UNIT-5

INTRODUCTION DIGITAL SIGNAL PROCESSORS AND ITS APPLICATION

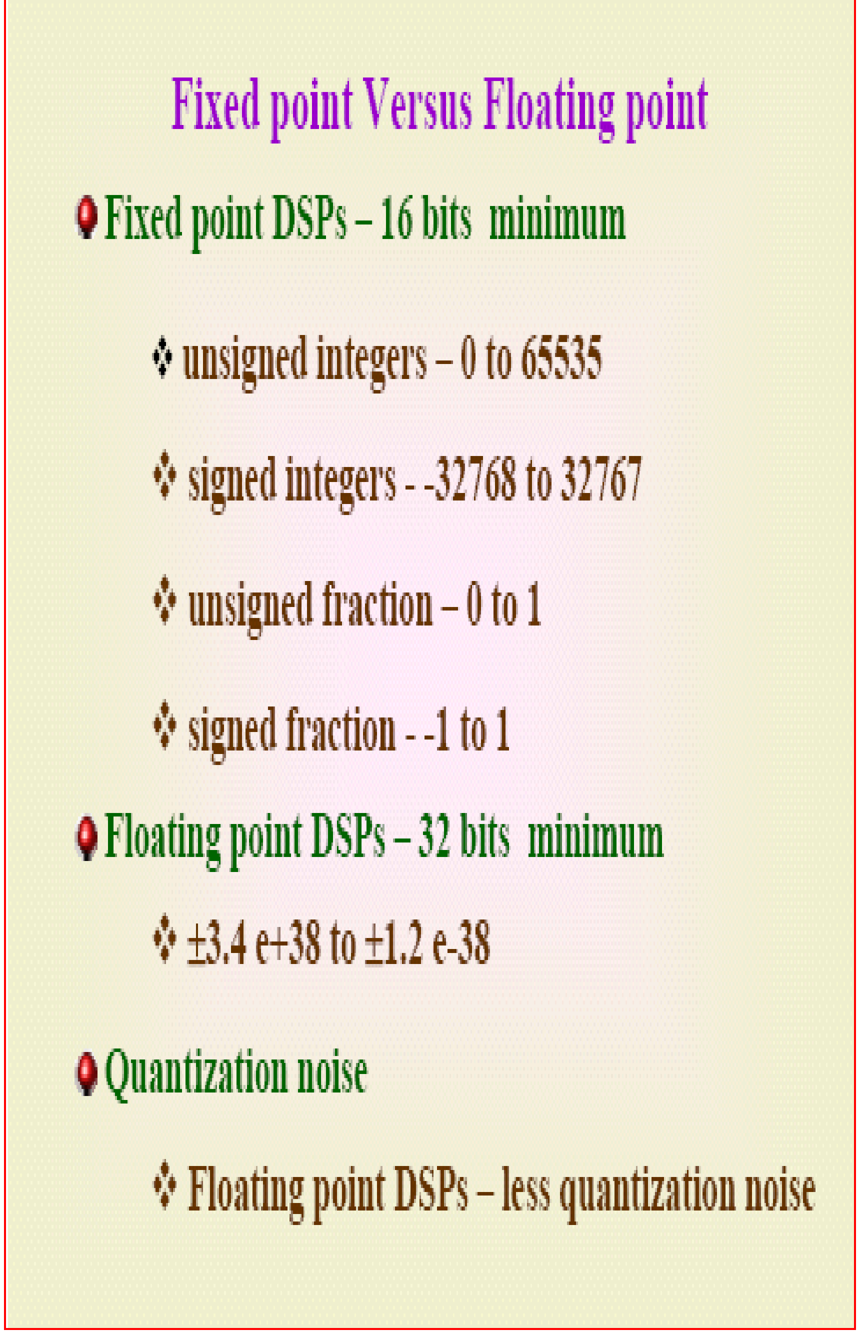
Dr. Ramesh R

Introduction

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

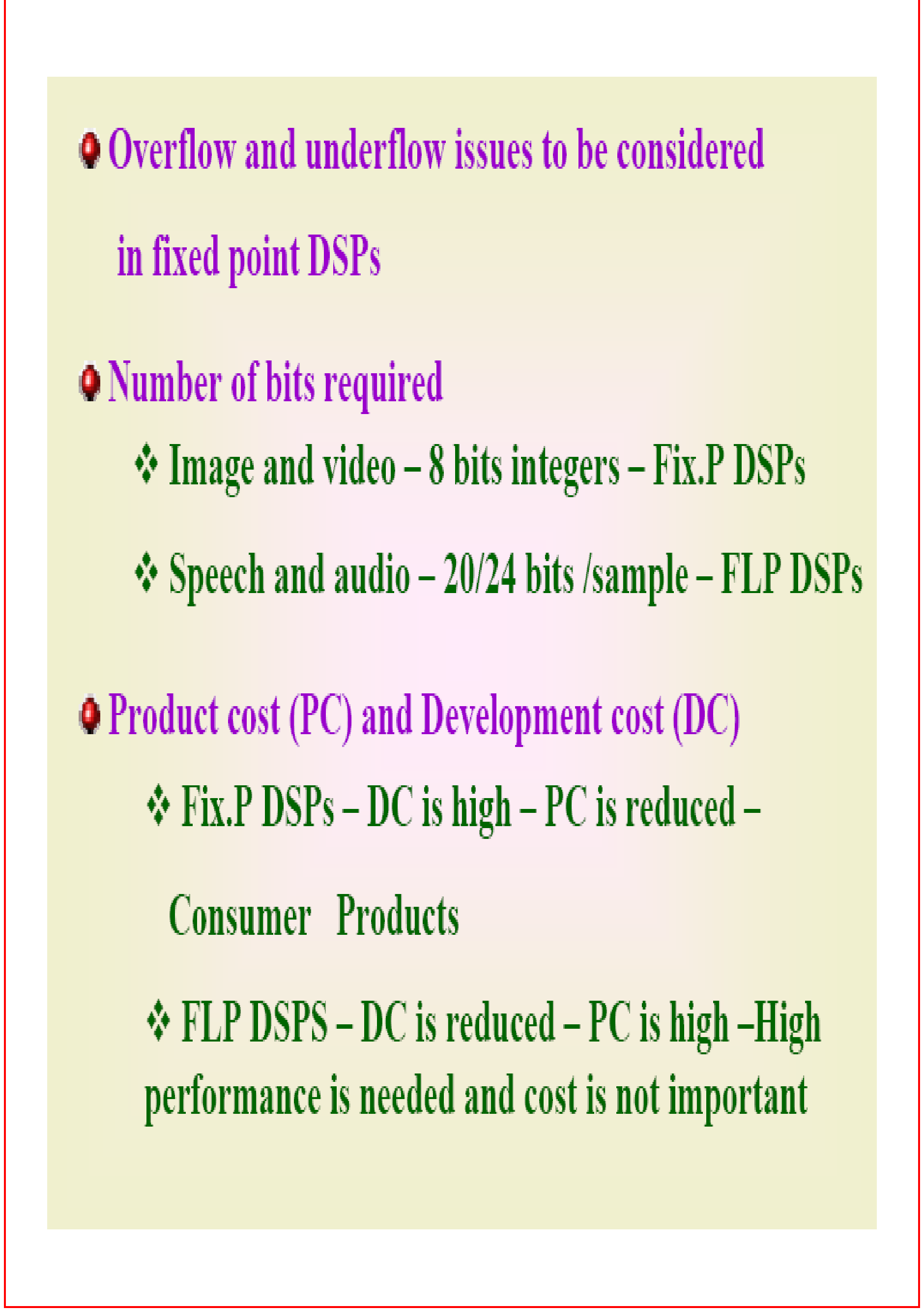
Examples of Fixed and Floating

- 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 Versus Floating point

- Fixed point DSPs – 16 bits minimum
  - unsigned integers – 0 to 65535
  - signed integers - -32768 to 32767
  - unsigned fraction – 0 to 1
  - signed fraction - -1 to 1
- Floating point DSPs – 32 bits minimum
  - ±3.4 e+38 to ±1.2 e-38
- Quantization noise
  - Floating point DSPs – less quantization noise



- Overflow and underflow issues to be considered in fixed point DSPs
- Number of bits required
  - Image and video – 8 bits integers – Fix.P DSPs
  - Speech and audio – 20/24 bits /sample – FLP DSPs
- Product cost (PC) and Development cost (DC)
  - Fix.P DSPs – DC is high – PC is reduced – Consumer Products
  - FLP DSPS – DC is reduced – PC is high –High performance is needed and cost is not important

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

Fixed Point Vs Floating Point

Floating Point

Applications

- Modems
- Digital Subscriber Line (DSL)
- Wireless Basestations
- Central Office Switches
- Private Branch Exchange (PBX)
- Digital Imaging
- 3D Graphics
- Speech Recognition
- Voice over IP

Fixed Point

Applications

- Portable Products
- 2G, 2.5G and 3G Cell Phones
- Digital Audio Players
- Digital Still Cameras
- Electronic Books
- Voice Recognition
- GPS Receivers
- Headsets
- Biometrics
- Fingerprint Recognition

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]

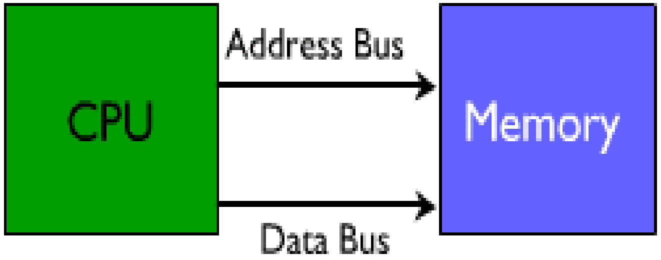
Computer Architecture for Signal Processing

Techniques

- Harvard Architecture
- Pipelining
- Fast, dedicated hardware multiplier/Accumulator
- Special instruction dedicated to DSP
- Replication
- On-chip memory/cache
- Extended parallelism- SIMD, VLIW and static superscalar processing

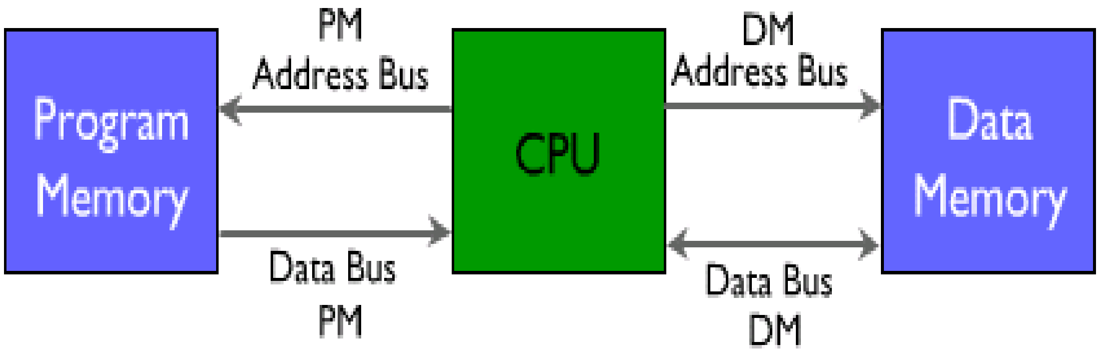
Difference in Architecture

Von Neumann Architecture



- All General Purpose processors follow Von Neumann architecture
- In this architecture :
  - Single memory is shared by both the program instructions and data
  - Single data bus is used to transfer data to and from the CPU
  - CPU places Memory addresses on the address bus and then fetches
  - instruction using the same data bus
- Most computers and traditional microprocessors in use today are built on this architecture
- Disadvantage: consider a MAC instruction, which consists of four operations that must be executed. => 4 Cycles needed
- Limitations: A single, shared memory and single, shared bus structure

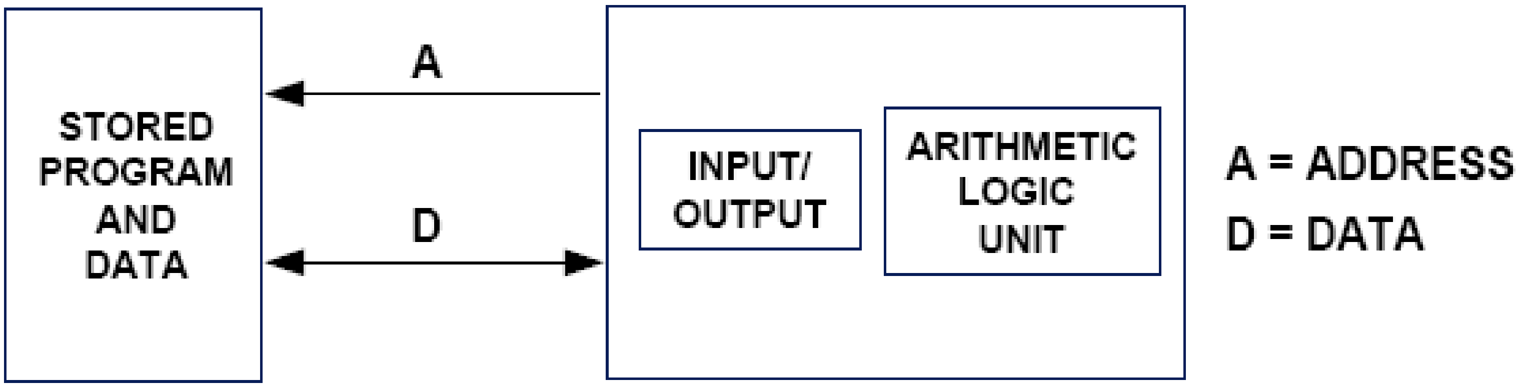
Harvard Architecture



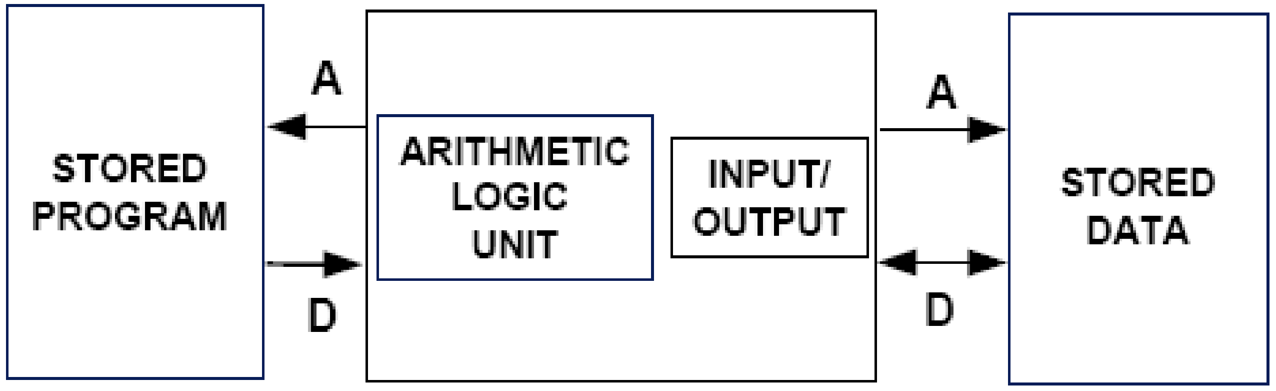
- DSPs follow Harvard architecture
- In this architecture ,
  - Two separate memories, a program memory (PM) for instructions, and a data memory (DM) for data
  - Program instructions and data can be fetched at the same time, increasing overall processing speed.
  - Help to implement specialized instructions like MAC *p,*q ,A (A = A + *p * *q) which executes in single cpu clock cycle
- Today, a modified Harvard architecture is also available, which incorporates three memory banks, each with its own set of buses. These memory banks include two data memories and one program memory.

