УДК. 681.3

А.И. Корнилов, М.Ю. Семенов,
О.В. Ласточкин, В.С. Калашников

# Применение современных методов проектирования при реализации модулярных вычислительных процедур

*(Институт проблем проектирования в микроэлектронике РАН)*

Показано применение современных методов проектирования и логического синтеза при построении эффективных, с точки зрения быстродействия и занимаемой площади, модулярных вычислительных блоков для отдельных значений модулей.

К цифровым устройствам постоянно предъявляются требования повышения быстродействия. Применение аппарата модулярной арифметики при построении специализированных вычислителей позволяет повысить производительность таких систем за счет естественного распараллеливания трактов обработки данных без какого-либо изменения существующих технологий. В 60-70-е годы были проведены значительные теоретические исследования в области модулярной арифметики (в том числе и в России) и реализован ряд высокоэффективных вычислительных систем на ее основе. Однако данное направление не получило дальнейшего широкого развития во многом из-за проблем в реализации этих устройств, связанных с элементной базой, принципиально ориентированной на двоичную

булеву арифметику. В настоящее время с развитием интегральной схемотехники появляются возможности по использованию современных методов проектирования при реализации модулярных вычислительных процедур. В свою очередь, эффективная реализация отдельных модулярных вычислительных блоков приводит к повышению эффективности всего устройства в целом.

Рассмотрим более подробно, какие методы проектирования могут быть использованы при реализации модулярных вычислительных процедур.

Методы логического синтеза быстродействующих модулярных сумматоров на основе BDD-технологии.

Одним из возможных методов логического синтеза быстродействующих цифровых схем является использование представления логических функций на основе Диаграмм Двоичных Решений (ДДР) или в англоязычной терминологии - Binary Decision Diagram (BDD). Применение диаграмм двоичных решений может быть продемонстрировано при построении быстродействующих сумматоров по модулям вида ($2^n$-1) и ($2^n$+1). Модули данного типа находят широкое применение при построении систем с использованием аппарата модулярной арифметики.

*Построение быстродействующих сумматоров по модулю вида ($2^n$-1) на основе BDD.*

Сумматоры по модулям вида ($2^n$-1) могут быть построены на основе сумматоров с циклическим переносом из высшего в низший разряд. Это связано с тем, что суммирование по модулю ($2^n$-1) может быть преобразовано в суммирование по модулю $2^n$ в соответствии со следующей формулой:

$$|a+b|_{2^n-1} = |a+b+c_{out}|_{2^n} \qquad (1)$$

В классической BDD каждое звено диаграммы соответствует разложению Шеннона, а в аппаратном исполнении реализации мультиплексора. Однако для традиционных КМОП схем целесообразно использовать монотонный базис, применяя известные булевы функции генерации ($g_i$) и распространения переноса ($p_i$):

$$p_i = a_i \vee b_i \qquad (2)$$

$$g_i = a_i \wedge b_i \qquad (3)$$

$$c_{i+1} = g_i \vee p_i c_i \qquad (4)$$

В таких сумматорах быстродействие определяет критический путь при вычислении функции выходного переноса. Построим функциональную диаграмму для системы функций переноса сумматоров по модулю ($2^n$-1) в соответствии с формулой (1), основываясь на функциональной диаграмме обычного двоичного сумматора в соответствии с формулами (2) – (4). В этом случае каждое звено диаграммы соответствует аппаратной реализации КМОП-элемента 2И-ИЛИ. Для обозначения таких диаграмм, которые, фактически, являются одной из разновидностей BDD диаграмм, будем использовать название FDD - Functional Decision Diagram.

На рис. 1(a) приведена FDD для n-разрядного двоичного сумматора с учетом формул (2)-(4). Для сумматоров по модулю ($2^n$-1), в которых выходной перенос равен входному (см. формулу (1)), FDD принимает вид, показанный на рис. 1(b).

