Санкт-Петербургский Государственный Университет

Физический факультет

Вечернее отделение, 5 курс

Макарченко Иван Павлович

Программирование логических микросхем

РЕФЕРАТ

Санкт-Петербург

февраль 2009

Введение

Современная микроэлектроника делает гигантские шаги в своем развитии. Одним из наиболее интенсивно развивающихся направлений является развитие и использование программируемых логических интегральных схем – ПЛИС. В русскоязычной литературе этот класс микросхем не редко называют программируемой логикой (ПЛ), поэтому в настоящее время, кроме аббревиатуры ПЛИС широко распространены сокращения ПЛ и СБИС ПЛ.

Прародители современных ПЛИС появились в 70-х годах прошлого столетия, и в то время, в силу своей матричной структуры, они назывались программируемыми логическими матрицами – ПЛМ, из-за чего этот термин перекочевал и в современный язык, где иногда употребляется в среде схемотехников, как сленговое название для ПЛИС, потому что аббревиатуру ПЛМ можно расшифровывать и как программируемая логическая микросхема. Тем не менее, правильными терминами следует считать аббревиатуры ПЛИС и СБИС ПЛ. Еще один устаревший термин, используемый для программируемой логики – ПМЛ – программируемая матричная логика, используется для обозначения устаревших микросхем, явившихся промежуточным звеном между ПЛМ и современными большими ПЛИС.

Вторым прародителем ПЛИС следует признать микросхемы базовых матричных кристаллов – БМК, типичных для 60–70-х годов прошлого столетия. БМК программировались с помощью технологических фотолитографических матриц, что во времена их использования было достаточно прогрессивно, но к настоящеему времени потеряло актуальность.

В англоязычной литературе микросхемы ПЛИС разделяются на несколько достаточно больших классов, имеющих разные аббревиатуры. Первые микросхемы программируемой логики назывались Programmable Logic Device (PLD), затем появились Programmable Array Logic (PAL) и Gate Array Logic (GAL). В настоящее время эти классы микросхем переродились в CPLD (Complex Programmable Logic Device), которые по сравнению с современными большими ПЛИС уже нельзя назвать сложными (complex) или большими, потому что сложность миросхем ПЛ второго крупнейшего современного класса – Field Programmable Gate Array (FPGA) – давно перешагнула мечты инженеров 80-х, когда FPGA только зарождались.

Заканчивая краткий обзор типов микросхем программируемой логики следует отметить существование еще одного класса микросхем, тесно связанных с ПЛИС. Это ASIC (Application Specific Integrated Circuit) – микросхемы так называемого класса HardCopy, которые проектируются в ПЛИС, а затем перепроектируются (производи-

телем на основе кристаллов своих ПЛИС и предоставленным заказчиком описанием схемы на специальном языке описания аппаратуры – HDL – hardware design language) для выпуска крупными тиражами, что позволяет уменьшить стоимость и потребляемую мощность, а так же увеличить быстродействие ASIC микросхемы по отношению к прототипу на FPGA.

Первые микросхемы программируемой логики выпускались фирмами Intel, AMD, Lattce. В 1983 году подразделение фирмы Intel, занимавшееся программируемой логикой, выделилось в отдельную фирму, выросшую к настоящему времени в ALTERA Co., являющуюся одним из лидеров производителей ПЛИС. Вторым (по алфавиту, но не по состоянию дел) лидером производителей ПЛИС следует назвать фирму Xilinx Inc., появившуюся в 1984 году. В настоящее время ALTERA и Xilinx являются неоспоримыми лидерами в области разработок и производства ПЛИС, и развиваются, как говорится «ноздря в ноздрю», практически не отставая и не перегоняя друг от друга.

Иные известные фирмы, производящие ПЛИС – Atmel, Actel, Lattice – занимают доли процентов на мировом рынке ПЛИС. Менее известны – Exel, Signetics.

Выбор, какие микросхемы ведущих производителей использовать в разработках (ALTERA или Xilinx), достаточно сложен, и делается более из соображений экономического и человеческого планов, нежели по параметрам ПЛИС, потому что по параметрам новейшие серии ПЛИС обоих лидеров, «примерно» одинаковы. Слово «примерно» в кавычках, потому что точное сравнение невозможно из-за различия идеологий построения ПЛИС у ALTERA и Xilinx. Тем не менее, различие этих идеологий до сих пор не привело к естественному отбору лучшей, потому что обе фирмы ведут достаточно целеустремленную политику по удерживанию пользователей своей продукции. Конкурентная борьба на примере фирм ALTERA и Xilinx может служить показательным примером положительного влияния конкурентности в капиталистическом обществе. Именно она – причина того, что мы имеем сейчас в своем распоряжении достаточно мощные ПЛИС по относительно низким ценам и, вместе с тем, совершенно нетипичное в капиталистическом обществе бесплатное программное обеспечение для разработки схем на ПЛИС, которое свободно распространяется через internet с сайтов $http://www.altera.com$ и $http://www.xilinx.com$.

