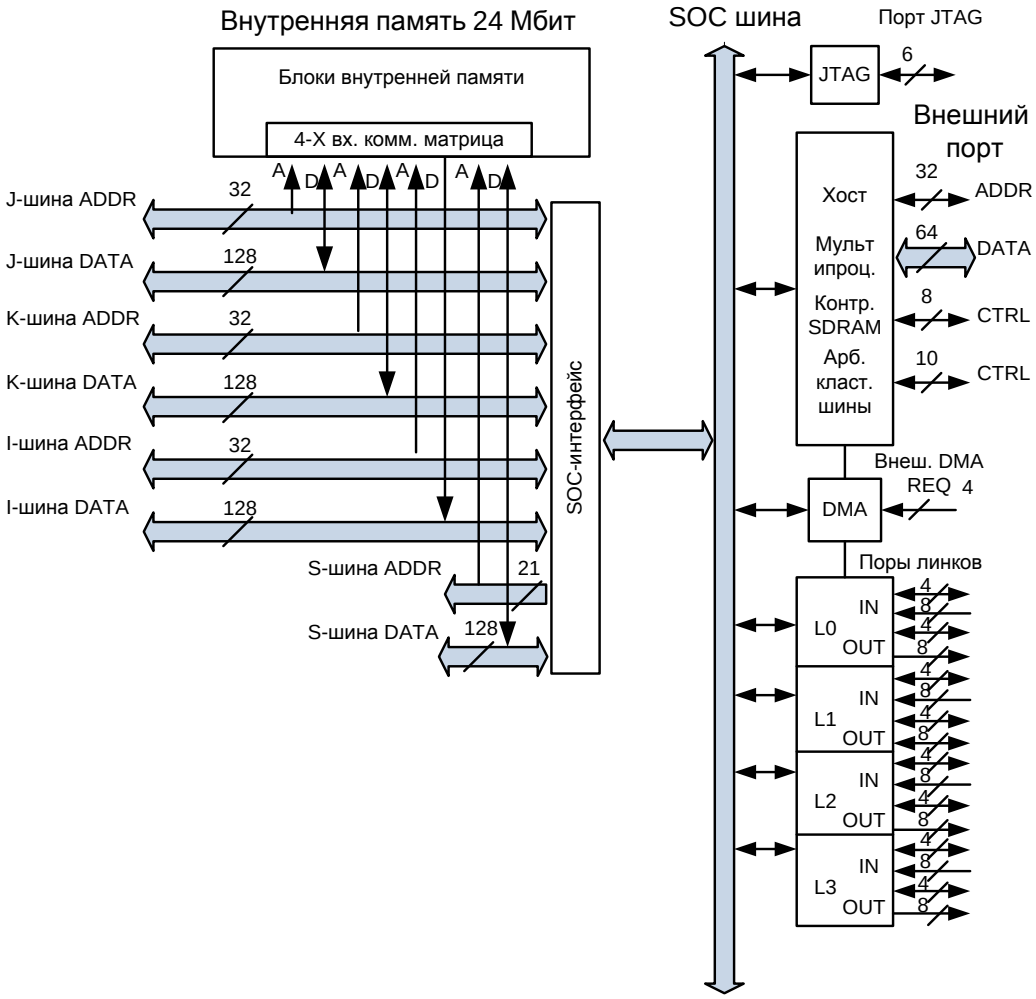


Рисунок 5 – Схема подключения периферийных устройств к ядру процессора

Внешний порт процессора обеспечивает интерфейс с внешней памятью, с устройствами ввода/вывода, отображенными в память, с хост-процессором, а также с дополнительными процессорами. Внешний порт выполняет арбитраж внешней шины и подает управляющие сигналы на общую глобальную память, SDRAM и устройства ввода/вывода.

На рисунке 6 показана типовая однопроцессорная система. Мультипроцессорная система показана на рисунке 7.

В процессоре имеется ряд особенностей, которые упрощают создание систем на его базе. Эти особенности заключаются в трех ключевых моментах:

- поддержка форматов с плавающей точкой по стандарту IEEE;
- средства последовательного сканирования и внутрикристальной эмуляции в соответствии со стандартом IEEE 1149.1 JTAG;
- архитектурные особенности, обеспечивающие поддержку языков высокого уровня и операционных систем.

Особенности процессора, которые непосредственно обеспечивают поддержку компиляторов языков высокого уровня и операционных систем, включают в себя:

- простые ортогональные команды, позволяющие компилятору эффективно использовать поля многокомандных строк;
- файлы регистров данных общего назначения и регистров целочисленных ALU;

- аппаратно поддерживаемые 32-, 64-битные (в соответствии со стандартом IEEE 754/854) и 40-битные типы данных с плавающей точкой, а также аппаратно поддерживаемые 8-, 16-, 32- и 64-битные типы данных с фиксированной точкой;
- большое адресное пространство;
- возможность непосредственной модификации адреса;
- легко поддерживаемая перемещаемость кода и данных;
- быстрое сохранение и восстановление регистров процессора в стеках во внутренней памяти.

Архитектура данного процессора ближе к архитектуре VLIW (Very Large Instruction Word), где исполняющее устройство (одно из нескольких) для команды назначается заранее (статически) – на этапе компиляции или на этапе написания программы, если программа на ассемблере. Высокая производительность процессора достигается отчасти за счет возможности исполнения до четырех 32-битных команд в одном такте. Не обязательно выравнивание команд на какие-то границы в памяти, что позволяет не расходовать программную память впустую.

Внутренняя память объемом 24 Мбит разбита на шесть блоков памяти по 128 Кслов. Каждая из четырех пар внутренних шин «адрес/данные» подсоединена ко всем шести блокам памяти через коммутационную матрицу. Шесть блоков памяти поддерживают до четырех обращений в каждом такте, причем каждый блок памяти может выполнить 128-битное обращение за такт.

Кластерная шина, на которую выходит внешний порт, имеет ширину в 64 бита. Высокая пропускная способность ввода/вывода сочетается с высокой скоростью работы ядра. Для достижения высокой тактовой частоты процессор использует конвейерную внешнюю шину с программируемой глубиной конвейера для межпроцессорных обменов, а также для синхронной статической памяти (SSRAM) и для синхронной динамической памяти (SDRAM).

Высокую пропускную способность передачи данных из точки в точку поддерживают четыре LINK-порта в режиме LVDS. Каждый LINK-порт обеспечивает полнодуплексную связь.

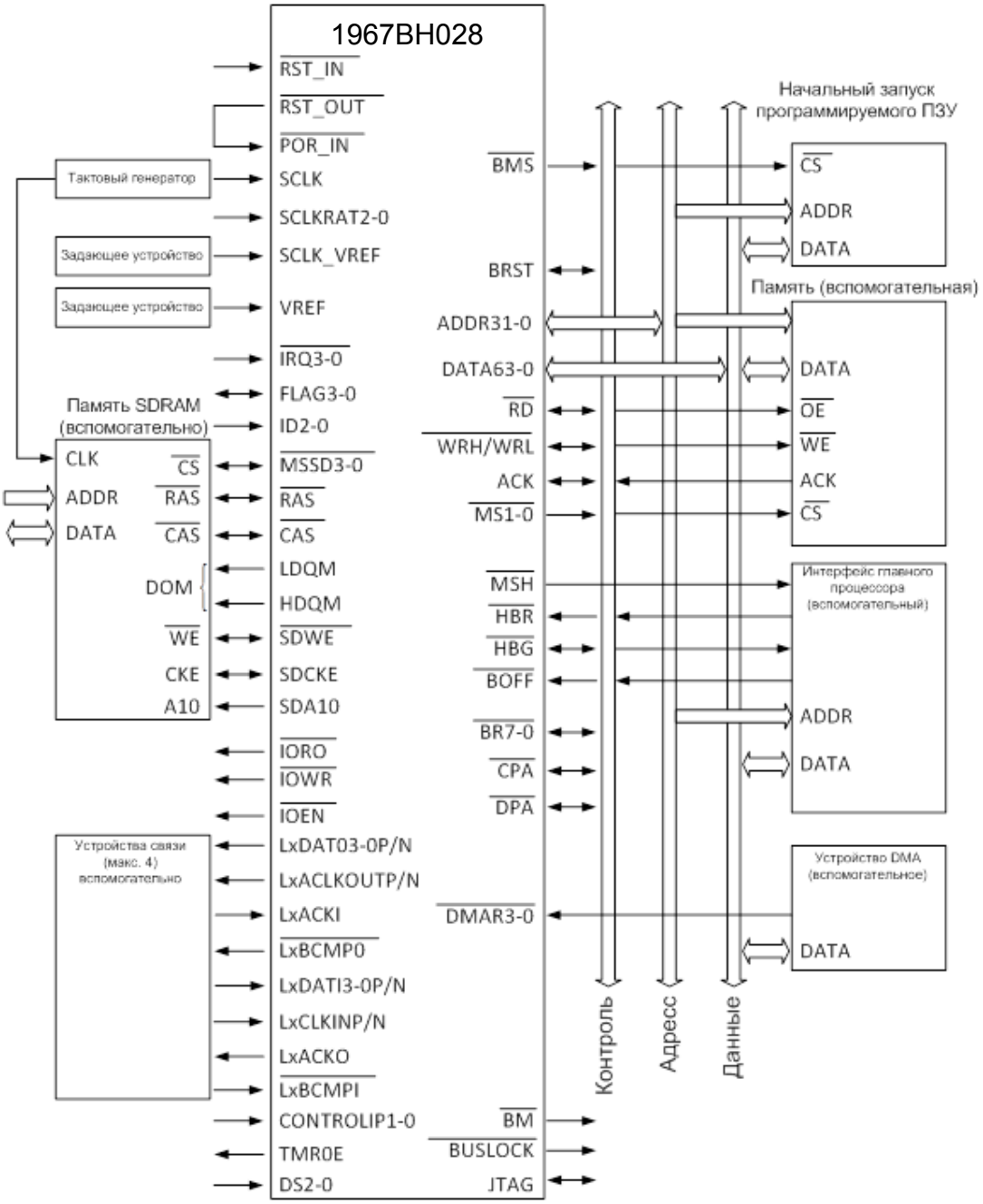


Рисунок 6 – Однопроцессорная конфигурация

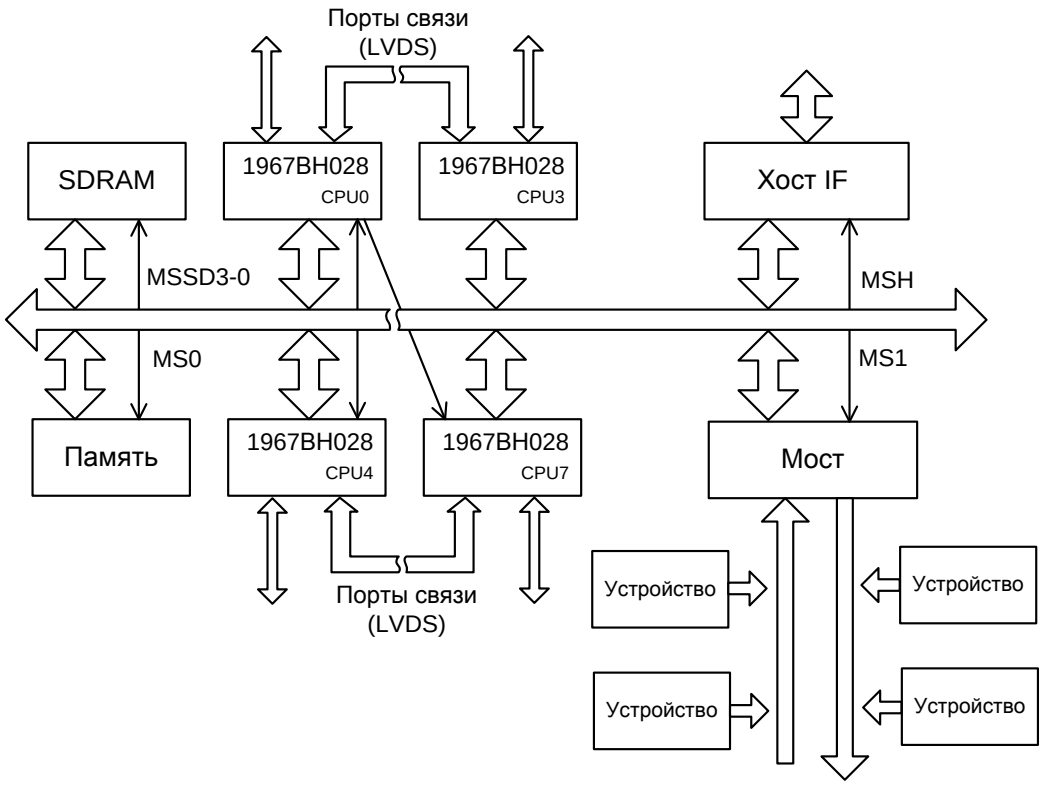


Рисунок 7 – Мультипроцессорная кластерная конфигурация

7.1 Ядро процессора

Ядро процессора состоит из пяти самостоятельных функциональных модулей:

- модули X и Y – вычислительные модули (устройства);
- модули J и K – целочисленные АЛУ;
- модуль S – устройство управления.

В модулях X и Y происходит основная обработка данных, и они в свою очередь включают в себя различные устройства обработки данных. Модули X и Y абсолютно симметричны, т.е. выполняют одинаковые функции. Каждый модуль обрабатывает свой поток команд. При этом существует возможность управлять модулями X и Y с помощью одной команды, как одним единым модулем.

Модули J и K формируют адреса для доступа к памяти, а также выполняют функции по обработке данных. При этом данные (как и адреса) могут иметь только тип Integer, что соответствует 32-разрядному целому числу. Модули J и K также симметричны, и каждый из них выполняет свой поток команд.

Модуль S формирует адреса команд, управляет потоком команд, а также управляет работой всего конвейера ядра. Модуль S также имеет способность выполнять несколько потоков команд определенных типов. Среди них основными являются команды переходов и вызовов.

Модульная архитектура процессора с независимыми потоками команд позволяет организовать высокопараллельные вычисления. Так в процессе интенсивных вычислений одно или оба целочисленных ALU вычисляют или генерируют адреса для выборки до двух операндов размером в квадрослово из двух блоков памяти, в то время как устройство управления одновременно извлекает следующую четверку команд из третьего блока памяти. Параллельно вычислительные устройства могут обрабатывать ранее считанные операнды, а устройство управления подготавливать переход.

Пока ядро процессора занято вышеописанными действиями, каналы DMA могут в фоновом режиме обновлять содержимое внутренней памяти квадрословами данных, либо из внешнего порта, либо из портов линков.

Высокая производительность вычислительного ядра процессора при цифровой обработке сигналов достигается за счет следующих особенностей:

- вычислительный конвейер;
- пара вычислительных устройств;
- исполнение до четырех команд за такт;
- выборка/запись до восьми слов памяти за такт.

Два идентичных вычислительных устройства (X и У) выполняют арифметические операции как с плавающей, так и с фиксированной точкой. Эти устройства за один такт выполняют до шести операций с плавающей точкой или до 24 операций с фиксированной точкой.

Модули J и K выполняют SISD-обработку, модули X и У могут выполнять SISD- и SIMD-обработку. Модули X и У имеют сложную структуру и включают в себя разнообразные вычислительные блоки.

7.1.1 Вычислительные модули

Ядро процессора содержит два вычислительных модуля, каждый из которых содержит регистровый файл и четыре независимых вычислительных блока: ALU, CLU, умножитель и сдвиговое устройство. Вычислительные блоки осуществляют обработку данных в нескольких форматах представления с фиксированной и плавающей точкой.

Форматы данных с фиксированной точкой

В число форматов данных с фиксированной точкой входят:

- 64-битное длинное слово (Long Long);
- 32-битное обычное слово (Integer);
- 32-битное комплексное (16-битная действительная и 16-битная мнимая часть) слово;
- 16-битное короткое слово (Short);
- 8-битное однобайтное слово (Char).

Для коротких слов арифметики с фиксированной точкой учетверенные операции над данными, выровненными на границу квадрослова, обеспечивают быструю обработку вектора данных. Байтовые операции поддерживаются также для данных, выровненных на границу октослова (т.е. 256-битного слова).

Форматы данных с плавающей точкой

В число форматов данных с плавающей точкой входят:

- 32-битное обычное слово (float);
- 64-битное двойное слово (double);
- 40-битное расширенное слово.

Операции с плавающей точкой выполняются с одинарной, двойной и расширенной точностью. Формат обычного слова с плавающей точкой соответствует стандартному IEEE-формату, а 40-битное число формата с увеличенной точностью размещается в двойном слове (64 бита), занимая дополнительно к 32 битам восемь наименее значащих битов (Least Significant Bits – LSBs) мантиссы для достижения большей точности.

Каждый вычислительный модуль имеет многопортовый регистровый файл, содержащий регистры общего назначения для обмена данными между вычислительными блоками и шинами данных, а также для хранения промежуточных результатов. Ко всем этим регистрам можно обращаться как к обычному, двойному или квадрорегистру. Все операции вычислительных модулей исполняются на двух стадиях конвейера. Результат операции доступен только на второй стадии, поэтому если следующая команда вычислительного модуля использует результат текущей команды как операнд-источник, то всегда возникает такт простоя процессора (пузырь).

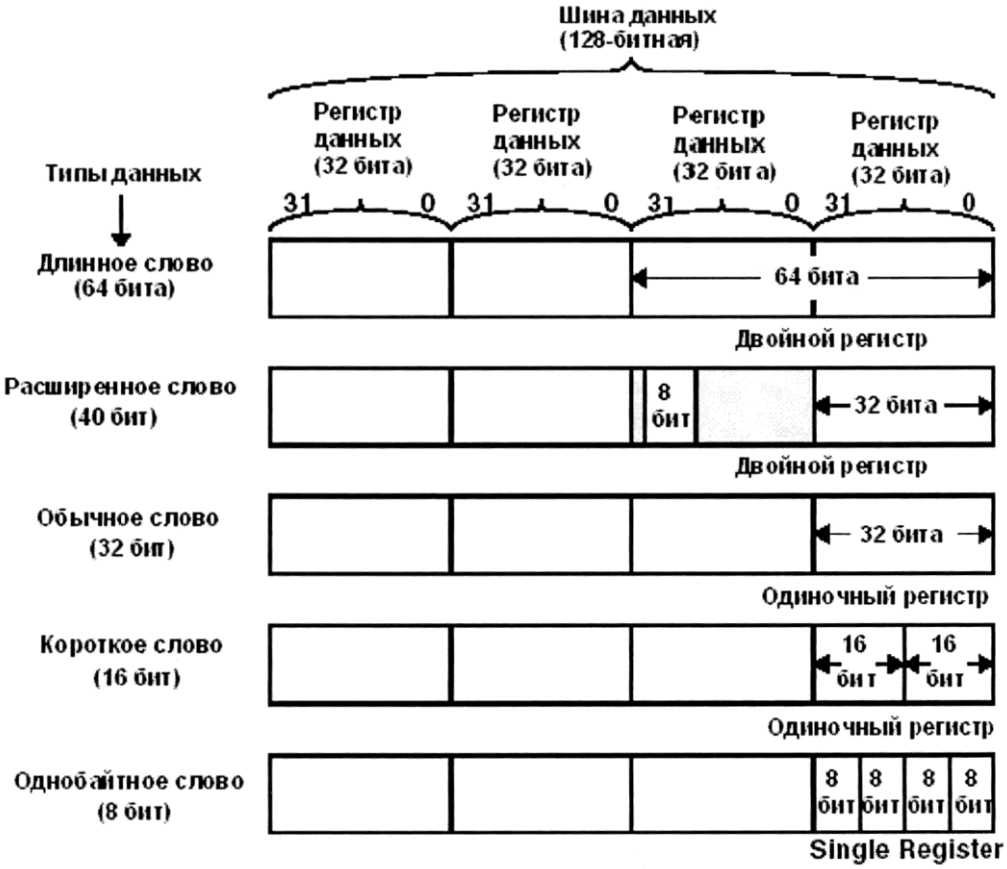


Рисунок 8 – Определения форматов слов

7.1.1.1 Арифметико-логическое устройство (ALU)

ALU выполняет арифметические операции над данными с фиксированной и с плавающей точкой, а также логические операции над данными с фиксированной точкой. Источником и приемником большинства операций ALU является регистровый файл вычислительного блока. Операнды ALU могут иметь размер 32, либо 64 бита, что соответствует одиночным операциям над типами Int, Float, Double, Long. Для типов меньшей разрядности (Short, Char) поддерживается векторная обработка. Количество элементов в векторах для типа Int равно двум, для типа Short – двум, либо четырем, а для типа Char – четырем, либо восьми. Т.к. модули X и Y могут работать как один сдвоенный модуль, длина обрабатываемых векторов может быть увеличена в 2 раза, т.е. за такт процессор способен обрабатывать 4 типа Int, 8 типов Short или 16 типов Char. Все это соответствует максимальной ширине шины памяти в 128 бит.

7.1.1.2 Коммуникационно-логическое устройство (CLU)

Каждый вычислительный модуль содержит вычислительное устройство специального назначения – коммуникационно-логическое устройство (CLU). Команды CLU предназначены для поддержки различных алгоритмов, используемых в коммуникационных приложениях, а именно алгоритмов:

- декодирования по Витерби;
- декодирования турбо кода;
- сжатия [частотной полосы сигнала] (despreading) для систем множественного доступа с кодовым разделением (CDMA);
- кросс-корреляций, используемых для отыскания пути.

Особенностью данного блока является то, что он имеет собственный 32 словный набор регистров общего назначения. Таким образом, команды CLU могут использовать операнды из РОН вычислительного модуля и из РОН собственного блока.

7.1.1.3 Умножитель с накоплением (Умножитель)

Блок умножителя выполняет умножение чисел с фиксированной или плавающей точкой, а также операции умножения с накоплением для чисел с фиксированной точкой. Умножитель поддерживает несколько типов данных для форматов с фиксированной и плавающей точкой. Для плавающей точки – это стандартные форматы одинарный и двойной точности, а также формат с расширенной точностью. Источником и приемником для большинства операций является регистровый файл вычислительного блока. Умножитель также поддерживает операции умножения над комплексными числами, представленными в виде пары 16-разрядных коротких слов, упакованных в одно 32-разрядное слово. Младшие значащие биты этого слова представляют собой действительную часть комплексного числа, а старшие значащие биты представляют мнимую часть. Команды умножения с накоплением используют в качестве приемника специальные регистры-аккумуляторы.

7.1.1.4 Побитное сдвиговое устройство (сдвиговое устройство)

Сдвиговое устройство выполняет логические и арифметические сдвиги, манипуляции с битами, заполнение полей и извлечение полей. Сдвиговое устройство выполняет операции с одним 64-битным, одним или двумя 32-битным, двумя или четырьмя 16-битными, а также с четырьмя или восемью 8- битными операндами с фиксированной точкой. Операции сдвигового устройства включают в себя:

- сдвиги и циклические сдвиги влево и вправо;
- операции манипуляции с битами, включая установку, очистку, переключение и проверку;
- операции манипуляции с битовыми полями, включая извлечение, заполнение с использованием регистра BFOTMP (внутреннего регистра сдвигового устройства);
- операции с однобитным FIFO для поддержки потоков битов с полями переменной длины;
- поддержка операций преобразования фиксированной и плавающей точки (таких, как извлечение порядка, определение числа ведущих единиц и нулей).

7.1.2 Целочисленное арифметико-логическое устройство (IALU)

Целочисленные ALU могут исполнять стандартные автономные операции ALU с регистровыми файлами IALU, операции загрузки/сохранения регистров и пересылки из регистра в регистр, а также формировать адреса при обмене данными между памятью и регистрами. Процессор имеет пару IALU (J-IALU и K-IALU), что позволяет формировать адреса одновременно для двух параллельно исполняемых транзакций по 128 бит. Наличие целочисленных ALU позволяет вычислительным операциям исполняться с максимальной эффективностью, поскольку вычислительные устройства могут быть заняты исключительно обработкой данных.

Каждое IALU имеет многопортовый 32-словный регистровый файл. Любая вычислительная операция IALU исполняется за один такт. При генерации адресов целочисленные ALU поддерживают пред-модификацию регистров без их обновления и пост-модификацию с обновлением. Кольцевые буферы реализованы аппаратно. Целочисленные ALU поддерживают следующие типы команд:

- обычные команды IALU;
- команды пересылки данных (из регистра в регистр);
- команды загрузки константы в регистр;
- команды загрузки/сохранения данных с модификацией значением регистра;
- команды загрузки/сохранения данных с модификацией непосредственным значением.

При косвенной адресации (команды с модификацией) значение одного из регистров регистрового файла может быть модифицировано значением другого регистра файла или непосредственным 8- или 32-битным значением, либо до (пред-модификация), либо после (пост-модификация) выполнения обращения к памяти. При адресации кольцевых буферов каждому из четырех первых регистров IALU может быть сопоставлено значение длины для выполнения автоматической адресации по ее модулю внутри такого буфера; границами кольцевых буферов могут служить любые адреса памяти. Кольцевые буферы делают возможной эффективную реализацию линий задержки и других структур данных, используемых обычно в цифровых фильтрах и преобразованиях Фурье. Переход адресного указателя через границу кольцевого буфера обрабатывается процессором автоматически, что позволяет упростить программную реализацию алгоритмов.

Целочисленные ALU поддерживают также бит-реверсивную адресацию, ориентированную на алгоритм БПФ. Бит-реверсивная адресация реализуется с использованием сложения с обратным переносом, подобного обычному сложению, но бит переноса, которого берется из старших битов и направляется в младшие.

IALU обеспечивают гибкость в пересылках данных обычными, двойными или квадрословами. Каждая команда может исполняться с производительностью одна на такт. Обычно задержки, обусловленные зависимостью между командами IALU, отсутствуют, но если таковые имеются, то их длительность может достигать трех тактов. Задержки могут появляться, если текущая команда производит загрузку данных из памяти в регистр IALU, либо пересылку регистра в регистр IALU, а следующая команда использует загружаемый регистр в качестве операнда-источника. В таком случае возникает три такта задержки до готовности операнда. Необходимо отметить, что IALU обладает практически такими же вычислительными возможностями по обработке данных типа Integer, как и вычислительный модуль. Наиболее проблемным местом является только отсутствие в IALU операции умножения.

7.1.3 Устройство управления потоком команд

Устройство управления (далее УУ) предоставляет адреса для выборки команд из памяти. УУ совместно с IALU позволяет вычислительным операциям выполняться с максимальной эффективностью. Эффективность ветвления обеспечивается устройством управления за счет использования буфера целевых адресов перехода (BTB), позволяющего сократить задержки на исполнение условных и безусловных переходов. УУ выполняет две задачи:

- декодирование извлеченных из памяти команд, т е. выделение команд из полей командной строки и отсылка их на соответствующие исполнительные устройства (вычислительные блоки, целочисленные ALU или УУ);
- управление ходом исполнения программы.

Команды УУ подразделяются на два типа:

- команды управления ходом исполнения программы, используемые для смены текущего адреса исполнения (перехода) и для условного исполнения индивидуальных команд;
- команды непосредственного расширения, используемые для расширения числовых полей, указываемых в непосредственных операндах для УУ и IALU.

Команды управления ходом исполнения программы подразделяются, в свою очередь, на две категории:

- явные переходы и вызовы, основанные на непосредственно указываемом в коде команды адресном операнде. Например, по команде 'if <cond> jump 100;' всегда осуществляется переход по адресу 100, если только вычисленное значение выражения <cond> есть «истина».
- Косвенные переходы, базирующиеся на адресе, предоставляемом регистром. Команды, используемые для указания на условное исполнение командной строки, представляют собой подкатегорию косвенных переходов. Например, переход по команде 'if <cond> cjmp;' является переходом по адресу, содержащемуся в регистре CJMP.

Примечание – Команды управления ходом исполнения должны располагаться в первом поле командной строки. Непосредственными расширениями могут обладать команды IALU и УУ (в последнем случае только команды управления ходом исполнения).

Эти команды не задаются программистом явно, а порождаются автоматически в случае, если размер непосредственных данных, использованных в командах, достаточно велик. Программист должен поместить команду, которая требует непосредственного расширения, в первое поле командной строки и оставить свободное поле в этой строке (задействовав только три поля) с тем, чтобы ассемблер мог поместить непосредственное расширение во второе поле командной строки. В одной командной строке может присутствовать только одно непосредственное расширение.

Процессор достигает своей высокой скорости исполнения за счет использования 9-ти стадий конвейера. Каждая стадия соответствует одному такту частоты процессора. Упрощенная структура конвейера приведена ниже (Рисунок 9).

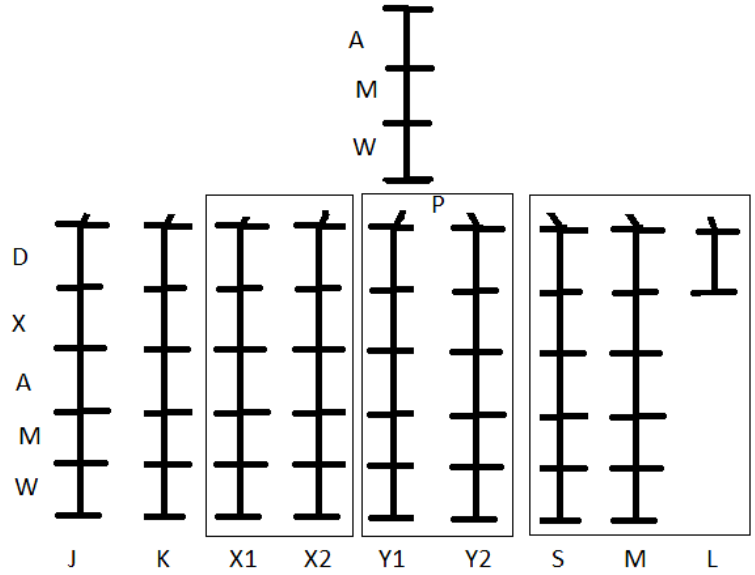


Рисунок 9 – Структура конвейера процессора

Горизонтальным линиям конвейера (см. Рисунок 9) соответствуют регистры для хранения информации между стадиями конвейера, а вертикальные линии соответствуют задержке обрабатывающей логики. Самая верхняя горизонтальная линия соответствует регистру IP – указателю на текущую команду устройства управления. IP содержит адрес, который отправляется в подсистему памяти для выборки команды.

Стадия адреса (стадия A) соответствует логике декодирования адреса, а также логике арбитража доступа к памяти.

Следующая стадия M соответствует чтению данных из накопителя в буфер памяти.

На стадии W происходит передача команды из буфера памяти в буфер выравнивания команд IAB. Шина команд имеет ширину 128 бит, что соответствует выборке 4-х команд за такт.

На стадии P одна линия конвейера чтения команд разветвляется на 9 линий (ветвей) конвейера исполнения команд.

Модули J и K имеют по одной линии конвейера, модули X и Y по две линии конвейера каждый, а модуль управления включает три ветви S, M, L. Ветвь L короткая, т.к. она обрабатывает команды-расширители константного поля, и вся обработка этой команды сводится к определению, какой линии конвейера соответствует расширение и передачи данных на эту линию. Для модулей обработки J, K, X, Y последняя горизонтальная линия конвейера соответствует РОН, в который записывается результат операции. Название стадий конвейера обработки данных D, X, A, M, W выбрано исходя из логики выполнения команды загрузки из памяти модулей J и K:

D – декодирование команды;
X – исполнение команды (вычисление адреса доступа);
A – передача адреса в подсистему памяти;
M – чтение данных из накопителя в буфер;
W – передача данных из буфера памяти в регистр ядра.

В зависимости от типа команды операции, выполняемые на данных стадиях конвейера, будут различны. Так для модулей X и Y наиболее важными являются стадии A, M, W:

- на стадии A происходит декодирование команды;

- на стадии M – первый такт исполнения;
- на стадии W – второй (заключительный такт исполнения).

Одной из задач УУ является отслеживание зависимостей по операндам во всех линиях конвейера и приостановка конвейера в случае, если операнды не готовы. Все линии конвейера продвигаются синхронно. При этом максимум шесть линий из 9-ти могут одновременно исполнять команды.

Для УУ основной линией конвейера является линия S, на которой происходит обработка команд переходов. Для такого глубоко конвейеризированного процессора скорость выполнения переходов является одним из ключевых факторов эффективности исполнения программы. Переходы могут быть безусловными и условными. Условные переходы могут быть двух типов: переходы, анализируемые на стадии X конвейера и на стадии W. Последние (на стадии W) соответствуют анализу флагов операций в модулях X и У. В случае выполнения перехода на стадии X процессор потеряет пять тактов, прежде чем следующая команда достигнет стадии X. Для ветвлений на стадии W таких тактов будет уже восемь. Для уменьшения потерь, связанных с переходами, УУ содержит буфер целевых адресов перехода (BTB) и имеет механизм статического прогнозирования переходов (выполняется пользователем). Если переход помечен как прогнозируемый, то он выполняется на стадии D линии S конвейера. Информация о переходе заносится в строку BTB. При первом выполнении данного перехода теряются четыре такта, однако при последующих его выполнениях потери будут равны нулю.

Также одной из задач УУ является обработка запросов прерываний.

Процессор имеет четыре внешних линии запроса прерываний общего назначения (nIRQ3-0), может генерировать прерывания от двух таймеров, каналов DMA, линк-портов, по арифметическим исключениям, а также генерировать векторные прерывания межпроцессорного взаимодействия и программные пользовательские прерывания. Есть возможность программно обеспечить обработку вложенных прерываний. Прерывания имеют низкую латентность и не могут сбросить с конвейера текущие команды уже начавшие свое исполнение. Прерывание векторизуется непосредственно по заданному пользователем адресу в регистровом файле таблицы (векторов) прерываний.

Запрос прерывания поступает в УУ и если прерывания разрешены, то стадия P конвейера прекращает поставлять команды на линии обработки команд, а УУ помещает в регистр IP адрес вектора прерывания (сбрасывая содержимое конвейера чтения команд) и запускает чтение нового потока.

7.1.4 Управление ядром процессора

Существует несколько режимов работы процессора, а также средств управления его работой. В данном подразделе описываются режимы работы, которые не управляются посредством выполнения команд. Эти средства управления включают управление:

- доменами синхронизации;
- режимом работы;
- режимом загрузки.

7.1.4.1 Домены синхронизации

Процессор имеет внешний вход тактовой частоты SCLK, которая используется для синхронизации работы внешней шины, а также в качестве опорной для

внутреннего умножителя частоты (PLL). Структура формирования частот процессора показана на рисунке 10.

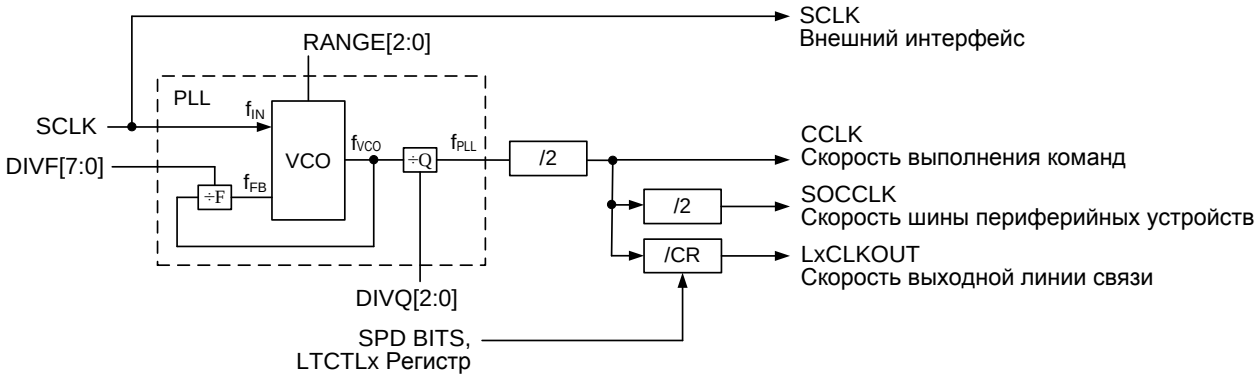


Рисунок 10 – Источники тактовой частоты

В процессоре можно выделить три основных домена синхронизации (см. Рисунок 11):

- ядро процессора (CCLK);
- SOC-шина периферийных устройств (SOCCLK);
- внешний интерфейс (SCLK).

Дополнительными доменами синхронизации являются четыре порта связи, каждый из которых может иметь индивидуальную скорость передачи.

Выводы DIVF[7:0], DIVQ[2:0]. RANGE[2:0] определяют умножение тактовой частоты SCLK (Таблица 6). Значения коэффициентов подбираются таким образом, чтобы частота f_{VCO} находилась в диапазоне 1200 – 2400 МГц, а частота f_{IN} была равна частоте f_{FB} и находилась в диапазоне, определяемом значением RANGE. Тактовая частота передатчика LVDS порта генерируется из CCLK через программируемый делитель (биты SPD регистра LTCTL). Встроенная SOC-шина работает при SOCCLK = ½ CCLK.

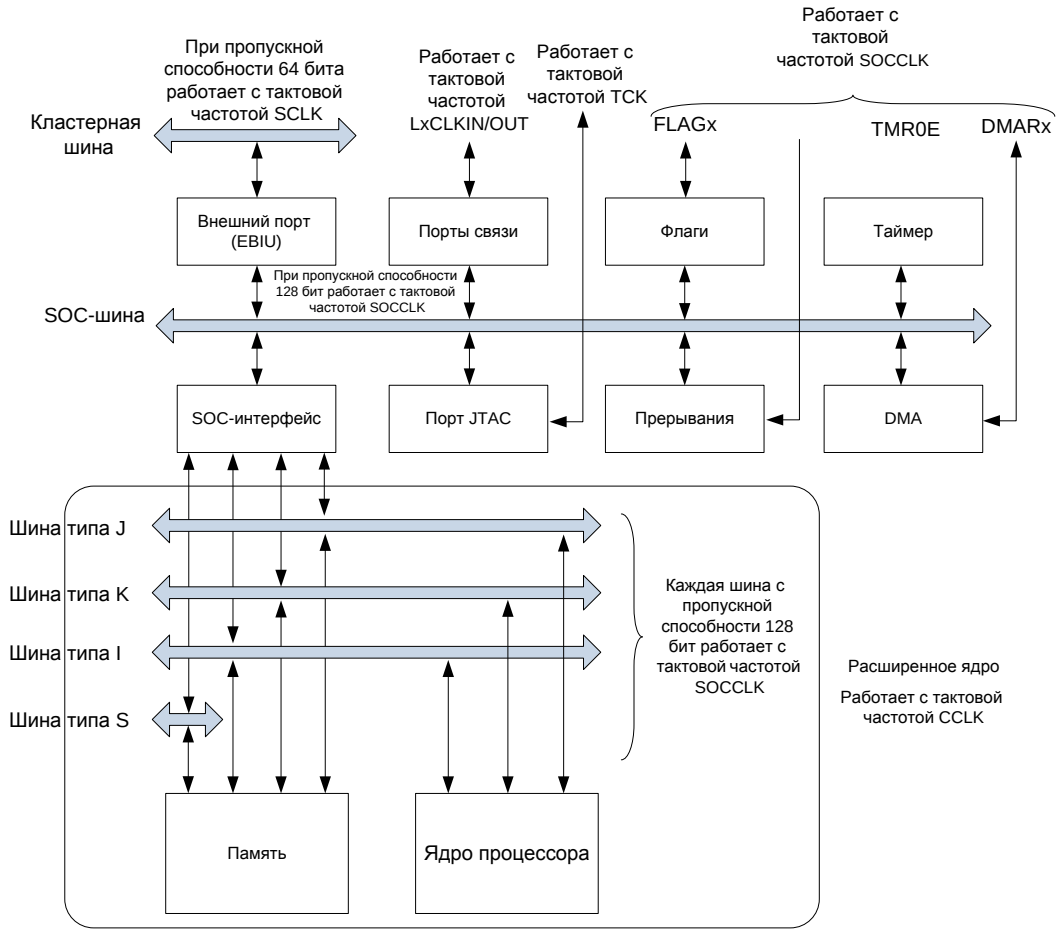


Рисунок 11 – Домены тактовой частоты

7.1.4.2 Режимы работы

Возможны три режима работы процессора:

- пользовательский;
- супервизор;
- эмулятор.

В режиме пользователя и супервизора все команды выполняются стандартным образом (т.е. путем чтения из памяти). Отличие режима пользователя от режима супервизора заключается только в ограничении доступа к отдельным компонентам системы для пользователя. Режим также влияет на приём и обработку исключений. Приоритеты режимов от наименьшего приоритета к наивысшему: пользовательский, супервизор, эмулятор.

Режим пользователя

Работа в пользовательском режиме используется для алгоритмов и управляющего кода, который не требует манипуляций с системными ресурсами. Многие системные ресурсы недоступны в пользовательском режиме. Если процессор пытается получить доступы к таким ресурсам, возникает исключение. Ядро ОС работает в режиме супервизора, но пользовательский код ограничен пользовательским режимом.

Следующие регистры доступны ядру программы в пользовательском режиме:

- Группы регистров от 0×00 до 0×09 – регистры вычислительных модулей;
- Группы регистров от 0×0C до 0×0F – регистры целочисленных АЛУ;
- Регистры УУ – CJMP, регистры счетчика циклов LC0 и LC1, и регистр статичного флага (SFREG).

Все другие регистры не могут быть записаны ядром программы в пользовательском режиме. Попытка записи в защищенный регистр вызовет исключение.

Режим супервизора

Режим супервизора позволяет программе получить доступ ко всем ресурсам процессора. Для работы процессора в режиме супервизора необходимо выполнение одного из условий:

- бит NMOD в SQCTL установлен;
- процедура прерывания запущена.

При аппаратном или программном сбросе процессор переходит в состояние ожидания прерывания. При получении прерывания он переходит в состояние исполнения программы в режиме супервизора.

Если бит NMOD в SQCTL сброшен, процессор входит в пользовательский режим после выхода из прерывания, если только прерывание не является вложенным.

Режим эмулятора

Режим эмулятора используется при управлении процессором отладочным инструментом через порт JTAG. Процессор входит в режим эмулятора при генерации специального исключения. Исключение эмулятора генерируется при одном из событий:

- EMUTRAP команда;
- срабатывание точки наблюдения для входа в эмулятор;
- передний фронт TMS при установленном бите TEME в EMUCTL.

Исключения эмулятора имеют высочайший приоритет среди исключений и прерываний.

Пока процессор находится в режиме эмулятора, единственным источником команд для него является регистр EMUIR. Регистр EMUIR загружается через порт тестового доступа JTAG. При вхождении в режим эмулятора, внешний контроллер JTAG должен быть включен.

После работы в режиме эмулятора, единственным способом выхода из данного режима является выполнение команды возврата из прерывания (RTI).

В режиме эмулятора, доступ к регистрам отладки (группа регистров 0×1B) может быть получен путем выполнения команд пересылки «регистр-регистр» или командой непосредственной загрузки данных. Эти регистры не могут быть загружены из памяти.

BTB, счетчик циклов, мониторы производительности и буфер трассировки неактивны в режим эмулятора. Кроме команды RTI (пр), все управляющие команды (переход, вызов, RDS, условные переходы, и т.д.) и команда IDLE не должны быть использованы в режиме эмулятора. Команда RTI может быть использована только безусловно.

7.1.4.3 Режимы старта

После сброса процессор возможны четыре опции загрузки кода программы для начала работы:

- загрузка из EPROM;
- загрузка хост-процессором;
- загрузка через порты связи;
- отсутствие загрузки.

Для выбора загрузки из EPROM или ее отсутствия системой используется вывод nBMS. После того, как режим загрузки выбран, процессор загружает (в ведущем режиме) или загружается (в подчиненном режиме) данными загрузки объемом 256 слов. Обычно данные загрузки содержат ядро загрузки для выбранного режима. После завершения первоначальной загрузки, ядро загрузки начинает работу, загружая программу разработчика в процессор и начиная выполнение этой программы.

Ниже приведен перечень внешних выводов процессора, значение которых во время сброса используется для конфигурирования процессора (Таблица 18).

Таблица 18 – Внешние конфигурационные выводы

Вывод	Значение во время сброса	Выполняемая функция
nBMS	0(низкий)	Выбирает режим загрузки из внешнего EPROM
	1(высокий)	Отключает возможность использования внешнего EPROM для стартовой загрузки.
nBM	0(низкий)	Запрещает прием прерываний по входам nIRQ3:0
	1(высокий)	Разрешает прием запросов прерываний по входам nIRQ3:0 по уровню сигнала (активный низкий).
TMR0E	0(низкий)	Выбор ширины шины каналов портов связи 1 бит
	1(высокий)	Выбор ширины шины каналов портов связи 4 бита
nBUSLOCK	0(низкий)	Однократная запись в регистры SYSCON,SDRCON
	1(высокий)	Многократная запись в регистры SYSCON,SDRCON

При нормальном функционировании выводы, приведенные в таблице 18, являются выходами. Во время сброса выход отключен, и вывод работает как вход, позволяя записать конфигурационную информацию.

Выполнение операции загрузки процессора

После сброса ядро процессора находится в состоянии ожидания прерывания. Если поступит запрос прерывания, процессор начнет исполнение программы согласно вектору прерывания.

В случае загрузки из EPROM (вывод nBMS равен нулю во время сброса), канал 0 DMA выполняет загрузку кода из внешнего EPROM во внутреннюю память и посылает запрос прерывания с адрес-вектором равным 0.

Хост-процессор может загрузить программу во внутреннюю память процессора, а затем, используя векторное прерывание, запустить процессор на исполнение кода.

Внешнее устройство может загрузить 256 слов кода через порт связи, после чего соответствующий приемнику порта связи канал DMA сформирует запрос прерывания к процессору с адрес-вектором 0.

Если события, описанные выше, не происходят, то может возникнуть внешнее прерывание по одному из входов nIRQ3-0. Каждому входу запроса прерывания соответствует вектор во внешней или внутренней памяти. Процессор может начать исполнение кода из внешней памяти.

Загрузка EPROM

Загрузка процессора из EPROM является загрузочным режимом, в котором процессор начинает и контролирует процесс пересылки данных.

Для того чтобы сконфигурировать процессор для загрузки из EEPROM необходимо чтобы во время сброса на внешнем выводе nBMS был низкий уровень.

По окончании сброса процессор защелкивает состояние внешнего вывода nBMS во внутреннем флаге. Это значение доступно в регистре SYSTAT. Если процессор сконфигурирован под загрузку из EPROM, nBMS является активным во время последовательности загрузки и должен подключаться к сигналу выбора кристалла EPROM.

При выборе режима загрузки EPROM процессор инициализирует канал 0 внешнего порта DMA для передачи 256-ти 32-разрядных слов кода из загрузочного EPROM в ячейки 0x00-0xFF блока внутренней памяти 0 процессора. Соответствующий вектор прерывания (для DMA канала 0) инициализируется по адресу 0x00 во внутренней памяти. После завершения работы DMA, процессор начинает выполнение программы из адреса ячейки 0x00.

Эти 256 слов кода действуют в качестве загрузчика для инициализации остальной памяти процессора.

Загрузка хоста

Загрузка процессора хост-процессором является режимом, в котором процессор ожидает пока внешнее устройство начнет загрузку кода в его внутреннюю память. Любое ведущее устройство на кластерной шине может загружать процессор через запись в его внутреннюю память или посредством autoDMA.

Для того, чтобы перевести процессор в режим загрузки хоста, необходимо поместить нагрузочный повышающий резистор между nBMS и VDD_IO. Тогда во время сброса значение nBMS будет равно 1, что укажет на отсутствие загрузки из EPROM.

Когда выбран режим загрузки хоста или порта связи, процессор переходит в нерабочее состояние после сброса, ожидая пока хост-процессор или порт связи осуществит загрузку. Загрузка хоста может использовать регистры AUTODMA процессора (оба канала), из которых оба инициализированы для передачи 256 слов кода в блок 0 ячейки 0x00-0xFF внутренней памяти процессора. Соответствующий вектор прерывания инициализируется по адресу 0x00 во внутренней памяти. Таким образом, после завершения DMA, процессор начнет выполнение программы с адреса 0. Эти 256 слов кода действуют в качестве загрузчика для инициализации остальной памяти процессора.

Загрузка порта связи

Загрузка процессора из порта связи является загрузочным режимом, в котором процессор ожидает внешнее устройство, подключенное к порту связи, чтобы принять и поместить во внутреннюю память загрузочную программу.

Все четыре принимающих канала DMA портов связи инициализируются после сброса для передачи блока 256 слов в адреса внутренней памяти с 0 по 255 и для вызова прерывания после приема блока (аналогично каналу 0 DMA). Соответствующие прерывания этих каналов DMA устанавливаются по нулевому адресу. Эти 256 слов кода действуют в качестве загрузчика для инициализации остальной памяти процессора.

При выборе загрузки посредством порта связи имеет значение состояния вывода TMR0E во время сброса. Этот вывод позволяет выбрать ширину шины порта связи: 1 или 4 бита.

Отсутствие загрузки

Запуск процессора в режиме «Без загрузки» является режимом, в котором процессор начинает исполнение программы с фиксированного адреса во внешней или внутренней памяти. Выбор стартового адреса осуществляется посредством подачи запроса прерывания по одной из линий nIRQ3-0.

Для конфигурации процессора в режиме без загрузки необходимо подключение вывода nBMS к высокому уровню посредством резистора. Также

необходимо подключить высокий уровень к выводу nBM. Именно значение 1 на входе nBM во время сброса и разрешает анализ запросов прерывания по входам nIRQ3:0. Запрос прерывания осуществляется активный низким уровнем сигнала.

Каждый вход запроса прерывания имеет соответствующий ему вектор. Значение векторов прерывания приведено ниже (Таблица 19). Как видно из таблицы, процессор может выполнить старт с адреса внешней памяти банков памяти MS0 или MS1, а также из хост-пространства. Возможен также старт из внутренней памяти с адреса 0.

Таблица 19 – Векторы прерывания nIRQ3–0 после сброса

Прерывание	Адрес
nIRQ0	0x30000000 (MS0)
nIRQ1	0x38000000 (MS1)
nIRQ2	0x80000000 (nMSH)
nIRQ3	0x00000000 (внутренняя память)

Необходимо отметить, что выбор загрузки из EPROM (nBMS равен 0) не означает невозможность загрузки посредством портов связи или хостом, или генерации прерывания посредством nIRQ3:0. Загрузка из EPROM не запрещает других действий. Поэтому нужно очень внимательно относиться к процедуре сброса и избегать неоднозначности в вариантах начального старта процессора.

7.2 Контроллер прерываний

У процессора есть четыре внешних прерывания общего назначения nIRQ3-0, а также внутренние прерывания от двух таймеров, DMA каналов, портов связи, арифметических исключений, мультипроцессорных векторных прерываний и программных прерываний, задаваемых пользователем. Прерывания могут быть вложенными. При обработке прерывания выполняется переход по вектору прерывания, значение которого хранится в таблице векторов прерываний.

Процессор поддерживает обработку прерываний, являющихся индикаторами различных внешних или внутренних событий. Прерывания делятся на программные и аппаратные.

Большая часть запросов прерываний процессора является специализированной. Четыре внешних входа и два регистра прерываний поддерживают прерывания общего назначения. Все другие источники прерываний вызываются событиями или специализированной аппаратной частью.

Для каждого запроса прерывания существует:

- регистр в таблице векторов прерываний (группы регистров 0x38 и 0x39);
- бит в регистре защелки прерывания ILAT;
- бит в регистре маски прерывания IMASK;
- бит в регистре приоритета прерывания PMASK.

Номера битов в регистрах одинаковы и соответствуют номеру прерывания. Этот же номер имеет и регистр вектора прерывания в таблице.

Программные прерывания вызываются специальными командами или определенными ситуациями, которые возникают при выполнении команд.

Аппаратные прерывания вызываются различными событиями, которые происходят в периферийных устройствах, либо входным сигналом на выводах внешних прерываний (nIRQ3–0).

Прерывание может быть вызвано аппаратными ошибками в процессоре, такими как:

- ошибка в работе контроллера DMA;
- доступ к каналу AutoDMA, когда он выключен;
- чтение мультипроцессоров собственной области памяти;
- ошибка порта связи;
- широковещательное чтение из внешней памяти.

Прерывания могут быть чувствительны к фронту или уровню сигнала:

- Чувствительные к фронту прерывания фиксируются при их появлении установкой определенного флага в регистре ILAT. Флаги хранятся до тех пор, пока не будут обслужены или сброшены по команде. Если флаг не сбрасывается во время обработки, то новое событие не обнаруживается и запрос игнорируется.
- Чувствительные к уровню прерывания должны поддерживать активный уровень запроса до начала обработки, в противном случае они не видны. Если запрос продолжается после обработки, это считается новым запросом прерывания.

Аппаратные прерывания (nIRQ3–0) могут быть сконфигурированы любым из вариантов в соответствии с регистром INTCTL.

Чувствительные к уровню прерывания обычно появляются как некоторый активный результат в определенном доступном регистре и аннулируются чтением этого регистра или записью в него неактивного значения. Чувствительные к фронту прерывания инициируются событием (например, срабатыванием таймера).

Каждое из прерываний процессора имеет регистр вектора прерываний, который является адресом процедуры, обслуживающей данное прерывание. Весь регистровый файл называется Таблица Вектора Прерываний (IVT). Каждый регистр таблицы 32-битный для того, чтобы подключать процедуры обработки прерываний из внешней или внутренней памяти. Для реализации процедуры начального старта некоторые регистры векторов прерываний при сбросе инициализируются конкретными адресами. При загрузке из EPROM или загрузке из портов связи, DMA инициализируется для загрузки данных, начиная с адреса памяти 0x0. Соответственно после выполнения загрузки вырабатывается запрос прерывания, которому в таблице векторов прерываний должен соответствовать вектор со значением 0.

7.2.1 Операции с прерываниями

Операции с прерываниями включают установку запросов прерываний, обработку прерываний или исключений и возврат из прерывания. Контроллер прерываний включает следующий регистры установки и управления:

- регистр управления прерываниями (INTCTL);
- регистры маски прерываний (IMASK) – IMASKH и IMASKL;
- регистры макси приоритета (PMASK) – PMASKH и PMASKL;
- Регистры защелки прерываний (ILAT) – ILATH и ILATL.

Все эти регистры подробно были описаны в главе «Группа регистров управления контроллером прерываний». Наличие запроса прерывания отражается активным (равен 1) битом в регистре ILAT. Если соответствующий ему бит в регистре маски IMASK установлен, то запрос прерывания поступает на дальнейшую обработку. Регистр приоритетов PMASK хранит запросы прерываний, которые процессор принял к обработке. Приоритет прерывания – это номер бита в регистре PMASK (от 0 до 63). Регистр приоритетов позволяет блокировать низкоприоритетные запросы прерываний до момента окончания обработки более высокоприоритетного. Если прерывание разрешено и его приоритет выше текущего обрабатываемого прерывания (или нет обработки прерываний в данный момент) тогда контроллер прерываний формирует активный уровень запроса прерывания к процессору, сопровождая его номером прерывания и адрес-вектором процедуры обслуживания данного прерывания. Если бит GIE (бит глобального разрешения прерываний) в регистре SQCTL установлен, то процессор прерывает текущий ход программы и переходит к выполнению процедуры обработки прерывания. Как только прерывание начинает обслуживаться, в регистре PMASK устанавливается в 1 соответствующий данному прерыванию бит. Данная установка запрещает прохождение запросов прерываний с приоритетом ниже обрабатываемого. При этом процедура обработки прерывания может разрешить обработку более высокоприоритетных прерываний. Как только обработка прерывания заканчивается, соответствующий ему бит в PMASK сбрасывается. Это происходит при выполнении команд RTI или RDS. При этом если команда RTI означает завершение процедуры обработки, то команда RDS означает, что обработка текущего прерывания может быть прервана другим прерыванием, т.е. возможны вложенные прерывания.

Активный запрос прерывания вызывает установку соответствующего бита в регистре ILAT. Существует возможность программно установить бит в этом регистре, т.е. приложение может эмулировать прерывание посредством записи в

регистр ILAT. Запись по адресам ILATSTL или ILATSTH обновляет регистр ILAT значением логического «ИЛИ» старого содержимого и нового записываемого. В результате, каждый установленный бит в записанных данных устанавливает соответствующий бит в регистре ILAT. Установка бита прерывания вызывает в процессоре предположение о возникновении соответствующего прерывания. Биты прерывания могут быть также сброшены записью в адреса сброса ILATCLL или ILATCLH. Такая запись применяет оператор AND к записываемым данным и старым данным регистра ILAT. В этом случае нулевой бит во входных данных сбрасывает соответствующий бит в ILAT, тогда как установленный бит остается неизменным. Таким образом, приложение может сбросить ждущее прерывание до его исполнения.

Операции сброса или установки бит в ILAT должны выполняться при следующих условиях:

- Незадействованные (резервные) биты не могут быть установлены. При записи по адресам установки или сброса, резервные биты должны быть нулями.
- Биты программного исключения и эмуляторного исключения не могут быть установлены через ILATST. Для того чтобы вызвать такие исключения следует использовать команды TRAP и EMUTRAP.
- Прерывания, которые запускаются уровнем, не должны изменяться.
- Записи в ILATL и ILATH не должны предприниматься (т.е. только сброс или установка).
- Прерывание должно быть сброшено в то время, когда оно запрещено, иначе оно может быть обслужено до того, как будет сброшено.
- Адреса установки и сброса работают с одинарными словами.

7.2.2 Программа обработки аппаратного прерывания

При обработке прерываний часть действий выполняются контроллером прерываний, другая часть – процессором. Рассмотрим последовательность обработки запроса прерывания на примере прерывания от таймера 0. Действия контроллера прерываний будут следующие:

- Если таймер включен, и его счетчик имеет в текущем такте SOCCLK значение 1, то вырабатывается активный уровень запроса прерывания. Активный запрос существует только один такт, чего достаточно, чтобы в следующем такте биты 2 и 52 регистра ILAT установились в 1. Бит 2 соответствует запросу прерывания от таймера 0 с низким уровнем приоритета, а бит 52 с высоким уровнем.
- Предположим, что мы ожидаем прерывание с высоким уровнем приоритета. Для этого бит 52 в регистре IMASK должен быть установлен в 1. Логическое «И» между битом запроса прерывания и битом маски равно 1 и это говорит о том, что прерывание может быть принято к обслуживанию.
- Далее выполняется анализ регистра PMASK. Если биты регистра с 52 по 63-й равны нулю, то в обработке нет прерываний равного или более высокого приоритета, чем ожидаемое. В противном случае необходимо дождаться окончания обслуживания более высокоприоритетного прерывания.
- Выполнения описанных выше условий достаточно, чтобы контроллер прерываний сформировал запрос прерывания к процессору. Кроме активного уровня запроса на линии, контроллер передает в процессор номер прерывания (52), а также извлекает из таблицы векторов прерываний

значение регистра с номером 52 и передает его в процессор. Это значение есть адрес процедуры обработки нашего прерывания.

Когда запрос прерывания поступает в процессор (в устройство управления) первым действием процессора является проверка бита глобального разрешения прерываний GIE регистра управления SQCTL. Если бит равен 1 – прерывания разрешены и процессор выполняет следующие действия:

1 Запоминает номер и адрес-вектор обработки пришедшего прерывания.
2 Выполняет сброс всего конвейера чтения команд вместе с буфером IAB.
3 Запускает чтение команд по адрес-вектору прерывания.
4 Конвейер обработки команд продолжает и заканчивает обработку всех команд, которые находились на конвейере в момент сброса конвейера чтения команд.
5 Когда последняя команда конвейера обработки завершает свое выполнение, процессор повторно проверяет бит разрешения прерываний GIE и, если прерывания по-прежнему разрешены, первая команда процедуры обработки прерывания поступает из буфера команд на конвейер обработки.
6 При достижении первой командой последней стадии конвейера (W) происходит формирование сигнала о том, что данное прерывание взято на обработку. Этот факт приводит к установке бита 52 регистра PMASK и сброс бита 52 в регистре ILAT. Установленный бит в регистре PMASK запрещает все запросы прерывания уровнем ниже 53 (т.е. новое прерывание от таймера 0 также запрещено). Вместе с этим процессор запоминает в специальном регистре RETI адрес следующей команды, которая должна быть выполнена при выходе из прерывания. Дополнительно ко всему процессор устанавливает внутренний флаг EXE_ISR, который запрещает все прерывания. Флаг EXE_ISR указывает на то, что процессор находится в состоянии обработки прерывания. Флаг доступен для чтения как 20-й бит регистра SQSTAT.
7 Переход на обработку прерывания вызывает установку режима супервизора (если до этого был режим пользователя).
8 Сброс бита 52 в регистре ILAT означает, что бит стал доступен для приема нового запроса прерывания от таймера 0.
9 Далее процессор выполняет все команды, которые соответствуют процедуре обслуживания данного прерывания. Для выхода из процедуры обработки используется команда RTI.
10 Выполнение команды RTI приводит к сбросу бита 52 в регистре PMASK, что разрешает прохождение всех прерываний, которые ранее блокировались. Также данная команда вызывает сброс флага EXE_ISR, что снимает блокировку прерываний и включает функции бита GIE. Процессор переходит к выполнению команд по адресу из регистра RETI.

Процедура, описанная выше, соответствует типичной ситуации обработки при отсутствии каких-либо аномалий. Однако, возможны различные ситуации:

- Если перед повторной проверкой разрешения прерываний (после полного освобождения конвейера обработки) оказывается, что прерывания запрещены (это может быть сделано командой сбрасывающей бит GIE), тогда все действия по обработке прерывания отменяются и конвейер перезапускается с адреса, следующего за адресом последней выполненной команды.
- Если запрос прерывания из контроллера пришел в процессор, но бит GIE равен нулю, прерывание будет ждать. В это время в контроллере

прерываний может появиться запрос с более высоким приоритетом, чем наше. Контроллер прерываний обработает его и передаст в процессор номер и адрес-вектор нового прерывания. Запрос от таймера 0, в этом случае, будет отложен до момента окончания обработки нового прерывания.

- Возможна ситуация, когда запрос пришел из контроллера, но в это же время процессор выполнял сброс бита запроса прерывания в регистре ILAT. Эта ситуация приведет к тому что запрос из контроллера прерываний будет стабильным в течение одного или нескольких тактов. Тем не менее, запрос может быть обработан. Если снятие запроса произошло после или во время сброса конвейера чтения команд, то это означает, что запрос будет обслужен. Для надежного выполнения подобных операций необходимо предварительно запрещать прерывания битом GIE, а затем производить сброс в контроллере прерываний.

Процедура обработки прерывания начинается с сохранения контекста процессора, т.е. сохранения на стеке всех регистров процессора, которые будут использованы программой обработки прерывания. Это необходимо для того, чтобы после обработки прерывания восстановить прежнее состояние процессора и дать возможность прерванной программе корректно возобновить работу. Факт обработки прерывания отражается в установке флага EXE_ISR и это запрещает все прерывание, в том числе и более высокоприоритетные, чем ожидаемое. Если время обработки прерывания значительно, и это может повредить обработке запроса более приоритетного прерывания, программа может разрешить обслуживание более высокоприоритетных прерываний. Для этого необходимо обязательно сохранить на стеке содержимое регистра RETI, который хранит адрес возврата из прерывания. Для сохранения содержимого RETI используется специальный псевдоним этого регистра – RETIB.

Выполнение команды **j0 = RETIB;;** вызовет копирование регистра RETI в регистр j0 и сброс флага EXE_ISR. Сам регистр j0 должен быть предварительно сохранен на стеке. После выполнения указанной команды процессор имеет возможность обслуживать более высокоприоритетные запросы, чем запрос 52. Описываемая ситуация вносит коррективы в последовательность действий при выходе из обработки прерывания. Перед выходом содержимое регистра возврата RETI неопределенно (новые прерывания могли изменить его). Поэтому адрес возврата из регистра j0 (или иного места, где он хранился) необходимо вернуть обратно в RETI. Однако выполнение обычной пересылки RETI = j0;; будет некорректным, т.к. после этой команды возможно возникновение прерывания, и адрес возврата может быть утерян. В данном случае необходимо использовать регистр-псевдоним RETIB. Пересылка RETIB = j0;; не только пересылает адрес возврата в RETI, но и устанавливает флаг EXE_ISR. При такой пересылке прерывания запрещены, и весь контекст можно безопасно восстановить, после чего выполнить финишную команду RTI.

В выше описанной процедуре разрешения прерываний с более высоким приоритетом, чем текущее, разрешаются прерывания с приоритетом от 53 и выше, но прерывание с приоритетом 52 запрещено. Это означает, что обрабатывая запрос от таймера 0, существует возможность принимать новый запрос прерывания от таймера, т.к. бит 52 в регистре ILAT был сброшен и теперь имеет возможность зафиксировать новое событие, но нет возможности его обслуживать т.к. бит 52 в регистре PMASK установлен. Также нет возможности обслуживать запросы с приоритетом ниже 52. При необходимости разрешить обслуживание запросов с приоритетом ниже 52 используется команда RDS.

Выполнение команды j0 = RETIB;; разрешает только прерывания выше 52-го, т.к. бит 52 в регистре PMASK установлен. Выполнение команды RDS вызывает следующие действия процессора:

- сбрасывается в ноль текущий самый высокоприоритетный установленный бит в регистре PMASK;
- сбрасывается флаг EXE_ISR;
- выполняется переход в режим работы, который предшествовал обработке прерывания;
- выполняется перезапуск конвейера в новых условиях.

Перед выполнением RDS необходимо сохранить адрес возврата. Если переход в прежний режим нежелателен (например, возврат в режим пользователя с его ограничениями), то необходимо предварительно сохранить регистр SQCTL и затем установить в нем бит режима супервизора. В этом случае после команды RDS останется режим супервизора.

Действия, описанные выше, позволяют реализовать вложенные прерывания. Выход из подобных прерываний выполняется с использованием регистра RETIB.

Описанные действия относятся к обработке любого аппаратного прерывания, а не только 52-го.

Выполнение команды возврата из прерывания RTI имеет еще ряд особенностей. Данная команда является командой перехода и может выполняться с опцией предсказания или без. Использование опции предсказания позволяет ускорить выход из прерывания на несколько тактов процессора. Однако необходимо помнить, что предсказание будет полезным, если содержимое регистра RETI корректно в момент выполнения предсказания, т.е. в момент передачи в память адреса линии команд содержащей RTI. Если регистр RETI загружается непосредственно перед командой RTI или это делается в нескольких тактах до RTI, то лучше использовать команду RTI без предсказания перехода т.к. все равно предсказание будет ошибочным и все выполненные действия будут отменены.

7.2.3 Особенности сохранения контекста процессора

Во время выполнения программы процессор может находиться в различных режимах. Можно выделить два основных режима работы:

- режим супервизора (ядра или системный);
- режим пользователя.

Режим супервизора может иметь несколько подрежимов (или видов):

- режим выполнения задачи системы;
- режим обработки прерывания;
- режим обработки программной исключительной ситуации;
- режим отладки.

Произведя анализ регистра состояния SQSTAT, можно определить тип подрежима. Для каждого из подрежимов процессор имеет соответствующие им регистры связи:

- CJMP – регистр связи для выполнения вызовов процедур при нормальном ходе программы как в режиме супервизора, так и в режиме пользователя.
- RETI – регистр связи при возникновении прерывания.
- RETS – регистр связи при возникновении программной исключительной ситуации.
- DBGE – регистр связи при вызове эмулятора.

Регистры связи RETI, RETS, DBGE используются только при входе в исключительную ситуацию и при выходе из нее. При нормальном ходе программы для связи используется регистр CJMP. В процессоре используется один регистр-указатель стека, общий для всех режимов. Однако в качестве указателя стека может быть использован любой от 0-го до 30-го регистр IALU, но по умолчанию компилятор использует в качестве указателя стека регистр J27/K27. На аппаратном уровне процессор поддерживает раздельные регистры J27/K27 для режима супервизора (KSP или SP) и режима пользователя (USP).

Режим работы с двумя указателями стеков возможен только при установленном бите SQCTL[14]. Если бит SQCTL[14] равен нулю, процессор работает с общим для всех режимов указателем стека. В режиме ядра регистр j27/k27 соответствует KSP, а также имеется возможность чтения и записи указателя USP через резервный регистр номер 27 группы регистров базы и длины циклической адресации модулей IALU. В режиме пользователя j27/k27 доступен только как USP. Выбор указателя стека осуществляется следующим образом:

- Если процессор находится в режиме обработки прерывания – используется KSP.
- Если процессор находится в режиме обработки исключительной ситуации – используется KSP.
- Если процессор находится в режиме эмулятора – используется KSP.
- Если процессор исполняет прикладную задачу и бит SQCTL[9] равен нулю – используется USP.
- Если процессор исполняет прикладную задачу и бит SQCTL[9] равен единице – выбор указателя определяется битом SQCTL[7]: если бит равен нулю – KSP, иначе – USP.

Таким образом, в режиме супервизора возможна работа как с указателем ядра, так и с указателем пользователя.

Основные трудности в выборе указателя могут возникнуть в момент смены режима, так как из-за наличия конвейера фазы чтения и записи регистра разнесены во времени и различным моментам времени могут соответствовать различные режимы.

Процесс смены режима посредством модификации бита SQCTL[9] наиболее безопасный, так как каждый раз после записи регистра SQCTL происходит перезапуск конвейера. Это гарантирует, что команды, следующие за сменой режима, извлекаются и выполняются в требуемом режиме.

Смена режима посредством программной исключительной ситуации (установка SWI_mode) безопасна во время входа в обработчик, так как возникновение исключительной ситуации приводит к сбросу конвейера, его очистке и последующему переходу на программу обработки. Безопасный выход из обработчика должен быть реализован соответствующим образом. Выход из обработчика (сброс SWI_mode) осуществляется посредством выполнения команды RTI. Данная команда должна исполняться с опцией RTI (NP). Это гарантирует сброс конвейера после смены режима и последующее корректное использование указателя стека (а также работу устройства защиты).

Смена режима посредством аппаратного прерывания (установка ISR_mode) безопасна во время входа в обработчик. Это гарантируется аппаратурой процессора. Признак прерывания движется по конвейеру одновременно с командами обработчика и осуществляет выбор корректного указателя стека, если первые команды обработчика используют указатель. Безопасный выход из прерывания (сброс ISR_mode) должен быть реализован соответствующим образом. Выход осуществляется посредством выполнения команды RTI. Данная команда должна исполняться с опцией RTI (NP). Это гарантирует сброс конвейера после

смены режима и последующее корректное использование указателя стека (а также работу устройства защиты).

Сброс режима обработки аппаратного прерывания или программного прерывания может быть выполнен не только командой RTI, но и командой RDS. Опасность использования команды RDS при работе с разными указателями стека, а также с устройством защиты, связано с тем, что выполнение команды RDS приводит к возврату в режим, предшествовавший прерыванию. При работе в режиме раздельных указателей стека после выполнения команды RDS будет выполнен перезапуск конвейера. До выполнения команды RDS будет доступен указатель стека супервизора, после RDS – это может быть указатель стека супервизора или пользователя.

Поскольку возможно использование сразу двух указателей стека, сохранение контекста может быть ускорено, если стеки размещаются в различных банках памяти. Это позволяет за такт сохранить восемь регистров процессора. Если использование двух стеков неудобно, можно использовать факт разбиения каждого банка внутренней памяти на две половины, одна из которых – четные квадрослова, вторая – нечетные. В этом случае регистр К27 может хранить копию указателя J27, и при сохранении контекста возможна работа сразу с парой квадрослов. Это позволяет сохранять и восстанавливать сразу восемь регистров.

7.2.4 Обработка программных прерываний (исключений)

Исключения – программные прерывания, вызываемые исполняемым программным кодом. При этом могут быть специальные команды (TRAP, EMUTRAP) или ситуации которые возникают при выполнении команд. Среди таких ситуаций могут быть аномалии при вычислении с плавающей запятой, доступ к длинным словам по невыровненному адресу и другие. Обработка программного прерывания отличается от аппаратного:

- Бит GIE не имеет значения при обработке исключений. Для программных прерываний используется специальный бит глобального разрешения SWIE.
- Часть программных прерываний имеет индивидуальные биты разрешения. Это касается исключений, которые возникают в вычислительных модулях X и У. Другие исключения не имеют специальных бит для разрешения, кроме глобального.
- Программное прерывание происходит сразу за выполнением команды инициировавшей это событие.
- Для хранения адреса возврата используется специальный регистр RETS, а не RETI.
- Возникновение исключение приводит к установке 62-го бита регистра PMASK, что означает блокировку всех аппаратных прерываний.
- В качестве адреса-вектора процедуры обработки исключений используется содержимое регистра IVSW.
- Возникновение исключения приводит к установке специального бита EXE_SWI. Бит виден в 21-м разряде регистра SQSTAT. Бит может быть установлен только при возникновении исключительной ситуации. Бит может быть сброшен только при выходе из обработки исключения либо командой RDS. Никаких других механизмов установки или сброса данного бита нет.
- Установка бита EXE_SWI приводит к переходу в режим супервизора (если ранее был режим пользователя). Также это приводит к запрещению всех программных и аппаратных прерываний.
- Если при обработке исключительной ситуации возникло новое исключение, то оно не будет обработано.

Последовательность действий процессора, выполняемая при возникновении исключительной ситуации:

- В поле SQSTAT[11:8] процессор сохраняет код исключительной ситуации. Если ситуация была вызвана командой TRAP, то в поле SQSTAT[7:3] дополнительно сохраняется параметр данной команды. Все это позволяет идентифицировать источник исключения.
- Сбрасывается весь конвейер процессора и начинается чтение команд по адресу из регистра IVSW.
- В регистре RETS сохраняется адрес возврата, а именно адрес первой команды линии следуемой за линией команд вызвавшей исключение.
- Бит PMASK[62] устанавливается в 1, запрещая все аппаратные прерывания.
- Устанавливается внутренний флаг EXE_SWI, переводя процессор в режим супервизора и запрещая все аппаратные и программные прерывания.
- Далее процессор выполняет все команды процедуры обработки исключительной ситуации.

- Перед выходом из прерывания программа копирует содержимое регистра RETS в регистр RETI с помощью команды RETI = RETS;;. Регистр RETIB в данном случае не должен использоваться.
- Выход из обработчика осуществляется с помощью команды RTI.

Поскольку при возврате из программного прерывания используется регистр RETI, то последовательность выхода может быть такой:

[j31+temporary address] = RETI;;
RETI = RETS;;
RTI (NP); RETI = [j31+temporary address];;

Таким образом, видно, что при возникновении программного прерывания создаются очень строгие ограничения в поведении процессора. Также, как и при аппаратном. Однако, возникновение аппаратного прерывания не запрещает обработку программного прерывания, если оно возникнет при выполнении команд аппаратного обработчика. При обработке программного прерывания можно разрешить аппаратные прерывания. Для этого необходимо выполнить команду RDS, что приведет к следующему:

- Произойдет сброс бита PMASK[62], т.е. все аппаратные прерывания в контроллере прерываний станут доступны.
- Произойдет сброс бита EXE_SWI. Аппаратные прерывания станут управляться битом GIE, а программные SWIE. Произойдет возврат к предыдущему режиму работы.

При использовании команды RDS необходимо заранее принять меры, чтобы последующие события не повлияли на корректное исполнение программного обработчика.

Программное прерывание может быть вложенным в аппаратное. Этот факт можно отследить, если оба биты EXE_ISR и EXE_SWI равны 1. В подобной ситуации команды RTI или RDS оказывают влияние только на бит EXE_SWI, и их исполнение означает переход на уровень аппаратного обработчика.

7.2.5 Обработка прерываний эмулятора

Прерывания эмулятора – это события, которые приводят к переходу процессора в режим эмуляции, когда все его действия управляются посредством команд через интерфейс JTAG. Подобное прерывание может быть вызвано:

- командой EMUTRAP;
- срабатыванием точки наблюдения;
- аппаратным запросом из JTAG интерфейса.

Обработка данного типа прерываний имеет следующие особенности:

- Специальный бит глобального разрешения DBGEN. Биты GIE и SWIE не влияют.
- Прерывания от точек наблюдения и из JTAG имеют дополнительные биты разрешения.
- Прерывание происходит сразу за командой, которая покинула последнюю стадию конвейера W в момент возникновения запроса.
- Для хранения адреса возврата используется специальный регистр DBGE.
- В качестве адреса-вектора процедуры обработки используется фиксированный адрес 0x001F_03A4.

- Возникновение исключения приводит к установке специального бита EMUL. Бит виден в 22-м разряде регистра SQSTAT. Бит может быть установлен только при возникновении данной исключительной ситуации. Бит может быть сброшен только при выходе из обработки исключения командой RTI. Иных механизмов установки или сброса данного бита не существует.
- Установка бита EMUL приводит к переходу в режим эмулятора. Также это приводит к запрещению всех программных и аппаратных прерываний.

Последовательность действий процессора выполняемых при возникновении прерывания эмулятора следующая:

- В поле SQSTAT[15:12] процессор сохраняет код исключительной ситуации. Это позволяет идентифицировать источник исключения.
- Сбрасывается весь конвейер процессора и начинается чтение команд по адресу 0x001F_03A4. При этом изменения адреса не происходит. Данный адрес соответствует регистру команд интерфейса JTAG, т.е. процессор все время исполняет те команды, которые ему выставляет JTAG интерфейс.
- В регистре DBGE сохраняется адрес возврата.
- Устанавливается внутренний флаг EMUL, переводя процессор в режим эмулятора и запрещая все аппаратные и программные прерывания.
- Далее процессор выполняет все команды, которые предоставляет ему JTAG интерфейс.
- Перед выходом из прерывания программа копирует содержимое регистра DBGE в регистр RETI с помощью команды RETI = DBGE;;. Регистр RETIB в данном случае не должен использоваться.
- Выход из обработчика осуществляется с помощью команды RTI.

Так как при возврате из программного прерывания используется регистр RETI, то последовательность выхода может быть такой:

```
[j31+temporary address] = RETI;;
RETI = DBGE;;
RTI (NP); RETI = [j31+temporary address];;
```

Прерывание эмулятора накладывает серьезные ограничения на перечень команд, которые должны поставляться интерфейсом JTAG. Например, запрещены любые команды перехода и т.д. При переходе в режим эмулятора блокируются некоторые аппаратные ресурсы. Например, таймеры останавливаются и не меняют свое значения до момента выхода из режима эмуляции.

7.2.6 Источники прерываний процессора

Таблица 20 – Таблица вектора прерываний

Приоритет	Прерывание	Описание	Срабатывание
63 (самый высокий)	Резервные	Резервные	
62	SW	Программное исключение	По уровню
61–58	Резервные	Резервные	
57	HWERR	Аппаратная ошибка	По уровню
56–54	Резервные	Резервные	
53	TIMER1H	Высокий приоритет таймера 1	По фронту
52	TIMER0H	Высокий приоритет таймера 0	По фронту
51	Резервные	Резервные	
50	nBUSLOCK	Блокировка шины	По фронту

Приоритет	Прерывание	Описание	Срабатывание
49	Резервные	Резервные	
48	VIRPT	Векторное прерывание	По фронту
47–45	Резервные	Резервные	
44	nIRQ3	Вывод nIRQ3	По фронту или по уровню
43	nIRQ2	вывод nIRQ2	По фронту или по уровню
42	nIRQ1	вывод nIRQ1	По фронту или по уровню
41	nIRQ0	вывод nIRQ0	По фронту или по уровню
40–39	Резервные	Резервные	
38	DMA13	DMA канал 13	По фронту
37	DMA12	DMA канал 12	По фронту
36–33	Резервные	Резервные	
32	DMA11	DMA канал 11	По фронту
31	DMA10	DMA канал 10	По фронту
30	DMA9	DMA канал 9	По фронту
29	DMA8	DMA канал 8	По фронту
28–26	Резервные	Резервные	
25	DMA7	DMA канал 7	По фронту
24	DMA6	DMA канал 6	По фронту
23	DMA5	DMA канал 5	По фронту
22	DMA4	DMA канал 4	По фронту
21–18	Резервные	Резервные	
17	DMA3	DMA канал 3	По фронту
16	DMA2	DMA канал 2	По фронту
15	DMA1	DMA канал 1	По фронту
14	DMA0	DMA канал 0	По фронту
13–10	Резервные	Резервные	
9	LINK3	Запрос порта связи 3	По уровню
8	LINK2	Запрос порта связи 2	По уровню
7	LINK1	Запрос порта связи 1	По уровню
6	LINK0	Запрос порта связи 0	По уровню
5–4	Резервные	Резервные	
3	TIMER1L	Низкий приоритет таймера 1	По фронту
2	TIMER0L	Низкий приоритет таймера 0	По фронту
1	Резервные	Резервные	
0 (самый низкий)	KERNEL	ядро	По фронту

7.2.6.1 Прерывания по срабатыванию таймера

Имеется четыре прерывания от таймеров: два для каждого таймера, одно – с высоким приоритетом и одно – с низким. Два приоритетов для одного таймера реализованы для возможности выбрать пользователю уровень приоритета.

Регистры вектора прерываний таймера: IVTIMER0HP, IVTIMER1HP, IVTIMER0LP, и IVTIMER1LP. В случае прерывания от таймера, устанавливаются оба бита приоритетов (высокий и низкий) в ILAT (например, для таймера 0 – это биты 2 и 52).

Сброс или установка бита запроса прерывания от таймера в регистре ILAT с помощью адресов ILATSTx или ILATCLx оказывает влияние сразу на оба бита таймера, т.е. нельзя разграничить запрос низкого от запроса высокого приоритета.

После сброса процессора прерывания таймеров отключены, и векторы не инициализированы.

7.2.6.2 Прерывания на обслуживание порта связи

Существуют четыре канала портов связи, которые обычно работают с выделенными им каналами DMA. Регистры вектора обслуживания порта связи: IVLINK0, IVLINK1, IVLINK2, и IVLINK3. Когда данные поступают в буфер приемника порта связи и соответствующий ему канал DMA не инициализирован, порт связи формирует запрос прерывания на обслуживание. После сброса прерывания портов связи отключены, и векторы не инициализированы.

