Троичная ЭВМ “Сетунь 70”

Брусенцов Н.П., к.т.н., Рамиль Альварес Х., к.ф.м.н.

ramil@cs.msu.su

На основе положительного опыта первой троичной машины «Сетунь» [1] в НИВЦ МГУ был разработан и в 1970 году построен опытный образец нового троичного миникомпьютера «Сетунь 70» [2,3]. Стремление повысить компактность представления программ и создать благоприятные условия для оптимальной компиляции привело к использованию в качестве машинного языка польской инверсной записи (ПОЛИЗ). В архитектуре «Сетуни 70» нет традиционного понятия машинной команды как слова, содержащего код операции и адреса операндов. Программа представляет собой последовательность трайтов-операндов и трайтов-операций.

Адресуемой единицей памяти является слог (трайт) из 6 тритов. Память 1-го уровня машины разбита на 27 страниц по 81 трайту в каждой: 9 страниц оперативной памяти (ОЗУ) и 18 страниц постоянной памяти (ПЗУ). Обмен между памятью 1-го уровня и памятью 2-го уровня на магнитном барабане производится постранично (запись и считывание), объем магнитного барабана составляет 243 страницы.

Использование ПОЛИЗа как машинного языка потребовало реализации магазинной памяти (стека). Адресные слоги управляют пересылкой данных из памяти 1-го уровня в стек, операционные – преобразованием значений в стеке и пересылкой данных из стека в оперативную память. Операции выполняются над вершиной стека (последним значением в стеке) и, возможно, над подвершиной (предпоследним значением стека) или одним из регистров машины. Одна позиция стека содержит три трайта, таким образом, в странице помещается 27 позиций стека. Номер страницы ОЗУ, которая используется под стек, и номер позиции вершины стека указываются в регистре P (ph и pa соответственно).

В «Сетуни 70» применена косвенная адресация. Имеются три регистра приписки (H1, H2, H3), значения которых есть номера страниц памяти 1-го уровня, доступных для чтения и записи (только ОЗУ). Возможна пересылка в стек и из стека значений длиной в один, два или три трайта. При засылке в стек одного трайта или двух трайтов производится дополнение справа недостающих трайтов нулями. При засылке из стека в ОЗУ производится запись требуемого числа трайтов из первых слева трайтов стека.

Адресный слог имеет вид

k1	k2	ka
1	2	3 … 6

где k1 – длина данного, k2 – регистр приписки, ka – адрес старшего трайта данного.

Имеются три режима работы машины: режим пользователя, режим макрооперации и режим прерывания. В регистр машины (C) указывается текущий режим работы (c1), номер страницы выполняемой программы (ch) и номер выполняемого слога в этой странице (ca).

В машине «Сетунь 70» различаются два уровня операции: аппаратно реализуемые и макрооперации. Аппаратно реализуемые операции бывают двух типов: 27 основных операций, выполнение которых разрешено в любом режиме, и 27 служебных, выполнение которых в режиме пользователя запрещено. Макрооперации разрешены только в режиме пользователя. При выполнении макрооперации устанавливается режим макрооперации и происходит переход на соответствующую подпрограмму, расположенную в постоянной памяти.

Операционный слог имеет вид

0	0	k3	ko
1	2	3	4 … 6

где k3 – вид операции: -1 – макрооперация, 0 – основная, 1 – служебная; ko – код операции.

Система прерываний позволяет обнаруживать в программе особых ситуаций (исчерпание или переполнение страницы стека, захват адресным слогом следующей страницы, запрещенный слог, исчерпание страницы программы) и получения сигналы от внешних устройств (памяти 2-го уровня, устройства ввода/вывода). При возникновении прерывания включается соответствующий режим, производится переключения стека операндов и переход на подпрограммы обработки прерывания.

Прерывания от сигналов внешних устройств позволяют повысить эффективность выполнения программ за счет сокращения простоя процессора при обмене информации с этими устройствами.

