БАЗОВЫЕ ТИПЫ ДАННЫХ УПРАВЛЯЮЩИХ ЭВМ СЕРИИ 5Э26 И СОВРЕМЕННЫЕ ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Леонид Евгеньевич Карпов
Институт системного программирования им. В.П. Иванникова РАН,
Московский государственный университет им. М.В. Ломоносова,
Москва, Российская Федерация, mak@ispras.ru

***Аннотация* – Во второй половине 1960-х годов в Советском Союзе началась работа над созданием вычислительных машин серии 5Э26. Разработанные ЭВМ предназначались для применения в составе систем противовоздушной обороны С-300 различных конфигураций. На этих машинах также работали несколько систем программирования, выполнялась трансляция программ с различных языков программирования, велась подготовка текстовой документации. Реализация технического задания, подразумевавшего решение такого разнообразия задач, привела к аппаратной поддержке большого разнообразия типов данных: чисел с плавающей запятой, целых чисел различных типоразмеров и адресных значений программных компонентов (переменных, констант, процедур, меток), которые связаны со структурами данных современных языков программирования. Для работы с упакованными целыми числами была разработана оригинальная система адресации, обеспечивавшая доступ к элементам памяти с точностью до отдельного разряда. Принятые при разработке аппаратуры решения оказались вполне пригодными для организации программирования на ЭВМ ряда 5Э26 не только на машинно-зависимых языках, но и на языках значительно более высокого уровня – Фортране, Паскале, Си, а также более новых языках, возникших на их основе.**

***Ключевые слова* – управляющие и универсальные ЭВМ, базовые типы данных, аппаратная поддержка языков высокого уровня.**

I. Введение

В середине 1960-х годов предприятия Министерства радиопромышленности СССР получили утвержденное Министерством обороны страны техническое задание на разработку вычислительных средств для новой системы противовоздушной обороны С-300 (по классификации НАТО SA-10 Grumble). Долгое время это задание оказывалось невостребованным: предприятиям-разработчикам содержащиеся в нем требования представлялись слишком завышенными и невыполнимыми. Однако ближе к концу десятилетия за реализацию поставленной в техническом задании задачи взялся Институт точной механики вычислительной техники Академии наук СССР (ИТМ и ВТ АН СССР). Главными конструкторами новых вычислительных машин стали [1, 2] академик Сергей Алексеевич Лебедев и его ученик Всеволод Сергеевич Бурцев, а на последнем этапе Евгений Александрович Кривошеев (рис. 1).

С.А. Лебедев

В.С. Бурцев

Е.А. Кривошеев

Рис. 1. Главные конструкторы серии ЭВМ 5Э26

Трудности с реализацией выработанного технического задания были связаны не только с высокими требованиями к аппаратной части нового управляющего вычислительного комплекса, но и вполне осознанными к этому времени требованиями к поддержке программирования на языках высокого

уровня. Сергей Алексеевич взялся за разработку новой ЭВМ, так как он был полностью к этому готов. В это время ИТМ и ВТ уже завершил разработку ЭВМ 5Э92б, прошедшей апробацию в составе крупномасштабного макета системы противоракетной обороны (системы «А») и завершал подготовку к серийному выпуску вычислительной машины 5Э51.

По итогам первых опытов создания вычислительных средств для систем противоракетной обороны Генеральный конструктор системы А-35 Григорий Васильевич Кисунько (рис. 2) рассчитал, что для надежного сопровождения и уничтожения малозаметной баллистической цели необходимо достичь быстродействия вычислительной машины на уровне 100 миллионов операций в секунду. Это значительно превышало возможности существовавших в то время ЭВМ, построенных на дискретных полупроводниковых элементах, и в ИТМ и ВТ началось обсуждение архитектуры совершенно нового многопроцессорного вычислительного комплекса «Чегет» (будущего МВК «Эльбрус-1»).

Не предполагая скорого начала финансирования амбициозного проекта, Лебедев решил воспользоваться разработкой по принятому им на себя техническому заданию на ЭВМ для противосамолетной системы, чтобы создать сразу два коллектива и вести две крупные разработки на одни и те же деньги. Это было вполне оправданно, так как, несмотря на различие в предполагаемых условиях функционирования противосамолетного и противоракетного комплексов, с точки зрения программирования для них вырабатывались достаточно близкие требования: программное обеспечение должно было создаваться на языках программирования высокого уровня.

