Вопросы производительности, количества ресурсов, задержки и максимальной тактовой частоты

Вся гибкость FPGA замечательно проявляется в том, что разработчик практически не ограничен в архитектуре устройства, которое он разрабатывает. В свою очередь именно архитектура определяет такие важные характеристики цифрового устройства, как производительность, количество задействованных ресурсов и задержка.

Высокая производительность vs Количество ресурсов

Под производительностью цифрового устройства мы будем понимать количество битов или количество *N*-разрядных слов, которое оно обрабатывает за единицу времени. В качестве таковой обычно принимают один такт сигнала тактовой частоты. Например, производительность декодера помехоустойчивого кода 1 бит за 1 такт и на тактовой частоте 400 МГц такой декодер будет уметь обрабатывать данные, поступающие на вход со скоростью до 400 Мбит/с. Или, например, производительность фильтра – 0,1 16-разрядного отсчета за такт. Такой фильтр сможет на частоте 500 МГц обрабатывать 16-разрядные отсчеты, следующие с частотой дискретизации не выше 50 МГц.

При рассмотрении устройства с позиций его производительности не важно, с какой задержкой появляются данные на выходе цифрового устройства (рис. 1). Например, производительность блока быстрого преобразования Фурье (БПФ) может быть один отсчет на такт, но при этом задержка может достигать сотен и даже тысяч тактов.

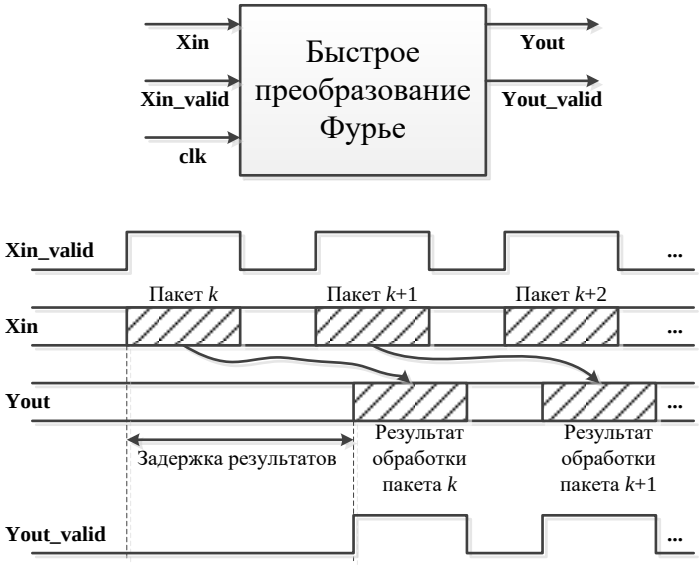

Рис. 1. Пример задержки результатов на выходе устройства. Блок БПФ выдает результаты обработки с некоторой большой задержкой, но далее результаты в среднем появляются с постоянной скоростью

Классическое решение задачи повышения производительности известно со времен мануфактур: разбить весь процесс на цепочку подпроцессов таким образом, что результаты одного процесса используются на входе следующего. Применительно к цифровым устройствам такой подход называется конвейеризацией. Важнейшим свойством конвейера является то, что он может начинать обработку новых входных данных, не дожидаясь окончания обработки предыдущих. С точки зрения реализации конвейер предполагает “развертывание” циклов.

Рассмотрим в качестве примера операцию возведения числа x в степень 5. Для определенности будем считать, что операция умножения выполняется за один такт. Тогда реализация x^5 в цикле (рис. 2) будет выглядеть так: $x \cdot x \cdot x \cdot x \cdot x$.

