

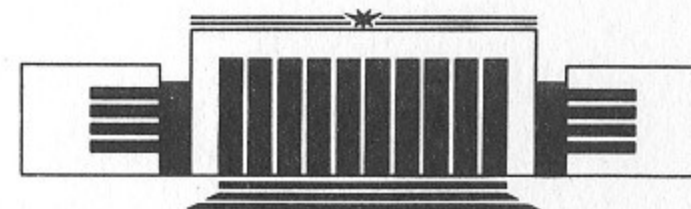


**Г.А. Аксенов, А.В. Кислицын,
В.Я. Сазанский, А.Г. Чертовских**

**УНИВЕРСАЛЬНЫЙ АРИФМЕТИЧЕСКИЙ
ПРОЦЕССОР АП-32**

**4. Арифметика с плавающей запятой,
сопроцессор элементарных функций**

ПРЕПРИНТ 90-26



НОВОСИБИРСК

Универсальный арифметический
процессор АП-32

4. Арифметика с плавающей запятой,
сопроцессор элементарных функций

Г.А. Аксенов, А.В. Кислицын,
В.Я. Сазанский, А.Г. Чертовских

Институт ядерной физики
630090, Новосибирск 90, СССР.

АННОТАЦИЯ

Описан процессор чисел с плавающей запятой универсального процессора АП-32.

8. ПРОЦЕССОР ЧИСЕЛ С ПЛАВАЮЩЕЙ ЗАПЯТОЙ

Процессор АП-32 [1] оснащен быстродействующим процессором чисел с плавающей запятой (ППЗ), реализующим не только арифметические операции, но и вычисления некоторых тригонометрических и трансцендентных функций. Он состоит из трех одноплатных модулей: умножителя, сумматора-вычитателя и сопроцессора элементарных функций (рис. 8.1).

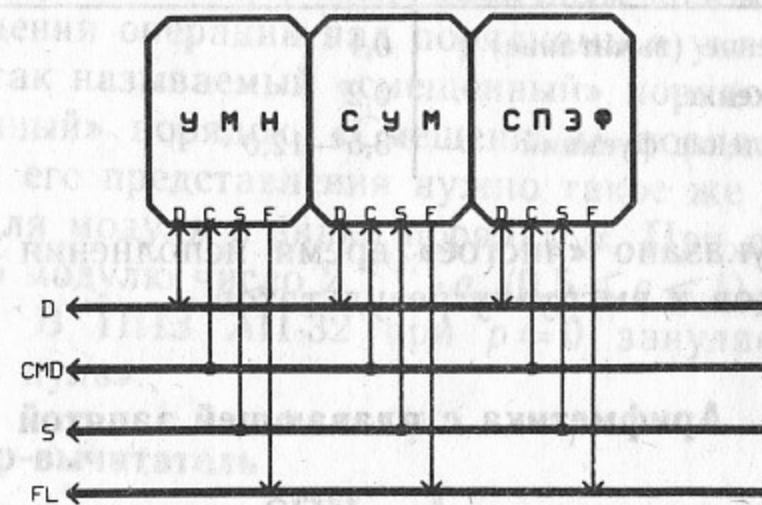


Рис. 8.1. Структурная схема ППЗ.

Все операции ППЗ выполняются с 32-разрядными результатами (однократная точность).

Источниками команд, данных, синхросигналов для модулей ППЗ являются шины CMD(16), D(32), S(4). Признаки результатов операций (флаги) по шине FL передаются в блок управления

АП-32 (автономный режим) или в интерфейс ЭВМ «накачки» (режим тестирования). Структура ППЗ позволяет увеличивать его возможности установкой нескольких однотипных модулей, специализированных сопроцессоров и т. п.

Системой команд АП-32 [1] ППЗ интерпретируется как набор специализированных F-регистров. На языке Ассемблера каждому из них присвоено символическое обозначение и код адреса в соответствующем поле команды обращения.

Инициализация УМН, СУМ, СПЭФ производится загрузкой операндов во входные F-регистры. Причем источниками операндов, как это и предусмотрено системой команд АП-32 [1], могут быть либо память данных, либо выходные F-регистры самих модулей. В первом случае инициализация модулей осуществляется командами типа память-регистр, во втором — командами типа регистр-регистр.

Применение в схемотехнических решениях методов распараллеливания операций всюду, где только возможно, позволило создать на отечественной элементной базе (микросхемы серий К500, К1500, К1800, К1802) арифметическое устройство, выполняющее операции с весьма высокой скоростью (табл. 8.1).

Таблица 8.1

Операции	Время исполнения, мкс
Сложение (вычитание)	0,1
Умножение	0,2
Вычисление функций	3,5—12,0

В табл. 8.1 указано «чистое» время исполнения без учета на загрузку операндов и выгрузку результатов.

8.1. Арифметика с плавающей запятой

Из арифметических операций в ППЗ реализуются три: сложение, вычитание (СУМ), умножение (УМН). Они являются основными, поскольку с их помощью программным путем могут быть выполнены и другие операции. Например, итерационными процессами Ньютона—Рафсона, Герона можно реализовать соответственно деление, извлечение корня квадратного.

8.1.1. Формат данных

В ППЗ АП-32 формат представления чисел с плавающей запятой (рис. 8.2) аналогичен формату DEC F, принятому в ЭВМ семейств «Электроника», VAX.

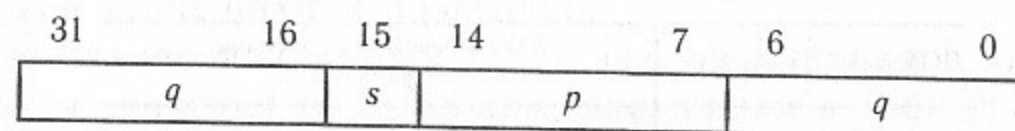


Рис. 8.2. Формат данных в ППЗ.

Здесь: p — порядок характеристики числа, q — мантисса, s — знак мантиссы (0 — плюс, 1 — минус).

В указанном формате числа используются в нормализованном виде: старший разряд мантиссы с весовым коэффициентом 2^{-1} должен быть всегда равен единице. Поэтому этот разряд («скрытый») в памяти АП-32 не хранится, на шину данных D не передается, а автоматически восстанавливается в модулях ППЗ при вычислениях. Таким образом, внутренний формат данных в них оказывается 33-разрядным (1 — знак, 8 — порядок, 24 — мантисса). Как для положительных, так и для отрицательных чисел порядок и мантисса изображаются прямым двоичным кодом.

Для упрощения операций над порядками в указанном формате используется так называемый «смещенный» порядок: $p = \alpha + 128_{10}$, где α — «истинный» порядок. «Смещенный» порядок всегда положителен и для его представления нужно такое же количество разрядов, как и для модуля и знака порядка α . При $p=0$ получается наименьшее по модулю число $2^{-127} \cdot q$, ($0,5 \leq q < 1$), — «нормализованный нуль». В ППЗ АП-32 при $p=0$ зануляется и мантисса — «истинный нуль».

8.1.2. Сумматор-вычитатель

Сумматор (вычитатель) — арифметическое устройство статического типа: в выходном регистре C всегда находится сумма (разность) двух операндов, хранящихся во входных регистрах A и B . Причем регистры A и B доступны только для записи операндов (командами типа LD), а регистр C — только для чтения результата операции (командами типа ST). Предусмотрена также возможность обмена данными между регистром C и регистрами A и B (C — источник, A и B — приемники) командами типа MVF FSR, FDR.

