CS 240
Computer Systems

#4: ISAs and Instruction Sets, File Types, & Memory

Jan. 27, 2022 · Wade Fagen-Ulmschneider

Instruction Set Architecture (ISA)

Every CPU has a set of commands it understands known as it's Instruction Set Architecture or ISA. Two ISAs are **very** common:

1.

2.

An ISA defines the function of the hardware in the CPU.

CPU Registers

Each CPU core has an extremely limited number of ___________ that are used for general purpose CPU operations:

- x64: 16 registers of 64 bits
- ARMv8: 31 registers of 64 bits

With very few exceptions ________________________________.

Instruction Sets

Every ISA defines a set of instructions that a CPU can execute:

Move:	MOV, XCHG, PUSH, POP, ...
Arithmetic (int):	ADD, SUB, MUL, DIV, NEG, CMP, ...
Logic:	AND, OR, XOR, SHR, SHL, ...
Control Flow:	JMP, LOOP, CALL, RET, ...
Synchronization:	LOCK, ...
Floating Point:	FADD, FSUB, FMUL, FDIV, FABS, ...

ARM processors have significantly fewer instructions and are known as _____________________________ while x64 processors have a greater set of instructions and known as ______________________.

Q: Advantages of RISC / CISC?

CPU Instruction in a Real Program

	04.c	gcc 04.c objdump -d ./a.out	

3	int main() {	f3 0f 1e fa
		55
		48 89 e5
		48 83 ec 10
			endbr64
			push %rbp
			mov %rsp,%rbp
			sub $0x10,%rsp

4	int a = 0;	c7 45 fc 00 00 00 00	movl $0x0,-0x4(%rbp)
5	a = a + 3;	83 45 fc 03	addl $0x3,-0x4(%rbp)
6	a = a - 2;	83 6d fc 02	subl $0x2,-0x4(%rbp)
7	a = a * 4;	c1 65 fc 02	shll $0x2,-0x4(%rbp)

8	a = a / 2;	8b 45 fc
		89 c2
		c1 ea 1f
		01 d0
		d1 f8
		89 45 fc
			mov -0x4(%rbp),%eax
			mov %eax,%edx
			shr $0x1f,%edx
			add %edx,%eax
			sar %eax
			mov %eax,-0x4(%rbp)

9	a = a * 5;	8b 55 fc
		89 d0
		c1 e0 02
		01 d0
		89 45 fc
			mov -0x4(%rbp),%edx
			mov %edx,%eax
			shl $0x2,%eax
			add %edx,%eax
			mov %eax,-0x4(%rbp)

10	printf("Hi");	48 8d 3d f0 0d 00 00 b8 00 00 00 00 e8 42 fe ff ff	lea 0xdf0(%rip),%rdi # 2004 <_IO_stdin_used+0x4> mov $0x0,%eax callq 1060<printf@plt>

11	a = a * 479;	8b 45 fc
		69 c0 df 01 00 00
		89 45 fc
			mov -0x4(%rbp),%eax
			imul $0x1df,%eax,%eax
			mov %eax,-0x4(%rbp)

Program Counter (PC):

Operation Timings:

Endianness:
One major difference between ISAs is how multi-byte characters are stored. Knowing that **`sizeof(int) == 4`**, what do we expect from the following program?

04b.c

```c
int i = 3 + (2 << 8) + (1 << 16);  // 66051
char *s = (char *)&i;
printf("%02x %02x %02x %02x\n", s[0], s[1], s[2], s[3]);
```

x86/x64 - Big Endian:

s[0]	s[1]	s[2]	s[3]

ARM/A64 - Little Endian:

s[0]	s[1]	s[2]	s[3]

What is “Host Byte Order”'? What is “Network Byte Order”?

Beyond Characters: Files and File Types
Using binary digits, often represented as characters using an encoding like UTF-8, we can build more complex file types.

File Extensions: An Easy Identifier
The most common way to identify the contents of a file is by the **file extension**. The file extension is defined as:

Examples:

`cs240.png`	`mp1.c`	`mp1.h`	`taylor.swift.mp4`

Which files are “plain text files”?

Memory Hierarchy:
The third foundation of a computer system is the “memory” -- the storage of data to be processed by our CPU. There are many different types of common **memory** and **storage** in a system:

1.

2.

3.

4.

5.

6.

Does knowing something about memory matter?

Sample Programs:

04cr.c

```c
  for (unsigned int c = 0; c < SIZE; c++) {
    for (unsigned int r = 0; r < SIZE; r++) {
      array[(r * SIZE) + c] = (r * SIZE) + c;
    }
  }
```

-vs-

04rc.c

```c
  for (unsigned int r = 0; r < SIZE; r++) {
    for (unsigned int c = 0; c < SIZE; c++) {
      array[(r * SIZE) + c] = (r * SIZE) + c;
    }
  }
```

...what is different about **`04cr.c`** and **`04rc.c`**?

Running Times: **`04cr.c`** (Program #1):

`04rc.c` (Program #2):