Clock cycles: cpu, memory, gpu, bus

$\frac{\text{cycles}}{\text{second}}$ or $\frac{\text{seconds}}{\text{cycle}}$ Hz

200 MHz - frequency bus $\equiv$ 200 000 000 $\frac{\text{cycles}}{\text{second}}$

$\equiv$ 5 $\frac{\text{nano seconds}}{\text{cycle}}$

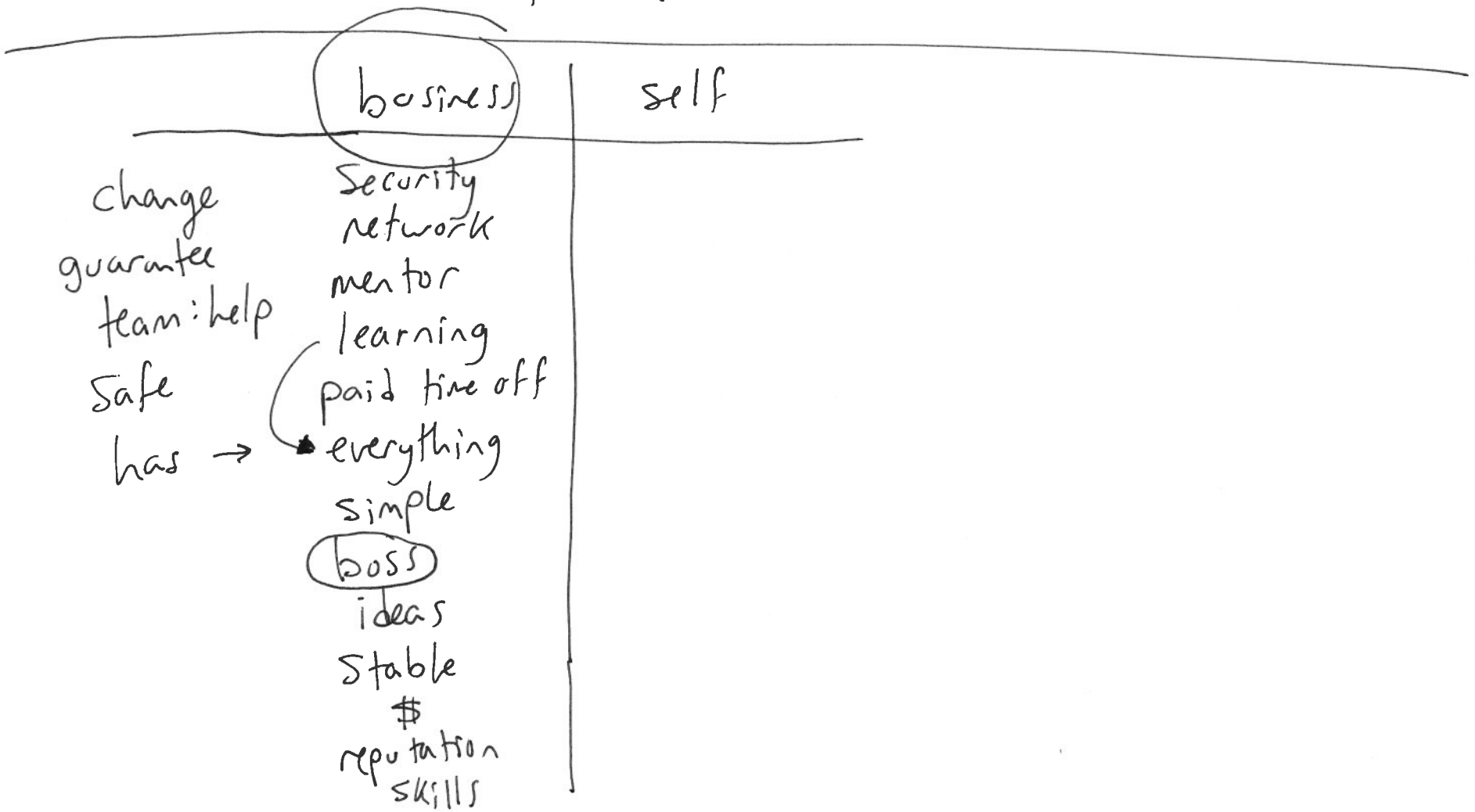
Practice: frequency $\longleftrightarrow$ clock cycle

Pg 63-65: Instruction Execution

1. All instr sb executed directly by hardware
   faster but complex, expensive.
2. Issue instructions as fast as possible
3. Instr sb easy to decode. backwards compatability
4. Only LOAD/STORE should reference MM. Registers instead.
   Cannot avoid the store.
5. Provide lots of registers. $

Goal: feed CPU.

dfn: interpreter: a software program (aka a virtual machine; slower and cheaper than a real machine) that FDE the instructions of another program. The output is another program, simpler to run, for the HW.