В командах АП-32 [1] типы операций арифметики с плавающей запятой кодируются не полем кода операций, а полем адреса F-регистров. Поэтому «физические» регистры А, В, С сумматора интерпретируются как набор «логических» F-регистров (табл. 8.2).

Таблица 8.2

Регистр сумматора	Обозначение и назначение F-регистра
А	AD — первое слагаемое SB — уменьшаемое
В	AR — второе слагаемое SR — вычитаемое
С	AF — сумма арифметическая AM — модуль суммы SF — разность арифметическая SM — модуль разности

Как следует из табл. 8.2, одному и тому же «физическому» регистру, например А, присвоено несколько логических адресов. Такой способ кодирования операций обладает одним весьма важным преимуществом: он позволяет увеличивать возможности ППЗ (установкой нескольких одинаковых модулей, разработкой новых специализированных устройств и т. п.) за счет расширения в заданных пределах набора F-регистров без модификации системы команд в целом.

Как известно, алгоритм сложения (вычитания) двух чисел $C = A \pm B$ основан на выравнивании порядков. Если вычислить разность $k = p_a - p_b$ (m_k — двоичный код числа k), то возможны следующие случаи:

1. $p_a = p_b = 0$. В этом случае в качестве результата может быть взят «истинный нуль», ($p_c = 0, q_c = 0$).
2. $p_a = 0$ или $p_b = 0$ или $m_k \geq 24_{10}$. В этих случаях по знаку числа k определяется операнд с большим порядком G , который вместе со своим знаком и берется в качестве результата ($p_c = p_g, q_c = q_g, \text{sgn} C = \text{sgn} G$).
3. $0 \leq m_k < 24_{10}$. По знаку числа k определяются операнды с большим (G) и меньшим (L) порядками.

Порядок p_g берется за основу порядка результата. Мантисса q_l операнда L сдвигается на m_k позиций вправо (денормализация),

затем проводится суммирование (вычитание) мантисс. Далее, если необходимо, осуществляется нормализация мантиссы результата путем сдвига ее кода либо влево на N позиций, либо на одну позицию вправо. Соответственно этому корректируется и порядок результата.

Поскольку в формате DEC F положительные и отрицательные числа представлены в прямом коде, то операции над знаками и мантиссами выполняются отдельно.

В представленной версии СУМ знакам операндов G и L присваиваются значения по правилам, указанным в табл. 8.3.

Таблица 8.3

Исполняемые операции	sgn G		sgn L	
	$k \geq 0$	$k < 0$	$k \geq 0$	$k < 0$
Сложение	sgn A	sgn B	sgn B	sgn A
Вычитание	sgn A	$-\text{sgn} B$	$-\text{sgn} B$	sgn A

Процесс формирования порядка и мантиссы результата однозначно определяется только знаками операндов G и L (табл. 8.4).

Таблица 8.4

sgn $G = \text{sgn} L$		sgn $G \neq \text{sgn} L$	
p_c	q_c	p_c	q_c
$p_g + v$	$(q_g + 2^{-k} q_l) 2^{-v}$	$p_g - n$	$(q_g - 2^{-k} q_l) 2^n$

Здесь v — значение выхода сумматора «Перенос» после суммирования мантисс (0 — нет переноса, 1 — есть перенос), n — количество разрядов, необходимое для нормализации результата после вычитания мантисс.

Если при образовании суммы $p_g + v$ устанавливается «Перенос» сумматора порядков, то должно быть зафиксировано состояние «переполнение разрядной сетки». Аналогично, в случае $n > p_g$ фиксируется состояние «антипереполнение», а порядок и мантисса результата зануляются ($p_c = 0, q_c = 0$).

Знак результата определяется знаками операндов G и L , и состоянием разряда переполнения сумматора-вычитателя мантисс (табл. 8.5).

Таблица 8.5

Условие	sgn C
$v=1$	sgn G
$v=0$	sgn L

Таким образом, в рассматриваемом случае ($0 \leq m_k < 24_{10}$) алгоритм работы СУМ должен состоять из последовательности следующих основных действий.

1. Сравнением порядков исходных чисел A и B определить, какое из них является G , а какое — L . В зависимости от заданной операции присвоить этим операндам знаки в соответствии с табл. 8.3.
2. В зависимости от соотношений между знаками G и L установить на сумматоре-вычитателе мантисс команду суммирования или вычитания (табл. 8.4).
3. По знакам операндов G и L и по признаку результата операции суммирования-вычитания мантисс (v) сформировать знак результата (табл. 8.5).

Функциональная схема СУМ показана на рис. 8.3. СУМ полностью реализован на комбинационных схемах без элементов запоминания промежуточных результатов.

Командами обращения к входным F-регистрам исходные данные загружаются в два 32-разрядных регистра PA и PB. Команды дешифрируются схемой управления ввода/вывода (СУВВ), и признак типа операции, задаваемый обращением к соответствующим F-регистрам (см. табл. 8.2), запоминается триггером операций на весь цикл работы СУМ.

Порядки чисел A и B сравниваются схемой сравнения порядков (ССП), состоящей из 8-разрядного быстродействующего вычитателя, собранного на микросхемах К500ИП181, и нескольких логических схем.

Для каждого из трех возможных случаев соотношений порядков (пп.1—3 алгоритма) ССП формирует соответствующие логические сигналы, с помощью которых управляет коммутаторами порядков (КП) и мантисс (КМ). При этом порядок операнда G подается на схему коррекции порядка (СКП), а мантисса операнда L — на денормализатор (ДНМ). Одновременно ССП формирует также и код m_k для управления ДНМ.

