Адаптер сети Альфа

Руководство администратора

Листов 41

РФЯЦ-ВНИИТФ

РОСАТОМ

Версия 1.8

АННОТАЦИЯ

Данный документ является руководством по настройке и использованию комплекса программного обеспечения адаптера сети Альфа Альфа (далее по тексту — комплекс программного обеспечения) в рамках многоузловой системы.

Документ предназначен как для администратора многоузловой системы, так и для пользователей без административных полномочий, которые хотят выполнять на многоузловой системе свои задачи.

В руководстве приводятся инструкции по установке, настройке, проверке и использованию комплекса программного обеспечения.

Настоящее руководство распространяется на комплекс программного обеспечения и не заменяет учебную и справочную литературу на остальные программные и аппаратные компоненты многоузловой системы (в том числе процессор, материнскую плату и операционную систему).

Содержание

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Сеть Альфа

Межузловая коммуникационная сеть (МКС) Альфа создана в РФЯЦ – ВНИИТФ и имеет уникальную архитектуру, разработанную отечественными специалистами.

Сеть Альфа предназначена для передачи данных между узлами вычислительной системы с высокой скоростью и малой коммуникационной задержкой. Она используется при создании многопроцессорных вычислительных систем, в суперкомпьютерах и высокопроизводительных системах хранения и обработки данных, требующих высоких скоростей и низких задержек при передаче данных.

Сетевое оборудование Альфа состоит из следующих компонентов:

- коммутатор сети Альфа (рис. 1), который представляет собой выдвижной 1U блок, монтируемый в 19-дюймовую стойку;
- адаптер сети Альфа в форм-факторе PCI Express Low-Profile (рис. 2), который устанавливается в вычислительный узел;
- кабели QSFP-DD (рис. 3), обеспечивающие соединение адаптеров и коммутаторов.

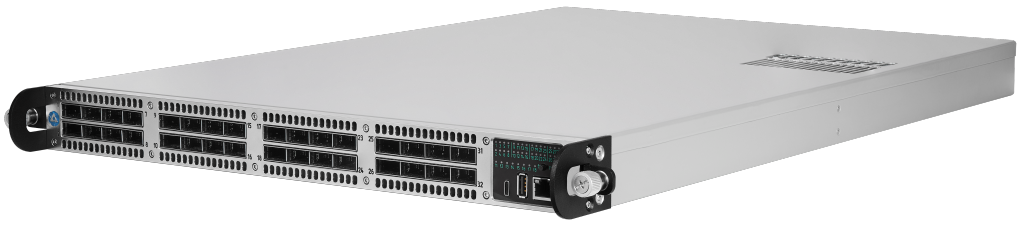

Рисунок 1 – Коммутатор сети Альфа

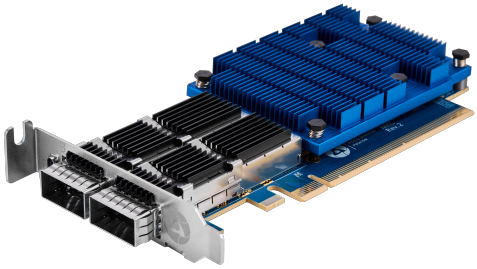

Рисунок 2 – Адаптер сети Альфа

Рисунок 3 – Кабель QSFP-DD

1.2. Адаптер сети Альфа

Адаптер сети Альфа (рис. 4) устанавливается в каждый вычислительный узел и обеспечивает передачу пакетов/сообщений между узлами. Двухпортовый адаптер сети Альфа оснащен интерфейсом PCI Express 3.0 и может быть подключен к соседнему адаптеру или к коммутатору сети Альфа медным, либо оптическим кабелем.

Таблица 1 – Характеристики адаптера сети Альфа.

Характеристика	Значение
Количество портов	2 x QSFP-DD
Пропускная способность порта	80 Гбит/с
Задержка между соседними узлами	1,7 мкс
Интерфейс связи с процессором узла	PCI Gen 3.0 x16 (в режиме бифуркации x8x8)
Система охлаждения	Воздушная пассивная
Потребляемая мощность	29,5 Вт
Форм-фактор	HH/HL PCI Express card
Габаритные размеры	142,25 x 17,25 x 68,9 (ГхВхШ, мм)

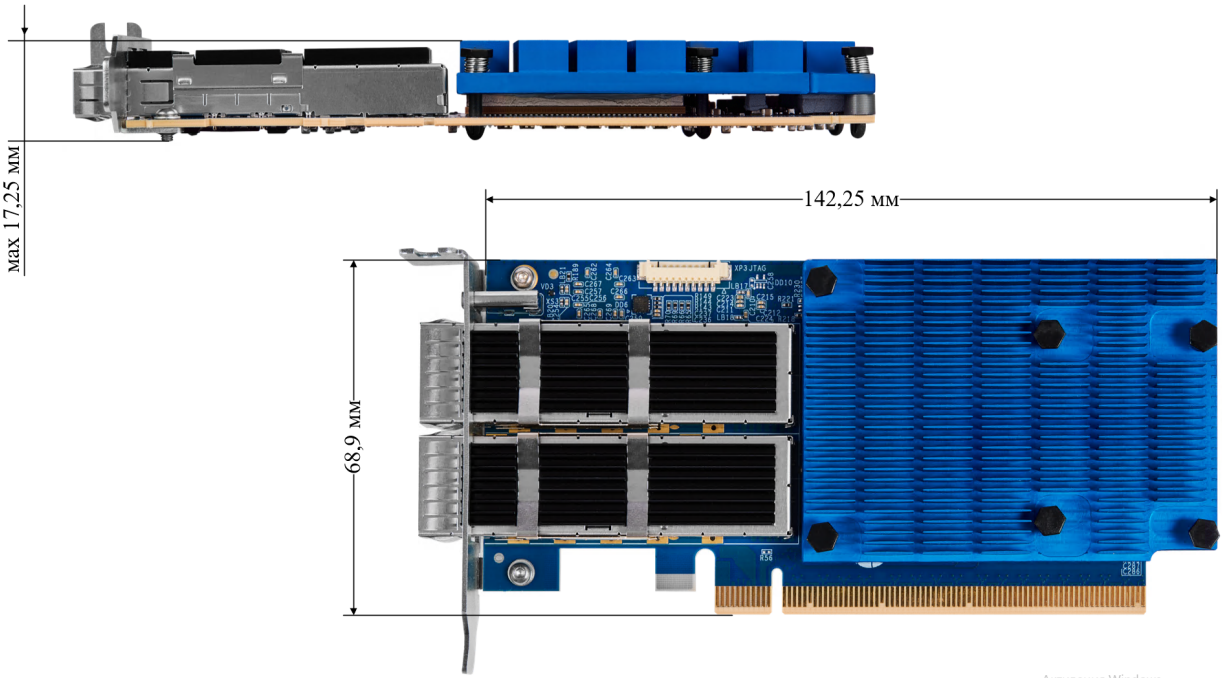

Рисунок 4 – Габаритные размеры адаптера МКС Альфа

Функциональные возможности адаптера сети Альфа

- Инжекция пакетов в сеть:
  - режим PIO – готовый пакет копируется в инжекционный конвейер адаптера;
  - режим DMA – адаптер самостоятельно работает с памятью хоста (формирует, отправляет и получает пакеты).
- Эжекция пакетов из сети – в режиме DMA.
- Отправка и получение данных происходит без участия ядра ОС.
- Поддержка следующих типов сетевых операций:
  - PUT – запись данных в память удаленного узла;
  - GET – чтение данных из памяти удаленного узла;
  - ADD/CAS – атомарная операция с ячейкой памяти удаленного узла с возможностью возврата значения (FADD/FCAS);
- Аппаратная поддержка семантики кольцевого буфера.
- Поддержка прерываний MSI-X:
  - вызов прерывания на удаленном узле (поддерживается до 512 прерываний);
  - нулевое прерывание на хосте системы в случае ошибок адаптера.

В адаптере МКС Альфа реализованы следующие механизмы диагностики:

- выявление ошибок при инжекции:
  - попытка передачи пакета с неправильным заголовком;
  - не удаётся отправить пакет в течение длительного времени;
- обработка ошибок на канале связи:
  - количество переповторов передаваемых пакетов за промежуток времени достигло порогового значения;
  - количество обрывов связи за промежуток времени достигло порогового значения;
  - порт долгое время не инициализируется;
  - переход порта из состояния «выключен» в состояние «включен»;
- определение перегрева адаптера сети Альфа;
- оценка загруженности и здоровья сети:
  - счетчики загруженности каналов связи (переданные и принятые сетевые пакеты);
  - счетчики для оценки качества канала связи (переповторы во время передачи, обрывы связи, время нахождения канала связи в состоянии обрыва).

1.3. Комлпекс программного обеспечения

Комплекс программного обеспечения предназначен для обеспечения работоспособности сети Альфа в составе многоузловой системы.

Основными функциями комплекса являются:

- настройка и управление компонентами сети Альфа в рамках вычислительного узла;
- предоставление интерфейса параллельного программирования для языка C (библиотеки SHMEM и MPI);
- поддержка передачи TCP/IP трафика поверх сети Альфа;
- запуск и контроль выполнения параллельных задач;
- тестирование и отладка работы компонентов сети Альфа.

Комплекс программного обеспечения может работать на системах с архитектурой процессора, поддерживающего набор инструкций x86_64, aarch64 или Эльбрус, под управлением операционной системы на базе ядра Linux.

2. ПОСТРОЕНИЕ ТОПОЛОГИЙ

2.1. Соединение адаптеров

В адаптере сети Альфа есть 2 порта, которые отвечают за направления передачи данных. Физический порт XS2 соответствует направлению «+», XS1 соответствует направлению «-» (рисунок 5).

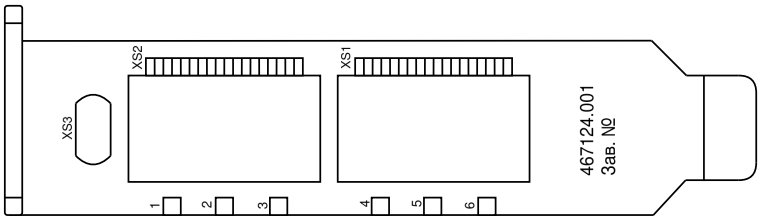

Рисунок 5 – Схематичное изображение адаптера Альфа

Коммутация адаптеров происходит после установки адаптера в серверную стойку. В дальнейших схемах подключения будет рассмотрено использование 1-юнитовых серверов, где в сервере расположен 1 адаптер (рисунок 6). В сервере может быть место для размещения разного количества адаптеров в зависимости от запроса Заказчика.