Несмотря на то, что программная надстройка, на которую рассчитывалась при создании машины, не была осуществлена, «Сетунь 70» удалось эффективно использовать в деле, специфика которого не была учтена в процессе разработки, а именно в автоматизированной системе обучения [4]. При разработке программного оснащения системы обучения трудности составления и отладки значительно увеличились вследствие программирования на языке машины и из-за страничной организации памяти. Стремясь уменьшить трудоемкость и повысить эффективность программистского труда, мы стали внедрять в практику нашей работы структурированное программирование [5]. Однако управляющие команды машины плохо соответствовали операторам управления, используемым в структурированных программах, и поэтому машинная эффективность этих программ получалась низкой. Решили перестроить архитектуру машины применительно к требованиям структурированного программирования, чтобы структурированные программы могли стать машинно-эффективными [6].

Архитектура «Сетуни 70» оказалась исключительно благоприятной для указанной перестройки. Вместо неиспользовавших в программном оснащении системы обучения команд E=0, E+1, E-1 и T=C были введены операционные трайты позволяющие конструировать команды:

JSR A - вызова подпрограммы-процедуры A;

BRT A1 A2 A3 - альтернативного вызова одной из трех процедур в зависимости от знака значения вершины стека операндов;

DOW A - циклического вызова процедуры A, пока текущее значение вершины стека не равно нулю.

Адреса процедур в этих командах (A, A1, A2 и A3) имеют вид:

ch	ca
1 2	3 … 6

т.е. процедуры могут находиться только в страницах оперативной памяти.

Имевшийся в машине аппарат упрятывания и восстановления значения программного счетчика, рассчитанный на три состояния: счетчик программы, счетчик макрооперации и счетчик обработки прерывания, был расширен в стек адресов возврата, содержащим 26 позиций, что обеспечивает значительную глубину вложенности процедур. Процедура заканчивается служебной операцией возврата (RMC), восстанавливающей программный счетчик значением вершины стека возврата. В следствии этого основным режимом машины стал режим макроопераций.

Правила работы с регистром H2 были изменены: по нему всегда доступна страница (ch), в которой находится выполняемый в данный момент трайт. Это значительно облегчило доступ к необходимым константам программы.

В Приложении приведены таблицы основных и служебных операций измененной машины. При описаниии операций использованы обозначения из работы [2].

Процедурное программирование, реализованное на измененной машине «Сетунь 70», на практике подтвердило заявленные Дейкстрой преимущества его метода: трудоемкость создания программ сократилась в 5-7 раз, благодаря исключению традиционной «отладки» тестированием на конкретных примерах, причем программы обрели надлежащую надежность, упорядоченность, понятность и модифицируемость.

Базовое программное оснащение модифицированной машины «Сетунь 70» [7] включает операционную систему, редактор текстов, ассемблер и комплекс сервисных программ.

Операционная система обеспечивает начало работы, организацию обмена информацией с внешней памятью, реализацию аппарата макроопераций и реакцию на прерывания. Начало работы означает загрузку первой страницы программы с перфоленты или с магнитного барабана. В ОС реализованы средства динамической отладки, т.е. задаваемые программистом при выполнении программы, и средства статической отладки, т.е. задаваемые непосредство в программе. Операционная система расположена в постоянной памяти и занимает 10 страниц.

Редактор текстов позволяет вводить текст с пишущей машинки или с перфоленты; вносить в текст изменения, удалять, переставлять и вставлять строки текста; выводить текст на пишущую машинку или на перфоленты для дальнейшего использования.

Ассемблер «Сетуни 70» предназначен для составления только структурированных программ. За счет некоторых ограничений на текст программы удалось создать однопроходный ассемблер.

Комплекс сервисных программ использует магнитный барабан для хранения отлаживаемой программы что позволяет один раз ввести программу с перфоленты, а при многократных выполнениях программы в процессе отладки загружать ее с магнитного барабана. С помощью комплекса можно вносить изменения в программу на языке машины, распечатывать или задавать значения рабочих величин.

