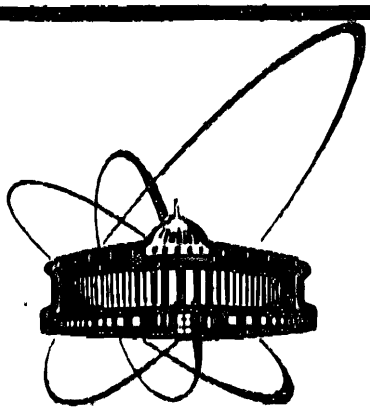

сообщения
объединенного
института
ядерных
исследований
дубна

К431

Р11-88-798

А.С.Кирилов

СРЕДСТВА ОТЛАДКИ АВТОНОМНЫХ ПРОГРАММ ДЛЯ СИСТЕМ НА ОСНОВЕ 8086/8088 МИКРОПРОЦЕССОРОВ

1988

Под автономными в данной работе понимаются программы, работающие вне операционной системы.

Уточним терминологию. Рабочей системой будем называть тот аппаратный объект /построенный на базе 8086/8088 микропроцессора/, для которого предназначается отлаживаемая программа. Эта программа далее будет также именоваться рабочей программой или просто программой.

Первоначальной задачей автора было изыскание или разработка средств отладки программы-диспетчера - одного из модулей графической рабочей станции/1/. В этом качестве используется плата, сходная по архитектуре с основной платой ПЭВМ "Правец-16" /далее просто "Правец"/. Не останавливаясь специально на архитектуре станции, отметим только, что в ее рамках диспетчер связан с "Правецем" через общую коммуникационную память.

Для подготовки программ был использован программный продукт фирмы Intel - система разработки матобеспечения с ЭВМ Intellec/2/, адаптированная для "Правеца".

Поиск был начат с исследования монитора той же фирмы. Эта программа объединяет функции простейшей операционной системы и отладчика. Как правило, она базируется в ЛЗУ, но нам удалось получить ее текст на языке PLM-86.

ИСХОДНАЯ ВЕРСИЯ

Монитор был быстро проверен в работе. Его простая, но функционально полная система команд предусматривает следующие операции:

- просмотр и задание содержимого регистров, памяти и портов;
- перепись блоков памяти;
- запуск программ в пошаговом и нормальном режимах с возможностью задания точки останова;
- ввод/вывод на перфоленту.

Число или адрес могут задаваться тремя различными способами:

- абсолютным значением;
- в формате база : смещение /если база опущена, то подразумевается CS/;

Объ... институт ядерных исследований БИБЛИОТЕКА

- в формате регистр_процессора +/- абсолютное смещение. /Регистр_процессора задается общепринятой аббревиатурой, например AX, CS и т.д./.

Передача управления между монитором и программой выполняется как вход/выход из прерывания /int3 в нормальном режиме и int1 в пошаговом/.

Для диалога с пользователем в состав рабочей системы должен входить консольный терминал.

Изучение возможностей исходной версии монитора показало, что она вполне годится как основа для более мощного и эффективного отладчика, который бы по своим свойствам более соответствовал современному уровню. Необходимость в таких дополнениях определяется относительной сложностью программы диспетчера.

НОВЫЕ ВОЗМОЖНОСТИ

В результате развития исходной версии монитора появилась система из трех программ:

- нового варианта монитора;
- загрузчика кодов /программ/;
- загрузчика команд.

Две последних являются вспомогательными.

В монитор были добавлены следующие качественно новые возможности:

- задание до восьми точек останова;
- применение имен переменных программы для задания адресов;
- задание до восьми отслеживаемых переменных и отслеживаемой области памяти;
- редактирование команд;
- загрузка программы /и таблицы символов/ с "Правеца";
- выполнение командных файлов;
- прерывание выполнения программы в произвольный момент времени.

Кроме того, вместо команд ввода/вывода на перфоленту добавлены команды для заполнения областей памяти и их сравнения.

Рассмотрим добавления подробнее.

ТОЧКИ ОСТАНОВА

Монитор записывает по адресу точки останова код команды int3, организуя таким образом самовызов. Оригинальное значение этого байта сохраняется и восстанавливается после "попадания" программы в данную точку, чтобы программа далее выполнялась правильно. Таким образом, производится самоудаление точки останова после ее срабатывания. Чтобы сделать точки постоянными до момента явного указания об их отмене, был предложен способ их восстановления, который, видимо, используется и в других отладчиках. После самоудаления точки монитор запоминает этот факт. Затем по команде продолжения выполнения программы он скрыто от пользователя переводит процессор в пошаговый режим. После выполнения одного шага монитор вновь получает управление, восстанавливает точку, нормальный режим работы процессора и возвращает управление программе.

