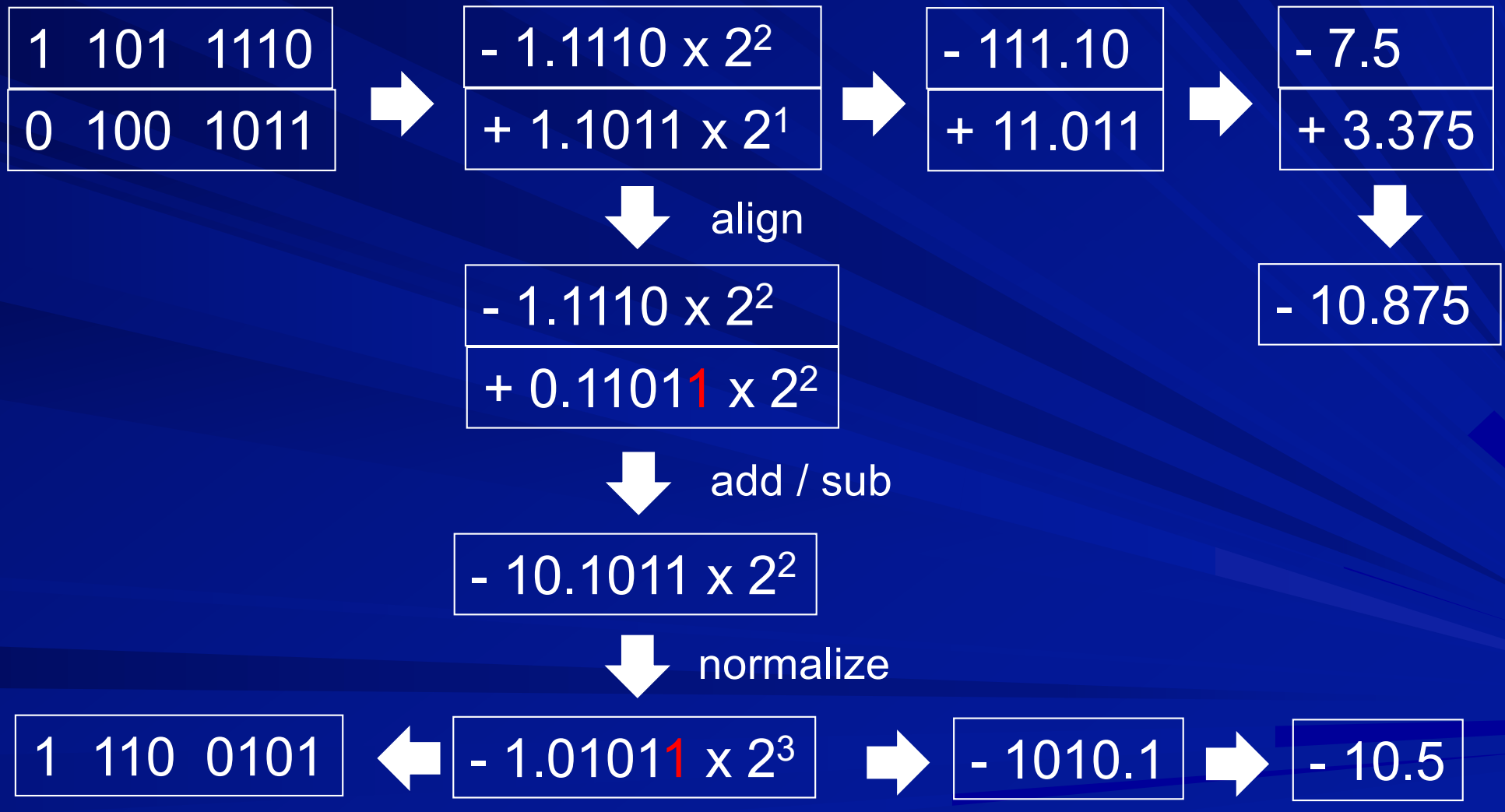
Small word size example

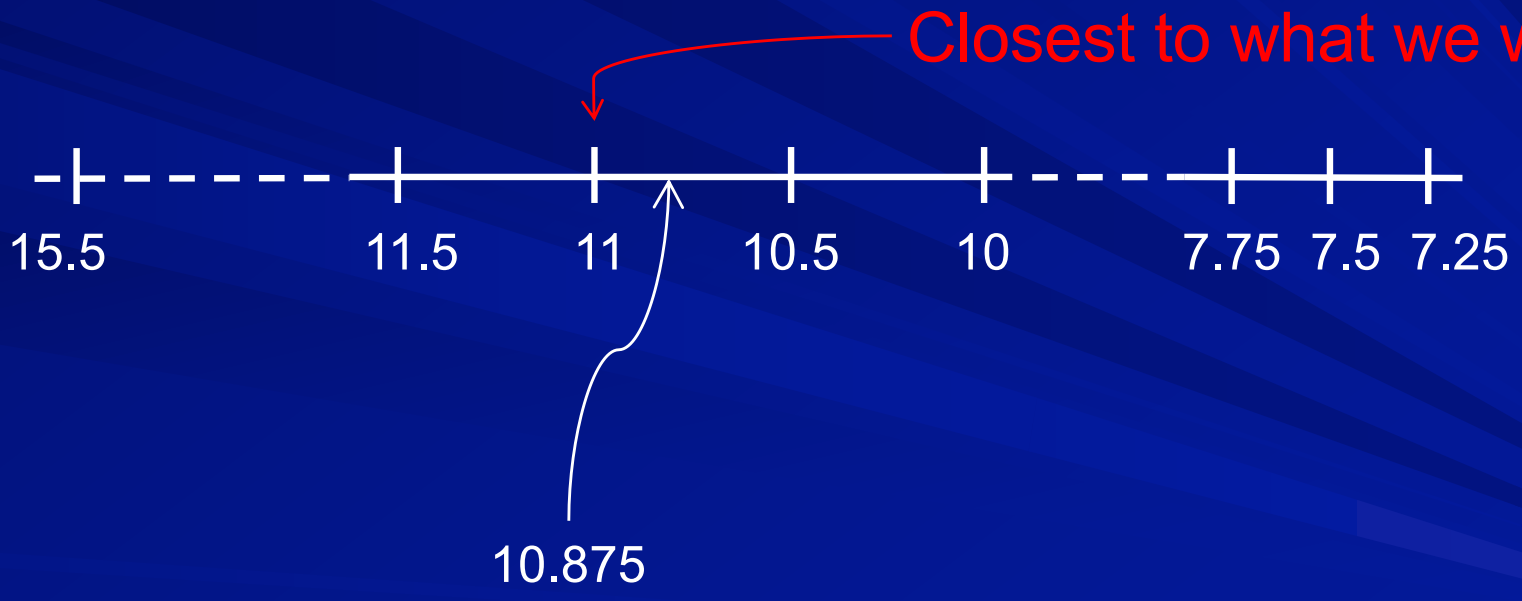
- Range of E = -2 to 3
  - represented by 001 to 110 (bias = 3)
- Range of F = 1.0 to 1.9375
  - represented by 1.0000 to 1.1111
- resolution = .0625 x 2^E

