EESTI NSV TEADUSTE AKADEEMIA TOIMETISED. 31. KÖIDE
FÜÜSIKA * MATEMAATIKA. 1982, NR. 1

ИЗВЕСТИЯ АКАДЕМИИ НАУК ЭСТОНСКОЙ ССР. ТОМ 31
ФИЗИКА * МАТЕМАТИКА. 1982, № 1

https://doi.org/10.3176/phys.math.1982.1.02

УДК 681.3.016

И. ПАЛЛ, В. ФОМИЧЕВ

ОРГАНИЗАЦИЯ ВНУТРЕННЕЙ ПАМЯТИ СПЕЦИАЛИЗИРОВАННОГО ПРОЦЕССОРА ДЛЯ ОБРАБОТКИ ДАННЫХ СТРУКТУРИРОВАННОГО ТИПА

(Представил Э. Тыугу)

Расширение области применения вычислительных средств для решения научно-технических задач происходит за счет усложнения как алгоритмов, так и структур обрабатываемых данных. В настоящее время обработка структурированных данных выполняется либо с помощью подпрограмм, включаемых в основную программу на этапе трансляции [1], либо с использованием ассоциативной памяти [2, 3]. Однако высокая стоимость хранения единицы данных в ассоциативной памяти, необходимость дополнительной памяти для хранения индексов, а также трудности, возникающие при работе с динамическими упорядоченными структурами, побуждают искать новые подходы к проблеме. В настоящей работе рассматривается возможность применения специализированного, ориентированного на обработку структурированных данных процессора (ПСД), обладающего собственной оперативной памятью. Включение такого процессора в вычислительную систему может быть выполнено с помощью канального процессора, общей шины или путем непосредственной связи с центральным процессором (ЦП). Независимо от способа связи, работа ПСД заключается в выполнении команд, передаваемых ЦП, которые должны идентифицировать компоненты данных и вид требуемой операции. Закончив выполнение очередной команды, ПСД обязан передать ЦП сообщение о результатах выполнения.

Обработка структурированных данных с использованием ПСД позволяет: 1) увеличить степень параллелизма за счет совмещения во времени работы ЦП и ПСД, 2) уменьшить объем проблемных программ и трансляторов за счет использования в них команд ПСД, 3) уменьшить время ЦП на решение задачи динамического распределения. Все это повышает эффективность работы вычислительной системы.

Набор команд и типы структурированных данных, представляемых в ПСД, определяются входным языком системы программирования.

Допустим, что в языке есть четыре типа структурированных данных: массив, список, каталог, дерево; каждый тип, кроме массива, может иметь не только статический, но и динамический вид. Назовем совокупность компонент данных конкретного структурного типа блоком данных. Условимся также, что для блоков статического вида определены две операции: чтение и запись компонентов блока, а для блоков динамического вида — четыре операции: чтение, запись, добавление и исключение компонентов блока.

Использование ПСД предполагает, что для каждого блока данных транслятор входного языка строит дескриптор вида

$$D_{\text{Б}} = \{I_{\text{Б}}, T, V, A_{\text{Б}}, Z_1, Z_2, \ldots, Z_k\},$$

где $I_Б$ — имя блока данных, T — указатель типа, V — указатель вида блока, $A_Б$ — адрес первого элемента блока данных, Z_1, Z_2, ... Z_k — характеристики блока данных, число которых зависит от типа блока. Дескрипторы данных пользователя пересылаются в память ПСД в случае активизации программы и используются ПСД при выполнении команд, передаваемых ему ЦП. Следовательно, для ЦП и системы программирования ПСД представляет собой структурированную память, которая настраивается на требуемый тип с помощью дескрипторов блоков.

Центральной задачей, определяющей эффективность применения ПСД, является задача управления его внутренней памятью.

Допустим, что множество элементов $M = \{m_1, m_2, \ldots, m_n\}$ образует память ПСД, причем каждый $m_i \in M$ имеет свой адрес $a_i = a(m_i)$. Назовем подмножество смежных элементов памяти сегментом и обозначим $M_j = \{m_k, m_{k+1}, \ldots, m_{k+q}\}$. Адрес сегмента определяется адресом его первого элемента, т. е. $a(M_j) = a(m_k)$. Число элементов сегмента назовем размером сегмента и обозначим $d(M_j) = |M_j|$. Каждый сегмент поделим на участки $M_j = \{M_{j1}, M_{j2}, \ldots, M_{jp}\}$, полагая, что любой из них предназначен для хранения компонента блока данных. Размер участка определяется числом образующих его элементов памяти $d(M_{ji}) = |M_{ji}|$. С целью уменьшения фрагментации при выделении сегментов память ПСД делится на страницы. Сегмент памяти состоит из одной или нескольких смежных страниц, и размер сегмента определяется как $d(M_j) = |M_j|/K$, где K — число элементов памяти в странице.

