Арифметико-логические схемы, триггеры, регистры

Луцив Дмитрий Вадимович
Кафедра системного программирования СПбГУ

Содержание

Логические схемы

- Арифметико-логическое устройство
- Поразрядные логические операции и сдвиг

Понятие логической схемы

Логическая схема — устройство, вычисляющее значения одной или нескольких логических функций

Понятие логической схемы

Логическая схема — устройство, вычисляющее значения одной или нескольких логических функций

- Как правило, всё-таки электронное
- *Как правило*, асинхронное
- Не имеет *состояния*, т.е. после стабилизации выходов, их значения полностью определяются входами

Устройство и действие сумматора

- **Сумматор (Adder)** — логическая схема, получающая на входе биты слагаемых и выдающая биты их суммы

Складываем $A + B = S$.

При суммировании двоичных чисел у i-го разряда

- на вход поступают a_i, b_i, $c_{i-1 \to i}$
- на выходе имеем: s_i, $c_{i \to i+1}$

При этом:

- $s_i = a_i \oplus b_i \oplus c_{i-1 \to i}$
- $c_{i \to i+1} = (a_i \wedge b_i) \vee (b_i \wedge c_{i-1 \to i}) \vee (c_{i-1 \to i} \wedge a_i)$

Таким образом, сумматор — логическая схема.

Схема 1-разрядного сумматора

1-разрядный полусумматор

Полусумматор (2 входа)

- $s = a \oplus b$
- $c = a \wedge b$

1-разрядный полный сумматор

- Оптимизируем формулу для переноса с предыдущего слайда:

$$c_{i \to i+1} = (a_i \wedge b_i) \vee (c_{i-1 \to i} \wedge (a_i \oplus b_i))$$

- Это позволит создать 1-битный сумматор на основе двух полусумматоров и ещё 1 вентиля
  - на самом деле входы равноправны

Схема многоразрядного сумматора

- Собирается через соединение переносов
- Узкое место по времени — как раз перенос

