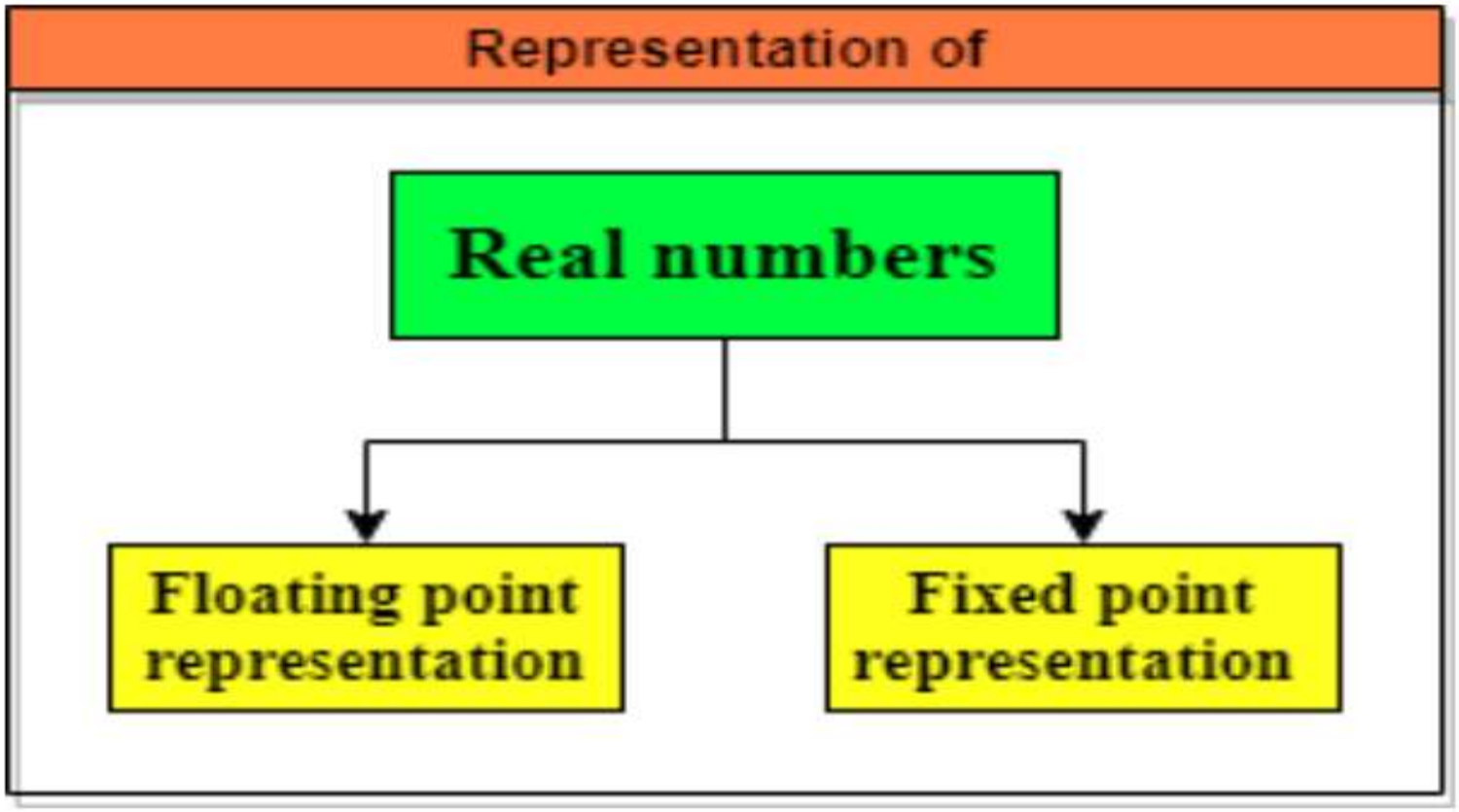
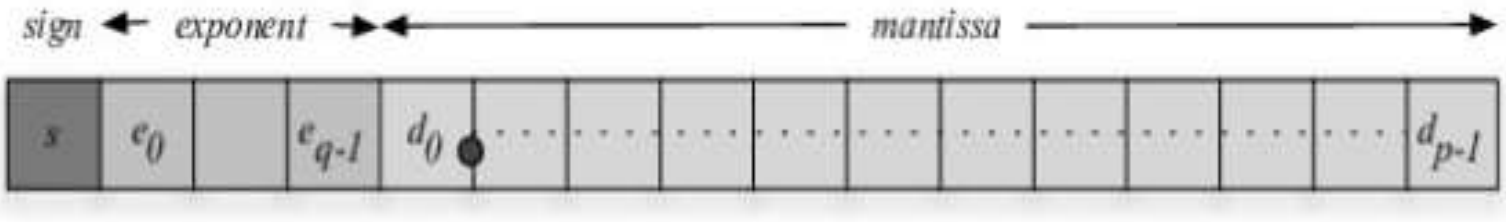
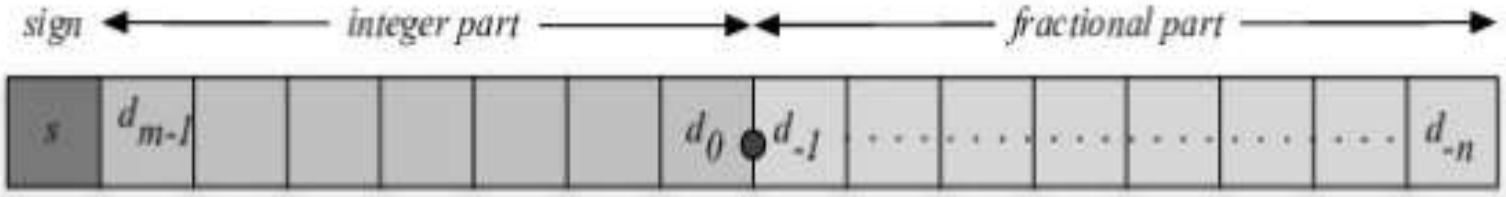
PROCESSORS