Digital Computers

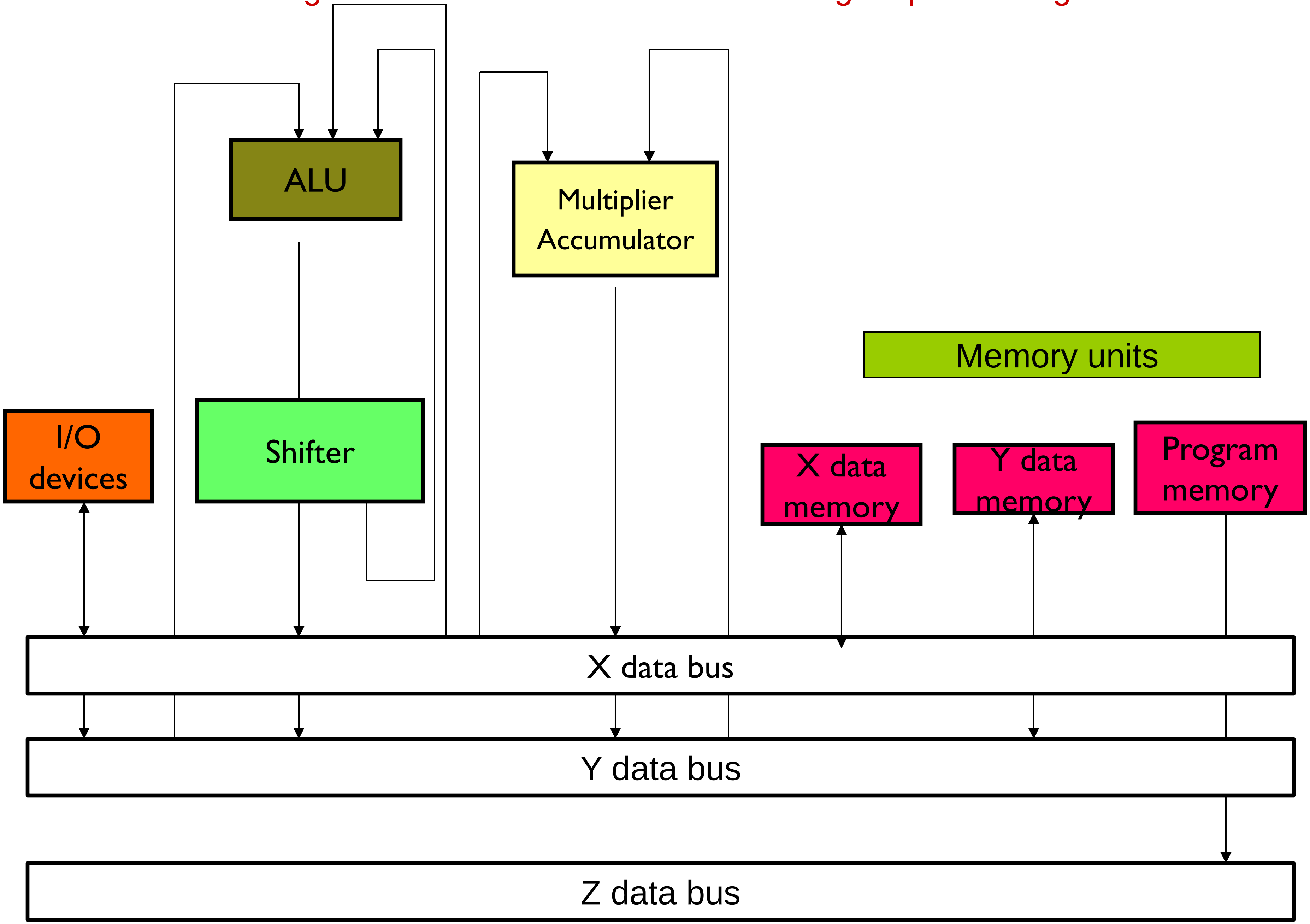
von Neuman Machine



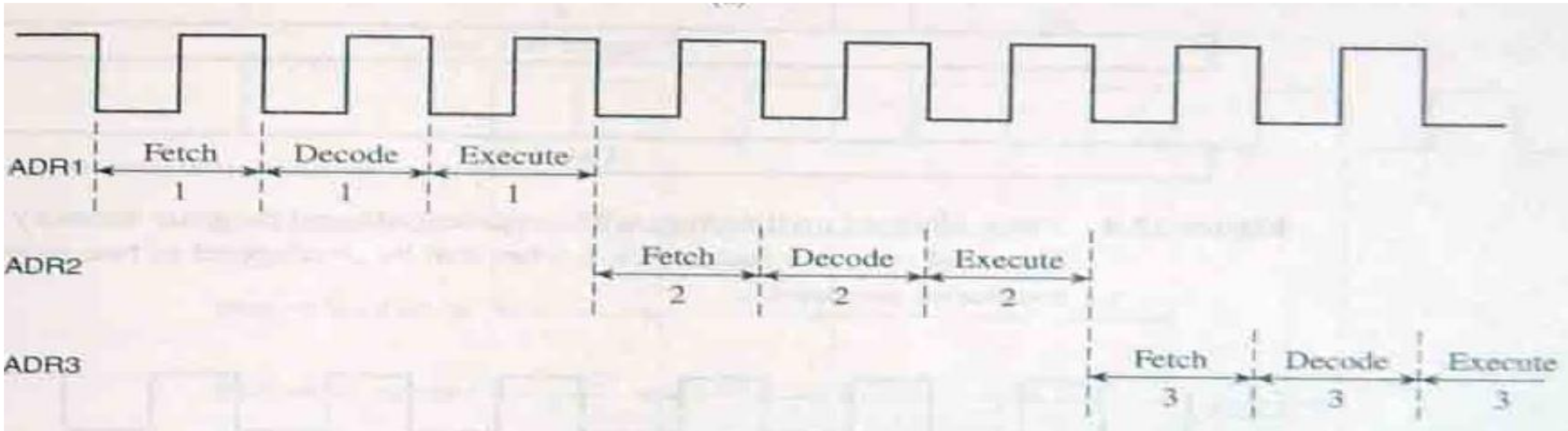
Harvard Architecture

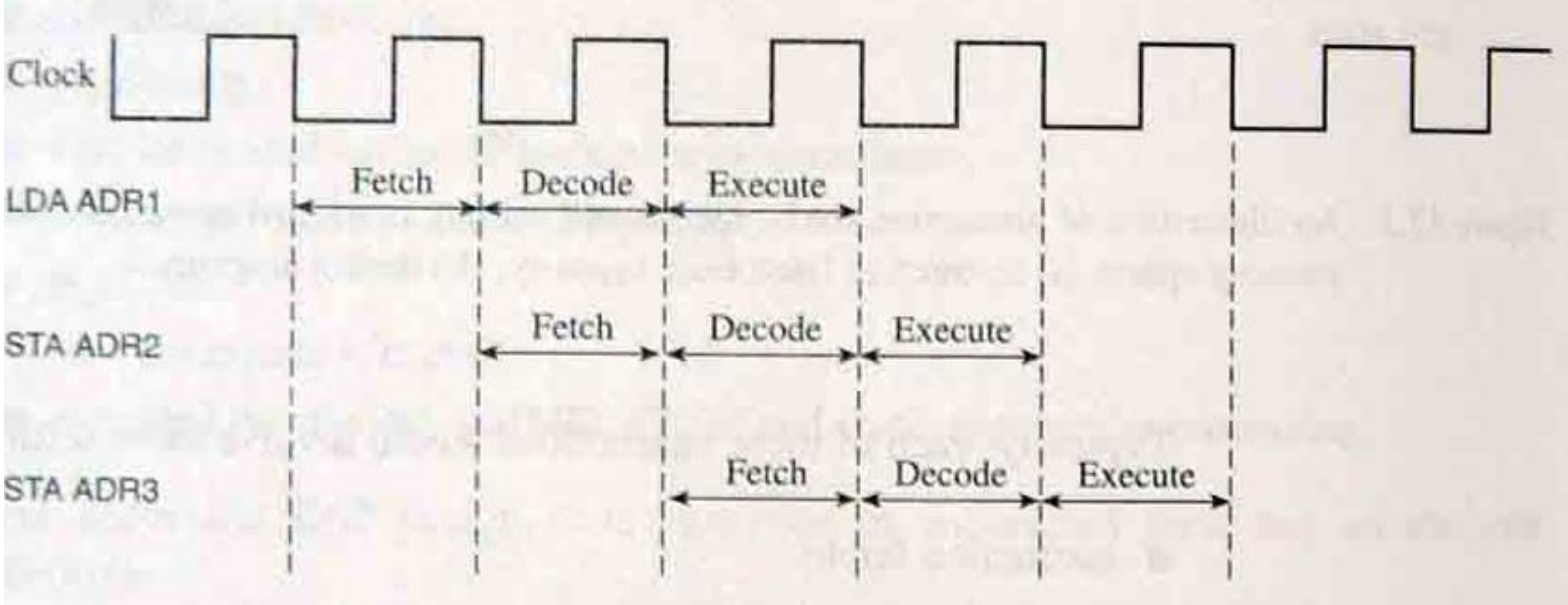


Basic generic hardware architecture for signal processing

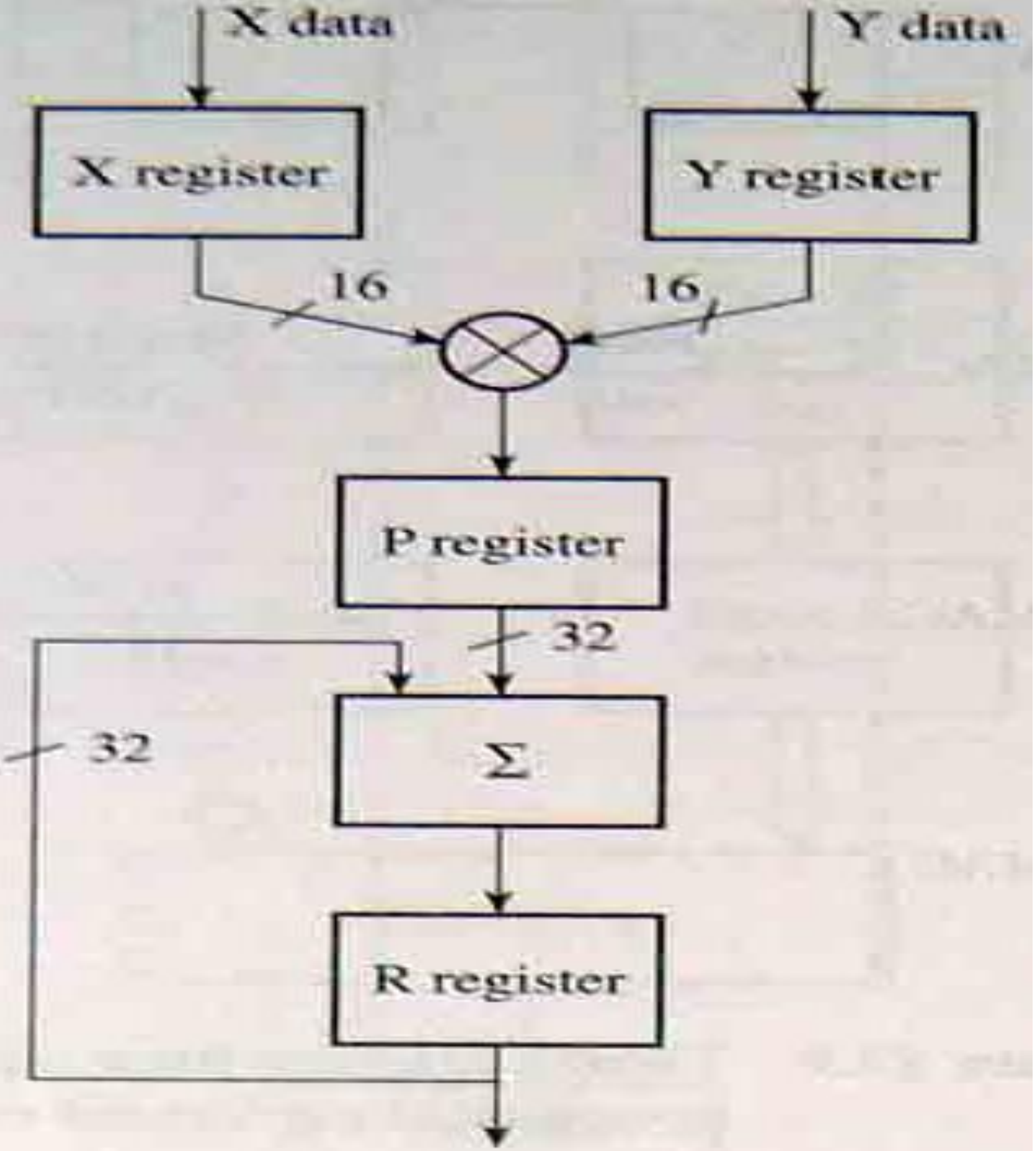


Harvard Architecture





Hardware Multiply and accumulate


X data
Y data
X register
Y register
16
16
P register
32
32
Σ
R register

Multiply and Add

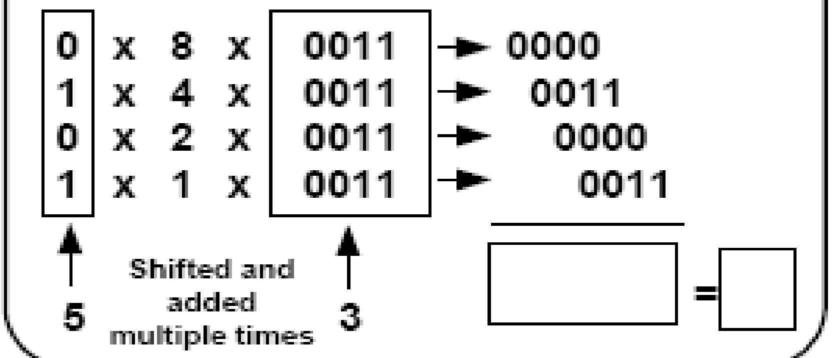
Add

1+2 = 3

$$\begin{array}{r} 0001 \\ +\quad 0010 \\ \hline 0011 \end{array}$$

Multiply

5*3 = 15



0 x 8 x 0011 → 0000

1 x 4 x 0011 → 0011

0 x 2 x 0011 → 0000

1 x 1 x 0011 → 0011

5

Shifted and added multiple times

3

=

Most Common Operation in DSP

$$\boxed{A = B*C + D}$$

$$E = F*G + A$$

⋮

Multiply, Add, and Accumulate

MAC Instruction

MAC Operation

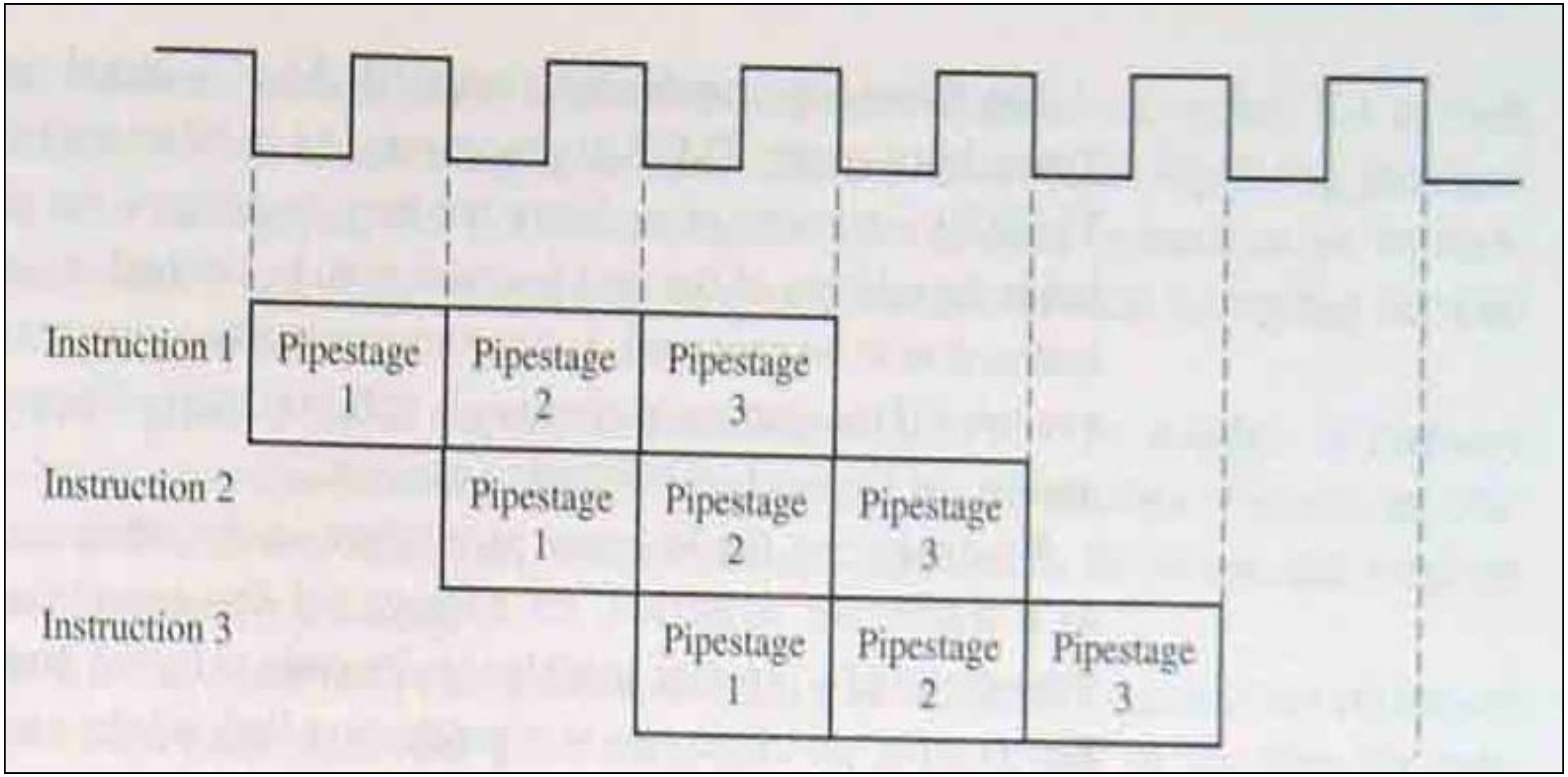
Typically 70 Clock Cycles With Ordinary Processors

Typically 1 Clock Cycle With Digital Signal Processors

Pipelining

- A technique which allows two or more operations to overlap during execution.
- It is used extensively in DSP to increase speed.
- The simultaneous functions going on are:
  - instruction fetch
  - instruction decode
  - instruction execution

Stages of pipelining



Instruction 1	Pipestage 1	Pipestage 2	Pipestage 3		
Instruction 2		Pipestage 1	Pipestage 2	Pipestage 3	
Instruction 3			Pipestage 1	Pipestage 2	Pipestage 3

- In a Harvard architecture (DSP processor with pipelining) :

