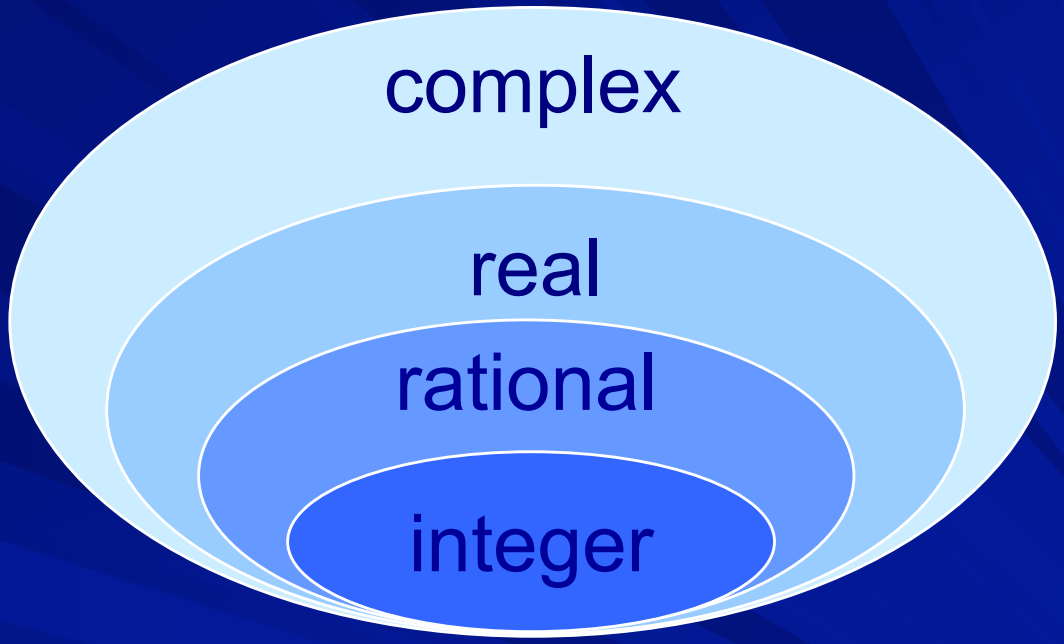
# COL216
# Computer Architecture

## Concluding Remarks:
## Floating point numbers and arithmetic
## 31st March 2022

# Need to go beyond integers

- integer 7
- rational 5/8
- real $\sqrt{3}$
- complex 2 - 3 i




Extremely large and small values:

- distance pluto - sun = $5.9\ 10^{12}$ m
- mass of electron = $9.1 \times 10^{-28}$ gm

# Representing fractions

- Integer pairs (for rational numbers)

| 5 | 8 | = 5/8 |
|---|---|---|

- Strings with explicit decimal point

| - | 2 | 4 | 7 | . | 0 | 9 |
|---|---|---|---|---|---|---|

- Implicit point at a fixed position

| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

implicit point

- Floating point

  fraction x base $^{\text{power}}$

# Numbers with binary point

$101.11 = 1x2^2 + 0x2^1 + 1x2^0 + 1x2^{-1} + 1x2^{-2}$

$= 4 + 1 + 0.5 + 0.25 = 5.75_{10}$

$0.6 = 0.1001100110011001100110011001.....$

$.6 \times 2 = 1 + .2$

$.2 \times 2 = 0 + .4$

$.4 \times 2 = 0 + .8$

$.8 \times 2 = 1 + .6$

# FP numbers with base = 2

$(-1)^S \times F \times 2^E$

S = sign

F = fraction (fixed point number)
usually called mantissa or significand

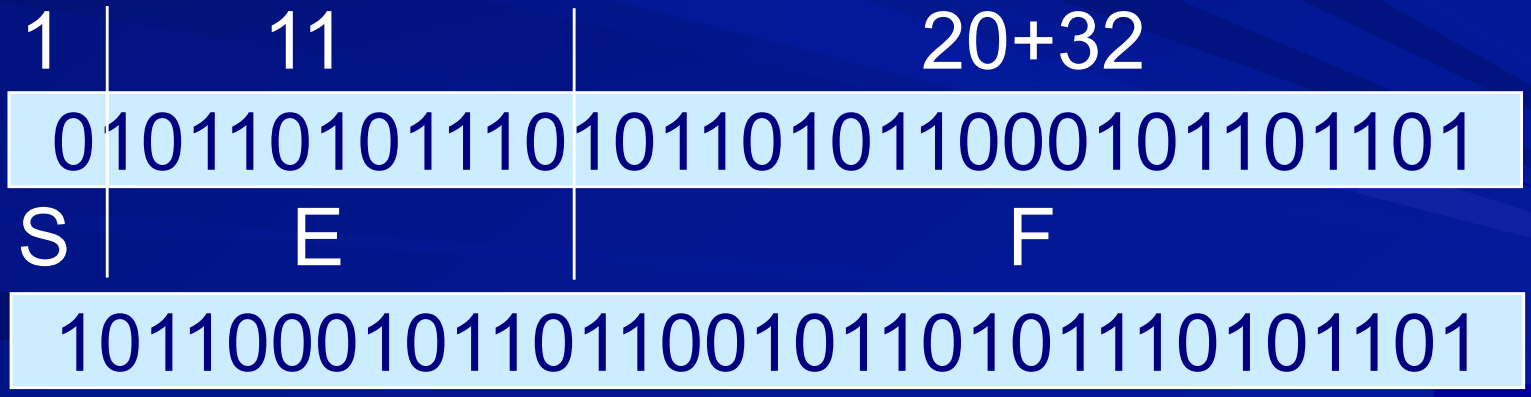
E = exponent (positive or negative integer)

- How to divide a word into S, F and E?
- How to represent S, F and E?

# IEEE 754 standard

- Single precision numbers

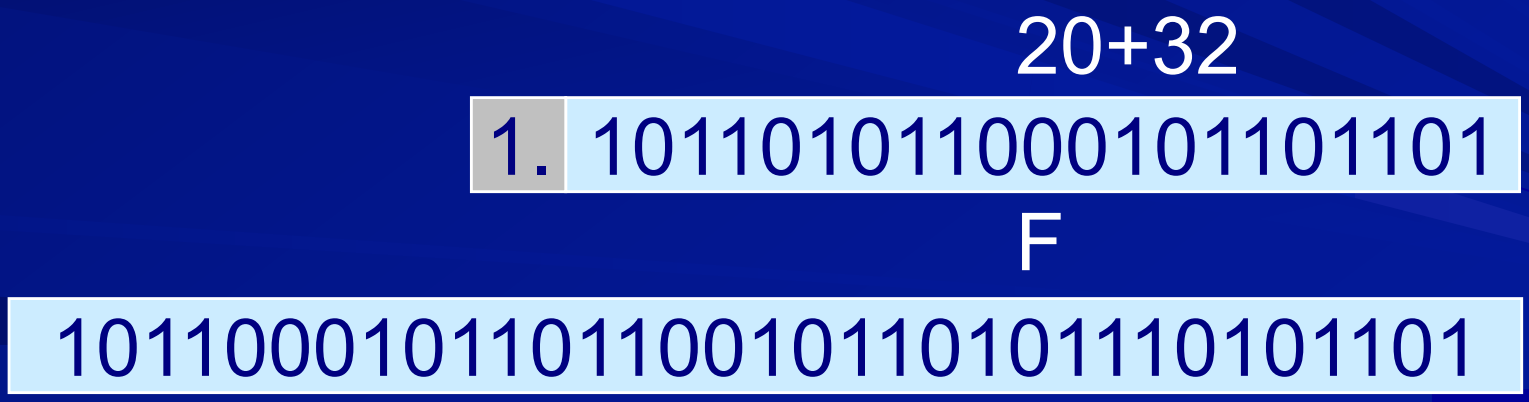

- Double precision numbers



# Representing F in IEEE 754

- Single precision numbers


- Double precision numbers



# Value range for F

- Single precision numbers

$1 \le F \le 2 - 2^{-23}$ or $1 \le F < 2$

- Double precision numbers

$1 \le F \le 2 - 2^{-52}$ or $1 \le F < 2$

These are “normalized”.

