Проект МГВС моделирования экзафлопсного суперкомпьютера

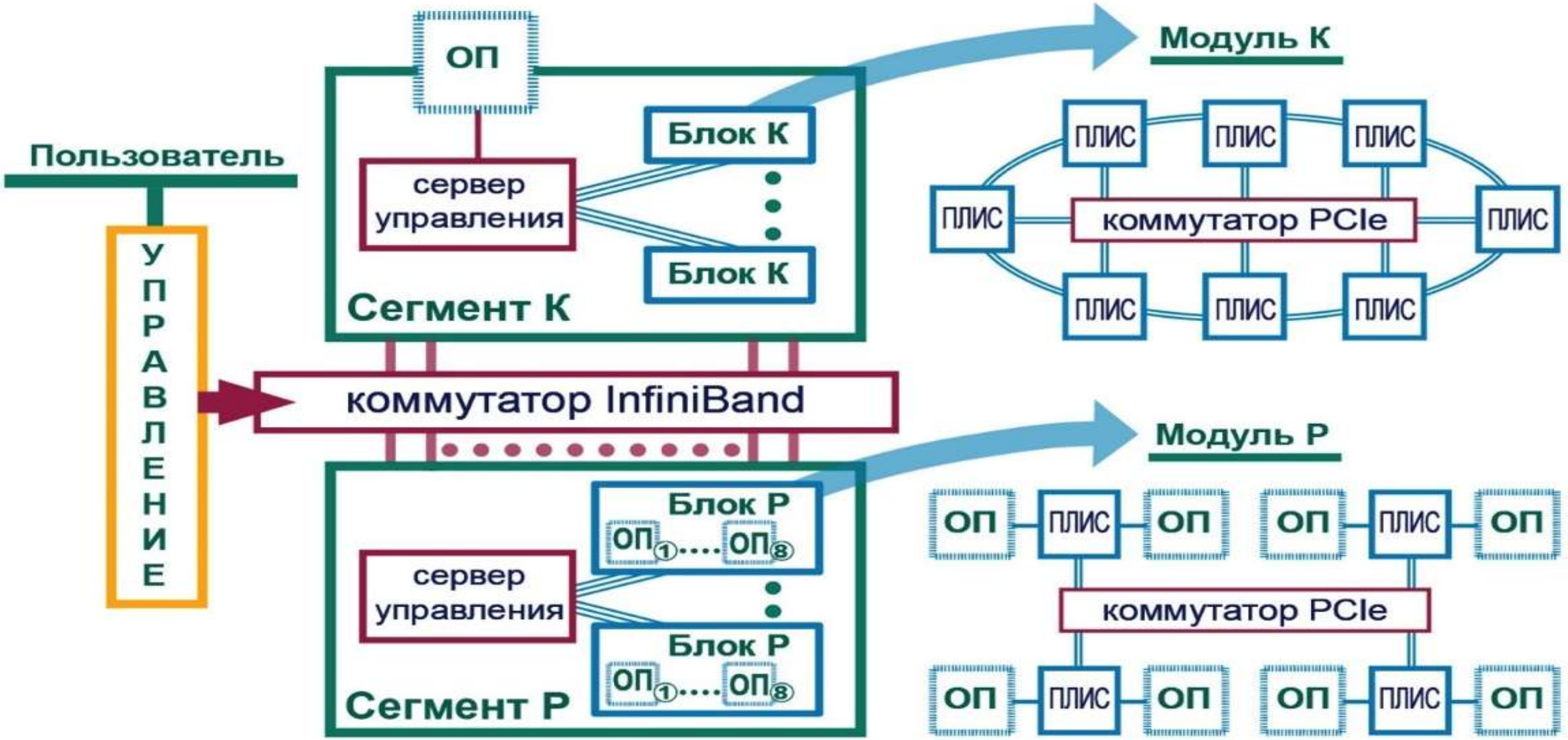
В.В. Корнеев, С.Г. Елизаров

ФГУП «НИИ «Квант», МГУ им. М.В. Ломоносова

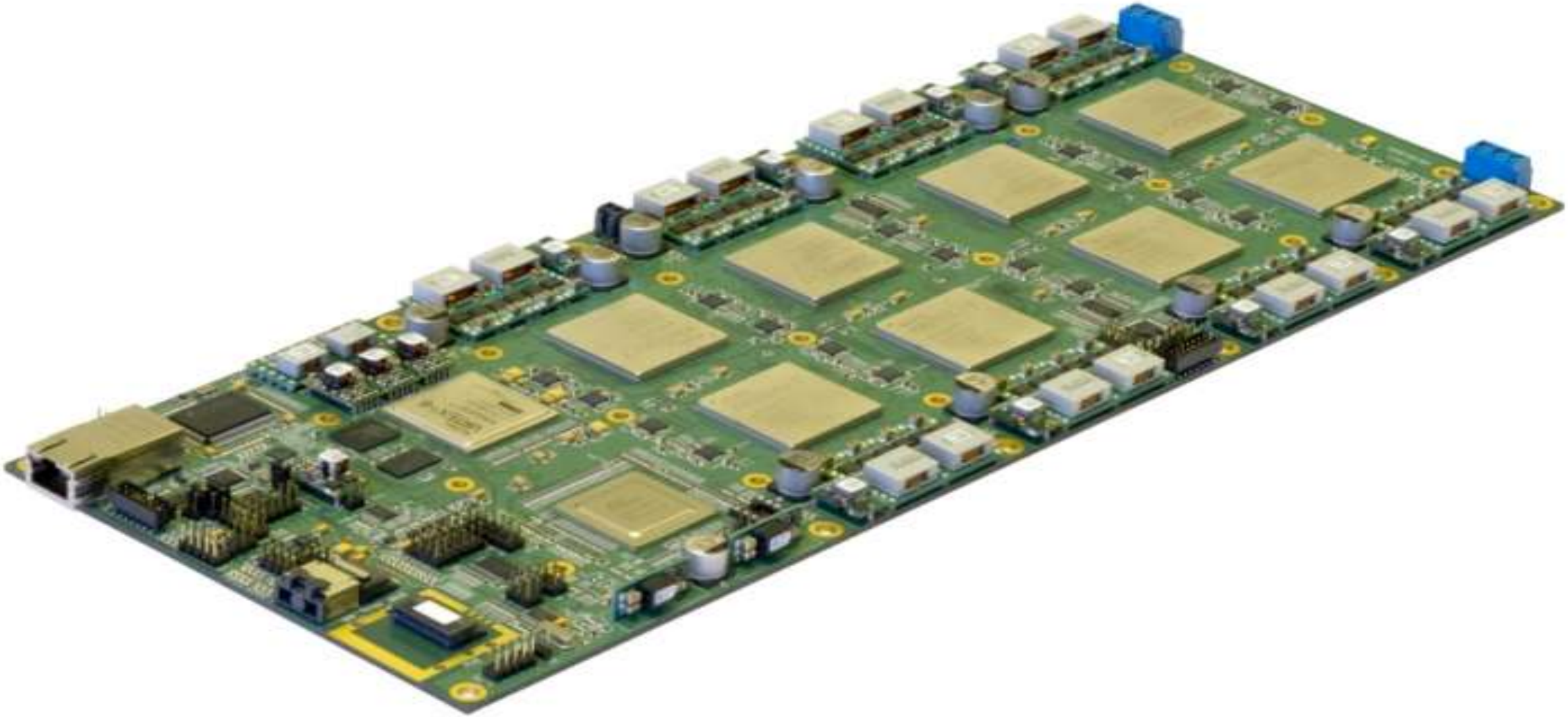
Цель проекта МГВС

- создание моделирующей гибридной вычислительной системы (МГВС) для разработки имитационных и эмуляционных моделей высокопроизводительных систем с инновационной архитектурой;
- создание моделей наиболее характерных вариантов суперкомпьютеров со сверхвысоким уровнем распараллеливания вычислений в рамках работ по созданию в РФ вычислительных систем с производительностью на уровне 10^{18} оп/с.;
- разработка моделей и средств программирования суперкомпьютеров со сверхвысоким уровнем распараллеливания вычислений, а также систем поддержки времени исполнения таких программ (runtime systems).

