ISA

- ISA defines how the bits of the instruction are interpreted
  - NUCLEO - Cortex-M4, ARMv7 ISA

Compiler Optimization

Knowing your ISA

- Getting the most out of your CPU requires knowing what instructions it has.
  - Example: Copy an 8-bit signed value (`val8`) to a 32-bit variable (`val32`):
    - In C, this is done as:

```
if (val8 & 0x80 !=0)
  val32 = (0xFFFFFF00|val8);
else
  val32 = val8;
```

assembly

```
    and   r0, r0, #128
    cmp   r0, #1
    beq   .SIGNED
    ldrb  r0, [r11, #-1]
    ldr   r1, 0xFFFFFF00
    orr   r0, r0, r1
.SIGNED
   ldrb r0, [r11, #-1]
```

  - Or on ARM-M4, you can do: `SXTB Rd, Rm`
    - Sign-extend a copy of the byte in `Rm` into a 32-bit register `Rd`.
  - But the compiler won't always do this automatically!

- Worth it to spend some time seeing what instructions your CPU has.

SIMD

ARM-M4

- z = __SADD16(x, y) → Macro to add 16|16 uint32_t and store in 16|16 in uint32_t
  - __ASM - write assembly directly in C. Volatile - prevent compiler optimization
- instruction for FP addition
- use libraries
  - CMSIS-DSP - signal filtering and FFT, Mel-frequency cepstrum(MFCC) speech processing
  - CMSIS-NN

Embedded CPU

- factors
  - bit-width - impacts total number of cycles
  - hardware support - encryption accelerator/multiply/FP
  - fastest - throughput (BW) OR latency (clock rate)?
- user-time related to CPU clock (cycles)

$$ExecutionTime = ClockCycles \times ClockPeriod$$

- bit-width affects total number of cycles

Cycles per Instruction (CPI)

$$TotalClockCycles = \sum_{i \in I} |i| \times cyclesperInstruction$$

Factors affecting CPI

- depends on architecture - w/wo hardware accelerator (hardware multiplier)
- # of instructions depends on
  - hardware
    - ISA
    - architecture
  - software
    - algorithm - more efficient
    - programming language
    - compiler
- A program can have more instruction, but if lower CPI, execution time is smaller

Given two programs with the instruction mixes shown:

	P1	P 2	CPI
Ins. A	2	4	2
Ins. B	1	1	1
Ins. C	2	2	3

- Which program executes the most instructions?
  - P1 → 2 + 1 + 2 = 5 total instructions
  - P2 → 4 + 1 + 2 = 6 total instructions
- Which program is faster? (i.e., takes fewer clock cycles to finish)
  - Clock cycles = Σ(CPI x # of instructions), for each instruction type
  - P1 → 2x2 + 1x1 + 2x3 = 11
  - P2 → 4x2 + 1x1 + 2x3 = 15
- What is the average CPI for each program?
  - CPI = (total Cycle Count)/IC
  - P1 → 11/5 = 2.2
  - P2 → 15/6 = 2.5

CPU Performance

$$Execution\ Time = (\sum_{i \in I} \#\ of\ Instrctions \times cycles\ per\ instruction) \times\ clock\ period$$

Clock period

	Freq.	CPI
Sys. A	4 GHz	2.0
Sys B	2 GHz	1.2

- Now consider the **same** program (using the same ISA) running on TWO different systems.
- At first glance, A runs at 4GHz, and B runs at 2GHz. So, is A faster?
- Clock periods: System A = 250 ps and system B = 500 ps
- If both run N instructions:
  - For Computer A, time = N x 2 x 250 = 500N
  - For Computer B, time = N x 1.2 x 500 = 600N
  - Computer A outperforms computer B.

IPS (Instructions Per Second)

$$IPS = \frac{Instruction\ count}{Execution\ time} = \frac{Instruction\ count}{\frac{Instruction\ count \times CPI}{Cycle\ time}} = \frac{Cycle\ time}{CPI}$$

- MIPS (Million IPS) (10^6)

FP Hardware Example

1. You want to switch to an M4 CPU with floating point support, where these operations take just 1 cycle. What is the speed-up you can get when moving from M3 to M4 if the frequencies are the same for both?

Insn. type	# insn.	CPI
Float	2N	100
Other	20N	1

Solution:

Since the frequencies are the same, we can compare just the clock cycles taken on each system. The formula for this is:

Clock cycles = Σ(CPI x # of instructions), for each instruction type

M3 clock cycles = (2N x 100) + (20N x 1) = 200N + 20N = 220N

M4 clock cycles = (2N x 1) + (20N x 1) = 2N + 20N = 22N

Speed-up = M3 clock cycles ÷ M4 clock cycles = 220N ÷ 22N = 10x

FP hardware example

2. The M3 system runs at 100MHz, while the M4 system runs at 80 MHz. What is the speed-up now when using the M4 vs. the M3?

Solution:

For this question, the frequencies are different. So, we must compare the execution time on each system. First, we must calculate the clock period of each system

M3 clock period = 1 ÷ 100 MHz = 10 ns ; M4 clock period = 1 ÷ 80 MHz = 12.5 ns

Next, we compute the execution time using: Execution time = Clock cycles x clock period

M3 execution time = 220N x 10 ns = 2,200N ns

M4 execution time = 22N x 12.5ns = 275N ns

Speed up = 2200N ns ÷ 275N ns = 8x

1e^-8 = 10ns
1e^-9 = 1ns

1e^-6 = 1us
1e^-3 = 1ms

FP hardware example

3. You want to see how fast the M3 system runs when using fixed point. With fixed point, your code increases to 26 instructions per loop, and they all take 1 cycle each. What is the execution time for the M3-fixed system?

Solution:

In the table on slide 25, there are ‘float’ instructions and ‘other’ instructions. So, if float instructions are replaced with integer instructions, they would now also take 1 cycle each.

M3-fixed execution time = 26N x 10ns = 260N ns.