# Representing E in IEEE 754

- Single precision numbers

  8

  10110101 — 54

  E — (+ bias 127)

- Double precision numbers

  11

  10110101110 — 431

  E — (+ bias 1023)

# Value range for E

- Single precision numbers

  $-126 \leq E \leq 127$

  (all 0's and all 1's have special meanings)
- Double precision numbers

  $-1022 \leq E \leq 1023$

  (all 0's and all 1's have special meanings)

# Representing zero

- All bits of F are zero, no explicit “1” to the left

  (not a normalized value)
- E can have any value, we choose to have all bits zero
- Sign bit is also zero

- Exactly same as integer zero!

# Testing and comparing

- Test for zero, negative and positive for FP numbers is same as that for integers
- Magnitude comparison of FP numbers is same as magnitude comparison for integers

  Why?

  Because exponent is in biased notation and is located to the left of mantissa

# Overflow and underflow

largest positive/negative number (SP) =
$\pm(2 - 2^{-23}) \times 2^{127} \cong \pm 2 \times 10^{38}$
smallest positive/negative number (SP) =
$\pm 1 \times 2^{-126} \cong \pm 2 \times 10^{-38}$

largest positive/negative number (DP) =
$\pm(2 - 2^{-52}) \times 2^{1023} \cong \pm 2 \times 10^{308}$
smallest positive/negative number (DP) =
$\pm 1 \times 2^{-1022} \cong \pm 2 \times 10^{-308}$

