Foundations of Programming Languages

Foundations: Memory

Prof. Dr. Christoph Reichenbach

Fachbereich 12 / Institut für Informatik

17. Oktober 2014

Data

```
user@host:~$ hexdump -C hello-world.o
...
0200  b8 01 00 00 00 bf 01 00  00 00 48 be 00 00 00 00  |..........H.....|
0210  00 00 00 00 ba 0d 00 00  00 0f 05 b8 3c 00 00 00  |............<...|
0220  bf 00 00 00 00 0f 05 00  00 00 00 00 00 00 00 00  |................|
0230  48 65 6c 6c 6f 2c 20 57  6f 72 6c 64 0a 00 e8 03  |Hello, World....|
0240  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
...
```

Data

```
user@host:~$ hexdump -C hello-world.o
...
0200  b8 01 00 00 00 bf 01 00  00 00 48 be 00 00 00 00  |..........H.....|
0210  00 00 00 00 ba 0d 00 00  00 0f 05 b8 3c 00 00 00  |............<...|
0220  bf 00 00 00 00 0f 05 00  00 00 00 00 00 00 00 00  |................|
0230  48 65 6c 6c 6f 2c 20 57  6f 72 6c 64 0a 00 e8 03  |Hello, World....|
0240  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
...
```

Number (1000)

Data

```
user@host:~$ hexdump -C hello-world.o
...
0200  b8 01 00 00 00 bf 01 00  00 00 48 be 00 00 00 00  |.........H.....|
0210  00 00 00 00 ba 0d 00 00  00 0f 05 b8 3c 00 00 00  |...........<...|
0220  bf 00 00 00 00 0f 05 00  00 00 00 00 00 00 00 00  |...............|
0230  48 65 6c 6c 6f 2c 20 57  6f 72 6c 64 0a 00 e8 03  |Hello, World....|
0240  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |...............|
...
```

- Number (1000)
- Machine language

Data

```
user@host:~$ hexdump -C hello-world.o
...
0200  b8 01 00 00 00 bf 01 00  00 00 48 be 00 00 00 00  |.........H......|
0210  00 00 00 00 ba 0d 00 00  00 0f 05 b8 3c 00 00 00  |............<...|
0220  bf 00 00 00 00 0f 05 00  00 00 00 00 00 00 00 00  |................|
0230  48 65 6c 6c 6f 2c 20 57  6f 72 6c 64 0a 00 e8 03  |Hello, World....|
0240  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
...
```

- Number (1000)
- Machine language
- String ("Hello, World\n")

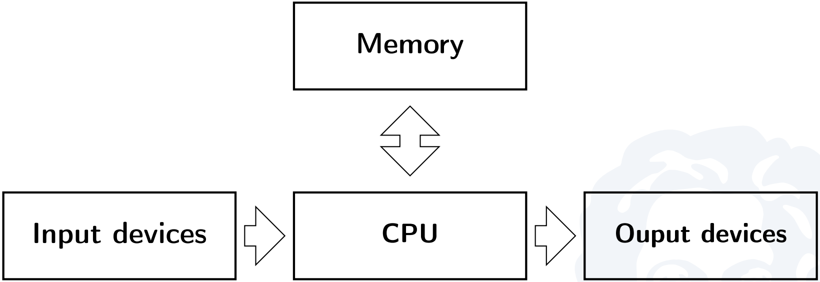
Data

```
user@host:~$ hexdump -C hello-world.o
...
0200  b8 01 00 00 00 bf 01 00  00 00 48 be 00 00 00 00  |..........H.....|
0210  00 00 00 00 ba 0d 00 00  00 0f 05 b8 3c 00 00 00  |............<...|
0220  bf 00 00 00 00 0f 05 00  00 00 00 00 00 00 00 00  |................|
0230  48 65 6c 6c 6f 2c 20 57  6f 72 6c 64 0a 00 e8 03  |Hello, World....|
0240  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
...
```

- Number (1000)
- Machine language
- String ("Hello, World\n")

Memory can contain all sorts of data, often freely mixed

The Computer



CPU + RAM Interaction

Address	Memory
0	`e8 03 00 00 00 00 00 00`
8	`fe ff ff ff ff ff ff ff`

CPU

`$t0` $= 0$

`$t1` $= 0$

- RAM behaves like array: maps *addresses* to bytes

CPU + RAM Interaction

Address	Memory
0	`e8 03 00 00 00 00 00 00`
8	`fe ff ff ff ff ff ff ff`

$\Uparrow$

CPU

`$t0` = 0

`$t1` = 0

- RAM behaves like array: maps *addresses* to bytes
- To operate on memory:
  - CPU loads RAM contents into *registers* such as `$t0`, `$t1`
  - CPU operates on registers
  - CPU writes back registers into RAM

Representing Numbers

- Numbers can be represented in a variety of ways
- Here: 64-bit little-endian two's complement numbers