FP subtraction example

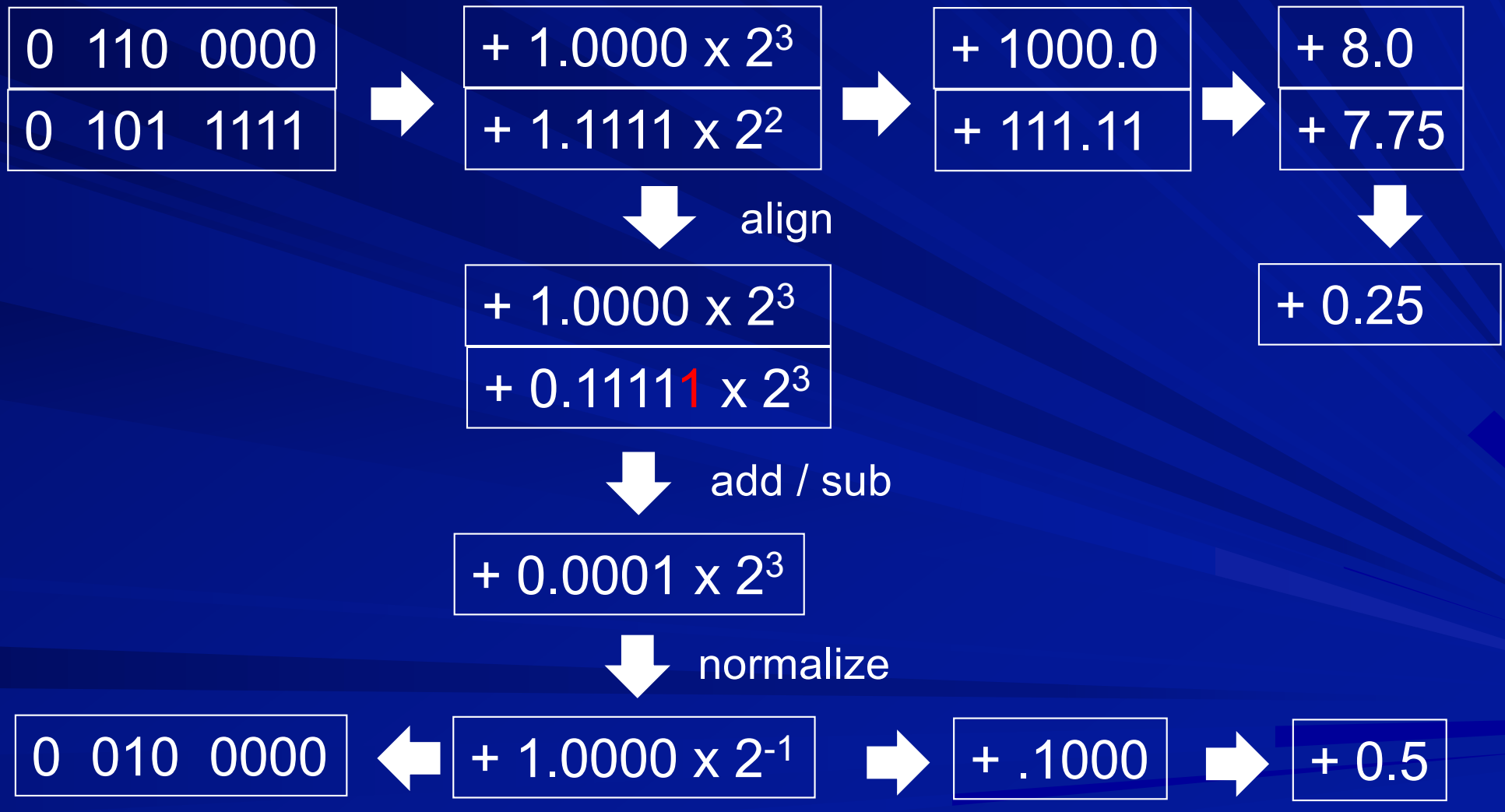


Can we do better?

The values that can be represented



Another FP subtraction example