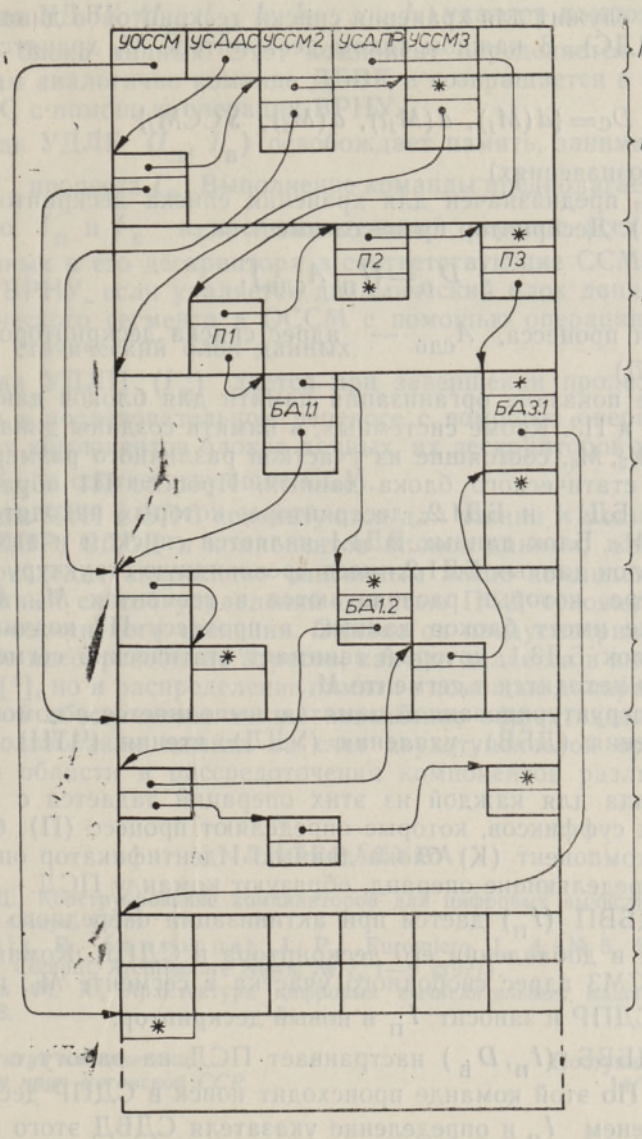
Учитывая, что списки, каталоги и деревья предполагают использование структурированной памяти спискового типа, выберем в качестве базового механизма управления памятью цепной список свободных мест (ССМ). Такое решение обеспечит функциональную однородность ПСД. Основной ССМ (ОССМ) используется в ПСД для учета свободных страниц и выделения сегментов, которые могут быть статическими или динамическими. Первые применяются для представления блоков данных статического вида, а вторые — для хранения компонентов одинакового размера, принадлежащих различным динамическим блокам. В каждом динамическом сегменте строится локальный список свободных мест сегмента (ССМС) для контроля наличия свободных участков памяти.

Управление свободной памятью ПСД осуществляется с помощью внутренних операций выделения (ВДЛ) и возвращения (ВРН). Каждая из них имеет две модификации, определяемые суффиксом сегмента (С) или суффиксом участка (У). Эти операции с конкретизирующими их параметрами образуют группу внутренних команд управления памятью ПСД.

Выполнение команды ВДЛС (⟨размер сегмента⟩) предполагает поиск в ОССМ сегмента требуемого размера, его исключение из ОССМ при удачном завершении поиска и занесение начального адреса в регистр ответа. Если же сегмент требуемого размера в ОССМ отсутствует, то в регистре ответа формируется специальный признак сложившейся ситуации.

Выполнение команды ВДЛУ (⟨размер участка⟩) заключается в поиске динамического сегмента с участком требуемого размера, исключении участка из найденного ССМС и занесении адреса участка в регистр ответа. Если такой сегмент отсутствует или ССМС пуст, выполняется поиск сегмента, у которого размер участка допустимо превышает заданный. Если и такой сегмент отсутствует, строится новый динамический сегмент с участком требуемого размера.