Основным использованием машины «Сетунь 70» стала автоматизированная система обучения «Наставник» [4]. Для этого к машине был подключен класс с 27 терминалами учащихся. Программное оснащение «Наставника» позволяет проводить:

- групповое обучение учащихся различным предметам по печатным материалам в формате «Наставника»;
- автоматизированные коллоквиумы, экзамены и контрольные работы;
- проверки умений в режиме контроля или самопроверки;
- подготовку управляющей информации для выполнения вышеуказанных действий;
- обработку результатов, полученных при обучении, экзаменах и тестах, для совершенствования учебных материалов.

Особое место занимает применение аппаратуры «Наставника» для проведения автоматической диагностики цветового зрения, осуществленное совместно с кафедрой психофизиологии психологического факультета МГУ под руководством профессора Е.Н. Соколова. Аппаратура «Наставника» была доукомплектована цветным телевизором, управляемым компьютером. Система выдавала на телевизор стимул в виде двух световых сигналов различного цвета и испытуемый оценивал по девятибалльной шкале различия в каждом стимуле, сообщая системе свои оценки нажатием соответствующей клавиши на клавиатуре терминала. Обработка полученных на протяжении получаса данных позволяла построить объективные характеристики цветового зрения каждого из испытуемых [8].

Многолетнее использование системы «Наставник» в учебном процессе подтвердило надежность машины «Сетуни 70» и программного оснащения.

Двухстековая архитектура и процедурное программирование «Сетуни 70» послужили основой диалоговой системы структурированной программирования ДССП, реализованной на двоичных компьютерах [9].

Литература:

1. Брусенцов Н.П., Маслов С.П., Розин В.П., Тишулина А.М. Малая цифровая вычислительная машина «Сетунь» – М.: Изд-во Моск. ун-та, 1965. 145 с.
2. Брусенцов Н.П., Жоголев Е.А. Структура и алгоритм функционирования малой вычислительной машины // Вычислительная техника и вопросы кибернетики. Вып. 8. – Л: Изд-во Ленингр. ун-та, 1971. С. 34-51.
3. Брусенцов Н.П., Жоголев Е.А., Маслов С.П. Общая характеристика малой цифровой машины «Сетунь-70» // Вычислительная техника и вопросы кибернетики. Вып. 10. – Л.: Изд-во Ленингр. ун-та, 1974. С. 3-21.
4. Брусенцов Н.П., Маслов С.П., Рамиль Альварес Х. Микрокомпьютерная система обучения «Наставник». – М.: Наука, Физматлит, 1990. 223 с.
5. Дал У., Дейкстра Э., Хоор К. Структурное программирование. – М., "Мир", 1975.
6. Брусенцов Н.П., Рамиль Альварес Х. Структурированное программирование на малой цифровой машине // Вычислительная техника и вопросы кибернетики. Вып. 15. – М.: Изд-во Моск. ун-та, 1978. С. 3-8.
7. Рамиль Альварес Х. Базовое программное оснащение ЦМ "Сетунь 70" // Вычислительная техника и вопросы кибернетики. Вып 17. 1981 . – М.: Изд-во Моск. ун-та, 1981. С. 22-26.
8. Брусенцов Н.П., Соколов Е.Н., Зимачев М.М., Измайлов Ч.А., Маслов С.П. Рамиль Альварес Х. Автоматизированная диагностика цветового зрения // Психологический журнал, Т. 1, № 3, 1980. С. 58-84.
9. Брусенцов Н.П., Златкус В., Руднев И.А. ДССП – диалоговая система структурированного программирования // Программное оснащение микрокомпьютеров. – М.: Изд-во Моск. ун-та, 1982. С. 11-40.