Несмотря на такую двойственность стоявшей перед Лебедевым задачи, он отнесся к участию ИТМ и ВТ в создании вычислительных средств для системы противовоздушной обороны со всей серьезностью. Он лично ездил к Генеральному конструктору системы С-300 академику Борису Васильевичу Бункину и его соратнику Александру Алексеевичу Леманскому (рис. 2), представлял им своих инженеров и программистов, рассказывал о своем видении будущей вычислительной машины, о том как лучше построить взаимодействие нескольких крупных коллективов.

Г.В. Кисунько

Б.В. Бункин

А.А. Леманский

Рис. 2. Главные конструкторы систем управления, использовавших ЭВМ серии 5Э26

В результате работа коллективом института была выполнена. С конца 1960-х годов за 30 лет была создана серия из 5 типов вычислительных комплексов. Разработанные комплексы успешно использовались в составе систем противовоздушной обороны С-300 различных конфигураций и лет выпуска. За эту работу коллектив разработчиков дважды был отмечен Государственными премиями СССР (1980 год) и Российской Федерации (1997 год). Члены коллектива были награждены орденами и медалями за огромный вклад в создание новой вычислительной техники.

II. Серия управляющих ЭВМ 5Э26

Вычислительные машины, составившие эту серию, которую будем называть 5Э26 (в нее входили пять моделей ЭВМ – 5Э261, 5Э262, 5Э265, 5Э266 и 40У6), оказались очень удачными, способными решать самые разные задачи – расчеты траекторий как аэродинамических, так и баллистических целей, а также собственных ракет, распознавание образов, управления различными устройствами по стандартным и специализированным каналам обмена информацией. На машинах 5Э26 [3-6] работали системы программирования, выполнялась трансляция программ с различных языков программирования, велась подготовка текстовой документации. Именно такие ЭВМ можно вполне заслуженно называть

универсальными, хотя основное их предназначение – работа в составе системы управления, работающей в условиях жесткого реального времени.

Решения, принятые разработчиками при начале разработки серии управляющих ЭВМ, доказали свою жизненность и продолжают делать это до сих пор, так как эти машины и сейчас работают в составе различных систем управления. Но, если для параллельно с ними разрабатывавшихся многопроцессорных комплексов «Эльбрус» опубликовано достаточно много материалов [1, 2, 9], то эти машины, которые вполне можно рассматривать как суперЭВМ в классе мобильных вычислительных комплексов, оказались менее известными.

Между тем, структуры данных и система команд, выбранные в середине 1960-х годов, вполне пригодны для работы с самыми современными языками программирования, конечно, с учетом основных архитектурных ограничений, присущих всем моделям серии 5Э26. Для понимания этих ограничений укажем, что все эти машины имеют размер машинного слова в 32 двоичных разряда, размер памяти даже теоретически не может превысить 512 Мб (конкретные модели обладали значительно меньшей оперативной памятью), а диапазон чисел расширен за счет их необычного представления. Впрочем, в те времена общепринятого стандарта еще не существовало, его разработка началась только в 1976 году, когда компания Intel начала работу над сопроцессором для обработки чисел с плавающей запятой, а закончилась в 1985 году, когда стандарт был наконец утвержден [7].

Рассмотрим основные аппаратные возможности по представлению базовых структур данных современных языков программирования.

III. Полноразрядные числа

Языки программирования возникли из желания автоматизировать формульные вычисления. Лишь позже появились возможности по обработке логических значений. В настоящее время существует огромное множество языков программирования, построенных в рамках различных парадигм программирования – известны примеры процедурных и функциональных языков, языков разметки и моделирования, языков описания бизнес-процессов, параллельного и графического программирования. Однако необходимость проводить численные расчеты никуда не ушла, и языки, предназначенные для записи численных алгоритмов, продолжают свое развитие.

Рост разрядности вычислительных устройств не помешал введению в языки программирования возможностей вычислений с повышенной точностью, в частности, чисел с плавающей запятой двойной точности, а также длинных (и даже «длинных-длинных») и коротких целых чисел. Интересно отметить, что первые языки программирования (за исключением, конечно, языков ассемблеров) пытались более точно следовать математическим понятиям, скрывая реальную разрядность элементов памяти, отводимых для хранения значений переменных и констант [8], и вводя для обозначения их типов обобщенные наименования (LOGICAL, INTEGER, REAL). В дальнейшем, однако, проявилась тенденция не только следования математическим моделям, но и использования явных указаний разрядности при определении типов, как это было в языке Эль-76 [9], в котором определение имени должно сопровождаться указанием формата, например, с помощью спецификатора ***Ф64***, или в более новом языке Go [10], где наряду с типом ***int***, имеются типы ***int8***, ***int16***, ***int32*** и ***int64***.