7.2.6.3 Прерывания завершения работы каналов DMA

Имеется 14 каналов DMA. Каждый канал DMA может генерировать прерывание. Регистры вектора: IVDMA0, IVDMA1, IVDMA2, IVDMA3, IVDMA4, IVDMA5, IVDMA6, IVDMA7, IVDMA8, IVDMA9, IVDMA10, IVDMA11, IVDMA12, и IVDMA13. Если бит разрешения прерывания в регистре управления передачей DMA установлен, то канал DMA генерирует прерывание, когда канал завершает передачу блока. После сброса прерывания DMA включены, и векторы инициализированы нулями для загрузочных целей.

7.2.6.4 Прерывания от внешних источников (nIRQ3–0)

Существует четыре внешних прерывания общего назначения, которые могут быть запрограммированы как чувствительные к уровню или фронту. Регистры вектора внешних прерываний (nIRQ3–0): IVIRQ0, IVIRQ1, IVIRQ2, и IVIRQ3. Когда активируется один из входов, запускается прерывание (если они разрешены). Тип прерывания по фронту или уровню программируется в регистре INTCTL. Выводы внешних прерываний (nIRQ3–0) позволяют процессору получать сигналы прерываний от внешних устройств.

После сброса прерывания nIRQ могут быть включены или выключены. Это зависит от выбранного режима старта процессора.

7.2.6.5 Векторное прерывание

Векторное прерывание – прерывание общего назначения для использования другим мастером. Другое ведущее устройство или процессор могут записывать адрес в этот регистр. Запись вызывает установку запроса прерывания, а записанное значение в регистре – адрес обработки прерывания. Регистр вектора прерывания – VIRPT.

После сброса векторное прерывание разрешено, но вектор не инициализирован.

7.2.6.6 Прерывание от захвата шины

Прерывание захвата шины происходит, когда установлен бит захвата шины в регистре nBUSLOCK и процессор становится мастером шины. Регистр вектора захвата шины – IVBUSLK. Данное прерывание используется для напоминания о том, что процессор захватил внешнюю шину. Процессор может использовать данное прерывание для сокращения возможного времени простоя при доступе к внешней памяти в мультипроцессорной системе.

После сброса прерывание захвата шины выключено, и вектор не инициализирован.

7.2.6.7 Прерывание от аппаратной ошибки

Прерывание аппаратной ошибки указывает на одну из возможных ошибок:

- Ошибка в DMA. При инициализации некоторого канала обнаружена ошибочная ситуация.
- Ошибка в AutoDMA каналах. Указывает на то, что при выключенном канале в него была произведена запись данных. Бит 17 в SYSTAT активен, если произошло данное событие.
- Широковещательное чтение из внешней памяти. Бит 16 в SYSTAT активен, если произошла данная ошибка.
- Доступ к отключенной SDRAM. Бит 18 в SYSTAT активен, если произошло данное событие.
- Мультипроцессорное авточтение. Бит 19 в SYSTAT активен, если произошло данное событие.
- Ошибка порта связи. В одном из регистров LSTATx в полях RER или TER активна ошибка.

Регистр вектора прерываний аппаратных ошибок – IVHW.

После сброса прерывание от аппаратной ошибки выключено, и вектор не инициализирован.

7.2.6.8 Программные исключения

Программные исключения – прерывания, появляющиеся во время выполнения программного кода. Если происходит исключение и бит SWIE равен нулю – исключение теряется. Вектор прерываний программных исключений IVSW. Программные исключения включают в себя следующие ситуации:

- ошибка работы с плавающей точкой, если бит IVEN в регистре XSTAT/YSTAT установлен;
- исчезновение результата, если бит UEN в регистре XSTAT/YSTAT установлен;
- переполнение результата, если бит OEN в регистре XSTAT/YSTAT установлен;
- команда TRAP;
- совпадение точки наблюдения, когда на это событие запрограммировано как исключение;
- неопределенные команды (в некоторых случаях);
- недопустимая комбинация команд (в некоторых случаях);
- внешний доступ к недопустимому пространству (широковещательное мультипроцессорное чтение при выполнении команды загрузки);
- доступ к длинным данным (64-битовое или 128-битовое слова) при невыровненном адресе на границе слова;
- нарушение защиты – доступ к регистру супервизора в пользовательском режиме;
- две команды в одной строке с одним назначением;
- больше трех результатов в одной строке вычислительной команды (например, сложение и вычитание, умножение, сдвиг);
- две 32-битных непосредственных команды в одной строке.

После сброса программные исключения выключены, и вектор не инициализирован.

7.3 Память и регистры

В данном разделе приведено описание карты памяти и регистров. Для получения информации об использовании регистров для вычислений и памяти для загрузки и хранения, необходимо обращаться к руководству по программированию процессора. Для информации по использованию регистров для конфигурирования процессорной периферии, необходимо смотреть соответствующие разделы описания периферийных модулей.

Процессор имеет шесть внутренних блоков памяти как показано на карте памяти (Рисунок 12). Каждый блок памяти состоит их 4 Мбит пространства памяти и сконфигурирован как 128 К слова, каждое шириной 32 бита. Существует четыре отдельных 128-битных внутренних шины, каждая из которых может получать доступ к блокам памяти. Процессор ввода-вывода (DMA) имеет свою собственную внутреннюю шину, так что он не соревнуется с ядром за использование внутренней шины. Таким образом, за один цикл может происходить до четырех 128-битных передач внутри ядра: две передачи данных, одна команда и одна передача интерфейса ввода-вывода.

Ядро процессора имеет три шины (I-шина, J-шина, и K-шина), каждая из которых соединена со всеми блоками внутренней памяти. Эти шины имеют ширину 128 бит для передачи до четырех команд или четырех слов данных в пределах одного цикла. Внешние по отношению к ядру устройства (DMA, хост-процессор, другие процессоры кластера) используют S-шину для доступа к памяти. В каждом цикле возможен только один доступ к каждому блоку памяти, т.е. если в течение одного такта мы имеем четыре запроса к различным модулям памяти, то все запросы будут обслужены без потери скорости.

Большинство регистров процессора могут быть классифицированы как универсальные регистры (Ureg). Имеются команды для пересылки данных между двумя любыми Ureg, между Ureg и памятью или для непосредственной загрузки константы в Ureg.

Процессор имеет 32-битные адресные шины, обеспечивающие адресное пространство до четырех гигаслов. Такое адресное пространство общее для кластера процессоров, разделяющих одну кластерную шину. Процессор поддерживает словную адресацию памяти. *При этом возможна байтовая адресация, но она поддерживается только для внутренней памяти*. Данная глава определяет карту памяти для каждого процессора в системе и показывает, где в пространстве памяти определяется каждый элемент. Зоны пространства памяти составлены из следующих областей:

- пространство внешней памяти – область для стандартной адресации внешней памяти, включая SDRAM, банк 0 (MS0), банк 1 (MS1) и хост (nMSH);
- Внешнее мультипроцессорное пространство – внутренняя память всех других процессоров в объединенной мультипроцессорной системе;
- пространство внутренней памяти – область для стандартной внутренней адресации.

Общая карта памяти показана ниже (Рисунок 12).

Для быстродействия процессора большое значение имеет конфигурация банков внутренней памяти. Шесть банков внутренней памяти обеспечивают возможность одновременного параллельного доступа к ним в течение одного такта ядра. Такое уровень быстродействия можно достичь только если процессор будет, например, производить чтение команд из банка 0, чтение данных по шине J из банка 1, чтение данных по шине K из банка 2, чтение-запись по шине S внешним мастером

банка 3, отложенная запись по шине J в банк 4 и отложенная запись по шине K в банк 5.

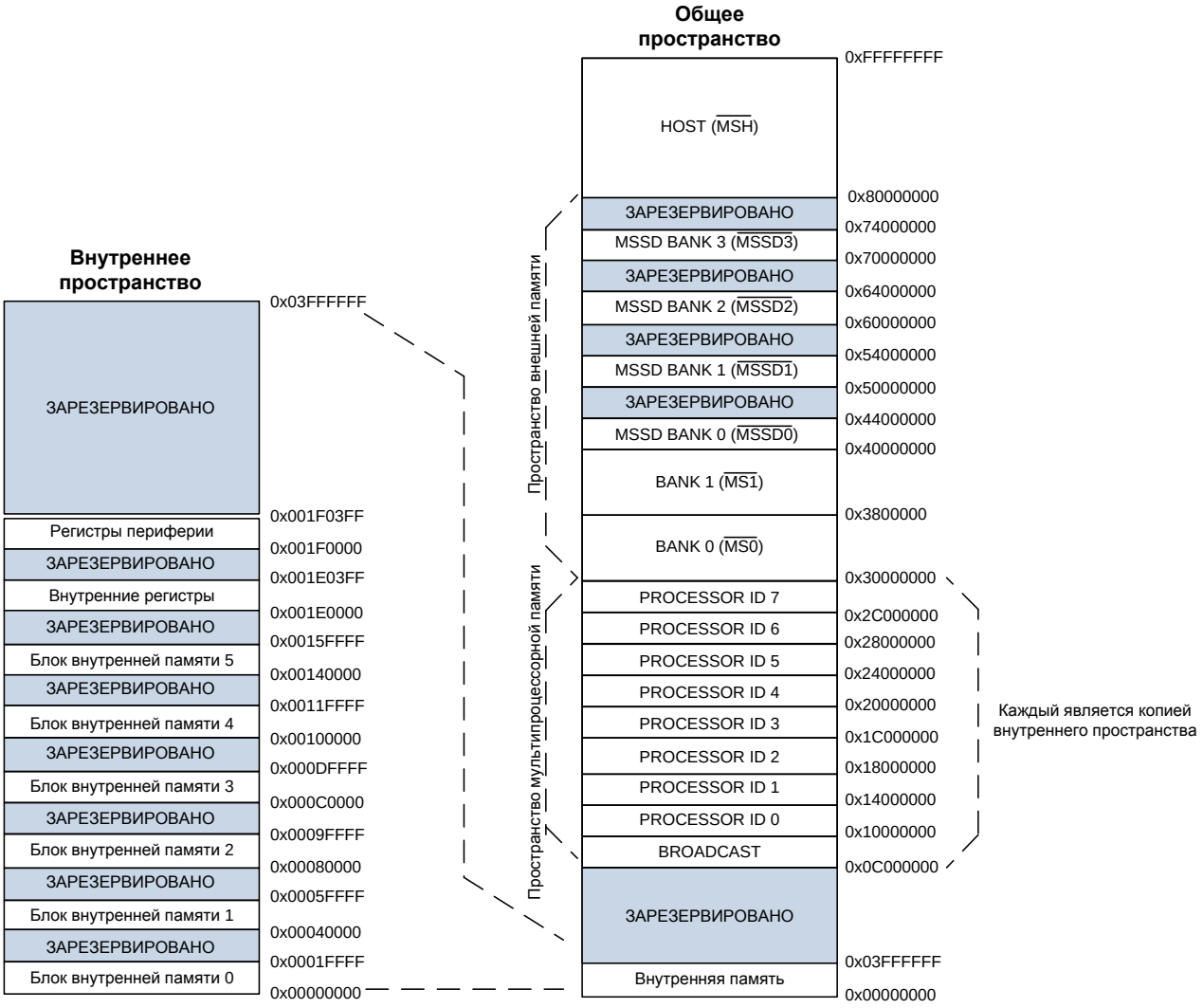


Рисунок 12 – Карта памяти 1967ВН028

Дополнительно к вышеописанной параллельности, возможны параллельные операции и при работе с одним банком. Каждый банк внутренней памяти разбит на два модуля: в одном хранятся четные квадрослова, а в другом нечетные. Имеется возможность выполнять два доступа в одном такте к одному банку, если происходит работа с разными по четности квадрословами. Это позволяет в некоторых случаях (сохранение и восстановление контекста, пересылка данных) обеспечивать ширину шины данных до 256 разрядов.

7.3.1 Пространство адресов хоста

Пространство адресов хоста – это пространство, определяемое для хост-процессора при доступе к нему как к ведомому. При использовании этого пространства, конвейерный или асинхронный протокол используется в соответствии с битами управления в регистре SYSCON. Адресное пространство хоста имеет размер два гигаслова и определяется значением старшего (31-го) бита адресной шины. Если бит ADDR31 равен 1, то это указывает на доступ к адресному пространству хост-процессора. Если же бит равен нулю, то это может быть внутренняя память, внешняя память кластера или мультипроцессорное пространство.

7.3.2 Пространство банка внешней памяти

Это адресное пространство памяти относится к внешней памяти и устройствам ввода-вывода (SDRAM, SRAM, периферия ввода-вывода и другие стандартные устройства памяти). Внешний интерфейс поддерживает разбиение общего банка внешней памяти на отдельные банки и ставит им в соответствие специальные сигналы выборки и другие управляющие сигналы. Имеется набор банков для SDRAM и доступ к ним производится через выводы выбора внешней памяти nMSSD0, nMSSD1, nMSSD2, и nMSSD3; доступ к этим банкам осуществляется по протоколу SDRAM. Другой набора банков определен выводами выбора внешней памяти MS0 и MS1, где протокол доступа конфигурируется пользователем как конвейерный или протокол медленного устройства, и параметры этих банков определяются регистром SYSCON. Адрес внешней памяти разделяется на поля как показано ниже (Таблица 21).

Таблица 21 – Пространство банка внешней памяти

Биты	Имя	Определение
ADDR25–0	Address	Биты которые осуществляют выбор данных внутри адресного пространства определенного банка
ADDR30–26	MS/PRID	Выбор типа банка внешней памяти: 01100 – банк 0 (MS0) 01110 – банк 1 (MS1) 10000 – SDRAM банк 0 (nMSSD0) 10001 – зарезервировано 10100 – SDRAM банк 1 (nMSSD1) 10101 – зарезервировано 11000 – SDRAM банк 2 (nMSSD2) 11001 – зарезервировано 11100 – SDRAM банк 3 (nMSSD3) 11101 – зарезервировано
ADDR31	Host Select	Определение типа адреса. Если ADDR31 = 0, то адрес может находиться во внешнем, внутреннем или мультипроцессорном адресном пространстве

7.3.3 Мультипроцессорное пространство

Мультипроцессорное пространство показывает, как каждый процессор кластера видит внутреннюю память любого другого процессора. Это позволяет одному процессору легко записывать и считывать данные других процессоров в

мультипроцессорной системе. Широковещательное пространство (broadcast) позволяет получать доступ по записи во все процессоры кластера одновременно. Каждый процессор в кластере имеет свой ID. Количество процессоров в кластере – от 2 до 8. Внешние мультипроцессорные адреса разделены на поля, как показано ниже (Таблица 22).

Таблица 22 – Пространство мультипроцессора

Биты	Биты	Биты
ADDR25–0	Address	Биты которые осуществляют выбор данных внутри адресного пространства определенного банка.
ADDR30–26	MS/PRID	Выбор банка внутренней памяти процессора кластера: 00011 – широковещательная запись 00100 – процессор ID0 00101 – процессор ID1 00110 – процессор ID2 00111 – процессор ID3 01000 – процессор ID4 01001 – процессор ID5 01010 – процессор ID6 01011 – процессор ID7
ADDR31	Host Select	Определитель типа адреса. Если ADDR31=0, то адрес может находиться во внешнем, внутреннем или мультипроцессорном адресном пространстве.

Собственное внутренне пространство процессора, включая универсальные регистры, может быть доступно через мультипроцессорное пространство, включая широковещательное пространство, только для операций записи. Из-за исполнения данных операций через внешнюю шину, они должны использоваться только в особых случаях, когда данные должны пройти через внешний интерфейс процессора.

7.3.4 Внутреннее адресное пространство

Внутреннее адресное пространство соответствует собственному адресному пространству процессора и универсальным регистрам. Внутреннее пространство адресов показано выше (Рисунок 12). Оно используется для передачи внутри процессора и доступ к этому пространству памяти не отображается на шине кластера. Оно используется для доступа к блокам внутренней памяти. Из рисунка видно, что все универсальные регистры процессора, а также регистры периферии имеют адрес в общей карте памяти. Однако доступ к регистрам по адресу памяти возможен только через мультипроцессорное пространство. Невозможно обратится к регистрам, используя их внутренний адрес вне мультипроцессорной области.

В командах загрузки\хранения адрес не может указывать на регистры через внутреннее пространство. DMA также не может напрямую получать доступ к регистру.

7.3.5 Пространство универсальных регистров

Универсальные регистры в карте памяти распределены по группам, в соответствии с их отношением к различным модулям процессора. Например, регистры общего назначения (РОН) данных (XR31–0, YR31–0, или XYR31–0) вычислительных блоков находятся в одной группе регистров, тогда как РОН данных для целочисленного АЛУ (J30–0 или K30–0) – в других. Каждая группа регистров

соответствует определенному диапазону адресов памяти, т.к. все регистры в группе имеют свой адрес в карте памяти. Термин «универсальный регистр» указывает на несколько важных особенностей:

1. Регистр имеет адрес в карте памяти, что указывает на возможность доступа других процессоров в мультипроцессорной системе к данному регистру посредством его адреса.
2. Содержимое регистра может быть загружено из памяти, сохранится в памяти или передаваться в\из других универсальных регистров процессора.
3. Регистр может быть доступен в одинарный, двойной и счетверенный (квадрорегистр). Правда ряд регистров может быть доступен только как одинарные.

Мультипроцессор получает доступ к универсальным регистрам другого процессора с использованием адресов памяти, которые находятся в таблицах групп регистров как часть адреса. Но при доступе к регистру определенного процессора необходимо добавление смещения для каждого конкретного процессора (Рисунок 12). Следующий пример кода показывает мультипроцессорные доступы к универсальным регистрам другого процессора. В примере один процессор в мультипроцессорной системе переносит содержимое своего регистра XR1 в регистр XR0 процессора P1.

J0 = P1_OFFSET_LOC ;;

/* P1_OFFSET_LOC равен 0x1400 0000, что указывает на процессор P1*/

J1 = XR0_LOC ;;

/* XR0_LOC – 0x001E 0000; внутренний адрес регистра XR0 в процессоре*/

[J0 + J1] = XR1 ;;

/* перенос регистра XR1 текущего процессора в регистр XR0 процессора P1 */

Процессор не может выполнить операцию чтения собственных регистров, используя мультипроцессорный доступ. Подобный тип чтения вызывает генерацию процессором исключительной ситуации.

Загрузка, сохранение и пересылка содержимого универсальных регистров доступна внутри процессора с использованием имени регистра в командах процессора, а не по адресу в памяти. Имена регистров содержатся в таблице групп регистров. Команда загрузки регистра читает данные из памяти процессора и помещает их в регистр. Команда сохранения регистра берет данные из регистра и помещает их в память процессора. Команда пересылки копирует данные из одного регистра в другой. Имеются также команды загрузки константы в регистр, когда значение операнда берется из кода команды и помещается в регистр. Следующий пример показывает доступы загрузки, сохранения и передачи одного регистра.

```
XR0 = 0x76543210 ;; /* загружает в XR0 конкретные 32-бит данные */
XR4 = [J31 + 0x43] ;; /* загружает в XR0 данные из ячейки памяти с адресом 0x43 */
[J0 + J4] = YR0 ;; /* сохраняет YR0 в память */
XR7 = SQSTAT ;; /* передает содержимое SQSTAT в XR7 */
```

При возможности выполнения процессором до 4-х команд одновременно можно выполнить:

- две команды загрузки регистров из памяти;
- две команды сохранения в память;
- две команды пересылки;
- две загрузки регистров константами.

Все эти типы команд выполняются в целочисленных J и K модулях.

Для одновременного доступа к двум или четырем регистрам, образующим длинные слова и квадрослова, предусмотрена специальная техника образования нового имени регистра. При этом, изменение имени регистра указывает на диапазон регистров, а не на новый регистр. Например, имя сдвоенного регистра XR1:0 указывает регистр R1 и R0 в вычислительном блоке X, а счетверенное имя регистра J31:28 указывает регистры J31, J30, J29, и J28 в J-IALU. Обязательно выравнивание границ адресов сдвоенных и счетверенных регистров. Для сдвоенного регистра его младший номер должен быть кратен двум (например, R1:0, R3:2, R27:26), а для счетверенного кратен 4 (например, R3:0, R7:4, R31:28). Следующий пример кода показывает загрузку, сохранение и передачу сдвоенного и счетверенного регистра.

```
YR5:4 = L [J31 + 0x1ff0] ;; /* загрузка YR5:4 из памяти */
XR3:0 = Q [K4 += K5] ;; /* загрузка XR3:0 из памяти */
L [J0 + J2] = YR31:30 ;; /* сохранение YR31:30 в память */
Q [K31 + K28] = YR7:4 ;; /* сохранение YR7:4 в память */
DCS0 = YR11:8 ;; /* передача YR11:8 в DCS0 (DP, DY, DX, DI) */
```

Обычно цикл загрузки или сохранения регистра в памяти происходит между регистром и словом памяти одинакового размера – 32, 64 и 128 бит. В этом случае операция называется «нормальный доступ чтения \ записи».

Для использования преимуществ обработки «одна команда – много данных» (SIMD), архитектура шины процессора поддерживает типы доступа, в которых содержимое одного слова памяти может быть загружено в два регистра (широковещательное чтение), и тип доступа, в котором несколько регистров могут быть загружены\сохранены с различными данными, используя одно слово в памяти (доступ объединенного чтения или записи).

Отметим, что доступ широковещательного чтения происходит только между памятью и регистрами внутри процессора, тогда как мультипроцессорная широковещательная запись может происходить одновременно по всем процессорам в мультипроцессорной системе.

Пространство регистров состоит их 64 регистровых групп с количеством регистров в группе равным 32. Группы регистров определены в диапазоне 0x3F–0 (63–0). Группы 0x1F–0 (31-0) доступны для любых команд передачи данных (загрузка константой, пересылка регистра, загрузка и сохранение в памяти) Группы 0x3F–0x20 доступны только командам пересылки регистров и другим мастерам кластера.

Группы регистров:

- 0x00–0x09 вычислительные модули X и У;
- 0x0C–0x0F целочисленные J и K АЛУ;
- 0x0A,0x1B модуль отладки;
- 0x1A устройство управления;
- 0x1E–0x1F устройство защиты памяти;
- 0x20 – 0x23 контроллер DMA;
- 0x24 контроллер внешнего порта;
- 0x25 – 0x27 порты связи;
- 0x38, 0x39, и 0x3A контроллер прерываний;
- 0x3D модуль JTAG;
- 0x3F каналы autoDMA.

Прочие группы зарезервированы. Данные регистры не могут быть доступны для приложений, т.к. они могут вызвать непредсказуемую реакцию процессора.

Существует прямая связь между адресом универсального регистра в карте памяти и группой, в которой он находится. Важно знать номер группы, которой принадлежит регистр, т.к. они могут иметь различные ограничения доступа. Для определения номера группы регистров см. Рисунок 13.

Рисунок 13 – Номер группы регистров в зависимости от адреса регистра

Также важно обратить внимание на адрес регистра в плане его (регистра) расположения в процессоре: в ядре процессора (внутренние универсальные регистры) или в периферийном модуле SOC-шины (Рисунок 12), т.к. регистры имеют различные ограничения доступа. Некоторые из регистров не используют все 32 бита. Неиспользованные биты зарезервированы. При записи в регистр с резервными битами, резервные биты должны быть записаны нулевым значением. При чтении регистра с резервными битами резервные биты могут иметь любое значение.

7.3.6 Группы регистров вычислительных модулей

Файл регистров вычислительного модуля содержит 32 регистра общего назначения (РОН). Существует два подобных файла, по одному в каждом вычислительном модуле (X и Y). Однако, фактически имея две группы по 32 регистра, регистры вычислительных модулей используют 10 номеров групп (с 0 по 9) для своей адресации. Можно сказать, что каждый регистр имеет собственное имя и несколько псевдонимов. Использование подобных псевдонимов удобно в командах пересылки регистров, а также в командах чтения-записи при работе с памятью. С помощью псевдонима происходит дополнительное кодирование выполняемой с регистром операции.

В таблицах ниже (Таблица 78 – Таблица 82) указаны виды доступов к регистрам вычислительного блока в соответствии с их распределением в памяти.

Таблица 23 – Группа регистров вычислительного блока X

Имя	Адрес	Значение по умолчанию	Имя	Адрес	Значение по умолчанию
XR0	0x1E 0000	Не определено	XR16	0x1E 0010	Не определено
XR1	0x1E 0001	Не определено	XR17	0x1E 0011	Не определено
XR2	0x1E 0002	Не определено	XR18	0x1E 0012	Не определено
XR3	0x1E 0003	Не определено	XR19	0x1E 0013	Не определено
XR4	0x1E 0004	Не определено	XR20	0x1E 0014	Не определено
XR5	0x1E 0005	Не определено	XR21	0x1E 0015	Не определено
XR6	0x1E 0006	Не определено	XR22	0x1E 0016	Не определено
XR7	0x1E 0007	Не определено	XR23	0x1E 0017	Не определено
XR8	0x1E 0008	Не определено	XR24	0x1E 0018	Не определено
XR9	0x1E 0009	Не определено	XR25	0x1E 0019	Не определено

XR10	0x1E 000A	Не определено	XR26	0x1E 001A	Не определено
XR11	0x1E 000B	Не определено	XR27	0x1E 001B	Не определено
XR12	0x1E 000C	Не определено	XR28	0x1E 001C	Не определено
XR13	0x1E 000D	Не определено	XR29	0x1E 001D	Не определено
XR14	0x1E 000E	Не определено	XR30	0x1E 001E	Не определено
XR15	0x1E 000F	Не определено	XR31	0x1E 001F	Не определено

Таблица 24 – Группа регистров вычислительного блока Y

Имя	Адрес	Значение по умолчанию	Имя	Адрес	Значение по умолчанию
YR0	0x1E 0040	Не определено	YR16	0x1E 0050	Не определено
YR1	0x1E 0041	Не определено	YR17	0x1E 0051	Не определено
YR2	0x1E 0042	Не определено	YR18	0x1E 0052	Не определено
YR3	0x1E 0043	Не определено	YR19	0x1E 0053	Не определено
YR4	0x1E 0044	Не определено	YR20	0x1E 0054	Не определено
YR5	0x1E 0045	Не определено	YR21	0x1E 0055	Не определено
YR6	0x1E 0046	Не определено	YR22	0x1E 0056	Не определено
YR7	0x1E 0047	Не определено	YR23	0x1E 0057	Не определено
YR8	0x1E 0048	Не определено	YR24	0x1E 0058	Не определено
YR9	0x1E 0049	Не определено	YR25	0x1E 0059	Не определено
YR10	0x1E 004A	Не определено	YR26	0x1E 005A	Не определено
YR11	0x1E 004B	Не определено	YR27	0x1E 005B	Не определено
YR12	0x1E 004C	Не определено	YR28	0x1E 005C	Не определено
YR13	0x1E 004D	Не определено	YR29	0x1E 005D	Не определено
YR14	0x1E 004E	Не определено	YR30	0x1E 005E	Не определено
YR15	0x1E 004F	Не определено	YR31	0x1E 005F	Не определено

Таблица 25 – Объединенная группа регистров вычислительного модуля XY

Имя	Адрес	Значение по умолчанию	Имя	Адрес	Значение по умолчанию
XYR0	0x1E 0080	Не определено	XYR16	0x1E 0090	Не определено
XYR1	0x1E 0081	Не определено	XYR17	0x1E 0091	Не определено
XYR2	0x1E 0082	Не определено	XYR18	0x1E 0092	Не определено
XYR3	0x1E 0083	Не определено	XYR19	0x1E 0093	Не определено
XYR4	0x1E 0084	Не определено	XYR20	0x1E 0094	Не определено
XYR5	0x1E 0085	Не определено	XYR21	0x1E 0095	Не определено
XYR6	0x1E 0086	Не определено	XYR22	0x1E 0096	Не определено
XYR7	0x1E 0087	Не определено	XYR23	0x1E 0097	Не определено
XYR8	0x1E 0088	Не определено	XYR24	0x1E 0098	Не определено
XYR9	0x1E 0089	Не определено	XYR25	0x1E 0099	Не определено

XYR10	0x1E 008A	Не определено	XYR26	0x1E 009A	Не определено
XYR11	0x1E 008B	Не определено	XYR27	0x1E 009B	Не определено
XYR12	0x1E 008C	Не определено	XYR28	0x1E 009C	Не определено
XYR13	0x1E 008D	Не определено	XYR29	0x1E 009D	Не определено
XYR14	0x1E 008E	Не определено	XYR30	0x1E 009E	Не определено
XYR15	0x1E 008F	Не определено	XYR31	0x1E 009F	Не определено

Таблица 26 – Объединенная группа регистров вычислительного модуля YX

Имя	Адрес	Значение по умолчанию	Имя	Адрес	Значение по умолчанию
YXR0	0x1E 00C0	Не определено	YXR16	0x1E 00D0	Не определено
YXR1	0x1E 00C1	Не определено	YXR17	0x1E 00D1	Не определено
YXR2	0x1E 00C2	Не определено	YXR18	0x1E 00D2	Не определено
YXR3	0x1E 00C3	Не определено	YXR19	0x1E 00D3	Не определено
YXR4	0x1E 00C4	Не определено	YXR20	0x1E 00D4	Не определено
YXR5	0x1E 00C5	Не определено	YXR21	0x1E 00D5	Не определено
YXR6	0x1E 00C6	Не определено	YXR22	0x1E 00D6	Не определено
YXR7	0x1E 00C7	Не определено	YXR23	0x1E 00D7	Не определено
YXR8	0x1E 00C8	Не определено	YXR24	0x1E 00D8	Не определено
YXR9	0x1E 00C9	Не определено	YXR25	0x1E 00D9	Не определено
YXR10	0x1E 00CA	Не определено	YXR26	0x1E 00DA	Не определено
YXR11	0x1E 00CB	Не определено	YXR27	0x1E 00DB	Не определено
YXR12	0x1E 00CC	Не определено	YXR28	0x1E 00DC	Не определено
YXR13	0x1E 00CD	Не определено	YXR29	0x1E 00DD	Не определено
YXR14	0x1E 00CE	Не определено	YXR30	0x1E 00DE	Не определено
YXR15	0x1E 00CF	Не определено	YXR31	0x1E 00DF	Не определено

Таблица 27 – Группа регистров широковещательного доступа вычислительного модуля XY

Имя	Адрес	Значение по умолчанию	Имя	Адрес	Значение по умолчанию
XYR0	0x1E 0100	Не определено	XYR16	0x1E 0110	Не определено
XYR1	0x1E 0101	Не определено	XYR17	0x1E 0111	Не определено
XYR2	0x1E 0102	Не определено	XYR18	0x1E 0112	Не определено
XYR3	0x1E 0103	Не определено	XYR19	0x1E 0113	Не определено
XYR4	0x1E 0104	Не определено	XYR20	0x1E 0114	Не определено
XYR5	0x1E 0105	Не определено	XYR21	0x1E 0115	Не определено
XYR6	0x1E 0106	Не определено	XYR22	0x1E 0116	Не определено
XYR7	0x1E 0107	Не определено	XYR23	0x1E 0117	Не определено
XYR8	0x1E 0108	Не определено	XYR24	0x1E 0118	Не определено

XYR9	0x1E 0109	Не определено	XYR25	0x1E 0119	Не определено
XYR10	0x1E 010A	Не определено	XYR26	0x1E 011A	Не определено
XYR11	0x1E 010B	Не определено	XYR27	0x1E 011B	Не определено
XYR12	0x1E 010C	Не определено	XYR28	0x1E 011C	Не определено
XYR13	0x1E 010D	Не определено	XYR29	0x1E 011D	Не определено
XYR14	0x1E 010E	Не определено	XYR30	0x1E 011E	Не определено
XYR15	0x1E 010F	Не определено	XYR31	0x1E 011F	Не определено

Отметим, что дополнительно имеются номера групп, которые используются командами альтернативного доступа с использованием буфера DAB и кольцевых буферов.

7.3.7 Регистры вычислительного модуля без адреса в памяти

Есть несколько регистров в вычислительных модулях, которые не являются универсальными и которые не доступны как универсальные регистры посредством их адресации в памяти. Эти регистры доступны только при выполнении специальных команд, которые передают данные между ними и РОН вычислительного модуля.

7.3.7.1 Регистры статуса вычислительных модулей (XSTAT/YSTAT)

Регистры XSTAT и YSTAT – 32-битные регистры вычислительных модулей, которые хранят состояние флагов модулей. Каждый флаг регистра обновляется при выполнении соответствующей команды. Как известно вычислительный модуль состоит из вычислительных блоков: ALU, умножитель, сдвигатель, CLU. Каждый из этих блоков формирует специальные флаги-признаки результата операции. Регистр состояний хранит данные флаги. Флаги имеют разный тип. Большинство флагов регистра изменяет свое значение при выполнении соответствующих команд, однако есть специальные флаги (sticky) которые после их установки в 1-е значение уже невозможно сбросить. Сброс возможен только командой загрузки регистра. Stiky биты очень удобны, когда нужно в конце алгоритма проверить были ли аномальные ситуации при выполнении вычислений на протяжении всего алгоритма. В отличие от них другие флаги нужно проверять (если необходимо) сразу после выполнения операции. Подробное описание бит регистров X/YSTAT (значение после сброса = 0x0000 0000) приведено ниже (Таблица 28).

Таблица 28 – Регистр состояния вычислительного модуля

Бит	Имя	Назначение
0	AZ	ALU. Признак нулевого результата операции
1	AN	ALU. Признак отрицательного результата операции
2	AV	ALU. Признак переполнения результата операции
3	AC	ALU. Признак переноса
4	MZ	Умножитель. Признак нулевого результата операции
5	MN	Умножитель. Признак отрицательного результата операции
6	MV	Умножитель. Признак переполнения результата операции
7	MU	Умножитель. Признак исчезновения (underflow) результата операции
8	SZ	Сдвигатель. Признак нулевого результата операции
9	SN	Сдвигатель. Признак отрицательного результата операции
10	BF0	Сдвигатель. Флаг блока операций с плавающей запятой
11	BF1	Сдвигатель. Флаг блока операций с плавающей запятой
12	AI	Ошибочная (Invalid) операция АЛУ с плавающей запятой

13	MI	Ошибочная (Invalid) операция умножителя с плавающей запятой
14	TROV	CLU. Переполнение операции trellis
15	TRSOV	CLU. Переполнение операции trellis. Stiky бит
16	-	
17	-	
18	-	
19	-	
20	UEN	Разрешение (если 1) генерации исключительной ситуации при возникновении флага исчезновения (underflow) результата
21	OEN	Разрешение (если 1) генерации исключительной ситуации при возникновении флага переполнения(overflow) результата
22	IVEN	Разрешение (если 1) генерации исключительной ситуации при возникновении флага ошибочного (invalid) результата при операциях с плавающей запятой
23	-	
Stiky биты		
24	AUS	Underflow в АЛУ с плавающей запятой
25	AVS	Переполнение в АЛУ с плавающей запятой
26	AOS	Переполнение в АЛУ с фиксированной точкой
27	AIS	Ошибочный результат в АЛУ с плавающей запятой
28	MUS	Underflow в умножителе с плавающей запятой
29	MVS	Переполнение в умножителе с плавающей запятой
30	MOS	Переполнение в умножителе с фиксированной точкой
31	MIS	Ошибочный результат в умножителе с плавающей запятой

7.3.7.2 Регистры АЛУ

Регистры параллельных результатов (PR0 и PR1) – 32-битные регистры, используемые для специальных команд суммирования.

7.3.7.3 Регистры умножителя

Регистры результатов умножителя (MR3–0 и MR4) используются для накопления при выполнении операций умножения с накоплением, т.е. выполняют функции аккумуляторов.

7.3.7.4 Регистры сдвигателя

Регистр BFOTMP – 64-битный регистр для команды PUTBITS, используемой для организации потока бит.

7.3.7.5 Регистры блока CLU

Модуль CLU использует дополнительный набор регистров для выполнения своих команд. Это регистры данных (TR31–0), регистры истории (THR3–0) и регистр управления (CMCTL). Каждый регистр имеют ширину 32 бита.

7.3.8 Группы регистров целочисленного АЛУ

Каждое целочисленное АУ имеет две группы универсальных регистров. Первая группа – файл регистров общего назначения, включающий 32 слова. Вторая группа – регистры для указания параметров кольцевой буферизации. Группы регистров целочисленного АЛУ показаны в таблицах 29 – 32.

Файл регистров кольцевого буфера содержит следующие регистры:

- 3–0 это база кольцевого буфера JB3–0 и KB3–0;
- 7–4 это длина кольцевого буфера JL3–0 и KL3–0;
- 26–8 это резерв;
- 27 это регистр указателя стека режима пользователя (если включено разрешение на его использование);
- 31–28 это резерв.

Таблица 29 – Группа регистров целочисленного АЛУ типа J (J-IALU)

Имя	Адрес	Значение по умолчанию	Имя	Адрес	Значение по умолчанию
J0	0x1E 0180	Не определено	J16	0x1E 0190	Не определено
J1	0x1E 0181	Не определено	J17	0x1E 0191	Не определено
J2	0x1E 0182	Не определено	J18	0x1E 0192	Не определено
J3	0x1E 0183	Не определено	J19	0x1E 0193	Не определено
J4	0x1E 0184	Не определено	J20	0x1E 0194	Не определено
J5	0x1E 0185	Не определено	J21	0x1E 0195	Не определено
J6	0x1E 0186	Не определено	J22	0x1E 0196	Не определено
J7	0x1E 0187	Не определено	J23	0x1E 0197	Не определено
J8	0x1E 0188	Не определено	J24	0x1E 0198	Не определено
J9	0x1E 0189	Не определено	J25	0x1E 0199	Не определено
J10	0x1E 018A	Не определено	J26	0x1E 019A	Не определено
J11	0x1E 018B	Не определено	J27	0x1E 019B	Не определено
J12	0x1E 018C	Не определено	J28	0x1E 019C	Не определено
J13	0x1E 018D	Не определено	J29	0x1E 019D	Не определено
J14	0x1E 018E	Не определено	J30	0x1E 019E	Не определено
J15	0x1E 018F	Не определено	J31	0x1E 019F	Не определено

Таблица 30 – Группа регистров целочисленного АЛУ типа K (K-IALU)

Имя	Адрес	Значение по умолчанию	Имя	Адрес	Значение по умолчанию
K0	0x1E 01A0	Не определено	K16	0x1E 01B0	Не определено
K1	0x1E 01A1	Не определено	K17	0x1E 01B1	Не определено
K2	0x1E 01A2	Не определено	K18	0x1E 01B2	Не определено
K3	0x1E 01A3	Не определено	K19	0x1E 01B3	Не определено
K4	0x1E 01A4	Не определено	K20	0x1E 01B4	Не определено
K5	0x1E 01A5	Не определено	K21	0x1E 01B5	Не определено
K6	0x1E 01A6	Не определено	K22	0x1E 01B6	Не определено
K7	0x1E 01A7	Не определено	K23	0x1E 01B7	Не определено
K8	0x1E 01A8	Не определено	K24	0x1E 01B8	Не определено

K9	0x1E 01A9	Не определено	K25	0x1E 01B9	Не определено
K10	0x1E 01AA	Не определено	K26	0x1E 01BA	Не определено
K11	0x1E 01AB	Не определено	K27	0x1E 01BB	Не определено
K12	0x1E01AC	Не определено	K28	0x1E 01BC	Не определено
K13	0x1E01AD	Не определено	K29	0x1E 01BD	Не определено
K14	0x1E 01AE	Не определено	K30	0x1E 01BE	Не определено
K15	0x1E 01AF	Не определено	K31	0x1E 01BF	Не определено

Таблица 31 – Группа регистров циклического буфера целочисленного АЛУ типа J (J-IALU)

Имя	Адрес	Значение по умолчанию
JB0	0x1E 01C0	Не определено
JB1	0x1E 01C1	Не определено
JB2	0x1E 01C2	Не определено
JB3	0x1E 01C3	Не определено
JL0	0x1E 01C4	Не определено
JL1	0x1E 01C5	Не определено
JL2	0x1E 01C6	Не определено
JL3	0x1E 01C7	Не определено
резервные	0x1E 01C8– 0x1E 01DA	Не определено
JUSP	0x1E 01DB	Не определено
резервные	0x1E 01DC– 0x1E 01DF	Не определено

Таблица 32 – Группа регистров циклического буфера K-IALU

Имя	Адрес	Значение по умолчанию
KB0	0x1E 01E0	Не определено
KB1	0x1E 01E1	Не определено
KB2	0x1E 01E2	Не определено
KB3	0x1E 01E3	Не определено
KL0	0x1E 01E4	Не определено
KL1	0x1E 01E5	Не определено
KL2	0x1E 01E6	Не определено
KL3	0x1E 01E7	Не определено
резервные	0x1E 01E8– 0x1E 01FA	Не определено
KUSP	0x1E 01FB	Не определено
резервные	0x1E 01FC– 0x1E 01FF	Не определено

7.3.8.1 Регистры статусы целочисленного АЛУ (J31/JSTAT и K31/KSTAT)

Регистры статуса JSTAT (для J-IALU) и KSTAT (для K-IALU) хранят состояния флагов и обновляются в результате выполнения различных команд целочисленного АЛУ. При записи возможно обращение к регистру JSTAT как J31. При использовании в качестве операнда в арифметических, логических и функциональных операциях J31 рассматривается как ноль. Регистры J31/JSTAT (K31/KSTAT) (значение сброса = 0x0000 0000) описаны ниже (Таблица 33).

Таблица 33 – Регистр состояния целочисленного АЛУ

Бит	Имя	Назначение
0	Z	Признак нулевого результата операции
1	N	Признак отрицательного результата операции
2	V	Признак переполнения результата операции
3	C	Признак переноса
4	-	Резервные. Всегда 0

7.3.9 Группы регистров устройства управления

Группа содержит 32 регистра, однако не все регистры используются. Часть регистров зарезервирована. Все регистры группы выполняют специальные функции. Особенностью данной группы является то, что регистры доступны только как однословные.

Таблица 34 – Группа регистров устройства управления

Имя	Описание	Адрес	Значение по умолчанию
CJMP	Вычисленный адрес перехода	0x1E 0340	Не определено
-	-	0x1E 0341	-
RETI	Возврат из прерывания	0x1E 0342	Не определено
RETIB	Псевдоним RETI, для вложенных прерываний	0x1E 0343	Не определено
RETS	Возврат из исключительной ситуации	0x1E 0344	Не определено
DBGE	Возврат из эмуляции	0x1E 0345	Не определено
-	-	0x1E 0346–0x1E 0347	-
LC0	Счетчик циклов 0	0x1E 0348	Не определено
LC1	Счетчик циклов 1	0x1E 0349	Не определено
-	-	0x1E 034A–0x1E 034F	-
IVSW	Адрес-вектор для программных исключений	0x1E 0350	Не определено
-	-	0x1E 0351–0x1E 0353	-
FLAGREG	Регистр управления флагами процессора	0x1E 0354	0x0000 0000
FLAGREGST	Установка бит регистра флага	0x1E 0355	Не определено
FLAGREGCL	Сброс бит регистра флага	0x1E 0356	Не определено
-	-	0x1E 0357	-
SQCTL	Регистр управления	0x1E 0358	0x0000 0004
SQCTLST	Установка бит регистра управления	0x1E 0359	Не определено
SQCTLCL	Сброс бит регистра управления	0x1E 035A	Не определено
SQSTAT	Регистр статуса, только чтение	0x1E 035B	0x0000 FF04
SFREG	Регистр специальных статических флагов	0x1E 035C	0x0000 0000
-	-	0x1E 035D	-
EXT_FUN	Включение дополнительных функций	0x1E 035E	0
-	-	0x1E 035F	-

7.3.9.1 Регистр управления флагом (FLAGREG)

Регистр FLAGREG – 32-битный регистр, который контролирует направление вывода флага (вход или выход) и обеспечивает значение (1 или 0), когда вывод сконфигурирован как выход. Регистр FLAGREG (значение сброса = 0x0000 0000) описывается ниже (Таблица 35).

Таблица 35 – Биты регистра управления флагом

Бит	Имя	Назначение
3:0	FLAGx_EN	Направление флага 0 – прием 1 – выдача
7:4	FLAGx_OUT	Значение флага при выдаче
31-8	-	Не используются и всегда равны нулю

Вывод FLAG0 использует биты 0 и 4, FLAG1 использует биты 1 и 5, FLAG2 использует биты 2 и 6, FLAG3 использует биты 3 и 7.

7.3.9.2 Регистр управления SQCTL

Устройство управления последовательностью команд контролируется значением регистра SQCTL. После сброса значение регистра равно 0x0000_0004. Подробное описание всех бит регистра приведено ниже (Таблица 36).

Таблица 36 – Биты регистра управления SQCTL

Бит	Имя	Назначение
0	-	
1	-	
2	GIE	Глобальное разрешение прерываний: 1 – прерывания разрешены 0 – запрещены
3	SWIE	Разрешение генерации исключительных ситуаций (программных прерываний): 1 – разрешено 0 – запрещено
4	-	
5	-	
6	-	
7	KUSP	Выбор указателя стека в режиме супервизора (NMOD=1) при включенном режиме работы с раздельными указателями (USP=1): 0 – используется регистр KSP; 1 – используется регистр USP
8	DBGEN	Разрешение отладки: 1 – разрешено 0 – запрещено
9	NMOD	Режим работы: 0 – пользователь 1 – супервизор
10	TRCBEN	Включение буфера трасс: 0 – буфер трасс выключен и хранит старое значение 1 – включен и следит за исполнением переходов.
11	TRCBEXEN	Разрешение генерации исключительной ситуации от буфера трасс: 0 – запрещено 1 – разрешено

12	DF_IEEE	Разрешает использование команд обработки данных с плавающей запятой двойной точности. 0 – используется расширенная точность. 1 – код команды расширенной точности соответствует коду команды двойной точности
13	BYTE_ON	Разрешение использование кодов команды TRAP 32-63 в качестве кодов префикса для создания новых команд. 0 – команды TRAP 32-63 используются для генерации исключений. 1 – команды TRAP 32-63 используются в качестве префикса.
14	USP	Разрешение использования двух указателей стеков (регистр JK27). 0 – регистр JK27 один для пользователя и супервизора 1 – для пользователя и супервизора используются разные регистры
31-15	-	Зарезервированы и не используются. При чтении всегда равны нулю

Биты 12, 13, 14 регистра управления соответствуют дополнительным возможностям процессора и их включение должно быть дополнительно разрешено битом регистра EXT_FUN.

7.3.9.3 Регистр управления (SQCTLST). Установка бит

Бит SQCTLST является псевдонимом для SQCTL. При записи по данному адресу, ‘1’ в любом бите записываемых данных устанавливает соответствующий бит в SQCTL, тогда как ‘0’ в записываемых данных не меняет значение бита.

7.3.9.4 Регистр управления (SQCTLCL). Сброс бит

Бит SQCTLCL является псевдонимом для записи в SQCTL. При записи по данному адресу ‘0’ в любом бите записываемых данных сбрасывает соответствующий бит в SQCTL, тогда как ‘1’ в записываемых данных не меняет значение бита.

7.3.9.5 Регистр статических флагов (SFREG)

Статические флаги используются как статические копии условий. При желании сохранить значение условия, до того, как другая команда поменяет его значение, можно скопировать значение в регистр SFREG и использовать его позже как условие. Все статические флаги условий группируются в регистре SFREG. Регистр SFREG после сброса равен нулю и описание его разрядов приведено ниже (Таблица 37).

Таблица 37 – Биты регистра статических флагов

Бит	Имя	Назначение
0	GSCF0	Статический флаг целочисленных АЛУ
1	GSCF1	Статический флаг целочисленных АЛУ
2	XSCF0	Статический флаг вычислительного модуля X
3	XSCF1	Статический флаг вычислительного модуля X
4	YSCF0	Статический флаг вычислительного модуля У
5	YSCF1	Статический флаг вычислительного модуля У
31-6	-	Не используются и всегда равны нулю

7.3.9.6 Регистр включения расширенных функций (EXT_FUN)

Регистр состоит из одного бита, значение которого может быть прочитано в регистре SQSTAT, бит 30 – EXT_MODE. Этот бит по сбросу устанавливается в 0 (все расширенные функции запрещены) и меняет свое значение на противоположное всякий раз после выполнения команды 0x8B5E1A3C. Этот код суть команда пересылки из регистра EXT_FUN в этот же регистр EXT_FUN (EXT_FUN=EXT_FUN).

7.3.9.7 Регистр статуса устройства управления (SQSTAT)

Данный регистр доступен только для чтения и содержит информацию о текущем статусе устройства управления. Подробное описание разрядов регистра приведено ниже (Таблица 38).

Таблица 38 – Биты регистра SQSTAT

Бит	Имя	Назначение
1:0	MODE	Текущий режим работы процессора: 00 – пользователь 01 – супервизор 11 – эмулятор Другие состояния невозможны
2	IDLE	Процессор в состоянии ожидания прерывания (если 1)
7:3	SPVCMD	Значение параметра последней выполненной команды TRAP
11:8	EXCAUSE	Код последней исключительной ситуации: 0000 – команда TRAP 0001 – точка наблюдения или буфер трасс 0010 – команда плавающей запятой 0011 – ошибочная линия команд 0100 – невыровненный доступ 0101 – попытка записи защищенного регистра 0110 – переполнения счетчика статистики 0111 – ошибочный доступ в IALU 1000 – x 1001 – x 1010 – x 1011 – x 1100 – x 1101 – x 1110 – x 1111 – не было ошибок с момента сброса
15:12	EMCAUSE	Код ситуации, вызвавшей переход в режим эмулятора: 1111 – не было перехода в режим эмулятора после сброса 0000 – команда EMUTRAP 0001 – запрос JTAG 0010 – сработала точка наблюдения Все другие коды зарезервированы и не могут присутствовать
16	FLAG0	Состояние внешнего вывода FLAG0
17	FLAG1	Состояние внешнего вывода FLAG1
18	FLAG2	Состояние внешнего вывода FLAG2
19	FLAG3	Состояние внешнего вывода FLAG3

20	EXE_ISR	Флаг устанавливается в 1 при: - переходе процессора в состояние обработки прерывания, - записи данных в регистр RETIB. Установленный флаг запрещает прерывания. Флаг сбрасывается: - командой выхода из прерывания RTI, - командой RDS (если нет обработки исключительной ситуации), - чтением регистра RETIB. Сброшенный флаг означает возможность обработки нового прерывания процессором.
21	EXE_SWI	Признак того что процессор находится в состоянии обработки исключительной ситуации (когда равен 1)
22	EMUL	Признак того что процессор находится в режиме эмулятора (когда равен 1)
23	ISR_MODE	Флаг устанавливается в 1 при переходе процессора в состояние обработки прерывания. Флаг сбрасывается: - командой выхода из прерыванияRTI; - командой RSD (если нет обработки исключиткльной ситуации). Установленный флаг означает состояние обработки прерывания в режиме супервизора.
24	BTBEN	
25:27	-	
28	BTBEN	Буфер предсказания переходов включен (если 1), либо выключен (если 0)
29	-	
30	EXT_MODE	Флаг включения расширенных операций процессора. После аппаратного сброса значение 0 (расширенные операции запрещены)
31	I_LOCK	Флаг блокировки прерываний (1 – прерывания блокированы)

7.3.10 Группа регистров устройства защиты памяти.

Выше упоминалось, что процессор имеет несколько режимов работы и особенностью режима пользователя является то, что для него ограничен доступ к некоторым аппаратным ресурсам. Однако внутренняя и внешняя память одинаково доступны как для супервизора, так и для пользователя. Вместе с тем в ряде приложений желательно защитить код или данные системы от разрушения некорректным поведением пользовательской программы. Для этих целей в процессоре имеется группа регистров, с помощью которой можно описать отдельные регионы внутренней памяти и способы доступа к ним.

Все регистры данной группы доступны только как однословные. Запись к регистрам разрешена только если включен режим использования расширенных функций процессора (бит регистра EXT_FUN).

Таблица 39 – Группа регистра устройства защиты памяти

Имя	Описание	Адрес	Значение по умолчанию
PU0	Регистр защиты 0	0x1E 03E0	Не определено
PU1	Регистр защиты 1	0x1E 03E1	Не определено
PU2	Регистр защиты 2	0x1E 03E2	Не определено
PU3	Регистр защиты 3	0x1E 03E3	Не определено

PU4	Регистр защиты 4	0x1E 03E4	Не определено
PU5	Регистр защиты 5	0x1E 03E5	Не определено
PU6	Регистр защиты 6	0x1E 03E6	Не определено
PU7	Регистр защиты 7	0x1E 03E7	Не определено
-	-	0x1E 03E8–0x1E 03FB	0
PU_CR	Регистр управления	0x1E 03FC	0
-	-	0x1E 03FD–0x1E 03FF	0

7.3.10.1 Регистры защиты (Pix)

Каждый регистр защиты позволяет описать определенную область памяти и способы доступа к ней. Работа регистра защиты включается соответствующим битом в регистре управления. Подробное описание бит регистров защиты приведено ниже (Таблица 40).

Таблица 40 – Регистр PU

Бит	Имя	Назначение
10:0	STA	Начальный адрес модуля памяти
11	-	
22:12	ENDA	Конечный адрес модуля памяти
23	-	
25:24	JK_AP	Режим доступа к модулю со стороны шин J и K. 00 – полный доступ 01 – супервизор чтение и запись, пользователь чтение 10 – всем только чтение 11 – супервизор только чтение
27:26	I_AP	Режим доступа к модулю со стороны шины I. 00 – полный доступ 01 – только супервизор 10 – доступ запрещен 11 – доступ запрещен
29:28	H_AP	Режим доступа к модулю со стороны шины S (хост, DMA). 00 – чтение и запись 01 – только чтение 10 – доступ запрещен 11 – доступ запрещен
31:30	-	резерв

Поля STA и ENDA используются для сравнения со значениями адресных шин J, K, I, S. При этом в сравнении участвуют только биты 20:10 указанных шин. Проверка выполняется только при доступе к внутренней памяти. Так для K шины доступа попадание в некоторый модуль означает истинность следующего выражения:

Hit_PU = (K[20:10]>=STA) && (K[20:10]<=ENDA).

Минимальный размер модуля равен странице из 1 K слов. Модуль может рассматриваться как множество из 1 K страниц.

При нарушении прав доступа к модулю памяти вырабатывается исключительная ситуация.

7.3.10.2 Регистр управления (PU_CR)

Регистр осуществляет включение в работу всех функций модуля защиты. После сброса значение регистра равно нулю. Подробное описание бит регистра управления приведено ниже (Таблица 41).

Таблица 41 – Регистр PU_CR

Бит	Имя	Назначение
7:0	-	Зарезервировано. Биты общего назначения
8	PSD_fun	Управление генерацией исключительной ситуации при загрузке регистров устройства управления и модуля отладки из внешней памяти 0 – разрешена генерация 1 – исключительная ситуация только при нарушении прав доступа
9	SP_byte	Управление автоматическим сдвигом указателя стека и фрэйма при работе с байтами и короткими 0 – функция выключена 1 – функция включена
10	MIN_MAX	Включение функции поиска номера максимума-минимума 0 – выключено 1 – включено
11	LC_sup	Выключение анализа выхода из цикла 0 – разрешение использования процедуры анализа завершения цикла. Позволяет сократить время выхода из цикла 1 – при реализации циклов с помощью LC0, LC1 будут потери в скорости при выполнении последнего предсказания
15:12	-	Зарезервировано. Биты общего назначения
16	PU0_EN	Разрешение работы модуля защиты 0 1 – разрешено 0 – запрещено
17	PU1_EN	Разрешение работы модуля защиты 1 1 – разрешено 0 – запрещено
18	PU2_EN	Разрешение работы модуля защиты 2 1 – разрешено 0 – запрещено
19	PU3_EN	Разрешение работы модуля защиты 3 1 – разрешено 0 – запрещено
20	PU4_EN	Разрешение работы модуля защиты 4 1 – разрешено 0 – запрещено
21	PU5_EN	Разрешение работы модуля защиты 5 1 – разрешено 0 – запрещено
22	PU6_EN	Разрешение работы модуля защиты 6 1 – разрешено 0 – запрещено
23	PU7_EN	Разрешение работы модуля защиты 7 1 – разрешено 0 – запрещено
31:24	-	Зарезервировано. Биты общего назначения

7.3.11 Группы регистров контроллера прерываний

Данные группы регистров относятся к векторам прерываний, управлению прерываниями и регистрам статуса (Таблица 42 – Таблица 44). Все регистры контроллера прерываний доступны только как однословные.

7.3.11.1 Группы регистров векторов прерываний

Таблица 42 – Группа A регистров таблицы вектора прерываний

Имя	Описание	Адрес	Значение по умолчанию
IVKERNEL	Прерывание ядра VDK	0x1F 0300	Не определено
-	-	0x1F 0301	-
IVTIMER0LP	Низкий приоритет таймера #0	0x1F 0302	Не определено
IVTIMER1LP	Низкий приоритет таймера #1	0x1F 0303	Не определено
-	-	0x1F 0304– 0x1F 0305	-
IVLINK0	порт связи #0	0x1F 0306	Не определено
IVLINK1	порт связи #1	0x1F 0307	Не определено
IVLINK2	порт связи #2	0x1F 0308	Не определено
IVLINK3	порт связи #3	0x1F 0309	Не определено
-	-	0x1F 030A– 0x1F 030D	-
IVDMA0	Регистр DMA #0	0x1F 030E	0x0000 0000
IVDMA1	Регистр DMA #1	0x1F 030F	0x0000 0000
IVDMA2	Регистр DMA #2	0x1F 0310	0x0000 0000
IVDMA3	Регистр DMA #3	0x1F 0311	0x0000 0000
-	-	0x1F 0312– 0x1F 0315	-
IVDMA4	Регистр DMA #4	0x1F 0316	0x0000 0000
IVDMA5	Регистр DMA #5	0x1F 0317	0x0000 0000
IVDMA6	Регистр DMA #6	0x1F 0318	0x0000 0000
IVDMA7	Регистр DMA #7	0x1F 0319	0x0000 0000
-	-	0x1F 031A– 0x1F 031C	-
IVDMA8	Регистр DMA #8	0x1F 031D	0x0000 0000
IVDMA9	Регистр DMA #9	0x1F 031E	0x0000 0000
IVDMA10	Регистр DMA #10	0x1F 031F	0x0000 0000

Таблица 43 – Группа B регистров таблицы вектора прерываний

Имя	Описание	Адрес	Значение по умолчанию
IVDMA11	Регистр DMA #11	0x1F 0320	0x0000 0000
-	-	0x1F 0321– 0x1F 0324	-
IVDMA12	Регистр DMA #12	0x1F 0325	0x0000 0000
IVDMA13	Регистр DMA #13	0x1F 0326	0x0000 0000

-	-	0x1F 0327– 0x1F 0328	-
IVIRQ0	Вывод регистра nIRQ0	0x1F 0329	0x3000 0000
IVIRQ1	Вывод регистра nIRQ1	0x1F 032A	0x3800 0000
IVIRQ2	Вывод регистра nIRQ2	0x1F 032B	0x8000 0000
IVIRQ3	Вывод регистра nIRQ3	0x1F 032C	0x0000 0000
-	-	0x1F 032D– 0x1F 032F	-
VIRPT	Векторный регистр VIRPT	0x1F 0330	Не определено
-	-	0x1F 0331	-
IVBUSLK	Вектор блокировки шины	0x1F 0332	Не определено
-	-	0x1F 0333	-
IVTIMER0HP	Высокий приоритет таймера 0	0x1F 0334	Не определено
IVTIMER1HP	Высокий приоритет таймера 1	0x1F 0335	Не определено
-	-	0x1F 0336– 0x1F 0338	-
IVHW	Аппаратная ошибка	0x1F 0339	Не определено
-	-	0x1F 033A– 0x1F 033F	-

7.3.11.2 Группа регистров управления контроллером прерываний

Таблица 44 – Группа регистров управления прерываниями

Имя	Описание	Адрес	Значение по умолчанию
ILATL	ILAT младшие разряды	0x1F 0340	0x0000 0000
ILATH	ILAT старшие разряды	0x1F 0341	0x0000 0000
ILATSTL	ILAT младшие разряды, установка	0x1F 0342	Не определено
ILATSTH	ILAT старшие разряды, установка	0x1F 0343	Не определено
ILATCLL	ILAT младшие разряды, сброс	0x1F 0344	Не определено
ILATCLH	ILAT старшие разряды, сброс	0x1F 0345	Не определено
PMASKL	PMASK младшие разряды	0x1F 0346	0x0000 0000
PMASKH	PMASK старшие разряды	0x1F 0347	0x0000 0000
IMASKL	IMASK младшие разряды	0x1F 0348	0xE3C3 C000
IMASKH	IMASK старшие разряды	0x1F 0349	См. Описание
-	-	0x1F 034A– 0x1F 034D	-
INTCTL	Управление прерыванием	0x1F 034E	См. Описание
-	-	0x1F 034F	-
TIMER0L	Текущее значение таймера 0 младшие разряды	0x1F 0350	Не определено
TIMER0H	Текущее значение таймера 0. Старшие разряды	0x1F 0351	Не определено
TIMER1L	Текущее значение таймера 1. Младшие разряды	0x1F 0352	Не определено

TIMER1H	Текущее значение таймера 1 старшие разряды	0x1F 0353	Не определено
TMRIN0L	Значение инициализации таймера 0. Младшие разряды	0x1F 0354	Не определено
TMRIN0H	Значение инициализации таймера 0 старшие разряды	0x1F 0355	Не определено
TMRIN1L	Значение инициализации 1 младшие разряды	0x1F 0356	Не определено
TMRIN1H	Значение инициализации 1 старшие разряды	0x1F 0357	Не определено
-	-	0x1F 0358– 0x1F 035F	-

7.3.11.3 Регистр управления прерываниями (INTCTL)

Регистр INTCTL – 32-битный регистр, который управляет чувствительностью внешних входов прерываний nIRQ3-0 (прерываний чувствительных к фронту или уровню) и обеспечивает управление стартом и остановкой таймеров. Подробное описание разрядов регистра приведено ниже (Таблица 45).

Таблица 45 – Регистр INTCTL

Бит	Имя	Назначение
0	IRQ0_EDGE	Запрос прерывания по входу nIRQ0 вызывается: 0 – фронтом сигнала (из высокого в низкий) 1 – уровнем (низкий)
1	IRQ1_EDGE	Запрос прерывания по входу nIRQ1 вызывается: 0 – фронтом сигнала (из высокого в низкий) 1 – уровнем (низкий)
2	IRQ2_EDGE	Запрос прерывания по входу nIRQ2 вызывается: 0 – фронтом сигнала (из высокого в низкий) 1 – уровнем (низкий)
3	IRQ3_EDGE	Запрос прерывания по входу nIRQ3 вызывается: 0 – фронтом сигнала (из высокого в низкий) 1 – уровнем (низкий)
4	TMR0RN	Разрешение работы таймера 0: 0 – таймер остановлен 1 – таймер работает
5	TMR1RN	Разрешение работы таймера 1: 0 – таймер остановлен 1 – таймер работает
31:6	-	Не используются и при чтении всегда равны нулю.

Во время сброса процессора можно выбрать способ генерации прерывания для внешних входов nIRQ3-0. Это определяется внешним выводом nBM.

Если значение сигнала на выводе nBM =1 при сбросе, то значением сброса для INTCTL является 0x0000 000F.

Если значение сигнала на выводе nBM =0 при сбросе, то значением сброса для INTCTL является 0x0000 0000.

7.3.11.4 Регистры защелки прерываний (ILATL/ILATH)

Все запросы прерываний, которые поступают в контроллер прерываний фиксируются (запоминаются) в регистре ILAT. Регистр ILAT – это 64-битный регистр доступный только по чтению как два 32-битных регистра ILATH и ILATL:

- ILAT – младшая часть ILAT (биты 31:0);
- ILATH – старшая часть ILAT (биты 63:32).

Каждый бит соответствует одному прерыванию и устанавливается при появлении этого прерывания. Порядок битов запросов прерываний устанавливается в соответствии с приоритетами прерываний: 0 – самый низкий приоритет.

Регистр ILAT имеет псевдонимы для упрощения процедур установки и сброса бит регистра. Для установки используются регистры ILATSTL или ILATSTH, а для сброса ILATCLL или ILATCLH. Регистры-псевдонимы доступны только по записи. При установке записываемое значение 1 приводит к установке бита, ноль не изменяет значение. При сбросе запись нуля вызывает сброс, а запись единицы не изменяет значение.

7.3.11.5 Регистры маскирования прерываний (IMASKL/IMASKH)

Каждый бит запроса прерываний, зафиксированный в регистре ILAT имеет соответствующий ему бит разрешения обслуживания прерывания (или бит маскирования прерывания). Значение 1 в бите маски разрешает прохождение соответствующего ему запроса прерывания в ILAT для дальнейшей обработки.

Регистр IMASK – 64-битный регистр, доступный как два 32-битных регистра IMASKL и IMASKH. После сброса значение IMASKL всегда равно 0xE3C3C000. Значение регистра IMASKH после сброса зависит от состояния во время сброса внешнего вывода IRQEN и может быть равно 0x00010061 (nBM=0) либо 0x00011E61 (nBM=1).

7.3.11.6 Регистры приоритетного маскирования прерываний (PMASKL/PMASKH)

Регистр PMASK – 64-битный регистр, доступный как два 32-битных регистра PMASKL и PMASKH. Регистры доступны только по чтению и отображают состояние запросов прерываний, которые поступают на приоритетный шифратор контроллера прерываний.

Каждый бит регистра запроса прерываний соответствует определенному источнику прерываний и имеет соответствующий ему (с тем же номером) бит в регистре маскирования IMASK и бит регистре PMASK. Назначение бит всех этих регистров приведено ниже (Таблица 101, Таблица 102).

Таблица 46 – Биты регистров ILATL, IMASKL, PMASKL

Бит	Имя	Описание
0	INT_KERNEL	Прерывание ядра VDK
1	-	-
2	INT_TIMER0LP	Низкий приоритет таймера #0
3	INT_TIMER1LP	Низкий приоритет таймера #1
5:4	-	-
6	INT_LINK0	порт связи #0
7	INT_LINK1	порт связи #1
8	INT_LINK2	порт связи #2
9	INT_LINK3	порт связи #3

13:10	-	-
14	INT_DMA0	Регистр DMA #0
15	INT_DMA1	Регистр DMA #1
16	INT_DMA2	Регистр DMA #2
17	INT_DMA3	Регистр DMA #3
21:18	-	-
22	INT_DMA4	Регистр DMA #4
23	INT_DMA5	Регистр DMA #5
24	INT_DMA6	Регистр DMA #6
25	INT_DMA7	Регистр DMA #7
28:26	-	-
29	INT_DMA8	Регистр DMA #8
30	INT_DMA9	Регистр DMA #9
31	INT_DMA10	Регистр DMA #10

Таблица 47 – Биты регистров ILATH, IMASKH, PMASKH

Бит	Имя	Описание
0	INT_DMA11	Регистр DMA #11
4:1	-	-
5	INT_DMA12	Регистр DMA #12
6	INT_DMA13	Регистр DMA #13
8:7	-	-
9	INT_IRQ0	Вывод регистра nIRQ0
10	INT_IRQ1	Вывод регистра nIRQ1
11	INT_IRQ2	Вывод регистра nIRQ2
12	INT_IRQ3	Вывод регистра nIRQ3
15:13	-	-
16	INT_VIRPT	Векторный регистр VIRPT
17	-	-
18	INT_BUSLOCK	Вектор блокировки шины
19	-	-
20	INT_TIMER0HP	Высокий приоритет таймера 0
21	INT_TIMER1HP	Высокий приоритет таймера 1
24:22	-	-
25	INT_HWERR	Аппаратная ошибка
31:26	-	-

7.3.11.7 Регистры таймера (TIMERxH/L)

Регистры TIMERx – одинарные 64-битные регистры доступные как два 32-битных регистра TIMERxL и TIMERxH. Регистры TIMERx содержат текущее значение для таймеров:

- TIMER0L Таймер 0 младшая часть текущего значения, только для чтения;
- TIMER0H Таймер 0 старшая часть текущего значения, только для чтения;
- TIMER1L Таймер 1 младшая часть текущего значения, только для чтения;
- TIMER1H Таймер 1 старшая часть текущего значения, только для чтения.

7.3.11.8 Регистры начального значения таймеров (TMRINxH/L)

Регистры TMRINx – одинарные 64-битные регистры доступные как два 32-битных регистра TMRINxL и TMRINxH. Регистры TMRINx содержат начальные значения для таймеров:

- TMRIN0L Таймер 0 младшая часть начального значения;
- TMRIN0H Таймер 0 старшая часть начального значения;
- TMRIN1L Таймер 1 младшая часть начального значения;
- TMRIN1H Таймер 1 старшая часть начального значения.

7.3.12 Группа регистров управления и статуса DMA

Общее описание группы регистров управления и статуса приведено ниже (Таблица 48). Далее более подробно рассматривается каждый регистр в отдельности.

Таблица 48 – Группа регистров управления DMA

Имя	Описание	Адрес	Значение по умолчанию
DCNT	Управление DMA	0x1F 0060	0x0000 0000
-	-	0x1F 0061– 0x1F 0063	-
DCNTST	Управление DMA, биты установки	0x1F 0064	Не определено
-	-	0x1F 0065– 0x1F 0067	-
DCNTCL	Управление DMA, биты сброса	0x1F 0068	Не определено
-	-	0x1F 0069– 0x1F 006B	-
DSTAT	Статус DMA	0x1F 006C	0x0000 0000
-	-	0x1F 006D– 0x1F 006F	-
DSTATCL	Статус DMA, биты сброса	0x1F 0070	Не определено
-	-	0x1F 0071– 0x1F 007F	-

7.3.12.1 Регистр управления DMA (DCNT)

Биты регистра управления выполняют одну единственную функцию приостановки работы соответствующего им канала DMA. Подробное описание разрядов регистра управления приведено ниже (Таблица 49).

Таблица 49 – Регистр DCNT

Бит	Имя	Описание
0	DMA0_P	Пауза (если 1) для канала 0 DMA
1	DMA1_P	Пауза (если 1) для канала 1 DMA
2	DMA2_P	Пауза (если 1) для канала 2 DMA
3	DMA3_P	Пауза (если 1) для канала 3 DMA
4	DMA4_P	Пауза (если 1) для канала 4 DMA
5	DMA5_P	Пауза (если 1) для канала 5 DMA
6	DMA6_P	Пауза (если 1) для канала 6 DMA
7	DMA7_P	Пауза (если 1) для канала 7 DMA
9:8		
10	DMA8_P	Пауза (если 1) для канала 8 DMA
11	DMA9_P	Пауза (если 1) для канала 9 DMA
12	DMA10_P	Пауза (если 1) для канала 10 DMA
13	DMA11_P	Пауза (если 1) для канала 11 DMA
15:14		
16	DMA12_P	Пауза (если 1) для канала 12 DMA
17	DMA13_P	Пауза (если 1) для канала 13 DMA
31:18		

Регистр управления DMA имеет три адреса:
- DCNT – регулярный адрес для чтения и записи.
- DCNTST – адрес для установки бит. Доступен только по записи. Запись 1 устанавливает соответствующий бит, а запись 0 не изменяет его значения.
- DCNTCL – адрес для очистки бит. Доступен только по записи. Запись 0 сбрасывает значение бита, а запись 1 не изменяет его значение.

Назначение бита паузы состоит в том, что с его установкой в 1 соответствующий канал DMA приостанавливает свою работу и возобновляет её только после того как бит станет равен 0.

Регистры управления DMA (DCNT, DCNTST, DCNTCL) могут быть доступны как нормальное, длинное и счетверенное слово.

7.3.12.2 Регистр статуса DMA (DSTAT)

Регистр статуса DSTAT предназначен только для чтения и отражает текущее состояние каналов контроллера. Регистр имеет разрядность 64 бита и два адреса доступа. Второму адресу соответствует имя DSTATCL регистра. При чтении регистра DSTATCL все коды ошибок в регистре статуса очищаются и состояние канала изменяется на состояние выключено. Регистры статуса DMA (DSTAT, DSTATCL) могут быть доступны только как двойное и счетверенное слово. В регистре состояния каждому каналу контроллера отведено 3 бита. С помощью этих бит кодируется состояние канала. Возможны следующие значения состояния канала:
- 000 – выключен;
- 001 – работает;
- 010 – завершил работу;
- 011 – резерв;
- 100 – ошибка. Инициализация активного канала;
- 101 – ошибка. Запрещенная конфигурация канала;
- 110 – резерв;
- 111 – ошибка. Некорректный адрес.

Назначение разрядов регистра состояния приведено ниже (Таблица 50).

Таблица 50 – Регистр DSTAT

Бит	Имя	Описание
2:0	CH0	Состояние канала 0
5:3	CH1	Состояние канала 1
8:6	CH2	Состояние канала 2
11:9	CH3	Состояние канала 3
14:12	CH4	Состояние канала 4
17:15	CH5	Состояние канала 5
20:18	CH6	Состояние канала 6
23:21	CH7	Состояние канала 7
31:24	-	-
34:32	CH8	Состояние канала 8

37:35	CH9	Состояние канала 9
40:38	CH10	Состояние канала 10
43:41	CH11	Состояние канала 11
49:44	-	-
52:50	CH12	Состояние канала 12
55:53	CH13	Состояние канала 13
63:56	-	-

7.3.13 Группа регистров ТСВ каналов 0-3 DMA

Доступ к регистрам данной группы осуществляется только как к счетверенным словам, т.е. операции выполняются сразу с четырьмя регистрами.

Таблица 51 – Группа регистров ТСВ каналов 0-3 DMA

Имя	Описание	Адрес	Значение по умолчанию
DCS0:DI	канал 0, адрес источника	0x1F 0000	0x0000 0000
DCS0:DX	канал 0, модификатор X источника	0x1F 0001	0x0100 0004
DCS0:DY	канал 0, модификатор У источника	0x1F 0002	0x0000 0000
DCS0:DP	канал 0, управление источником	0x1F 0003	0xD300 0000
DCD0:DI	канал 0, адрес приемника	0x1F 0004	0x0000 0000
DCD0:DX	канал 0, модификатор X приемника	0x1F 0005	0x0100 0004
DCD0:DY	канал 0, модификатор У приемника	0x1F 0006	0x0000 0000
DCD0:DP	канал 0, управление приемником	0x1F 0007	0x5300 0000
DCS1:DI	канал 1, адрес источника	0x1F 0008	Не определено
DCS1:DX	канал 1, модификатор X источника	0x1F 0009	Не определено
DCS1:DY	канал 1, модификатор У источника	0x1F 000A	Не определено
DCS1:DP	канал 1, управление источником	0x1F 000B	Не определено
DCD1:DI	канал 1, адрес приемника	0x1F 000C	Не определено
DCD1:DX	канал 1, модификатор X приемника	0x1F 000D	Не определено
DCD1:DY	канал 1, модификатор У приемника	0x1F 000E	Не определено
DCD1:DP	канал 1, управление приемником	0x1F 000F	Не определено
DCS2:DI	канал 2, адрес источника	0x1F 0010	Не определено
DCS2:DX	канал 2, модификатор X источника	0x1F 0011	Не определено
DCS2:DY	канал 2, модификатор У источника	0x1F 0012	Не определено
DCS2:DP	канал 2, управление источником	0x1F 0013	Не определено
DCD2:DI	канал 2, адрес приемника	0x1F 0014	Не определено
DCD2:DX	канал 2, модификатор X приемника	0x1F 0015	Не определено
DCD2:DY	канал 2, модификатор У приемника	0x1F 0016	Не определено
DCD2:DP	канал 2, управление приемником	0x1F 0017	Не определено
DCS3:DI	канал 3, адрес источника	0x1F 0018	Не определено
DCS3:DX	канал 3, модификатор X источника	0x1F 0019	Не определено

DCS3:DY	канал 3, модификатор У источника	0x1F 001A	Не определено
DCS3:DP	канал 3, управление источником	0x1F 001B	Не определено
DCD3:DI	канал 3, адрес приемника	0x1F 001C	Не определено
DCD3:DX	канал 3, модификатор Х приемника	0x1F 001D	Не определено
DCD3:DY	канал 3, модификатор У приемника	0x1F 001E	Не определено
DCD3:DP	канал 3, управление приемником	0x1F 001F	Не определено

Из таблицы выше (Таблица 51) видно, что только канал 0 имеет определенное начальное состояние. При этом активное состояние может быть установлено при сбросе только если выбран режим загрузки с внешнего EPROM.

7.3.14 Группа регистров ТСВ порта связи. Каналы передачи

Доступ к регистрам данной группы осуществляется только как к счетверенным словам, т.е. операции выполняются сразу с четырьмя регистрами.

В отличие от каналов 0-3, где источник и приемник задается, каналы 4-7 жестко закреплены за соответствующими им передатчиками порта связи. В этом случае необходимо определить только параметры источника.

Таблица 52 – Группа регистров ТСВ порта связи. Передача

Имя	описание	адрес	Значение по умолчанию
DC4:DI	канал 4, адрес источника	0x1F 0020	Не определено
DC4:DX	канал 4, модификатор Х источника	0x1F 0021	Не определено
DC4:DY	канал 4, модификатор У источника	0x1F 0022	Не определено
DC4:DP	канал 4, управление источником	0x1F 0023	Не определено
DC5:DI	канал 5, адрес источника	0x1F 0024	Не определено
DC5:DX	канал 5, модификатор Х источника	0x1F 0025	Не определено
DC5:DY	канал 5, модификатор У источника	0x1F 0026	Не определено
DC5:DP	канал 5, управление источником	0x1F 0027	Не определено
DC6:DI	канал 6, адрес источника	0x1F 0028	Не определено
DC6:DX	канал 6, модификатор Х источника	0x1F 0029	Не определено
DC6:DY	канал 6, модификатор У источника	0x1F 002A	Не определено
DC6:DP	канал 6, управление источником	0x1F 002B	Не определено
DC7:DI	канал 7, адрес источника	0x1F 002C	Не определено
DC7:DX	канал 7, модификатор Х источника	0x1F 002D	Не определено
DC7:DY	канал 7, модификатор У источника	0x1F 002E	Не определено
DC7:DP	канал 7, управление источником	0x1F 002F	Не определено
-	-	0x1F0030–0x1F003F	-

7.3.15 Группа регистров ТСВ порта связи. Каналы приема.

Доступ к регистрам данной группы осуществляется только как к счетверенным словам, т.е. операции выполняются сразу с четырьмя регистрами.

В отличие от каналов 4-7, каналы 8-11 жестко закреплены за соответствующими им приемниками порта связи. В этом случае необходимо определить параметры приемника информации. К этой группе относятся также и каналы 12 и 13 DMA.

Таблица 53 – Группа регистров ТСВ порта связи. Прием

Имя	Описание	Адрес	Значение по умолчанию
DC8:DI	канал 8, адрес приемника	0x1F 0040	0x0000 0000
DC8:DX	канал 8, модификатор X приемника	0x1F 0041	0x0100 0004
DC8:DY	канал 8, модификатор У приемника	0x1F 0042	0x0000 0000
DC8:DP	канал 8, управление приемником	0x1F 0043	0x5780 0000
DC9:DI	канал 9, адрес приемника	0x1F 0044	0x0000 0000
DC9:DX	канал 9, модификатор X приемника	0x1F 0045	0x0100 0004
DC9:DY	канал 9, модификатор У приемника	0x1F 0046	0x0000 0000
DC9:DP	канал 9, управление приемником	0x1F 0047	0x5780 0000
DC10:DI	канал 10, адрес приемника	0x1F 0048	0x0000 0000
DC10:DX	канал 10, модификатор X приемника	0x1F 0049	0x0100 0004
DC10:DY	канал 10, модификатор У приемника	0x1F 004A	0x0000 0000
DC10:DP	канал 10, управление приемником	0x1F 004B	0x5780 0000
DC11:DI	канал 11, адрес приемника	0x1F 004C	0x0000 0000
DC11:DX	канал 11, модификатор X приемника	0x1F 004D	0x0100 0004
DC11:DY	канал 11, модификатор У приемника	0x1F 004E	0x0000 0000
DC11:DP	канал 11, управление приемником	0x1F 004F	0x5780 0000
-	-	0x1F 0050– 0x1F 0057	-
DC12:DI	канал 12, адрес приемника	0x1F 0058	0x0000 0000
DC12:DX	канал 12, модификатор X приемника	0x1F 0059	0x0100 0001
DC12:DY	канал 12, модификатор У приемника	0x1F 005A	0x0000 0000
DC12:DP	канал 12, управление приемником	0x1F 005B	0x5380 0000
DC13:DI	канал 13, адрес приемника	0x1F 005C	0x0000 0000
DC13:DX	канал 13, модификатор X приемника	0x1F 005D	0x0100 0001
DC13:DY	канал 13, модификатор У приемника	0x1F 005E	0x0000 0000
DC13:DP	канал 13, управление приемником	0x1F 005F	0x5380 0000

Каналы 12 и 13 называются autoDMA каналами в связи с тем, что при выполнении записи данных в них ведущим устройством, они автоматически выполняют пересылку принятых данных в запрограммированный в них приемник.

7.3.16 Группа регистров AutoDMA

Регистры данной группы относятся к каналам 12 и 13 DMA и используются для реализации режима «подчиненный» (slave) DMA. Данные регистры могут быть доступны только через мультипроцессорное адресное пространство.

