Средства вызова подпрограмм и задач

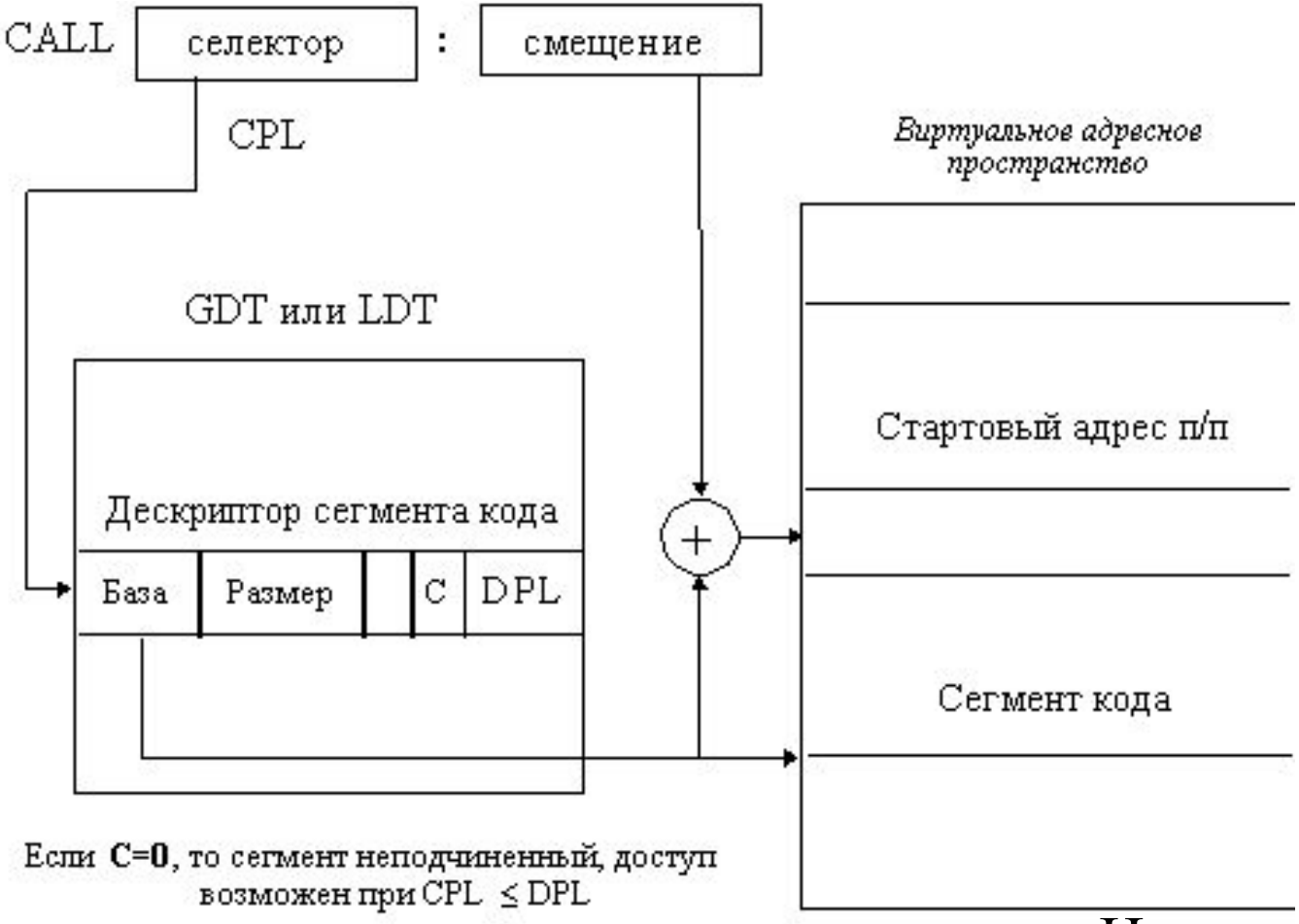
Операционная система, как однозадачная, так и многозадачная, должна предоставлять задачам средства

