**Чем отличается ARM и X86**

x86 процессоры используют сложный набор инструкций, который называется CISC - Complex Instruction Set Computing.  
  
ARM процессоры наоборот используют упрощенный набор инструкций — RISC - Reduced Instruction Set Computing.  
  
Кстати ARM расшифровывается как Продвинутые RISC машины - Advanced RISC Machines.  
  
Наборы инструкций ещё принято назвать архитектурой или ISA - Instruction Set Architecture.

Второе отличие — это микроархитектура. Что это такое?

От того на каком языке говорят процессоры, зависит и то, как они проектируются. Потому как для выполнения каждой инструкции на процессоре нужно расположить свой логический блок. Соответственно, разные инструкции — разный дизайн процессора. А дизайн — это и есть микроархитектура.

* x86 — CISC
* ARM — RISC

Итак, запомнили. Говорим x86 — подразумеваем архитектуру CISC, ARM — это RISC.  
  
Но как так произошло, что процессоры стали говорить на разных языках?

Недостатки CISC

Но был ли такой подход оптимальным??? С точки зрения разработчиков — да. Но вот микроархитектура страдала.  
  
Представьте, вы купили квартиру и теперь вам нужно обставить её мебелью. Площади мало, каждый квадратный метр на счету.  И вот представьте, если бы CISC-процессор обставил мебелью вам гостиную, он бы с одной стороны позаботился о комфорте каждого потенциального гостя и выделил бы для него своё персональное место.  
  
С другой стороны, он бы не щадил бюджет. Диван для одного человека, пуф для другого, кушетка для третьего, трон из Игры Престолов для вашей Дейенерис. В этом случае площадь комнаты бы очень быстро закончилась. Чтобы разместить всех вам бы пришлось увеличивать бюджет и расширять зал. Это не рационально. Но самое главное, CISC-архитектура существует очень давно и те инструкции, которые были написаны в 60-х годах сейчас уже вообще не актуальны. Поэтому часть мебели, а точнее исполнительных блоков, просто не будут использоваться. Но многие из них там остаются. Поэтому появился RISC…

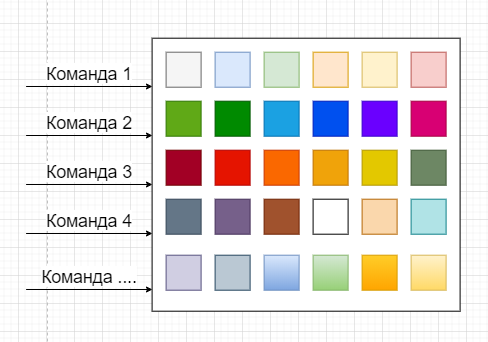
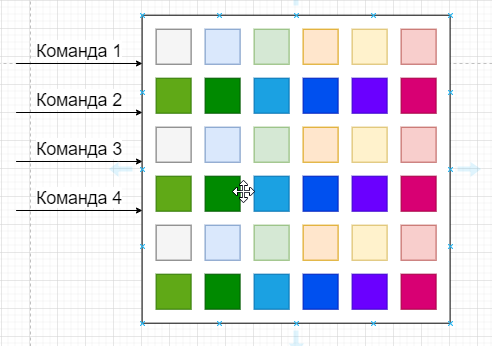
Преимущества RISC

С одной стороны писать на Assembler'е под RISC процессоры не очень-то удобно. Если в лоб сравнивать код, написанный под CISC и RISC процессоры, очевидно преимущество первого.  
  
Так выглядит код одной и той же операции для x86 и ARM.  
  
**x86**

* MOV AX, 15; AH = 00, AL = 0Fh
* AAA; AH = 01, AL = 05
* RET

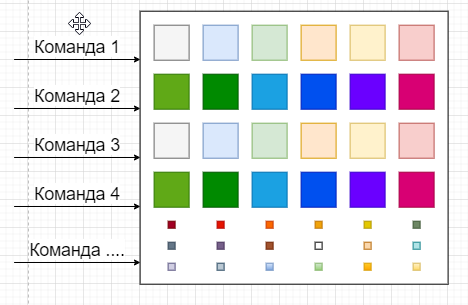
**ARM**

* MOV R3, #10
* AND R2, R0, #0xF
* CMP R2, R3
* IT LT
* BLT elsebranch
* ADD R2. #6
* ADD R1. #1
* elsebranch:
* END

Но так было раньше. На ассемблере уже давно никто не пишет.  Сейчас за программистов всё это делают компиляторы, поэтому никаких сложностей с написанием кода под RISC-процессоры нет. Зато есть преимущества.  
  
Представьте, что вы проектируете процессор. Расположение блоков на х86 выглядело бы так.  
  