Таблица 54 – Группа регистров AUTO DMA

Имя	Описание	Адрес	Значение по умолчанию
AUTODMA0:0	Регистр AutoDMA 0, разряды данных 31–0	0x1F 03E0	Не определено
AUTODMA0:1	Регистр AutoDMA 0, разряды данных 63–32	0x1F 03E1	Не определено
AUTODMA0:2	Регистр AutoDMA 0, разряды данных 95–64	0x1F 03E2	Не определено
AUTODMA0:3	Регистр AutoDMA 0, разряды данных 127–96	0x1F 03E3	Не определено
AUTODMA1:0	Регистр AutoDMA 1, разряды данных 31–0	0x1F 03E4	Не определено
AUTODMA1:1	Регистр AutoDMA 1, разряды данных 63–32	0x1F 03E5	Не определено
AUTODMA1:2	Регистр AutoDMA 1, разряды данных 95–64	0x1F 03E6	Не определено
AUTODMA1:3	Регистр AutoDMA 1, разряды данных 127–96	0x1F 03E7	Не определено
-	-	0x1F 03E8– 0x1F 03FF	-

Регистры данных канала 0 AutoDMA содержат 32-, 64-, или 128-битные данные. При записи в этот регистр, канал 12 DMA передает записанные данные по адресу внутренней памяти, запрограммированному в DMA. Этот регистр не может читаться и может быть записан только через мультипроцессорное пространство адресов. Если канал 12 не инициализирован, данные теряются.

Регистры данных канала 1 AutoDMA содержат 32-, 64-, или 128-битные данные. При записи в этот регистр, канал 13 DMA передает записанные данные по адресу внутренней памяти, запрограммированному в DMA. Этот регистр не может читаться и может быть записан только через мультипроцессорное пространство адресов. Если канал 13 не инициализирован, данные теряются.

7.3.16.1 TCB регистры

адреса (_I),

модификатора X (_X),

модификатора Y (_Y),

управления (_P)

Каждый канал DMA управляется одним или двумя TCB регистрами, состоящими из четырех слов. Каждое из слов имеет определенное назначение:

- Регистр адреса (_I)

Это 32-битный регистр который содержит адрес источника данных либо приемника данных и используется для адресации внутренней или внешней памяти.

- Регистр модификатора X (_Xx)
  Этот 32-битный регистр содержит в своих 16-ти младших разрядах (биты 15:0) значение на которое изменяется адрес после каждого цикла обмена, а в старших 16-ти разрядах (биты 31:16) содержится количество слов, которые канал должен передать. Есть режимы обмена, когда значение модификации занимает 22 бита (биты 21:0), а счетчик числа циклов 10 бит (биты 31:22). При включенном двухмерном (2D) режиме обмена структура регистра такая же, но выполняемые изменения относятся к X координате.
- Регистр модификатора Y (_Yx)
  Данный регистр используется вместе с модификатором _X при включенном двухмерном режиме. Он содержит 16\22-битное значение изменений и 16\10-битное значение счетчика, но изменения относятся к оси У.
- Регистр управления каналом (_Px)
  Данный регистр задает режим работы канала. Подробное описание разрядов регистра приведено ниже (Таблица 55). Режимы работы каналов подробно описаны в подразделе контроллера DMA.

Таблица 55 – Регистр DP

Бит	Имя	Назначение
18:0	CHPT	Адрес-указатель на следующую цепочку обмена
21:19	CHTG	Канал-приемник следующей цепочки: 000 – канал связи 0, приемник 001 – канал связи 1, приемник 010 – канал связи 2, приемник 011 – канал связи 3, приемник 100 – канал связи 0, передатчик 101 – канал связи 1, передатчик 110 – канал связи 2, передатчик 111 – канал связи 3, передатчик
22	CHEN	Разрешение загрузки следующей цепочки: 1 – разрешено 0 – запрещено
23	DRQ	Разрешение анализа внешнего запроса: 1 – разрешено 0 – запрещено Если разрешено, то канал выполняет цикл обмена только по внешнему запросу. Если запрещено, то канал выполняет обмены без ожидания запроса.
24	INT	Генерация запроса прерывания после окончания работы канала: 1 – разрешено 0 – запрещено
26:25	LEN	Длина передаваемых данных в одном цикле обмена: 00 – резерв 01 – слово 32 бита 10 – длинное слово 64 бита 11 – квадрослово 128 бит
27	2D	Включение режима 2-х мерной пересылки: 1 – двумерная пересылка 0 – одномерная пересылка

28	PR	Приоритет циклов обмена: 1 – высокий 0 – обычный
31:29	TY	Тип обмена (выбор источника либо приемника): 000 – канал выключен 001 – линк порт 010 – внутренняя память (16\16) 011 – внутренняя память (22\10) 100 – внешняя память (16\16) 101 – внешнее устройство Flyby 110 – загрузочное EPROM 111 – внешняя память (22\10)

7.3.17 Группа регистров портов связи

Регистры буфера портов связи доступны как счетверенные слова. Две группы регистров данного типа описаны в таблицах 2-36 и 2-37. Регистры управления и статуса доступны как одинарные слова. Базовый адрес для всех регистров LINK-портов 0x001F0000.

Таблица 56 – Группа регистров буфера приема/передачи порта связи

Имя	Описание	Адрес	Значение по умолчанию
LBUFTX0:0	Данные передатчика порта связи 0	0x1F 04A0	Не определено
LBUFTX0:1	Данные передатчика порта связи 0	0x1F 04A1	Не определено
LBUFTX0:2	Данные передатчика порта связи 0	0x1F 04A2	Не определено
LBUFTX0:3	Данные передатчика порта связи 0	0x1F 04A3	Не определено
LBUFRX0:0	Данные приемника порта связи 0. Только чтение	0x1F 04A4	Не определено
LBUFRX0:1	Данные приемника порта связи 0. Только чтение	0x1F 04A5	Не определено
LBUFRX0:2	Данные приемника порта связи 0. Только чтение	0x1F 04A6	Не определено
LBUFRX0:3	Данные приемника порта связи 0. Только чтение	0x1F 04A7	Не определено
LBUFTX1:0	Данные передатчика порта связи 1	0x1F 04A8	Не определено
LBUFTX1:1	Данные передатчика порта связи 1	0x1F 04A9	Не определено
LBUFTX1:2	Данные передатчика порта связи 1	0x1F 04AA	Не определено
LBUFTX1:3	Данные передатчика порта связи 1	0x1F 04AB	Не определено
LBUFRX1:0	Данные приемника порта связи 1. Только чтение	0x1F 04AC	Не определено
LBUFRX1:1	Данные приемника порта связи 1. Только чтение	0x1F 04AD	Не определено
LBUFRX1:2	Данные приемника порта связи 1. Только чтение	0x1F 04AE	Не определено
LBUFRX1:3	Данные приемника порта связи 1. Только чтение	0x1F 04AF	Не определено
LBUFTX2:0	Данные передатчика порта связи 2	0x1F 04B0	Не определено
LBUFTX2:1	Данные передатчика порта связи 2	0x1F 04B1	Не определено

LBUFTX2:2	Данные передатчика порта связи 2	0x1F 04B2	Не определено
LBUFTX2:3	Данные передатчика порта связи 2	0x1F 04B3	Не определено
LBUFRX2:0	Данные приемника порта связи 2. Только чтение	0x1F 04B4	Не определено
LBUFRX2:1	Данные приемника порта связи 2. Только чтение	0x1F 04B5	Не определено
LBUFRX2:2	Данные приемника порта связи 2. Только чтение	0x1F 04B6	Не определено
LBUFRX2:3	Данные приемника порта связи 2. Только чтение	0x1F 04B7	Не определено
LBUFTX3:0	Данные передатчика порта связи 3	0x1F 04B8	Не определено
LBUFTX3:1	Данные передатчика порта связи 3	0x1F 04B9	Не определено
LBUFTX3:2	Данные передатчика порта связи 3	0x1F 04BA	Не определено
LBUFTX3:3	Данные передатчика порта связи 3	0x1F 04BB	Не определено
LBUFRX3:0	Данные приемника порта связи 3. Только чтение	0x1F 04BC	Не определено
LBUFRX3:1	Данные приемника порта связи 3. Только чтение	0x1F 04BD	Не определено
LBUFRX3:2	Данные приемника порта связи 3. Только чтение	0x1F 04BE	Не определено
LBUFRX3:3	Данные приемника порта связи 3. Только чтение	0x1F 04BF	Не определено

Таблица 57 – Группа регистров статуса и контроля порта связи

Имя	Описание	Адрес	Значение по умолчанию
LRCTL0	управление приемника порта связи 0	0x1F 00E0	0x0000 0001 или 0x0000 0011
LRCTL1	управление приемника порта связи 1	0x1F 00E1	0x0000 0001 или 0x0000 0011
LRCTL2	управление приемника порта связи 2	0x1F 00E2	0x0000 0001 или 0x0000 0011
LRCTL3	управление приемника порта связи 3	0x1F 00E3	0x0000 0001 или 0x0000 0011
LTCTL0	управление передатчика порта связи 0	0x1F 00E4	0x0000 0000
LTCTL1	управление передатчика порта связи 1	0x1F 00E5	0x0000 0000
LTCTL2	управление передатчика порта связи 2	0x1F 00E6	0x0000 0000
LTCTL3	управление передатчика порта связи 3	0x1F 00E7	0x0000 0000
-	-	0x1F 00E6–0x1F 00EF	Нет данных
LRSTAT0	статус приемника порта связи 0	0x1F 00F0	0x0000 0000
LRSTAT1	статус приемника порта связи 1	0x1F 00F1	0x0000 0000
LRSTAT2	статус приемника порта связи 2	0x1F 00F2	0x0000 0000
LRSTAT3	статус приемника порта связи 3	0x1F 00F3	0x0000 0000
LTSTAT0	статус передатчика порта связи 0	0x1F 00F4	0x0000 0002
LTSTAT1	статус передатчика порта связи 1	0x1F 00F5	0x0000 0002

LTSTAT2	статус передатчика порта связи 2	0x1F 00F6	0x0000 0002
LTSTAT3	статус передатчика порта связи 3	0x1F 00F7	0x0000 0002
LRSTATC0	сброс статуса приемника порта связи 0	0x1F 00F8	Не определено
LRSTATC1	сброс статуса приемника порта связи 1	0x1F 00F9	Не определено
LRSTATC2	сброс статуса приемника порта связи 2	0x1F 00FA	Не определено
LRSTATC3	сброс статуса приемника порта связи 3	0x1F 00FB	Не определено
LTSTATC0	сброс статуса передатчика порта связи 0	0x1F 00FC	Не определено
LTSTATC1	сброс статуса передатчика порта связи 1	0x1F 00FD	Не определено
LTSTATC2	сброс статуса передатчика порта связи 2	0x1F 00FE	Не определено
LTSTATC3	сброс статуса передатчика порта связи 3	0x1F 00FF	Не определено

7.3.17.1 Регистр управления приемником порта связи (LRCTLx)

Регистры LRCTLx определяют параметры приема для порта связи. При этом ширина порта связи может быть 4 или 1 бит. Имеется возможность задать ширину порта связи во время сброса. Значение сброса для регистров LRCTLx равно 0x0000_0001, если при сбросе значение на выводе TMR0E = 0 (RDSIZE = 0). Если при сбросе значение на выводе TMR0E = 1, то значение сброса для регистров LRCTLx является 0x0000 0011 (RDSIZE=1). Подробное описание разрядов регистра приведено ниже (Таблица 58).

Таблица 58 – Регистр LRCTL

Бит	Имя	Назначение
0	REN	Бит включения приемника: 1 – включен 0 – выключен
1	RVERE	Разрешение контроля при приеме 1 – разрешено 0 – запрещено
2	RTOE	Разрешение прерывания в случае ситуации time out
3	RBCMPE	Разрешение анализа входа nLxBCMPI: 1 – разрешено 0 – запрещено
4	RDSIZE	Размер шины приема: 1 – 4 бита 0 – 1 бит
5	ROVRE	Разрешение прерывания при переполнении буфера приемника: 1 – разрешено 0 – запрещено
6	RINIF	Разрешение проведения процедуры инициализации: 1 – выполнить инициализацию с помощью RINIV значения 0 – не выполнять инициализацию
7	RINIV	Значение данных которые используются при процедуре инициализации
31:8	-	зарезервировано

7.3.17.2 Регистр управления передатчиком порта связи (LTCTLx)

Регистры устанавливают параметры передачи для порта связи. Значением сброса для регистров LTCTLx является 0. Подробное описание разрядов регистра приведено ниже (Таблица 59).

Таблица 59 – Регистр LTCTL

Бит	Имя	Назначение
0	TEN	Бит включения передатчика: 1 – включен 0 – выключен
1	TVERE	Разрешение формирования контрольной суммы при передаче: 1 – разрешено 0 – запрещено
2	TTOE	Разрешение прерывания в случае ситуации time out
3	TBCMPE	Разрешение формирования выхода nLxBCMPO: 1 – разрешено 0 – запрещено
4	TDSIZE	Размер шины передачи: 1 – 4 бита 0 – 1 бит
7:5	SPD	Частота передачи LxCLKOUT равна: 000 – CCLK 001 – CCLK деленный на 1.5 010 – CCLK деленный на 2 100 – CCLK деленный на 4 Другие значения зарезервированы.
31:8	-	зарезервировано

7.3.17.3 Регистр состояния приемника порта связи (LRSTATx)

Регистры LRSTATx указывают статус приемника порта связи. Значением сброса для регистров LRSTATx является 0x0000 0000. Подробное описание разрядов регистра приведено ниже (Таблица 60).

Таблица 60 – Регистр LRSTAT

Бит	Имя	Назначение
1:0	RSTAT	Состояние буфера приемника: 00 – пуст 01 – значение буфера достоверно 11 – буфер полон 10 – буфер не готов (принимает данные)
2	RTER	1 – приемник обнаружил ситуацию time out. 0 – нет ошибки
3	RWER	1 – приемник обнаружил ошибку записи 0 – нет ошибки
4	RCSER	1 – обнаружена ошибка контрольной суммы 0 – нет ошибки
5	ROVER	1 – ошибка переполнения приемника 0 – нет ошибки
6	RINIT	Состояние инициализации приемника: 1 – проинициализирован 0 – не инициализировался
31:7	-	Всегда ноль

7.3.17.4 Регистр состояния передатчика порта связи (LTSTATx)

Регистры LTSTATx указывают статус передатчика порта связи. Значением сброса регистров LTSTATx является = 0x0000 0002. Подробное описание разрядов регистра приведено ниже (Таблица 61).

Таблица 61 – Регистр LTSTAT

Бит	Имя	Назначение
0	TVACANT	1 – буфер может принять данные 0 – буфер не может принять данные
1	TEMPTY	1 – передатчик пуст 0 – передатчик занят
2	TTER	1 – ошибка передачи time out 0 – нет ошибки
3	TWER	1 – ошибка записи 0 – нет ошибки
31:4	-	Всегда ноль

7.3.18 Группа регистров интерфейса внешней шины

Регистры внешнего порта представлены ниже (Таблица 62).

Таблица 62 – Группа регистров интерфейса шины

Имя	Описание	Адрес	Значение после сброса
SYSCON	Регистр конфигурации внешней шины	0x1F 0080	0x0000 9067
-	-	0x1F 0081– 0x1F 0082	-
nBUSLOCK	Регистр блокировки внешней шины	0x1F 0083	См. Определение
SDRCON	Регистр конфигурации SDRAM	0x1F 0084	См. Определение
-	-	0x1F 0085	-
SYSTAT	Регистр статуса системы; только чтение	0x1F 0086	См. Определение
SYSTATCL	Регистр статуса системы; адрес для сброса ошибок. Только чтение	0x1F 0087	Не определено
-	-	0x1F 0088– 0x1F 008B	-
BMAX	Максимальное число циклов для равнодоступности шины	0x1F 008C	См. Определение
BMAXC	Текущее значение BMAX; только чтение	0x1F 008D	Не определено
-	-	0x1F 008E– 0x1F 009F	-

7.3.18.1 Регистр конфигурации внешней шины (SYSCON)

Регистр SYSCON определяет конфигурацию внешней шины. Для данного регистра имеется специальное управление количеством записей в регистр.

Вывод nBUSLOCK во время сброса определяет количество записей в регистр (одна запись или без ограничений). Если определена одна запись, после ее выполнения другие попытки записи игнорируются. Такие же ограничения касаются регистра SDRCON. Для записи в регистр SYSCON (и регистра SDRCON) более одного раза необходимо удерживать в 1 вывод nBUSLOCK во время сброса. Запись возможна только в режиме супервизора. Подробное описание разрядов регистра приведено ниже (Таблица 63).

Таблица 63 – Регистр SYSCON

Бит	Имя	Назначение
0	BNK0IDLE	Вставка пустого цикла между операциями доступа к банку памяти MS0: 1 – вставить цикл 0 – нет
2:1	BNK0WAIT	Количество внутренних циклов ожидания при обращении к банку MS0: 00 – 0 циклов 01 – 1 цикл 10 – 2 цикла 11 – 3 цикла
4:3	BNK0PIPE	Глубина конвейера банка памяти MS0: 00 – 1 цикл 01 – 2 цикла 10 – 3 цикла 11 – 4 цикла
5	BNK0SLOW	Тип протокола обмена для банка MS0: 0 – синхронный (конвейерный) 1 – медленный (асинхронный)
6	BNK1IDLE	Вставка пустого цикла между операциями доступа к банку памяти MS1: 1 – вставить цикл 0 – нет
8:7	BNK1WAIT	Количество внутренних циклов ожидания при обращении к банку MS1: 00 – 0 циклов 01 – 1 цикл 10 – 2 цикла 11 – 3 цикла
10:9	BNK1PIPE	Глубина конвейера банка памяти MS1: 00 – 1 цикл 01 – 2 цикла 10 – 3 цикла 11 – 4 цикла
11	BNK1SLOW	Тип протокола обмена для банка MS1: 0 – синхронный (конвейерный) 1 – медленный (асинхронный)
12	HOSTIDLE	Вставка пустого цикла между операциями доступа к области памяти хоста: 1 – вставить цикл 0 – нет
14:13	HOSTWAIT	Количество внутренних циклов ожидания при обращении к области хоста: 00 – 0 циклов 01 – 1 цикл 10 – 2 цикла 11 – 3 цикла

16:15	HOSTPIPE	Глубина конвейера для области хоста: 00 – 1 цикл 01 – 2 цикла 10 – 3 цикла 11 – 4 цикла
17	HOSTSLOW	Тип протокола обмена для области хоста: 0 – синхронный (конвейерный) 1 – медленный (асинхронный)
18	-	
19	MEMWIDTH	Ширина шины данных при обращении к внешней памяти 1 – 64 бита 0 – 32 бита
20	MPWIDTH	Ширина шины данных при обращении к мультипроцессорной области памяти 1 – 64 бита 0 – 32 бита
21	HSTWIDTH	Ширина шины данных при обращении к памяти хоста 1 – 64 бита 0 – 32 бита
23:22		Всегда 0
24		Режим работы входного ФИФО внешнего интерфейса 0 – нормальный режим 1 – ускорение передачи данных Бит можно только установить. Сброс только аппаратно.
25		Режим работы выходного буфера внешнего интерфейса 0 – нормальный режим 1 – ускорение передачи данных Бит можно только установить. Сброс только аппаратно.
31:26		Всегда 0

7.3.18.2 Регистр управления блокировкой шины (nBUSLOCK)

Регистр nBUSLOCK определяет состояние запроса блокировки шины. В этом 32-разрядном регистре определен только один нулевой бит. Все другие биты (с 1 по 31) не используются. Запись значения 1 в нулевой бит вызывает запрос на блокировку шины со стороны процессора. Если процессор получит шину в результате арбитража, то он заблокирует её за собой и будет мастером до тех пор, пока бит блокировки не будет сброшен. Факт захвата шины сигнализируется специальным флагом в регистре состояния.

7.3.18.3 Регистр конфигурации SDRAM (SDRCON)

Регистр SDRCON определяет конфигурацию внешней SDRAM. Значение регистра после сброса равно нулю. Подробное описание разрядов регистра приведено ниже (Таблица 64).

Таблица 64 – Регистр SDRCON

Бит	Имя	Назначение
0	SDREN	Включение контроллера SDRAM 1 –включен 0 – выключен
2:1	nCAS	Задержка nCAS 00 -1 цикл 01 – 2 цикла 10 – 3 цикла 11 – резерв
3	PIPE	Дополнительный конвейер: 1 – используется 0 – нет
5:4	PAGE	Размер страницы: 00 – 256 слов 01 – 512 слов 10 – 1024 слов 11 – резерв
6	-	
8:7	REF	Период регенерации памяти (в тактах SCLK) 00 – 1100 циклов 01 – 1850 циклов 10 – 2200 циклов 11 – 3700 циклов
10:9	PRC2RAS	Задержка от подзаряда до строба nRAS (такты SCLK) 00 – 2 01 – 3 10 – 4 11 – 5
13:11	RAS2PRC	Задержка от строба nRAS до команды подзаряда: 000 – 2 110 – 8
14	INIT	Выбор последовательности инициализации: 1 – команда MRS после регенерации памяти 0 – команда MRS перед регенерацией памяти
15	EMREN	Использование дополнительного регистра режима: 1 – разрешено 0 – не используется
31:16	-	Всегда 0

7.3.18.4 Регистр статуса системы (SYSTAT)

Регистр SYSTAT предназначен только для чтения и указывает на состояние некоторых параметров системы. Значением сброса регистра SYSTAT является 0xXX00 SS0S (где XX не определены, SS – вывод). Подробное описание разрядов регистра приведено ниже (Таблица 65).

Таблица 65 – Регистр SYSTAT

Бит	Имя	Назначение
1:0	ID	Идентификатор процессора. Возможные значение от 0 до 3
2	-	Всегда 0
3	-	Всегда 0
5:4	BUSMSTR	ID процессора который в данный момент является мастером на шине
6	-	Всегда 0

Бит	Имя	Назначение
7	HSTMSTR	1 – мастером на шине является хост-процессор 0 – мастер на шине процессор BUSMSTR
10:8		Значение контактных шариков RANGEFIL
11	-	Всегда 0
12	BOOTMDE	Режим загрузки: 0 – из внешнего EPROM 1 – из другого источника
13	MRSCOMP	Признак окончания процедуры инициализации динамической памяти: 1 – завершена. Память готова к использованию. 0 – нет
14	BUSLKACT	Признак захвата шины: 1 – текущий процессор стал мастером на шине и удерживает её. 0 – нет захвата шины.
15	-	Всегда 0
16	BROADREADERR	Ошибка доступа к широковещательному адресному пространству: 1 – была попытка чтения из указанного пространства 0 – нет ошибки
17	AUTODMAERR	Признак ошибки в работе каналов autoDMA: 1 – была попытка записи в выключенный канал 0 – нет ошибки
18	SDRAMERR	Признак ошибки доступа к SDRAM. 1 – была попытка доступа неинициализированной внешней динамической памяти. 0 – нет ошибки
19	MPREADERR	Признак ошибки чтения процессором собственного мультипроцессорного адресного пространства: 1 – ошибка. Было чтение 0 – нет ошибки
31:20	-	Всегда 0

Программы могут считывать регистр SYSTAT, используя имена SYSTAT или SYSTATCL (сброс SYSTAT):

- SYSTAT – чтение без изменений в содержимом регистра;
- SYSTATCL – разряды 19–16 будут сброшены после чтения.

7.3.18.5 Регистр ограничения времени управления шиной (BMAX)

В регистр BMAX загружается максимальное число циклов (SOCCLK) для которых процессор может сохранять владение внешней шиной. Используются только 16 младших бит регистра. Старшие биты всегда равны нулю. После сброса значение регистра равно 0xFFFF. Это значение соответствует выключению счетчика.

7.3.18.6 Регистр текущего ограничения времени управления шиной (BMAXC)

Данный регистр является счетчиком и может каждый такт SOCCLK изменять свое значение (уменьшается на 1). Адрес BMAXC доступен только по чтению. Определены только 16 младших разрядов счетчика. Старшие всегда равны нулю. После сброса BMAXC равен 0xFFFF. Это соответствует выключенному состоянию. Счетчик загружает BMAXC загружает новое значение при записи по адресу BMAX. Если записанное значение не равно 0xFFFF, то это означает включение счетчика в

работу. Если текущий процессор захватил шину и использует ее, то каждый такт SOCCLK счетчик будет уменьшать свое значение на 1. При достижении значения 0 счетчик проверяет наличие запросов к шине со стороны других процессоров. Если их нет – счетчик перезагружает значение из BMAX и продолжает счет. Если запросы есть – счетчик останавливается и требует от процессора освободить шину.

7.3.19 Группа регистров эмулятора JTAG

Таблица 66 – Группа регистров JTAG

Имя	Описание	Адрес	Значение по умолчанию
EMUCTL	Управление	0x1F 03A0	0x0000 0000
EMUSTAT	Состояние	0x1F 03A1	0x0000 0002
EMUDAT	Данные	0x1F 03A2	Не определено
EMUIR:0	Команда 0	0x1F 03A4	0x33C0 0000
EMUIR:1	Команда 1	0x1F 03A5	0x33C0 0000
EMUIR:2	Команда 2	0x1F 03A6	0x33C0 0000
EMUIR:3	Команда 3	0x1F 03A7	0xB3C0 0000
-	-	0x1F 03A8	-
OSPID	OSPID	0x1F 03A9	Не определено
-	-	0x1F 03AA–0x1F 03BF	-

7.3.19.1 Регистр управления (EMUCTL)

Регистр EMUCTL устанавливает параметры эмуляции. Значением сброса регистра EMUCTL является 0. Большинство разрядов регистра EMUCTL используются эмулятором и не должны модифицироваться программой. Единственными разрядами, предназначенными для использования программой, являются SWRST (программный сброс) и BOOTDIS (запрет загрузки). Подробное описание разрядов регистра приведено ниже (Таблица 67).

Таблица 67 – Регистр EMUCTL

Бит	Имя	Назначение
0	EMEN	Включение эмулятора 1 – включен 0 – выключен
1	TEME	Разрешение генерации запроса к процессору по входу TMS: 1 – разрешено 0 – запрещено
2	EMUOE	Разрешение активной выдачи информации на выход TDO 1 – разрешено 0 – запрещено
3	-	Бит общего назначения
4	SWRST	Запрос программного сброса: 1 – сброс 0 – рабочий режим
5	BOOTDSBL	Запрещение загрузки 1 – запрещено 0 – разрешено Установка данного бита запрещает загрузку из внешнего EPROM.
31:6	-	Всегда ноль

7.3.20 Регистр состояния эмуляции (EMUSTAT)

Регистр EMUSTAT отражает состояние эмуляции. Значением сброса регистра EMUCTL является 0x0000 0002. Подробное описание разрядов регистра приведено ниже (Таблица 68).

Таблица 68 – Регистр EMUSTAT

Бит	Имя	Назначение
0	EMUMOD	1 – процессор в состоянии эмуляции 0 – нормальный режим работы
1	IRFREE	Флаг готовности команды в регистре EMUIR 1 – команда готова 0 – нет
2	INRESET	Флаг сброса эмулятора
31:3	-	Всегда ноль

7.3.21 Группы регистров отладки

Группы отладки описываются ниже (Таблица 69). Доступ к регистрам отладки возможет только как к отдельным словам. К регистрам отладки может быть применена только команда пересылки «регистр-регистр» или командам немедленной загрузки данных. Эти регистры не могут быть загружены из памяти или сохранены напрямую в память. После каждой записи в регистры группы отладки происходит перезапуск конвейера процессора.

Таблица 69 – Группа регистров отладки

Имя	Описание	Адрес	Значение по умолчанию
WP0CTL	Управление точкой наблюдения 0	0x1E 0360	0x0000 0000
WP1CTL	Управление точкой наблюдения 1	0x1E 0361	0x0000 0000
WP2CTL	Управление точкой наблюдения 2	0x1E 0362	0x0000 0000
-	-	0x1E 0363	-
WP0STAT	Состояние точки наблюдения 0; (RO)	0x1E 0364	0x0000 0000
WP1STAT	Состояние точки наблюдения 1; (RO)	0x1E 0365	0x0000 0000
WP2STAT	Состояние точки наблюдения 2; (RO)	0x1E 0366	0x0000 0000
-	-	0x1E 0367	-
WP0L	Нижний адрес точки наблюдения 0	0x1E 0368	Не определено
WP0H	Верхний адрес точки наблюдения 0	0x1E 0369	Не определено
WP1L	Нижний адрес точки наблюдения 1	0x1E 036A	Не определено
WP1H	Верхний адрес точки наблюдения 1	0x1E 036B	Не определено
WP2L	Нижний адрес точки наблюдения 2	0x1E 036C	Не определено
WP2H	Верхний адрес точки наблюдения 2	0x1E 036D	Не определено

WPDR	Регистр данных точки наблюдения 1	0x1E 036E	0
WPMR	Регистр маски точки наблюдения 1	0x1E 036F	0
CCNT0	Нижняя часть счетчика циклов	0x1E 0370	0
CCNT1	Верхняя часть счетчика циклов	0x1E 0371	0
PRFM	Маска монитора производительности	0x1E 0372	0x0000 0000
PRFCNT	Счетчик монитора производительности	0x1E 0373	0x0000 0000
TRCBMASK	Маска буфера трассировки (RO)	0x1E 0374	0x0000 0000
TRCBPTR	Указатель буфер трассировки (RO)	0x1E 0375	0x0000 0000
-	-	0x1E 0376–0x1E 037B	-
TRCBVAL	Допустимый буфер трассировки (RO)	0x1E 037C	0x0000 0000
-	-	0x1E 037D–0x1E 037F	-
TRCB31–0	Буфер трассировки (RO)	0x1E 0140–0x1E015F	0x0000 0000

Точка наблюдения представляет собой механизм описания условия, при выполнении которого модуль отладки формирует запрос к процессору на обработку исключительной ситуации либо на переход в режим эмуляции.

7.3.21.1 Регистр управления точкой наблюдения (WPxCTL)

У каждой из трех точек наблюдения есть регистр управления (WP0CTL, WP1CTL и WP2CTL), который используется для определения действий точки наблюдения. Значением сброса регистров WPxCTL является 0x0000 0000. Каждая из точек наблюдения осуществляет мониторинг определенных шин процессорного ядра. Точка 0 – шину I, точка 1 – шины J и K, точка 2 – шину S.

Подробное описание разрядов регистра WP0CTL приведено ниже (Таблица 70).

Таблица 70 – Регистр WP0CTL

Бит	Имя	Назначение
1:0	OPMODE	Режим работы; 00 – выключена 01 – включена. Анализ совпадения адреса. 10 – включена. Анализ попадания в интервал. 11 – включена. Анализ выхода вне интервала.
3:2	EXTYPE	Действия при выполнении условия: 00 – нет действий 01 – программное исключение 10 – переход в режим эмулятора 11 – резерв
4	SSTP	Режим пошагового выполнения команд: 1 – включен 0 – выключен
15:5	-	Всегда ноль
31:16	WP0CNT	Счетчик количества срабатываний точки наблюдения до генерации исключительной ситуации.

Подробное описание разрядов регистра WP1CTL приведено ниже (Таблица 71).

Таблица 71 – Регистр WP1CTL

Бит	Имя	Назначение
1:0	OPMODE	Режим работы; 00 – выключена 01 – включена. Анализ совпадения адреса. 10 – включена. Анализ попадания в интервал. 11 – включена. Анализ выхода вне интервала.
3:2	EXTYPE	Действия при выполнении условия: 00 – нет действий 01 – программное исключение 10 – переход в режим эмулятора 11 – резерв
4	READ	Мониторинг циклов чтения: 1 – включен 0 – выключен
5	WRITE	Мониторинг циклов записи: 1 – включен 0 – выключен
6	Jbus	Мониторинг шины J: 1 – включен 0 – выключен
7	Kbus	Мониторинг шины K: 1 – включен 0 – выключен
15:8	-	Всегда ноль
31:16	WP1CNT	Счетчик количества срабатываний точки наблюдения до генерации исключительной ситуации.

Подробное описание разрядов регистра WP2CTL приведено ниже (Таблица 72).

Таблица 72 – Регистр WP2CTL

Бит	Имя	Назначение
1:0	OPMODE	Режим работы; 00 – выключена 01 – включена. Анализ совпадения адреса. 10 – включена. Анализ попадания в интервал. 11 – включена. Анализ выхода вне интервала.
3:2	EXTYPE	Действия при выполнении условия: 00 – нет действий 01 – программное исключение 10 – переход в режим эмулятора 11 – резерв
4	READ	Мониторинг циклов чтения: 1 – включен 0 – выключен
5	WRITE	Мониторинг циклов записи: 1 – включен 0 – выключен
15:6	-	Всегда ноль
31:16	WP2CNT	Счетчик количества срабатываний точки наблюдения до генерации исключительной ситуации.

7.3.21.2 Регистры состояния точки наблюдения (WPxSTAT)

У каждой из трех точек наблюдения есть регистр состояния (WP0STAT, WP1STAT и WP2STAT). Значениями сброса регистров WPxSTAT являются 0x0000 0000. Подробное описание разрядов регистра приведено ниже (Таблица 73).

Таблица 73 – Регистр WPxSTAT

Бит	Имя	Назначение
15:0	WPxCNT	Текущее значение счетчика точки наблюдения. Счетчик вычитает 1 каждый раз при срабатывании точки.
17:16	EX	Состояние точки наблюдения 00 –выключена 01 – активна 10 – резерв 11 – переполнился счетчик
31:18	-	Всегда 0

7.3.21.3 Регистры указателя адреса точки наблюдения (WPxL/WPxH)

Указатели адреса точки наблюдения являются 32-битными указателями, определяющими адрес или диапазон адресов точки наблюдения. После сброса их значение не определено.

7.3.21.4 Регистр маски монитора производительности (PRFM)

Регистр маски монитора производительности (PRFM) определяет, какие события в процессоре отслеживаются и подсчитываются счетчиком монитора производительности (PRFCNT). Значением сброса регистра PRFM является 0x00000000. Подробное описание разрядов регистра приведено ниже (Таблица 74).

Таблица 74 – Регистр PRFM

Бит	Имя	Назначение
7:0	-	резерв
8	Jexe	Подсчет команд, выполненных на линии конвейера J 1 – включен 0 – выключен
9	Kexe	Подсчет команд, выполненных на линии конвейера К 1 – включен 0 – выключен
10	Xexe	Подсчет команд, выполненных на линиях конвейера X1 и X2 1 – включен 0 – выключен
11	Yexe	Подсчет команд, выполненных на линиях конвейера У1 и У2 1 – включен 0 – выключен
12	Sexe	Подсчет команд, выполненных на линии конвейера S 1 – включен 0 – выключен
15:13	-	резерв
16	STALL	Подсчет команд остановов конвейера вызванных ожиданием данных или другими блокирующими конвейер ситуациями 1 – включен 0 – выключен
17	BTB_true	Подсчет количества правильных предсказаний буфера переходов 1 – включен 0 – выключен

18	ABORT	Подсчет количества программных исключительных ситуаций 1 – включен 0 – выключен
19	Uexe	Подсчет количества линий команд, выполненных в режиме пользователя 1 – включен 0 – выключен
20	BTB_false	Подсчет количества ошибочных предсказаний буфера переходов 1 – включен 0 – выключен
21	BTB_load	Подсчет количества загрузок в буфер переходов 1 – включен 0 – выключен
31:22	-	резерв

Все события, которые отслеживаются монитором производительности суммируются по «ИЛИ» и при их наступлении происходит увеличение счетчика PRFM на 1. Логичным является разрешение подсчета одного условия из многих.

7.3.21.5 Регистр счетчика монитора производительности (PRFCNT)

Назначение счетчика производительности – увеличивать свое значение на 1 каждый раз, когда происходит отслеживаемое событие. Отслеживаемое событие определяется регистром маски монитора производительности (PRFM). Регистр очищается после сброса.

7.3.21.6 Регистры счетчика циклов (CCNTx)

Длина счетчика циклов равна 64 разряда, однако доступен он как два универсальных 32-разрядных регистра – CCNT0 и CCNT1. После сброса значение счетчика равно нулю.

В связи с тем, что счетчик 64 бита, а доступ возможен только к частям регистра, необходимо соблюдение специальной процедуры чтения и записи регистра. При чтении сначала считывается нижняя часть (CCNT0), а затем верхняя часть (CCNT1). При чтении нижней части верхняя копируется в буфер и это гарантирует её корректное значение. При записи сначала пишется нижняя часть (она попадает в буфер), а затем верхняя. При записи верхней полное 64-битное значение попадает в счетчик.

7.3.21.7 Регистры указателя и буфера трассировки (TRCBx/TRCBPTR)

Каждый раз, когда процессор выполняет переход на непоследовательно исполняемую команду, адрес непоследовательно выбранной команды (адрес перехода) записывается в один из регистров буфера трассировки. Первая запись осуществляется в буфер трасировки 0, вторая – в 1, и далее циклично. Указатель буфера трассировки определяет последний записанный буфер трассировки.

Таблица 75 – Группа регистров буфера трассировки

Имя	Описание	Адрес	Значение по умолчанию
TRCB0	Буфер трассировки 0; только чтение	0x1E 0140	0x0000 0000
TRCB1	Буфер трассировки 1; только чтение	0x1E 0141	0x0000 0000
TRCB2	Буфер трассировки 2; только чтение	0x1E 0142	0x0000 0000

TRCB3	Буфер трассировки 3; только чтение	0x1E 0143	0x0000 0000
TRCB4	Буфер трассировки 4; только чтение	0x1E 0144	0x0000 0000
TRCB5	Буфер трассировки 5; только чтение	0x1E 0145	0x0000 0000
TRCB6	Буфер трассировки 6; только чтение	0x1E 0146	0x0000 0000
TRCB7	Буфер трассировки 7; только чтение	0x1E 0147	0x0000 0000
TRCB8	Буфер трассировки 8; только чтение	0x1E 0148	0x0000 0000
TRCB9	Буфер трассировки 9; только чтение	0x1E 0149	0x0000 0000
TRCB10	Буфер трассировки 10; только чтение	0x1E 014A	0x0000 0000
TRCB11	Буфер трассировки 11; только чтение	0x1E 014B	0x0000 0000
TRCB12	Буфер трассировки 12; только чтение	0x1E 014C	0x0000 0000
TRCB13	Буфер трассировки 13; только чтение	0x1E 014D	0x0000 0000
TRCB14	Буфер трассировки 14; только чтение	0x1E 014E	0x0000 0000
TRCB15	Буфер трассировки 15; только чтение	0x1E 014F	0x0000 0000
TRCB16	Буфер трассировки 16; только чтение	0x1E 0150	0x0000 0000
TRCB17	Буфер трассировки 17; только чтение	0x1E 0151	0x0000 0000
TRCB18	Буфер трассировки 18; только чтение	0x1E 0152	0x0000 0000
TRCB19	Буфер трассировки 19; только чтение	0x1E 0153	0x0000 0000
TRCB20	Буфер трассировки 20; только чтение	0x1E 0154	0x0000 0000
TRCB21	Буфер трассировки 21; только чтение	0x1E 0155	0x0000 0000
TRCB22	Буфер трассировки 22; только чтение	0x1E 0156	0x0000 0000
TRCB23	Буфер трассировки 23; только чтение	0x1E 0157	0x0000 0000
TRCB24	Буфер трассировки 24; только чтение	0x1E 0158	0x0000 0000
TRCB25	Буфер трассировки 25; только чтение	0x1E 0159	0x0000 0000
TRCB26	Буфер трассировки 26; только чтение	0x1E 015A	0x0000 0000
TRCB27	Буфер трассировки 27; только чтение	0x1E 015B	0x0000 0000
TRCB28	Буфер трассировки 28; только чтение	0x1E 015C	0x0000 0000
TRCB29	Буфер трассировки 29; только чтение	0x1E 015D	0x0000 0000
TRCB30	Буфер трассировки 30; только чтение	0x1E 015E	0x0000 0000
TRCB31	Буфер трассировки 31; только чтение	0x1E 015F	0x0000 0000

7.3.21.8 Регистр данных точки наблюдения 1 (WPDR)

Точка наблюдения 1 имеет дополнительные возможности по мониторингу обмена данными ядра процессора с памятью. Выше описывался механизм срабатывания исключительной ситуации в случае совпадения адресов точки наблюдения с адресами шин J и K. Однако имеется возможность дополнить сравнение адресов еще и сравнением данных. При чтении имеется возможность контролировать какие данные считывает процессор. Если дополнительно к совпадению адресов происходит совпадение и прочитанных данных – точка срабатывает. Также можно контролировать и процесс записи данных. Необходимые для контроля данные записываются в регистр WPDR. Необходимо отметить, что шина чтения или записи имеет разрядность 128 бит. Разрядность регистра данных только 32 бита. Поэтому сравнение выполняется только для 32-разрядного слова шины данных определяемого младшими битами шины адреса независимо от того выполняется чтение или запись 64-х или 128-разрядного слова.

7.3.21.9 Регистр маски точки наблюдения 1 (WPMR)

При использовании шины данных в описании точки наблюдения 1, не все биты шины данных могут понадобиться при анализе читаемых или записываемых данных. Регистр маски позволяет установить какие биты регистра данных WPDR должны участвовать в сравнении. Если бит установлен в 1 – соответствующий ему бит данных сравнивается с битом шины данных. После сброса регистр маски равен нулю и данные не участвуют в работе точки наблюдения.

7.4 Внешний порт и интерфейс SDRAM

Процессор может быть использован как в качестве самостоятельного процессора, так и в качестве элемента мультипроцессорной системы. Он может функционировать как отдельный процессор или система, а может управляться компьютером (например, система процессора может быть установлена на плате ПК).

В данном разделе приведено описание интерфейса внешней шины процессора, который включает:

- внутреннюю логику интерфейса;
- арбитр шины;
- шину адреса;
- шину данных;
- шину управления.

Наиболее быстрым протоколом обмена является конвейерный протокол. Процессор использует этот протокол для соединения с другими процессорами в кластере. Также процессор может использовать этот протокол для соединения через интерфейс с хостом и системами памяти. Протокол имеет максимальную производительность при одной передаче данных во время каждого тактового цикла шины и может поддерживать уровень производительности близко к максимальному значению.

Другим быстрым протоколом является протокол SDRAM. Данный протокол определяется производственными стандартами микросхем SDRAM. Контроллер SDRAM расположен на кристалле процессора. Он управляет всеми контрольными сигналами SDRAM (nRAS, nCAS, nSDWE, SDCKE, LDQM и HDQM) и поддерживает инициализацию и регенерацию микросхем SDRAM. Максимальная производительность SDRAM при передаче данных равна одной передаче за один тактовый цикл. Производительность может удерживаться близко к максимальной, если последовательные доступы выполняются к той же странице

Процессор поддерживает также протокол медленного устройства. Этот протокол следует использовать для устройств не критичных в плане производительности. В большинстве систем такие устройства следует подключать на вспомогательную шину, поскольку они могут увеличить нагрузку на шину и замедлить большое число критических доступов.

Внешняя шина включает в себя шину данных, с пропускной способностью 32 или 64 разрядов, 32-разрядную шину адреса и контрольные сигналы. Большинство сигналов являются двунаправленными, поскольку процессор может быть как ведущим, так и ведомым на внешней шине.

Блок интерфейса внешней шины может быть сконфигурирован под различные настройки путем записи в конфигурационные регистры SYSCON и SDRCON внешнего порта. Установочные параметры этих двух регистров должны быть одинаковыми для всех процессоров в системе. Широковещательная запись рекомендована при инициализации регистров SYSCON и SDRCON.

При описании работы внешнего интерфейса в мультипроцессорной системе использованы следующие термины:

- Мастер (ведущий) шины. Устройство на внешней шине, которое в настоящий момент управляет всеми линиями интерфейса.
- Подчиненный (ведомый). Процессор, память, устройство ввода-вывода к которому может выполнять обращение мастер шины.
- Хост-процессор. Внешнее по отношению к кластеру устройство, которое имеет доступ ко всем ресурсам кластера и управляет специальными линиями внешнего интерфейса.

7.4.1 Особенности внешней шины

- Ширина шины данных: 64 или 32 разряда, программируется отдельно для памяти, мультипроцессорного обмена и интерфейса хоста.
- Конвейерные передачи с программируемым числом этапов конвейера.
- Программируемые состояния простоя (IDLE).
- Протокол поддерживает циклы ожидания, добавляемые ведомым устройством с использованием вывода ACK.
- Интерфейс EPROM и FLASH: 8-разрядная шина данных с фиксированным числом циклов ожидания, поддержкой чтения и записи.
- Хост-интерфейс.
- SDRAM-интерфейс.
- Поддержка медленных устройств ввода/вывода.
- Мультипроцессорная система, основанная на распределенном арбитраже шины без дополнительной логики.
- Поддержка передач через DMA для внешних устройств ввода/вывода в режиме квитирования.
- Поддержка режима flyby для передач между памятью SDRAM и устройством ввода/вывода в DMA.
- Поддержка арбитража кластерной шины пока процессор находится в состоянии программного сброса.

7.4.2 Внешние вводы/выводы интерфейса шины

Ниже приведено описание основных выводов внешнего интерфейса. Буква «n» в начале названия вывода указывает на низкий активный уровень сигнала.

- **ADDR31-0**
  Шина адреса с тремя состояниями. В режиме мастера процессор выдает на эти линии адрес устройства, к которому он выполняет доступ. В режиме подчиненного процессор анализирует адрес для отслеживания доступа к его внутренним ресурсам.

- **DATA63–0**
  Шина данных с тремя состояниями. Процессор выдает данные на эти выводы или принимает данные или команды с помощью этих выводов. Имеется возможность задания размера шины (32 или 64 бита) для работы с различными устройствами системы.

- **nRD**
  Чтение памяти. В режиме мастера nRD активируется всякий раз, когда процессор осуществляет чтение из подчиненного устройства системы. Когда процессор подчиненный, nRD является входом и указывает на транзакции чтения, имеющие доступ к внутренней памяти или универсальным регистрам. nRD изменяется одновременно с выводами ADDR. Активный уровень низкий. Не используется при доступе к SDRAM.

- **nWRL**
  Запись младшего слова. В режиме мастера nWRL активируется в двух случаях. Первый, когда шина данных имеет ширину 64 бита и происходит

запись в слово четного адреса внешнего устройства. Второй, когда шина имеет ширину 32 бита и происходит запись по любому адресу. Когда процессор подчиненный, nWRL является входом и указывает на транзакции записи, имеющие доступ к внутренней памяти или универсальным регистрам. nWRL изменяется одновременно с выводами ADDR. Активный уровень низкий. Не используется при доступе к SDRAM.

– **nWRH**
Запись старшего слова. В режиме мастера nWRH активируется, когда размер шины данных равен 64 бита и выполняется запись двойного слова (64 разряда), либо выполняется запись слова по нечетному адресу внешней памяти при 64-разрядной шине данных. Если размер шины данных 32 бита, nWRH не используется. Когда процессор подчиненный, nWRH является входом и указывает на транзакции записи, имеющие доступ к внутренней памяти или универсальным регистрам. nWRH изменяется одновременно с выводами ADDR. Активный уровень низкий. Не используется при доступе к SDRAM.

– **ACK**
Подтверждение готовности. В режиме мастера процессор анализирует вход ACK во время доступа к устройствам. Внешние подчиненные устройства могут деактивировать (установить в низкий уровень) ACK чтобы добавить состояние ожидания во время доступа к ним. ACK используется подчиненными устройствами на фазе данных. Когда процессор подчиненный и к нему выполняется доступ, ACK является выходом и может использоваться процессором для приостановки обмена в случае неготовности процессора. Высокий уровень на линии ACK означает готовность устройства, низкий – требование подождать. Устройства могут выставлять только активный низкий уровень. Высокий уровень на линии ACK обеспечивается резистором. ACK не используется при доступе к SDRAM.

– **nBMS**
Выбор загрузочной памяти. Вывод является многофункциональным. Во время сброса значение на линии nBMS указывает на выбор варианта начального старта процессора. В режиме мастера nBMS активизируется, когда процессор выполняет доступ к загрузочной памяти (EPROM или флэш). В режиме подчиненного процессор анализирует вход nBMS для обнаружения доступа к внутренним ресурсам. nBMS изменяется одновременно с выводами ADDR. Активный уровень низкий.

– **nMS1-0**
Выбор памяти. В режиме мастера nMS0 или nMS1 активируются в то время, когда процессор желает получить доступ к банкам памяти 0 или 1 соответственно. Выходы nMS1–0 являются декодированными сигналами адреса памяти и изменяются параллельно с выводами ADDR. Когда линии

ADDR31–27 равны 0b00110, nMS0 активирован. Когда линии ADDR31–27 равны 0b00111, nMS1 активирован. В режиме подчиненного процессор анализирует входы nMS1–0 для обнаружения доступа к внутренним ресурсам. Активный уровень низкий.

- **nMSH**
  Выбор хост-памяти. В режиме мастера nMSH активируется, когда процессор выполняет доступ к адресному пространству хост-процессора (ADDR31 равен 1). nMSH является декодированным сигналом адреса памяти, который изменяется параллельно с выводами ADDR. В режиме подчиненного процессор анализирует вход nMSH для обнаружения доступа к внутренним ресурсам. Активный уровень низкий.

- **nBRST**
  Пакетный режим обмена. В режиме мастера процессор активизирует эту линию, когда он выполняет несколько обменов по последовательным адресам. Это позволяет подчиненному устройству не анализировать последующие адреса и заранее подготавливать необходимые данные. В режиме подчиненного процессор анализирует вход nBRST для обнаружения последовательного доступа к его внутренним ресурсам. Активный уровень низкий.

7.4.3 Микроархитектура процессора

Функции процессора могут изменяться в зависимости от системы, в которой они используются. Он может быть представлен и как отдельный процессор, и в качестве элемента мультипроцессорной системы на общей внешней шине (Рисунок 14).

В мультипроцессорной системе каждый процессор должен определяться с помощью своего ID (устанавливается выводами ID[2:0). В системе должен быть процессор с ID равным нулю, поскольку процессор ID0 используется для инициализации внешней памяти SDRAM. Также процессор ID0 имеет внутренние резисторы, повышающие или понижающие логический уровень на некоторых линиях, которые активны во время или после сброса.

Другими устройствами, определяемыми на внешней шине, могут быть: модули памяти, устройства ввода/вывода, мосты на другие шины. К каждому процессору и хост-процессору в системе может быть организован доступ как к ведомому устройству.

Передаваемые на внешней шине данные могут быть стандартными словами (32 разряда), двойными словами (64 разряда) или счетверенными словами (128 разрядов). Адрес этих слов должен быть выровнен по их размеру. Поскольку ширина шины (32 или 64 разряда) может быть меньше размера передачи, то передача данных распределяется по нескольким циклам внешней шины.

Внешний порт (EBIU) может быть мастером или подчиненным на внешней шине, а также мастером и подчиненным на внутренних шинах процессора. Ниже (Рисунок 15) показана структура внутренних шин, соединяющих внешний порт с внутренними модулями процессора. Три главных элемента передачи данных в структуре внешнего порта это:

- выходное FIFO (OFIFO) передачи запросов на внешнюю шину;
- входное FIFO (IFIFO) входных запросов с внешней шины или возвращаемых данных с внешней шины;

- выходной буфер (OBUF) для возвращения данных из внутренней памяти или регистров в ответ на внешний запрос чтения.

Буфер IFIFO используется в следующих случаях:

- Передача запросов от внешних ведущих устройств к внутренним ресурсам системы для выполнения операций чтения или записи. Внешний интерфейс в данной ситуации работает как подчиненный и отслеживает все доступы к внутренним ресурсам его процессора. Далее запросы отправляются на SOC-шину или в ядро процессора (в IFIFO SOC-интерфейса).
- Передача данных во внутренний приемник в ответ на чтение ядром процессора или контроллером DMA устройств на внешней шине. В этой ситуации внешний интерфейс работает как мастер на внешней шине и возвращает прочитанные данные. Далее данные отправляются на SOC-шину (контроллер DMA или порты связи) или в ядро процессора (в IFIFO SOC-интерфейса).

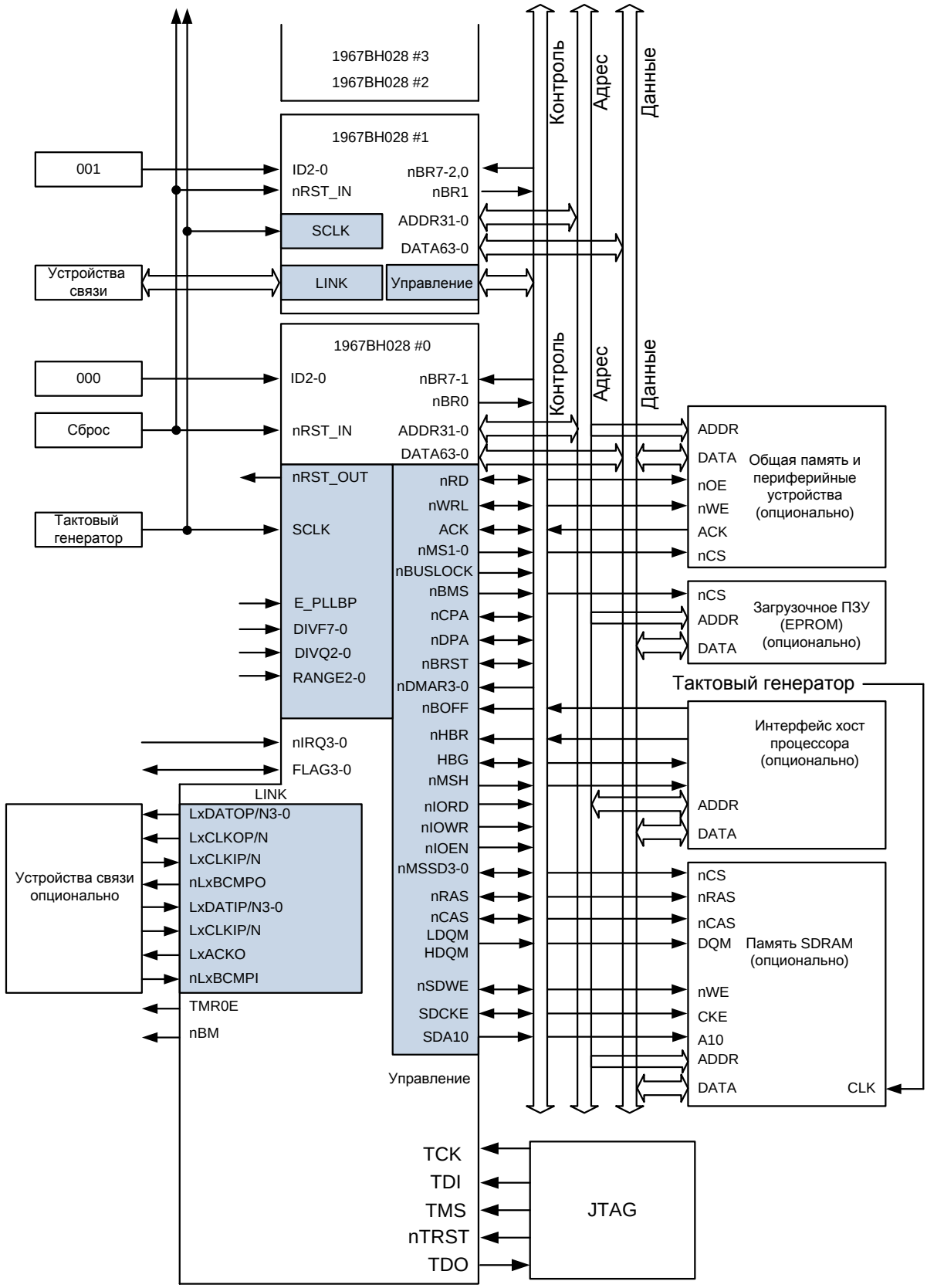


Рисунок 14 – Типичная конфигурация мультипроцессорного кластера

Выходной буфер OFIFO внешнего порта используется для всех транзакций, направленных во внешнее адресное пространство. Это могут быть запросы процессорного ядра или контроллера DMA.

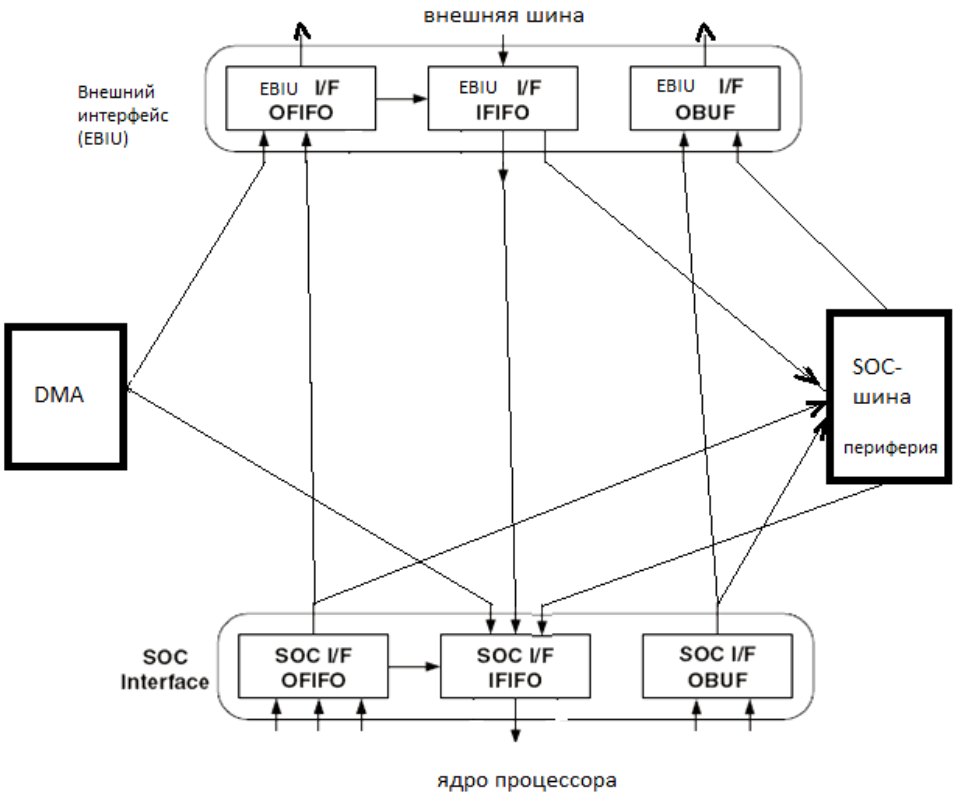


Рисунок 15 – Структура соединений модулей SOC-шины

На рисунке ниже подробно представлены потоки данных, которые имеют место при различных вариантах передач (Рисунок 16). Возможны операции чтения и операции записи. Операции, выполняемые ядром, контроллером DMA или внешним мастером.

Если операция является операцией записи, то она завершается:

- для ядра процессора в момент записи данных в SOC OFIFO;
- для контроллера DMA в момент записи в EBIU OFIFO;
- для внешнего мастера в момент записи в EBIU IFIFO.

Если выполняется операция чтения, то данные проходят следующие этапы:

- При чтении ядром процессора, запрос помещается в SOC OFIFO, далее он перемещается в EBIU OFIFO, где читается и анализируется внешним интерфейсом. Выполняется цикл чтения внешнего устройства и прочитанные данные размещает в EBIU IFIFO, откуда они пересылаются в SOC IFIFO. SOC интерфейс принятые данные отправляет в регистр назначения.
- При чтении контроллером DMA, запрос помещается в EBIU OFIFO, где затем читается и анализируется внешним интерфейсом. Выполняется цикл чтения внешнего устройства и прочитанные данные размещает в EBIU IFIFO, откуда они пересылаются в буфер контроллера либо буфера передатчиков портов связи
- При чтении внешним мастером внутренней памяти процессора запрос помещается в EBIU IFIFO, далее он перемещается в SOC IFIFO, где читается и анализируется SOC-интерфейсом. Выполняется цикл чтения внутренней памяти или внутреннего регистра ядра и прочитанные данные размещает в SOC OBUF, откуда они пересылаются в EBIU OBUF. Внешний интерфейс принятые данные выставляет на внешнюю шину данных для того чтобы мастер шины мог их принять.

- При чтении внешним мастером регистра внутренней периферии процессора на SOC-шине, запрос помещается в EBIU IFIFO, далее он посылается на SOC-шину. Выполняется цикл чтения регистра устройства и прочитанные данные пересылаются в EBIU OBUF. Внешний интерфейс принятые данные выставляет на внешнюю шину данных для того чтобы мастер шины мог их принять.

Передачи данных выполняются в соответствии с картой памяти и ID процессора. При доступе к ресурсам со стороны многих устройств используется приоритет доступа. Из рисунка выше (Рисунок 15), что EBIU IFIFO участвует в доступе к SOC-шине и к SOC-IFIFO. При доступе к SOC-шине приоритет следующий:

- SOC OBUF (наивысший);
- EBIU IFIFO;
- процессорное ядро (низший).

При доступе к SOC IFIFO приоритет следующий:

- чтение ядром регистра SOC шины;
- EBIU IFIFO;
- контроллер DMA.

Приоритет устройств фиксированный и не зависит от приоритета транзакции DMA или приоритета транзакции внешнего мастера.

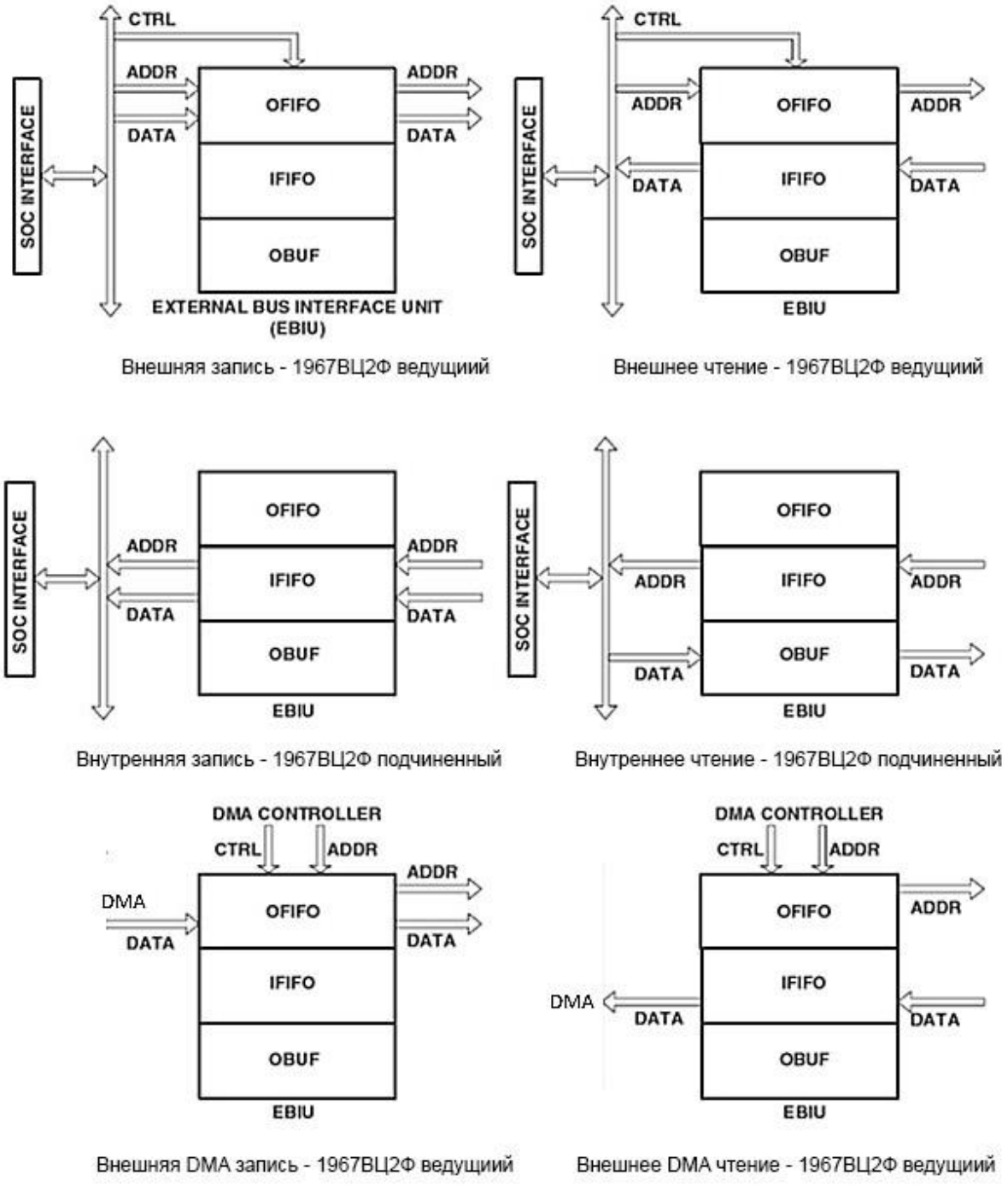


Рисунок 16 – Передача данных через интерфейс внешней шины (EBIU)

7.4.4 Программирование регистра SYSCON

Регистр SYSCON является регистром конфигурации системы и должен быть запрограммирован до того, как начнется передача данных на шине (если значения по умолчанию должны изменяться). В режиме пользователя доступ к SYSCON невозможен. В режиме супервизора количество записей в регистр зависит от вывода nBUSLOCK (фиксируется во время сброса). Этот вывод позволяет выбрать между режимами однократной записи и многократной записи. Описание разрядов регистра приведено в таблице ниже (Таблица 76). Рассмотрим подробно назначение различных разрядов регистра.

Таблица 76 – Регистр SYSCON

Бит	Имя	Назначение
0	BNK0IDLE	Вставка пустого цикла между операциями доступа к банку памяти MS0: 1 – вставить цикл 0 – нет
2:1	BNK0WAIT	Количество внутренних циклов ожидания при обращении к банку MS0: 00 – 0 циклов 01 – 1 цикл 10 – 2 цикла 11 – 3 цикла
4:3	BNK0PIPE	Глубина конвейера банка памяти MS0: 00 – 1 цикл 01 – 2 цикла 10 – 3 цикла 11 – 4 цикла
5	BNK0SLOW	Тип протокола обмена для банка MS0: 0 – синхронный (конвейерный) 1 – медленный (асинхронный)
6	BNK1IDLE	Вставка пустого цикла между операциями доступа к банку памяти MS1: 1 – вставить цикл 0 – нет
8:7	BNK1WAIT	Количество внутренних циклов ожидания при обращении к банку MS1: 00 – 0 циклов 01 – 1 цикл 10 – 2 цикла 11 – 3 цикла
10:9	BNK1PIPE	Глубина конвейера банка памяти MS1: 00 – 1 цикл 01 – 2 цикла 10 – 3 цикла 11 – 4 цикла
11	BNK1SLOW	Тип протокола обмена для банка MS1: 0 – синхронный (конвейерный) 1 – медленный (асинхронный)
12	HOSTIDLE	Вставка пустого цикла между операциями доступа к области памяти хоста: 1 – вставить цикл 0 – нет
14:13	HOSTWAIT	Количество внутренних циклов ожидания при обращении к области хоста: 00 – 0 циклов 01 – 1 цикл 10 – 2 цикла 11 – 3 цикла
16:15	HOSTPIPE	Глубина конвейера для области хоста: 00 – 1 цикл 01 – 2 цикла 10 – 3 цикла 11 – 4 цикла
17	HOSTSLOW	Тип протокола обмена для области хоста: 0 – синхронный (конвейерный) 1 – медленный (асинхронный)

18	-	
19	MEMWIDTH	Ширина шины данных при обращении к внешней памяти 1 – 64 бита 0 – 32 бита
20	MPWIDTH	Ширина шины данных при обращении к мультипроцессорной области памяти 1 – 64 бита 0 – 32 бита
21	HSTWIDTH	Ширина шины данных при обращении к памяти хоста 1 – 64 бита 0 – 32 бита
23:22		Всегда 0
24		Режим работы входного ФИФО внешнего интерфейса 0 – нормальный режим 1 – ускорение передачи данных Бит можно только установить. Сброс только аппаратно.
25		Режим работы выходного буфера внешнего интерфейса 0 – нормальный режим 1 – ускорение передачи данных Бит можно только установить. Сброс только аппаратно.
31:26		Всегда 0

7.4.4.1 Ширина шины данных

Физически шина данных внешнего интерфейса имеет 64 линии. Однако это не значит, что подключаемая память или внешнее устройство обязательно должны иметь такую же разрядность шины данных. Ширина шины задается отдельно для доступа к общей памяти кластера (активизируемая с помощью nMS1–0 или nMSSD3–0), доступа к памяти хост-процессора (с использованием nMSH) и для мультипроцессорного доступа к внутренней памяти процессоров кластера (с использованием мультипроцессорных адресов). Возможны различные комбинации ширины шин. Допустимые значения приведены в таблице 77.

Таблица 77 – Настройки пропускной способности шины хоста, мультипроцессорной шины и шины внешней памяти

Разряды SYSCON			Пропускная способность шины
21	20	19	
0	0	0	Пропускная способность шины хоста, мультипроцессорной шины и шины внешней памяти 32-разряда
0	0	1	Недопустимая комбинация
0	1	0	Пропускная способность шины хоста и шины внешней памяти 32-разряда. Пропускная способность мультипроцессорной шины 64-разряда
0	1	1	Пропускная способность шины хоста 32-разряда. Пропускная способность мультипроцессорной шины и шины внешней памяти 64-разряда
1	0	0	Недопустимая комбинация
1	0	1	Недопустимая комбинация
1	1	0	Пропускная способность шины хоста и мультипроцессорной шины 64-разряда Пропускная способность шины внешней памяти 32-разряда
1	1	1	Пропускная способность шины хоста, мультипроцессорной шины и шины внешней памяти 64-разряда

Некоторые комбинации запрещены из-за того, что при их установке работа системы будет некорректной. Процессор не отслеживает запрещенные комбинации, и вся ответственность за их использование лежит на пользователе.

7.4.4.2 Протокол медленного устройства для шины

Для настройки протокола медленного устройства обнуляется разряд протокола медленного устройства BNK#SLOW (HOSTSLOW) (разряд 5 для банка 0, разряд 11 для банка 1, и разряд 17 для хоста). В случае выбора протокола медленного устройства некоторые поля (поле глубины конвейера PIPE, разряд IDLE) являются безразличными и не используются. Поле ожидания определяет число внутренних циклов ожидания при медленных доступах. Если число внутренних ожиданий равно нулю, то механизм внешних ожиданий (анализ входа АСК) не может использоваться для данных транзакций.

7.4.4.3 Конвейерный протокол для шины

Для выбора конвейерного протокола бит BNK#SLOW соответствующего банка памяти устанавливается в 1, а поле PIPE устанавливается на заданную глубину конвейера (0b00 для одного цикла, 0b01 для двух циклов, 0b10 для трех циклов и 0b11 для четырех циклов). Так определяется глубина конвейера только для транзакций чтения. Глубина конвейера для транзакций записи всегда равна одному циклу. Разряд IDLE устанавливается, если на шине есть несколько подчиненных устройств в одном банке памяти, с целью предотвращения конфликта на шине данных между различными подчиненными устройствами при последовательном чтении. Если банк памяти содержит только одно подчиненное устройство, то разряд IDLE сбрасывается. Поле внутреннего ожидания безразлично при конвейерных транзакциях.

7.4.4.4 Начальные значения для работы шины

После сброса регистр SYSCON имеет начальное значение равное 0x9067. Начальное значение определяет:

- банк 0: медленный протокол, три состояния ожидания, включен холостой режим;
- банк 1: конвейерный протокол, конвейер глубиной 1 цикл, нет состояний ожидания, включен холостой режим;
- Хост: конвейерный протокол, конвейер глубиной два цикла, нет состояний ожидания, включен холостой режим.

7.4.5 Интерфейс конвейерного протокола

Процессор использует конвейерный протокол для соединения с другими процессорами, хостом и быстрой синхронной памятью, подключаемой к nMS1-0.

Транзакции также выполняются согласно конвейерному протоколу, когда к процессору выполняется доступ как к подчиненному устройству. Конвейерный протокол обеспечивает передачу данных со скоростью частоты внешней шины. При конвейерном доступе адрес и управляющие разряды выдаются одновременно в одном такте, а данные лишь спустя несколько циклов: от одного до четырех в зависимости от направления передачи и установленных параметров. Процессор может выдавать адрес следующей транзакции еще до прихода данных предыдущей транзакции.

Управляющие сигналы, используемые в конвейерных транзакциях:

- **nRD** – если активен, указывает на транзакцию чтения.
- **nWRH** и **nWRL** – транзакция записи, если один из этих сигналов активный.

Два сигнала позволяют определить, какое слово на 64-разрядной шине данных является достоверным.

- **nBRST** – указывает на блочный цикл обмена, когда считывается группа слов, расположенных в последовательных адресах. Например, при доступе к счетверенному слову на шине шириной 32 разряда, nBRST будет активным в течение первых трех циклов транзакции, состоящей из четырех циклов.
- **ACK** – управляется адресуемым подчиненным устройством в течение цикла данных. Если имеет высокий уровень, то подчиненное устройство готово завершить цикл данных, в противном случае мастер шины должен ждать готовности подчиненного устройства.

7.4.5.1 Базовая конвейерная транзакция

Временная диаграмма базовой конвейерной транзакции представлена на рисунке ниже (Рисунок 17). Во время адресного цикла, адрес передается вместе с сигналами nRD или WRx (nWRL, nWRH или оба).

Цикл данных может начинаться с задержкой от одного до четырех тактов, в соответствии с параметрами, указанным в регистре конфигурации для подчиненного устройства, и направлением транзакции. Задержка между циклом адреса и циклом данных есть глубина конвейера. В примере на рисунке глубина конвейера равна 2. В цикле данных данные передаются в соответствии с направлением транзакции (чтение, либо запись). Если подчиненное устройство готово к завершению транзакции, то оно активирует сигнал ACK (равен 1) и принимает, либо выставляет данные. Если подчиненное устройство не готово, оно деактивирует сигнал ACK (равен 0) и задерживает данные.

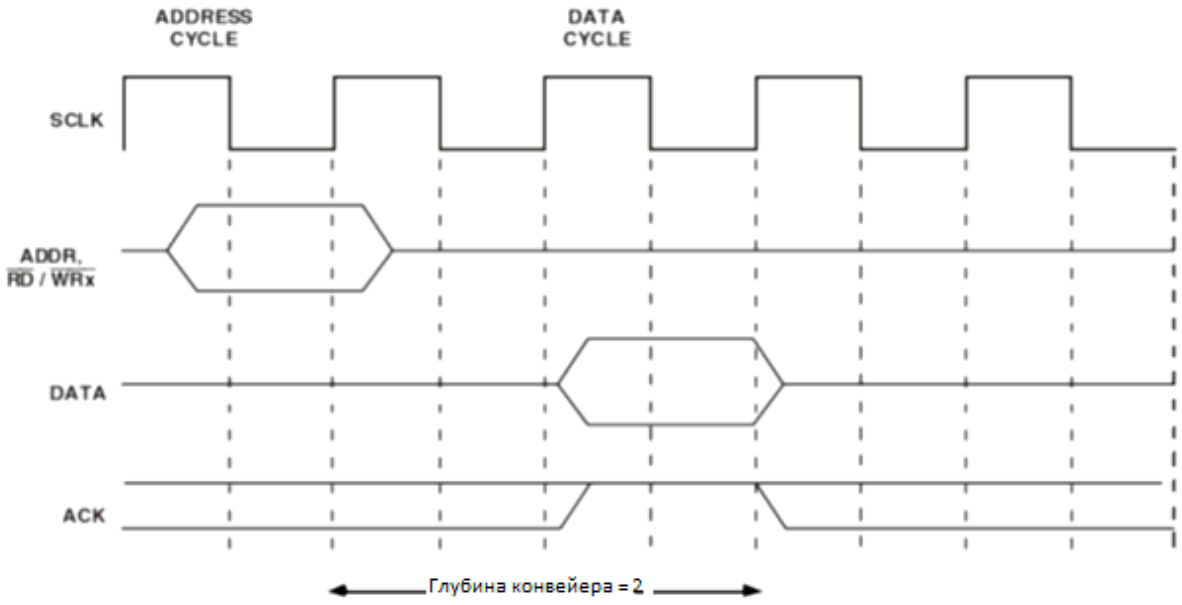


Рисунок 17 – Основные конвейерные передачи

Глубина конвейера программируется для каждого из банков памяти, а также зависит от типа транзакции (чтение или запись). Для записи глубина конвейера всегда равна 1, т.е. данные выставляются мастером на шину в следующем такте после выдачи адреса. При чтении из другого процессора в мультипроцессорных системах глубина конвейера всегда 4. При чтении из внешних банков памяти или пространства памяти хоста глубина конвейера программируется пользователем и может быть до 4 циклов. Глубина конвейера может быть выбрана индивидуально для каждого их банков в регистре SYSCON после сброса.

Одиночные транзакции занимают различное число циклов в зависимости от размера транзакции и ширины внешней шины:

- транзакция стандартного слова: один цикл;
- транзакция двойного слова на 64-разрядной шине: один цикл;
- транзакция двойного слова на 32-разрядной шине: два цикла;
- транзакция счетверенного слова на 64-разрядной шине: два цикла;
- транзакция счетверенного слова на 32-разрядной шине: четыре цикла.

Тип транзакции определяется сигналами nRD и nWRL/nWRH во время цикла адреса. Сигналы nWRL и nWRH также определяют, какое слово на 64-разрядной шине должно быть записано в память.

7.4.5.2 Пакетные конвейерные транзакции

Конвейер используется эффективно, когда мастер выполняет последовательные доступы (доступ к словам, расположенным в последовательных адресах), которые имеют одинаковую глубину конвейера. Последовательные доступы могут быть результатом одной транзакции, которая слишком широка для ширины шины. Этот случай называется пакетной передачей и идентифицируется выводом nBRST.

Сигнал nBRST используется для обозначения продолжительности одной транзакции в циклах конвейерного протокола. Он указывает, что следующий цикл является последовательным и принадлежит той же транзакции, что и текущий цикл.

Сигнал nBRST является важным для процессора, когда он выступает в качестве подчиненного устройства. Он позволяет процессору определить, что

некоторое множество циклов данных являются одной транзакцией мастера и должны интерпретироваться как одна транзакции внутри процессора.

Например, мультипроцессорная система, где один ведущий процессор пытается получить доступ к регистрам ТСВ контроллера DMA другого процессора. Поскольку доступ к регистрам ТСВ должен выполняться только с помощью счетверенных слов, то передача из нескольких слов через системную шину может быть подтверждена сигналом nBRST, и это позволит подчиненному процессору правильно интерпретировать этот доступ как доступ счетверенного слова. Без сигнала nBRST, такой же доступ будет интерпретироваться как четыре раздельных доступа нормального слова (32 разряда), что повлечет за собой недопустимый доступ к регистру ТСВ.

Настраиваемое подчиненное устройство (т.е., устройство, не являющееся подчиненным процессором) может использовать сигнал nBRST для принятия первого адреса в пакетной передаче и затем автоматически увеличивать адрес по мере поступления слов данных.

Сигнал nBRST должен быть подключен в мультипроцессорной системе с целью идентификации обменов двойными и счетверенными словами между процессорами. Сигнал nBRST может использоваться и при обменах между хост-процессором и процессором. Возможность определения пакетного доступа позволяет эффективно использовать внутренние 128-разрядные шины и минимизировать количество конфликтов доступа.

Рисунки ниже (Рисунок 17, Рисунок 18) иллюстрируют примеры различных типов последовательных передач на 32-разрядной шине. Запись счетверенного слова выполняется за четыре цикла (Рисунок 17), где вывод nBRST активируется в первых трех циклах транзакции. Глубина конвейера равна одному циклу, поскольку это транзакция записи.

Ниже представлены две транзакции чтения счетверённых слов (Рисунок 18), где каждая транзакция занимает четыре цикла шины. Две транзакции разделены холостым циклом. Холостой цикл вставляется в следующих случаях:

- за чтением одного банка памяти следует чтение из другого банка памяти, разных процессоров, хоста или комбинаций всего вышеуказанного и если при этом глубина конвейера второго чтения меньше либо равна глубине конвейера первого чтения;
- за чтением следует чтение из того же банка памяти и установлен разряд IDLE данного банка памяти регистра SYSCON;
- за чтением следует запись, либо за записью следует чтение, если глубина конвейера второй транзакции меньше либо равна глубине конвейера первой транзакции;
- за записью следует запись в мультипроцессорное пространство;
- за записью в мультипроцессорное пространство следует запись в другое подчиненное устройство;
- запись или чтение в/из мультипроцессорного пространства, если следующая транзакция связана с SRAM или памятью хоста. Холостые циклы вставляются, пока не закончится транзакция.