Can we represent 0.25 ?

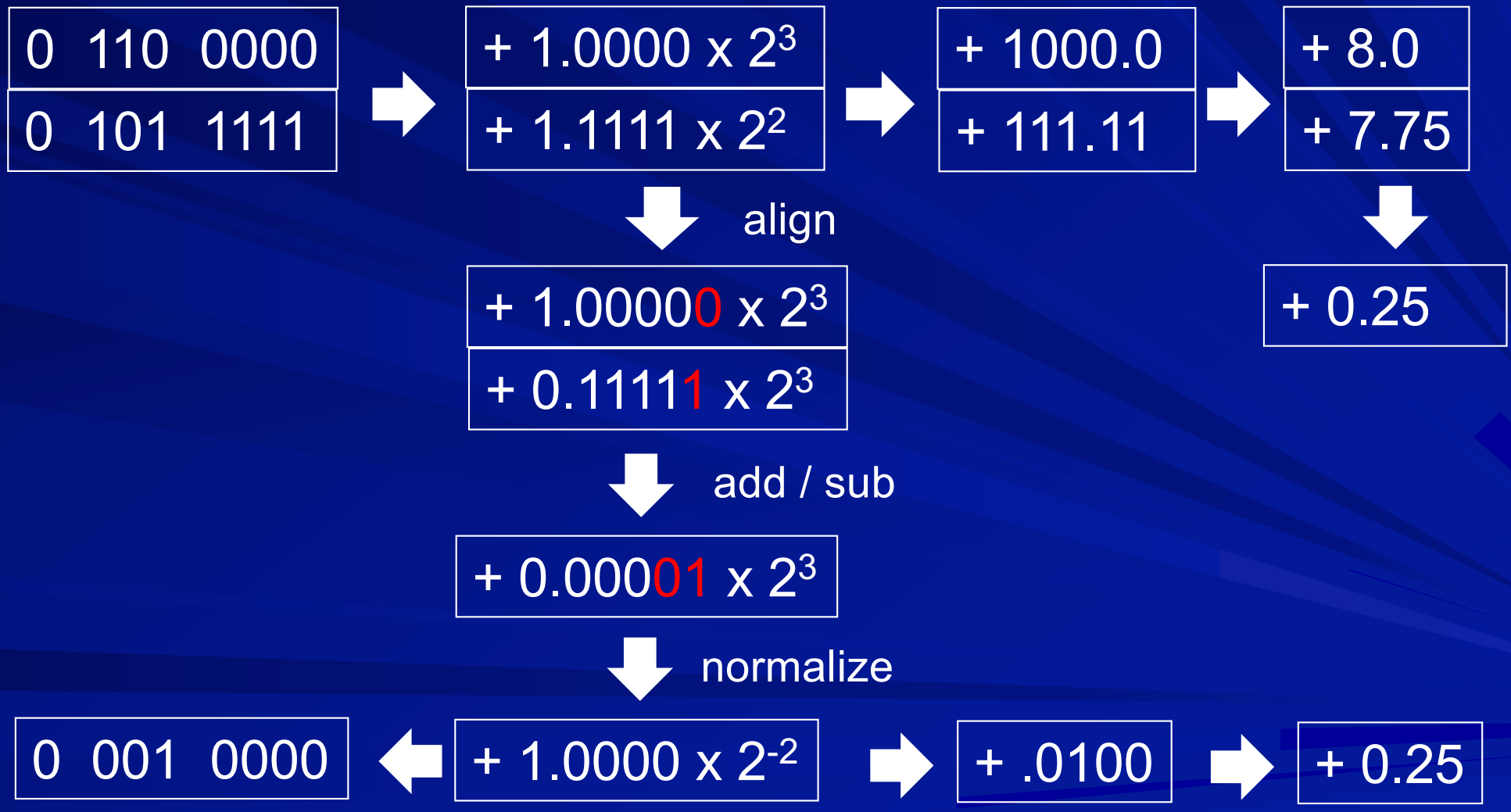
Yes.

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

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