ENGR 101
Engineering Technology

Dr. Kerese Manueli

School of Engineering and Computer Science
Victoria University of Wellington

Victoria
UNIVERSITY OF WELLINGTON
Te Whare Wānanga
o te Ūpoko o te Ika a Māui
CAPITAL CITY UNIVERSITY

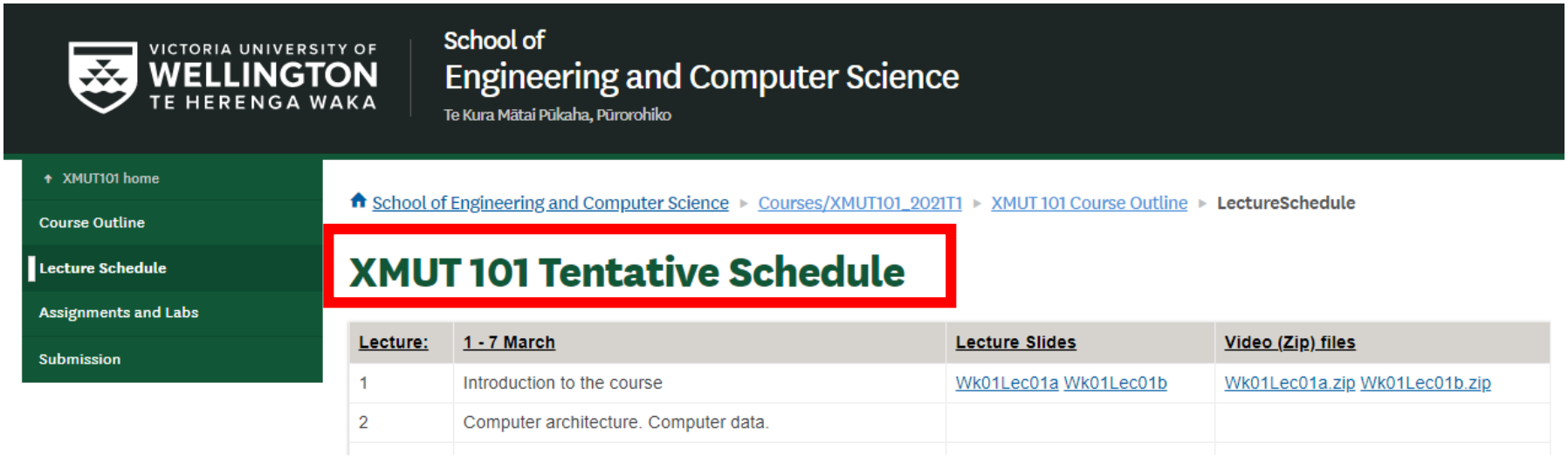
Week 1 Lecture 2b

- Course web page:

https://ecs.wgtn.ac.nz/Courses/XMUT101_2021T1/

- kerese@ecs.vuw.ac.nz

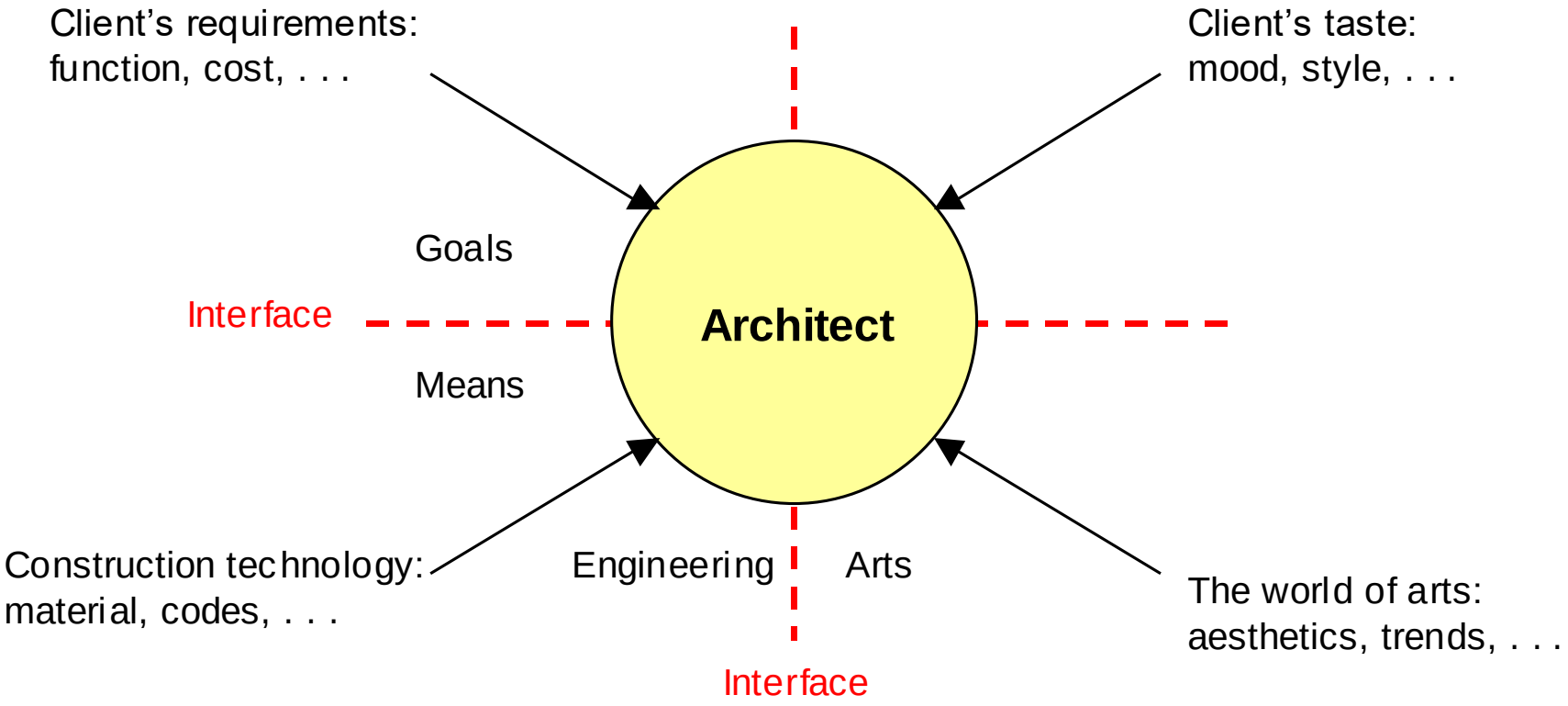
https://ecs.wgtn.ac.nz/Courses/XMUT101_2021T1/LectureSchedule



Lecture:	1 - 7 March	Lecture Slides	Video (Zip) files
1	Introduction to the course	Wk01Lec01a Wk01Lec01b	Wk01Lec01a.zip Wk01Lec01b.zip
2	Computer architecture. Computer data.		

What is (Computer) Architecture?

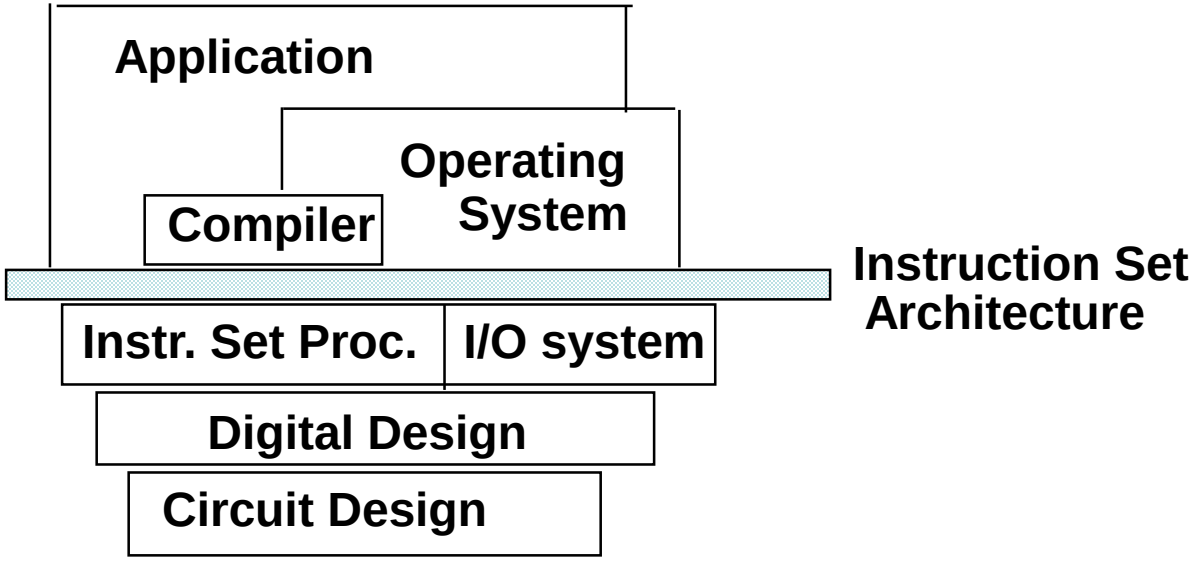
Like a building architect, whose place at the engineering/arts and goals/means interfaces is seen in this diagram, a computer architect reconciles many conflicting or competing demands.