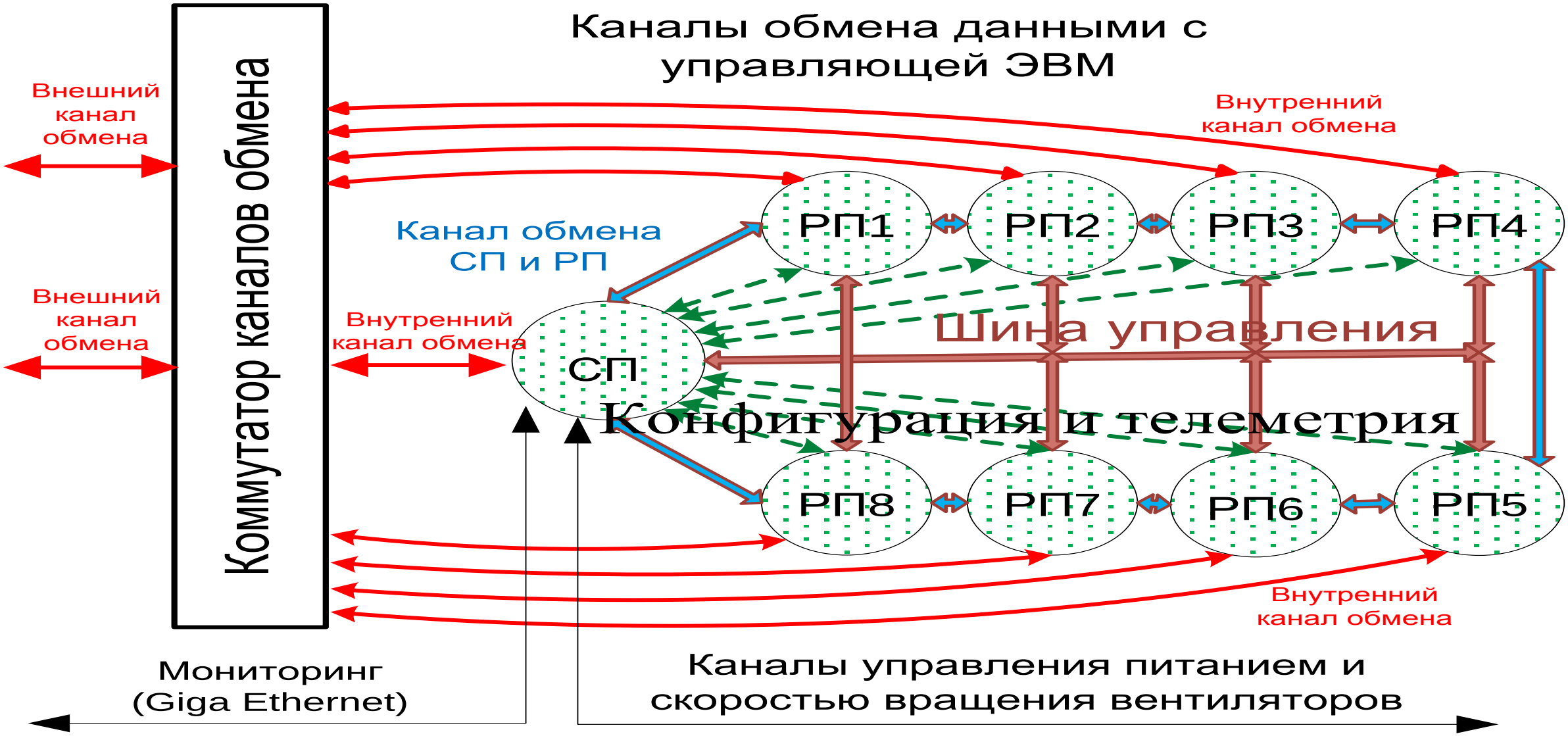
Структура МГВС



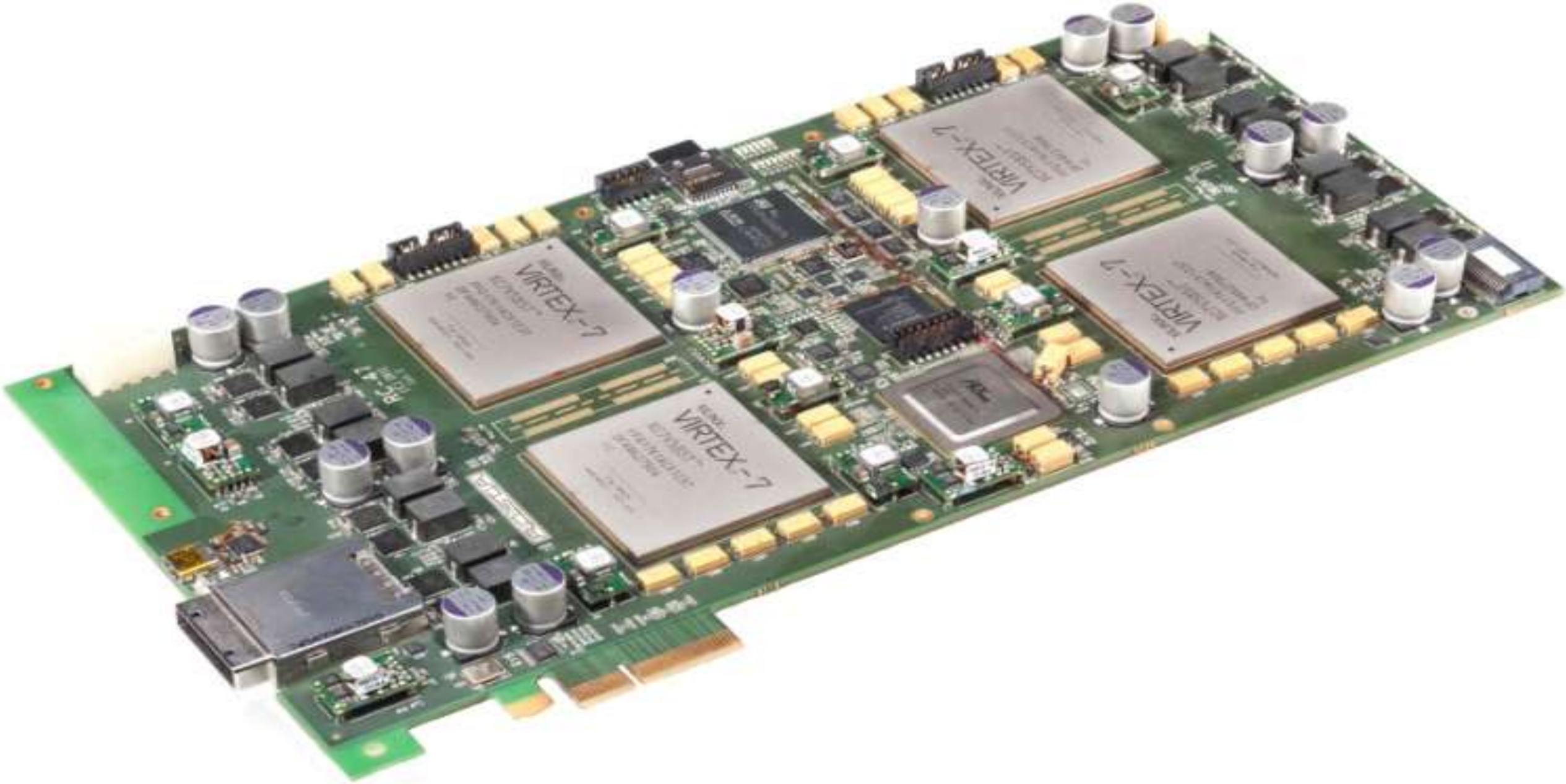
Реконфигурируемый модуль К



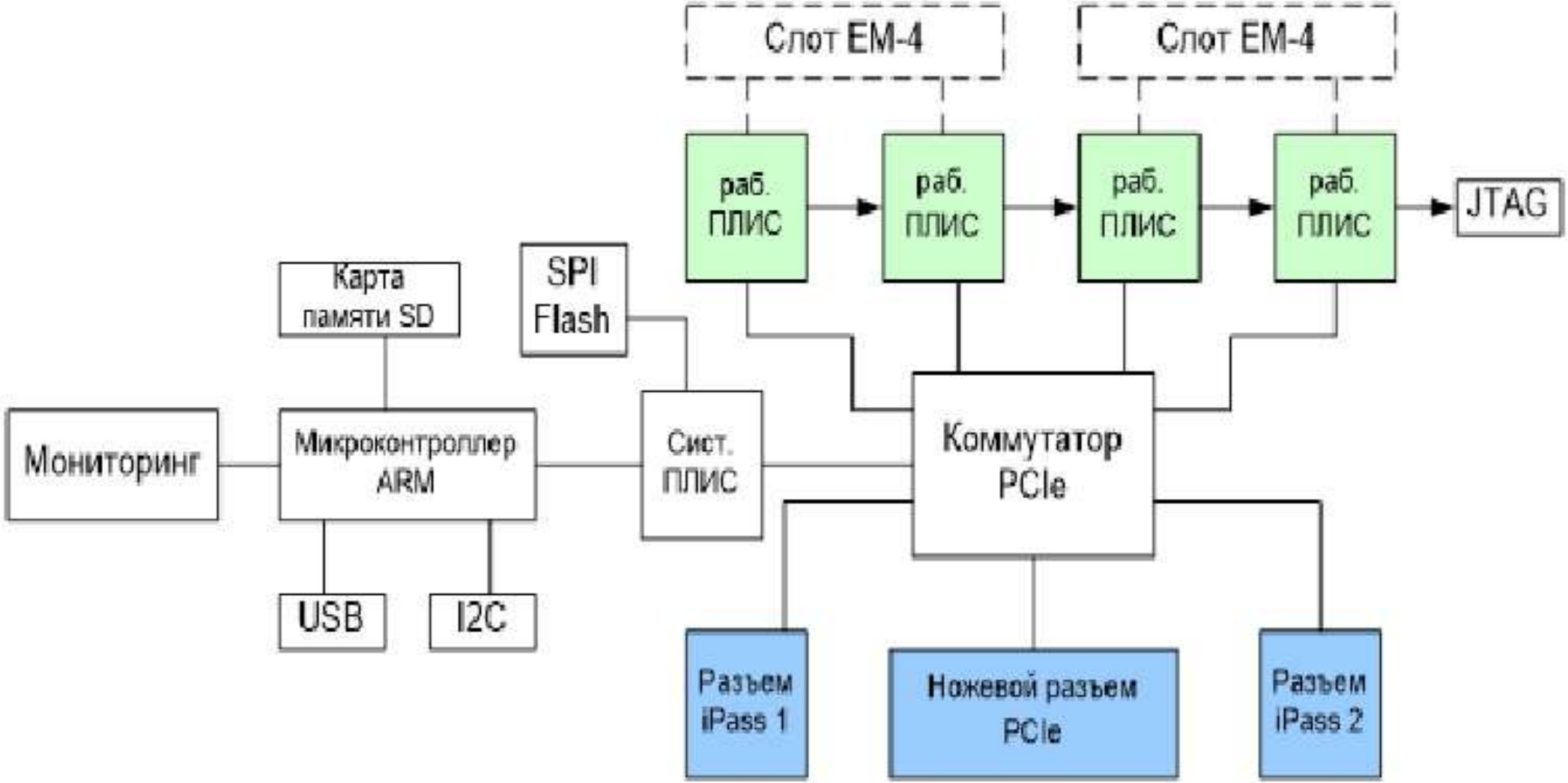
Структура модуля К

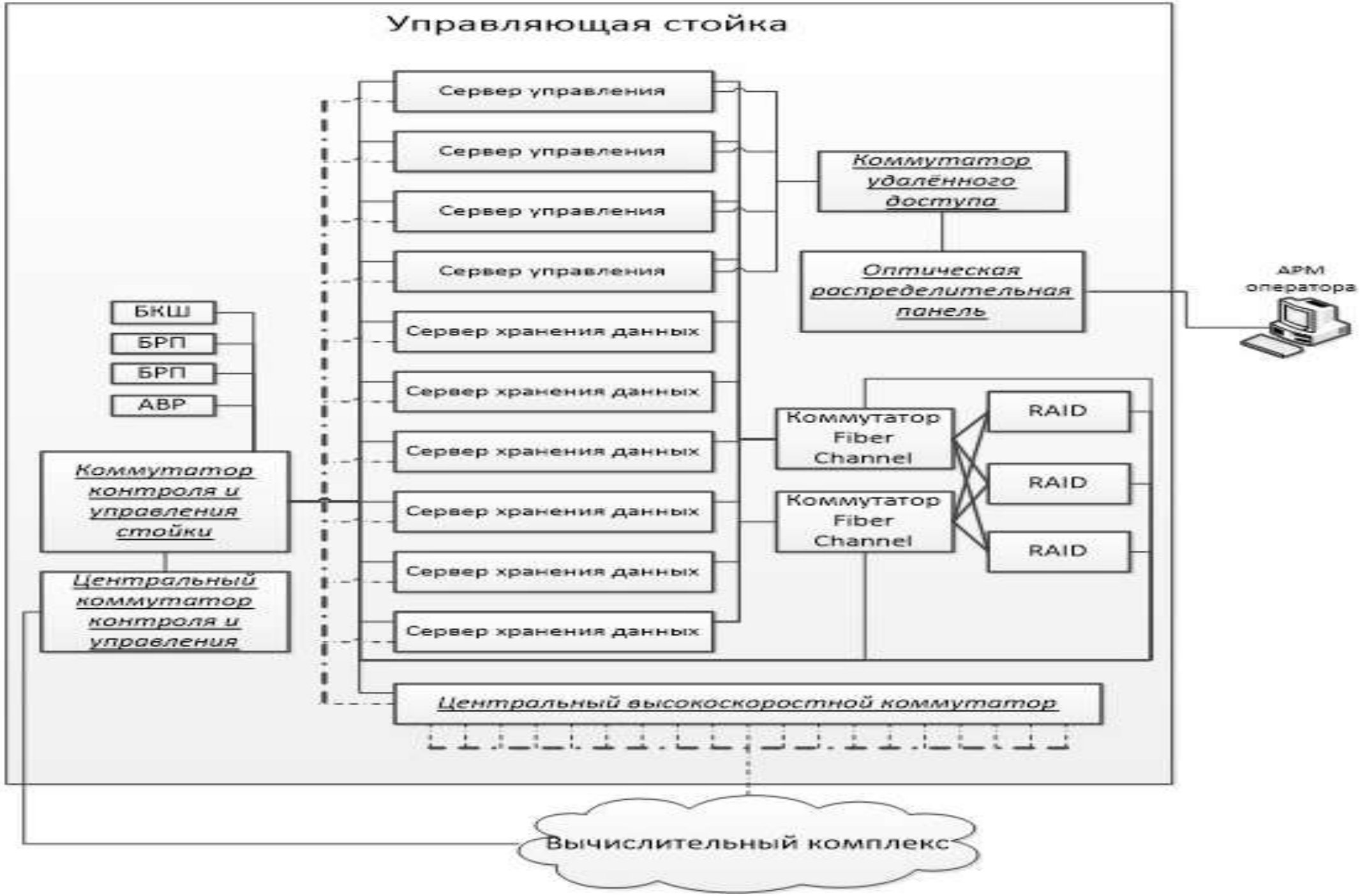


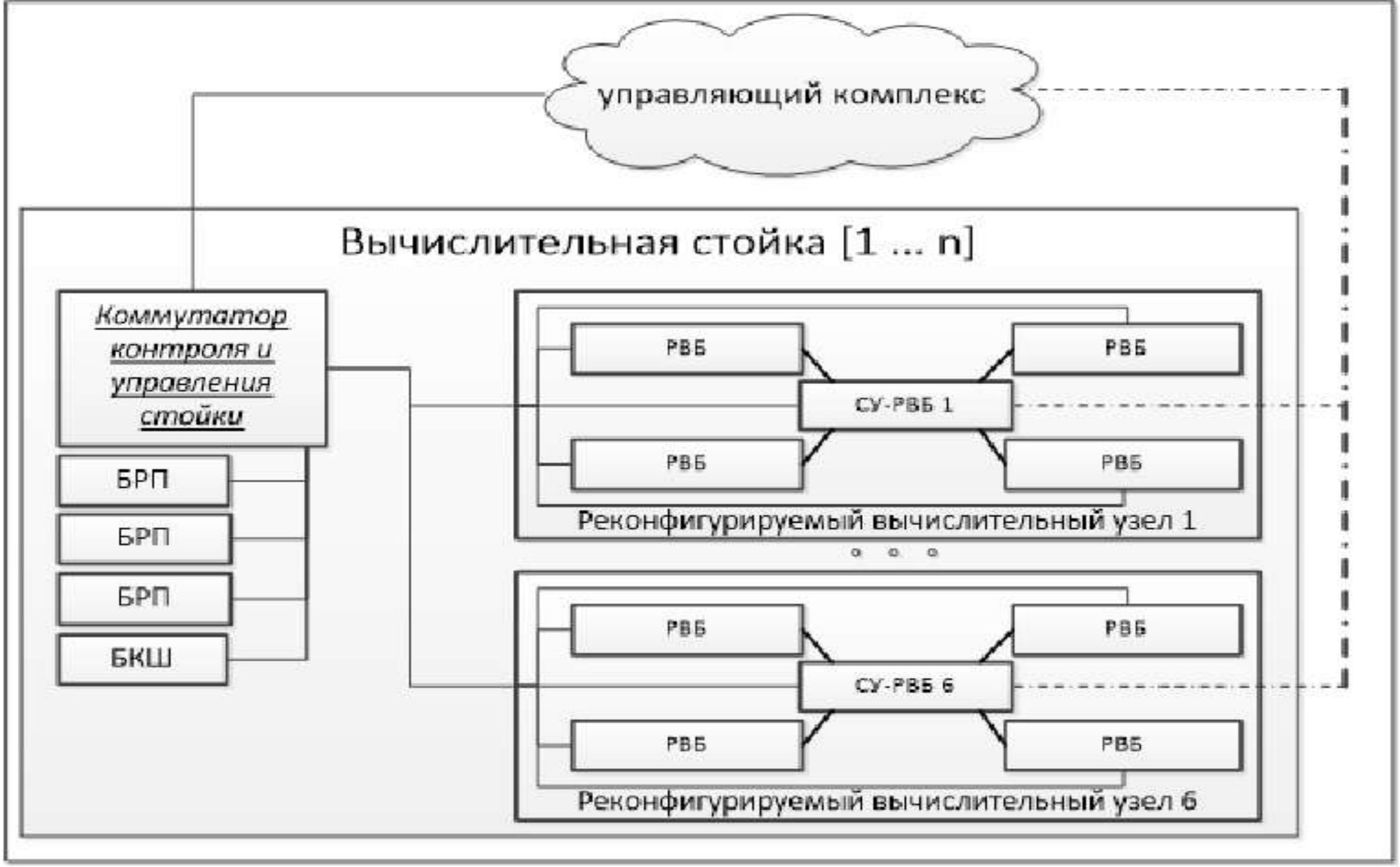
Реконфигурируемый модуль Р



Структура модуля P




Управляющая стойка
Сервер управления
