УДК004.31

Бортовая информационно-управляющая система на основе MIPS архитектуры с применением расширения системы команд UDI и аппаратным ускорителем тригонометрических вычислений

Е.В. Ливенцев, А.Л. Переверзев, Е.В. Примаков, А.М. Силантьев

Национальный исследовательский университет «МИЭТ», primakov@org.miet.ru

***Аннотация* — В статье рассмотрена задача построения интегрированных бортовых информационно-управляющих систем для малогабаритных беспилотных летательных аппаратов на базе ПЛИС. Основу рассмотренной системы составляет синтезируемый 32-разрядный RISC процессор MicroAptiv UP. Предложено решение проблемы увеличения скорости расчетов тригонометрических функций за счет использования аппаратного ускорителя на базе алгоритма CORDIC. Рассмотрены два варианта подключения ускорителя к микропроцессорному ядру: системная шина AHB-Lite и интерфейс пользовательских команд CorExtend (UDI). Проведён сравнительный анализ производительности системы при данных способах подключения. Показано, что при использовании аппаратного ускорителя, подключенного через интерфейс UDI, по сравнению с программной реализацией алгоритма CORDIC было получено увеличение скорости вычислений в 7 раз, а по сравнению с аппаратной реализацией на шине AHB-Lite – увеличение скорости на 18%. Аппаратные затраты, необходимые для реализации ускорителя составили 1% от затрат на реализацию процессорного ядра.**

***Ключевые слова* — ИУС, аппаратный ускоритель, MIPS, UDI, CorExtend, ПЛИС, FPGA.**

I. Введение

Управление малогабаритными подвижными объектами, например, аэромоделями, самолетами-мишенями и т.д. осуществляется при помощи бортовых информационно-управляющих систем. Одним из способов совершенствования массогабаритных и энергетических характеристик таких систем, а также снижения их стоимости при крупном серийном производстве является повышение степени интеграции. Применение программируемых логических интегральных схем (ПЛИС) позволяет не только повысить степень интеграции, но и уменьшить аппаратную избыточность за счет создания специализированных структур, включающих в себя процессорные ядра, пользовательские модули, ОЗУ и контроллеры различных интерфейсов передачи данных. Помимо этого, ПЛИС обеспечивают высокую производительность в задачах потоковой обработки информации.

Пример специализированной структуры, обеспечивающей управление малогабаритным беспилотным летательным аппаратом, представлен на рис. 1. Данная система осуществляет прием, обнаружение и дешифрацию команд курса и высоты, поступающих по радиоканалу управления, выполнение функционального алгоритма, формирование телеметрической информации [1].

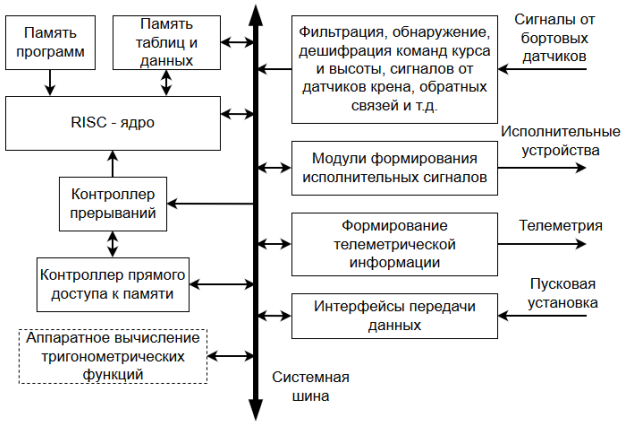

Рис. 1. Фрагмент структуры бортовой информационно-управляющей системы малогабаритного беспилотного летательного аппарата

Традиционно на аппаратном уровне реализуется первичная обработка сигналов с бортовых датчиков и систем передачи информации, а на программном – выполнение основного функционального и вспомогательных алгоритмов.

Выполнение основного функционального алгоритма включает в себя решение задач навигации и позиционирования объекта управления в пространстве. Для решения этих задач требуется проводить расчёт тригонометрических функций в режиме реального времени с относительно высоким быстродействием. На практике не всегда можно добиться необходимой скорости вычислений в рамках требований по энергопотреблению, стоимости и габаритам, используя исключительно программные средства. В таком случае, одним из эффективных способов увеличения производительности является использование аппаратных ускорителей (на рис. 1 обозначен пунктирной линией).