Computer Architecture

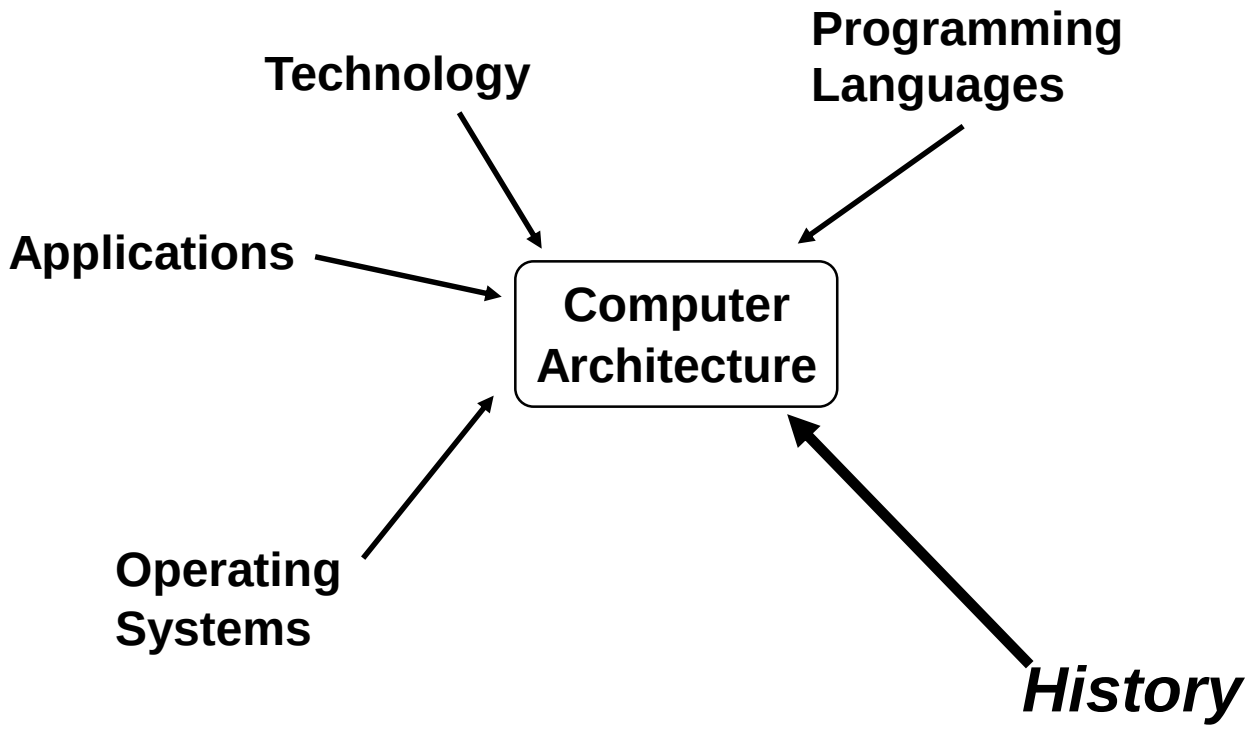
A system concept integrating software, hardware, and firmware to specify the design of computing systems

