QUIZ WHAT ARE 3 TRADITIONAL WAYS HW DESIGNERS MAKE COMPUTERS RUN FASTER?

- [x] FASTER CLOCKS
- [ ] LARGER HARD DISK
- [ ] LONGER CLOCK PERIOD
- [x] MORE PROCESSORS
- [x] MORE WORK/ CLOCK CYCLE
- [ ] REDUCE AMOUNT OF MEMORY

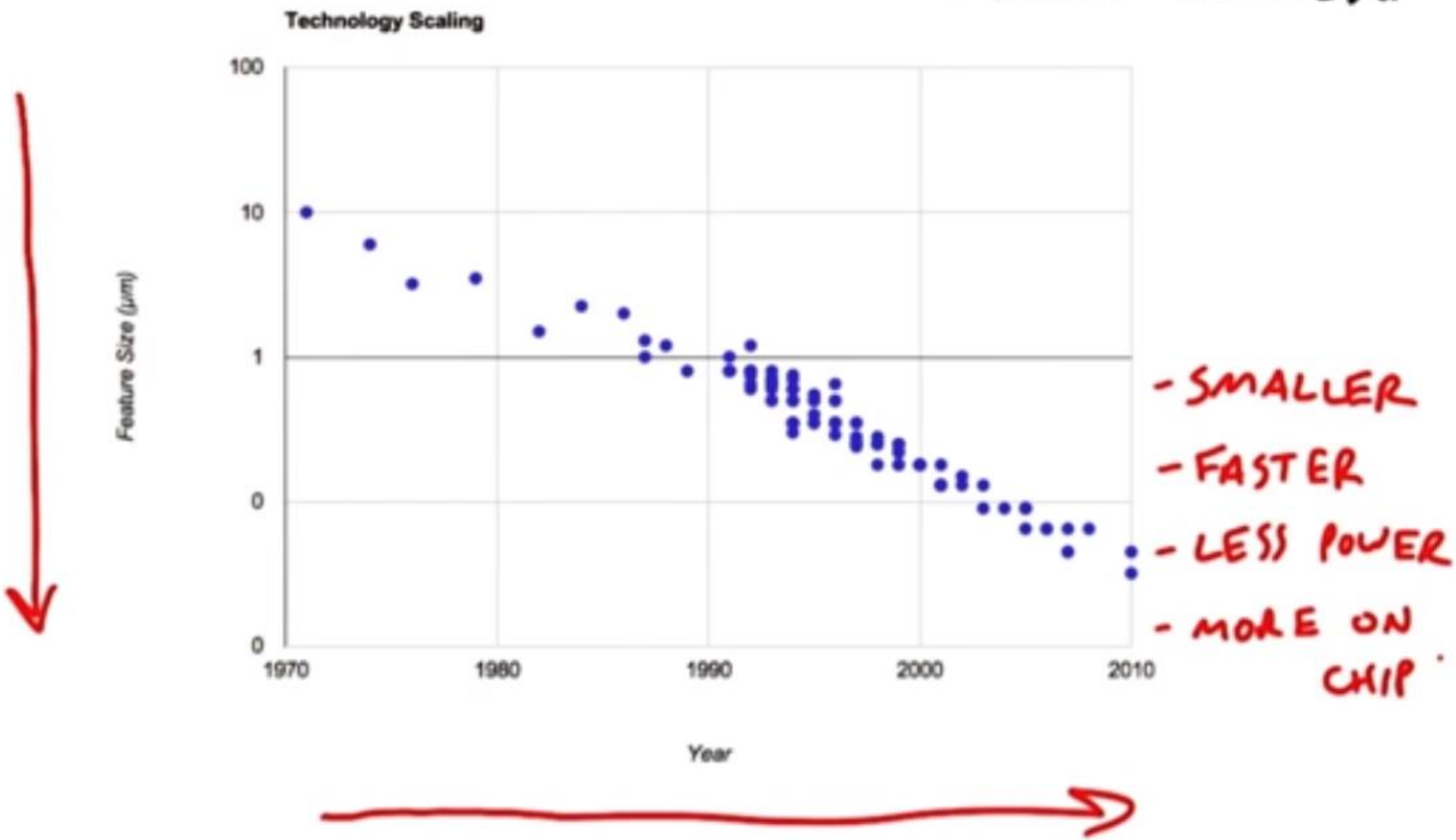
SEYMOUR CRAY: WOULD YOU RATHER PLOW A FIELD WITH TWO STRONG OXEN OR 1024 CHICKENS?