Включение такого ускорителя в состав системы – задача, которую можно решить подключением его к системной шине процессора. Такой подход имеет недостатки, например, задержки при обмене данными по общей шине. В ряде случаев создатели процессорных ядер предоставляют разработчикам более эффективные возможности интеграции внешних модулей в систему, например, в виде сопроцессора или модуля на специализированной шине, отличной от системной.

В настоящей работе исследованы способы интеграции аппаратного ускорителя тригонометрических вычислений с синтезируемым 32-разрядным RISC-процессором MIPSfpga на основе ядра MicroAptiv UP, а также выполнен сравнительный анализ быстродействия систем с использованием:

- расширения набора команд CorExtend (UDI);
- подключения ускорителя к системной шине AHB-Lite.

Предложенные структуры цифровой части системы описаны на Verilog HDL и могут быть реализованы как на основе ПЛИС, так и на основе базового матричного кристалла или заказной интегральной схемы.

II. Анализ способов подключения аппаратных ускорителей вычислений к ядру MicroAptiv UP

A. *Подключение с помощью интерфейса UDI*

Механизм пользовательских команд (user defined instructions, CorExtend UDI) позволяет обращаться к внешним по отношению к процессору модулям через подмножество инструкций UDI0-UDI15. При выполнении этих инструкций данные поступают на обработку в пользовательский модуль вместо АЛУ. Взаимодействие модуля с конвейером процессора при обработке пользовательских команд показано на рис. 2. Как правило, UDI используется для подключения модулей, реализующих операции с несколькими входными операндами и одним результатом.

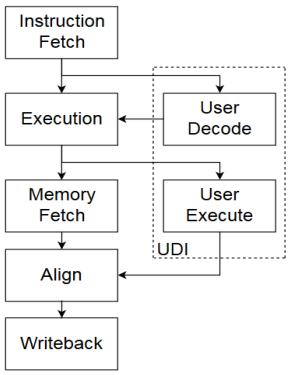

Рис. 2. Принцип взаимодействия блока UDI с конвейером процессора

Пользовательский модуль, подключенный к интерфейсу UDI, может обращаться к регистровому файлу процессора, записывать результат напрямую в регистры и иметь собственную память. Также модуль может приостанавливать конвейер процессора на время своей работы [2]-[3].

К достоинствам CorExtend можно отнести удобство использования команд UDI при разработке низкоуровневого программного обеспечения системы.

B. *Подключение к системной шине AHB-Lite*

Интерфейс AHB-Lite – системная шина с одним мастер-устройством, которая используется для подключения периферийных блоков к микроядру. Типовая схема устройства с системной шиной AHB-Lite представлена на рис. 3 [4]. Подключенный к системной шине модуль не имеет прямого доступа к регистрам процессора на чтение или запись. При таком подключении регистры периферийного модуля отображаются в системную память, для обмена данными используются операции чтения и записи в память. Проверка готовности данных осуществляется, например, операцией чтения из статусного регистра модуля.

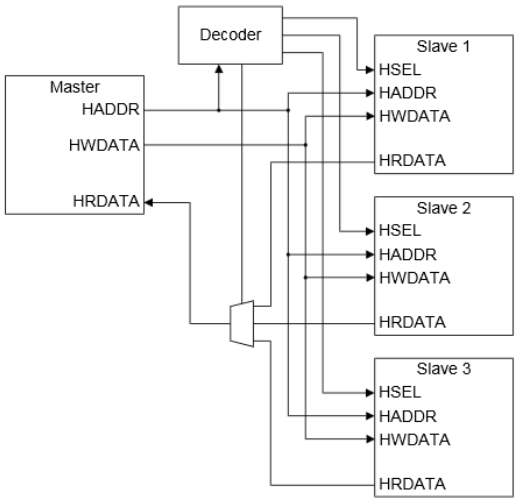

Рис. 3. Схема устройства с шиной AHB-Lite

III. Реализация аппаратного ускорителя тригонометрических вычислений на основе алгоритма CORDIC

Для реализации аппаратного ускорителя был выбран алгоритм CORDIC (*COordinate Rotation DIgital Computer*) – итерационный метод вычисления значений тригонометрических функций с помощью поворота вектора на плоскости.