ИМЕНА ПЕРЕМЕННЫХ

Возможность использования для адресации имен программных переменных является мощнейшим средством ускорения отладки. Известно, что компиляторы фирмы Intel при трансляции с параметром DEBUG помещают таблицу символов в объектный модуль, откуда она далее попадает и в абсолютный загрузочный модуль. Из-за отсутствия описаний необходимые знания о структуре таблицы символов были получены эмпирически. В качестве начального адреса для размещения этой таблицы в памяти был принят адрес символа MEMORY - адрес свободной после загрузки области памяти. Выяснилось, что в загрузочном модуле он упоминается первым, что облегчило задачу.

Поскольку в общем случае итоговый загрузочный модуль может быть образован из нескольких программных, то вместе с именем переменной необходимо обязательно указывать и имя модуля, в котором она определена. Поэтому была принята схема задания адресов:

```
адрес =:: имя [+/- смещение]
имя =:: *[имя_модуля :] переменная|
        *[имя_модуля :]# номер_строки_программы
смещение =:: [регистр] +/- число
регистр =:: AX|BX|CX|DX|CS|SS|ES|DS|BP|IP|SI|DI|FL|SP
```

Здесь использованы обозначения, ставшие общепринятыми: [...] заключают параметр, который можно опустить; знак | разграничивает варианты. Следует отметить, что знаки * и # являются ключевыми.

Введено понятие текущего модуля, выбираемого произвольно. При адресации внутри текущего модуля его имя можно опускать.

Следует отметить, что все прежние возможности адресации сохранены.

ОТСЛЕЖИВАНИЕ ПЕРЕМЕННЫХ

Можно задать до восьми переменных, значения которых вместе с именами и адресами будут выдаваться всякий раз, когда управление возвращается из программы в монитор. Это облегчает контроль динамики изменения этих значений.

В качестве отслеживаемых могут быть избраны любые переменные, массив или просто адрес памяти. Можно задать и формат представления каждой переменной: побитный, байтовый, ASCII, в виде слова, целого или вещественного числа, а также адреса.

Помимо переменных можно избрать для отслеживания и произвольную область памяти.

Отслеживание целиком можно выключать и включать вновь, не изменяя состава избранных переменных.

РЕДАКТИРОВАНИЕ КОМАНД

В оригинальной версии монитора каждый вводимый символ немедленно интерпретируется. Поэтому любая ошибка в наборе команды требует повторения ее ввода сначала. Это малоприятно, особенно если в команде есть адреса.

Чтобы сгладить это неудобство, в новой версии введена буферизация команд. В один из буферов команда помещается по мере ее ввода. В другом хранится предыдущая, причем если она привела к ошибке, то ее последний символ из буфера исключается. /При синтаксической ошибке последний символ команды является заведомо неправильным/.

Применение буферизации позволило включить в монитор простейшие /но полезные/ средства редактирования команд, разрешающие в момент набора новой команды использовать полностью или частично предыдущую. Кроме того, появилась возможность выполнять командные файлы /смотри далее/.

ЗАГРУЗКА ПРОГРАММ

Необходимость собственного загрузчика диктуется двумя причинами:

- потребностью загружать программы в память рабочей системы;
- потребностью загрузки таблицы символов.

Загрузка программ выполняется с дисков "Правеца", для чего потребовалось составить соответствующую программу - загрузчик кодов. Загружаемые программы должны быть подготовлены в формате абсолютного объектного файла Intel. Напомним, что в нашей рабочей системе "Правец" связан с диспетчером через коммуникационную память /КП/. Она имеет емкость 128К байт. Возник вопрос о разделении функций при загрузке и выборе способа связи между процессорами.

Так как ОС "Правеца" однопрограммная и не содержит средств поддержки программ реального времени, загрузчик на "Правеце" может быть вызван и работать только до запуска рабочей программы.

Связь монитора с загрузчиком кодов организована через двойной буфер в КП. Инициатива в общении принадлежит загрузчику кодов, т.е. "Правецу". Монитор в процессе ожидания команды периодически проверяет "почтовый ящик" на предмет появления задания и, если таковое обнаруживается, немедленно переходит к его выполнению.

