БАЗЫ ДАННЫХ
DATABASE

УДК 004.02: 004.312

В. Н. Жураковский, канд. техн. наук, доц.,
С. И. Силин, канд. техн. наук, ассистент, e-mail: sm2-2@inbox.ru,
Московский государственный технический университет имени Н. Э. Баумана

Выбор алгоритмов обработки данных в современной вычислительной технике на основе системного анализа

Рассмотрены проблемы выбора алгоритмической реализации параллельной обработки данных на базе ПЛИС. Рассмотрены различные параметры, которые влияют на возможность реализации алгоритма в ПЛИС. Показано, что исследование и оптимизацию процесса разработки целесообразно проводить с помощью методики и средств системного анализа. Проведена формализация критериев выбора формы реализации на основе приведенных показателей.

Ключевые слова: *ПЛИС, системный анализ, алгоритм, параллельная обработка сигналов, радиолокация, встроенные системы, системы на кристаллах, микроэлектроника, схемотехника, системотехника, проектирование систем*

Введение

К бортовым встраиваемым системам предъявляется ряд требований по массогабаритным характеристикам, энергопотреблению, стоимости, удобству настройки, снижению зависимости параметров алгоритмов от характеристик используемой элементной базы. В связи с этим в последнее время стало широко распространено применение цифровых программируемых микросхем, таких как цифровые сигнальные процессоры (ЦСП) и программируемые логические интегральные схемы (ПЛИС).

В данной статье речь пойдет о проблемах, возникающих при использовании ПЛИС [1]. Эти микросхемы применяются все более и более часто в большом числе приложений. Подобный рост популярности связан с их высоким быстродействием, массовым параллелизмом, простотой отработки проектов, возможностью изменения и настройки параметров алгоритмов в зависимости от конкретных условий их применения. Также в последнее время появилась технология, позволяющая переносить проекты, разработанные для ПЛИС, в заказные интегральные схемы с жесткой логикой без изменения самого проекта, что позволяет упростить и удешевить процесс запуска серийного производства изделий.

Особенности реализации параллельных алгоритмов

Несмотря на все достоинства разработка программно-математического обеспечения для ПЛИС также имеет ряд трудностей, с которыми приходится столкнуться разработчику. Каждая из микросхем обладает определенными характеристиками по производительности (максимальной тактовой частоте) и числу доступных для использования ячеек (объему).

Сами ячейки являются разнородными по типу и назначению [2, 3]:

- специализированные входные элементы (рис. 1, позиция *а*);
- комбинационные логические блоки и регистры, в некоторых ПЛИС объединенные в логические модули (рис. 1, позиция *б*);
- оперативная память (рис. 1, позиция *в*);
- блоки цифровой обработки сигнала (ЦОС, рис. 1, позиция *г*);
- модули фазовой автоподстройки частоты (ФАПЧ); и др.

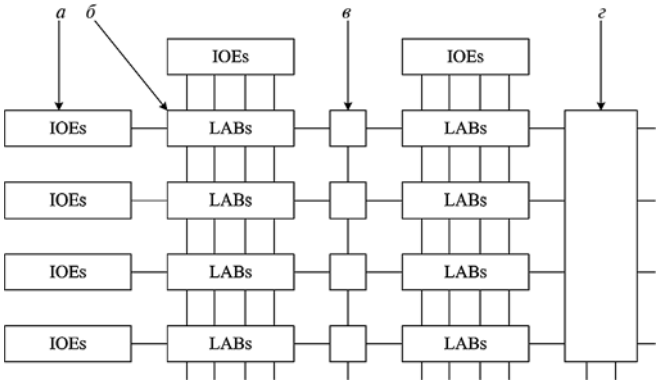

Рис. 1. Типовая архитектура ПЛИС на примере ПЛИС Stratix II компании Altera

Эти ячейки не являются взаимозаменяемыми, и каждый из них располагается внутри кристалла в определенном месте. Специфика и расположение используемого блока относительно других используемых блоков оказывает влияние на производительность.

Существуют методы проектирования, позволяющие разработать модули таким образом, чтобы получить требуемую производительность [4, 5]. Но многие алгоритмы, используемые в современных системах высокоскоростной обработки (системах цифровой связи, системах первичной обработки радиосигналов или системах траекторного сопровождения [6]), требуют существенных вычислительных затрат и большого объема оперативной памяти. Например, при использовании гистограммных алгоритмов определения координат излучающего объекта [7] необходимо за несколько миллисекунд построить проекцию следа пеленгационного конуса на земной поверхности, для чего приходится вычислять значения трансцендентых математических функций в арифметике с плавающей точкой, что достаточно проблематично реализовать в ПЛИС. При этом все построенные и накопленные проекции следов должны сохраняться в памяти вычислителя, что требует десятки мегабайтов, а при использовании параметрической селекции число карт увеличивается кратно числу разделений, что приводит к необходимости использования модулей оперативной памяти на несколько гигабайтов и установки вычислительных модулей, поддерживающих 64-разрядную адресацию. В качестве другого примера можно привести современные алгоритмы фильтрации, предполагающие использование фильтров Калмана. Такие алгоритмы используются при оценке точного положения объекта на траектории [8] и связаны с большим объемом матричных вычислений, точность которых также зависит от использования целочисленных вычислений или арифметики плавающей точки. Все это приводит к росту сложности системы, а это критически сказывается на возможности использования аппаратуры на борту, а не в стационарном режиме. Поэтому зачастую выбор того или иного алгоритма обусловливается поиском компромисса между желаемой точностью результата, желаемой скоростью обработки и затратой аппаратных ресурсов на его получение. При этом различные реализации одного и того же алгоритма могут требовать различные ресурсы, например, вычислительного времени, числа регистров, ячеек памяти, умножителей и пр. В статье [9] рассмотрены особенности применения ПЛИС к различным задачам, возникающим в процессе разработки различного оборудования.