Суть метода заключается в последовательном приближении к заданному углу путём поворота вектора по и против часовой стрелки на угол, уменьшающийся с каждой итерацией. Графическая интерпретация метода представлена на рис. 4.

При реализации этого алгоритма в виде цифрового устройства можно свести операции поворота вектора, интенсивно использующие операцию умножения, к операциям сдвига и сложения, что существенно уменьшает аппаратные затраты на устройство и увеличивает его производительность [5].

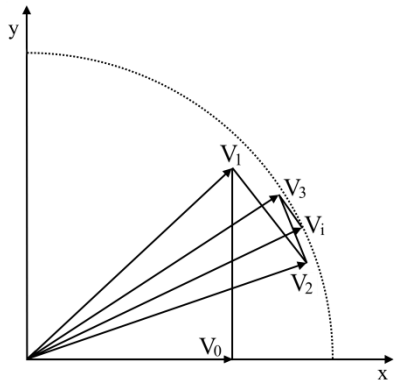

Рис. 4. Принцип работы алгоритма CORDIC (V0 – начальный вектор, Vi – вектор после i-й итерации)

Пример структурной схемы блока поворота приведён на рис. 5. Блок принимает координаты вектора (X_i; Y_i), угол поворота Z_i и производит элементарный поворот системы координат. На выход модуля выводится преобразованный вектор (X_{i+1} ; Y_{i+1}) и угол Z_{i+1} для следующей итерации. После окончания вычислений X будет содержать косинус начального угла, а Y – синус. Таким образом, алгоритм CORDIC позволяет выполнять расчёт синуса и косинуса одного аргумента одновременно [5].

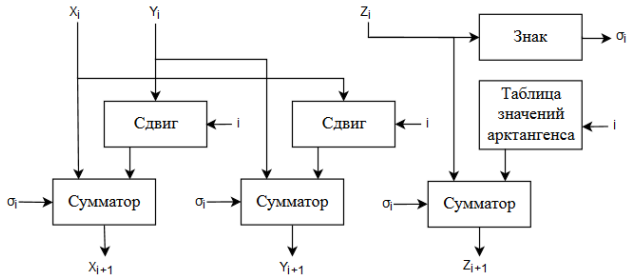

Рис. 5. Структурная схема блока поворота системы координат

На рис. 6 представлена структура разработанного ускорителя тригонометрических вычислений на базе алгоритма CORDIC, который состоит из устройства управления (УУ) на основе конечного автомата, блока поворота системы координат и буферного регистра. Программное управление ускорителем и обмен данными с процессорным ядром осуществляется посредством взаимодействия с УУ, которое реализует интерфейсы UDI и AHB-Lite.

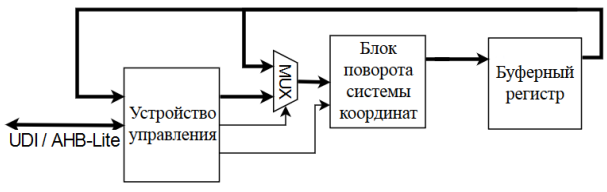

Рис. 6. Структурная схема ускорителя на основе алгоритма CORDIC

При подключении через UDI, после получения команды от процессора, УУ:

1. производит декодирование команды, и, если полученная команда корректна, то УУ возвращает процессору адрес регистра общего назначения (РОН), в который будет произведена запись результата, в противном случае – вызывает системное исключение;
2. передаёт в итерационный блок поворота системы координат начальные данные и запускает процесс вычисления;
3. на время вычисления устанавливает флаг, приостанавливающий конвейер процессора;
4. ожидает окончания вычислений;
5. по окончании вычислений результат из буферного регистра направляется через интерфейс UDI в процессорное ядро для записи в РОН.

Принцип работы УУ при подключении через AHB-Lite:

1. процесс вычисления инициируется записью аргумента функции в регистр УУ;
2. УУ ожидает окончания вычисления;
3. после окончания расчёта результаты становятся доступны для чтения из регистров УУ и устанавливается флаг, сигнализирующий о завершении вычислений;
4. результаты считываются процессором из регистров УУ.

Ускоритель работает с 32-х битными числами в формате с фиксированной точкой: старший бит – знаковый, бит 30 – целая часть числа, биты 29-0 дробная часть. Одна операция вычисления пары тригонометрических функций выполняется за 31 итерацию.