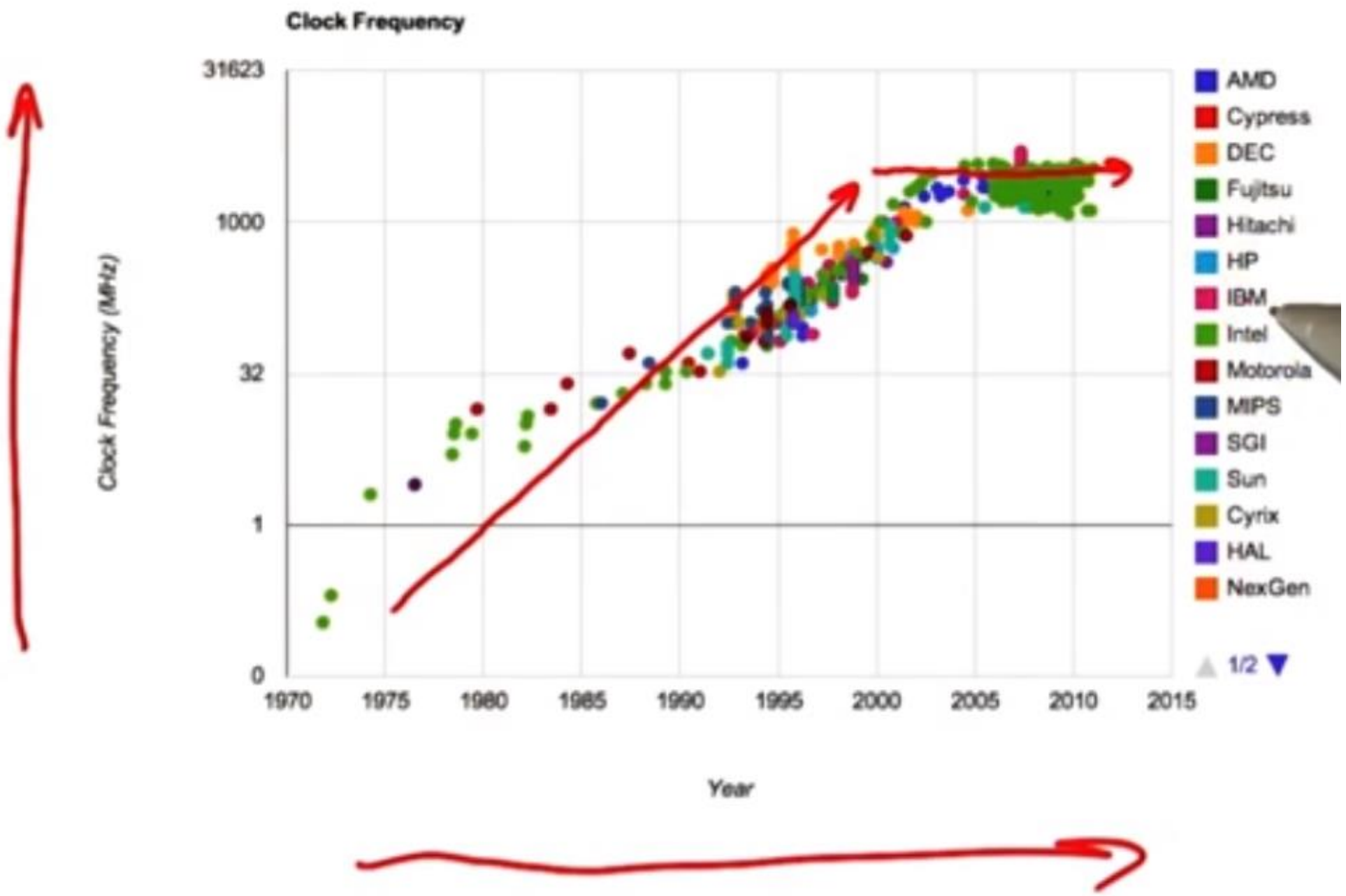
I ♡ CHICKENS!