ДНМ производит логический сдвиг вправо 24-разрядного числа

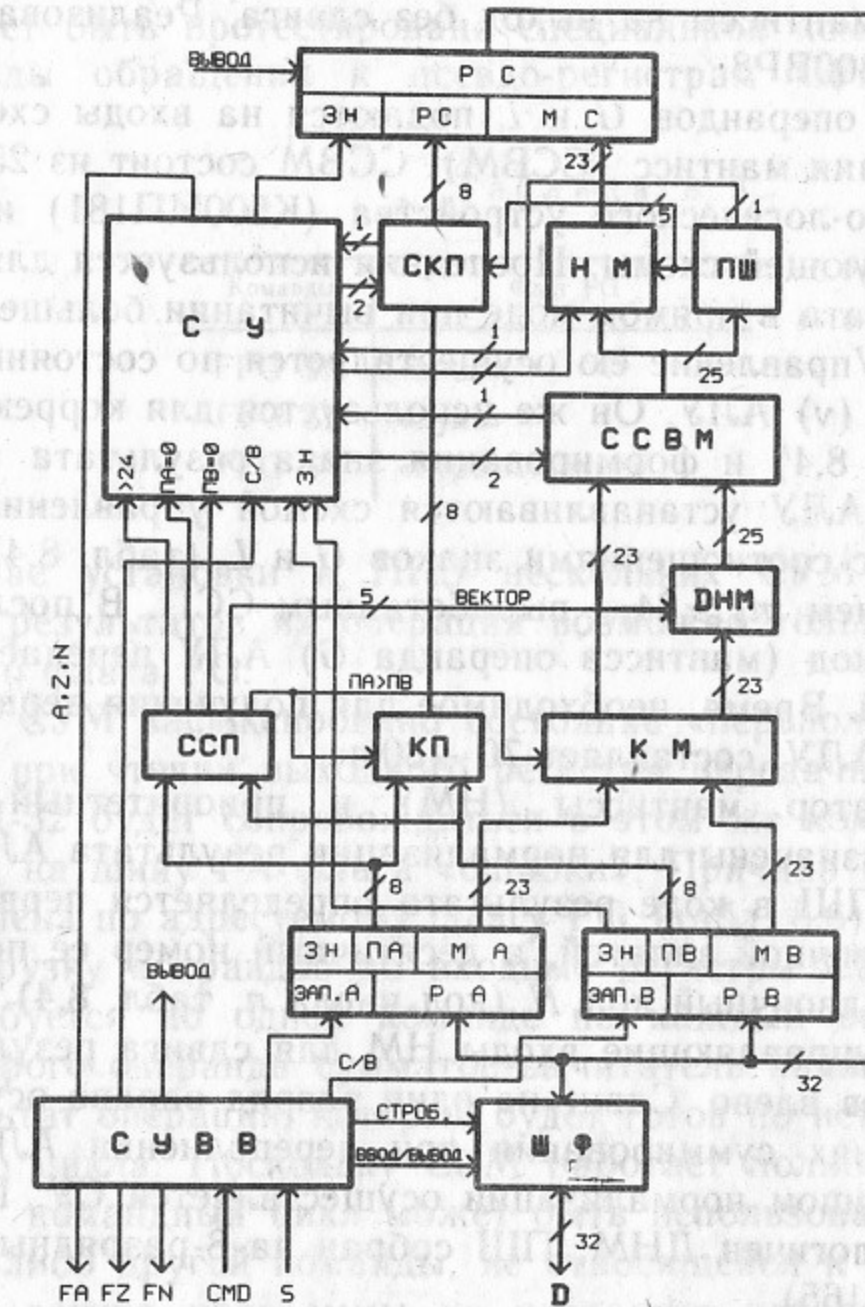


Рис. 8.3. Функциональная схема СУМ.

на m_k разрядов с весьма высоким быстродействием (15—20 нс). Кроме того, ДНМ восстанавливает «скрытый» бит мантиссы операнда L . При равенстве порядков ($p_a = p_b$, $m_k = 0$) ДНМ передает входной код мантиссы на выход без сдвига. Реализован ДНМ на трех БИС К1800ВР8.

Мантиссы операндов G и L подаются на входы схемы сложения и вычитания мантис (ССВМ). ССВМ состоит из 25-разрядного арифметико-логического устройства (К500ИП181) и управляемой инвертирующей схемы. Последняя используется для представления результата в прямом коде при вычитании большей мантиссы из меньшей. Управление ею осуществляется по состоянию разряда переполнения (v) АЛУ. Он же используется для коррекции порядка (см. табл. 8.4) и формирования знака результата (табл. 8.5). Команды на АЛУ устанавливаются схемой управления (ССУ) в соответствии с соотношениями знаков G и L (табл. 8.4) или логическим условием $m_k \geq 24_{10}$, выработанным ССП. В последнем случае входной код (мантисса операнда G) АЛУ передает на выход без изменений. Время, необходимое для получения верного результата работы АЛУ, составляет 70—80 нс.

Нормализатор мантиссы (НМ) и приоритетный шифратор (ПШ) предназначены для нормализации результата АЛУ мантиссы. С помощью ПШ в коде результата определяется первая единица справа от двоичной запятой, а десятичный номер ее позиции преобразуется в двоичный код N (код числа n , табл. 8.4). Последний подается на управляющие входы НМ для сдвига результата АЛУ на N разрядов влево. Сдвиг на один разряд вправо осуществляется в операциях суммирования при переполнении АЛУ мантиссы. Управление типом нормализации осуществляется СУ. По исполнению НМ аналогичен ДНМ, ПШ собран на 8-разрядных шифраторах (К500ИВ165).

Код порядка результата (p_c) формируется схемой коррекции порядка СКП (8-разрядное быстродействующее АЛУ на микросхемах К500ИП181) согласно табл. 8.4. Как и на АЛУ мантиссы, команды на АЛУ СКП устанавливаются СУ. Схема управления формирует также знак и флаги результата.

СУМ имеет два неадресуемых флага: FZ — «нуль», FN — «знак». Эти флаги по одноименным шинам передаются непосредственно в БУ АП-32 и могут быть протестированы командами переходов. Значения флагов не изменяются до тех пор, пока не изменится содержимое какого-либо входного регистра. Предусмотрена возможность исследования признаков результата по адре-

суемому флагу FG. В этом случае командами обращения (FFG flg) к псевдо-регистрам СУМ признаки результата устанавливаются на шину FG на промежуток времени, равный командному циклу АП-32. Значение флага FG запоминается в БУ АП-32 и уже затем может быть протестировано специальной командой перехода. Команды обращения к псевдо-регистрам СУМ указаны в табл. 8.6.

Таблица 8.6

Команды	Флаг FG
FFG SN	«знак»
FFG SZ	«нуль»
FFG SV	«переполнение»

В случае установки в ППЗ нескольких СУМ исследование признаков результатов их операций возможно только с помощью адресуемого флага FG.

Если в СУМ зафиксировано состояние «переполнение рабочей сетки», то при чтении выходного регистра передача результата в память АП-32 будет сопровождаться в этом же командном цикле установкой на шину FA флага «ошибка». Причина ошибки может быть выяснена по адресуемому флагу FG (табл. 8.6).

На загрузку операндов во входные регистры СУМ из памяти АП-32 требуется по одной команде на каждый операнд. После записи второго операнда сумматор-вычитатель начинает формировать результат операции, который будет готов по истечении одного командного цикла. Поскольку СУМ работает полностью автономно, то этот командный цикл может быть использован на исполнении какой-либо другой команды, не относящейся к данной операции. Если логика программы не позволяет использовать такую возможность, то в этом случае, как показано в примере 1, должна быть исполнена команда NOP («нет операции»).

Пример 1.

```

LD AD, A ; A
LD AR, B ; B
NOP      ; A + B
ST AF, C ; C = A + B

```


8.1.3. Умножитель

Так же, как и СУМ, умножитель — устройство статического типа: в выходном регистре С всегда хранится произведение содержимых входных регистров А и В. Инициализация работы УМН производится командами обращения к его входным F-регистрам (табл. 8.7).

Таблица 8.7

Регистры умножителя	Обозначение и назначение F-регистров
А	MP — множимое
В	MR — множитель
С	MF — произведение

Входные F-регистры доступны только по записи, а выходной — по чтению.

Алгоритм умножения чисел в формате DEC F сводится к следующему:

- 1) перемножению мантисс сомножителей;
- 2) суммированию порядков сомножителей за вычетом «смещения» (128_{10});
- 3) нормализации мантиссы результата путем сдвига ее кода влево на один разряд с последующей коррекцией порядка результата.

