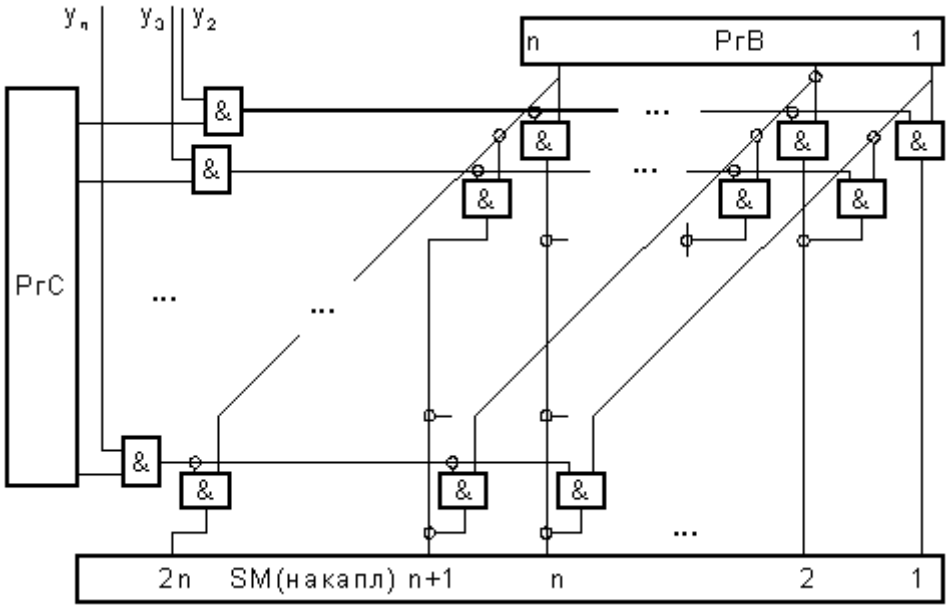
Аппаратные способы ускорения умножения в организации АЛУ.

Рассматриваются только два конкретных решения.

Один из способов сокращения $T_У$ (времени умножения) основан на сокращении или исключении сдвигов. Идея заключается *в создании «функционального пространства сдвигов»*.

Рис. 14. 1.



Из УУ после записи в регистры B и C и обнуления SM (y_1) поступает y_2, открываются вентили & и информация о множимом записывается на триггеры накапливающего сумматора (в n младших разрядов). Затем идет сигнал y_3 и частичные произведения со сдвигом на один разряд влево подается на SM и т.д.

$$Ty \cong nT$$

Еще больше время сократится, если монтажные «ИЛИ» заменить разрядами комбинационного сумматора, так как получится матричная схема умножения.

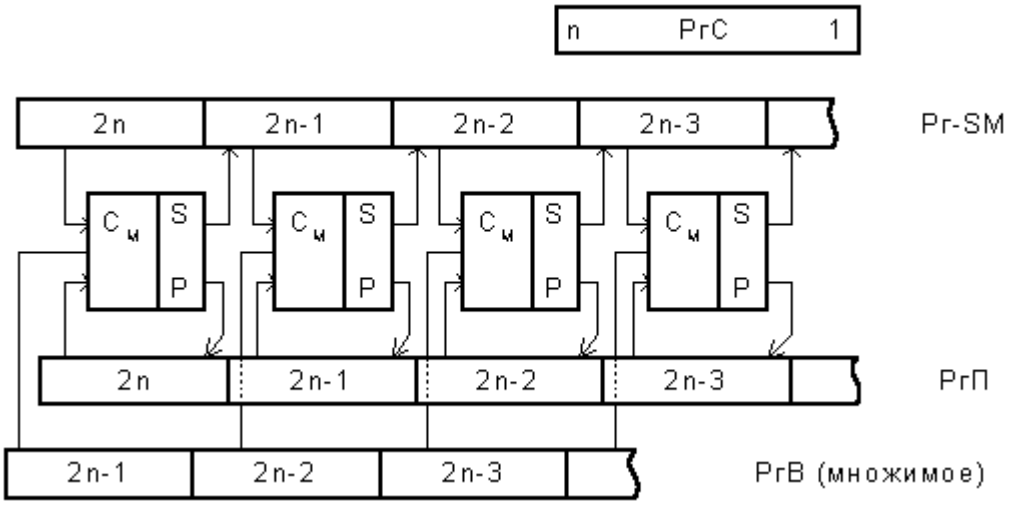
Второй аппаратный способ, позволяющий получить высокое быстродействие по умножению, базируется на схеме с использованием сумматоров с запоминанием переносов.