Рисунок 6 – Адаптер в сервере

Коммутация адаптеров сети Альфа между собой осуществляется между направлением «+» одного адаптера и направлением «-» другого. Схема подключения двух адаптеров представлена на рисунке 7.

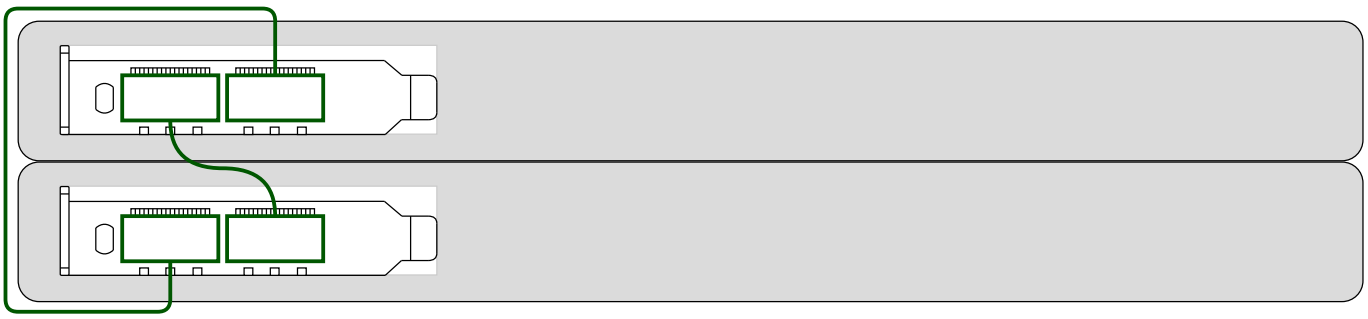

Рисунок 7 – Коммутация двух адаптеров

Адаптеры сети Альфа соединяются в топологию «кольцо». Пример коммутации 3 адаптеров представлен на рисунке 8.

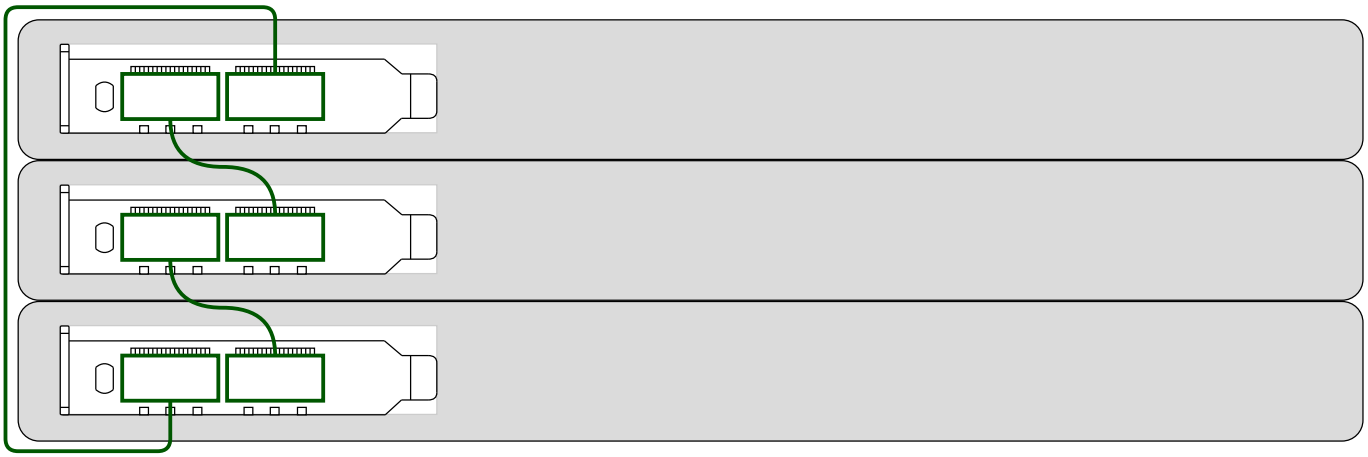

Рисунок 8 – Коммутация трех адаптеров

Количество адаптеров в кольце влияет на скорость и отказоустойчивость собранной системы:

– От 3 до 8 адаптеров в кольце. Рекомендуемый вариант, обеспечивающий максисальную скорость и допускающий повреждения 1 линка в системе.
– От 9 до 29 адаптеров в кольце. Реализуется по требованию Заказчика.

Схема подключения N количества адаптеров приведена на рисунке 9.

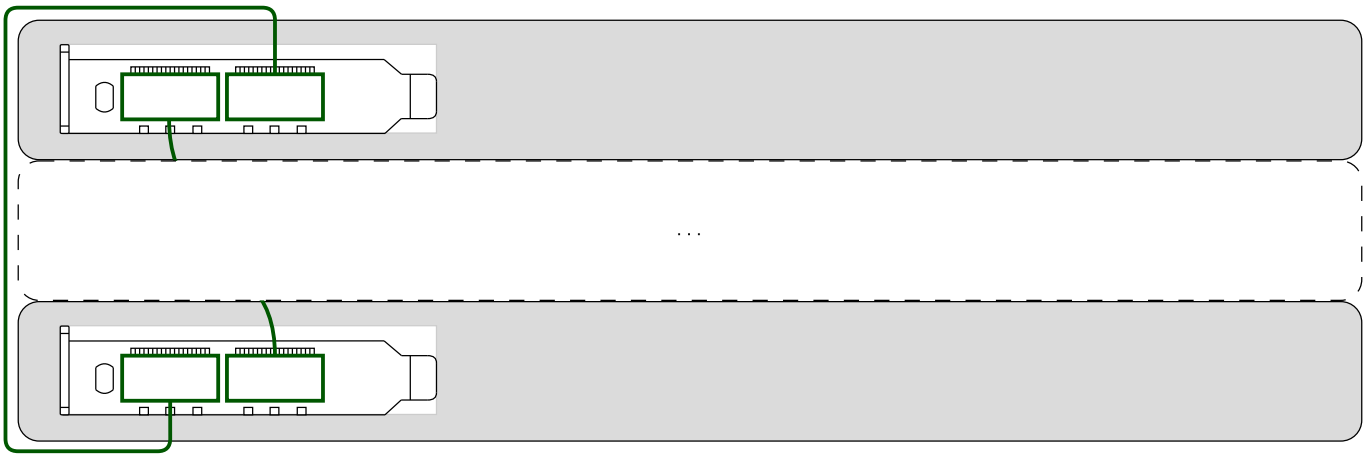

Рисунок 9 – Коммутация N адаптеров

2.2. Топологии с коммутаторами

В коммутаторе сети Альфа есть 32 порта, физически разделенных на 4 модуля коммутации по 8 портов (рис. 10).

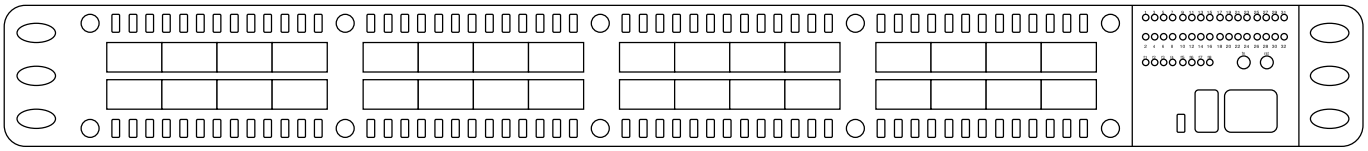

Рисунок 10 – Схематичное изображение коммутатора сети Альфа

В каждом модуле коммутации (sw0c0, sw0c1, sw0c2, sw0c3) определенные порты отвечают за логические направление («+» или «-») и измерение (X, Y, Z, K) передачи данных (рис. 11). При простроении топологий с коммутатором порты X и Y стоит использовать для соединений с другими коммутаторами, Z и K – для подключения к коммутаторам адаптеров сети Альфа.

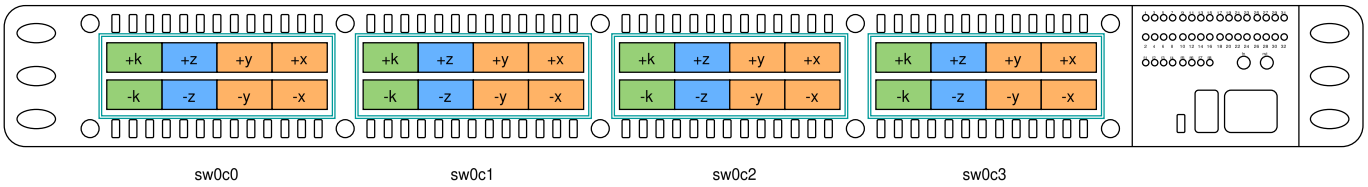

Рисунок 11 – Логическое обозначение портов Коммутатора Альфа

2.2.1. Коммутация адаптеров с коммутаторами

Адаптеры сети Альфа подключаются к коммутаторам по портам, отнесенным к измерениям X и Y. При отсутствии необходимости подключения других коммутаторов Альфа допустимо подключать адаптеры также к измерениям Z и K, увеличив количество подключаемых адаптеров Альфа до 32.

Подключение одного адаптера к коммутатору можно осуществить следующими вариантами:

– С помощью одного кабеля. Рекомендуемый вариант, представлен на рисунке 12.