[](http://droider.ru/wp-content/uploads/2020/09/image1-2.png)  
  
Каждый цветной квадрат — это отдельные команды. Их много и они разные. Как вы поняли, здесь мы уже говорим про микроархитектуру, которая вытекает из набора команд. А вот ARM-процессор скорее выглядит так.  
  
[](http://droider.ru/wp-content/uploads/2020/09/image3-2.png)  
  
Ему не нужны блоки, созданные для функций, написанных 50 лет назад.  
  
По сути, тут блоки только для самых востребованных команд. Зато таких блоков много. А это значит, что можно одновременно выполнять больше базовых команд. А раритетные не занимают место.  
  
Еще один бонус сокращенного набора RISC: меньше места на чипе занимает блок по декодированию команд. Да, для этого тоже нужно место. Архитектура RISC проще и удобнее, загибайте пальцы:

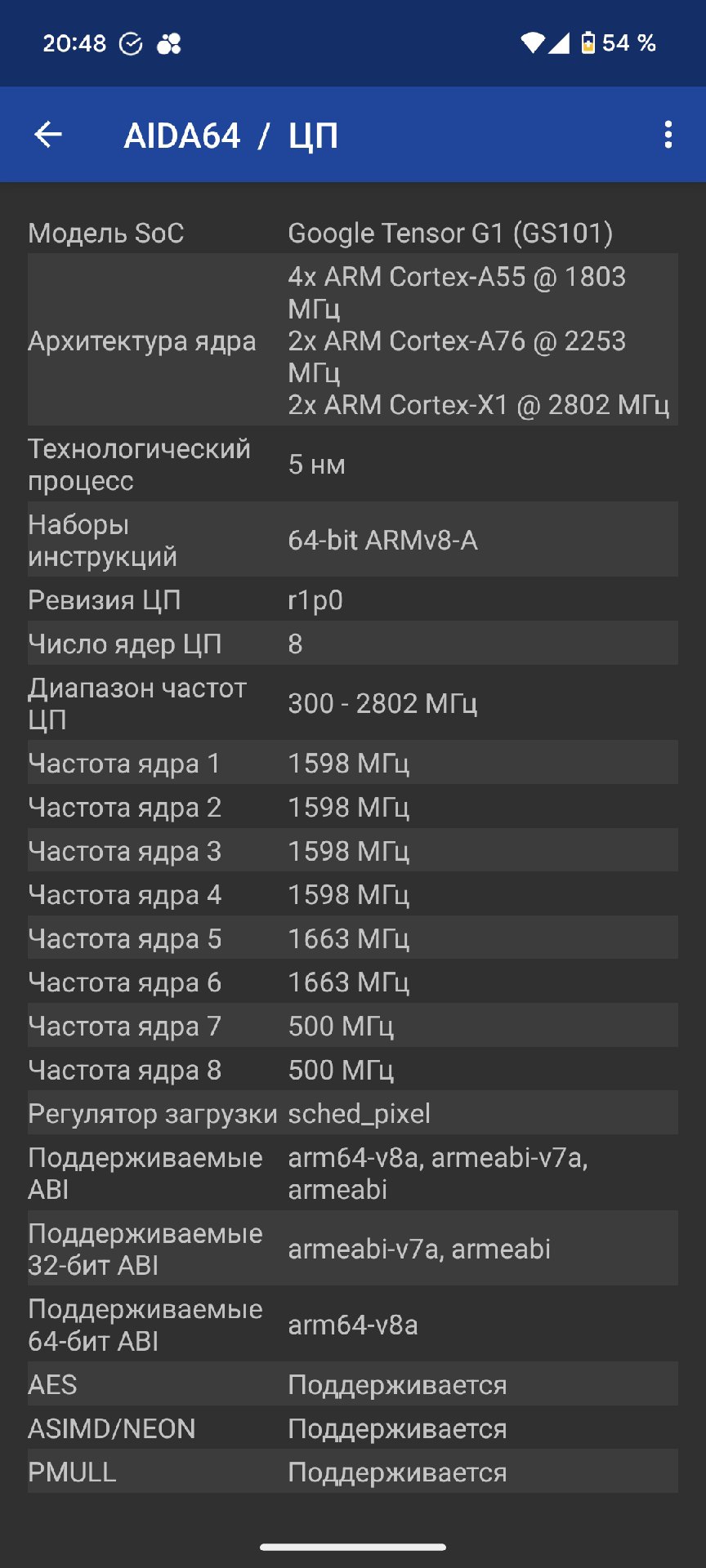
* проще работа с памятью,
* более богатая регистровая архитектура,
* легче делать 32/64/128 разряды,
* легче оптимизировать,
* меньше энергопотребление,
* проще масштабировать и делать отладку.