- **Co-ordination of *levels of abstraction***

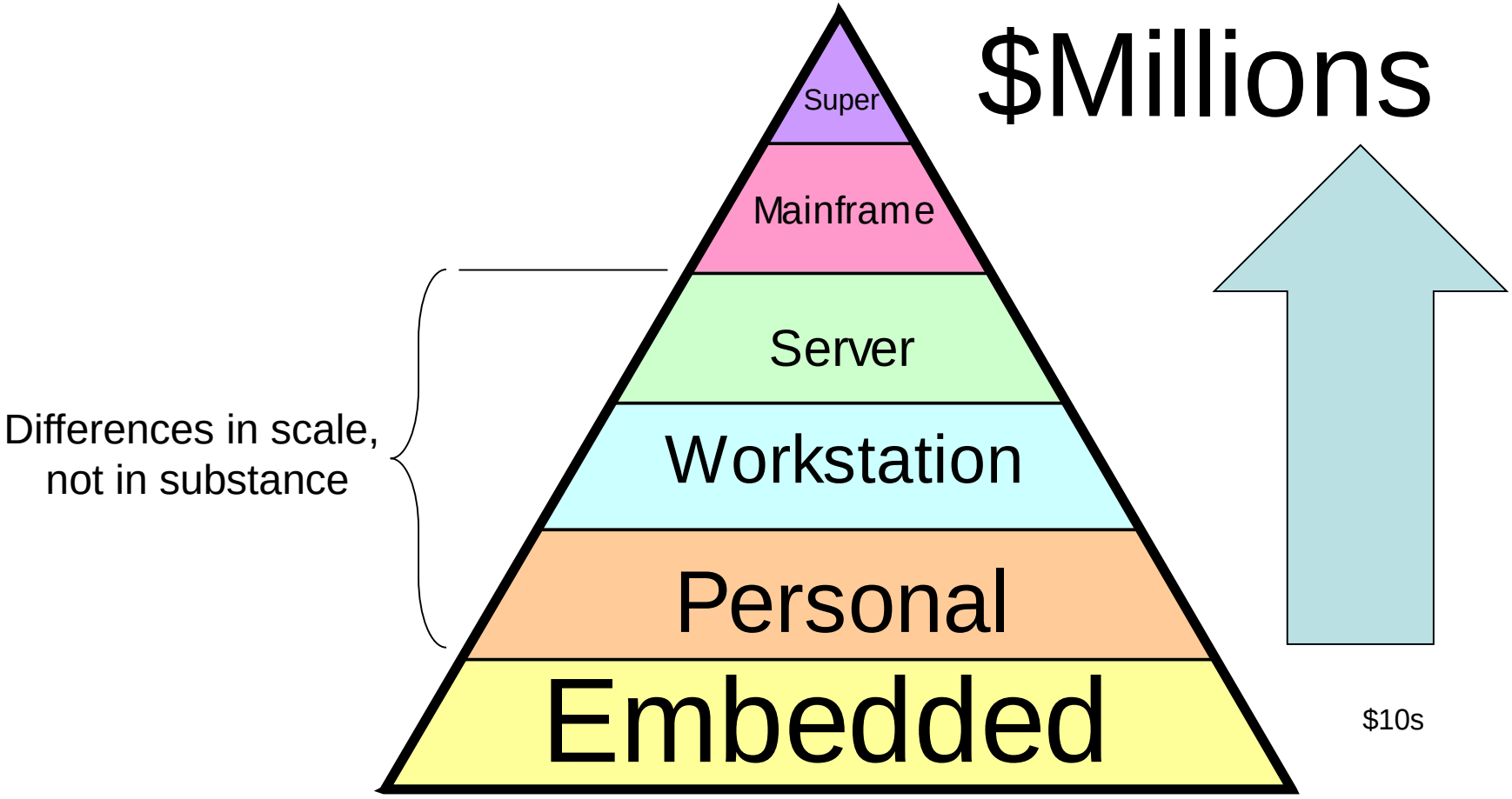


- **Under a set of rapidly changing *Forces***

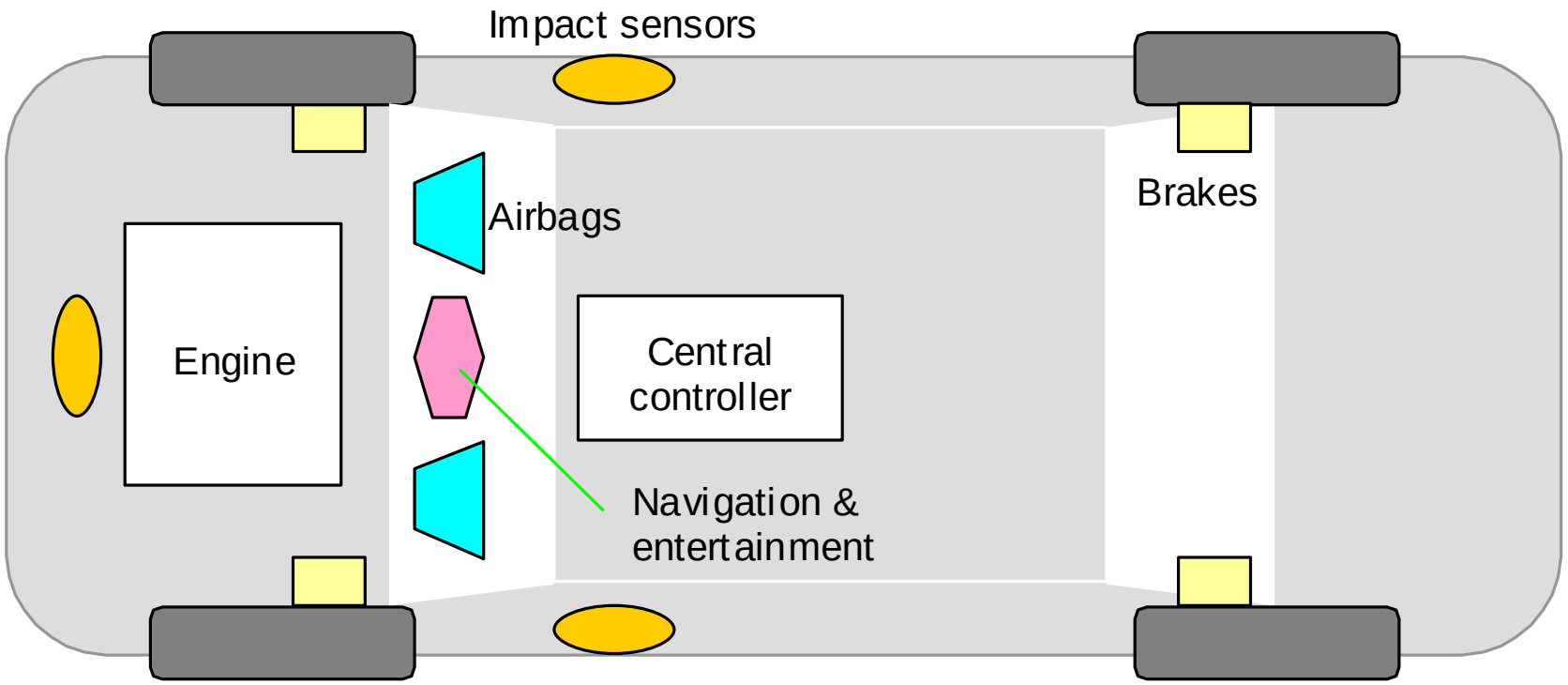
Forces on Computer Architecture



Computer Price/Performance Pyramid

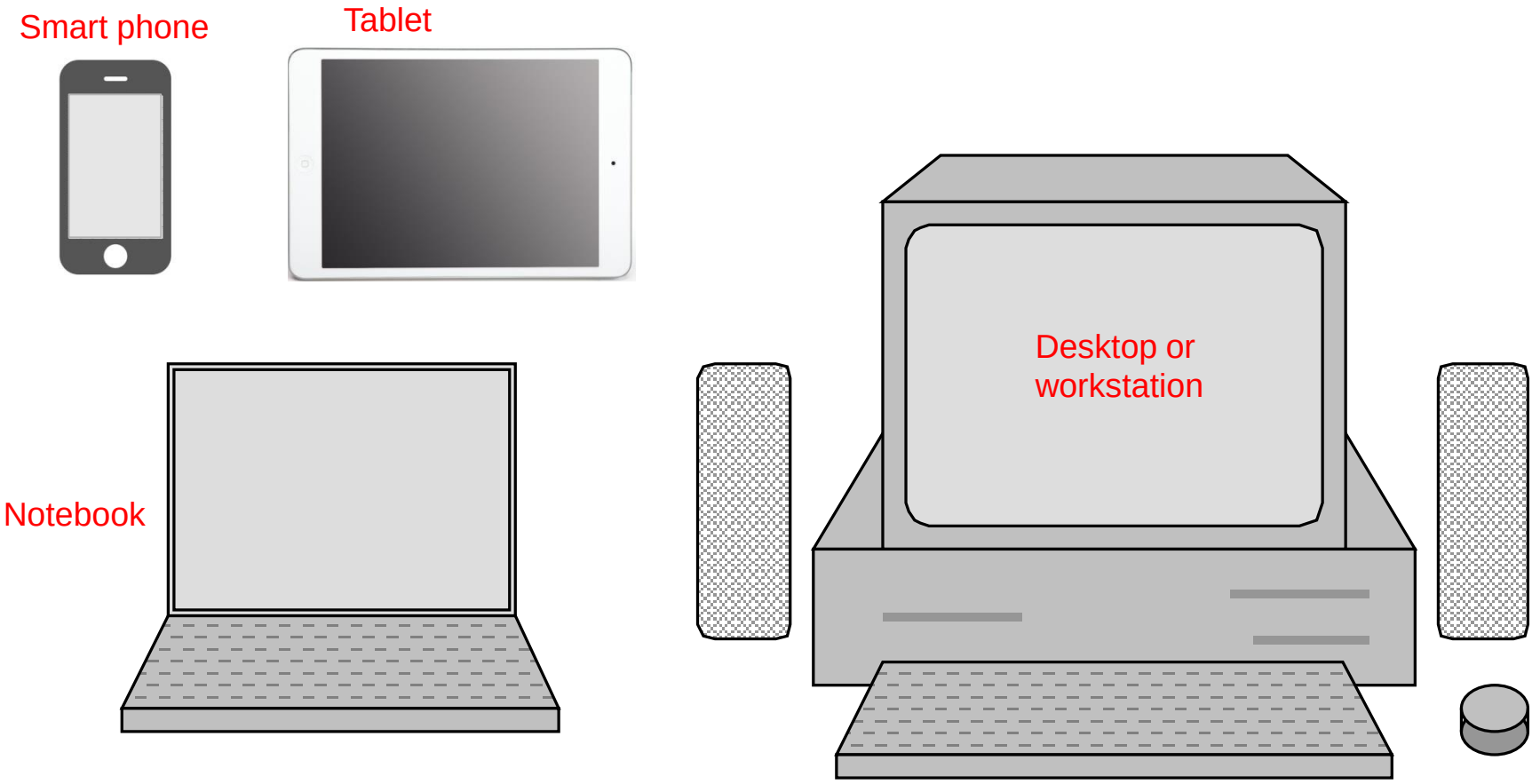


Automotive Embedded Computers



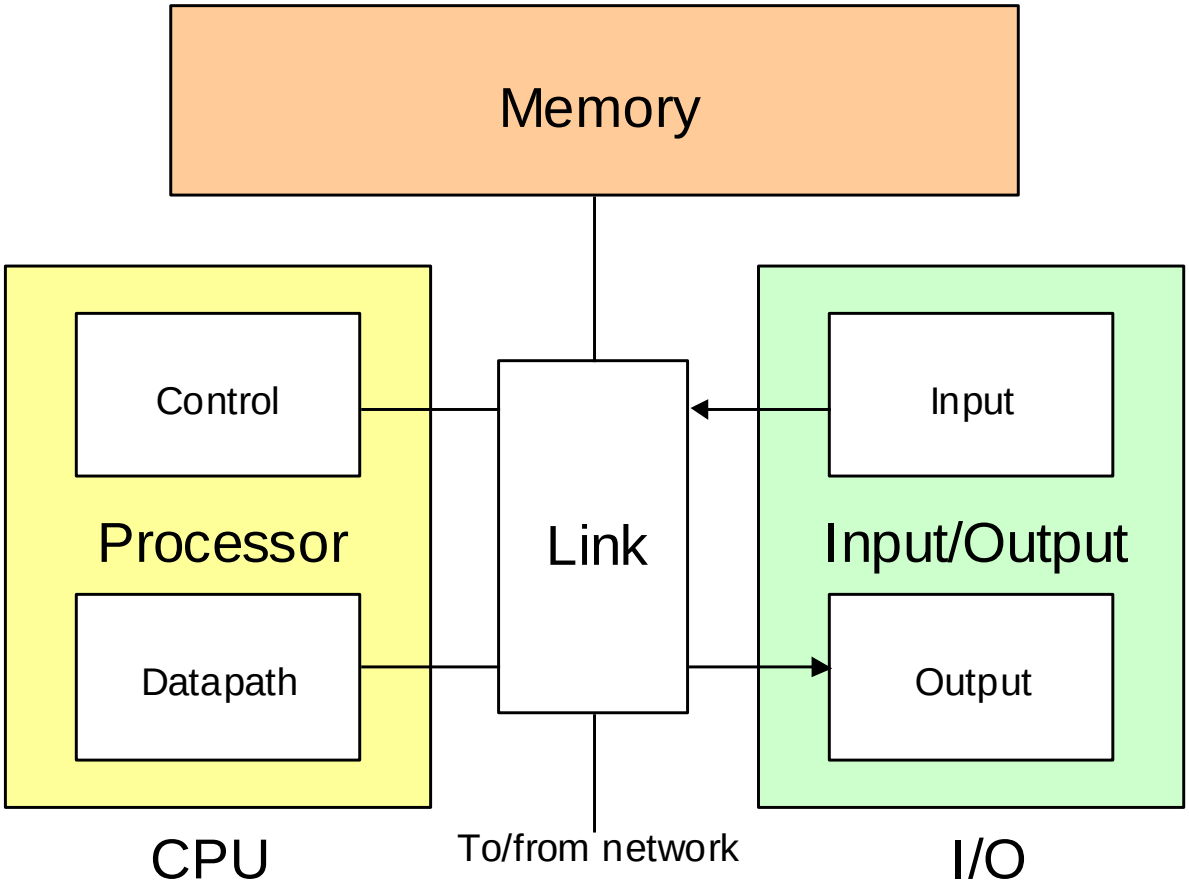
Embedded computers are ubiquitous, yet invisible. They are found in automobiles, home appliances, and many other places.

Personal Computers and Workstations



Notebooks, a common class of portable computers, are much smaller than desktops but offer substantially the same capabilities.

Digital Computer Subsystems



The six main units of a digital computer are the CPU, which comprised of the control unit and the datapath, memory, the I/O devices which are linked together by a simple bus or a more elaborate network.

Generations of Progress

The 5 generations of digital computers.

Generation (begun)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment

Generations of Progress

The 5 generations of digital computers, and their ancestors.

