Операционные системы

Многопроцессорные системы

Типы многопроцессорных систем

- Мультипроцессоры (многопроцессорные, многоядерные)
- Мультикомпьютеры

Мультипроцессоры

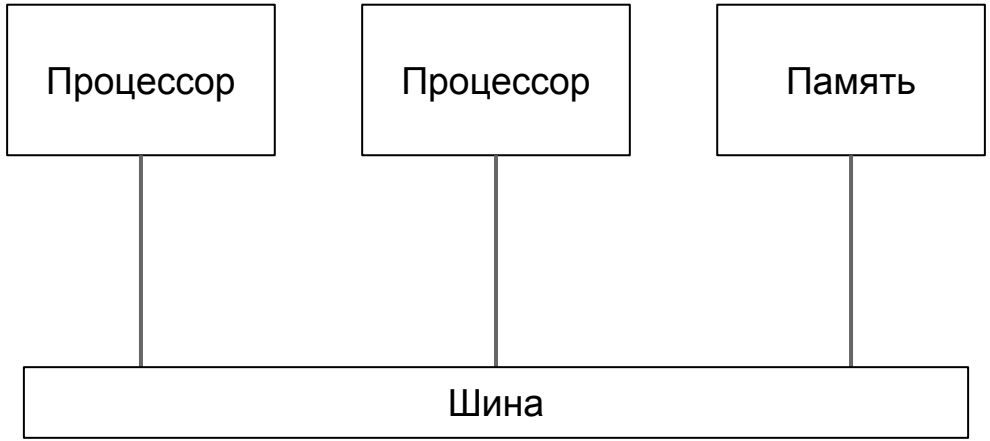
несколько процессоров имеют доступ к общей оперативной памяти.

Примеры: многопроцессорные серверы, многоядерные смартфоны

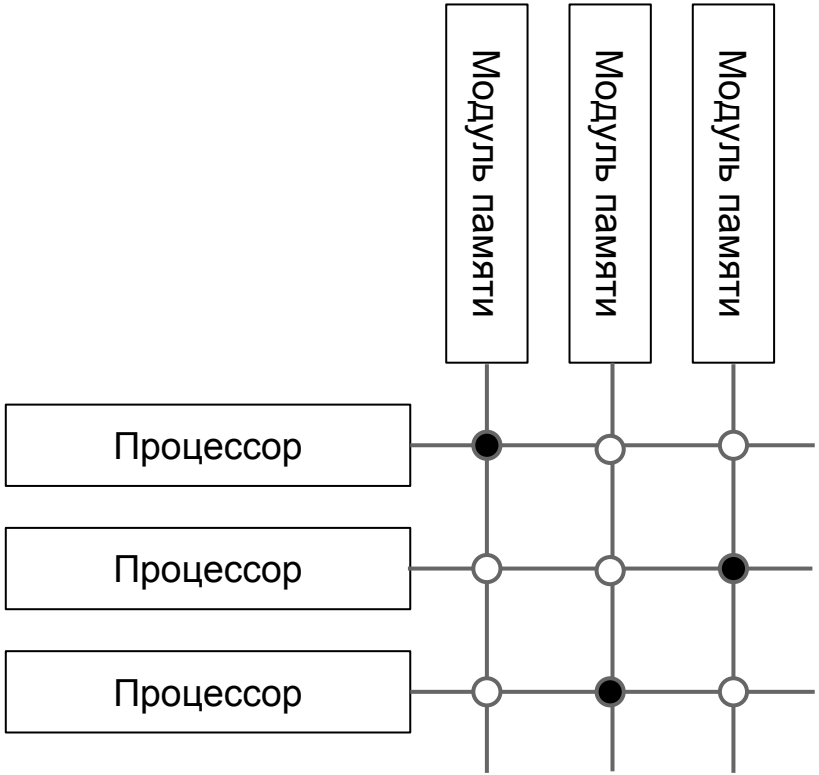
Доступ к памяти

- UMA (Uniform Memory Access) - время доступа не зависит от процессора или области памяти
- NUMA (Nonniform Memory Access) - в противном случае