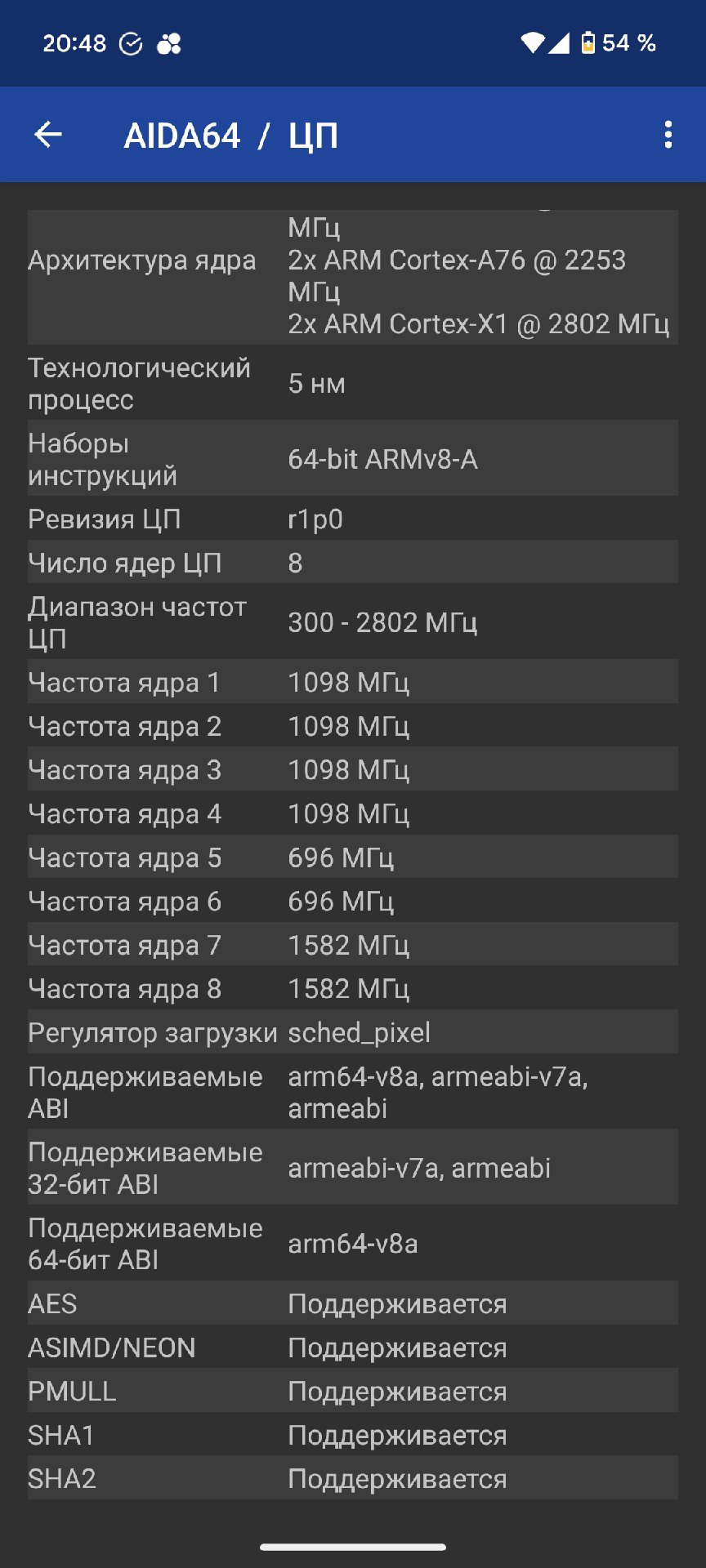
Для примера вот два процессора одного поколения. ARM1 и Intel 386. При схожей производительности ARM вдвое меньше по площади. А транзисторов на нем в 10 раз меньше: 25 тысяч против 275 тысяч. Энергопотребление тоже отличается на порядок: 0.1 Ватт против 2 Ватт у Intel. Шок.  
  
Поэтому наши смартфоны, которые работают на ARM процессорах с архитектурой RISC, долго живут, не требуют активного охлаждения и такие быстрые.

[](http://droider.ru/wp-content/uploads/2020/09/image7-2.png)  
  
Но уже достаточно давно процессоры Intel и AMD разбивают входные инструкции на более мелкие микро инструкции (micro-ops), которые в дальнейшем — сейчас вы удивитесь — исполняются RISC ядром.  
  
Да-да, ребята! Те самые 4-8 ядер в вашем ПК — это тоже RISC-ядра!  
  
Надеюсь, тут вы окончательно запутались. Но суть в том, что разница между RISC и CISC-дизайнами уже сейчас минимальна.  
  
А что остается важным — так это микроархитектура. То есть то, насколько эффективно все организовано на самом камне.

Ход работы

* 1. Устанавливаем аиду на телефон и заходим в вкладку цпу





Устанавливаем Cpuz, вхолдим в кладку SOC

