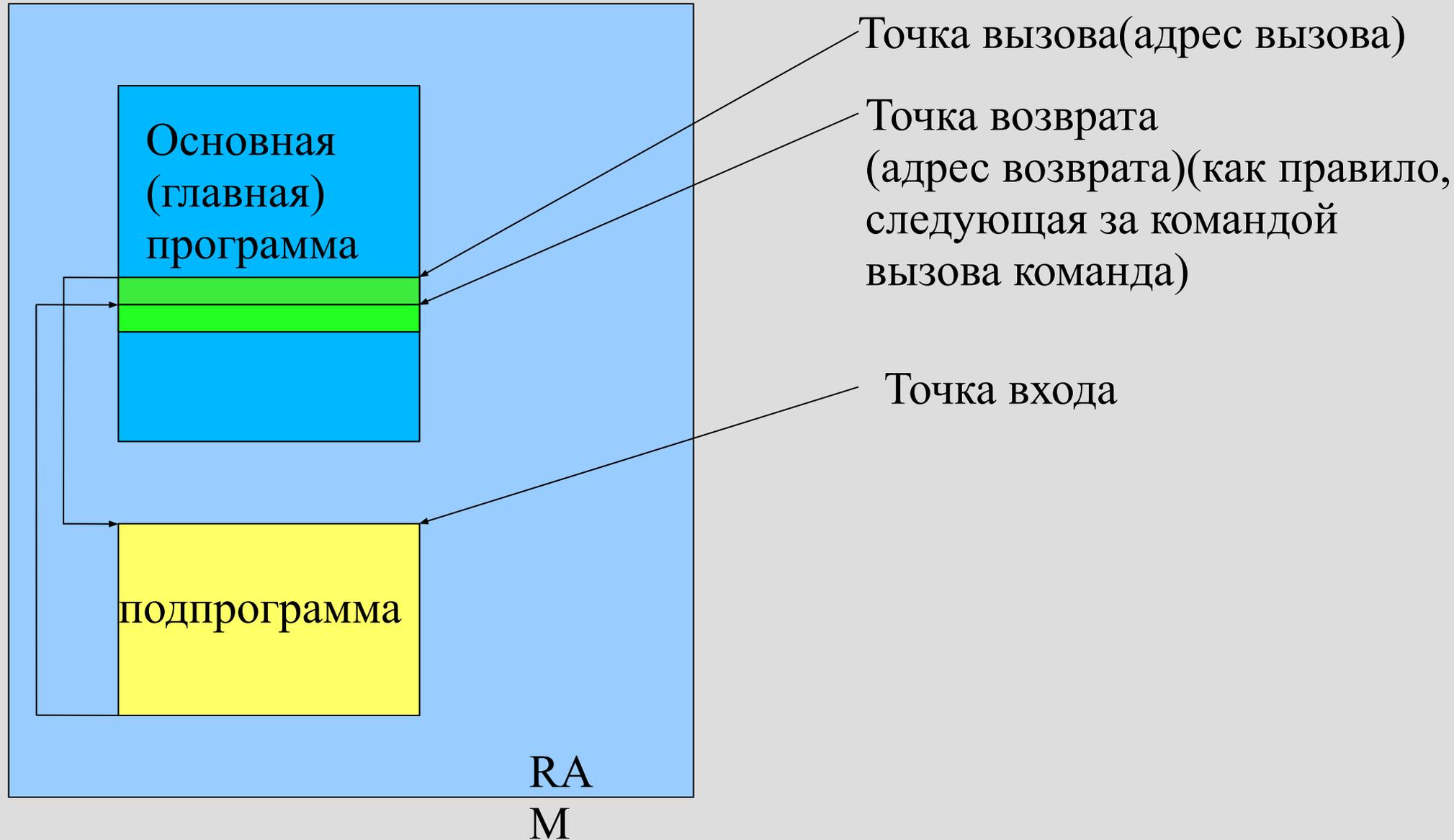


Механизмы организации подпрограмм и прерываний

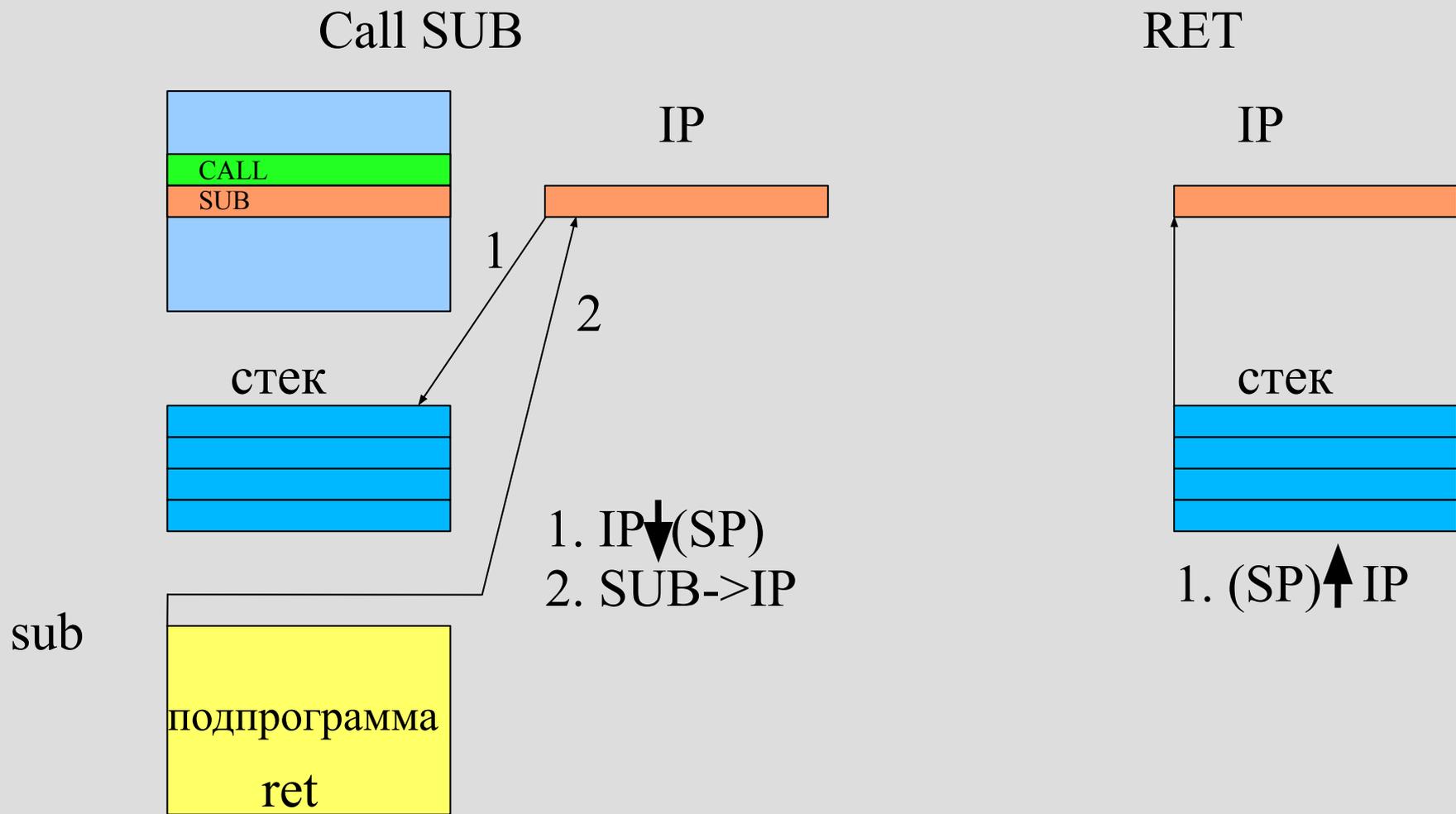
Преимущества использования подпрограмм и модульного подхода

1. Появляется возможность выделить в программе одинаковые фрагменты и оформить их отдельным модулем, используя его каждый раз, когда требуется выполнить соответствующие вычисления. Это позволяет значительно сократить объем программы.
2. В подавляющем большинстве случаев гораздо легче спроектировать и отладить десяток небольших модулей, реализующих достаточно простые функции, а затем