1. Числа с плавающей запятой

Все модели серии 5Э26 поддерживают только один формат вещественных чисел. Для хранения вещественного числа предназначено одно слово памяти, то есть 32 двоичных разряда. Вещественные числа занимают в памяти полное слово и, как правило, представляются в нормализованном виде по основанию 4, то есть в виде

$$4^{p} \times \mathbf{M},$$

где p – порядок числа, а **M** – его мантисса. Порядок и мантисса вещественного числа представляют собой целые значения со знаком (рис. 3).

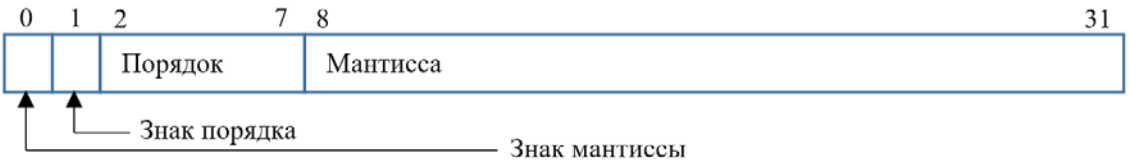

Рис. 3. Структура вещественного числа

Нулевой и первый двоичные разряды машинного слова хранят знаки мантиссы и порядка числа соответственно. Положительные порядки и мантиссы имеют нулевое значение знака, отрицательные порядки и мантиссы представляются с единицей в знаковом разряде.

Мантисса представляется как целое четверично нормализованное число, задаваемое в дополнительном коде в двоичных разрядах с 8-го по 31-й. Такой выбор основания системы счисления сделан для расширения представляемых чисел, хотя четверичная нормализация обеспечивает меньшую точность по сравнению с двоичной. Выбранный подход к представлению мантиссы приводит к фиксированному положению запятой мантиссы правее 31-го двоичного разряда. Для нормализованных неотрицательных вещественных чисел старший четверичный разряд (8-й и 9-й двоичные разряды) числа всегда отличен от нуля. Для нормализованных отрицательных чисел значение этого разряда всегда отлично от 3_4.

Значение целой мантиссы представляется в дополнительном коде в двенадцати четверичных разрядах всегда находится в диапазоне

$$-4^{12} \le M \le 4^{12} - 1$$

Двоичные разряды от второго до седьмого включительно предназначены для хранения порядка числа, то есть целого числа, представленного в обратном коде. Так как под порядок отведено три четверичных разряда, его значение всегда находится в диапазоне

$$-(4^3 - 1) \le П \le +(4^3 - 1) \text{ или } -63 \le П \le +63$$

Таким образом, диапазон представимых в аппаратуре ЭВМ вещественных чисел составляет от -4^{75} до $+4^{75}$ или приблизительно от $-1{,}427\times10^{45}$ до $+1{,}427\times10^{45}$.

Наименьшее представимое по абсолютной величине число составляет примерно 5×10^{-32}, а точнее

$$4^{-63}\times4^{11} = 4^{-52}$$

Представление порядков в обратном коде приводит к тому, что нулевой порядок может задаваться либо как $+0$, либо как -0. Два машинных слова, содержащих в поле порядка четверичные значения $0{,}000_4$ и $1{,}333_4$, задают числа с нулевым порядком (рис. 4). Это свойство чисел, представляемых в обратном коде, используется для того, чтобы отличать целые числа от вещественных чисел с нулевым порядком. Для вещественных чисел с нулевым порядком значение разряда, содержащего знак порядка, всегда выбирается противоположным знаку мантиссы (то есть собственно знаку числа).

0	1	2 7	8		31
0	1	1 1 1 1 1 1	0 0 0 0	нули	0 0 0 0
0	1	1 1 1 1 1 1	1 0 0 0	нули	0 0 0 0
1	0	0 0 0 0 0 0	1 0 0 0	нули	0 0 0 0

Рис. 4. Примеры представления вещественных чисел 0, 2^{23} и -2^{23}

В аппаратуре ЭВМ имеется также пара команд, регулирующих режим округления результата выполнения арифметических операций – сложения, вычитания, умножения и деления. Само округление проводится по-разному в разных моделях ЭВМ 5Э26. Первые модели выполняли округление установкой в единицу значения самого младшего из старших 32-х разрядов результата в случае, если хотя бы один из более младших разрядов (отбрасываемых после выполнения операции) имеет ненулевое значение. Последняя модель серии выполняет округление иначе, то есть таким образом, который более соответствует принятому в математике методу округления. В модели 40У6 округление выполняется установкой в единицу значения самого младшего из старших 32-х разрядов результата в случае, если ненулевое значение имеет самый старший из отбрасываемых после выполнения операции разрядов.