Representation of
Real numbers
Floating point representation
Fixed point representation



Floating-Point Format



Fixed-Point Format

Introduction

- DSP processors can be divided into two broad categories:
  - » General Purpose
  - » Special Purpose
- Further DSP processors include
  - » Fixed point devices
  - » Floating point devices

Examples of Fixed and Floating

- Low End **Fixed Point**
  - TMS320C2XX, ADSP21XX, Motorola (DSP56XXX)
- High End **Fixed Point**
  - TMS320C55XX, DSP16XXX,
  - ADSP215XX, DSP56800
- Floating Point
  - TMS320C3X, C67XX, ADSP210XX(SHARC processor), DSP96000, DSP32XX

Fixed Point Vs Floating Point

- fixed point processor are :
  - cheaper
  - smaller
  - less power consuming
  - Harder to program
    - Watch for errors: truncation, overflow, rounding
  - Limited dynamic range
  - Used in 95% of consumer products
- floating point processors
  - have larger accuracy
  - are much easier to program
  - can access larger memory

Special purpose hardware

- Hardware designed for efficient execution of specific DSP algorithms such as digital filter, FFT. [Algorithm-specific digital signal processor]

- Hardware designed for specific application, for example telecommunications, digital audio, or control applications.[Application -specific digital signal processor]

Parallelism

- To achieve increased computational performance
- Three techniques used to achieve parallelism are
    - SIMD
    - VLIW
    - Superscalar processing

Very Long Instruction word (VLIW) architecture

- It consists of many functional units connected to a large central register file
- Each functional unit have two read ports and one write port
- Register file would have enough memory bandwidth to balance the operand usage rate of functional units

VLIW characteristics

- VLIW contains multiple primitive instructions that can be executed in parallel
- The compiler **packs** a number of primitive, independent **instructions into a very long instruction word**
- The **compiler must guarantee that multiple primitive instructions** which group together **are independent** so they can be executed in parallel.