Generation (begun)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet

Generations of Progress

The 5 generations of digital computers, and their ancestors.

Generation (begun)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet
2 (1960s)	Transistor	Magnetic core	Drum, printer, text terminal	Room-size mainframe

Generations of Progress

The 5 generations of digital computers, and their ancestors.

Generation (begun)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet
2 (1960s)	Transistor	Magnetic core	Drum, printer, text terminal	Room-size mainframe
3 (1970s)	SSI/MSI	RAM/ROM chip	Disk, keyboard, video monitor	Desk-size mini

SSI – Small Scale Integration { logic gates: AND. OR, NAND, NOR; 1-10 / chip}

MSI – Medium Scale Integration { Flip-flops, adders/counters, MUX & DEMUX

Generations of Progress

The 5 generations of digital computers, and their ancestors.

Generation (begun)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet
2 (1960s)	Transistor	Magnetic core	Drum, printer, text terminal	Room-size mainframe
3 (1970s)	SSI/MSI	RAM/ROM chip	Disk, keyboard, video monitor	Desk-size mini
4 (1980s)	LSI/VLSI	SRAM/DRAM	Network, CD, mouse,sound	Desktop/ laptop micro

LSI – Large Scale Integration { 500 – 20,000 transistors/chip}

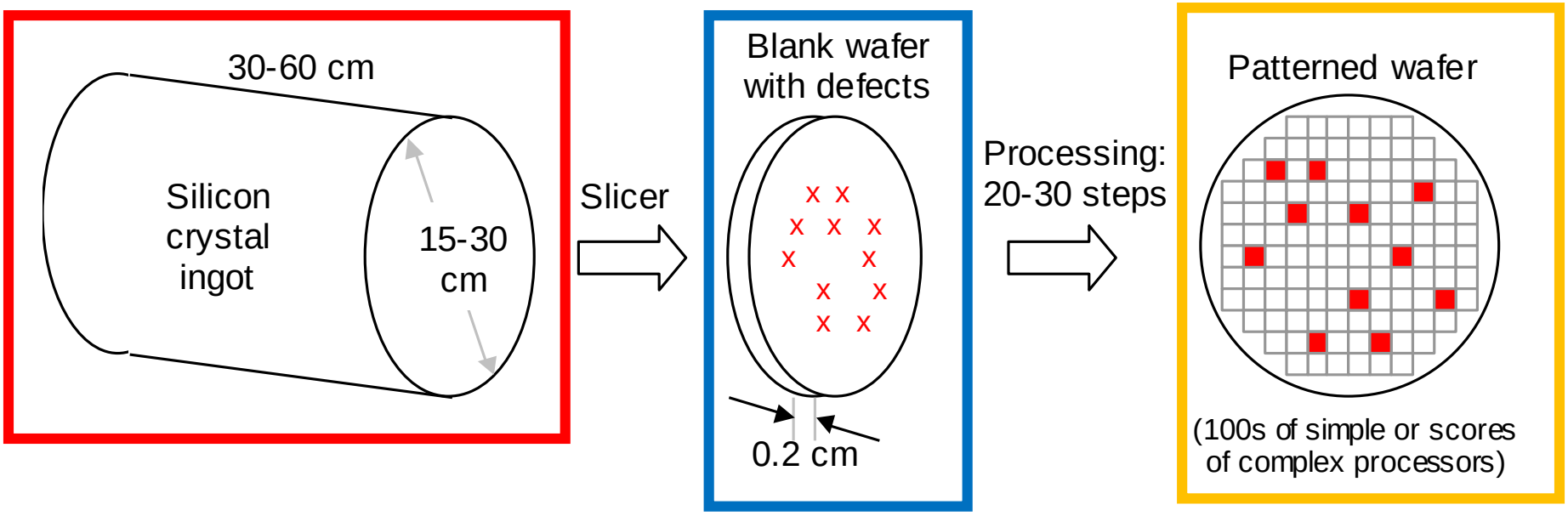
VLSI – Very Large Scale Integration { 20,000 – 1,000,000,000 transistors/chip

Generations of Progress

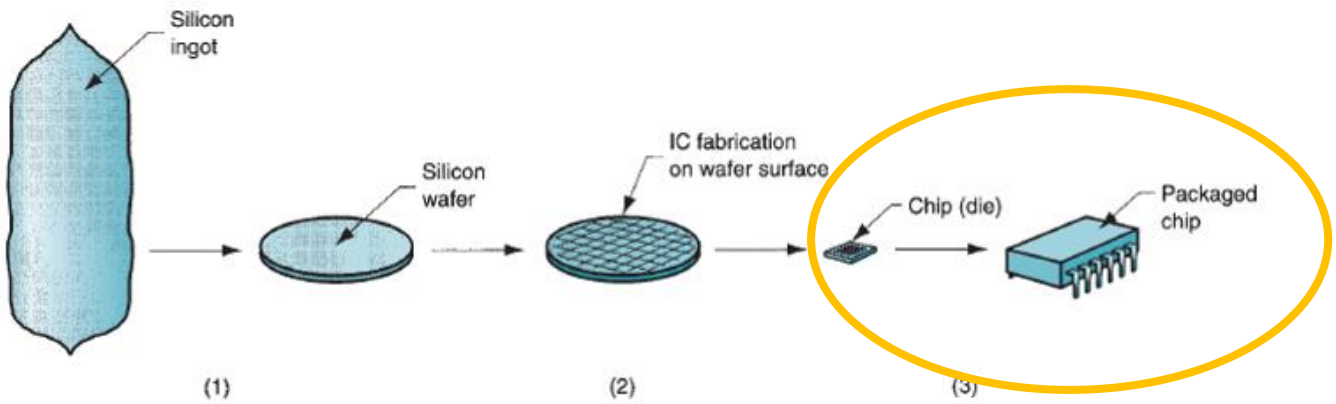
The 5 generations of digital computers, and their ancestors.

Generation (begun)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet
2 (1960s)	Transistor	Magnetic core	Drum, printer, text terminal	Room-size mainframe
3 (1970s)	SSI/MSI	RAM/ROM chip	Disk, keyboard, video monitor	Desk-size mini
4 (1980s)	LSI/VLSI	SRAM/DRAM	Network, CD, mouse,sound	Desktop/ laptop micro
5 (1990s)	ULSI/GSI/ WSI, SOC	SDRAM, flash	Sensor/actuator, point/click	Invisible, embedded