связать их между собой с помощью опять же небольшой управляющей программы, чем проектировать и отлаживать большую программу, созданную единым модулем, часто со сложной, сетевой структурой.
3. появляется возможность коллективной разработки крупных проектов программ коллективами программистов, при этом каждый из них выполняет отдельный, функционально автономный и самостоятельный модуль в соответствии с полученными спецификациями.

Вызов подпрограммы

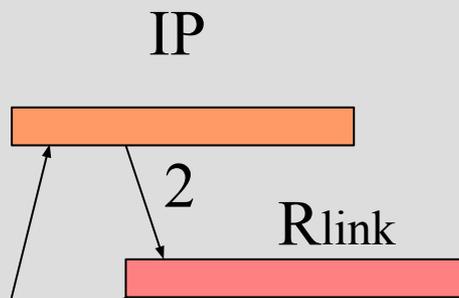
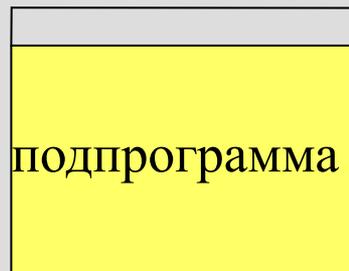
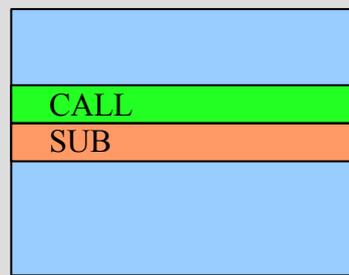