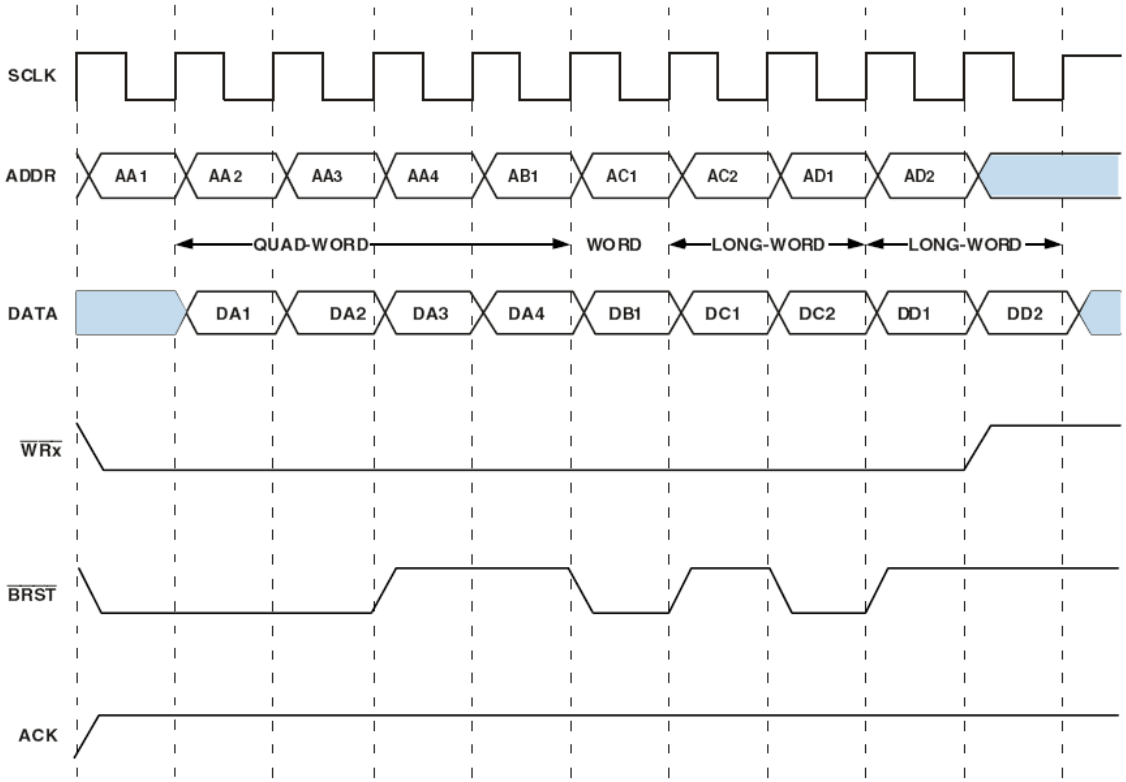


Рисунок 18 – Конвейерные транзакции – последовательные записи. Глубина конвейера = 1, пропускная способность шины = 32

Если соответствующий разряд IDLE сброшен, последовательные записи или последовательности чтения из одного и того же подчиненного устройства не разделяются циклом простоя.

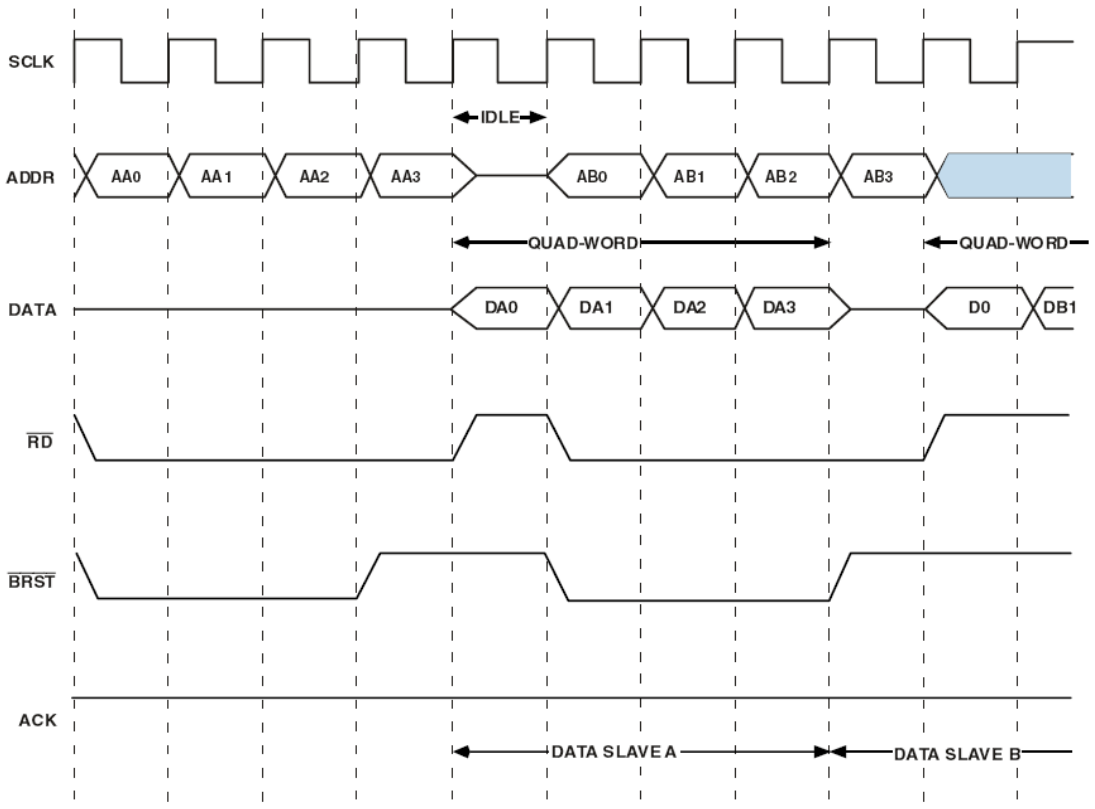


Рисунок 19 – Групповое чтение с последующим групповым чтением в том же банке памяти. Глубина конвейера = 4, пропускная способность шины = 32, установка разряда IDLE

Последовательные доступы с различной глубиной конвейера получают специальную обработку. Если первая транзакция имеет глубину конвейера меньшую, чем вторая транзакция, то адресные циклы выдаются без разрывов, даже в том случае, когда разрыв возможен между циклами данных. Если глубина конвейера первой транзакции больше глубины конвейера второй транзакции, то такие транзакции не могут быть последовательными. Как показано на рисунке ниже (Рисунок 20), адресный цикл второй транзакции задерживается после первой транзакции.

Если разница между глубинами конвейеров равна единице, то вводится холостой цикл. Адресный цикл второй транзакции начинается спустя два цикла после окончания первой транзакции.

Последовательность, представленная на рисунке, предполагает различную глубину конвейеров и ширину шины 32 разряда.

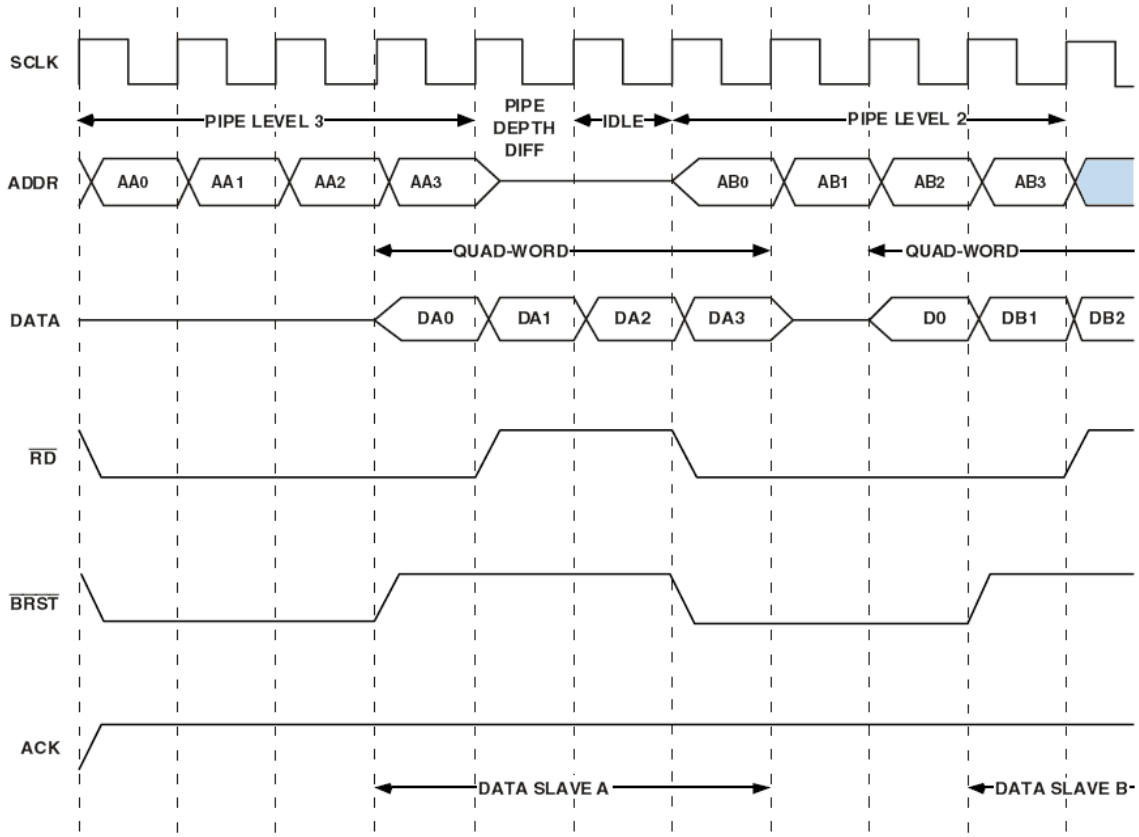


Рисунок 20 – Чтение с последующим чтением из другого подчиненного устройства

7.4.5.3 Циклы ожидания

В транзакциях чтения, если подчиненное устройство вовремя готово к циклу данных выполняемой транзакции, то подчиненное устройство активирует сигнал ACK в цикле данных. Если подчиненное устройство не готово, оно деактивирует сигнал ACK в цикле данных и держит его неактивным (равным 0) пока не будет готово продолжить транзакцию. ACK также может быть деактивирован после транзакции записи, если подчиненное устройство не может вовремя отправить данные в место назначения.

Особенности в работе сигнала ACK:

- Подчиненные устройства могут выставлять на линию ACK только активный низкий уровень. Это говорит о том, что ACK может быть переведен в низкий уровень несколькими подчинёнными процессорами одновременно без конфликтов.
- ACK поддерживается в высоком уровне внутренним нагрузочным подтягивающим резистором 50 Ом/5 кОм.
- ACK переводится в низкий уровень подчиненным процессором с целью чтобы остановить доступ.
- ACK переводится в высокий уровень внутренними подтягивающими резисторами 50 Ом/5 кОм. На высоком уровне сигнала SCLK площадка ACK подключена к питанию через резистор 50 Ом, на низком уровне сигнала SCLK – через резистор 5 кОм. И далее удерживается в высоком уровне этим резистором. Отметим одно очень важное требование к переводу сигнала ACK в высокий уровень. Когда подчиненное устройство прекращает выставлять активный низкий уровень на выход ACK (по положительному фронту сигнала SCLK), оно предполагает, что в течение следующего такта линия ACK будет иметь высокий уровень за счет использования повышающего резистора. Если резистор не способен обеспечить такую скорость заряда в течение одного тактового периода, то поведение системы непредсказуемо. В этом случае нужно снижать частоту системной шины или установить на плату дополнительный внешний резистор, например, номиналом 300 Ом.

Сигнал ACK может быть выставлен в любом цикле данных в многоцикловых транзакциях. Нет никаких ограничений на продолжительность удерживания ACK в нуле.

Рисунок 21 демонстрирует использование сигнала ACK. В этом примере, данные DA1 не готовы вовремя и выставляются на один такт позже. Сигнал ACK деактивируется во время цикла данных DA1 и активируется в следующем цикле, чтобы указать на достоверные данные. Из-за этого цикла ожидания:

- мастер берет с шины данные на один цикл позже.
- адресный цикл, следующий за сигналом ACK (AB2) растягивается на один цикл.
- все подчиненные устройства с приостановленными транзакциями во время цикла, следующего за сигналом ACK, замедляются на число циклов, которые были деактивированы сигналом ACK. Например, между адресным циклом AB0 и циклом данных DB0 проходит пять циклов, хотя глубина конвейера равна только четырем.

При этом адресные и управляющие сигналы на линиях в цикле, где ACK был в нуле, должны защелкиваться подчиненными устройствами.

Транзакции записи используют циклы ожидания ACK не так как транзакции чтения. В случае записи подчиненное устройство активирует сигнал ACK, пока у него имеется свободное место в буфере приема.

Рисунок 21 представляет пример, в котором адресный цикл AA3 заставляет подчиненное устройство деактивировать сигнал ACK. Это может быть вызвано тем, что подчиненное устройство приняло 128-разрядное слово в буфер, и он занят для следующей транзакции. Остановка удлиняет конвейер транзакции на два цикла. Циклы, которые были остановлены ACK, начинаются спустя один такт после деактивации сигнала ACK – например, AB2 и DB1 дополнены двумя циклами.

При этом все подчиненные устройства принимают адрес AB1, а устройство, которому адресованы данные DB0, также принимает эти данные во внутренний буфер.

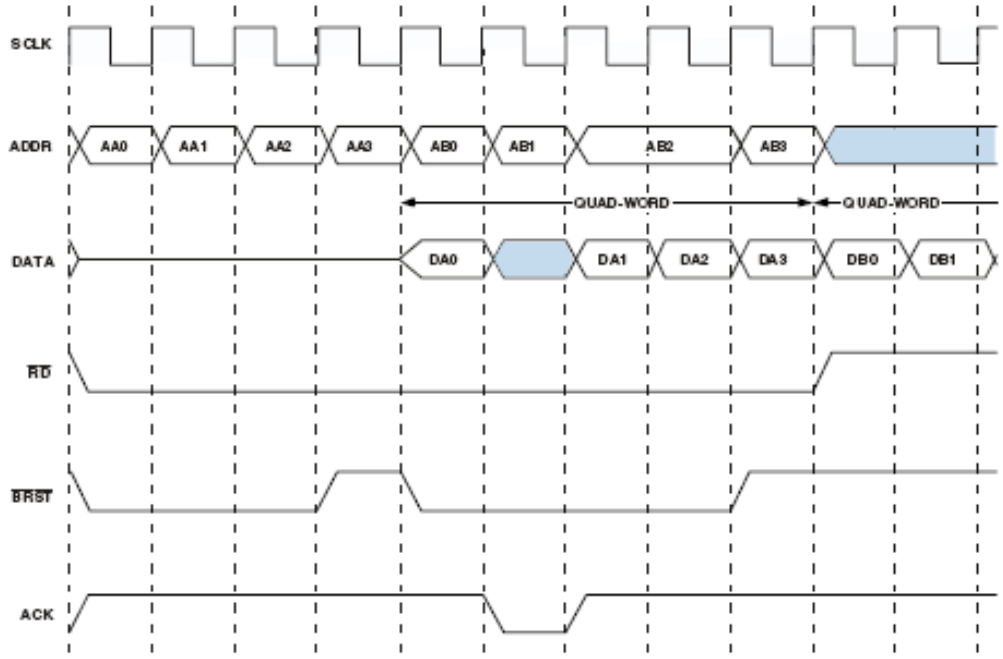


Рисунок 21 – Доступ к пакетному чтению с последующим пакетным чтением

На рисунке ниже (Рисунок 22) транзакции AB0 и AB1 адресуют другое подчиненное устройство. Если первое подчиненное устройство переводит сигнал ACK на низкий уровень, и следующие транзакции соответствуют другому подчиненному устройству, то новое подчиненное устройство транзакции не может управлять сигналом ACK до тех пор, пока первое подчиненное устройство не завершит транзакцию.

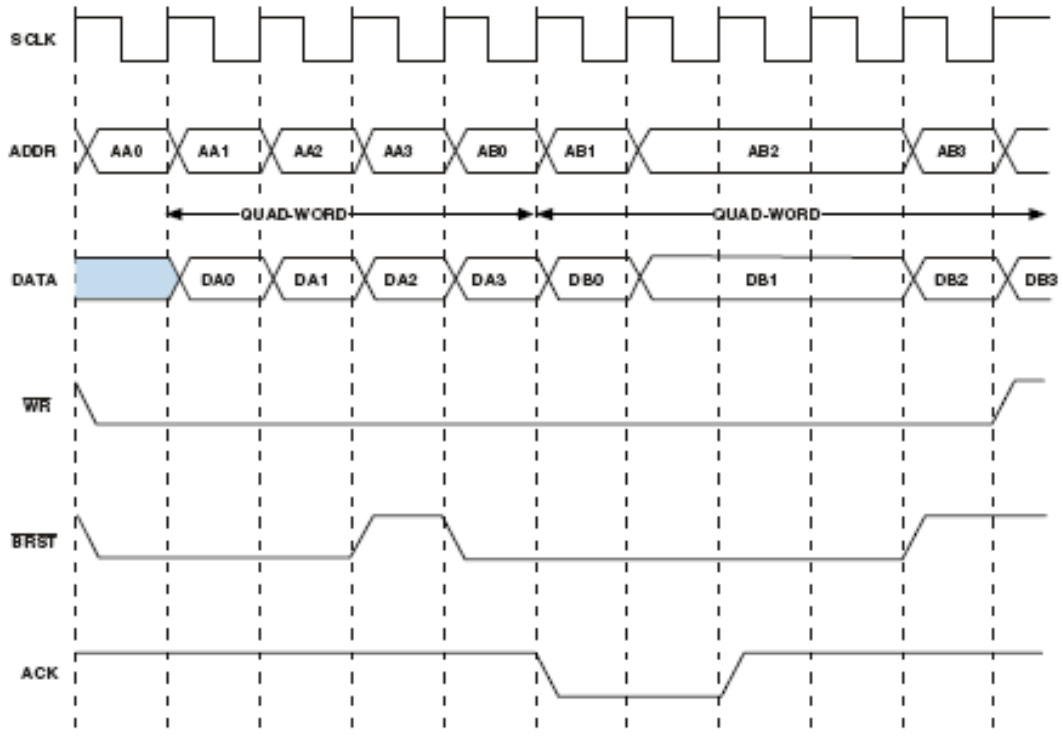


Рисунок 22 – Доступ к пакетной записи с последующей пакетной записью

7.4.6 Интерфейс протокола медленного устройства

Процессор поддерживает протокол медленных устройств, который может быть использован для простых устройств. Этот протокол может быть сконфигурирован для адресных пространств, принадлежащих банку 0, банку 1 или хосту. Протокол устанавливается посредством программирования соответствующих разрядов в регистре SYSCON:

- бит BNK#SLOW (HOSTSLOW) равен 1;
- биты PIPE глубины конвейера равны нулю;
- циклы ожидания WAIT программируются в соответствии с требованиями системы;
- цикл IDLE (холостой цикл) равен 1.

Целью таких установок является прямое соединение с простыми, медленными, некритическими модулями памяти или периферийными устройствами. Протокол может работать с синхронными и асинхронными устройствами. Базовый протокол с конфигурацией «без циклов ожидания» представлен на рисунках ниже (Рисунок 23, Рисунок 24).

В первом цикле выдается адрес и активируется линия выбора соответствующего банка памяти.

Во втором цикле активируются сигналы nRD или WRx (в зависимости от типа транзакции). Если транзакция записи, то в этом цикле процессор выдает данные. В случае транзакции чтения в этом цикле подчиненное устройство начинает выдавать данные. В случае чтения, в конце цикла процессор защелкивает данные с шины во внутреннем регистре.

В третьем цикле процессор продолжает удерживать адрес и данные (если выполняется запись), но деактивирует сигналы nRD или WRx. В случае чтения устройство перестает выдавать данные на шину.

Для конфигурации «без циклов ожидания», циклы ожидания не могут быть добавлены деактивирующим сигналом ACK.

Одно из ключевых требований при организации протокола медленного устройства – гарантированное время удержания адреса и данных. Из временной диаграммы видно, что процессор обеспечивает один такт предустановки адреса и один такт удержания адреса и данных. Дополнительные такты предустановки данных могут обеспечиваться внутренними программируемыми циклами ожидания или внешним сигналом ACK.

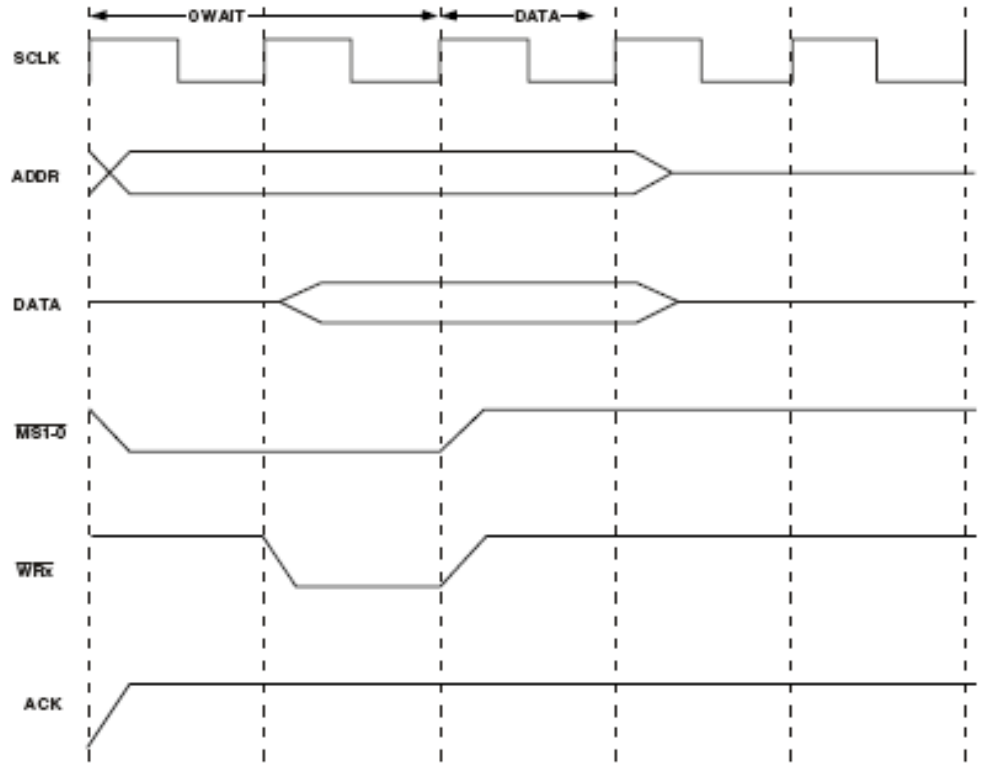


Рисунок 23 – Медленный протокол. Запись с нулевым временем ожидания

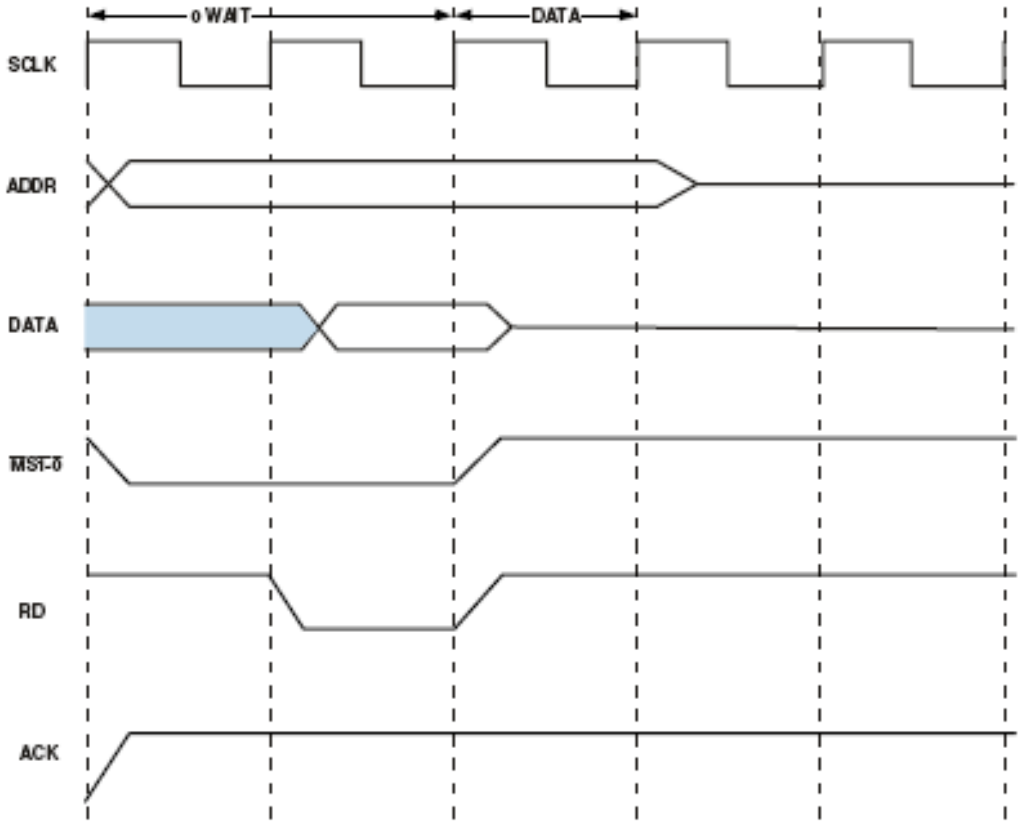


Рисунок 24 – Медленный протокол. Чтение с нулевым временем ожидания

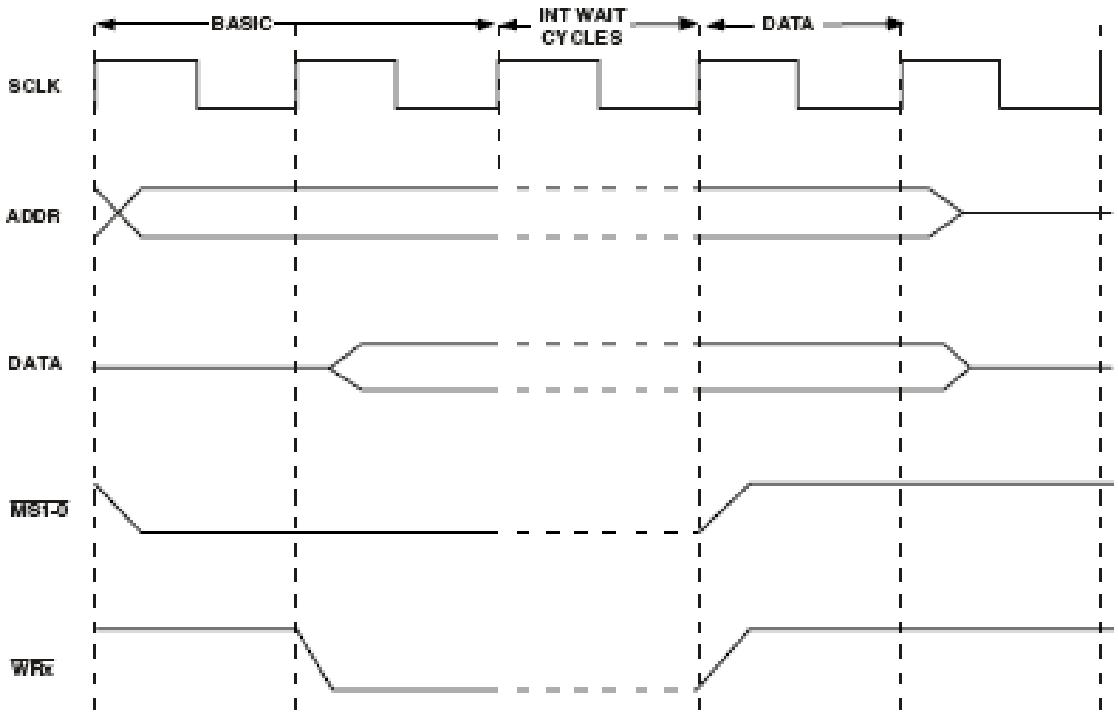


Рисунок 25 – Медленный протокол. Запись с циклами ожидания

В протоколе медленного устройства при необходимости могут быть вставлены циклы ожидания. Регистр SYSCON включает в себя поле задания некоторого числа внутренних циклов ожидания. Значения могут быть от нуля до трех.

На рисунках (Рисунок 25, Рисунок 26) показаны медленные транзакции с одним или более циклами ожидания. Ситуация похожа на транзакцию без циклов ожидания, но второй цикл (цикл в котором активированы nRD или WRx) повторяется в соответствии с числом циклов внутреннего ожидания.

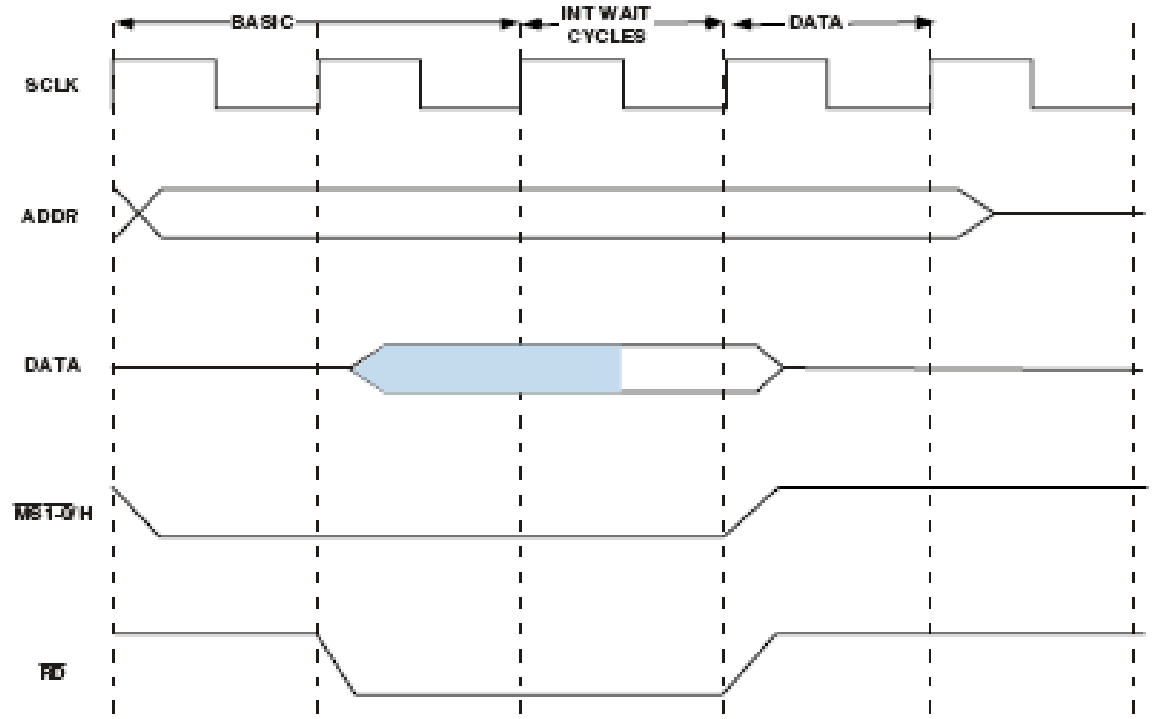


Рисунок 26 – Медленный протокол. Чтение с циклами ожидания

Дополнительные циклы внешнего ожидания могут быть вставлены путем деактивации сигнала ACK. Это может быть сделано только тогда, когда в регистре SYSCON поле WAIT не равно нулю. Для того, чтобы добавить циклы внешнего ожидания, сигнал ACK должен быть деактивирован в течение одного или более циклов перед последним циклом ожидания (см. Рисунок 27).

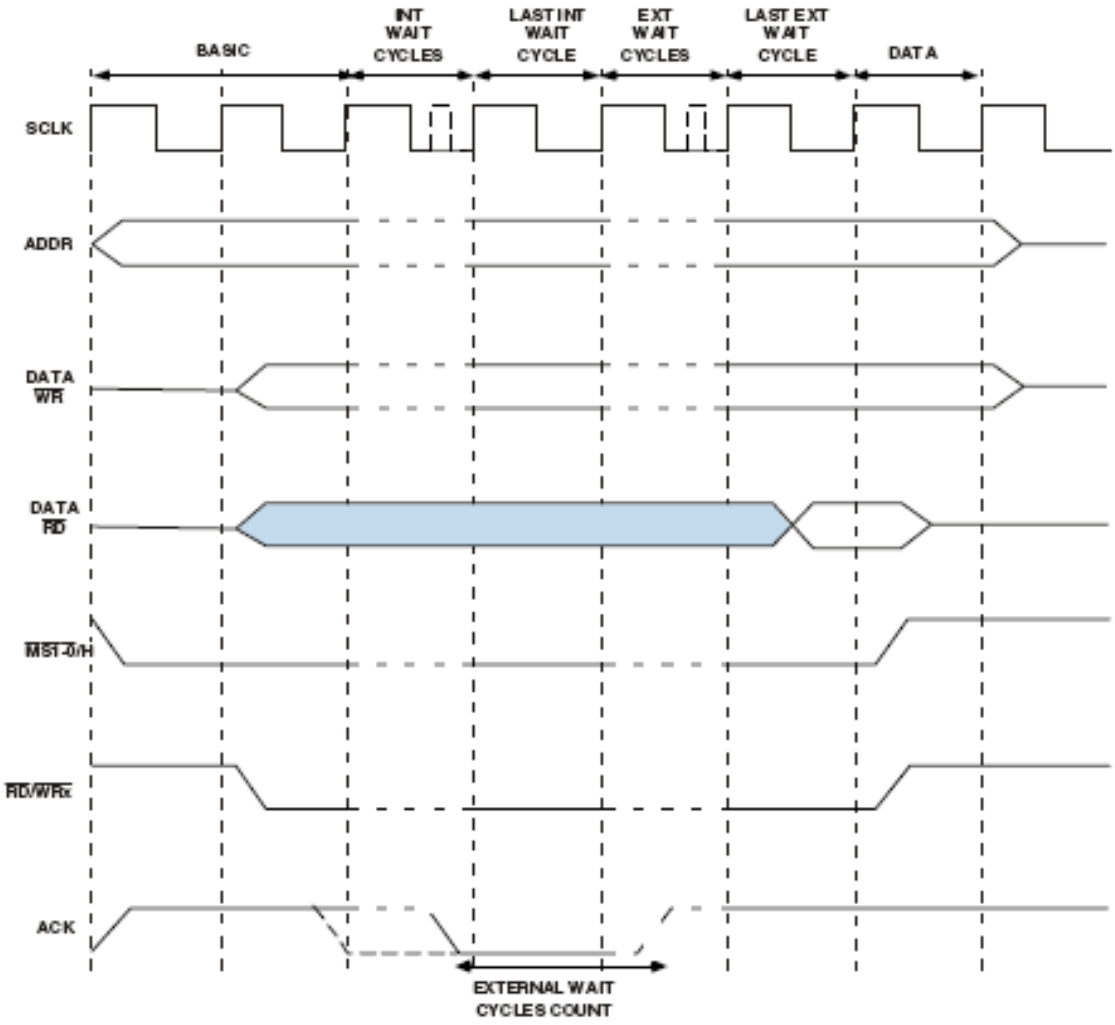


Рисунок 27 – Медленный протокол. Запись/чтение с внешними циклами ожидания

7.4.7 Интерфейс EPROM

Процессор имеет возможность работать с внешней 8-разрядной перепрограммируемой памятью типа EPROM. Во время сброса процессор может быть сконфигурирован для загрузки из внешнего EPROM. В этом случае программа загружается из EPROM во внутреннюю память автоматически как часть последовательности сброса. Для этой цели используется канал 0 DMA.

Поскольку разрядность шины данных памяти EPROM составляет 8 бит, внешний интерфейс процессора упаковывает принимаемые байты в 32-разрядные данные.

Процессор может работать с EPROM не только во время сброса. Имеется возможность доступа и в процессе выполнения программы. Однако в работе с EPROM имеется одна особенность – память EPROM не является частью адресного пространства процессора. Процессор поддерживает словную адресацию внешней памяти, а память EPROM имеет байтовую организацию. В связи с этим для работы с EPROM можно использовать только канал общего назначения DMA. Для работы с EPROM тип обмена в ТСВ канала устанавливается равным 6. При чтении EPROM, обмен может происходить только между EPROM и внутренней памятью. Внешний интерфейс определяет, что текущий запрос есть запрос DMA и его тип равен 6. Это позволяет интерфейсу выбрать для обмена протокол медленного устройства с фиксированным числом циклов внутреннего ожидания равным 16. При выполнении внешней транзакции процессор активизирует линию nBMS для выбора EPROM. Для передачи данных используются биты с 0 по 7-й шины данных.

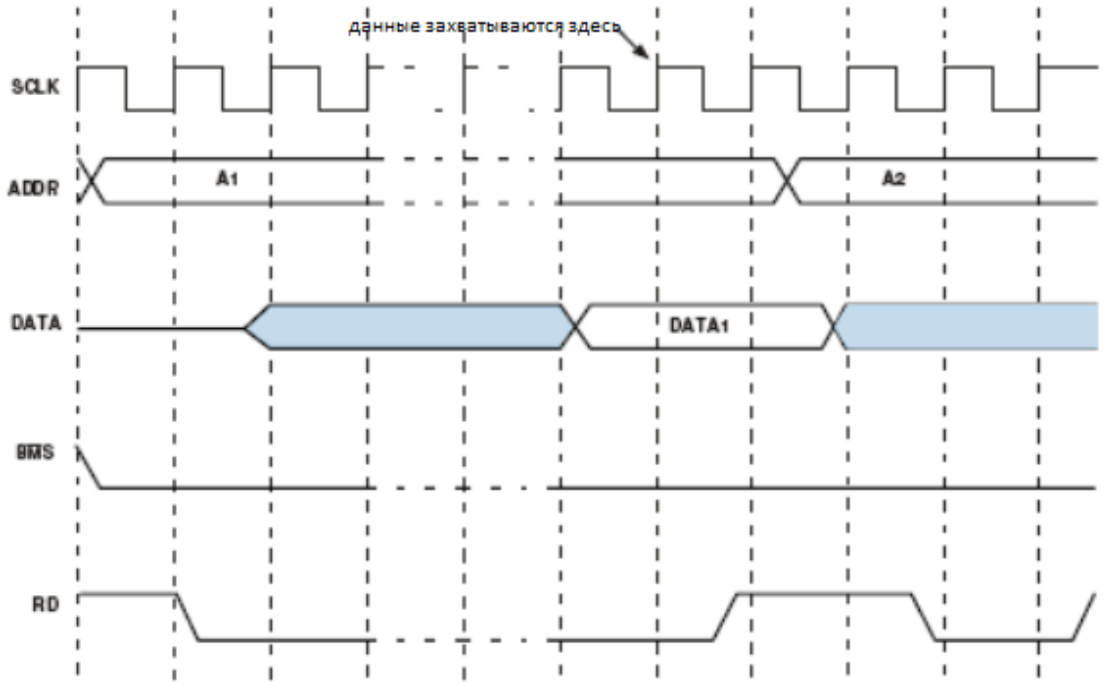


Рисунок 28 – Доступ к загрузке EPROM – 16 циклов ожидания

Возможно как чтение внешней EPROM, так и операция записи для программирования памяти. Во время чтения интерфейс упаковывает байты в слова и передает их во внутреннюю память. Операция записи идентична чтению, но с одним отличием – при записи нет распаковки слова в байты. Процессор может выдавать на внешнюю шину только 32-разрядные слова, а EPROM может принимать только младший байт. Поэтому при записи программист должен самостоятельно

подготовить информацию в удобном для записи виде. При записи EPROM источником данных может быть как внутренняя, так и внешняя память.

Во внутреннем или внешнем ОЗУ процессора данные должны быть организованы так, как показано на рисунке ниже (Рисунок 29). Слева исходные данные для программирования. Справа показаны распакованные в слова байты для передачи во внешнюю EPROM.

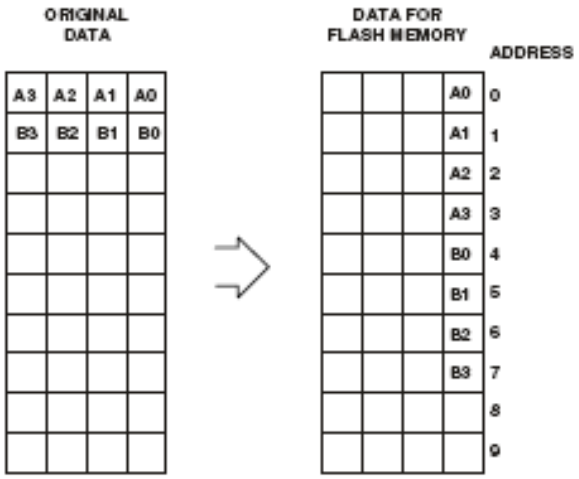


Рисунок 29 – Подготовка данных для записи в 8-разрядную EPROM

При выполнении обмена с EPROM, внешний интерфейс всегда устанавливается размер шины данных 32 разряда, несмотря на установки в SYSCON. При этом адрес EPROM не должен быть в используемом адресном пространстве хоста, SDRAM или банков MS1 и MS0. Лучше всего старшие биты адреса установить в ноль. При чтении внешний интерфейс автоматически устанавливает размер данных равным квадрослову. Это позволяет произвести 4 цикла чтения, взять из них по одному байту и сформировать одно слово. Канал DMA при этом должен выполнять обмен словами, но увеличивать адрес на 4. Таким образом, первое считывание будет по адресам 0, 1, 2, 3, а второе – по адресам 4, 5, 6, 7. Такая особенность используется при чтении.

При записи используется размер данных, который задает канал DMA. При записи внешний интерфейс работает с EPROM так, как будто эта память имеет разрядность 32 бита и передает ей слова данных. Основной моментом здесь является предварительная подготовка данных для записи в памяти.

Настройки канала DMA для работы с внешней EPROM имеет следующие особенности:

При чтении регистр TCB источника имеет следующее значение:

DI – адрес байта в памяти EPROM. Начальное значение может быть любым. Процессор всегда читает за одну транзакцию 4 последовательных байта DI, DI+1, DI+2, DI+3.

DXM – модификатор адреса. Хранит значение, которое прибавляется после каждой транзакции к регистру DI. Имеет значение количества байт. После чтения 4-х байт следующее значение регистра DI будет равно DI+DXM. Поэтому рекомендуемым значением для последовательного чтения блока байт всегда является число 4.

DXC – *количество слов,* которые будут прочитаны.

При записи регистр TCB приемника имеет следующее значение:

DI – адрес байта в памяти EPROM. Начальное значение может быть любым. Процессор всегда пишет за одну транзакцию один байт по адресу DI.

DXM – модификатор адреса. Хранит значение, которое прибавляется после каждой транзакции к регистру DI. Имеет значение количества байт. После записи текущего байта, следующее значение регистра DI будет равно DI+DXM. Поэтому рекомендуемым значением для последовательной записи блока байт всегда является число 1.

DXC – *количество байт,* которые будут записаны. Так как во внутренней или внешней памяти для хранения записываемого байта используется 32-разрядное слово (байт должен размещаться в младших битах слова), то данный счетчик эквивалентен количеству слов.

7.4.8 Мультипроцессорный интерфейс

Процессор имеет способность работать в мультипроцессорной системе с общей шиной. Для поддержки мультипроцессорных возможностей внешний интерфейс процессора имеет возможность быть мастером шины либо работать как подчиненное устройство. Также процессор включает в себя арбитражное логическое устройство, чем поддерживает распределенный арбитраж в мультипроцессорной системе. К одной внешней шине могут быть подключены до четырех процессоров и хост-процессор.

Для арбитража доступа к шине и для организации взаимодействия с хост-процессором, внешний интерфейс процессора использует специальные выводы.

- **ID2–0**
  Мультипроцессорный идентификатор (номер процессора в многопроцессорной системе). Этот номер позволяет процессору определить свое место в карте памяти системы, а также указывает процессору какой запрос шины (nBR0–nBR7) активизировать при запросе шины: 000 = BR0, 001 = BR1, 010 = BR2, 011 = BR3, 100 = BR4, 101 = BR5, 110 = BR6, 111 = BR7. Идентификатор должен иметь постоянное значение во время работы системы и может изменяться только во время сброса.
- **nBR7–0**
  Выводы запроса мультипроцессорной шины. Используется процессором в мультипроцессорной системе для осуществления арбитража доступа к внешней шине. Каждый процессор управляет своей линией nBR и осуществляет мониторинг всех остальных. В системе, где меньше четырех процессоров, необходимо установить высокий уровень сигнала на неиспользуемых выводах nBR.
- **nBM**
  Мастер шины. Текущий мастер шины активизирует вывод nBM. Вывод является информативным и не анализируется процессорами. Во время сброса является анализируемым выводом.
- **nBOFF**
  Сигнал возврата к началу цикла. Во время работы системы возможна ситуация взаимоблокировки, когда хост-процессор и процессор пытаются одновременно осуществлять чтение памяти друг друга. При взаимоблокировке хост может активировать nBOFF чтобы принудить процессор освободить шину.
- **nBUSLOCK**
  Индикатор блокировки шины. Указывает на то, что мастер шины заблокировал шину. Во время сброса является анализируемой выводом.

- **nHBR**
  Запрос мультипроцессорной шины хост-процессором. Хост может активировать nHBR для запроса доступа к внешней шине процессора. Когда nHBR активирован в мультипроцессорной системе, текущий мастер шины освобождает шину и активирует nHBG.
- **nHBG**
  Предоставление шины хост-процессору. Подтверждает nHBR и указывает на то, что хост-процессор может контролировать внешнюю шину. Во время освобождения шины, мастер переводит в третье состояние разряды ADDR31–0, DATA63–0, nMSH, nMSSD3–0, MS1–0, nRD, nWRL, nWRH, nBMS, nBRST, nIORD, nIOWR, nIOEN, nRAS, nCAS, nSDWE, SDA10, SDCKE, LDQM и HDQM. Также процессор переводит SDRAM в режим самогенерации. Процессор-мастер удерживает активным nHBG до тех пор, пока хост не деактивирует nHBR. В мультипроцессорных системах текущий мастер шины активизирует nHBG, а все подчиненные процессоры осуществляют мониторинг.
- **nCPA**
  Приоритетный доступ ядра. Активизируется, когда ядро процессора желает выполнить доступ к внешней памяти. Этот вывод разрешает подчиненному процессору прерывать передачи процессора-мастера и получить управление внешней шиной для выполнения операций, инициированных ядром. Мастер может прервать свои передачи, только если их приоритет ниже, т.е. если это передачи DMA. Процессоры могут выдавать на линию только активный низкий уровень. Высокий уровень на nCPA обеспечивается встроенным резистором.
- **nDPA**
  Приоритетный доступ DMA. Активизируется, когда высокоприоритетный DMA канал процессора желает выполнить доступ к внешней памяти. Этот вывод разрешает высокоприоритетному DMA каналу на подчиненном процессоре прерывать передачи процессора-мастера и получить управление внешней шиной для выполнения операций. Мастер может прервать свои передачи, только если их приоритет ниже, т.е. если это передачи низкоприоритетных каналов DMA. Процессоры могут выдавать на линию только активный низкий уровень. Высокий уровень на DPA обеспечивается встроенным резистором.

Все арбитры кластера работают в синхронном режиме. Арбитраж доступа к шине выполняется с использованием выводов nBR7-0, nHBR и nHBG.

Выводы nBR7-0 необходимы для арбитража между процессорами кластера, а nHBR и nHBG используются для арбитража между мастером шины и хост-процессором. Если в системе меньше четырех процессоров, то любой неиспользуемый вывод nBR должен быть деактивирован.

Выбор одного из нескольких процессоров для доступа к шине осуществляется на основе приоритета. Для данной схемы арбитража используется циклически изменяемый приоритет процессора. После сброса процессор с ID равным 00 становится мастером и далее приоритет изменяется в соответствии с циклическим алгоритмом, переходя от текущего мастера к следующему. Таким образом, приоритет процессоров после сброса от высокого к низкому будет 0, 1, 2, 3. Если процессор с номером 2 запросит шину в отсутствие запросов от других процессоров, то процессор 2 станет мастером и приоритет станет 2,3,0,1. Данная схема приоритетов действует, когда линии nCPA и nDPA неактивны, т.е. для транзакций низкоприоритетных каналов DMA. Смена приоритета прерывается, когда хост

запрашивает шину или, когда процессор запрашивает шину посредством активации выводов nCPA или nDPA.

На внешней шине могут образовываться три группы запросов имеющий различный приоритет по отношению друг к другу:

- запросы с активным nCPA;
- запросы с активным nDPA;
- запросы с неактивными nCPA и nDPA.

Внутри каждой группы используется циклическая схема приоритетов. Таким образом приоритет системы в порядке убывания:

- хост-процессор;
- процессор с активными nBR и nCPA. Соответствует транзакции ядра процессора на внешней шине;
- процессор с активными nBR и nDPA. Соответствует транзакции высокоприоритетного канала DMA на внешней шине;
- процессор с активным nBR. Соответствует транзакции низкоприоритетного канала DMA на внешней шине.

Текущий мастер не может бесконечно удерживать шину и должен передавать управление шиной другому процессору в следующих условиях:

- текущий мастер не запрашивает шину, а другое подчиненное устройство активирует свой nBR или имеется запрос nHBR;
- текущий мастер запрашивает шину, а другое подчиненное устройство активирует свой nBR и nCPA. Текущий мастер не активирует nCPA;
- текущий мастер запрашивает шину, а другое подчиненное устройство активирует свой nBR и nDPA. Текущий мастер не активирует nCPA или nDPA;
- текущий мастер запрашивает шину, а хост активирует свой nHBR;
- текущий мастер запрашивает шину, но значение регистра BMAX истекает и другое подчиненное устройство активирует свой nBR. Регистр BMAX определяет число циклов, в течение которых процессор может занимать шину. Счет происходит во внутренних циклах.

7.4.8.1 Протокол арбитража шины

Каждый такт SCLK процессоры кластера защелкивают в буфере арбитра состояние линий nBR. Когда одному из процессоров необходимо стать ведущим устройством (мастером), он активирует собственную линию nBR и в то же самое время отслеживает другие линии nBR. Текущий мастер, после выполнения своей транзакции деактивирует свой nBR. Тогда управление шиной переходит новому запрашивающему устройству. Новый мастер шины сохраняет управление шиной путем удерживания своего nBR постоянно активным.

Когда текущий мастер освобождает шину, ему присваивается низший приоритет. Когда другие подчиненные устройства собираются стать ведущими, каждый из них активирует свой nBR и подчиненное устройство с самым высоким приоритетом получает управление шиной. Оставшиеся подчиненные устройства держать активными свои nBR до момента получения шины в соответствии с приоритетом. Если текущий мастер видит, что должен освободить шину, он завершает все свои начатые транзакции до того, как освободит шину.

По причине циклов смены мастера, происходит холостой цикл, когда управление переходит от текущего к новому мастеру. На рисунке ниже (Рисунок 30) показан пример арбитража доступа для случая транзакций низкоприоритетных каналов DMA. В течение 4-х первых тактов процессор 0 мастер шины и он выполняет транзакцию. В течение этих тактов процессоры 1 и 2 формируют запросы

доступа к шине. В 5-м такте процессор 0 освобождает шину, снимая запрос с линии BR0. В 6-м такте в соответствии с циклическим алгоритмом приоритетов мастером шины становится процессор 1. В течение тактов с 6 по 9-й он выполняет свою транзакцию. В 10-м такте процессор освобождает шину, снимая запрос с линии BR1. В 11-м такте процессоры анализируют состояние запросов и видно, что есть запросы от процессоров 0 и 2. В соответствии с циклическим алгоритмом мастером шины станет процессор 2.

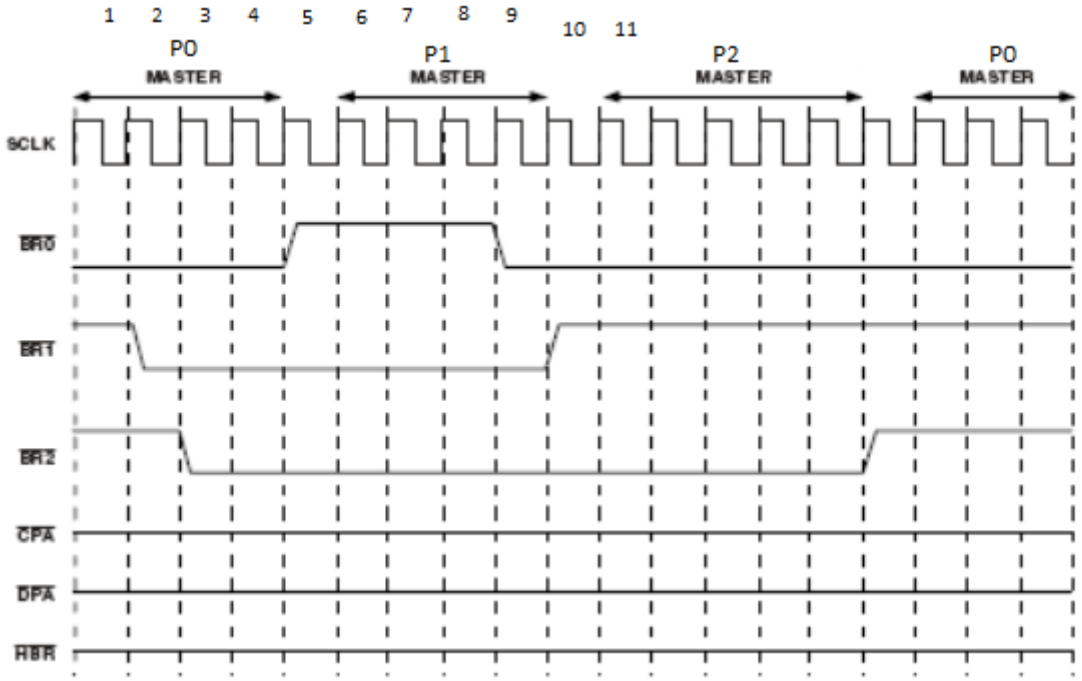


Рисунок 30 – Последовательность арбитража внешней шины, nCPA/nDPA/nHBR неактивны

Арбитраж шины позволяет каждому процессору получать временный приоритет над текущим мастером. Это достигается с использованием двух выводов nCPA и nDPA.

Хост-процессор может быть мастером шины на внешней шине процессора. Среди четырех процессоров кластера всегда есть один процессор, который в текущий момент является мастером. Именно с этим процессором договаривается хост-процессор об использовании шины. Хост использует линии nHBR и nHBG для получения контроля над шиной. Для того чтобы хост стал мастером шины, он должен активировать nHBR и ожидать пока nHBG не будет активирован текущим процессором-мастером. Процессор-мастер освобождает внешнюю шину и свидетельствует об этом активацией nHBG. При этом процессор, предоставившей шину хосту, остается формальным мастером кластера. Хост держит nHBR активным столько времени, сколько это необходимо для выполнения его транзакций. Мастер, предоставивший шину хосту, будет удерживать свою линию nBR активной, пока хост выполняет свои транзакции. Таким образом, когда хост деактивирует nHBR, формальный мастер снова станет реальным мастером.

7.4.8.2 Вывод приоритетного доступа ядра(nCPA)

Вывод nCPA активируется, когда ядро процессора желает получить доступ к внешней памяти. Это позволяет подчиненному процессору прерывать фоновые передачи канала DMA, принадлежащие процессору-мастеру, и получить возможность управления внешней шиной.

Последовательность действий арбитража может быть следующей:

- некоторый подчиненный процессор желает выполнить транзакцию ядра к внешней памяти. Для этого он выставляет активный уровень на линию nCPA и активный уровень на линию nBR.
- все другие подчиненные процессоры смотрят, что появился nCPA запрос и, если их запросы ниже приоритетом (запросы DMA), они деактивируют свои линии nBR.
- Мастер шины видит, что появился nCPA запрос и запрос его текущей транзакции ниже (транзакция DMA). Мастер завершает текущую транзакцию и деактивирует свою линию nBR. При этом он остается формальным мастером шины.
- После выдачи nCPA запроса, на шине будут активны линии nBR только у процессоров, которые желают выполнить транзакции ядра. Эти процессоры будут выполнять арбитраж шины между собой. При этом процессор получивший шину станет ***временным*** мастером. После завершения транзакции ядра он снимет свой nCPA запрос, и предыдущий мастер опять станет реальным мастером шины. Если вдруг появится запрос от хост-процессора, то именно данный временный мастер будет выполнять арбитраж с хостом.

На рисунке ниже (Рисунок 31) показан пример арбитража при наличии nCPA запроса. Пример усложнен дополнительным запросом от хост-процессора. На рисунке процессоры 1 и 2 одновременно желают выполнить транзакции ядра и выставляют nCPA запросы. Процессор 0 анализирует эти запросы и снимает свой запрос. Временным мастером становится процессор 1 (согласно циклическому алгоритму). В это время приходит запрос от хоста, и временный мастер берет на себя обязанности за предоставление шины хост-процессору. После завершения транзакций хостом, процессор 1 возвращает себе управление шиной.

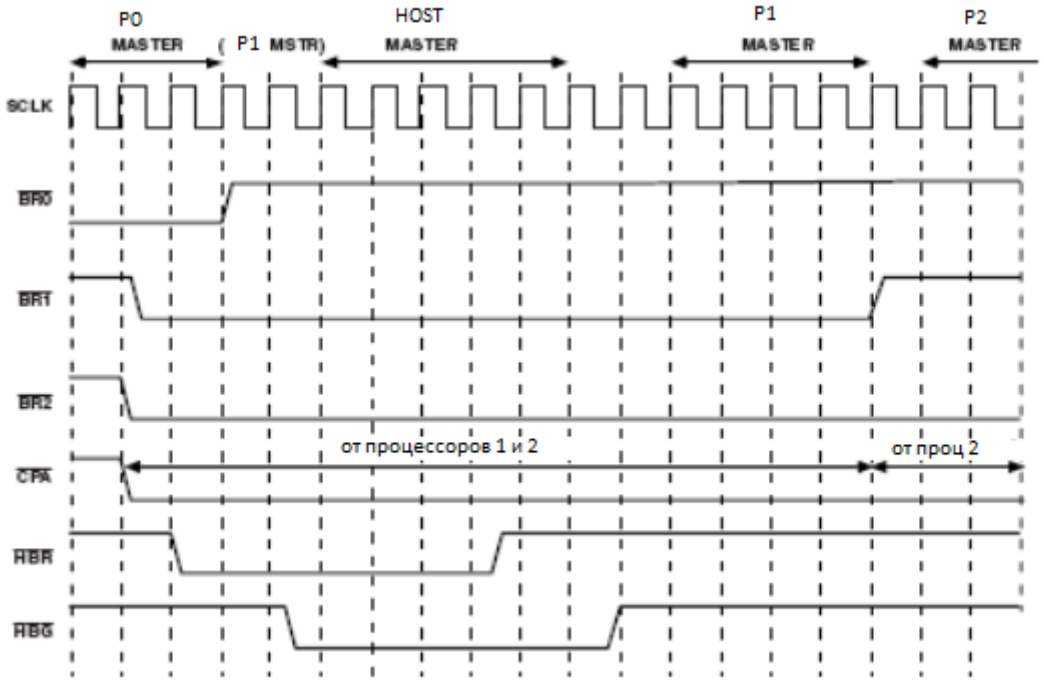


Рисунок 31 – Последовательность арбитража внешней шины; Более одного активного nCPA; nHBR активный

7.4.8.3 Вывод приоритетного доступа DMA (nDPA)

Вывод nDPA активируется, когда высокоприоритетный канал DMA процессора получает доступ к внешней шине. Это позволяет каналу DMA с высоким приоритетом, принадлежащему подчиненному процессору, прерывать фоновые передачи канала DMA с обычным приоритетом, принадлежащего управляющему процессору. Запросы nDPA могут активироваться только в случае, когда nCPA не активирован, т.е. они имеют более низкий приоритет, чем запросы ядра. В случае наличия nCPA запроса, все nDPA запросы деактивируются как обычные запросы.

В случае появления nDPA запроса, текущий мастер останавливает свои транзакции (если приоритет его транзакций ниже) и передает управление шиной запрашивающему процессору. Передача шины выполняется посредством деактивации линии nBR. Когда nDPA активирован, то только процессоры с высокоприоритетными транзакциями DMA могут направлять запрос шине.

Последовательность действий процессоров в системе в случае nDPA запроса полностью аналогична действиям в случае nCPA запроса, описанного выше. Пример такого запроса отображен на рисунке ниже (Рисунок 32). На рисунке процессор 2 выставляет nDPA запрос. В этот момент процессор 0 является мастером шины, а процессор 1 выставил обычный запрос. В ответ на nDPA запрос процессоры 0 и 1 деактивируют свои линии nBR. Процессор 2 становится временным мастером и выполняет транзакцию. После выполнения транзакции снимается nDPA и nBR. Управление переходит к процессору 0 (прерванный мастер).

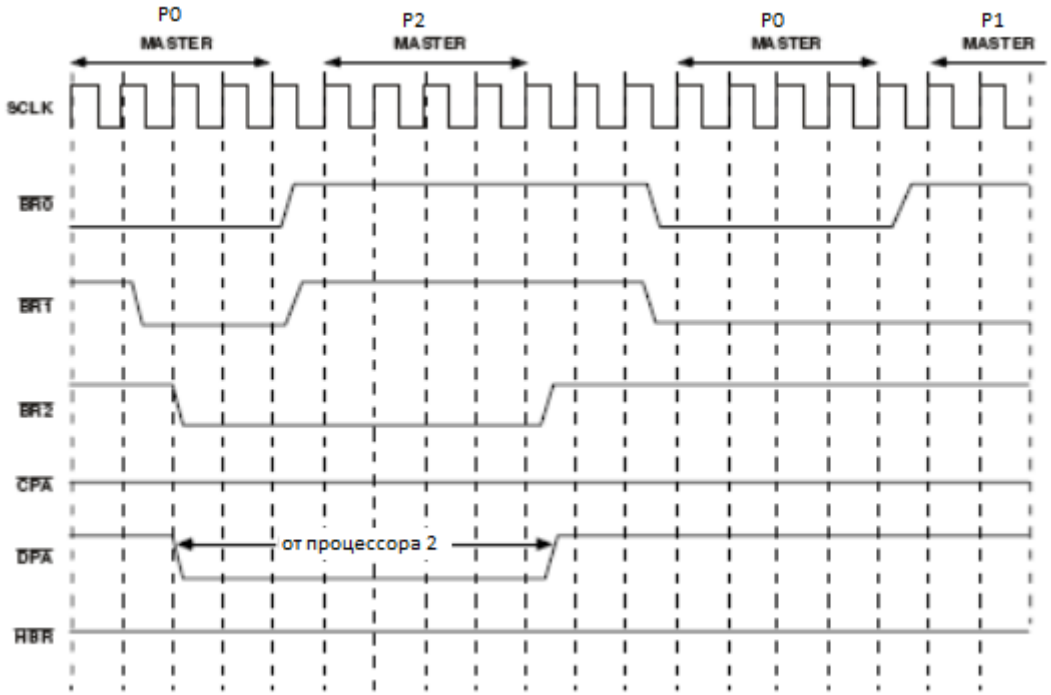


Рисунок 32 – Последовательность арбитража внешней шины; только один активный nDPA; nHBR неактивный

В процессе арбитража, запрашивающие шину процессоры с активным nCPA имеют более высокий приоритет, чем запрашивающие шину процессоры с активным nDPA. Это означает, что процессоры с активным nDPA деактивируют свои запросы nBR после обнаружения того, что линия nCPA активирована другими процессорами шины.

Если шина принадлежит процессору, который установил блокировку шины (используя регистр nBUSLOCK), то приоритет блокировки шины является

практически самым высоким приоритетом в системе. В этом случае арбитр внешнего интерфейса не освободит шину, даже если имеются nCPA, nDPA или nHBR запросы. Также не освобождает шину и окончание отсчета счетчика BMAX. Только сбросив бит блокировки шины возможно освобождение шины для других процессоров.

7.4.8.4 Регистр управления арбитражем шины (BMAX)

Системному разработчику может понадобиться ограничить время удержания шины одним процессором с целью предотвращения недостатка шины для других процессоров в системе. Регистр BMAX определяет максимальный период времени в течение, которого процессор может быть мастером шины. Время определяется в циклах SOCCLK (или CCLK/2).

Счетчик BMAX инициализируется определенным значением в момент, когда процессор захватывает шину и становится мастером. С этого момента счетчик начинает обратный отсчет. Если к мастеру будет выполнен запрос шины со стороны хост-процессора, то счетчик остановит обратный отсчет на время пока хост пользуется шиной. Когда счетчик достигает нулевого значения, арбитр шины проверяет наличие запросов от других процессоров. Если таковых нет, то счетчик опять инициализируется и начинает обратный отсчет. Если запросы имеются, процессор освобождает шину и предоставляет её другому процессору.

Значение 0xFFFF используется для выключения счетчика. Такое значение имеет счетчик после сброса. В этом случае он не влияет на работу арбитра шины.

Когда устанавливается разряд блокировки шины (BUSLK) в регистре nBUSLOCK, то завершение счета BMAX игнорируется до тех пор, пока не сбрасывается блокировка шины. Как только бит блокировки сбрасывается – шина освобождается.

7.4.8.5 Регистр блокировки шины (nBUSLOCK)

Функция захвата шины выполняется с целью поддержки атомарных операций чтение-модификация-запись. Когда процессору необходимо получить доступ к семафору, процессор запрашивает захват шины посредством установки разряда BUSLK в регистр nBUSLOCK. Установка этого разряда заставляет арбитра шины сделать запрос шины, выполнив активацию nBR в течение всего периода, пока установлен разряд BUSLK. Как только разряд BUSLK сбрасывается, арбитраж приобретает стандартный порядок.

После установки бита захвата шины, перед доступом к семафору, процессор может проверить выполнил ли арбитр захват шины. Это может быть сделано посредством чтения регистра состояния SYSTAT.

Процессор также может проверить является ли он мастером шины с помощью специальной команды условного перехода.

Также имеется возможность генерации прерывания при захвате шины процессором.

7.4.8.6 Операция программного сброса ядра процессора

Внешний порт продолжает свою работу даже тогда, когда процессор находится в состоянии программного сброса (SWRST бит равен 1). В состоянии программного сброса интерфейс шины обеспечивает следующее:

- арбитраж шины – арбитраж на кластерной шине продолжается, но процессор, находящийся в программном сбросе, не может сделать запрос шины.
- реакция транзакции – внешний порт IFIFO реагирует на транзакции кластерной шины, нацеленные на процессор, но эти транзакции задерживаются в IFIFO, пока не завершится программный сброс.

7.4.9 Интерфейс хост-процессора

Под хост-процессором в данном случае понимаем управляющее устройство, внешнее по отношению к кластеру процессоров. Интерфейс хост-процессора соединяет хост с внешней мультипроцессорной шиной процессорного кластера.

Хост может быть подчиненным или ведущим устройством шины. Когда хост является подчиненным устройством, для доступа к нему со стороны процессора может использоваться конвейерный протокол или протокол медленного устройства, в зависимости от конфигурации интерфейса, указанной в регистре SYSCON. Когда хост становится ведущим устройством, он может получить доступ к процессору кластера, используя тот же конвейерный протокол, что и любой другой процессор. Также хост может иметь доступ к любому из подчиненных устройств в системе (память, устройства ввода-вывода), используя свой собственный протокол.

Текущий процессор-мастер шины предоставляет шину хост-процессору только после полного завершения своей транзакции. Мастер возобновляет транзакции только после того как хост снял запрос шины. Пример цикла чтения хост-процессором отражен на рисунке ниже (Рисунок 33).

Когда хост выполняет доступ к процессору, то его протокол должен полностью отвечать всем правилам конвейерных транзакций процессора. Наиболее важные правила:

- глубина конвейера всегда 4 для чтения и 1 для записи;
- не разрешается чтение из широковещательного адресного пространства;
- хост должен работать с той же конфигурацией системы, что и другие процессоры на кластерной шине;
- хост не получает доступа к шине пока процессор (ID 0) не инициализировал SDRAM (если запрос от хоста пришел во время инициализации);
- процессор переводит SDRAM в режим саморегенерации до того, как освободит шину для хоста. Хост, с целью использования SDRAM, должен изменить режим SDRAM на нормальный рабочий режим;
- перед возвратом шины процессору хост должен снова поместить SDRAM в режим саморегенерации.

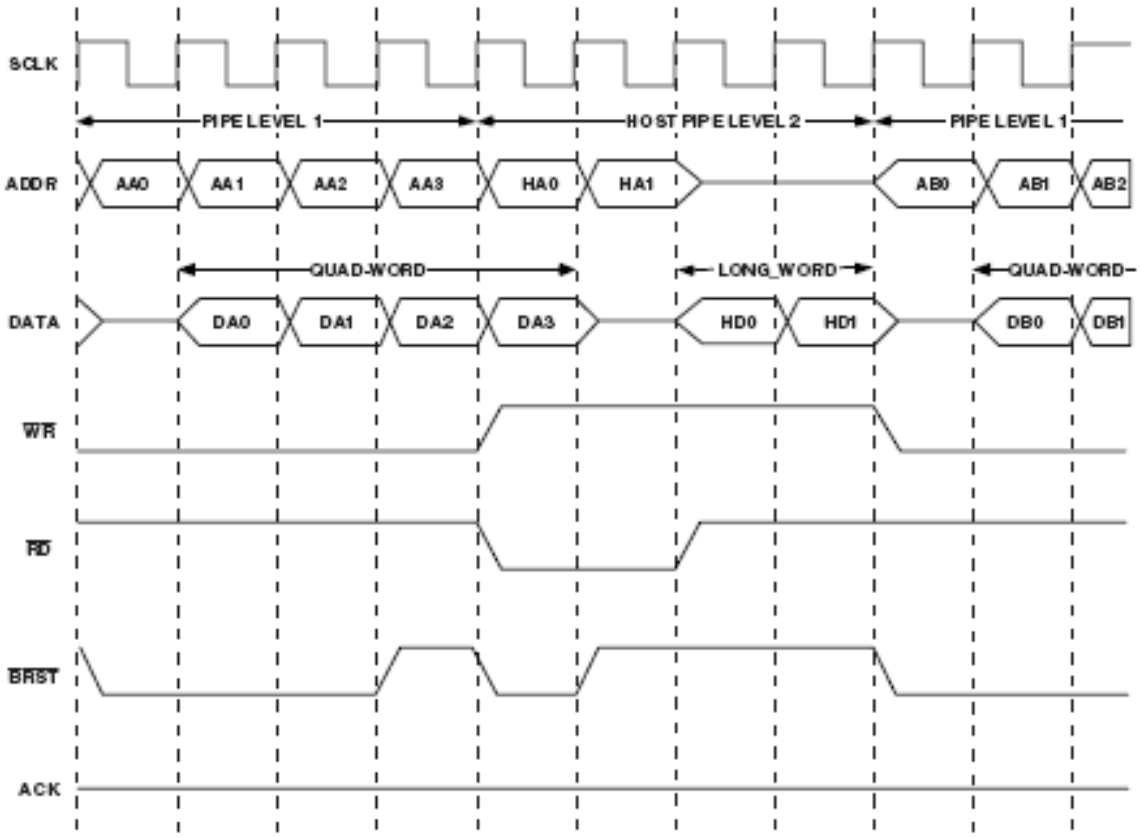


Рисунок 33 – Операция записи в хост с последующим циклом чтения из хоста процессором. Процессор, пропускная способность шины = 32

Когда хост выполняет пакетные транзакции с процессором, то процессор позволяет постоянную передачу пакетов из более, чем четырех слов данных. Хост формирует начальный адрес пакетной передачи. Пока активирован nBRST, процессор увеличивает адрес внутри. Вывод nBRST используется для индикации продолжительности транзакции, в этом случае транзакция завершается на каждом адресе счетверенного слова. Первая транзакция должна начинаться, и последняя транзакция должна заканчиваться в адресах, выровненных по счетверенному слову.

7.4.9.1 Сигналы отмены транзакции nBOFF

Хост-процессор, являясь одним из управляющих устройств в системе, может запрашивать шину, как это описано в разделе арбитража шины. В некоторых случаях может произойти ситуация взаимоблокировки. Это происходит, когда хост и процессор пытаются одновременно произвести чтение с шины друг у друга (см. Рисунок 34). В этом случае процессор управляет своей системной шиной, хост управляет своей системной шиной, а оба осуществляют транзакцию чтения друг из друга. В результате мост хоста делает запрос на обе шины для выполнения обеих транзакций.

Ни одна из шин не освобождается до того, пока текущее управляющее устройство не завершит транзакцию. Аналогично, ни одна из транзакций не завершается, потому что шины не освобождаются. Для устранения взаимоблокировки, одно из запрашивающих устройств должно оставить шину, позволяя другому запрашивающему устройству завершить транзакцию чтения. В этой ситуации процессор может предоставить шину хосту, когда последний активирует nBOFF, сигнализируя процессору о том, что необходимо предоставить шину.

В ответ процессор активирует nHBG, позволяя хосту выполнять транзакции. После того как хост завершит транзакции и освободит шину, процессор осуществляет транзакции чтения, которые были приостановлены, когда был активирован nBOFF.

Механизм nBOFF может не использоваться для остановки транзакции записи в хост (запись данных может выполняться в мост). Механизм nBOFF может быть активным, только когда процессор достигает фазы данных транзакций, ориентированных на хост. Это происходит после того как хост деактивирует вывод ACK.

Если хост запрограммирован таким образом, что у процессора есть к нему доступ через медленный протокол с нулевым циклом ожидания, то механизм nBOFF не может использоваться, поскольку в этом случае сигнал ACK не может быть активирован для передачи.

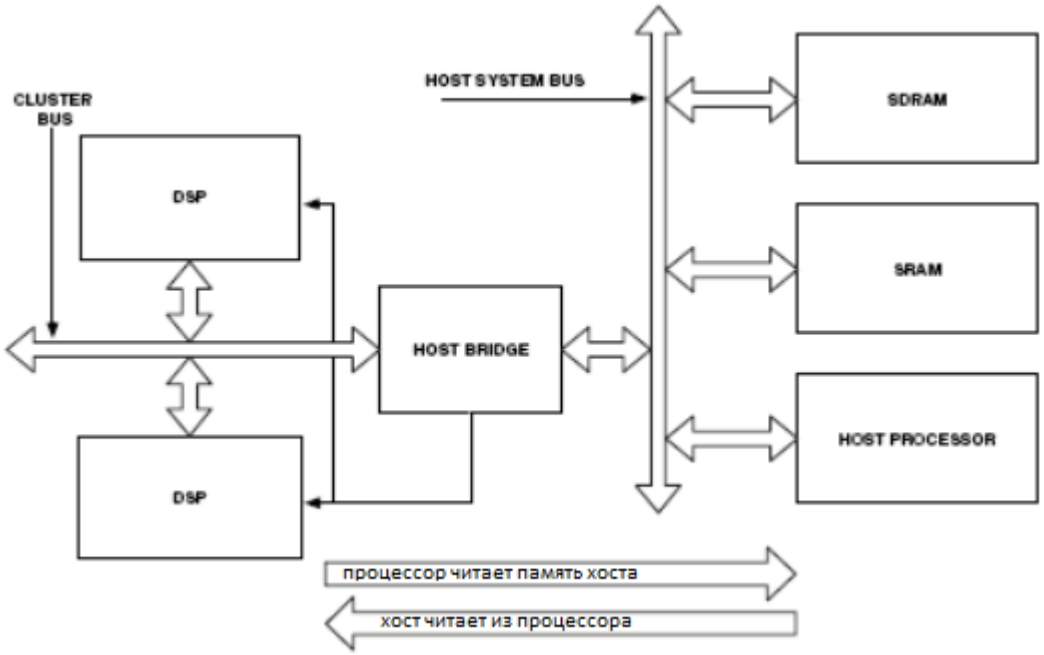


Рисунок 34 – Сценарий взаимоблокировки

7.4.10 Интерфейс SDRAM

Процессор имеет выделенное адресное пространство для подключения синхронной динамической памяти SDRAM. Четыре банка в этом адресном пространстве выбираются линиями выборки nMSSD3–0. SDRAM доступно в адресном диапазоне с 0x4000 0000 до 0x7FFF FFFF.

Процессор поддерживает стандартную (3,3 В) и пониженной мощности (2,5 В) SDRAM. Память работает с использованием частоты внешней шины SCLK (системная частота). Все входы защелкиваются и все выходы корректны при положительном (из 0 в 1) перепаде частоты. Синхронный интерфейс позволяет выполнять передачу данных каждый такт. SDRAM поддерживает несколько типов пакетных доступов в зависимости от инициализации регистра режима SDRCON. Каждый тип доступа предваряется запуском соответствующей команды в SDRAM. Специальные режимы SDRAM должны инициализироваться в регистре SDRCON.

Инициализация SDRAM выполняется процессором с ID 0. Процессор с таким ID всегда должен присутствовать в любом кластере процессоров.

SDRAM имеет внутреннюю организацию в виде двух или четырех банков. Выводы выбора банка SDRAM определяют к какому банку происходит обращение.

SDRAM имеет программируемый параметр задержки чтения, который должен инициализироваться приложением в соответствии с типом устройства и рабочей тактовой частотой.

Для соответствия требуемым временным характеристикам SDRAM, процессор может выполнять конвейерную передачу адреса и управляющих команд SDRAM. Для этого используется бит глубины конвейера в регистре SDRCON.

Микросхемы SDRAM поставляются различными производителями. Каждый поставщик имеет свои временные требования касательно параметров задержки команд ACT – PRE (tRAS) и PRE – ACT (tRP). Для поддержки всех основных поставщиков и различных уровней скорости, регистр SDRCON программируем для того, чтобы разработчик мог удовлетворить временные требования микросхемы SDRAM.

Такими параметрами являются битовые поля: nCAS задержка, PRE – nRAS задержка, nRAS – PRE задержка.

Наилучшая скорость обмена с SDRAM получается при использовании блочного обмена, когда процессор выполняет несколько последовательных чтений или записей в один банк памяти. Интерфейс процессора имеет возможность отслеживания и поддержки пакетного доступа.

Особенностью работы динамической памяти является необходимость регенерации всех её ячеек памяти в течение заданного интервала времени. Это гарантирует сохранность информации. Процессор имеет возможность программировать частоту регенерации для соответствия временным требованиям микросхемы.

При описании интерфейса SDRAM используются следующие определения:

- Команда активации банка ACT

  Активирует выбранный банк и фиксирует в нем новую строку. Должна использоваться перед командой чтения или записи.

- Длина пакета (Burst Length)

  Определяет количество слов, которые поступают на вход или выход SDRAM после детектирования команды чтения или записи. Микросхема памяти всегда программируется для длины пакета равного полной странице.

- Тип пакетирования (Burst Type)

  Определяет порядок, в котором SDRAM отправляет или принимает пакетные данные после детектирования команды чтения или записи. Процессор поддерживает только последовательный доступ.

- Задержка nCAS (nCAS Latency)

  Задержка, в тактах, между тем, когда SDRAM определяет команду чтения и когда данные поступают на выходные выводы. Величина запаздывания определяется уровнем скорости устройства и тактовой частотой шины. Приложение должно программировать этот параметр в регистре SDRCON.

- Маскирование ввода-вывода HDQM/LDQM данных

  Выводы HDQM/LDQM используются для маскирования контроллером операций записи.

- Регистр SDRCON

Регистр, который содержит программируемые конфигурационные параметры для возможности работы контроллера SDRAM с микросхемой SDRAM.

- Регистр режима (Mode register)
  Регистр конфигурации SDRAM, который содержит определяемые пользователем параметры, согласованные с регистром SDRCON. Этот регистр находится внутри микросхемы памяти.

- Размер страницы (Page Size)
  Процессор поддерживает 1024-, 512-, и 256-словные размеры страниц. Размер страницы может быть запрограммирован в регистре SDRCON.

- Команда подзарядки (Precharge)
  Подзаряжает активный банк.

- Период регенерации (Refresh Rate)
  Программируемая величина в регистре SDRCON. Период регенерации позволяет приложениям координировать скорость SCLK с требуемой частотой регенерации SDRAM.

- Саморегенерация (Self-Refresh)
  Состояние микросхемы памяти в котором она использует внутренний таймер и периодически инициирует автоматические регенерации памяти без внешних команд. Этот режим работы обеспечивает SDRAM режим низкого потребления.

- tRAS
  Требуемая задержка между запуском команды активации строки банка и запуском команды подзарядки. Величина зависит от производителя и задается в регистре SDRCON.

- tRC
  Требуемая задержка между последовательными командами активации одного банка. Данный параметр зависит от производителя и определяется как tRC = tRP + tRAS. Процессор фиксирует значение этого параметра, так что это непрограммируемая опция.

- tRCD. nRAS – nCAS задержка
  Требуемая задержка между командой ACT и началом первой операции записи или чтения. Данный параметр зависит от производителя и определяется как tRCD = CL. Процессор использует фиксированное значение этого параметра, так что это непрограммируемая опция.

- tRP
  Требуемая задержка между запуском команды подзарядки и командой активации. Данный параметр зависит от производителя и определяется в SDRCON.

На рисунке 35 показан интерфейс контроллера SDRAM между внутренним ядром процессора и внешним устройством SDRAM.

Процессор обычно генерирует адрес внешней памяти, который затем активизирует соответствующую линию выбора SDRAM памяти (nMSSD3–0), а также

указывает на тип операции обмена: чтение или запись. Эта информация анализируется контроллером SDRAM. Внутренняя 32-битная шина адреса мультиплексируется контроллером SDRAM для создания соответствующего сигнала выбора памяти, адреса строки, адреса колонки и банка в SDRAM.