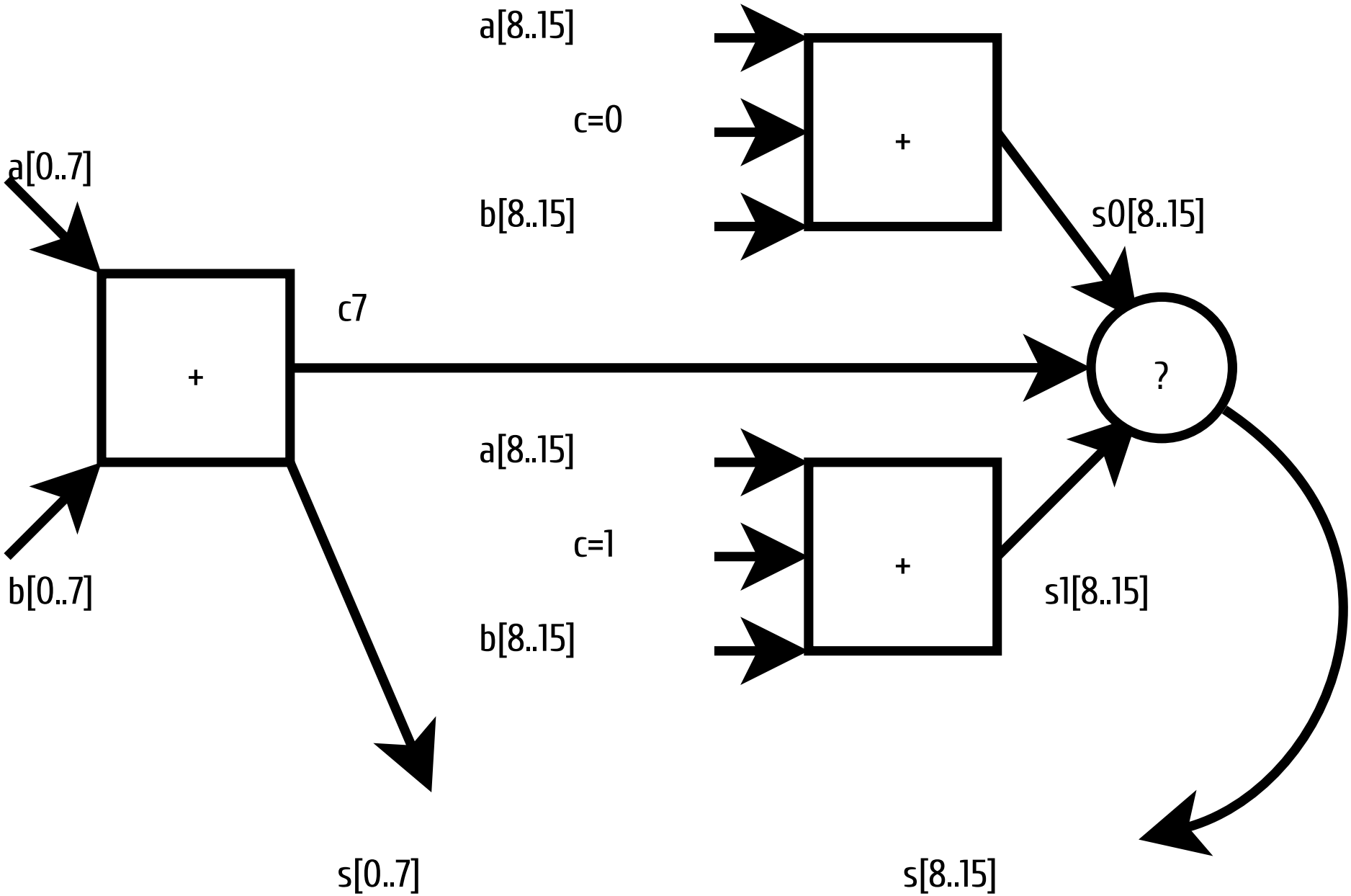
Замечания

Перенос через сумматор распространяется постепенно, поэтому:

- в старших разряде «ответ» есть сразу, но он неправильный
- правильный ответ появляется через какое-то время
- чем больше разрядность операции, тем больше время, даже для одного и того же процессора

Последний бит переноса $c_{N-1\rightarrow}$ можно запомнить и при следующем суммировании использовать в качестве $c_{\rightarrow 0}$. Это позволит соединить несколько сумматоров или несколько раз использовать один и тот же для реализации арифметики произвольной разрядности.

Оптимизация: упреждающий сегментированный сумматор (1)



Оптимизация: упреждающий сегментированный сумматор (2)

- (?) — *мультиплексор* — логическая схема, подающая на выход один из входов под действием управляющего сигнала
- Вынашивание полутора детей за 4,5 месяца. Да, оказывается можно!
- Можно сделать 3, 4 или больше сегментов
- Важна золотая середина: сегменты выбираются последовательно $\Longrightarrow$ грубая оценка — порядка $\sqrt{N}$ (на самом деле меньше)

Инвертор (для вычитания)

$x - y = x + (-y)$. Проблема — вычислить $-y$, зная y.
$[-y] = [P - y] = [P - 1 + 1 - y] = [1] + [(P - 1) - y]$ — уже легче, осталось вычислить $(P - 1) - y$ (а сумматор, только прибавляющий 1, можно дополнительно оптимизировать)

Для $P = 2^N$ справедливо $P - 1 = \sum_{i=0}^{N-1} 2^i$.

Т.е. это число со всеми двоичными единицами. Чтобы вычесть из него y, достаточно инвертировать все биты y при помощи вентиля НЕ, т.к. заёма при вычитании не возникает

Умножение и деление: Мультипликатор

При умножении на 1 бит числа x мы можем пользоваться логической схемой:

$$x \times b = x \wedge b.$$

Тогда для многоразрядных чисел справедливо ($\ll$ — операция сдвига):

$$x \times y = \sum_{i=0}^{N-1} (y_i \wedge (x \ll i))$$

Т.е. выразили умножение через известные операции.

Разрядность произведения — сумма разрядностей множителей.

Умножение и деление: Делитель

Алгоритм деления «в столбик» для системы счисления с произвольным основанием:

1. сдвигаем делитель влево, пока он меньше делимого;
2. вычитаем, пока вычитается, прибавляя к очередному разряду частного;
3. сдвигаем делитель вправо (и меняем позицию в частном), пока нельзя вычитать, потом (2);
4. если делитель вернулся на исходную позицию относительно сдвига, то от делимого остался остаток.