В этом случае АЛУ имеет дополнительный регистр переноса, в котором запоминается результат переноса, а в PrSM запоминается промежуточная сумма без учета переносов. При поступлении очередного слагаемого (накопление частичного произведения) выполняется суммирование разрядов трех регистров: Pr-SM; регистра переносов и «нового» слагаемого. Перенос же через все разряды реализуется только при определении окончательного результата и может быть выполнен любым способом.

Сложение в данном случае выполняется как поразрядная операция без затрат времени на распространение переносов.

Соответствующая несколько упрощенная функциональная схема АЛУ:

Рис. 14.2.



Очень *существенно*, что в данной схеме не предусмотрены цепи сдвига содержимого SM вправо. Этот сдвиг сумм частичных произведений получают за счет схемы соединений разрядов Рг-SM и См. В заключительном такте производится суммирование содержимого Рг-SM и РгП $_{с}$ распространением волны переносов. В итоге:

$$Ty = n\tau_{S.P.} + T_c,$$

где $\tau_{S.P.}$ – время задержки распространения сигнала переноса в одном разряде + время «сдвига»; T_C – время окончательного суммирования.

В конечном счете выигрыш по быстродействию по сравнению с классической схемой $\approx$ в n/2 раз.

Алгоритмические (логические) способы ускорения умножения в организации АЛУ.

Дополнительные затраты оборудования в этих способах не зависят от разрядности АЛУ, т.к. усложняется, в основном, схема управления АЛУ.

Рассмотрим первую группу способов, позволяющих уменьшить количество суммирований частичных произведений в ходе умножения. Главное за счет чего это можно сделать: *выявить в множителе группы из нескольких единиц (а также нескольких нулей)*. Тогда используется простое соотношение:

$$\sum_{i=k}^{n} 2^i = 2^{n-1} - 2^k,$$

например, $101101 \times 011110 = 101101 \times (2^5 - 2^1) = 101101 \times 2^5 - 101101 \times 2^1$

(здесь используется соотношение: $011110_2 = 2^5 - 2^1 = 30$),

n и k – номера разрядов числа, между которыми во всех разрядах в числе записаны единицы. Но тогда нужно лишь множимое сдвинуть вначале на n-1 разряд, а затем на k разрядов, и вслед за этим вычесть из 1-го числа второе.

Обобщенно это *приводит к* такому *алгоритму*, подлежащему реализации в АЛУ:

Шаг 1. Просматривая множитель, начиная с младшего разряда, сдвигать содержимое сумматора (частичное произведение) вправо до появления первой единицы в множителе.

Шаг 2. На первой единице передать множимое инверсным кодом в сумматор.

Шаг 3. Сдвигать содержимое сумматора синхронно с просмотром разрядов множителя до появления первого «0».

Шаг 4. Передать на первом «0» в сумматор множимое прямым кодом и перейти к первому шагу.

Завершается алгоритм с просмотром последнего (старшего разряда) множителя.

Этот *алгоритм будет плох* лишь *при обработке* комбинаций *с чередующимися* «0» и «1»: …01010101…, так как окажется, что на каждые 2 разряда вместо одного сложения (в обычном алгоритме) нужно выполнять одно сложение и одно вычитание. Поэтому этот алгоритм используется в усовершенствованном виде в *алгоритме Лемана*. Идея этого алгоритма основана на соотношениях:

первое: $+2^{k+1}-2^{k}=+2^{k}$ – можно использовать, когда переходим от ожидаемого «0» к ожидаемой «1».