Отечественные микросхемы программируемой логики, являвшиеся аналогами некоторых ПЛМ фирмы Intel, выпускались еще в доперестроечное время (например: К556РТ1, К556РТ2, КМ1556ХП4, ХП6, ХП8, ХЛ8). В настоящее время отечественная промышленность микросхемы программируемой логики не выпускает. Есть только непроверенные слухи, о том, что где-то появились отечественные аналоги ПЛИС

фирмы ALTERA из серии FLEX10K, что не вызывает энтузиазма у схемотехников, потому что эта серия – не более чем вчерашний день технологии ПЛИС.

Старые ПЛМ (PLD) и множество устаревших серий ПЛИС, относящихся к CPLD и FPGA, в современных разработках практически не используются, подобно тому, как не используются микросхемы средней и малой степени интеграции. Причина этого в том, что ПЛИС неуклонно развиваются (согласно известному закону Мура) и достигли такого уровня, когда в одном корпусе возможно размещение огромных схем, которые элементной базе 80-х занимали бы целые шкафы и даже комнаты. Кроме того, развитие компьютерной техники позволяет вести разработку схем для ПЛИС с помощью мощных САПР, кардинально ускоряющих и облегчающих разработку проектов, на которые в прежние времена уходили месяцы и годы работы огромных коллективов разработчиков.

Логические объемы современных ПЛИС давно перешагнули миллионный рубеж в эквивалентных вентилях (элементарных функциях 2И-НЕ), и в настоящее время подходят к милионному рубежу в логических ячейках (логическая ячейка – минимальная структура FPGA, состоящая из 4-входового логического элемента, мультиплексора, D-тригера и схем управления). Это позволяет реализовывать на них сложнейшие схемы цифровой обработки сигналов, специализированные вычислители, процессоры.

Для примера приведем параметры ПЛИС современной серии Cyclone-III (фирмы ALTERA) очень скромного объема EP3C25F324C6:

- 25000 логических ячеек (LE) ($\approx$ 250000 эквивалентных вентилей)
- 66 блоков внутренней памяти M9K (0.6 мегабит)
- 16 встроенных умножителей 18x18бит
- 4 блока PLL (ФАПЧ)
- 214 выводов ввода-вывода.

И подсчитаем предельное число математических и логических функций, которые можно реализовать в такой ПЛИС. Для вычислений используются ресурсы логических ячеек и встроенных умножителей. Скорость работы умножителей определяется предельной тактовой частотой, с которой работают умножители в ПЛИС. Для выбранной ПЛИС она составляет 250MHz, и это означает, что ПЛИС способна выполнить до $66 * 250 \approx 16.5$ миллиардов целочисленых умножений в секунду. Логические ячейки позволяют реализовывать более простые функции – сумматоры и логические функции, требуя, как правило, по одной LE на разряд. Считая обрабатываемые числа 32-хразрядными (т.е. на одну операцию требуется 32 ячейки) и имеющиеся 25000 ячеек, получаем, что рассматриваемой ПЛИС достаточно для реализации до

≈ 800 параллельно работающих 32-хразрядных функций, что при тактовой частоте 250MHz соответствует $2 \cdot 10^{11}$ операций в секунду.

Разумеется, эти числа являются предельными и реально недостижимы по объективным причинам. Каждая схема для ПЛИС характеризуется некоторым параметром, называемым «разводимость схемы». Разводимость определяет, можно ли физически разместить (развести) схему в конкретной ПЛИС? Попытка развести схему, требующую 100% ресурсов ПЛИС, чаще всего заканчивается неудачей разводчика. Например, САПР Quartus II в подобном случае и в случае, если количество требующихся ресурсов достигает 80% объема конкретного чипа, рекомендует использовать для проекта ПЛИС большего объема.

Главными направлениями использования ПЛИС являются:

1. Проектирование систем Цифровой обработки Сигналов – ЦОС (Digital Signal Processing – DSP);
2. Проектирование систем-на-кристалле (System on Chip – SoC);
3. Прототипирование (prototyping) – создание прототипов для ASIC.

Применительно к Физике, ПЛИС могут использоваться для решения задач автоматизации и управления экспериментами, задач обработки результатов экспериментов в реальном времени, а так же для решения задач математической и вычислительной физики.

Современный способ описания схем для программируемой логики

Появление СБИС ПЛ высокой логической емкости, развитие средств автоматизации проектирования и уровень сложности создаваемых цифровых систем предопределили существенные изменения в методологии проектирования схем.

