ОПЫТ ВОССТАНОВЛЕНИЯ ФУНКЦИОНАЛЬНОСТИ АРХИВНОЙ СИСТЕМЫ МАРС ДЛЯ БЭСМ-6 И РАБОТА ПО ЕЕ ДИЗАССЕМБЛИРОВАНИЮ

Леонид Александрович Брухис
Synopsys Inc, Санта Клара, Калифорния, США, leob@mailcom.com

***Аннотация* – В сообщении рассказывается о восстановлении функциональности архивной системы МАРС БЭСМ-6 (частью не сохранившейся СУБД «КОМПАС») и работе по ее дизассемблированию.**

***Ключевые слова* – БЭСМ-6, эмуляция, СУБД КОМПАС, МАРС-6, мониторная система «Дубна», Паскаль.**

I. Введение. Упоминания системы МАРС

В юбилейном сборнике «50 лет ВЦ РАН: История, люди, достижения» читаем: в 1977-1978 гг. [В.И. Филипповым] была разработана «архивная система» Марс-6 для БЭСМ-6, представляющая собой интерпретируемый «микрокод» основных и вспомогательных операций обращения к базам данных, широко используемая в ВЦ АН СССР и других организациях страны для реализации систем хранения и СУБД различных моделей.

В 1979 г. совместно с И.В. Величко была реализована сетевая СУБД Альма-1 с доступом (и реализацией через Алгол-60, а в 1980 г. – сетевая СУБД Компас с доступом (и реализацией) через язык Паскаль БЭСМ-6 [4, с. 121].

Среди сохранившейся документации программного обеспечения БЭСМ-6, описывающей особенности входного языка Паскаль [1], встречается упоминание и краткое описание реализованного интерфейса к системе хранения данных МАРС-6, реализованной В.И. Филипповым (ВЦ АН СССР), со ссылкой на готовящуюся публикацию «В.И. Филиппов, И.В. Величко. Архивная система МАРС-6 и её интерфейсы. Обработка символьной информации № 5, в печати».

К сожалению, найти 5-й выпуск сборника «Обработка символьной информации» ни в каталоге РГБ, ни в других каталогах не удалось. Встал вопрос о возможности восстановить функциональность этой системы в режиме эмуляции, базируясь на имеющейся документации и двоичных образах дисков.

II. Интерфейс Паскаль-МАРС

Согласно [1], для работы с архивной системой МАРС-6 нужно в первую очередь указать, где располагается её двоичный код («При использовании в языке Паскаль системы МАРС необходима зона кодов системы МАРС ..., которая переписывается отдельно от личной системной библиотеки-компилятора»). Судя по тому, что там же упоминается способ создания области базы данных с помощью системы «Пульт», можно предположить, что искомая зона присутствует в составе системы «Пульт», которая сохранилась на образах дисков, используемых в эмуляторе БЭСМ-6.

В [3] находим упоминание «База данных МАРС в окончательном варианте будет представлять реализацию подмножества предложений КОДАСИЛ. ... каждая запись занимает во внешней памяти столько ячеек, сколько фактически требуется для ее размещения плюс 2-3 служебных ячейки. Имеются приказы: считать, записать, исключить, распечатать каталог, открыть подчиненную базу данных. Каталог в БД лексически упорядочен. Большим преимуществом является то, что все записи доступны из прикладных программ, написанных на АЛГОЛЕ, ФОРТРАНЕ, ПАСКАЛЕ и т.д.».

Действительно, в составе системы «Пульт» в образе диска 2048 (с кодами систем программирования), полученного из ИТМ и ВТ АН СССР, в соответствии с каталогом модулей системы Пульт в [3] обнаруживается зона 1026_8 с кодом для обработки диалоговой команды БД, названная в каталоге «КОБЛА», и следующая за ней зона 1027_8 с кодом собственно процедур работы с базой данных, названная в каталоге «БАНДИТ». В слове 1 этой зоны находится ключевое слово «БАНДИТ» («БАНк Данных И Текстов», личное сообщение В.И. Филиппова).

Считая, что именно эта зона имеется в виду под «зоной кодов системы МАРС», попробуем определить, насколько функционален интерфейс Паскаля к ней. Для этого скомпилируем программу, состоящую исключительно из вызова одной из процедур интерфейса к системе МАРС, и дизассемблируем её.

```
1 PROGRAM MAIN(OUTPUT);(*=P-,T-,S8*)
2 VAR A:ARRAY [1..100] OF INTEGER;
3 BEGIN PUTD('ARRAY', A) END.
```

Эта программа вызывает процедуру PUTD, записывающую массив из 100 слов в базу данных с ключом «ARRAY».