Механизм передачи и возврата управления с использованием стека



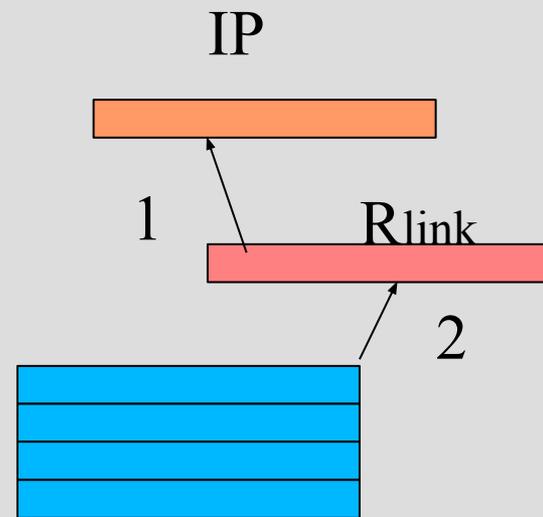
Механизм передачи и возврата управления с использованием стека и регистра связи

Call SUB



1. (Rlink)↓(SP)
2. (IP)-> Rlink
3. SUB->IP

RET



1. Rlink->IP
2. (SP)↑(Rlink)

Требования к подпрограмме при передаче управления

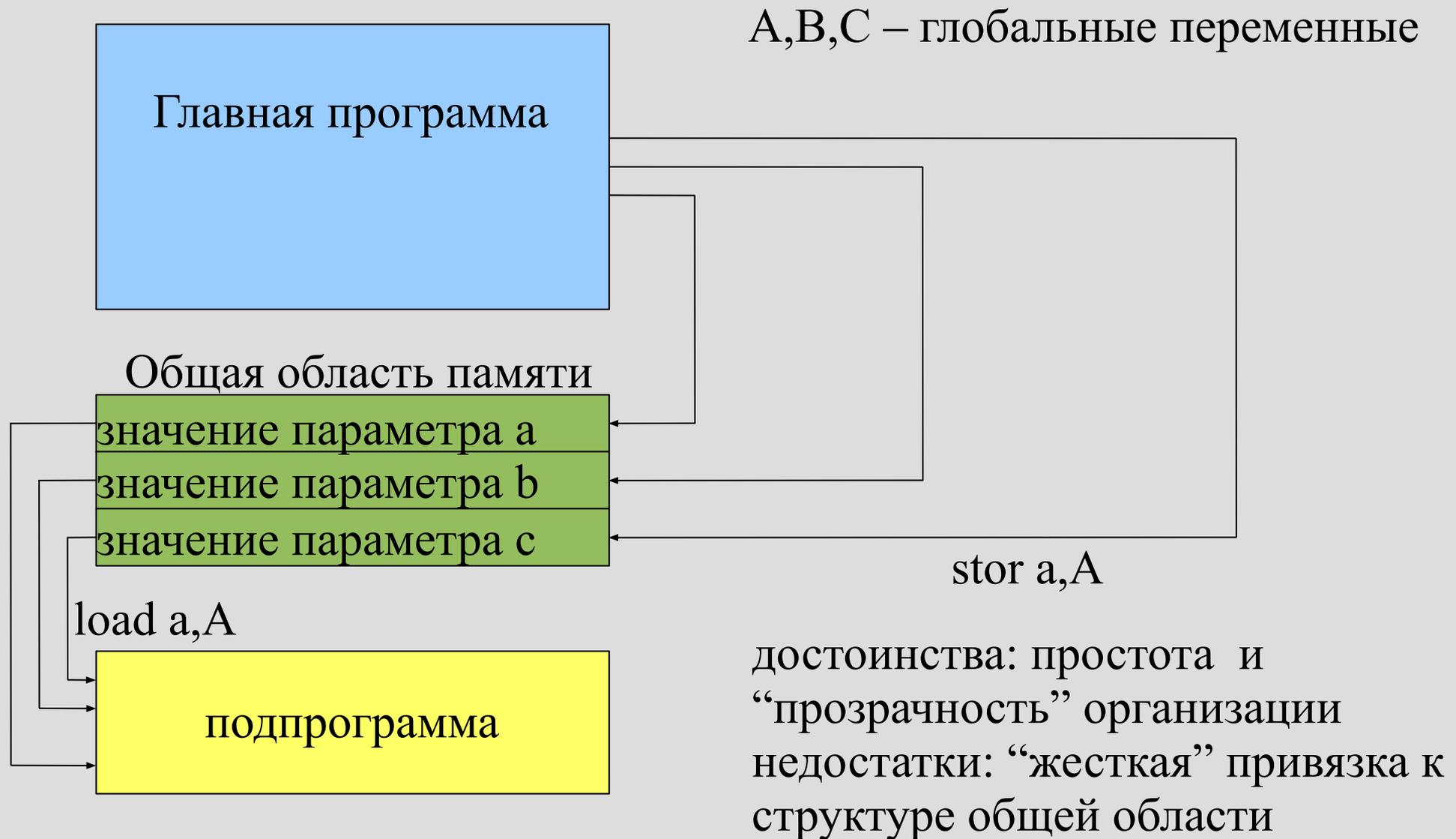
1. Подпрограмма может в ходе своей работы использовать регистры процессора. Для того, чтобы не испортить значения регистров основной программы, подпрограмма должна начинать свою работу с сохранения содержимого используемых ей регистров процессора, а перед возвратом в основную программу обеспечивать восстановление их содержимого. Содержимое регистров может сохраняться либо в стеке, либо в области памяти, доступной подпрограмме.
2. Перед возвратом управления из подпрограммы в основную программу стек должен быть *сбалансирован*, т.е. при выходе из подпрограммы он должен иметь то же состояние, что и при входе в подпрограмму(по крайней мере, его вершина).