Сервер управления
Сервер управления
Сервер управления
Сервер хранения данных
Сервер хранения данных
Сервер хранения данных
Сервер хранения данных
Сервер хранения данных
Сервер хранения данных
Коммутатор удалённого доступа
Оптическая распределительная панель
АРМ оператора
БКШ
БРП
БРП
АВР
Коммутатор контроля и управления стойки
Центральный коммутатор контроля и управления
Коммутатор Fiber Channel
Коммутатор Fiber Channel
RAID
RAID
RAID
Центральный высокоскоростной коммутатор
Вычислительный комплекс


управляющий комплекс
Вычислительная стойка [1 ... n]
Коммутатор контроля и управления стойки
БРП
БРП
БРП
БКШ
РВБ
РВБ
СУ-РВБ 1
РВБ
РВБ
Реконфигурируемый вычислительный узел 1
РВБ
РВБ
СУ-РВБ 6
РВБ
РВБ
Реконфигурируемый вычислительный узел 6

Восток-1 (Сегмент Р)

План Проекта (на 3 года)

Выполнено

10 cores
160 kb local memory
128 Mb shared memory
Int. bus up to 0.8 Gb/s
Ext. mem. up to 0.2 Gb/s

Basic system
on 1 x Virtex5
Sept. 2011-12

Выполнено

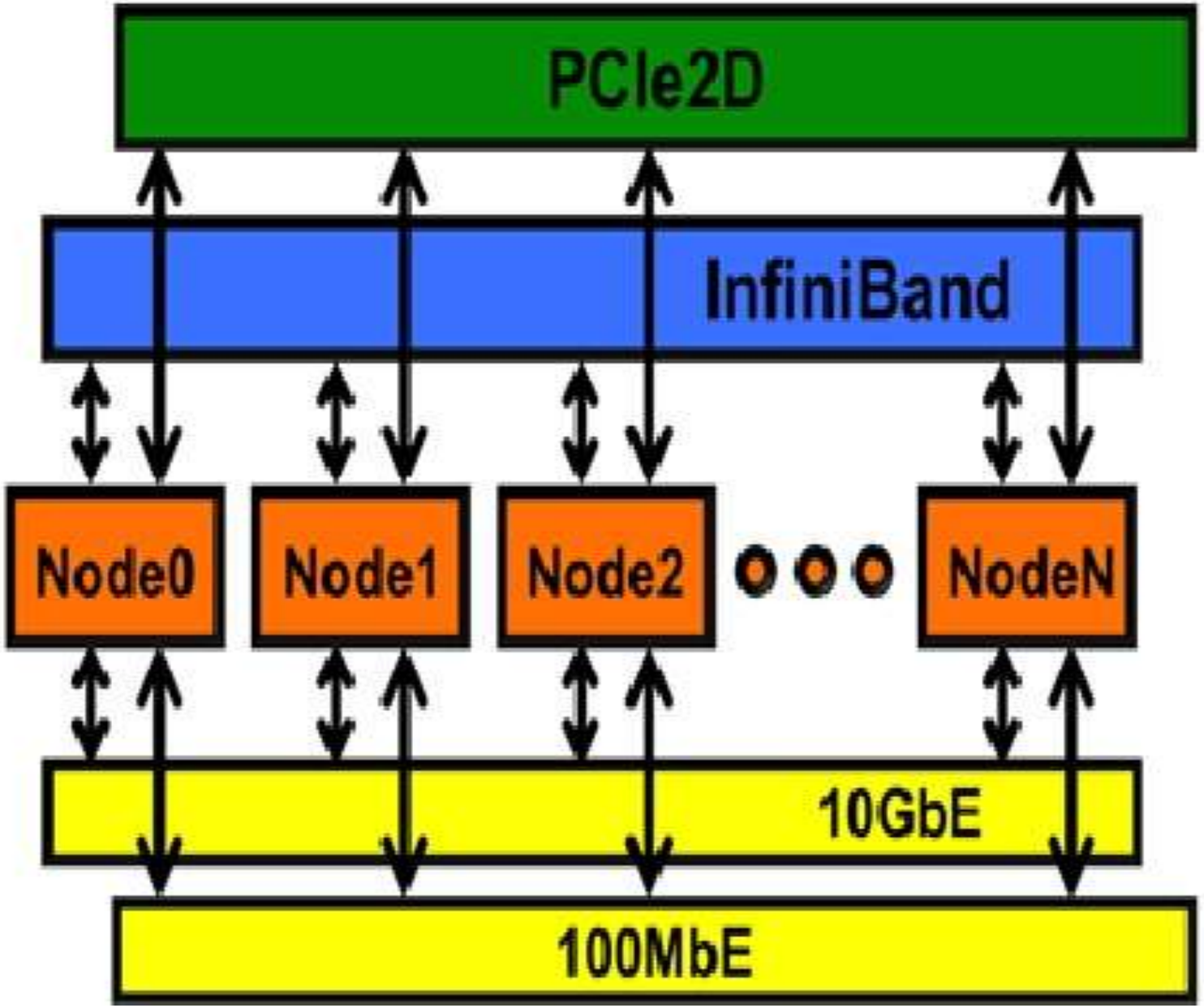
100 cores
1.6 Mb local memory
1 Gb shared memory
Int. bus up to 12.8 Gb/s
Ext. mem. up to 1.6 Gb/s