- The program instructions and data lie in separate memory spaces.

- Due to this the fetching of the next instruction can overlap the execution of the current instruction.

- Each of the step is known as pipeline stage

REGISTERS

- In TMS320 number of registers are used to achieve pipelining.

- **Pre-fetch counter**: holds the address of the next instruction to be fetched.
- **Instruction register**: holds the instruction to be executed.
- **Queue instruction register**: stores the instruction to be executed if the current is still in process of execution.
- **Program counter**: contains the address of the next instruction to be executed

Parameters used in pipelining

- Throughput is determined by the number of instructions through the pipe per unit time.

- In a perfect pipeline the average time per instruction is given by(Hennesy and Patterson,1990)

$$= \frac{\text{time per instruction(non-pipeline)}}{\text{number of pipe stages}}$$

- Speedup= $\dfrac{\text{Average instruction time (nonpipeline)}}{\text{Average instruction time (pipeline)}}$

ADVANTAGES

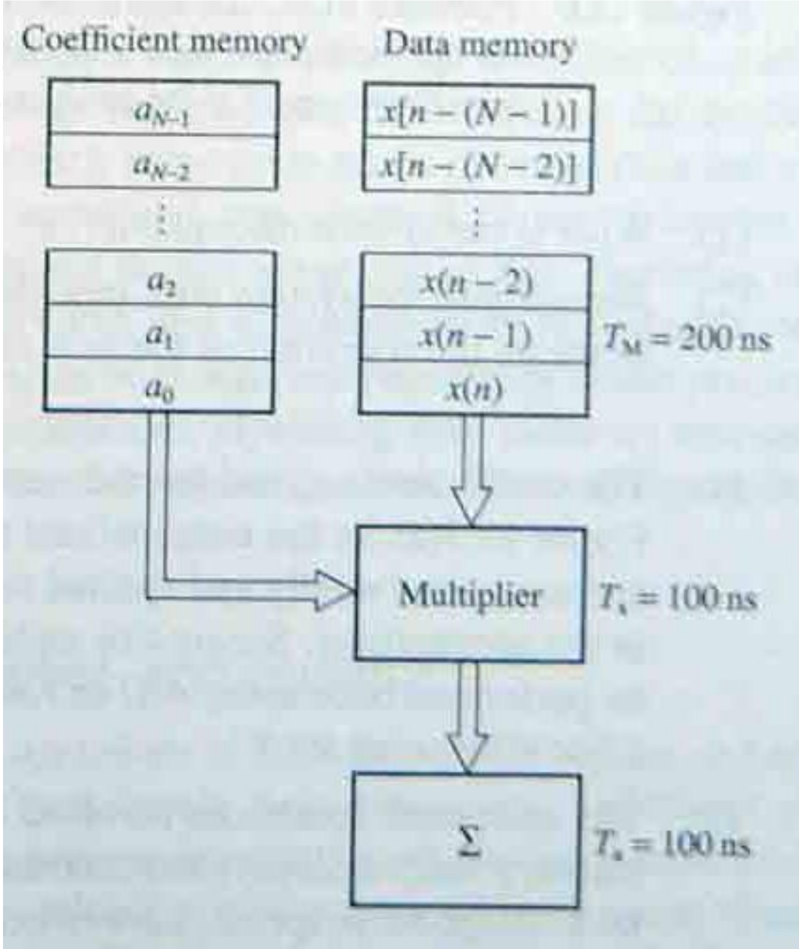
- The cycle time of the processor is reduced, thus increasing instruction bandwidth in most cases.

DISADVANTAGES

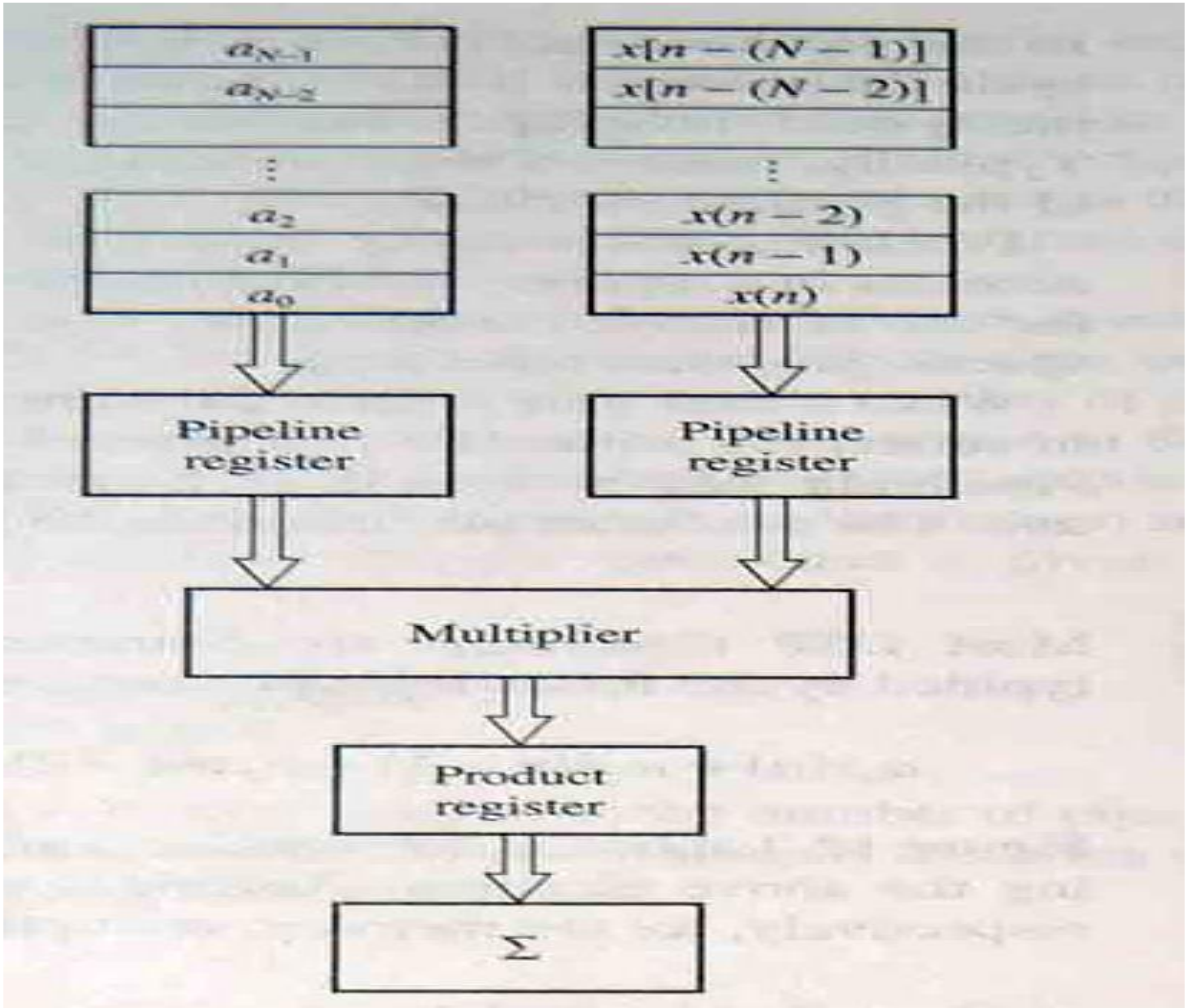
- The design is complex due to addition of extra flip flops.
- The manufacture cost is high.
- The performance of a pipelined processor is much harder to predict and may vary more widely between different programs due to unstable instruction bandwidth.