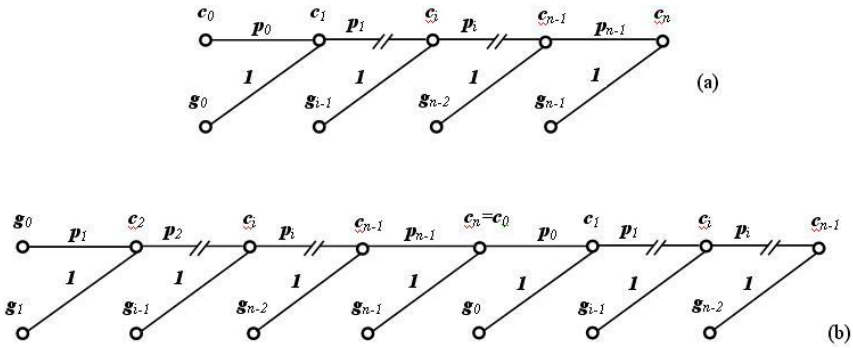


Рис. 1. FDD для n-разрядного двоичного сумматора (a) и преобразованная FDD для сумматора по модулю ($2^n$-1) (b).

Применим к полученной диаграмме для сумматоров по модулю ($2^n$-1) алгоритм декомпозиции, описанный в работах [1], [2]. Правило преобразования заключается в том, что длинная структура FDD может быть сведена обрывом ребра к двум диаграммам, при этом эти декомпозированные части реализуются параллельно (примени-

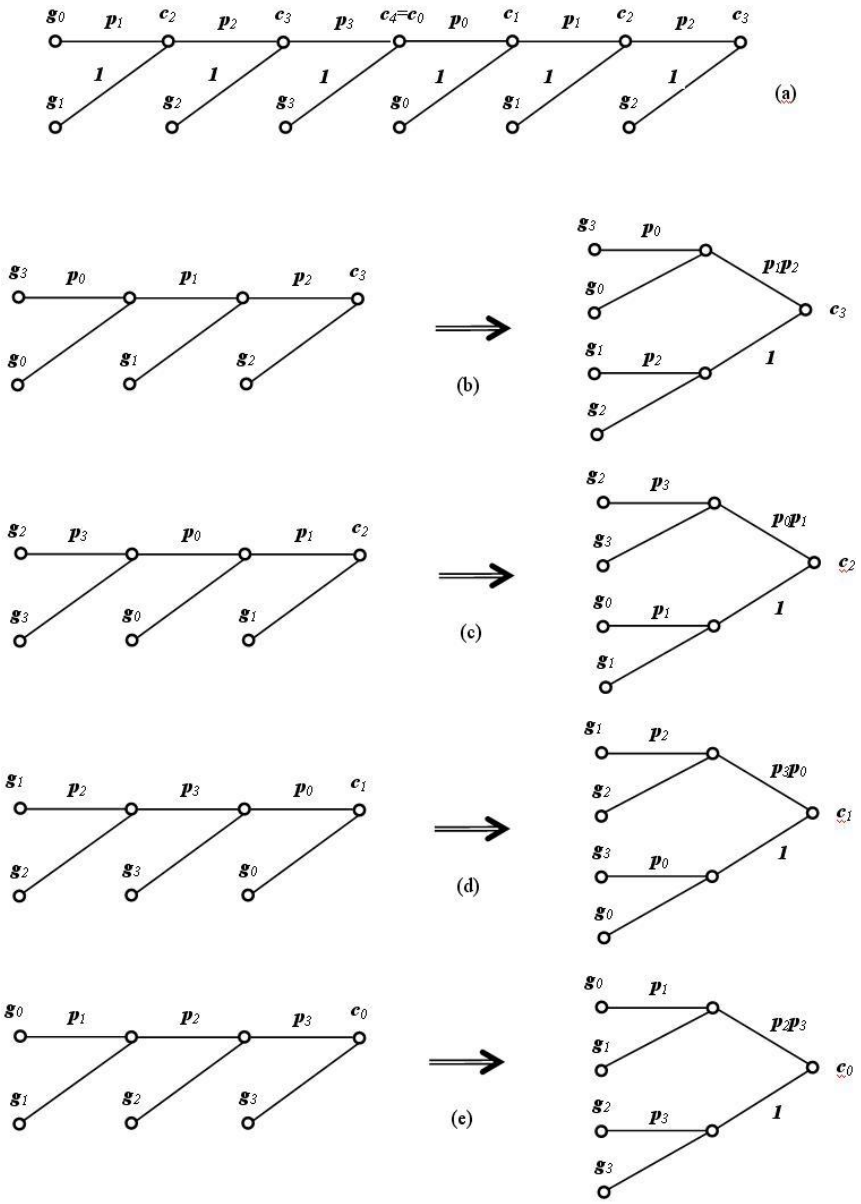


Рис.2. FDD для сумматора по модулю 15 и этапы декомпозиции: (a) – исходная FDD,
(b) – перенос $c_3$, (c) – перенос $c_2$, (d) - перенос $c_1$, (e) - перенос $c_0$.

