УДК 681.327

Е. И. Гурин, И. В. Огнев

ПОСТРОЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ БЛОКОВ НА ОСНОВЕ ПРОГРАММИРУЕМЫХ ЛОГИЧЕСКИХ ИНТЕГРАЛЬНЫХ СХЕМ СО СПЕЦИАЛИЗИРОВАННЫМИ СОПРОЦЕССОРАМИ

Аннотация. Рассматриваются вопросы построения встроенных вычислительных блоков на основе программируемых логических интегральных схем со специализированными процессорами, реализованных с применением скоростного интерфейса FSL. Приводится пример реализации и результаты экспериментального исследования.

Ключевые слова: специализированный сопроцессор, программируемые логические интегральные схемы, интерфейс FSL.

Abstract. The article considers problems of construction of the integrated computers on the basis of FPGA with specialized processors, realized with application of FSL high-speed interface. The authors also give an example of realization and results of experimental research.

Key words: specialized co-processor, Field Programmable Gate Array, interface FSL.

Введение

Во многих приложениях с целью ускорения вычислений применяются специализированные блоки на основе программируемых логических интегральных схем [1–4]. В частности, на программируемых логических интегральных схемах (ПЛИС) фирмы Xilinx для повышения производительности могут добавляться специализированные процессоры, которые разрабатываются пользователем для конкретной задачи и подключаются к основному процессорному блоку. Для подключения таких блоков используется интерфейс Fast Simplex Link (FSL) [5], что дословно переводится как «быстрая простая связь». Он позволяет реализовать быструю связь между любыми двумя блоками разрабатываемой системы. В этом интерфейсе используется шина FSL Bus – однонаправленная двухточечная шина канала связи. Всего может быть создано до восьми каналов. Процессор общается с сопроцессорами с помощью специальных команд, предназначенных для работы с FSL, передает данные через буфер типа FIFO в блок сопроцессора, последний выполняет необходимые вычисления и возвращает результаты в процессор. Ускорение достигается за счет того, что сложные функции, требующие длительных вычислений, выполняются аппаратно в специализированном устройстве, структура которого разрабатывается пользователем для конкретного случая.

1. Разработка структуры вычислительных блоков

Типовая структура вычислительного блока с сопроцессором приведена на рис. 1. Процессор по специальным командам, предназначенным для работы с FSL, передает данные через буфер типа FIFO в блок пользователя (сопроцессор), последний выполняет необходимые вычисления и возвращает результаты в процессор через буфер.

Рассмотрим операции ввода-вывода со стороны сопроцессора. Временная диаграмма передачи данных из периферийного устройства (сопроцессо-

ра) на шину FSL показана на рис. 2. Передача производится по сигналу FSL_M_Write (в режиме ведущего). Передаваемые данные формируются на шине FSL_M_Data и сопровождаются единичным сигналом FSL_M_Write. На рис. 2 показана запись в основной процессор трех чисел D1, D2 и D3, после записи третьего числа приемный буфер переполняется (об этом сообщает сигнал FSL_M_Full, равный единице). Затем после двух тактов ожидания буфер освобождается и записывается число D4.