Выбор алгоритма и правильного способа его реализации эвристическими методами приводит к трудно предсказуемым результатам, поэтому целью данной работы является разработка методики, позволяющей провести анализ используемых реализаций тех или иных функциональных модулей и выполнить многокритериальную оценку проекта. Подобная методика должна позволять сравнить несколько систем между собой и сделать выбор лучшей из них по некоторым критериям.

Для решения подобной задачи целесообразно использовать методы системного анализа, предполагающего возможность декомпозиции системы на ряд подсистем (модулей) и проведения оценки отдельных подсистем и синтеза системы, оптимальной по заданным условиям [10, 11].

Применение системного анализа для получения критериев качества алгоритмов

Согласно идеологии системного анализа декомпозиция системы на модули имеет смысл до тех пор, пока членение подсистем низшего уровня не приводит к потере системообразующих свойств [10, 11]. Подобный подход соответствует минимизации числа связей между отдельными подсистемами, так как при потере системообразующих свойств связи между подсистемами начинают соответствовать так называемым промежуточным данным, которые используются для получения того или иного свойства.

Другая сторона системного анализа позволяет свести оценку качества системы к задачам численной оптимизации, т. е. поставить в соответствие некоторым параметрам системы некоторую численную величину, характеризующую то, насколько система соответствует заявленным критериям качества.

Таким образом, чтобы получить возможность применить системный анализ к разработке программного обеспечения для ПЛИС, необходимо выработать критерии и выделить целевую функцию.

Очевидно, что критерии должны удовлетворять определенным физическим процессам, происходящим в кристалле. Основные параметры, которые стремятся улучшить разработчики систем реального времени, — это точность результата, время реакции, разрешение по времени, количество затраченных системных ресурсов (ячеек ПЛИС) и быстродействие.

Под *быстродействием* (производительностью) будем понимать максимальную частоту тактирования ПЛИС f_{max}, при которой разрабатываемый проект способен функционировать в заданном кристалле.

Под *временем реакции* будем понимать максимальное время $t_{обр}$, которое должно пройти с момента появления данных на входе до момента появления результата на выходе.

Под *разрешающей способностью* по времени будем понимать минимальный интервал времени между появлением двух данных на входе, чтобы система могла провести их корректную обработку. Обозначим эту величину как $t_{вх.\ min}$.

Под *количеством затраченных ресурсов* будем понимать число занятых ячеек общего назначения, число ячеек памяти и число специализированных ячеек (модулей ЦОС). Введем две величины M — число использованных ячеек данного типа и $M_{\max}$ — максимальное число ячеек данного типа в выбранной схеме. Введем $\varepsilon = M/M_{\max}$ — относительное число использованных ячеек. Полному использованию одного типа блоков соответствует значение 1. Тогда полному заполнению ПЛИС соответствует значение 3.

Чем меньше использовано ресурсов, тем быстрее можно провести сборку проекта (трассировку между блоками внутри кристалла), тем проще получить желаемую производительность, поэтому при заданном размере кристалла оптимизацию можно проводить на основе минимизации величины

$$\varepsilon_M(\text{alg}) = \frac{M_{mem}(\text{alg})}{M_{mem\max}} + \frac{M_{\log ic}(\text{alg})}{M_{\log ic\max}} + \frac{M_{dsp}(\text{alg})}{M_{dsp\max}}, \quad (1)$$

где ε_M — относительное заполнение ПЛИС при реализации данного алгоритма; M_{mem} — число используемых алгоритмом ячеек памяти; $M_{\log ic}$ — число логических ячеек общего назначения, используемых алгоритмом; M_{dsp} — число ячеек цифровой обработки сигналов (умножителей и фильтров), используемых алгоритмом; alg — обозначение выбранного алгоритма.

При этом критерием для выбора из двух алгоритмов alg_j и alg_i будет являться выражение

$$\varepsilon_M(\text{alg}_i) > \varepsilon_M(\text{alg}_j) \Rightarrow \text{alg}_j \succ \text{alg}_i. \quad (2)$$

Учтем, что невозможно использовать ячеек больше, чем число доступных блоков, т. е. для каждого типа ячеек $\varepsilon_{type} \leqslant 1$ и

$$M_{type}(\text{alg}) \leqslant M_{type\ \max}, \quad (3)$$

где type — тип ячеек: logic, mem, dsp.

Условные ограничения на использование алгоритмов

На проект накладываются ограничения по требуемому быстродействию, разрешающей способности и времени реакции, которые позволяют исключить возможность применения той или иной реализации проекта, если не выполняются условия по достигнутым параметрам:

$$\begin{aligned} f^{\text{д}}_{\max} &\geqslant f_{\max}; \\ t^{\text{д}}_{\text{вх}\ \min} &\leqslant t_{\text{вх}}, \end{aligned} \quad (4)$$