MODERN GPU:
- THOUSANDS OF ALUs
- HUNDREDS OF PROCESSORS
- TENS OF THOUSANDS OF CONCURRENT THREADS

THIS CLASS: LEARN TO THINK IN PARALLEL (LIKE THE CHICKENS)

CPUDB.STANFORD.EDU




Clock Frequency
31623
1000
32
1
0
Clock Frequency (MHz)
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Year
AMD
Cypress
DEC
Fujitsu
Hitachi
HP
IBM
Intel
Motorola
MIPS
SGI
Sun
Cyrix
HAL
NexGen
1/2

QUIZ

ARE PROCESSORS TODAY GETTING FASTER BECAUSE

- [ ] WE'RE CLOCKING THEIR TRANSISTORS FASTER
- [ ] WE HAVE MORE TRANSISTORS AVAILABLE FOR COMPUTATION?

WHY DON'T WE KEEP INCREASING CLOCK SPEED?

HAVE TRANSISTORS STOPPED GETTING SMALLER & FASTER?

NO.

INSTEAD HEAT!

WHAT MATTERS TODAY: POWER!

CONSEQUENCE:

- SMALLER, MORE EFFICIENT PROCESSORS
- MORE OF THEM.

WHAT KIND OF PROCESSORS WILL WE BUILD?

(MAJOR DESIGN CONSTRAINT: POWER.)

CPU: — COMPLEX CONTROL HARDWARE

↑ FLEXIBILITY + PERFORMANCE!

↓ EXPENSIVE IN TERMS OF POWER

GPU: — SIMPLER CONTROL HARDWARE

↑ MORE HW FOR COMPUTATION

↑ POTENTIALLY MORE POWER EFFICIENT (OPS/WATT)

↓ MORE RESTRICTIVE PROGRAMMING MODEL

QUIZ

WHICH TECHNIQUES ARE COMPUTER DESIGNERS USING TODAY TO BUILD MORE POWER-EFFICIENT CHIPS?

- [ ] FEWER, MORE COMPLEX PROCESSORS
- [x] MORE, SIMPLER PROCESSORS
- [ ] MAXIMIZING THE SPEED OF THE PROCESSOR CLOCK
- [ ] INCREASING THE COMPLEXITY OF THE CONTROL HW

LET'S BUILD A (POWER-EFFICIENT) HIGH PERFORMANCE PROCESSOR!

LATENCY
(TIME)
(SECONDS)

THROUGHPUT
(STUFF/TIME)
(JOBS/HOUR)

LATENCY

(TIME)

(SECONDS)