Приложение

Таблица 1. Основные операции

Название	Трайт	Описание
LST	000---	x[1:18] := S; x[19:36] := Y; x := x*3↑t; S := x[1:18]; Y := x[19:36]; INCRC; pa := pa-1
COT	000--0	x[1:18] := T; **if** x[1]≠0 **then** ca := t **else** INCRC; pa := pa-1

XNN	000--+	x[1:18] := T; x[19:36] := Y;
		if x[1]≠0 **then begin** i := 1; x := x/3 **end**
		else
		begin for i := 0 **step** -1 **until** -34 **do**
		if x[2] ≠0 **then go to** EX **else** x := x*3;
		i := 0
		end else;
		EX: e := e+i; T := x[1: 18]; Y := x[19:36]; INCRC

DOW	000-0-	**if** T ≠ 0 **then** SUBR (ca+1, ca) **else begin** INCRC; INCRC **end**
BRT	000-00	INCRC; **if** T < 0 **then** z := a **else if** T = 0 **then** z := a+1**else** z := a+2; SUBR (z, ca+3); pa := pa-1
JMP	000-0+	z := T; pa := pa-1; c[1:4] := z[3:6]; c[5:8] := z[9:12]

T-E	000-+-	T := T-e*3↑12; INCRC
E=T	000-+0	e := t; INCRC; pa := pa-1
T+E	000-++	T := T+e*3↑12; INCRC
CLT	0000--	**if** S<0 **then** ca := t **else** INCRC; pa := pa-1
CET	0000-0	**if** S=0 **then** ca := t **else** INCRC; pa := pa-1
CGT	0000-+	**if** S>0 **then** ca := t **else** INCRC; pa := pa-1
JSR	00000-	INCRC; SUBR (ca, ca+1)
R=T	000000	R := T; INCRC; pa := pa-1
C=T	00000+	ca := t; pa := pa-1
T=W	0000+-	k := t; REFSYL
YFT	0000+0	Y := 0; INCRC; pa := pa-1

W=S	0000++	k := t;
		if c≠1 & ka+k1+1>40 **then begin** w := -29; **go to** IR **end**;
		if abs(h[k2])≤4 **then**
		m [h[k2], ka: ka+k1+1] := m[ph,3*(pa-1)-1:3*(pa-1)+k1] ;
		INCRC; pa := pa-1

SMT	000+--	x[1;18] := T; z := S;
		for i := 1 **step** 1 **until** 18 **do**
		z[i] := z[i]*x[i];
		S := z; INCRC; pa := pa-1

Y=T	000+-0	Y := T; INCRC; pa := pa-1

SAT	000+-+	x[1:18] := T; z := S;
		for i := 1 **step** 1 **until** 18 **do**
		if z[i]=0 **then** z[i] := x[i] **else if** z[i]≠x[i] & x[i]≠0 **then** z[i] := 0;
		S := z; INCRC; pa := pa-1

S-T	000+0-	S := S-T; INCRC; pa := pa-1
TDN	000+00	T := -T; INCRC
S+T	000+0+	S := S+T; INCRC; pa := pa-1
LBT	000++-	x[1:18] := S; x[19:36] := 0; x := x+T*R*3↑2; S := x[1:18]; Y := x[19:36]; INCRC; pa := pa-1
L*T	000++0	R := S; x := T*R*3↑2; S := x[1:18]; Y := x[19:36]; INCRC; pa := pa-1
LHT	000+++	x[1:18] := S; x[19:36] := Y; x := x+T*R*3↑2; S := x[1:18]; Y := x[19:36]; INCRC; pa := pa-1

Пояснения к таблице 1

REFSYL реализует слог-ссылку.

INCRC осуществляет последовательное увеличение ca с проверкой выхода за пределы страницы.

Регистр приписки H2 всегда указывает на CH - страницу программы.

В алгоритм описания машины “Сетунь 70” [2] надо добавить следующие описания массивов:

ternary array SR[-13:13, 1:12], psr [1:3]

(SR[-13:13, 1:12] – страница стека возвратов, а psr [1:3] – указатель окна стека возвратов)

и в спецификацию массивов

TST ~ SR[psr, 1:12] - окно стека возвратов.

```
procedure SUBR (adr, ret); ternary ret. adr;
   comment SUBR – переход на подпрограмму adr с запоминанием возврата ret;
      begin
```

```
    if ret>40 then begin w := -29; go to IR end;
    z := 0 ; z[3] := c1; z[5:6] := ch; z[9:12] := ret;
    psr := psr+1; TSR := z;
    x := m[ch, ca]; ch := x [1:2] ; ca := x [3:6]
end SUBR;
```

Если в названии операции последняя буква T, после выполнения операции происходит удаление позиции в стеке.