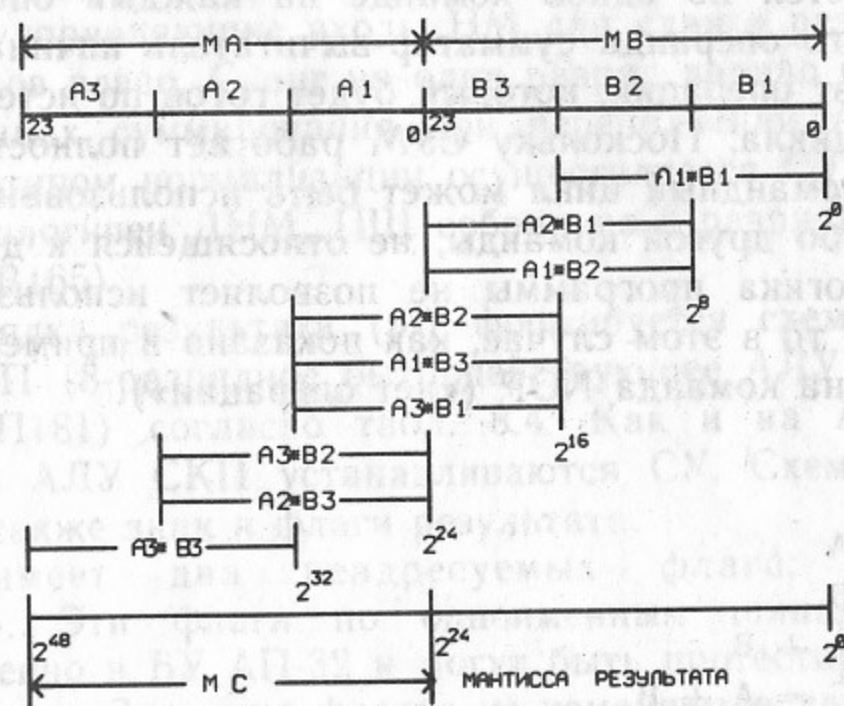


Рис. 8.4. Способ перемножения мантисс.

При $p_a=0$ или $p_b=0$ в качестве результата берется «истинный ноль».

Наиболее длительной по времени и громоздкой по схемотехнической реализации является операция перемножения мантисс. В УМН используется метод суммирования частичных произведений (рис. 8.4).

Каждое произведение $A_i B_j$ образуется с помощью параллельных умножителей типа 8·8 и с учетом своего весового коэффициента суммируется с остальными. Поскольку произведение имеет одинарную длину (24 разряда), нет смысла удваивать длину сумматора частичных произведений. Известно, что достаточно увеличить его длину на количество разрядов S , удовлетворяющего условию: $S \geq \ln(N-S)$, где N — число разрядов сомножителей ($N=24$). В УМН принято $S=8$, поэтому произведение $A_1 B_1$ не формируется, а из $A_2 B_1$ и $A_1 B_2$ используются только старшие 8 разрядов.

Функциональная схема УМН показана на рис. 8.5.

Сомножители загружаются в регистры РА и РВ. Общей частью этих регистров является регистр мантисс РМ, играющий роль буферного для схемы частичных произведений (СЧП). С помощью РМ мантиссы сомножителей по очереди загружаются во внутренние регистры умножителей К1802ВРЗ.

Знак результата формируется схемой ФЗ суммированием по модулю 2 знаков сомножителей.

Порядки сомножителей суммируются сумматором порядков (СП). Формируемый им код корректируется вычитанием «смещения» и единицы в случае необходимости нормализации мантиссы результата. Корректор порядка (КП) представляет собой 10-разрядный быстродействующий вычитатель, в котором два дополнительных разряда используются для фиксации «переполнения» или «антипереполнения» разрядной сетки. Если порядок результата оказался равным нулю, что определяется компаратором КО, то зануляется и мантисса.

СЧП состоит из массива умножителей и сумматоров. Она формирует 25-разрядную мантиссу произведения, которая затем нормализуется с помощью НМ.

УМН имеет только один неадресуемый флаг — FGT — «переполнение разрядной сетки». Остальные флаги — адресуемые (табл. 8.8) и тестируются командами типа FFG flg.

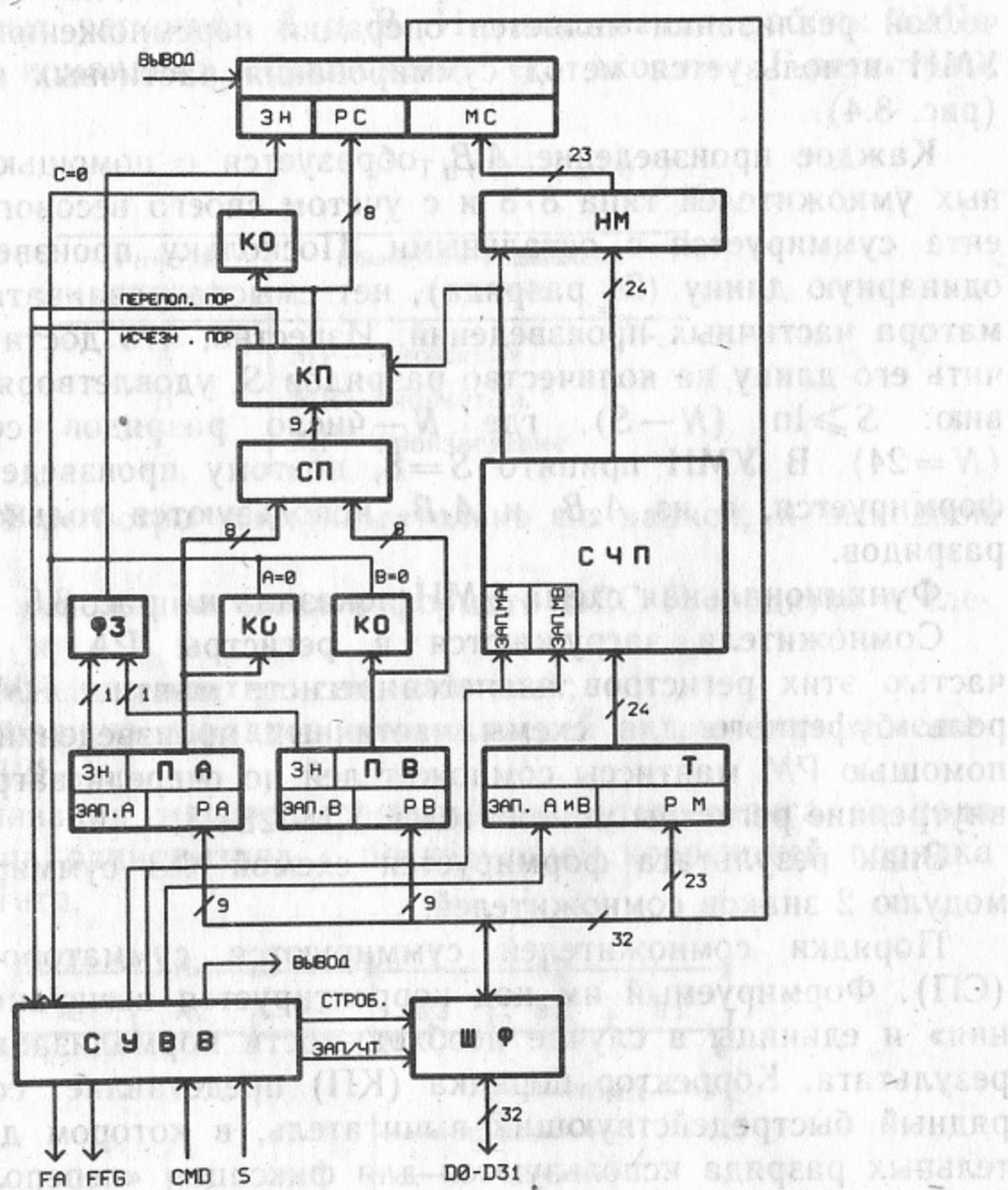


Рис. 8.5. Функциональная схема УМН.

Таблица 8.8

Команды	Флаг FG
FFG MN	«знак»
FFG MZ	«нуль»
FFG MV	«антипереполнение»

Если в УМН зафиксировано состояние «переполнение разрядной сетки», то команда чтения произведения будет сопровождаться установкой флага FA («ошибка»). При «антипереполнении» результат операции УМН зануляется.