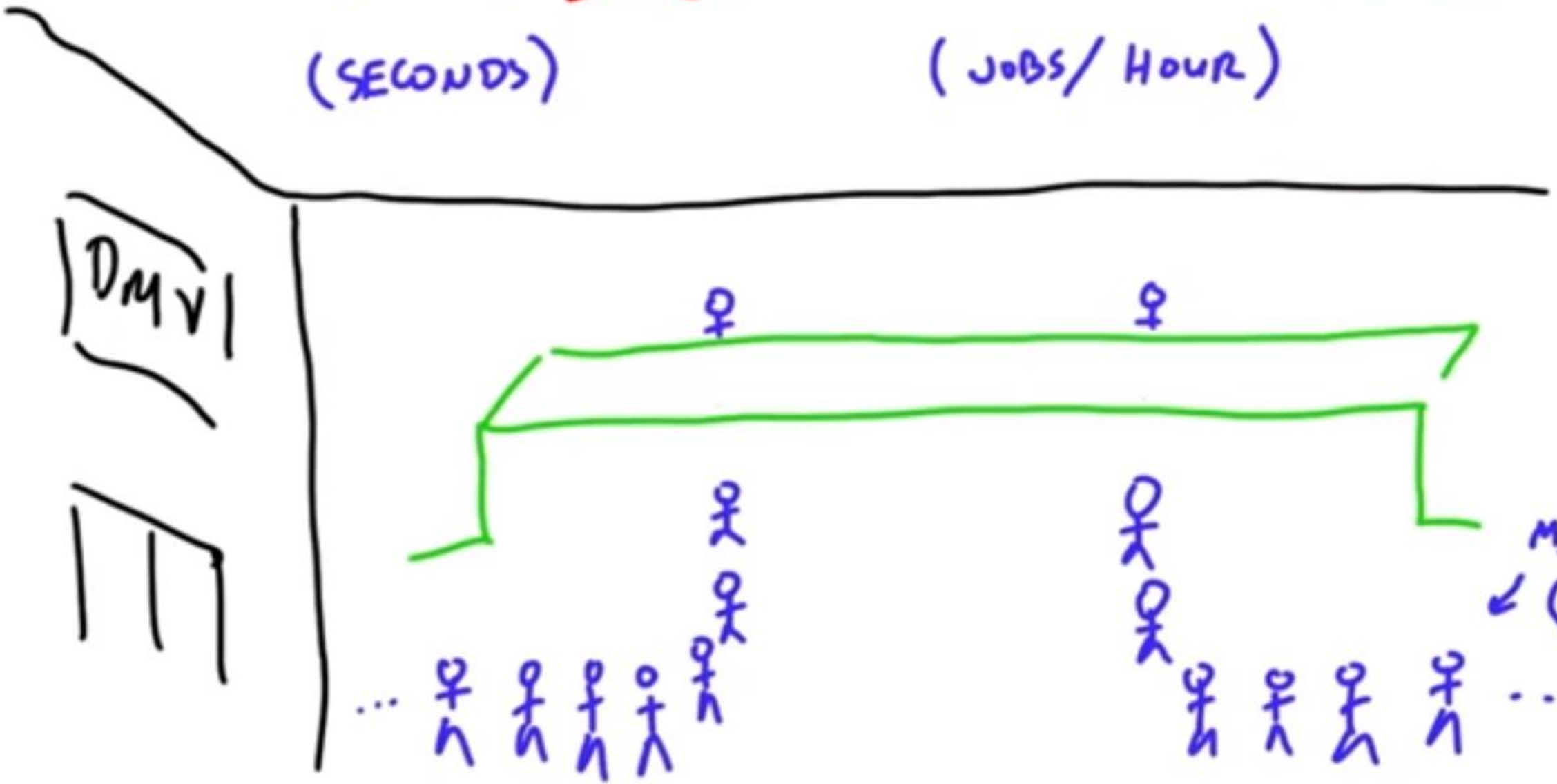
CPU

THROUGHPUT

(STUFF/TIME)

(JOBS/HOUR)

GPU.



$$d = 4500km$$

Car

$$2people\ at\ v = 200km/h$$

So we find for time to discover the latency:

$$v = \frac{d}{t}$$

$$tv = d \rightarrow t = \frac{d}{v}$$

Replacing

$$t = \frac{4500km}{200km/h} = \frac{4500km * h}{200km} = \frac{4500}{200}h = 22.5h$$

$$throughput = \frac{2people}{22.5h} = 0.0888people/h \cong 0.089people/h$$

Bus

$$40people\ at\ v = 50km/h$$

We find for time to discover the latency:

$$t = \frac{d}{v}$$

Replacing

$$t = \frac{4500km}{50km/h} = 90h$$

we find for throughput $(people/hour)$

$$throughput = \frac{40people}{90h} = 0.444people/h \cong 0.45people/h$$

CORE GPU DESIGN TENETS

1. LOTS OF SIMPLE COMPUTE UNITS
   TRADE SIMPLE CONTROL FOR MORE COMPUTE
2. EXPLICITLY PARALLEL PROGRAMMING MODEL
3. OPTIMIZE FOR THROUGHPUT NOT LATENCY.