тельно к функциям переноса сумматора по модулю ($2^n$-1) это соответствует левой и правой частям FDD), а также вводится дополнительная логика. Параллельная реализация обеспечивает более быстродействующую структуру с некоторым увеличением аппаратных затрат.

Таким образом, исходя из общей диаграммы, изображенной на рис. 1(b), возможно построить FDD для каждой из функций переноса $c_i$ для сумматора по модулю ($2^n$-1). При этом необходимо руководствоваться основным принципом, согласно которому, при построении диаграммы для переноса $c_i$ разрез должен осуществляться по ребру $p_i$ [3]. Данное правило является определяющим при построении FDD для функций переноса сумматоров по модулю ($2^n$-1).

В качестве примера, на рис. 2(a) приведена общая FDD для сумматора по модулю m=15. Диаграммы для функций переноса $c_3$, $c_2$, $c_1$ и $c_0$ формируются разрывом соответствующих ребер $p_3$, $p_2$, $p_1$ и $p_0$. На рис. 2(b), 2(c), 2(d), 2(e) показаны FDD для каждого из переносов и полученные из них декомпозированные диаграммы.

Схемная реализация для переноса $c_3$, соответствующая построенной оптимизированной FDD, приведена на рис. 3. При этом каждое звено диаграммы соответствует реализации на основе элемента 2И-ИЛИ (рис. 3(a)). Для КМОП-базиса, используя правила де Моргана, можно легко получить схему с использованием элементов 2И-ИЛИ-НЕ/2ИЛИ-И-НЕ(см. рис. 3(b)).Аналогичным образом, можно получить схемные реализации для других функций переноса.

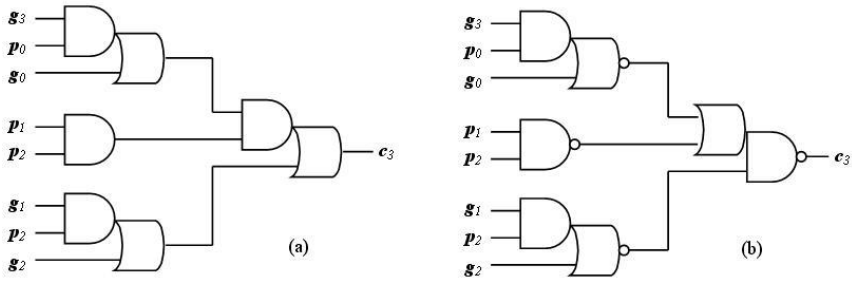


Рис.3. Аппаратная реализация функции переноса:
• $c_3$ в базисе элементов 2И-ИЛИ (a),
• в базисе элементов 2И-ИЛИ-НЕ/2ИЛИ-И-НЕ (b).

Следует отметить, что для полной реализации модулярного сумматора необходимо введение дополнительной логики, обеспечивающей единственное представление нуля. Эта логика практически не приводит к увеличению задержки на критическом пути, так как реализуется параллельно с основной структурой.

Таким образом, сумматор по модулю ($2^n$-1) включает следующие основные стадии:

- блок, реализующий функции генерации ($g_i$) и распространения переноса ($p_i$) в соответствии с формулами (2) и (3);

- блок, реализующий функции переноса $c_i$ и построенный на основе декомпозированных FDD;

- блок, вычисляющий финальное значение модулярной суммы ($S_i = a_i \wedge b_i \wedge c_i$).

Представленная структура означает, что сумматор по модулю ($2^n$-1) не уступает по быстродействию двоичным сумматорам такой же разрядности.