Так широкое применение графического описания проектируемого устройства, базирующееся на ручном и в большинстве случаев неформализованном (эмпирическом) синтезе управляющих автоматов, остается в прошлом. На смену приходит другая методология, основанная на двух ключевых моментах:

- Текстовое описание – применение высокоуровневых языковых конструкций (языков описания аппаратуры – Hardware Design Language) для задания алгоритма работы создаваемого устройства.
- Автоматический синтез – процедура машинного перевода текстового описания в схемное описание на заданном элсментном базисе, выполняемая с помощью систем автоматизированного проектирования (САПР).

Применение этой методологии позволяет не только ускорить и удешевить создание цифрового устройства, но и добиться принципиально более высокой степени

проработки проекта. В связи с этим разработчику аппаратуры становится принципиально необходимо знать языки описания аппаратуры и соответствующие системы автоматизированного проектирования.

В настоящее время существует множество языков описания аппаратуры, часть из которых универсальны (VHDL, Verilog), часть подходит только для САПР конкретной фирмы (AHDL – ALTERA). Выбор HDL для последующего изучения зависит от преследуемых целей и выбора ПЛИС, которые будут использоваться в проекте. Если выбор ПЛИС еще не сделан, лучше изучать универсальный язык VHDL или Verilog. Если выбор уже сделан в пользу ПЛИС фирмы ALTERA (например, заказчиком), то можно изучать AHDL, как наиболее простой язык для проектирования схем в ПЛИС этой фирмы. Тем не менее, даже сама ALTERA рекомендует изучать для проектирования схем VHDL. VHDL так же используется и для проектирования схем в ПЛИС фирмы Xilinx, поэтому дальше в реферате речь пойдет о VHDL, хотя почти все сказанное о языке в принципиальном плане, относится и к другим языкам описания аппаратуры. Соотношение наиболее распространенных языков по сложности языковых конструкций и изучения можно проиллюстрировать их сравнением с традиционными языками программирования:

- AHDL – Паскаль;
- VHDL – Ада, Алгол;
- Verilog – Си.

Расшифровка аббревиатуры VHDL имеет необычный вид. Первая буква V – означает сокращение от другой аббревиатуры – VHSIC – Very High Speed Integrated Circuit. Остальные три – HDL – Hardware Design Language.

История VHDL начинается с 1980 года, когда американский департамент обороны основал первый стандарт на HDL для програмы моделирования высокоскоростных микросхем VHSIC. В 1987 году американский Institute of Electrical and Electronics Engineers (IEEE) ратифицировал стандарт для VHDL – IEEE Standard 1076. И, в 1993 году стандарт на VHDL был обновлен до IEEE 1076'93.

Основные концепции VHDL

1. VHDL используется и для описания проектируемого модуля, и для описания тестов, с помощью которых производится верификация устройства или модуля. При первом знакомстве с VHDL инженера более интересует возможности схемотехнического описания модулей.

2. Способы описания – поведенческое – описание алгоритма работы устройства, –

структурное – описание проектируемого модуля в виде взаимосвязанных компонентов более низкого уровня в иерархии описаний, и смешанное – смесь поведенческого и структурного описания.

3. Параллельность выполнения операторов. Этот пункт существенно отличает любой язык описания аппаратуры от традиционных языков процедурного программирования. Приниципиальное отличие заключается в том, что все элементы описываемой аппаратуры работают одновременно, тогда как в традиционной программе, все описываемые действия выполняются последовательно. Это означает, что записи на VHDL вида:

```
A<=not(B and R); B<=not(A and S and C);
```

и

```
B<=not(A and S and C); A<=not(B and R);
```

совершенно эквивалентны, независимо от того, какой из операторов написан первым, а какой вторым.

Оператор <= в VHDL является оператором соединения сигналов. Запись A<=C – означает, что сигнальный провод C присоединен к сигнальному проводу A. При этом обратного соединения не происходит, то есть сигнал с C попадает на A словно через буфер с нулевым временем прохождения сигнала, но с A на C – сигнал не попадает, если только рядом нет оператора соединения в обратную сторону. При переложении описания в схему этот буфер, разумеется, будет иметь ненулевое время срабатывания, но для функционального описания обычно не требуется учитывать время распространения сигнала в логическом элементе. Оператор and – означает логическую функцию «И», not – логическое отрицание, т.е. в описанной схеме на провод A подсоединен выход логического элемента 2И-НЕ, на входы которого поданы сигналы B и R. В то же время на провод B подсоединен выход логического элемента 3И-НЕ, на входы которого поданы сигналы A, S и C. Схемотехнически эти строчки описывают простейший RS-тригер с одним входом сброса R и двумя входами установки S и C.

