ОРГАНИЗАЦИЯ СЕГМЕНТНО-СТРАНИЧНОЙ ОРГАНИЗАЦИИ ПАМЯТИ

СЕГМЕНТ

На самом деле здесь нужен способ избавить программиста от необходимости усмирения увеличивающихся и уменьшающихся в размерах таблиц аналогично тому, как вирту- альная память устраняет беспокойство по поводу организации программы в оверлеи.

Простым и универсальным решением является предоставление машины с большим количеством совершенно независимых адресных пространств, называемых **сегментами**. Каждый сегмент состоит из линейной последовательности адресов от 0 до некоторого максимума. Длина каждого сегмента может иметь любое значение от 0 до максимально разрешенного. Различные сегменты могут быть разной длины, как это обычно и случается. Кроме того, длина сегмента может изменяться в процессе выполнения про- граммы. Длина сегмента стека может увеличиваться при поступлении в него данных и уменьшаться при их извлечении из него.

ОРГАНИЗАЦИЯ СЕГМЕНТНО-СТРАНИЧНОЙ ОРГАНИЗАЦИИ ПАМЯТИ MULTICS

При большом размере сегментов может стать неудобно или даже невозможно хранить их целиком в оперативной памяти. Это наталкивает на идею применения к ним страничной организации, чтобы иметь дело только с теми страницами сегмента, которые нужны в данный момент. Поддержка страничных сегментов реализована в нескольких важных для нас системах

Каждая программа в системе MULTICS использовала таблицу сегментов, в которой имелось по одному дескриптору на каждый сегмент.

Адрес в системе MULTICS состоял из двух частей: сегмента и адреса внутри сегмен- та. Последний, в свою очередь, делился на номер страницы и слово внутри страницы (рис. 3.32).

Когда происходило обращение к памяти, выполнялся следующий алгоритм:

1. Номер сегмента использовался для нахождения дескриптора сегмента.

2. Проверялось, находится ли таблица страниц сегмента в памяти. Если таблица страниц присутствовала в памяти, определялось ее местоположение. Если таблица в памяти отсутствовала, возникала ошибка отсутствия сегмента. При нарушении защиты возникала ошибка (происходило системное прерывание).

1. Изучалась запись в таблице страниц для запрашиваемой виртуальной страницы. Если страница не находилась в памяти, возникала ошибка отсутствия страницы. Если она была в памяти, из записи таблицы страниц извлекался адрес начала страницы в оперативной памяти.
2. К адресу начала страницы прибавлялось смещение, что давало в результате адрес в оперативной памяти, по которому располагалось нужное слово.
3. И наконец, осуществлялось чтение или сохранение данных.

**Сегментация со страничной организацией памяти: система Intel x86**