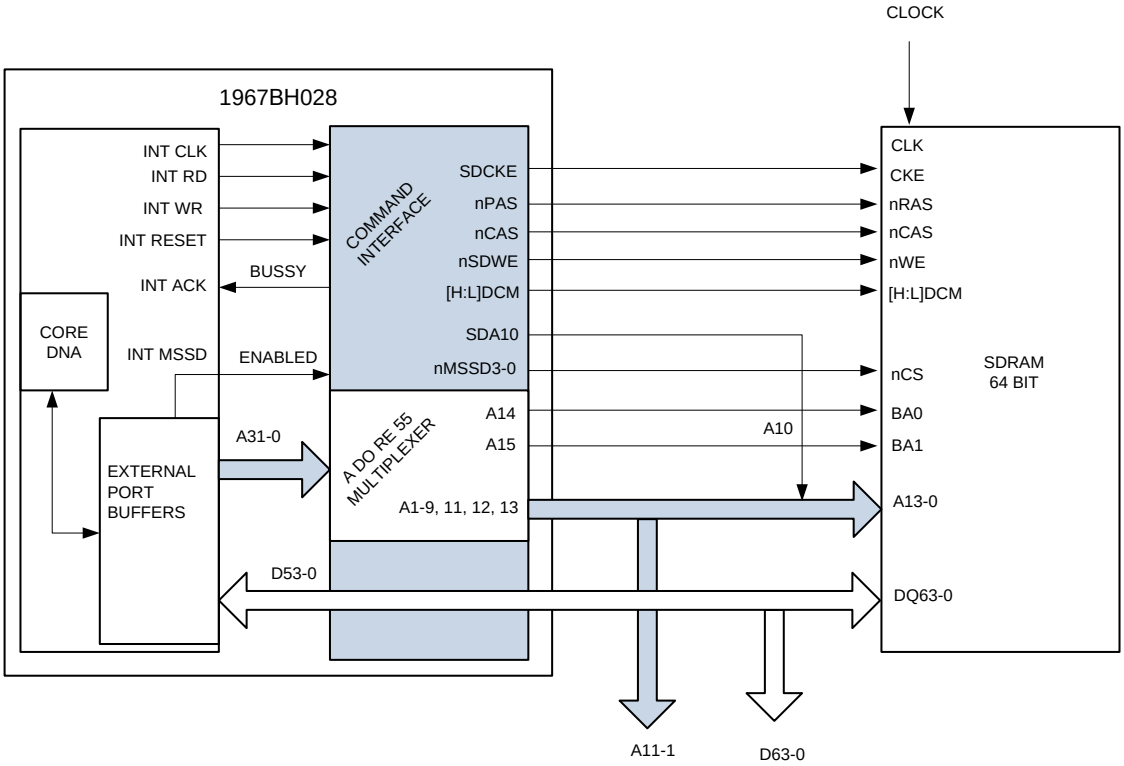


Рисунок 35 – Интерфейс контроллера SDRAM (для конфигурации с 64-разрядной шиной)

7.4.10.1 Вводы/выводы интерфейса SDRAM

Интерфейс SDRAM порта внешней шины использует для обмена с SDRAM памятью общую шину данных и часть шины адреса. Для управления микросхемой памяти используются следующие дополнительные линии:

– nMSSD3–0

Выборка памяти SDRAM. nMSSD0, nMSSD1, nMSSD2 или nMSSD3 активны (один из четырех), всякий раз, когда процессор выполняет доступ к пространству памяти SDRAM. nMSSD3–0 являются выводами декодированного адреса памяти и активны в то время, когда процессор выполняет командный цикл SDRAM. В мультипроцессорной системе мастер управляет nMSSD3–0. Некоторые команды активируют сразу все четыре линии.

– nRAS

Выборка адреса строки. Во время чтения или записи SDRAM низкий уровне сигнала nRAS указывает на то, что на шине адреса находится достоверный адрес строки. В остальных случаях доступа к SDRAM он определяет тип операции для выполнения в соответствии со спецификацией SDRAM.

– nCAS

Выборка адреса столбца. Во время чтения или записи SDRAM низкий уровне сигнала CAS указывает на то, что на шине адреса находится

достоверный адрес столбца. В остальных случаях доступа к SDRAM он определяет тип операции для выполнения в соответствии со спецификацией SDRAM.

- LDQM
  Маска младшего слова данных SDRAM. При высоком уровне сигнала переводит буферы данных DQ SDRAM в третье состояние. При чтении всегда равен нулю. При операциях записи LDQM является активным (равным 1), когда выполняется доступ к нечетному слову на 64-разрядной шине памяти для запрещения записи младшего слова.

- HDQM
  Маска старшего слова данных SDRAM. При высоком уровне сигнала переводит буферы данных DQ SDRAM в третье состояние. Используется только при 64-разрядной шине данных. Для 32-разрядной шины всегда равен 1. При чтении всегда равен нулю. При операциях записи HDQM является активным (равным 1), когда выполняется доступ к четному адресному слову на 64-разрядной шине памяти для запрещения записи старшего слова.

- SDA10
  Вывод разряда 10 адреса SDRAM. SDRAM использует отдельный вывод адресной линии номер 10 из-за того, что она используется в специальных операциях и может задействоваться в то время, когда процессор выполняет на внешней шине транзакцию, не связанную с SDRAM.

- SDCKE
  Разрешение тактового сигнала SDRAM. Вход синхронизации микросхемы динамической памяти может быть постоянно активным, но микросхема будет управляться данным синхросигналом, только если на линии SDCKE высокий уровень. Низкий уровень сигнала SDCKE используется для перевода памяти в режим саморегенерации.

- nSDWE
  Разрешение записи SDRAM. При низком уровне сигнала во время активного nCAS, nSDWE указывает на выполнение записи SDRAM. При высоком уровне сигнала во время активного nCAS, nSDWE указывает на выполнение чтения SDRAM. В остальных операциях SDRAM, nSDWE определяет тип операции для выполнения в соответствии со спецификацией SDRAM.

Выводы данных и адреса SDRAM могут соединяться напрямую с выводами данных и адреса процессора. Ширина шины данных может быть выбрана 32-х или 64-х разрядов при программировании регистра SYSCON.

Способ подключения линий адреса зависит от выбранной ширины шины данных. Для 32-разрядной шины соединение представлено ниже:

- разряды 9-0 адреса SDRAM соединяются с выводами ADDR9–0 процессора;
- разряд 10 адреса SDRAM соединяется с выводом SDA10 процессора;
- разряды 15-11 адреса SDRAM соединяются с выводами ADDR15–11 (столько разрядов сколько требуется) процессора.

При 64-разрядной шине данных адрес должен быть сдвинут. Схема должна быть следующей:

- разряды 9-0 адреса SDRAM соединяются с выводами ADDR10–1 процессора;
- разряд 10 адреса SDRAM соединяется с выводом SDA10 процессора;
- разряды 14-11 адреса SDRAM соединяются с выводами ADDR15–12 (столько разрядов сколько требуется) процессора.

Управляющие сигналы являются специальными сигналами контроля SDRAM. Эти сигналы перечислены ниже:

- nMSSD3–0 – выбор чипа SDRAM;
- nRAS – строб адреса строки;
- nCAS – строб адреса столбца;
- nSDWE –запись SDRAM;
- LDQM – маска данных для младшего слова (разряды данных 31-0);
- HDQM – маска данных для старшего слова (разряды данных 63-32);
- SDCKE – разрешение тактового сигнала.

В больших системах памяти обычно присутствует много чипов SDRAM, которые могут вызвать перегрузку шины адреса. Так система памяти может быть основана на микросхемах SDRAM с шириной шины данных 8 бит. В этом случае при 64-разрядной шине данных мы будем иметь 8 нагрузок на линиях адреса и одну нагрузку на линии данных. Процессор предлагает два варианта снижения нагрузки на адресную шину.

Разряды 15-0 адреса SDRAM дублируются на выводах ADDR31–16 процессора. Это позволяет часть микросхем подключать к младшей половине адресной шины, а остальные к старшей половине адресной шины. Для 32-разрядной шины следующая схема соединений данных:

- разряды 9-0 адреса SDRAM соединяются с выводами ADDR25–16 процессора.
- разряд 10 адреса SDRAM соединяется с выводом SDA10 процессора.
- разряды 15-11 адреса SDRAM соединяются с выводами ADDR31–27 (столько разрядов сколько требуется) процессора.

Схема для 64-разрядной шины следующая:

- разряды 9-0 адреса SDRAM соединяются с выводами ADDR26–17 процессора
- разряд 10 адреса SDRAM соединяется с выводом SDA10 процессора
- разряды 14-11 адреса SDRAM соединяются с выводами ADDR31–28 (столько разрядов сколько требуется) процессора

Другой способ сократить нагрузку – буферизировать сигналы адреса и контроля. Нагрузочная способность буфера должна быть достаточной для обеспечения требуемой частоты. Буферизация сигналов вызывает задержку в один такт в цикле обмена. В этом случае должен быть установлен разряд глубины конвейера в SDRCON.

7.4.10.2 Выводы выбора банка

Внутренняя организация микросхемы динамической памяти включает в себя два или четыре банка. Для выбора банка микросхема имеет специальные выводы BS1-0. Эти выводы подключаются к адресным линиям процессора. Соединение выводов адреса процессора в качестве линий выбора банка для устройства SDRAM меняется в зависимости от следующих факторов:

- рабочее напряжение, используемое для SDRAM (стандартно 3,3 В или 2,5 В SDRAM с пониженной мощностью);
- числа банков.

Для двух банков, BS равен ADDR11 или любой адресной линии в диапазоне ADDR15–11. Для четырех банков, BS1–0 равен ADDR12–11 или BS1–0 может быть равен другой паре адресов в диапазоне ADDR15–11.

Для SDRAM с пониженной мощностью использование линий адресов в качестве сигналов выбора банка ограничено ADDR15-14. Для двух банков может быть использована любая из двух адресных линий ADDR14 или ADDR15. Для четырех банков должны использоваться ADDR15–14.

7.4.10.3 Физическое соединение внутреннего адреса процессора и адреса микросхемы SDRAM

Выше отмечалось, что SDRAM может подключаться к шине данных шириной 64 бита либо 32 бита. Выбор подключения зависит от того какой объём памяти пользователь хочет установить в системе и какую скорость обмена хочет получить.

Рисунок 36, Таблица 78, Таблица 79, Таблица 80 описывают выравнивание данных внешнего порта и соединения SDRAM для 64-разрядной системной шины.

Рисунок 37, Таблица 81, Таблица 82, Таблица 83 описывают выравнивание данных внешнего порта и соединения SDRAM для 32-разрядной системной шины.

В таблицах приведено соответствие между внешними выводами адреса процессора и выводами адреса микросхемы SDRAM, а также указаны разряды внутреннего 32-битного адреса, которые выдаются на выводы в разных фазах доступа (т.е. в фазе активации банка и выбора активного ряда (строки)) и в фазе чтения или записи при выборе колонки (номера слова в строке).

При 64-разряной шине чтение (nRD) всегда возвращает 64 бита данных. При записи может писаться сразу длинное слово (64 бита) либо одно 32-разрядное слово (по четному или нечетному адресу). При записи каждому слову соответствует свой сигнал записи (HDQM и LDQM для SDRAM).

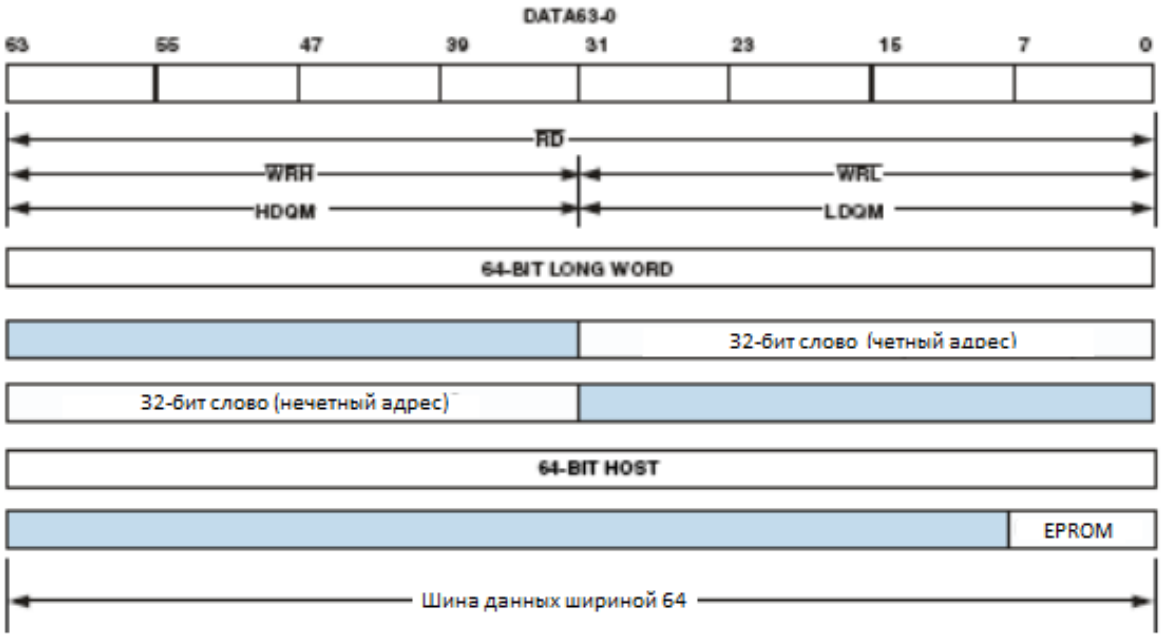


Рисунок 36 – Шина данных внешнего порта 64 разряда

Таблица 78 – Размер страницы 256 слов, 64-разрядная шина

Физический вывод процессора	Активация банка Адрес строки Бит внутреннего адреса	Выбор колонки в строке банка. Бит внутреннего адреса	Физический вывод SDRAM
A0	8	0	NC
A1	9	1	A0
A2	10	2	A1
A3	11	3	A2
A4	12	4	A3
A5	13	5	A4
A6	14	6	A5
A7	15	7	A6
A8	16	8	A7
A9	17	9	A8
A10	18	10	A9
SDA10	19	нулевой	A10/AP
A11	несущественный	несущественный	NC
A12	20	20	A11 или банк
A13	21	21	A12 или банк
A14	22	22	A13 или банк
A15	23	23	A14 или банк

Таблица 79 – Размер страницы 512 слов, 64-разрядная шина

Физический вывод процессора Процессор	Внутренний адрес активного цикла банка	Внутренний адрес периода выборки столбца	Физический вывод SDRAM
A0	9	0	NC
A1	10	1	A0
A2	11	2	A1
A3	12	3	A2
A4	13	4	A3
A5	14	5	A4
A6	15	6	A5
A7	16	7	A6
A8	17	8	A7
A9	18	9	A8
A10	19	10	A9
SDA10	20	нулевой	A10/AP
A11	несущественный	несущественный	NC
A12	21	21	A11 или банк
A13	22	22	A12 или банк
A14	23	23	A13 или банк
A15	24	24	A14 или банк

Таблица 80 – Размер страницы 1K слов, 64-разрядная шина

Физический вывод процессора Процессор	Внутренний адрес активного цикла банка	Внутренний адрес периода выборки столбца	Физический вывод SDRAM
A0	10	0	NC
A1	11	1	A0
A2	12	2	A1
A3	13	3	A2
A4	14	4	A3
A5	15	5	A4
A6	16	6	A5
A7	17	7	A6
A8	18	8	A7
A9	19	9	A8
A10	20	10	A9
SDA10	21	нулевой	A10/AP
A11	несущественный	несущественный	NC
A12	22	22	A11 или банк
A13	23	23	A12 или банк
A14	24	24	A13 или банк
A15	25	25	A14 или банк

При 32-разряной шине чтение (nRD) всегда возвращает 32 бита данных. При записи также пишется только одно 32-разрядное слово по любому адресу. При записи используется только один сигнал записи (LDQM для SDRAM).

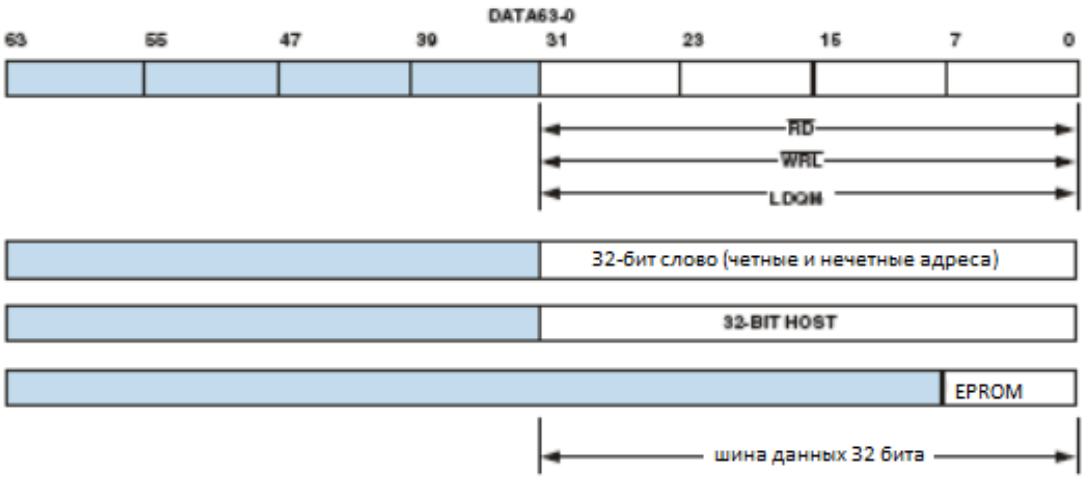


Рисунок 37 – Шина данных внешнего порта 32 разряда

Таблица 81 – Размер страницы 256 слов, 32-разрядная шина

Физический вывод процессора Процессор	Внутренний адрес активного цикла банка	Внутренний адрес периода выборки столбца	Физический вывод SDRAM
A0	8	0	A0
A1	9	1	A1
A2	10	2	A2
A3	11	3	A3
A4	12	4	A4
A5	13	5	A5
A6	14	6	A6
A7	15	7	A7
A8	16	8	A8
A9	17	9	A9
A10	несущественный	несущественный	NC
SDA10	18	0	A10/AP
A11	19	19	A11 или банк
A12	20	20	A12 или банк
A13	21	21	A13 или банк
A14	22	22	A14 или банк
A15	23	23	A15 или банк

Таблица 82 – Размер страницы 512 слов, 32- разрядная шина

Физический вывод процессора Процессор	Внутренний адрес активного цикла банка	Внутренний адрес периода выборки столбца	Физический вывод SDRAM
A0	9	0	A0
A1	10	1	A1
A2	11	2	A2
A3	12	3	A3
A4	13	4	A4
A5	14	5	A5
A6	15	6	A6
A7	16	7	A7
A8	17	8	A8
A9	18	9	A9
A10	несущественный	несущественный	NC
SDA10	19	нулевой	A10/AP
A11	20	20	A11 или банк
A12	21	21	A12 или банк
A13	22	22	A13 или банк
A14	23	23	A14 или банк
A15	24	24	A15 или банк

Таблица 83 – Размер страницы 1K слов, 32- разрядная шина

Физический вывод процессора Процессор	Внутренний адрес активного цикла банка	Внутренний адрес периода выборки столбца	Физический вывод SDRAM
A0	10	0	A0
A1	11	1	A1
A2	12	2	A2
A3	13	3	A3
A4	14	4	A4
A5	15	5	A5
A6	16	6	A6
A7	17	7	A7
A8	18	8	A8
A9	19	9	A9
A10	несущественный	несущественный	NC
SDA10	20	нулевой	A10/AP
A11	21	21	A11 или банк
A12	22	22	A12 или банк
A13	23	23	A13 или банк
A14	24	24	A14 или банк
A15	25	25	A15 или банк

7.4.10.4 Программирование параметров SDRAM

Микросхемы SDRAM могут иметь различные требования к последовательности включения и временным параметрам, так как могут быть получены от разных поставщиков. С целью возможности работы с большим количеством поставщиков, в процессоре предусмотрен регистр SDRCON, в котором можно запрограммировать некоторые параметры SDRAM. В таблице 84 приведено описание разрядов регистра.

Таблица 84 – Регистр SDRCON

Бит	Имя	Назначение
0	SDREN	Включение контроллера SDRAM 1 –включен 0 – выключен
2:1	nCAS	Задержка nCAS 00 – 1 цикл 01 – 2 цикла 10 – 3 цикла 11 – резерв
3	PIPE	Дополнительный конвейер: 1 – используется 0 – нет
5:4	PAGE	Размер страницы: 00 – 256 слов 01 – 512 слов 10 – 1024 слов 11 – резерв
6	-	

8:7	REF	Период регенерации памяти (в тактах SCLK) 00 – 1100 циклов 01 – 1850 циклов 10 – 2200 циклов 11 – 3700 циклов
10:9	PRC2RAS	Задержка от подзаряда до строба nRAS (такты SCLK) 00 – 2 01 – 3 10 – 4 11 – 5
13:11	RAS2PRC	Задержка от строба nRAS до команды подзаряда: 000 – 2 …… 110 – 8
14	INIT	Выбор последовательности инициализации: 1 – команда MRS после регенерации памяти 0 – команда MRS перед регенерацией памяти
15	EMREN	Использование дополнительного регистра режима: 1 – разрешено 0 – не используется
31:16	-	Всегда 0.

Регистр SDRCON хранит информацию о конфигурации интерфейса SDRAM. Ниже подробно описывается назначение каждого из параметров регистра.

Включение контроллера SDRAM

Разряд SDREN должен быть установлен, если в системе есть SDRAM. В противном случае, он должен быть сброшен. Любой доступ к SDRAM, пока этот разряд сброшен, вызывает аппаратное прерывание ошибки доступа. Установка данного бита включает в работу контроллер SDRAM внешнего интерфейса. При включении контроллер сразу же начинает процедуру инициализации внешней динамической памяти. Данную процедуру может выполнять только процессор с номером ID равным нулю. Поэтому при включении контроллера SDRAM, процессор должен быть мастером на шине.

Выбор значения задержки nCAS Latency

Значение задержки nCAS определяет задержку в тактовых циклах системы (SCLK), между временем, когда SDRAM обнаруживает команду чтения и временем, когда он выдает данные на его внешние выводы. Этот параметр позволяет процессору знать, в каком такте после передачи в SDRAM команды чтения он может принять прочитанные данные.

Задержка nCAS не используется в циклах записи.

Опция буферизации SDRAM. Глубина конвейера

Ранее были описаны ситуации, при которых пользователю может понадобиться использовать буферизацию адресных и управляющих сигналов. Связано это с максимальной нагрузочной способностью выводов процессора и требуемой скоростью обмена. В случае использование буферизации в конвейере чтения данных появляется дополнительный цикл. Чтобы данный цикл был учтен контроллером SDRAM и используется бит PIPE. Шина данных не буферизируется. Поэтому при установленном разряде PIPE (1), контроллер SDRAM задерживает данные в цикле записи в течение одного такта.

В случае чтения контроллер SDRAM осуществляет захват данных на один цикл позже.

Ниже приведен пример одного процессора, в котором интерфейс SDRAM соединяется с множеством банков SDRAM (Рисунок 38). Микросхема SDRAM имеет шину данных шириной 4 бита. Для обеспечения 32-разрядной шины данных используется 8 микросхем. Чтобы уменьшить нагрузку на внешние выводы процессора используется буферный регистр, который состоит из двух регистров A и B каждый из которых запоминает одинаковую информацию, но передает ее на свои 4 микросхемы памяти.

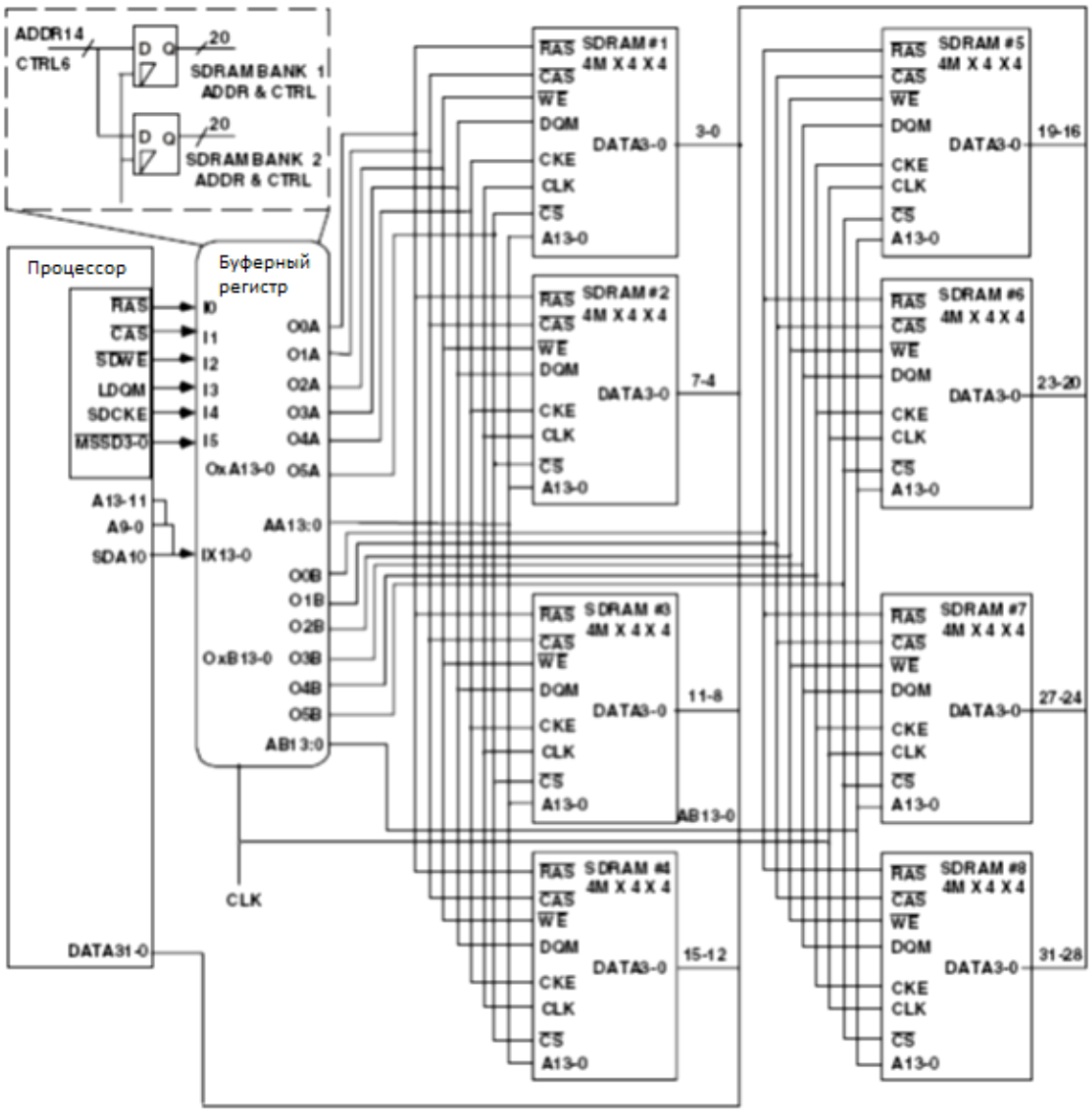


Рисунок 38 – Однопроцессорная система со стандартными устройствами SDRAM (32-разрядная шина)

Выбор размера страницы SDRAM

Размер страницы определяется конкретным типом микросхемы SDRAM. Разряды PAGE определяют размер страницы банков SDRAM, в количестве слов. Знание размера страницы необходимо контроллеру SDRAM для отслеживания ситуаций обращения к одной и той же странице. Это позволяет существенно увеличить скорость обмена. Также размер страницы влияет на распределение разрядов внутреннего адреса между адресами строки, столбца и выбора банка памяти.

Период регенерации памяти

Память SDRAM требует, чтобы в течение определенного интервала времени все её ячейки были обновлены (регенерированы). Это гарантирует сохранность информации. В связи с этим контроллер SDRAM должен периодически посылать в микросхемы памяти команды регенерации. Период подачи команд программируется. Период внутри контроллера задается в тактах клока SOC шины.

Количество тактов в периоде рассчитывается исходя из следующего уравнения.

$$f_{cycles} = \left(sdramclock x \frac{t_{ref}}{rows} \right) = (clock \times refresh\_rate)$$

Период регенерации всей памяти Tref делится на количество рядов (строк) в одном банке памяти. Получаем период для регенерации одного ряда. Далее это значение умножается на частоту динамической памяти, и получается период подачи команды регенерации в тактах динамической памяти. Также можно получить этот же период в тактах SOC шины. В регистре конфигурации можно задать несколько фиксированных значений периода. Мастер шины всегда ответственен за своевременную регенерацию памяти. При этом все подчиненные процессоры отслеживают подачу команды мастером и корректируют свои счетчики.

Задержка от подзаряда до строба nRAS

Протокол обмена с динамической памятью требует, чтобы при доступе к новому ряду банка или перед выполнением регенерации памяти, выполнялся подзаряд (закрытие) текущей активной строки. Для подзаряда используется команда PRE. После выполнения подзаряда невозможно сразу перейти к активизации следующего ряда т.к. спецификация микросхемы требует, чтобы между подзарядом и активизацией нового ряда была обеспечена некоторая минимальная задержка. Значение этой задержки, выраженное числом системных тактовых циклов (SCLK), и программируется в поле PRC2RAS.

Выбор задержки между nRAS и предварительным зарядом

Аналогично предыдущему параметру, существует требование минимального интервала времени между подачей команды nRAS (активизация ряда банка) и подачей команды PRE подзаряда банка. Данный интервал задается числом системных тактовых циклов SCLK. Значение поля RAS2PRC может изменяться от 0 до 6, это соответствует числу циклов SCLK от 2 до 8. Если, например, поле RAS2PRC равно 011 это означает, что минимальный интервал равен 5 циклов. Это не означает, что после команды nRAS минимум через 5 тактов должна быть команда PRE. Это означает, что если процессор выполнил команду nRAS, а затем по каким-то причинам (например, регенерация) ему необходимо выполнить подзаряд, то процессор должен проверить прошло ли с момента команды nRAS как минимум 5 тактов. Тогда он может подавать команду подзаряда.

Выбор последовательности инициализации SDRAM

Для того чтобы микросхема SDRAM корректно работала, она должна быть предварительно проинициализирована. Процессор обеспечивает две наиболее часто используемые последовательности инициализации.

Бит последовательности инициализации INIT в регистре SDRCON выбирает режим включения SDRAM. Когда бит установлен (1), контроллер SDRAM последовательно генерирует: команду PRE, восемь циклов регенерации и команду MRS (установка регистра режима). Когда бит INIT сброшен (0), контроллер SDRAM

работает в следующем порядке: команда PRE, команда MRS и восемь циклов регенерации.

Разрешение регистра расширенного режима (EMR)

Бит EMREN в регистре SDRCON разрешает программирование регистра расширенного режима. Когда бит установлен (1), контролер SDRAM формирует цикл EMRS, предшествующий команде MRS. Когда бит сброшен (0), последовательность инициализации происходит так, как описано в главе выше.

Бит разрешающий EMR должен быть установлен только во время обмена данными с устройствами SDRAM пониженной мощности (2,5 В). Именно в таких микросхемах используется дополнительный регистр режима.

7.4.10.5 Flyby передачи

Транзакции Flyby используются контроллером DMA для передачи данных между внешним устройством ввода/вывода и внешней памятью SDRAM. Передача может происходить от памяти SDRAM к устройству ввода/вывода (чтение) или от устройства ввода/вывода к памяти SDRAM (запись). Транзакция flyby вызывается каналом DMA, который запрограммирован на выполнение таких транзакций с использованием поля TY регистра DP.

В транзакциях flyby процессор выбирает параллельно внешнюю память SDRAM и устройство ввода/вывода. Память SDRAM управляется с помощью соответствующих сигналов контроллера SDRAM (nMSSDx, nRAS, nCAS и nSDWE), а устройство ввода/вывода управляется выводами nIORD, nIOWR и nIOEN. Передача данных осуществляется напрямую по шине данных без участия процессора.

Во время транзакции чтения flyby (чтение внешней памяти SDRAM и запись устройства ввода/вывода), сигнал nIOWR является активным во время цикла данных. Память выдает данные после фронта SCLK и устройство ввода/вывода захватывает данные на фронте SCLK, когда сигнал nIOWR активный.

Рисунок 39 показывает пример транзакции чтения flyby. В транзакциях данные с адресами CA0 и CA2 считываются текущим мастером шины, а устройство ввода/вывода захватывает данные с адресом CA1.

С целью поддержки транзакций чтения SDRAM в режиме flyby, устройства ввода/вывода должны быть синхронными.

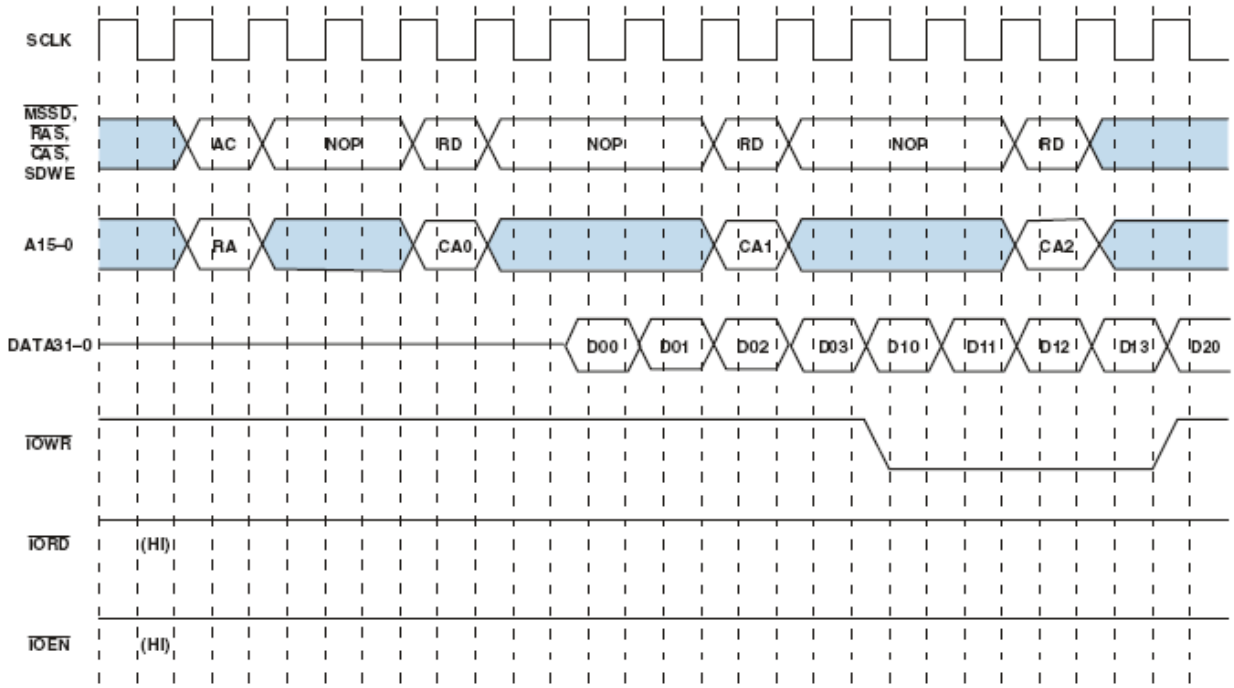


Рисунок 39 – Транзакция чтения FLYBY

Во время транзакции записи данные передаются от устройства ввода-вывода к памяти. Для этого на устройство подается сигнал выбора nIOEN и сигнал чтения nIORD. Эти сигналы подаются на такт раньше команды записи данных для SDRAM. По положительному фронту SCLK сигнал чтения защелкивается в устройстве и в следующем такте устройство выставляет на шину данные. Эти данные вместе с командой записи поступают в SDRAM.

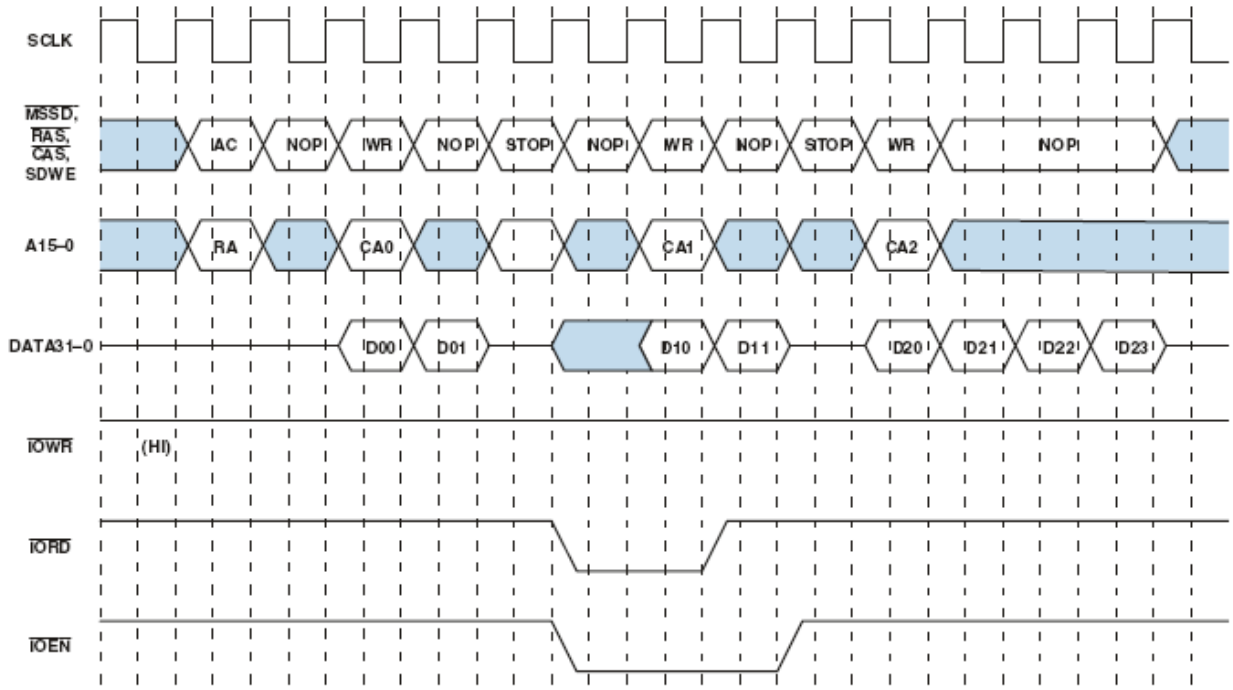


Рисунок 40 – Транзакция записи FLYBY

7.4.10.6 Мультипроцессорный режим работы

В мультипроцессорной среде, SDRAM распределяется между двумя и более процессорами. Входные сигналы SDRAM всегда контролируются мастером шины. Подчиненные процессоры отслеживают команды, которые поступают от управляющего процессора к SDRAM. Эта особенность позволяет контроллерам SDRAM всех процессоров обновлять свои счетчики и предотвращать ненужные операции регенерации.

В мультипроцессорных системах, где управление шиной передается от процессора к процессору, текущий мастер шины всегда формирует команду предварительного заряда (PRE), перед тем как шина освобождается для нового мастера.

Таким образом, новый мастер шины может безопасно начинать доступ к SDRAM с помощью обычной команды активации (ACT).

В мультипроцессорных системах, где используется SDRAM, последовательность инициализации SDRAM определяется процессором с ID равным 00.

7.4.10.7 Включение после сброса

После сброса содержимое регистра SDRCON равно нулю и контроллер SDRAM выключен. Как только приложение пользователя выполняет запись в регистр

SDRCON данных с установленным битом включения контроллера SDRAM, контроллер тут же инициирует выбранную последовательность включения динамической памяти. Последовательность инициализации определяется разрядом INIT и разрядом регистра расширенного режима в регистре SDRCON. В мультипроцессорной среде, последовательность включения инициируется процессором с ID равным 00. Программный сброс не вызывает сброса контроллера SDRAM и не инициирует повторно последовательность включения.

7.4.11 Команды контроллера SDRAM

Контроллер SDRAM использует следующие команды для управления интерфейсом SDRAM:

- MRS
  (установка регистра режима). Инициализирует рабочие параметры SDRAM во время последовательности включения.
- PRE
  (подзаряд). Осуществляет подзаряд банка.
- ACT
  (активация банка). Активирует страницу в запрашиваемом банке.
- Read
  Чтение из SDRAM.
- Write
  Запись в SDRAM.
- REF
  (регенерация). Переводит SDRAM в режим регенерации.
- SREF
  (саморегенерация). Помещает SDRAM в режим саморегенерации, в котором память управляет своими операциями регенерации изнутри.
- NOP
  (нет операций). Передается после чтения или записи с целью разрешения пакетных операций или с целью установления режима ожидания для различных доступов SDRAM. Во время этой команды nMSSD3–0 деактивирован.

7.4.11.1 Команда установки регистра режима (MRS)

Установка регистра режима является частью последовательности инициализации SDRAM. Каждая микросхема SDRAM имеет внутри регистр режима (количество регистров режима может достигать четырех) и команда MRS позволяет передать данные, используя разряды адреса ADDR13–0, в микросхему для записи в регистр. Таким образом, MRS инициализирует рабочие параметры SDRAM. Команда MRS может подаваться только процессором с ID равным 0 и должна быть использована для первого доступа к SDRAM после включения.

Регистр режима (регистр номер 0) имеет разрядность 13 бит и команда MRS инициализирует следующие параметры:

Биты 2:0 – Burst length. Значение 0b111, что соответствует полной странице.

Бит 3 – Burst type. Значение 0, что соответствует последовательному изменению адреса.

Биты 6:4 – nCAS Latency. Значение параметра определяется в соответствии с программированием регистра SDRCON[2:1].

Все оставшиеся биты регистра режима программируются нулевым значением.

При выполнении команды MRS, SDRAM должна находиться в состоянии подзаряда во всех своих банках.

Передача команды осуществляется соответствующей комбинацией значений на линиях управления. Эти значения приведены ниже (Таблица 85). Происходит запись сразу во все микросхемы памяти.

Таблица 85 – Состояние выводов во время выполнения команды MRS

Разряд	Состояние
nMSSD3–0	0 (все)
nCAS	0
nRAS	0
nSDWE	0
SDCKE	1
A14 == A30	0
A15 == A31	0 – MRS 1 – EMRS

Если в микросхеме памяти имеется дополнительный регистр режима и для инициализации требуется его запись, то в регистре SDRCON должны быть сделаны соответствующие установки (бит EMREN должен быть равен 1). Во время последовательности инициализации контроллер SDRAM запишет нулевое значение в дополнительный регистр режима. Выполняется это с помощью той же команды MRS, но бит 15 адреса (и бит 31) будет установлен в 1, что соответствует дополнительному регистру режима. С помощью адресных линий 14 и 15 можно задать номер дополнительного регистра режима как 01 или 10.

7.4.11.2 Команда подзаряда (PRE)

Особенность работы микросхемы памяти SDRAM состоит в том, что в некоторых ситуациях необходимо выполнять подзаряд (предварительный заряд) внутренних шин банков памяти. Команда PRE (Таблица 86) исполняется в двух ситуациях:

- Прерывание цикла чтения или записи. Точный момент, когда данная команда генерируется контроллером SDRAM, зависит от последовательности выполняемых транзакций и адресов, по которым выполняется доступ.
- Предварительная зарядка активного банка. При выполнении команды на линии A10 всегда высокий уровень, что соответствует подзаряду всех банков микросхемы. Однако подзаряжается только активный банк. Алгоритм работы контроллера SDRAM такой, что он поддерживает активным только один банк.

Передача команды PRE возможна в следующих случаях:

- во время последовательной инициализации SDRAM;
- перед командой ACT (доступ к новой странице). Доступ к новой странице требует, чтобы активная страница была закрыта. Это делается путем выполнения команды подзаряда.
- перед циклом регенерации. Выполнение команды регенерации требует, чтобы все банки памяти были подзаряжены.
- перед тем, как процессор освобождает внешнюю шину. Это гарантирует однозначные условия начала работы нового мастера шины.

Передача команды PRE в микросхему памяти требует, чтобы в течение определенного времени к микросхеме не посылались команды активации. Данное время программируется в регистре SDRCON в соответствии с характеристиками микросхемы памяти.

Таблица 86 – Состояние выводов во время выполнения команды PRE

Вывод	Состояние
nMSSD3–0	0 (все)
nSDWE	0
nRAS	0
nCAS	1
SDCKE	1
SDA10	1

7.4.11.3 Команда выбора активного банка (ACT)

Команда ACT (Таблица 87) посылается контроллером перед любым чтением или записью страницы, которая в данный момент не является активной. Перед командой ACT идет команда PRE, если в каком-то банке имеется другая активная страница. Команда ACT открывает доступ к странице SDRAM в некотором банке, и данная страница остается открытой до тех пор, пока не будет закрыта следующей командой предварительной зарядки PRE.

Таблица 87 – Состояние вывода во время выполнения команды ACT

Вывод	Состояние
nMSSD3–0	0 (один из четырех)
nCAS	1
nRAS	0
nSDWE	1
SDCKE	1

7.4.11.4 Команда чтения (Read)

Команда Read (Таблица 88) выполняет чтение данных из активной страницы памяти SDRAM. Если чтение страницы выполняется первый после активации командой ACT, то между командами ACT и Read должна быть задержка, которая определяется параметром tRCD. Если это уже не первое чтение, то отслеживание данной задержки не выполняется. Команда Read задает начальный адрес слова в странице, с которого начинается чтение. Микросхема памяти будет использовать переданный адрес как стартовый и каждый такт увеличивать его, если необходимо выполнить последовательное чтение блока памяти. После подачи команды Read данные появляются на внешних выводах микросхемы через время равное задержке nCAS Latency.

Одна транзакция чтения занимает различное количество циклов шины в зависимости от размера операнда и ширины внешней шины:

- чтение одинарного слова – 1 цикл;
- чтение двойного слова на 64-разрядной шине – 1 цикл;
- чтение двойного слова на 32-разрядной шине – 2 цикла;
- чтение счетверенного слова на 64-разрядной шине – 2 цикла;
- чтение счетверенного слова на 32-разрядной шине – 4 цикла.

Если после команды Read на SDRAM не поступает какая-то другая команда, то SDRAM продолжает последовательное чтение данных, наращивая внутренний адрес. Этот режим чтения называется «страничный режим» или «пакет». Такой режим удобен, когда транзакция чтения длится более одного цикла.

Когда транзакция полностью завершена, то SDRAM может продолжить работу по нескольким путям:

- Если есть следующая транзакция чтения SDRAM на той же странице, то новая транзакция начинается с подачи команды Read.
- Если нет новых транзакций SDRAM, то контроллер посылает команду BSTOP. Эта команда прерывает последовательное чтение страницы.
- Если после чтения приходит транзакция записи SDRAM на той же странице, то команда BSTOP также останавливает чтение и запись начинается после приема данных.
- Если есть доступ SDRAM к другой странице, то поступает команда BSTOP и после неё посылается команда закрытия активной страницы PRE.

Таблица 88 – Состояние вывода во время выполнения команды Чтение

Вывод	Состояние
nMSSD3–0	0 (один из четырех)
nCAS	0
nRAS	1
nSDWE	1
SDCKE	1

Ниже приведены временные диаграммы использования команды Read для случая ширины шины 64 бита (рисунок 41) и ширины шины 32 бита (рисунок 42). Для обоих случаев задержка nCAS Latency равна 2.

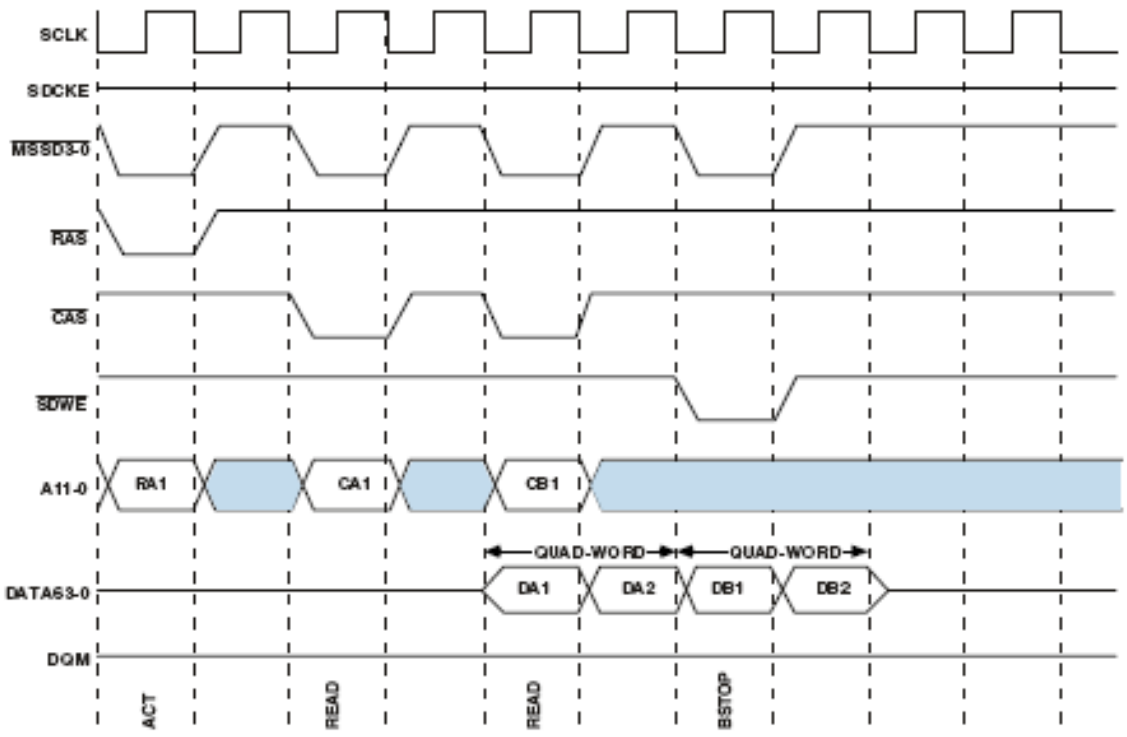


Рисунок 41 – Шина 64 разряда
(пакетное чтение с последующим пакетным чтением на той же странице)

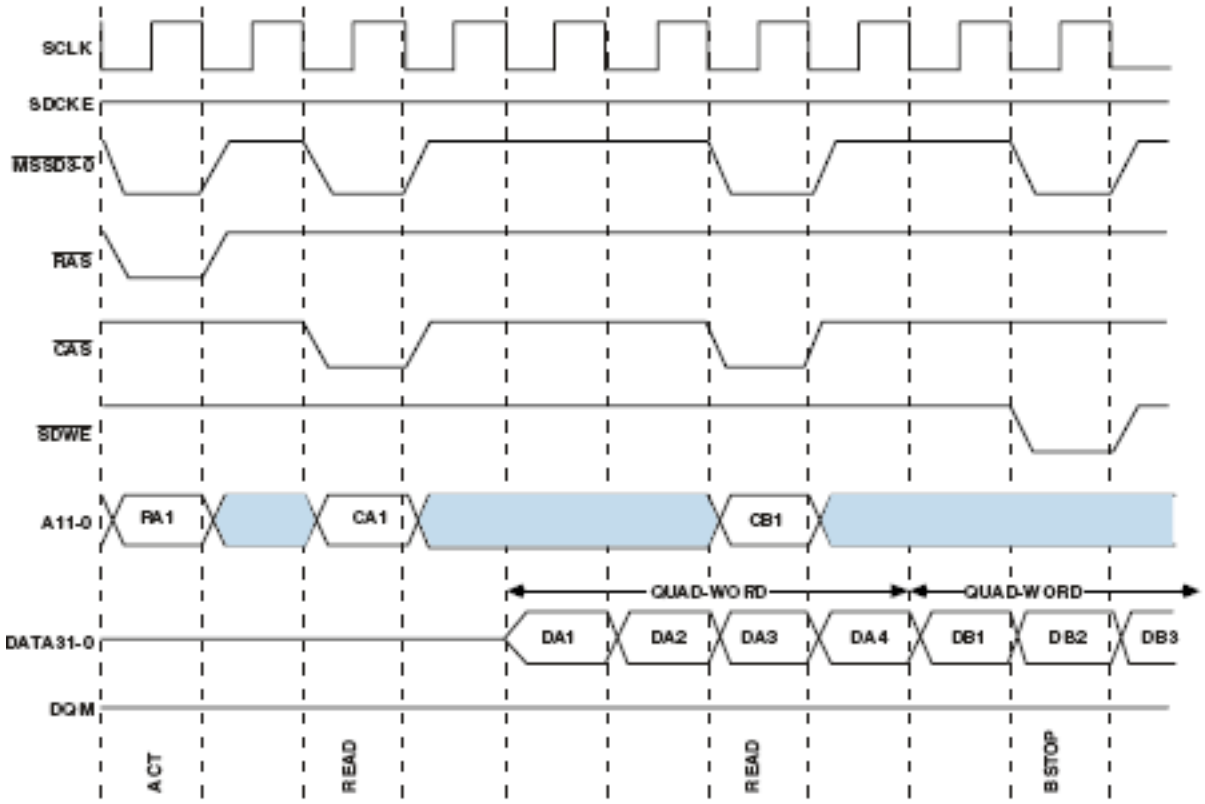


Рисунок 42 – Шина 32 разряда
(пакетное чтение с последующим пакетным чтением на той же странице)

7.4.11.5 Команда записи (write)

Команда Write (Таблица 89) выполняет запись данных в активную страницу. Если это первая запись после активизации страницы, то перед подачей команды записи должен быть соблюден интервал времени tRCD. Команда записи устанавливает начальный адрес слова страницы, с которого начнется запись. Как и в случае чтения, если после записи нет других команд, то следующем такте будет выполняться запись по последовательно увеличенному адресу. Если запись нужно остановить, контроллер SDRAM должен передать команду BSTOP.

Одна транзакция записи занимает разное количество циклов в зависимости от размера операнда транзакции и ширины внешней шины.

- запись одного слова – 1 цикл;
- запись двойного слова на 64-разрядной шине – 1 цикл;
- запись двойного слова на 32-разрядной шине – 2 цикла;
- запись счетверенного слова на 64-разрядной шине – 2 цикла;
- запись счетверенного слова на 32-разрядной шине – 4 цикла.

Одновременно с подачей начального адреса, процессор выставляет на шину данных и записываемые данные. Последовательная запись блока данных называется «страничный режим» или «пакет». При этом в последующих циклах нет необходимости подавать адрес, достаточно только передавать следующие данные.

Когда транзакция записи завершена, контроллер SDRAM может продолжить работу по нескольким путям:

- Если нет новых транзакций SDRAM, то поступает команда BSTOP которая указывает на завершение записи блока данных.

- Если следующая транзакция есть тоже запись в эту же страницу, то контроллер может подавать новую команду записи с новым начальным адресом и данными.
- Если следующая транзакция – это чтение SDRAM на той же странице, то команда BSTOP останавливает транзакцию записи и транзакция чтения начинается с нового цикла.
- Если есть доступ SDRAM к другой странице, то поступает команда BSTOP и далее подаются команды закрыть текущую страницу (PRE) и активизировать следующую. При этом соблюдаются все необходимые задержки, запрограммированные в регистре конфигурации.

Таблица 89 – Состояние вывода во время выполнения команды Запись

Вывод	Состояние
nMSSD3–0	0 (один из четырех)
nCAS	0
nRAS	1
nSDWE	0
SDCKE	1

Ниже приведены временные диаграммы использования команды Write для случая ширины шины 64 бита (Рисунок 43) и ширины шины 32 бита (Рисунок 44). Для обоих случаев задержка nCAS Latency равна 2.

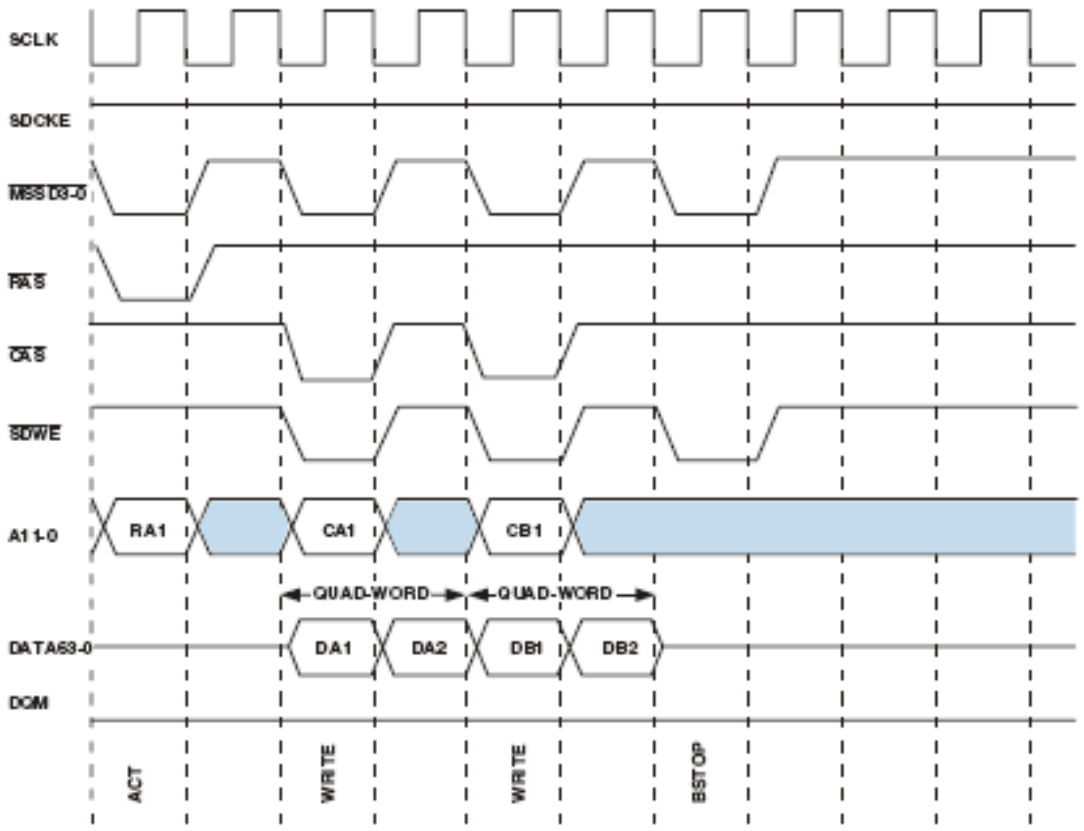


Рисунок 43 – Шина 64 разряда
(пакетная запись с последующей пакетной записью на той же странице)

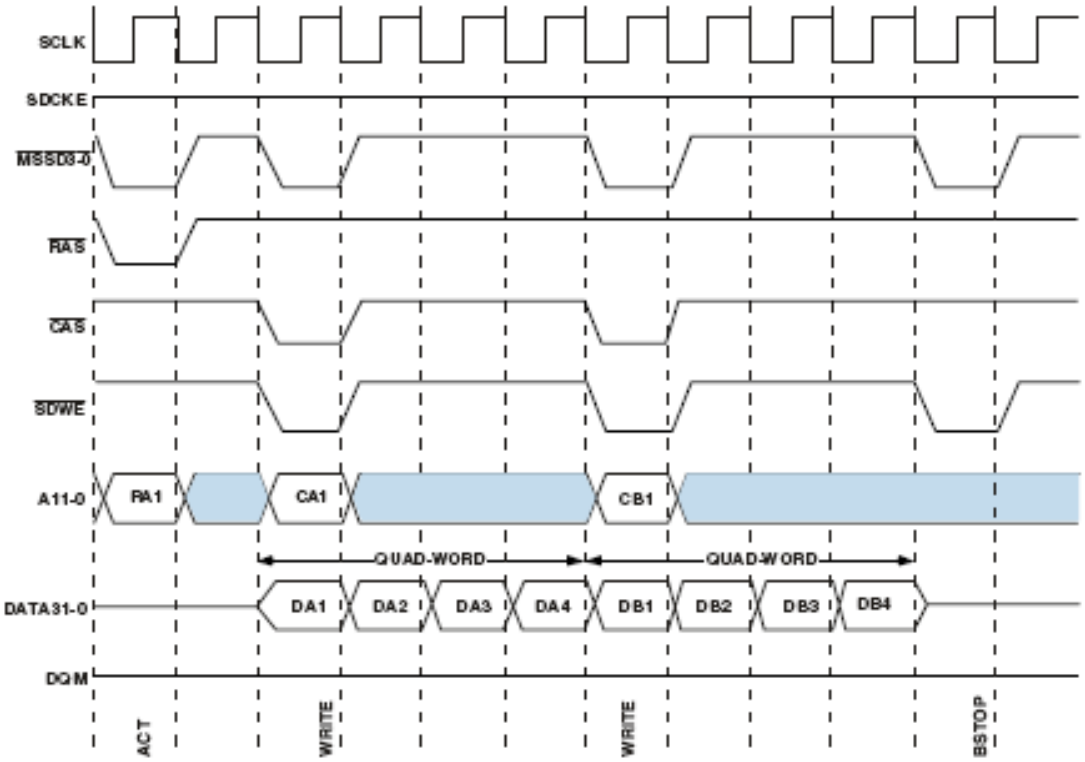


Рисунок 44 – Шина 32 разряда
(пакетная запись с последующей пакетной записью на той же странице)

7.4.11.6 Команда регенерации (REF)

Память SDRAM требует периодической регенерации своих ячеек (строк) для сохранности информации. Команда REF (Таблица 90) является запросом к SDRAM для выполнения транзакции регенерации. Эта команда генерируется автоматически процессором и вызывает регенерацию данных по адресу, который находится внутри SDRAM. Перед выполнением команды REF контроллер SDRAM выполняет проверку наличия активных страниц и, если они имеются – выполняет команду предварительной зарядки (PRE) для активного банка.

Следующая команда активации (ACT) выдается контроллером только после минимальной задержки, равной tRC, где:

$$t_{RC} = t_{RP} + t_{RAS} + 1$$

Частота регенерации (период подачи команд регенерации) устанавливается в регистре SDRCON. В мультипроцессорной системе регенерация осуществляется текущим мастером шины. Каждый подчиненный процессор отслеживает внешнюю шину и сбрасывает свой собственный счетчик при обнаружении цикла регенерации.

Таблица 90 – Состояние выводов во время выполнения команды REF

Вывод	Состояние
nMSSD3–0	0(все)
nCAS	0
nRAS	0
nSDWE	1
SDCKE	1

7.4.11.7 Команда саморегенерации (SREF)

Описанная выше команда регенерации REF требует от процессора непрерывно отслеживать период повторения циклов регенерации и своевременно посылать команды REF. Эта команда может выполняться на фоне выполнения транзакций процессором к другим типам памяти (не SDRAM). Также существует режим, когда сама память отвечает за процедуру регенерации и самостоятельно поддерживает данные достоверными. Этот режим называется режимом саморегенерации, и процессор переводит SDRAM в этот режим путем подачи команды SREF (см. таблицу 91). В данном режиме память недоступна для использования.

Процессор переводит память SDRAM в режим саморегенерации только при передаче шины хост-процессору. Это гарантирует сохранность информации в памяти, если хост-процессор не имеет интерфейса к SDRAM и не может поддерживать процедуру регенерации.

Если хост желает и имеет возможность обмениваться данными с SDRAM, то он выводит SDRAM из режима саморегенерации и работает с памятью в нормальном режиме. По окончании работы, хост обязан вернуть память в прежний режим. При выполнении команды саморегенерации должны быть выполнены все те же временные требования, что и при выполнении команды регенерации.

Перед выполнением команды SREF контроллер подзаряжает все банки микросхем. После команды SREF память выполняет операции регенерации самостоятельно без внешнего контроля. После выхода из режима саморегенерации контроллер SDRAM ждет количество циклов tRC перед выполнением команды активизации банка (ACT).

Таблица 91 – Состояние вывода во время выполнения команды SREF

Вывод	Состояние
nMSSD3–0	0(все)
nCAS	0
nRAS	0
nSDWE	1
SDCKE	0

Режим саморегенерации поддерживается тем, что на линии SDCKE постоянно низкий уровень, что запрещает использование входа синхронизации. Низкий уровень на SDCKE обеспечивается за счет встроенного резистора, подключенного к земле. В нормальном режиме работы процессор имеет резистор, который подтягивает уровень SDCKE к 1.

7.5 DMA контроллер

Прямой доступ к памяти (Direct Memory Access) – это механизм передачи данных без исполнения команд в ядре процессора. Встроенный контроллер DMA избавляет процессор от необходимости передачи данных между внутренней памятью и внешними устройствами\памятью или между портами связи и внешней или внутренней памятью. Полностью интегрированный контроллер DMA позволяет ядру процессора или внешнему устройству указать каналу DMA команду передачи данных и вернуться к нормальной работе, тогда как контроллер DMA выполнит передачу данных в фоновом режиме.

Модуль DMA может быть ведущим или ведомым на шине SOC. При доступе к регистрам DMA со стороны процессорного ядра, модуль DMA выступает в роли ведомого устройства. При работе канала DMA по пересылке данных, модуль DMA выступает в роли ведущего устройства и соревнуется в доступе к различным ресурсам с другими ведущими устройствами. Рисунок 15 показывает ресурсы, к которым посылает свои запросы контроллер DMA. Это интерфейсы внутренней или внешней памяти.

Существуют термины, которые постоянно используются в этой главе. Сюда входят:

- Блок управления передачей (TCB). Блок из четырех слов, который определяет набор параметров для DMA операций.
- Регистр TCB. 128-разрядный регистр, который содержит TCB.
- Загрузка цепочки TCB. Процесс, при котором контроллер DMA загружает новый TCB из памяти и автоматически инициализирует регистр TCB.
- Регистры AutoDMA. Два виртуальных DMA регистра, которые могут быть доступны ведущему устройству кластерной шины. Эти регистры позволяют данным поступать в блок, определенный TCB регистром канала.
- Транзакция канала DMA. Последовательность действий канала по пересылке одного операнда.
- Операнд канала. Порция данных (одно, два или четыре слова) размер которой задается регистром управления и которой оперирует канал в одной транзакции.
- Режим квитирования. Режим работы канала, в котором инициатором одной транзакции выступает внешний запрос.
- Выровненный адрес. Означает, что для двойного слова (64 бита) младший бит адреса всегда равен 0, а для квадрослова (128 бит) два младших разряда адреса должны быть равны нулю.

7.5.1 Архитектура DMA

Контроллер DMA включает 14 каналов предназначенных для выполнения различных типов передачи.

- Каналы с 0 по 3 являются универсальными и позволяют задавать источник и приемник.
- Каналы с 4 по 7 жестко закреплены за передатчиками интерфейсов связи, т.е. источник данных у этих каналов DMA программируемый, а приемник фиксированный и соответствует передатчикам каналов связи с 0-го по 3-й.
- Каналы с 8 по 11 жестко закреплены за приемниками интерфейсов связи, т.е. источник данных этих каналов фиксированный (соответствует приемникам каналов связи с 0-го по 3-й), а приемник данных программируемый.

– Каналы 12 и 13 используют в качестве источника данных внутренний буфер, а приемник данных программируемый, но с ограничением – только внутренняя память процессора. Внутренний буфер каналов 12 и 13 может заполняться данными внешними ведущими устройствами через кластерную шину или процессорным ядром.

Каналы DMA позволяют передачу следующих типов:
– Внутренняя память ⇔ Внешняя память или периферийные устройства в карте внешней памяти;
– Внутренняя память ⇔ Внутренняя память (один процессор);
– Внутренняя память ⇔ Внутренняя память (разные процессоры);
– Внутренняя память ⇔ Хост-процессор;
– Внутренняя память ⇔ Порт связи;
– Внешняя память или периферийные устройства в карте внешней памяти ⇔ Порт связи;
– Внешняя память ⇔ Внешние периферийные устройства;
– Внешняя память ⇔ Внешняя память;
– Порт связи => Порт связи;
– Ведущий на кластерной шине => Внутренняя память через AutoDMA.

Передачи DMA характеризуются направлением потока данных – от источника к приемнику. Если в роли источников или приемников выступает память, то они описываются регистром TCB. Универсальные каналы имеют два TCB регистра: для описания источника и приемника, каналы портов связи и AutoDMA имеют по одному регистру TCB, т.к. в них источник либо приемник фиксирован и не требует дополнительного описания. Рисунок 45 показывает блок-схему DMA контроллера.

DMA поддерживает так называемые цепочки операций, когда одна запрограммированная в канале пересылка после завершения работы вызывает автоматическое чтение из внутренней памяти новой команды пересылки без участия процессора. Эта техника в первую очередь используется для одного DMA канала, но в некоторых случаях может использоваться и для разных каналов.

Процессор также имеет четыре вывода внешнего запроса DMA (nDMAR3–0), которые позволяют внешним устройствам ввода\вывода запрашивать DMA сервис. В ответ на внешний запрос по входу nDMARi, соответствующий i-й канал DMA выполняет передачу в соответствии с его инициализацией. Для инициализации канала программа должна записать в блок управления передачей (TCB) командное слово.

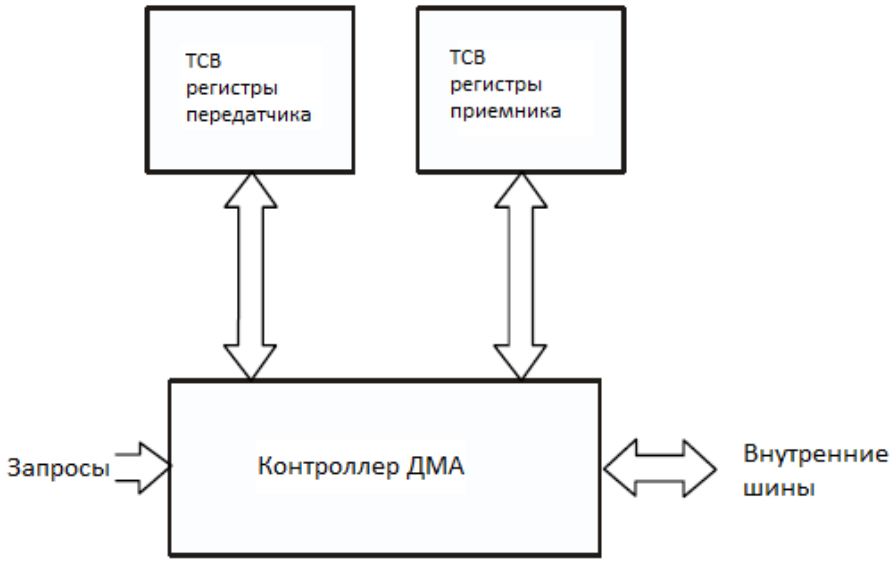


Рисунок 45 – Блок-схема DMA

Блок управления передачей ТСВ является регистром счетверенного слова (128-бит), который содержит информацию, необходимую для работы канала DMA. Вид регистра ТСВ представлен ниже (Рисунок 46). В одном канале может присутствовать приемник и передатчик, либо только приемник или только передатчик. Названия ТСВ передатчика и приемника имеют внутриканальный смысл: ТСВ передатчика передает данные ТСВ приемнику. Сам же ТСВ передатчик берет данные у источника информации, а ТСВ приемника передает принятые данные в пункт назначения.

В случае ТСВ передатчика, четыре слова содержат данные адреса источника информации (DI), количество слов для передачи (DX/DY), приращение адреса (DX/DY), контрольные разряды (DP).

В случае ТСВ приемника эти четыре слова содержат адрес назначения (DI), количество слов, которое нужно получить (DX/DY), приращение адреса (DX/DY), контрольные разряды (DP).

Рисунок 46 – блока управления передачей DMA

Одна передача (пересылка или транзакция) при работе DMA означает копирование порции данных(операнда) из источника данных в приемник данных. Размер порции данных описывается в регистре управления DP и может быть равен одному, двум или четырем 32-разрядным словам. В регистре DX задается количество слов для передачи. Понятно, что количество транзакций будет равно количеству слов, деленному на размер порции данных. После каждой транзакции канал осуществляет изменение адреса на величину приращения, задаваемую в регистре DX(DY). Канал DMA может поддерживать двухмерные пересылки данных, когда адрес изменяется по горизонтальной составляющей X и по вертикальной составляющей У. Поэтому в ТСВ присутствует описатели DX и DY. Обычный режим передачи – одномерный и для него используется только регистр DX. Регистр DY используется только в двухмерном режиме.

Контроллер поддерживает словную адресацию памяти и минимальной адресуемой единицей данных для него является 32-разрядное слово.

Ниже приведено соответствие между номером канала и его характеристиками (Таблица 92).

Таблица 92 – Особенности программирования каналов контроллера.

Номер канала	Источник данных	Приемник данных	Инициирование транзакции
0	Программируется в ТСВ передатчика	Программируется в ТСВ приемника	Программируется. Может использовать вход nDMAR0
1	Программируется в ТСВ передатчика	Программируется в ТСВ приемника	Программируется. Может использовать вход nDMAR1
2	Программируется в ТСВ передатчика	Программируется в ТСВ приемника	Программируется. Может использовать вход nDMAR2
3	Программируется в ТСВ передатчика	Программируется в ТСВ приемника	Программируется. Может использовать вход nDMAR3
4	Программируется в ТСВ передатчика	Передатчик порта связи канал 0.	Всегда только по запросу передатчика канала связи 0
5	Программируется в ТСВ передатчика	Передатчик порта связи канал 1.	Всегда только по запросу передатчика канала связи 1
6	Программируется в ТСВ передатчика	Передатчик порта связи канал 2.	Всегда только по запросу передатчика канала связи 2
7	Программируется в ТСВ передатчика	Передатчик порта связи канал 3.	Всегда только по запросу передатчика канала связи 3
8	Приемник порта связи канал 0	Программируется в ТСВ приемника	Всегда только по запросу приемника канала связи 0
9	Приемник порта связи канал 1	Программируется в ТСВ приемника	Всегда только по запросу приемника канала связи 1
10	Приемник порта связи канал 2	Программируется в ТСВ приемника	Всегда только по запросу приемника канала связи 2
11	Приемник порта связи канал 3	Программируется в ТСВ приемника	Всегда только по запросу приемника канала связи 3
12	Запись данных внешним ведущим устройством в регистр канала	Программируется в ТСВ приемника	Всегда при наличии данных во внутреннем буфере
13	Запись данных внешним ведущим устройством в регистр канала	Программируется в ТСВ приемника	Всегда при наличии данных во внутреннем буфере

7.5.2 Настройка DMA передач

DMA операции могут программироваться ядром процессора, внешним хост-процессором или ведущим процессором кластерной шины. Операция программируется записью в регистры ТСВ. Регистр ТСВ – регистр счетверенных слов, описывающий передачу DMA блока. DMA канал настраивается записью счетверенного слова в каждый из регистров ТСВ. Каждый регистр должен быть загружен начальным адресом для блока, приращением адреса, количеством слов и управляющей информацией.

Если ТСВ запрограммирован на генерацию прерывания, то запрос прерывания формируется, когда блок данных полностью передан.

Регистры ТСВ могут быть доступны только как счетверенные слова

7.5.2.1 Регистры блока управления передачей (TCB)

Каждый TCB регистр имеет длину 128 бит и разделен на четыре 32-битных регистра (Рисунок 47, Таблица 115):

- регистр индекса (DI);
- регистр X количества и приращения (DX);
- регистр Y количества и приращения (DY);
- регистр управления и указателя цепочки (DP);

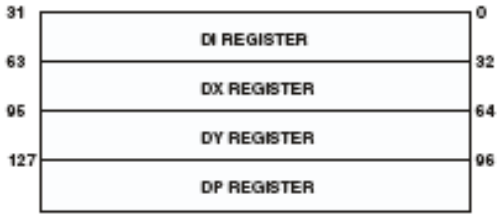


Рисунок 47 – Регистры блока управления передачей

Эти четыре регистра образуют счетверенное слово TCB, которое может быть загружено как выровненное счетверенное слово из внутренней памяти посредство цепочки или с помощью ядра.

Для инициирования нового обмена, после завершения текущего, программа должна записать новые параметры в регистры TCB.

Таблица 93 – Каналы DMA и связанные регистры TCB

Канал DMA	Регистры TCB	Описание
0	DCS0 DCD0	TCB регистр источника канала 0 TCB регистр назначения канала 0
1	DCS1 DCD1	TCB регистр источника канала 1 TCB регистр назначения канала 1
2	DCS2 DCD2	TCB регистр источника канала 2 TCB регистр назначения канала 2
3	DCS3 DCD3	TCB регистр источника канала 3 TCB регистр назначения канала 3
4	DC4	TCB регистр источника канала 4 для передатчика порта связи 0
5	DC5	TCB регистр источника канала 5 для передатчика порта связи 1
6	DC6	TCB регистр источника канала 6 для передатчика порта связи 2
7	DC7	TCB регистр источника канала 7 для передатчика порта связи 3
8	DC8	TCB регистр назначения канала 8 для приемника порта связи 0
9	DC9	TCB регистр назначения канала 9 для приемника порта связи 1
10	DC10	TCB регистр назначения канала 10 для приемника порта связи 2
11	DC11	TCB регистр назначения канала 11 для приемника порта связи 3
12	DC12	TCB регистр назначения канала 0 автоDMA
13	DC13	TCB регистр назначения канала 1 автоDMA

Регистр DI

Данный 32-разрядный регистр содержит начальный адрес блока данных и используется для прямого доступа к памяти. Он может указывать на адрес внешней памяти или внутренней памяти.

Регистр DX

Этот 32-разрядный регистр имеет два поля:

- DXM – младшие биты (с 0-го по 15-й или с 0-го по 21-й) хранят значение модификатора адреса (DX_MODIFY), которое используется для изменения адреса после каждой транзакции;
- DXC – старшие биты (с 16-го по 31-й или с 22-го по 31-й) хранят значение количества слов (DX_COUNT) в блоке данных, который необходимо передать.

Например, если необходимо передать 16 слов данных, непрерывно размещенных в памяти, порциями по 4 слова в каждой передаче, то параметр DXC будет равен 16, а значение DXM будет равно 4. Длина операнда (порции данных) в регистре DP устанавливается равной счетверенному слову.

Существуют ограничения, которые следует принимать во внимание при программировании TCB:

- указатель DI должен содержать адрес, выровненный на границе операнда, определяемого в регистре DP;
- Значение DXM должно быть кратно размеру операнда;
- Значение DXC должно быть кратным длине операнда.

Выбор длины полей модификатора и количества определяется типом обмена, задаваемым полем TY в регистре DP.

Регистр DY

Этот 32-разрядный регистр используется вместе с регистром DX, когда двумерный режим DMA разрешен. Как и DX, регистр DY содержит два поля:

- DYM – младшие биты (с 0-го по 15-й или с 0-го по 21-й) хранят значение модификатора адреса (DY_MODIFY), которое используется для изменения адреса после завершения передачи блока по координате X. Модификация Y – это разница между адресом последнего элемента данных в строке и адресом первого элемента в следующей строке.
- DYC – старшие биты (с 16-го по 31-й или с 22-го по 31-й) хранят значение количества попыток (DY_COUNT) передач по координате X.

Если двумерный (2D) режим DMA запрещен, то регистр DY загружается значением 0 или любым другим допустимым значением.

Например, мы имеем в памяти массив данных в виде матрицы 100x200, т.е. в строке 200 слов данных и таких строк 100. Пусть нам необходимо выполнить передачу блока данных содержащего 15 первых элементов каждой из 10-ти первых строк. В этом случае режим 2D будет иметь следующие параметры: DXC равно 15, DXM равен 1, DYC равно 10, а DYM будет равен 186 (200-15+1). Такое значение модификатора DY позволяет после передачи 15-го элемента текущей строки перейти сразу к первому элементу следующей строки.

Выбор длины полей модификатора и количества определяется типом обмена, задаваемым полем TY в регистре DP.

Регистр DP

Данный регистр задает режим работы передатчика\приемника канала. Подробное описание разрядов регистра приведено ниже (Таблица 94).

Таблица 94 – Описание разрядов регистра DP

Бит	Имя	Назначение
18:0	CHPT	Указатель цепочки. Эти разряды включают в себя разряды 20–2 адреса внутренней памяти, где находится значение следующего регистра TCB. Разряды 1–0 адреса не указываются поскольку адрес должен быть выровненным на границе квадрослова, т.е. биты равны нулю. Например, если содержимое следующего TCB сохраняется в памяти 0x87128–0x8712B тогда поле CHPT будет содержать b0010_0001_1100_0100_1010 (0x21C4A).
21:19	CHTG	Канал-приемник следующей цепочки. Это поле имеет значение только для каналов 4-11 порта связи и указывает, в какой канал DMA будет загружено новое значение TCB: 000 – канал 8 (порт связи 0, приемник) 001 – канал 9 (порт связи 1, приемник) 010 – канал 10 (порт связи 2, приемник) 011 – канал 11 (порт связи 3, приемник) 100 – канал 4 (порт связи 0, передатчик) 101 – канал 5 (порт связи 1, передатчик) 110 – канал 6 (порт связи 2, передатчик) 111 – канал 7 (порт связи 3, передатчик)
22	CHEN	Разрешение загрузки следующей цепочки: 1 – разрешено 0 – запрещено При разрешении загрузки, после окончания передачи всего блока данных, канал выполнит чтение новой TCB из внутренней памяти и загрузит её в канал назначения.
23	DRQ	Разрешение анализа внешнего запроса: 1 – разрешено 0 – запрещено Если разрешено, то канал выполняет одну транзакцию только по внешнему запросу. Если запрещено, то канал выполняет передачу всего блока без ожидания запроса. Бит имеет значение для каналов 0-3 и позволяет им анализировать внешние запросы nDMARx. Для каналов 4-11 он не имеет значения т.к. они всегда работают только по запросу от портов связи.
24	INT	Генерация запроса прерывания после окончания работы канала: 1 – разрешено 0 – запрещено
26:25	LEN	Длина передаваемых данных (операнда) в одном цикле обмена: 00 – резерв 01 – слово 32 бита 10 – длинное слово 64 бита 11 – квадрослово 128 бит Длина операнда накладывает ограничения на адрес источника или приемника – они всегда должны быть выравнены. Также накладываются ограничение на модификаторы и количество передаваемых слов в X.
27	2D	Включение режима 2-х мерной пересылки: 1 – двумерная пересылка 0 – одномерная пересылка

28	PR	Приоритет циклов обмена: 1 – высокий 0 – обычный Уровень приоритета используется во время арбитража каналов между собой. Также уровень приоритета оказывает влияние на вывод nDPA внешней шины.
31:29	TY	Тип обмена (выбор источника либо приемника): 000 – канал выключен 001 – линк порт (только в случае передачи линк-линк) 010 – внутренняя память (16\16) 011 – внутренняя память (22\10) 100 – внешняя память (16\16) 101 – внешнее устройство Flyby 110 – загрузочное EPROM. Вывод nBMS. 111 – внешняя память (22\10)

Ограничения при программировании регистра DP

Неправильное задание конфигурации TCB может вызвать генерацию ошибки каналом. *На использование регистра управления TCB* накладываются следующие *ограничения*:

- Установка длины операнда (LEN). В каналах 0-3 поле LEN должно быть одинаковым в обоих TCB.
- Если поле TY источника или получателя TCB имеет тип 6 (загрузочная EPROM), поле LEN должно быть 0x1 (стандартное слово).
- В каналах 4-11 поле LEN всегда должно быть счетверенным словом (0x3).
- Для каналов AutoDMA, поле LEN должно быть одинаковым с транзакциями в регистр AutoDMA, т.е. если в поле LEN указана длина двойного слова, то запись в регистр AutoDMA должна выполняться только двойными словами.