Advanced system
on 2 x Virtex6
Sept. 2012-13

1000 cores
16 Mb
local memory
128 Gb
shared memory
Int. bus up
to 200 Gb/s
Ext. mem.
up to
25 Gb/s

1000 core module
on 4 x Virtex7
Sept. 2013-14

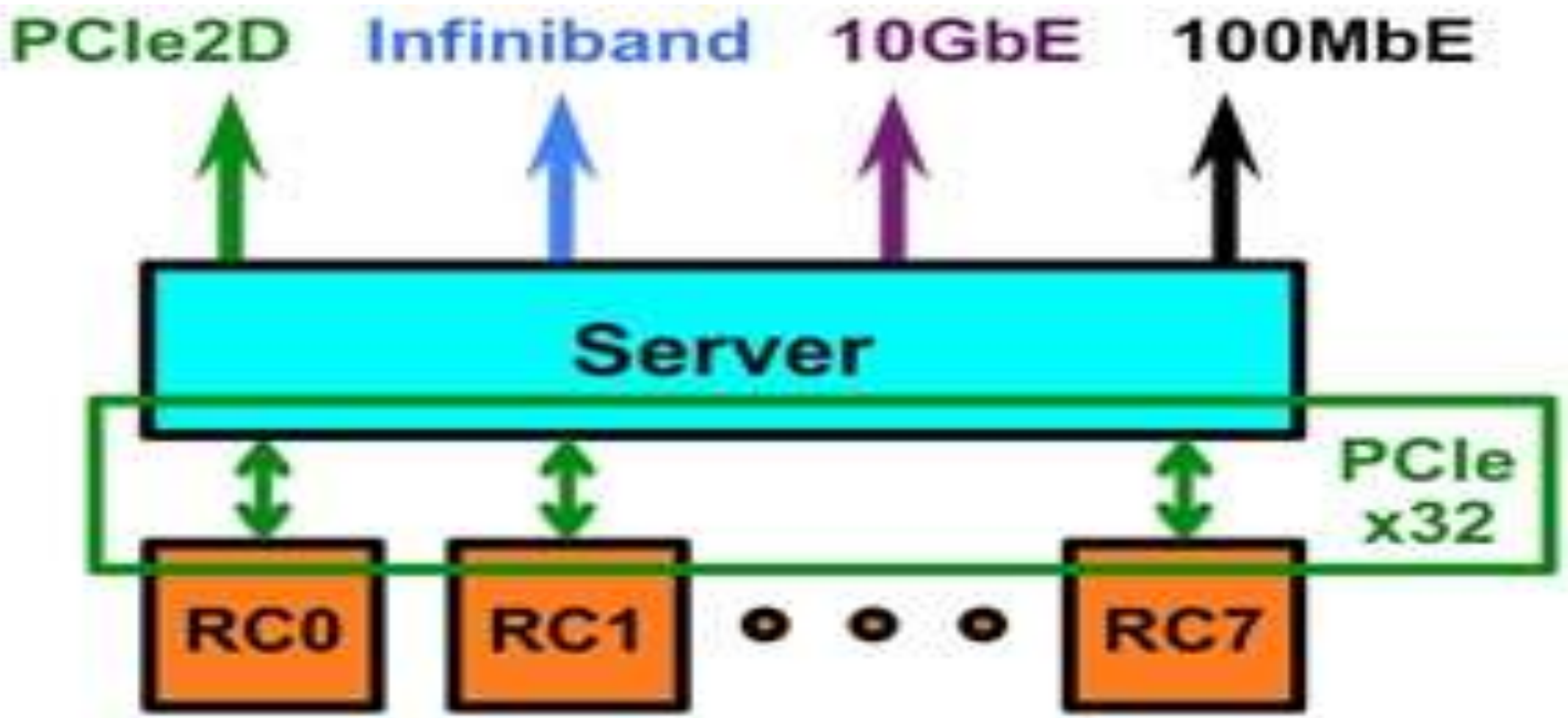
Структура Восток-1



PCIe Gen3 4x lane

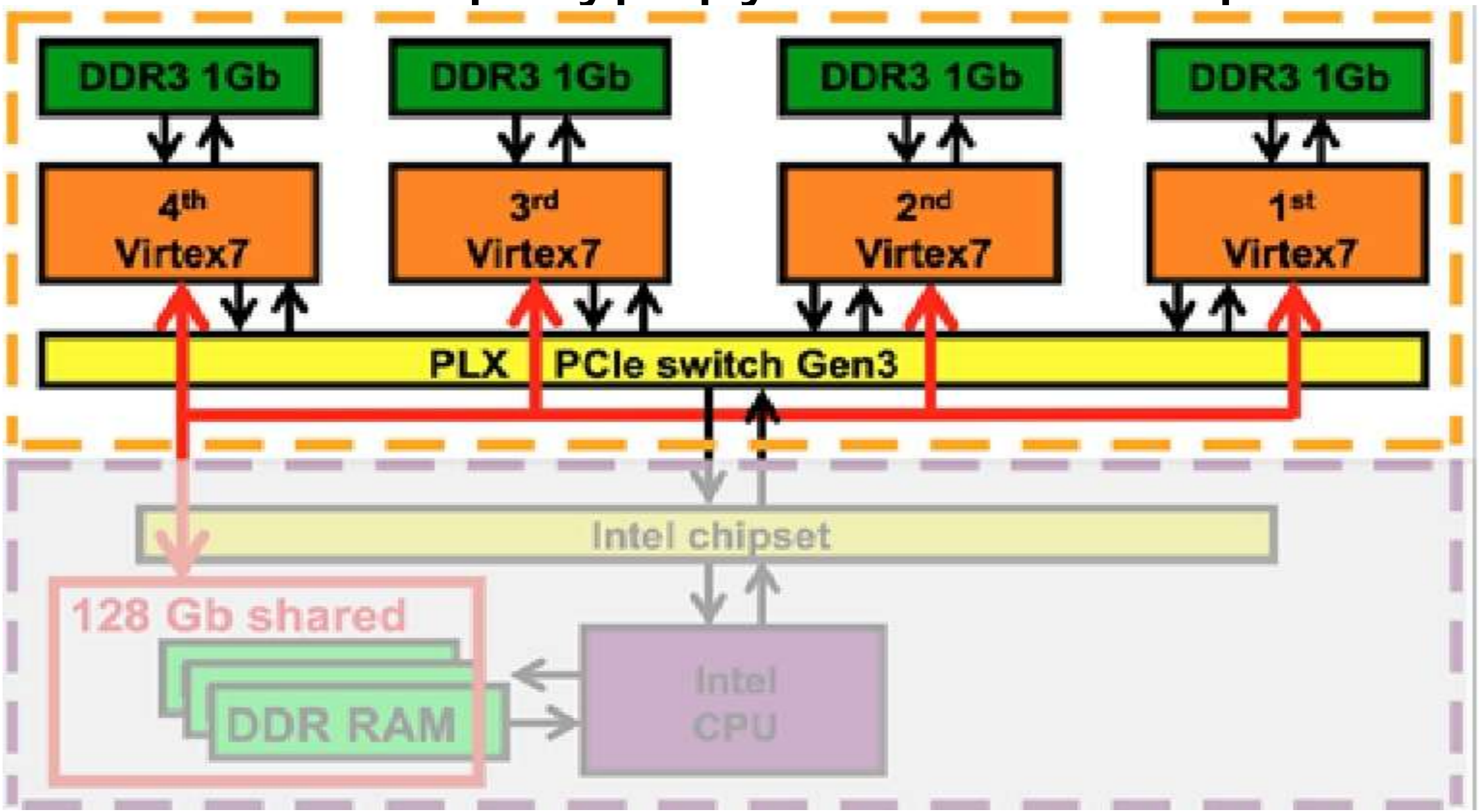
сеть загрузки и управления

сеть мониторинга

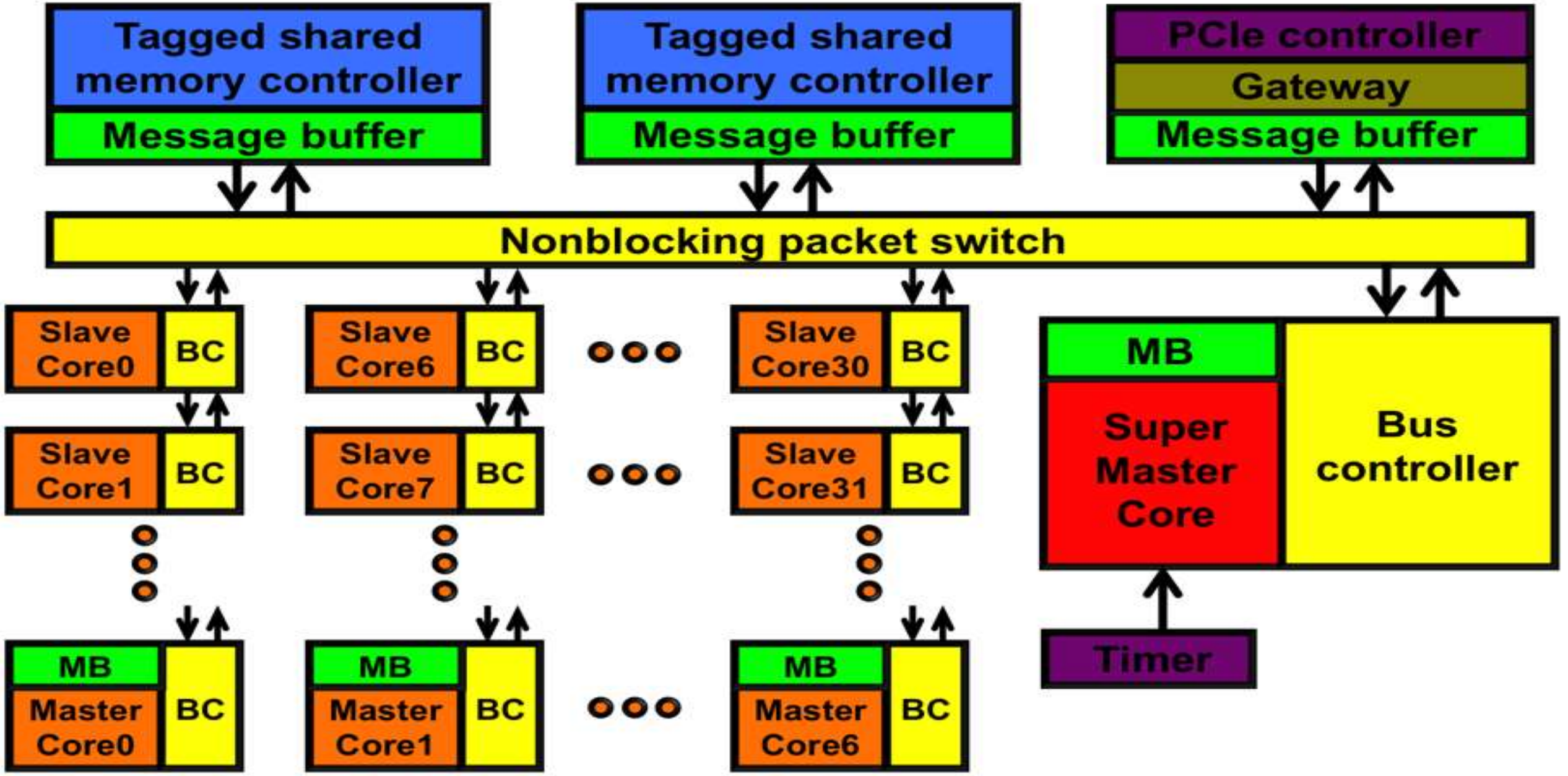


RC-реконфигурируемый кластер

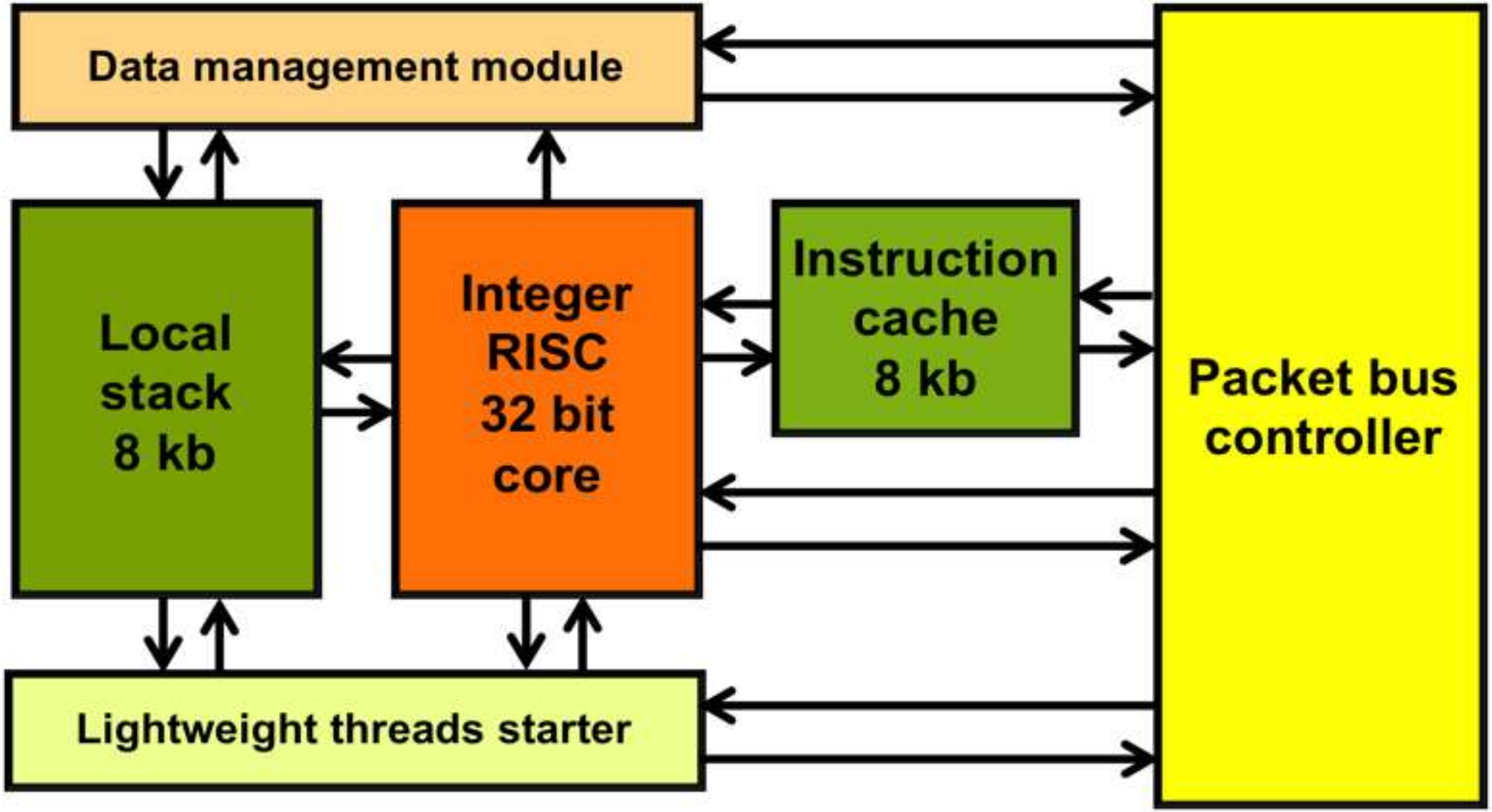
Реконфигурируемый кластер



Структура, формируемая в ПЛИС



Структура вычислительного ядра



Структура, формируемая в ПЛИС

Модель программирования Parallel Random Access Model (PRAM)

- Процессоры Pi, i=1, …, n, имеют доступ к общей памяти (distributed shared memory - DSM) или секционированной общей памяти (partitioned global address space - PGAS), физически распределенной по вычислительным модулям (процессор+блок памяти), но имеющей общее адресное пространство и логически доступной всем процессорам
- Конфликты доступа разных процессоров к одной ячейке разделяемой памяти exclusive-read exclusive-write (EREW) и concurrent-read exclusive-write (CREW).

Язык программирования

- Языком параллельного программирования может служить расширение языка С (С+ 3 дополнительные команды+библиотека синхронизации и коммуникаций):