Фактически загрузчик кодов только читает объектный файл с диска и по определенной процедуре передает его монитору, который и выполняет собственно загрузку.

ВЫПОЛНЕНИЕ КОМАНДНЫХ ФАЙЛОВ

При отладке сложных программ часто бывает необходимо в начале сеанса выполнить некоторый постоянный набор действий, например, задать значение каким-то переменным или установить нужный режим работы. Удобно хранить эту последовательность команд монитора в виде файла на диске "Правеца". Такой файл принято называть командным.

Для поддержки этой функции на "Правеце" была составлена программа загрузчика команд, которая использует тот же протокол связи с монитором, что и загрузчик кодов.

ПРЕРЫВАНИЕ ВЫПОЛНЯЕМОЙ ПРОГРАММЫ

Актуальность возможности экстренно прервать процесс выполнения отлаживаемой программы не нуждается в пространном обосновании. Поскольку эта возможность отсутствовала в исходной версии, то она была добавлена и реализована через механизм прерываний как попытка ввода с клавиатуры. Таким образом, в момент прогона программы нажатие на любую клавишу клавиатуры консольного терминала возвратит управление в монитор, если прерывания не запрещены самой программой.

О ПЕРЕНОСЕ МОНИТОРА

Монитор, строго говоря, не является универсальной программой. Такой цели автор перед собой и не ставил. По своему назначению монитор - это встроенный отладчик, хранящийся в ПЗУ конкретной рабочей системы. Он должен быть настроен на ее конкретную архитектуру /занимаемое адресное пространство, адреса и состав портов и т.д./. Однако, поскольку мы располагаем текстом монитора и процедурой получения кода для ПЗУ, его адаптация для другой рабочей системы, в принципе, возможна.

Если эта другая рабочая система включает "Правец" или родственный персональный компьютер, описываемые здесь свойства монитора доступны в полном объеме. В противном случае теряются возможности загрузки, следовательно, и использования таблицы символов программы и ряда других. Тем не менее, поскольку сохраняются функциональная полнота монитора как отладчика, его применение возможно и в этом случае.

Способ связи "Правеца" с процессором рабочей системы не ограничивает возможность применения монитора, поскольку это лишь вопрос составления соответствующих коммуникационных процедур /подпрограмм/. В нашем случае использована коммуникационная память, но пригодны и другие средства связи, например последовательный интерфейс.

Во всех случаях монитор нуждается в консольном терминале для диалога с пользователем.

Ориентация данного монитора на программное обеспечение фирмы Intel не представляется серьезным ограничением по причине отсутствия в ОИЯИ других столь же развитых систем разработки автономных программ.

Монитор почти целиком написан на языке PLM-86. Его текст составляет 49К байт. Есть несколько коротких программ на ассемблере. Объем кода в ПЗУ - около 9К байт.

В заключение автор выражает искреннюю благодарность своим коллегам, прежде всего К.Бруку и Й.Хайницу, за полезные предложения и помощь в работе.

ПРИЛОЖЕНИЕ

СИСТЕМА КОМАНД МОНИТОРА

Обозначения

[...] - параметр, который можно опустить;
{...} - параметр, который может быть повторен неоднократно;
| - разделитель возможных вариантов.

Общие замечания

Все команды заканчиваются нажатием "RETURN".

Выражение [W] в имени команды означает, что она может применяться для манипуляции как с байтами, так и со словами. Так, команда S применяется для просмотра и изменения содержимого памяти побайтно, а SW - пословно.

Общение с монитором - постоянный диалог. Он немедленно обрабатывает каждый введенный символ и в ряде случаев выводит некоторую информацию на экран, не дожидаясь нажатия клавиши "RETURN". Так, в командах изменения и просмотра содержимого памяти или регистров монитор вслед за введенным адресом или обозначением регистра сначала выдает его текущее значение, а затем ожидает ввода нового.

В случае ошибки монитор выдает знак # и ожидает новой команды.

Последнюю введенную команду или часть ее можно повторить символ за символом, нажимая ESC.

Нажатие клавиши CTRL/S вызывает приостановку вывода на консольный терминал до ввода CTRL/Q.

Нажатие CTRL/C прерывает вывод на терминал и выполнение текущей команды.

Для того, чтобы пользоваться при отладке именами переменных, необходимо транслировать программу с параметром DEBUG.