где верхний индекс обозначает достигнутое значение соответствующей величины

$$t^{\text{д}}_{\text{обр}} \leqslant t_{\text{обр}}.$$

Проводить анализ проекта целиком не очень удобно, поэтому целесообразно применить методологию анализа и синтеза из области системного анализа. Тогда мы можем рассмотреть наш проект как систему, которую можно расчленить на ряд подсистем, каждую их которых можно разбить еще дальше, до тех пор, пока дальнейшее дробление потеряет под собой физическое обоснование. Применительно к программно-аппаратному обеспечению подобный подход будет означать разбиение на отдельные функциональные блоки — алгоритмы. Тогда величину относительного заполнения для отдельного алгоритма можно представить, как

$$\varepsilon_{Mi}(\text{alg}_i) = \frac{M_{mem\,i}(\text{alg}_i)}{M_{mem\ \max}} + \frac{M_{\log ici}(\text{alg}_i)}{M_{\log ic\max}} + \frac{M_{dsp}(\text{alg}_i)}{M_{dsp\max}}, \quad (5)$$

а суммарную величину (1) при условии использования выбранного перечня алгоритмов alg как

$$\varepsilon_M(\text{alg}) = \sum_i \varepsilon_{Mi}(\text{alg}_i). \quad (6)$$

Тогда из (1) и (6) следует, что

$$\varepsilon_M(\text{alg}) = \frac{M_{mem}(\text{alg})}{M_{mem\ \max}} + \frac{M_{\log ci}(\text{alg})}{M_{\log ic\max}} + \frac{M_{dsp}(\text{alg})}{M_{dsp\max}}. \quad (7)$$

Полученная формула (7) может применяться только в том случае, если все функциональные блоки синхронизированы между собой, т. е. если выходы каждого из блоков настроены таким образом, чтобы соответствующие друг другу выходы блоков соответствовали друг другу по времени.

Рассмотрим конструкцию из N параллельно функционирующих модулей с общим источником данных D_k в квант времени k. Выходные данные каждого из модулей i Pi_k являются реакцией на входные данные, поступившие в k-й квант времени, считываются потребителем, включенным последовательно с параллельной цепью (рис. 2). Каждый из

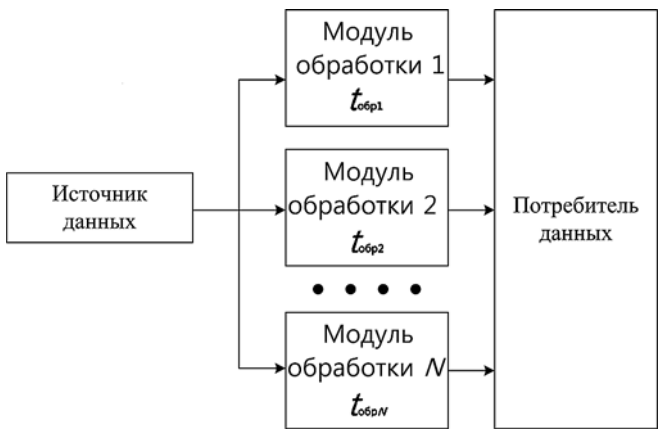

Рис. 2. Схема параллельной обработки данных

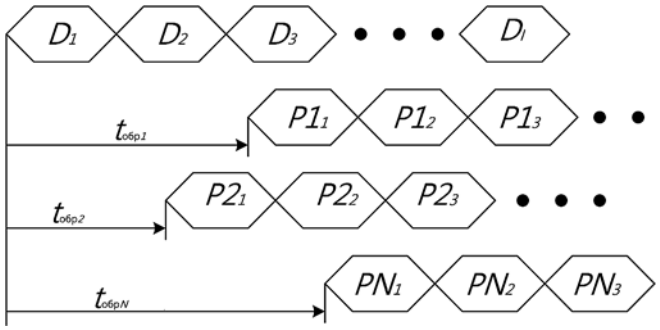

Рис. 3. Временные диаграммы при параллельной обработке данных

модулей обладает своим временем обработки $t_{\text{обр}i}$, поэтому данные поступают на входы потребителя асинхронно (рис. 3). При этом во многих случаях они должны анализироваться в один момент времени (для диаграммы на рис. 3 в момент времени $t_{\text{обр}N}$), т. е. необходимо ввести согласующие линии задержки в остальные каналы поступления данных.

Эквивалентное число тактов для времени обработки данных модулем i при заданном алгоритме alg и требуемых времени реакции $t_{\text{обр}i}$ и предельного быстродействия $f_{i\max}$ для этого модуля составит

$$N_{\text{обр}i}(\text{alg}) = \frac{t_{\text{обр}i(\text{alg})}}{f_{i\max}}.$$

Для корректного анализа данные должны поступить на вход в одно и то же время.

Значит, число тактов, на которое надо задержать сигнал, составит для каждого модуля

$$\Delta N_{\text{обр}i}(\text{alg}) = \max(N_{\text{обр}i}(\text{alg})) - N_{\text{обр}i}(\text{alg}).$$

Число элементов, которые будут затрачены на согласование времени прихода информации от разных модулей, определяется выражением

$$M_{delayi}(\text{alg}) = \Delta N_{\text{обр}i}(\text{alg}) N_{out}(\text{alg}),$$

где N_{out} — разрядность выходной шины модуля при заданном алгоритме.

Суммарное число элементов определяется выражением

$$M_{delay}(\text{alg}) = \sum_i M_{delayi}(\text{alg}).$$

Задержка может быть построена на блоках общего назначения, тогда

$$M_{delay}(\text{alg}) = M_{delay}^{\log ic}(\text{alg}) + M_{delay}^{mem}(\text{alg}),$$

где $M_{delay}^{\log ic}$ — число ячеек общего назначения, используемых для согласования по времени; M_{delay}^{mem} — число ячеек памяти, используемых для согласования.

То есть наличие большого рассогласования во временах обработки параллельных модулей приводит к существенным дополнительным затратам на согласующие линии задержки.