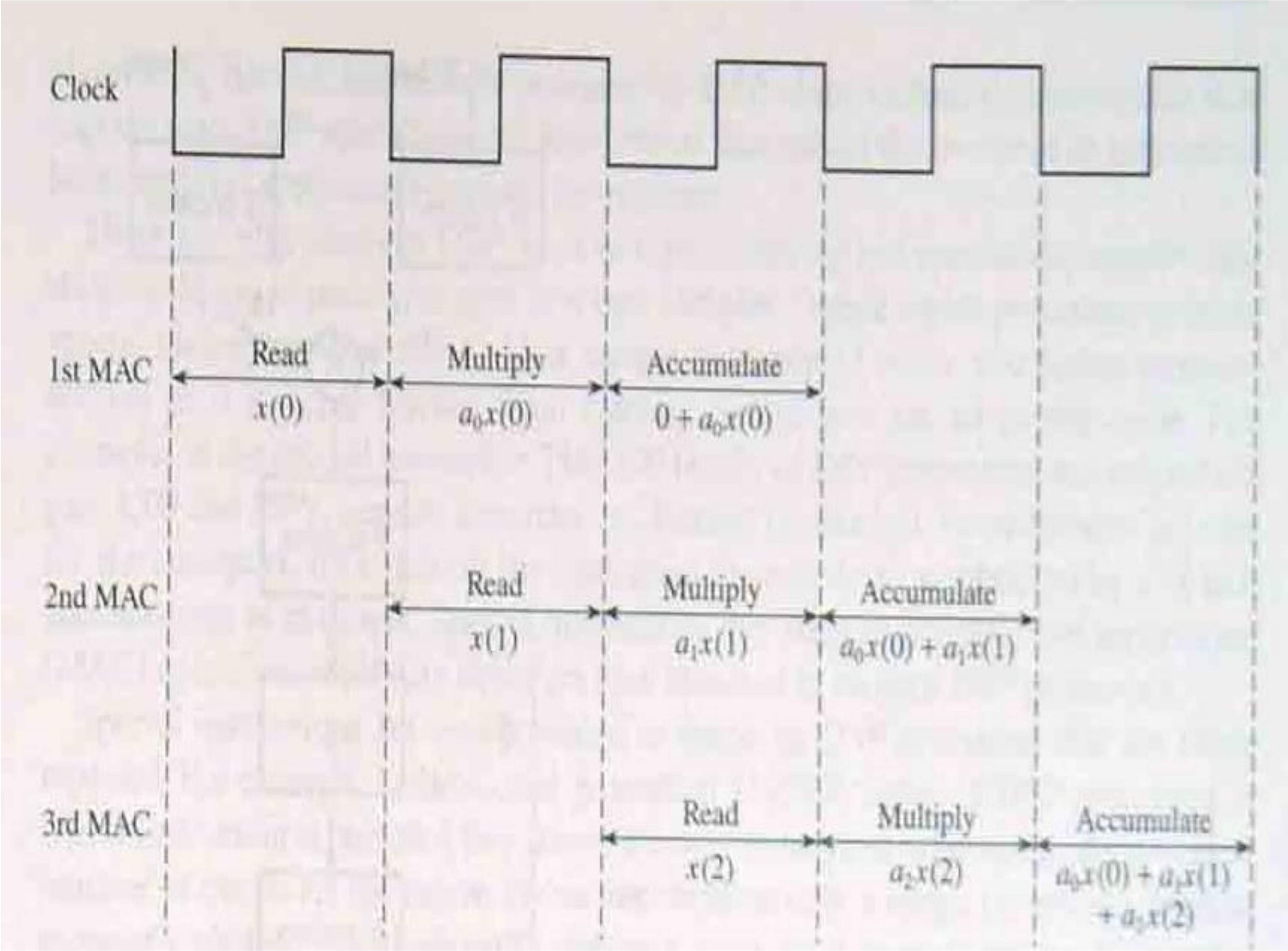
Examples (**pipelining)**

- Multiply and accumulate operations typified by the following equation
- $a_0x(n)+a_1x(n-1)+a_2x(n-2)+\ldots..+A_{N-1}x(N-1$
- Non pipeline of the above equation



MAC Pipeline operation




Clock
1st MAC
Read
$x(0)$
Multiply
$a_0x(0)$
Accumulate
$0 + a_0x(0)$
2nd MAC
Read
$x(1)$
Multiply
$a_1x(1)$
Accumulate
$a_0x(0) + a_1x(1)$
3rd MAC
Read
$x(2)$
Multiply
$a_2x(2)$
Accumulate
$a_0x(0) + a_1x(1)$
$+ a_2x(2)$

Parallelism

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

SIMD (Single instruction multiple data)

- Used to increase number of operation performed per instruction
- Multiple data path and multiple execution unit
- Single instruction may be issued to multiple execution unit to process the block of data simultaneously and in this way number of operation performed in one cycle is increased

SIMD

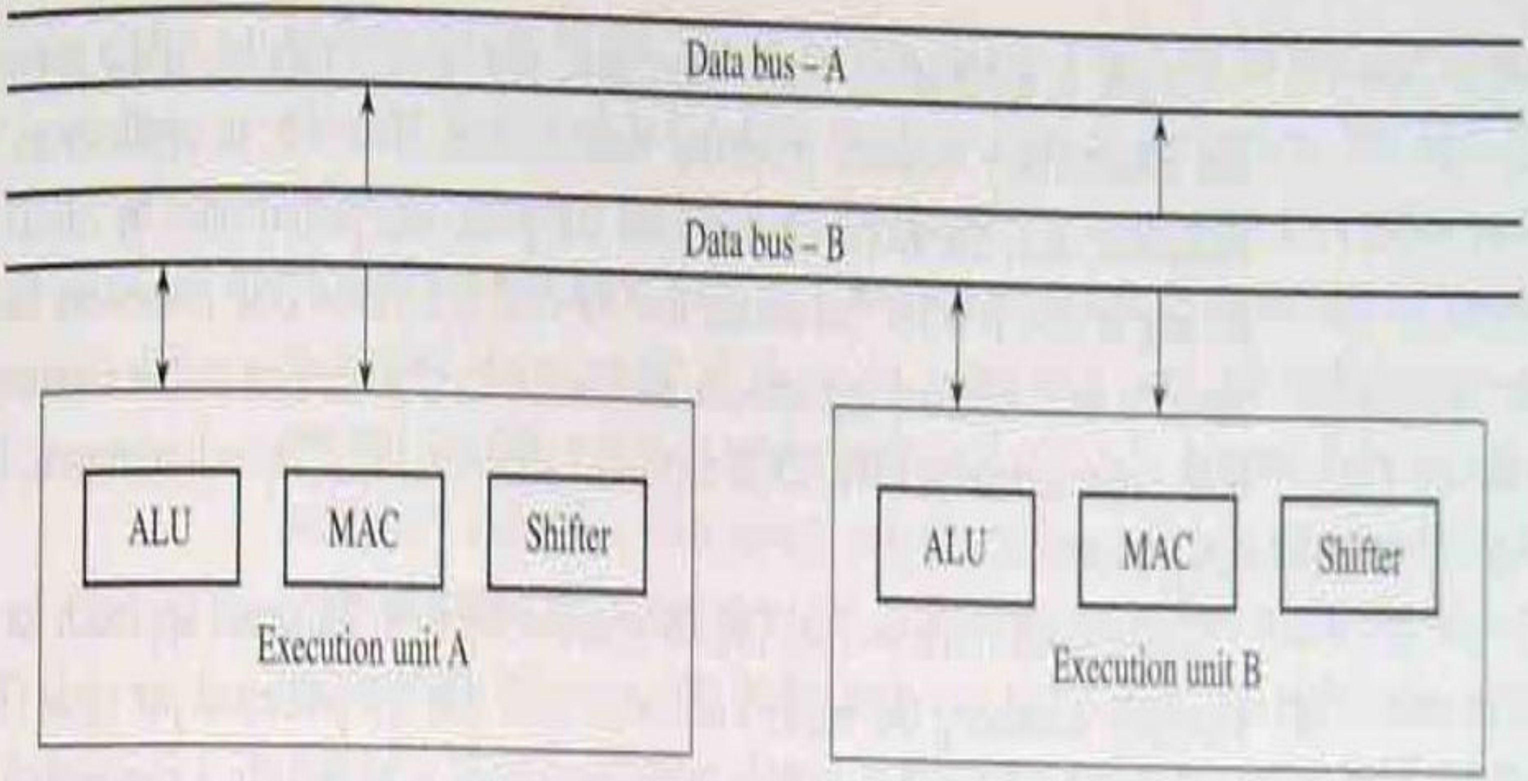
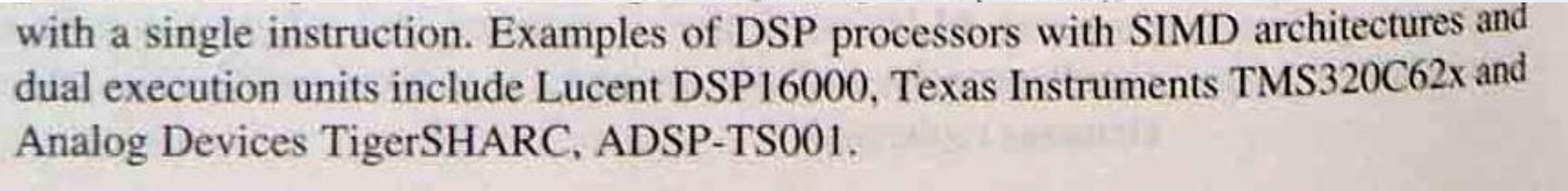
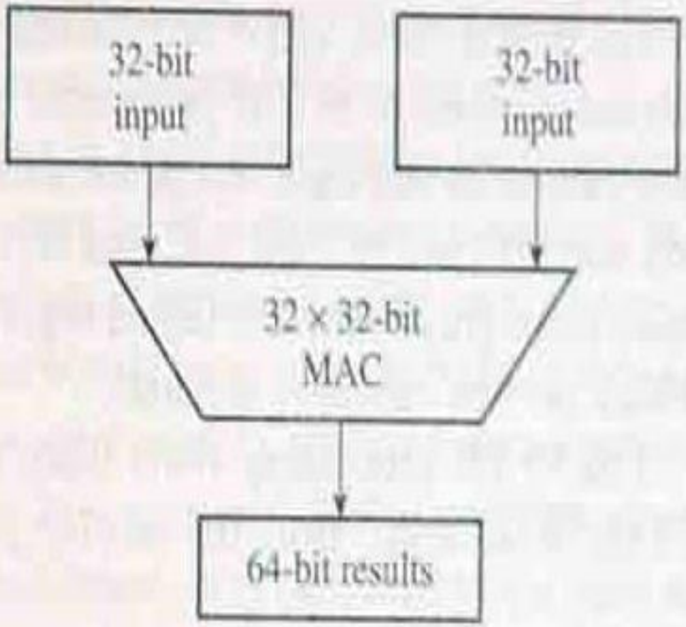


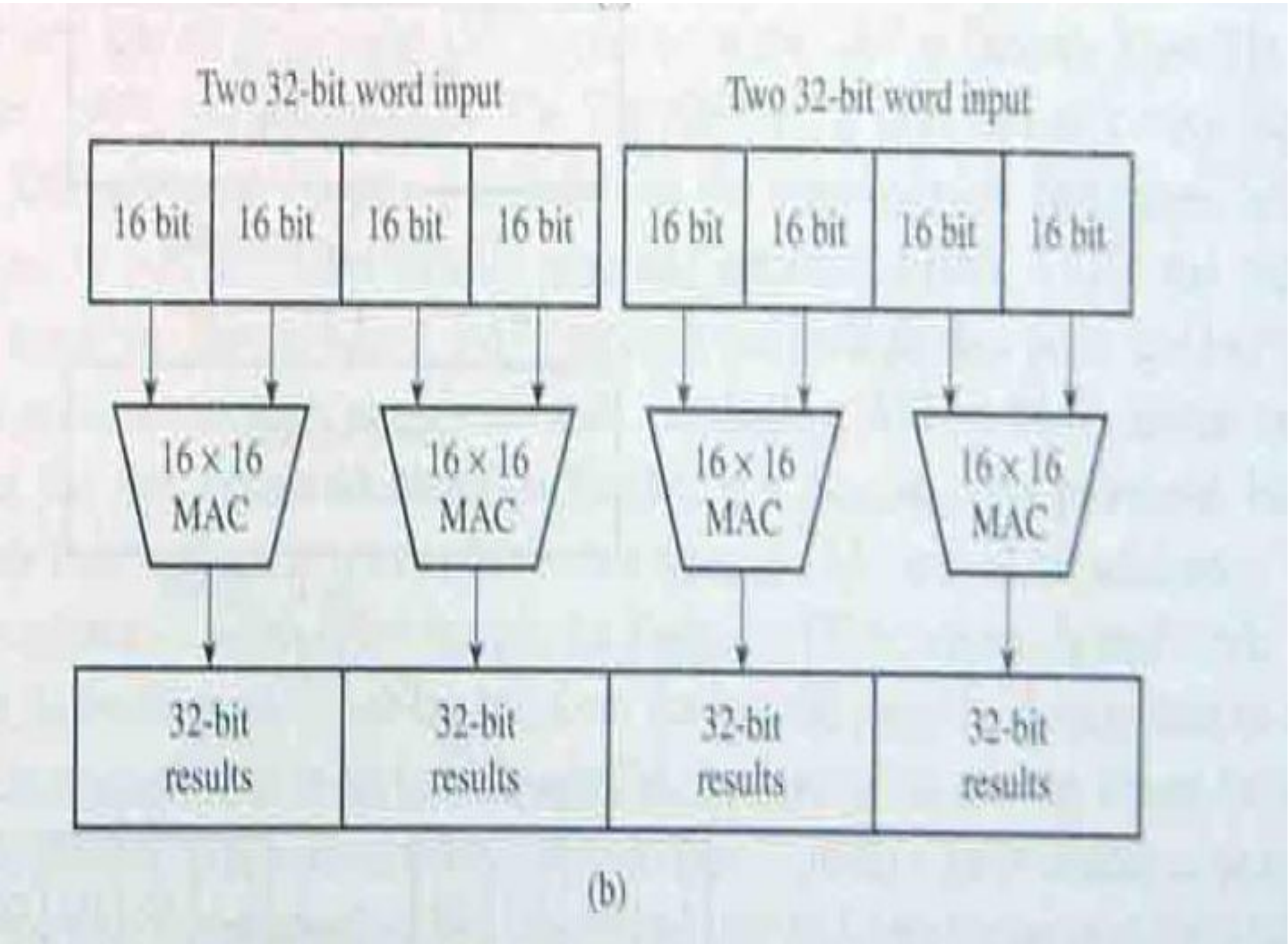
Figure 12.11 Dual arithmetic units with dual data paths for SIMD processing.



with a single instruction. Examples of DSP processors with SIMD architectures and dual execution units include Lucent DSP16000, Texas Instruments TMS320C62x and Analog Devices TigerSHARC, ADSP-TS001.



