# Содержание

# *Предисловие*

**Военное дело просто и вполне доступно здравому уму человека. Но воевать сложно.**
***Карл фон Клаузевиц***

**Кто мешает тебе выдумать порох непромокаемый?**
***Козьма Прутков***

С момента появления первого микропроцессора (I8080, будем так считать) всегда казалось, что новые конструкции, сильно отличающиеся от лидера в этой области, серьезной конкуренции ему не составят. Конечно, все время появлялись новые микропроцессоры, они даже конкурировали с лидером (например, M68xx и x86) на протяжении некоторого периода, но затем отходили на периферию внимания потребителей. И тем не менее все время появляются новые микропроцессоры, и некоторым из них удается завоевать основные позиции в какой-либо области применения. Например, это процессоры ARM, которые сначала заняли нишу в области высокопроизводительных контроллеров, затем мобильных приложений, а теперь уже теснят x86 в его основной сфере применения вплоть до серверных приложений. Долгое время казалось, что процессорам ARM ничего не угрожает, его «одноклассники» (например, MIPS) сильно ему уступают по распространенности. Но опять же не так давно появились процессоры с новой архитектурой – RISC V – которые уже составляют архитектуре ARM конкуренцию. Это ситуация в топовых приложениях.

Наряду с ними существует масса микропроцессоров совсем не таких мощных, как упомянутые выше, которые используются в более специфических приложениях – микроконтроллерах, то есть микросхемах, предназначенных для выполнения задач управления, где не нужна большая вычислительная мощность, а иногда ценятся совсем другие качества – минимальное энергопотребление, компактность, низкая стоимость. В самом деле, для управления хлебопечкой не требуется много вычислять, а для использования в качестве счетчика воды или электроэнергии нужна небольшая потребляемая мощность, чтобы контроллер мог работать от батареи достаточно большое время, и так далее.

Причины появления новых архитектур могут быть разные.

Это может быть действительно новое слово в области построения процессоров (RISC -архитектуры), желание обойти лицензионные ограничения уже имеющихся конструкций (RISC V), использование в специфической области (DSP - архитектуры), требование повышения производительности по сравнению с традиционными процессорами, тем не менее, не выходящими далеко за границы известных архитектуры (WLIW - архитектуры) и так далее. Перечисленные факторы и сейчас не утратили свою актуальность.

Все сказанное говорит о том, что нужда в новых микропроцессорах разных архитектур постоянна, и задача разработки микропроцессоров вполне актуальна. Это относится как к простым микропроцессорам, так и к устройствам высокой вычислительной мощностью класса ARM, Pentium, MIPS и других.

В этой книге дается необходимый базис, позволяющий выполнить разработку микропроцессора небольшому коллективу, или даже отдельному разработчику – для этого вовсе не надо быть супераппаратчиком (введем здесь такое понятие, аналогичное суперпрограммисту в области разработки аппаратуры).

Обычно алгоритмы в литературе по программированию и теории алгоритмов приводятся на псевдокоде или псевдо-Алголе. Но когда речь идет об аппаратуре, процедурных псевдоязыков оказывается недостаточно: процессы в аппаратуре выполняются параллельно, а такие псевдоязыки отсутствуют; поэтому почти все алгоритмы и описания работы аппаратуры, приведенные в книге, написаны на языке VHDL. Дополнительным аргументом в пользу этого решения является то, что при желании их можно воспроизвести, промоделировать, синтезировать и, самое главное, модифицировать для получения лучших результатов.

Все характеристики алгоритмов, такие, как площадь и время выполнения функции, получены синтезом алгоритмов с использованием одной и той же библиотеки элементов и одного САПР, что позволяет сравнивать алгоритмы как вдоль, так и поперек, то есть сравнивать алгоритмы одной группы и делать оценку разных групп и даже отдельных алгоритмов из разных групп сравнением их между собой.

В двух приложениях приведены сведения из теории автоматов и теории алгоритмов.

Сведения из теории автоматов переработаны автором для единообразия и приведены в том виде, который удобен разработчикам аппаратуры (если речь идет об конечных автоматах, то они довольно привычны для аппаратчиков; автоматы с магазинной памятью обычно используются только при разработке программного обеспечения, компиляторов в первую очередь).

**Стратегические просчеты не могут быть компенсированы тактическими средствами.**

***Карл фон Клаузевиц***

# Разработка архитектуры и структуры процессора

В программировании, как известно, есть два подхода к проектированию – нисходящий и восходящий.

Нисходящий предполагает, что сначала разрабатывается структура программного обеспечение на самом высоком уровне, то есть определяется, какие именно функции оно должно выполнять, проясняется взаимодействие компонентов программного обеспечения на самых высоких уровнях иерархии, а затем каждый компонент разбивается на отдельные задачи и т.д., до тех пор, пока не будут разработаны все компоненты программы. Этот подход также называется иерархическим.

