

CS126 Precept 6

Meeting 8: TOY and TOY
assignment

Randy Wang

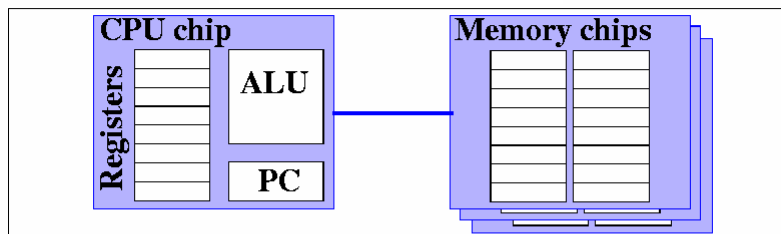
First Exam

- Wednesday, 3/24, 7:30 – 9:30pm
- Friend 101
- Extra review session:
3/23, 7:30pm, CS 105

Number Conversions

- Binary \rightarrow decimal, hex \rightarrow decimal
- Decimal \rightarrow binary, decimal \rightarrow hex
- Binary \rightarrow hex, hex \rightarrow binary
- Negative decimal \rightarrow 2's complement
- 2's complement arithmetic
- Bit-wise operations:
 - \sim & | \wedge \gg \ll
 - $\gg i$ is equivalent to division by 2^i
 - $\ll i$ is equivalent to multiplication by 2^i

TOY



- ALU (arithmetic logic unit) -- executes instructions to manipulate data
- 16 registers -- the fastest form of storage, on-chip in modern computers, used as scratch space during computation
- PC (program counter) -- a register with special meaning, keeps track of the next instruction to be executed
- 256 16-bit words of memory -- stores both code and data

TOY Instruction Set

- You should be able to trace a TOY program
- You should be able to write a TOY program
- The instruction set cheat sheet would always be provided to you---no need to memorize it

Some TOY Instruction Details

- “Bit-whacking”
- Standard input and output
- Function calls

TOY Assignment: Running Programs

- Java webstart TOY application
 - Editor
 - Debugger
 - Simulator
- TOY.java

Writing TOY Programs

- Either with jedit or the visual TOY simulator
- Line number warning: it's in HEX
19 is followed by 1A, not 20
- Comments: not just “dumb” ones
generated automatically by the simulator

encode.toy

```
read four 'bits' (actually four TOY words - m1 m2 m3 m4)
compute three parity 'bits' (p1 p2 p3)
output seven 'bits' (m1 m2 m3 m4 p1 p2 p3)
repeat as long as there are input bits
```

decode.toy

```
read seven bits (four message bits followed by three parity bits)
compute three parity bits
compare computed parity bits with transmitted parity bits
correct a message bit if needed
output correct message bits (m1 m2 m3 m4)
repeat as long as there are input bits
```

tips

- In readme, list register use for each program
- Tips:
 - * may want to write C-like pseudo-code (or actual code) to start and then 'translate' to TOY
 - * write all code on paper first
 - * keep track of registers carefully (readme must include description of register usage)
 - * be careful about line numbers (1A comes after 19)
 - * remember that all numbers are hex values
 - * the pc is initially 10, so code must start with instruction 10
 - * TOY code must be commented: make sure comments and code always match

Common Mistakes

- conflicting use of registers
- confusing hex numbers with decimal
- line number problems (skipping)
- not starting program with instruction 10