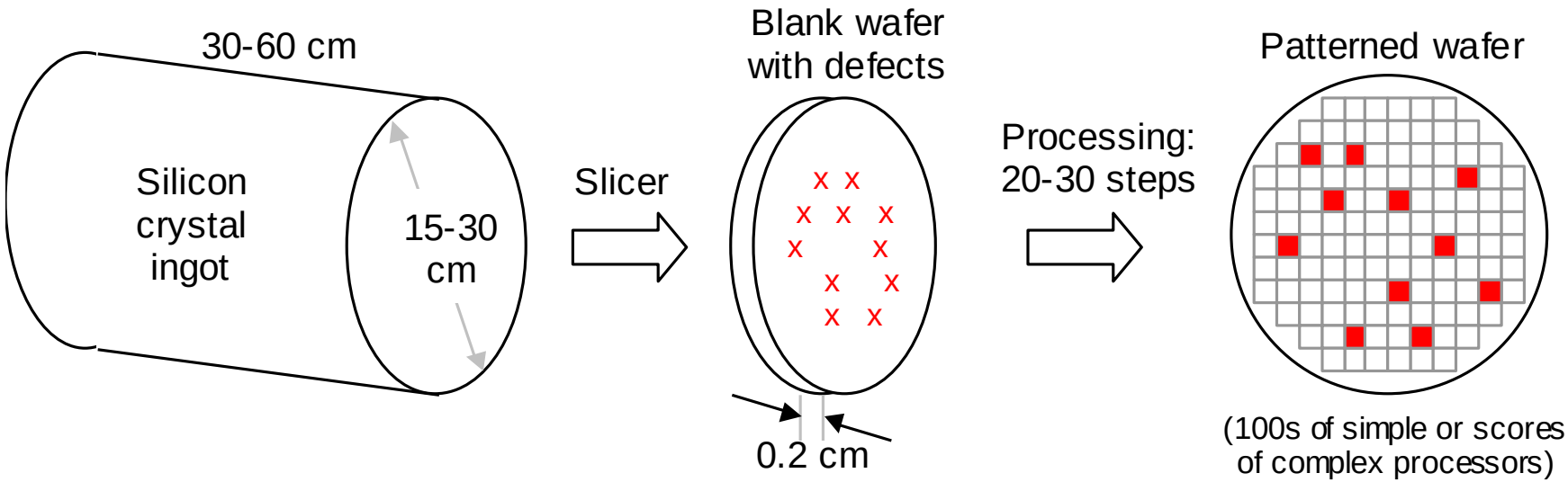
IC Production and Yield

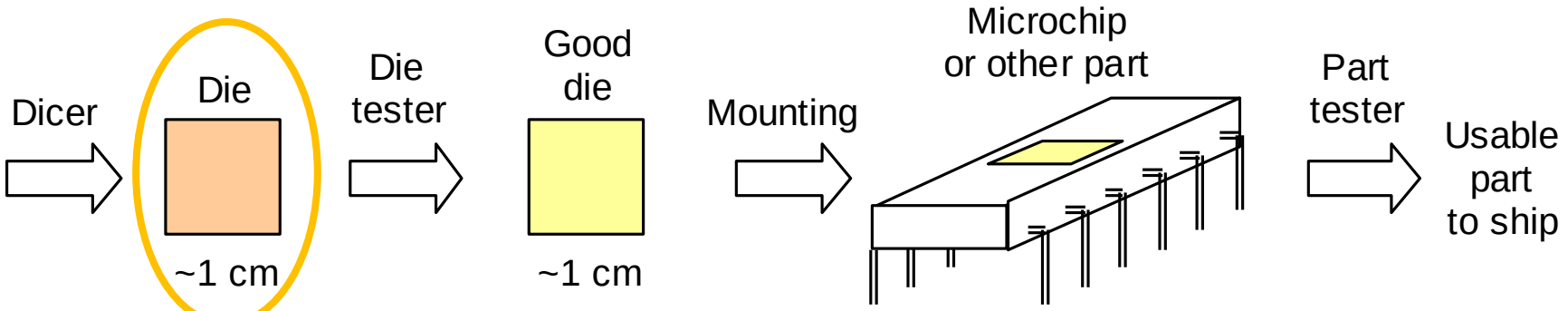


The manufacturing process for an IC part.



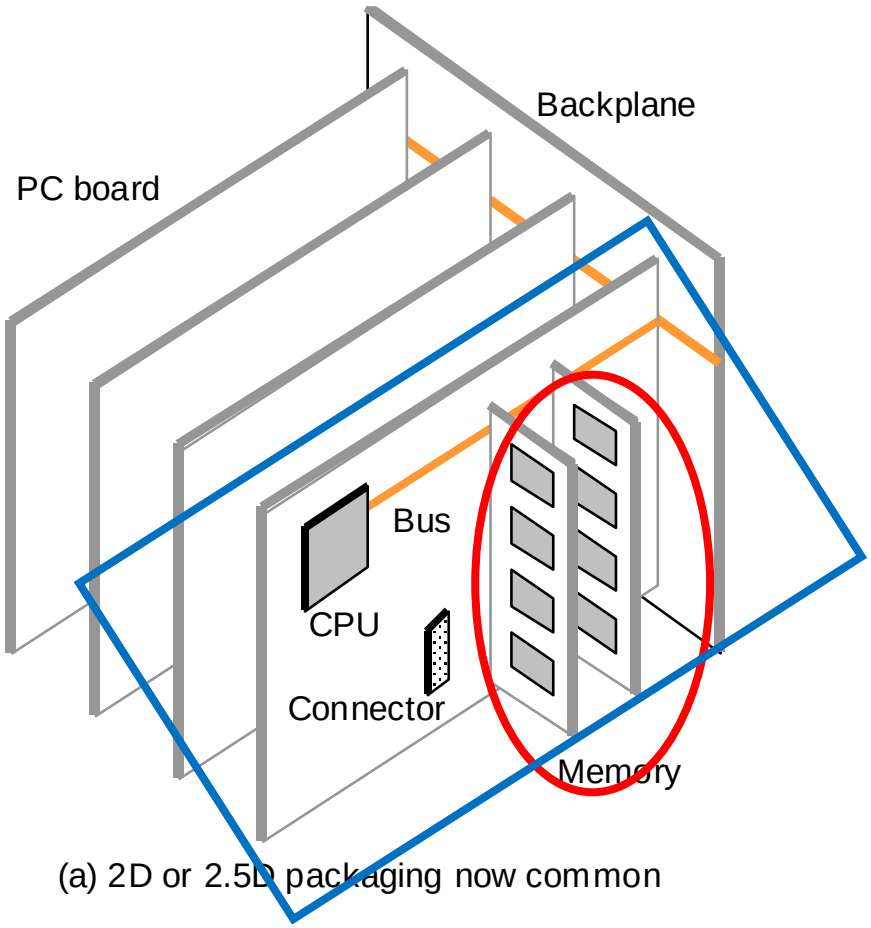
IC Production and Yield





The manufacturing process for an IC part.

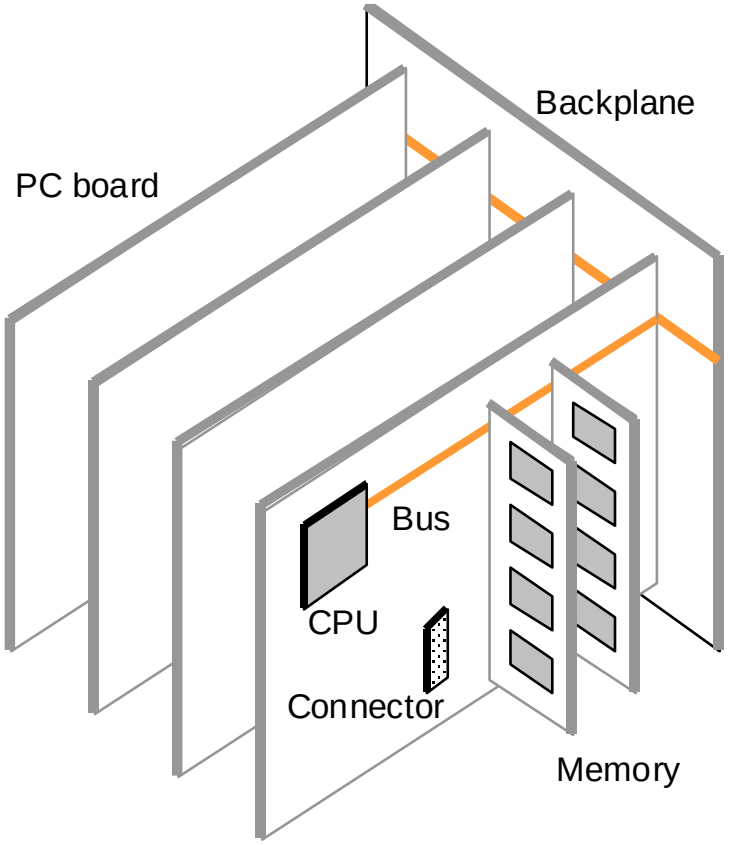
Processor and Memory Technologies



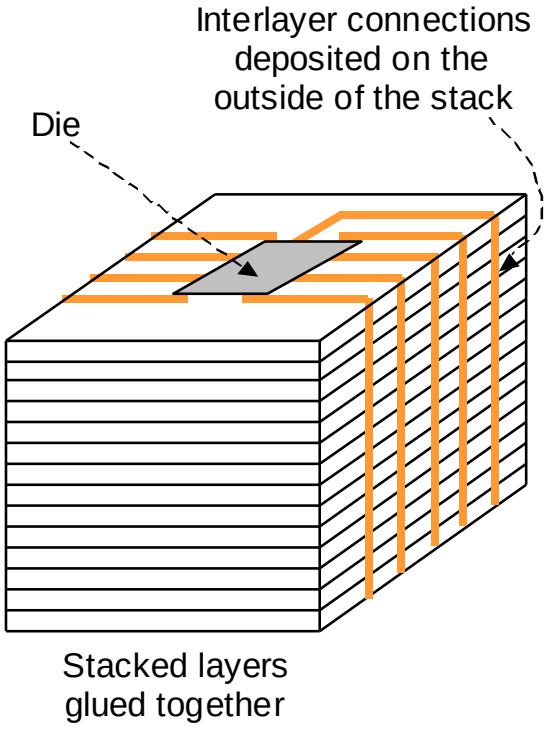
(a) 2D or 2.5D packaging now common

Packaging of processor, memory, and other components.