# Floating point operations

- Add/subtract

$[(-1)^{S1} \times F1 \times 2^{E1}] \pm [(-1)^{S2} \times F2 \times 2^{E2}]$

suppose E1 > E2, then we can write it as

$[(-1)^{S1} \times F1 \times 2^{E1}] \pm [(-1)^{S2} \times F2' \times 2^{E1}]$

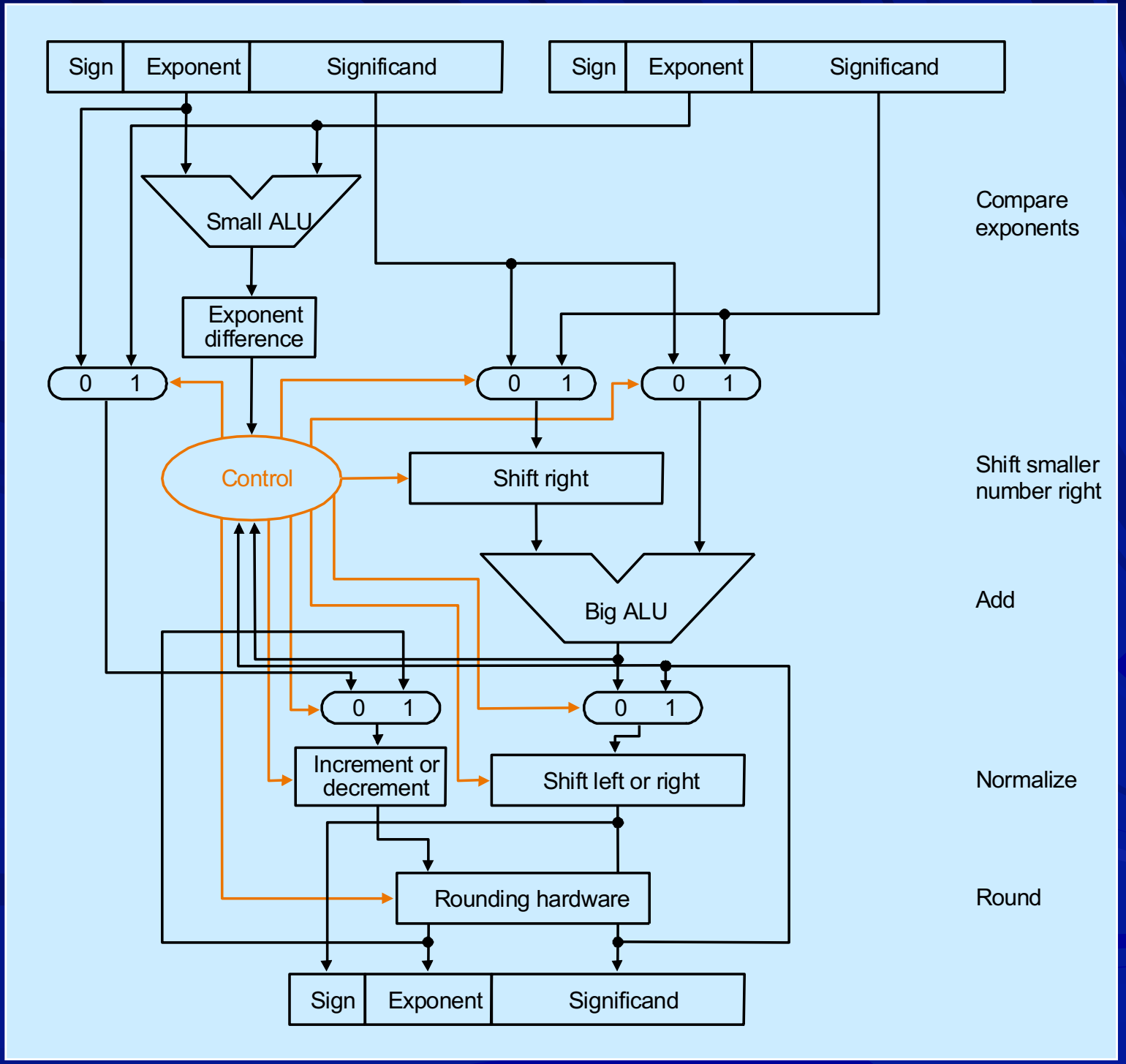
where $F2' = F2 / 2^{E1-E2}$,

The result is

$(-1)^{S1} \times (F1 \pm F2') \times 2^{E1}$

It may need to be normalized

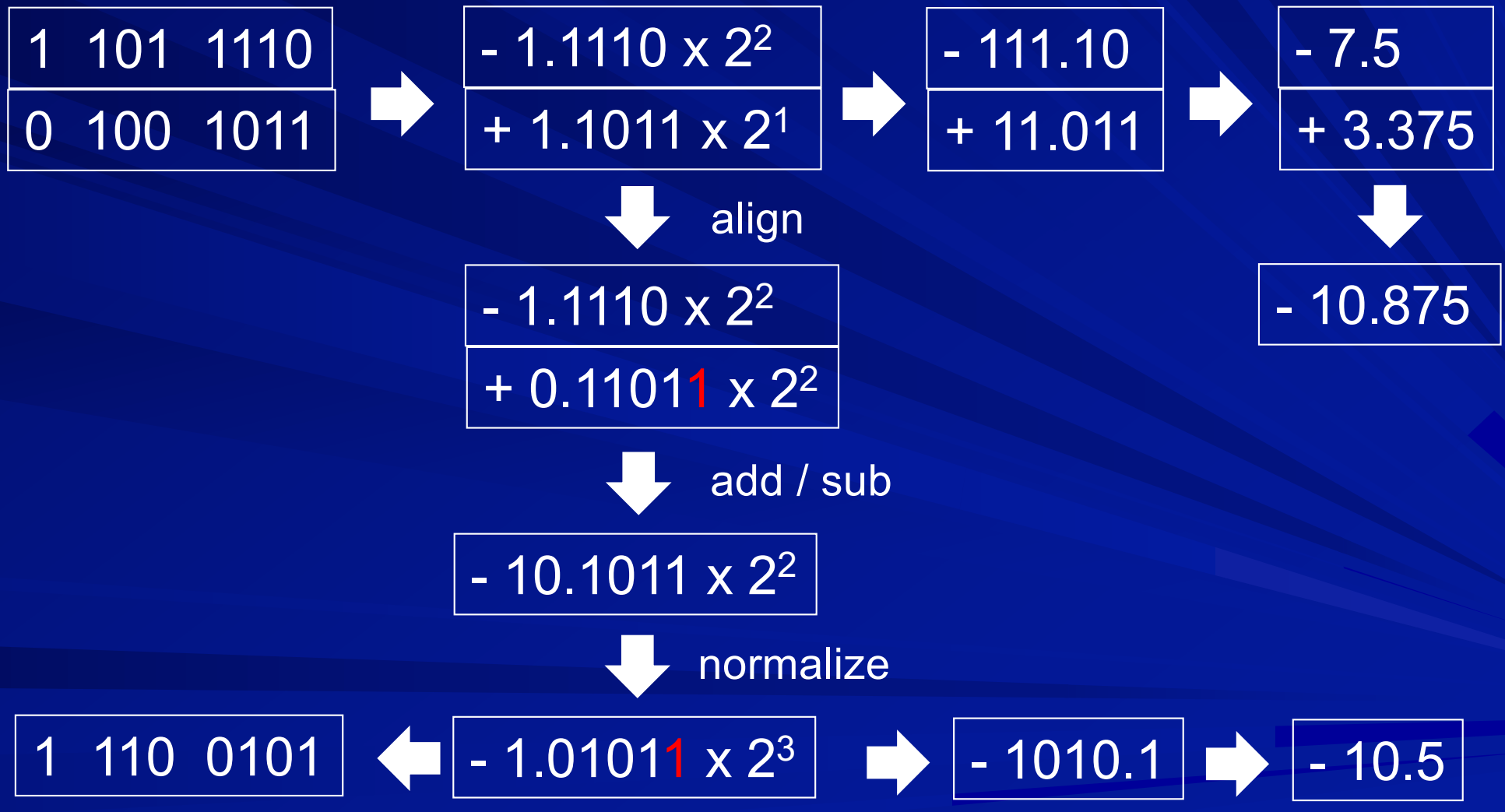
# FP Add/Sub Unit



# Small word size example

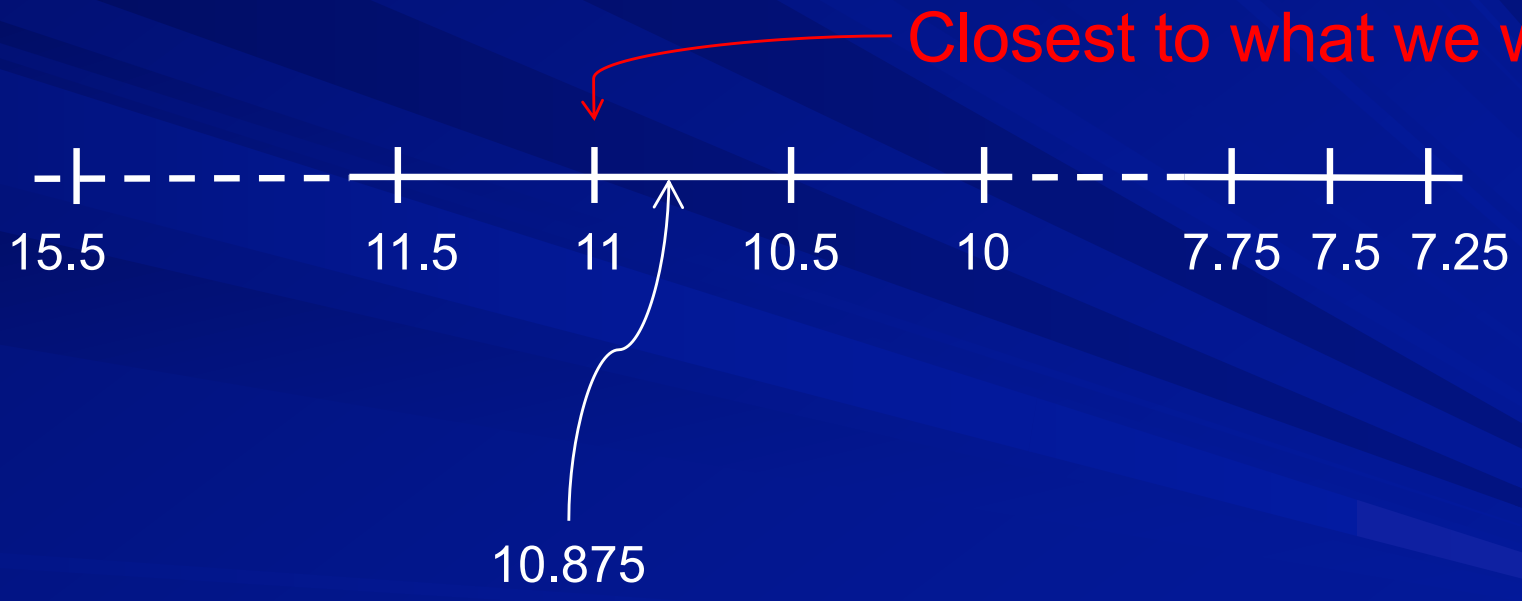

- Range of E = -2 to 3
  - represented by 001 to 110 (bias = 3)
- Range of F = 1.0 to 1.9375
  - represented by 1.0000 to 1.1111
- resolution = .0625 x $2^E$