Однако поддержка вещественных вычислений не ограничивается только выполнением операций над числами известной структуры в том или ином режиме. Аппаратура также способна выполнять преобразование чисел и обеспечивать проверку корректности операндов и результатов выполнения арифметических операций. Такой подход во многом совпадает с теми возможностями, которые предлагают программистам современные языки программирования.

Все вещественные вычисления всегда проводятся только с нормализованными числами, хотя в памяти ЭВМ могут храниться любые числа, даже ненормализованные. В системе команд центрального процессора имеется специальная команда, которая преобразует ненормализованное число к нормализованному виду. Предполагается, что наличие таких команд в нужных местах программ

должны обеспечивать программисты, составляющие программы на каком-либо языке, близком к машинному, или компиляторы с языков высокого уровня.

Вещественный вычислитель центральных процессоров воспринимает целые числа как ненормализованные. Чтобы сделать возможными смешанные вычисления, в командах, выполняющих арифметические операции, в которых один из операндов представляет собой нормализованное вещественное число, а второй имеет правильный вид целого числа, второе число автоматически преобразуется к вещественному виду и нормализуется. Такое преобразование возможно всегда, так как любое представимое в ЭВМ целое число имеет свой точный аналог среди вещественных чисел.

При выполнении операций над вещественными числами возможно возникновение аппаратных прерываний, связанных либо с некорректностью операндов (например, делитель в командах деления, равный нулю), либо с некорректностью результата (переполнением или исчезновением значимости). Имеющееся во многих современных языках понятие исключительной ситуации включает в себя ситуации, возникающие при возбуждении аппаратного прерывания. В современных языках программирования, например, в языке C# [11], некоторые аппаратные прерывания включены в состав классов стандартных исключительных ситуаций. Аппаратные прерывания ЭВМ 5Э26, возникающие при работе с вещественными числами, перечислены в табл. 1, там же показаны те стандартные исключительные ситуации языка C#, которые им соответствуют.

Таблица 1
Аппаратные прерывания 5Э26, возбуждаемые при работе с вещественными числами,
и стандартные исключительные ситуации C#

Прерывание	**Исключение C#**
Исчезновение порядка	System.ArithmeticException
Некорректность аргумента	System.ArithmeticException
Некорректность деления	System.DivideByZeroException
Ненормализованное число	System.ArithmeticException
Переполнение вещественное	System.OverflowException
Потеря значимости	System.ArithmeticException

Таким образом, единственным аппаратно поддерживаемым форматом вещественных чисел в машинах серии 5Э26 является оригинальный 32-разрядный формат. Все остальные форматы, если они нужны для обеспечения повышенной точности вычислений или для расширения диапазона представимых чисел, должны моделироваться программным образом. Такой опыт у разработчиков системного программного обеспечения имелся.

2. Целые числа

Целые числа представляются в дополнительном коде. Целые числа занимают машинное слово целиком. Как и вещественные числа, они имеют размер в 32 двоичных разряда (или в 16 четверичных разрядов).

Целые числа всегда имеют нулевой порядок, представленный в обратном коде, знак которого совпадает со знаком мантиссы (то есть, фактически, со знаком числа). Только 24 младших двоичных разряда таких чисел являются значащими, при этом все оставшиеся 8 двоичных разрядов (старшие разряды слова) заполняются знаком числа (рис. 5). Тем самым значение знакового (самого старшего) разряда числа размножается так, что все двоичные разряды с 1-го по 7-й имеют одинаковые значения, повторяющие знак числа (нулевой разряд слова).

0	1	2 7	8		31
0	0	0 0 0 0 0 0 0	0 0 0 0	нули	0 0 0 0
0	0	0 0 0 0 0 0 0	1 0 0 0	нули	0 0 0 0
1	1	1 1 1 1 1 1	1 0 0 0	нули	0 0 0 0

Рис. 5. Примеры представления целых чисел $\boldsymbol{0}$, $\boldsymbol{2^{23}}$ и $\boldsymbol{-2^{23}}$

В системе команд центральных процессоров 5Э26 имеется команда, выполняющая преобразование вещественных чисел к целому виду. Такое преобразование возможно, если операнд представляет собой нормализованное вещественное число, и находится в диапазоне представимых целых 32-разрядных чисел. В противном случае возбуждается какое-либо из прерываний, связанных с ненормализованностью операнда, либо целым переполнением.