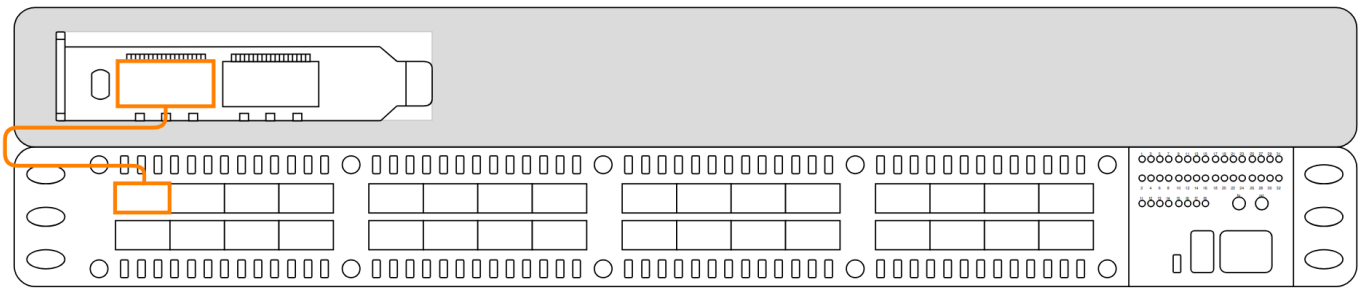

Рисунок 12 – Подключение адаптера к коммутатору посредством одного кабеля

– С помощью двух кабелей. В необходимых случаях снижает задержку при передаче данных внутри коммутатора, однако сокращает возможности расширения сети из-за ис-

пользования двух портов коммутатора вместо одного. Вариант подключения представлен на рисунке 13.

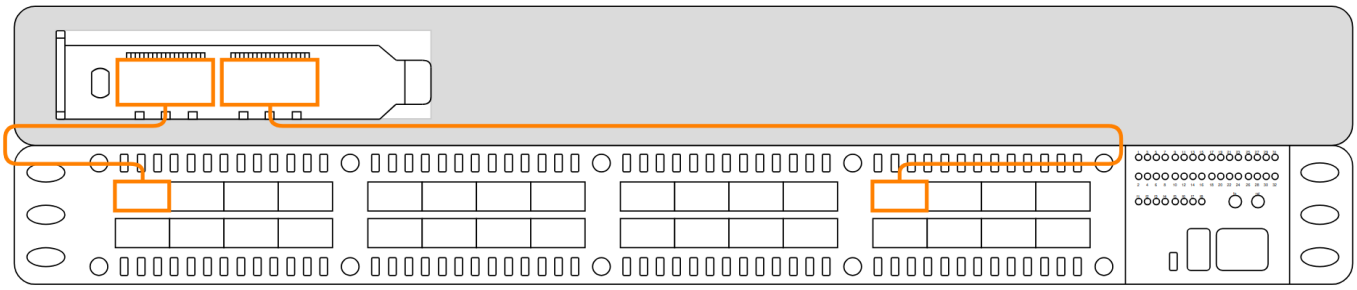

Рисунок 13 – Подключение адаптера к коммутатору посредством двух кабелей

При необходимости соединения нескольких адаптеров к одному коммутатору рекомендуется выполнять их подключение в пределах одного модуля коммутации (sw0c0, sw0c1, sw0c2, sw0c3) рисунка 11. При заполнении всех четырех портов одного модуля коммутации рекомендуется выбирать следующий модуль коммутации для подключения исходя из таблицы 2.

Таблица 2 – Порядок подключения к модулям коммутации коммутатора.

Первый заполненный модуль коммутации коммутатора	Приоритетный для подключения следующий модуль коммутации	Наименее приоритетный
sw0c0	sw0c1, sw0c2	sw0c3
sw0c1	sw0c3, sw0c0	sw0c2
sw0c2	sw0c3, sw0c0	sw0c1
sw0c3	sw0c2, sw0c1	sw0c0

Один адаптер также можно подключить к двум коммутаторам. Такой вариант подключения обеспечивает дополнительную отказоустойчивость сети (рис. 14).

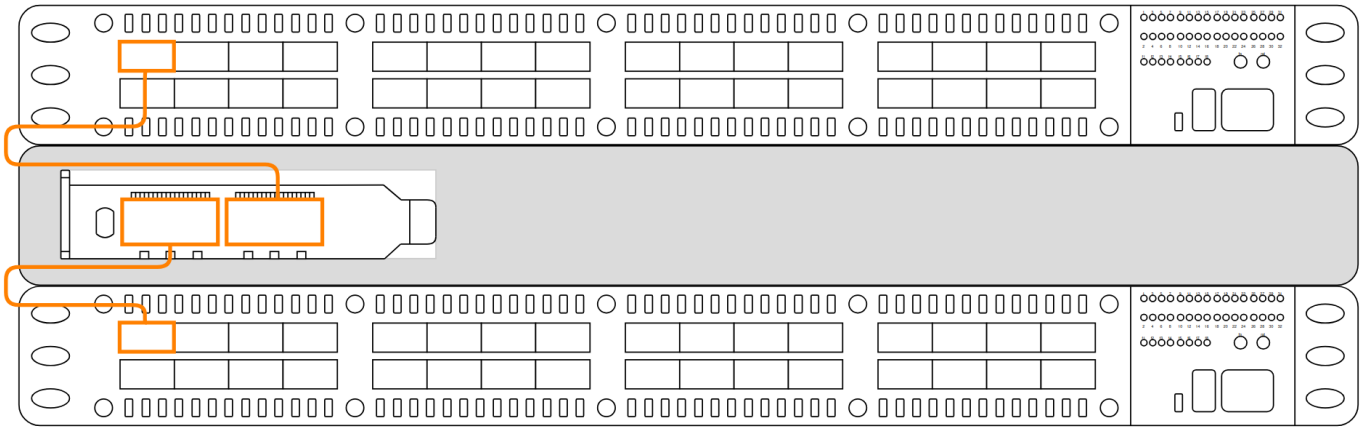

Рисунок 14 – Подключение одного адаптера к двум коммутаторам

2.2.2. Варианты соединения коммутаторов между собой

При соединении коммутатора с другим коммутатором рекомендуется выполнять подключение по измерениям X и Y.

Подключение двух коммутаторов выполняется с помощью соединения их по измерению X, при этом направления подключения должны отличаться (порт +X одного коммутатора должен быть подключен к порту -X другого). Соответствующее подключение проводится по всем четырем модулям коммутации, оно представлено на рисунке 15.

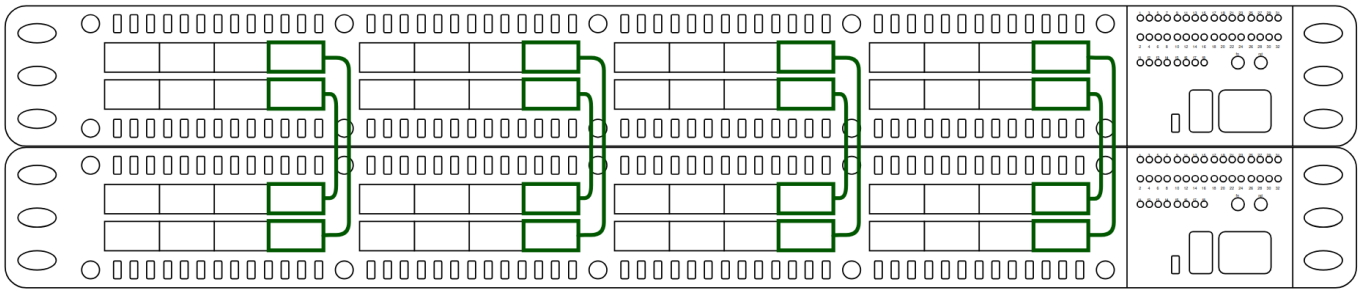

Рисунок 15 – Соединение двух коммутаторов

Подключение трех коммутаторов в кольцо показано на рисунке 16.

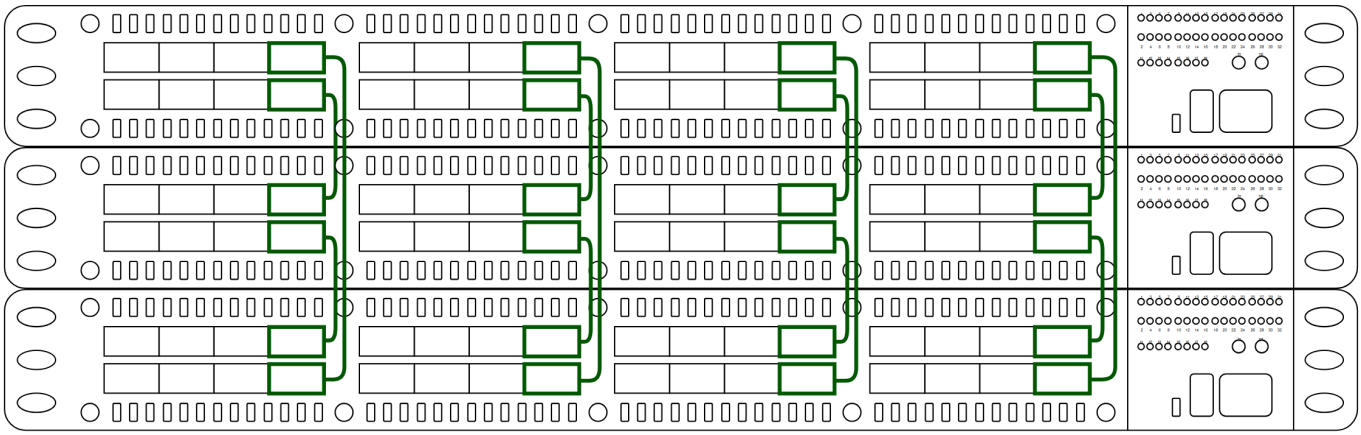

Рисунок 16 – Соединение трех коммутаторов в кольцо по измерению X

Начиная с четырех коммутаторов, подключение можно выполнять двумя способами:

– Топология «Кольцо». Подключение выполняется аналогнично предыдушим примерам с 2 и 3 коммутаторами. Допускается подключение до 8 коммутаторов в кольце.
– Топология «2D-Top». При подключении добавляются порты измерения Y. Допускается подключение до 64 коммутаторов в 2D-Top.

Соединение четырех коммутаторов показано на рисунках 17, 19. Упрощенные схемы соединений показаны на рисунках 18, 20.