Синхронизация работы УМН с блоком управления АП-32 осуществляется так же, как и для СУМ, с помощью команд NOP. Однако поскольку время исполнения операции УМН составляет 200 нс, то таких команд необходимо исполнить две (см. пример 1). В сложных алгебраических вычислениях применение команды MVF FSR, FDR (регистр-регистр), равно как и других из набора команд АП-32, позволяет оптимизировать программу за счет исключения «пустых» команд.

Создание быстродействующих устройств арифметики с плавающей запятой на отечественной элементной базе оказывается довольно сложной схемотехнической задачей. Каждый модуль ППЗ состоит из 150 микросхем ЭСЛ разной степени интеграции и собран на плате размером 350×400 мм.

9. СОПРОЦЕССОР ЭЛЕМЕНТАРНЫХ ФУНКЦИЙ

СПЭФ реализует программно-аппаратный метод вычисления элементарных функций, входящих в базовый набор ФОРТРАНа-4: SIN(X), COS(X), ARCTG(X), SQRT(X), LN(X), EXP(X).

Использование данного модуля в составе процессора АП-32 не является обязательным и будет целесообразным только при большом объеме вычислений, включающих заданный набор функций, а также производных от них. В частности, это оказывается полезным в реализации алгоритмов отбора «полезных» событий при обработке экспериментальных данных с детекторов элементарных частиц.

При отсутствии СПЭФ функции могут вычисляться полностью программным способом, что является обычным для серийных ЭВМ.

9.1. Алгоритм вычисления функций

Для вычисления упомянутых функций используется алгоритм Волдера (CORDIC-алгоритм), математические основы которого изложены в [2—4].

Важным достоинством этого алгоритма является то, что для вычисления всего набора элементарных функций применяются однотипные системы итерационных уравнений. Они описывают вращение вектора (x, y) в некотором координатном пространстве и в зависимости от вида координатной системы делятся на 3 типа:

- 1) вращение вектора в круговой координатной системе;
- 2) вращение вектора в линейной координатной системе;
- 3) вращение вектора в гиперболической координатной системе.

Существуют также две процедуры поворота вектора, которые получили название ROTATION (вращение) и VECTORING (векторизация).

Исходными данными для режима ROT являются координаты (x_1, y_1) начального вектора и угол, на который его нужно повернуть. Результатом поитерационной процедуры вычислений будут координаты (x_n, y_n) нового вектора. Для режима VEC исходными данными являются только координаты (x_1, y_1) , а результатом — модуль и аргумент вектора.

Обобщенную систему уравнений CORDIC-алгоритма можно записать следующим образом:

$$\begin{aligned} x_{i+1} &= x_i + S_x y_i 2^{-l}; \\ y_{i+1} &= y_i + S_y x_i 2^{-l}; \\ z_{i+1} &= z_i + S_z C_i, \quad i=1, 2, \dots, n. \end{aligned} \quad (9.1)$$

Здесь i — номер итерационного шага, l — номер итерации. В общем случае i и l друг с другом не совпадают (табл. 9.1).

Тип вычисляемой функции задается тремя значениями начальных условий x_1, y_1, z_1 , параметрами S_x, S_y, S_z и последовательностью констант C_i , участвующих в итерационном процессе.

Таблица 9.1

Координатная система	Режим	l	S_x	S_y	S_z	C_i
Круговая	ROT VEC	$i-1$	$-\text{sgn } z_i$ $\text{sgn } y_i$	$\text{sgn } z_i$ $-\text{sgn } y_i$	$-\text{sgn } z_i$ $\text{sgn } y_i$	$\text{arctg}(2^{-l})$
Линейная	ROT VEC	i	0 0	$\text{sgn } z_i$ $-\text{sgn } y_i$	$-\text{sgn } z_i$ $\text{sgn } y_i$	2^{-l}
Гиперболическая	ROT VEC	i итер. 4, 13, 40, ..., k , $3k+1$ выполняются дважды	$\text{sgn } z_i$ $-\text{sgn } y_i$	$\text{sgn } z_i$ $-\text{sgn } y_i$	$-\text{sgn } z_i$ $\text{sgn } y_i$	$\text{arth}(2^{-l})$

Вид констант C_i определяется типом координатной системы, а их количество на каждую функцию равно числу итерационных шагов n , которое, в свою очередь, зависит от желаемой точности аппроксимации. Константы подсчитываются заранее и хранятся в постоянном запоминающем устройстве сопроцессора. В процессе решения они извлекаются из ПЗУ по одной на каждой l -итерации.

Параметры S_x, S_y, S_z определяются либо текущим знаком переменной z_i (режим ROT), либо текущим знаком переменной y_i (режим VEC).

Функции SIN(X), COS(X), ARCTG(X)

Эти тригонометрические функции вычисляются итерационными уравнениями (9.1) в круговой координатной системе (табл. 9.2).

Таблица 9.2

Режим	Начальные условия	Результат	Область сходимости
ROT	$x_1 = 1/K_n$ $y_1 = 0$ $z_1 = \varphi - \pi/2, \varphi \geq 0$ $x_1 = -1/K_n$ $y_1 = 0$ $z_1 = \varphi + \pi/2, \varphi < 0$	$x_n = \cos \varphi$ $y_n = \sin \varphi$	$ \varphi \leq 1,1\pi/2$
VEC	$x_1 = x$ $y_1 = y$ $z_1 = 0$	$z_n = \text{arctg}(x/y)$	не ограничена

Здесь $K_n = \prod_{i=1}^n \sqrt{1+2^{-2i}}$ — коэффициент удлинения вектора ($K_n \approx 1,647$).

Для большинства функций, вычисляемых с помощью CORDIC-алгоритма, итерационный процесс сходится только в ограниченном интервале изменения аргумента. Поскольку в принятом для АП-32 представлении вещественных чисел (в формате с плавающей запятой) аргументы элементарных функций могут изменяться в гораздо более широких пределах, чем позволяет область сходимости, то возникает необходимость приведения аргумента в нужный диапазон. Воспользовавшись некоторыми свойствами вычисляемых функций, а также свойствами чисел, представленных в формате с плавающей запятой, можно сформулировать довольно простые методы приведения аргумента к интервалу, удовлетворяющему условию сходимости алгоритма.

Наиболее просто формулы приведения записываются для периодических функций $\text{SIN}(X)$, $\text{COS}(X)$. Если выделить целую часть M и остаток φ от деления X на $\pi/2$, то

$$\sin\left(\frac{\pi}{2}M + \varphi\right) = \begin{cases} \sin \varphi, & \text{mod}_4 M = 0 \\ \cos \varphi, & \text{mod}_4 M = 1 \\ -\sin \varphi, & \text{mod}_4 M = 2 \\ -\cos \varphi, & \text{mod}_4 M = 3 \end{cases}$$

Для функции $\text{ARCTG}(X)$, как указано в табл. 9.2, область сходимости бесконечна. Однако ограничение здесь накладывает внутренний формат представления данных в сопроцессоре, что будет обсуждаться ниже. Поэтому искомую функцию удобно вычислять, пользуясь следующими соотношениями:

$$\text{arctg}(x) = -\pi/2 - \text{arctg}(1/x), \quad x \leq 0;$$

$$\text{arctg}(x) = \pi/2 - \text{arctg}(1/x), \quad x > 0.$$

Функция $1/X$

Функция $1/X$ вычисляется в линейной координатной системе (табл. 9.3), в АП-32 используется в операциях деления.