При выполнении операций над целыми числами возможно возникновение аппаратных прерываний, связанных либо с некорректностью операндов (например, делитель в командах деления, равный нулю), либо с невозможностью занесения результата в отведенное для него место в памяти ЭВМ. Последняя ситуация может возникнуть, если значение, полученное в регистре сумматора и трактуемое как знаковое, записывается не в полное слово, а в некоторую его часть – полуслово, байт или одиночный разряд (см. раздел IV). При выполнении такой записи осуществляется аппаратный контроль совпадения всех отбрасываемых (старших) разрядов сумматора со старшим (знаковым) разрядом записываемой части. В случае, если эти разряды не одинаковы по своему значению и не совпадают со знаком короткого целого числа, возбуждается соответствующее аппаратное прерывание. В машине 40У6 аналогичное прерывание может возбуждаться, если осуществляется запись числа со знаком в переменное поле, и отбрасываемые (старшие) разряды сумматора не совпадают со знаком записываемого значения. Аппаратные прерывания ЭВМ 5Э26, возникающие при работе с целыми числами, перечислены в табл. 2, там же показаны те стандартные исключительные ситуации языка C#, которые им соответствуют.

Таблица 2
Аппаратные прерывания 5Э26, возбуждаемые при работе с целыми числами,
и стандартные исключительные ситуации C#

Прерывание	**Исключение C#**
Некорректность деления	System.DivideByZeroException
Ненормализованное число	System.ArithmeticException
Переполнение целое	System.OverflowException
Переполнение целое при записи с переменным полем	System.OverflowException

IV. Упакованные данные

Система адресации, принятая в ЭВМ серии 5Э26, позволяет непосредственно оперировать с упакованными данными. Такие значения, в зависимости от используемых алгоритмов обработки упакованных данных, могут быть знаковыми и беззнаковыми. По своему размеру упакованные данные могут быть регулярными (полуслово, байт или разряд) и переменными полями. Первые модели серии 5Э26 имели возможность работать только с регулярными упакованными данными, переменные поля появились только в последней модели серии (40У6).

В памяти ЭВМ регулярные упакованные данные могут занимать 16, 8 или 1 разряд. При выборке упакованных данных из памяти на регистр сумматора центрального процессора выбираемое неполнословное число расширяется до 32-х разрядов.

Если в команде выборки указан арифметический режим считывания, осуществляется размножение старшего двоичного разряда выбираемого кода, что дополняет этот код до полных 32-х двоичных разрядов. Тем самым при считывании на сумматор упакованное число превращается в обычное целое число, отрицательное или неотрицательное, в зависимости от значения старшего разряда выбираемого упакованного числа. Логический режим считывания дополняет выбираемый из памяти код нулями, помещая сам код в младшие разряды сумматора и превращая число в обычное целое неотрицательное число.

В табл. 3 показаны диапазоны представления целых чисел, занимающих разное количество разрядов в памяти ЭВМ.

Таблица 3
Диапазоны чисел разных типоразмеров, представимых в ЭВМ серии 5Э26

Типоразмер целого числа	**Диапазон представления**
Тридцатидвухразрядное целое (A32)	$-16777216 \le A32 \le 16777215$
Шестнадцатиразрядное целое со знаком (A16)	$-32768 \le A16 \le 32767$
Шестнадцатиразрядное целое без знака (L16)	$0 \le L16 \le 65536$
Восьмиразрядное целое со знаком (A8)	$-128 \le A8 \le 127$
Восьмиразрядное целое без знака (L8)	$0 \le L8 \le 255$
Одноразрядное целое со знаком (A1)	$-1 \le A1 \le 0$
Одноразрядное целое без знака (L1)	$0 \le L1 \le 1$

1. Полусловные данные

Полусловные данные представляются в памяти ЭВМ 16-разрядными двоичными целыми значениями. Эти значения всегда выровнены и в памяти располагаются, начиная либо с нулевого (самого старшего) разряда, либо с шестнадцатого разряда (рис. 6). В наибольшей степени семантически они соответствуют коротким знаковым (***signed short***) и беззнаковым (***unsigned short***) целым данным языка Си и других языков, близких к нему по способам представления целочисленных значений.

При записи полусловного значения в память ЭВМ старшие 16 разрядов регистра сумматора отбрасываются, и в память попадают только младшие разряды. Эти данные размещаются либо в старшем, либо в младшем полуслове по исполнительному адресу, указанному в команде записи, в зависимости от номера полуслова, также имеющемуся в полном исполнительном адресе.

В арифметическом режиме записи осуществляется проверка равенства всех отбрасываемых старших разрядов регистра сумматора старшему разряду записываемого кода, то есть 16-му разряду сумматора, рассматриваемому в качестве знакового разряда короткого шестнадцатиразрядного целого числа. Если хотя бы один из отбрасываемых разрядов не совпадает со знаком короткого целого, возбуждается прерывание «Переполнение целое при записи».