Для двоичной:

- вычитаем, **если** *(а не пока)* вычитается.

Умножение и деление: Делитель

Алгоритм деления «в столбик» для системы счисления с произвольным основанием:

1. сдвигаем делитель влево, пока он меньше делимого;
2. вычитаем, пока вычитается, прибавляя к очередному разряду частного;
3. сдвигаем делитель вправо (и меняем позицию в частном), пока нельзя вычитать, потом (2);
4. если делитель вернулся на исходную позицию относительно сдвига, то от делимого остался остаток.

Для двоичной:

- вычитаем, **если** *(а не пока)* вычитается.

Деление медленное, многие компиляторы умеют заменять деление на константу умножением, пользуясь свойствами кольца, например, для 64-битных машин $\mathbb{Z}/2^{64}$. Так,
`a / 7`
компилируется для x86_64 в
`imul rcx, rax, 0xffffffff92492493`

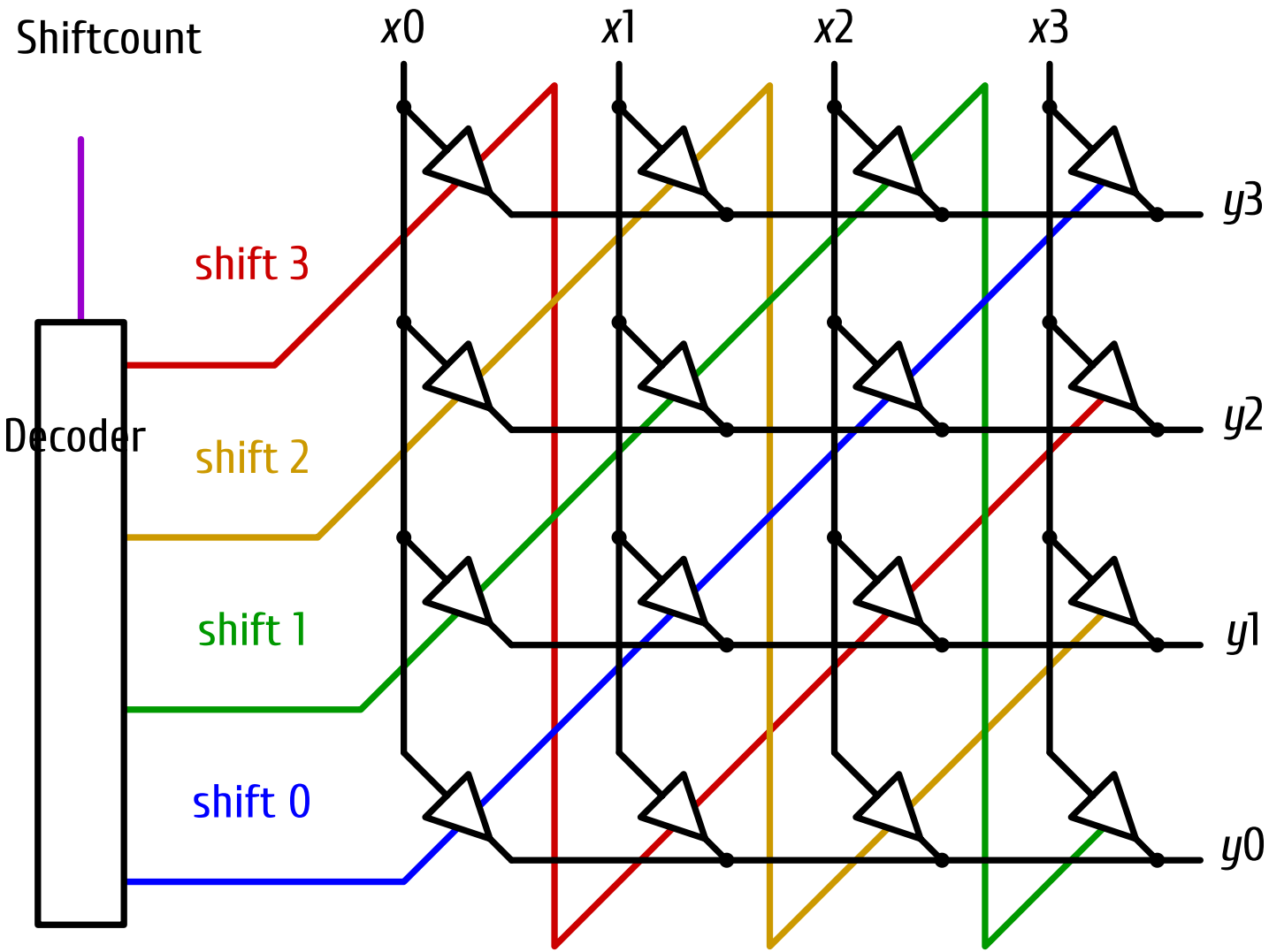
Умножение и деление: замечания

- Мультипликатор, как и сумматор, можно реализовать логической схемой, но очень громоздкой, поэтому часто его делают микропрограммой. А на простых архитектурах — программой.
- Есть аппаратные логические делители — большие и «горячие» микросхемы, используются очень редко. Делитель — почти всегда микропрограмма (или программа).
- Умножение чисел произвольной длины возможно, однако перенос эффективнее делать по слову (старшей половине произведения), а не по биту.

Поразрядные логические операции

Очевидно