Тогда из (3) мы получаем новое ограничивающее условие:

$$\begin{aligned}
&\sum_i (M_{memi}(\text{alg}_i) + M_{delayi}^{mem}(\text{alg}_i)) < M_{mem\ \max},\\
&\sum_i (M_{\log ici}(\text{alg}_i) + M_{delayi}^{\log ic}(\text{alg}_i)) < M_{\log ic\ \max},\\
&\sum_i M_{dspi}(\text{alg}_i) < M_{dsp\ \max}.
\end{aligned} \qquad (8)$$

А выражение (6) для алгоритма в составе системы записывается как

$$\varepsilon_{Mi}(\text{alg}_i) = \frac{\sum_i M_{memi}(\text{alg}_i) + M_{delayi}^{mem}(\text{alg}_i)}{M_{mem\max}} + \frac{\sum_i M_{\log ici}(\text{alg}_i) + M_{delayi}^{\log ci}(\text{alg}_i)}{M_{\log ic\max}} + \frac{\sum_i M_{dspi}(\text{alg}_i)}{M_{dsp\max}}. \qquad (9)$$

Таким образом, относительное заполнение, которое является основным критерием выбора предпочтительности системы, оказывается зависимым от времени обработки каждого из алгоритмов. Попробуем получить выражение для расчета параметров $t_{\text{обр}}^{\text{д}}$, $t_{\text{вх min}}^{\text{д}}$ и $f_{\max}^{\text{д}}$ в зависимости от того, какой алгоритм используется в данном конкретном случае.

В первом приближении можно считать, что время реакции и разрешающая способность при прочих равных параметрах обратно пропорциональны быстродействию:

$$\frac{t_{\text{вх min2}}}{t_{\text{вх min1}}} = \frac{t_{\text{обр2}}}{t_{\text{обр1}}} = \frac{f_{\max 1}}{f_{\max 2}}$$

или

$$t_{\text{вх min}} = \frac{K_{\text{вх min}}}{f_{\max}},\ t_{\text{обр}} = \frac{K_{\text{обр}}}{f_{\max}}. \qquad (10)$$

Коэффициенты $K_{\text{вх min}}$ и $K_{\text{обр}}$ имеют размерность числа тактов системной частоты и показывают, сколько тактов требуется на реакцию алгоритма и обработку соответственно. В общем случае они не являются константами, а зависят от параметров и структуры системы.

Во-первых, они зависят от выбранного алгоритма и его технической реализации. Для примера сравним два алгоритма преобразования Фурье — прямого преобразования (DFT) и быстрого (FFT). Для определенности допустим, что в один момент времени можно проводить только попарные арифметические операции, тогда сумма S четырех чисел a, b, c и d

$$S = a + b + c + d$$

вычисляется за два такта как попарные суммы S_1 и S_2 за первый такт и последующее вычисление суммы S во второй:

$$S_1 = a + b;$$

$$S_2 = c + d;$$

$$S = S_1 + S_2.$$

Тогда при условии полной параллельности всех вычислений выполнение алгоритма DFT займет N тактов системной частоты. Алгоритм FFT потребует $[\log_2 N] + 1$ тактов, где N — число точек, по которым считается преобразование. Подобная реализация затруднительна для большого числа точек, так как хранить большие объемы информации в неспециализированных ячейках с памятью (регистрах) нерационально, а специализированные ячейки (оперативная память RAM) не позволяет получить доступ к нескольким позициями одновременно. Поэтому в основном используется последовательная реализация алгоритма, при которой затраты по времени возрастают в N раз, т. е. $([\log_2 N] + 1)N$ для FFT и N^2 для DFT.

Это значит, что минимально возможное число тактов, за которые произойдет реакция системы,

$$K_{\text{обр min}}(\text{alg}) = o(\text{alg}),$$

т. е. невозможно получить время реакции, меньшее чем число операций, соответствующих математической сложности алгоритма в его текущей реализации. Подобное справедливо только для условий идеальных математических операций (т. е. любая операция занимает ровно один такт). Для реальных математических операций в ПЛИС существенную роль играет разрядность M. В этом случае одна операция в зависимости от $f_{\max}$ может занимать не один квант времени, а несколько. Обозначим эту величину как $N_{\text{доп}}(f_{\max}, M)$. Дополнительно на времени вычисления могут сказываться дополнительные расходы $N_{\text{расх}}$, обусловленные принципом функционирования ячеек. Например, если алгоритм не предусматривает последовательное чтение памяти, а выбирает следующую ячейку для чтения исходя из результатов последней операции, то сказывается дополнительная задержка на оперативной памяти в два такта (один такт на запись адреса, один такт на выдачу результата). Вместе с тем, если проводится последовательная обработка в некотором алгоритме, например, сложение всех чисел в блоке оперативной памяти, то выбор следующей позиции для чтения не зависит от значения предыдущей позиции. Поэтому данные будут прочитаны за число тактов, равное размеру памяти в словах, плюс дополнительные два такта на работу с памятью — задержку получения данных на выходе памяти относительно входа. То есть подобный вклад может быть аддитивным, мультипликативным или комбинированным (когда есть и аддитивная, и мультипликативная добавки). Исходя из этого можно записать выражение для числа тактов обработки в общем виде:

$$K_{\text{обр}}(\text{alg}) =$$

$$= (N_{\text{доп}}(f_{\max}, M) + N^{\text{мул}}_{\text{расх}}(\text{alg}))o(\text{alg}) + N^{\text{адд}}_{\text{расх}}(\text{alg}), \quad (11)$$

где $N^{\text{адд}}_{\text{расх}}$ — аддитивные дополнительные расходы, а $N^{\text{мул}}_{\text{расх}}$ — мультипликативные.