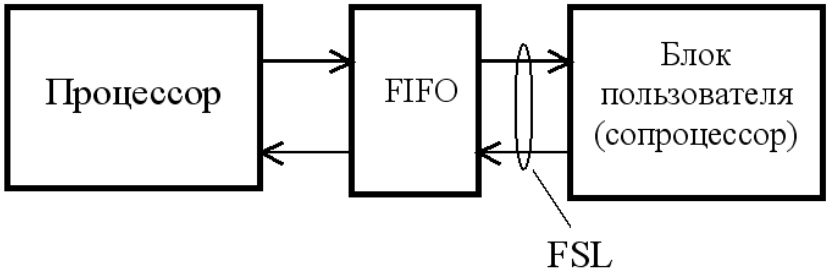

Рис. 1. Схема подключения сопроцессора

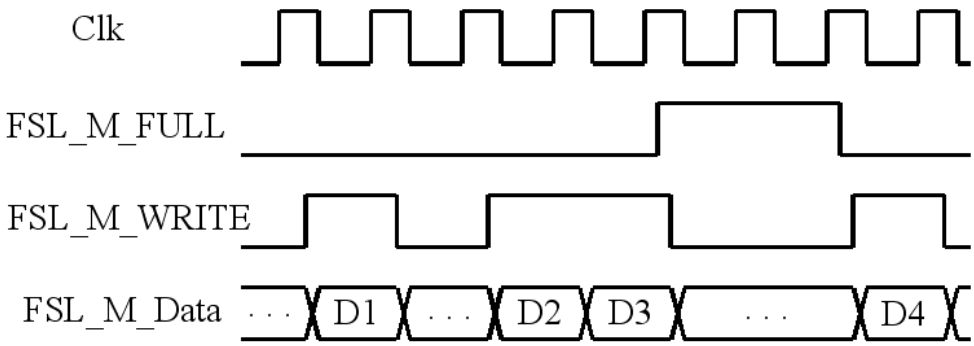

Рис. 2. Временные диаграммы обмена данными

Операция чтения данных в сопроцессор управляется сигналом FSL_S_Read (рис. 3). Когда данные в FSL на шине FSL_S_Data готовы, формируется сигнал FSL_S_Exists, равный единице, чтение выполняется в режиме ведомого. Чтение сопровождается сигналом FSL_S_Read, установленным в единицу в течение одного тактового цикла. Если сигнал FSL_S_Read задерживается, то задерживаются также сигналы FSL_S_Data и FSL_S_Exists.

Подключение сопроцессора с использованием интерфейса FSL производится в системе EDK (Embedded Development Kit), предназначенной для построения систем на ПЛИС с процессорными блоками. Сначала создается проект, включающий основной процессор на процессорном ядре MicroBlaze. Затем инициируется создание нового блока в режиме Create or Import Peripheral (создание или добавление периферийных устройств), при этом необходимо задать тип интерфейса – Fast Simplex Link (FSL). В поле Input FSL Interface задается количество входных данных, которые планируется передавать в создаваемый блок за один цикл работы. Этот параметр будет использоваться в дальнейшем при создании заготовки VHDL-описания. В поле Output FSL Interface задается количество данных результата, которые плани-

руется принимать из создаваемого блока за один цикл. После создания шаблона выполняется доработка схемы сопроцессора в соответствии с заданными требованиями.

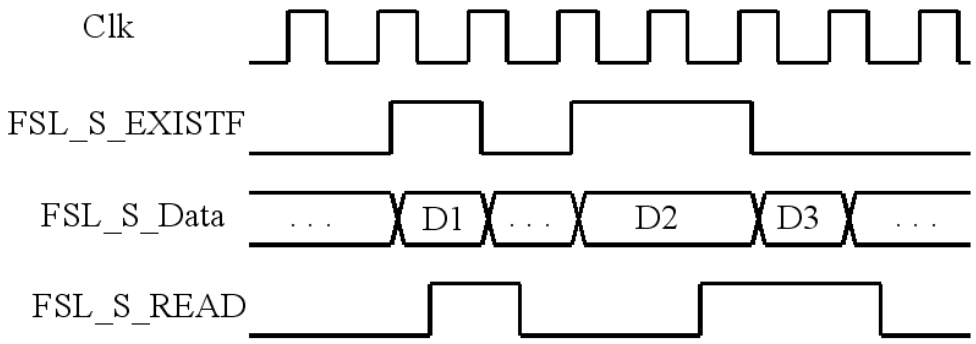

Рис. 3. Диаграммы обмена данными

На практике часто процессорное ядро входит в состав другой системы на правах подчиненного блока [4, 6]. Структура системы с подчиненным процессорным ядром приведена на рис. 4.

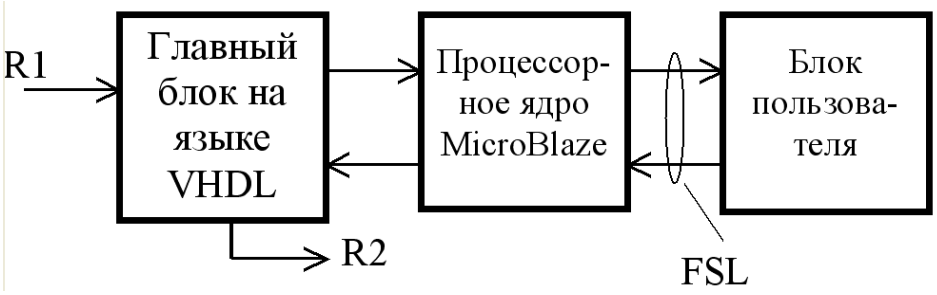

Рис. 4. Структура системы

На вход системы поступает сигнал R1, над ним выполняются необходимые вычисления, результат вычислений выдается на выход R2. В этой системе основным блоком является модуль на языке VHDL, процессорный блок на базе ядра MicroBlaze входит в состав системы как компонент. После создания процессорного блока к нему подключается пользовательский блок (сопроцессор) по интерфейсу FSL.