Имеются *ограничения на задание полей количества* (DX и DY):

- Для каналов 0-3 общее количество слов передатчика должно равняться количеству слов приемника.
- Для обычного режима обмена количество слов равно DXC.
- Для двухмерного режима количество слов равно DXC × DYC. При этом бит 2D может быть установлен в одном из TCB и сброшен в другом (для каналов с 0-го по 3-й).
- Нет ограничений для регистра количества в DMA каналах связи. Во всех случаях, если канал не простаивает, DXC не может быть нулевым. Если 2D режим установлен, DYC не может быть 0 или 1

Из поля TY регистра управления видно, что можно задать 7 типов обмена. Однако в зависимости от номера канала существует ограничения в выборе типа. Так в таблице показаны разрешенные (отмечены +) комбинации типов в каналах 0-3.

Таблица 95 – Допустимые комбинации передач каналов с 0 по 3

Тип передатчика	**Тип приемника**						
	1	**2**	**3**	**4**	**5**	**6**	**7**
1							
2				+		+	+
3				+		+	+
4		+	+	+	+	+	+
5				+			+
6		+	+				
7		+	+	+	+	+	+

Все другие комбинации при их установке в каналах 0-3 вызовут генерацию ошибки канала.

Для каналов 4-7 также существует ограничение в выборе типа обмена: в качестве источника данных могут быть выбраны или внутренняя (тип 2, 3), или внешняя (тип 4, 7) память.

Ограничения в выборе типа для каналов 8-11: в качестве приемника данных могут быть выбраны только внутренняя (2, 3), внешняя (4, 7) память либо порт связи (тип 1).

Для каналов 12 и 13 единственным типом может быть только внутренняя память (тип 2, 3).

Ограничения запросов DMA

В каналах 0-3 бит DRQ должен быть одинаковым в обоих регистрах TCB.

Ограничения выравнивания

Если поле LEN равно 2 (двойное слово) следующие поля должны быть четными (бит 0 очищен):

- DI;
- DXM;
- DYM (при необходимости);
- DXC.

Если поле LEN равно 3 (счетверенное слово) значения этих же полей должны быть кратны 4 (биты 1–0 очищены).

Для типа 6 (загрузочная EPROM) данные регистры должны быть все кратны 4 (биты 1–0 очищены).

Ограничения диапазона адресов

Следующая связь должна существовать в любой момент работы TCB между полем TY в регистре DP и адресом:

- TY = 1 (канал связи) ⇒ адрес – один из регистров передачи порта связи.
- TY = 2 или 3 (внутренняя память) ⇒ адрес – в диапазоне внутренней памяти и не в регистрах.
- TY = 4 или 7(внешняя память) ⇒ адрес – в диапазоне внешней памяти (биты 31–22 отличны от нуля). Это может быть пространство хост-процессора, внешней памяти или мультипроцессорное пространство.

7.5.2.2 Регистр статуса DMA (DSTAT/DSTATC)

Регистр статуса DSTAT предназначен только для чтения и отражает текущее состояние каналов контроллера. Регистр имеет разрядность 64 бита и два адреса доступа. Второму адресу соответствует имя DSTATCL регистра. При чтении регистра DSTATCL все коды ошибок в регистре статуса очищаются, и состояние канала изменяется на состояние выключено. Регистры статуса DMA (DSTAT, DSTATCL) могут быть доступны только как двойное или счетверенное слово. В регистре состояния каждому каналу контроллера отведено 3 бита. С помощью этих бит кодируется состояние канала. Возможны следующие значение состояния канала:

- 000 – выключен;
- 001 – активен (работает);
- 010 – завершил работу;
- 011 – резерв;

- 100 – ошибка. Инициализация активного канала;
- 101 – ошибка. Запрещенная конфигурация канала;
- 110 – резерв;
- 111 – ошибка. Некорректный адрес.

Назначение разрядов регистра состояния приведено ниже (Таблица 96).

Таблица 96 – Биты регистра состояний

Бит	Имя	Описание
2:0	CH0	Состояние канала 0
5:3	CH1	Состояние канала 1
8:6	CH2	Состояние канала 2
11:9	CH3	Состояние канала 3
14:12	CH4	Состояние канала 4
17:15	CH5	Состояние канала 5
20:18	CH6	Состояние канала 6
23:21	CH7	Состояние канала 7
31:24	-	-
34:32	CH8	Состояние канала 8
37:35	CH9	Состояние канала 9
40:38	CH10	Состояние канала 10
43:41	CH11	Состояние канала 11
49:44	-	-
52:50	CH12	Состояние канала 12
55:53	CH13	Состояние канала 13
63:56	-	-

- Активный статус DMA устанавливается, когда канал DMA включен (когда поле TY регистра DPx установлено).
- Статус завершения DMA устанавливается, когда последняя транзакция DMA завершена.
- Статус ошибки DMA устанавливается, если обнаружено запрещенное или ошибочное состояние.

В случае если обнаруживается ошибочное состояние, DMA может генерировать аппаратное прерывание, если оно включено, и биты статуса могут быть очищены только чтением из регистра DSTATC. Регистры DSTAT и DSTATC должны читаться как сдвоенное или счетверенное слово. Чтение этих регистров как стандартное слово недопустимо.

7.5.2.3 Регистр управления DMA

Биты регистра управления выполняют одну единственную функцию – приостановка работы соответствующего им канала DMA. Подробное описание разрядов регистра управления приведено ниже (Таблица 97).

Таблица 97 – Биты регистра управления

Бит	Имя	Описание
0	DMA0_P	Пауза (если 1) для канала 0 DMA
1	DMA1_P	Пауза (если 1) для канала 1 DMA
2	DMA2_P	Пауза (если 1) для канала 2 DMA
3	DMA3_P	Пауза (если 1) для канала 3 DMA

4	DMA4_P	Пауза (если 1) для канала 4 DMA
5	DMA5_P	Пауза (если 1) для канала 5 DMA
6	DMA6_P	Пауза (если 1) для канала 6 DMA
7	DMA7_P	Пауза (если 1) для канала 7 DMA
9:8		
10	DMA8_P	Пауза (если 1) для канала 8 DMA
11	DMA9_P	Пауза (если 1) для канала 9 DMA
12	DMA10_P	Пауза (если 1) для канала 10 DMA
13	DMA11_P	Пауза (если 1) для канала 11 DMA
15:14		
16	DMA12_P	Пауза (если 1) для канала 12 DMA
17	DMA13_P	Пауза (если 1) для канала 13 DMA
31:18		

Регистр управления DMA имеет три адреса:

- DCNT – регулярный адрес для чтения и записи.
- DCNTST – адрес для установки бит. Доступен только по записи. Запись 1 устанавливает соответствующий бит, а запись 0 не изменяет его значения.
- DCNTCL – адрес для очистки бит. Доступен только по записи. Запись 0 сбрасывает значение бита, а запись 1 не изменяет его значение.

При установке бита паузы в 1 соответствующий канал DMA приостанавливает свою работу и возобновляет её только после того, как бит станет равен 0.

Регистры управления DMA (DCNT, DCNTST, DCNTCL) могут быть доступны как нормальное, удлиненное и счетверенное слово. Начальное значение DCNT после сброса равно 0.

Управление битом паузы каналами с 4 по 13-й довольно простое т.к. установка бита паузы для них означает остановку генерации транзакций каналом. При этом все уже запущенные каналами транзакции спокойно завершаются.

Управление битом паузы для каналов с 0 по 3-й более сложное. Это связано с тем, что транзакции этих каналов состоят из двух самостоятельных частей: транзакции чтения данных из источника в буфер DMA и затем транзакции записи данных из буфера по адресу приемника. Канал может запускать до 8-ми транзакций чтения, прежде чем хотя бы одна транзакция записи выполнится. Бит паузы влияет только за запуск транзакций чтения. Но установка бита паузы не означает, что канал остановил работу. Это будет сделано, только когда все запущенные транзакции чтения загрузят в буфер данные и затем выполнятся все соответствующие им транзакции записи. Пользователь, после установки бита паузы, может захотеть изменить настройки канала. В результате это приведет к потере прочитанных данных т.к. операции записи могут быть сброшены. Чтобы этого не произошло, канал DMA после установки бита паузы для каналов 0-3 смотрит, есть ли у него незавершенные транзакции записи и, если таковые имеются – канал генерирует сигнал блокировки доступа к SOC шине со стороны ядра. Таким образом, если процессор сразу после записи бита паузы пытается произвести операцию на SOC-шине, то он может быть остановлен на некоторое время. Данное время может быть значительным, если канал работает с внешней памятью. Это обстоятельство нужно учитывать при работе с каналами 0-3. По возможности не стоит использовать бит паузы для каналов 0-3, если источником данных для каналов является внешняя память и при этом необходима высокая скорость реакции на прерывание.

7.5.3 Операции DMA контроллера

7.5.3.1 DMA управление портами связи

Четыре порта связи процессора способны использовать восемь каналов DMA для обработки приема и передачи данных. Два DMA канала приписаны к каждому из портов связи. Имеется однозначное соответствие между приемником и передатчиком конкретного порта связи и соответствующим им каналом DMA:

- Приемник порта связи имеет соответствующий ему регистр TCB получателя в канале DMA.
- Передатчик порта связи имеет соответствующий ему регистр TCB приемника в канале DMA.

Для передачи из внутренней или внешней памяти в порты связи используются каналы 4-7. Регистры источника TCB программируются адресом внутренней/внешней памяти DI, инкрементом адреса DXM, количеством слов для передачи DXC и управляющими разрядами DP. По окончании программирования TCB, автоматически начинается передача и продолжается до завершения передачи блока. При этом транзакции всегда инициируются передатчиком порта связи в момент, когда он готов принять новую порцию данных. Данный процесс повторяется, пока блок данных не передается полностью.

Для передачи из порта связи во внутреннюю или внешнюю память, используются каналы 8-11. Регистры приемника TCB программируются адресом внутренней/внешней памяти DI, инкрементом адреса DXM, количеством слов для передачи DXC и управляющими разрядами DP. По окончании программирования TCB, автоматически начинается передача. При этом транзакции канала инициируются приемником порта связи в момент, когда у него есть принятые данные. После генерации запроса, принятые портом связи данные считываются соответствующим каналом DMA и отсылаются по адресу получателя. Данный процесс повторяется, пока блок данных не передается полностью.

Количество слов для передачи или приема определяется как количество DXC или количество DXC умноженное на количество DYC (если режим 2D установлен), но в любом случае разрешена передача или прием только счетверенными словами. Приемник порта связи может сигнализировать каналу DMA закончить отсчет количества передач до того, как счетчики DXC и DYC исчерпаны. Также DMA может сигнализировать передатчику порта связи, что была выполнена отправка последнего операнда.

7.5.3.2 DMA управление внешним портом

Передача данных через внешний порт, посредством кластерной шины, перемещает данные между внутренней памятью процессора и внешней памятью или внешними устройствами ввода-вывода. Для передачи из внутренней во внешнюю память, в регистры источника TCB записывается адрес внутренней памяти DI, приращение адреса DXM, количество слов для передачи DXC и управляющие разряды DP. Регистры TCB приемника заполняются адресом внешней памяти DI, приращением адреса DXM, количеством слов для передачи DXC и управляющими разрядами DP. При передаче в обратном направлении источник и приемник меняются местами.

После окончания настройки, передача начинается автоматически и продолжается до полной передачи блока. Передачи выполняются порциями (по одному операнду на транзакцию). При этом канал может инициировать транзакции непрерывно, либо по одной транзакции после каждого импульса nDMAR3-0.

Для внешнего устройства и внешней памяти (только SDRAM) имеется режим работы канала, позволяющий выполнить пересылку между ними напрямую по внешней шине без участия внутренних шин процессора (передача в режиме flyby).

Внешнее устройство ввода-вывода, в отличие от памяти, сообщает каналу DMA о готовности к передаче данных. Устройство-источник готово, если у него есть данные для записи. Приемник готов, если у него есть место в буфере записи. Существует две техники синхронизации этих устройств с DMA каналами процессора:

- Источник и приемник могут подтверждать запрос nDMARx каждый раз при готовности передачи новых данных. Запросы аккумулируются в канале, и при их наличии транзакция запускается по соответствующему каналу. Возможно аккумулирование до 15 запросов.
- Источник, который может быть ведущим на кластерной шине может записывать данные в регистр AutoDMA. После того как данные записаны в регистр AutoDMA, канал AutoDMA передает принятые данные в соответствии с его настройками.

Для передачи данных между внутренней и внешней памятью могут использоваться данные шириной 32, 64 и 128 бит. В основном для работы с внешней памятью используются каналы 0-3. Однако порты связи тоже могут передавать данные размещенные во внешней памяти либо принимать данные во внешнюю память.

Процессор имеет несколько внешних выводов, которые используются только для операций DMA. Описание этих выводов приведено ниже (Таблица 98).

Таблица 98 – Описание выводов DMA внешнего порта

Сигнал	Описание
nDMAR3–0	Выводы запроса сервиса DMA. Разрешают внешним устройствам ввода/вывода запрашивать сервисы DMA из процессора. При наличии активного запроса на некотором nDMARx, соответствующий ему канал DMA выполнит передачу операнда в соответствии с настройками канала. Канал DMA игнорирует внешний запрос если он не инициализирован либо если бит разрешения внешнего запроса сброшен. Выводы запроса nDMAR3-0 чувствительны к фронту.
nIOWR	Запись устройства ввода/вывода. Используется в режиме FLY_BY. Данный сигнал позволяет записать считываемые из внешней динамической памяти данные во внешнее устройство без использования ресурсов процессора.
nIORD	Чтение устройства ввода/вывода. Используется в режиме FLY_BY. Данный сигнал позволяет прочитать данные из внешнего устройства и сразу же записать их во внешнюю динамическую память без использования ресурсов процессора.
nIOEN	Разрешение выходного сигнала устройства ввода/вывода. Разрешает выходным буферам внешнего устройства ввода/вывода совершать транзакции FLY_BY между устройством и внешней памятью. Активный во время транзакций FLY_BY

При доступе к внешней памяти контроллер DMA использует те же интерфейсные сигналы, что и ядро процессора.

7.5.3.3 Регистр управления AutoDMA

Каналы AutoDMA (каналы 12 и 13) используют режим работы, в котором контроллер DMA выступает в роли ведомого устройства и виден для ведущих устройств как два виртуальных регистра: AutoDMA0 (канал 12) и AutoDMA1

(канал 13). Когда внешний ведущий (хост или другой процессор) записывает в один из регистров AutoDMA, этот регистр генерирует запрос к приписанному ему каналу и DMA канал передает данные во внутреннюю память в соответствии с запрограммированным в нем TCB. Для передачи данных между регистрами данных и внутренней памятью могут использоваться внутренние данные шириной 32, 64 и 128 бит.

Канал AutoDMA включен, если поле TY в регистре DP не равно нулю. Канал может поддерживать цепочки операций.

Если происходит запись в регистр AutoDMA, когда канал выключен и не занят загрузкой нового TCB цепочки, записанные данные теряются, происходит прерывание аппаратной ошибки и ошибка отражается в регистре SYSTAT.

Если запись в регистр AutoDMA происходит, когда канал выключен, но занят загрузкой TCB новой цепочки, то данные помещаются в буфер и передаются по назначению, когда загрузка TCB для канала завершается. В этом случае нет генерации ошибки.

7.5.4 DMA передача

Следующие подразделы описывают работу контроллера DMA.

7.5.4.1 Адресация памяти

Контроллер DMA поддерживает словную адресацию памяти. Это означает, что минимальной адресуемой единицей является 32-разрядное слово и адрес в 32-разрядном индексном регистре есть адрес слова.

Благодаря 32-разрядному регистру адреса, DMA может получать доступ к полному адресному пространству процессора. Каждый канал включает регистр индекса DI и поле модификации DXM в своём регистре TCB, которые используются для адресации буфера данных в памяти. 32-битный регистр TCB DI содержит начальный адрес блока и должен инициализироваться стартовым адресом буфера данных. Поле модификации DXM содержит значение со знаком (т.е. положительное или отрицательное), длиной 16 или 22 бита, и определяет величину изменения адреса после выполнения каждой транзакции. Измененное значение адреса используется в следующей транзакции. Само значение адреса в регистре DI не указывает на адресацию внутренней или внешней памяти. Для того чтобы указать какому типу памяти соответствует адрес, используется дополнительное поле TY в регистре управления DP блока TCB. Если в результате модификации адрес выйдет за границы существующей памяти или сменит область внутренней памяти на внешнюю (или наоборот), контроллер DMA все равно будет обращаться к области памяти в соответствии с полем TY.

Передаваемые данные могут быть стандартными, сдвоенными или счетверенными словами. Это указывается в поле LEN регистра управления DP.

Каждый канал DMA имеет поле количества DXC в регистре DX, которое должно быть инициализировано количеством слов для передачи, независимо от длины данных (32, 64 или 128 бит). Счетчик количества уменьшается после каждой передачи DMA в данном канале. Значения уменьшения равны 1, 2 или 4 и определяются полем LEN. Когда количество достигает значения ноль, DMA завершается и может быть сгенерировано прерывания для информирования процессора.

Если поле количества DXC в TCB DX инициализируется нулем, DMA передача на этом канале возможна и будет соответствовать выполнению передачи максимально возможного количества данных. Это происходит из-за того, что первая передача начинается до того, как проверяется значение количества.

Значение количества анализируется после вычитания из него 1. Верный способ отключить DMA канал – сбросить поле TY в соответствующем управляющем регистре.

7.5.4.2 Приоритеты каналов DMA

Система приоритетов используется при выборе канала DMA для обслуживания, т.к. в одном такте может поступить активный запрос более чем от одного канала DMA. Напомним, что в зависимости от бита PR в регистре DP блока TCB, канал может относиться к группе каналов с высоким приоритетом или к группе с низким приоритетом. Внутри каждой группы каналов используется фиксированная система приоритетов между каналами. Таблица 99 описывает каналы DMA в порядке уменьшения приоритетов.

Приоритет устанавливается исходя из следующих соображений:

- Входящие данные имеют более высокий приоритет, чем исходящие, для уменьшения вероятности простоя кластерной шины. Для этого каналы регистры AutoDMA имеют высочайший приоритет. Приемники портов связи имеют более высокий приоритет, чем передатчики.
- Каналы портов связи имеют более высокий приоритет, чем каналы общего назначения, т.к. их более низкая пропускная способность не должна блокировать шину.
- Каналы общего назначения дополнительно имеют циклически изменяемый приоритет в последовательности 3-2-1-0.

Контроллер DMA анализирует запросы от всех каналов внутри двух групп и в течение одного такта определяет канал с самым высоким приоритетом. Канал, выбранный внутри высокоприоритетной группы, имеет преимущество над каналом обычной группы. Выбранному каналу в текущем такте предоставляется право выполнить транзакцию. В следующем такте арбитраж повторяется.

Загрузка цепочки TCB получает приоритет канала. Например, передача данных канала с более высоким приоритетом производится перед загрузкой цепочки TCB с более низким приоритетом.

Таблица 99 – Приоритеты каналов DMA

Канал	Описание	Приоритет
Канал 13	Регистры AutoDMA	Наиболее высокий
Канал 12		
Канал 11 (Link 3)	Приемники портов связи	
Канал 10 (Link 2)		
Канал 9 (Link 1)		
Канал 8 (Link 0)		
Канал 7 (Link 3)	Передатчики портов связи	
Канал 6 (Link 2)		
Канал 5 (Link 1)		
Канал 4 (Link 0)		
Канал 3 (nDMAR3)	Каналы общего назначения	
Канал 2 (nDMAR2)		
Канал 1 (nDMAR1)		
Канал 0 (nDMAR0)		Наиболее низкий

7.5.4.3 Циклически присваиваемый приоритет

Для каналов 0-3 сделана дополнительная модификация – циклический приоритет внутри их группы. Это означает, что после каждого цикла арбитража приоритет каналов 0-3 меняется циклически. Приоритеты каналов после сброса следующие:

- канал 3 (самый высокий);
- канал 2;
- канал 1;
- канал 0 (самый низкий).

Каждый раз, если одному из каналов 0-3 предоставлялась возможность выполнить транзакцию, приоритет каналов меняется. Только что работавший канал получает высочайший приоритет, остальные приоритеты расставляются, как описано выше, но по циклу. Например, если канал 1 в текущем такте выполняет транзакцию, то в следующем такте приоритет каналов следующий:

- канал 1 (самый высокий);
- канал 0;
- канал 3;
- канал 2 (самый низкий).

Если следующую транзакцию делает канал 0, то приоритеты следующие:

- канал 0 (самый высокий);
- канал 3;
- канал 2;
- канал 1 (самый низкий).

Приоритеты каналов 0 – 3 остаются неизменными при выборе каналов с более высоким приоритетом (4-13). По окончании транзакции канала с более высоким приоритетом, порядок приоритетов в каналах 0-3 сохраняется прежним.

В примерах выше подразумевалось, что все четыре каналы не устанавливают бит приоритета в TCB или все устанавливают, т.е. принадлежат одной группе. Если один или более канал установлены на высокий приоритет, циклический алгоритм применяется для каждой группы в отдельности. Например, если после сброса у канала 2 установлен бит приоритета, приоритеты следующие:

- канал 2 (установлен бит);
- канал 3 (самый высокий);
- канал 1;
- канал 0 (самый низкий).

Если канал 1 запрашивает доступ и получает, схема приоритетов следующая:

- канал 2 (установлен бит);
- канал 1 (самый высокий);
- канал 0;
- канал 3 (самый низкий).

Если в течение работы канала 1, канал 2 генерирует запрос, то он получает доступ и работа канала 1 прерывается. После окончания передачи каналом 2, схема приоритетов для каналов с более низкими приоритетами не меняется и канал 1 выигрывает арбитраж и продолжает передачу. Если более одного канала имеют установленный бит приоритета, арбитраж между этими каналами проходит в аналогичной циклической манере.

7.5.4.4 Приоритет DMA на внутренних шинах процессора

Канал DMA, выигравший арбитраж приоритетов среди всех каналов внутри контроллера, получает возможность выполнить транзакцию и генерирует запрос к одному из двух ресурсов системы (Рисунок 15):

- SOC IFIFO – входной буфер SOC-интерфейса для доступа к внутренней памяти;
- EBIU OFIFO – выходной буфер внешнего интерфейса для доступа к внешней памяти.

При арбитраже запросов к SOC IFIFO контроллер DMA имеет самый низкий приоритет. Первыми получают доступ либо считываемые процессором с SOC-шины данные, либо запрос от EBIU IFIFO внешнего интерфейса.

При арбитраже запросов к EBIU OFIFO контроллер DMA также имеет самый низкий приоритет. Первым получает доступ ядро процессора. При этом не имеет значения высокоприоритетный запрос DMA или нет.

Каналы DMA 8-11, работающие с приемниками портов связи, получают данные из буферов приемника напрямую без арбитража доступа. Каналы 4-7, работающие с передатчиками портов связи, запускают транзакции чтения внешней или внутренней памяти и на этом их работа заканчивается. Прочитанные данные самостоятельно поступают в передатчик порта связи из EBIU IFIFO (при чтении внешней памяти) или SOC OBUF (при чтении внутренней памяти). При этом оба эти модуля при записи данных в порт связи участвуют в арбитраже за доступ к SOC-шине. Запрос SOC OBUF имеет наивысший приоритет, далее идет EBIU IFIFO и самый низкий приоритет у ядра процессора.

7.5.4.5 Цепочка DMA

Традиционный алгоритм работы DMA заключается в программировании процессором канала DMA, выполнение каналом передачи данных и генерации прерывания после завершения работы. В ответ на запрос прерывания процессор может запрограммировать канал на новую передачу.

Использование цепочек в DMA позволяет контроллеру автоматически инициализировать себя следующей передачей блока данных после завершения передачи текущего блока. При этом участие процессора не требуется, и он экономит время, которое потратил бы на обработку прерывания и программирование нового обмена. Используя механизм цепочки, могут быть организованы множественные операции DMA с различными атрибутами и устройствами ввода-вывода.

Для поддержки цепочки в первую очередь используются биты регистра TCB DP:

- Бит CHEN разрешения цепочки.
- Поле CHPT указателя на цепочку. Это адрес в памяти, который содержит следующий TCB для загрузки (где содержится адрес, инкремент, количество слов для передачи и управляющие разряды для следующего передаваемого блока).
- Поле CHTG приемника нового TCB. Прочитанное из памяти значение TCB загружается в канал DMA указанный в данном поле.

Таким образом, в каждом TCB канала имеется механизм, чтобы после окончания работы загрузить в регистр TCB новое значение командного слова. Из-за ограничения размера указателя, хранение значений цепочек возможно только во внутренней памяти процессора.

Существует несколько ограничений на цепочки DMA между каналами:

- Для каналов 0-3 возможна поддержка только одноканальной цепочки, т.е. поле CHTG не имеет значения, и новый TCB цепочки может быть загружен только в вызвавший её канал. При этом в обоих TCB канала бит CHEN должен быть установлен т.к. для работы этих каналов нужна загрузка двух TCB.
- Для каналов DMA портов связи возможна межканальная цепочка, когда один канал может запускать работу в другом канале. Например, канал 8, приняв блок данных и загрузив его в память, может инициировать канал 4 на передачу принятых данных.

Включение и выключение цепочек

Передачи DMA запускаются записью счетверенного слова в регистр TCB (каналы 0-3 требуют загрузки обоих регистров TCB). Цепочка образуется при установленном бите разрешения CHEN и, если указатель цепочки CHPT является разрешенным адресом.

Поле CHPT в TCB DP может быть загружено:

- При инициализации канала;
- Во время работы канала (вставка цепочки).

В первом варианте пользователь заранее знает, что необходимо выполнить несколько разнообразных передач и программирует цепочку операций, загружая первый TCB в канал, а все последующие TCB сохраняя во внутренней памяти.

Во втором случае пользователь запускает передачу одного блока, но в процессе выполнения программы (возможно другим процессом) возникает необходимость в передаче нового блока данных. В этом случае программа определяет, что требуемый канал активен и устанавливает для данного канала бит паузы. Это вызывает приостановку работы канала и позволяет программе

- Если цепочка операций в активном TCB не включена – организовать вставку цепочки путем установки бита CHEN в 1 и загрузки поля указателя CHPT необходимым значением. Значение TCB новой передачи сохраняется во внутренней памяти.

- Если цепочка операций включена – добавить в конец (или в любое место) списка цепочек свою запись.

После выполнения описанных действий процессор может сбросить бит паузы и канал продолжит работу. После сброса бита паузы, приостановленный активный канал выполняет повторный контроль своего состояния и, если все корректно – продолжает работу.

Генерация прерывания в цепочке операций

Пользователь может сам выбрать режим прерывания при работе цепочки. Если необходима генерация прерывания при передаче конкретного блока, бит INT в регистре TCB блока нужно установить в 1. Прерывание произойдет после передачи данного блока. Если необходимо сгенерировать прерывание только после окончания работы всей цепочки, бит INT должен быть установлен только в последнем TCB цепочки.

TCB и загрузка цепочек

Во время загрузки TCB цепочки, регистры TCB канала DMA загружаются значениями, полученными из внутренней памяти. TCB хранится в четырех последовательных словах выровненного счетверенного слова. Чтение нового TCB цепочки инициируется после передачи всего блока данных. Для каналов 0-3 необходимо чтение 2-х TCB. При необходимости чтения цепочки канал формирует запрос на чтение TCB со своим приоритетом, т.е. чтение цепочки каналом имеет такой же приоритет, как и чтение данных.

Двухмерный DMA

Данный раздел описывает изменения в работе, которые происходят при включении режима двухмерного DMA в процессоре. Режим двухмерного DMA включается установкой бита 2D в TCB DP регистре. Этот режим использует внешние и внутренние адреса. Преимущество двухмерной адресации состоит в перераспределении данных из одномерной строки (вектора) в представление в виде двухмерного массива (матрицы) с координатами X и Y. Измерения X и Y определяются в регистрах TCB передатчика и/или приемника. При этом измерение X определяет количество слов в строке массива, а измерение Y – количество строк.

- Измерения массива передатчика и приемника могут отличаться. Главное требование – итоговое число слов в источнике и получателе должно быть равно.
- Двухмерный массив памяти может быть перемещен в одномерный массив и наоборот, если итоговое число слов в источнике и получателе равно.
- Двухмерный блок в памяти может быть передан в порты связи, а блок, полученный из порта связи или из AutoDMA, может быть размещен в памяти как двухмерный массив.

Организация канала двухмерного DMA

В режиме двухмерного DMA адресация к массиву может быть выполнена на любом DMA канале, приемнике или передатчике. Регистр индекса (DI) загружается адресом первого элемента массива и обновляется после каждой передачи операнда на значение модификатора DXM, пока поле количества DXC не достигнет нуля. Величина модификатора DXM в регистре DX содержит смещение, добавляемое к текущему значению адреса для перехода на следующий элемент в измерении X (следующая колонка).

Для режима 2D поле количества DXC в регистре DX имеет дополнительную буферизацию (внутренний буфер CIX). При загрузке TCB, поле количества DXC и буфер CIX загружаются одинаковым значением. В процессе передачи по измерению X количество в DXC уменьшается до нуля и буфер CIX используется для перезагрузки количества в DXC. Можно сказать, что измерение X работает в режиме 2D также, как и в обычном режиме только после передачи всей строки X происходит перезагрузка значения количества.

Регистр DY в TCB используется для задания количества строк массива и для перехода от одной строки к другой. В процессе передачи текущей строки к адресу элемента прибавляется значение DXM и осуществляется переход к следующему элементу строки (элементы не обязательно последовательны). При передаче последнего элемента строки (поле DXC достигает значения ноль) к адресу последнего элемента прибавляется модификатор DYM (модификатор DXM не добавляется). Это позволяет перейти к первому элементу следующей строки. Значение модификации DYM в регистре DY является целочисленным числом со знаком, что позволяет уменьшать или увеличивать адрес.

Значение количества строк в DYC уменьшается на 1 и, если это была не последняя строка, количество DXC перезагружается из CIX.

Когда поле количества DYC достигает нуля, DMA передача блока окончена, образуя матрицу размером X на Y.

Рассмотрим пример задания двухмерной передачи. Пусть мы имеем матрицу с количеством колонок COL и количеством строк ROW. Каждый элемент матрицы имеет размер TXS (он определяется полем LEN в DP и может быть равен 1, 2 или 4). Элементы матрицы размещаются в памяти последовательно, причем сначала элементы первого столбца, затем второго и т.д. Пусть нам необходимо выполнить передачу массива построчно. Следующая формула поможет быть применена для расчета параметров DX и DY.

DX = TXS*COL << 16; // количество слов данных в строке

DX |= TXS*ROW; // шаг от текущего элемента строки к следующему

DY = ROW << 16; // количество строк

DY |= (-TXS*(((COL-1)*ROW)-1))&0xFFFF; // переход к следующей строке

Операция «И» с 0xFFFF использовалась для того чтобы ограничить длину отрицательного модификатора размеров в 16 бит перед операцией логического ИЛИ с количеством строк.

Двухмерные DMA операции

Рассмотрим подробно действия, которые происходят внутри канала при двухмерной передаче. Все действия происходят в течение одного такта, но имеют причинно-следственную связь.

- Берется текущий адрес в регистре DI и запускается транзакция передачи операнда длиной LEN (чтение либо запись).
- При подтверждении отправки транзакции происходит вычитание из DXC числа слов в операнде. Результат вычитания анализируется.
- Если результат вычитания не равен нулю, к содержимому регистра DI прибавляется значение модификации DXM. Результат вычитания помещается в поле количества DXC, а новое значение адреса в DI. На этом действия канала в текущей транзакции закончены.
- Если результат вычитания равен нулю, происходит вычитание единицы из поля количества DYC. Результат вычитания анализируется.

- Если результат вычитания количества DYC равен нулю – канал завершает работу.
- Если результат вычитания не равен нулю, в регистр количества DXC загружается начальное значение из буфера CIX, к значению регистра DI прибавляется значение модификатора DYM, результат вычитания загружается в поле количества DYC. Процесс повторяется.

Особенность двухмерного DMA (и любого DMA) в том, что первая передача начинается до анализа количества. Это означает, что DMA не может быть выключен установкой в ноль количества в регистре DXC или DYC. Для отключения двухмерного DMA необходимо очистить бит 2D в регистре управления DP. Для отключения канала DMA, необходимо очистить поле TY в регистре DP.

7.5.4.6 Прерывания DMA

Прерывание окончания DMA генерируется, если бит INT в регистре канала DP установлен.

DMA генерирует прерывание, когда количество передач в активном канале DMA доходит до нуля и передача заканчивается. Когда назначением DMA является внутренняя память, окончание передачи означает, что последний элемент данных попал в буфер SOC IFIFO.

Когда назначением DMA является внешняя память, окончание передачи означает, что последний элемент данных попал в буфер EBIU OFIFO.

Если назначением передачи является передатчик порта связи, окончание передачи означает, что последний элемент данных был загружен в буфер передатчика.

Видно, что генерация прерывания для случая, когда пункт назначения данных в памяти, не означает, что последние данные уже находятся в памяти. Это произойдет с задержкой. Для случая внутренней памяти задержка может достигать 32 тактов процессора (если запись идет в блок памяти, куда одновременно обращаются с чтением все три внутренних процессорных шины),

16 тактов (если запись идет в блок памяти без конфликтов). Минимальная задержка 3 такта. Во всех случаях процессор не в состоянии за это время обработать запрос прерывания и попытаться прочитать последнее слово переданных данных.

Задержка для внешней памяти непредсказуема, т.к. зависит от частоты системной шины, от успешности получения доступа к шине при арбитраже и других факторов.

Каждый канал DMA имеет собственный бит запроса прерывания в контроллере прерываний. Прерывания DMA фиксируются в регистре ILAT и разрешаются в регистре IMASK. Приоритет запросов прерываний от каналов DMA фиксированный и определен в контроллере прерываний.

Опрос регистра DSTAT может использоваться как альтернатива прерываниям для определения окончания одиночной последовательности DMA. Если включены цепочки, опрос DSTAT нельзя использовать, т.к. следующая последовательность DMA может быть на подходе в момент возврата статуса по результатам опроса.

7.5.4.7 Запуск и окончание последовательностей DMA

Данная глава рассматривает запуск, приостановку, возобновление и окончание последовательностей DMA.

Запуск последовательности DMA

Последовательность DMA для каналов 0-3 запускается в одном из случаев:
- включен канал DMA, бит запроса DRQ равен нулю;
- включен канал x DMA, установлен бит запроса DRQ и на входе nDMARx происходит переход от высокого к низкому.

Последовательность DMA для каналов 4-7 запускается в случае:
- включен канал DMA и имеется запрос от передатчика канала связи на прием данных в буфер передатчика. Бит запроса DRQ не имеет значения.

Последовательность DMA для каналов 8-11 запускается в случае:
- включен канал DMA и имеется запрос от приемника канала связи о наличии данных в буфере приемника. Бит запроса DRQ не имеет значения.

Последовательность DMA для каналов 12-13 запускается в случае:
- включен канал DMA и имеются данные в буфере канала. Буфер каждого канала может принять до 4-х операндов. Бит запроса DRQ не имеет значения.

Для запуска новой последовательности DMA после завершения текущей, программа должна записать новые параметры в регистры TCB. Запись активного TCB в активный TCB регистр вызывает ошибку.

Приостановка последовательности DMA

Последовательность операций DMA в некотором канале приостанавливается, когда соответствующий ему бит паузы в регистре DCNT установлен. Установка бита паузы может произойти в любой момент работы канала: канал может работать, загружать цепочку или завершить работу. Поэтому после установки бита паузы пользователь должен проанализировать состояние канала, если он намерен выполнять с каналом какие-то действия.

Если момент установки паузы пришелся на загрузку цепочки, бит паузы не остановит загрузку цепочки, и до момента загрузки цепочки поле TY регистра TCB будет читаться как 0. Хотя биты состояния в этот момент будут показывать, что канал активен.

Если момент установки бита паузы пришелся на завершение работы каналом, поле TY регистра TCB будет равно нулю и регистр состояния укажет, что канал не активен.

Установка бита паузы в момент нормальной работы канала (пересылка данных) вызывает останов операций передачи. При этом каналы с 4 по 13 немедленно прекращают операции пересылки. Каналы с 0 по 3 немедленно прекращают операции чтения данных из источника, но продолжают выполнять незавершенные операции записи. В случае наличия незавершенных операций записи, после установки бита паузы SOC-шина становится недоступной для ядра процессора. Разблокировка доступа произойдет только после завершения операций записи.

Установка бита паузы позволяет записать новое активное значение TCB (поле TY не равно нулю) в активный регистр TCB. Ранее упоминалось, что такая запись приводит к генерации аппаратной ошибки. Однако в случае паузы такая запись разрешена. Если случай записи без бита паузы рассматривается как сбой в программе, то запись с установленным битом паузы есть осознанная последовательность действий по смене режима работы канала. Это позволяет отложить задачу, выполняемую каналом, и запрограммировать канал на более

приоритетную задачу. После её завершения состояние канала может быть восстановлено.

Имеется особенность в перепрограммировании каналов с 0 по 3. Если новое активное TCB имеет поле TY такое же, как активный регистр TCB канала, то данная запись рассматривается только как попытка модификации регистра DP (например, попытка вставить цепочку). В этом случае при записи обновляется только значение регистра DP. Все другие значения неизменны. Поэтому, чтобы задать новый обмен для каналов с 0 по 3 необходимо:

- сохранить текущие параметры регистра TCB;
- записать в регистр TCB значение с полем TY равным нулю;
- записать активное значение в TCB регистр.

Возобновление последовательности DMA

Последовательность операций DMA в некотором канале возобновляется, когда соответствующий ему бит паузы в регистре DCNT сбрасывается в ноль. Для каналов с 4 по 13 это означает немедленное возобновление работы в соответствии с командой TCB. Для каналов с 0 по 3 сброс бита паузы может привести к процедуре проверки правильности конфигурации TCB, если во время паузы были записи в TCB регистры.

Окончание последовательности DMA

Последовательность DMA заканчивается в одном из случаев:

- Регистры количества равны нулю и цепочка заканчивается.
- Канал отключен сбросом поля TY в одном из TCB и DP регистрах. При этом такая запись не активного TCB может происходить в любой момент. Она очень неприятна для каналов с 0 по 3 т.к. до 8-ми транзакций чтения может быть запущено во внутреннюю или внешнюю память и после отключения канала все эти данные поступят во внутренний буфер контроллера DMA и будут утеряны, если канал выключен. Если к этому времени канал будет включен для новой задачи, то оставшиеся от предыдущей задачи данные вызовут ошибку в работе канала т.к. будет обнаружено что произошла запись данных, которые не читались.

7.5.5 Каналы DMA общего назначения

Четыре канала DMA с 0 по 3 являются каналами общего назначения, т.к. обладают двумя регистрами TCB и позволяют программировать источник и приемник данных. Регистры TCB каналов определяют следующее:

- Канал включен, если оба поля TY в TCB DP не равны нулю.
- Приоритет канала высокий, если бит PR установлен в одном из регистров TCB DP.
- Прерывание DMA включено, если бит INT установлен в одном из регистров TCB DP.
- Режим запроса DMA включен, если бит DRQ установлен в одном из регистров TCB DP.
- Поле LEN должно быть одинаковым в обоих регистрах.
- Бит CHEN должен быть одинаковым в обоих регистрах.
- Передача внутренней и внешней памяти устанавливается в поле TY регистров TCB DP. Это позволяет эффективно передавать данные между внутренней памятью процессора и внешней памятью или устройством.

– Поле CHTG не имеет значения для этих каналов. В случае цепочки каждый регистр TCB загружается новое значение в свой канал в соответствии с его полем CHPT.

Передача DMA между внутренней памятью процессора и внешней памятью требует от одного канала DMA генерации адресов для обоих типов памяти. Внешний адрес генерируется в соответствии с данными в регистре TCB внешней памяти. Адрес внутренней памяти генерируется в соответствии с данными в регистре TCB внутренней памяти. Если бит 2D в TCB DP установлен в одном из регистров TCB, то адрес этого TCB также обновляется в соответствии со значением модификатора в регистре DY. Передачи внутри внешней памяти и внутренней памяти поддерживаются. Также поддерживается передача между внешней памятью и внешними устройствами.

Поле TY в регистре DP указывает тип передачи. В таблице 55 приводилось описание допустимых комбинаций типов в одном канале. Каждый канал с 0 по 3 имеет регистр TCB передатчика и приемника, где передатчик считывает данные из памяти и передает их приемнику, а приемник отсылает данные в память.

Следует отметить, что большинство параметров TCB не имеют отношения к типу памяти, а являются описателями блока памяти (который может размещаться в любом месте) или описывают поведение канала во время работы и при завершении обмена.

Параметры, описывающие блок памяти:

– регистры DX и DY задают размеры блока и расположение элементов блока в памяти;
– бит 2D определяет, как рассматривается блок данных: как линейный или как двухмерный;
– биты LEN определяют длину операнда блока.

Параметры, описывающие поведение канала во время работы и при завершении обмена:

– бит PR указывает, что канал имеет высокий приоритет;
– бит DRQ определяет активацию транзакции по внешнему запросу ил без него;
– бит INT разрешает формирование запроса прерывания к процессору по окончании обмена;
– бит CHEN разрешает цепочку операций;
– поле CHPT хранит адрес цепочки;
– поле CHTG определяет приемник цепочки.

Все эти параметры не зависят от типа обмена. Поэтому при рассмотрении различных вариантов обмена будем уделять внимание только тем параметрам, которые зависят от выбранного типа памяти.

Рассмотрим различные варианты обменов для каналов 0-3.

7.5.5.1 Передача из внешней памяти во внутреннюю память

Для передачи данных из внешней памяти во внутреннюю необходимо запрограммировать TCB передатчика для чтения внешней памяти, а TCB приемника для записи во внутреннюю память. TCB передатчика программируется следующим образом:

– Регистр DI

  Инициализируется начальным адресом блока данных во внешней памяти.

– Регистр DP

Поле TY должно быть равно 4 или 7, что указывает на внешнюю память. Выбор между типом 4 и 7 влияет только на длину полей DXC, DXM, DYC, DYM.

ТСВ приемника данных описывается аналогичным образом со следующими особенностями, связанными с выбором внутренней памяти:

- Регистр DP
  Поле TY должно быть равно 2 или 3, что указывает на внутреннюю память. Выбор между типом 2 и 3 влияет только на длину полей DXC, DXM, DYC, DYM приемника.
- Регистр DI
  Инициализируется начальным адресом блока данных во внутренней памяти.

Все другие параметры ТСВ для передатчика и приемника описывают блоки памяти и поведение канала. Параметры, описанные выше, имеют ограничения, основными из которых являются:

- поле LEN должно быть одинаковым в обоих ТСВ;
- бит CHEN должен быть одинаковым в обоих ТСВ.

Выбор между типами обменов 2 и 3 или 4 и 7 зависит от размера величины модификации адреса. Если при переходе от одного элемента к другому нам нужно изменить адрес на величину более чем 32768 (в положительную сторону или отрицательную), то нам недостаточно 16 бит поля DXM для хранения такого значения. Поэтому мы вынуждены использовать тип обмена 3 или 7 место 2 или 4, чтобы поле модификации было 22 бита. В этом случае, уменьшается размер поля количества.

7.5.5.2 Передача из внутренней памяти во внешнюю память

В случае передачи из внутренней памяти во внешнюю память нет никаких особенностей. По сравнению с режимом, описанным в подразделе «Передача из внешней памяти во внутреннюю память», ТСВ передатчика и приемника просто меняются местами.

ТСВ передатчика:

- Регистр DP
  Поле TY должно быть равно 2 или 3, что указывает на внутреннюю память. Выбор между типом 2 и 3 влияет только на длину полей DXC, DXM, DYC, DYM приемника.
- Регистр DI
  Инициализируется начальным адресом блока данных во внутренней памяти.

ТСВ приемника:

- Регистр DP
  Поле TY должно быть равно 4 или 7, что указывает на внешнюю память. Выбор между типом 4 и 7 влияет только на длину полей DXC, DXM, DYC, DYM приемника.
- Регистр DI
  Инициализируется начальным адресом блока данных во внешней памяти.

7.5.5.3 Передача из внешней памяти во внешнюю память

Подобный тип передачи выполняется в случае, когда в ТСВ приемника и передатчика поля типа TY равны 4 или 7. Адреса в регистрах DI должны указывать на блоки данных во внешней памяти.

7.5.5.4 Передача из внутренней памяти во внутреннюю память

Данный тип передачи программируется следующим образом.

ТСВ передатчика:

- Регистр DP
  Поле TY должно быть равно 2 или 3, что указывает на внутреннюю память. Выбор между типом 2 и 3 влияет только на длину полей DXC, DXM, DYC, DYM приемника.
- Регистр DI
  Инициализируется начальным адресом блока данных во внутренней памяти.

ТСВ приемника:

- Регистр DP
  Поле TY должно быть равно 4 или 7, что указывает на внешнюю память. Выбор между типом 4 и 7 влияет только на длину полей DXC, DXM, DYC, DYM приемника.
- Регистр DI
  Инициализируется начальным адресом блока данных внутренней памяти, но в мультипроцессорном адресном пространстве.

Таким образом данный тип передач напоминает передачу из внутренней во внешнюю память, но в качестве внешней памяти указывается адрес внутренней памяти того же процессора, но в мультипроцессорном адресном пространстве. В этом случае канал DMA определяет, что возможна передача без задействования кластерной шины и выполняет внутреннюю пересылку. Необходимо отметить ограничения такой пересылки:

- Контроллер не проверяет: относится ли адрес внутренней памяти мультипроцессорного пространства к памяти общего назначения или регистрам. Контроллер всегда рассматривает этот адрес как адрес памяти.
- Невозможно выполнить пересылку из внутренней памяти в регистры ядра или регистры периферии, используя подобный тип передачи.

Пересылки во внутренней памяти с помощью DMA выполняются как минимум вдвое медленнее, чем такие же пересылки, выполненные ядром процессора. Однако полезным может быть тот факт, что пересылка выполняется в фоновом режиме и процессор может выполнять в это время другую работу.

7.5.5.5 Передача из внешнего устройства ввода/вывода во внешнюю память (FLYBY)

Источником данных является внешнее устройство, поэтому ТСВ передатчика имеет следующие особенности:

- Передача может инициироваться только внешним запросом nDMARx поэтому бит DRQ в регистре DP должен быть 1.
- Источником данных является устройство, которое на внешней шине не имеет адреса и использует специальные внешние выводы для управления, поэтому поле адреса источника безразлично и поле модификатора DXM и DYM должно быть равно нулю.

- Бит 2D должен быть равен нулю.
- Количество данных, считываемых из устройства, определяется полем DXC.
- Поле LEN должно указывать на слово или двойное слово. Это не может быть 128-битовое слово, т.к. разрядность внешней шины не более 64 бит.
- Поле TY равно 5. Указывает на режим FLYBY.

Приемником данных является внешняя память, поэтому TCB приемника канала описывается стандартным образом, как и в предыдущих передачах. Имеется только одна особенность: регистр адреса DI должен указывать на внешнюю динамическую память SDRAM. Только с ней возможен режим FLYBY.

Особенность FLYBY передачи состоит в том, что канал DMA по внешнему запросу от устройства, сразу инициирует транзакцию записи во внешнюю память, хотя данных для записи у него пока нет. Когда транзакция записи появляется на внешней мультипроцессорной шине, интерфейс внешнего порта информирует внешнее устройство, что оно может выставить на шину данных свои данные. Эти данные и используются в цикле записи во внешнюю память.

Особенностью данного режима является прямая пересылка между внешним устройством и внешней памятью с минимальным участием процессора.

7.5.5.6 Передача из внешней памяти во внешнее устройство ввода/вывода (FLYBY)

Выше был рассмотрен случай передачи данных из внешнего устройства во внешнюю память. Возможна и обратная передача, когда внешнее устройство выступает в роли приемника данных.

В этом случае источником данных выступает внешняя динамическая память и TCB передатчика может быть описано стандартным образом. TCB приемника описывает устройство ввода-вывода и имеет особенности:

- Передача может инициироваться только внешним запросом nDMARx, поэтому бит DRQ в регистре DP должен быть 1.
- Приемником данных является устройство на внешней шине, которое не имеет адреса и которое использует специальные внешние выводы для управления, поэтому поле адреса DI приемника безразлично и поле модификатора DXM и DYM должно быть равно нулю.
- Бит 2D должен быть равен нулю.
- Количество данных, записываемых в устройство, определяется полем DXC.
- Поле LEN должно указывать на слово или двойное слово. Оно должно быть равно полю LEN в TCB передатчика. Это не может быть 128-битовое слово, т.к. разрядность внешней шины не более 64 бит.
- Поле TY равно 5. Указывает на режим FLYBY.

Особенность данной FLYBY передачи состоит в том, что канал DMA по внешнему запросу от устройства инициирует транзакцию чтения внешней память. Когда транзакция чтения появляется на внешней мультипроцессорной шине и внешняя память выдает данные на шину, интерфейс внешнего порта информирует внешнее устройство, что оно может принять данные с шины. На этом цикл чтения заканчивается. Процессор не пересылает прочитанные данные в DMA контроллер.

7.5.5.7 Семафоры DMA

Каналы DMA с 0 по 3-й являются каналами общего назначения и могут использоваться различными ведущими устройствами или процессами. Устройство, которое хочет использовать канал DMA должно определить, доступен ли он в текущий момент, т.к. каналы DMA являются системным ресурсом. Рекомендовано делать это программным методом, используя согласованную область в памяти

каждого процессора для определения доступности каналов. Эта область памяти должна меняться только при использовании операции «чтение-модификация-запись».

7.5.5.8 Режим квитирования

Каждый из четырех каналов DMA могут запускаться внешним запросом – nDMAR0, nDMAR1, nDMAR2 и nDMAR3.

Доступ запрашивается, когда внешние устройства выставляет низкий уровень на nDMARx. Фронт внешнего сигнала фиксируется процессором и синхронизируется с тактовой частотой SOC-шины процессора.

Внешнее устройство не обязано ждать действительного прохождения транзакции на шине, прежде чем сделать следующий запрос по линии nDMARx. Запросы хранятся во внутреннем счетчике канала. Счетчик хранит до 15 запросов, так что внешнее устройство может создавать до 15 запросов, прежде чем первый будет обработан.

Более 15 запросов без передачи DMA могут вызвать непредсказуемые результаты. Если канал DMA отключен, то соответствующий сигнал nDMARx игнорируется.

7.5.6 DMA порта связи

Четыре канала DMA определяют передачи от портов связи к внутренней памяти, внешней памяти или другим портам связи. Другие четыре канала определяют передачи из внутренней памяти, внешней памяти и одних портов связи к другим портам связи.

Особенностью каналов DMA портов связи является то, что все они имеют только один регистр TCB. Для каналов с 4 по 7 это TCB передатчика, который читает данные из внешней или внутренней памяти и отправляет данные в передатчик порта связи. Для каналов с 8 по 11 это TCB приемника, который берет данные из приемника порта связи и пересылает их во внутреннюю или внешнюю память.

Как и в случае каналов общего назначения, большинство параметров TCB не имеют отношения к порту связи и описывают только блок данных либо поведение канала во время работы или при завершении передачи. Поле TY регистра TCB указывает на тип передачи.

7.5.6.1 Порт связи во внутреннюю \ внешнюю память

Для переноса данных от портов связи во внутреннюю или внешнюю память должны использоваться только каналы с 8 по 11. Приемник TCB этих каналов должен быть запрограммирован соответствующим образом.

Для передачи во внутреннюю память:

- Регистр DP
  Поле TY должно быть равно 2 или 3, что указывает на внутреннюю память.
- Регистр DI
  Инициализируется начальным адресом блока данных во внутренней памяти, куда будут пересылаться данные из порта связи.

Для передачи во внешнюю память

- Регистр DP
  Поле TY должно быть равно 4 или 7, что указывает на внешнюю память.
- Регистр DI
  Инициализируется начальным адресом блока данных во внешней памяти, куда будут пересылаться данные из порта связи.

Передача всегда инициируется приемником канала связи. Как только буфер приемника загружается принятыми данными, посылается запрос к каналу DMA, и он выполняет пересылку данных в блок данных описываемый регистром TCB.

Каналы DMA портов связи имеют одну особенность при организации цепочки операций: если в универсальном канале новое значение TCB может быть загружено только в регистр того же канала, то для каналов порта связи можно указать номер канала-приемника, следующего TCB. Правда номер канала-приемника лежит в диапазоне от 4 до 11. Поэтому после окончания работы один канал может инициировать работу другого канала.

7.5.6.2 Внутренняя/внешняя память – порт связи

Для пересылки данных из внутренней или внешней памяти в порты связи должны использоваться только каналы с 4 по 7. Передатчик TCB этих каналов должен быть запрограммирован соответствующим образом.

Для передачи из внутренней памяти:

- Регистр DP
  Поле TY должно быть равно 2 или 3, что указывает на внутреннюю память.
- Регистр DI
  Инициализируется начальным адресом блока данных во внутренней памяти, откуда будут читаться данные для порта связи.

Для передачи из внешней памяти

- Регистр DP
  Поле TY должно быть равно 4 или 7, что указывает на внешнюю память.
- Регистр DI
  Инициализируется начальным адресом блока данных во внешней памяти, откуда будут читаться данные для порта связи

Передача всегда инициируется передатчиком порта связи. Как только буфер передатчика становится пуст – посылается запрос к каналу DMA, и он выполняет пересылку данных из блока данных, описываемого регистром TCB, в буфер передатчика порта связи.

7.5.6.3 Пересылка между портами связи

Контроллер DMA поддерживает возможность пересылки данных между портами связи без использования промежуточных буферов в памяти. Понятно, что передача может идти от приемника порта связи в передатчик порта связи. Поскольку инициатором передачи в данном случае выступает приемник порта связи, то такие передачи поддерживаются только каналами DMA с 8 по 11. Отличием такой пересылки от пересылки в память является необходимость дополнительного контроля готовности передатчика канала связи принять данные от приемника. Особенности программирования TCB приемника каналов с 8 по 11состояит в следующем:

- Регистр DP.
  Поле TY должно быть равно 1, что указывает на порт связи.
- Регистр DI.
  Инициализируется адресом передатчика порта связи, который будет принимать данные. Адрес 0x1F04A0 для порта связи 0, 0x1F04A8 для порта связи 1, 0x1F04B0 для порта связи 2, 0x1F04B8 для порта связи 3. На самом деле в данном адресе важны только биты 3 и 4. С их помощью кодируется номер передатчика порта связи. Все другие биты безразличны.
- Модификатор DXM должен быть сброшен.

Передача всегда инициируется готовностью приемника канала связи и готовностью передатчика канала связи. Как только буфер приемника загружается принятыми данными, посылается запрос к каналу DMA. Канал DMA анализирует состояние буфера передатчика, в который он запрограммирован записать данные. Если буфер передатчика канала связи пуст – происходит пересылка.

7.5.7 Каналы autoDMA

Два канала autoDMA с номерами 12 и 13 поддерживают возможность пересылки данных от внешнего ведущего устройства во внутреннюю память процессора. Блок данных в TCB приемника описывается стандартным образом. Параметры работы канала и завершения работы канала определяются стандартным образом. Единственным ограничением в TCB является то, что тип обмена TY должен быть равен 2 или 3, т.е. приемник данных всегда внутренняя память.

Каждый канал autoDMA имеет буфер данных. Канал запрашивает выполнение транзакции, как только в буфере появляются данные. Размер буфера данных составляет четыре 128-разрядных слова. Требованием в работе канала является соответствие размера данных, записываемых в буфер, размеру операнда, который определяется полем LEN в регистре TCB.

Канал принимает данные в буфер, только когда он активен. Если происходит запись в буфер при неактивном канале – генерируется флаг ошибки.

7.5.8 Быстродействие DMA

Данная глава описывает общее быстродействие контроллера DMA при условии наличия нескольких активных каналов, пытающихся одновременно получить доступ к внешней \ внутренней памяти.

7.5.8.1 DMA внутренней памяти

Каналы DMA получают доступ к внутренней памяти согласно своим приоритетам. В целом потеря производительности при переключении между каналами отсутствует. Любая комбинация портов связи и других портов для передачи имеет одинаковую максимальную скорость внутренней передачи, составляющую одну передачу DMA на такт.

С точки зрения быстродействия каналы связи DMA (с 4 по 11) имеют следующие особенности:

- Каналы с 4 по 7 при работе с передатчиками портов связи максимально загружают шины процессора в случае, когда передатчики работают на максимальной частоте и при ширине шины в 4 бита. В этом случае передача одного 128-разрядного слова (без контроля) происходит за 16 тактов частоты процессора (8 тактов SOC-шины). Поэтому каналу DMA необходимо за 8 тактов SOC-шины успеть прочитать новые данные из внутренней памяти. Если при чтении не будет конфликтов доступа, канал DMA может сделать чтение за 7 тактов. При этом если работают все 4 передатчика, контроллер DMA будет генерировать транзакции каждые 4 такта из 8-ми.
- Каналы с 8 по11 при работе с приемниками портов связи максимально загружают шины процессора в случае, когда приемники работают на максимальной частоте и при ширине шины в 4 бита. В этом случае прием одного 128-разрядного слова (без контроля) происходит за 16 тактов частоты процессора (8 тактов SOC-шины). Поэтому каналу DMA необходимо за 8 тактов SOC-шины успеть записать новые данные из приемника во внутреннюю память. Если при записи не будет конфликтов доступа, то канал DMA может это сделать за 2 такта (такт на анализ

запроса от приемника и такт на арбитраж доступа). При этом если работают все 4 приемника, то контроллер DMA будет генерировать транзакции каждые 4 такта из 8-ми.

Таким образом, если включить все 8 каналов портов связи на максимальной частоте, то мы будем иметь запрос к внутренней памяти со стороны DMA каждый такт SOC шины. Можно сказать, что каналы с 0 по 3 не смогут получить доступ к внутренней памяти в это время. Если каналы 12 и 13 будут иметь приоритет выше чем каналы связи, то они будут притормаживать обмен по каналам связи. Если же каналы 12 и 13 будут иметь приоритет ниже, то возможна ошибка в их работе, т.к. они не смогут получить доступ к внутренней памяти и тогда их буфер переполнится. Если порты связи будут работать не на максимальной частоте, а вдвое меньшей, то для доступа к внутренней памяти остается 50 % времени.

На скорость обмена каналов DMA с внутренней памятью влияет также и работа процессора. Если процессор часто обращается на SOC-шину для чтения данных с периферийных регистров, то он может притормаживать DMA, если цикл чтения попадает на запрос от DMA. Тогда при доступе к SOC IFIFO процессор будет иметь более высокий приоритет, и контроллер DMA будет притормаживать.

С точки зрения быстродействия каналы общего назначения (с 0 по 3) имеют следующие особенности:

- Транзакции каналов разделены на две части. В первой части выполняется чтение данных из памяти во внутренний буфер DMA. Во второй части идет запись данных из буфера в память.
- Четыре канала имеют общий буфер размером восемь 128-разрядных слов.
- Каналы могут запускать до 8-ми операций чтения, не дожидаясь выполнения операций записи.

Возможность генерации сразу восьми транзакций чтения одним каналом значительно увеличивает скорость обмена. Если бы вторая транзакция начиналась только после выполнения фазы записи первой транзакции, скорость работы канала была бы практически в 10 раз медленнее. Вместе с тем при работе нескольких каналов общего назначения самый приоритетный канал практически полностью захватывает трафик контроллера, и следующий канал может начать передачу только после полного завершения обмена более высокоприоритетного канала. Поэтому может оказаться, что цепочка из 4-х операций на одном канале DMA может выполниться с той же скоростью, как и 4 обмена одновременно на разных каналах. Это справедливо для случая пересылок между внутренней и внешней памятью, когда бит DRQ равен нулю и канал не ждет внешнего запроса.

7.5.8.2 DMA внешней памяти

Работа с внешней памятью имеет особенность, связанную с тем, что доступ к этой памяти намного медленнее, чем к внутренней. Если доступ к внутренней памяти для контроллера DMA может происходить каждый такт SOC-шины, то доступ к внешней памяти будет притормаживаться из-за более низкой частоты внешней шины. В лучшем случае (например, при доступе к SDRAM) это может быть один такт внешней шины.

Контроллер DMA может отсылать каждый такт SOC шины запрос к внешней памяти. Эти запросы попадают в OFIFO внешнего интерфейса. Буфер этого интерфейса имеет ограниченный размер – не более 8 записей. При этом запись в буфер идет со скоростью SOCCLK, а чтение со скоростью SCLK. Обычно частота внешней шины как минимум в 2 раза ниже частоты SOC шины. Поэтому заполнив буфер за 8 тактов, контроллер DMA будет ожидать наличие свободного места в

буфере, и в дальнейшем скорость записи в буфер будет соответствовать скорости чтения. Скорость чтения зависит от многих факторов. Например, для SDRAM это:

- Наличие прав доступа к внешней шине (это нужно для любого типа памяти);
- Отсутствие регенерации памяти в момент запроса доступа;
- Отсутствие необходимости генерировать команду ACT;
- Длина запрашиваемого операнда и ширина шины данных внешней памяти.

По сути пока один запрос к памяти не обслужится, второй запрос не поступит на обработку. В процессоре имеется поддержка для выполнения более быстрого обмена с SDRAM. Контроллер внешней памяти отслеживает ситуацию последовательного доступа к одной странице динамической памяти и поддерживает конвейерный доступ в пределах одной страницы. Это позволяет значительно ускорить обмен. Этому способствует и тот фактор, что каналы с 0 по 3-й могут формировать до 8-ми последовательных запросов. Для динамической памяти очень неприятным моментом являются частые доступы в различные страницы памяти т.к. это вызывает необходимость часто выполнять подзаряд текущей страницы и заново активировать следующую страницу.

Дополнительная сложность возникает при работе в мультипроцессорной системе. Если процессор захватил шину для обмена, для него очень важно иметь блок запросов к внешней памяти т.к. в этом случае он может выполнить все их вместе, не освобождая шину после одного запроса. Здесь возможен выигрыш в скорости из-за отсутствия необходимости часто выполнять арбитраж доступа.

Каналы связи DMA также могут сохранять свои данные во внешней памяти или брать данные из внешней памяти. В принципе, если работает только один канал (на прием или передачу) то при работе с внешней динамической памятью есть вероятность поддержки высокой скорости обмена. При последовательном доступе к памяти и при ширине шины в 64 бита можно получить скорость данных близкую к максимальной скорости канала связи. Однако если работает несколько каналов, то эффективность обмена будет снижаться из-за непоследовательного доступа к памяти. В случае каналов связи более скоростным вариантом может оказаться прием данных из порта связи в буфер внутренней памяти, а затем пересылка этого буфера во внешнюю память, чем прямая пересылка из канала связи во внешнюю память.

7.6 Порты связи (LVDS)

Процессор содержит четыре полнодуплексных порта связи, каждый из которых обеспечивает дополнительные возможности ввода-вывода данных: четырёхбитный прием и четырехбитную передачу, используя технологию LVDS.

Благодаря способности работать на двойной скорости передачи данных – данные в защелке на обоих фронтах тактового сигнала – с частотой до 500 МГц, каждый порт связи может поддерживать передачу до 500 Мбайт в секунду в каждом направлении для комбинированной максимальной пропускной способности 4 Гбайт/с.

Порты связи обеспечивают дополнительный коммуникационный канал, который используется в мультипроцессорных системах для осуществления двухточечной межпроцессорной связи. Приложения также могут использовать порт для загрузки.

Управление передачей ядром процессора выполняется посредством механизмов прерывания и опроса. Управление передачей контроллером DMA выполняется через назначенные DMA-каналы приема/передачи. При этом все каналы DMA портов поддерживают цепочки операций, а также могут быть использованы при загрузке процессора во время старта.

7.6.1 Архитектура портов связи

Как показано ниже (Рисунок 48), порты связи подключены к SOC-шине как периферийное устройство.

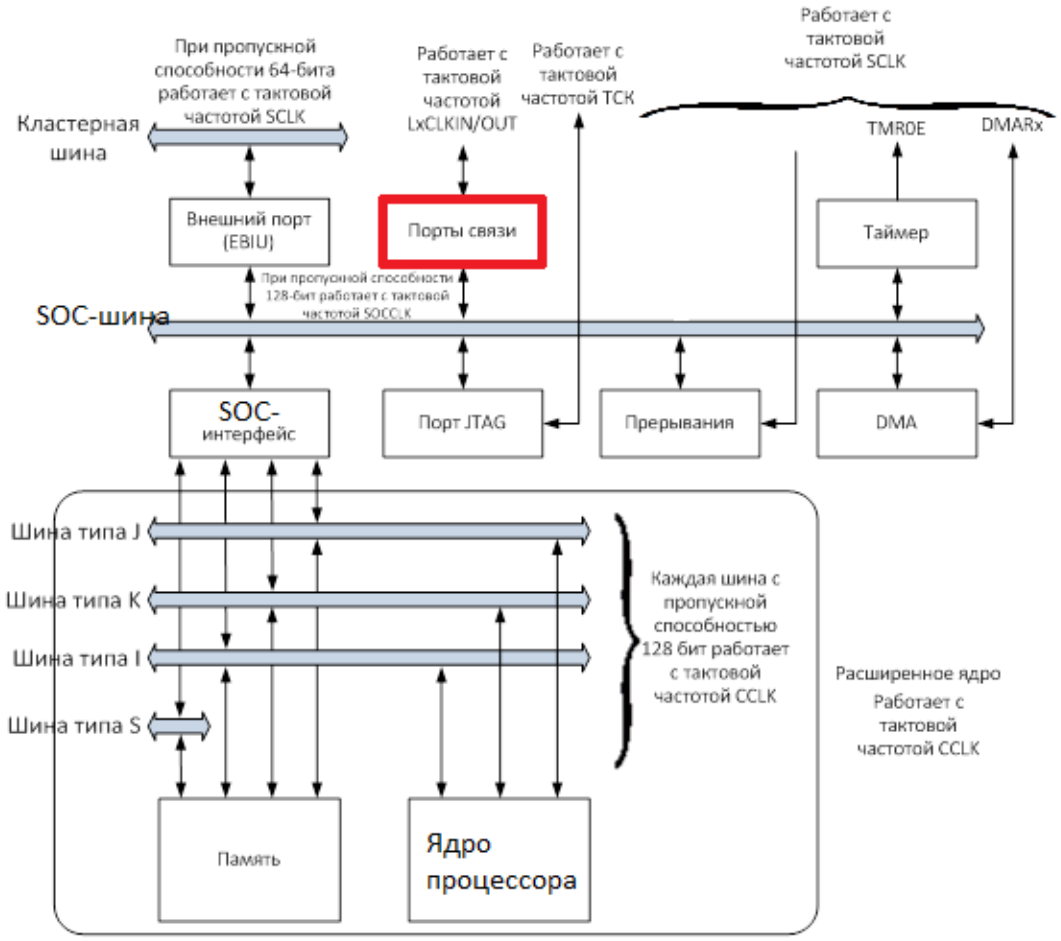


Рисунок 48 – Подключение портов связи на кристалле

Порт связи состоит из двух частей – передатчик и приемник. Каналы передачи и приема имеют тройной буфер, как показано ниже (Рисунок 49). Буферные регистры, подключенные к SOC-шине, доступны программисту как регистры LBUFTX и LBUFRX. Они являются 128-разрядными, отображаемыми в памяти универсальными регистрами. Регистры дополнительных буферов и сдвиговые регистры являются программно недоступными.

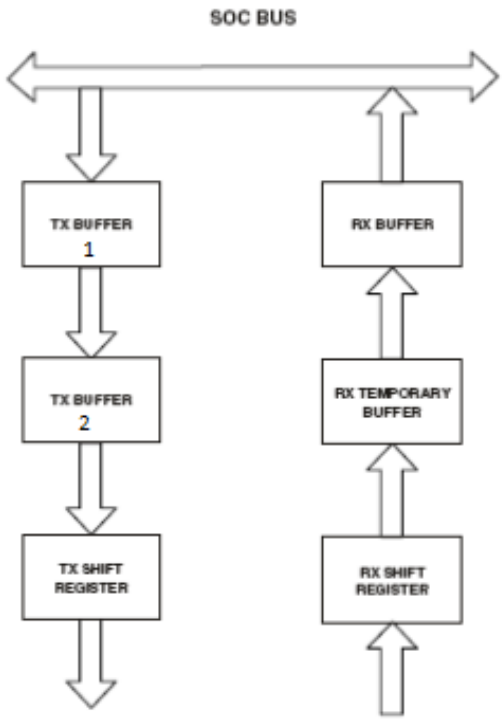


Рисунок 49 – Архитектура порта связи

Дополнительная буферизация позволяет конвейеризировать операции приема-передачи порта связи и операции контроллера DMA. Также имеет значение, что сдвиговые регистры приема-передачи имеют собственную частоту работы, которая отличается от частоты SOC-шины.

7.6.2 Внешние выводы портов связи

Для выполнения обмена данными с внешними устройствами порты связи используют внешние выводы микросхемы, которые описаны ниже (Таблица 100). В таблице приведен перечень выводов одного порта связи. Знак ‘x’ в имени сигнала указывает на порт связи – 0, 1, 2 или 3. В набор внешних выводов порта связи входят как дифференциальные выводы, так и обычные выводы. Для каждого дифференциального вывода существует пара линий: P и N. На линии P передается истинное значение сигнала, а на линии N противоположное. Раздельные шины данных для приема и для передачи имеют разрядность 4 бита каждая. При этом имеется режим передачи с использованием только одного разряда шины данных из четырех.

Таблица 100 – Описание выводов – порты связи

Сигнал	Описание
LxDATO3–0P	Шина данных 3–0 передача LVDS P
LxDATO3–0N	Шина данных 3–0 передача LVDS N
LxCLKOUTP	тактовый генератор передачи LVDS P
LxCLKOUTN	тактовый генератор передачи LVDS N

Сигнал	Описание
LxACKI	входной сигнал подтверждения передатчика. Используя этот сигнал, приемник указывает передатчику, что можно продолжать передачу
LxBCMPO	Завершение блока. Когда передача выполняется с использованием DMA, данный сигнал указывает приемнику, что передаваемый блок закончился. Во время сброса выводы nL1BCMPO, nL2BCMPO и nL3BCMPO используются как входы для конфигурирования процессора. Перед началом инициализации порта, порт связи выдает сигнал высокого уровня на LxBCMPO (деактивирован), указывая приемнику, что порт подсоединен
LxDATI3–0P	данные 3–0 прием LVDS P
LxDATI3–0N	данные 3–0 прием LVDS N
LxCLKINP	тактовый генератор приема LVDS P
LxCLKINN	тактовый генератор приема LVDS N
LxACKO	выходной сигнал подтверждения приемника. Используя этот сигнал, приемник указывает передатчику, что можно продолжать передачу.
nLxBCMPI	Завершение блока. Когда передача выполняется с использованием DMA, данный сигнал указывает приемнику, что передаваемый блок закончился. После сброса сигнал высокого уровня на nLxBCMPI указывает на то, что к приемнику порта подключен внешний передатчик

Каждый порт связи имеет два независимых канала, один для приема и второй для передачи, которые могу работать одновременно. Канал передачи передает данные на другое устройство, а канал приема получает данные с другого устройства. Каждый канал осуществляет обмен данными, используя до четырех битов данных и используя сигналы LxCLKOUTP/N, LxACKI, LxCLKINP/N и LxACKO для управления передачей данных.

Сигналы nLxBCMPI и nLxBCMPO используются для информирования о том, что текущая передача блока данных завершена. Порты связи должны соединяться так, как это показано ниже (Рисунок 50, Рисунок 51).

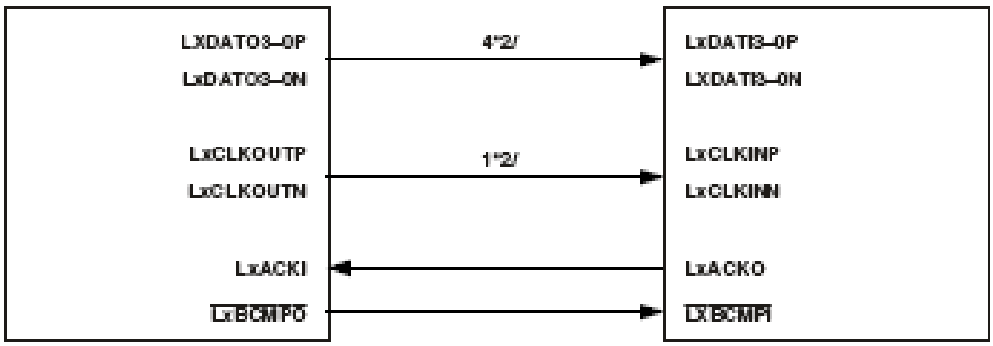


Рисунок 50 – Конфигурация порта связи (4-разрядный режим)

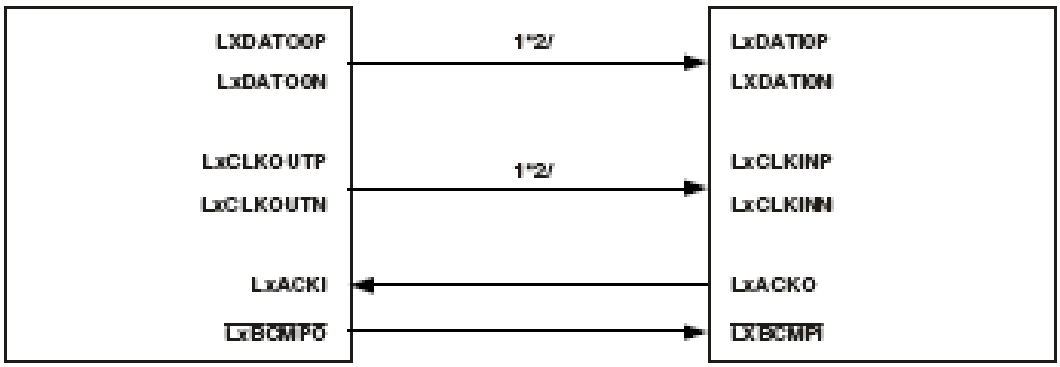


Рисунок 51 – Конфигурация порта связи (1-разрядный режим)

Порты связи процессора являются тактируемыми высокоскоростными LVDS портами данных. LVDS является стандартом для дифференциальной передачи сигналов между удаленными элементами. LVDS обеспечивает более высокую частоту, более высокий уровень устойчивости к шумам, более низкое энергопотребление и меньшее число электромагнитных помех.

Передача сигналов LVDS требует дифференциального завершения. Ниже показан процесс передачи через порт от одного процессора к другому (Рисунок 52). Внутренние 100 Ом резисторы завершения (RT) на стороне приемника обеспечивают выполнение данного требования. Линии на плате должны быть по возможности одинаковыми для того, чтобы поддерживать одинаковое время задержки для всех выводов данных и тактовых сигналов.

Линии несимметричных сигналов (LxACKI, LxACKO, nLxBCMPI и nLxBCMPO) не так критичны, но их задержки должны были близки к задержкам соответствующих им дифференциальных сигналов.

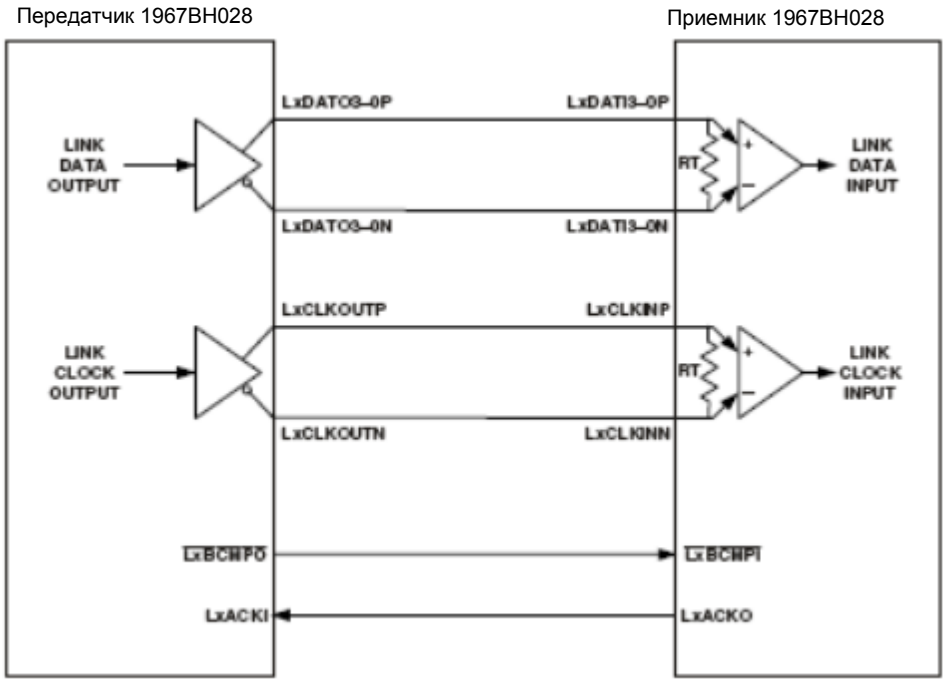


Рисунок 52 – Конфигурация передачи между процессорами

7.6.3 Прием и передача данных

Обмен данными осуществляется посредством записи в буфер передачи и чтением из буфера приема. Длина передаваемых данных всегда равна 128 битам. Все данные, записанные в буфер передачи, копируются в сдвиговый регистр, как только он становится пустым, и после этого передаются. Приемник разрешает продолжение приема данных, только когда его сдвиговый регистр пуст или, когда в его буферных регистрах есть достаточно места для приема данных из сдвигового регистра, когда прием этого счетверенного слова будет завершен. После того, как целое счетверенное слово получено, приемник перемещает данные из сдвигового регистра в буфер приема, когда он свободен.

Если приемник не готов принять следующее слово данных, он использует линию подтверждения LxACK0 для приостановки работы передатчика.

Ниже приведены имена регистров, которые пользователь может использовать при программировании портов связи (Таблица 101, Таблица 102).

Таблица 101 – Регистры буферов передатчика и приемника порта связи

Регистр	Название	Адрес	Значение по умолчанию
LBUFTX0:0	Данные передатчика порта связи 0	0x 04A0	Не определено
LBUFTX0:1	Данные передатчика порта связи 0	0x 04A1	Не определено
LBUFTX0:2	Данные передатчика порта связи 0	0x 04A2	Не определено
LBUFTX0:3	Данные передатчика порта связи 0	0x 04A3	Не определено
LBUFRX0:0	Данные приемника порта связи 0. Только чтение	0x 04A4	Не определено
LBUFRX0:1	Данные приемника порта связи 0. Только чтение	0x 04A5	Не определено
LBUFRX0:2	Данные приемника порта связи 0. Только чтение	0x 04A6	Не определено
LBUFRX0:3	Данные приемника порта связи 0. Только чтение	0x 04A7	Не определено
LBUFTX1:0	Данные передатчика порта связи 1	0x 04A8	Не определено
LBUFTX1:1	Данные передатчика порта связи 1	0x 04A9	Не определено
LBUFTX1:2	Данные передатчика порта связи 1	0x 04AA	Не определено
LBUFTX1:3	Данные передатчика порта связи 1	0x 04AB	Не определено
LBUFRX1:0	Данные приемника порта связи 1. Только чтение	0x 04AC	Не определено
LBUFRX1:1	Данные приемника порта связи 1. Только чтение	0x 04AD	Не определено
LBUFRX1:2	Данные приемника порта связи 1. Только чтение	0x 04AE	Не определено
LBUFRX1:3	Данные приемника порта связи 1. Только чтение	0x 04AF	Не определено
LBUFTX2:0	Данные передатчика порта связи 2	0x 04B0	Не определено
LBUFTX2:1	Данные передатчика порта связи 2	0x 04B1	Не определено
LBUFTX2:2	Данные передатчика порта связи 2	0x 04B2	Не определено
LBUFTX2:3	Данные передатчика порта связи 2	0x 04B3	Не определено
LBUFRX2:0	Данные приемника порта связи 2. Только чтение	0x 04B4	Не определено
LBUFRX2:1	Данные приемника порта связи 2. Только чтение	0x 04B5	Не определено

LBUFRX2:2	Данные приемника порта связи 2. Только чтение	0x 04B6	Не определено
LBUFRX2:3	Данные приемника порта связи 2. Только чтение	0x 04B7	Не определено
LBUFTX3:0	Данные передатчика порта связи 3	0x 04B8	Не определено
LBUFTX3:1	Данные передатчика порта связи 3	0x 04B9	Не определено
LBUFTX3:2	Данные передатчика порта связи 3	0x 04BA	Не определено
LBUFTX3:3	Данные передатчика порта связи 3	0x 04BB	Не определено
LBUFRX3:0	Данные приемника порта связи 3. Только чтение	0x 04BC	Не определено
LBUFRX3:1	Данные приемника порта связи 3. Только чтение	0x 04BD	Не определено
LBUFRX3:2	Данные приемника порта связи 3. Только чтение	0x 04BE	Не определено
LBUFRX3:3	Данные приемника порта связи 3. Только чтение	0x 04BF	Не определено

Таблица 102 – Регистры управления и состояния портов связи

Регистр	Название	Адрес	Значение по умолчанию
LRCTL0	Регистр управления приемником порта связи 0	0x00E0	0x00000001
LRCTL1	Регистр управления приемником порта связи 1	0x00E1	0x00000001
LRCTL2	Регистр управления приемником порта связи 2	0x00E2	0x00000001
LRCTL3	Регистр управления приемником порта связи 3	0x00E3	0x00000001
LTCTL0	Регистр управления передатчиком порта связи 0	0x00E4	0x00000000
LTCTL1	Регистр управления передатчиком порта связи 1	0x00E5	0x00000000
LTCTL2	Регистр управления передатчиком порта связи 2	0x00E6	0x00000000
LTCTL3	Регистр управления передатчиком порта связи 3	0x00E7	0x00000000
	Зарезервированы	0x00E8-0x00EF	
LRSTAT0	Регистр состояния приемника порта связи 0	0x00F0	0x00000040
LRSTAT1	Регистр состояния приемника порта связи 1	0x00F1	0x00000040
LRSTAT2	Регистр состояния приемника порта связи 2	0x00F2	0x00000040
LRSTAT3	Регистр состояния приемника порта связи 3	0x00F3	0x00000040
LTSTAT0	Регистр состояния передатчика порта связи 0	0x00F4	0x00000002
LTSTAT1	Регистр состояния передатчика порта связи 1	0x00F5	0x00000002
LTSTAT2	Регистр состояния передатчика порта связи 2	0x00F6	0x00000002
LTSTAT3	Регистр состояния передатчика порта связи 3	0x00F7	0x00000002

LRSTATC0	Регистр сброса состояния приемника порта связи 0	0x00F8	Не определено
LRSTATC1	Регистр сброса состояния приемника порта связи 1	0x00F9	Не определено
LRSTATC2	Регистр сброса состояния приемника порта связи 2	0x00FA	Не определено
LRSTATC3	Регистр сброса состояния приемника порта связи 3	0x00FB	Не определено
LTSTATC0	Регистр сброса состояния передатчика порта связи 0	0x00FC	Не определено
LTSTATC1	Регистр сброса состояния передатчика порта связи 1	0x00FD	Не определено
LTSTATC2	Регистр сброса состояния передатчика порта связи 2	0x00FE	Не определено
LTSTATC3	Регистр сброса состояния передатчика порта связи 3	0x00FF	Не определено

7.6.4 Связь с DMA

Каждый порт связи соединяется с двумя каналами DMA. Один канал используется для передачи данных, в то время как второй используется для приема данных. Оба канала DMA связаны через интерфейс с внутренней памятью, внешней памятью или другими буферами портов связи. Передатчики портов связи с 0 по 3-й соединены соответственно с 4-го по 7-й каналами DMA. Приемники портов связи с 0 по 3-й соединены соответственно с 8-го по 11-й каналами DMA. Каналы DMA с 4-го по 7-й называются каналами передачи, т.к. передают данные передатчикам порта связи, а каналы с 8 по 11-й называются каналами приема т.к. принимают данные от приемников порта связи.