GPUs FROM THE POINT OF VIEW OF THE SOFTWARE DEVELOPER

— IMPORTANCE OF PROGRAMMING IN PARALLEL

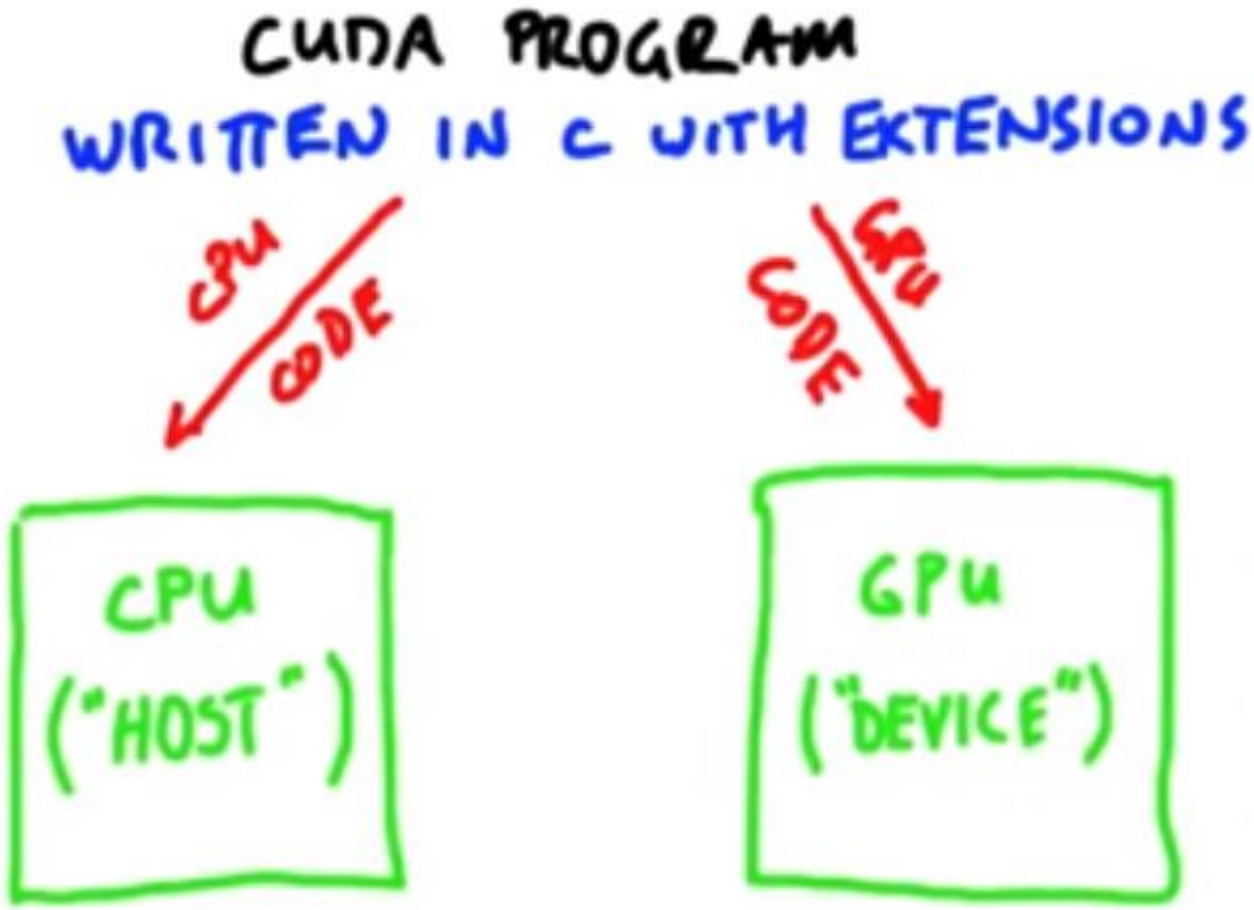
8 CORE IVY BRIDGE (INTEL)