Алгоритм CORDIC работает только с аргументами из диапазона [0; π/2] [5]. Для устранения этого ограничения в устройство управления добавлена схема, расширяющая область определения вычислителя на четвертый квадрант с использованием свойств симметрии тригонометрических функций. Таким образом, разработанный ускоритель работает в диапазоне аргументов [-π/2; π/2].

IV. Программное взаимодействие ядра MIPSFPGA с вычислителем CORIDC через интерфейс UDI

Реализованы следующие ассемблерные команды:

- UDI0 RS; RD; расчёт косинуса от операнда RS, значение записывается по адресу RD. Останавливает конвейер на 31 такт.
- UDI1 RS; RD; расчёт синуса от операнда RS, значение записывается по адресу RD; Останавливает конвейер на 31 такт.
- UDI2 RD; возвращение значения парной функции по отношению к последней рассчитанной от того же аргумента, значение записывается по адресу RD.

Для реализации команды UDI2 используется особенность алгоритма CORDIC – одновременный расчёт синуса и косинуса аргумента. Рассчитанное значение парной функции сохраняется в ускорителе и может быть извлечено с помощью команды UDI2 без проведения повторного расчета. В случае с ранее рассчитанным синусом UDI2 вернёт косинус, и наоборот.

Упрощение использования команд UDI в языке C возможно с помощью функции с ассемблерными вставками (на примере косинуса – UDI0):

```
int cordic_cos(int x)
{
 int* px = &x;
 int Result;

 __asm__(
 "lw $t1, %1 \n\t" // Чтение из памяти в рег.
                   // $t1
 "udi0 $t1, 0, $t2, 0 \n\t" // Вызов
                            // $t2=udi0($t1)
 "sw $t2, %0" // Запись результата из $t2 в
              // память
 : "=m" (Result)
 : "m" (*px)
 );
 return Result;
}
```

Листинг 1. Пример использования UDI0 в виде функции языка C

V. Методика тестирования производительности

Сравнение производительности тригонометрических вычислений произведено для следующих способов расчёта:

- программный расчёт по алгоритму CORDIC;
- аппаратно-ускоренный расчёт с применением разработанного модуля CORDIC, подключенного к системной шине процессора AHB-Lite;
- аппаратно-ускоренный расчёт с применением разработанного модуля CORDIC, подключенного через интерфейс UDI.

Для тестирования выбрана характерная бортовым информационно-управляющим системам задача: расчёт косинуса от аргумента в диапазоне $[-\pi/2; \pi/2]$.

Компиляция тестовой программы производилась пакетом gcc с флагом оптимизации “-o2”. Чтение аргументов для теста проводилось из некэшируемой области памяти, чтобы минимизировать влияние работы механизма кэширования на результаты.

За меру производительности принято среднее время одного вычисления при выполнении набора из 1000 последовательных вычислений. Результаты округлены до ближайших целых. Измерение времени производилось с помощью системного таймера.

Во всех тестах аппаратный ускоритель CORDIC был запущен на тактовой частоте процессорного ядра. Для дальнейшего увеличения производительности возможна работа ускорителя на частотах, в 2-2.5 раза превышающих тактовую частоту ядра.

VI. Результаты тестирования

Результаты тестирования в соответствии с предложенной методикой представлены на рис. 7 и в таблице 1. Из приведенных данных видно, что при использовании интерфейса UDI, по сравнению с программным расчётом по алгоритму CORDIC было получено увеличение скорости вычислений в 7.2 раза. При этом аппаратные затраты на реализацию CORDIC+UDI составляют приблизительно 1% от аппаратных затрат на реализацию ядра MicroAptiv UP.

Рис. 7. Результаты расчёта косинуса

Таблица 1

Сводная таблица результатов тестирования

Реализация	Время выполнения одной операции
Программный CORDIC	346 тактов
Аппаратный CORDIC с UDI	48 тактов
Аппаратный CORDIC с AHB-Lite	57 тактов

По сравнению с конфигурацией на основе AHB-Lite, UDI обеспечивает меньшие задержки при обмене данными, сокращая время, затрачиваемое на выполнение одной задачи на 18%.

VII. Выводы

Проведён обзор двух способов подключения пользовательских модулей к процессору MircoAptiv UP: интерфейс CorExtend (UDI) и системная шина AHB-Lite.