Таблица 9.3

Режим	Начальные условия	Результат	Область сходимости
VEC	$x_1 = X$ $y_1 = 1$ $z_1 = 0$	$z_n = 1/X$	$ X \geq 1$

Чтобы получить формулу приведения аргумента X в область сходимости, достаточно воспользоваться представлением его в форме числа с плавающей запятой:

$$X = 2^p q, \quad 0,5 \leq q < 1, \quad (9.2)$$

если записать $X = 2^{p-1} Q$, $Q = 2q$, $1 \leq Q < 2$, то

$$1/X = 2^{1-p} 1/Q, \quad 0,5 \leq 1/Q < 1.$$

Таким образом, для вычисления $1/X$ достаточно:

- 1) выделить удвоенную мантиссу (Q) заданного аргумента;
- 2) вычислить CORDIC-алгоритмом величину $1/Q$;
- 3) сформировать порядок искомого числа $1/X$, используя порядок p исходного аргумента.

Функции $\text{EXP}(X)$, $\text{LN}(X)$, \sqrt{X}

Указанные функции вычисляются в гиперболической координатной системе (табл. 9.4).

Таблица 9.4

Режим	Начальные условия	Результат	Область сходимости
ROT	$x_1 = 1/K_n$ $y_1 = 1/K_n$ $z_1 = \varphi$	$x_n = \exp(\varphi)$ $y_n = \exp(\varphi)$	$ \varphi \leq 1,13$
	$x_1 = 1/K_n$ $y_1 = -1/K_n$ $z_1 = \varphi$	$x_n = \exp(-\varphi)$ $y_n = \exp(-\varphi)$	
VEC	$x_1 = \varphi + 1$ $y_1 = \varphi - 1$ $z_1 = 0$	$z_n = (1/2) \ln \varphi$	$0,1 \leq \varphi < 9,58$
	$x_1 = \varphi + 0,25 / K_n^2$ $y_1 = \varphi - 0,25 / K_n^2$ $z_1 = 0$	$x_n = \sqrt{\varphi}$	$0,03 \leq \varphi < 2,42$

Здесь $K_n = \prod_{i=1}^n \sqrt{1-2^{-2i}}$ — коэффициент удлинения вектора ($K_n \approx 0,80$).

Вычисление EXP(X) можно осуществить по формуле:

$$\text{EXP}(X) = \exp(M) \exp(\varphi),$$

где M — целая часть аргумента, а φ — дробная часть, $0 \leq \varphi < 1$.

Таким образом, для вычисления экспоненты достаточно:

- 1) выделить из аргумента целую и дробную части;
- 2) вычислить CORDIC-алгоритмом величину $\exp(\varphi)$;
- 3) умножить результат итерационного процесса на константу $\exp(M)$.

Константы $\exp(M)$ могут храниться в памяти процессора АП-32, общее их количество не превышает 88.

Для вычисления натурального логарифма формула (9.2) предоставляет возможность сужения интервала изменения аргумента до необходимых пределов:

$$\text{LN}(X) = \ln(2^p q) = p \ln 2 + \ln q; \quad 0,5 \leq q < 1.$$

Таким образом, вычисление LN(X) от произвольного аргумента сводится к вычислению CORDIC-алгоритмом логарифма мантиссы аргумента с добавлением константы, которая определяется его

порядком и хранится в памяти АП-32. Количество констант не превышает 64.

Вычисление квадратного корня от аргумента, представленного в виде (9.2), можно осуществить по следующей формуле:

$$\sqrt{X} = \begin{cases} 2^{p/2} \sqrt{q}, & p \text{ — четное,} \\ 2^{(p-1)/2} \sqrt{2q}, & p \text{ — нечетное.} \end{cases}$$

Величины q и $2q$, как видно из табл. 9.4, удовлетворяют условию сходимости и могут служить аргументами для CORDIC-алгоритма. Как и для функции $1/X$, здесь необходимо добавить преобразование порядка аргумента.

Из вышеизложенного следует, что для того, чтобы получить значение искомого элементарных функций, необходимо:

- 1) преобразовать заданный аргумент в аргумент, интервал изменения которого удовлетворяет условию сходимости алгоритма;
- 2) CORDIC-алгоритмом вычислить значение элементарной функции для нового аргумента по соответствующей итерационной схеме и при надлежащих начальных условиях;
- 3) вычислить значение функции для исходного аргумента по результату п.2.

В общем случае пп.1, 3 указанной последовательности действий присущи и любым другим методам аппроксимации (степенными рядами, дробно-рациональными функциями и т. п.) и выполняются программным путем. Аналогично, и в нашем случае аппаратным способом реализуется только п.2, однако часть операций, предусмотренных пп.1, 3, удастся осуществить параллельно с п.2, поскольку они выполняются другими модулями АП-32.

9.2. Структура СПЭФ

Итерационная схема (9.1) CORDIC-алгоритма может быть реализована разными способами. Для достижения максимального быстродействия в представленной версии СПЭФ используется параллельная схема вычислений переменных x_i, y_i, z_i .

Специализированное арифметическое устройство, реализующее параллельную схему вычислений, помимо других необходимых узлов требует, согласно (9.1), наличия трех сумматоров X, Y, Z. При этом в зависимости от вида вычисляемой функции итерационная схема должна быть настроена на соответствующую координатную систему, режим поворота вектора, а тип арифметических опе-

раций на каждой l -й итерации должен определяться текущими знаками переменных x_i, y_i, z_i .

Формализуем алгебраические уравнения (9.1) в логических терминах. Двоичным словом $\langle R \rangle = (R_2, R_1, R_0)$ закодируем тип вычисляемой функции (табл. 9.5).

Таблица 9.5

Код			Координатная система	Режим	Функция
R_2	R_1	R_0			
0	0	0	линейная	VEC	$1/X$
0	0	1	гиперболическая	VEC	$\text{LN}(X), \sqrt{X}$
0	1	0	*	*	
0	1	1	круговая	VEC	$\text{ARCTG}(Y/X)$
1	0	0	*	*	
1	0	1	гиперболическая	ROT	$\text{EXP}(X)$
1	1	0	*	*	
1	1	1	круговая	ROT	$\text{SIN}(X), \text{COS}(X)$

* — не имеет значения.

Здесь разрядом R_2 задается режим поворота вектора, а разрядами R_1, R_0 — тип координатной системы.

Напомним, что при заданном режиме поворота вектора и заданном типе координатной системы конкретный вид вычисляемой функции определяется только начальными условиями (x_1, y_1, z_1) итерационного процесса (см. табл. 9.2, 9.3, 9.4).

Обозначив логическими переменными $Z31, Y31$ текущие знаки (0 — плюс, 1 — минус) переменных z_i, y_i и поставив им в соответствие алгебраические величины $\text{sgn } z_i, \text{sgn } y_i$, введем логическую функцию:

$$W = Z31 \& R_2 + \overline{Y31} \& \overline{R_2}$$

С учетом соотношений между S_y, S_z (см. табл. 9.1) таблицу истинности для W можно представить в виде (табл. 9.6):

Таблица 9.6

Режим	R	$Z31$	$Y31$	W	$S_y = \text{sgn } z_i$	$S_z = -\text{sgn } z_i$
ROT	1	0	*	0	+ 1	- 1
	1	1	*	1	- 1	+ 1
VEC	0	*	0	1	- 1	+ 1
				0	+ 1	- 1
	0	*	1	1	- 1	+ 1
				0	+ 1	- 1

Из табл. 9.7 следует, что вопрос о типе арифметических операций сумматоров Z, Y решается достаточно просто:

Таблица 9.7

W	сумматор Y	сумматор Z
1	вычитание	сложение
0	сложение	вычитание

Поскольку $S_y = -S_z$ (см. табл. 9.1), то, пользуясь табл. 9.5, можно составить следующую таблицу (табл. 9.8) относительно параметра S_x .