Processor and Memory Technologies



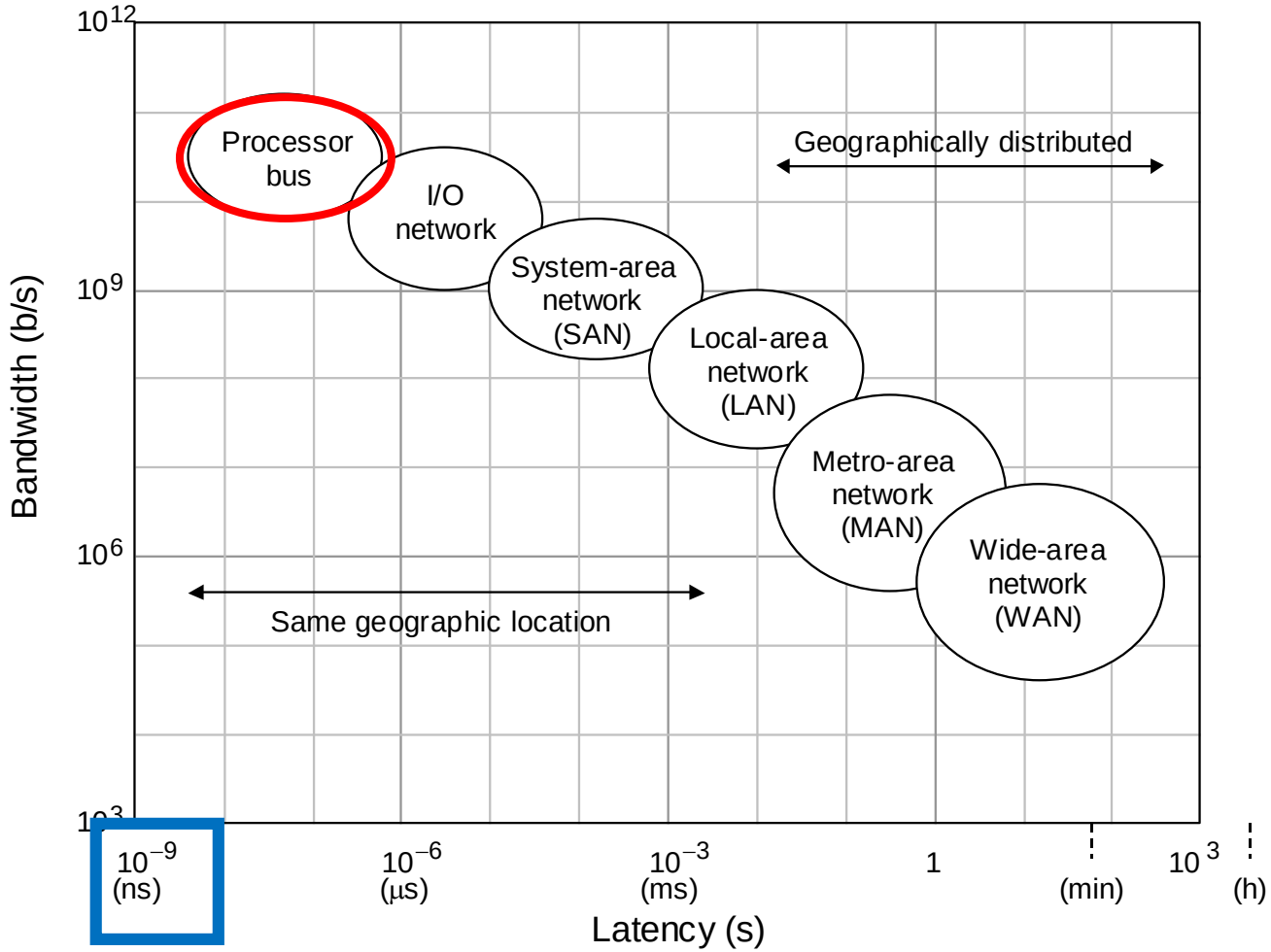
(a) 2D or 2.5D packaging now common



(b) 3D packaging of the future

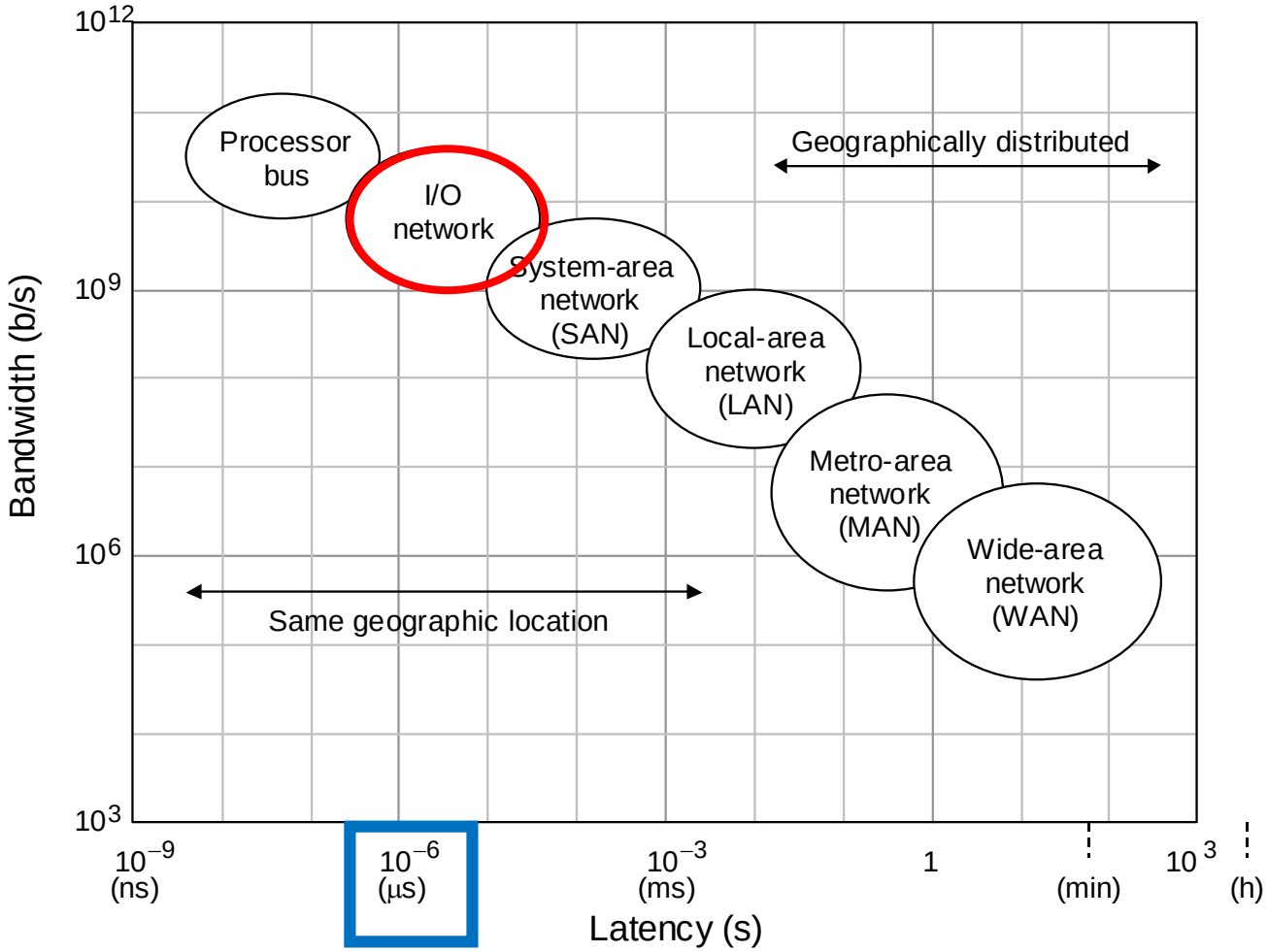
Packaging of processor, memory, and other components.

Communication Technologies



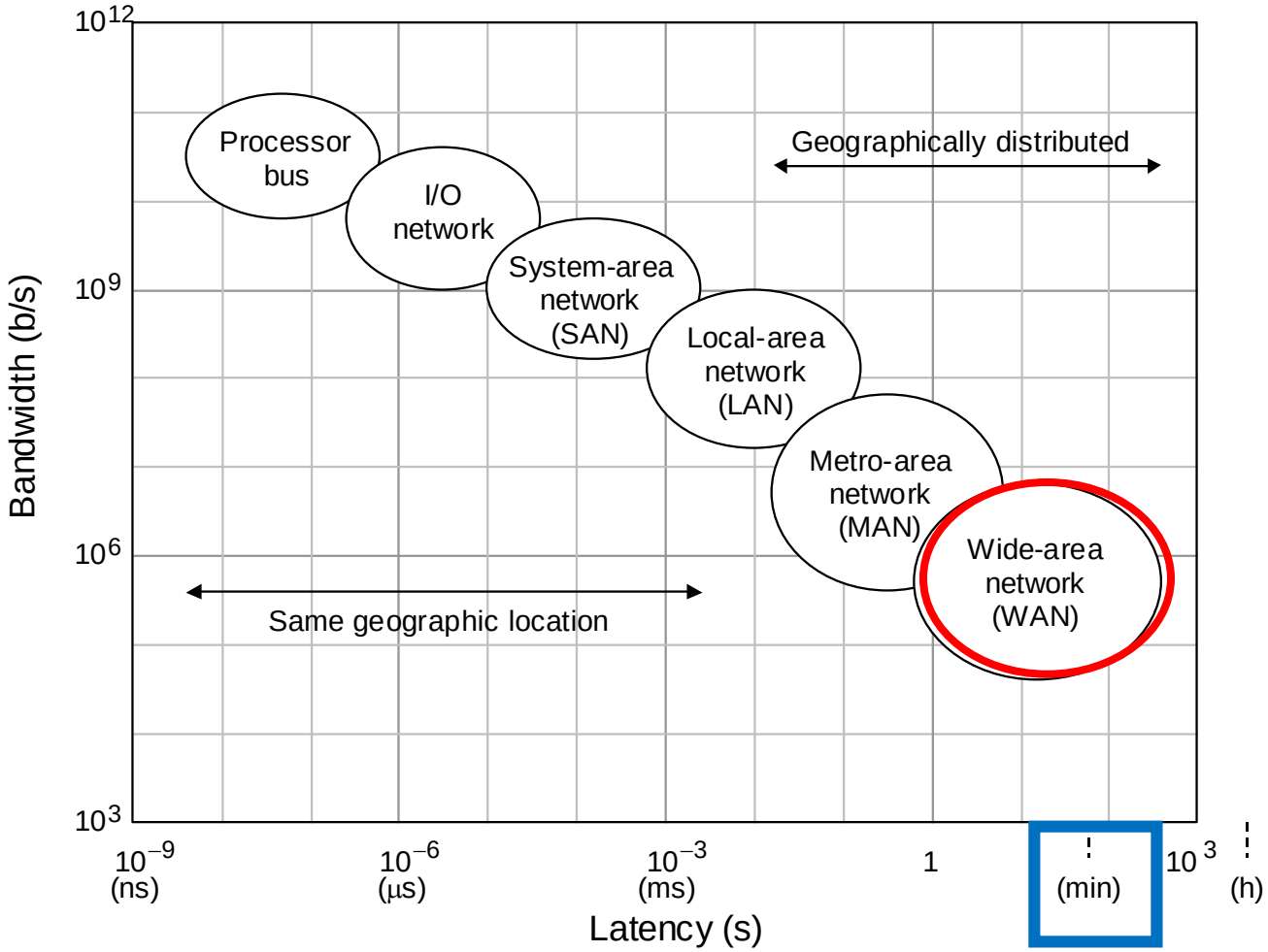
Latency and bandwidth characteristics of different classes of communication links.