# FP subtraction example

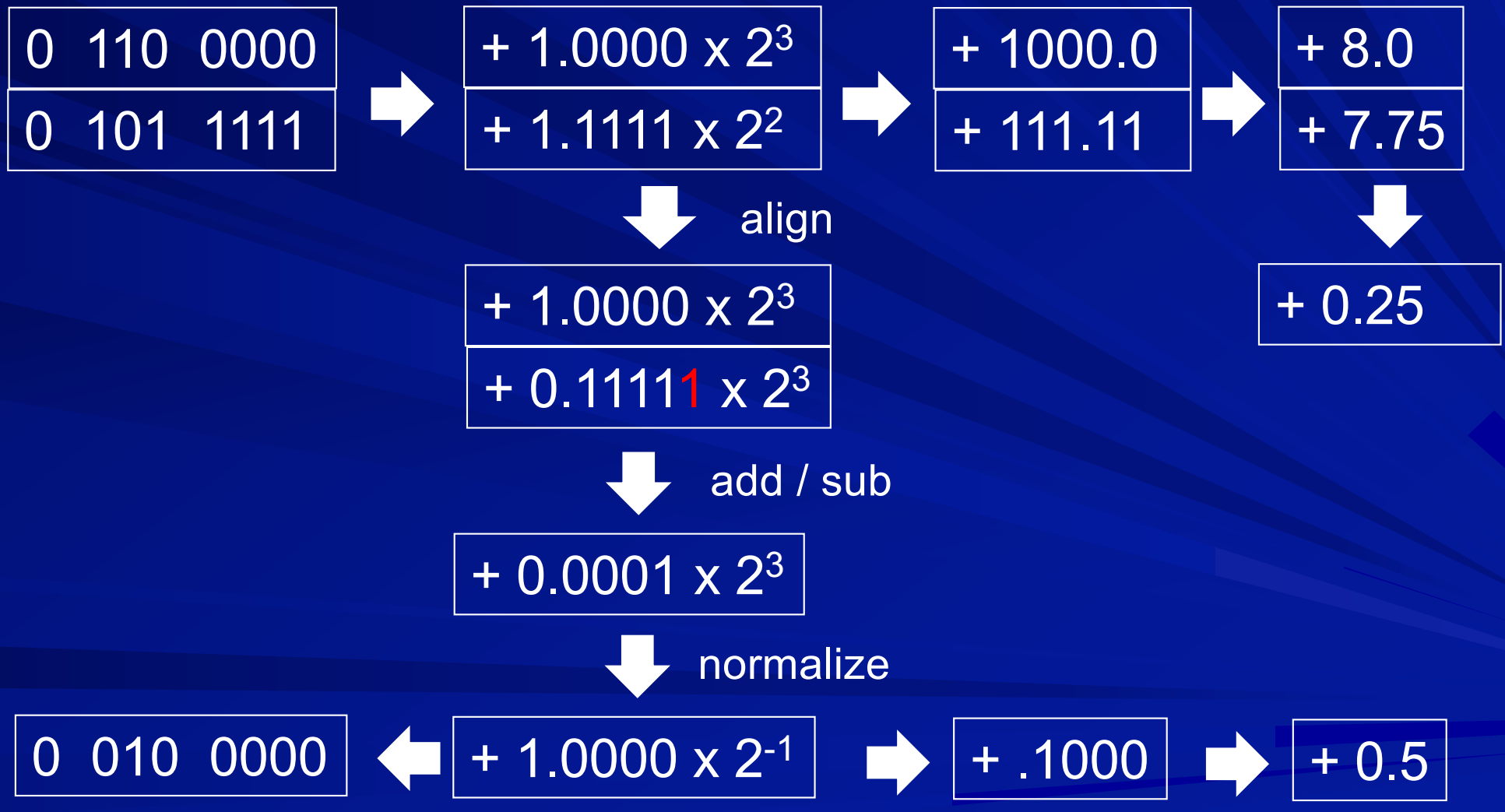


# Can we do better?

The values that can be represented



# Another FP subtraction example



# Can we represent 0.25 ?

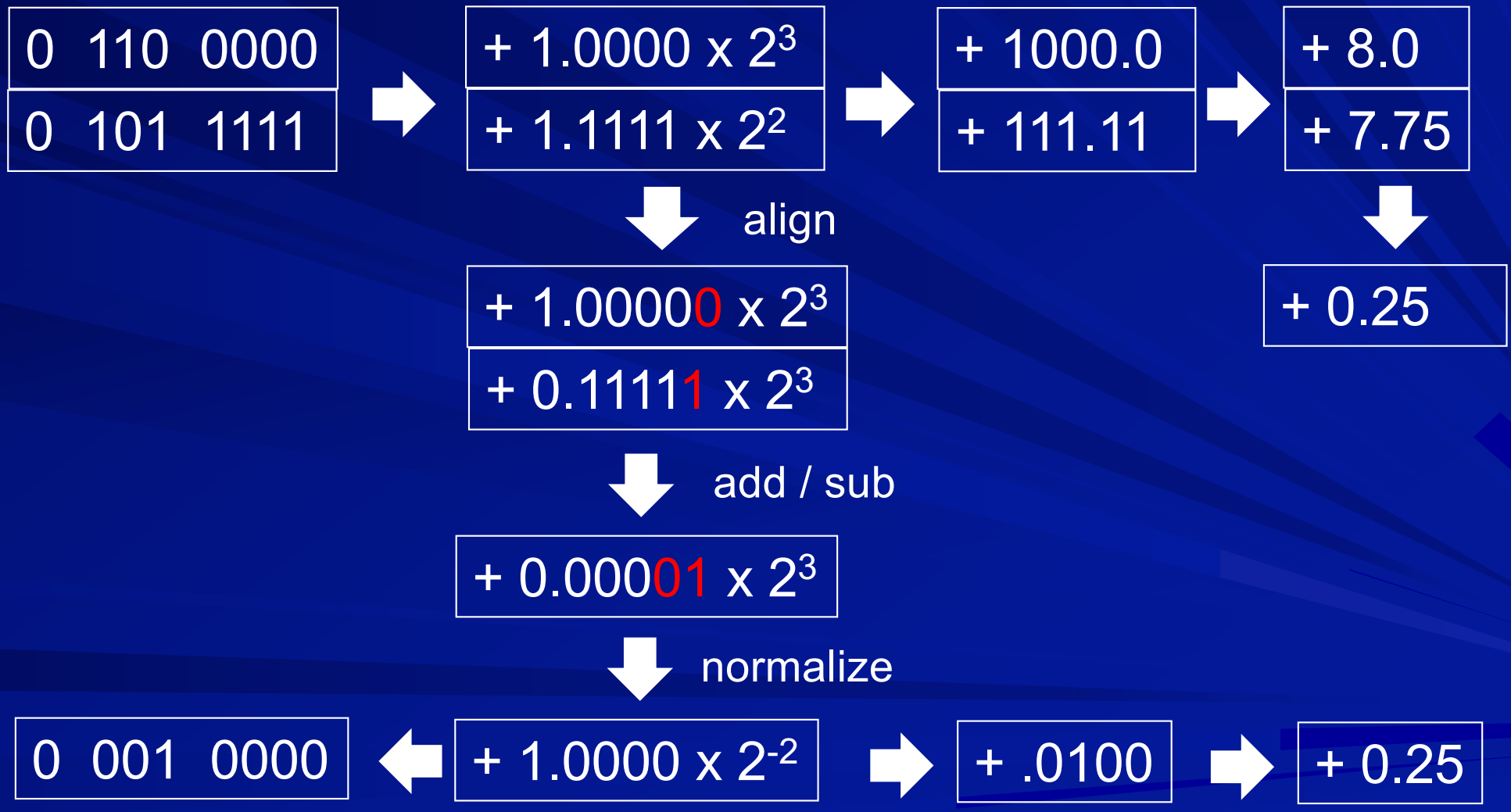
Yes.

$0.25 = 1.0 \times 2^{-2}$

0 0 0 1 0 0 0 0

So where is the problem?

# If we had more bits to work with…



# Improving precision of computation

- Precision is lost when some bits are shifted to right of the rightmost bit or are thrown

# How many bits more?

- How many bits we lose in alignment that can potentially be recovered during normalization?
- many bits lost during alignment when difference in operands is large
- many bits brought in during normalization when difference is small

# Improving precision of computation

- Three extra bits are used internally -
  G (guard), R (round) and S (sticky)
  - G and R are simply the next two bits after LSB
  - S = 1 iff any bit to right of R is non-zero

| | | |
|---|---|---|
| 1. | 1101011010110001011011 01 | GRS |

# Rounding using G, R and S

- if G=1 & R=1, add 1 to LSB
- if G=1 & R=0, look at S
  - if S=1, add 1 to LSB
  - if S=0, round to the nearest “even”
    i.e., add 1 to LSB if LSB = 1
- if G=0 & R=0 or 1, no change

# Floating point operations

- Multiply

$[(-1)^{S1} \times F1 \times 2^{E1}] \times [(-1)^{S2} \times F2 \times 2^{E2}]$

$= (-1)^{S1 \oplus S2} \times (F1 \times F2) \times 2^{E1+E2}$

Since $1 \leq (F1 \times F2) < 4$,

the result may need to be normalized

# Floating point operations

- Divide

$[(-1)^{S1} \times F1 \times 2^{E1}] \div [(-1)^{S2} \times F2 \times 2^{E2}]$

$= (-1)^{S1 \oplus S2} \times (F1 \div F2) \times 2^{E1-E2}$

Since $.5 < (F1 \div F2) < 2$,

the result may need to be normalized

(assume $F2 \neq 0$)

# Important points

- Floating point representation is an approximation of the real values
- Floating point operations are not exact real operations
- Associativity of operations need not hold
  - (A + B) + C may differ from A + (B + C)

# Special numbers

| Single precision | | Double precision | | object |
|---|---|---|---|---|
| exponent | mantissa | exponent | mantissa | |
| 0 | 0 | 0 | 0 | zero |
| 0 | nonzero | 0 | nonzero | denorm |
| 1-254 | any | 1-2046 | any | norm |
| 255 | 0 | 2047 | 0 | infinity |
| 255 | nonzero | 2047 | nonzero | NaN |

# Co-processors / processor extensions

- Extend the capability of main processors
- Modularize the design – available as options
- Co-processor registers
- Load/store instructions
- Instructions for movement of data between co-processor registers and main registers
- Co-processor operations

# Examples

- ARM
  - Floating point co-processor
  - VFP
  - Neon
- Intel
  - Floating point co-processor
  - MMX (multimedia extension)
  - SSE (streaming SIMD extension)