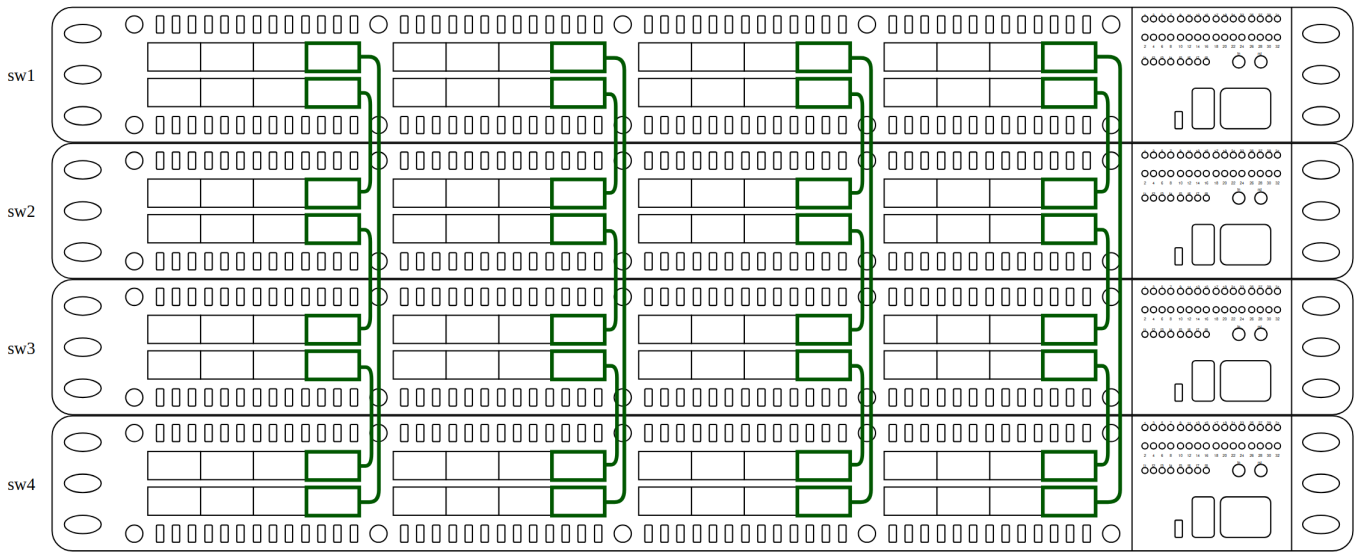

Рисунок 17 – Соединение четырех коммутаторов в кольцо по измерению X

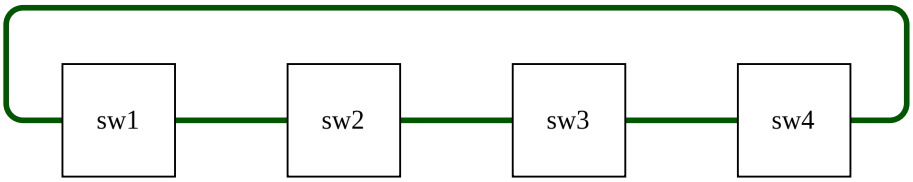

Рисунок 18 – Упрощенная схема соединения четырех коммутаторов в кольцо по измерению X

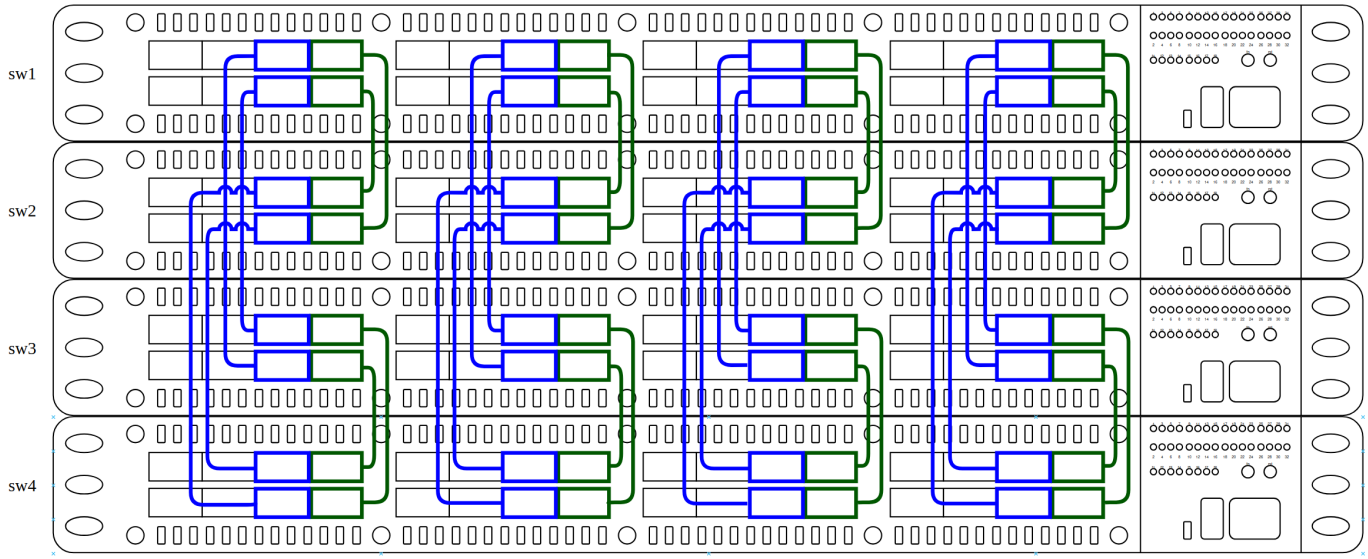

Рисунок 19 – Соединение четырех коммутаторов в 2D-Top размерностью 2x2 по измерениям X, Y

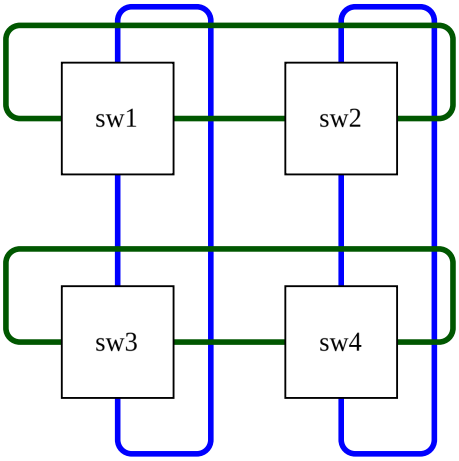

Рисунок 20 – Упрощенная схема соединения четырех коммутаторов в 2D-Top размерностью 2x2 по измерениям X, Y

Подключение более, чем 8 коммутаторов осуществляется только через топологию 2D-Top. Упрощенная схема соединения 9 коммутаторов показана на рисунке 21.

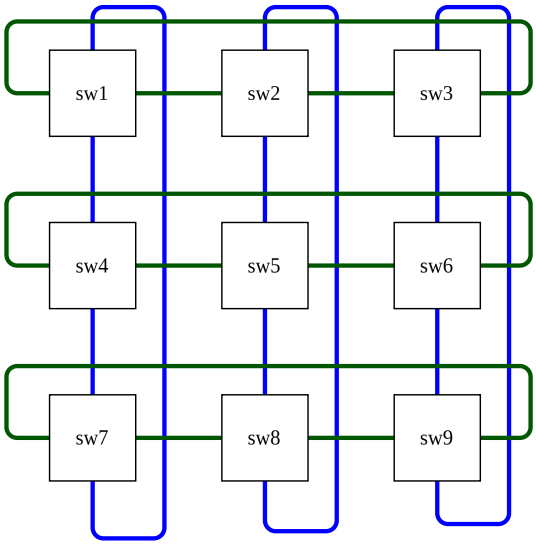

Рисунок 21 – Упрощенная схема соединения 9 коммутаторов в 2D-Top размерностью 3x3 по измерениям X, Y

Выполнение подключения большего количества коммутаторов изображено на рисунке 22.

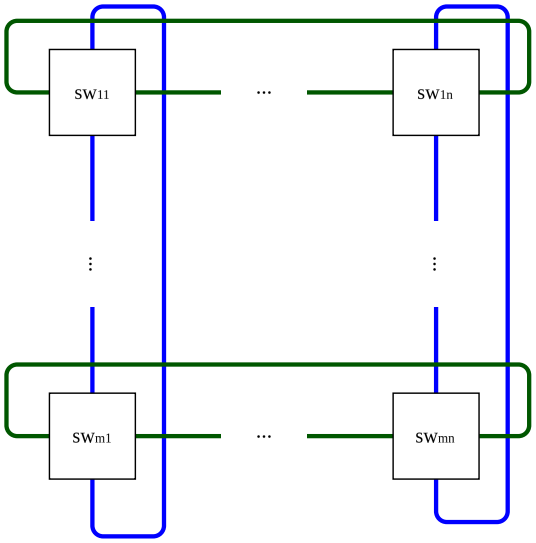

Рисунок 22 – Упрощенная схема соединения коммутаторов в 2D-Top по измерениям X, Y

При этом, максимальное число коммутаторов по индексам n и m не должно быть больше 8.

Количество соединенных коммутаторов влияет на скорость и отказоустойчивость собранной системы:

- От 2 до 64 коммутаторов. Рекомендуемый вариант. Обеспечивает для подключения до 1024 узлов.
- От 65 до 512 коммутаторов (до 4096 узлов). Реализуется по требованию Заказчика.

3. СТРУКТУРА КОМПЛЕКСА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

3.1. Список компонентов

Комплекс программного обеспечения включает в себя следующие компоненты:

- драйвер, оформленный в виде модуля ядра Linux;
- модуль поддержки стека протоколов TCP/IP;
- низкоуровневую библиотеку передачи данных;
- библиотеки параллельного программирования SHMEM и MPI;
- тест проверки работоспособности;
- утилиту мониторинга и управления.

3.2. Драйвер

Драйвер отвечает за инициализацию адаптера сети Альфа в системе, его первоначальную настройку, а также за управление ресурсами адаптера и их распределение между пользовательскими процессами. Взаимодействие пользовательских программ с драйвером осуществляется через интерфейс символьного устройства.

Основные опции загрузки драйвера:

- `dmamem_size` — размер DMA-региона (обязательный параметр);
- `a3eth_dmamem_size` — размер DMA-региона из которого будет выделяться память для модуля поддержки стека протоколов TCP/IP;
- `numanode` — номер сокета (узла NUMA) для выделения DMA-региона;
- `nodenum` — уникальный номер узла в рамках сети (используется модулем поддержки стека протоколов TCP/IP);
- `use_wc` — использовать режим Write Combining при записи пакетов в инжекционный конвейер (включен по умолчанию);
- `verbose` — включить вывод отладочной информации.