Отметим также, что в ряде команд /VS, VM и т.д./, в которых имена переменных и модулей являются параметрами, монитор в качестве подсказки выдает в нужном месте символ *.

Введено понятие текущего модуля, каковым может быть избран любой. Текущий модуль задается командой VM. При адресации переменных текущего модуля его имя может быть опущено.

Команды монитора

X [регистр][число]{,регистр [число]} - изменение и просмотр содержимого регистров; X - вывод содержимого всех регистров.

Пример: XDS2500 - запись 2500h в регистр DS.

S[W] адрес [значение]{,адрес [значение]} - изменение или просмотр содержимого памяти.

Пример: S4000:0 - просмотр байта по адресу
4000:0.

G [адрес][,адрес_точки_останова] - запуск программы по указанному адресу;если адрес опущен, то подразумевается текущее значение CS:IP.

Пример: G,2500:FA - запуск программы с текущего адреса CS:IP до адреса 2500:FA.

N [адрес]{,[адрес]} - запуск программы в пошаговом режиме; по умолчанию адреса с CS:IP.

M адрес_1,число_смещение_2,адрес_3 - перепись блока памяти, заданного двумя первыми параметрами по адресу_3.

D[W] адрес_1[,число_смещение_2] - вывод содержимого заданной области памяти на консольный терминал.

I[W] порт ,{,} - ввод значения с указанного порта и вывод его на консольный терминал.

O[W] порт,число - вывод заданного числа по адресу указанного порта.

F адрес_1,число_смещение_2,число - заполнение области памяти заданным значением.

C адрес_1,число_смещение_2,адрес_3 - сравнение двух областей памяти.

BSi,адрес - задание точки останова i по указанному адресу (i=0,...,7).

BCi - удаление точки останова i /если i=*, то удаляются все точки останова/.

BL - выдача списка всех точек останова.

VS имя [,тип[длина]] - включение указанной переменной в число отслеживаемых;

тип = W - слово /16-разр./;
P - адрес;
A - ASCII;
I - целое /16-разр./;
F - вещественное число /32-разр. выдается как 4 последовательных байта/;
S - байт побитно;
проч. - байт числом.

VCi - исключение переменной i из числа отслеживаемых /если i = **, то исключаются все переменные/.

VL - выдача таблицы отслеживаемых переменных.

VM имя_модуля - выбор текущего модуля.

VD[W] адрес_1,смещение_2 - задание границ отслеживаемой области памяти.

VT - включение/выключение процесса отслеживания.

VP - выдача на консольный терминал таблицы символов полностью.

VPM - выдача на консольный терминал списка имен модулей.

VPI имя_модуля - выдача на консольный терминал таблицы символов заданного модуля.

ЛИТЕРАТУРА

1. Bruck K.et al. - Microproc. and Microprogr., 1988, 23, p.375.
2. MCS-86 Software Development Utilities Operating Instructions for ISIS-II Users, Intel corp., No.98006398.

Рукопись поступила в издательский отдел
11 ноября 1988 года.

Кирилов А.С. Р11-88-798

Средства отладки автономных программ
для систем на основе 8086/8088 микропроцессоров

Приводится описание основных возможностей системы утилитов, состоящей из монитора, загрузчика кодов /программ/ и загрузчика команд. Она предназначена для отладки автономных, т.е. работающих вне среды операционной системы, программ для систем на основе 8086/8088 микропроцессоров. В приложении дан набор команд монитора. Система в основном предназначена для работы с программами, подготовленными с помощью системы разработки программного обеспечения фирмы Intel, адаптированной на ПЭВМ "Правец-16".

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Сообщение Объединенного института ядерных исследований. Дубна 1988

Перевод О.С.Виноградовой

Kirilov A.S. P11-88-798

The Utilities for Debugging of Stand-Alone
Programs for systems Built on the Base of
8086/8088 Microprocessors

A description of main features of the utility system, consisting of the monitor, the batch code loader and the batch command loader is done. This system is devoted to debugging of stand-alone programs (which work without an operating system environment) for a hardware built on the base of 8086/8088 microprocessors. There is the monitor command list in the appendix. The system is mainly intended for a use with the Intel software development system adopted to Pravetz-16.

The investigation has been performed at the Laboratory of Computing Techniques and Automation , JINR.

Communication of the Joint Institute for Nuclear Research. Dubna 1988