Псевдокомментарий (*=P-,T-,S8*) означает выключение отладочной информации и всех проверок во время исполнения для сокращения размера получаемого кода и облегчения его анализа.

В результате получаем ассемблерный код, в части, относящейся к вызову процедуры PUTD, выглядящий следующим образом:

```
, XTA ,*0023B .=6HARRAY
14, VTM ,144B (100 десятичное)
, ITS ,14
15, ATX ,
11, VTM ,GAK/7 (адрес локального массива)
14, VTM ,3 (3 – код операции PUTD)
13, VTM ,P/EF (переход на завершение после возврата)
, UJ ,PAIB (вызов интерфейсной процедуры)
```

Здесь можно увидеть, что значение ключа и длина данных передается в стеке, а адрес данных и код операции – на регистрах. Варьируя вызываемые процедуры, описанные в [1] («Для работы с системой МАРС в языке Паскаль-Монитор разработаны следующие стандартные процедуры … позволяющие отводить на некоторой памяти (барабаны, диски, ленты) область МАРСа (NEWD), открывать область (OPEND), после открытия засылать значения паскаль-переменных в область (PUTD), выбирать их из неё (GETD), исключать элементы из области (DELD) и модифицировать их значения (MODD)»), получаем следующую таблицу:

Процедура	Код операции
NEWD	2
OPEND	0
PUTD	3
GETD	4
DELD	1
MODD	5

Заметим также, что в компиляторе Паскаля имеется задел для нереализованной процедуры FIND.

Рассмотрим, что происходит в интерфейсной процедуре PAIB. В ее коде[1] перед переходом непосредственно на выполнение системы МАРС, находим перекодировку номеров процедур в «микрокод», упомянутый в юбилейном сборнике ВЦ РАН.

Номер процедуры	Микрокод (восьмеричный)
0 (OPEND)	25 12 14 11 31
1 (DELD)	27 23 14 11
2 (NEWD)	26 21 15 11 31 / 10 12 14 11
3 (PUTD)	26 21 15 11
4 (GETD)	22 14 11
5 (MODD)	20 40 26 21 00 15 11

(Процедура NEWD выполняется с помощью двух последовательных вызовов системы МАРС, с передачей системе первого и второго из двух упомянутых кодов соответственно.)

[1] В дизассемблированном виде (автокод Мадлен)
https://github.com/besm6/besm6.github.io/blob/master/sources/mars/pasbdi.asm

В то же время, в коде зоны 1027_8 диска 2048[2] не находим ничего, походившего бы на исполнение микрокоманд. Более того, в тех словах, на которые производится переход из интерфейсной процедуры Паскаля, располагаются данные, что, при попытке использования этой зоны в качестве кода системы МАРС-6 приводит к потере управления и аварийному останову. Для исполнения определенной команды в указанной зоне имеются отдельные точки входа, соответствующие шести перечисленным процедурам.

Исходя из этого, можно заключить, что версия системы «Пульт», имеющаяся на сохранившемся образе диска, содержит в себе более ранний вариант системы МАРС-6, нежели та, на которую рассчитывает интерфейс, реализованный в языке Паскаль. На этом работы по восстановлению функциональности системы были прерваны на долгое время.

III. Микрокод МАРС-6

При внимательном изучении содержимого диска 2048, полученного из учебного центра ВМФ в Сосновом Бору, в составе диалоговой системы «РЭКС-1981», о которой не удалось найти какой-либо информации, обнаружилась зона с системой МАРС-6, соответствующая ожиданиям интерфейса языка Паскаль, что позволило продолжить изыскания.

Попытаемся выяснить значения отдельных микрокодов. Обратим внимание, что код 31 встречается только для процедур работы с каталогом массивов данных. Можно предположить, что код 31 означает выбор каталога массивов данных в качестве рабочего массива. Тогда из этого будет следовать, что микрокод исполняется справа налево, от младших разрядов к старшим. Дизассемблирование зоны системы МАРС-6 подтверждает эти предположения.

Рассмотрим теперь код 11, встречающийся в микрокоде всех шести процедур. Поскольку все эти процедуры работают с ключом записи, будь то запись в составе массива данных, или имя массива для процедур NEWD и OPEND, логично предположить, что это поиск записи по ключу.