Error-correcting codes : Theory

① Error detection
② Error correction

eg Hamming code

Store + retrieve words in MM

(A) Raw data: "data word" or "memory word"

(B) Encoded data: "code word"

example code A: has a vocabulary of just two valid 8-bit words:

00000000 } all 254 other words are not valid
11111111

Retrieve: 11111110 } error detection can be trivial

probably was supposed to be 11111111

---

example code B: 00000000 } these are the only 2 invalid words
11111111

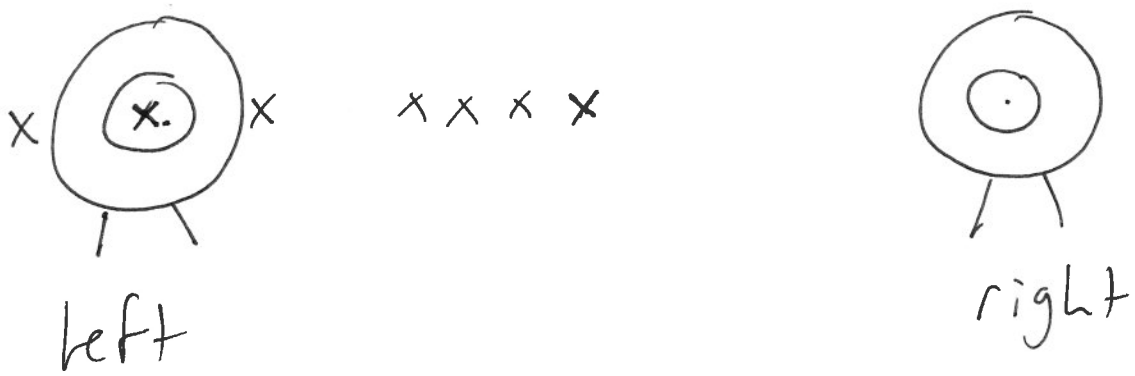
retrieve: 10111001 } valid but possibly wrong

★ TRADEOFF: clarity vs efficiency

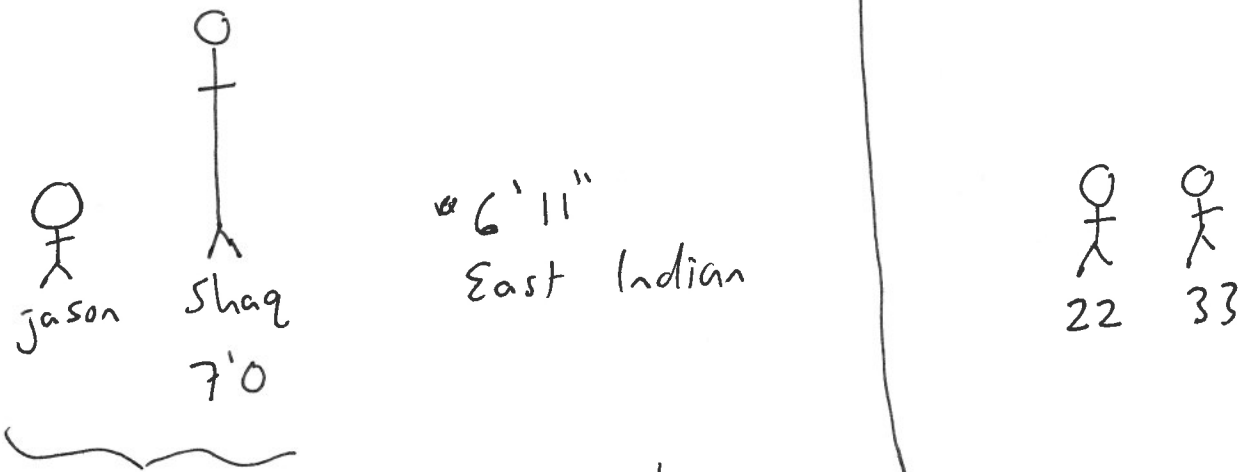
Code A is better: we want large gaps between valid codewords

☆ identify errors
identify corrections

large gaps = clarity



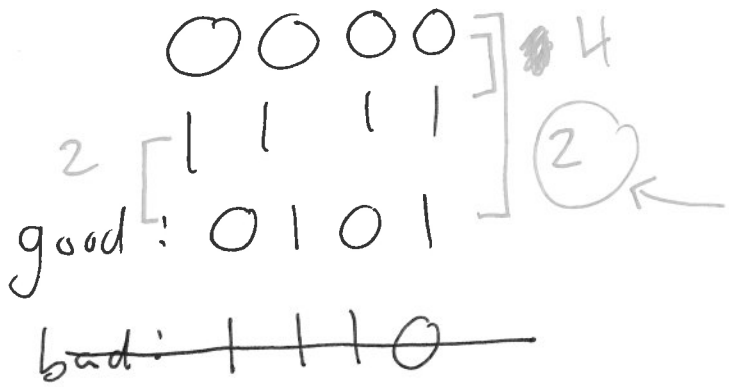
murder



"bald male"
"chinese"
"6'2"

large gaps

code C : valid codewords:



Hamming Distance: the minimum number of bits between any two codewords.

We want a high Hamming distance.

Hamming distance of code A: 8 bits

" " " " B: 1 bit : useless

Quiz: latency, BW, ~~[illegible]~~ today's lesson, freq ↔ clock cycle, ☆ date, pipelines, interpreter, bottlenecks,