- spawn (a, n) - породить заданное параметром n количество тредов с последовательными номерами, начиная с номера a,
- joint - завершить тред, исполняющий команду join,
- fetch_and_add(e, x) - выполнить как неделимую атомарную операцию присвоение переменной значения параметра x и увеличить значение x, прибавив к x значение e.

Пример: массив B из ненулевых элементов A

- int x;
- x=0;
- spawn(1, n) {
- int e;
- e=1;
- if (A[$]!=0) {
- a= fetch_and_add(e, x)
- B[a]=A[$];
- }
- }
- $ - номер текущего процесса

Средства межпроцессных коммуникаций и синхронизации

- **добавление к каждому слову памяти full/empty (FE) бита**
- **изменение семантики команд обращения к памяти: аналогично Cray XMT для синхронизации на базе FE битов введены синхронизирующие переменные x$ и аппаратные функции (generic functions) для выполнения операций чтения и записи:**
- **purge x$ - присвоение FEбиту x$ значения empty;**
- **writeqr x$, g–запись в x$ значения g, если значение FEбита x$ равно q или ожидание записи пока значение FEбита не станет q; после записи значение FEбита становится равным r;**
- **readqr x$ - чтение значения x$, если значение FEбита x$ равно q или ожидание чтения пока значение FEбита не станет q; после чтения значение FEбита становится равным r.**

Особенности синхронизирующих переменных

- if ((x$ >= 10)&&(x$ <= 100)){}
- Не то же, что:

- tmpx = x$;