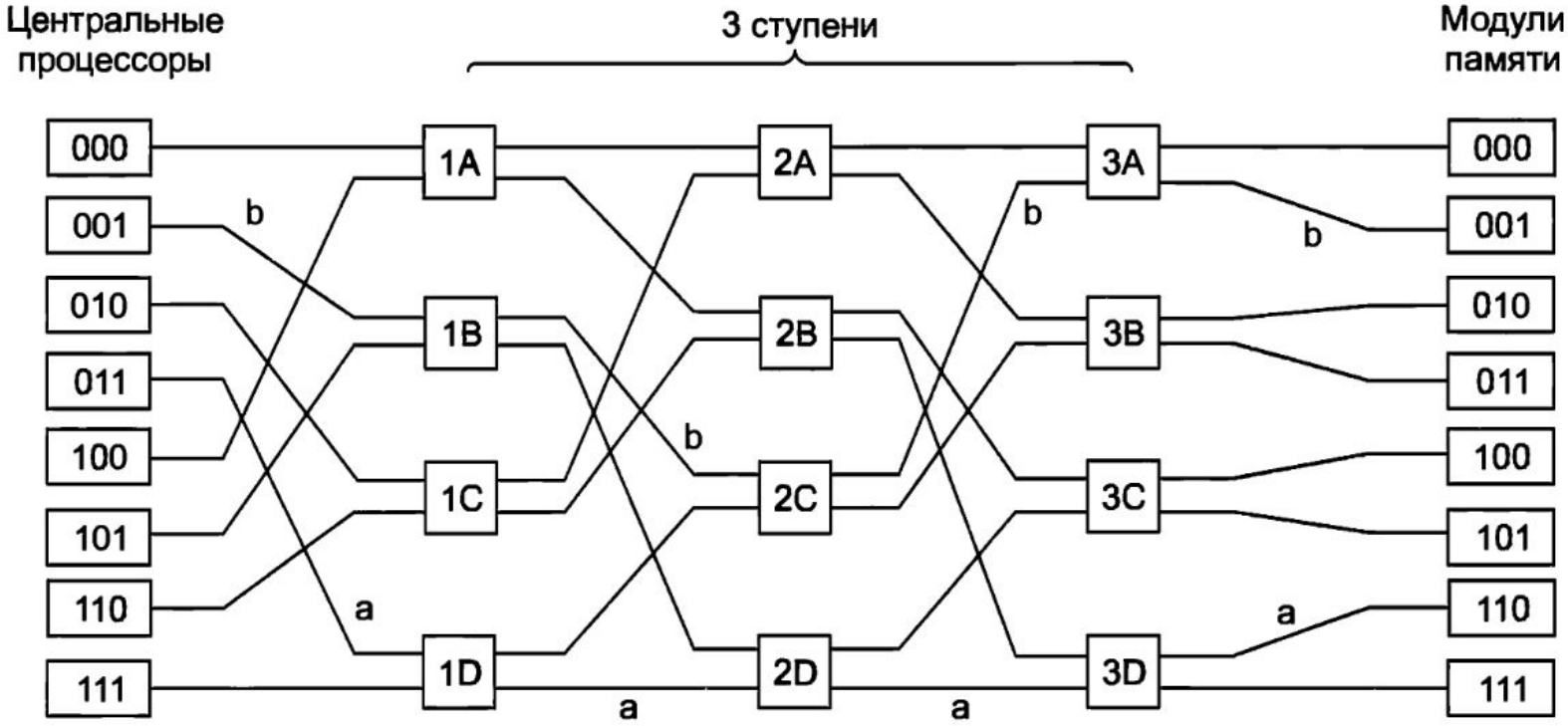
UMA с шинной архитектурой



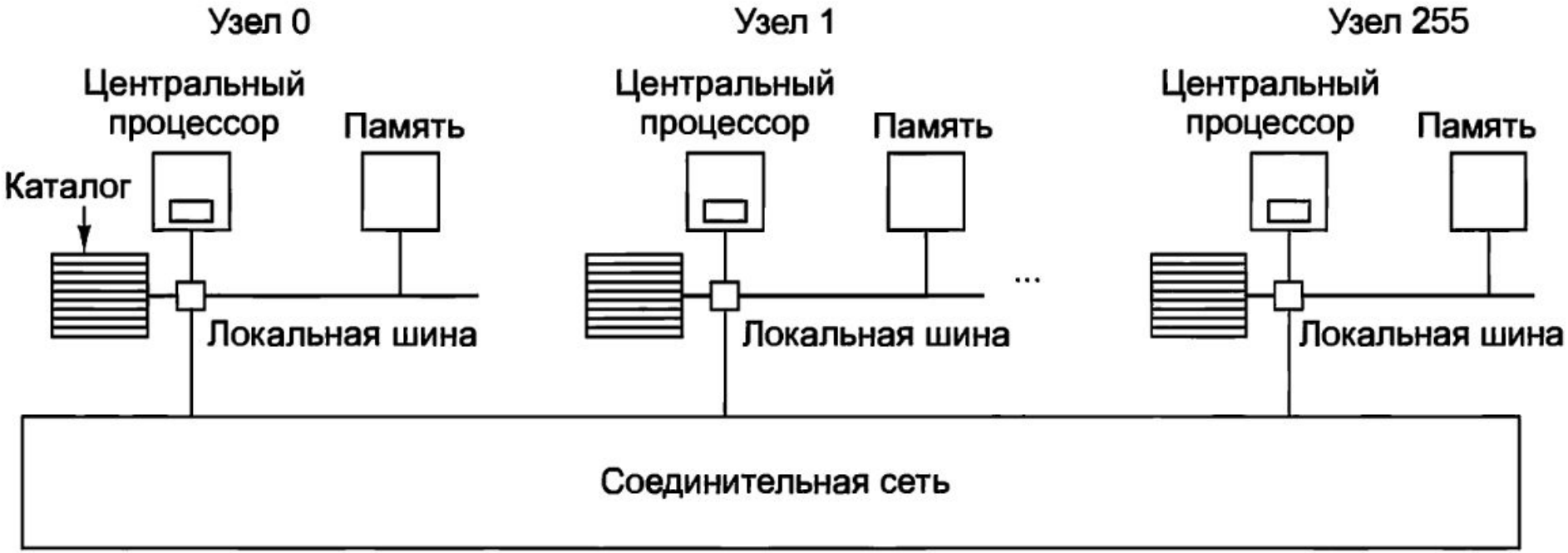
UMA с координатным коммутатором



UMA с многоступенчатой коммутацией



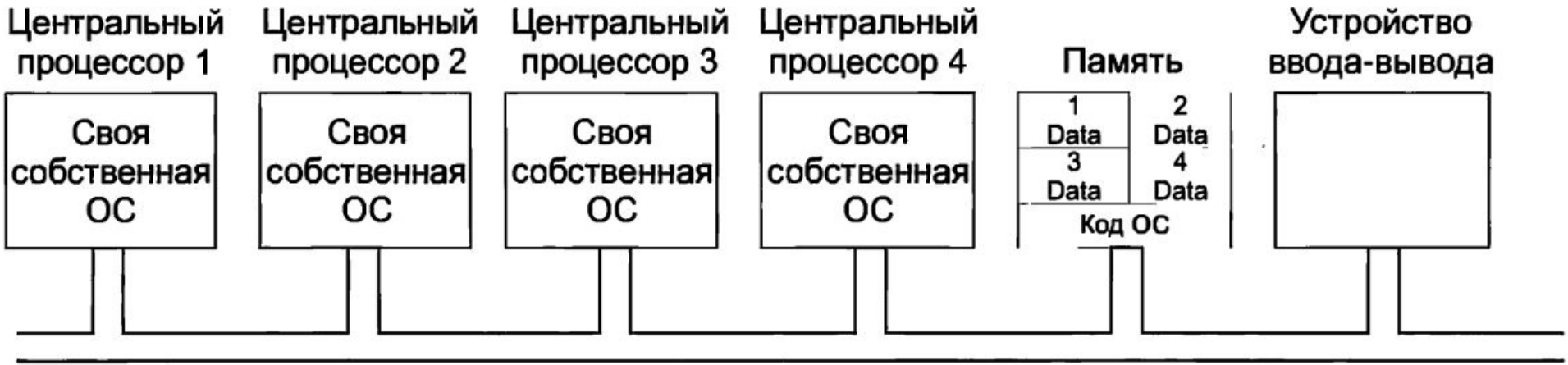
NUMA


Узел 0
Центральный процессор
Память
Каталог
Локальная шина
Узел 1
Центральный процессор
Память
Локальная шина
...
Узел 255
Центральный процессор
Память
Локальная шина
Соединительная сеть

ОС для мультипроцессоров

- Собственная ОС
- Ассиметричные мультипроцессоры
- Симметричные мультипроцессоры

Собственная ОС для каждого процессора



Свойства

Недостатки:

- процессы не мигрируют, неравномерная загрузка
- дублирование программных кешей
- статическое распределение памяти

Достоинства:

- ОС работают независимо, повышается надёжность, безопасность

AMP (assymetric multiprocessing)