2. Разработка программного обеспечения

При разработке программного обеспечения для основного процессора использовались средства GNU C. Программа обработки позволяет принять входные данные с входа R1, передать их в сопроцессор, принять результаты из сопроцессора и выдать их на выход R2. Программа составлена для случая, когда в сопроцессор передается три слова и два слова принимаются обратно в основной процессор. Сопроцессор вычисляет сумму и выдает ее дважды. В качестве результата получается одно число, но для иллюстрации процессов обмена оно выдается два раза. Программа обработки имеет следующий вид:

```
#include "xparameters.h"
#include "xgpio.h"
#include "f2.h"
#define WRITE_F3_0(val)
  write_into_fsl(val, XPAR_FSL_F2_0_INPUT_SLOT_ID)
#define READ_F3_0(val)
  read_from_fsl(val, XPAR_FSL_F2_0_OUTPUT_SLOT_ID)

int main (void)
{
  int i, x, v;
  XGpio_mSetDataDirection(XPAR_LEDS_BASEADDR, 1, 0);
  i=0;
  while(1)
  {
    x=XGpio_mGetDataReg(XPAR_DIP_SWITCHES_BASEADDR, 1);
    for (i=0; i<3; i++) WRITE_F3_0(x);
    for (i=0; i<2; i++) READ_F3_0(v);
    XGpio_mSetDataReg(XPAR_LEDS_BASEADDR, 1, v);
  }
}
```

Файлы xparameters.h, xgpio.h и f2.h создаются системой проектирования EDK, в частности, в файле f2.h приводятся описания функций write_into_fsl и read_from_fsl, которые обеспечивают обмен информацией по интерфейсу FSL. Для обращения к FSL можно также использовать операторы putfsl(val, id) и getfsl(val, id). Функции XGpio_mGetDataReg и XGpio_mSetDataReg предназначены для приема входной информации и выдачи результатов на индикаторы.

Для отладки сопроцессора на первом этапе можно использовать отладчик GNU Debugger (GDB). На рис. 5 приведен фрагмент программы, показанный в GDB. На этом рисунке приведены операторы программы на языке C, ассемблерные команды и адреса этих команд.

```
    7 int main (void) {int i, x, v;
    8   XGpio_mSetDataDirection(XPAR_LEDS_BASEADDR, 1, 0);
0x168 <main>:                    imm     16386
0x16c <main+4>:                  swi     r0, r0, 4 // 0x40020004
    9   i=0;
   10   while(1)
   11     {x = XGpio_mGetDataReg (XPAR_DIP_SWITCHES_BASEADDR, 1);
0x170 <main+8>:                  imm     16384
0x174 <main+12>:                 lwi     r3, r0, 0 // 0x40000000
   12       for (i=0; i<3; i++)   WRITE_F3_0(x);
0x178 <main+16>:                 put     r3, rfsl0
0x17c <main+20>:                 put     r3, rfsl0
0x180 <main+24>:                 put     r3, rfsl0
   13       for (i=0; i<2; i++)   READ_F3_0(v);
0x184 <main+28>:                 get     r3, rfsl0
0x188 <main+32>:                 get     r3, rfsl0
   14       XGpio_mSetDataReg(XPAR_LEDS_BASEADDR, 1, v);
0x18c <main+36>:                 imm     16386
0x190 <main+40>:                 swi     r3, r0, 0 // 0x40020000
0x194 <main+44>:                 bri     -36     // 0x170 <main+8>
```

Рис. 5. Фрагмент программы в отладчике GDB

Для отладки можно также использовать имитационное моделирование. На рис. 6 приведены результаты моделирования, которые соответствуют рис. 2 и 3.