- if ((tmpx >= 10) && (tmpx <= 100)){}

Потоковое вычислениеF(i, j)=0.25(F(i-1, j)+F(i-1, j-1)+F(i, j-1)+F(i, j) в области 0<i<n+1, 0<j<n+1 при заданных значениях F(0, 0), F(0, j), F(i, 0)*

- spawn (1, n) {
- int i;
- i=$;
- spawn (1, n) {
- int j;
- j=$;
- purge (&(F[i, j]);
- }
- }

```
spawn (1, n) {
int i;
i=$;
spawn (1, n) {
int j;
j=$;
double East=readff(&(F[i, j-1]);
double SEast=readff(&(F[i-1, j-1]);
double South=readff(&(F[i-1, j]);
double t= readee(&(F[i, j]);
t=0.25*(t+East+SEast+South);
writeef(&(F[i, j]), t);
}
}
```

Заключение

- **Текущим предметом исследований по МГВС служат реализуемые в архитектуре и программах экзафлопсных компьютеров модели организации вычислений:**
- **- применение функциональных, непроцедурных и потоковых моделей программ для выражения на уровне пользователя параллелизма;**
- **массовая однородная и неоднородная по объему (зернистости) и типу выполняемых операций мультитредовость, ориентированная на приложения специализация;**
- **- выделение в приложениях асинхронных процессов доступа к данным и вычислительных процессов;**
- **- статическое и динамическое распараллеливание приложений;**
- **- иерархическая организация глобально адресуемой памяти, адаптивная локализация данных и вычислений на уровнях иерархии;**
- **- мелкозернистая синхронизация процессов на элементах памяти.**