Свойства

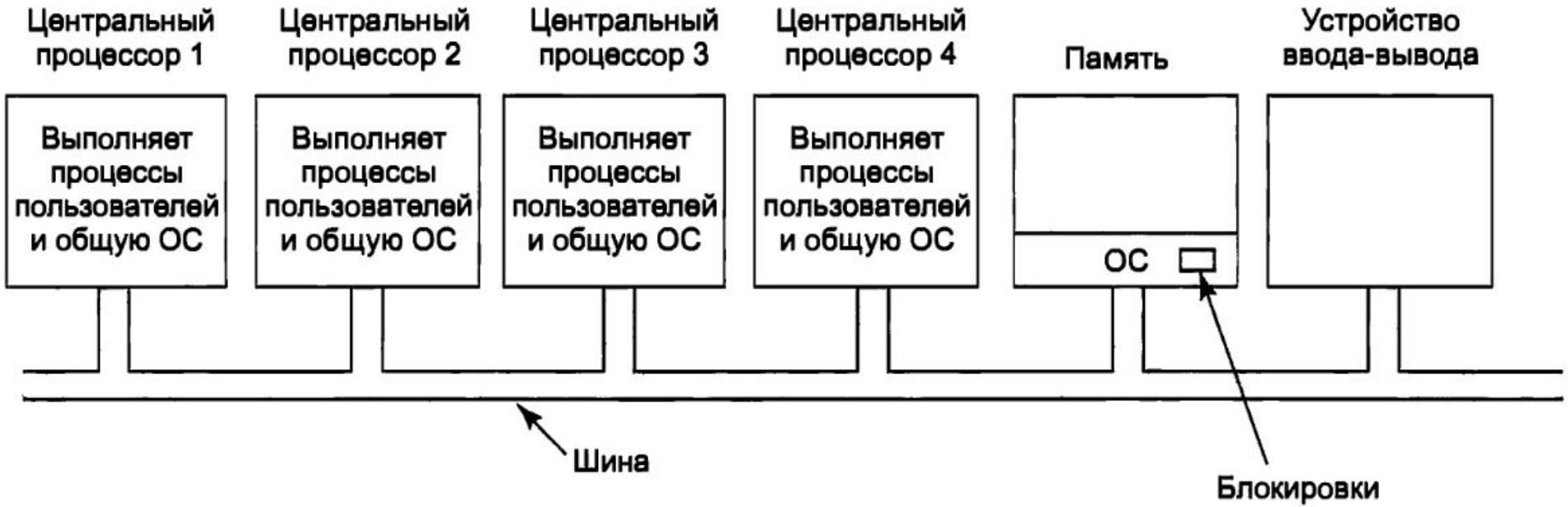
Достоинства:

- Распределение ресурсов
- Возможно использовать процессоры разных архитектур

Недостатки:

- Управляющий процессор - узкое место
- Усложнение ОС: одна версия для управляющего процессора, другая для рабочих процессоров

SMP (Symmetric Multiprocessor)



Планирование мультипроцессоров

- Единая очередь готовых потоков
- + стараться не перемещать потоки на другой процессор

или

- По очереди на процессор
- + перемещать потоки процессор, если тот начинает простаивать

Реентерабельность

свойство частей программ, позволяющее выполнять одну и ту же последовательность инструкций нескольким потокам.

Нереентерабельные части должны быть защищены.

spinlock - мотивация

irq_lock/unlock в мультипроцессорной среде недостаточно для обеспечения атомарности последовательности инструкций.

Нужен специальный примитив, обеспечивающий атомарность в мультипроцессорной среде.

spinlock

Критическая секция на основе цикла и специальных атомарных инструкций

Примерная реализация:

```
spin_lock(spin_t *spin) {
    while (LOCKED == test_and_set(spin));
}
spin_unlock(spin_t *spin) {
    *spin = UNLOCKED;
}
```

test_and_set

```
test_and_set(ptr) {
    bool old = *ptr
    if (!*ptr) {
        *ptr = true;
    }
    return old;
}
```

Другие атомарные операции

- fetch_and_add
- compare_and_swap

Замечания

В такой версии test&set не учитваются кэши.

Такой наивный подход требует постоянного блокирование шины, что может быть дорого (~100 тактов)

Исследования давно сосредоточены на создании lock-free и wait-free алгоритмов

Мультикомпьютеры

системы, с собственным процессором и оперативной памятью, часто без накопительных устройств, но может быть обратное.

Часто построен на базе множества персональных компьютеров, соединенных высокопроизводительной сетью

Способы взаимодействия узлов

- Посылка сообщений
- Удалённый вызов
- Обмен страницами

Посылка сообщений

Вызовы:

- send
- receive

Удалённый вызов

1. Вызов обычной функции,
2. которая отправляет сообщение
3. По приёму сообщения,
4. вызывается требуемая функций

Обмен страницами

Создание видимости обладания общей памятью.

Общая память разбивается на блоки- *страницы,* каждой страницей владеет один узел. Если другой узел требует страницу, она перемещается.

Важно обеспечивать минимальное количество перемещений страниц.

Неизменяемые страницы можно *реплицировать* - содержать на нескольких узлах

Для приложений (unix)

- get/setaffinity
- spinlock, test_and_set, ...