Выполнение команд ВРНС (⟨адрес сегмента⟩, ⟨размер сегмента⟩) и



ВРНУ (⟨адрес участка⟩, ⟨размер участка⟩) заключается в пополнении соответствующих ССМ. Если ПСД включен в состав вычислительной системы, работающей в режиме мультипрограммирования, то при выборе способа управления памятью необходимо учитывать возможность протекания нескольких процессов в системе, а также то, что каждый процесс может иметь несколько подчиненных блоков данных. Организацию внутренней памяти ПСД в этом случае можно описать следующим образом. В начале памяти находятся три системных сегмента M_1, M_2, M_3, которые создаются при выполнении команды начальной загрузки ПСД. Первый сегмент M_1 статического вида резервируется для хранения входных параметров памяти. Первый элемент сегмента M_1 представляет собой указатель основного списка свободных мест (УОССМ), второй элемент — указатель списка дескрипторов динамических сегментов (УСДДС), третий — указатель списка свободных мест второго системного сегмента (УССМ2), четвертый — указатель списка дескрипторов процессов (УСДПР) и пятый — указатель списка свободных мест третьего системного сегмента (УССМ3).

Сегмент M_2 служит для хранения списка дескрипторов динамических сегментов (СДДС). В каждом таком дескрипторе M_j хранятся четыре величины:

$$D_{\text{С}}=\{d(M_j),\ d(M_{ji}),\ a(M_j),\ \text{УССМ}_j\}$$

(в прежних обозначениях).

Сегмент M_3 предназначен для хранения списка дескрипторов процессов (СДПР). Дескриптор процесса имеет вид

$$D_{\text{П}}=\{I_{\text{П}}, A_{\text{СДБ}}\},$$

где $I_{\text{П}}$ — имя процесса, $A_{\text{СДБ}}$ — адрес списка дескрипторов блоков данных (СДБД).

На рисунке показана организация памяти для блоков данных процессов П1, П2 и П3. Кроме системных, в памяти созданы динамические сегменты M_4, M_5, M_6, состоящие из участков различного размера, и сегмент M_7 для статического блока данных. Процесс П1 обрабатывает блоки данных БД1.1 и БД1.2, дескрипторы которых находятся в сегментах M_4 и M_5. Блок данных БД1.1 является списком и находится в сегменте M_5, блок данных БД1.2 имеет древовидную структуру с узлами разного размера, которые располагаются в сегментах M_4, M_5 и M_6. Процесс П2 не имеет блоков данных, а процессу П3 подчинен один статический блок БД3.1, который занимает статический сегмент M_7, а его дескриптор находится в сегменте M_4.

Работа со структурированной памятью выполняется с помощью операций добавления (ДБВ), удаления (УДЛ), чтения (ЧТН) и записи (ЗПС).

Род операнда для каждой из этих операций задается с помощью одного из трех суффиксов, которые определяют процесс (П), блок данных (Б) или компонент (К) блока данных. Идентификатор операции и параметры, определяющие операнд, образуют команду ПСД.

Команда ДБВП ($I_{\text{П}}$) дается при активизации очередного процесса и заключается в добавлении его дескриптора в СДПР. Команда определяет по УССМ3 адрес свободного участка в сегменте M_3, переносит его в начало СДПР и заносит $I_{\text{П}}$ в новый дескриптор.

Команда ДБВБ ($I_{\text{П}}, D_{\text{Б}}$) настраивает ПСД на работу с данными пользователя. По этой команде происходит поиск в СДПР дескриптора процесса с именем $I_{\text{П}}$ и определение указателя СДБД этого процесса. С помощью операции ВДЛУ выделяется участок памяти требуемого размера для размещения дескриптора блока данных $D_{\text{Б}}$. Этот участок включается в начало СДБД, и в него заносятся данные дескриптора $D_{\text{Б}}$. ПСД анализирует вид блока и для статического блока выделяет по команде ВДЛС статический сегмент памяти и заносит его адрес в дескриптор блока. Для компонентов динамических блоков данных память при передаче дескрипторов не выделяется, поскольку такие блоки создаются путем последовательного включения компонентов.

Команда ДБВК ($I_{\text{П}}, I_{\text{Б}}, I_1, I_2, \ldots, I_k$) добавляет очередной компонент в динамический блок данных. По заданным в команде параметрам происходит последовательный поиск дескриптора процесса $I_{\text{П}}$ в СДПР, дескриптора списка $I_{\text{Б}}$ в СДБД и места добавления в найденном блоке данных по индексам $I_1, I_2, \ldots I_k$. С помощью операции ВДЛУ выделяется участок из соответствующего сегмента и включается в ранее локализованное место блока данных.