второе: $-2^{k+1}+2^{k}=-2^{k}$ – можно использовать при переходе от ожидаемой «1» к ожидаемому «0».

Пояснение-иллюстрация:

а) …000100… Если две группы нулей разделены «1» в k-ой позиции, то вместо вычитания в k-ой позиции и сложения в k+1-ой достаточно только выполнить сложение в k-ой позиции.

б) …110111… Если две группы единиц разделены «0» в k-ой позиции, то достаточно выполнить только вычитание в k-ой позиции.

Итак, надо знать:

1) по каким сдвигам-ожиданиям («0» или «1») происходит просмотр множителя;
2) содержимое следующего (соседнего старшего) разряда за анализируемым.

Первое очень просто реализовать с помощью триггера (начальная установка в «0»), а тогда тип передачи множимого на сумматор задается таблицей:

Содержимое триггера режима	Содержимое k+1-го разряда	Содержимое k-го разряда	Тип передачи	Примечание
Движение по «0»	0	1	+В	Изолированная «1»! Режим не меняем.
Движение по «0»	1	1	–В	Меняем режим (состояние Тг реж)
Движение по «1»	1	0	–В	Изолированный «0»! Режим не меняем.
Движение по «1»	0	0	+В	Меняем режим (состояние Тг реж)

Следовательно, достаточно несложного конечного автомата и наличия двух цепей передачи множимого в сумматор.

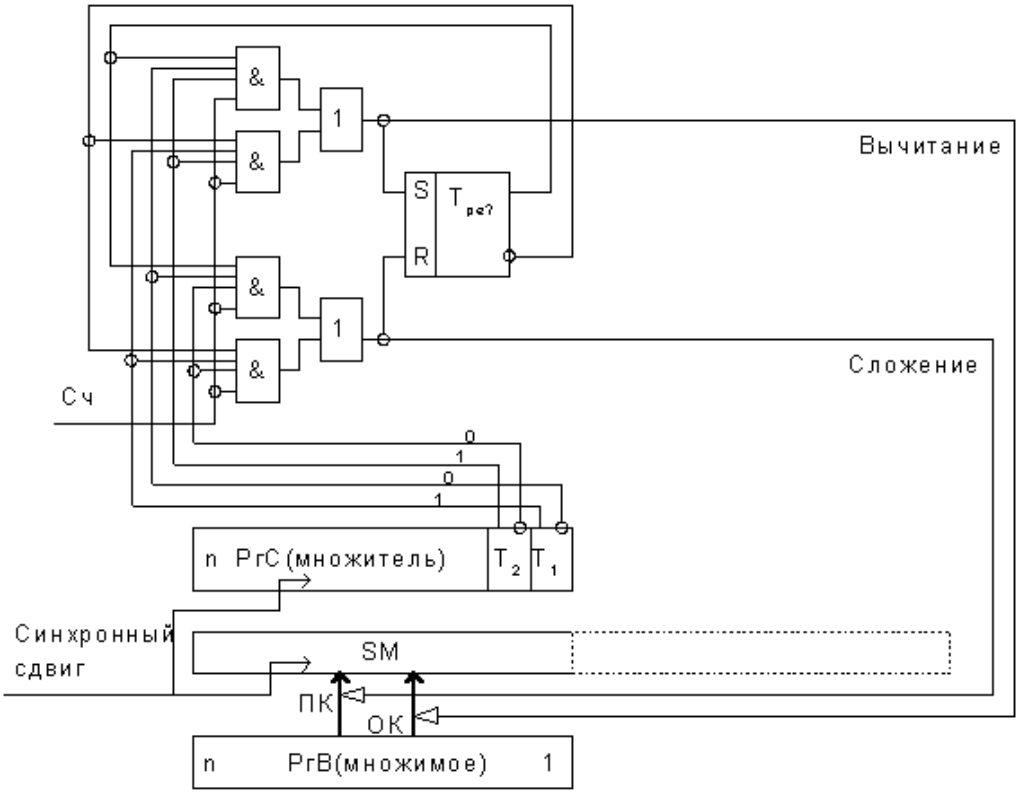


Рис. 14.3.