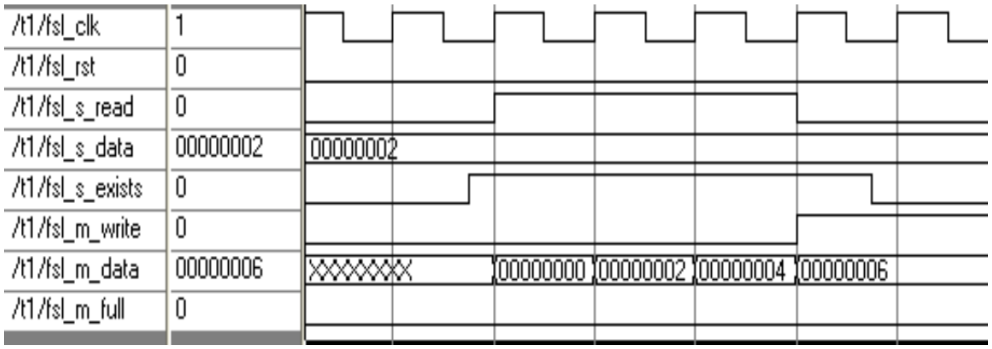

Рис. 6. Результаты моделирования

На вход в течение трех тактов (при fsl_s_read=1) поступает число 2, после трех сложений получается число 6, которое дважды передается в интерфейс FSL (при fsl_m_write=1).

3. Экспериментальное исследование

Экспериментальная проверка разработанного устройства производилась на отладочной плате Spartan-3E Starter Kit. Для отладки в проект был встроен логический анализатор ChipScope Pro фирмы Xilinx, который подключался к основному модулю. На рис. 7 приведены диаграммы работы системы, диаграмма получена с помощью логического анализатора.

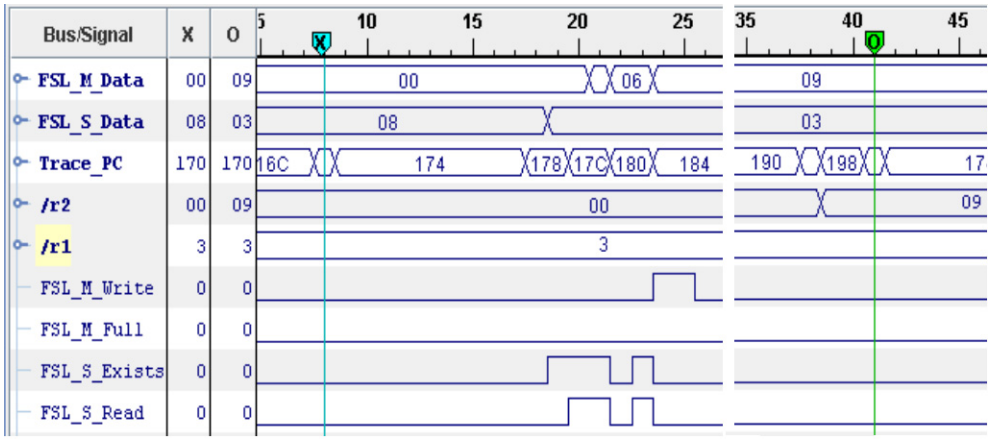

Рис. 7. Временные диаграммы работы системы

На рис. 7 сигнал r1 – входной сигнал для процессорного блока; r2 – выходной сигнал. На входе r1 постоянно присутствует 3, в момент t=18 это число из процессора выдается на FSL_S_Data. Именно это число трижды поступает в сопроцессор (пользовательский блок FSL). Результат FSL_M_Data=9 формируется в момент t=23, он дважды записывается в основной процессор. Из процессора этот результат выдается в момент t=38 на шине r2. Состояние

программного счетчика отражено сигналом Trace_PC и соответствует адресам команд на рис. 5.

Запуск производится по адресу Trace_PC=0x168 (0001_0110_1000 – шина инвертируется разрядами). Здесь сначала записывается два числа, потом через такт еще одно, после чего в ответ выдается два одинаковых числа.

Если без изменения аппаратуры изменить количество операций чтения:

```
for (i=0; i<3; i++) WRITE_F3_0(x);
for (i=0; i<3; i++) READ_F3_0(v);
```

то программа зависает: она ждет третьего числа, а аппаратура его не выдает.

Если уменьшить количество операций чтения:

```
for (i=0; i<3; i++) WRITE_F3_0(x);
for (i=0; i<1; i++) READ_F3_0(v);
```

то через некоторое время происходит переполнение буфера (рис. 8).