Разработано Verilog-описание ускорителя на основе алгоритма CORDIC, обеспечивающее вычисление

тригонометрических функций в диапазоне аргументов [-π/2; π/2].

Проведено сравнительное тестирование производительности трёх конфигураций: без использования ускорителя, с использованием ускорителя, подключенного по шине AHB-Lite и с использованием ускорителя CORDIC, подключенного через интерфейс UDI.

В результате тестирования была продемонстрирована эффективность реализации ускорителя вычислений CORDIC, подключенного к процессору MIPS MicroAptiv UP через интерфейс UDI. По сравнению с программным расчётом было получено увеличение скорости вычислений в 7.2 раза. Аппаратные затраты на реализацию предложенной структуры составили 1% от затрат на реализацию процессорного ядра. Ускоритель, подключенный через интерфейс UDI, продемонстрировал прирост производительности 18% по сравнению с модулем, подключенным к системной шине.

В перспективе планируется запуск ускорителя на тактовой частоте, превышающей частоту процессорного ядра, а также конвейеризация потоковых вычислений.

Литература

[1] Беклемишев Д.Н., Переверзев А.Л., Твердунов Д.В. Однокристальный вычислитель для беспилотного летательного аппарата // Известия вузов. Электроника. М.: МИЭТ, 2010. №6. С. 33-38.

[2] MIPS32 MicroAptiv UP Processor Core Family Integrator's Guide // Imagination Technologies, 2014. pp. 20-24.

[3] Zats, K. Using MIPS microAptiv UP Processor CorExtend UDI interface, 2016. URL: http://zatslogic.blogspot.com/2016/01/using-mips-microaptiv-up-processor.html (дата обращения: 31.03.2016)

[4] MIPS32 MicroAptiv UP Processor Core AHB-Lite Interface // Imagination Technologies, 2014. P. 2-16.

[5] Ercegovac M.D., Lang T., Digital Arithmetic // Morgan Kaufmann Publishers, 2014. pp. 609-648.

On-board flight control system based on the MIPS architecture with CorExtend user-defined instructions and hardware-accelerated trigonometry calculations

E.V. Liventsev, A.L. Pereverzev, E.V. Primakov, A.M. Silantiev

National Research University of Electronic Technology (MIET), primakov@org.miet.ru

***Keywords* — control system, hardware accelerator, MIPS, UDI, CorExtend, FPGA.**

Abstract

The article considers the issue of creating integrated FPGA-based on-board control systems for small unmanned aircraft. The system is based on a synthesizable 32-bit RISC microAptiv UP processor core. There is problem in similar systems that software-based solutions for calculating trigonometric functions may not be fast enough for real time calculations. The suggested solution to the problem is using a hardware accelerator based on CORDIC algorithm. Two options of connecting an accelerator to the microprocessor are considered: AHB-Lite system bus and user-defined instructions UDI interface. The article gives a comparative performance analysis of systems with CORDIC module connected with AHB-Lite interface and UDI interface. It is reported that calculation speed with hardware accelerator compared to software calculation had increased by 7.2 times for the calculation of sine or cosine functions. It is also reported that using UDI interface provides 18% increase in performance of hardware accelerator compared to AHB-Lite interface. Implementation of the proposed circuit requires additional 1% more hardware cost compared to the cost of the processor core.

References

[1] Beklemishev D.N., Pereverzev A.L., Tverdunov D.V. Odnokristal'nyj vychislitel' dlja bespilotnogo letatel'nogo apparata // Izvestija vuzov. Jelektronika. Moscow, MIET, 2010. No. 6. P. 33-38 (in Russian).

[2] MIPS32 MicroAptiv UP Processor Core Family Integrator's Guide // Imagination Technologies, 2014. P. 20-24.

[3] Zats, K. Using MIPS microAptiv UP Processor CorExtend UDI interface, 2016. URL: http://zatslogic.blogspot.com/2016/01/using-mips-microaptiv-up-processor.html (access date: 31.03.2016)

[4] MIPS32 MicroAptiv UP Processor Core AHB-Lite Interface // Imagination Technologies, 2014. P. 2-16.

[5] Ercegovac M.D., Lang T., Digital Arithmetic // Morgan Kaufmann Publishers, 2014. P. 609-648.