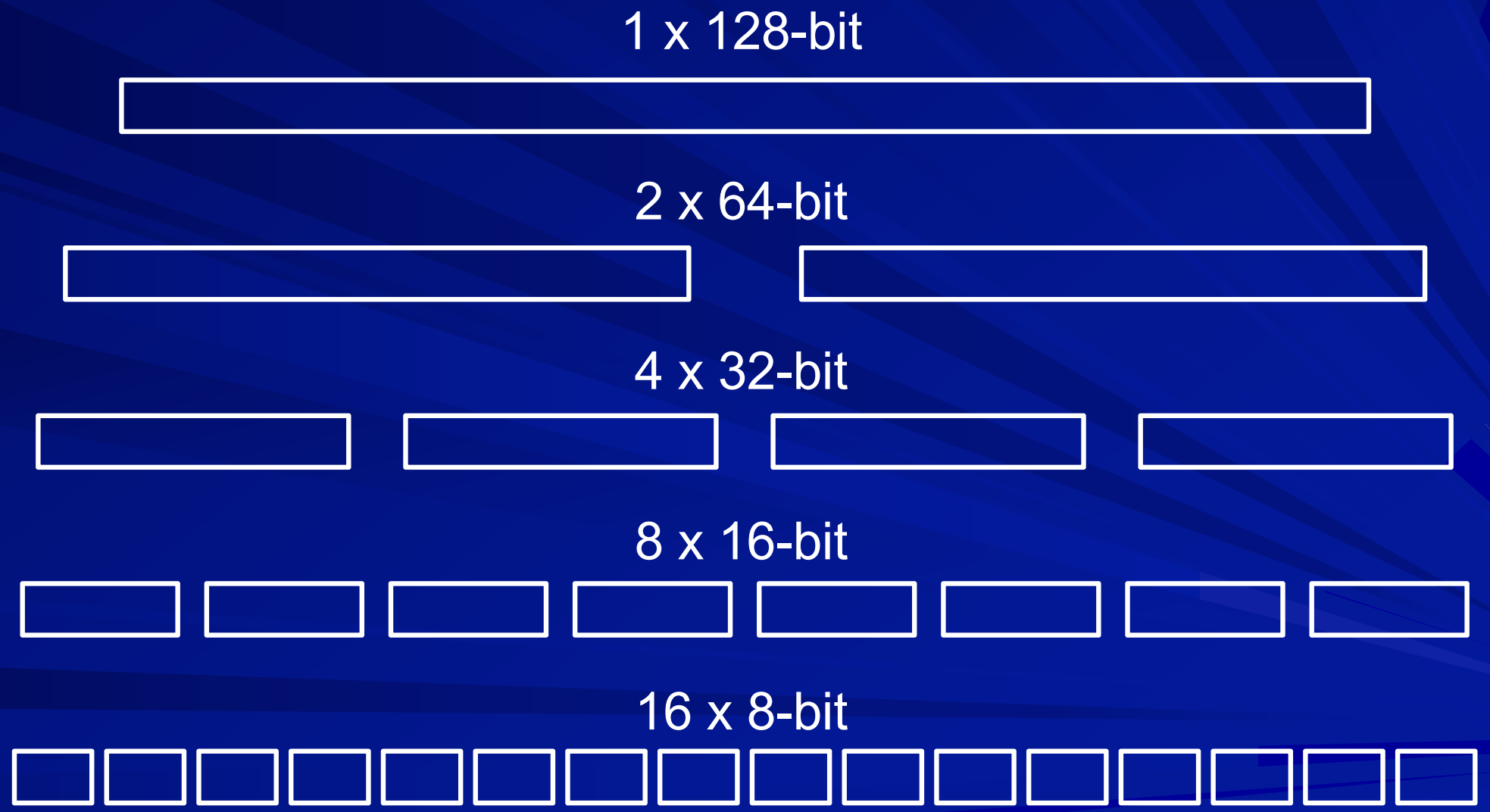
# Floating Point processors

- Floating point registers
- Load/store instructions
- Instructions for movement of data between FP and integer registers
- Arithmetic instructions (SP, DP, …)
- Comparison instructions
- Instructions for format conversion

# SIMD extensions

- SIMD : Single Instruction Multiple Data
  - same operation repeated over multiple units of data
  - example – addition of two vectors
- A large register may be interpreted as an array of smaller registers
- Hardware may perform operations on small vectors of larger registers or large vectors of smaller registers

# SIMD extensions



# COL216
# Computer Architecture

## Concluding Remarks:
## What lies beyond the present course
## 31st March 2022

# What have we studied

- Instruction set architecture and assembly language programming
- Arithmetic operations, how to build an ALU
- Constructing a processor to execute instructions – Micro architecture
- Performance issues
- Pipelining to improve performance
- Memory: caches and virtual memory
- Input / output

# What we have not studied

- Advanced pipelining techniques
- Advanced concepts in memory hierarchy design
- Superscalars
- VLIW architectures
- Vector processors
- Multiprocessors
- GPUs

# How to achieve high performance?

- Do things faster
- Do more things at the same time
- Achieve same thing by doing less
- Remove bottle necks – achieve balance
- What do $N_{instr}$, $CPI_{avg}$ and $T_{clock}$ depend on?
  - Technology
  - Circuit / logic design
  - Architecture
  - Compiler, OS
  - Algorithm

# Technology trends (increase/year)

- Transistor density : 35%
- Die size : 10-20%
- Transistor count : 40-55% (double in 18-24 mo)
  - Moore's Law
- DRAM capacity : 25-40% (double in 2-3 yr)
- Flash chip capacity : 50-60%
- Disk capacity : 40% (30 => 60 => 100 => 40)

# Forms of Parallelism

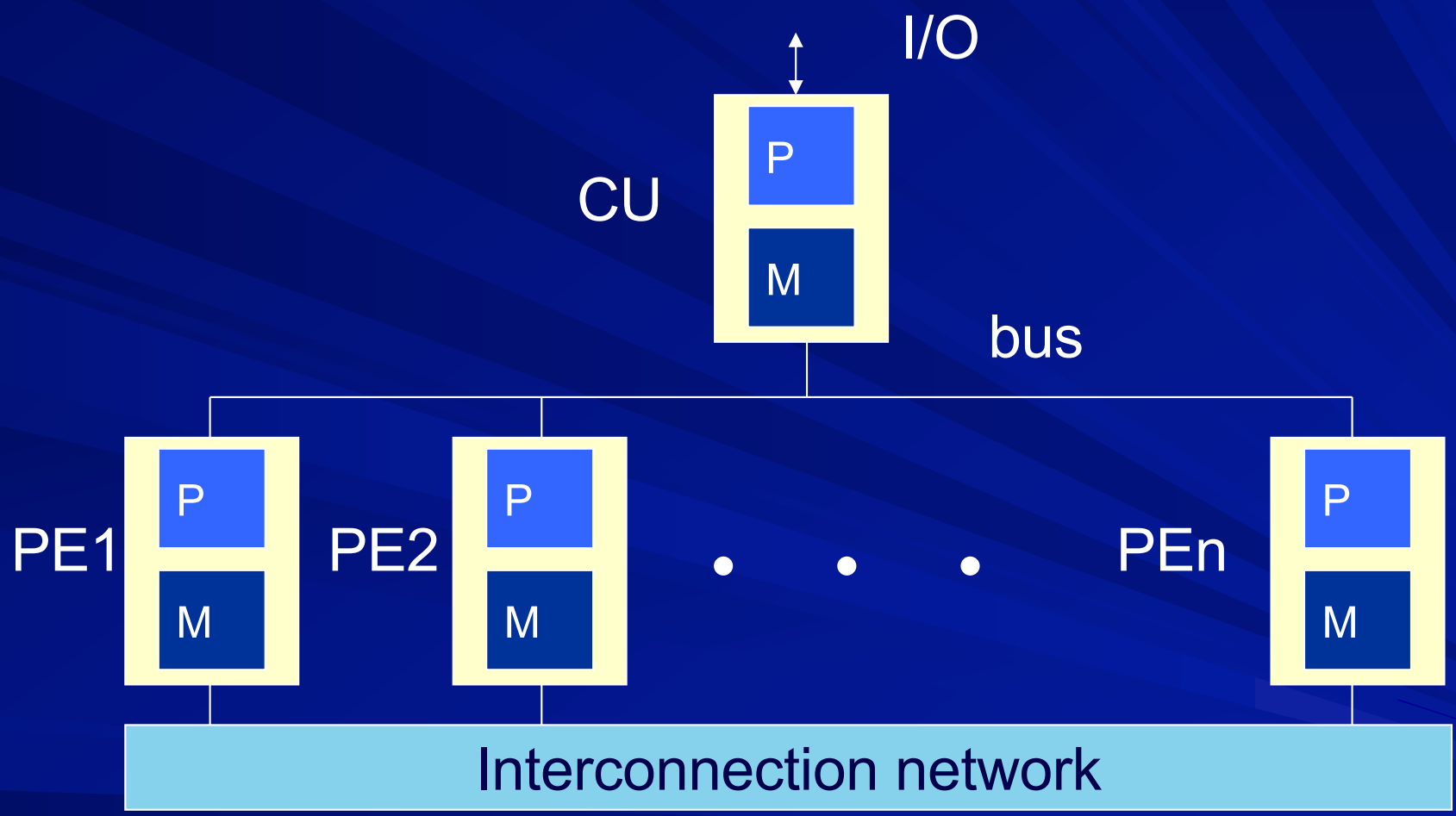
- Data Parallelism
- Functional Parallelism