Передача параметров подпрограммам

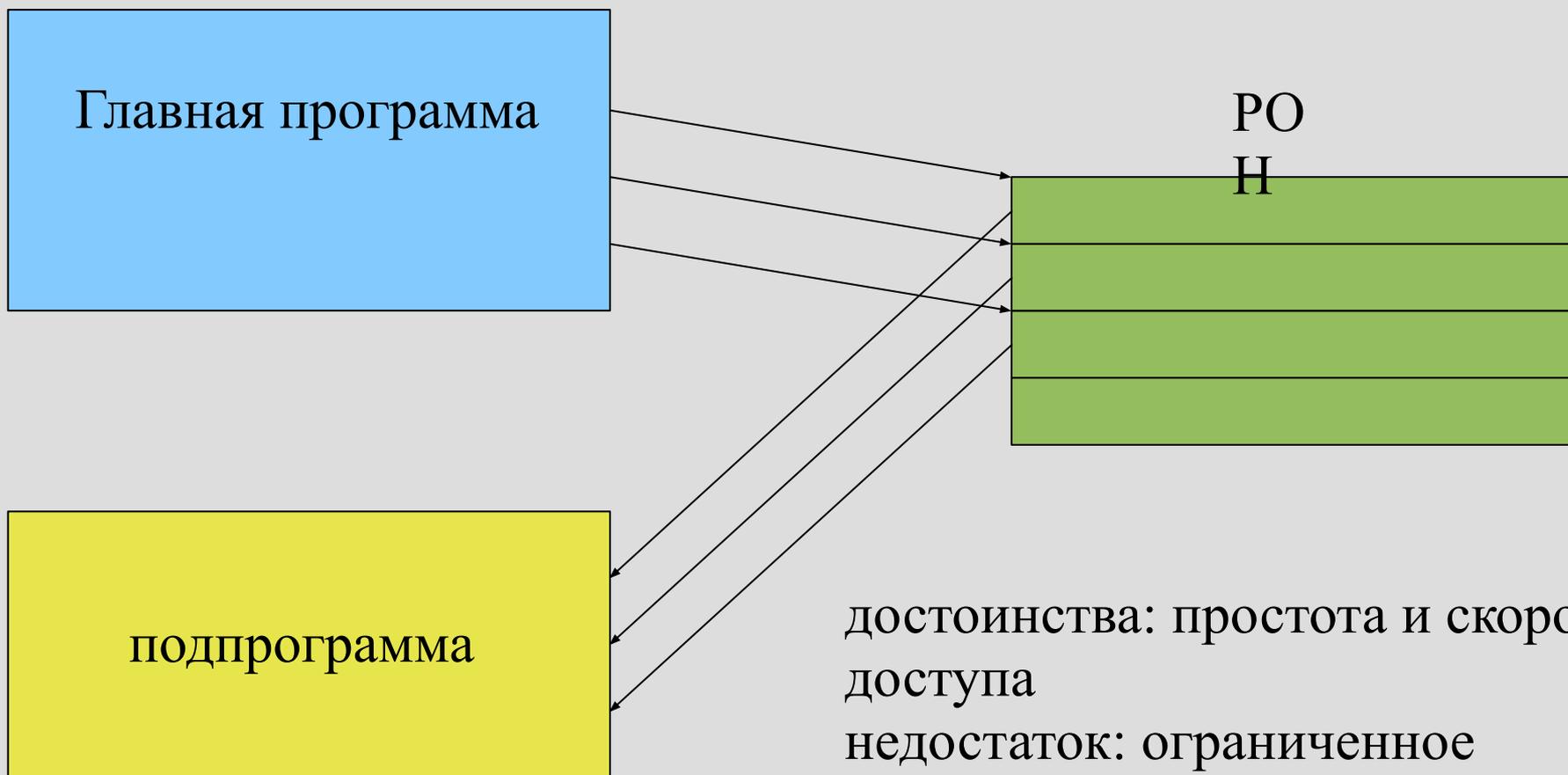
При рассмотрении внутренних механизмов передачи параметров можно выделить несколько способов передачи, а именно:

1. передача параметров с использованием общей области памяти;
2. передача параметров через регистры процессора;
3. передача параметров через стек;
4. передача параметров комбинированными способами, в том числе через использование таблицы адресов параметров.