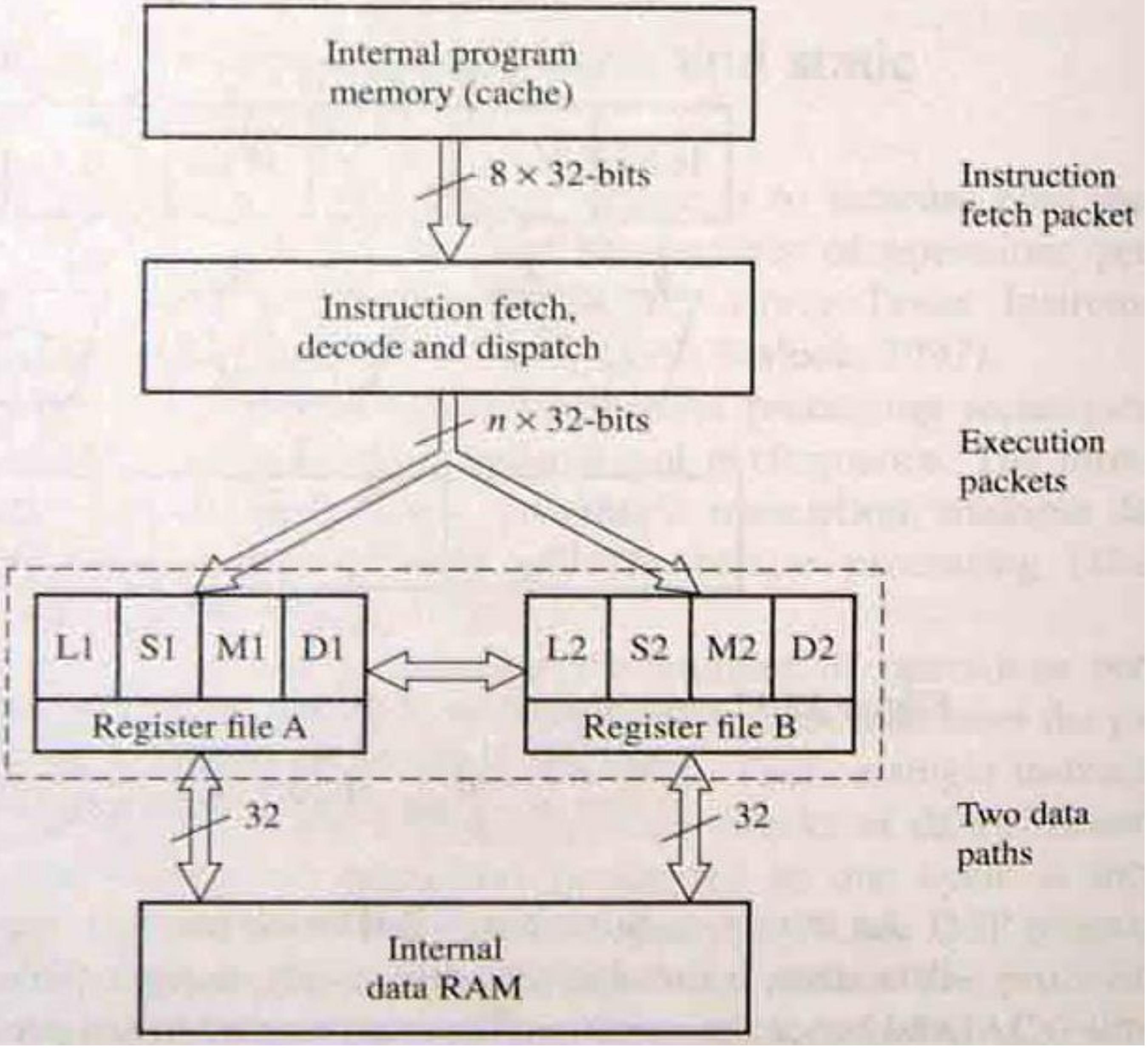
(a)


Two 32-bit word input
Two 32-bit word input
16 bit
16 bit
16 bit
16 bit
16 bit
16 bit
16 bit
16 bit
16 × 16 MAC
16 × 16 MAC
16 × 16 MAC
16 × 16 MAC
32-bit results
32-bit results
32-bit results
32-bit results

(b)

VLIW (Very Long Instruction Word)

- Substantially increasing the number of instruction that are processed per cycle
- Essentially a concatenation of several short instruction and require multiple execution units running in parallel to carry out the instructions in a single cycle

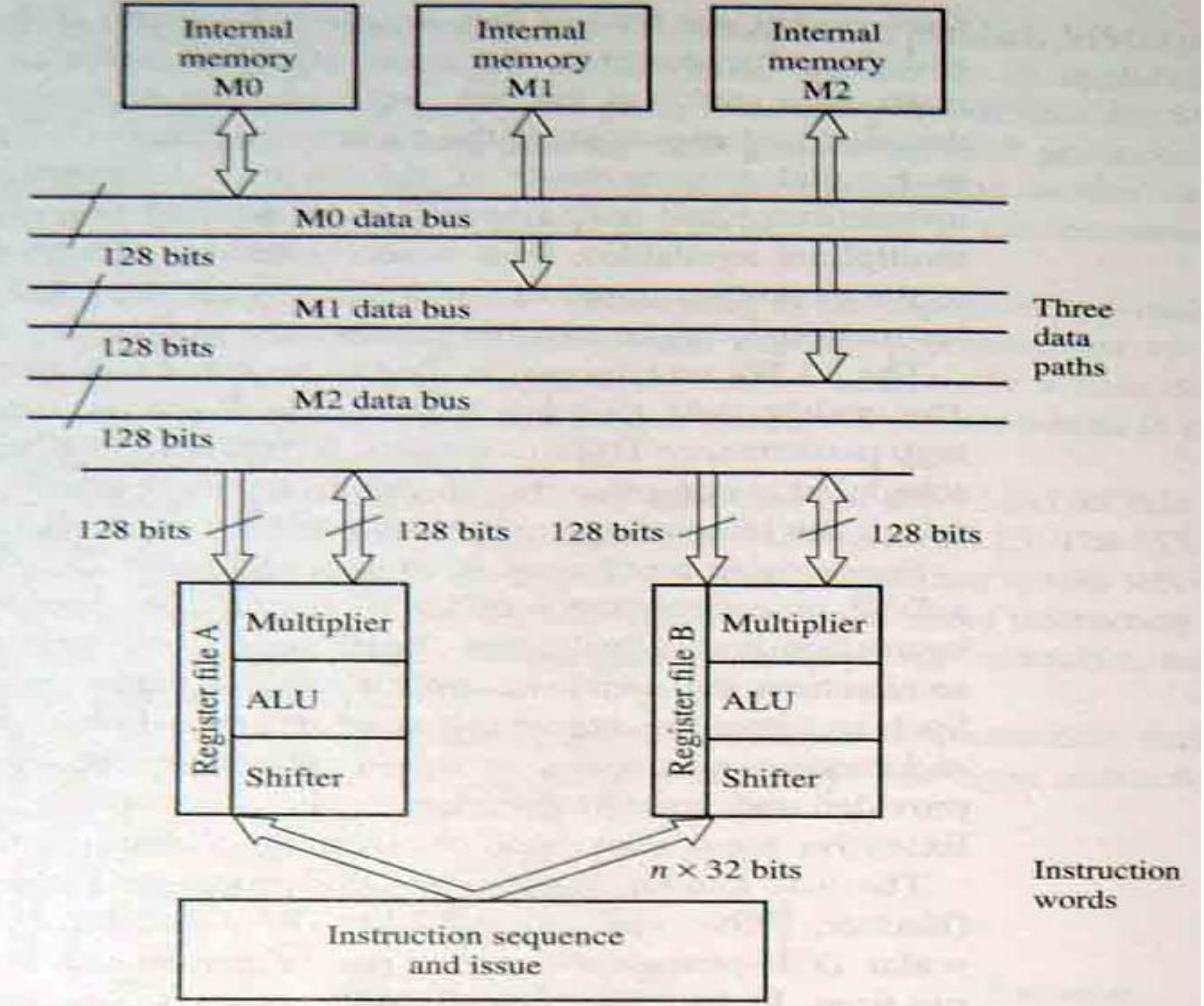

Internal program memory (cache)
8 × 32-bits
Instruction fetch packet
Instruction fetch, decode and dispatch
n × 32-bits
Execution packets
L1
S1
M1
D1
L2
S2
M2
D2
Register file A
Register file B
32
32
Two data paths
Internal data RAM

Features of VLIW

- The cpu contain two data paths and eight independent execution units, organized in two sets –(L1,S1,M1,D1) and (L2,S2,M2,D2)
- Each short instruction is 32 bits wide and eight of these are linked together to form a very long instruction word packed which may be executed in parallel.
- VLIW architecture is clearly designed to support instruction level parallelism, together with fast clock speeds-200MHz

Super scalar processing

- Increasing the instruction rate of a DSP processor (the number of instruction processed in a cycle) by exploiting instruction-level parallelism.
- Super scalar- computer architecture that enable multiple instruction to be executed in one cycles
- It is widely used in general purpose processor such as power PC, and Pentium processor
- Best known super scalar processor is the Analog devices Tiger SHARC


Internal memory M0
Internal memory M1
Internal memory M2
M0 data bus
128 bits
M1 data bus
128 bits
M2 data bus
128 bits
Three data paths
128 bits
128 bits
128 bits
128 bits
Register file A
Multiplier
ALU
Shifter
Register file B
Multiplier
ALU
Shifter
$n \times 32$ bits
Instruction words
Instruction sequence and issue

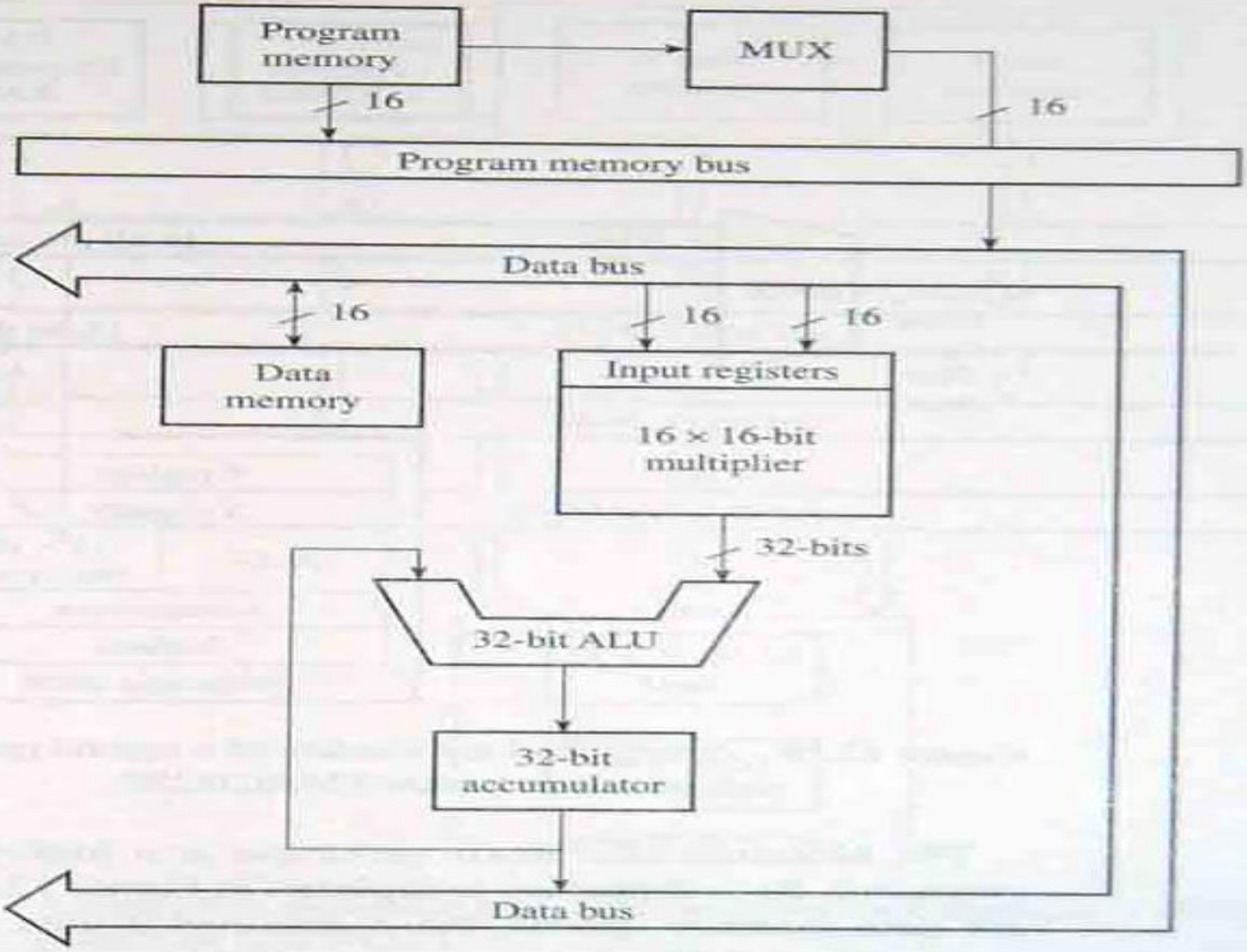
General Purpose Digital Signal processor

- It is a High speed microprocessor with hardware architecture

- Instruction sets are optimized for DSP operations

- Extensive use of parallelism, Harvard architecture, pipelining and dedicated hardware, shifting, scaling, multiplication and so on

Fixed point Digital Signal Processors



A simplified architecture of a first generation fixed-point DSP processor (Texas Instruments TMS320C10).