- Fine Grain Parallelism
- Coarse Grain Parallelism

# Data Parallel Architectures

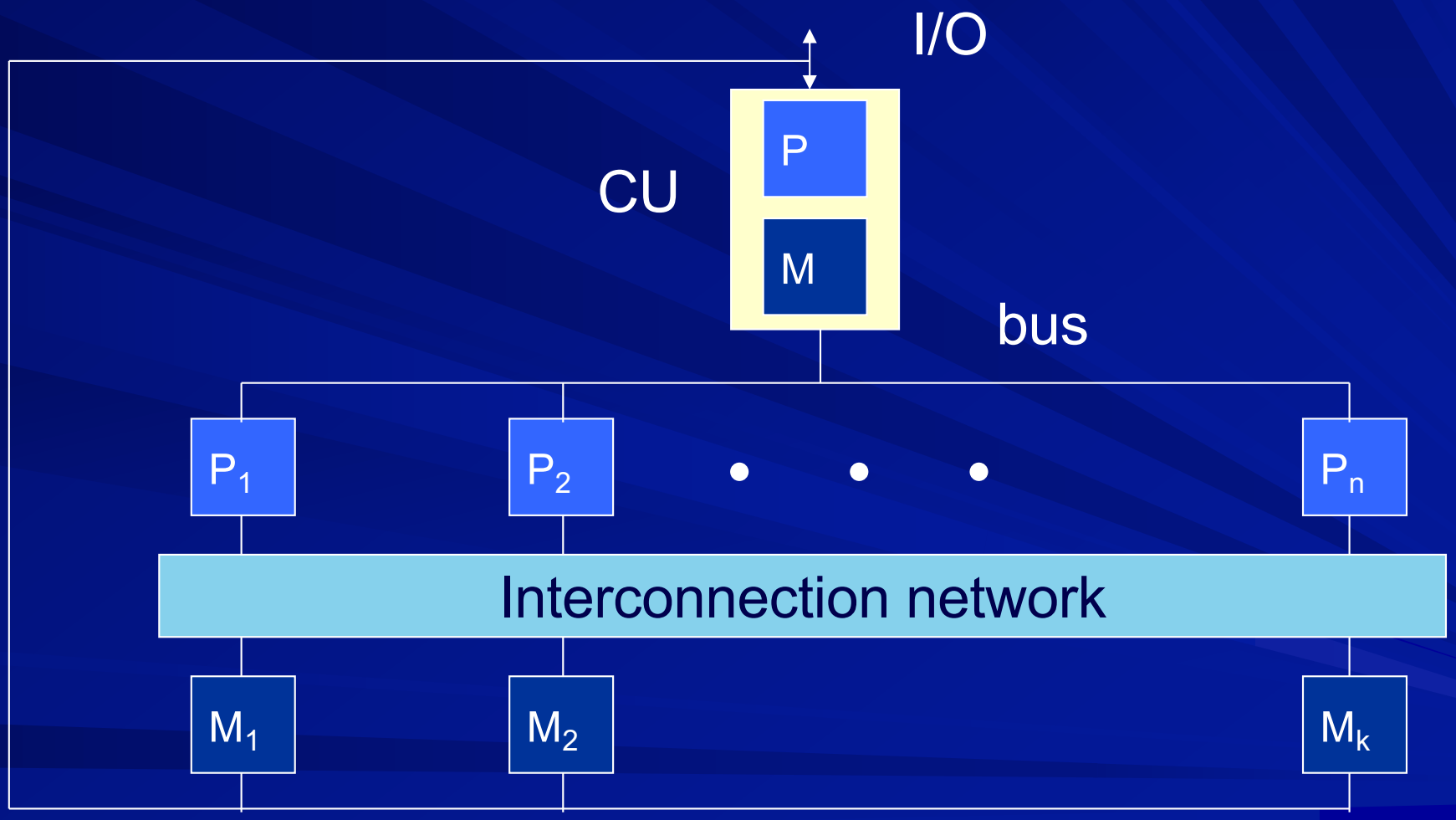
- SIMD Processors
  - Multiple processing elements driven by a single instruction stream
- Vector Processors
  - Uni-processors with vector instructions

# ILLIAC IV SIMD




Planned for 64 x 4 PEs, built only 64

# Burroughs Scientific Processor



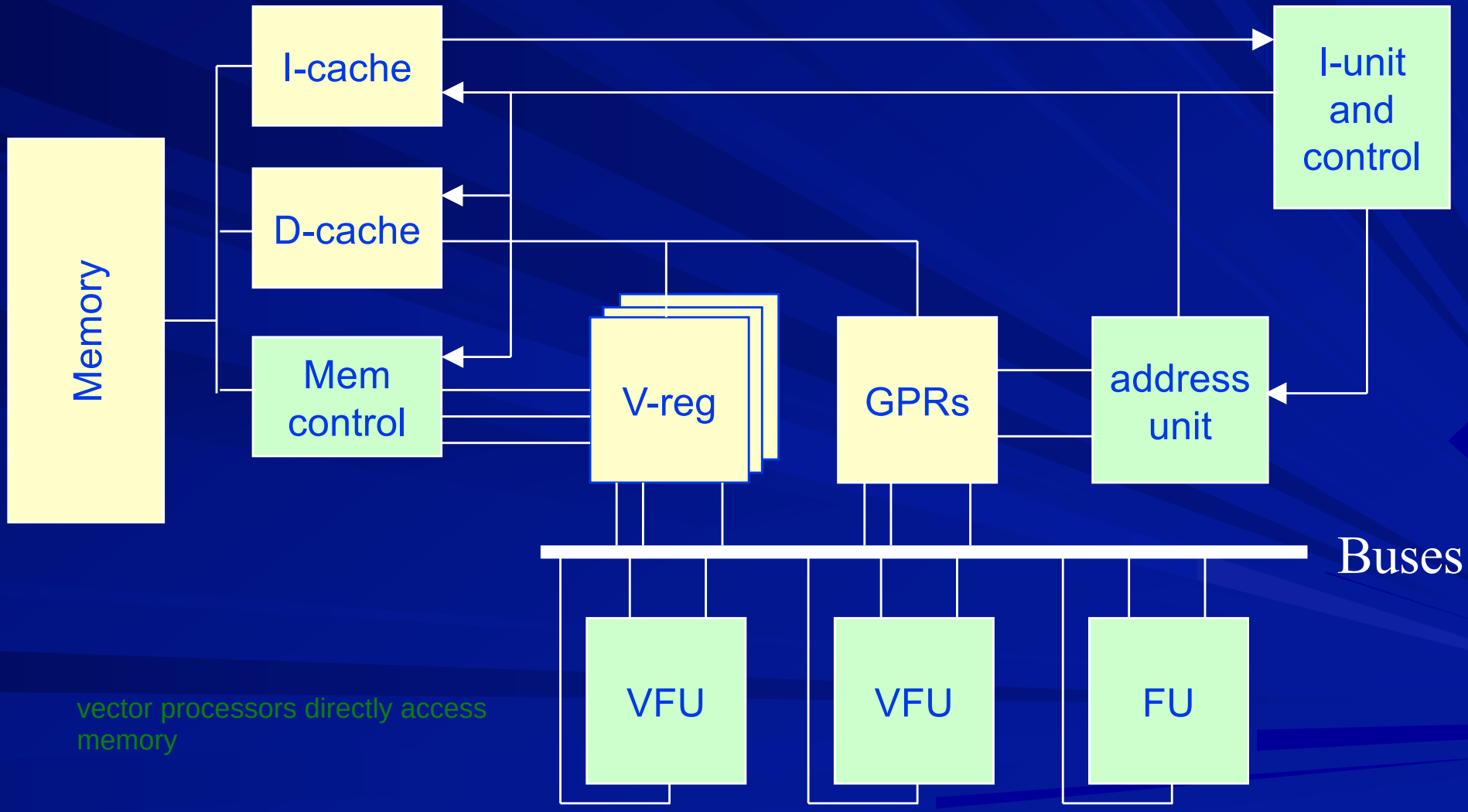
# Rise and fall of SIMDs

- Introduced in 60's (e.g. Illiac, BSP)
- Problems:
  - not cost effective
  - serial fraction and Amdahl's law
  - I/O bottle neck
- Overshadowed by Vector Processors
- Resurrected in 80's
- Did not survive because of high cost
- Basic ideas used in some modern systems