Таким образом, из выражений (10) и (11) следует:

$$t_{\text{обр}}(\text{alg}) =$$

$$= \frac{(N_{\text{доп}}(f_{\max}, M) + N^{\text{мул}}_{\text{расх}}(\text{alg}))o(\text{alg}) + N^{\text{адд}}_{\text{расх}}(\text{alg})}{f_{\max}},$$

где M определяется исходя из ряда соображений, например, таких как требуемая точность $\varepsilon_{\text{треб}}$ и диапазон значений оцифрованного параметра $\Delta_{\text{диап}}$.

То есть

$$M(\text{alg}) = \frac{\Delta_{\text{диап}}}{\varepsilon_{\text{треб}}} + M_{\text{доп}}(\text{alg}). \quad (12)$$

Ввод добавки $M_{\text{доп}}(\text{alg})$ обусловлен тем, что для работы некоторых алгоритмов (например, алгоритмов последовательного приближения) требуется, чтобы разряд, обеспечивающий заданную точность, был не самым младшим в вычислителе. Например, для вычисления арктангенса по алгоритму CORDIC с точностью до заданного бита требуется увеличить разрядность операций на два-три бита, т. е. $M_{\text{доп}}(\text{atan2}_{\text{CORDIC}}) = 2$.

Максимальное число разрядов, которое можно получить при заданной тактовой частоте, можно рассчитать, приняв во внимание соображение, что в современных кристаллах задержка на связях между элементами много больше, чем задержка переключения ячеек. Тогда можно воспользоваться следующей формулой:

$$N_{\text{доп}}(f_{\max}, M) = \left[\frac{M}{M_0(f_{\max})}\right] + 1,$$

где M_0 — это число ячеек, через которые успеет пройти сигнал между активными фронтами тактовых сигналов, которое, в свою очередь, зависит от времени распространения сигнала между ячейками. Примем упрощение, при котором будем считать, что сигналы распространяются между соседними ячейками. Тогда

$$N_{\text{доп}}(f_{\max}, M) = \left\lfloor \frac{M}{\frac{1}{f_{\max} t_0}} \right\rfloor + 1 = [M f_{\max} t_0] + 1, \quad (13)$$

где t_0 — время распространения сигнала между соседними ячейками. Подставив формулу (12) в выражение (13), получим

$$N_{\text{доп}}(f_{\max}, \Delta_{\text{диап}}, \varepsilon_{\text{треб}}) = \left\lfloor \frac{M}{\frac{1}{f_{\max} t_0}} \right\rfloor + 1 =$$

$$= \left\lfloor \left(\frac{\Delta_{\text{диап}}}{\varepsilon_{\text{треб}}} + M_{\text{доп}}(\text{alg}) \right) f_{\max} t_0 \right\rfloor + 1. \quad (14)$$

Таким образом, из выражений (10), (11) и (14) мы получаем формулу, которой можно описать зависимость времени обработки одного пакета данных:

$$t_{\text{обр}}(f_{\max}, \Delta_{\text{диап}}, \varepsilon_{\text{треб}}, \text{alg}) =$$

$$= \frac{\left(\left(\frac{\Delta_{\text{диап}}}{\varepsilon_{\text{треб}}} + M_{\text{доп}}(\text{alg}) \right) f_{\max} t_0 + 1 + N^{\text{мул}}_{\text{расч}}(\text{alg}) \right) o(\text{alg}) + N^{\text{адд}}_{\text{расч}}(\text{alg})}{f_{\max}}. \quad (15)$$

Второй важной характеристикой, как уже указывалось, является величина $t_{\text{вх min}}$. Данный параметр определить проще всего. Минимальный интервал времени между двумя пакетами, который модуль обработки сможет корректно принять, зависит от характера алгоритма и характера его реализации. Характер алгоритма зависит от того, является ли алгоритм итерационным или непрерывным, т. е. выполняет действия с пришедшим пакетом сразу при его поступлении или ему требуется время на обработку. Например, интегратор является непрерывным по данной терминологии, так как для его работы ему достаточно прибавить поступающие данные к уже имеющимся. Вместе с тем, цифровой фильтр с конечной импульсной характеристикой (КИХ-фильтр) является итерационным, так как в общем случае на каждой итерации необходим пересчет свертки всех записанных данных со всей импульсной характеристикой (ИХ), т. е. потребуется $W_{\text{итер}}$ операций умножения и сложения.

Характер реализации представляет собой способ реализации алгоритма в цифровой технике — итерационный или конвейерный. В первом случае мы экономим ресурсы ценой увеличения $t_{\text{вх min}}$, во втором ценой ресурсов снижаем $t_{\text{вх min}}$. Для описания воспользуемся длиной конвейера $L_{\text{конв}}$ (параллельно вычисляемых операций). Если $L_{\text{конв}} = W_{\text{итер}}$, то новые данные можно подавать каждый такт, так как все операции будут обрабатываться одновременно. Если же $L_{\text{конв}} = 1$, то число одновременно выполняемых операций равно единице, поэтому потребуется $W_{\text{итер}}$ тактов на вычисление результата, в течение которых новые данные невозможно будет обработать.