*Построение быстродействующих сумматоров по модулю вида ($2^n$+1) на основе BDD.*

Аналогичным образом могут быть построены сумматоры по модулям вида ($2^n$+1). Суммирование по модулю ($2^n$+1) преобразовывается в суммирование по модулю $2^n$ следующим образом [4]:

$$|a+b|_{2^n+1} = |a+b+z|_{2^n} + \bar{c}_{out}, \qquad (5)$$

где $z = 2^{n+1} - (2^n + 1) = 2^n - 1$ – дополнение значения модуля до $2^{n+1}$, а $c_{out}$ - старший бит суммы $(a+b+z)$. Из соотношения (5) следует, что для определения суммы $|a+b|_{2^n+1}$ необходимо сначала вычислить сумму $(a+b+z)$, а затем прибавить к ней инверсию старшего бита, задействовав при этом, лишь младшие $n$ бит (т.е. взяв ее по модулю $2^n$). Следует отметить, что старший бит результата определяется специальным образом для обеспечения единственности представления нуля.

Добавление константы $z$ целесообразно реализовывать с помощью сумматора с запоминанием переносов (CSA-carry-save adder), который сводит сложение трех чисел к сложению двух за время за-

держки полного одноразрядного сумматора [5], [6]. Поскольку данная константа известна заранее для каждого конкретного значения модуля и имеет вид $2^n$-1, то одноразрядные сумматоры могут быть заменены на полусумматоры (half-adder, HA), из которых $n$ имеют в качестве входного переноса “1” а один является обычным полусумматором (рис. 4).

Следующим шагом необходимо сложить полученные два числа $u$ и $v$, определив тем самым значение $\bar{c}_{out}$. Для этого снова воспользуемся известными булевыми функциями генерации и распространения переноса (2) и (3).

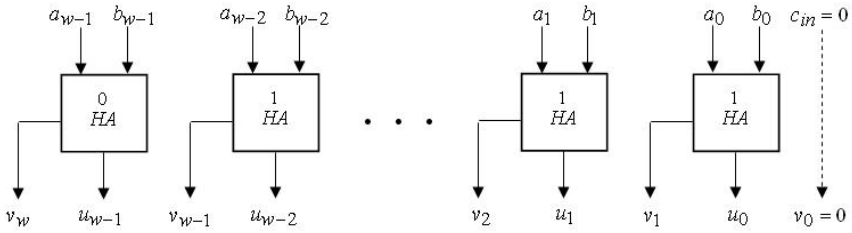


Рис.4. Блок добавления константы z в сумматорах по модулю ($2^n$+1).

Как и в сумматорах по модулю вида ($2^n$-1), воспользуемся диаграммами двоичных решений для вычисления $c_{out}$. Поскольку теперь, в качестве входного переноса необходимо использовать инверсию старшего бита $\bar{c}_{out}$, то реализация циклического переноса на основе BDD в данном случае представляется невозможной, вследствие немонотонного базиса. Поэтому для вычисления $c_{out}$ используется обычная FDD для двоичного сумматора на рис. 1(а). Очевидно, что инверсия $\bar{c}_{out}$ может быть легко получена выбором соответствующих КМОП-элементов при реализации данного блока.

Вычислив $\bar{c}_{out}$, необходимо определить сигналы переноса из младших разрядов с учетом $c_{in} = \bar{c}_{out}$. Для этого достаточно найти их значения исходя из условия $c_{in} = 0$:

$$c_1' = g_0 = 0,$$

$$c'_2 = g_1,$$

$$c'_3 = g_2 \vee (p_2 \wedge c_2) = g_2 \vee (p_2 \wedge g_1),$$

…

$$c'_n = g_{n-1} \vee (p_{n-1} \wedge c_{n-1}) = g_{n-1} \vee (p_{n-1} \wedge g_{n-2}) \vee (p_{n-1} \wedge p_{n-2} \wedge g_{n-3}) \vee \mathbf{K} \vee (p_{n-1} \wedge p_{n-2} \wedge \mathbf{K} \wedge p_2 \wedge g_1),$$

а затем скорректировать эти промежуточные переносы следующим образом:

$$c_1 = c'_1 \vee (p_0 \wedge \bar{c}_{out}) = p_0 \wedge \bar{c}_{out},$$

$$c_2 = c'_2 \vee ((p_1 \wedge p_0) \wedge \bar{c}_{out}) = c'_2 \vee (p_{10} \wedge \bar{c}_{out}),$$

$$c_3 = c'_3 \vee ((p_2 \wedge p_1 \wedge p_0) \wedge \bar{c}_{out}) = c'_3 \vee (p_{210} \wedge \bar{c}_{out}),$$