В псевдокоде описанная операция будет выглядеть так (без округлений и насыщений):

```
Result = 1;
for (k = 0; k < 5; k = k+1)
     Result = Result * x;
end for
```

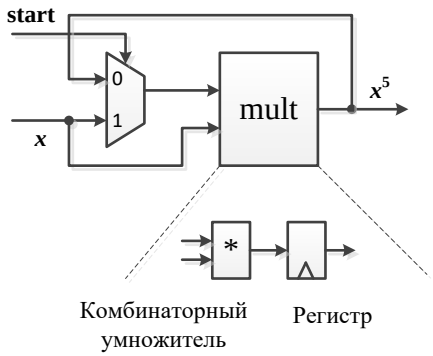

Рис. 2. Пример реализации операции x^5 в цикле

В схеме на рис. 2 расчеты для нового x нельзя начинать, пока не завершены текущие вычисления. Более того необходимы специальные сигналы, указывающие на старт и окончание вычислений. Реализация на рис. 2 требует 1 умножитель, выполняется за 4 такта и имеет производительность один отсчет за 4 такт.

Для большей производительности мы хотим иметь возможность на каждый такт подавать на вход новое число и не ждать окончания вычислений. Для этого разобьём операцию возведения в степень на несколько простых однотипных операций и соберем из них конвейер. На рис. 3 представлена конвейерная реализация на основе линии задержки и четырех последовательно расположенных умножителей, а на рис. 4 – конвейер на основе трех каскадов из четырех умножителей. В первом случае операция возведения в степень занимает 4 такта, во втором – 3 такта. Производительность обоих решений равна 16 битов за такт при задержке 4 и 3 такта соответственно.

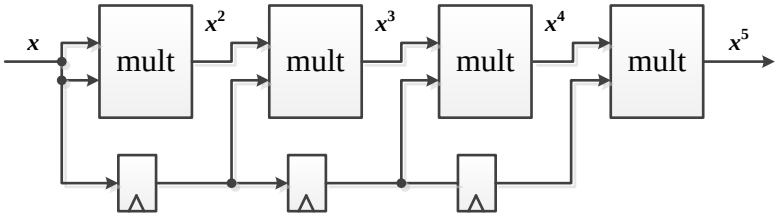

Рис. 3. Конвейерная реализация операции x^5 на основе четырех последовательных умножителей

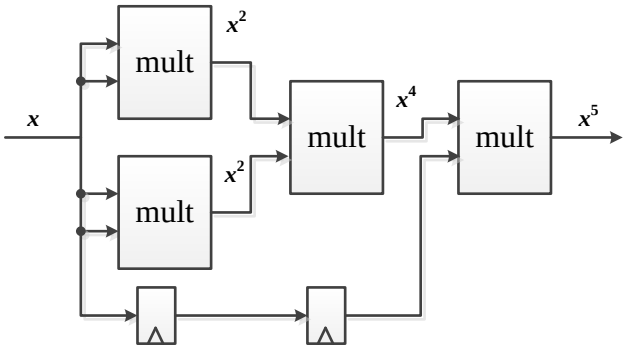

Рис. 4. Конвейерная реализация операции x^5 на основе трех каскадов четырех умножителей

Таким образом, в обмен на увеличение количества задействованных ресурсов производительность устройства повысилась в разы.

Справедлив и обратный подход: в случаях, когда не нужна высокая производительность один и тот же ресурс необходимо задействовать для выполнения разных операций. Классическим примером такого подхода является последовательно-параллельная реализация фильтров с конечной импульсной характеристикой (рис. 5). На рис. 5 блок MAC (Multiply Accumulate) выполняет операцию умножения с накоплением – умножение входного отсчета на коэффициент фильтра и сложения результата с предыдущей частичной суммой. Если отсутствует необходимость в высокой производительности (частота следования отсчетов в разы меньше тактовой частоты), то один ресурс (ячейка фильтра, включающая умножитель и сумматор) можно использовать многократно. И даже свести весь фильтр к одной ячейке!