Передатчик порта связи формирует сервисный запрос к каналу передачи DMA, когда буферный регистр LBUFTX пуст и канал DMA включен. Приемник порта связи формирует запрос к каналу приема DMA, когда он записывает счетверенное слово данных в буферный регистр LBUFRX и канал DMA включен.

Канал-приемник DMA (с 8 по 11) может также использоваться для транзитных передач посредством записи данных из приемника своего порта связи в буферный регистр передатчика любого другого порта связи. Канал DMA при этом отслеживает, что буферный регистр передатчика свободен.

7.6.5 Завершение блочной передачи

Функция завершения передачи блока позволяет передатчику информировать приемник о том, что блочная передача завершена.

Поскольку канал передачи DMA точно знает, сколько слов необходимо передать и может отследить момент передачи последнего слова, то вместе с записью последнего слова в буфер передатчика, он формирует сигнал завершения передачи блока и передает его в передатчик вместе с данными. Порт связи активизирует nLxBCMPO, когда он передает последнее счетверенное слово и в регистре управления LTCTL установлен разряд BCMPE.

Когда вход nLxBCMPI активен, то принимающий порт связи передает эту информацию каналу DMA вместе с запросом. Это позволяет каналу приемнику DMA отследить завершение приема и закончить свою работу.

7.6.6 Прерывания портов связи

Порты связи имеют специализированные прерывания для управления потоком данных, когда порт связи осуществляет передачу с использованием ядра процессора (в отличие от передачи с использованием каналов DMA). Прерывание от приемника порта связи является активным только тогда, когда канал DMA для порта связи не включается. Порт приема активизирует прерывание, когда счетверенное слово, получаемое портом связи, ожидает в регистре LBUFRX. Прерывание приема порта связи является чувствительным по уровню и поэтому, если канал DMA, взаимодействующий с портом связи, становится активным после разрешения прерывания порта связи, то прерывание приема порта связи будет деактивировано и будет сформирован запрос к DMA.

7.6.7 Инициализация после сброса и загрузка через порт связи

Старт процессора может быть выполнен посредством загрузки начального кода программы через порты связи. В этом режиме порт работает как подчиненное устройство, настроенное на прием данных. После сброса все четыре канала DMA порта приема инициализируются для передачи блока из 256 слов (загрузка ядра) во внутреннюю память по адресам с 0 до 255 и настраиваются для формирования прерывания по окончании передачи блока (аналогично каналу 0 DMA). Соответствующие вектора прерываний для каналов DMA устанавливаются в нулевой адрес.

При сбросе или запрещении портов связи через регистры управления, содержимое буферов порта связи сбрасывается. Последовательность загрузки порта связи после сброса следующая:

- Приемник каждого порта связи включен, т.к. значение аппаратного сброса для регистра LRCTL разрешает прием порта связи (REN=1).
- На выводе nBMS должен быть высокий уровень сигнала, что означает отсутствие загрузки из внешнего EPROM. В этом случае процессор ждет прерывания от других источников.
- Когда внешнее устройство готово загрузить процессор через порт связи, оно деактивирует (равен 1) вывод nLxBCMPI для инициализации порта. После этого порт может принять 256 слов данных, загрузить их во внутреннюю память и сформировать запрос прерывания к процессору.

7.6.8 Программная инициализация

Приемник порта связи имеет флаг инициализации. Данный флаг устанавливается аппаратным способом после сброса процессора. Никаких других функций данный флаг не имеет. Он не оказывает никакого влияния на работу приемника.

Данный флаг может быть установлен или сброшен программным способом. Для этого необходимо выключить приемник и в бит 7 регистра управления записать значение, которое должно быть загружено во флаг инициализации. Дальше устанавливается, как минимум на один такт бит 6 регистра управления и затем сбрасывается (без изменения значения бита 7). Такая процедура приводит к записи во флаг инициализации требуемого значения.

7.6.9 Протокол передачи данных порта связи

Порт связи осуществляет передачу данных через 1- или 4-разрядную шину данных по каждому направлению (TX или RX), с использованием трех типов контрольных сигналов. В одноразрядном режиме для передачи используются выводы LxDATI0P/N и LxDATO0P/N.

Каждый порт связи имеет два независимых канала, которые могут работать одновременно. Канал передачи передает данные на внешнее устройство, канал приема получает данные с внешнего устройства. Каждый канал осуществляет передачу данных с использованием одного или четырех разрядов данных, используя сигналы LxCLKOUTP/N, LxACKI, LxCLKINP/N и LxACKO для управления передачей данных. Сигналы nLxBCMPI и nLxBCMPO используются для оповещения о завершении текущей передачи блока.

Существует несколько общих правил, которые применяются в протоколе порта связи (Рисунок 53 – Рисунок 61). Знание этих правил позволяет понять примеры, приведенные на диаграммах. Общие правила таковы:

- Первые данные (1-разрядные или 4-разрядные) передаются на нарастающем (из 0 в 1) фронте тактового сигнала порта связи (условно LXCLKOUTP).
- последние данные (1-разрядные или 4-разрядные) передаются на спадающем (из 1 в 0) фронте тактового сигнала порта связи (условно LXCLKOUTP).
- LXCLKOUTP переводится в низкий уровень, когда порт связи находится в состоянии простоя.

Минимальный объём передачи данных через порт равен счетверенному слову. Счетверенное слово может быть передано в течение 16 тактовых циклов, когда используются четыре разряда шины данных или в течение 64 циклов, когда используется только один разряд данных. Ниже (Рисунок 53 – Рисунок 56) показаны оба случая.

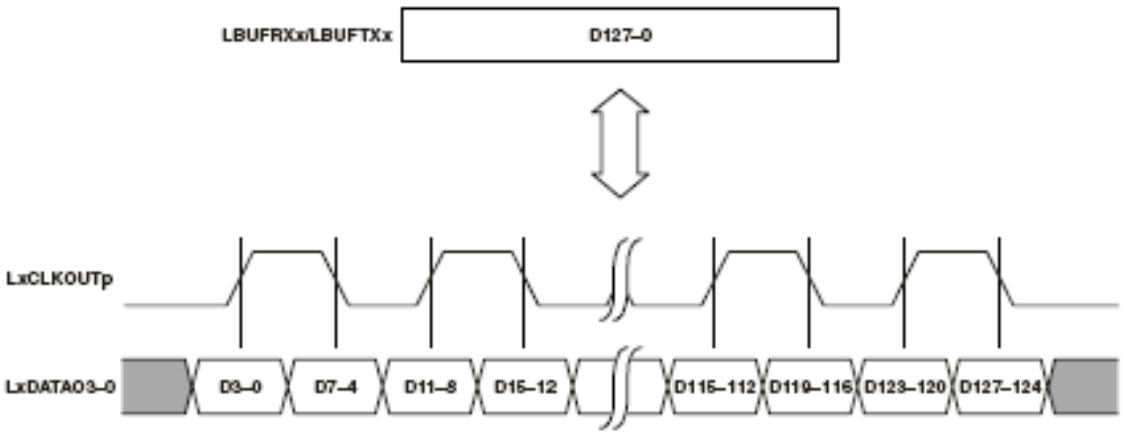


Рисунок 53 – Шина данных 4 разряда

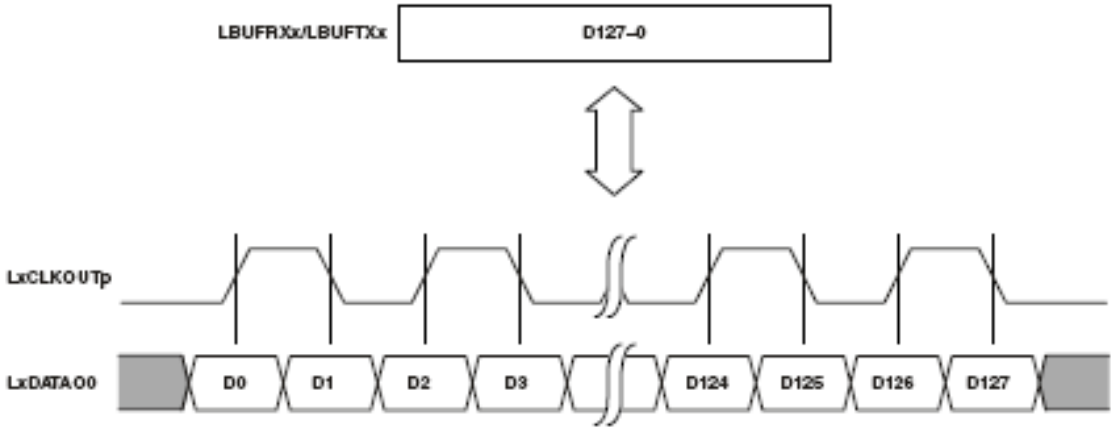


Рисунок 54 – Шина данных 1 разряд

Передача данных начинается, когда на LxACKI высокий уровень сигнала, что обозначает, что буфер приема свободен. Как показано ниже (Рисунок 55, Рисунок 56), первые биты данных достоверны до первого нарастающего фронта LxCLKOUTP и следующие за ними биты данных достоверны до спадающего фронта тактового сигнала.

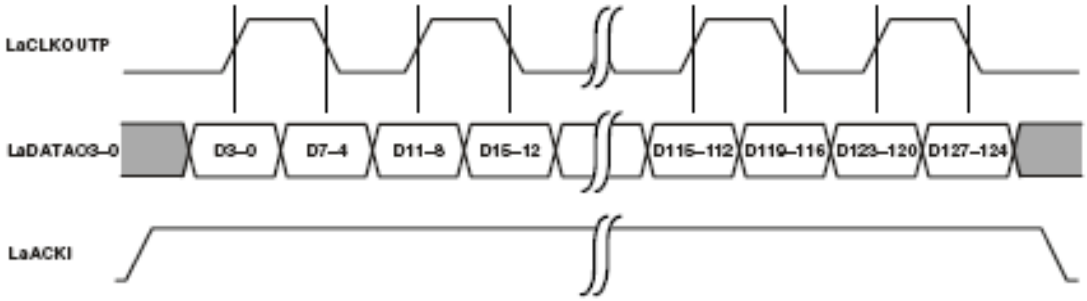


Рисунок 55 – Порт связи “a” передает одно счетверенное слово порту связи “b”

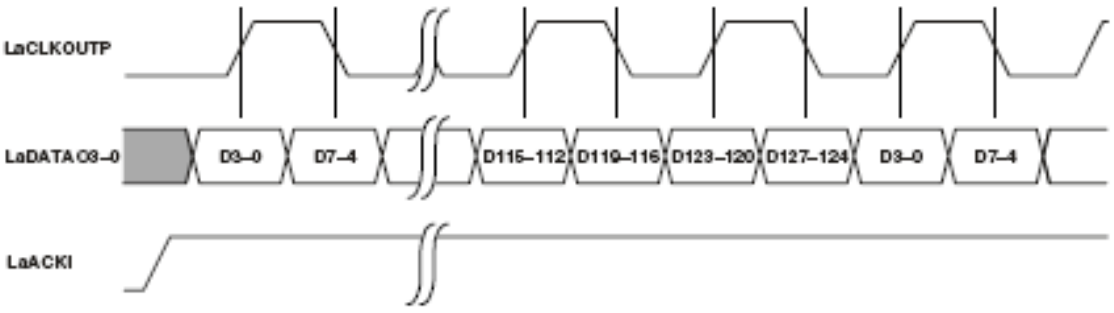


Рисунок 56 – Порт связи “a” передает слова порту связи “b” одно за другим

Рисунок 57 иллюстрирует состояние порта связи “b”, когда он не готов принимать больше данных. В этом случае после передачи текущего слова данных передатчик будет ждать пока сигнал подтверждения не примет высокий уровень.

На последовательность передачи влияет установка разрядов верификации данных. Это разряд RVERE в регистре LRCTL и разряд TVERE в регистре LTCTL. Если установлен разряд TVERE, то байт контрольной суммы отправляется после последнего байта в счетверенном слове. За байтом контрольной суммы всегда

следует «пустой» байт. Отправка байта контрольной суммы (биты Vx) и «пустого» байта (Xx) выполняется в течение двух циклов для шины 4-разрядных данных и 8 циклов для шины 1-разрядных данных. Рисунок 58 иллюстрирует последовательность передачи для случая 1-разрядной шины данных.

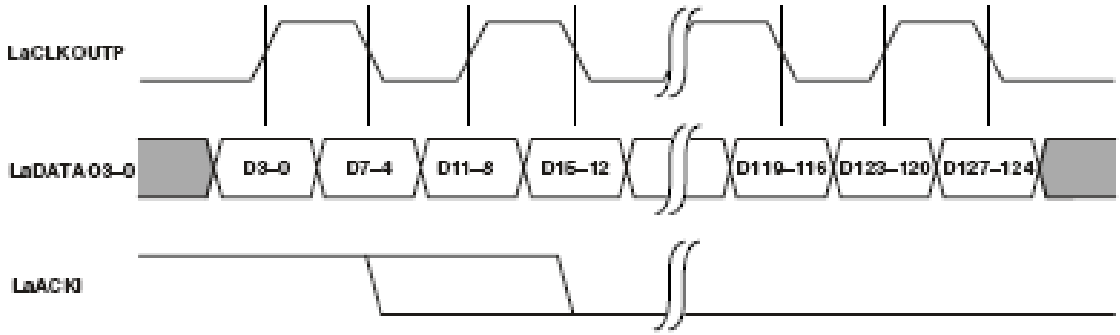


Рисунок 57 – Порт связи “a” передает порту связи “b” (приемник заполнен)

Сообщения между различными устройствами в системе обычно имеют различную длину. В некоторых случаях блоки данных также имеют различную длину. При этом знание длины блока не всегда доступно для принимающего устройства и не может быть передано передающим устройством. В этом случае на помощь приходят управляющие сигналы nLxBCMPO и nLxBCMPI.

Передающий порт связи указывает порту связи приема, что блок завершен с использованием выходного сигнала nLxBCMPO передатчика, который связан с входным сигналом приемника nLxBCMPI.

Когда приемник распознает этот сигнал, то он передает информацию каналу DMA о том, что блок данных завершен. В результате этого канал DMA игнорирует то, что его счетчик слов еще не достиг значения нуля, и работает так, как при завершении блока.

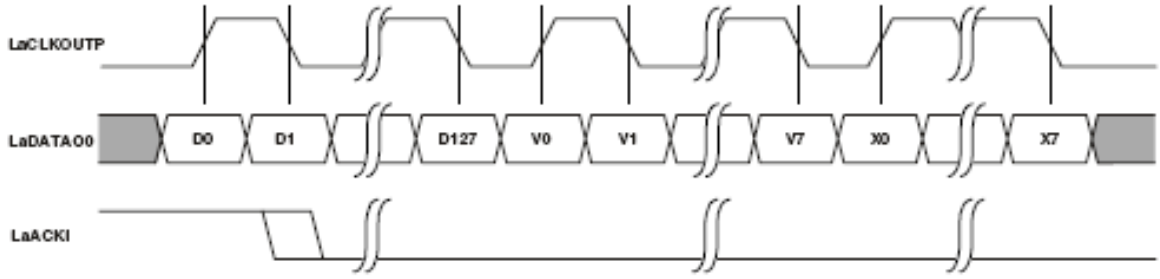


Рисунок 58 – Порт связи “a” передает порту связи “b” с верификацией

Сигнал nLxBCMPO указывает на завершение блока переходом на низкий уровень после первого нарастающего фронта LxCLKOUT в последнем счетверенном слове блока. На nLxBCMPO устанавливается высокий уровень сигнала (неактивный) перед последним спадающим фронтом сигнала LxCLKOUT того же счетверенного слова. Сигнал является неактивным, если разряд TBCMPE сбрасывается в регистр LTCTL или, когда передачи порта связи выполняются под управлением ядра процессора.

Рисунок 59 показывает, как порт связи “a” передает сигнал порту связи “b” о том, что текущий буфер завершен.

Рисунок 59 – Порт связи “a” передает порту связи “b” с завершением блока

На рисунке ниже (Рисунок 60) порт связи “a” передает сигналы порту связи “b” о том, что текущий буфер заполнен. Новое счетверенное слово следует за завершенным блоком.

Рисунок 60 – Порт связи “a” передает порту связи “b” с завершением блока (новый блок из порта связи “a” следует немедленно после завершения первого блока)

7.6.10 Задержки передачи через порт связи

Порты связи должны быть в состоянии преодолевать задержки между источником и пунктом назначения. Задержки для различных типов сигналов (LxCLKOUT и LxDAT3–O) должны соответствовать допускам, указанным в техническом описании. Существует два типа ограничений для задержек в системе:

- задержка от LxCLKOUT в передатчике к LxCLKIN в приемнике плюс задержка от LxACKO в приемнике к LxACKI в передатчике должны быть меньше чем половина периода передачи счетверенного слова.
- максимальная разница между задержкой трассировки nLxBCMPO и задержкой трассировки тактового сигнала (и данных) составляет одну треть периода передачи счетверенного слова.

Первое ограничение связано со следующим фактором. Если передатчик отправляет данные, а приемник передает сигнал о том, что он не готов принять больше данных через LxACKO, то передатчик должен сначала обработать LxACKI до того, как он отправит следующие данные. Если LxACKI не проанализирован вовремя, то данные будут утеряны приемником.

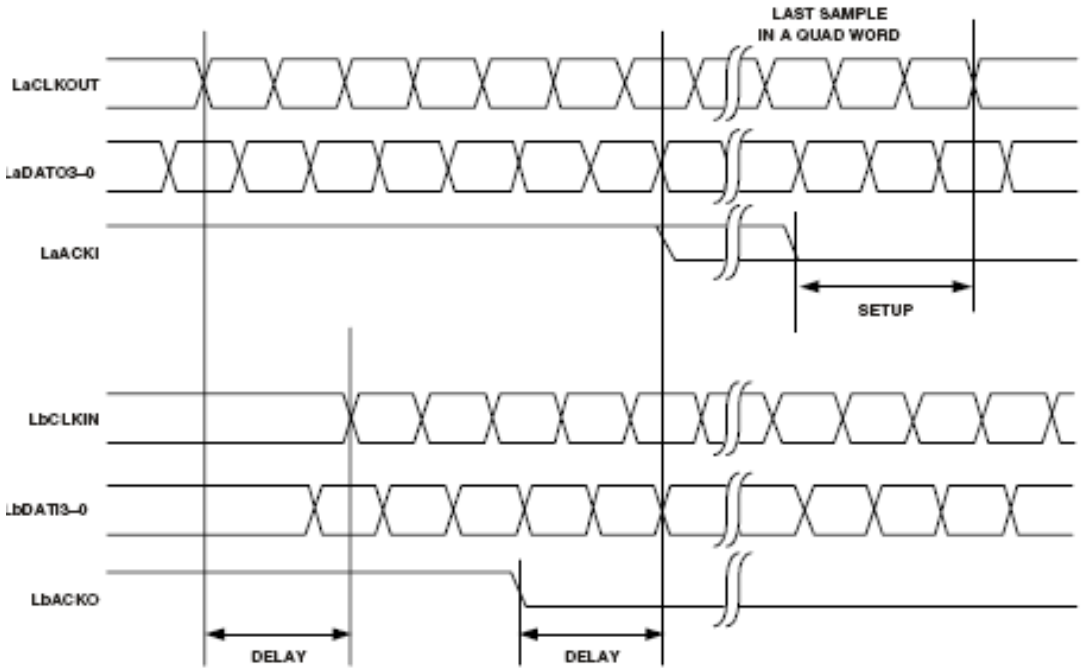


Рисунок 61 – Порт связи “a” передает порту связи “b” с задержкой

7.6.11 Механизмы определения ошибок порта связи

Порты связи поддерживают функции обнаружения ошибок в процессе работы. Когда порт связи обнаруживает состояние ошибки, он продолжает работу в соответствии со следующим алгоритмом:

- Если обнаружена ошибка истечения времени, то устанавливаются соответственно разряды TTER (ошибка истечения времени передачи) в регистр LTSTAT или RTER (ошибка истечения времени приема) в регистр LRSTAT. Если ошибки истечения времени разрешены в регистрах LRCTL или LTCTL, то активируется аппаратное прерывание.
- Если обнаружено переполнение данных в приемнике, то разряд ROVER устанавливается в LRSTAT. Если разрешена ошибка переполнения данных приема в LRCTL, то активируется аппаратное прерывание.
- Если верификация разрешена в LRCTL и обнаружена неверная контрольная сумма, то разряд RCSER устанавливается соответственно в LRSTAT и активируется аппаратное прерывание.
- Если активное значение записывается в LRCTL или LTCTL и значение в регистре управления является активным, то перезапись регистра предотвращается, разряд RWER устанавливается в LRSTAT или разряд TWER устанавливается в LTSTAT соответственно, активируется аппаратное прерывание.

Поля в LRSTAT и LTSTAT, указывающие на ошибку, могут быть сброшены посредством чтения соответствующих регистров LRSTATC и LTSTATC.

7.6.11.1 Время ожидания передачи через порт связи

Если в передатчике LxACKI деактивирован в течение периода 2048 тактовых циклов и в передатчике есть данные, готовые к передаче, то происходит ошибка времени ожидания (time-out) и разряд TTER устанавливается в LTSTAT. На линии LxACKI имеется слабый резистор, понижающий логический уровень. В случае, если нет подключения приемника, то вход будет в неактивном состоянии. Данная ошибка позволяет обнаружить либо отсутствие приемника вообще, либо сбой в его работе.

Также эта ошибка позволяет предотвратить возможное зависание системы в случае ожидания готовности передачи.

7.6.11.2 Время ожидания приемника

Если принимающий буфер приемника полон и нет доступа для чтения в течение 2048 тактовых циклов, то происходит ошибка окончания времени ожидания и разряд RTER устанавливается в LRSTAT. Данная ошибка информирует процессор о возможной некорректной работе программы обслуживания канала связи.

7.6.11.3 Ошибка верификации порта связи

Передатчик имеет возможность формировать байт контрольной суммы и отправлять его вместе с передаваемым счетверенным словом. Передача контрольного байта разрешается специальным битом регистра управления. Контрольный байт рассчитывается как сумма всех байтов данных, которые были переданы. 128-разрядное слово состоит из 16 байтов и контрольный байт отправляется в конце передачи каждых 16 байтов. Приемник принимает первые 16 байт данных, производит вычисление собственной контрольной суммы и сравнивает её с принятым значением. Если два байта различаются, то разряд RCSER устанавливается в регистр LSTAT и формируется аппаратное прерывание.

Алгоритм контрольной суммы следующий:

Checksum = младший байт от суммы (B0 + B1 + … + B15)

7.6.11.4 Ошибка записи приема/передачи через порт связи

Запись активного значения в регистр управления приемника или передатчика является допустимой только при выключенном приемнике или передатчике. Запись активного значения в активный управляющий регистр вызывает ошибку записи. Таким образом, для того, чтобы поменять некоторые параметры обмена необходимо предварительно выключить соответствующий приемник или передатчик и затем включить его снова с требуемым значением.

7.6.12 Регистр управления приемником порта связи (LRCTLx)

Регистры LRCTLx определяют параметры приема для порта связи. При этом ширина порта связи может быть 4 или 1 бит. Существует возможность задать ширину порта связи во время сброса. Значение сброса для регистров LRCTLx равно 0x0000_0001, если при сбросе значение на выводе LINK_DWIDTH = 0 (RDSIZE=0). Если при сбросе значение на выводе LINK_DWIDTH = 1, то значение сброса для регистров LRCTLx является 0x0000 0011 (RDSIZE=1). Подробное описание разрядов регистра приведено ниже (Таблица 103).

Таблица 103 – Регистр LRCTL

Бит	Имя	Назначение
0	REN	Бит включения приемника: 1 – включен 0 – выключен
1	RVERE	Разрешение контроля при приеме 1 – разрешено 0 – запрещено
2	RTOE	Разрешение прерывания в случае обнаружения ошибки time out

3	RBCMPE	Разрешение анализа входа nLxBCMPI: 1 – разрешено 0 – запрещено
4	RDSIZE	Размер шины данных приема: 1 – 4 бита 0 – 1 бит
5	ROVRE	Разрешение прерывания при переполнении буфера приемника: 1 – разрешено 0 – запрещено
6	RINIF	Разрешение проведения процедуры инициализации: 1 – выполнить инициализацию с помощью RINIV значения 0 – не выполнять инициализацию
7	RINIV	Значение данных, которые используются при процедуре инициализации
31:8	-	Зарезервировано

Регистр не может изменяться в процессе выполнения операции обмена. Запись активного значения в контрольный регистр допускается, только когда регистр имеет неактивное значение (REN сброшен). С целью изменения настроек во время работы порта связи, должно быть записано неактивное значение, чтобы прекратить работу порта, после чего следует записать новое активное значение. Игнорирование данного правила может вызвать аппаратное прерывание ошибки и новое значение не будет записано.

7.6.13 Регистр управления передатчиком порта связи (LTCTLx)

Регистр устанавливает параметры передачи для порта связи. Значением сброса для регистров LTCTLx является 0. Подробное описание разрядов регистра приведено ниже (Таблица 104).

Таблица 104 – Регистр LTCTL

Бит	Имя	Назначение
0	TEN	Бит включения передатчика: 1 – включен 0 – выключен
1	TVERE	Разрешение формирования контрольной суммы при передаче: 1 – разрешено 0 – запрещено
2	TTOE	Разрешение прерывания в случае обнаружения ситуации time out
3	TBCMPE	Разрешение формирования выхода nLxBCMPO: 1 – разрешено 0 – запрещено
4	TDSIZE	Размер шины передачи: 1 – 4 бита 0 – 1 бит
7:5	SPD	Частота передачи LxCLKOUT равна: 000 – CCLK 001 – CCLK деленный на 1.5 010 – CCLK деленный на 2 100 – CCLK деленный на 4 Другие значения зарезервированы.
31:8	-	Зарезервировано

Регистр не может изменяться в процессе выполнения операции обмена. Запись активного значения в контрольный регистр допускается, только когда регистр имеет неактивное значение (TEN сброшен). С целью изменения настроек во время работы порта связи, должно быть записано неактивное значение, чтобы прекратить работу порта, после чего следует записать новое активное значение. Игнорирование данного правила может вызвать аппаратное прерывание ошибки и новое значение не будет записано.

7.6.14 Регистр состояния приемника порта связи (LRSTATx)

Регистры LRSTATx указывают статус приемника порта связи. Значением сброса для регистров LRSTATx является 0x0000_0040. Подробное описание разрядов регистра приведено ниже (Таблица 105).

Таблица 105 – Регистр LRSTAT

Бит	Имя	Назначение
1:0	RSTAT	Состояние буфера приемника: 00 – пуст 01 – значение буфера достоверно 11 – буфер полон 10 – буфер не готов (принимает данные)
2	RTER	1 – приемник обнаружил ситуацию time out. 0 – нет ошибки Бит может быть сброшен чтением регистра LRSTATCx
3	RWER	1 – приемник обнаружил ошибку записи 0 – нет ошибки Бит может быть сброшен чтением регистра LRSTATCx
4	RCSER	1 – обнаружена ошибка контрольной суммы 0 – нет ошибки Бит может быть сброшен чтением регистра LRSTATCx
5	ROVER	1 – ошибка переполнения приемника 0 – нет ошибки Бит может быть сброшен чтением регистра LRSTATCx
6	RINIT	Состояние инициализации приемника: 1 – проинициализирован 0 – не инициализировался
31:7	-	Всегда ноль

7.6.15 Регистр состояния передатчика порта связи (LTSTATx)

Регистры LTSTATx указывают статус передатчика порта связи. Значением сброса регистров LTSTATx является = 0x0000 0002. Подробное описание разрядов регистра приведено ниже (Таблица 106).

Таблица 106 – Регистр LTSTAT

Бит	Имя	Назначение
0	TVACANT	1 – буфер может принять данные 0 – буфер не может принять данные
1	TEMPTY	1 – передатчик пуст 0 – передатчик занят Бит может быть использован при выключении передатчика.

2	TTER	1 – ошибка передачи time out 0 – нет ошибки Бит может быть сброшен чтением регистра LTSTATCx
3	TWER	1 – ошибка записи 0 – нет ошибки Активное значение было записано в регистр управления во время, когда передатчик был включен. Бит может быть сброшен чтением регистра LTSTATCx
31:4	-	Всегда ноль

7.6.16 Регистры сброса состояния приемника/передатчика порта связи (LRSTATCx и LTSTATCx)

Чтение регистров возвращает значение регистров статуса приемника/передатчика с автоматическим обнулением некоторых бит регистров статуса приемника/передатчика (см. подразделы «Регистр состояния приемника порта связи (LRSTATx)» и «Регистр состояния передатчика порта связи (LTSTATx)».

7.7 SoC-интерфейс

SoC-интерфейс представляет собой модуль процессора, выполняющий связующую роль между ядром и внутренними периферийными устройствами. Со стороны ядра к интерфейсу подключены четыре шины J-, K-, I-, и S-типов, а с другой стороны одна SoC-шина периферийных устройств. SoC-шина имеет пропускную способность 128 бит, и ее тактовый генератор (SOCCLK) работает на частоте, равной половине частоты ядра (CCLK). Периферийными устройствами, которые могут быть ведущими на SoC-шине являются (Рисунок 62):

- SoC – интерфейс;
- внешний порт (EBIU);
- контроллер DMA.

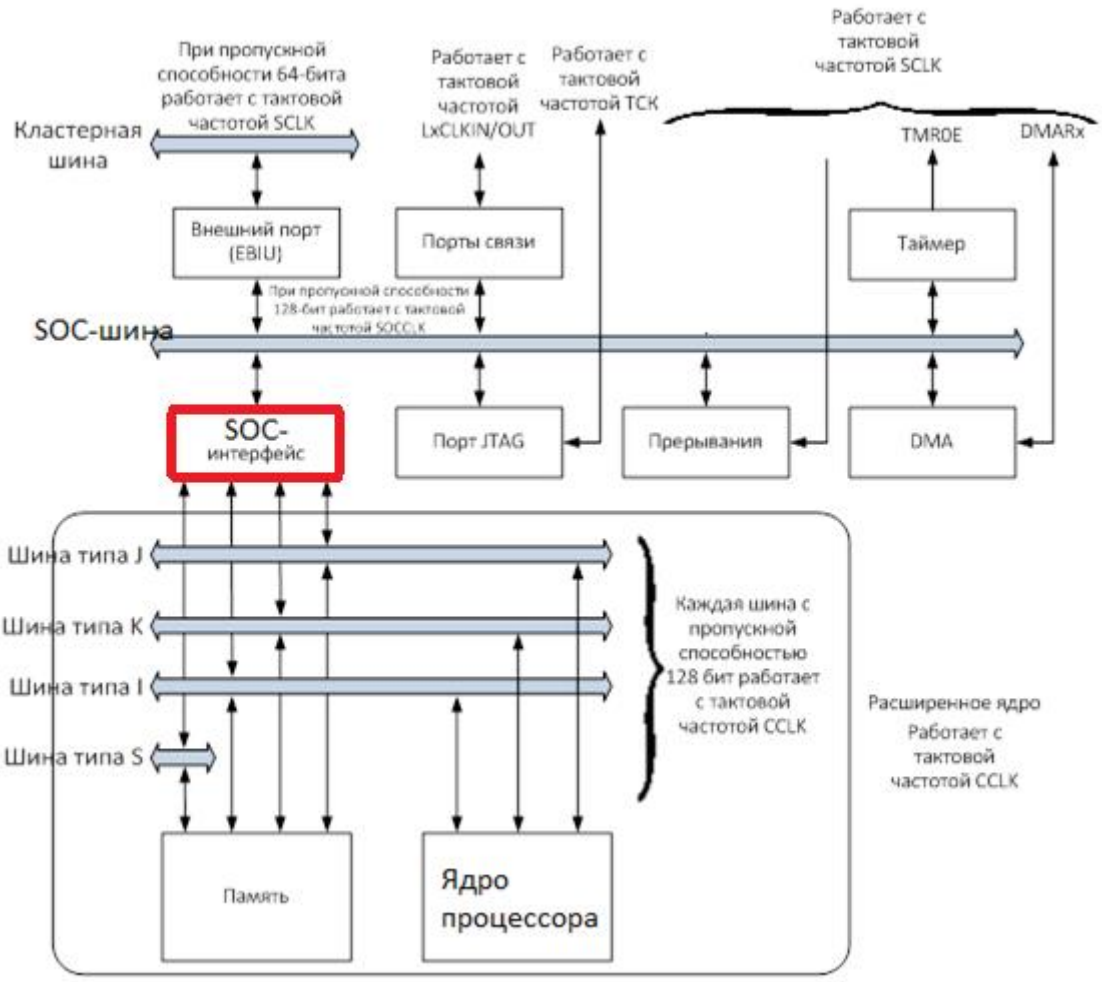


Рисунок 62 – Подключение периферийных модулей

Периферийные устройства, которые могут быть подчиненными на SoC-шине:

- Порты связи (линк-порты);
- Контроллер прерываний;
- Регистры контроллера DMA;
- Таймеры;
- Порт JTAG;
- Внешний порт (EBIU);
- SOC-интерфейс.

Ниже показана структура SOC-интерфейса (Рисунок 63) и архитектура шин, соединяющих различные модули SOC-шины (Рисунок 64).

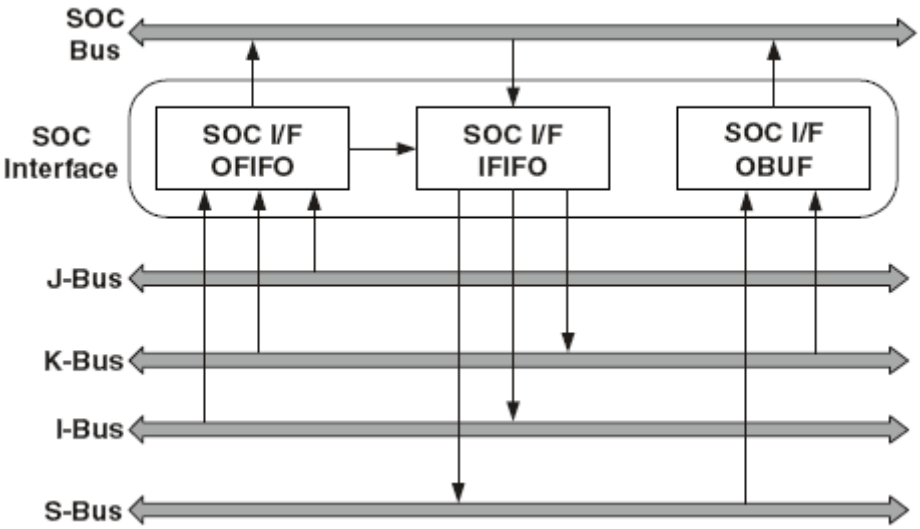


Рисунок 63 – Поток данных через SOC-интерфейс

В связи с тем, что на SOC-шине может быть несколько ведущих устройств, при каждом доступе к шине выполняется арбитраж. При этом доступ к периферийным устройствам возможен со стороны процессора и со стороны внешнего интерфейса. Со стороны процессора представителем является SOC-интерфейс и при этом он имеет два источника запросов: выходное FIFO интерфейса (OFIFO) и выходной буфер (OBUF). Приоритеты доступа при арбитраже фиксированы. Наивысший приоритет имеет выходной буфер интерфейса OBUF который может возвращать прочитанное значение в контроллер DMA либо в порт связи. Далее идет внешний интерфейс и наименьший приоритет имеет OFIFO.

SOC интерфейс состоит из трех устройств FIFO через которые проходят все потоки данных. Если процессор обращается во внешнюю память, то запрос из SOC-интерфейса поступает сразу во внешний интерфейс. Если шины процессорного ядра посылают в SOC-интерфейс запрос на чтение, то прочитанные данные должны вернуться обратно в интерфейс. Для возвращаемых данных используется входное IFIFO. Также в данный буфер помещаются запросы всех внешних ведущих устройств, которые обращаются к внутренней памяти ядра или регистрам ядра. Если модуль IFIFO получил прочитанные данные, то он пересылает их во внутренний регистр ядра. Если же был получен запрос на запись со стороны ведущего устройства, то записываемые данные пересылаются по адресу-приемнику (используется шина S). В случае получения от внешнего ведущего устройства запроса на чтение, производится чтение внутренней памяти (используется шина S) или регистра и прочитанные данные помещаются в выходной буфер OBUF. Таким образом, в OBUF попадают только считываемые внешними ведущими устройствами данные ядра.

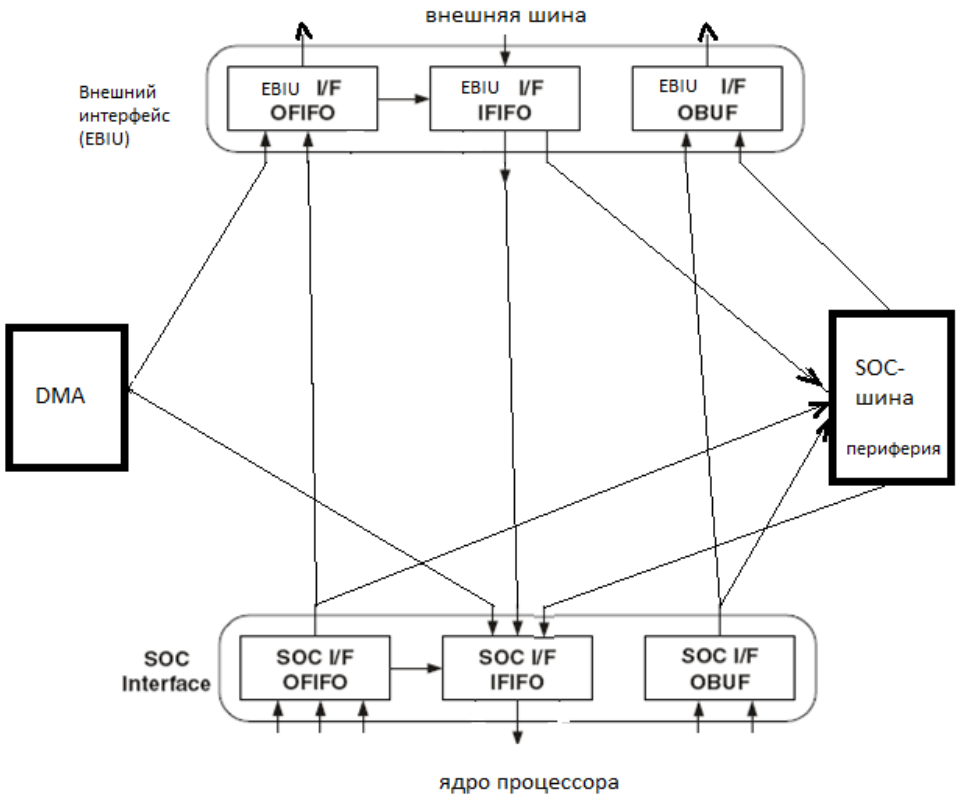


Рисунок 64 – Структура соединений модулей SOC-шины

7.7.1 Транзакции SOC OFIFO

Все запросы ядра по шинам J, K, I, которые не попадают во внутреннюю память, помещаются в выходное OFIFO. Это могут быть:

- запись из ядра процессора по шинам J и K в устройства SOC-шины или внешнее адресное пространство;
- запросы чтения по шинам J и K из устройств SOC-шины или внешнего адресного пространства;
- запрос чтения команды по шине I из внешней памяти или из регистра инструкций эмулятора (EMUIR).

Все запросы ядра поступают в OFIFO с частотой ядра (CCLK). Считывание запросов из буфера выполняется с частотой SOC-шины (SOCCLK). На выходе буфера проверяется к какому ресурсу выполняется запрос (внешняя память либо внутренняя периферия) и далее запрос пересылается в пункт назначения. Если выполняется запрос к периферии, то он имеет наименьший приоритет в арбитраже доступа. Если выполняется запрос к внешней памяти, то также происходит арбитраж доступа к выходному FIFO внешнего интерфейса. За доступ к внешнему ресурсу OFIFO соревнуется с контроллером DMA (Рисунок 64). SOC-интерфейс в этом арбитраже всегда имеет наивысший приоритет.

С точки зрения конвейера процессора, команда завершается, когда запрос на запись или на чтение помещается в SOC OFIFO. При этом имеется следующая особенность: если это операция записи, то она считается выполненной и конвейер ядра продолжает движение, а если операция чтения, то конвейер останавливается до момента возвращения данных. При этом неважно имеется зависимость по данным или нет. При чтении команд из внешней памяти после помещения двух запросов в OFIFO конвейер команд останавливается до момента получения

прочитанной команды. При этом конвейер обработки команд может продолжать выбирать команды из буфера выравнивания и отправлять на исполнение.

Буфер OFIFO имеет размер 8 записей. Это означает что он может принять до 8-ми запросов без остановки конвейера процессора. При этом имеется следующие особенности. Если, например, по шинам J и K мы имеем одновременно две записи, то процессор потратит один такт чтобы поместить их в OFIFO. При этом одна транзакция (J) помещается сразу в OFIFO, а вторая (K) во временный буфер. Из временного буфера запрос пересылается в OFIFO в следующем такте. Если процессор не требует доступа к OFIFO в следующем такте, то не будет никаких задержек конвейера. Напомним, что запросы в OFIFO помещаются с частотой ядра, а считываются с частотой SOC-шины, т.е. в два раза меньшей. Поэтому в случае интенсивного потока записи во внешнюю память возможно заполнение буфера и это вызовет приостановку конвейера процессора.

7.7.2 Транзакции SOC IFIFO

Входной буфер IFIFO SOC-интерфейса обрабатывает два типа транзакций:

- Запросы чтения ядра процессора: SOC IFIFO принимает прочитанные данные, полученные в результате транзакций чтения, которые были переданы через OFIFO.
- Запросы чтения или записи внутренней памяти (или регистров) ядра от внешнего по отношению к ядру ведущего устройства. Это может быть DMA, процессор кластера, хост-процессор.

В первом случае возвращаемые данные записываются в регистр ядра, используя шину K. Во втором случае IFIFO использует шину S для доступа к внутренней памяти либо шину K для доступа к регистрам. В случае операции чтения внутреннего ресурса, прочитанные данные помещаются в выходной буфер OBUF.

Имеется три источника запросов к IFIFO (Рисунок 64): DMA, внешний интерфейс и SOC-шина. Наивысший приоритет имеет SOC-шина т.к. её запрос означает чтение регистра периферии по запросу ядра, далее идет запрос внешнего интерфейса и наименьший приоритет у контроллера DMA. Запрос от внешнего интерфейса может означать как возвращение данных в ответ на запрос чтения внешней памяти ядром, а также это может быть цикл обмена внешнего ведущего устройства. Приоритеты доступа расставлены так чтобы запросы ядра получали наивысший приоритет. В случае чтения, это минимизирует простои процессора.

7.7.3 Транзакции SOC OBUF

Выходной буфер OBUF SOC-интерфейса задействуется только в операциях чтения внутренних ресурсов (памяти или регистров) ядра внешними ведущими устройствами. Если запрос чтения инициируется контроллером DMA, то данные из OBUF могут пересылаться в DMA или порты связи. Если чтение выполняет внешнее ведущее устройство, данные из SOC OBUF пересылаются в выходной буфер внешнего порта.

7.7.4 Программирование SOC интерфейса

Проблемы программирования, которые могут встретить разработчики при работе с SOC интерфейсом, относятся к пониманию проблем, которые влияют на производительность при работе с SOC интерфейсом. Чтение процессором внешней памяти вызывает передачу запроса от ядра через несколько доменов

синхронизации. При доступе к мультипроцессорной шине необходимо выполнить захват шины. Все это может приводить к длительным остановам процессора.

Ниже приведены некоторые рекоменации к программированию с использованием SOC-интерфейса:

- Следует избегать исполнения кода из внешней памяти. Скорость исполнения будет в несколько десятков раз медленнее, чем из внутренней памяти.
- Следует избегать чтения данных из внешней памяти или записи данных во внешнюю память при помощи команд ядра. Предпочтительнее для этих целей использовать контроллер DMA.
- При необходимости выполнения операций обмена ядром на внешней шине в мультипроцессорном кластере лучше использовать бит блокировки шины и после захвата шины выполнять обмен.
- Необходимо максимально использовать DMA для пересылок данных, а также сигнал прерывания от DMA по завершении работы.
- Необходимо обеспечить работу процессорного ядра и DMA с различными банками внутренней памяти. Это уменьшит количество конфликтов доступа.
- При пересылках данных следует использовать максимальную ширину шины в 128 бит.
- При чтении нескольких данных ядром процессора из внешней памяти или SOC-периферии предпочтительнее использовать в одной линии команд две операции чтения J и K шин. В этом случае обе операции чтения выполнятся практически как одна и за тоже время, что и одна операция чтения.
- При работе с внешней памятью в мультипроцессорной системе следует использовать регистр BMAX для задания времени владения шиной. Это минимизирует простои других процессоров от длительного ожидания шины.

7.8 Таймер и универсальные линии ввода/вывода

Процессор имеет два 64-битных таймера общего назначения: таймер 0 и таймер 1. Таймеры – это автономно работающие счетчики, имеющие следующие функции:

- загрузка начальным значением;
- декремент значение до нуля;
- индикация завершения счета;
- автоматическая перезагрузка;
- повторение цикла.

Индикатор завершения счета (такт после вычитания счетчика, когда он был равен 1) является прерыванием, но для таймера 0 дополнительно есть функция выдачи индикатора на внешний вывод. Каждый таймер содержит два 64-битных регистра: начальное значение (TMRINxH/L) счета и текущее значение (TIMERxH/L) счетчика. Регистр TMRINxH/L доступен для чтения и записи, тогда как TIMERxH/L предназначен только для чтения. Хотя в действительности оба регистра имеют длину 64 бита, но для программиста каждый регистр доступен только как две 32-разрядные половины: старшая и младшая. Доступные 32-битные регистры: TMRIN0H, TMRIN0L, TMRIN1H, TMRIN1L, TIMER0H, TIMER0L, TIMER1H, и TIMER1L. При этом нет никакой буферизации при считывании или записи различных частей регистра. Регистр TIMERx выполняет функцию счетчика и обновляется каждый такт частоты SOCCLK. Если счетчик в текущем такте имеет значение 1, то в следующем такте в него загружается начальное значение из регистра TMRINx.

Каждый из таймеров имеет бит разрешения счета, который хранится в регистре управления INTCTL. Установка бита TMRxRN в 1 разрешает счет, а сброс бита в 0 останавливает счет.

7.8.1 Операции таймера

После сброса таймеры остановлены т.к. биты TMRxRN в регистре INTCTL сбрасываются. Для запуска и работы таймера необходимо установить соответствующий бит в TMRxRN. После установки бита, значение в TMRINxH/L копируется в регистр TIMERxH/L, и таймер начинает обратный отсчет.

Программы могут останавливать таймер, сбрасывая бит TMRxRN и запускать его, устанавливая бит TMRxRN. Отметим что загрузка нового значения в регистр TMRINxH/L не вызывает немедленной загрузки этого же значения в регистр счетчика TIMERxH/L. Новое начальное значение будет загружено в счетчик только в двух ситуациях:

- при изменении бита TMRxRN из 0 в 1, т.е. при включении таймера;
- при работе таймера в момент достижения счетчиком значения 0.

При работающем таймере в момент достижения счетчиком значения нуля, таймер:

- генерирует два прерывания с высоким и низким приоритетом;
- загружает в счетчик начальное значение;
- начинает обратный отсчет.

Таймер имеет биты запроса прерывания в регистрах ILATx (INT_TIMER0LP, INT_TIMER1LP, INT_TIMER0HP, или INT_TIMER1HP). Прерывание от таймера импульсное и после установки оно может быть сброшено при обслуживании данного запроса.

Используйте следующую процедуру для управления таймерами:

1. Если таймер работает, отключите его сбросом бита TMRxRN в регистре INTCTL.
2. Если прерывание должно произойти, установите высокий или низкий приоритет прерывания таймера:
   a. Назначьте адрес(а) вектора прерываний для программы, загрузив регистр(ы) вектора прерываний (IVTIMERxH/LP).
   b. Разрешите прерывания таймеров, установив бит INT_TIMERxH/LP в регистре IMASK.
   c. Разрешите глобальные прерывания, установив бит SQCTL_GIE в регистре SQCTL.
3. Загрузите начальное значение в регистры TMRINxH/L.
4. Запустите таймер, установив бит TMRxRN в регистре INTCTL.

7.8.2 Вывод срабатывания таймера (TMR0E)

Внешний вывод микросхемы TMR0E показывает момент завершения счета таймером 0. В момент, когда таймер изменяет свое значение с 1 на начальное, процессор генерирует высокий уровень на вывод TMR0E в течение четырех тактов SOCCLK.

7.9 Флаги

Процессор имеет четыре внешних вывода общего назначения (FLAG3–0) которые могут выполнять функцию флагов. Можно сказать, что данные выводы являются портом общего назначения. Каждый вывод может быть сконфигурирован как вход или выход индивидуально. При конфигурации как выход, программы могут использовать выводы флагов для сигналов событий или условий для любого внешнего устройства. При конфигурации как входы, программы могут считывать состояние внешних выводов, используя регистр FLAGREG или использовать значение входа как условие в условных командах. После сброса FLAG3–0 работают как входы с внешним нагрузочным резистором, который держит их в высоком логическом уровне.

Для подготовки и использования флагов следует использовать следующую процедуру:

1. С помощью битов FLAGx_EN регистра FLAGREG сконфигурируйте каждый из выводов FLAG3–0 как вход (FLAGx_EN=0) или выход (FLAGx_EN=1).
2. С помощью битов FLAGx_OUT регистра FLAGREG выберите значения выхода для каждого вывода FLAG3–0 (для выводов, сконфигурированных как выход).
3. Прочитайте биты FLGx в регистре SQSTAT для определения входного значения для каждого вывода FLAG3–0, или используйте входные значения флагов (FLAGx_IN) в условных командах.

Выводы флагов (FLAG3–0) позволяют процессору посылать сигналы внешним устройствам или получать входные сигналы от них. При необходимости формирования импульсов определенной длительности отметим, что регистр FLAGREG относится к группе регистров устройства управления и тактируется клоком ядра CCLK.

7.10 Порт JTAG и модуль отладки

Функциональность, которая описана в данном подразделе, может использоваться:

- для отладки программ;
- для разработки ядра операционной системы.

Описываемые ниже функции реализованы в аппаратной части процессора для того, чтобы позволить программе-отладчику и операционным системам выполнять более сложные задачи. Функциональность разбита на группы по нескольким уровням. Эти уровни выстроены друг над другом, что позволяет отладчику или ядру ОС осуществлять более качественный контроль и анализ процессора.

Отладчик может осуществлять контроль процессора двумя путями:

- используя отладочный монитор (работающий в режиме супервизора);
- используя встроенный эмулятор (ICE).

При отладке с помощью программного монитора в режиме супервизора, некоторые системные ресурсы используются только монитором (например, память, канал коммуникации с хостом, прерывание, флаги и т. П.).

При отладке с использованием встроенного эмулятора, используется резервный коммуникационный канал (порт доступа к тестированию JTAG) для контроля процессора.

Порт JTAG является подчиненным устройством на SOC-шине. Многие отладочные функции являются востребованными как при использовании программного метода отладки, так и при аппаратном.

Функции и возможности процессора для поддержки отладки программ включают:

- Точки наблюдения
  Позволяет пользователю указывать конкретный диапазон адресов точки наблюдения и условия, которые останавливают процессор. После остановки, пользователь может проанализировать состояние процессора, восстановить прежнее состояние и продолжить выполнение команд. Программные точки останова и одношаговые операции также выполняются с помощью точки наблюдения.
- Мультипроцессорные возможности
  Позволяют пользователю выполнять одношаговые операции и устанавливать точки останова в мультипроцессорной системе.
- Запись истории
  Специальный буфер трасс позволяет проследить за программным счетчиком с целью восстановления хода программы. Буфер трассировки на кристалле сохраняет последние 32 перехода (все дискретные значения счетчика программ) выполненные устройством управления ядра, позволяя восстановить последний программный путь.
- Счетчик тактов
  Обеспечивает функцию подсчета количества процессорных тактов для всех функций программного кода.
- Мониторинг производительности
  Позволяет осуществлять мониторинг некоторых внутренних событий ядра процессора при выполнении программного кода. Отладчик может точно указать, какое из внутренних событий нуждается в мониторинге.
- Доступ к защищенным регистрам

Защищенные регистры доступны только в режиме супервизора или ICE. К ним нет доступа в режиме пользователя.

- Счетчики точки наблюдения
  Позволяет пользователю искать n-е возникновение события до остановки программы.
- Исключение
  Событие, которое отменяет выполнение всех последующих команд.

7.10.1 Рабочие режимы

Процессор работает в одном из трех режимов: режим эмулятора, супервизора или пользователя. В режимах пользователя и супервизора все команды выполняются стандартно. Тем не менее, в режиме пользователя ограничен доступ к регистрам. В режиме эмуляции имеются ограничения на использование некоторых типов команд.

7.10.2 Ресурсы отладки

В процессоре предусмотрены несколько ресурсов отладки, включая:

- специальные команды;
- точки наблюдения;
- буфер трассировки адреса команды.

7.10.3 Специальные команды

Процессор поддерживает специальные команды, которые используются для помощи в отладке системы. Данные команды нужны для выполнения программных остановов в отладчиках и вызовов операционной системы.

7.10.4 Точки наблюдения

Точки наблюдения адреса позволяют пользователю проверять состояние процессора во всех случаях, когда происходит выполнение условий доступа к памяти, определенной пользователем. Существует три точки наблюдения, которые работают параллельно. Во время работы каждой точки наблюдения пользователь может определять условия срабатывания при доступе к шине и действия, которые должен выполнить процессор.

Точка наблюдения имеет два адресных регистра: WpiL и WpiH. Когда точка наблюдения запрограммирована на один адрес, то используется только регистр с младшим адресом (WpiL). Если точка наблюдения запрограммирована на диапазон адресов, то в WpiL содержится младший адрес, а в WpiH – старший адрес. Необходимо установить регистры указателя адреса до того, как установлен регистр управления WPiCTL, который переводит точку наблюдения в активное состояние. Адресные регистры точки наблюдения не могут быть изменены, пока точка наблюдения активна.

Работа точки наблюдения определяется ее регистром управления WPiCTL, (где: i – это 0, 1 или 2 в зависимости от номера точки наблюдения).

Следующая информация должна быть запрограммирована в регистрах до начала поиска:

- адрес или диапазон адресов;
- отсчет.

Когда регистр управления устанавливается в активное состояние, поле отсчета устанавливается в начало отсчета. С этого момента аппаратная часть точки

наблюдения отслеживается внутренние шины с целью совпадения адресов передач, что указывается в регистрах точек наблюдения.

При каждом совпадении значение счетчика уменьшается. После того, как счетчик достигает значение нуля, инициируется исключение: программное исключение или вызов эмулятора. Выбор действий определяется разрядами 3-2 регистра WPiCTL.

С этого момента регистр состояния указывает на завершение работы точки наблюдения. Чтобы начать новый поиск, пользователю необходимо снова записать регистр управления. Три точки наблюдения связаны с тремя различными шинами.

Точка наблюдения 0 используется для мониторинга на I-шине. Точка наблюдения 0 поддерживает пошаговую опцию (SSTP), которая используется при эмуляции для выполнения одношаговых операций.

Точка наблюдения 1 используется для наблюдения за доступом к данным через J- и K-шины. При этом мониторинг может выполняться для одной из шин или для обеих сразу. Точка наблюдения 1 поддерживает также выбор типа доступа: только чтение, только запись, чтение и запись.

Точка наблюдения 2 используется для наблюдения за доступом к данным через S-шину. Точка наблюдения 2 поддерживает также выбор типа доступа: только чтение, только запись, чтение и запись.

Регистр WPiSTAT указывает на текущее состояние точки наблюдения. Он хранит текущее значение счетчика поиска и режима работы точки наблюдения. Описание разрядов регистра состояния приведено ниже (Таблица 107).

Таблица 107 – Описание разрядов регистра WPiSTAT

Разряды	Имя	Опиание
15–0	VALUE	Значение точки наблюдения
		Определяет текущие значения счетчика точки наблюдения.
17–16	EX	Разряд выполнения
		Выполнение точки наблюдения 00 – точка наблюдения запрещена 01 – поиск совпадений 11 – окончание работы счетчика точки наблюдения
31–18	Reserved	Резервные

7.10.4.1 Буфер трассировки адреса команды (TBUF)

Поскольку программный счетчик недоступен вне кристалла в реальном времени в процессе выполнения команд, то буфер трассировки используется для помощи в отладке программы.

Буфер трассировки включает в себя 32 регистра, в которых хранится история последних от 16 до 32-х переходов, выполненных устройством управления ядра процессора, что дает возможность пользователю полностью восстановить путь выполнения программы в течение последних переходов.

Переходы циклично записываются в регистры буфера трассировки. Первый записывается в TRCB0, следующий в TRCB1, и т. Д. до TRCB31. Тридцать второй переход снова записывается в TRCB0, и так далее. Для того, чтобы определить какой из регистров был записан последним, пользователь должен обратиться к регистру указателя буфера трассировки, который указывает на последний записанный вход.

Буфер трассировки всегда запоминает адрес команды, вызвавшей переход. Если адрес перехода вычислялся с использованием дополнительного регистра

(не только PC и смещение в коде команды), то кроме адреса команды перехода в буфере трасс запоминается и адрес перехода.

Для того, чтобы различать какой тип информации записан в линии буфера трасс, используется регистр маски TRCBMASK. Каждый разряд в RCBMASK связан с соответствующим регистром TRCBi. Если он сброшен, то TRCBi хранит адрес команды перехода. Если же он установлен, то TRCBi содержит адрес перехода.

7.10.5 Мониторинг производительности

Целью мониторинга производительности является оптимизация функционирования рабочих приложений. Аппаратные возможности процессора обеспечивают подсчет числа циклов, которые были затрачены на выполнение кода программы или подсчет числа заданных событий при выполнении программы. Мониторинг обеспечивается с помощью трех регистров: счетчик циклов, маска монитора производительности и счетчик монитора производительности.

Счетчики циклов и монитора производительности должны быть установлены пользователем в ноль перед тем, как начнется определенный период работы. Соотношение между общим счетчиком и счетчиком монитора производительности и дает требуемое значение.

Когда счетчик монитора производительности переполняется, это вызывает исключительную ситуацию.

7.10.5.1 Регистр маски монитора производительности (PRFM)

Регистр маски монитора производительности (PRFM) определяет, какие события в процессоре отслеживаются и подсчитываются счетчиком монитора производительности (PRFCNT). Значением сброса регистра PRFM является 0x0000 0000. Подробное описание разрядов регистра приведено в таблице ниже.

Таблица 108 – Описание разрядов регистра

Бит	Имя	Назначение
7:0	-	Резерв
8	Jexe	Подсчет команд, выполненных на линии конвейера J 1 – включен 0 – выключен
9	Kexe	Подсчет команд, выполненных на линии конвейера К 1 – включен 0 – выключен
10	Xexe	Подсчет команд, выполненных на линиях конвейера X1 и X2 1 – включен 0 – выключен
11	Yexe	Подсчет команд, выполненных на линиях конвейера У1 и У2 1 – включен 0 – выключен
12	Sexe	Подсчет команд, выполненных на линии конвейера S 1 – включен 0 – выключен
15:13	-	Резерв
16	STALL	Подсчет команд остановов конвейера вызванных ожиданием данных или другими блокирующими конвейер ситуациями 1 – включен 0 – выключен
17	BTB_true	Подсчет количества правильных предсказаний буфера переходов 1 – включен 0 – выключен

18	ABORT	Подсчет количества программных исключительных ситуаций 1 – включен 0 – выключен
19	Uexe	Подсчет количества линий команд, выполненных в режиме пользователя 1 – включен 0 – выключен
20	BTB_false	Подсчет количества ошибочных предсказаний буфера переходов 1 – включен 0 – выключен
21	BTB_load	Подсчет количества загрузок в буфер переходов 1 – включен 0 – выключен
31:22	-	Резерв

Все события, которые отслеживаются монитором производительности, суммируются по «ИЛИ» и при их наступлении происходит увеличение счетчика PRFM на 1. Логичным является разрешение подсчета одного условия из многих.

7.10.5.2 Регистр счетчика монитора производительности (PRFCNT)

Назначение счетчика производительности – увеличивать свое значение на 1 каждый раз, когда происходит отслеживаемое событие. Отслеживаемое событие определяется регистром маски монитора производительности (PRFM). Счет прекращается, когда процессор переходит в режим эмуляции. Регистр очищается после сброса.

7.10.5.3 Регистры счетчика циклов (CCNTx)

Длина счетчика циклов равна 64 разряда, однако доступен он как два универсальных 32-разрядных регистра – CCNT0 и CCNT1. После сброса значение счетчика равно нулю.

В связи с тем, что счетчик 64 бита, а доступ возможен только к частям регистра, необходимо соблюдение специальной процедуры чтения и записи регистра. При чтении сначала считывается нижняя часть (CCNT0), а затем верхняя часть (CCNT1). При чтении нижней части верхняя копируется в буфер и это гарантирует её корректное значение. При записи сначала пишется нижняя часть (она попадает в буфер), а затем верхняя. При записи верхней полное 64-битное значение попадает в счетчик.

Счет прекращается, когда процессор переходит в режим эмуляции.

7.10.5.4 Регистр данных точки наблюдения 1 (WPDR)

Точка наблюдения 1 имеет дополнительные возможности по мониторингу обмена данными ядра процессора с памятью. Выше описывался механизм срабатывания исключительной ситуации в случае совпадения адресов точки наблюдения с адресами шин J и K. Однако имеется возможность дополнить сравнение адресов еще и сравнением данных. При чтении имеется возможность контролировать, то какие данные считывает процессор. Если дополнительно к совпадению адресов происходит совпадение и прочитанных данных – точка срабатывает. Также можно контролировать и процесс записи данных. Необходимые для контроля данные записываются в регистр WPDR. Необходимо отметить, что шина чтения или записи имеет разрядность 128 бит. Разрядность регистра данных только 32 бита. Поэтому сравнение выполняется только для 32-разрядного слова шины данных определяемого младшими битами шины адреса, независимо от того выполняется чтение или запись 64-х или 128-разрядного слова.

7.10.5.5 Регистр маски точки наблюдения 1 (WPMR)

При использовании шины данных в описании точки наблюдения 1, не все биты шины данных могут понадобиться при анализе читаемых или записываемых данных. Регистр маски позволяет установить, какие биты регистра данных WPDR должны участвовать в сравнении. Если бит установлен в 1 – соответствующий ему бит данных сравнивается с битом шины данных. После сброса регистр маски равен нулю и данные не участвуют в работе точки наблюдения.

7.10.6 JTAG интерфейс

Процессор поддерживает порт доступа к средствам тестирования, соответствующий стандарту IEEE 1149.1.

7.10.6.1 Выводы JTAG порта

Таблица 109 описывает выводы процессора, используемые в аппаратной части эмуляции JTAG. Подчеркнутые снизу названия выводов и выводы с чертой над названием указывают на активные низкие сигналы.

Таблица 109 – Выводы порта связи и эмуляции устройства ввода/вывода

Сигнал	Тип	Описание
nTRST	вход	Сброс JTAG. Сбрасывает интерфейс JTAG. Сигнал nTRST должен быть активирован после включения питания, чтобы убедиться в правильной работе JTAG. Вход nTRST имеет внутренний повышающий резистор
TCK	вход	Синхронизация JTAG. Обеспечивает тактовый сигнал для работы JTAG
TDI	вход	Входной сигнал данных JTAG. Имеет внутренний повышающий резистор.
TDO	выход	Выходной сигнал данных JTAG
TMS	вход	Выбор режима тестирования JTAG. Управляет машиной состояний тестирования. Имеет внутренний повышающий резистор/
nEMU	выход	Эмуляция Признак перехода процессора в режим эмуляции.

В дополнение к стандартным функциям JTAG, выводы TMS и nEMU выполняют следующие функции отладки:

- вход TMS (когда разряд TEME установлен в регистре EMUCTL) выполняет функцию запроса к процессору на переход в режим эмуляции. Исключение эмуляции формируется при каждом нарастающем фронте сигнала TMS. Данная функция является дополнительной к функциональности TMS в JTAG.
- выходной сигнал nEMU является сигналом с открытым стоком, активизируемый, только если разряд EMUOE установлен в регистре EMUCTL. Вывод nEMU указывает на следующее:
- произошло срабатывание точки наблюдения, но регистр управления имеет в поле типа (поле EXTYPE в WPiCTL) значение ноль. Уровень сигнала nEMU в этом случае снижается на 20 циклов CCLK.
- каждый раз, когда процессор находится в режиме эмуляции и готов к выполнению новых линии команд, подаваемой в регистр EMUIR, уровень сигнала на выводе nEMU снижается до тех пор, пока такая команда добавляется.

7.10.6.2 Регистр команды JTAG

Регистр команды JTAG имеет длину, равную пяти разрядам, и позволяет контроллеру доступа порта (TAP) выбрать любой из сканируемых регистров данных. Младший значащий бит первым выдвигается на TDO. Кодирование команд приведено ниже (Таблица 110).

Таблица 110 – Декодирование команды доступа порта JTAG

Если значение IR4–0 равно...	Соответствующая команда...	Выполнить...
00000	EXTEST	Выполнить EXTEST
10000	Sample/Preload	Выполнить сэмплирование и предзагрузку
01000	EMUIR	Сканирование в регистре EMUIR
10100	EMUDAT	Сканирование в регистре EMUDAT
11110	SAMPLEPC	Сэмплирование ПК, к которому подключен процессор
10001	EMUTRAP	Выполнить перехват эмуляции
10011	OSPID	Сканирование регистра OSPID
00100	EMUCTL	Сканирование регистра EMUCTL
01100	EMUSTAT	Сканирование регистра EMUSTAT
11111	BYPASS	Параллельный режим

7.10.6.3 Регистры данных

Регистры данных выбираются через регистр команд. Как только соответствующее значение регистра данных записывается в регистр команд, и состояние TAP изменяется на SHIFT-DR, данные выбранного регистра начинают выдвигаться на TDO и приниматься с TDI. Больше деталей в спецификации стандарта IEEE 1149.1.

Регистры данных:

- Обход (Байпас)
  Регистр, отвечающий требованиям IEEE 1149.1, является одноразрядным регистром.
- Пограничные регистры
  Регистры, используемые многими командами JTAG. Все три команды JTAG соответствуют требованиям IEEE 1149.1
- EMUIR
  Команда JTAG загружает 48 разрядов, из которых 32 наименьших значащих разряда загружаются в регистр EMUIR. EMUIR является 128-разрядным регистром. В режиме эмуляции регистр загружается четыре раза последовательно (одной командой JTAG) или загружается до тех пор, пока в загружаемом слове не установится разряд EX (тогда остальные слоты EMUIR считаются командами NOP). Каждый раз, когда данные загружаются в EMUIR – сначала загружаются разряды 31–0, затем загружаются разряды 63–32 и так далее.

Когда четыре загрузки EMUIR завершены (или линия команд была завершена установленным разрядом EX), команды, загруженные в JTAG, передаются в устройство управления ядра и выполняются.
Когда счетверенное слово загружается в EMUIR, оно должно включать одну полную линию команд. Линия команд не должна продолжаться в следующем счетверенном слове. Лишние слоты должны быть заполнены NOP командами.
Запись в EMUIR является недопустимой операцией, когда процессор не находится в режиме эмуляции.

- EMUDAT
  Это 48-разрядный регистр, в котором 32 старших значащих разряда используются для сканирования Ureg с таким же именем. Эмулятор использует этот регистр для чтения и записи всех остальных регистров в процессоре.
  Эмулятор также использует данный регистр для доступа к памяти. Должна быть команда, использующая IALU для генерации адреса доступа к памяти, назначением или источником которой является EMUDAT. Когда эмулятор выполняет серию доступов к памяти, процессор должен вставить циклы удержания для того, чтобы завершить все доступы (как во время нормальной работы).

7.11 Сброс и запуск

Процессор поддерживает три уровня сброса:
- Сброс при включении питания – после включения системы (SCLK, все входные сигналы и конфигурационные входы находятся в устойчивом состоянии), вывод nRST_IN должен быть активен (низкий уровень сигнала).
- Нормальный сброс – любой сброс схемы, следующий после сброса при включении питания, вывод nRST_IN должен быть активен (низкий уровень сигнала).
- Сброс ядра DSP – при установке бита SWRST в EMUCTL сбрасывается ядро DSP, но не внешний порт и не устройство ввода-вывода.

После сброса возможны четыре варианта запуска процессора, описанные в Разделе 9 «Методы загрузки процессора».

7.12 Диапазон напряжения питания

Процессор имеет отдельные подключения к источнику питания для внутреннего ЛУ (VDD), тактовый генератор PLL (VDD_A), буфера устройства ввода-вывода (VDD_IO).

Для стабильной синхронизации, система должна обеспечивать чистое электропитание на входе VDD_A. Особое внимание при проектировании должно быть уделено фильтрации источника питания VDD_A.

Допустимые значения напряжений питания приведены в Таблице 123: для вывода VDD параметр U_{CC}, для VDD_A – U_{CCA}, для VDD_IO – U_{CCIO}.

8 Временные характеристики

Всех сигналы, за исключением выводов nDMAR3–0, nIRQ3–0, TMR0E и FLAG3–0 (только в режиме входа), являются синхронными по отношению к сигналу опорной частоты SCLK. В микросхеме реализовано всего несколько типов площадок ввода-вывода, и потому времена предустановки/удержания сигнала на входе, времена установки/удержания сигнала на выходе и времена включения/выключения буферов выходных сигналов, рассчитанные относительно фронта SCLK, являются одинаковыми для большинства выводов микросхемы.

Более подробная информация приведена в разделах «Основные временные характеристики переменного тока» и «LVDS-интерфейс, электрические и временные характеристики».

8.1 Основные временные характеристики переменного тока

Временные характеристики измерялись, когда сигнал пересекал уровень 1,25 В, как это показано на Рисунок 69. Все временные задержки (в наносекундах) измерялись между точкой, когда первый сигнал (SCLK) достигал уровня 1,25 В и точкой, когда второй сигнал достигал уровня 1,25 В.

Основные временные характеристики переменного тока представлены в таблицах 112 – 116. Все характеристики измеряются с определенной нагрузкой и с выходным потоком данных с уровнем 4.

Асинхронные данные для выводов nIRQ3–0, nDMAR3–0, FLAG3–0 и TMR0E представлены в таблице 111.

Таблица 111 – Характеристики асинхронного сигнала переменного тока

Обозначение	Описание	Ширина импульсов сигнала низкого уровня (мин)	Ширина импульсов сигнала высокого уровня (мин)
nIRQ3–0	Запрос прерывания	$2 \times t_{SOCCLK}$	$2 \times t_{SOCCLK}$
nDMAR3–0	Запрос DMA	$2 \times t_{SOCCLK}$	$2 \times t_{SOCCLK}$
FLAG3–0[1]	Вход FLAG3–0	$2 \times t_{SOCCLK}$	$2 \times t_{SOCCLK}$
TMR0E[2]	Срабатывание таймера 0	-	$4 \times t_{SOCCLK}$

[1] Выходные характеристики на выводах FLAG3–0 представлены в таблице 116.
[2] Этот вывод может являться конфигурационным входом.

Таблица 112 – Опорная частота – время цикла тактовых сигналов процессорного ядра (CCLK)

Параметр	Описание	Мин	Макс	Ед. изм.
t_{CCLK}	Время цикла тактовых сигналов процессорного ядра	2,2	26,7	нс

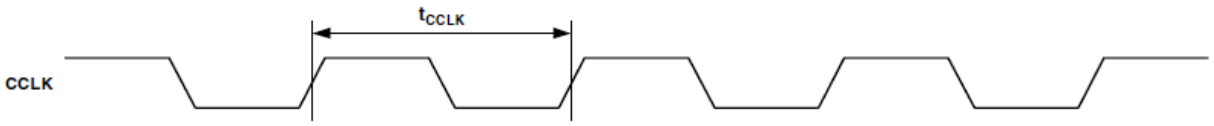


Рисунок 65 – Опорная частота – время цикла тактовых сигналов процессорного ядра (CCLK)

Таблица 113 – Опорная частота – время цикла тестовой синхронизации JTAG (TCK)

Параметр	Описание	Мин	Макс	Ед. изм.
T_{TCK}	Время цикла тестовой синхронизации (JTAG)	Наибольшее из 30 или t_{CCLK} × 4	—	нс
t_{TCKH}	Время цикла тестовой синхронизации сигналов высокого уровня (JTAG)	12	—	нс
t_{TCKL}	Время цикла тестовой синхронизации сигналов низкого уровня (JTAG)	12	—	нс

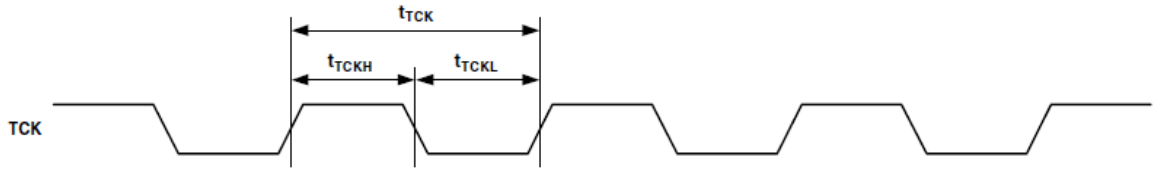


Рисунок 66 – Опорная частота—время цикла тестовой синхронизации JTAG (TCK)

Таблица 114 – Время сброса при включении питания

Параметр	Мин.	Макс.	Ед. изм.
Временные требования			
$t_{RST\_IN\_PWR}$ nRST_IN деактивирован после стабилизации V_{DD}, V_{DD_A}, V_{DD_IO}, SCLK и перехода в неизменное состояние статичных выводов/ конфигурационных входов	2		мс
$t_{TRST\_IN\_PWR}$[1] nTRST активирован во время сброса при включении питания	100 × t_{SCLK}		нс
Характеристики переключения			
$t_{RST\_OUT\_PWR}$ деактивация RST_OUT после деактивации nRST_IN	1,5		мс

[1] Применяется после того, как V_{DD}, V_{DD_A}, V_{DD_IO} и SCLK перейдут в стабильное состояние и ST_IN будет деактивирован.

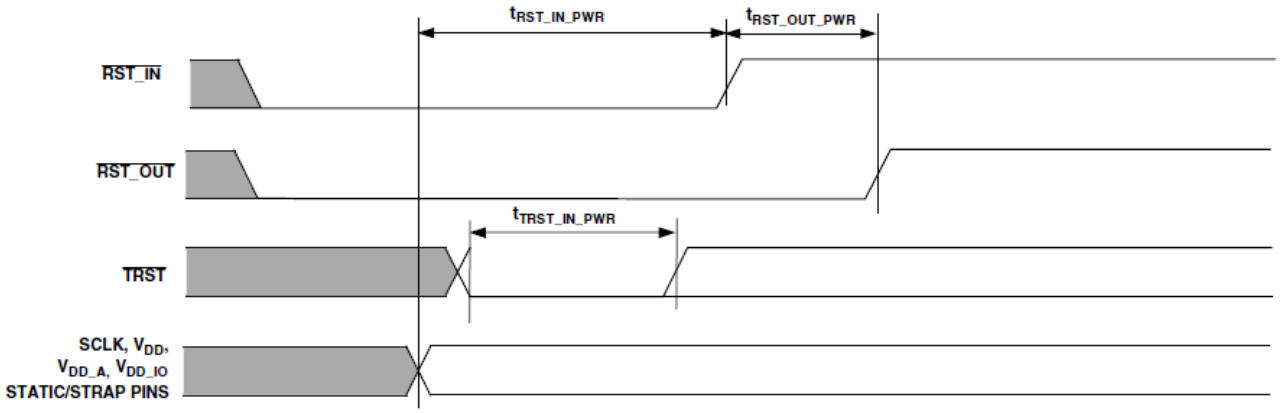


Рисунок 67 – Время сброса при включении питания

Таблица 115 – Время нормального сброса

Параметр	Мин.	Макс.	Ед. Изм.
Временные требования			
t_{RST_IN} nRST_IN активирован	2		мс
t_{STRAP} nRST_IN деактивирован после того, как конфигурационные входы будут в неизменном состоянии	1,5		мс
Характеристики переключения			
t_{RST_OUT} RST_OUT деактивирован после того, как деактивирован nRST_IN	1,5		мс

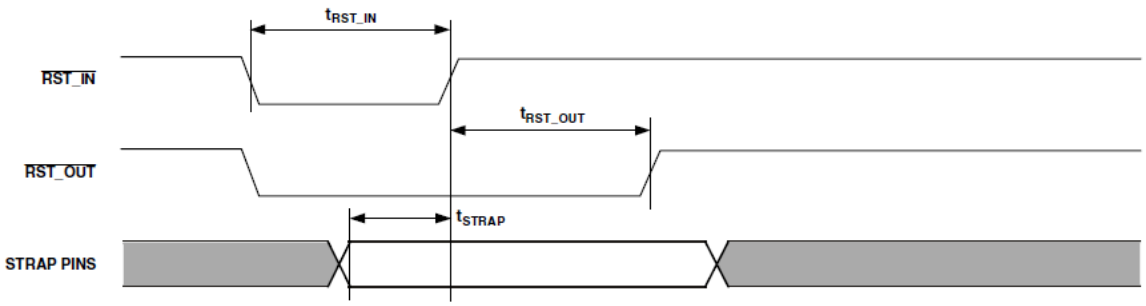


Рисунок 68 – Время нормального сброса

Таблица 116 – Характеристики сигнала переменного тока (все значения в таблице приведены в наносекундах)

Имя	Описание	Время предустановки входного сигнала (мин)	Время удержания входного сигнала(Макс)	Время установки выходного сигнала (макс)	Время удержания выходного сигнала (мин)	Время включения буфера выходного сигнала(мин)1	Время выключения буфера выходного сигнала (макс)1	Опорный тактовый сигнал
ADDR31–0	Внешняя адресная шина	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
DATA63–0	Внешняя шина данных	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
nMSH	Линия выбора памяти HOST	—	—	4.0	1.0	1.15	2.0	SCLK
nMSSD3–0	Линии выбора памяти SDRAM	1.5	0.5	4.0	1.0	1.0	2.0	SCLK
nMS1–0	Выбор памяти для статических блоков	—	—	4.0	1.0	1.15	2.0	SCLK
nRD	Чтение памяти	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
nWRL	Запись младшего слова	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
nWRH	Запись старшего слова	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
ACK	Подтверждение данных от высокого к низкому уровню Подтверждение данных от низкого к высокому уровню	1.5 1.5	0.5 0.5	3.6 4.2	1.0 0.9	1.15 1.15	2.0 2.0	SCLK SCLK
SDCKE	Включение тактового генератора SDRAM	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
nRAS	Выбор адресной строки	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
nCAS	Выбор адресного столбца	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
nSDWE	Включение записи SDRAM	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
LDQM	Маска данных младшего слова SDRAM	—	—	4.0	1.0	1.15	2.0	SCLK
HDQM	Маска слова старшего слова SDRAM	—	—	4.0	1.0	1.15	2.0	SCLK
SDA10	SDRAM ADDR10	—	—	4.0	1.0	1.15	2.0	SCLK
nHBR	Запрос шины хостом	1.5	0.5	—	—	—	—	SCLK
nHBG	Разрешение захвата шины хостом	1.5	0.5	4.0	1.0	1.15	2.0	SCLK
nBOFF	Запрос хоста на прекращение транзакции	1.5	0.5	—	—	—	—	SCLK
nBUSLOCK	Захват шины	—	—	4.0	1.0	1.15	2.0	SCLK
nBRST	Вывод импульса	1.5	0.5	4.0	1.0	1.15	2.0	SCLK

nBR7–0	Выводы запроса мультипроцессорной шины	1.5	0.5	4.0	1.0	—	—	SCLK
nBM	Помощь отладки контроллера шины	—	—	4.0	1.0	—	—	SCLK
nIORD	Вывод чтения I/O	—	—	4.0	1.0	1.0	2.0	SCLK
nIOWR	Вывод записи I/O	—	—	4.0	1.0	1.15	2.0	SCLK
nIOEN	Вывод включения I/O	—	—	4.0	1.0	1.15	2.0	SCLK
nCPA	Приоритетный доступ ядра от высокого к низкому Приоритетный доступ ядра от низкого к высокому	1.5 1.5	0.5 0.5	4.0 29.5	1.0 2.0	0.75 0.75	2.0 2.0	SCLK SCLK
nDPA	Приоритетный доступ DMA от высокого к низкому Приоритетный доступ DMA от низкого к высокому	1.5 1.5	0.5 0.5	4.0 29.5	1.0 2.0	0.75 0.75	2.0 2.0	SCLK SCLK
nBMS	Выбор загрузочной памяти	—	—	4.0	1.0	1.15	2.0	SCLK
FLAG3–0[2]	Выводы FLAG	—	—	4.0	1.0	1.15	2.0	SCLK
nRST_IN [3, 4]	Вывод глобального сброса	1.5	2.5	—	—	—	—	SCLK[5]
TMS	Выбор тестового режима (JTAG)	1.5	0.5	—	—	—	—	TCK
TDI	Вход тестовых данных (JTAG)	1.5	0.5	—	—	—	—	TCK
TDO	Выход тестовых данных (JTAG)	—	—	4.0	1.0	0.75	2.0	TCK [6]
nTRST [3, 4]	Тестовый сброс(JTAG)	1.5	0.5	—	—	—	—	TCK
nEMU [7]	Эмуляция от высокого к низкому	—	—	5.5	2.0	1.15	4.0	TCK или SCLK
ID2–0 [8]	Статические выводы – должны быть постоянными	—	—	—	—	—	—	—
STRAP SYS [9, 10]	Конфигурационные входы	1.5	0.5	—	—	—	—	SCLK
JTAG SYS [11, 12]	Системные выводы JTAG	+2.5	+10.0	+12.0	–1.0	—	—	TCK

[1] Протоколы обмена внешнего порта реализуют циклы шины IDLE для передачи управления шиной. Конфликты на шине при включении/выключении драйверов невозможны по причине того, что текущий драйвер гарантированно выключится раньше, чем включится другой драйвер.