…

$$c_n = c'_n \vee ((p_{n-1} \wedge p_{n-2} \wedge \mathbf{K} \wedge p_0) \wedge \bar{c}_{out}) = c'_n \vee (p_{n-1\mathbf{K}0} \wedge \bar{c}_{out}),$$

где значения $c'_1, \mathbf{K}, c'_n$ и соответствующие конъюнкции $p_{10}, p_{210}, \ldots, p_{n-1\mathbf{K}0}$ вычисляются параллельно с $\bar{c}_{out}$ на основе той же самой FDD и не вносят дополнительную задержку. Таким образом, вычисление сигналов переноса при условии $c_{in} = \bar{c}_{out}$ реализуется за одну стадию с помощью КМОП-элемента 2И-ИЛИ, как показано на рис. 5.

Стоит также отметить, что как и в сумматоре по модулю вида ($2^n$-1) в данном случае требуется введение дополнительной логики для вычисления старшего бита результата и обеспечения единственного представления нуля, которая также функционирует параллельно с основной структурой и не вносит дополнительной задержки.

Таким образом, сумматор по модулю ($2^n$+1) включает следующие основные стадии:

- блок, реализующий добавление константы $z$ к сумме входных операндов на основе CSA;

- блок, реализующий функции генерации ($g_i$) и распространения переноса ($p_i$) в соответствии с формулами (2) и (3);

- блок, реализующий функции переноса $c'_i$ и $\bar{c}_{out}$ (при условии

$c_{in} = 0$ ) и построенный на основе декомпозированных FDD;

- блок, реализующий коррекцию сигналов переноса с учетом $c_{in} = \overline{c}_{out}$ на основе КМОП-элемента 2И-ИЛИ;

- блок, вычисляющий финальное значение суммы ($S_i = a_i \wedge b_i \wedge c_i$).

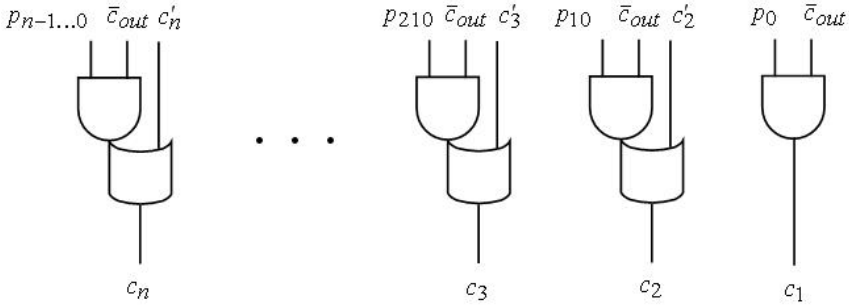


Рис.5. Блок вычисления переносов с учетом $c_{in} = \overline{c}_{out}$.

Представленная структура означает, что задержка сумматора по модулю ($2^n$+1) отличается от задержки обычных двоичных сумматоров такой же разрядности лишь на сумму задержек одного полусумматора и КМОП-элемента 2И-ИЛИ.

Методы построения быстродействующих модулярных умножителей на основе алгоритма Бута.

Существуют различные методы построения модулярных умножителей. Так, например, одним из распространенных методов является метод индексного (или дискретно-логарифмического) модулярного умножения [7]. Данный подход основан на преобразовании операндов в соответствующие индексы и заменой операции модулярного умножения на операцию модулярного сложения над полученными индексами. К недостаткам индексных умножителей можно отнести то, что они могут быть построены только для модулей, являющихся простыми числами, а также значительное возрастание общей площади умножителя при увеличении значения модуля [8].

*Построение быстродействующих умножителей по модулю вида ($2^n$-1)*

Для отдельных значений модулей при построении быстрых умножителей могут быть эффективно использованы такие же принципы

реализации, что и для быстродействующих двоичных умножителей, а именно, модифицированный алгоритм Бута [9], [10] и дерево Уоллеса [6].

В общем случае умножитель Бута состоит из селектора, генератора частичных произведений (декодер Бута), массива сумматоров для сложения частичных произведений. С помощью селектора осуществляется разложение множителя на отдельные группы бит (например, для разрежения спектра степеней в сумме частичных произведений через разряд используется группировка по три бита). Декодер Бута, в зависимости от значений полученных после селектора, осуществляет необходимые преобразования над вторым операндом. Массив сумматоров целесообразно реализовывать на основе дерева Уоллеса с финальным сумматором, построенным на основе алгоритмов быстрого сложения (например, на основе BDD-технологии). Обобщенная структура умножителя Бута представлена на рис. 6.