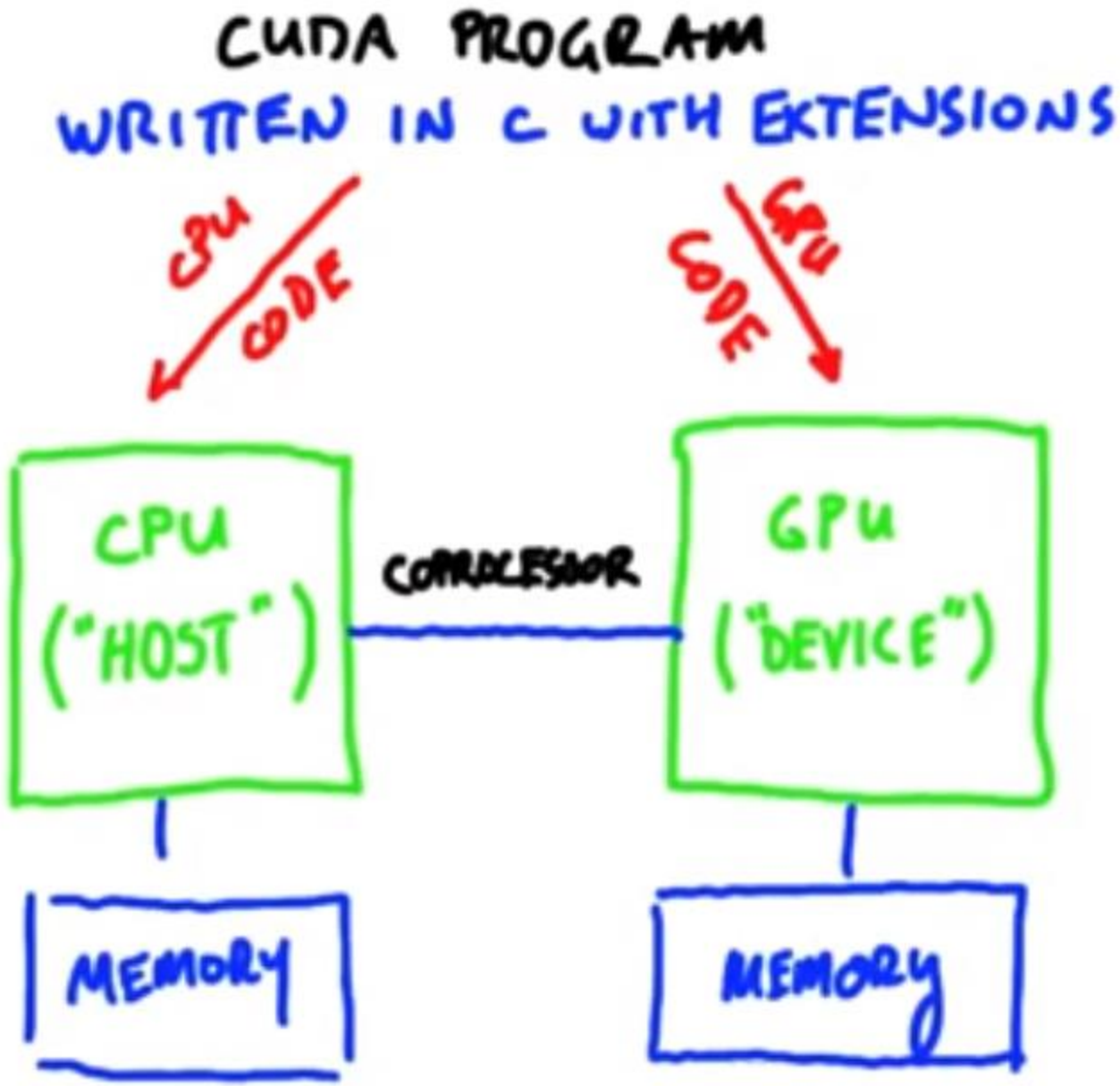
× 8-WIDE AVX VECTOR OPERATIONS / CORE

× 2 THREADS / CORE (HYPERTHREADING)