Восходящий подход основан на обратной процедуре: из имеющихся примитивов (готовых программ) собираются более крупные модули, которые по мере укрупнения образуют структуру разрабатываемой программы.

На практике, однако, в чистом виде реализация обоих подходов встречается редко.

Аналогично, можно и к проектированию аппаратуры, в том числе и процессоров, использовать оба подхода, тем не менее, нисходящий подход представляется гораздо более предпочтительным. Его использование уже на начальном этапе позволяет определить качество проекта. Разумеется, при этом разработчик должен представлять способы реализации тех блоков, из которых состоит проект, и их основные характеристики, которые важны для определения качества проекта.

К примеру, если взять некоторую функцию, которую должен выполнять процессор и не знать при этом, какие при ее конкретной реализации будут получены максимальная тактовая частота и площадь на кристалле, можно получить в целом неудовлетворительный результат, если эти характеристики для оценки проекта существенны. При выборе же восходящего подхода очень легко получить результат, который не будет удовлетворять ни одному требованию, предъявляемому к проекту.

Если имеется определенность, какой именно процессор (его назначение, основные свойства) необходимо разработать, начинать следует с разработки архитектуры. Это важнейший этап, результаты которого будут влиять на весь процесс проектирования и в конечном счете определят качество всего изделия в целом. Но просто разработать архитектуру, то есть определить состав и взаимодействие блоков, недостаточно, надо еще и осуществить «нарезку» прохождения данных по тактам работы процессора. Здесь, говоря о данных, подразумеваем как данные в общепринятом смысле, так и команды, выполняемые процессором. Попробуем теперь определить последовательность действий при определении архитектуры процессора по шагам.

1. Проектирование системы команд процессора – определение ISA (instruction set architecture);
   - определение набора команд процессора
   - проектирование топологии системы команд процессора
2. Анализ системы команд процессора;
3. Распределение команд по группам;
4. Определение поблочного состава процессора, обеспечивающего прохождение команд и данных;
5. «Нарезка» прохождения команд и данных по тактам с учетом блочной структуры процессора;
6. Разработка архитектуры АЛУ – арифметико-логического устройства;
7. разработка отдельных блоков процессора.
8. Моделирование работы процессора, отладка блоков и процессора в целом.

Пункты 1 и 2 в некотором смысле взаимоисключающие: пункт 1 используется в случае проектирования процессора «from scratch», то есть, с нуля, начиная с системы команд, а пункт 2 – для проектирования процессора с существующей системой команд. Проще говоря, пункт 1 нужен, когда проектируется новый процессор, а пункт 2 – когда реализуется уже известная система команд или даже уже известный процессор.

Целью анализа системы команд является уточнение архитектуры системы команд. Здесь требуется определить близкие по используемой аппаратуре или по назначению команды с целью дальнейшей группировки команд. Конечная цель группировки – минимизация числа блоков, из которых будет состоять процессор. Пример такой группировки – определение арифметических и логических команд, которые будут исполняться в одном блоке – АЛУ. Это, кстати, пример грубой группи-

ровки, так как уже в АЛУ команды надо будет группировать более тонко, хотя цель остается прежней – минимизация числа блоков. Понятно, минимизация количества используемых блоков также не является самоцелью. Истинная цель – обеспечить высокую тактовую частоту и малую площадь аппаратуры процессора. Представляется, что минимальное число блоков в некоторой степени обеспечивает эту задачу.

Далее определяются необходимые блоки, которые будут обеспечивать правильное функционирование процессора. Конечный пункт проектирования – разбиение диаграммы работы процессора по тактам. Если процессор конвейеризован – современные процессоры исполняют только такими – то задача формулируется так: разработать конвейер процессора.

Пункты 6 и 7 – собственно реализация нисходящего подхода, разбивка каждого блока на отдельные устройства (АЛУ тут выделен, так как здесь заранее ясно, что оно будет состоять из нескольких блоков) и реализация этих устройств на языках проектирования аппаратуры.

Далее, пункт 8 предполагает разработку набора тестов системы команд процессора и работы процессора с внешними сигналами (прерываниями и другими сигналами, внешними по отношению к работе процессора).

Отметим, что в общем случае разработка процессора имеет итерационный характер; в зависимости, например, от пункта 8 может потребоваться перепроектирование некоторых блоков процессора, или как минимум, их коррекция. Процессор – устройство, как правило, сложное, и одной итерации для реализации процессора с нужными характеристиками бывает недостаточно.

Чем лучше была разработана архитектура на верхнем уровне, тем меньше потребуется таких итераций; если архитектура разработана неудовлетворительно, никакая самая прекрасная реализация отдельных блоков не позволит достичь хороших характеристик (высокой тактовой частоты и /или небольшой площади) процессора в целом.

Обозначим набор блоков, в некотором роде типовой, которые имеются практически в каждом процессоре:

- дешифратор команд;
- блок управления центральным процессором;
- арифметико-логическое устройство;
- блок управления переходами;
- регистровый файл;
- интерфейс команд;
- интерфейс данных.