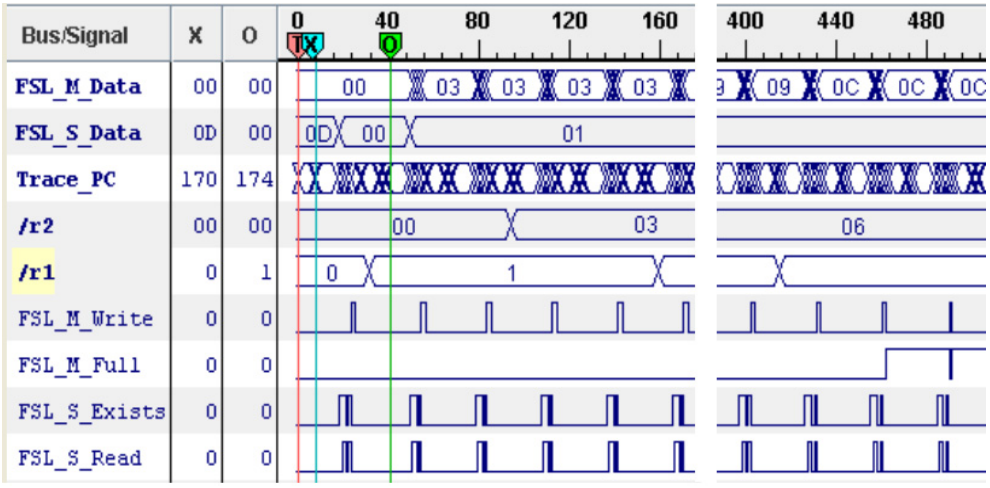

Рис. 8. Временные диаграммы работы системы

В каждом цикле программа считывает на одно число меньше, чем выдает аппаратура. В результате через 16 циклов (это емкость буфера FIFO) происходит переполнение буфера, о чем свидетельствует выработка FSL_M_FULL=1 на рис. 8.

Заключение

Разработаны структуры и программа обмена данными с сопроцессором для системы, содержащей основной процессор и специализированный процессор на основе программируемых логических интегральных схем, выполнено имитационное моделирование системы, произведена экспериментальная проверка работоспособности системы. Значимость полученных результатов для теории и практики построения систем на программируемых логических интегральных схемах с процессорными блоками состоит в том, что применение рассмотренных методов позволяет повысить быстродействие разрабатываемых систем.

Список литературы

1. Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation / ed. by Scott Hauck and Andrre DeHon. – Morgan Kaufmann Publishers, Elsevier Inc., 2008. – 908 p.
2. **Каляев, И. А.** Реконфигурируемые мультиконвейерные вычислительные структуры / И. А. Каляев, И. И. Левин, Е. А. Семерников, В. И. Шмойлов. – Ростов н/Д : Изд-во ЮНЦ РАН, 2008. – 398 с.
3. **Гурин, Е. И.** Спецпроцессор для решения задач определения диэлектрических и магнитных параметров материалов / Е. И. Гурин, Е. Е. Гришина // Новые информационные технологии и системы : тр. IX Междунар. науч.-техн. конф. – Пенза, 2010. – Ч. 1. – С. 169–176.
4. **Зотов, В. Ю.** Проектирование встраиваемых микропроцессорных систем на основе ПЛИС фирмы Xilinx / В. Ю. Зотов. – М. : Горячая линия-Телеком, 2006. – 520 с.
5. **Rosinger, H. P.** Connecting Customized IP to the MicroBlaze Soft Processor Using the Fast Simplex Link (FSL) Channel / Hans-Peter Rosinger. – Xilinx, XAPP529, 2004. – 12 p.
6. **Гурин, Е. И.** Построение систем на кристалле с подчиненным процессорным ядром Microblaze на ПЛИС фирмы Xilinx / Е. И. Гурин // Компоненты и технологии. – 2007. – № 9. – С. 115–118.

Гурин Евгений Иванович
доктор технических наук, профессор,
кафедра вычислительной техники,
Пензенский государственный университет

E-mail: gurin2@yandex.ru

Gurin Evgeny Ivanovich
Doctor of engineering sciences, professor,
sub-department of computer engineering,
Penza State University

Огнев Иван Васильевич
доктор технических наук, профессор,
кафедра вычислительной техники,
Московский энергетический институт
(технический университет)

E-mail: gurin2@yandex.ru

Ognev Ivan Vasilyevich
Doctor of engineering sciences, professor,
sub-department of computer engineering,
Moscow Institute of Energetics
(technical university)

УДК 681.327

Гурин, Е. И.

Построение вычислительных блоков на основе программируемых логических интегральных схем со специализированными сопроцессорами / Е. И. Гурин, И. В. Огнев // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2012. – № 1 (21). – С. 65–71.