# Vector Processors

- Instructions to operate on vectors
- Vectors are streamed into and out of highly pipelined functional units
- Vector registers hold vector data
- Vectors are streamed from memory into vector registers and from vector registers into memory, no cache
- Sequences of vector operations are chained

# Vector Processor Architecture




vector processors directly access memory

multiple instructions can be executed in parallel

# Fine Grain Functional Parallelism

- Scalar pipeline : IPC ≤ 1


- Instruction Level Parallelism

# Finding Instruction Level Paralleism

- Dynamic (Superscalar)
  - Using hardware
  - Expensive (hardware cost, power consumption)
  - Analyze instruction window
  - All delays and dependencies known
- Static (VLIW)
  - Using software (compiler)
  - Inexpensive
  - Analyze whole program
  - All delays and dependencies not known

hardware checks dependence of instructions

# How superscalars work

- Multiple pipelined EUs for parallel execution
- Fetch multiple instructions in a buffer, parallel decode – instruction window
- Dynamic scheduling - out of order
- Dependency check and resource check before execution
- Speculation
- Preserving the sequential consistency and exception processing - in order retirement

exceptions have to be thrown in order, so we don't need to update the state

hold back the updation of the state, the final step is done in order at the end

# Limits on ILP

- Limited ILP in the application (available ILP)
- Limitation of HW and SW in exploiting given ILP (achieved ILP)
  - limited EUs
  - limited instruction window
  - limited issue capability
  - renaming limitations
  - imperfect branch/jump prediction
  - imperfect load/store speculation
  - imperfect cache

# Beyond ILP

- Multithreading over ILP Datapath
- Multithreading over multiple cores

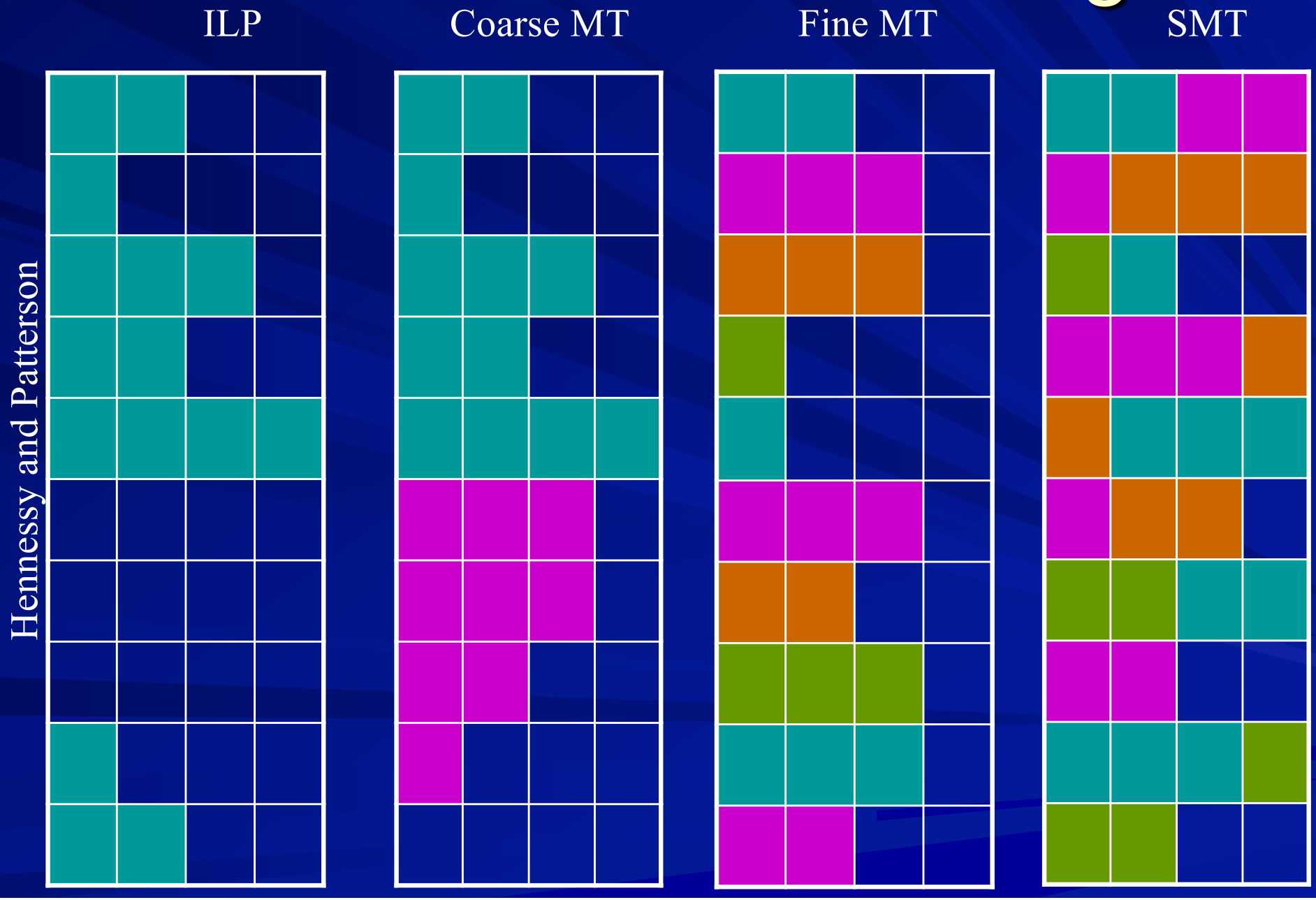
# Multithreading over ILP datapath

- The state needs to be replicated for each thread
  each thread must have its own state
  - program counter
  - registers
- Memory is shared
- Require ability to switch from one thread to other quickly

row represents the time axis, thus one row implies one cycle and columns denote different execution units - in case of SMT. we allow different threads to use
different execution units, thus overall improving performance