передача параметров с использованием общей области памяти



Передача параметров через регистры процессора

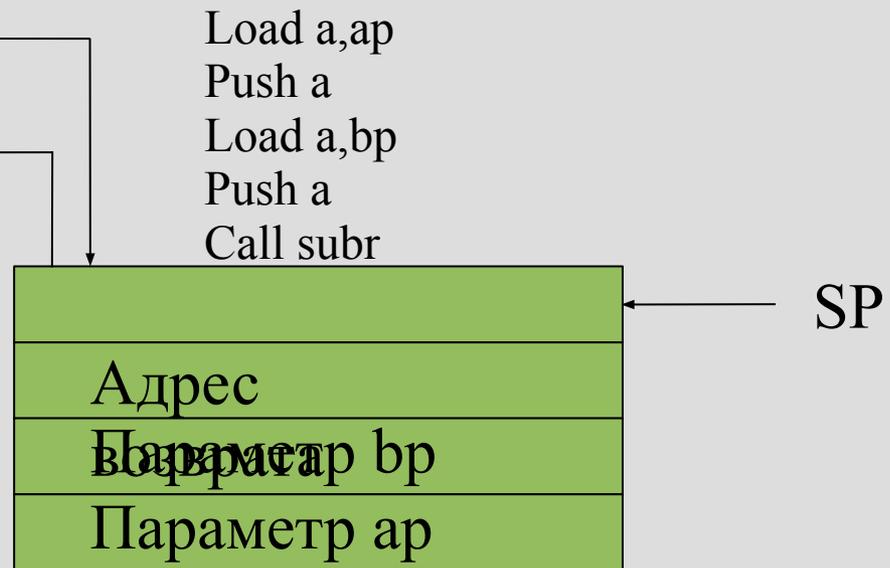


достоинства: простота и скорость доступа
недостаток: ограниченное количество регистров процессора
стандарт де-факто для функций

Передача параметров через стек

Главная программа

подпрограмма



Load a,ap
 Push a
 Load a,bp
 Push a
 Call subr

Pop a
 Stor a,retaddr
 Pop a
 Stor a,p1
 Pop a
 Stor a,p2

 Load a,retaddr
 Push a
 ret

Load a,2(sp)

 Load a,3(sp)

ИЛИ

Стандартный метод
 Передачи параметров

Комбинированные методы (таблица адресов параметров)

Главная программа

Доступ в п/п
Load a,1(a) ; в Ra адрес пар 2
Load a,0(a) ; в Ra-значение пар2

RO
N
Адрес таблицы адресов

Доступ в п/п
Load a,2(sp) ; в Ra адрес табл.
Или стек
Load a,1(a) ; в Ra-адр.пар.2
Load a,0(a) ; в Ra-пар-р 2