VLIW Architecture

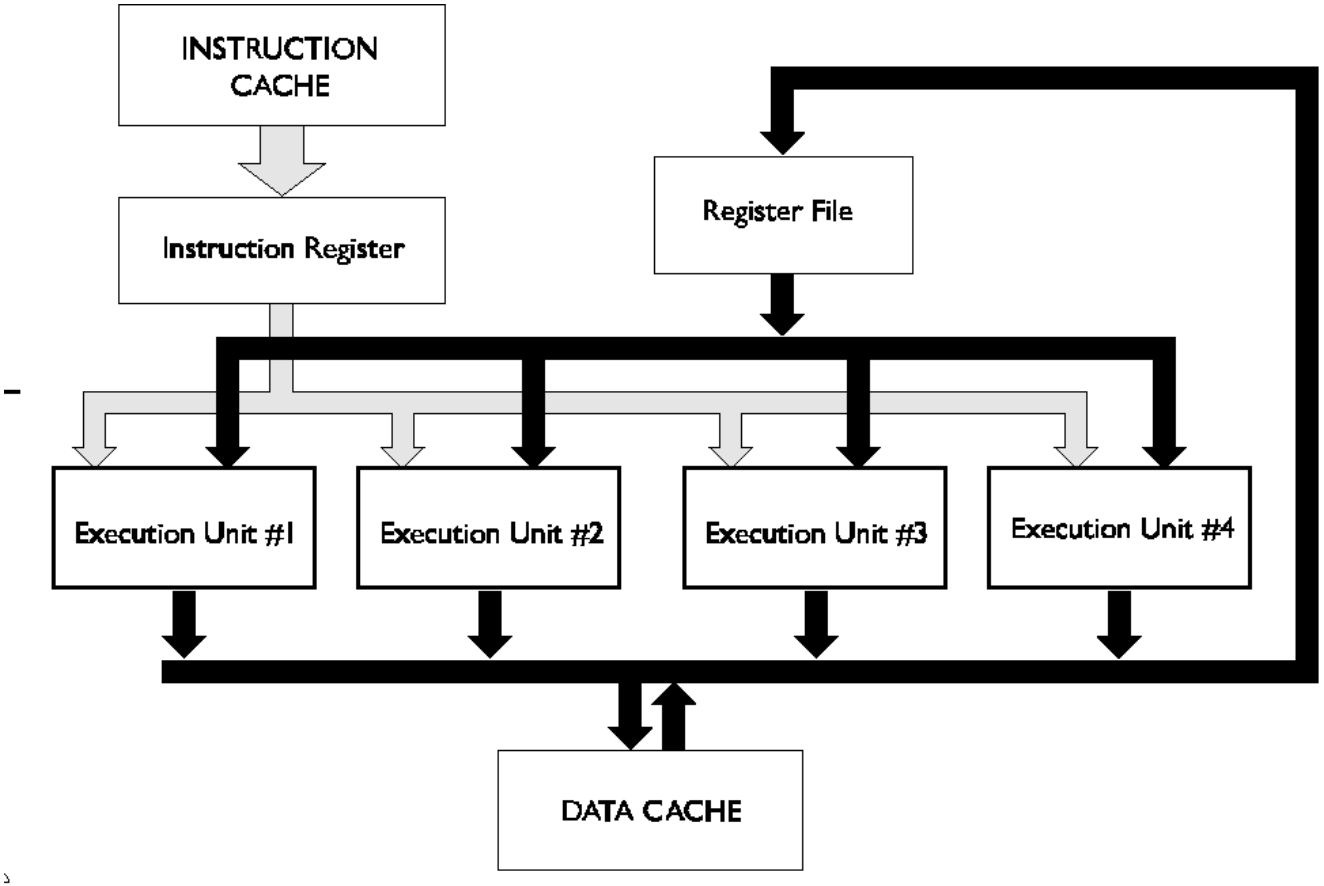
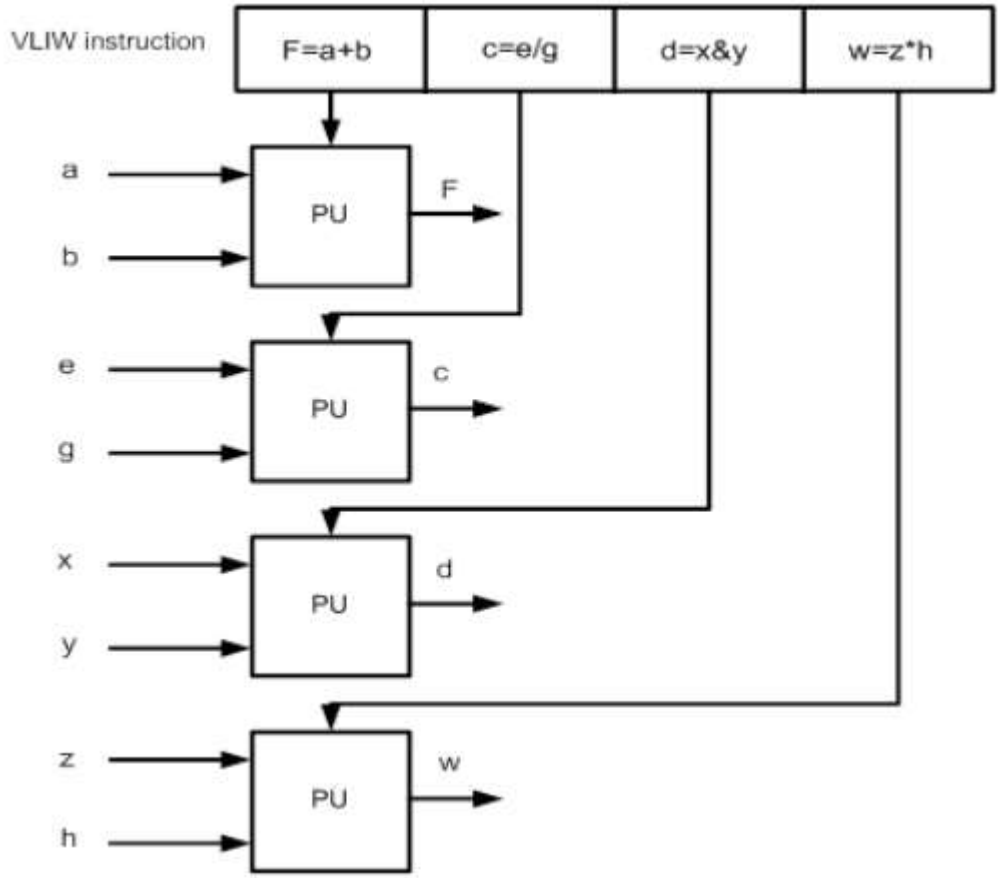


FIGURE 4

Example of a single VLIW instruction:

F=a+b; c=e/g; d=x&y; w=z*h;



VLIW principles

1. The compiler analyzes dependence of all instructions among sequential code and extracts as much parallelism as possible.
2. Based on analysis, the compiler re-codes the sequential code in VLIW instruction words.(One VLIW instruction word contains maximum 8 primitive instructions)
3. Finally VLIW hardware
   - Fetch the VLIWs from cache,
   - Decode them,
   - Dispatch the independent primitive instructions to corresponding functional units and
   - Execute

Finite word length

https://www.youtube.com/watch?v=4KYh36gydAM

- **Advantages of VLIW architecture**
- Increased performance.
- Potentially scalable i.e. more execution units can be added and so more instructions can be packed into the VLIW instruction.
- **Disadvantages of VLIW architecture**
- New programmer needed.
- Program must keep track of Instruction scheduling.
- Increased memory use.
- High power consumption.