Команда УДЛК ($I_П$, $I_Б$, I_1, I_2, ... , I_k) удаляет компонент из динамического блока данных. Этот компонент определяется по заданным параметрам аналогично команде ДБВК и возвращается в соответствующий ССМС с помощью операции ВРНУ.

Команда УДЛБ ($I_П$, $I_Б$) освобождает память, занимаемую блоком данных $I_Б$ процесса $I_П$. Выполнение команды предполагает поиск блока данных по $I_П$ и $I_Б$ и последовательный перенос всех компонентов блока данных и его дескриптора в соответствующие ССМС с помощью операции ВРНУ, если удаляется динамический блок данных, или перенос статического сегмента в ОССМ с помощью операции ВРНС, если удаляется статический блок данных.

Команда УДЛП ($I_П$) дается при завершении процесса $I_П$ и заключается в последовательном переносе с помощью операции ВРНУ и ВРНС всех компонентов блоков данных, их дескрипторов и дескриптора процесса $I_П$ в соответствующие ССМ.

Команды ЧТН и ЗПС используются для чтения и модификации элементов СДПР, СДБД и компонентов блоков данных, локализация которых происходит аналогично описанным выше командам.

Выбранный способ управления памятью ПСД основан на принципе странично-сегментного членения. Однако он предусматривает не только аппаратное преобразование адресов, как это делается в вычислительных машинах [4], но и распределение памяти с помощью аппаратных средств. Этот способ обеспечивает также повышение эффективности динамического использования памяти за счет двухступенчатой организации ее свободной области и рассредоточения компонентов различных блоков данных.

ЛИТЕРАТУРА

1. Грис Д., Конструирование компиляторов для цифровых вычислительных машин, М., «Мир», 1975.
2. Beaufils, R., Sansonnet, J. P., Euromicro J., **4**, № 5, 275—282 (1978).
3. Chu, Y., Comput. Architecture News, № 7, 1—9 (1977).
4. Карцев М. А., Архитектура цифровых вычислительных машин, М., «Наука», 1978.

Институт кибернетики
Академии наук Эстонской ССР

Поступила в редакцию
14/VIII 1981

I. PALL, V. FOMITŠEV

ANDMESTRUKTUURE TÖÖTLEVA PROTSESSORI MÄLUKORRALDUS

Artikkel käsitleb spetsialiseeritud, lokaalset mälu omava struktureeritud andmetöötlusprotsessori projekteerimisega seotud probleeme. Protsessor on orienteeritud nimistu, massiivi, kataloogi ja puu tüüpi andmete töötlemisele multiprogrammeerimisrežiimis. On esitatud kahetasemelise mälujaotuse meetod, mis võimaldab vähendada mälu fragmentatsiooni ja suurendada tema kasutamise efektiivsust, samuti protsessori käsusüsteem ja käskude täitmise kirjeldus.

I. PALL, V. FOMITSEV

MEMORY ORGANIZATION IN A SPECIALIZED DATA STRUCTURE PROCESSOR

Processing of complex structured data in present computers is carried out by software or by hardware, using associative memories. This paper discusses the problems of designing a specialized data structure processor (DSP), using location-addressable memory. Associative memories are ruled out for their prohibitive cost coupled with the inconvenience for handling dynamic ordered structure. The primary motivations for developing DSP are: 1) to make programs that manipulate structured data run more effectively, 2) to reduce the amount and complexity of compilers and users' programs, 3) to reduce the time needed for dynamic memory allocation by the central processing unit.

The efficiency of data processing is particularly determined by the memory allocation techniques used in the processor. A two-level memory allocation method that reduces inner memory fragmentation is proposed.

The instruction set of the processor is specified. The instructions are used to manipulate process descriptors (PD), data structure descriptors (DSD), and data structures (DS) thems elves. A PD is inserted into the process descriptor list when the process is activated. When it is accomplished, all memory blocks occupied by PD, DSD-s and components of DS are released and returned to the corresponding available space lists. DSD specify the representation of DS in the memory and are used by DS manipulating instructions. DSD is inserted into a DSD list before processing the DS. When DS is no longer needed, its descriptor and components are deleted and the memory blocks are returned to the corresponding available space lists. The DS manipulating instructions are used to insert, delete, read and modify the components of the user's data structures.