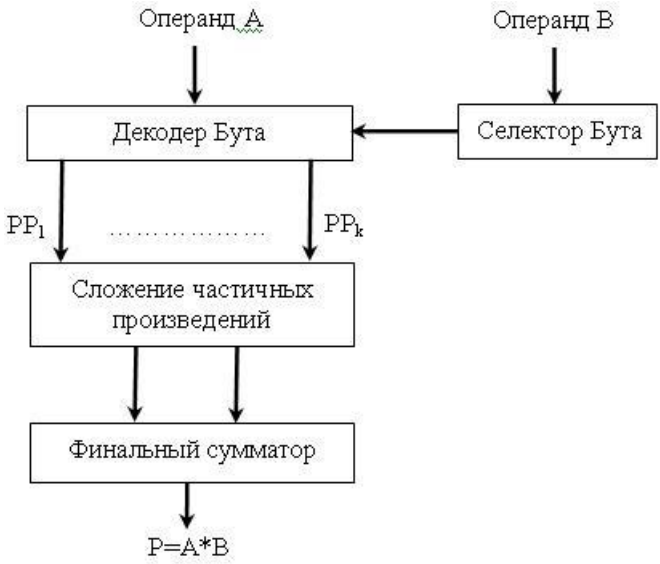


Рис.6. Обобщенная структурная схема умножителя на основе алгоритма Бута.

Особенности операций по модулю вида ($2^n$-1) позволяют эффективно реализовывать каждый элемент умножителя. В этом случае, частичные произведения $PP_1, \mathbf{K}, PP_k$ и конечное произведение $P$ будут вычислены, соответственно, по модулю ($2^n$-1). Селектор для модулярного умножителя строится таким же образом, как и селектор для обычного позиционного умножителя.

*1. Реализация декодера Бута в умножителях по модулю ($2^n$-1)*

Декодер Бута формирует значения:

$$\left\{A\big|_{2^n-1}, |2\cdot A\big|_{2^n-1}, |-2\cdot A\big|_{2^n-1}, |-A\big|_{2^n-1}, 0\right\} \quad (6)$$

при условии, что анализ операнда B выполняется на основе триад бит.

Покажем, что получить значения частичных произведений $\left\{|2\cdot A\big|_{2^n-1}, |-2\cdot A\big|_{2^n-1}, |-A\big|_{2^n-1}\right\}$ для декодера Бута можно комбинируя операцию инверсии каждого бита операнда A и/или его циклический сдвиг влево на соответствующее количество бит.

Рассмотрим генерацию частичных произведений в (6) более подробно. Пусть есть число $A = \sum_{i=0}^{n-1} a_i 2^i$, лежащее в диапазоне $\left[0, 2^n - 1\right)$. Выражение соответствующее умножению числа A на число 2, можно записать в следующем виде:

$$A\cdot 2 = \left(\sum_{i=0}^{n-1} a_i 2^i\right) 2 = \sum_{i=0}^{n-1} a_i 2^{i+1} = a_{n-1} 2^n + \sum_{i=0}^{n-2} a_i 2^{i+1} \quad (7)$$

Из формулы (7) можно найти значение выражения $|A\cdot 2|_{2^n-1}$:

$$|A\cdot 2|_{2^n-1} = \left|a_{n-1}2^n + \sum_{i=0}^{n-2} a_i 2^{i+1}\right|_{2^n-1} = \left|\left|a_{n-1}2^n\right|_{2^n-1} + \left|\sum_{i=0}^{n-2} a_i 2^{i+1}\right|_{2^n-1}\right|_{2^n-1} \quad (8)$$

Вычислим значение каждого слагаемого формулы (8), принимая во

внимание, что $\left|2^n\right|_{2^n-1} = 2^0$, а $\left|a_{n-1}\right|_{2^n-1} = a_{n-1}$ :

$$\left|a_{n-1}2^n\right|_{2^n-1} = \left|a_{n-1}\right|_{2^n-1} \cdot \left|2^n\right|_{2^n-1}\Big|_{2^n-1} = a_{n-1}2^0 \qquad (9)$$

$$\left|\sum_{i=0}^{n-2} a_i 2^{i+1}\right|_{2^n-1} = \sum_{i=0}^{n-2} a_i 2^{i+1} \qquad (10)$$

На основании полученных формул (9) и (10), умножение некоторого числа A на 2 по модулю ($2^n$-1) соответствует циклическому сдвигу влево на один бит.