В логическом режиме записи подобный контроль не выполняется, и старшие разряды сумматора просто отбрасываются.

2. Байтовые данные

Байтовые (восьмиразрядные) данные представляются в памяти ЭВМ 8-разрядными двоичными целыми значениями. Эти значения всегда выровнены и в памяти располагаются, начиная либо с нулевого (самого старшего) разряда, либо с восьмого, шестнадцатого или двадцать четвертого разряда слова (рис. 6).

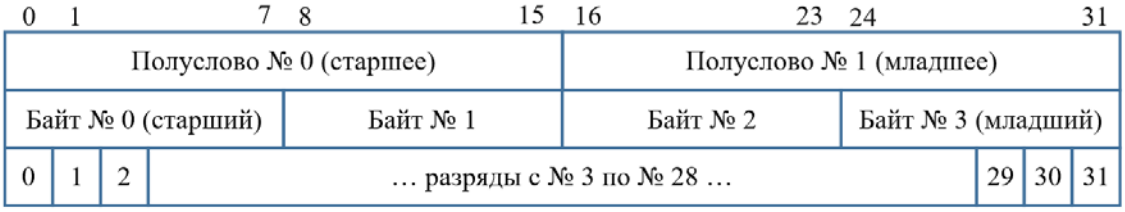

Рис. 6. Размещение целых чисел различных размеров в памяти ЭВМ

Семантически байтовые данные соответствуют символьным знаковым (***signed char***) и беззнаковым (unsigned char) данным языка Си и некоторых других языков.

При записи байтового значения в память ЭВМ старшие 24 разряда регистра сумматора отбрасываются, и в память попадают только младшие восемь разрядов. Эти данные размещаются в одном из четырех байтов слова памяти по исполнительному адресу, указанному в команде записи, в зависимости от номера байта, также имеющемуся в полном исполнительном адресе.

В арифметическом режиме записи осуществляется проверка значения регистра сумматора на равенство ***0*** или ***-1***, то есть на полное равенство значений всех отбрасываемых старших разрядов регистра сумматора значению записываемого в память разряда (31-го разряда сумматора). Если хотя бы один из отбрасываемых разрядов не совпадает со значением этого разряда, возбуждается прерывание «Переполнение целое при записи».

В логическом режиме записи подобный контроль не выполняется, и старшие разряды сумматора просто отбрасываются.

3. Одноразрядные (логические) данные

Логические данные представляются в памяти ЭВМ одноразрядными двоичными целыми значениями (рис. 6). Как и другие неполнословные значения, в зависимости от используемых алгоритмов обработки логических данных, в машинах серии 5Э26 они могут быть знаковыми и беззнаковыми. И в том, и в другом случае семантика таких данных полностью соответствует семантике данных типа ***bool*** языка C++.

При записи одноразрядного значения в память ЭВМ старшие 31 разряд регистра сумматора отбрасываются, и в память попадает только самый младший разряд регистра. Значение этого разряда размещается в том из разрядов слова памяти по исполнительному адресу, указанному в команде записи, номер которого указан в полном исполнительном адресе.

В арифметическом режиме записи осуществляется проверка значения регистра сумматора на равенство 0 или -1, то на полное равенство значений всех отбрасываемых старших разрядов регистра сумматора значению записываемого в память разряда, то есть 31-му разряду сумматора. Если хотя бы один из отбрасываемых разрядов не совпадает со значением этого разряда, возбуждается прерывание «Переполнение целое при записи». В логическом режиме записи подобный контроль не выполняется, и старшие разряды сумматора просто отбрасываются.

В случае, когда логическая информация трактуется как некоторое число без знака, допустимыми значениями, соответствующими понятиям «истина» и «ложь», становятся целые значения ***1*** и ***0***. Если же логическая информация представляется числами со знаком, эквивалентами тех же понятий являются целые числа ***-1*** и ***0***.

4. Битовые поля

В последней модели серии 5Э26 (40У6) к возможностям работы с короткими целыми числами разных размеров добавлена возможность работы со знаковыми и беззнаковыми целыми, занимающими произвольное число двоичных разрядов в одном машинном слове. Во многом такое переменное поле напоминает битовое поле, хорошо знакомое всем, программировавшим на языке Си и других языках программирования, основанных на его концепциях (рис. 7).

```
struct table_entry
{
    unsigned int type:3;     // диапазон значений от 0 до 7
    int value:5;             // диапазон значений от -16 до +15
};
```

Рис. 7. Представление битовых полей в языке Си

Переменное поле ЭВМ 40У6 определяется номером старшего двоичного разряда слова, начиная с которого в этом слове размещается поле, и длиной поля, выраженное в количестве двоичных разрядов, которое отведено для хранения поля. Номер начального (старшего) разряда и размер переменного поля не могут превышать значения 31. Если оказывается, что номера младших разрядов поля превышают значение 31, то есть выходят за границы машинного слова, поле закольцовывается, и его младшие разряды размещаются в соответствующем количестве старших разрядов того же слова памяти (рис. 8).