Communication Technologies



Latency and bandwidth characteristics of different classes of communication links.

Communication Technologies



Latency and bandwidth characteristics of different classes of communication links.

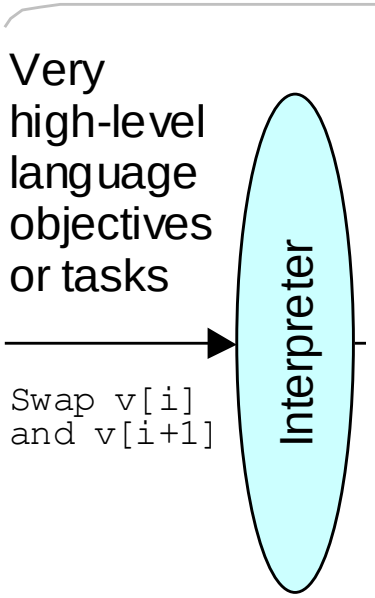
High- vs Low-Level Programming

More abstract, machine-independent;
easier to write, read, debug, or maintain

Models and abstractions in programming.

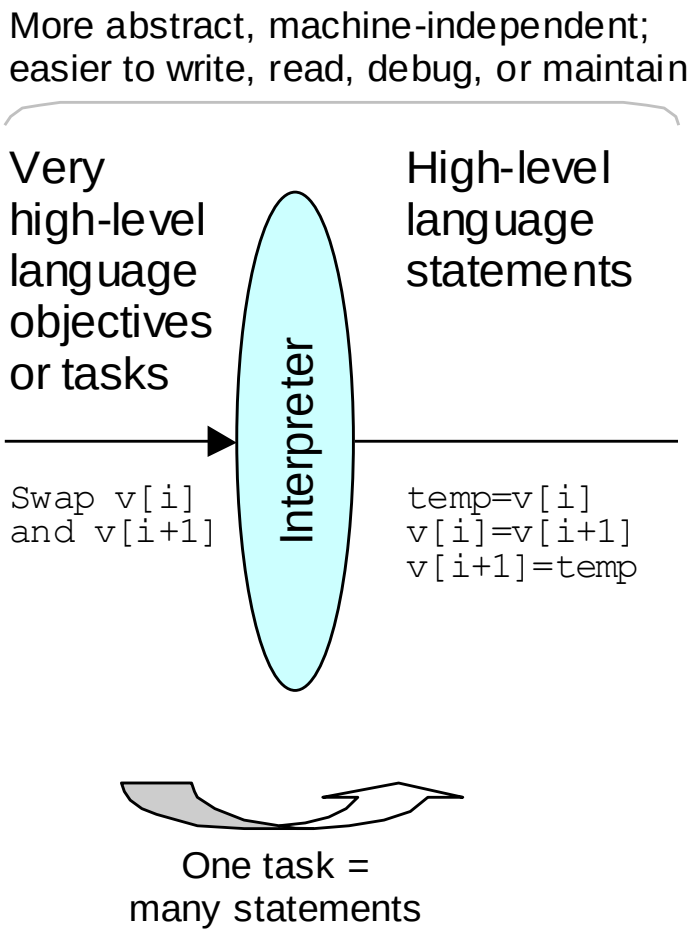
High- vs Low-Level Programming

More abstract, machine-independent;
easier to write, read, debug, or maintain



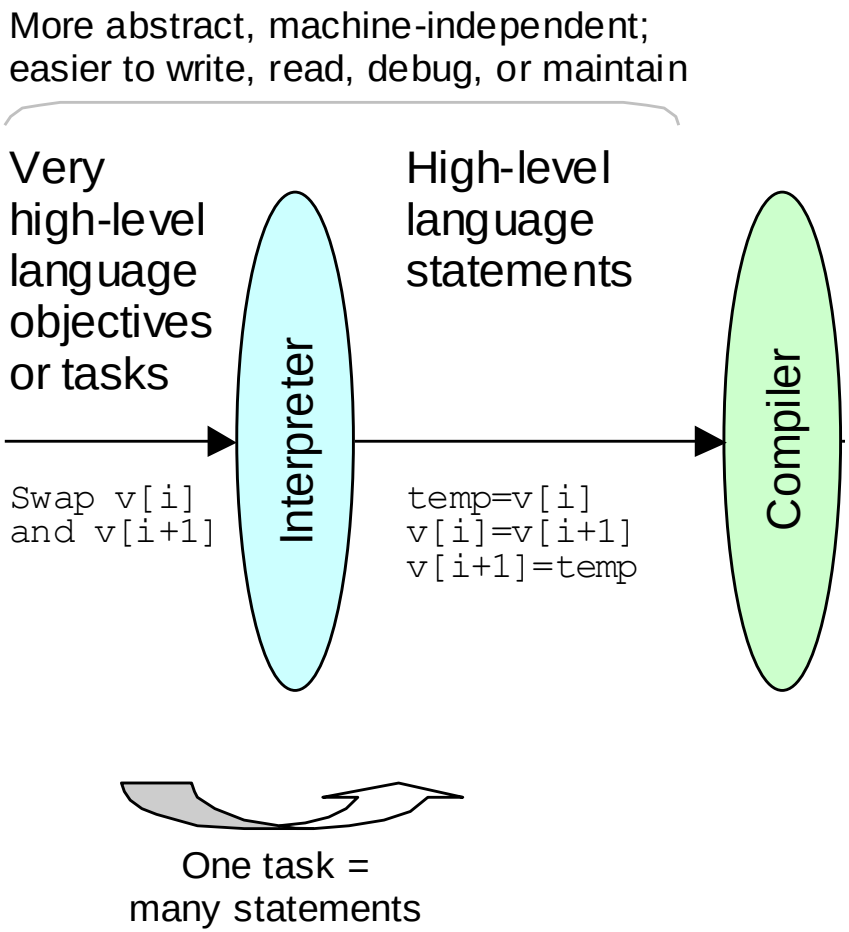
Models and abstractions in programming.

High- vs Low-Level Programming



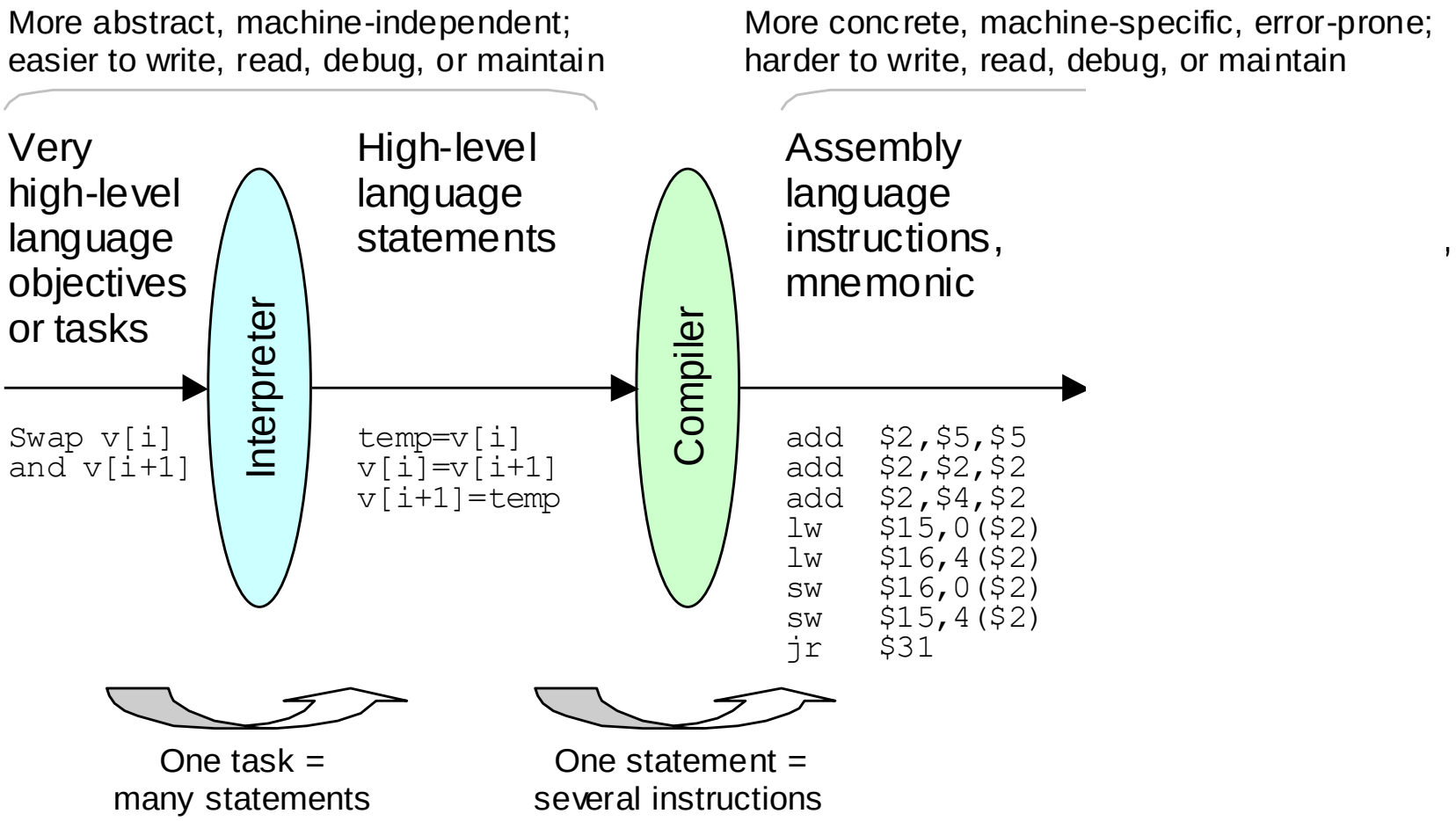
Models and abstractions in programming.