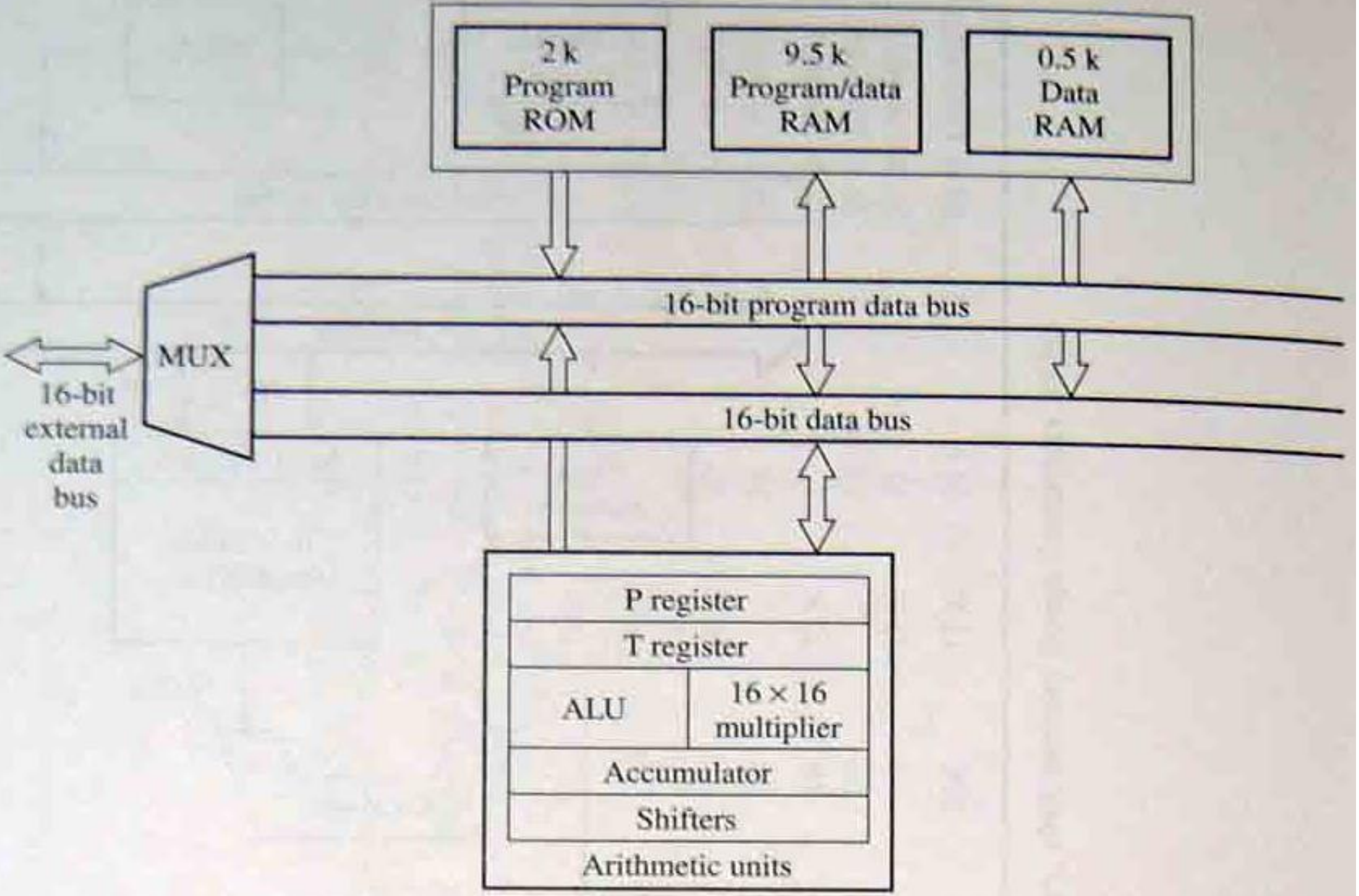


Figure 12.16 A simplified architecture of a second generation fixed-point DSP (Texas Instruments TMS320C50).

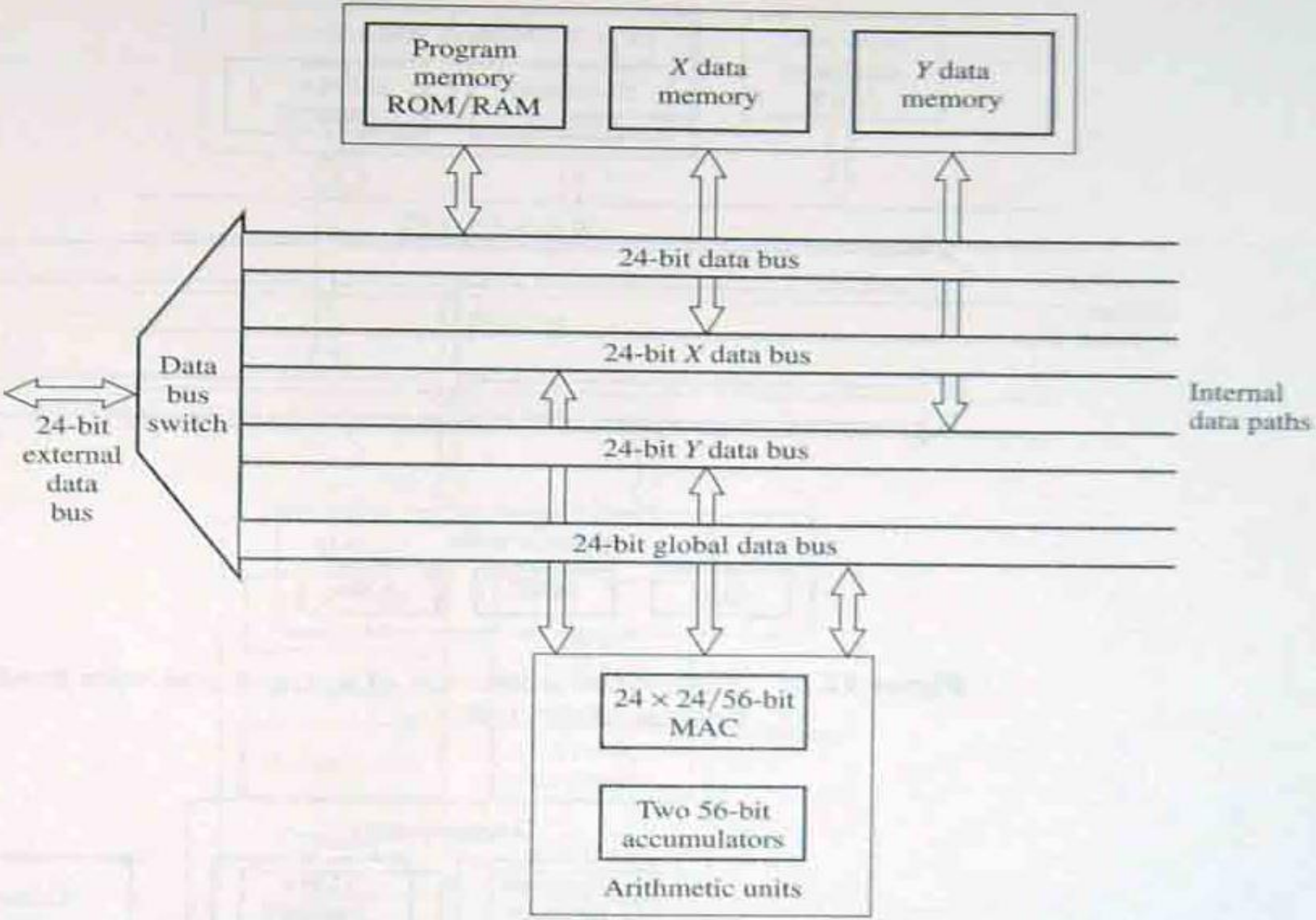


Figure 12.17 A simplified architecture of a second generation fixed-point DSP (Motorola DSP56002)

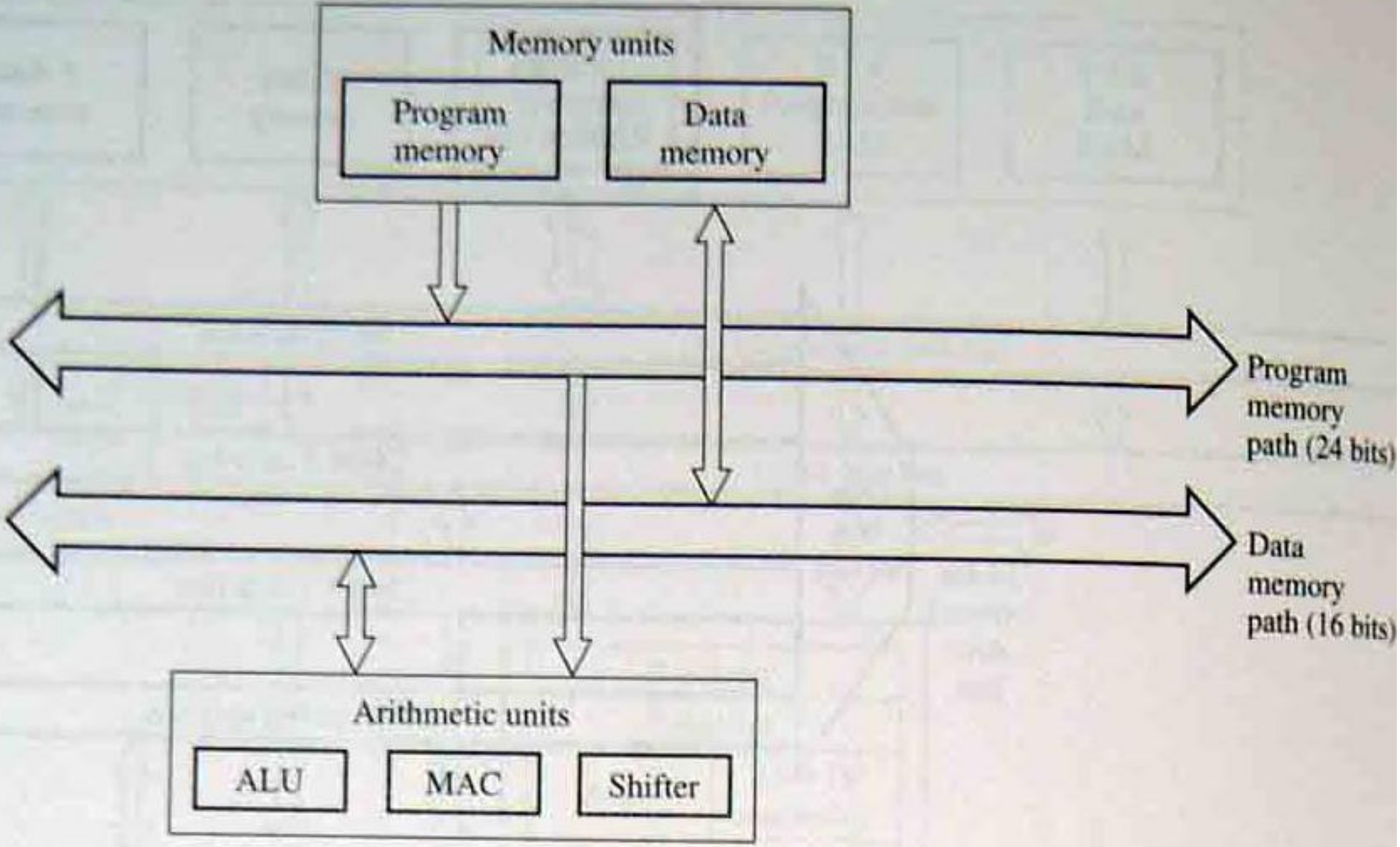
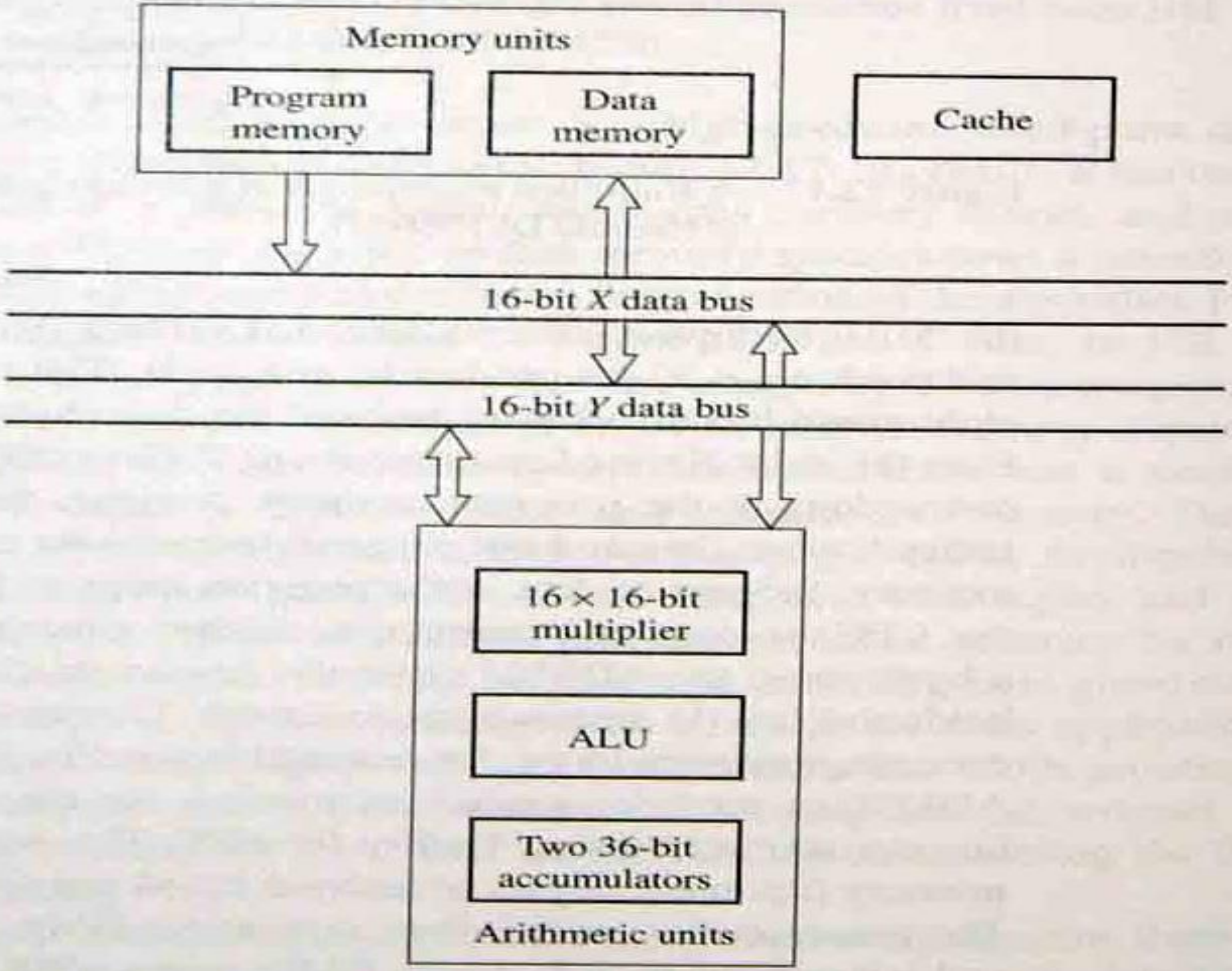
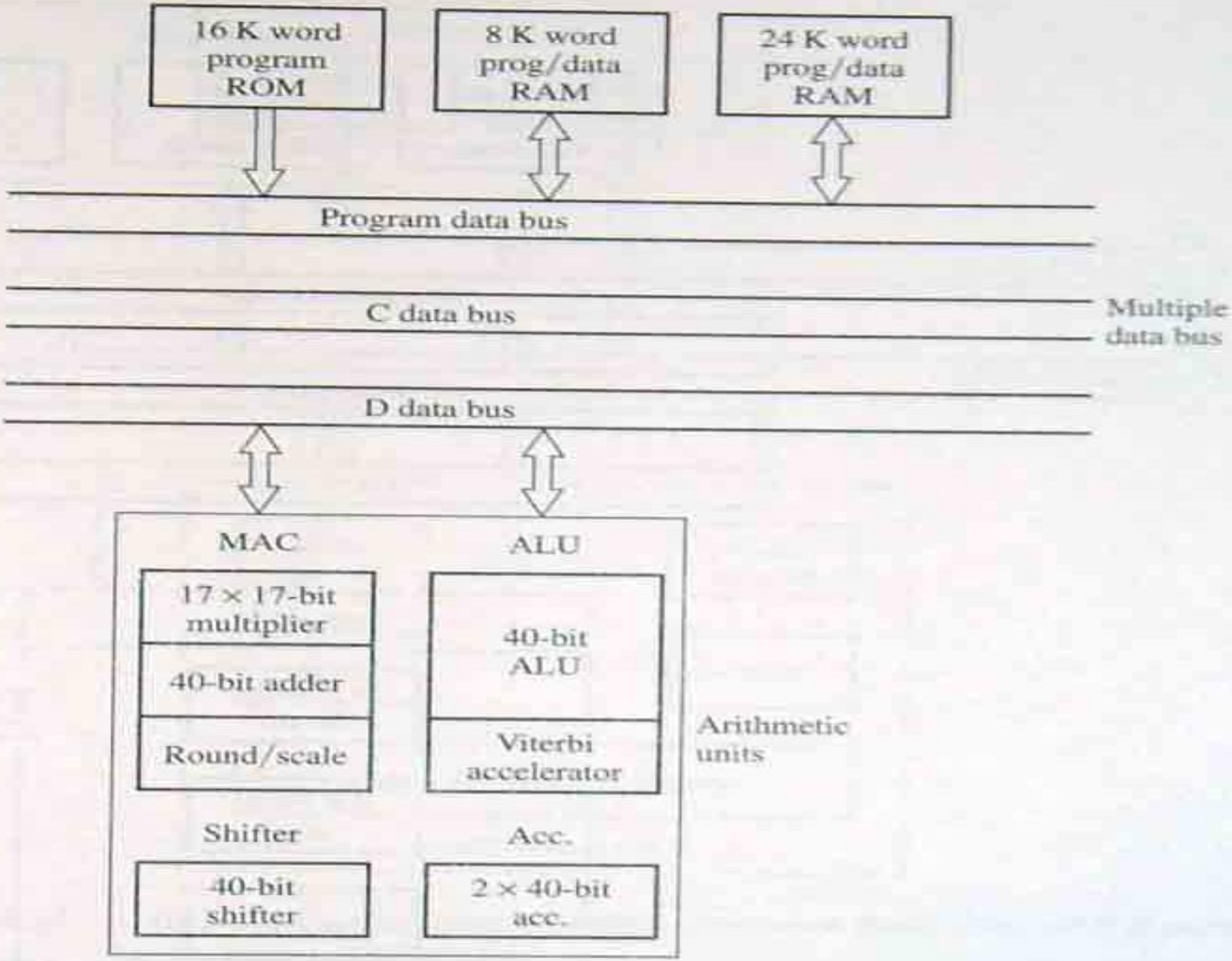