Рис. 8. Размещение переменных полей в памяти ЭВМ

В арифметическом режиме записи осуществляется проверка равенства всех отбрасываемых старших разрядов регистра сумматора старшему разряду записываемого кода, рассматриваемому в качестве знакового разряда неполноразрядного числа. Если хотя бы один из отбрасываемых разрядов не совпадает со знаком короткого целого, возбуждается прерывание «Переполнение целое при записи с переменным полем».

При считывании в арифметическом режиме переменного поля его старший разряд размножается до полных 32-х разрядов.

В логическом режиме записи старшие разряды сумматора, не входящие в переменное поле, просто отбрасываются. При чтении поля из памяти в логическом режиме значение, записанное в слове памяти по исполнительному адресу в границах переменного поля, передается в младшие разряды регистра сумматора. Старшие разряды этого регистра в этом режиме чтения заполняются нулями.

V. Адресная информация

Указателями и ссылками оперируют многие современные языки программирования, в которых разрешается явно использовать адреса переменных, констант, процедур, а иногда и меток. Все модели серии 5Э26 аппаратно поддерживают эти понятия, имея в системе команд центральных процессоров команду «Адрес исполнительный», которая, являясь полной аналогией операции взятия адреса языка Си, формирует специальную структуру данных, называемую «адресный дескриптор». С помощью таких дескрипторов можно осуществлять доступ к оперативной памяти вычислительного комплекса как при считывании информации на регистры процессора, так и при записи результатов вычислений в память. В составе дескриптора (рис. 9) присутствуют следующие поля:

- Режим работы с памятью (арифметический или логический);
- Размер адресуемого элемента (разряд, байт, полуслово, слово);
- Признак косвенного дескриптора (цепной разряд);
- Адрес слова в оперативной памяти;
- Уточнение адреса (до полуслова, байта или отдельного разряда).

Различия в структуре дескрипторов в разных моделях серии 5Э26 связаны только с размерами той части полного адреса элемента в дескрипторе, которая определяет слово памяти, в котором адресуемый элемент размещается. Если в первых моделях размер этого поля дескриптора составляет 18 двоичных разрядов, то в модели 40У6 его размер увеличен до 21 разряда.

Рис. 9. Структура словного, полусловного, байтового и разрядного дескрипторов

С помощью дескрипторов можно адресовать как 32-разрядные слова оперативной памяти (разряды 1 и 2 содержат код, равный $\boldsymbol{00}_2$), так и полуслова (разряды 1 и 2 содержат код, равный $\boldsymbol{01}_2$), байты (разряды 1 и 2 содержат код, равный $\boldsymbol{10}_2$) и двоичные разряды (разряды 1 и 2 содержат код, равный $\boldsymbol{11}_2$).

Если в дескрипторе признак косвенности (разряд № 3) имеет ненулевое значение, операндом операции, в которой используется дескриптор, становится не тот элемент, адрес которого находится в дескрипторе. Считается, что дескриптор адресует адресную информацию, цепной разряд которой снова проверяется на ненулевое значение, а процесс вычисления истинного исполнительного адреса продолжается до тех пор, пока не будет найден дескриптор с нулевым цепным разрядом. Для непосредственной работы с дескрипторами в системе команд процессора 5Э26 имеется возможность блокировать влияние цепного разряда на процесс формирования исполнительного адреса операнда команд считывания и записи.

Прямых аналогий дескрипторам с цепным разрядом в языках программирования нет, но хороший оптимизирующий компилятор вполне способен формировать цепочки указателей на связанные таблицы таким образом, чтобы для доступа к операнду было достаточно выполнить одну команду чтения по дескриптору, содержащему цепной разряд.

Заключение

Аппаратная поддержка базовых типов данных в ЭВМ серии 5Э26 такова, что делает эти управляющие вычислительные машины способными решать универсальные вычислительные задачи. Недостаточная точность вещественных вычислений, связанная с выбранными алгоритмами округления и нормализации результатов, вполне компенсируется широким диапазоном представимых чисел.