Таблица адресов

Адрес параметра 1

Адрес параметра 2

Адрес параметра 3

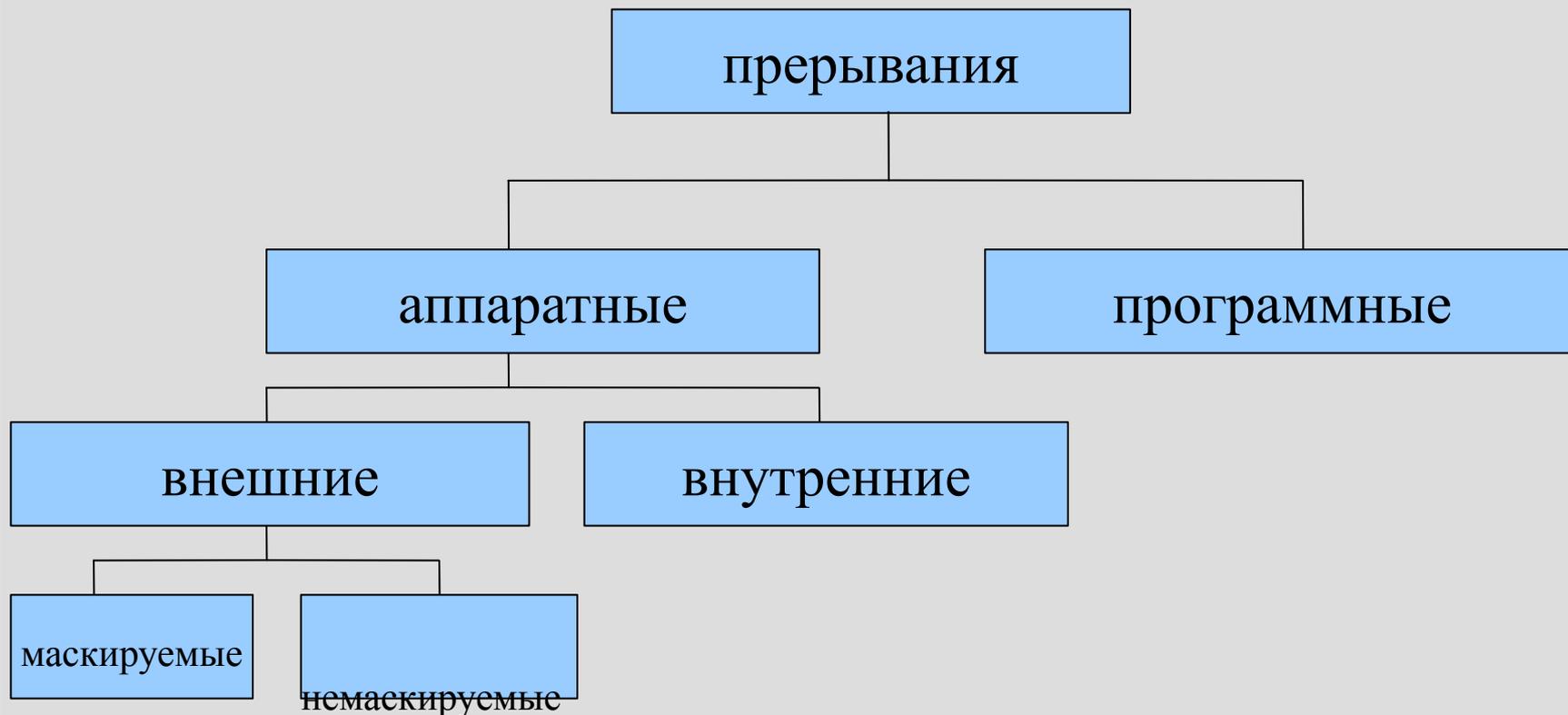
Адрес
Адрес таблицы адресов

подпрограмма

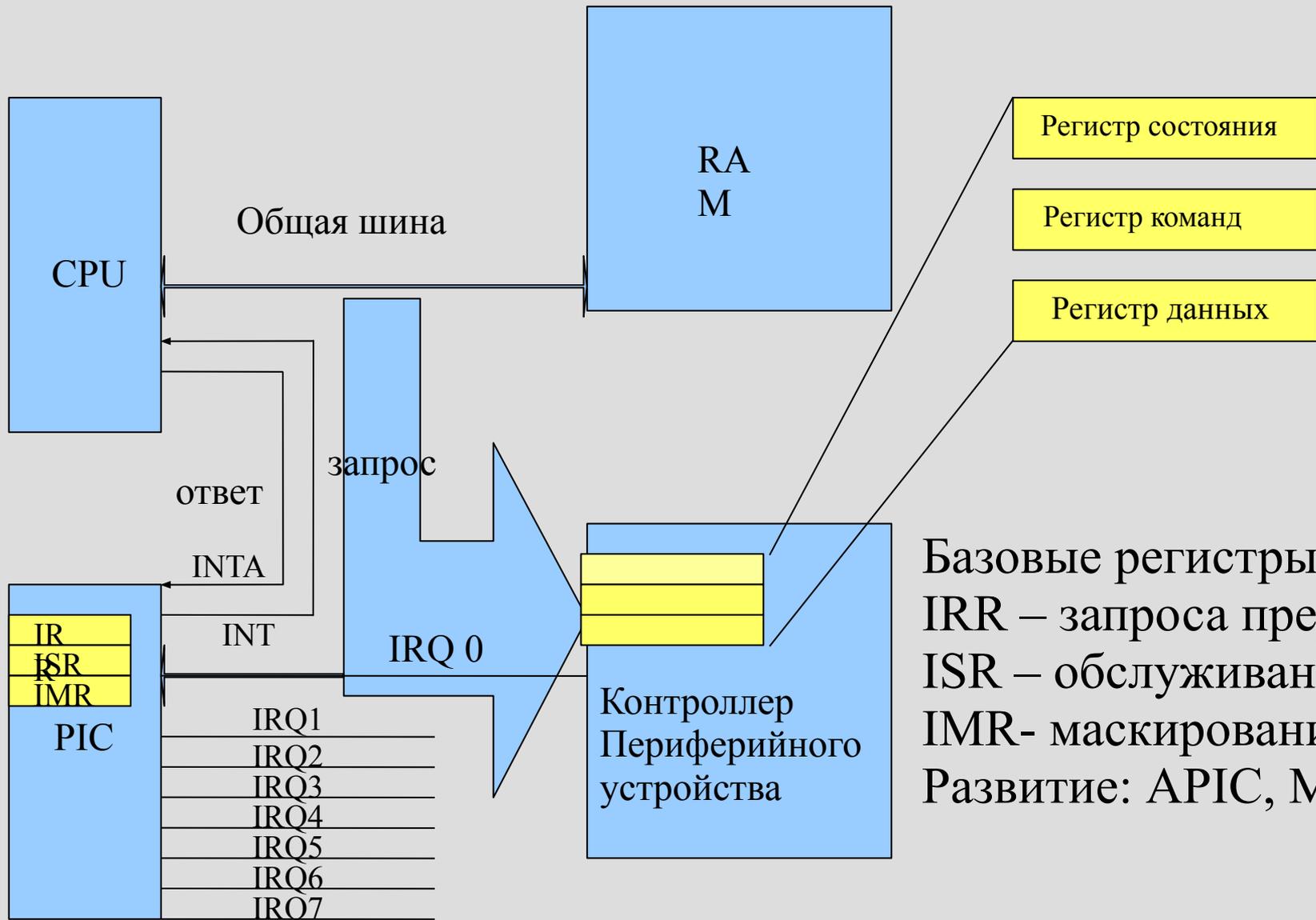
достоинства: произвольное число параметров, переменное число параметров
недостаток: косвенная адресация – долго работает

Прерывания

Прерывание - это временное прекращение выполнения процессором последовательности команд одной программы с целью выполнения другой, имеющей в данный момент времени более высокий приоритет.