Используя свойство модулярной арифметики для отрицательных чисел [6]: $\left|-Y\right|_X = \left|X-Y\right|_X$, можно показать, что значение выражения $\left|-A\right|_{2^n-1}$ соответствует инверсии всех битов исходного числа, т.е.

$$\left|-A\right|_{2^n-1} = \overline{a_{n-1}}\,\overline{a_{n-2}}\,\mathbf{K}\,\overline{a_0}$$

Выражение $\left|-A\cdot 2\right|_{2^n-1}$ соответствует циклическому сдвигу влево исходного слова, с инверсией каждого бита которого:

$$\left|-A\cdot 2\right|_{2^n-1} = \overline{a_{n-2}}\,\overline{a_{n-3}}\,\mathbf{K}\,\overline{a_0}\,\overline{a_{n-1}}$$

Если значение A равно нулю, тогда $\left|A\right|_{2^n-1} = 0 = 000\mathbf{K}00$.
*n бит*

*2. Реализация блока суммирования частичных произведений (мультиоперандный сумматор по модулю вида ($2^n$-1) )*

При анализе возможных методов реализации данного блока важно отметить, что частичные произведения, полученные на выходах декодера, имеют одинаковую разрядность. Этот факт позволяет для сложения частичных произведений эффективно использовать дерево Уоллеса. Как известно, дерево Уоллеса для суммирования чисел в двоичной системе счисления строится на основе сумматоров с запоминанием переносов (CSA). Учитывая свойство (1), аналогичным образом может быть построен сумматор с запоминанием пе-

реносов для модулей вида $2^n - 1$. Для этого достаточно использовать в качестве входного переноса следующего каскада выходной разряд переноса сумматора с запоминанием переносов, как показано на рис. 7. Очевидно, что задержка такого сумматора, как и его двоичного варианта, определяется максимальной задержкой одноразрядного полного сумматора (full-adder, FA), которую мы обозначим через $t_{з\max}^{FA}$.

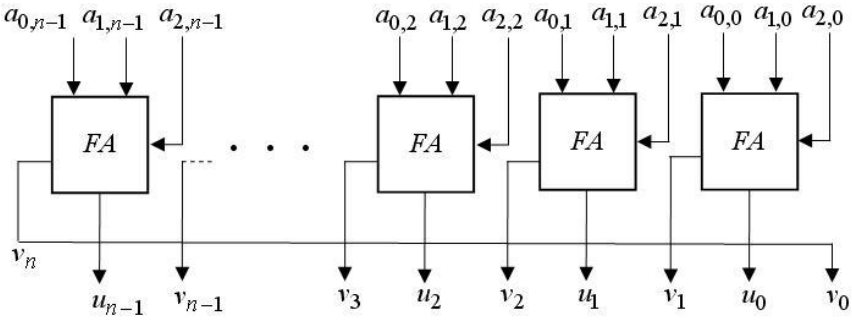


Рис. 7. Сумматор с запоминанием переносов для модулей вида $2^n - 1$.

На выходе такого сумматора формируются два числа одинаковой разрядности, что дает возможность построить регулярную структуру дерева Уоллеса, поскольку все входные и выходные операнды на каждом из каскадов дерева имеют одинаковую разрядность (рис. 8). Быстродействие данной структуры блока суммирования частичных произведений зависит только от количества операндов и не зависит от их разрядности, а значит и от значения модуля. Поскольку глубина дерева или количество этапов суммирования определяется как $\Theta(\log_2 k)$, а задержка любого из уровней структуры равна $t_{з\max}^{FA}$, то общую оценку быстродействия дерева Уоллеса для модулей вида $2^n - 1$ можно представить в виде $\Theta(\log_2 k) \cdot t_{з\max}^{FA}$, где k – количество входных операндов, $t_{з\max}^{FA}$ - задержка одноразрядного полного сумматора.