В принципе, вся логика могла бы быть выражена через элементы 2И-НЕ, однако перспектива описывать все схемы через базис Шеффера никого не прельщает (хотя и может быть интересна в академическом плане), поэтому в VHDL существуют:

4. Возможность работы с библиотеками. В библиотеках описываются стандартные элементы, применяемые для описания схем. Так же возможно создание библиотек пользователя, для их последующего использования в описаниях более высокого уровня.

5. Создание конфигурируемых (параметризованных) описаний. Конфигурируемое описание позволяет созданный однажды элемент использовать множество раз с разными параметрами. Например, при описании регистра его разрядность удобно задать параметром, а не конкретным числом, чтобы затем использовать это же описание для другого регистра не повторяя его описание с другим числом разрядов, а просто указав в параметре нужное число.

6. Повторное использование созданных компонентов. Каждый созданный компонент может использоваться в описании схемы столько раз, сколько нужно для схемы. Это позволяет создавать удобные и наглядные описания схем.

7. Возможность работы с файлами. Совершенно естественно (для современного языка), что части схем могут описываться раздельно в нескольких файлах. Это позволяет работать над проектом параллельно целому коллективу и избавляет разработчика от постоянного листания длинных исходных текстов в поисках места, куда надо внести очередные изменения.

8. Возможность переопределения операторов. Чем выше иерархия тем более высокоуровневые операции требуются для описания схем. Создав, например, однажды для некой схемы тип сигнала complex с нужной разрядностью, разработчик пожелает следующим шагом для таких сигналов задать возможность их суммирования, умножения, деления, делая это стандартными операторами. VHDL позволяет переопределить для своих типов операции, задаваемые стандартными элементами языка.

Выразительность языков описания аппаратуры и возможности современных САПР таковы, что опытный инженер способен в одиночку создавать и отлаживать схемы, на проектирование которых раньше потребовалась бы работа огромных коллективов, и выполнять такую работу не месяцами и годами, а за дни и недели. Разумеется, эти дни и недели предваряются годами опыта работы в сфере проектирования схем с использованием соответствующих САПР.

Заключение

Развитие программируемой логики неуклонно продолжается. Буквально несколько дней назад («ноздря в ноздрю» 10–11 февраля 2009 года) оба гиганта производителя ПЛИС сделали заявления о появлении новых серий ПЛИС, созданых по новейшей 40–45нм технологии. ALTERA объявила о начале выпуска серий Stratix IV GT и Arria II GX; Xilinx объявила о выпуске серий Spartan-6 и Virtex-6.

Вот заявленые параметры самых крупных микросхем:

Фирма	Микрохема	Число логических ячеек	Объем встроенной памяти	Число Встроенных умножителей
Xilinx	XC6VLX760	758764	25Mbit	864
Xilinx	XC6VSX475	476160	38Mbit	2016
ALTERA	EP4SGX540	531200	27Mbit	1024
ALTERA	EP4SE680	671100	31Mbit	1360

Работая с тактовыми частотами свыше 500MHz эти чипы позволят достичь предельных вычислительных мощностей на один чип до 10^{12} целочисленных умножений в секунду и до 10^{14} арифметическо-логических операций в секунду, что не под силу самым современным процессорам общего и специального назначения.

Отсюда можно сделать вывод, что в ближайшее время ПЛИС будут играть огромную роль в развитии науки и техники. И, быть может, дадут зеленый свет многим новым направлениям вычислительной науки за счет достижения высочайших предельных скоростей.

Литература:

1. ***Стешенко В.Б.*** *ПЛИС фирмы ALTERA: проектирование устройств обработки сигналов. – М.: ДОДЭКА, 2000.*

2. ***Антонов А.П.*** *Язык описания цифровых устройств AlteraHDL. – М.: РадиоСофт, 2001.*

3. ***Тарасов И.Е.*** *Разработка цифровых устройств на основе ПЛИС* Xilinx® *с применеием языка VHDL. – М.: Горячая линия – Телеком, 2005*

5. ***Стешенко В.Б.*** *ПЛИС фирмы ALTERA: элементная база,система проектирования и языки описания аппаратуры. 3-е изд. – М.: Додэка-XXI, 2007.*

5. ***Потехин Д.С., Тарасов И.Е.*** *Разработка систем цифровой обработки сигналов на базе ПЛИС. – М.: Горячая линия–Телеком, 2007.*

6. *Материалы* ***ALTERA Approved Traning course****, организованных при СПбГТУ фирмой «ЭФО», 2008.*

7. *Материалы с сайтов* ***http://www.altera.com*** *и* ***http://www.xilinx.com***

Реферат подготовлен с помощью пакета LaTeX.