Перед началом умножения Тг режима «гасится» (устанавливается в ноль сигналом y_0).

Даже в самом неблагоприятном случае число суммирований равно n/2, а в среднем и того меньше – n/3.

Теперь о втором способе: обработке за один шаг нескольких разрядов (обычно двух).

Пусть используем первый алгоритм умножения (см. п.2.2.3.) и рассматриваем сразу два младших разряда сдвигающего регистра, хранящего множитель. Какие возможны варианты?

00) Нужен простой сдвиг частичного произведения на два разряда.

01) К частичному произведению добавляется одинарное множимое и, далее, сдвиг на два разряда.

10) Добавление к частичному произведению удвоенного множимого и, далее, сдвиг на два разряда.

11) Утраивать?? Нет! Из суммы частичных произведений вычитается одинарное множимое. Но далее будет необходима корректировка на следующем шаге. (Можно, конечно, и утроить множимое, но за 2 такта: первый такт по варианту (10) и второй по варианту (01)). И всё же корректировка эффективнее!

Как производится корректировка?

Поскольку одинарное множимое вычитается вместо добавления утроенного, то коррекция результата предполагает далее прибавление учетверенного множимого!

Именно так! Но после сдвига на два разряда вправо сумма частичных произведений, противолежащая множимому, как бы увеличилась в 4 раза. Значит,

для корректировки на следующем этапе достаточно прибавить одинарное множимое.

Проще всего это сделать, преобразуя следующую просматриваемую пару разрядов множителя:

пару 00 - в пару 01
пару 01 - в пару 10
пару 10 - в пару 11
пару 11 - в пару 00, но с учетом дополнительной единицы для следующей пары.

Хранится дополнительная единица в специальном триггере. Либо она учитывается схемотехнически.

Два слова об удвоении множимого. Его можно реализовать:
а) сдвигом влево в Рг множимого и
б) переключением разрядов на другие входы сумматоров (схемный сдвиг).

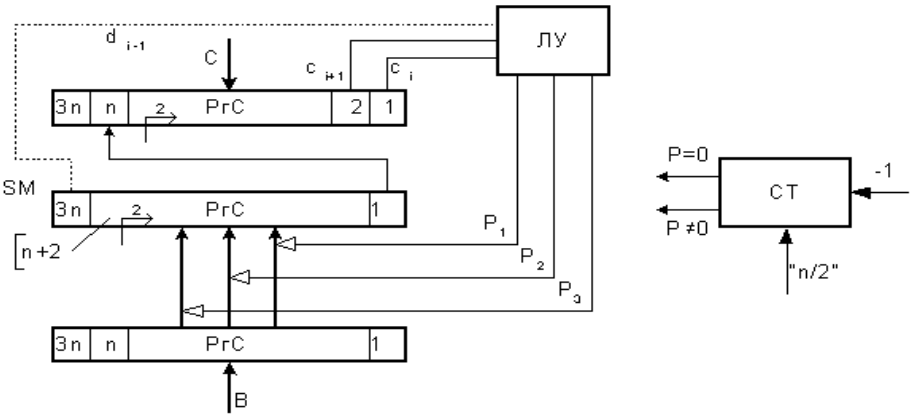


Рис. 14.4.

Пунктиром на рисунке показано, что вместо специального триггера можно использовать в качестве d_{i-1} знаковый разряд, но при этом

1) должен быть использован (при вычитании) дополнительный код;
2) SM должен иметь цепь арифметического сдвига вправо на два разряда (с размножением «1» в знаковом разряде)

$$SM := SM + 111. < РгВ >_{доп} \quad - \quad \text{при } p_3;$$

3) передача из SM в РгС младших разрядов происходит в прямом коде, либо нужен достаточно сумматор.

$$T_{умн} = \frac{n}{2}\left(\frac{3}{4}T_c + t_{сдв}\right); \qquad T_{умн} \approx \frac{7}{8}nT$$