Далее обратим внимание, что процедуры, ожидающие наличие записи с данным ключом (OPEND, DELD, GETD) после кода 11 исполняют код 14, а процедуры, ожидающие отсутствие записи (NEWD, PUTD) – исполняют код 15 (процедуру MODD, как единственную из шести, содержащую нетривиальное условное поведение, рассмотрим позднее). С помощью трассировки выяснилось, что при невыполнении соответствующего условия эти коды прерывают исполнение «микропрограммы» с диагностикой об ошибке. Последний оставшийся нерассмотренным код 22 в процедуре GETD, как нетрудно предположить – команда копирования блока данных из массива в память пользователя.

Две других базовых процедуры работы с элементами данных – DELD и PUTD. В функциональность PUTD должны входить выделение памяти под новую запись, копирование блока данных из памяти пользователя в массив, и включение вновь созданной записи в сохраняемую на внешнее устройство структуру данных. Исходя исключительно из номеров микрокоманд, установить, какие конкретные действия выполняют микрокоманды 21 и 26, не представляется возможным. Команда DELD выполняет обратную операцию. Как и в предыдущем случае, какие конкретные действия выполняют микрокоманды 23 и 27, пока неясно.

Оставшиеся нерассмотренными коды 12 и 25 в процедуре OPEND копируют найденный по ключу дескриптор массива в «текущий дескриптор», запускают открытие массива по дескриптору и устанавливают открытый массив в качестве текущего.

После сказанного, детали функциональности процедуры NEWD становятся прозрачными: она состоит из установки каталога массивов в качестве текущего, выполнения PUTD, что заносит запись об имени создаваемого массива и его местоположении в каталог, и выполнения OPEND с заменой кода 25 на код 10. Код 10 – это инициализация массива данных. Этот же код используется и для создания каталога массивов данных с помощью процедуры PASACD («Для заведения специальной области "КОБЛА", содержащей каталог остальных областей, используется фортран-процедура PASACD», [1]).

Обратим здесь внимание, что повторять код 31 второй раз не нужно, поскольку каталог массивов на момент исполнения кода 11 уже является текущим, а код 14 оказывается избыточным, поскольку сразу после создания записи с указанным ключом она обязательно найдётся. Таким образом, должно быть возможно упростить библиотечную функцию PAIB, уместив микрокод каждой из вызываемых процедур в одно слово.

[2] В дизассемблированном виде (автокод БЕМШ)
https://github.com/besm6/besm6.github.io/blob/master/sources/mars/re-bandit.asm

Рассмотрим теперь процедуру MODD (в порядке исполнения, коды 11 15 00 21 26 40 20). Можно заметить, что ее начало представляет собой практически ту же последовательность микроопераций, что и PUTD (в порядке исполнения, коды 11 15 21 26), но с нулевым кодом после проверки на отсутствие ключа.

Трассировкой можно выяснить, что при отсутствии записи в массиве записи с указанным ключом выполняются микрооперации с кодами 11 15 21 26 40, а при ее наличии, когда требуется лишь заменить элемент данных в уже имеющейся в массиве записи, выполняются коды 11 15 20 (где 20 – копирование данных в уже имеющийся элемент). Логику этого поведения удалось понять лишь после дизассемблирования соответствующей команды: если код, следующий за кодом 14 или 15 – нулевой, но после него в командном слове есть ненулевые коды, то невыполнение условия приведет не к возврату с диагностикой ошибок, а к пропусканию ровно трёх микрокоманд, следующих за нулевым кодом. Код 40 – завершение выполнения командного слова.

Дизассемблирование кода МАРС-6[3] позволило приблизительно понять значение некоторых других микрокодов. Так, например, коды 01-04 осуществляют работу с итераторами: установку итератора на начало или конец списка записей, лексикографически упорядоченных по значению ключа, и декремент-инкремент итератора.

Разъяснить семантику остальных микрокодов и их назначение для реализации функциональности полноценной СУБД ещё предстоит.