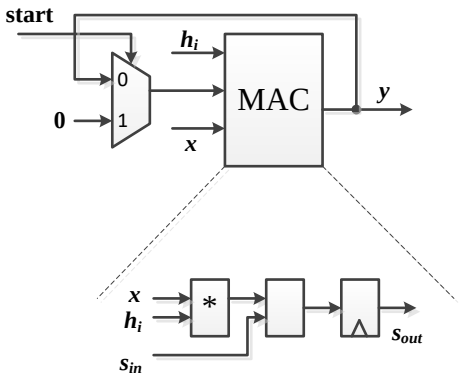

Рис. 5. Реализация транспонированной архитектуры фильтра на основе одного умножителя и сумматора

Итак, налицо закон сохранения энергии в мире цифровых устройств: производительность устройства и количество использованных при его реализации ресурсов связаны прямой зависимостью. Для улучшения одного показателя необходимо ухудшить другой.

Задержка цифрового устройства в тактах vs Максимальная тактовая частота

Другой характеристикой цифровых устройств является время, необходимое для расчета результата. Это время определяется с одной стороны количеством тактов, за которое выполняются необходимые операции, а с другой стороны – величиной тактовой частоты, на которой работает устройство.

Чем выше тактовая частота, тем меньше задержка. Чем меньше тактов на выполнение операции, тем меньше задержка.

Для снижения задержки необходимо применять распараллеливание и уменьшать задержку на элементах внутри устройства. Устройство с низкой задержкой предполагает в общем случае отказ или снижение числа шагов конвейера, а также отказ от триггеров, что приведет к увеличению длины критического пути и снизит максимально достижимую тактовую частоту.

Так, например, в конвейере на рис. 3 для сокращения задержки можно применять комбинационное умножение. Т.е умножение без защелкивания результатов. Задержка устройства в этом случае будет определяться задержкой выполнения комбинационного умножения. Для примера на рис. 6 показан конвейер с рис. 3 с комбинационным умножением на втором шаге конвейера. Задержка снизилась до трех тактов, но длина критического пути увеличилась.

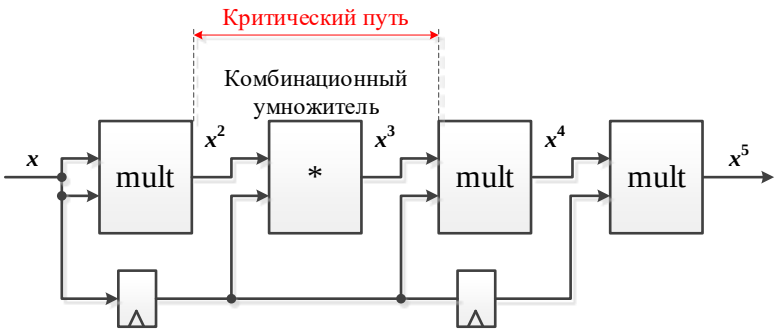

Рис. 6. Конвейерная реализация операции x^5 на основе четырех последовательных умножителей, второй умножитель – комбинационный

Цена снижения задержки путем удаления триггеров – увеличение длины критического пути (пути по комбинационной логике между двумя триггерами) и, как следствие, снижение максимальной тактовой частоты, на которой сможет работать устройство без нарушения времен установки и удержания.

Другим способом снижения задержки устройства в тактах является распараллеливание. Например, в конвейере на рис. 4 путем применения параллельных умножителей удалось снизить задержку устройства с четырех тактов до трех. При этом длина критического пути не увеличилась. К сожалению, распараллелить алгоритм не всегда возможно.

Итак, между величиной задержки устройства в тактах и максимальной тактовой частотой существует следующая взаимосвязь. Для **повышения** максимальной тактовой частоты устройства необходимо **разбивать критический путь триггерами** и, тем самым, увеличивать задержку устройства в тактах. Для **снижения** задержки в тактах необходимо **распараллеливать** алгоритм обработки сигналов, что приведет к увеличению ресурсов, либо **переходить к комбинационным вычислениям**, что уменьшит максимальную тактовую частоту.