Таблица 2. Служебные операции

Название	Трайт	Описание
CG1	00+---	TRANSIN(-1); INCRC
CG2	00+--0	TRANSIN(0); INCRC
CG3	00+--+	TRANSIN(1); INCRC
CF1	00+-0-	COPY (-1); INCRC; pa:=pa-1
CF2	00+-00	COPY (0); INCRC; pa:=pa-1
CF3	00+-0+	COPY (1); INCRC; pa:=pa-1
LQ1	00+-+-	z:=T; q [-1]:=z[5:12]; SUIT(-1); INCRC; pa:=pa-1
LQ2	00+-+0	z:=T; q [0]:=z[5:12]; SUIT(0); INCRC; pa:=pa-1
LQ3	00+-++	z:=T; q [1]:=z[5:12]; SUIT(1); INCRC; pa:=pa-1
CP	00+0--	z:=0; z[5:6]:=ph; z[9:11]:=pa T:=z; INCRC
EXP	00+0-0	z[1:5]:=p[1:5]; p[1:5]:=p[6:10]; p[6:10]:=z[1:5] ; INCRC
LP	00+0-+	z:=T; ph==z[5:6]; pa:=z[9:11] ; INCRC
CMC	00+00-	z := 0; z[1:12] := TSR; T := z; INCRC
RMC	00+00+	z := TSR; psr := psr-1; c[1:4] := z[3:6]; c[5:8] := z[9:12]
LMC	00+00+	z := T; psr := psr+1; TSR := z[1:12]; INCRC; pa:=pa-1
LH1	00+0+-	z:=T; h[-1]:=z[4:6] ; INCRC; pa:=pa-1
LH3	00+0+0	z :=T; h[0]:=z[4:6] ; INCRC; pa:=pa-1
LH2	00+0++	z :=T; h[1]:=z[4:6] ; INCRC; pa:=pa-1
LU1	00++--	u[-1,1:4] :=t; a[-1]:=1; INCRC; pa:=pa-1
LU2	00++-0	u[0,1:4] :=t; a[0]:=1; INCRC; pa:=pa-1
LU1	00++-+	u[1,1:4] :=t; a[1]:=1; INCRC; pa:=pa-1
LF1	00++0-	LOAD(-1); INCRC; pa:=pa-1
LF2	00++00	LOAD(0); INCRC; pa:=pa-1
LF3	00++0+	LOAD(1); INCRC; pa:=pa-1
LG1	00+++-	TRANSOUT(-1); INCRC; pa:=pa-1
LG2	00+++0	TRANSOUT(0); INCRC; pa:=pa-1
LG3	00++++	TRANSOUT(1); INCRC; pa:=pa-1

Пояснения к таблице 2

COPY(i) считывает страницу q(i) памяти 2-го уровня в страницу оперативной памяти, номер которой определяется t;

LOAD(i) записывает в страницу q(i) памяти 2-го уровня страницу памяти 1-го уровня, номер которой определяется t;

SUIT(i) осуществляет поиск страницы q(i) в памяти 2-го уровня.

```
procedure TRANSIN(i);
        comment TRANSIN(i) передает в T значение g(i), преобразуя двоичное значение g(i) в троичный код;
        begin T := 0;
          if g[i,7] = 1 then t := g[i,1:6] else t := -g[i,1:6]
        end TRANSIN;

procedure TRANSOUT(i);
        comment TRANSOUT(i) передает в g(i) значение t, преобразуя троичное значение t в двоичный код;
        begin x[1:6] := t; g[i,7] := 1; g[i,1:6] := 0;
          for k := 1 step 1 until 6 do
            if x[k] ≠ 0 then begin g[i,k] := 1; if x[k] = -1 then g[i,7] := 0 end
        end TRANSOUT;
```

Опубликовано: Труды SORUCOM-2011. Развитие вычислительной техники и ее программного обеспечения в России и странах бывшего СССР. Великий Новгород, 2011. С. 71-75