3.3. Модуль поддержки стека протоколов TCP/IP

Модуль поддержки стека протоколов TCP/IP (`a3eth`) выполняет приём/передачу Ethernet-кадров через сеть Альфа. Данный модуль создаёт сетевой интерфейс канального уровня (L2), который полностью аналогичен интерфейсу стандартного сетевого адаптера Ethernet, что позволяет взаимодействовать с ним посредством стандартных сетевых утилит в составе операционной системы на базе ядра Linux (например, iproute2, ifconfig, route). С точки зрения пользователя и сетевого приложения через данный интерфейс можно осуществлять приём/передачу, используя

стандартный механизм сетевых сокетов. Помимо непосредственного взаимодействия с эмулируемым сетевым интерфейсом поддерживается возможность добавления его в состав сетевого моста (Linux Bridge) и виртуального коммутатора (OpenVSwitch) для нужд взаимодействия виртуальных машин или контейнеров.

В текущей реализации программного обеспечения не поддерживается одновременное использование `a3eth` и запуск пользовательских задач с использованием SHMEM или MPI.

3.4. Низкоуровневая библиотека передачи данных

Низкоуровневая библиотека передачи данных (`a3verbs`) предназначена для доступа к функциональным возможностям адаптера сети Альфа из адресного пространства пользователя. На базе данной библиотеки (и её API) реализуются все доступные интерфейсы параллельного программирования (MPI, SHMEM). Библиотека отвечает за формирование и инжекцию сетевых пакетов, получение информации об устройстве, управление устройством.

3.5. Библиотека SHMEM

Реализация библиотеки параллельного программирования SHMEM содержит следующее подмножество функций стандарта OpenSHMEM 1.5:

- `int shmem_init()` — инициализация библиотеки, возвращает ноль в случае успеха; при повторном вызове никакие действия не выполняются, вызов завершается успешно;
- `void shmem_finalize()` — завершение библиотеки; при повторном вызове никакие действия не выполняются, вызов завершается успешно;
- `int shmem_num_pes()` — возвращает количество процессов;
- `int shmem_my_pe()` — возвращает номер текущего процесса в диапазоне от 0 до (`shmem_num_pes()` – 1);
- `void *shmem_alloc(size_t nbytes)` — выделение области общей памяти (памяти, доступной для чтения и записи другим процессам задачи) в байтах; содержит глобальный барьер, поэтому вызывать требуется во всех процессах и с одинаковыми аргументами; возвращает NULL в случае невозможности выделить память;
- `void shmem_barrier_all()` — барьерная синхронизация всех процессов задачи; завершение барьера для данного процесса гарантирует завершение всех операций put, адресованных ему и выданных другими процессами до барьера;
- `void shmem_uint32_put(uint32_t *dst, const uint32_t *src, size_t nelems, int pe)` — неблокирующий put в память другого процесса `pe` по указателю `dst` (из области памяти, выделенной с помощью `shmem_alloc()`) из локальной памяти процесса

по указателю `src` (не обязательно выделенному с помощью `shmem_alloc()`); размер данных `nelems` считается в 32-битных словах, все данные должны быть выровнены по границе 32 бита; функция возвращает управление, когда буфер `source` освобожден и его можно использовать, однако записи не обязательно доставлены адресату; после вызова `shmem_barrier_all()` гарантируется, что для всех выполненных операций put данные доставлены адресатам;

- `void shmem_uint32_get(uint32_t *dst, const uint32_t *src, size_t nelems, int pe)` — неблокирующий get из памяти процесса `pe` по указателю `src` (из области памяти, выделенной с помощью `shmem_alloc()`) в память вызывающего процесса по указателю `dst` (также выделенному с помощью `shmem_alloc()`); размер данных `nelems` считается в 32-битных словах, все данные должны быть выровнены по границе 32 бита; функция возвращает управление, когда операции чтения выданы, но могут быть еще не завершены.
- `void shmem_uint64_put_nbi(uint64_t *dst, const uint64_t *src, uint64_t nelems, int pe)` — неблокирующий put в память другого процесса `pe` по указателю `dst` (из области памяти, выделенной с помощью `shmem_alloc()`) из локальной памяти процесса по указателю `src` (также выделенному с помощью `shmem_alloc()`); размер данных `nelems` считается в 64-битных словах, все данные должны быть выровнены по границе 64 бита;
- `void shmem_uint32_atomic_add(uint32_t *dest, uint32_t value, int pe)` — атомарное сложение `value` со значением в памяти другого процесса `pe` по указателю `dst` (из области памяти, выделенной с помощью `shmem_alloc()`); данные должны быть выровнены по границе 32 бита; функция возвращает управление после отправки пакета в сеть, однако сама операция к этому моменту может ещё не завершиться;
- `void shmem_uint32_atomic_inc(uint32_t *dest, int pe)` — атомарное увеличение на единицу значения в памяти другого процесса `pe` по указателю `dst` (из области памяти, выделенной с помощью `shmem_alloc()`); функция возвращает управление после отправки пакета в сеть, однако сама операция к этому моменту может ещё не завершиться.

3.6. Библиотека MPI

Реализация библиотеки параллельного программирования MPI основана на реализации MPICH 4.0 стандарта MPI 4.0.

3.7. Тест проверки работоспособности

Тест проверки работоспособности (`bench`) позволяет проверять базовую работоспособность сетевого оборудования сети Альфа, а также измерять коммуникационную задержку и пропускную способность сети.

Формат вызова:

`bench` [`--test` *тест*] [*опции*]

Основные опции программы:

- `--help, -h` — выводит справочную информацию по использованию;
- `--version` — выводит версию программы;
- `--msgcount, -c` *число итераций* — задаёт число итераций для каждого размера; значение по умолчанию — 300000;
- `--operation, -o` *тип операции* — задаёт тип операции для текущего теста (смотри далее); значение по умолчанию — put;
- `--min_pkt, -p` *размер* — задаёт минимальный размер пакета в 8-байтных словах; значение по умолчанию — 1;
- `--max_pkt, -P` *размер* — задаёт максимальный размер пакета в 8-байтных словах; значение по умолчанию — 256;
- `--step, -s` *размер шага* — задаёт размер шага для тестов в 8-байтных словах; по умолчанию используются шаги по степеням двойки;
- `--test, -t` *тест* — задаёт тест (смотри далее); значение по умолчанию — lat.

Список доступных тестов:

- `a2a` — измерение пропускной способности на паттерне обмена «все — всем» (каждый с каждым);
- `lat` — измерение коммуникационной задержки между парами PE;
- `o2a` — измерение пропускной способности на паттерне обмена «один — всем»;
- `pairs` — измерение пропускной способности между парами PE;
- `uniform` — измерение пропускной способности при случайных равномерно распределённых обменах.

Список доступных операций:

– `put` — операция записи в память удаленного узла;
– `put0` — операция записи в память удаленного узла по нулевому виртуальному каналу;
– `putnb` — операция неблокирующей записи в память удаленного узла (в режиме DMA);
– `nop` — отправка пакета типа NOP;
– `get` — операция чтения из памяти удаленного узла;
– `rirq` — вызов прерывания на удаленном узле.

3.8. Утилита мониторинга и управления

Программа `a3ctl` позволяет получать список доступных устройств на узле, производить их настройку и получать информацию об их текущем статусе работы, а также считывать различные метрики производительности.

Формат вызова:

`a3ctl` [*опции*] [*команда*]

Основные опции данной команды:

– `-h` — выводит справочную информацию по использованию;
– `-d` *номер* — задаёт номер устройства в каталоге /dev, для которого будет выполнена команда; значение по умолчанию — 0;

Список основных команд:

– `reset` — выполняет аппаратный сброс устройства;
– `dinfo` — выводит информацию об устройстве, а также позволяет получать список доступных на узле устройств;
– `port` — позволяет получить текущий статус и счетчики ошибок порта (port info), вкл./выкл. порт (port power), управлять счетчиками производительности (port counters), а также получить информацию о вставленном кабеле (port qsfp);
– `except` — позволяет узнать о возникших исключительных ситуациях.
– `mcu` — позволяет узнать температуру и напряжения источников питания Для каждой команды доступна опция `-h`, позволяющая узнать подробности об использовании команды.

4. НАСТРОЙКА КОМПЛЕКСА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

4.1. Требования к аппаратной части

В качестве управляющего и вычислительных узлов должны использоваться серверы, оборудованные сетевыми интерфейсами Ethernet и процессорами, поддерживающими набор инструкций x86_64, aarch64 или Эльбрус. Управляющий узел должен иметь носитель информации, доступный на чтение и запись. В данном руководстве предполагается, что вычислительные узлы также имеют носители информации, доступные на чтение и запись. Для систем с малым числом узлов управляющий узел может совпадать с одним из вычислительных.

4.2. Предварительная настройка узлов

Этапы предварительной настройки управляющего узла:

1) установить дистрибутив операционной системы;
2) проверить наличие следующих пакетов, доустановить в случае их отсутствия: automake, autogen, kernel-devel, libtool, libnuma1, make, gcc, gcc-c++ (если планируется использование языка C++), gcc-fortran (если планируется использование языка Fortran), binutils, python, JQ;
3) для удобной работы с вычислительными узлами рекомендуется настроить монтирование (например, через NFS) по сети раздела `/home` с управляющего узла, аналогично для директории `/usr/lib/a3net`, куда устанавливается комплекс программного обеспечения;
4) настроить беспарольный доступ по SSH с управляющего узла на каждый из вычислительных узлов (для всех пользователей).

Предварительная настройка вычислительного узла выполняется в следующей последовательности:

1) установить дистрибутив операционной системы; в качестве hostname рекомендуется использовать имя вида `<префикс><число>`;
2) проверить наличие следующих пакетов, доустановить в случае их отсутствия: automake, autogen, kernel-devel, libtool, libnuma1, make, gcc, gcc-c++ (если планируется использование языка C++), gcc-fortran (если планируется использование языка Fortran), binutils, python, JQ;
3) для удобной работы рекомендуется настроить монтирование (например, через NFS) по сети раздела `/home` с управляющего узла, аналогично для директории `/usr/lib/a3net`, куда устанавливается комплекс программного обеспечения;
4) настроить беспарольный доступ по SSH с управляющего узла на каждый из вычисли-

тельных узлов (для всех пользователей);

5) в некоторых дистрибутивах для автоматической загрузки драйверов, не входящих в официальную поставку (таковым является драйвер сетевого оборудования сети Альфа), необходимо в конфигурационном файле `/etc/modprobe.d/unsupported-modules` параметр `allow_unsupported_modules` выставить в `1`.