High- vs Low-Level Programming



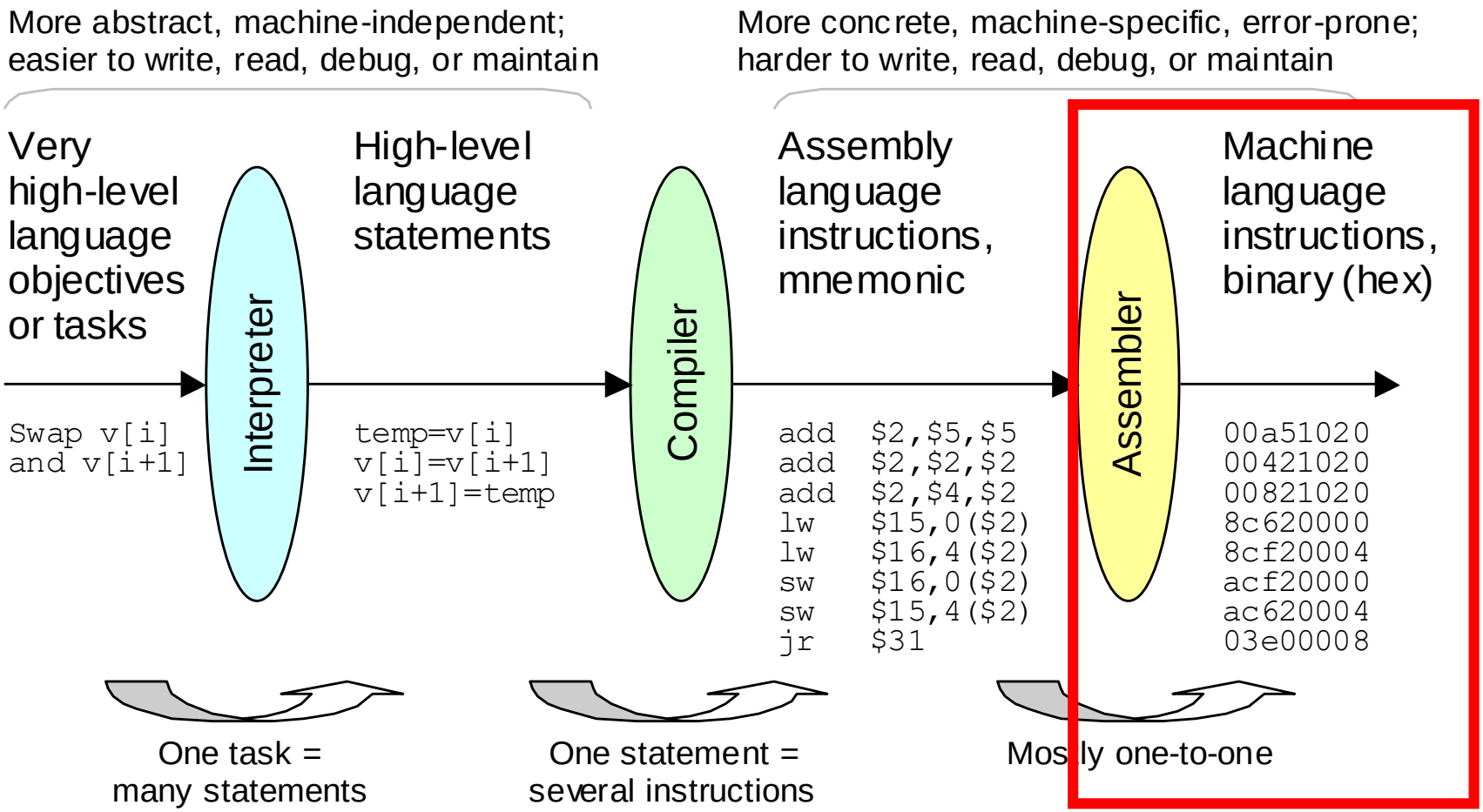
Models and abstractions in programming.

High- vs Low-Level Programming



Models and abstractions in programming.

High- vs Low-Level Programming

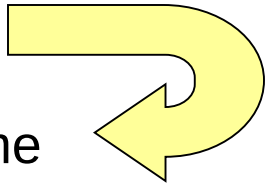


Models and abstractions in programming.

Concepts of Performance and Speedup

Performance = 1 / Execution time

Performance = 1 / CPU execution time



is simplified to

Concepts of Performance and Speedup

Performance = 1 / Execution time

is simplified to

Performance = 1 / CPU execution time

(Performance of M_1) / (Performance of M_2) = Speedup of M_1 over M_2
= (Execution time of M_2) / (Execution time M_1)

Terminology: M_1 is *x* times **as fast as** M_2 (e.g., 1.5 times as fast)
M_1 is 100(*x* – 1)% **faster than** M_2 (e.g., 50% faster)

Concepts of Performance and Speedup

Performance = 1 / Execution time

is simplified to

Performance = 1 / CPU execution time

(Performance of M_1) / (Performance of M_2) = Speedup of M_1 over M_2

= (Execution time of M_2) / (Execution time M_1)

Terminology: M_1 is *x* times as fast as M_2 (e.g., 1.5 times as fast)

M_1 is 100(*x* – 1)% faster than M_2 (e.g., 50% faster)

CPU time = Instructions × (Cycles per instruction) × (Secs per cycle)

= Instructions × CPI / (Clock rate)

Instruction count, CPI, and clock rate are not completely independent, so improving one by a given factor may not lead to overall execution time improvement by the same factor.

Elaboration on the CPU Time Formula

CPU time = Instructions × (Cycles per instruction) × (Secs per cycle)

= Instructions × Average CPI / (Clock rate)

Instructions: Number of instructions executed, not number of instructions in a program (dynamic count)

Elaboration on the CPU Time Formula

CPU time = Instructions × (Cycles per instruction) × (Secs per cycle)

= Instructions × Average CPI / (Clock rate)

Instructions: Number of instructions executed, not number of instructions in our program (dynamic count)

Average CPI: Is calculated based on the dynamic instruction mix and knowledge of how many clock cycles are needed to execute various instructions (or instruction classes)

Elaboration on the CPU Time Formula

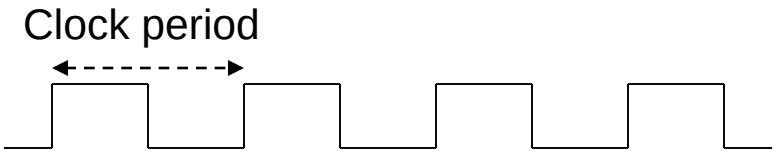
CPU time = Instructions × (Cycles per instruction) × (Secs per cycle)

= Instructions × Average CPI / (Clock rate)

Instructions: Number of instructions executed, not number of instructions in our program (dynamic count)

Average CPI: Is calculated based on the dynamic instruction mix and knowledge of how many clock cycles are needed to execute various instructions (or instruction classes)

Clock rate: 1 GHz = 10^9 cycles / s (cycle time 10^{-9} s = 1 ns)
200 MHz = 200 × 10^6 cycles / s (cycle time = 5 ns)



Week 1 Lecture 2b

- Introduction to Engineering Technology
  - Computer architecture
  - Computer prices and performance pyramid
  - Generational progress
  - IC manufacturing process
  - Computer languages, performance and speed

- Course web page:

  https://ecs.wgtn.ac.nz/Courses/XMUT101_2021T1/

- kerese@ecs.vuw.ac.nz