[2] Входные характеристики выводов FLAG3–0 см в таблица 111.

[3] Эти входные выводы являются асинхронными, и поэтому нет необходимости синхронизации с опорной частотой.

[4] Подробнее см. в разделе «Сброс и запуск».

[5] Данные для nRST_IN приведены относительно среза SCLK.

[6] Данные для TDO приведены относительно среза TCK.

[7] Частота, относительно которой приведены данные, зависит от функции вывода в тот или иной момент.

[8] Эти выводы могут изменяться только во время сброса; рекомендуется подключать их к VDD_IO/VSS.

[9] Выводы STRAP SYS включают: nBMS, nBM, nBUSLOCK, TMR0E.

[10] Характеристики применимы только во время сброса.

[11] Системные выводы JTAG SYS (режим boundary scan) включают в себя: nRST_IN, RST_OUT, nIRQ3–0, nDMAR3–0, nHBR, nBOFF, MS1–0, nMSH, SDCKE, LDQM, HDQM, nBMS, nIOWR, nIORD, nBM, nEMU, SDA10, nIOEN, nBUSLOCK, TMR0E, DATA63–0, ADDR31–0, nRD, nWRL, nWRH, nBRST, nMSSD3–0, nRAS, nCAS, nSDWE, nHBG, BR7–0, FLAG3–0, L0DATOP3–0, L0DATON3–0, L1DATOP3–0, L1DATON3–0, L2DATOP3–0, L2DATON3–0, L3DATOP3–0, L3DATON3–0, L0CLKOUTP, L0CLKOUTN, L1CLKOUTP, L1CLKOUTN, L2CLKOUTP, L2CLKOUTN, L3CLKOUTP, L3CLKOUTN, L0ACKI, L1ACKI, L2ACKI, L3ACKI, L0DATIP3–0, L0DATIN3–0, L1DATIP3–0, L1DATIN3–0, L2DATIP3–0, L2DATIN3–0, L3DATIP3–0, L3DATIN3–0, L0CLKINP, L0CLKINN, L1CLKINP, L1CLKINN, L2CLKINP, L2CLKINN, L3CLKINP, L3CLKINN, L0ACKO, L1ACKO, L2ACKO, L3ACKO, ACK, nCPA, nDPA, nL0BCMPO, nL1BCMPO, nL2BCMPO, nL3BCMPO, nL0BCMPI, nL1BCMPI, nL2BCMPI, nL3BCMPI, ID2–0, CTRL_IMPD1–0.

[12] Опорная частота времени системного выходного сигнала JTAG является убывающим фронтом TCK.

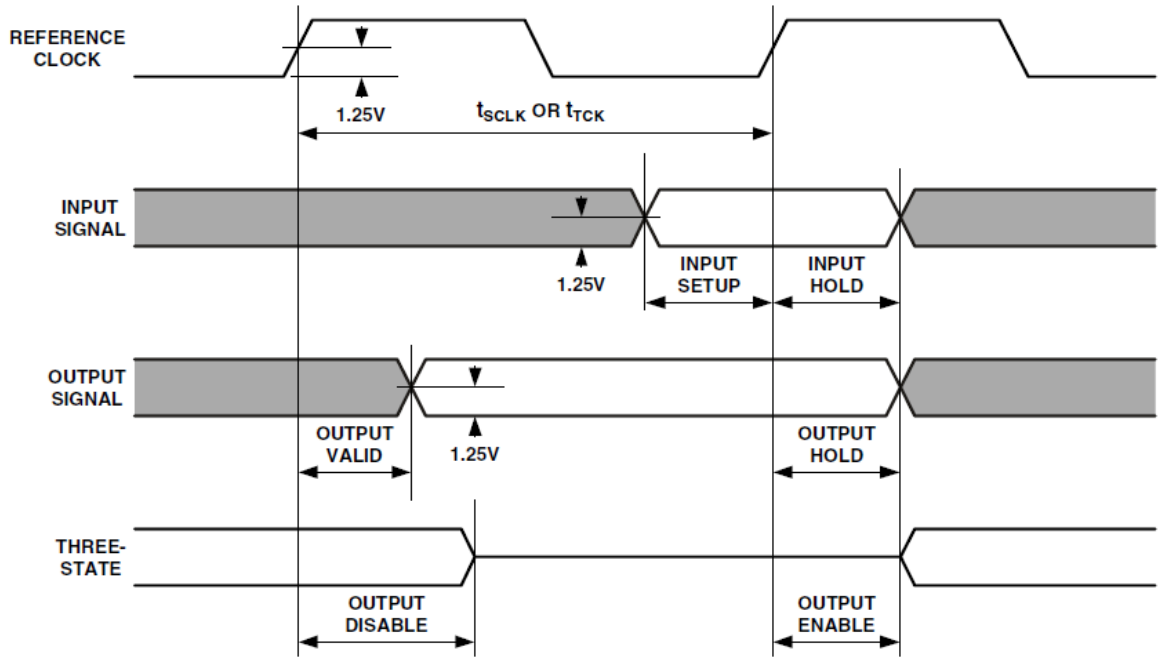


Рисунок 69 – Общие временные параметры переменного тока

8.2 LVDS-интерфейс, электрические и временные характеристики

Ниже представлены электрические характеристики портов связи LVDS (таблица 117, таблица 118, рисунок 70). Все параметры сигналов LVDS определяются при значениях V_{OD} = 0 В и используют обозначения сигналов без индексов N (negative) и P (positive), см. рисунок 71.

Таблица 117 – Электрические характеристики передачи порта связи LVDS

Параметр	Описание	Режим измерения	Мин.	Макс.	Ед. изм.
VOH	Высокий уровень выходного напряжения, VO_P или VO_N	RL = 100 Ом		1,85	В
VOL	Низкий уровень выходного напряжения, VO_P или VO_N	RL = 100 Ом	0,92		В
|VOD|	Выходное дифференциальное напряжение (модуль)	RL = 100 Ом	300	650	мВ
IOS	Выходной ток короткого замыкания	VO_P или VO_N = 0 В		+5/– 55	мА
		VOD = 0 В		±10	мА
VOCM	Выходное синфазное напряжение		1,20	1,50	В

Таблица 118 – Электрические характеристики приема порта связи LVDS

Параметр	Описание	Режим измерения	Мин.	Макс.	Ед. Изм.
|VID|	Входное дифференциальное напряжение (модуль)	t_{LDIS}/t_{LDIH} ≥ 0,20 нс	250	850	мВ
		t_{LDIS}/t_{LDIH} ≥ 0,25 нс	217	850	мВ
		t_{LDIS}/t_{LDIH} ≥ 0,30 нс	206	850	мВ
		t_{LDIS}/t_{LDIH} ≥ 0,35 нс	195	850	мВ
VICM	Входное синфазное напряжение		0,6	1,57	В

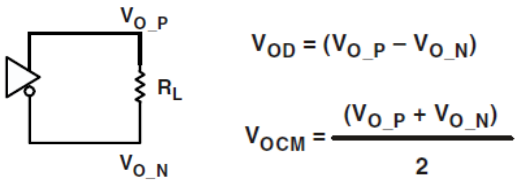
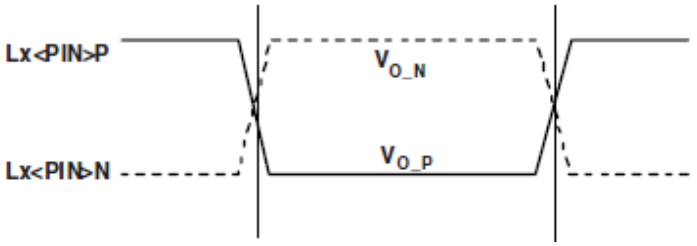


Рисунок 70 – Порт связи – электрические характеристики передачи

Колебание дифференциальной пары



Колебания дифференциального напряжения

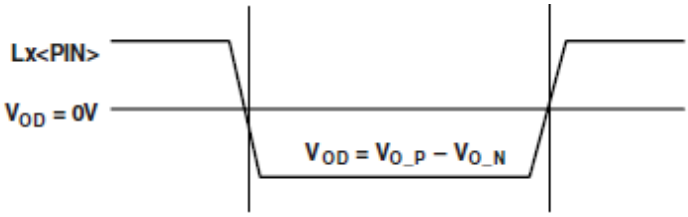


Рисунок 71 – Порт связи – обозначение сигналов

8.2.1 Порт связи – временные характеристики выходных сигналов

Ниже приведена информация о временных характеристиках выходных сигналов для портов связи LVDS (таблица 119, рисунок 72, рисунок 73, рисунок 74, рисунок 75, рисунок 76, рисунок 77).

Таблица 119 – Порт связи – временные характеристики вывода данных

Параметр	Описание	Мин.	Макс.	Ед. изм.
Выходы				
t_{REO}	Фронт сигнала (Рисунок 73)		350	пс
t_{FEO}	Срез сигнала (Рисунок 73)		350	пс
t_{LCLKOP}	Период LxCLKOUT (Рисунок 72)	Большее значение из 2,2 или 0,9 x LCR x t_{CCLK}[1,2]	Меньшее значение из 12,5 или 1,1 x LCR x t_{CCLK}[1,2]	нс
t_{LCLKOH}	Высокий уровень сигнала LxCLKOUT (Рисунок 72)	0,4 x t_{LCLKOP}[1]	0,6 x t_{LCLKOP}[1]	нс
t_{LCLKOL}	Низкий уровень сигнала LxCLKOUT (Рисунок 72)	0,4 x t_{LCLKOP}[1]	0,6 x t_{LCLKOP}[1]	нс
t_{COJT}	Джиттер LxCLKOUT (Рисунок 74)		±0,005 x t_{LCLKOP}[3]	пс
t_{LDOS}	Время предустановки выходного сигнала LxDATO (Рисунок 74)	0,25 x LCRxt_{CCLK} – 0,10 x t_{CCLK}[1,4,8] 0,25 x LCR x t_{CCLK} – 0,15 x t_{CCLK}[1,5,6,8] 0,25 x LCR x t_{CCLK} – 0,30 x t_{CCLK}[1,7,8]		нс нс нс
t_{LDOH}	Время удержания выходного сигнала LxDATO (Рисунок 74)	0,25 x LCRxt_{CCLK} - 0,10 x t_{CCLK}[1,4,8] 0,25 x LCR x t_{CCLK} - 0,15 x t_{CCLK}[1,5,6,8] 0,25 x LCR x t_{CCLK} - 0,30 x t_{CCLK}[1,7,8]		нс нс нс
t_{LACKID}	Время от фронта LxACKI до фронта синхроимпульса первой транзакции (Рисунок 75)		16 x LCR x t_{CCLK}[1,2]	нс
t_{BCMPOV}	Время установки сигнала nLxBCMPO (Рисунок 75)		2 x LCR x t_{CCLK}[1,2]	нс
t_{BCMPOH}	Время удержания сигнала nLxBCMPO (Рисунок 76)	3 x TSW-0.5[1,9]		нс

Параметр	Описание	Мин.	Макс.	Ед. изм.
Входы				
t_{LACKIS}	Время предустановки низкого уровня сигнала LxACKI для гарантии того, что передатчик прекратил передачу. (Рисунок 76) Время предустановки высокого уровня сигнала LxACKI для гарантии того, что передатчик продолжает передачу без прерываний (Рисунок 77)	16 x LCR x t_{CCLK}[1,2]		нс
t_{LACKIH}	Время удержания высокого уровня сигнала LxACKI (Рисунок 77)	0,51		

1 Измерения в моменты времени с дифференциальным напряжением, равным 0 (V_{OD} = 0).
2 LCR = 1, 1.5, 2, или 4. T_{CCLK} период частоты ядра CCLK.
3 Cycly-to-cycle джиттер.
4 LCR = 1.
5 LCR = 1,5.
6 LCR = 2.
7 LCR = 4.
8 Значения t_{LDOS} и t_{LDOH} включают джиттер LCLKOUT.
9 TSW это время передачи слова минимальной длины. Для 4-битной конфигурации равно 2xLCRxt_{CCLK}. Для 1-битной конфигурации равно 8xLCRxt_{CCLK} нс.

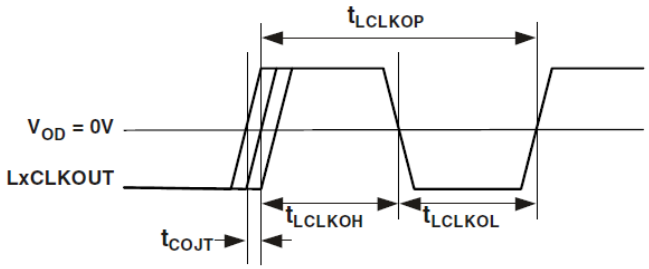


Рисунок 72 – Порты связи – выходной тактовый сигнал

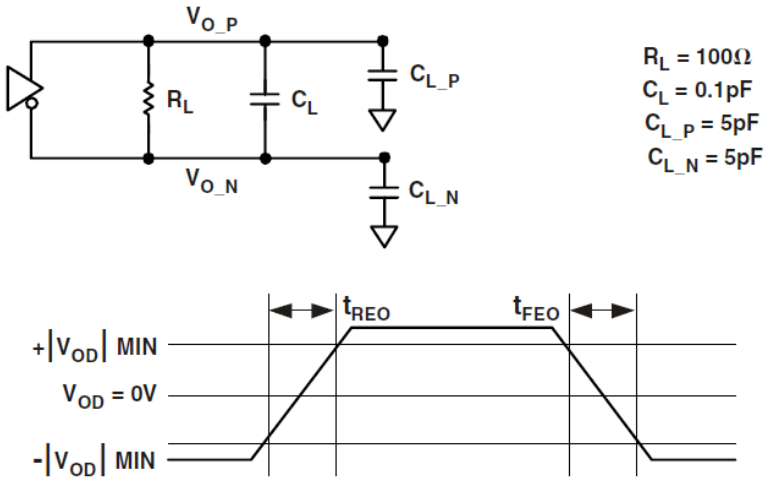


Рисунок 73 – Порты связи – время фронта/среза дифференциальных выходных сигналов

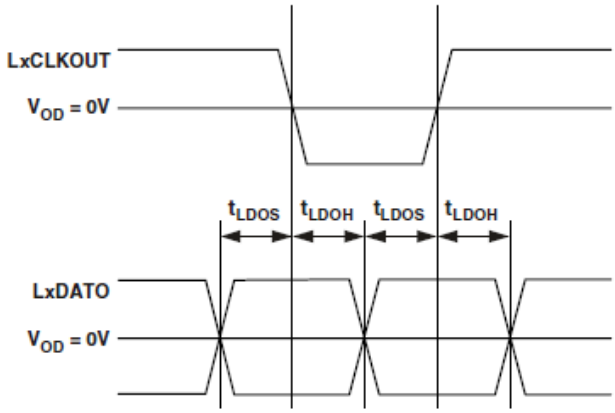


Рисунок 74 – Порты связи – время предустановки и удержания выходного сигнала данных[1]

[1] Параметры действительны как по отношению к фронту, так и по отношению к срезу тактового сигнала.

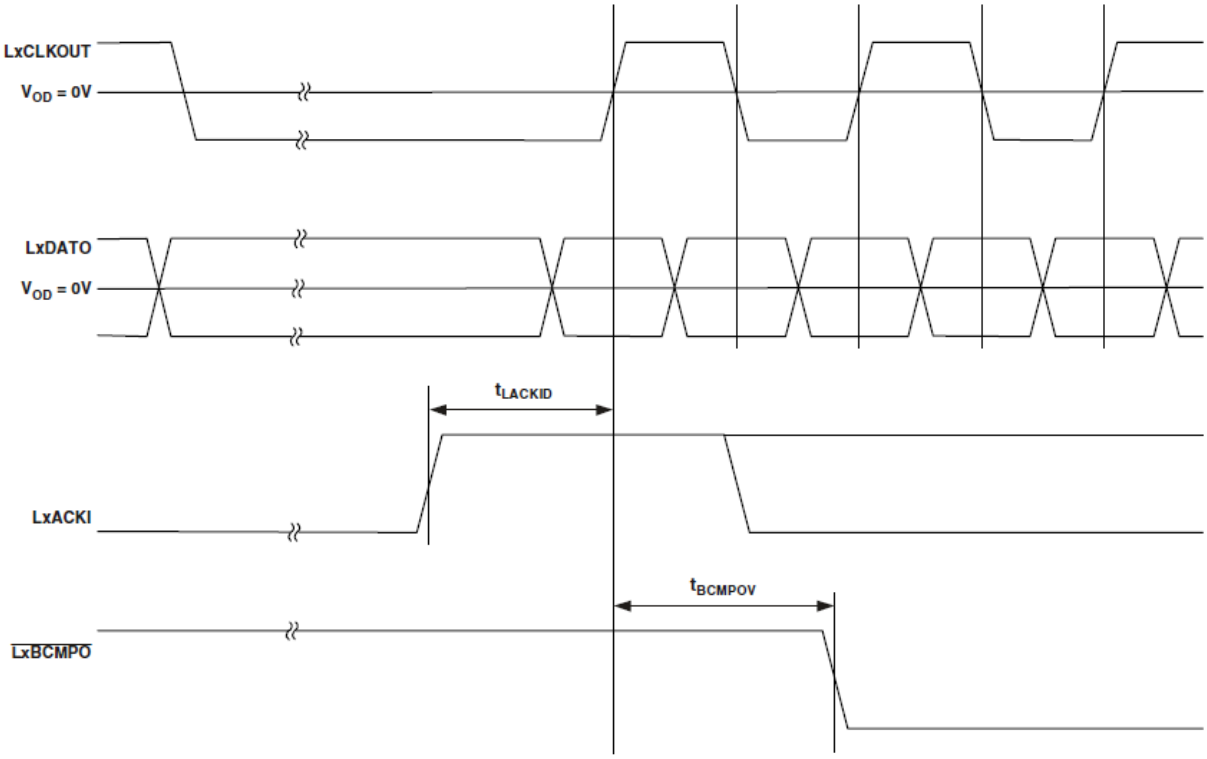


Рисунок 75 – Порты связи – начало передачи

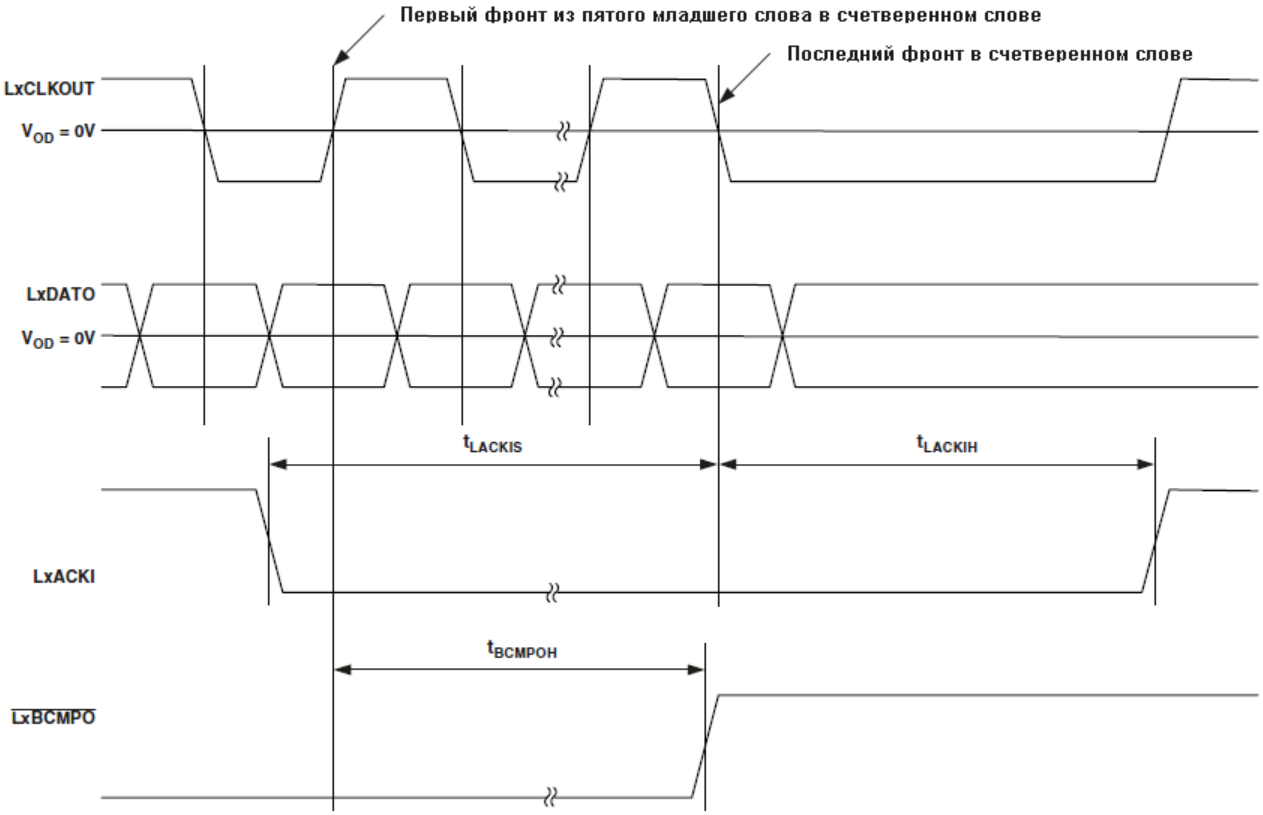


Рисунок 76 – Порты связи – окончание блочной передачи

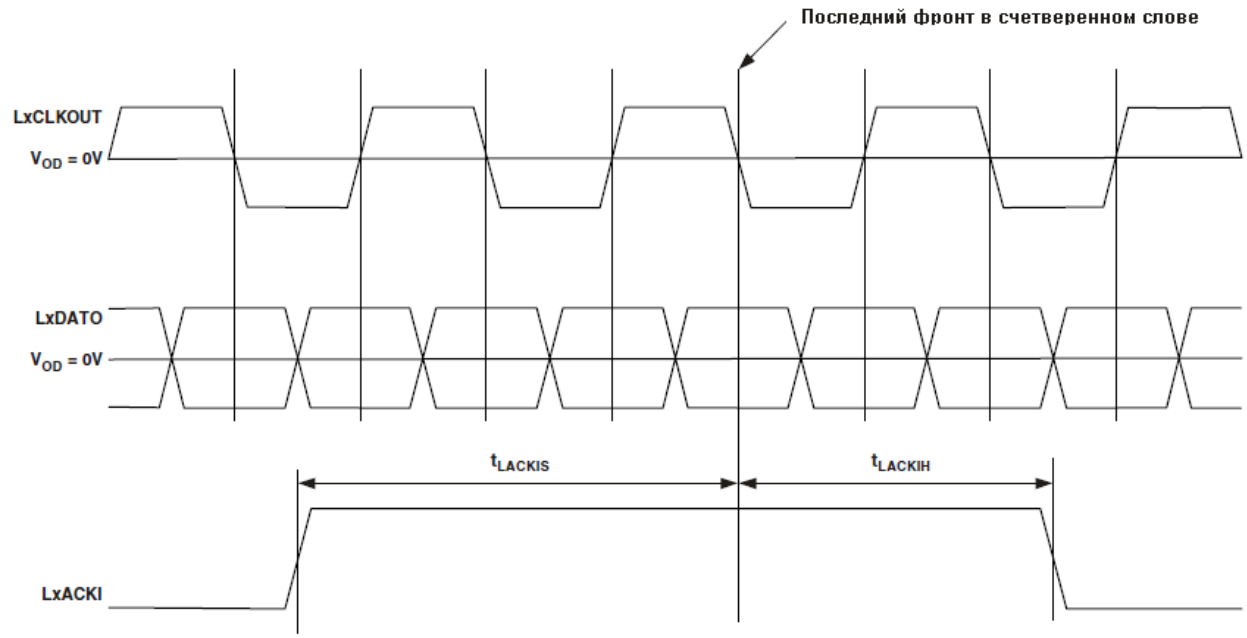


Рисунок 77 – Порт связи – временные характеристики сигнала LxACKI

8.2.2 Порт связи – временные характеристики ввода данных

Ниже приведена информация о временных характеристиках ввода данных для портов связи LVDS (таблица 120 и рисунок 78, рисунок 79).

Таблица 120 – Порт связи – временные характеристики данных на входе

Параметр	Описание	Мин.	Макс.	Ед. изм.
t_{LDIS}	Время предустановки входного сигнала LxDATI (Рисунок 79)	0,20[1, 2]		нс
		0,25[1, 3]		нс
		0,30[1, 4]		нс
		0,35[1, 5]		нс
t_{LDIH}	Время удержания входного сигнала LxDATI (Рисунок 79)	0,201, 2		нс
		0,25[1, 3]		нс
		0,30[1, 4]		нс
		0,35[1, 5]		нс
t_{LCLKIP}	Период LxCLKIN	Большее значение из 2,2 или $0,9 \cdot t_{CCLK}$	12,5	нс
t_{BCMPIS}	Установка nLxBCMPI (Рисунок 78)	$2 \times t_{LCLKIP}$[1]		нс
t_{BCMPIH}	Удержание nLxBCMPI (Рисунок 78)	$2 \times t_{LCLKIP}$[1]		нс

[1] Измерения в моменты временис с дифференциальным напряжением, равным 0 (VOD = 0).
[2] |VID| = 250 мВ.
[3] |VID| = 217 мВ.
[4] |VID| = 206 мВ.
[5] |VID| = 195 мВ.

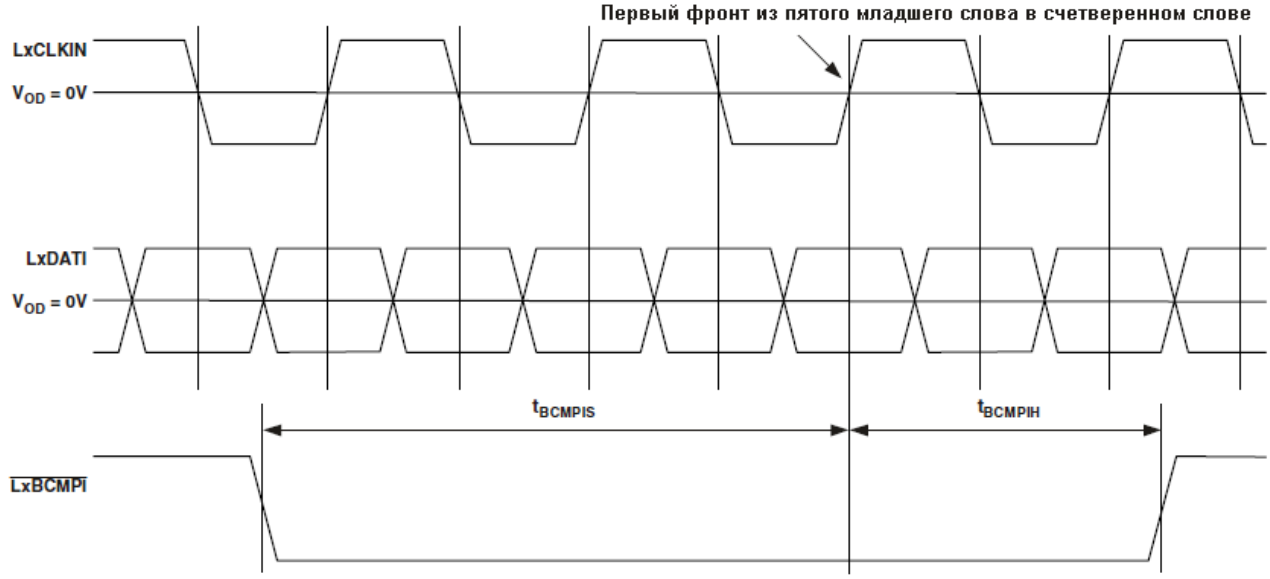


Рисунок 78 – Порты связи – последнее полученное счетверенное слово

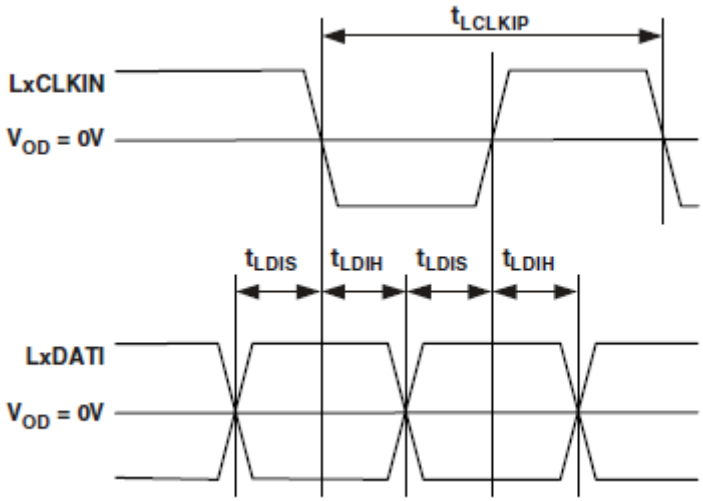


Рисунок 79 – Порты связи – время предустановки и удержания входного сигнала данных[1]

[1] Параметры действительны для обоих фронтов тактового сигнала.

9 Методы загрузки процессора

После сброса процессор имеет четыре опции загрузки кода программы для начала работы:

- загрузка из EPROM;
- загрузка хост-процессором;
- загрузка через порты связи;
- отсутствие загрузки.

Выбор загрузки из EPROM или ее отсутствие выбирается системой с помощью вывода nBMS. После того, как режим загрузки выбран, процессор загружает (в ведущем режиме) или загружается (в подчиненном режиме) данными загрузки объемом 256 слов. Обычно данные загрузки содержат ядро загрузки для выбранного режима. После завершения первоначальной загрузки ядро загрузки начинает работу, загружая программу разработчика в процессор и начиная выполнение этой программы.

В Таблице 121 приведен перечень внешних выводов процессора, значение которых во время сброса используется для конфигурирования процессора.

Таблица 121 – Внешние конфигурационные выводы

Вывод	Значение во время сброса	Выполняемая функция
nBMS	0(низкий)	Выбирает режим загрузки из внешнего EPROM
	1(высокий)	Отключает возможность использования внешнего EPROM для стартовой загрузки
nBM	0(низкий)	Запрещает прием прерываний по входам nIRQ3:0
	1(высокий)	Разрешает прием запросов прерываний по входам nIRQ3:0 по уровню сигнала (активный низкий)
TMR0E	0(низкий)	Выбор ширины шины каналов портов связи 1 бит
	1(высокий)	Выбор ширины шины каналов портов связи 4 бита
nBUSLOCK	0(низкий)	Однократная запись в регистры SYSCON, SDRCON
	1(высокий)	Многократная запись в регистры SYSCON, SDRCON

Выводы, приведенные в Таблице 121, при нормальном функционировании работают как выходы, но во время сброса выход отключен, и вывод работает как вход, позволяя записать конфигурационную информацию.

9.1 Выполнение операции загрузки процессора

После сброса ядро процессора находится в состоянии ожидания прерывания. Если поступит запрос прерывания, процессор начнет исполнение программы согласно вектору прерывания.

В случае загрузки из EPROM (вывод nBMS равен нулю во время сброса), канал 0 ДМА выполняет загрузку кода из внешнего EPROM во внутреннюю память и посылает запрос прерывания с адрес-вектором равным 0.

Хост-процессор может загрузить программу во внутреннюю память процессора, а затем, используя векторное прерывание, запустить процессор на исполнение кода.

Внешнее устройство может загрузить 256 слов кода через порт связи, после чего соответствующий приемнику порта связи канал ДМА сформирует запрос прерывания к процессору с адрес-вектором 0.

Если ничего описанного выше не происходит, то может возникнуть внешнее прерывание по одному из входов nIRQ3-0. Каждому входу запроса прерывания соответствует вектор во внешней или внутренней памяти. Процессор может начать исполнение кода из внешней памяти.

9.2 Загрузка EPROM

Загрузка процессора из EPROM является загрузочным режимом, в котором процессор начинает и контролирует процесс пересылки данных.

Для того, чтобы сконфигурировать процессор для загрузки из EEPROM необходимо, чтобы во время сброса на внешнем выводе nBMS был низкий уровень. По окончании сброса процессор защелкивает состояние внешнего вывода nBMS во внутреннем флаге. Это значение доступно в регистре SYSTAT. Если процессор сконфигурирован под загрузку из EPROM, nBMS является активным во время последовательности загрузки и должен подключаться к сигналу выбора кристалла EPROM.

При выборе режима загрузки EPROM процессор инициализирует канал 0 внешнего порта DMA для передачи 256-ти 32-разрядных слов кода из загрузочного EPROM в ячейки 0x00-0xFF блока внутренней памяти 0 процессора. Соответствующий вектор прерывания (для DMA канала 0) инициализируется по адресу 0x00 во внутренней памяти. После завершения работы DMA, процессор начинает выполнение программы из адреса ячейки 0x00. Эти 256 слов кода действуют в качестве загрузчика для инициализации остальной памяти процессора.

9.3 Загрузка хоста

Загрузка процессора хост-процессором является режимом, в котором процессор ожидает пока внешнее устройство начнет загрузку кода в его внутреннюю память. Любое ведущее устройство на кластерной шине может загружать процессор через запись в его внутреннюю память или посредством autoDMA.

Для того, чтобы перевести процессор в режим загрузки хоста, необходимо поместить нагрузочный повышающий резистор между nBMS и VDD_IO. Тогда во время сброса значение nBMS будет равно 1, что укажет на отсутствие загрузки из EPROM.

Когда выбран режим загрузки хоста или порта связи, процессор переходит в нерабочее состояние после сброса, ожидая пока хост-процессор или порт связи осуществит загрузку. Загрузка хоста может использовать регистры AUTODMA процессора (оба канала), из которых оба инициализированы для передачи 256 слов кода в блок 0 ячейки 0x00-0xFF внутренней памяти процессора. Соответствующий вектор прерывания инициализируется по адресу 0x00 во внутренней памяти. Таким образом, после завершения DMA, процессор начнет выполнение программы с адреса 0. Эти 256 слов кода действуют в качестве загрузчика для инициализации остальной памяти процессора.

9.4 Загрузка порта связи

Загрузка процессора из порта связи является загрузочным режимом, в котором процессор ожидает внешнее устройство, подключенное к порту связи, чтобы принять и поместить во внутреннюю память загрузочную программу.

Все четыре принимающих канала DMA портов связи инициализируются после сброса для передачи блока 256 слов в адреса внутренней памяти с 0 по 255 и для вызова прерывания после приема блока (аналогично каналу 0 DMA). Соответствующие прерывания этих каналов DMA устанавливаются по нулевому адресу. Эти 256 слов кода действуют в качестве загрузчика для инициализации остальной памяти процессора.

При выборе загрузки посредством порта связи имеет значение состояние на авыводе TMR0E во время сброса. Этот вывод позволяет выбрать ширину шины порта связи: 1 или 4 бита.

9.5 Отсутствие загрузки

Запуск процессора без загрузки является режимом, в котором процессор начинает исполнение программы с фиксированного адреса во внешней или внутренней памяти. Выбор стартового адреса осуществляется посредством подачи запроса прерывания по одной из линий nIRQ3-0.

Для конфигурации процессора в режиме без загрузки необходимо подключение вывода nBMS к высокому уровню посредством резистора. Также необходимо подключить высокий уровень к выводу nBM. Именно значение 1 на входе nBM во время сброса и разрешает анализ запросов прерывания по входам nIRQ3:0. Запрос прерывания осуществляется активный низким уровнем сигнала.

Каждый вход запроса прерывания имеет соответствующий ему вектор. Значение векторов прерывания приведено в Таблице 122. Как видно из таблицы, процессор может выполнить старт с адреса внешней памяти банков памяти MS0 или MS1, а также из хост-пространства. Возможен также старт из внутренней памяти с адреса 0.

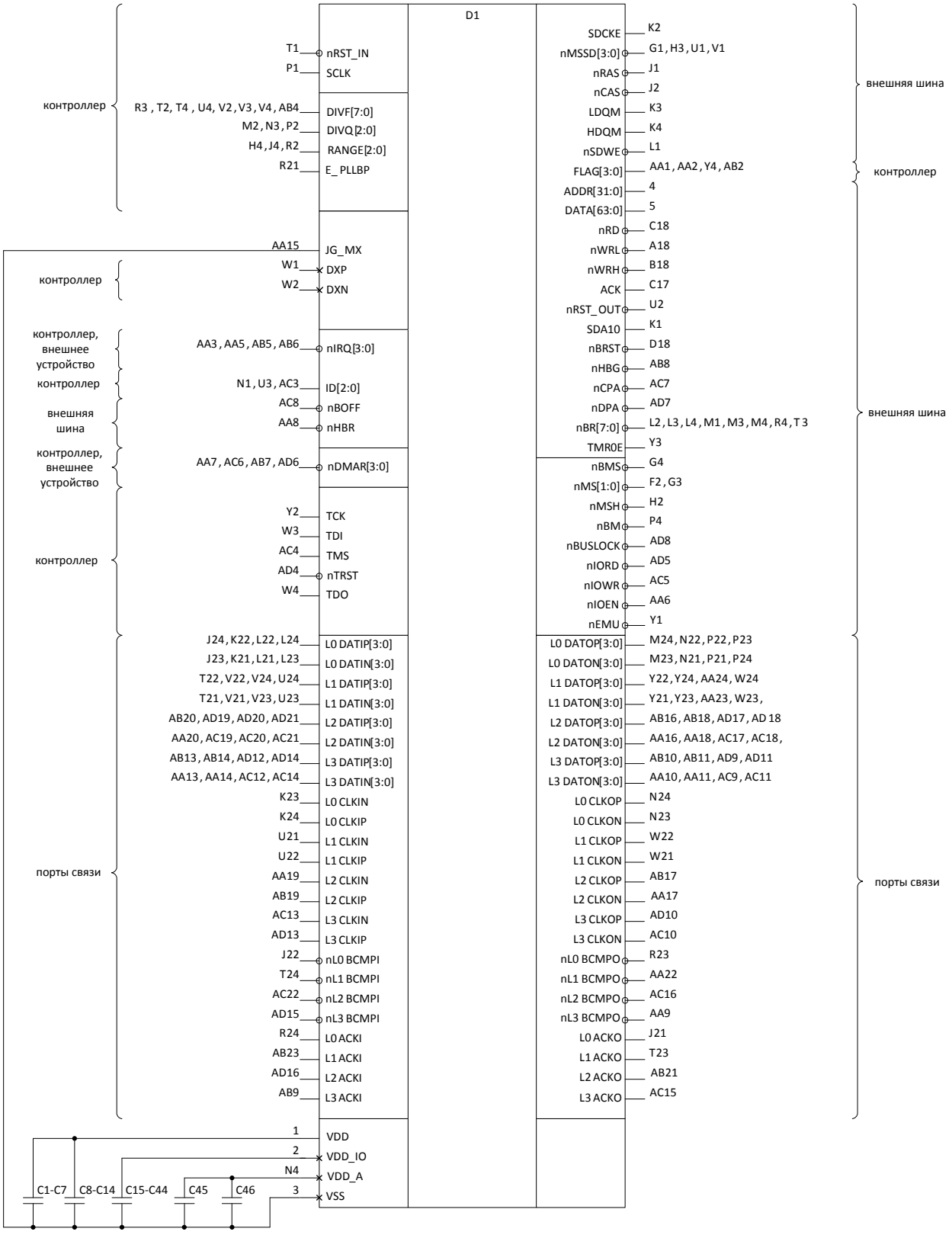
Таблица 122 – Векторы прерывания nIRQ3–0 после сброса

Прерывание	Адрес
nIRQ0	0x30000000 (MS0)
nIRQ1	0x38000000 (MS1)
nIRQ2	0x80000000 (nMSH)
nIRQ3	0x00000000 (внутренняя память)

Необходимо отметить, что выбор загрузки из EPROM (nBMS равен 0) не означает невозможность загрузки посредством портов связи или хостом, или генерации прерывания посредством nIRQ3:0. Загрузка из EPROM не запрещает других действий. Поэтому нужно внимательно относиться к процедуре сброса и избегать неоднозначности в вариантах начального старта процессора.

Следует иметь в виду, что порты связи после сброса по умолчанию готовы к приему данных. Поэтому при любом варианте загрузки, отличном от загрузки через порты связи, следует гарантировать отсутствие синхросигнала на приемниках портов связи.

10 Типовая схема включения



D1 – контролируемая микросхема;
C1 – C7, C15 – C44, C45 – конденсаторы емкостью 100 нФ;
C8 – C14, C46 – конденсаторы емкостью 10 нФ

Рисунок 80 – Типовая схема включения микросхем

11 Типовые зависимости

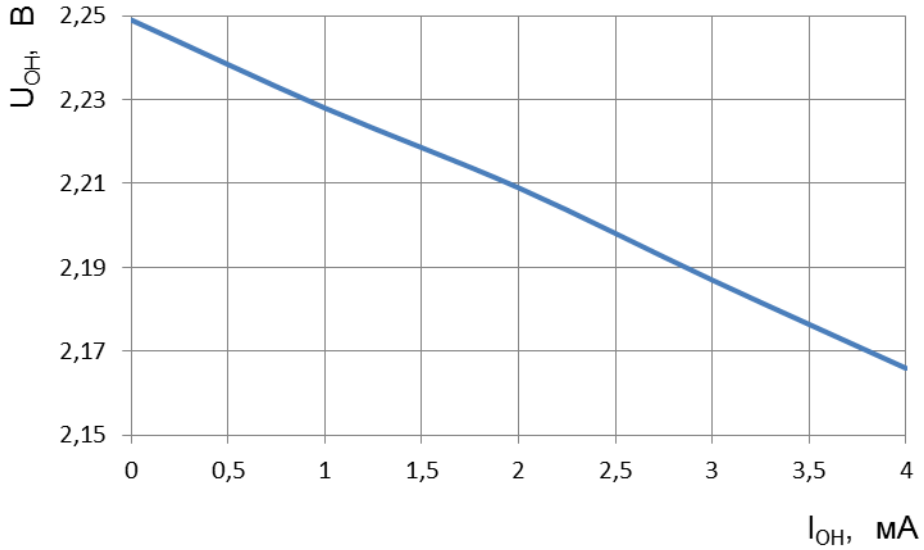


Рисунок 81 – Зависимость выходного напряжения высокого уровня U_{OH} от тока нагрузки при U_{CC}= 0,9 В, U_{CCA} =U_{CCIO} = 2,25 В, Т = 25 ºС

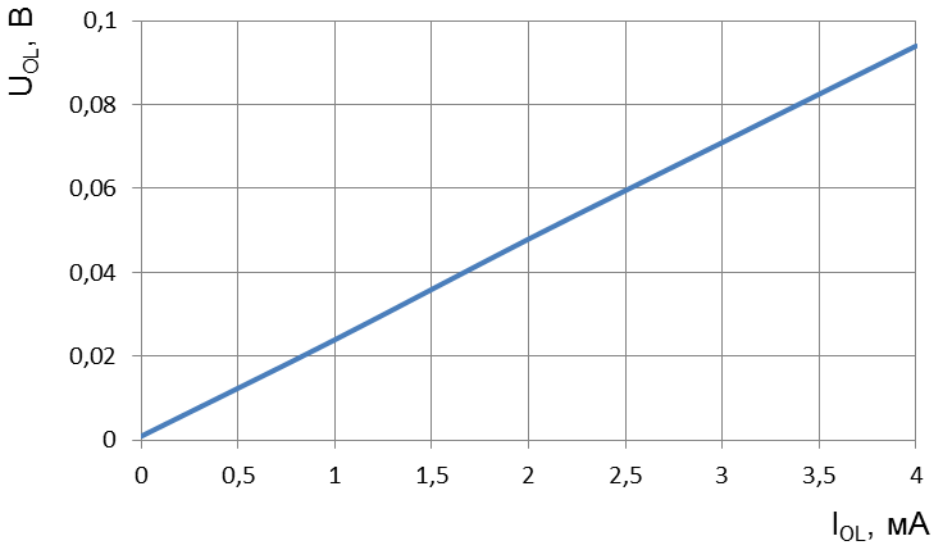


Рисунок 82 – Зависимость выходного напряжения низкого уровня U_{OL} от тока нагрузки при U_{CC}= 0,9 В, U_{CCA} =U_{CCIO} = 2,25 В, Т = 25 ºС

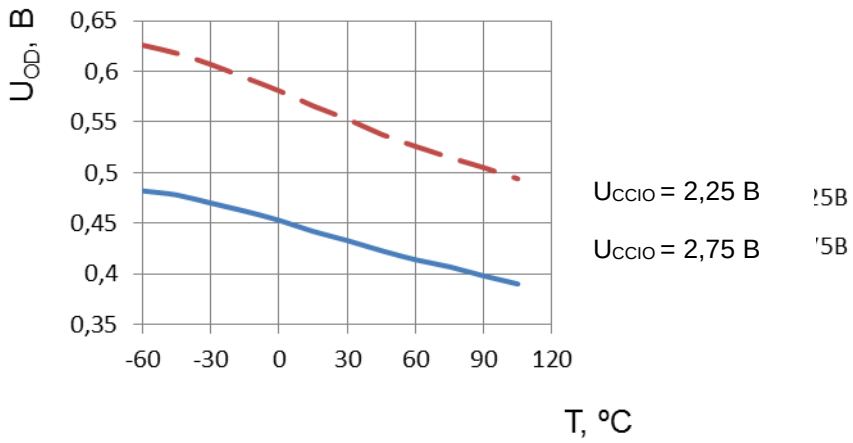


Рисунок 83 – Зависимость дифференциального выходного напряжения LINK-портов U_{OD} от температуры при R_L=100 Ом

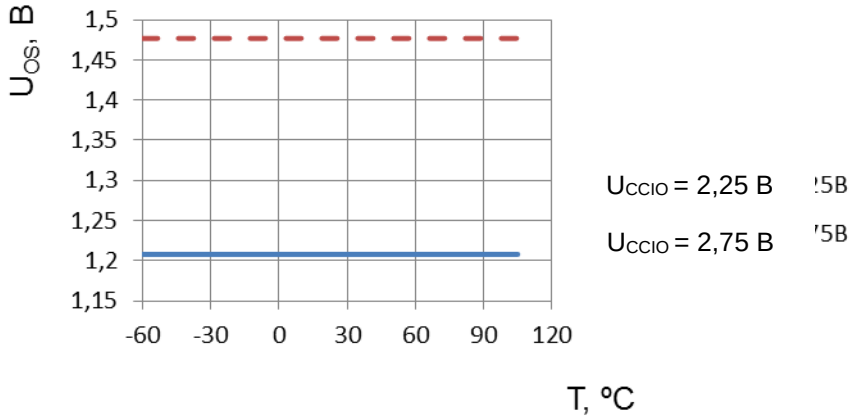


Рисунок 84 – Зависимость выходного синфазного напряжения LINK портов U_{OS} от температуры при R_L=100 Ом

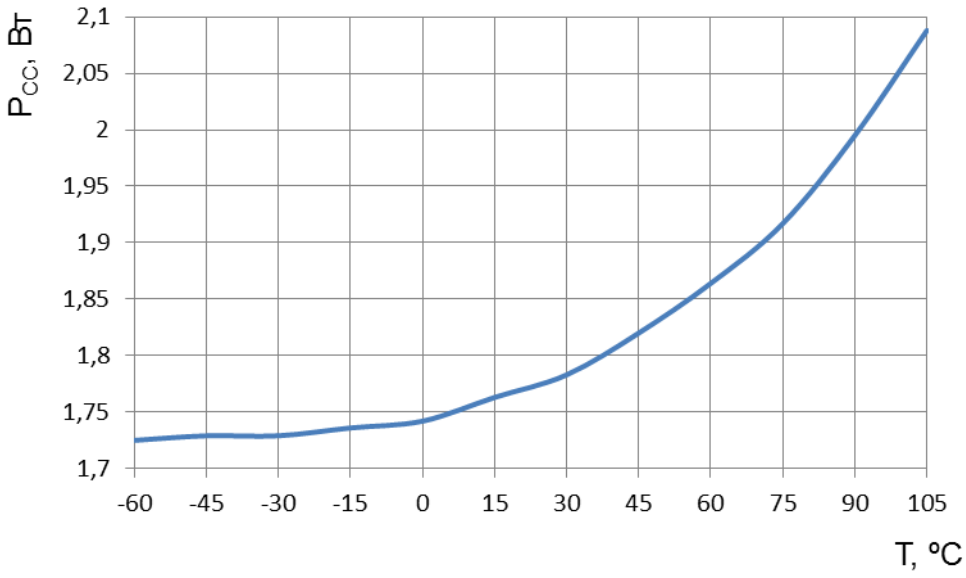


Рисунок 85 – Зависимость потребляемой мощности P_{CC} от температуры при U_{CC}= 1,1 В, U_{CCA} = U_{CCIO} = 2,75 В, f_C= 450 МГц, f_{C_PLL} = 100 МГц

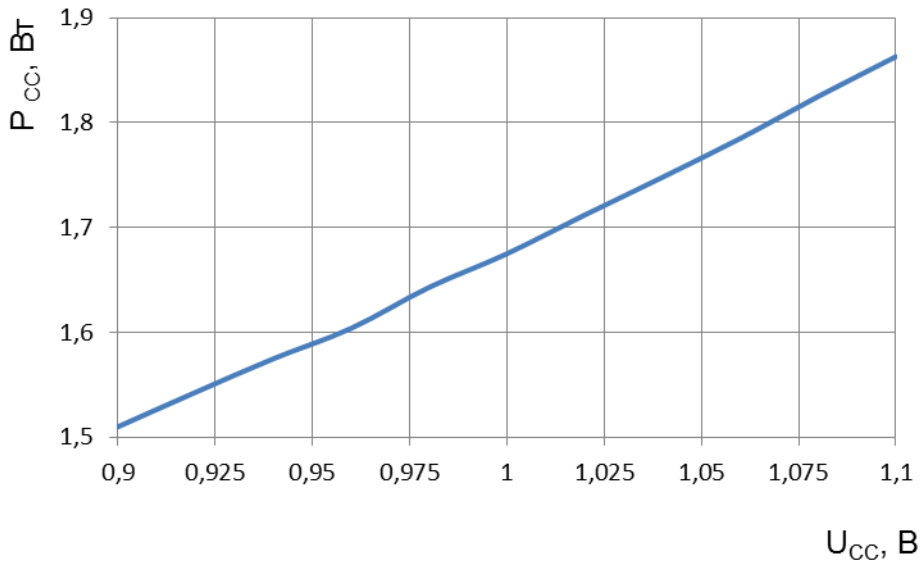


Рисунок 86 – Зависимость потребляемой мощности P_{CC} от напряжения питания ядра U_{CC} при $U_{CCA} = U_{CCIO} = 2,75$ В, $f_C = 450$ МГц, $f_{C\_PLL} = 100$ МГц, T = 25 ºC

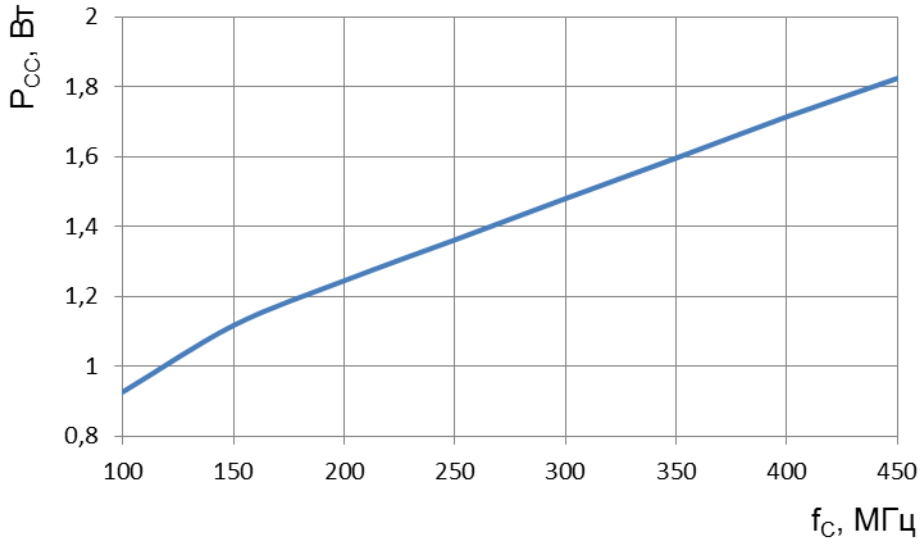


Рисунок 87 – Зависимость потребляемой мощности P_{CC} от тактовой частоты процессора f_C при $U_{CC} = 1,1$ В, $U_{CCA} = U_{CCIO} = 2,75$ В, $f_C = 450$ МГц, T = 25 ºC

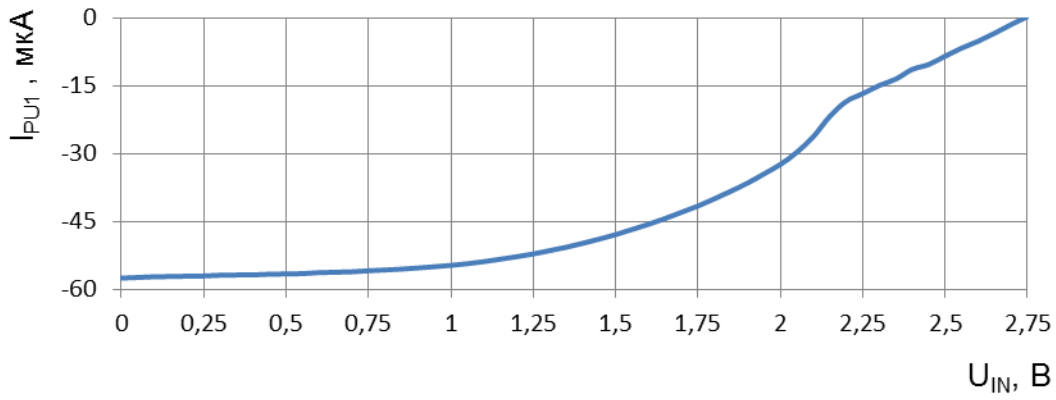


Рисунок 88 – Зависимость тока I_{PU1} от входного напряжения U_{IN} при T = 25 ºC

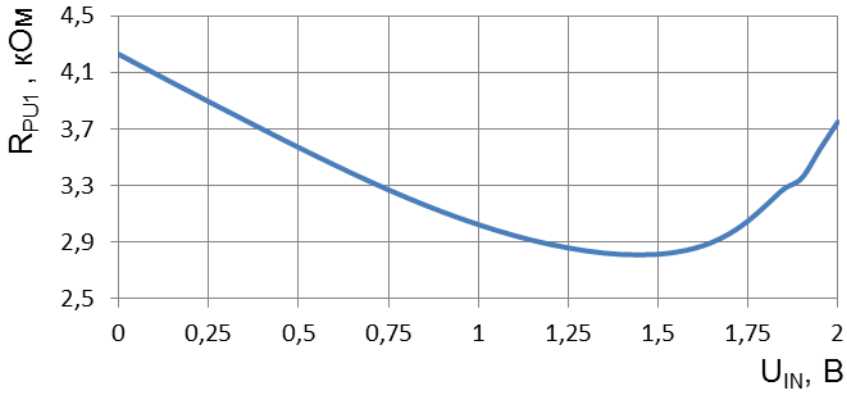


Рисунок 89 – Зависимость сопротивления подтягивающего к питанию резистора R_{PU1} от входного напряжения U_{IN} при U_{CC}= 1,1 В, U_{CCA} = U_{CCIO} = 2,75 В, U_{IL} = 0 В, T = 25 ºС

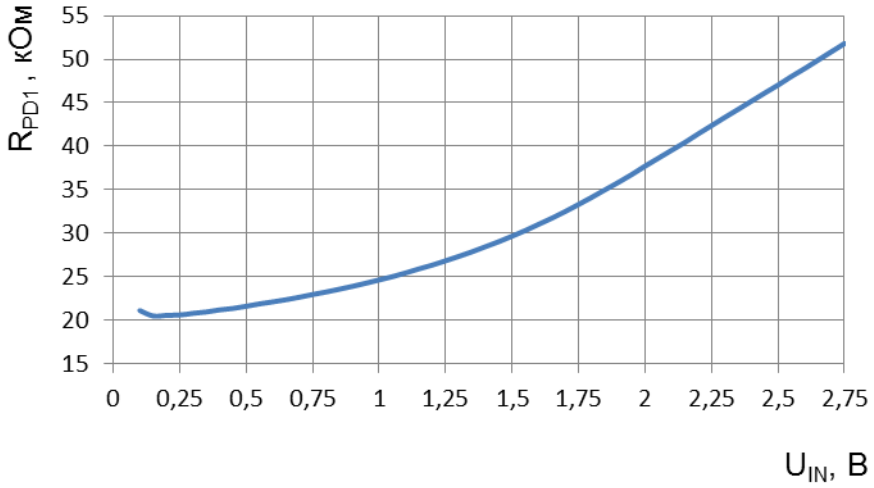


Рисунок 90 – Зависимость сопротивления подтягивающего к земле резистора R_{PD1} от входного напряжения U_{IN} при U_{CC}= 1,1 В, U_{CCA} = U_{CCIO} = U_{IH} = 2,75 В, T = 25 ºС

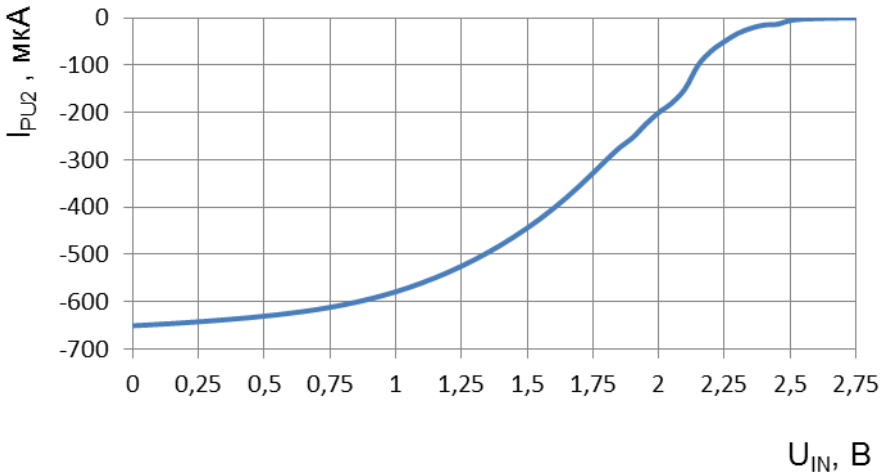


Рисунок 91 – Зависимость тока I_{PU2} от входного напряжения U_{IN} при температуре 25 ºС

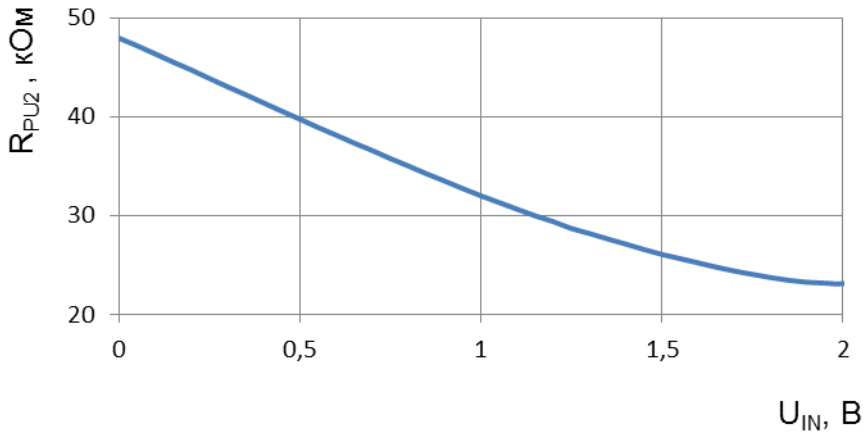


Рисунок 92 – Зависимость сопротивления подтягивающего к питанию резистора R_{PU2} от входного напряжения U_{IN} при U_{CC}= 1,1 В, U_{CCA} = U_{CCIO} = 2,75 В, U_{IL} = 0 В, T = 25 ºС

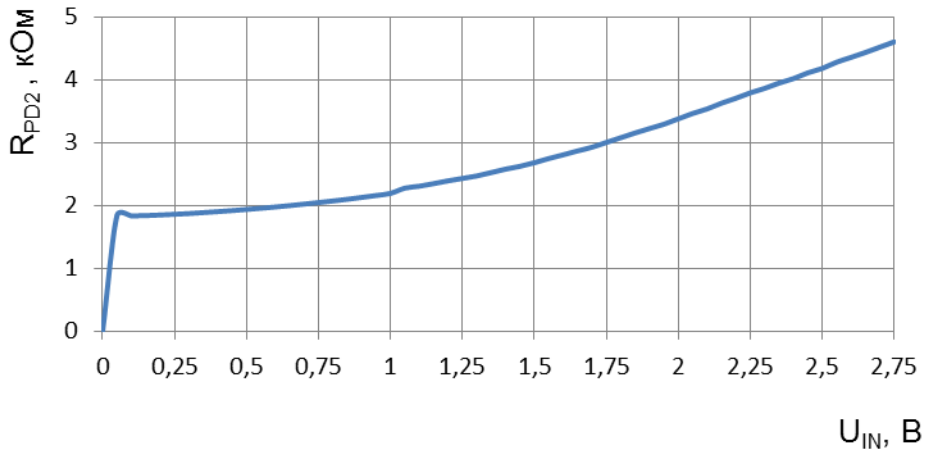


Рисунок 93 – Зависимость сопротивления подтягивающего к земле резистора R_{PD2} от входного напряжения U_{IN} при U_{CC}= 1,1 В, U_{CCA} = U_{CCIO} = U_{IH} = 2,75 В, T = 25 ºС

12 Предельно-допустимые режимы

Нагрузки, превышающие приведенные ниже значения, могут вызвать серьезное повреждение устройства. Предотвращение воздействия условий с максимально допустимыми значениями существенно повышает надежность устройства.

Таблица 123 – Предельно-допустимые режимы эксплуатации и предельные электрические режимы микросхем

Наименование параметра, единица измерения	Буквенное обозначение параметра	Норма параметра			
		Предельно-допустимый режим		Предельный режим	
		не менее	не более	не менее	не более
Напряжение питания ядра, В	U_{CC}	0,9	1,1	–	1,3
Напряжение питания ввода/вывода, В	U_{CCIO}	2,25	2,75	–	3,5
Напряжение питания аналоговых блоков, В	U_{CCA}	2,25	2,75	–	3,5
Входное напряжение высокого уровня, В	U_{IH}	1,7	3,6	–	3,9
Входное напряжения низкого уровня, В	U_{IL}	0	0,7	– 0,3	–
Входное дифференциальное напряжение LINK порта, В	U_{ID}	0,2	1,2	–	–
Входное синфазное напряжение LINK порта, В	U_{IS}	0,6	1,8	–	–
Выходной ток цифровых выходов высокого уровня, мА	I_{OH}	– 4	0	– 8	–
Выходной ток цифровых выходов низкого уровня, мА	I_{OL}	0	4	–	8
Тактовая частота процессора, МГц	f_C	–	450	–	–
Входная частота ФАПЧ, МГц	f_{C_PLL}	20	100	–	–
Тактовая частота LINK порта, МГц	f_{C_LINK}	–	450	–	–
Тактовая частота JTAG, МГц	f_{JTAG}	1	10	–	–
Примечание – Не допускается одновременное воздействие двух и более предельных режимов.					

13 Электрические параметры

Таблица 124 – Электрические параметры микросхем при приёмке и поставке

Наименование параметра, единица измерения, режим измерения	Буквенное обозначение параметра	Норма параметра		Температура среды (ккорпуса), °С
		не менее	не более	
Выходное напряжение высокого уровня, В	U_{OH}	2	–	25, (105), – 60
Выходное напряжение низкого уровня, В	U_{OL}	–	0,4	25, (105), – 60
Напряжение на термодиоде, мВ	U_{DO}	500	1000	25, (105), – 60
Дифференциальное выходное напряжение LINK портов, В	U_{OD}	0,3	0,8	25, (105), – 60
Синфазное выходное напряжения LINK портов, В	U_{OS}	1,1	1,6	25, (105), – 60
Ток утечки высокого уровня на входе, мкА на выводах группы C	I_{ILH}	–20	20	25, (105), – 60
Ток утечки низкого уровня на входе, мкА на выводах группы D	I_{ILL}	–20	20	25, (105), – 60
Входной ток высокого уровня на выводе с внутренним резистором доопределения до нуля, мкА $U_{IH} = U_{CCIO}$ на выводах группы G	I_{IH_PD}	–	800	25, (105), – 60
Входной ток низкого уровня на выводе с внутренним резистором доопределения до питания, мкА на выводах группы H	I_{IL_PU}	–800	–	25, (105), – 60
Динамический ток потребления схем ввода/вывода, мА	I_{OCC}	–	1000	25, (105), – 60
Потребляемая мощность, Вт	P_{CC}	–	3	25, – 60
			4	(105)
Сопротивление подтягивающего к земле резистора, кОм, $U_{IN} = U_{CCIO} = 2{,}75$ В - выводы группы E	R_{PD1}	10	70	25, (105), – 60
- выводы группы G	R_{PD2}	1,0	6,0	
Сопротивление подтягивающего к питанию резистора, кОм, - выводы группы F	R_{PU1}	10	70	25, (105), – 60
- выводы группы H	R_{PU2}	1,0	6,0	

Наименование параметра, единица измерения, режим измерения	**Буквенное обозначение параметра**	**Норма параметра**		**Температура среды (ккорпуса), °С**
		не менее	**не более**	
Сопротивление подтягивающего к питанию резистора, Ом - выводы AC7, AD7	R_{PU3}	220	280	25, (105), – 60
- вывод C17	R_{PU4}	45	80	
Сопротивление резистора, подключаемого между прямым и инверсным входами LINK портов, Ом	R_{ILV}	92,5	107,5	25, (105), – 60
Выходная частота блока PLL, МГц - максимальная	f_{PLL}	900	–	25, (105), – 60
- минимальная		–	37,5	

Примечание – Группы выводов:

C – A4-A21, B3-B22, C2, C4-C12, C14-C21, C23, D1-D3, D5-D12, D14-D20, D22-D24, E1-E4, E21-E24, F1-F4, F21-F24, G1, G3, G4, G21-G24, H2-H4, H21-H24, J1-J2, J4, K1-K4, L1-L4, M1-M4, N3, P1, P2, P4, R2-R4, T1-T4, U1, U4, V1-V4, W3, W4, Y1-Y4, AA1-AA3, AA5-AA9, AA22, AB2, AB4-AB8, AC4-AC8, AC16, AD4-AD8;

D – A18, B18, C18, D18, F2, G1, G3, G4, H2-H4, J1, J2, J4, J22, K1-K4, L1-L4, M1-M4, N1, N3, P1, P2, P4, R2-R4, R21, R24, T1-T4, T24, U1, U3, U4, V1-V4, W4, Y1-Y3, AA6-AA9, AA15, AA22, AB4, AB7-AB9, AB23, AC3, AC5-AC8, AC16, AC22, AD5-AD8, AD15, AD16;

E – J22, R21, R24, T24, AA15, AB9, AB23, AC22, AD15, AD16;

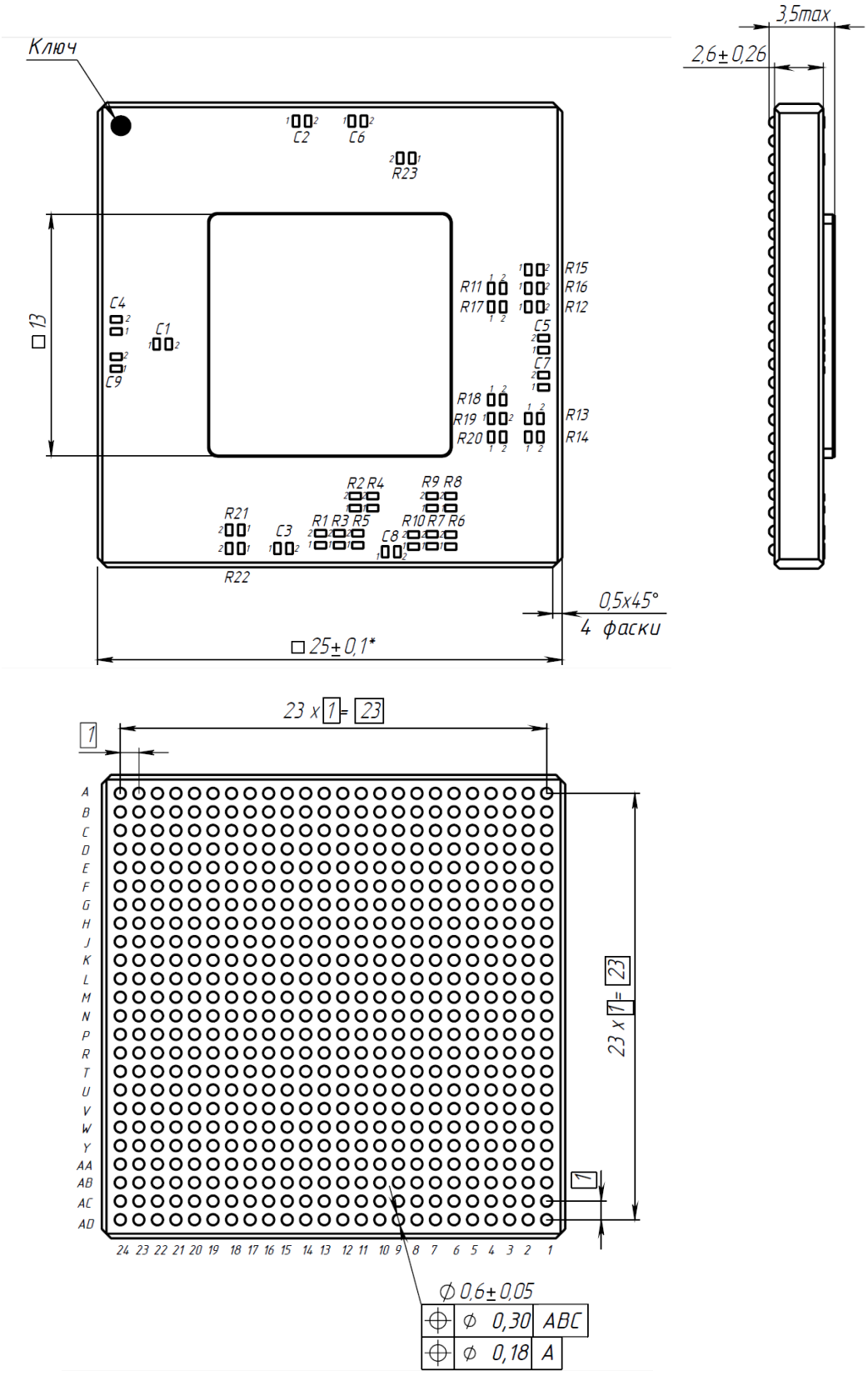
F – A18, B18, C18, D18, F2, G1, G3, G4, H2-H4, J1, J2, J4, J22, K1-K4, L1-L4, M1-M4, N1, N3, P1, P2, P4, R2-R4, R21, R24, T1-T4, T24, U1, U3, U4, V1-V4, W4, Y1-Y3, AA6-AA9, AA15, AA22, AB4, AB7-AB9, AB23, AC3, AC5-AC8, AC16, AC22, AD5-AD8, AD15, AD16;

H – A18, B18, C17, C18, D18, F2, G1, G3, H2, H3, J1, J2, K1, K3, K4, L1, U1, V1, Y4, AA1- AA3, AA5, AA6, AB2, AB5, AB6, AB8, AC5, AD5;

G – G4, N1, U3, AC3, AD8

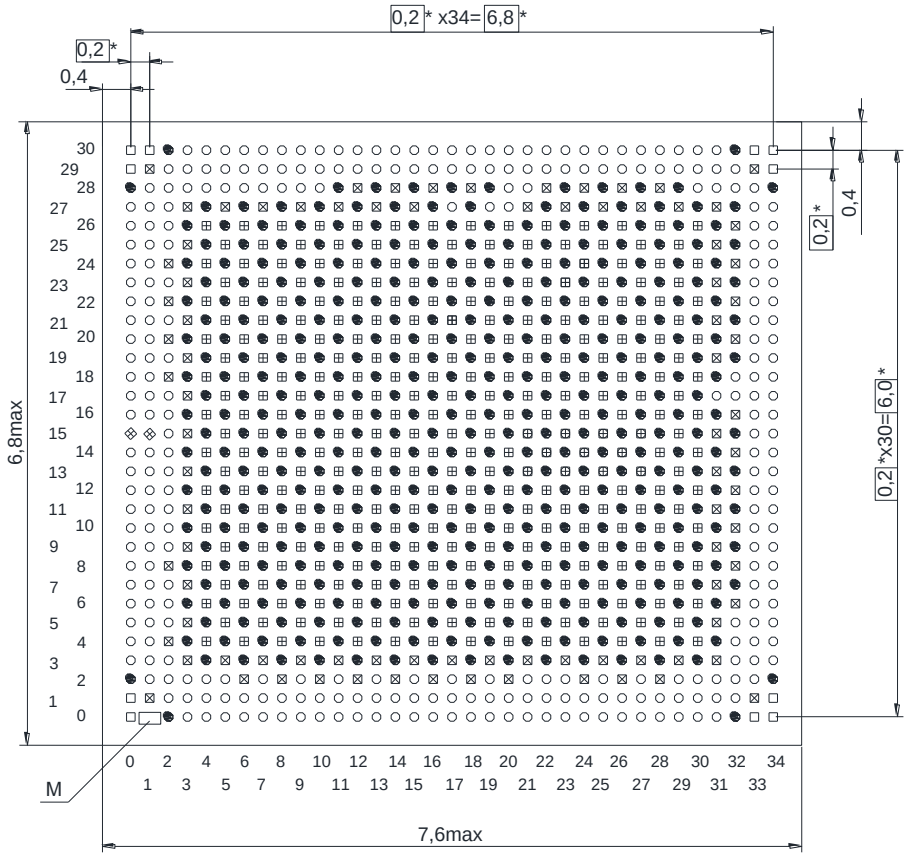
Микросхемы устойчивы к воздействию статического электричества с потенциалом не менее 2 000 В.

14 Габаритный чертеж микросхемы

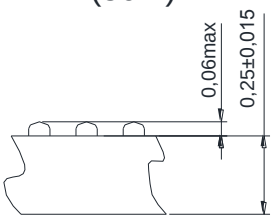


Нумерация выводов показана условно.

Рисунок 94 – Микросхема в корпусе МК8303.576-1



A(50:1)



1 * Размеры для справок.
2 M - маркировка кристалла MLDR84.
3 Размеры контактных площадок (КП) до формирования контактных шариков (КШ) - 90x90 мкм. Материал КШ - Sn5Pb95.
4 Форма КП до формирования КШ - восьмиугольник.
5 Форма контактных столбиков не регламентирована.
6 Номера КШ присвоены условно. Расположение КШ соответствует топологическому чертежу.
7 Обозначение функционального назначения КШ:
- ○ - функциональное назначение КШ приведено в таблице 1;
- ⊞ - VDD - питание ядра;
- ⊠ - VDD_IO - питание ввода/вывода;
- ⊗ - VDD_A - питание аналоговых блоков;
- ● - VSS -общий;
- □ - не используются.

Рисунок 95 – Кристалл (бескорпусное исполнение)

15 Информация для заказа

Обозначение	Маркировка	Тип корпуса	Температурный диапазон
1967ВН028	1967ВН028	МК8303.576-1	минус 60 – 105 °С
К1967ВН028	К1967ВН028	МК8303.576-1	минус 60 – 105 °С
К1967ВН028К	К1967ВН028●	МК8303.576-1	0 – 70 °С

Микросхемы с приемкой «ВП» маркируются ромбом.
Микросхемы с приемкой «ОТК» маркируются буквой «К».

Примечание – Микросхемы в бескорпусном исполнении поставляются в виде отдельных кристаллов, получаемых разделением пластины. Микросхемы поставляются в таре (кейсах) без потери ориентации. Маркировка микросхемы в бескорпусном исполнении – 1967ВН02Н4, К1967ВН02Н4 – наносится на тару.

Лист регистрации изменений

№ п/п	Дата	Версия	Краткое содержание изменения	№№ изменяемых листов
1	15.10.2014	0.1.0	Введена вновь	–
2	20.01.2015	0.2.0	Корректировка в соответствии с замечаниями разработчика	По тексту
3	06.02.2015	0.2.2	Исправлена маркировка даты на рисунке	1
4	26.06.2015	0.3.0	Корректировка в соответствии с замечаниями и дополнениями ГК	По тексту
5	29.01.2016	0.4.0	Внесено исправление в таблицу 3 Внесены дополнения в подраздел «Диапазон изменения тактовой частоты»	10 221
6	12.05.2016	1.0.0	Приведение в соответствие с ТУ и КД лит. А	По тексту
7	09.06.2016	2.0.0	Исправлен рисунок 65	236
8	14.06.2016	2.0.1	Корректировка подраздела «Домены синхронизации». Удален подраздел «Диапазон изменения тактовой частоты».	49
9	19.07.2016	2.0.2	В таблице 32 справлено название регистра NEW_FUN на EXT_FUN	80
10	29.07.2016	2.1.0	Введено бескорпусное исполнение	По тексту
11	30.08.2016	2.2.0	Исправления в таблице 2	24
12	15.09.2016	2.3.0	Исправлен рисунок 3	5
13	22.12.2016	2.4.0	Корректировка в соответствии с замечаниями разработчика	По тексту
14	17.01.2017	2.5.0	Корректировка подраздела «Особенности сохранения контекста процессора». Добавлено описание бита KUSP в таблице 35.	63 – 65, 86
15	27.03.2017	2.6.0	Исправление в таблице 16 Исправление в таблице 120	38 257
16	17.05.2017	2.7.0	Внесены исправления в описание бит BNK0SLOW, BNK1SLOW, HOSTSLOW регистра SYSCON (таблицы 63, 76). Внесены уточнения в подразделы 7.4.4.2, 7.4.4.3, 7.4.6	111, 112, 132, 133 134, 142
17	05.07.2017	2.8.0	Добавлен параметр «пиковая производительность» в раздел «Основные технические характеристики процессора»	1
18	14.08.2018	2.9.0	Обозначения выводов LxBCMPI, LxBCMPO исправлены на nLxBCMPI, nLxBCMPO. Исправлено назначение выводов P23, P24 на УГО (рисунок 2). Добавлено описание параметра t_{LCLKIP} в таблице 120 Внесено уточнение в подраздел «Программная инициализация». Исправлено значение по умолчанию для регистров LRSTAT0 – LRSTAT3 в таблице 102. Исправлено значение сброса в подразделе «Регистр состояния приемника порта связи (LRSTATx)»	По тексту 4 257 220 218 228