4.3. Установка комплекса программного обеспечения

Установка комплекса программного обеспечения выполняется на управляющем и вычислительных узлах следующим образом:

1) перейти в директорию, в которой расположен комплекс программного обеспечения;
2) в режиме суперпользователя выполнить команду установки: `./install.sh`;
3) перезагрузить узел.

Программа установки (install.sh) проверяет совместимость дистрибутива ПО с установленной версией ОС, устанавливает ПО и службы знакомства узлов, а также производит настройку.

4.3.1. Настройка менеджера подсети Альфа

Настройка менеджера подсети Альфа осуществляется с помощью json файлов, лежащих в директории `/usr/lib/a3net/etc`.

Примечание – Далее в документе будет употребляться запись:

`параметр json["key1"]["key2"] = value в файле file.json`,

которую необходимо понимать так, что в файле `file.json` должна быть определена структура `json`, в которой присутствует структура с ключами `"key1"` и `"key2"` со значением `value`:

```
{
    ...
    ...
    "key1": {
        ...
        ...
        "key2": value,
        ...
        ...
    },
    ...
    ...
}
```

Настройки:

– параметр `json["a3net"]["fssn"]` в файле a3net.json. Допустимое значение: `"0"`, `"1"`, `"none"`. Настройка первого шага по умолчанию, если не определено таблицами маршрутов. Данный параметр должен быть выставлен в `"0"` для режима `mirror`;

– параметр `json["a3snmgr"]["mirror"]["exclude"]` в файле a3net.json. Допустимое значение: список вычислительных узлов `["host1", "host2", "host3"]`, которые не будут учитываться при определении подсети для режима `mirror`;
– параметр `json["node"]` в файле ring.json. Допустимое значение: список вычислительных узлов `["host1", "host2", "host3"]` в топологии ring. Данный список должен быть одинаковым на всех вычислительных узлах для поддержки `a3eth`.

4.3.2. Запуск менеджера подсети Альфа

В зависимости от топологии сети Альфа на каждом вычислительном узле необходимо активировать и запустить одну из служб менеджера подсети:

– коммутаторная – `a3net.subnet-manager@switch.service`;
– кольцевая – `a3net.subnet-manager@ring.service`;
– зеркальная – `a3net.subnet-manager@mirror.service`.

После запуска одной из служб в директории `/run/a3snmgr/var/` будут созданы файлы с информацией о сети:

– `net.json` – описание топологии и узлов сети;
– `routes.json` – маршруты между вычислительными узлами сети.

4.3.3. Настройка a3eth

Если используется кольцевая топология (4.3.2), то на каждом вычислительном узле необходимо создать файл `/usr/lib/a3net/etc/ring.json`, в котором требуется перечислить список участников сети. После этого необходимо перезапустить менеджер подсети Альфа. Пример конфигурационного файла:

```
{
    "nodes": ["host1", "host2", "host3"]
}
```

После чего необходимо перезапустить сервис `a3net.subnet-manager@ring.service` на всех узлах.

По умолчанию сетевой Ethernet интерфейс `a3eth0` использует сеть `10.0.0.0/24`. Для её изменения необходимо задать параметр `json["a3net"]["ip"] = "10.0.0.0/24"` в файле `/usr/lib/a3net/etc/a3net.json` и перезапустить менеджер подсети Альфа (4.3.2);

Если в ОС используется менеджер сети, который может повлиять на конфигурацию сетевых интерфейсов (NetworkManager, systemd-networkd и т.п.), тогда необходимо добавить сетевой интерфейс `a3eth0` в его исключения.

Например, если используется менеджер сети NetworkManager, то необходимо добавить соответствующую конфигурацию: `/etc/NetworkManager/conf.d/90-a3eth.conf`:

```
[device-a3eth0-unmanaged]
match-device=interface-name:a3eth0
managed=0
```

4.3.4. Запуск a3eth

Для запуска a3eth необходимо сперва активировать и запустить службу `a3net.a3eth.service` (Менеджер подсети Альфа должен быть загружен).

После запуска службы должен создаться сетевой интерфейс:

```
...
6: a3eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 4014 qdisc mq state UNKNOWN group default
qlen 1000
    link/ether 02:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.1/24 scope global a3eth0
       valid_lft forever preferred_lft forever
```

4.3.5. Проверка a3eth

Проверить работоспособность интерфейса можно с помощью команды `ping`:

Пример вывода команды:

```
# ping -c 3 10.0.0.2 -I a3eth0
```

```
PING 10.0.0.2 (10.0.0.2) from 10.0.0.1 a3eth0: 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.079 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.078 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.073 ms
--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2050ms
rtt min/avg/max/mdev = 0.073/0.076/0.079/0.002 ms
```

Опция `-I` позволяет явно указать используемый сетевой интерфейс. Для корректной работы пинговать нужно другой узел, иначе пакеты будут перенаправлены в локальный интерфейс, и сеть Альфа задействована не будет.

Проверить пропускную способность можно с помощью утилиты `iperf`:

```
# iperf -s -B 10.0.0.1 -p 5001
```

```
# numactl -C 0-3 -m 0 iperf -c 10.0.0.2 -P 4 -i 2 -t 10
```

В первой команде утилита запускается в режиме сервера, а во второй — в режиме клиента. Сервер указывает свой адрес, клиент указывает адрес сервера.

При запуске в режиме клиента используется утилита `numactl`, ее параметры:

`-C 0-3` — список ядер, на которых будет выполняться `iperf`. Ядра должны принадлежать к тому же узлу NUMA, что и адаптер сети Альфа. Их количество должно быть не меньше, чем число параллельных потоков (`-P`).

`-m 0` — узел NUMA, на котором будет выделяться память. Должен совпадать с узлом NUMA, на котором находится адаптер сети Альфа. Это необходимо для того, чтобы избежать межсокетных обращений.

4.3.6. Отключение a3eth

Для выключение интерфейса `a3eth0` необходимо выполнить команду:

```
# systemctl stop a3net.a3eth.service
```

5. ПРОВЕРКА КОМПЛЕКСА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

5.1. Проверка работоспособности

Проверка работоспособности установленного комплекса программного обеспечения с управляющего узла выполняется следующим образом:

1) проверить, что сетевые адаптеры определились на узлах, выполнив на каждом узле команду:

   ```
   $ a3ctl dinfo list
   ```

   В результате выполнения команды на каждом узле должна быть выведена информация примерно следующего содержания:

   ```
   Found 4 devices
   PCI: 0000:31:00.0 fd: 0 chipid: 0.0
   PCI: 0000:31:00.1 fd: 1 chipid: 0.1
   PCI: 0000:32:00.0 fd: 2 chipid: 0.2
   PCI: 0000:32:00.1 fd: 3 chipid: 0.3
   ```

   Также необходимо проверить, что для каждого устройства создан файл в каталоге dev:

   ```
   $ ls -w1 /dev/a3net*
   ```

   В выводе команды должны присутствовать следующие строки:

   ```
   /dev/a3net0
   /dev/a3net1
   /dev/a3net2
   /dev/a3net3
   ```

   Если данные строки для некоторого узла выглядят иначе, данный узел необходимо перезагрузить;

2) проверить работоспособность программы запуска задач, выполнив команду:

   ```
   $ mpirun -hosts host1,host2 /usr/bin/hostname
   ```

   Корректный вывод должен содержать строки:

   ```
   host1
   host2
   ```

   Здесь вместо «`host1,host2`» должны быть имена узлов в используемой системе.

   Данная команда может завершиться неудачно, если:

   - не настроен беспарольный SSH-доступ с управляющего узла на какой-либо из вычислительных узлов;
   - у пользователя нет прав на чтение и запуск программ из `/usr/lib/a3net` (либо программы отсутствуют на каком-либо из узлов);

– на управляющем узле установлен MPI, отличный от поставляемого с сетевым оборудованием сети Альфа, и пути к `mpicc` и `mpiexec` ведут не к MPI для сетевого оборудования сети Альфа (вывод команд `which mpiexec` и `which mpicc` должен начинаться с пути `/usr/lib/a3net`); для решения данной проблемы требуется соответствующим образом разрешить конфликт выбора версии MPI, используемой по умолчанию (например, через изменение переменной окружения `PATH`);

3) запустить тест проверки работоспособности:

```
$ mpirun -hosts host1,host2 -ppn 8 bench -t pairs -e
```

Корректный вывод должен иметь примерно следующий вид:

```
Test: bandwidth Min pkt: 1      Max pkt: 256    Step: 0 Operations: 300000

Operation_type size hostname: bandwidth (Max / Min)

PUT 8B host1: 5347 Mbit/s (1.00)
PUT 8B host2: 5345 Mbit/s (1.00)

PUT 16B host1: 10682 Mbit/s (1.00)
PUT 16B host2: 10680 Mbit/s (1.01)

PUT 32B host1: 20116 Mbit/s (1.00)
PUT 32B host2: 20119 Mbit/s (1.00)

PUT 64B host1: 36997 Mbit/s (1.00)
PUT 64B host2: 36987 Mbit/s (1.00)

PUT 128B host1: 49972 Mbit/s (1.03)
PUT 128B host2: 49890 Mbit/s (1.01)

PUT 256B host1: 53224 Mbit/s (1.04)
PUT 256B host2: 53115 Mbit/s (1.01)

PUT 512B host1: 54702 Mbit/s (1.04)
PUT 512B host2: 54583 Mbit/s (1.01)

PUT 1024B host1: 54917 Mbit/s (1.02)
PUT 1024B host2: 54916 Mbit/s (1.02)

PUT 2048B host1: 54641 Mbit/s (1.02)
PUT 2048B host2: 54552 Mbit/s (1.01)
```

Данная команда может завершиться неудачно, если:

– какой-либо из кабелей, соединяющий сетевое оборудование сети Альфа с соседними узлами или с коммутатором, подсоединён неправильно (не соответствует конфигурации для таблиц маршрутов);

– возникла внутренняя ошибка (задача была снята некорректно; возник перегрев адаптера сети Альфа; возникли неустранимые ошибки при передаче по кабелю).

В этом случае необходимо на каждом узле выполнить команду “a3ctl reset” и начать выполнение проверок сначала.