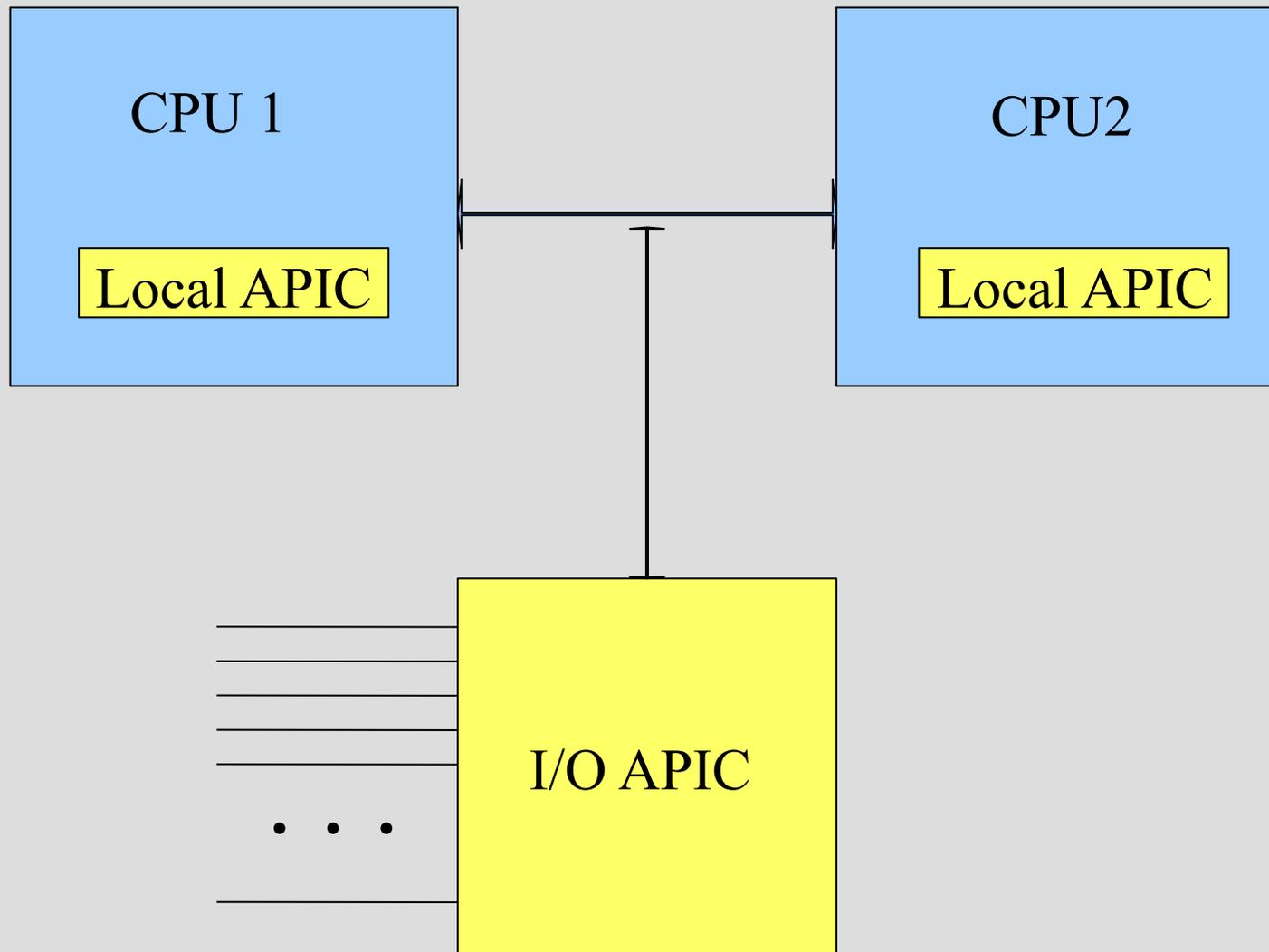


Внешние прерывания для организации ввода/вывода

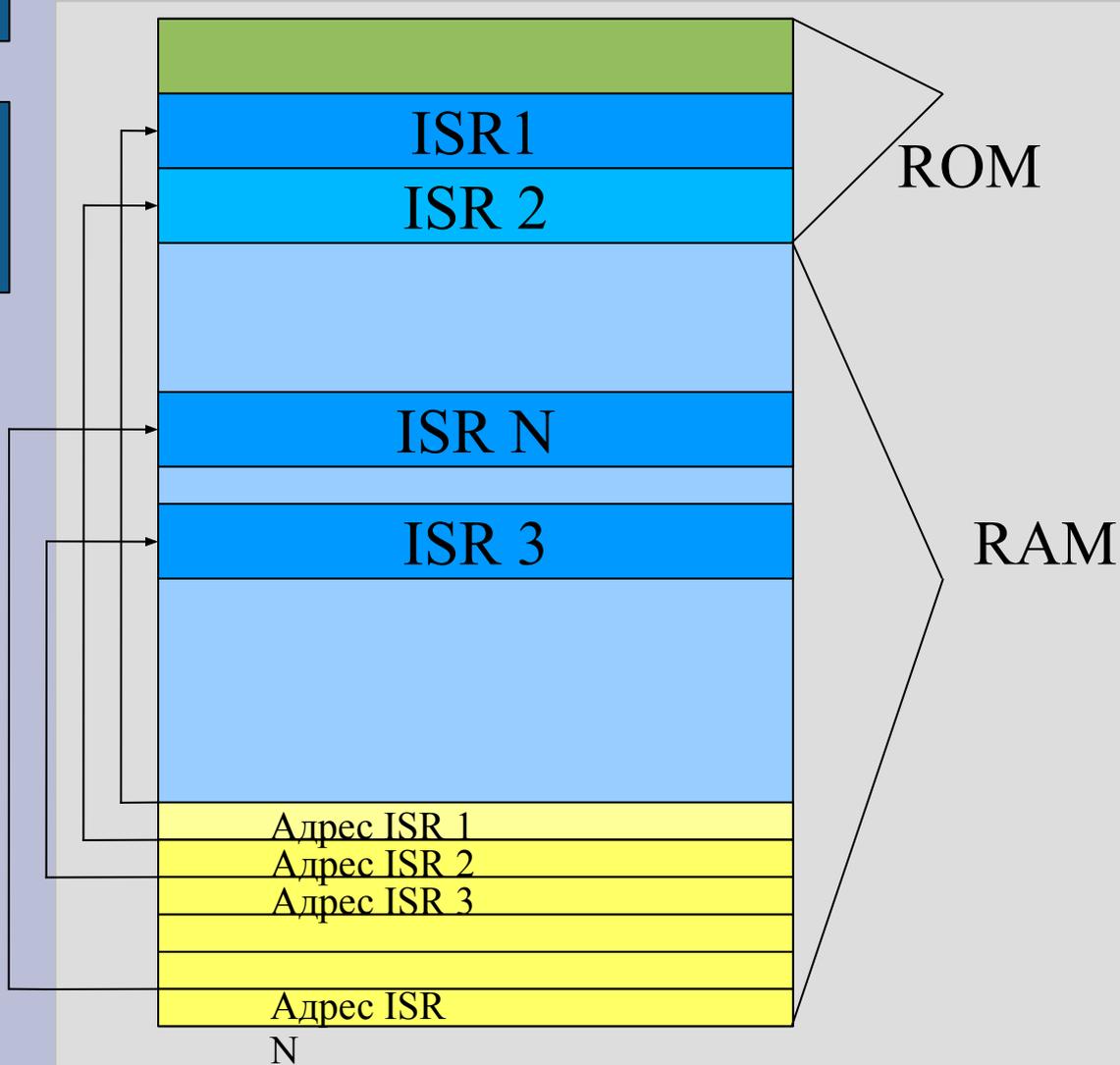


Базовые регистры PIC:
IRR – запроса прерывания
ISR – обслуживания прер.
IMR- маскирования прер.
Развитие: APIC, MSI

Структура подсистемы прерываний в SMP-системах



Обработка прерывания процессором



1. $PC \downarrow (SP)$, $F \downarrow (SP)$
2. по коду прерывания вычисляется вектор прерывания, содержащий адрес ISR (interrupt service routine)
3. Адрес ISR \rightarrow PC :
Начинает работать ISR
Перед завершением обработки прерывания при необходимости “сбрасывается” бит обработки в PIC
Выход из обработки – IRET :
 $(SP) \uparrow F$, $(SP) \uparrow PC$

Немаскируемые прерывания (NMI)

- Аппаратные прерывания, немаскируемые обычным способом, через регистр маски контроллера прерываний.
- Обработываются всегда, вне зависимости от состояния запрета обработки прерываний.
- Служат для сигнализации неисправимых аппаратных сбоев или для целей отладки

Внутренние аппаратные прерывания

- Сигнализируют о нарушении нормального хода вычислительного процесса;
- Немаскируются;
- Примеры:
 - деление на 0;
 - переполнение стека;
 - нарушение защиты памяти;
 - недопустимая инструкция;
 - отсутствие страницы в памяти;

Программные прерывания

- Являются синхронными и вызываются специальной инструкцией в процессе выполнения программы (INT N);
- Используются для обращения к функциям встроенного ПО (Firmware, например BIOS) и функциям операционной системы;
- Обеспечивают независимость программ от версии Firmware и версии операционной системы