Таблица 9.8

Координатная система	Режим	S_x	R_1	R_0
Круговая	ROT	S_z	1	1
	VEC	S_z		
Линейная	ROT	*	0	0
	VEC	*		
Гиперболическая	ROT	$-S_z$	0	1
	VEC	$-S_z$		

Воспользовавшись значениями S_z и W , представленными в табл. 9.6, получаем простой способ управления сумматором X (табл. 9.9).

Таблица 9.9

R_1	R_0	W	S_x	Сумматор X
0	0	*	0	тождество
0	1	0	+1	сложение
0	1	1	-1	вычитание
1	1	0	-1	вычитание
1	1	1	+1	сложение

В линейной координатной системе ($R_1=0, R_0=0$) параметр S_x в (9.1) равен нулю и уравнение для x_{i+1} превращается в тождество.

Таким образом, с помощью трехразрядного управляющего слова $\langle R \rangle$ итерационная структура арифметического устройства может быть настроена на вычисление любой элементарной функции из заданного набора, а логические соотношения между его разрядами и текущими знаками операндов y_i, z_i позволяют достаточно просто управлять сумматорами X, Y, Z на каждом шаге итерационной процедуры решения.

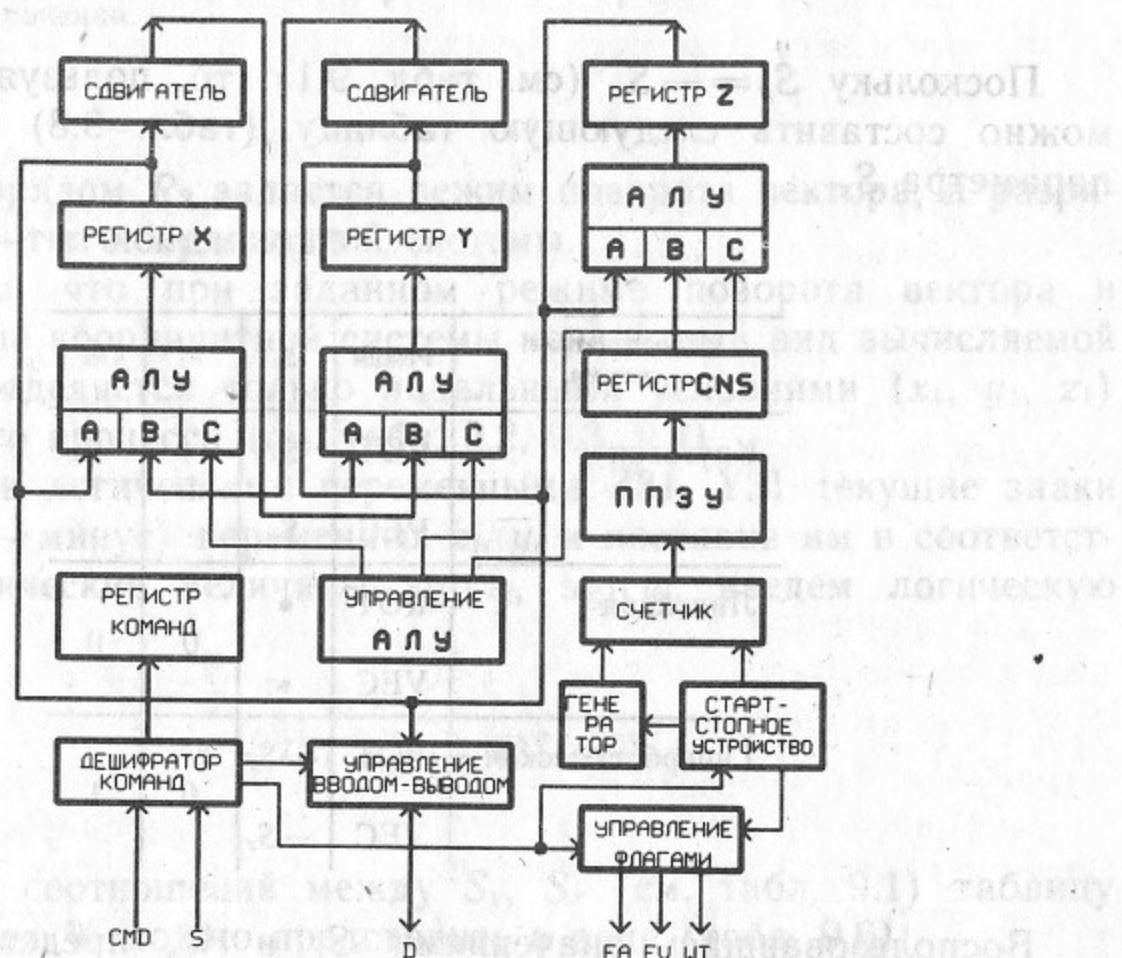


Рис. 9.1. Структурная схема СПЭФ.

В принятом для СПЭФ формате представления данных умножение операнда на 2^{-l} означает арифметический сдвиг его на l позиций вправо. Поэтому структура СПЭФ помимо трех сумматоров согласно (9.1) должна включать в себя и два сдвигателя.

Структурная схема СПЭФ представлена на рис. 9.1. Кроме сумматоров и сдвигателей в нее входят регистры (X, Y, Z) для загрузки начальных условий и записи промежуточных и конечных результатов, ПЗУ для хранения констант C_i и номеров итераций l , старт-стопное устройство с генератором синхросигналов и счетчиком номеров итерационных шагов i , дешифратор команд с регистром команд, узлы управления загрузкой-выгрузкой операндов и флагами.

Чтобы инициализировать работу СПЭФ, необходимо:

- 1) записать управляющее слово $\langle R \rangle$ в регистр команд (код вычисляемой функции);
- 2) последовательно загрузить начальные условия в регистры X, Y, Z .

После загрузки последнего регистра Z запускается счетчик шагов, который вырабатывает инкрементную последовательность двоичных чисел ($i=1, 2, \dots, n$). Вычислительный процесс в модуле организован таким образом, что выборка из ПЗУ констант C_i и номеров итераций l идет параллельно со сдвигом и сложением операндов. Другими словами, за время вычисления текущей итерации происходит выборка константы для следующей. Синхросигнал от внутреннего генератора подается одновременно на регистры X, Y, Z , регистр констант (CNS) и счетчик шагов. При выполнении необходимого количества итераций старт-стопное устройство останавливает вычислительный процесс и устанавливает флаг WT.

Управляющее слово $\langle R \rangle$ и показания счетчика шагов используются также для адресации ПЗУ. Адресное поле ПЗУ (8 адресных входов) поделено так, что на 3 старших входа подается код (R_2, R_1, R_0), а на остальные 5 входов — значение счетчика шагов ($i=1, 2, 3, \dots, n$). Таким образом, вся память разбивается на 8 физических областей (используется 5) по 32 ячейки. В каждой области хранятся соответствующие типу вычисляемой функции 31-разрядные константы C_i и коды номеров итераций l (5 разрядов).

Код l используется как код числа сдвигов операндов x_i, y_i и подается непосредственно на управляющие входы сдвигателей.