То есть

$$K_{\text{вх min}}(\text{alg}) = \frac{W_{\text{итер}}(\text{alg})}{L_{\text{конв}}(\text{alg})}. \quad (16)$$

Тогда из выражений (1) и (8) получаем разрешающую способность

$$t_{\text{вх min}}(\text{alg}) = \frac{W_{\text{итер}}(\text{alg})}{L_{\text{конв}}(\text{alg}) f_{\max}}. \quad (17)$$

Третьей важной величиной является достигнутое быстродействие $f_{\max}$. Эта величина определяет устойчивость работы микросхемы к изменяющимся условиям окружающей среды, допустим, к флуктуациям температуры или джиттеру тактового сигнала. Достигнутое быстродействие можно описать также как и желаемое быстродействие, то есть через максимальную длину, на которую распространится сигнал за период тактового сигнала

$$f_{\max} = \frac{1}{l_{\max}(\text{alg}) t_0}, \quad (18)$$

где $l_{\max}$ — максимальная длина пути (в ячейках) между двумя регистрами в модуле.

Полученные формулы (15), (17) и (18) позволяют рассчитать требуемые параметры алгоритмов при выбранных ограничивающих $f_{\max}$, $t_{\text{обр}}$ и $t_{\text{вх min}}$ либо рассчитать достигнутые параметры $f^{\text{д}}_{\max}$, $t^{\text{д}}_{\text{обр}}$ и $t^{\text{д}}_{\text{вх min}}$ при фиксированных алгоритмах.

Из выражения (18) видно, что тактовая частота получается в прямой зависимости от времени на обработку и минимального интервала между поступлениями данных, т. е. она является следствием необходимости обеспечения своевременной обработки данных. Таким образом, в выражения (15) и (17) можно подставить значение частоты из выра-

жения (18). Тогда выражения для расчета временных параметров алгоритма примут вид.

$$t_{\text{вх min}}(\text{alg}) = \frac{W_{\text{итер}}(\text{alg}) l_{\max}(\text{alg}) t_0}{L_{\text{конв}}(\text{alg})};$$

$$t_{\text{обр}}(l_{\max}, \Delta_{\text{диап}}, \varepsilon_{\text{треб}}, \text{alg}) =$$

$$= \left(\frac{\frac{\Delta_{\text{диап}}}{\varepsilon_{\text{треб}}} + M_{\text{доп}}(\text{alg})}{l_{\max}} + 1 + N^{\text{мул}}_{\text{расх}}(\text{alg}) \right) \times \tag{19}$$

$$\times\, o(\text{alg}) l_{\max}(\text{alg}) t_0 + N^{\text{адд}}_{\text{расх}}(\text{alg}) l_{\max}(\text{alg}) t_0.$$

Анализ данных величин позволяет найти в проекте участки, которые снижают возможность достижения требуемой производительности и, проанализировав их, изменить таким образом, чтобы они удовлетворяли заданным условиям.

Существуют два основных способа учета параметров (19) для ограничивающего условия (4): анализ "сверху вниз" и анализа "снизу вверх". В первом случае мы после разбиения системы на части задаем для каждой подсистемы свои временные ограничения, чтобы рассчитать временные параметры и применить ограничивающее условие (4) к каждой из подсистем. Второй вариант — рассчитать временные параметры для каждой из подсистем, из них получить достигнутые временные параметры системы, например, для времени обработки это будет время наиболее длинного пути распространения данных от входа к выходу. И применить ограничивающее условие (4) к временным параметрам самой системы.

Для первого случая и с введенным упрощением из (4) и (19) получаем ограничения, накладываемые на каждую подсистему в проекте:

$$\left(\left(\frac{\frac{\Delta_{\text{диап}}}{\varepsilon_{\text{треб}}} + M_{\text{доп}}(\text{alg})}{l_{\max}} + 1 + N^{\text{мул}}_{\text{расх}}(\text{alg}) \right) o(\text{alg}) + \right.$$

$$\left. + N^{\text{адд}}_{\text{расх}}(\text{alg}) \right) l_{\max}(\text{alg}) t_0 \leqslant t_{\text{обр}}; \tag{20}$$

$$\frac{W_{\text{итер}}(\text{alg}) l_{\max}(\text{alg}) t_0}{L_{\text{конв}}(\text{alg})} \leqslant t_{\text{вх}}.$$

Рассмотрим два простых примера, иллюстрирующих выбор ограничивающих параметров для условия (4) с упрощением по частоте и определение наилучшей системы согласно критерию (2) при расчете по формуле (9).

Выбор ограничивающих величин $t_{\text{вх}}$ и $t_{\text{обр}}$ диктуется требованиями к обработке и принципам работы модуля. Рассмотрим две ситуации, в которой на ПЛИС происходит обработка некоторого сигнала с частотой дискретизации $f_{\text{д}}$.

В зависимости от назначения устройства, мы можем получить разные $t_{\text{вх}}$. Представим два случая, в одном из них ПЛИС предназначена для цифровой обработки звука в музыкальной аппаратуре, во втором ПЛИС используется в ЦОС эхолота.

4. Примеры расчета критериев

В музыкальной аппаратуре поступающий сигнал необходимо обрабатывать постоянно. То есть аппаратура не может сделать паузу, подумать и после этого выдать результат. Таким образом, мы получаем величину $t_{\text{вх}}$, равную $1/f_{\text{д}}$. При этом данные будут обрабатываться в реальном времени. Данные, поступившие через $t_{\text{вх}}$, будут обработаны, а не буферизованы. Если бы $t^{\text{д}}_{\text{вх min}} \leqslant t_{\text{вх}}$ не выполнялось, то данные записывались бы в буфер в надежде, что будет время обработать их позже. Но так как в данном приложении все отсчеты и моменты времени равноправны, то увеличивалась бы задержка передачи данных от входа к выходу с течением времени, либо происходило бы неучтенное алгоритмами прореживание, что привело бы к неправильным или нестабильным результатам работы аппаратуры.