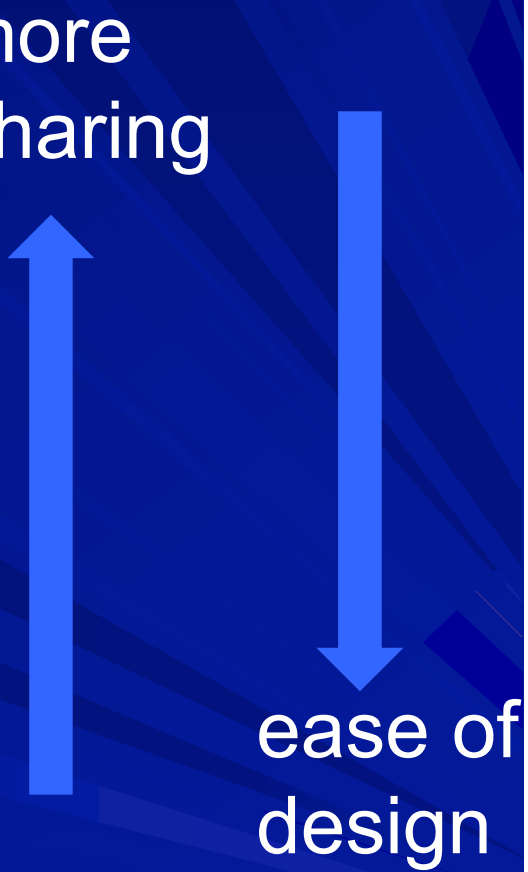
# ILP and Multithreading

simultaneous MT

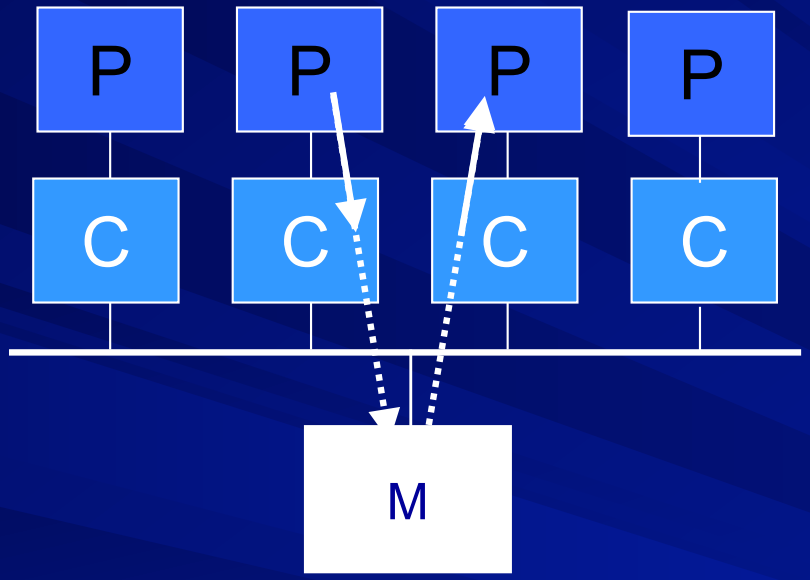


# ILP, SMT and Multicores

- ILP
  - Multiple execution units
  - RF, caches and memory shared
- SMT
  - Multiple execution units and RFs
  - Caches and memory shared
- Multicore
  - Multiple EUs, RFs and caches (upper levels)
  - Caches (lower levels) and memory shared



# Memory hierarchy in Multicores




- Multiple copies of data in caches may exist
- ⇒ Problem of cache coherence

Solution:

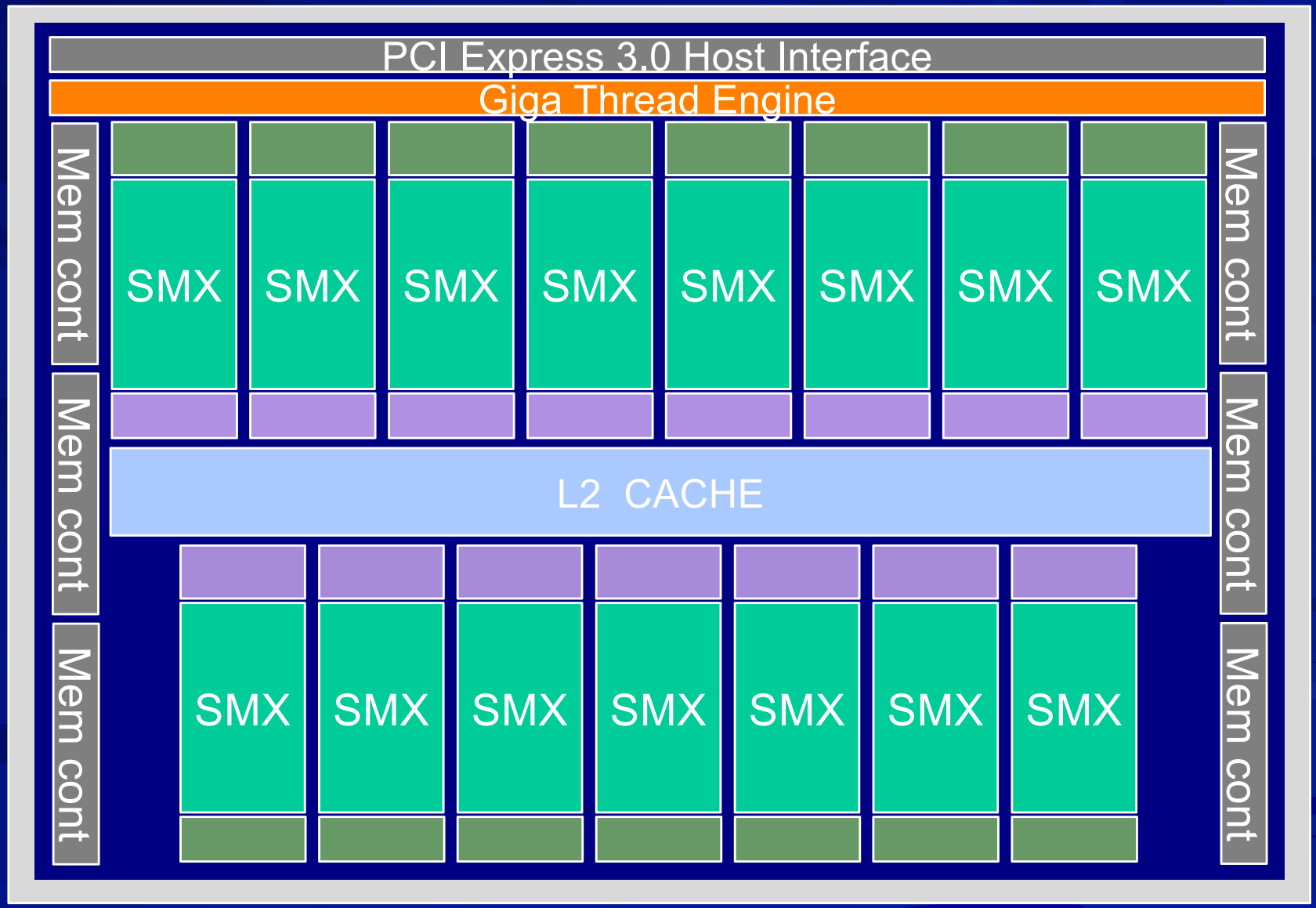
- Cache controllers follow “coherence protocol”
- Interaction and communication in a mutually agreed manner

# Graphics processors

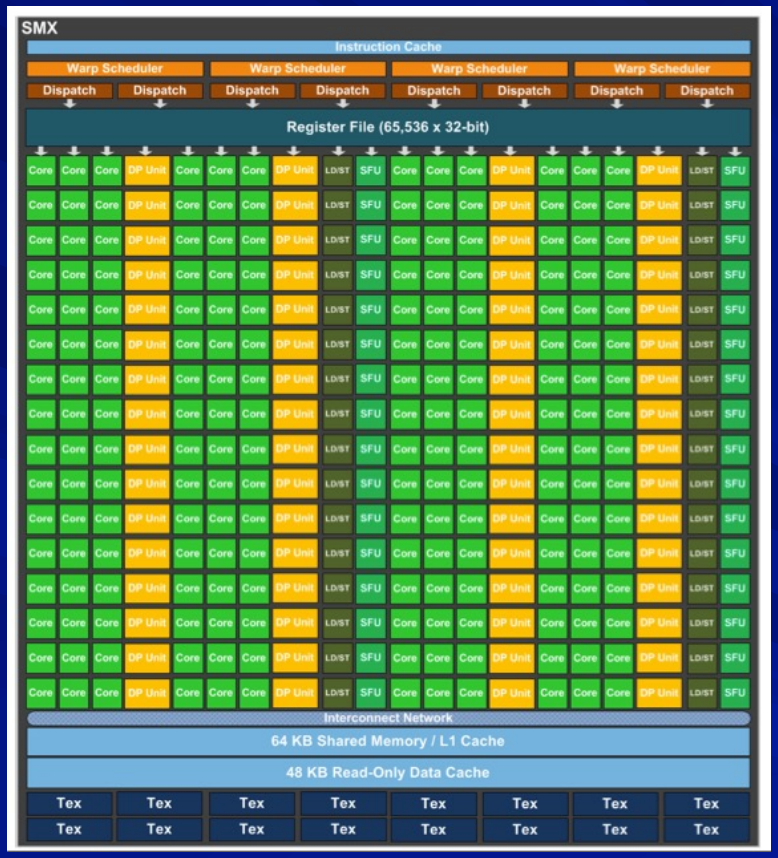
- Developed originally for accelerating graphics
- Now used for General Purpose computing as well
- Major manufacturers : Nvidia, AMD, Intel
- Support for programming : CUDA (for Nvidia devices), OpenCL (general)

SIMD kind fine grain parallelism

# Nvidia's Kepler GK110



# SMX: Streaming Multiprocessor




16 Rows

192 CUDA cores, 64 DPUs

32 LSUs,32 SFUs

# Thanks