5.2. Проверка работы MPI приложения

Для проверки работы MPI приложений необходимо скачать приложение Intel(R) MPI Benchmarks (например, по адресу https://github.com/intel/mpi-benchmarks). Для сборки приложения IMB-MPI1 необходимо перейти в директорию с приложением и выполнить команду:

```
$ make CC=mpicc CXX=mpicxx IMB-MPI1
```

Для запуска приложения IMB-MPI1 необходимо выполнить команду:

```
$ mpirun -hosts host1,host2 -ppn 8 ./IMB-MPI1 -npmin 16
```

Пример вывода одного из тестов:

```
#----------------------------------------------------------------
# Benchmarking Alltoall
# #processes = 16
#----------------------------------------------------------------
       #bytes #repetitions  t_min[usec]  t_max[usec]  t_avg[usec]
            0         1000         0.05         0.06         0.06
            1         1000         9.33        13.63        12.21
            2         1000         9.54        13.77        12.50
            4         1000         9.84        14.43        13.06
            8         1000         9.45        14.33        12.87
           16         1000         9.48        14.26        12.84
           32         1000        10.09        15.29        13.76
           64         1000        10.96        16.76        15.09
          128         1000        12.65        18.83        17.00
          256         1000        16.29        23.25        21.17
          512         1000        15.77        18.23        17.04
         1024         1000        23.36        27.51        25.33
         2048         1000        35.48        45.40        40.80
         4096         1000        61.23        79.40        72.60
         8192         1000       118.75       150.51       139.74
        16384         1000       222.96       282.06       263.53
        32768         1000       427.70       544.34       510.68
        65536          640      1036.66      1055.52      1047.26
       131072          320      2371.92      2385.27      2379.84
       262144          160      5832.13      5856.70      5845.53
       524288           80     13422.51     13469.49     13444.61
      1048576           40     27024.48     27104.11     27063.41
      2097152           20     53762.80     53916.85     53846.08
      4194304           10    107347.03    107629.47    107489.40
```

Проверка считается успешной, если все тесты отработали без ошибок.

5.3. Получение информации об адаптере как о PCIe-устройстве

Получение информации об адаптере как о PCIe-устройстве осуществляется с помощью стандартной утилиты `lspci` с опцией «`-d *:0250`» (наиболее полная информация доступна из-под пользователя `root`).

Пример — Вывод команды lspci для адаптера с одним PCIe Endpoint:

```
# lspci -vvv -d *:0250
10:00.0 Network controller: Xilinx Corporation Device 0250
        Subsystem: Xilinx Corporation Device 0000
        Control: I/O- Mem- BusMaster- SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B-
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR-
        Region 0: Memory at ea000000 (32-bit, non-prefetchable) [disabled] [size=32M]
        Region 1: Memory at ed000000 (32-bit, non-prefetchable) [disabled] [size=16M]
        Capabilities: [80] Express (v2) Endpoint, MSI 00
                DevCap:         MaxPayload 512 bytes, PhantFunc 0, Latency L0s <64ns, L1 <1us
                        ExtTag+ AttnBtn- AttnInd- PwrInd- RBE+ FLReset+ SlotPowerLimit 75.000W
                DevCtl:         Report errors: Correctable- Non-Fatal- Fatal- Unsupported-
                        RlxdOrd+ ExtTag+ PhantFunc- AuxPwr- NoSnoop+ FLReset-
                        MaxPayload 512 bytes, MaxReadReq 512 bytes
                DevSta:         CorrErr+ UncorrErr- FatalErr- UnsuppReq+ AuxPwr- TransPend-
                LnkCap:         Port #0, Speed 8GT/s, Width x8, ASPM L0s, Exit Latency L0s <256ns
                        ClockPM- Surprise- LLActRep- BwNot- ASPMOptComp+
                LnkCtl:         ASPM Disabled; RCB 64 bytes Disabled- CommClk-
                        ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
                LnkSta:         Speed 8GT/s, Width x8, TrErr- Train- SlotClk- DLActive- BWMgmt- ABWMgmt-
                DevCap2: Completion Timeout: Not Supported, TimeoutDis-, LTR-, OBFF Not Supported
                         AtomicOpsCap: 32bit- 64bit- 128bitCAS-
                DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis-, LTR-, OBFF Disabled
                         AtomicOpsCtl: ReqEn-
                LnkCtl2: Target Link Speed: 8GT/s, EnterCompliance- SpeedDis-
                         Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-
                         Compliance De-emphasis: -6dB
                LnkSta2: Current De-emphasis Level: -3.5dB, EqualizationComplete+, EqualizationPhase1+
                         EqualizationPhase2+, EqualizationPhase3+, LinkEqualizationRequest-
        Capabilities: [d0] MSI-X: Enable- Count=512 Masked-
                Vector table: BAR=1 offset=00010000
                PBA: BAR=1 offset=00012000
        Capabilities: [e0] MSI: Enable- Count=1/32 Maskable+ 64bit+
                Address: 0000000000000000  Data: 0000
                Masking: 00000000  Pending: 00000000
        Capabilities: [f8] Power Management version 3
                Flags: PMEClk- DSI- D1+ D2+ AuxCurrent=0mA PME(D0+,D1+,D2+,D3hot+,D3cold+)
                Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=0 PME-
        Capabilities: [100 v1] Vendor Specific Information: ID=1556 Rev=1 Len=008 <?>
        Capabilities: [128 v1] Alternative Routing-ID Interpretation (ARI)
                ARICap:         MFVC- ACS-, Next Function: 1
                ARICtl:         MFVC- ACS-, Function Group: 0
        Capabilities: [140 v1] Single Root I/O Virtualization (SR-IOV)
                IOVCap:         Migration-, Interrupt Message Number: 000
                IOVCtl:         Enable- Migration- Interrupt- MSE- ARIHierarchy+
                IOVSta:         Migration-
                Initial VFs: 3, Total VFs: 3, Number of VFs: 0, Function Dependency Link: 00
                VF offset: 8, stride: 1, Device ID: 0250
                Supported Page Size: 00000553, System Page Size: 00000001
                VF Migration: offset: 00000000, BIR: 0
        Capabilities: [1b0 v1] Transaction Processing Hints
                Device specific mode supported
                No steering table available
        Capabilities: [1c0 v1] Access Control Services
                ACSCap:         SrcValid- TransBlk- ReqRedir- CmpltRedir- UpstreamFwd- EgressCtrl+
                ACSCtl:         SrcValid- TransBlk- ReqRedir- CmpltRedir- UpstreamFwd- EgressCtrl-
        Capabilities: [200 v2] Advanced Error Reporting
                UESta:        DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC-
                UEMsk:        DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC-
                UESvrt:        DLP+ SDES- TLP- FCP+ CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP+ ECRC-
                CESta:        RxErr- BadTLP- BadDLLP- Rollover- Timeout- NonFatalErr-
                CEMsk:        RxErr- BadTLP- BadDLLP- Rollover- Timeout- NonFatalErr+
```

```
        AERCap:       First Error Pointer: 00, ECRCGenCap- ECRCGenEn- ECRCChkCap- ECRCChkEn-
               MultHdrRecCap- MultHdrRecEn- TLPPfxPres- HdrLogCap-
        HeaderLog: 04000001 0000230f 10070000 15d2be18
Capabilities: [300 v1] #19
Capabilities: [404 v1] #15
Capabilities: [800 v1] Vendor Specific Information: ID=dfea Rev=1 Len=038 <?>
```

5.4. Инструменты настройки и диагностики адаптера

В этом разделе перечислены некоторые команды утилиты `a3ctl`.

5.4.1. Получение общей информации об адаптере

```
$ a3ctl dinfo
```

Пример вывода:

```
id         name                raw         interpretation
 0 svn_revision         0x152               r338
 1 fw_build_time_utc    0x69a5c477          2026.03.02-17:10
22 fw_build_time_local  0x69a5c477          2026.03.02-20:10 (MSK +0300)
 2 version              0x1010112           router: 1.2 packet: 1 regmap: 1.1
 8 dims                 0x2                 2
 9 line_per_tx          0x2                 2
10 core_width           0x80                128
11 pcie_endpoint_count  0x4                 4
12 pcie_injection_count 0x4                 4
13 segment_count        0xb                 2048
14 qsfp_count           0x2                 2
15 qsfp_width           0x8                 8
16 tx_count             0x2                 2
17 core_count           0x1                 1
18 injbuf_size          0x9                 512
19 injbuf_cred_size     0xb                 11
20 flags                0x3c000000009       mask: 0x000003c0
21 type                 0x0                 lp
```

Основные поля:

`fw_buid_time` – отображает дату и время сборки прошивки ПЛИС;

`version` – отображает версию прошивки ПЛИС;

`pcie_endpoint_count` – отображает количество PCIe Endpoint;

`type` – отображает тип модуля.

Команда поддерживает вывод в формате json:

```
$ a3ctl dinfo -j
```

5.4.2. Получение информации о lid модуля (a3sn)

```
$ a3ctl dinfo a3sn
```

Пример вывода:

```
a3sn: 0x0000a
```

Данный lid используется при получении информации о соседнем узле (см. далее). Наличие слова `err` в выводе означает неправильный или незаданный lid модуля.

Команда поддерживает вывод в формате json:

```
$ a3ctl dinfo -j a3sn
```

Пример вывода:

```
{
    "a3sn": 10,
    "flags": 0,
    "core": 0,
    "type": 0,
    "crc": 11,
    "raw": 2952790026,
    "has_error": false,
    "errors": [
    ]
}
```

Значение поля `has_error: true` означает неправильный или незаданный lid модуля.

5.4.3. Получение информации с трансивера QSFP

```
$ a3ctl port -vvv qsfp
```

Пример вывода команды:

```
Port0:    undef
Port1:
    type:          DD
    vendor_name:   Amphenol
    part_number:   NDYYYF-0002
    revision:      N
    serial_number: APF24100024D8B
    Specification: 5.00v
    Up page type:  Paged
    Media Type:    Passive Copper Cable
    Temperature:   -1.00C
    Voltage 3.3v:  -1.00v
```

undef показывает, что не удалось прочитать информацию, например, если кабель не подключен. Параметр type указывает тип кабеля: DD для QSFP-DD, Single для QSFP. Если какие-то значения не удается прочитать, будет выведено значение -1.00. Трансиверы разных производителей предоставляют разное количество информации, отдельные модели не предоставляют никакой информации.

Для полноценной работы сети необходимо использовать кабели типа QSFP-DD.