вызова подпрограмм операционной системы, библиотечных

подпрограмм, а также иметь средства для запуска задач. Вызов подпрограммы отличается от запуска задачи тем, что в первом случае адресное пространство остается тем же (таблица LDT остается прежней), а при вызове задачи адресное пространство полностью меняется

Вызов подпрограмм без смены кодового сегмента в защищенном режиме процессора i386 производится аналогично вызову в реальном режиме с помощью команд JMP и CALL. Для вызова подпрограммы, код которой находится в другом сегменте (который может принадлежать библиотеке, другой задаче или операционной системе), процессор i386 предоставляет 2 варианта вызова, причем оба используют защиту с помощью прав доступа.

Первый способ состоит в непосредственном указании в поле команды JMP или CALL селектора сегмента, содержащего код вызываемой подпрограммы, а также смещение в этом сегменте адреса начала подпрограммы

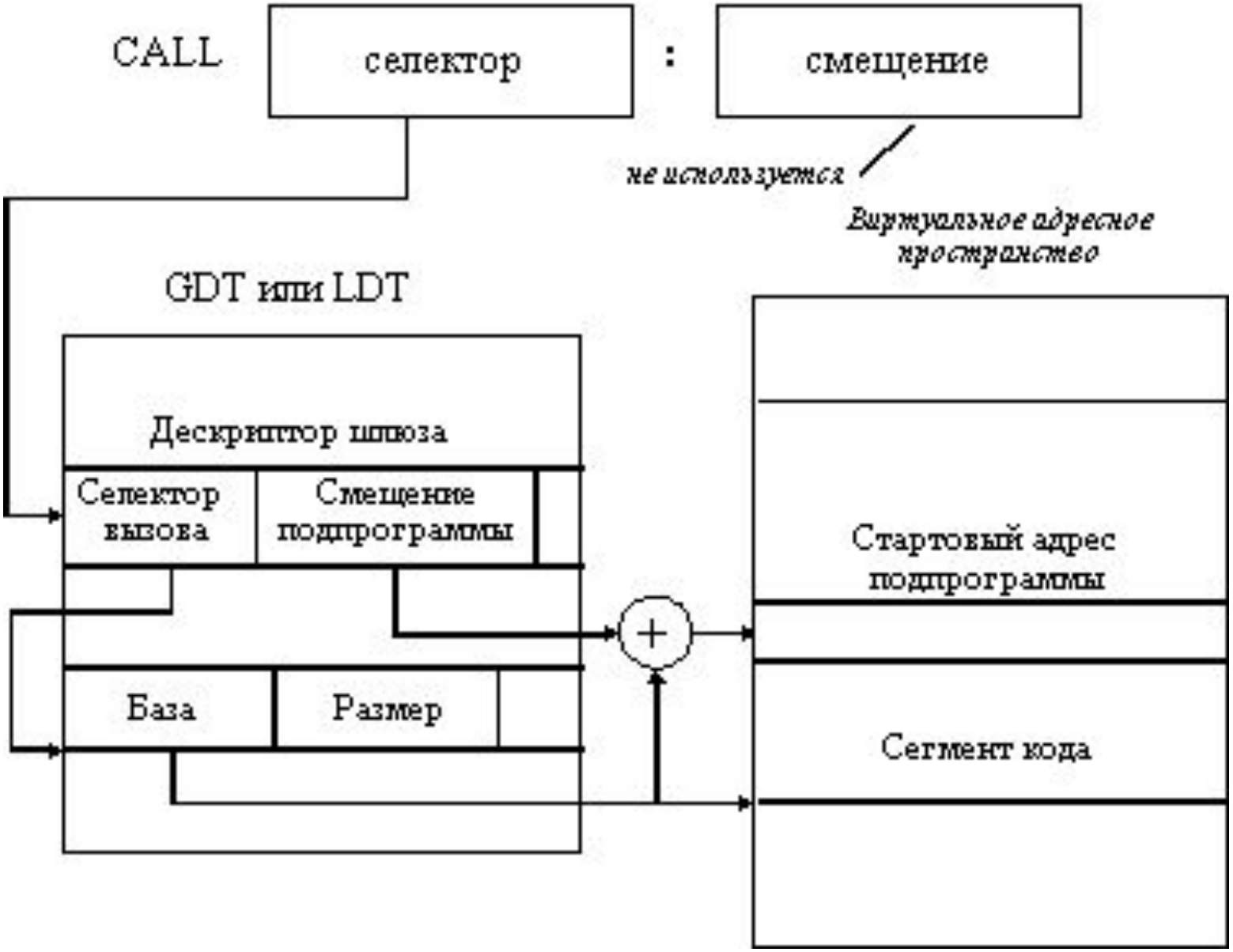


Непосредственный вызов подпрограммы

Разрешение вызова происходит в зависимости от значения поля C в дескрипторе сегмента вызываемого кода. При C=0 вызываемый сегмент не считается подчиненным, и вызов разрешается, только если уровень прав вызывающего кода не меньше уровня прав вызываемого сегмента. При C=1 вызываемый сегмент считается подчиненным и допускает вызов из кода с любым уровнем прав доступа, но при выполнении подпрограмма наделяется уровнем прав вызвавшего кода.

Селектор **0÷15**	Смещение**0÷15**					
Смещение**16÷32**	P	DPL	S=**0**	C (386) или 4 (286)		**0÷4** Счетчик слов (WC)

Формат дескриптора шлюза вызова подпрограммы



Вызов подпрограммы через шлюз вызова

Если вызывается подпрограмма, имеющая другой уровень привилегий, то из текущего стека в стек уровня доступа вызываемого сегмента копируется столько 32-разрядных слов, сколько указано в поле счетчика слов дескриптора шлюза.