Данные, представимые с помощью значений базовых типов могут использоваться для конструирования более сложных структур – записей и массивов. Для такого конструирования имеется значительная поддержка со стороны аппаратуры центральных процессоров (индексные регистры, регистры шага и предела для организации циклов, а также возможность использования содержимого индексных регистров в алгоритме формирования исполнительного адреса в исполняемых командах) и базовых языков программирования (в разные годы создавались языки-автокоды – АК-26 и АК-40, язык промежуточного уровня Ярус, строились компиляторы с языков Фортран, Паскаль, Си и других языков [6]).

Несколько последних десятилетий во многом изменили взгляды сообщества программистов на необходимую им для работы аппаратную поддержку работы с базовыми типами данных и сложными структурами. Однако и сейчас принятые много лет назад решения по выбору способов представления основных данных, набора и форматов команд по их обработке, не выглядят архаичными.

Архитектура ЭВМ серии 5Э26 может служить образцом при создании современных систем программирования, информационных систем и систем управления.

В целом, общий успех, достигнутый при разработке специальных систем, в центре которых находились вычислительные машины, созданные под руководством академика С.А. Лебедева и его ближайших учеников, определялся следующими положенными в основу выполнявшихся проектов принципов:

- сознательное ориентирование на возможности отечественной науки, отечественной промышленности, работы отечественных специалистов в сочетании с тщательным изучением мировых достижений в науке и технике;
- грамотная кадровая политика, позволившая массово привлечь к работам молодых специалистов, выпускников ведущих высших учебных заведений страны;
- создание всесторонней кооперации разработчиков, при которой разработка систем, в том числе вычислительных средств проводилась одновременными совместными скоординированными усилиями математиков-алгоритмистов, разработчиков электронных компонентов, схемных решений, системного и прикладного программного обеспечения, а также сотрудников заводских конструкторских бюро и технологов серийных производственных линий;
- решение проблемы производительности вычислительной аппаратуры разработкой оригинальной архитектуры вычислительных комплексов и построением многопроцессорных вычислительных структур;
- повышенное внимание к надежности производимой аппаратуры и систематическое внедрение методов аппаратного контроля, что позволило проводить эксплуатацию и поддержание в работоспособном состоянии вычислительных средств на протяжении более чем 35 лет при минимальном наборе запасного оборудования.

Благодарности

Автор благодарит всех своих коллег, на протяжении более чем 30 лет принимавших участие в разработке аппаратуры и системного программного обеспечения для ЭВМ серии 5Э26.

Список литературы

1. С.А. Лебедев. К 100-летию со дня рождения основоположника отечественной электронной вычислительной техники. Отв. ред. В.С. Бурцев. Составители: Ю.Н. Никольская, А.Н. Томилин, Ю.В. Никитин, Н.С. Лебедева. М.: ФИЗМАТЛИТ, 2002. 440 с.
2. Бурцев В.С. Развитие специализированных вычислительных систем ПВО и ПРО // В.С. Бурцев. Параллелизм вычислительных процессов и развитие архитектуры суперЭВМ. Сост.: В.П. Торчигин, Ю.Н. Никольская, Ю.В. Никитин. М.: Торус Пресс, 2006. С. 15-24.
3. История отечественных управляющих вычислительных машин (1955-1987 гг.). Под ред. Я.А. Хетагурова. М., 2011. 216 с. (Труды Виртуального компьютерного музея).
4. Бурцева Т.А., Карпов Л.Е., Карпова В.Б. Всеволод Бурцев и суперЭВМ // Открытые системы. СУБД. 2007. № 9. С. 70-73.
5. Карпова В.Б., Карпов Л.Е. СуперЭВМ – от задач к машине // Открытые системы. СУБД. 2010. № 4. С. 54-58.

6. Липаев В.В. Кросс-система программирования ЯУЗА-6 для специализированных ЭВМ реального времени (в 70 – 80-е годы прошлого века) // Труды Института системного программирования РАН. 2011. Т. 20. С. 95-110.
7. 754-1985 – IEEE Standard for Binary Floating-Point Arithmetic. https://ieeexplore.ieee.org/document/30711 (дата обращения 18.02.2023).
8. Backus J.W., Beeber R.J., Best S., et al. The Fortran Automatic Coding System for the IBM 704. Programmer's reference manual. IBM Corporation, N. Y., October 15, 1956.
9. Пентковский В.М. Язык программирования Эль-76. Принципы построения языка и руководство к пользованию. 2-е изд. испр. и доп. М.: Наука, Гл. ред. физ.-мат. лит., 1989. 366 с. (Б-чка программиста).
10. Донован, Алан А.А.; Керниган, Брайан. Язык программирования Go = The Go Programming Language. М.: Диалектика-Вильямс, 2020. 432 с.
11. ISO/IEC 23270:2018(E). Information technology. Programming languages. C#. Third edition, 2018.