Барабанная схема сдвига (Barrel shifter)



Схемы хранения состояния

- Триггеры
- Регистры

Базовые триггеры

Триггер — логическая схема, запоминающая состояние

RS-триггер — наше всё

1. RS-триггер
2. Некорректные состояния триггера

Синхронизация

1. Синхронизация по уровню тактового импульса
2. Синхронизация по фронту
3. D-триггер
4. Связка Master-Slave

Чередование состояний

1. JK-триггер
2. T-триггер
   - На основе JK
   - На основе D

Регистры

- **Регистр** — запоминающая схема на 1 или несколько битов
  - На практике — с возможностью выполнения некоторых операций, т.е. с частью логики

Счётчики

- На базе T и D, по фронту и по спаду
- Увеличивающие и уменьшающие
- С заданным начальным состоянием
  - С использованием сумматора

Сдвигающий регистр

- Загружающий
- Выгружающий
  - Требует мультиплексора

Хранящий регистр в процессоре

- Тоже требует мультиплексора

Хранящий регистр в процессоре

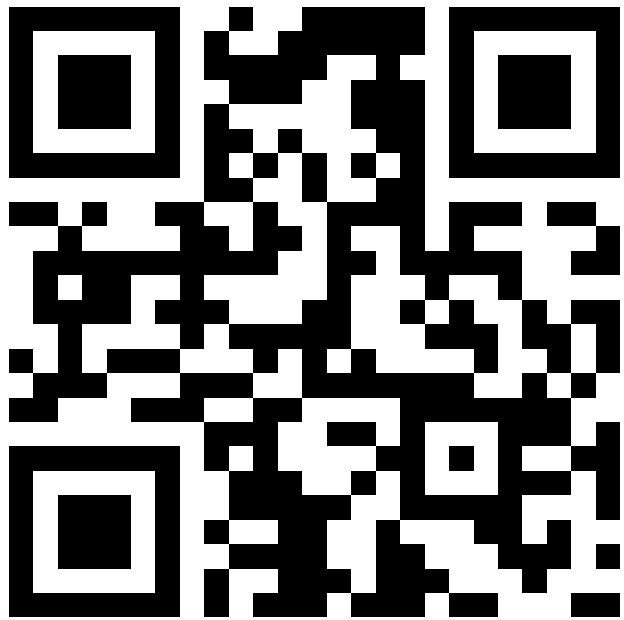
- Тоже требует мультиплексора

В процессоре на входе всех блоков мультиплексоры, которые выбирают, откуда прочитать входные данные по тактовому сигналу. Фактически (особенно в RISC) машинный код в необходимой последовательности переключает мультиплексоры, обеспечивая прохождение данных по нужным маршрутам.

Статическая память

Аналогично динамической (про которую позже), но на базе мультиплексоров и регистров из D-триггеров

Вопросы



EDU.DLUCIV.NAME