Очевидно, что первый способ непригоден для вызова функций операционной системы, имеющей обычно нулевой уровень прав, из пользовательской программы, работающей, как правило, на третьем уровне. Поэтому процессор i386 предоставляет способ вызова подпрограмм, основанный на том, что заранее определяется набор точек входа в привилегированные кодовые сегменты, и эти точки входа описываются с помощью специальных дескрипторов - дескрипторов шлюзов вызова подпрограмм. Этот дескриптор принадлежит к системным дескрипторам, и его структура отличается от структуры дескрипторов сегментов кода и данных . При вызове кодов, обладающих различны-ми уровнями привилегий, возникает проблема передачи параметров между различными стеками, так как для надежной защиты задачи различного уровня привилегий имеют различные сегменты стеков. Селекторы этих сегментов хранятся в контексте задачи - сегменте TSS (Task State Segment).

Как и в случае вызова подпрограмм, имеется две возможности вызова задачи - непосредственный вызов через указание селектора сегмента TSS нужной задачи в поле команды CALL и косвенный вызов через шлюз вызова задачи. Как и при вызове подпрограмм, непосредственный вызов возможен только в случае, если вызывающий код обладает уровнем привилегий, не меньшим, чем вызываемая задача. При вызове через шлюз (который может располагаться и в таблице LDT) достаточно иметь права доступа к шлюзу.

При переключении задач процессор выполняет следующие действия:

1) Выполняется команда CALL, селектор которой указывает на дескриптор сегмента типа TSS.

2) В TSS текущей задачи сохраняются значения регистров процессора. На текущий сегмент TSS указывает регистр процессора TR, содержащий селектор сегмента.

- 3) В TR загружается селектор сегмента TSS задачи, на которую переключается процессор 4) Из нового TSS в регистр LDTR переносится значение селектора таблицы LDT в таблице GDT задачи.
- 5) Восстанавливаются значения регистров процессора (из соответствующих полей нового сегмента TSS).
- 6) В поле селектора возврата заносится селектор сегмента TSS снимаемой с выполнения задачи для организации возврата к прерванной задаче в будущем.
- Вызов задачи через шлюз происходит аналогично, добавляется только этап поиска дескриптора сегмента TSS по значению селектора дескриптора шлюза вызова.