In-Memory Representation	Hexadecimal	Decimal
`00 00 00 00 00 00 00 00`	`0x0`	0
`01 00 00 00 00 00 00 00`	`0x1`	1
`10 00 00 00 00 00 00 00`	`0x10`	16
`0a 00 00 00 00 00 00 00`	`0xa`	10
`0f 00 00 00 00 00 00 00`	`0xf`	15
`00 01 00 00 00 00 00 00`	`0x100`	256
`ff ff ff ff ff ff ff ff`	`0xffffffffffffffff`	-1
`fe ff ff ff ff ff ff ff`	`0xfffffffffffffffe`	-2

Numbers vs. Bits: Notation

- $\text{num}_s^k(a)$ for interpreting signed $k-$bit strings:
  $\text{num}_s^4(1111) = -1$
- $\text{num}_u^k(a)$ for interpreting unsigned $k-$bit strings:
  $\text{num}_u^4(1111) = 15$

Notation for common bit-string activities

Numbers vs. Bits: Notation

- $\mathrm{num}_s^k(a)$ for interpreting signed $k-$bit strings:
  $\mathrm{num}_s^4(1111) = -1$
- $\mathrm{num}_u^k(a)$ for interpreting unsigned $k-$bit strings:
  $\mathrm{num}_u^4(1111) = 15$
- $\mathrm{repr}^k(n)$ for two's complement representation:
  $\mathrm{repr}^8(7) = \mathtt{00000111}$, $\mathrm{repr}^2(15) = 11$
  (Cuts off excess bits)

Notation for common bit-string activities

Numbers vs. Bits: Notation

- $\text{num}_s^k(a)$ for interpreting signed $k-$bit strings:
  $\text{num}_s^4(1111) = -1$
- $\text{num}_u^k(a)$ for interpreting unsigned $k-$bit strings:
  $\text{num}_u^4(1111) = 15$
- $\text{repr}^k(n)$ for two's complement representation:
  $\text{repr}^8(7) = \mathtt{00000111}$, $\text{repr}^2(15) = 11$
  (Cuts off excess bits)
- $a : b$ for bit string concatenation:
  $\mathtt{01001} : \mathtt{101} = \mathtt{01001101}$

Notation for common bit-string activities

Numbers vs. Bits: Notation

- $\text{num}_s^k(a)$ for interpreting signed $k-$bit strings:
  $\text{num}_s^4(1111) = -1$
- $\text{num}_u^k(a)$ for interpreting unsigned $k-$bit strings:
  $\text{num}_u^4(1111) = 15$
- $\text{repr}^k(n)$ for two's complement representation:
  $\text{repr}^8(7) = \mathtt{00000111}$, $\text{repr}^2(15) = 11$
  (Cuts off excess bits)
- $a : b$ for bit string concatenation:
  $\mathtt{01001} : \mathtt{101} = \mathtt{01001101}$
- $a[x : y]$ for bit string ranges:
  $\mathtt{0101101}[4 : 1] = \mathtt{0110}$

Notation for common bit-string activities

Numbers vs. Bits: Notation

- $\text{num}_s^k(a)$ for interpreting signed $k-$bit strings:
  $\text{num}_s^4(1111) = -1$
- $\text{num}_u^k(a)$ for interpreting unsigned $k-$bit strings:
  $\text{num}_u^4(1111) = 15$
- $\text{repr}^k(n)$ for two's complement representation:
  $\text{repr}^8(7) = \mathtt{00000111}$, $\text{repr}^2(15) = 11$
  (Cuts off excess bits)
- $a\!:\!b$ for bit string concatenation:
  $\mathtt{01001}\!:\!\mathtt{101} = \mathtt{01001101}$
- $a[x : y]$ for bit string ranges:
  $\mathtt{01}\mathtt{0110}\mathtt{1}[4 : 1] = \mathtt{0110}$

Notation for common bit-string activities

Numbers vs. Bits: Notation

- $\mathrm{num}_s^k(a)$ for interpreting signed $k-$bit strings:
  $\mathrm{num}_s^4(1111) = -1$
- $\mathrm{num}_u^k(a)$ for interpreting unsigned $k-$bit strings:
  $\mathrm{num}_u^4(1111) = 15$
- $\mathrm{repr}^k(n)$ for two's complement representation:
  $\mathrm{repr}^8(7) = 00000111$, $\mathrm{repr}^2(15) = 11$
  (Cuts off excess bits)
- $a\!:\!b$ for bit string concatenation:
  $01001\!:\!101 = 01001101$
- $a[x : y]$ for bit string ranges:
  $0101101[4 : 1] = 0110$
- a^n for repetition:
  $(01)^3 = 010101$

Notation for common bit-string activities

Summary

- All data is kept in RAM or in registers
  - RAM: lots of space, slow
  - Registers: very few, fast
- Code is data: CPU executes instructions from RAM
- Code can decide freely how to represent
  - Arrays
  - Data structures
  - Objects
  - Algebraic values

    ...
- Here, we work with 64-bit little-endian two's complement numbers