Теперь рассмотрим приложение ЦОС для эхолота. Предположим, что максимальная глубина, которую можно анализировать эхолотом, равна $h_{\max}$. Тогда время, которое будет потрачено на получение отсчетов сигнала, будет равно $t_{\text{ан}} = \frac{2h_{\max}}{v_{\text{с}}}$, где $v_{\text{с}}$ — скорость распространения сигнала в среде. Допустим, что эхолот посылает зондирующий сигнал каждые $t_{\text{з}}$. В этом случае мы должны принять $N = t_{\text{ан}} f_{\text{д}}$ отсчетов, но эти отсчеты мы можем обрабатывать в течение большего времени. Тогда $t_{\text{вх}} = \frac{t_{\text{з}}}{N} = \frac{t_{\text{з}}}{t_{\text{ан}} f_{\text{д}}}$. В зависимости от периода следования зондирующих сигналов мы получаем разные результаты. Например, если сигнал излучается через интервалы $t_{\text{ан}}$, то задача сводится к предыдущей, т. е. $t_{\text{вх}} = \frac{1}{f_{\text{д}}}$. Если же период излучения больше, то мы получаем два контура. Один из них должен успевать обрабатывать сигналы на входной частоте (частоте АЦП) и обеспечивать буферизацию входных сигналов, а второй — обеспечивать возможность чтения из буфера и обработку записанных отсчетов за время между посылками. Если среднее время обработки отсчета получится больше $\frac{t_{\text{з}}}{t_{\text{ан}} f_{\text{д}}}$, то возникнут те же самые проблемы, когда мы бу-

дем вынуждены либо терять информацию, либо "плыть" по времени, прошедшем от появления отсчета на входе до появления результата на выходе.

Аналогичная ситуация со временем обработки. Оно диктуется несколькими факторами. В глобальном смысле (в масштабе всего проекта) время определяется как максимальное время реакции на изменение. На примере той же обработки звука, если задержка будет слишком большой, то это будет сказываться на качестве игры, потому что фактически будет получаться, что исполнители не слышат ни того, как играют они, ни того, как играют соседи. То есть время реакции в данном случае определяется разрешающей способностью человека по времени.

Для эхолота время обработки зависит от сферы применения эхолота. Например, рассмотрим применение эхолота для поиска рифов при движении на плавательном транспортном средстве. Путь максимальное расстояние, на котором действует эхолот — $h_{\max}$, скорость транспортного средства v, время реакции транспортного средства на информационное сообщение об обнаруженных рифах $t_{\text{р}}$, скорость распространения звука в среде $v_{\text{зв}}$, а максимальное ускорение судна a. Тогда максимальное время остановки составит $t_{\text{ост}} = \frac{2h_{\max}}{v_{\text{зв}}} + t_{\text{р}} + \frac{v}{a} + t_{\text{обр}}$.

Пройденный путь составит

$$S_{\text{ост}} = v\frac{v}{a} - \frac{a\left(\frac{v}{a}\right)^2}{2} + v\left(\frac{2h_{\max}}{v_{\text{зв}}} + t_{\text{р}} + t_{\text{обр}}\right) = \frac{v^2}{2a} + v\left(\frac{2h_{\max}}{v_{\text{зв}}} + t_{\text{р}} + t_{\text{обр}}\right).$$

Пройденный путь должен быть меньше, чем $h_{\max}$, т. е.

$$\frac{v^2}{2a} + v\left(\frac{2h_{\max}}{v_{\text{зв}}} + t_{\text{р}} + t_{\text{обр}}\right) < h_{\max}.$$

Из этого выражения вычисляется требуемое время $t_{\text{обр}}$.

Накладываемые таким образом ограничения позволяют исключить из рассмотрения ряд алгоритмов, которые заведомо не удовлетворяют этим условиям.

Разработанные методы были применены к проектированию модулей цифровой обработки, что отражено в [8].

Заключение

Таким образом, необходимость поиска лучших алгоритмов для реализации заданной системы на ПЛИС ставит перед разработчиком ряд задач. Необходимо провести разбиение системы на составляющие блоки, рассчитать параметры возможных алгоритмов и граничные величины параметров (1) для каждого алгоритма. После этого необходимо провести сравнение алгоритмов по критерию (16) при учете системы условий (13) для каждого из алгоритмов, системы условий (15) для текущей реализации системы, а также при выполнении условий (13) для всей системы целиком.

Разработанная система критериев позволяет сократить время разработки программы, найти в проекте "узкие" места, не позволяющие достигнуть заложенных требований, выбрать из всех алгоритмов те, которые будут соответствовать заданным требованиям по скорости, точности и производительности. При этом появляется возможность оптимального использования алгоритмов на заданной аппаратной базе (выбранной микросхеме), либо, наоборот, становится возможным выбрать микросхему, на которой данный проект стоит реализовывать.

Список литературы

1. **Тарасов И. Е.** Разработка цифровых устройств на основе ПЛИС Xilinx с применением языка VHDL. М.: Горячая линия — Телеком, 2005. 253 с.

2. **Altera** corporation, Stratix II device handbook [Электронный ресурс]. URL: https://www.altera.com/en_US/pdfs/literature/hb/stx2/stratix2_handbook.pdf (дата обращения: 15.04.2015).

3. **Virtex-6** Family Overview [Электронный ресурс]. URL: http://www.xilinx.com/support/documentation/data_sheets/ds150.pdf (дата обращения: 15.04.2015).

4. **Johnson H. W., Graham M.** High-Speed Digital Design. USA: Prentice Hall, 1993. 384 с.

5. **Кондратов К. С., Жураковский В. Н., Силин С. И.** Принципы проектирования встраиваемых систем на основе программных средств // Инженерный вестник. 2014. №11. С. 536—544.