9.3. Формат данных, система команд, флаги

CORDIC-алгоритм базируется на использовании итерационного метода «цифра за цифрой», позволяющего последовательно в каждом цикле вычислений получить очередную верную цифру результата. При этом сам алгоритм не накладывает никаких требований на форму представления данных. Однако сложности аппаратной реализации арифметики с плавающей запятой и некоторые отмеченные выше особенности самого алгоритма сделали выбор представления данных в СПЭФ в формате с фиксированной запятой более предпочтительным. Действительно, в этом случае умножение операнда на 2^{-l} сводится к простому сдвигу на l позиций вправо. Кроме того, операция сложения (вычитания) чисел с плавающей запятой занимает значительно большее время, чем чисел с фиксированной запятой.

Применение в СПЭФ БИС К1800ВР8 (16-разрядный каскадируемый сдвигатель) и СИС К500ИП181 (4-разрядное арифметико-логическое устройство) позволило создать 32-разрядные сдвигатели и сумматоры с быстродействием в 20—25 нс. Это в свою очередь позволяет затрачивать на вычисление одной итерации не более 60—70 нс времени или 1,8—2,1 мкс на непосредственное вычисление функции (без формул приведения аргумента). Очевидно, что в настоящее время при современном уровне развития микроэлектроники получить такое быстродействие с помощью арифметики с плавающей запятой не представляется возможным.

Преобразование аргумента в формат СПЭФ учитывается в формулах приведения. Аналогично, преобразование результата вычисления СПЭФ в форму числа с плавающей запятой производится на окончательном этапе определения искомого значения функции. Эти преобразования в АП-32 осуществляются программным способом и состоят из небольшого количества несложных операций.

Разрядность СПЭФ зависит от требуемой точности вычисления функций, которая, в свою очередь, ограничена точностью задания аргумента, начальных данных, инструментальными погрешностями операций, а также количеством итераций. В АП-32 точность задания операнда определяется дробной частью числа — мантиссой (24 разряда при однократной точности). Следовательно, чтобы получить такую же точность вычислений, что и точность начальных данных, необходимо обеспечить в дробной части результата 24 значащих разряда. Численное моделирование СПЭФ показало, что

для этого требуется 30 итераций, включая и те, что исполняются дважды ($l=4, 13$; гиперболическая координатная система), и 27 разрядов дробной части операндов.

Формат чисел, с которым работает СПЭФ, показан на рис. 9.2.

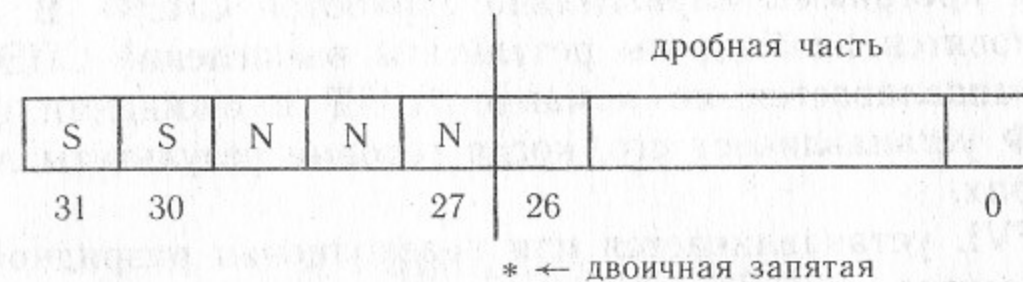


Рис. 9.2. Формат данных в СПЭФ.

Три разряда отводятся для целой части числа. Диапазон чисел, который он обеспечивает, очень мал, однако, как показано выше, метод вычислений и не требует большего. Два старших разряда слова — знаковые, поскольку рабочим кодом чисел является модифицированный дополнительный код.

Как и для других модулей ППЗ, источником команд и данных для СПЭФ служат шины CMD и D (рис. 8.1). Регистры X, Y, Z, имеющие на языке Ассемблера мнемоническое обозначение (VX, VY, VZ), могут быть как регистрами-приемниками, так и регистрами-источниками.

Как и для всех других F-регистров ППЗ, системой команд АП-32 предусматривается обмен данными только между ними и памятью данных, где хранятся начальные данные, некоторые константы и результаты вычислений. Поэтому взаимодействие СПЭФ с программой АП-32 осуществляется по очень простой схеме:

- 1) преобразование исходного аргумента функции в аргумент для CORDIC-алгоритма, запись нового аргумента в память данных;
- 2) инициализация работы СПЭФ (загрузка регистров начальными данными из ОЗУД командами типа LD);
- 3) выгрузка результатов вычисления в ОЗУД (командами типа ST).

Загрузке начальных данных должна предшествовать команда FFG RRR, в одном из полей которой содержится код вычисляемой функции (табл. 9.5). Кроме того, эта команда выполняет и ряд других специфических функций, подготавливающих сопроцессор к работе.

Для сигнализации о своей работе СПЭФ использует три флажка: WT, FVL, FA.

Флаг WT используется для синхронизации работы модуля с ЦП АП-32. После запуска сопроцессора АП-32 может продолжать исполнение программы параллельно с работой СПЭФ. В момент, когда становятся необходимы результаты вычислений СПЭФ, ЦП АП-32 останавливается по команде WAIT в ожидании флажка WT. СПЭФ устанавливает его, когда готовые результаты лежат в его регистрах.

Флаг FVL устанавливается при переполнении разрядной сетки арифметического устройства (возможно только при неверных начальных данных, технической неисправности СПЭФ) или остановке внутреннего генератора синхросигналов.

Флаг FA устанавливается на общую шину флага «Ошибка» при чтении регистров и наличии флага FVL.

Как и другие модули ППЗ, СПЭФ собран на микросхемах ЭСЛ и представляет собой довольно сложное арифметическое устройство. Наличие отечественных 32-разрядных сдвигателей, сумматоров, регистров позволило бы аппаратно реализовать практически все этапы алгоритма на одной плате тех же самых габаритов, что и СПЭФ. При этом было бы реальным получить быстродействие вычисления функции порядка 3—4 мкс при однократной точности и 7—9 мкс — при двукратной.

Для работы с СПЭФ создано программное обеспечение, реализующее все описанные выше методы вычисления элементарных функций. Пакет соответствующих подпрограмм является частью библиотеки ФОРТРАНа.

ЛИТЕРАТУРА

1. Аксенов Г.А. и др. Универсальный арифметический процессор АП-32. 1. Архитектура, система команд, технические характеристики. — Препринт 89-175 ИЯФ СО АН СССР. Новосибирск, 1989.
2. Volder J.E. The cordic trigonometric computing techniqe. — IRE trans. On electronic comp., 1959, v.8, N3.
3. Бахтияров Г.Д., Зиняков Н.Л. Алгоритм Волдера и его применение в вычислительной технике и цифровой обработке сигналов. — Зарубежная радиоэлектроника, № 3, 1983.
4. Байков В.Д., Смолов В.Б. Специализированные процессоры: итерационные алгоритмы и структуры. М.: Радио и связь, 1985.

Г.А. Аксенов, А.В. Кислицын,
В.Я. Сазанский, А.Г. Чертовских

Универсальный арифметический процессор АП-32

4. Арифметика с плавающей запятой, сoproцессор элементарных функций

Ответственный за выпуск С.Г. Попов

— Работа поступила 19.02 1990 г.
Подписано в печать 2.03 1990 г. МН 02135
Формат бумаги 60×90 1/16 Объем 2,5 печ.л., 2,0 уч.-изд.л.
Тираж 150 экз. Бесплатно. Заказ № 26

Набрано в автоматизированной системе на базе фото-
наборного автомата ФА1000 и ЭВМ «Электроника» и
отпечатано на ротапинтере Института ядерной физики
СО АН СССР,
Новосибирск, 630090, пр. академика Лаврентьева, 11.