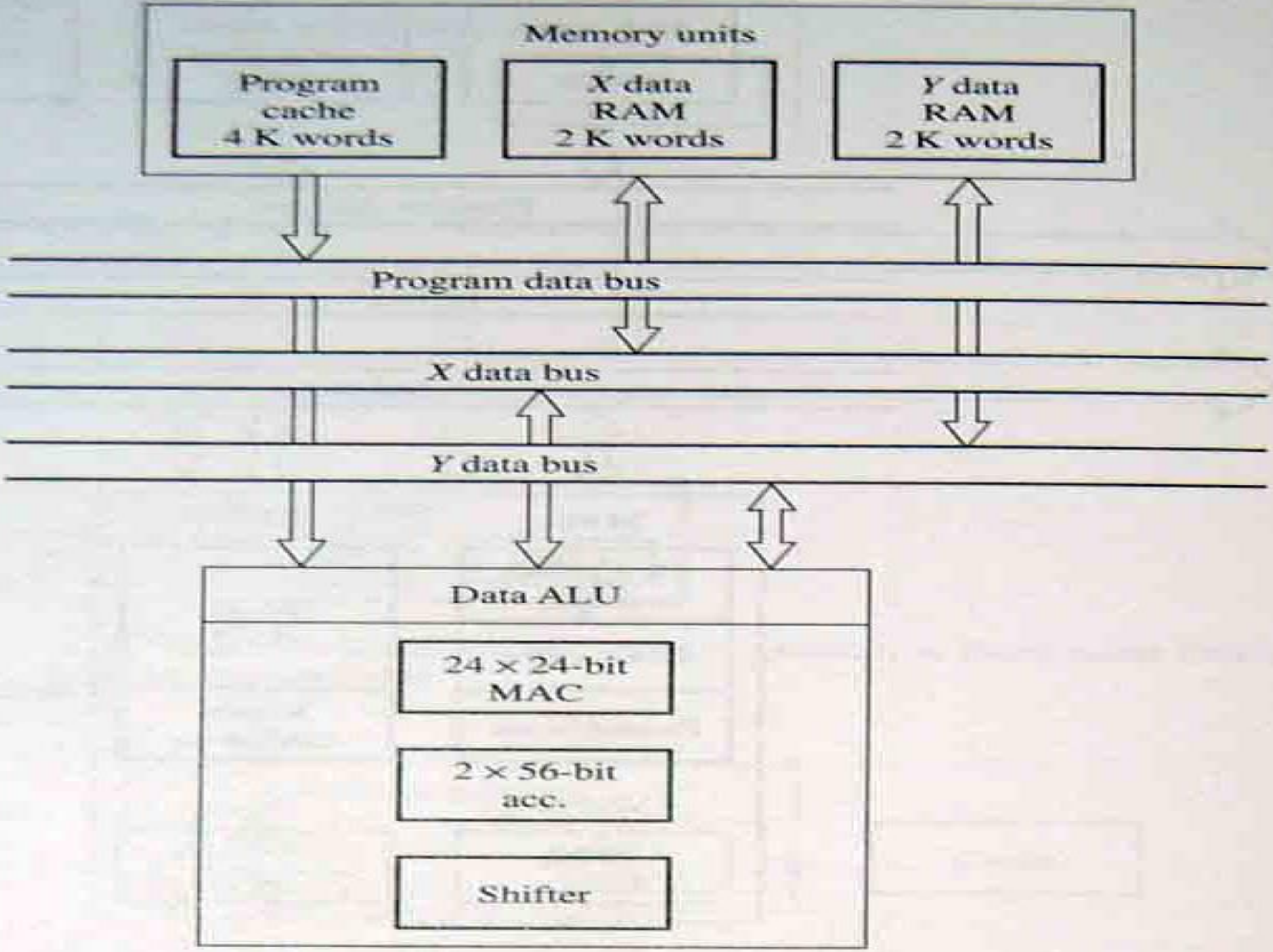
Figure 12.18 A simplified architecture of a second generation fixed-point DSP (Analog Devices ADSP2100).



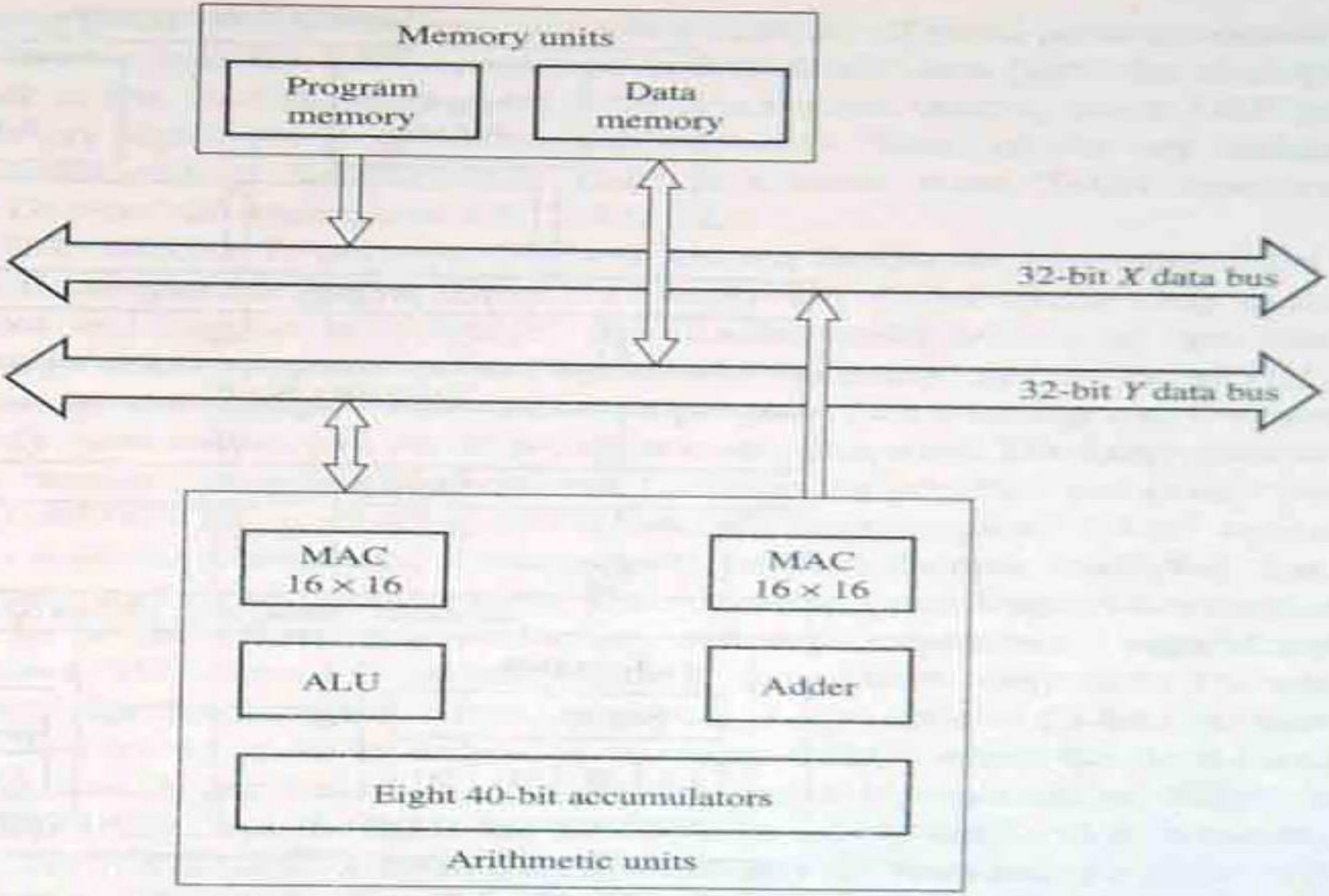
.9 A simplified architecture of Lucent Technologies' DSP16xx fixed-point DSP.