IV. Структура данных МАРС-6

Выясним, насколько эффективны были структуры данных, использованные в системе. Для этого напишем программу, создающую массив, и заносящую в этот массив записи той или иной длины до тех пор, пока не возникнет ошибка по переполнению БД:

```
 1 PROGRAM MAIN(OUTPUT, BDERRN);
 2 VAR L, BDERRN:INTEGER; A:ARRAY [1..10] OF INTEGER;
 3 PROCEDURE PASSETAR(I: INTEGER); EXTERNAL;
 4 PROCEDURE PASACD(VAR I: INTEGER); FORTRAN;
 5 PROCEDURE TRY(L:INTEGER);VAR I:INTEGER;
 6 BEGIN { процедура каждый раз создает базу данных заново }
 7 I := 1520000C; PASACD(I); PASSETAR(I);
 8 I := 1520001C; NEWD('ARRAY', I); OPEND('ARRAY'); {массив длиной 1 зона}
 9 (X) FOR I := 1 TO 1024 DO BEGIN
10 PUTD(I, A:L); {запись с ключом I, длиной L и произвольным содержимым}
11 IF BDERRN <> 0C THEN EXIT X; {если была ошибка, выход из цикла}
12 END;
13 WRITELN(' В 1024 СЛОВА ПОМЕЩАЕТСЯ ', I-1:3, ' ЗАПИСЕЙ ДЛИНЫ ', L:2);
14 BDERRN := 0C; {сброс ошибки}
15 END;
16 BEGIN FOR L := 0 TO 10 DO TRY(L) END. {пробуем длины от 0 до 10}
```

Программа выдаёт следующий результат (сообщения об ошибках, кроме первого, опущены):

```
01040B ADR, 06 DATA BASE ERROR, 002621 MICRO RUN, NAME= P
В 1024 СЛОВА ПОМЕЩАЕТСЯ 235 ЗАПИСЕЙ ДЛИНЫ 0
В 1024 СЛОВА ПОМЕЩАЕТСЯ 189 ЗАПИСЕЙ ДЛИНЫ 1
В 1024 СЛОВА ПОМЕЩАЕТСЯ 158 ЗАПИСЕЙ ДЛИНЫ 2
В 1024 СЛОВА ПОМЕЩАЕТСЯ 136 ЗАПИСЕЙ ДЛИНЫ 3
В 1024 СЛОВА ПОМЕЩАЕТСЯ 120 ЗАПИСЕЙ ДЛИНЫ 4
В 1024 СЛОВА ПОМЕЩАЕТСЯ 107 ЗАПИСЕЙ ДЛИНЫ 5
В 1024 СЛОВА ПОМЕЩАЕТСЯ 94 ЗАПИСЕЙ ДЛИНЫ 6
В 1024 СЛОВА ПОМЕЩАЕТСЯ 87 ЗАПИСЕЙ ДЛИНЫ 7
В 1024 СЛОВА ПОМЕЩАЕТСЯ 78 ЗАПИСЕЙ ДЛИНЫ 8
В 1024 СЛОВА ПОМЕЩАЕТСЯ 74 ЗАПИСЕЙ ДЛИНЫ 9
В 1024 СЛОВА ПОМЕЩАЕТСЯ 69 ЗАПИСЕЙ ДЛИНЫ 10
```

В тексте сообщения об ошибке присутствует адрес, код ошибки (6 – «ПЕРЕПОЛНЕНА»), три следующие микрокоманды, начиная с микрокоманды, вызвавшей ошибку (здесь – микрокоманда 21) и

[3] https://github.com/besm6/besm6.github.io/blob/master/sources/mars/re-mars.asm

текстовое представление ключа, с которым производились действия на момент ошибки. Поскольку использованный в программе ключ был не текстовым, а числовым, содержимое этого поля содержит «мусор».

Составив несложные уравнения, можно заключить, что размер заголовка архива составляет около 60 ячеек, а количество служебных ячеек в каждой записи, не считая слова-ключа, действительно, равно 3, как и сказано выше.

V. Выводы и заключение

Архивная система МАРС-6 для БЭСМ-6 представляет собой объектно-ориентированный процедурный интерфейс обобщенного ассоциативного контейнера на внешней памяти, обеспечивающий необходимую функциональность для реализации полноценной системы управления базами данных.

Остроумное использование микропрограммирования позволило снизить нагрузку на ограниченную по размеру оперативную память БЭСМ-6. Дальнейшие работы по более детальному изучению системы МАРС-6 позволят в полной мере оценить инженерные решения, использованные при её разработке.

Источник финансирования. Благодарности

Работа выполнена автором в качестве хобби. Автор благодарит В.И. Филиппова за информацию о структуре системы МАРС-6 и её месте в составе СУБД КОМПАС.

Список литературы

1. Пирин С.И. Язык Паскаль-Монитор и его использование. М.: ВЦ АН СССР, 1978. 55 с.
2. Мазный Г.Л. Программирование на БЭСМ-6 в системе «Дубна». М.: Наука, 1978. 272 с. (Библиотечка программиста.)
3. Система ПУЛЬТ-78. Общая структура и рекомендации по обслуживанию. Под ред. В.Л. Сметанина. М: ВЦ АН СССР, 1978. 68 с.
4. 50 лет ВЦ РАН: история, люди, достижения. М.: ВЦ РАН, 2005. 319 с.