128-WAY PARALLELISM

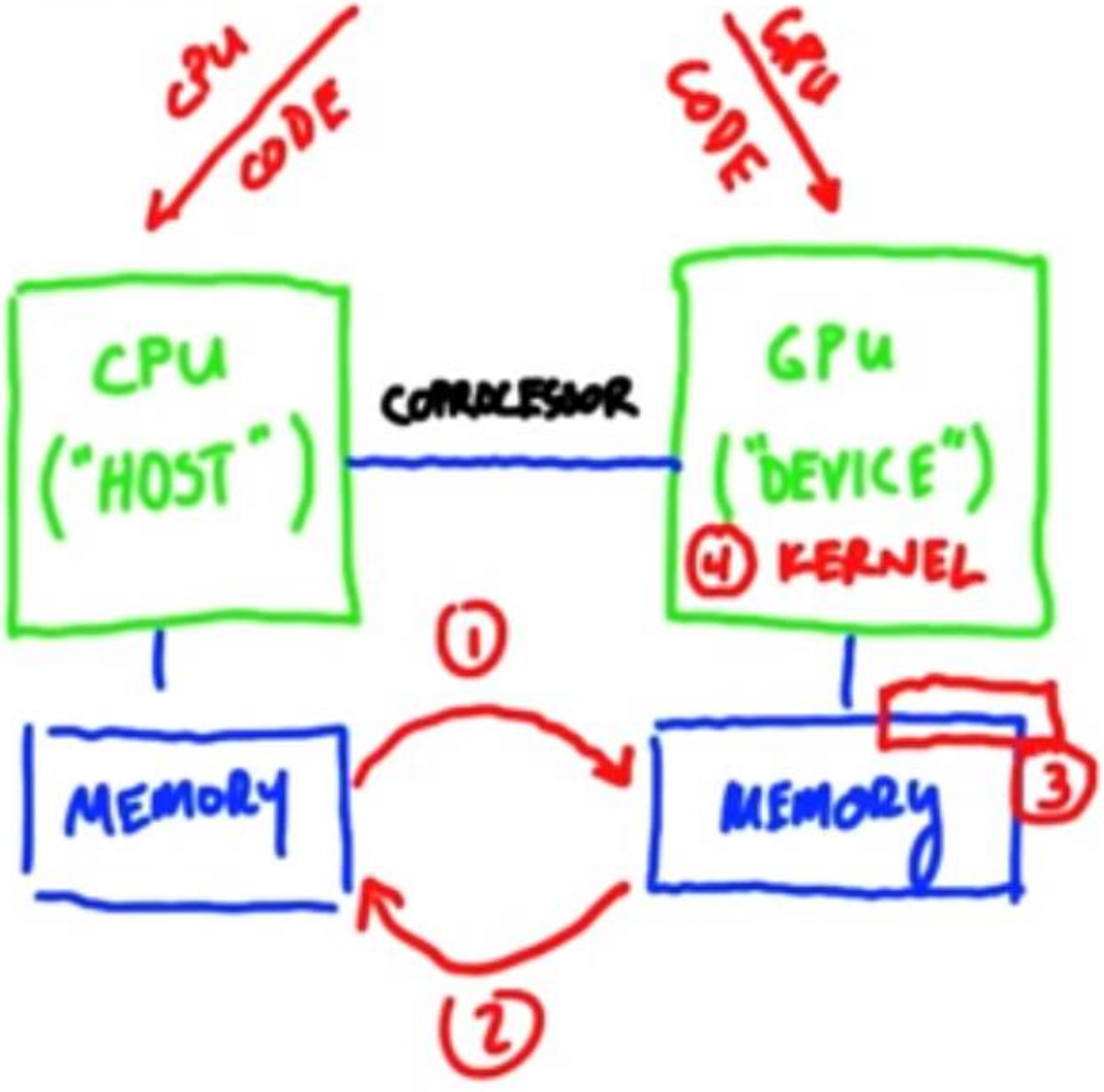
Heterogeneous System Architecture(HSA)


CUDA PROGRAM
WRITTEN IN C WITH EXTENSIONS
CPU CODE
GPU CODE
CPU ("HOST")
GPU ("DEVICE")


CUDA PROGRAM
WRITTEN IN C WITH EXTENSIONS
CPU CODE
GPU CODE
CPU ("HOST")
COPROCESSOR
GPU ("DEVICE")
MEMORY
MEMORY

CUDA PROGRAM

WRITTEN IN C WITH EXTENSIONS



1. DATA CPU → GPU
2. DATA GPU → CPU

(1),(2): cudaMemcpy

3. ALLOCATE GPU MEMORY

(3) cudaMalloc

4. LAUNCH KERNEL ON GPU

QUIZ THE GPU CAN DO THE FOLLOWING (T / F)

- [ ] INITIATE DATA SEND GPU → CPU
- [x] RESPOND TO CPU REQUEST TO SEND DATA GPU → CPU
- [ ] INITIATE DATA REQUEST CPU → GPU
- [x] RESPOND TO CPU REQUEST TO RECV DATA CPU → GPU
- [x] COMPUTE A KERNEL LAUNCHED BY CPU
- [ ] COMPUTE A KERNEL LAUNCHED BY GPU.

Now let’s read the introduction to CUDA C located at

report_src/additional_resources/Introduction to CUDA C - GPU Technology Conference.pdf