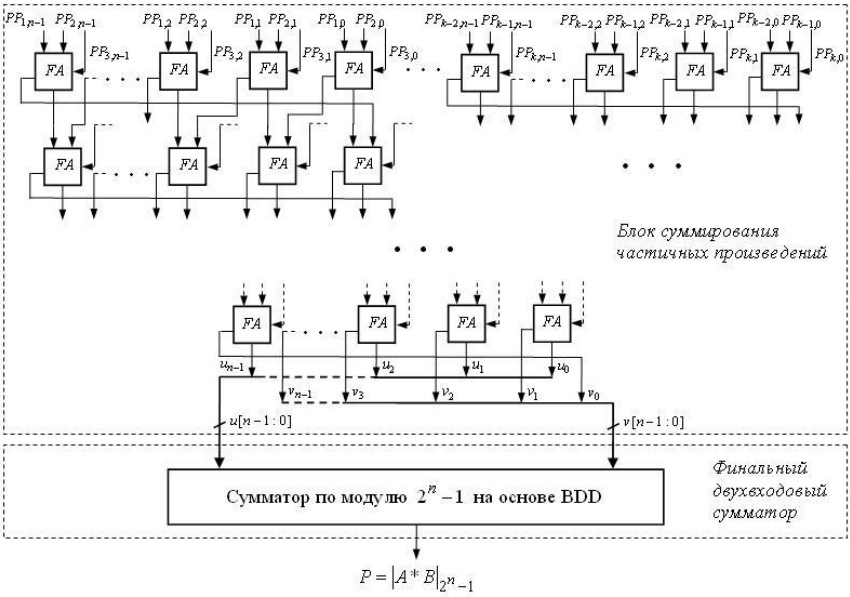


Рис. 8. Реализация блока суммирования частичных произведений (мультиоперандного сумматора по модулю вида $2^n-1$) на основе дерева Уоллеса.

Таким образом, преодолеваются основные недостатки дерева Уоллеса: нерегулярность структуры и резкое возрастание площади при увеличении количества операндов, которые необходимо сложить. Финальный сумматор может быть построен на основе BDD-технологии, как это показано ранее в данной работе.

Из проведенного анализа можно сделать вывод, что умножители по модулю вида $2^n-1$ могут быть построены также эффективно с точки зрения быстродействия и занимаемой площади, как и обычные двоичные умножители такой же разрядности.

Следует отметить, что принципы построения быстрых умножителей на основе алгоритма Бута также применимы при реализации умножителя по модулю вида $2^n+1$.

## Литература


1. Kornilov A., Isaeva T., Syngaevsky V. Carry Circuit Depth Optimization by BDD Based Decomposition // Proc. of PATMOS'97 Workshop – Louvain-la-Neuve, Belgium, Sep. 8-10. – 1997.-P.89-98.

2. Исаева Т.Ю., Корнилов А.И. Алгоритм декомпозиции логических функций, ориентированный на синтез быстродействующих цифровых устройств// Информационные технологии. - №4, 2001. - С.26-31.

3. Корнилов А.И., Исаева Т.Ю., Семенов М.Ю. Методы логического синтеза сумматоров с ускоренным переносом по модулю ($2^n$-1) на основе BDD-технологии // Известия ВУЗов. Электроника. – 2004. - №3. – С. 54-60.

4. A. A. Hiasat. High-Speed and Reduced-Area Modular Adder Structures for RNS // IEEE Transactions on Computers, vol. 51, no. 1, January 2002.

5. T. Kim, W. Jao, S. Tjiang. Circuit Optimization Using Carry-Save-Adder Cells // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 17, no. 10, October 1998.

6. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. - М: МЦНМО, 2001. – 960с.

7. D. Radhakrishnan, Yong Yuan. Novel Approaches to the Design of VLSI RNS Multipliers // IEEE Transactions on Circuits and Systems – II: Analog and Digital Signal Processing, Vol. 39. No 1. January 1992. P.: 52-57.

8. Корнилов А.И., Семенов М.Ю., Ласточкин О.В. Принципы построения модулярных индексных умножителей // Известия ВУЗов. Электроника. – 2004. - №2. – С. 48-55.

9. Угрюмов Е. Цифровая схемотехника – СПБ.: БХВ-Петербург, 2001. –528с.

10. H.I. Saleh, A.H. Khalil, M.A. Ashour, A.E. Salama. Novel serial-parallel multipliers // IEEE Proc.-Circuits Devices Systems, Vol. 148, No. 4, August 2001, P.: 183-189.