6. **Логвиненко А. С., Жураковский В. Н.** Повышение точности измерения параметров сигналов в цифровом тракте // Инженерный вестник. 2014. №10. С. 614—651.

7. **Жураковский В. Н., Кондратов К. С.** Алгоритм определения местоположения наземных объектов в условиях низкой точности входных данных // Наука и образование: электронное научно-техническое издание. 2013. № 12. С. 307—314.

8. **Шахтарин Б. И., Микаэльян С. В.** Траекторный фильтр в системе координат измерителя для системы слежения за целями по угломерным данным // Научный вестник Московского государственного технического университета гражданской авиации. 2013. № 193. С. 21—25.

9. **Smetana D.** Designing a general-purpose FPGA DSP card for EW, radar applications using the latest generation of FPGAs [Электронный ресурс] // Military Embedded Systems. 29.01.2014. URL: http://mil-embedded.com/articles/designing-general-purpose-applications-using-latest-generation-fpgas (дата обращения: 03.09.2015).

10. **Николаев В. И., Брук В. М.** Системотехника: методы и приложения. Л.: Машиностроение, Ленингр. отд-ние, 1985. 199 с.

11. **Антонов А. В.** Системный анализ. М.: Высшая школа, 2004. 454 с.

12. **Еременков А. И., Жураковский В. Н., Силин С. И.** Аналого-цифровой модуль приема и передачи данных. Разработка алгоритмов и программного обеспечения // Инженерный вестник. 2014. №10. С. 652—665.

V. N. Zhurakovskiy, Associate Professor, **S. I. Silin,** Assistant, e-mail: sm2-2@inbox.ru,
Bauman MSTU

System Analysis Applied to the Data Processing Algorithms Selection for Modern Computing Systems

Main purpose of this work is development of criteria system that could be applied to develop sophisticated devices based on field programmable gate array (FPGA), systems on the chip (SOC) and complex programmable logic devices (CPLD). In the first chapter of this article authors reviewed typical modern approaches to fulfilling this task. Authors provided explanation of the basis which the further apply to developed theory. In the second chapter authors developed criteria and objective function that could be used during system development to select the most suitable algorithm and realization. In the third chapter authors developed limitations to be used during selection process. Limitations are based on the typical structure of programmable integrated circuits and properties of typical problems arise during development process. In the fourth chapter authors placed example of different signal processing algorithms that could be analyzed with developed criteria and limitation systems. In the fifth chapter conclusion to work is placed. This theory suits to give mathematical and theoretical explanation of the selection process during DSP algorithms development.

Keywords: *FPGA, system analysis, algorithm, parallel signal-processing, radiolocation, embedded systems, systems on the chip, electronics, circuit technique, systems engineering, system design*

References

1. **Tarasov I. E.** *Razrabotka cifrovyh ustrojstv na osnove PLIS Xilinx s primeneniem jazyka VHDL*. Moscow: Gorjachaja linija — Telekom, 2005, 253 p.

2. **Altera** *corporation*, Stratix II device handbook [Electronic resource]. URL: https://www.altera.com/en_US/pdfs/literature/hb/stx2/stratix2_handbook. pdf (date of circulation: 15.04.2015).

3. **Virtex-6** *Family Overview* [Electronic resource]. URL: http://www.xilinx.com/support/documentation/data_sheets/ds150.pdf (date of circulation: 15.04.2015).

4. **Johnson H. W., Graham M.** *High-Speed Digital Design*. Prentice Hall, 1993, 384 p.

5. **Kondrashov K. S., Zhurakovskij V. N., Silin S. I.** Principy proektirovanija vstraivaemyh sistem na osnove programmnyh sredstv, *Inzhenernyj vestnik*, 2014, no. 11, pp. 536—544.

6. **Logvinenko A. S., Zhurakovskij V. N.** Povyshenie tochnosti izmerenija parametrov signalov v cifrovom trakte, *Inzhenernyj vestnik*, 2014, no. 10, pp. 641—651.

7. **Zhurakovskij V. N., Kondrashov K. S.** Algoritm opredelenija mestopolozhenija nazemnyh obektov v uslovijah nizkoj tochnosti vhodnyh dannyh, *Nauka i obrazovanie: jelektronnoe nauchno-tehnicheskoe izdanie*, 2013, no. 12, pp. 307—314.

8. **Shahtarin B. I., Mikajel'jan S. V.** Traektornyj fil'tr v sisteme koordinat izmeritelja dlja sistemy slezhenija za celjami po uglomernym dannym, *Nauchnyj vestnik Moskovskogo gosudarstvennogo tehnicheskogo universiteta grazhdanskoj aviacii*, 2013, no. 191, pp. 21—25.

9. **Smetana D.** Designing a general-purpose FPGA DSP card for EW, radar applications using the latest generation of FPGAs [Electronic resource], *Military Embedded Systems*. 29.01.2014. URL: http://mil-embedded.com/articles/designing-general-purpose-applications-using-latest-generation-fpgas (date of circulation: 03.09.2015).

10. **Nikolaev V. I., Bruk V. M.** *Sistemotehnika: metody i prilozhenija*. Leningrad: Mashinostroenie, Leningr. otd-nie, 1985, 199 p.

11. **Antonov A. V.** Sistemnyj analiz. Moscow: Vysshaja shkola, 2004. 454 p.

12. **Eremenkov A. I., Zhurakovskij V. N., Silin S. I.** Analogo-cifrovoj modul' priema i peredachi dannyh. Razrabotka algoritmov i programmnogo obespechenija, *Inzhenernyj vestnik*, 2014, no. 10, pp. 652—665.