5.4.4. Просмотр текущего статуса портов

```
$ a3ctl port info st
```

Пример вывода команды:

```
Port:0
    status: ready;
    power: Enable;
    retry_count: Exception counter is disabled;
    break_count: Exception counter is disabled;
    max_break_time: 1302422475;
    dropped_flits: 0;
    dropped_packages: 0;
    neighbor: 0x00002;
Port:1
    status: broken;
    power: Disable;
    retry_count: Exception counter is disabled;
    break_count: Exception counter is disabled;
    max_break_time: 0;
    dropped_flits: 0;
    dropped_packages: 0;
    neighbor: err;
```

Статус broken означает, что данный порт не подключен кабелем к порту исправно работающего устройства, либо что линк выключен.

Команда поддерживает вывод в формате json:

```
$ a3ctl port -j info st
```

5.4.5. Управление питанием портов

Получение информации о питании портов адаптера:

```
$ a3ctl port power status
```

Пример вывода команды:

```
Port:0
    power: Disable;
```

```
Port:1
    power: Enable;
```

Управление питанием портов:

```
$ a3ctl port -p <N> power <state>
```

где `<N>` – номер порта;

`<state>` – устанавливаемое значение (`enable` – включить, `disable` – выключить).

5.4.6. Просмотр информации об ответных портах

```
$ a3ctl port info nb
```

Пример вывода команды:

```
Port:0
    neighbor: 0x100001;
Port:1
    neighbor: err;
```

Число (`0x100001`) – это lid(a3sn) устройства с другой стороны линка. Вывод `err` значит, что не удалось получить информацию с другой стороны линка, либо она некорректна. Это может свидетельствовать об отсутствии соединения, либо устройство с другой стороны не сконфигурировано.

Команда поддерживает вывод в формате json:

```
$ a3ctl port -j info nb
```

5.4.7. Просмотр суммарной статистики по портам

```
$ a3ctl port counters
```

Пример вывода команды:

```
Counters:
    Port:0 416385717438 flit
    Port:1 0 flit
    Port:0 13339533735 packet
    Port:1 0 packet
```

5.4.8. Просмотр счетчиков переповторов и обрывов связи

Сначала нужно включить сбор статистики по данным метрикам:

```
$ a3ctl except -e
```

```
$ a3ctl except -E all
```

После этого появится дополнительная информация в статусе порта:

```
$ a3ctl port info st
```

```
Port:0
    status: ready;
    power: Enable;
    retry_count: 0;
    break_count: 0;
    max_break_time: 2787991713;
    dropped_flits: 0;
    dropped_packages: 0;
    neighbor: 0x00002;
Port:1
    status: broken;
    power: Disable;
    retry_count: 0;
    break_count: 0;
    max_break_time: 0;
    dropped_flits: 0;
    dropped_packages: 0;
    neighbor: err;
```

`retry_count` показывает число переповторов на порте, `break_count` показывает число обрывов на порте. Данные параметры обнуляются с периодом примерно 10 секунд. Стабильный рост значений `retry_count` и/или `break_count` в рамках периода может свидетельствовать о повреждении порта или кабеля.

`dropped_flits(packages)` показывает число отброшенных флитов (пакетов) на порте. Возрастающее число может свидетельствовать о перегрузке сети или повреждении кабеля.

Команда поддерживает вывод в формате json:

```
$ a3ctl port -j info st
```

5.4.9. Просмотр показаний датчиков температуры

```
$ a3ctl mcu temp
```

Пример вывода команды:

```
Temperature:
   PSU_1V0              52.000 C
   Board                47.250 C
   Chip                 52.250 C
```

`PSU_1V0` показывает температуру источника питания 1 В, `Board` показывает температуру платы адаптера, `Chip` показывает температуру ПЛИС. Температура ПЛИС выше 75 °C считается повышенной, при приближении к ней необходимо улучшить охлаждение адаптера. Температура ПЛИС 85 °C является критической, при её достижении отключается питание модуля.

5.4.10. Просмотр показаний датчиков напряжения

```
$ a3ctl mcu adc
```

Пример вывода команды:

```
ADC:
   ADC_1V2              1.221
   ADC_1V8              1.845
   ADC_1V8_AUX          1.832
```

Здесь показаны значения напряжения питания периферийных модулей ПЛИС. Нормальное значение напряжений соответствует числам в названиях датчиков. Например, для `ADC_1V2` нормой является 1.2 В.

Отклонение более чем на 10% от нормы может свидетельствовать о неисправности питания адаптера, либо внутренних цепей адаптера.

6. СООБЩЕНИЯ СИСТЕМНОМУ ПРОГРАММИСТУ

6.1. Типы сообщений

Возможные сообщения об ошибках делятся на:

– сообщения при установке комплекса программного обеспечения;

– сообщения при настройке комплекса программного обеспечения;

– сообщения при запуске программ;

– сообщения при выполнении программ.

6.2. Сообщения при установке комплекса программного обеспечения

Программа установки (`install.sh`) выводит информацию о текущем этапе установки и результат завершения этапа: `OK` (успешно) или `Fail` (неуспешно). В конце работы программы установки выводится сообщение `Installation successful` в случае успеха или `Installation failed` в случае возникновения ошибок.

Пример успешного завершения:

```
$ sudo ./install.sh


Checking:
 - a3eth
   - Checking kernel version 6.12.68-6.12-alt1: OK
 - a3run
   - Checking perl dependency: OK
 - a3driver
   - Checking kernel version 6.12.68-6.12-alt1: OK

Installing:
 - general
    - Applying assets  OK
 - a3reg_mmap
    - Unpacking OK
 - a3ctl
    - Unpacking OK
    - Applying assets  OK
 - a3eth
    - Unpacking OK
    - Applying assets  OK
 - a3verbs
    - Unpacking OK
 - a3run
    - Unpacking OK
 - bench
    - Unpacking OK
 - shmem
    - Unpacking OK
 - a3reg_verbs
    - Unpacking OK
```

```
 - a3driver
    - Unpacking OK
    - Applying assets  OK

Configuring:
 - a3driver
    - Configuring OK

Enable/Start:

Installation successful
```

Пример ошибки:

```
$ sudo ./install.sh


Checking:
 - a3eth
   - Checking kernel version 6.12.68-6.12-alt1: Fail

FAIL
Installation failed
More info: /home/user/tarball_20260305_1307/install.log
```

6.3. Сообщения при настройке комплекса программного обеспечения

Специальных сообщений при настройке комплекса программного обеспечения не предусмотрено. Настройка комплекса программного обеспечения происходит с помощью общесистемных утилит. Информацию о сообщениях следует искать в документации к используемой операционной системе.

6.4. Сообщения при запуске тестов

При запуске программы для тестирования работоспособности и измерения производительности сети корректный вывод имеет следующий вид (для двух взаимодействующий узлов):

```
Test: bandwidth Min pkt: 1      Max pkt: 256    Step: 0 Operations: 300000

Operation_type size hostname: bandwidth (Max / Min)

PUT 8B host1: 5347 Mbit/s (1.00)
PUT 8B host2: 5345 Mbit/s (1.00)

PUT 16B host1: 10682 Mbit/s (1.00)
PUT 16B host2: 10680 Mbit/s (1.01)

PUT 32B host1: 20116 Mbit/s (1.00)
PUT 32B host2: 20119 Mbit/s (1.00)

PUT 64B host1: 36997 Mbit/s (1.00)
PUT 64B host2: 36987 Mbit/s (1.00)
```

```
PUT 128B host1: 49972 Mbit/s (1.03)
PUT 128B host2: 49890 Mbit/s (1.01)

PUT 256B host1: 53224 Mbit/s (1.04)
PUT 256B host2: 53115 Mbit/s (1.01)

PUT 512B host1: 54702 Mbit/s (1.04)
PUT 512B host2: 54583 Mbit/s (1.01)

PUT 1024B host1: 54917 Mbit/s (1.02)
PUT 1024B host2: 54916 Mbit/s (1.02)

PUT 2048B host1: 54641 Mbit/s (1.02)
PUT 2048B host2: 54552 Mbit/s (1.01)
```

6.5. Сообщения при запуске программ

При запуске программ возможны следующие ошибки:

- `Error: invalid option ...` — некорректные опции запуска;
- `Error: program ... not found.` — файл исполняемой программы либо не найден, либо нет разрешения на его исполнение;
- `Error: host list is empty (use -H option).` — не задан список узлов для запуска программы (обязательный параметр);
- `Error: no executable given.` — в строке запуска не указан файл исполняемой программы.

6.6. Сообщения при выполнении программ

Программы, использующие параллельные библиотеки SHMEM и MPI, могут выдавать следующие сообщения об ошибках:

- `A3VERBS error: failed to get route to the host ...` — ошибка при старте низкоуровневой библиотеки передачи сообщений; основной причиной может быть некорректная настройка, либо некорректная установка комплекса программного обеспечения;
- `A3VERBS error: failed to open device ...` — не удалось открыть драйвер сетевого оборудования сети Альфа (либо некорректный формат драйвера, либо драйвер отсутствует);
- `A3VERBS error: ioctl failed ...` — ошибка запроса к драйверу (либо некорректный формат драйвера, либо используется драйвер другой версии, либо запрашиваемый ресурс занят);
- `A3VERBS error: failed to obtain injbuf` — не удалось выделить ресурсы для работы с сетевым оборудованием сети Альфа;

– `Timeout copy pkt to injbuf ...` — превышено время ожидания отправки сообщений; скорее всего, сломан линк сетевого оборудования сети Альфа.

6.7. Дополнительные сведения

Если происходила перекоммутация сети, может потребоваться выполнить команду «`a3ctl reset`» на тех узлах, где происходили изменения в подключении кабелей.

ПРИЛОЖЕНИЕ 1

ПРИНЯТОЕ ФОРМАТИРОВАНИЕ

Ниже приведены пояснения к различным видам форматирования, принятым в данном документе.

Пути к директориям, утилиты, пароли и прочие наименования, на которые следует обратить внимание, выделяются особым начертанием:

Перейдите в директорию `/home/user`.

Ввод команды от имени пользователя (`user`) выделяется знаком `$` в начале строки:

Выполните команду:

```
$ cd /home/user
```

Ввод команды от имени администратора (`root`) выделяется знаком `#` в начале строки:

Выполните команду:

```
# cd /home/user
```

Вывод содержимого файла или пример вывода выполненной команды показываются следующим образом:

Создайте файл `example.sh` со следующим содержимым:

```
#!/bin/sh
echo "Hello world!"
```