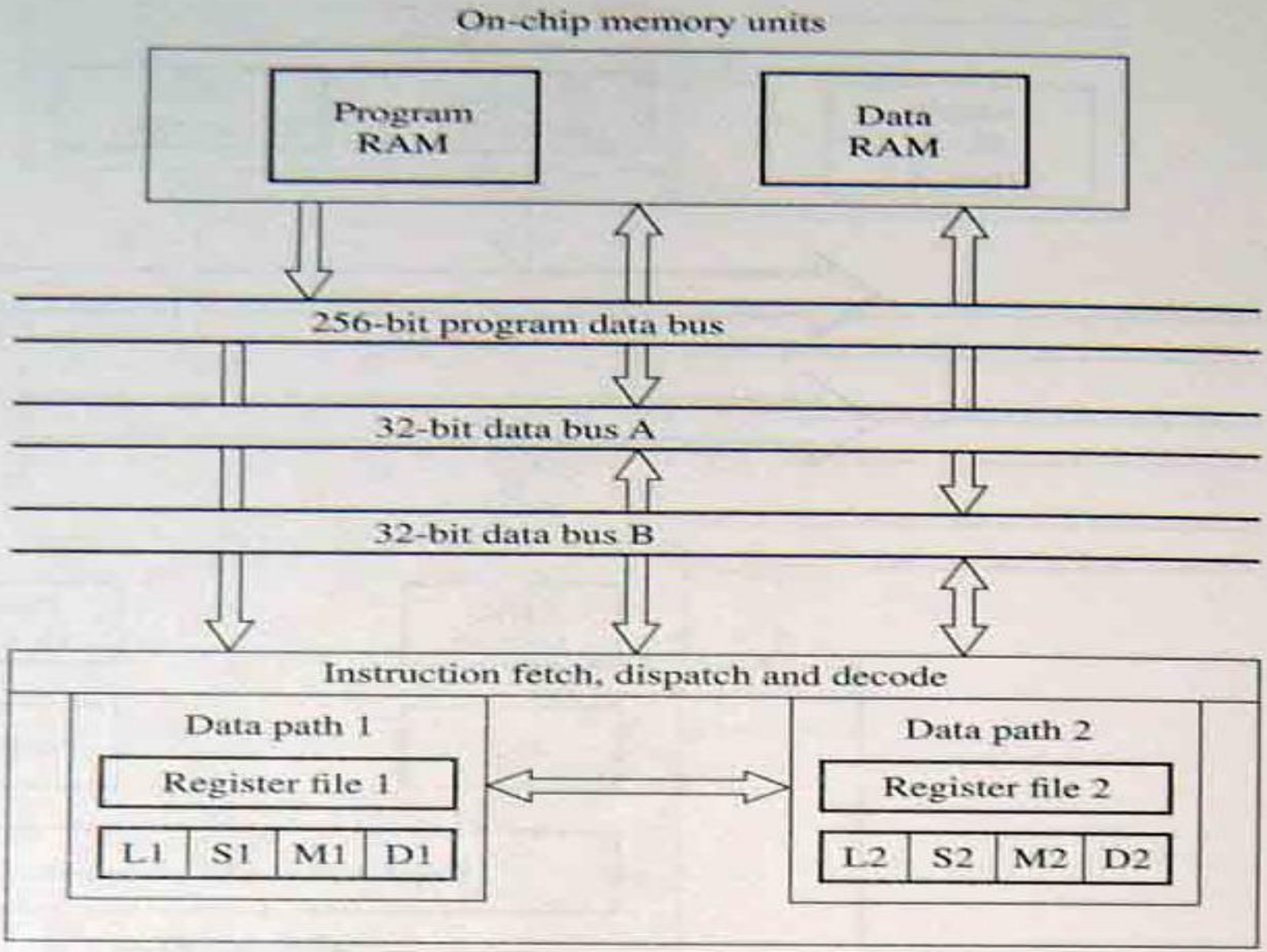
ıre 12.20 A simplified architecture of a third generation fixed-point DSP (Texas Instruments TMS320C54x).



.21 A simplified architecture of a third generation fixed-point DSP (Motorola DSP56300).



2.22 A simplified architecture of a third generation fixed-point DSP (Lucent Technologies DSP16000).



.23 A simplified architecture of a fourth generation fixed-point, very long instruction word, DSP processor (Texas Instruments TMS320C62x). Note the two independent arithmetic data paths, each with four execution units – L1, S1, M1 and D1; L2, S2, M2 and D2.

Fixed point Digital Signal processors

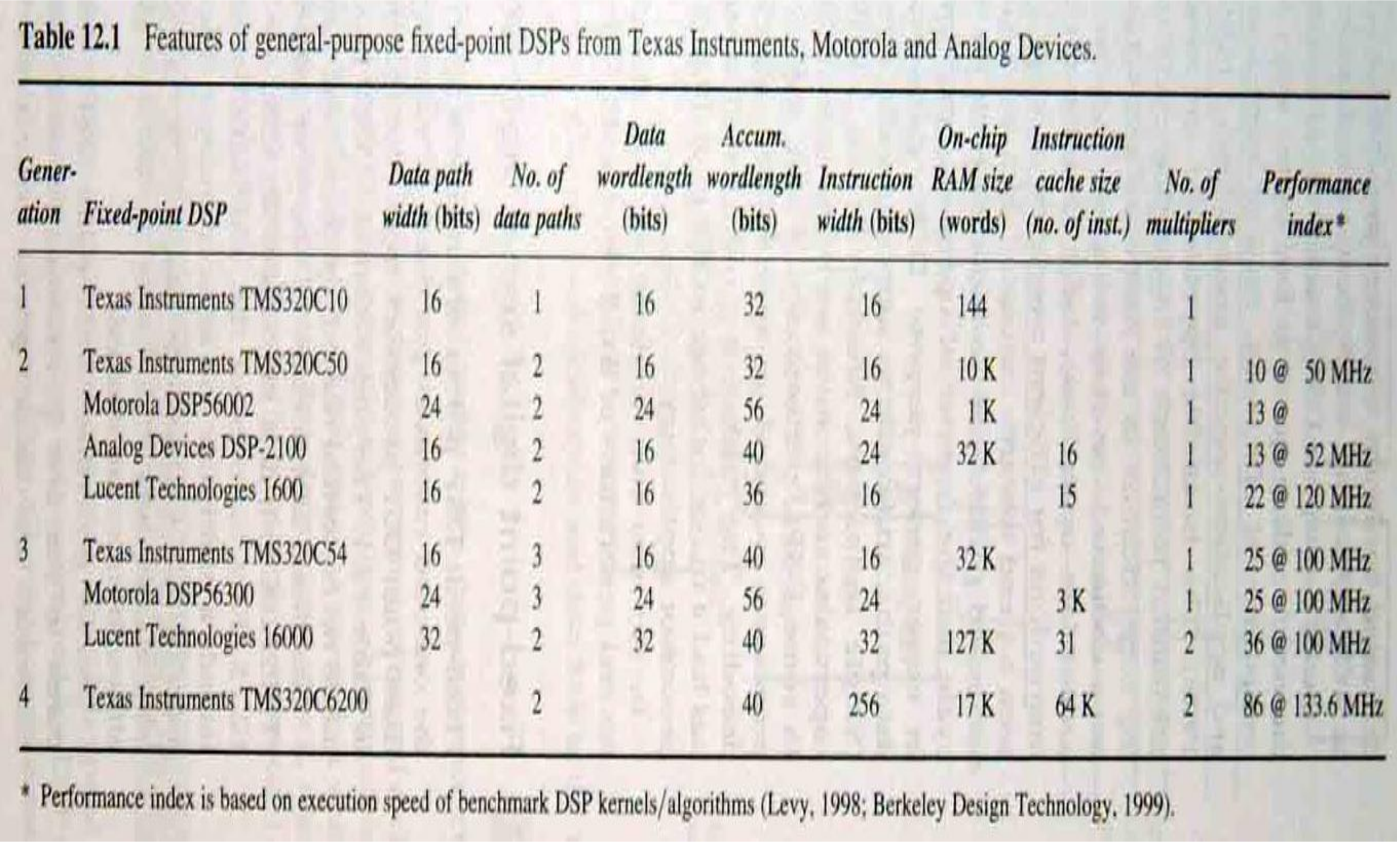


Table 12.1 Features of general-purpose fixed-point DSPs from Texas Instruments, Motorola and Analog Devices.

Gener-ation	*Fixed-point DSP*	*Data path width* (bits)	*No. of data paths*	*Data wordlength* (bits)	*Accum. wordlength* (bits)	*Instruction width* (bits)	*On-chip RAM size* (words)	*Instruction cache size* (*no. of inst.*)	*No. of multipliers*	*Performance index**
1	Texas Instruments TMS320C10	16	1	16	32	16	144		1	
2	Texas Instruments TMS320C50	16	2	16	32	16	10 K		1	10 @ 50 MHz
	Motorola DSP56002	24	2	24	56	24	1 K		1	13 @
	Analog Devices DSP-2100	16	2	16	40	24	32 K	16	1	13 @ 52 MHz
	Lucent Technologies 1600	16	2	16	36	16		15	1	22 @ 120 MHz
3	Texas Instruments TMS320C54	16	3	16	40	16	32 K		1	25 @ 100 MHz
	Motorola DSP56300	24	3	24	56	24		3 K	1	25 @ 100 MHz
	Lucent Technologies 16000	32	2	32	40	32	127 K	31	2	36 @ 100 MHz
4	Texas Instruments TMS320C6200		2		40	256	17 K	64 K	2	86 @ 133.6 MHz

* Performance index is based on execution speed of benchmark DSP kernels/algorithms (Levy, 1998; Berkeley Design Technology, 1999).

Floating Point Processors

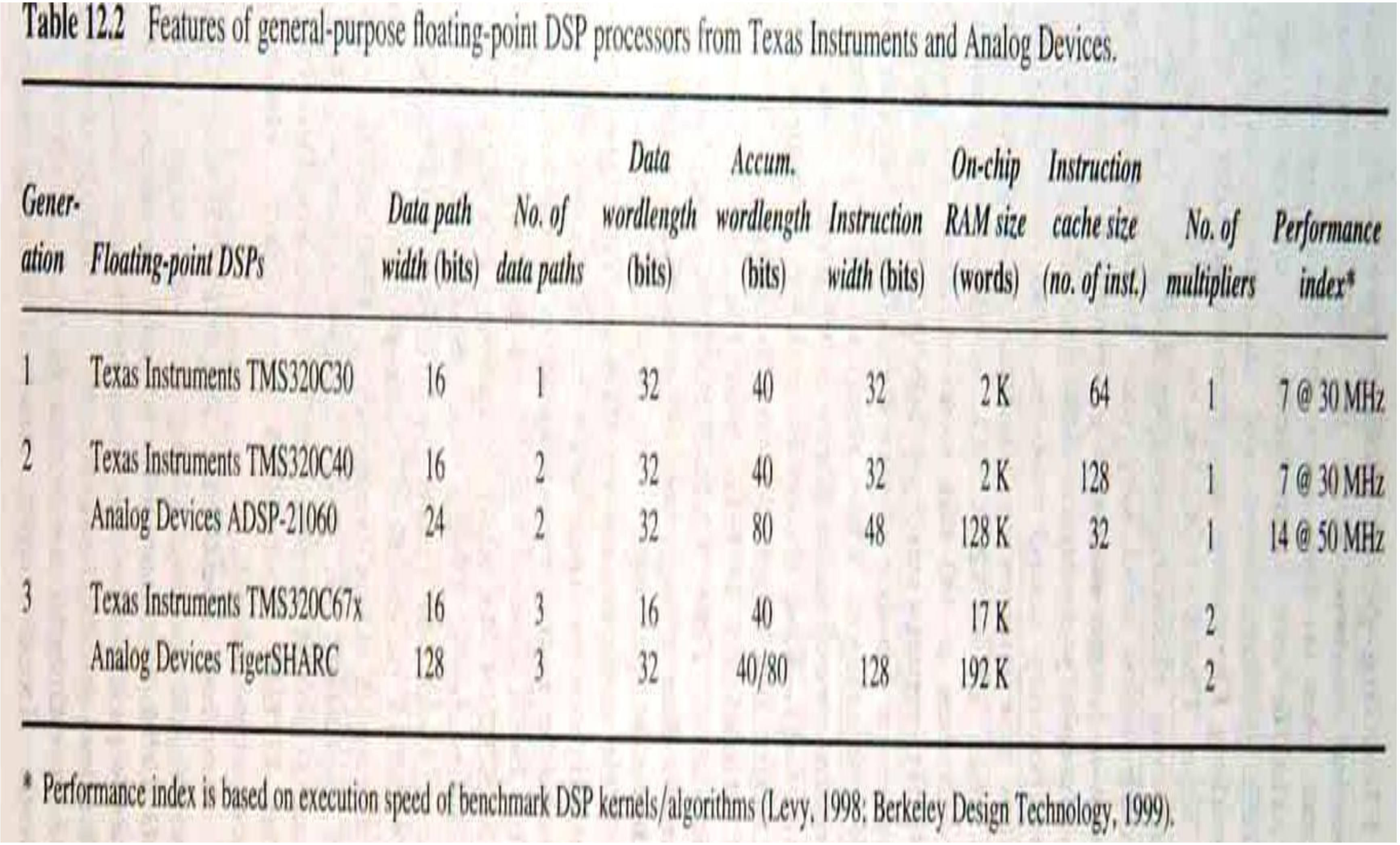


Table 12.2 Features of general-purpose floating-point DSP processors from Texas Instruments and Analog Devices.

Gener-ation	*Floating-point DSPs*	*Data path width* (bits)	*No. of data paths*	*Data wordlength* (bits)	*Accum. wordlength* (bits)	*Instruction width* (bits)	*On-chip RAM size* (words)	*Instruction cache size (no. of inst.)*	*No. of multipliers*	*Performance index**
1	Texas Instruments TMS320C30	16	1	32	40	32	2 K	64	1	7 @ 30 MHz
2	Texas Instruments TMS320C40	16	2	32	40	32	2 K	128	1	7 @ 30 MHz
	Analog Devices ADSP-21060	24	2	32	80	48	128 K	32	1	14 @ 50 MHz
3	Texas Instruments TMS320C67x	16	3	16	40		17 K		2	
	Analog Devices TigerSHARC	128	3	32	40/80	128	192 K		2	

* Performance index is based on execution speed of benchmark DSP kernels/algorithms (Levy, 1998; Berkeley Design Technology, 1999).

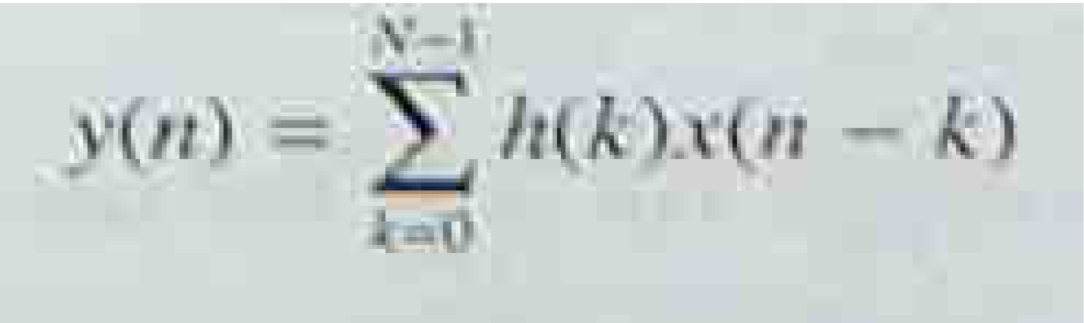
Selecting the Digital Signal Processor

- Architectural features
  - Size of on chip memory
  - Special Instruction
  - I/O capability
- Execution speed
  - Clock speed of the processor(MHz)
  - Number of instruction performed (MIPS) & (MFLOPS)

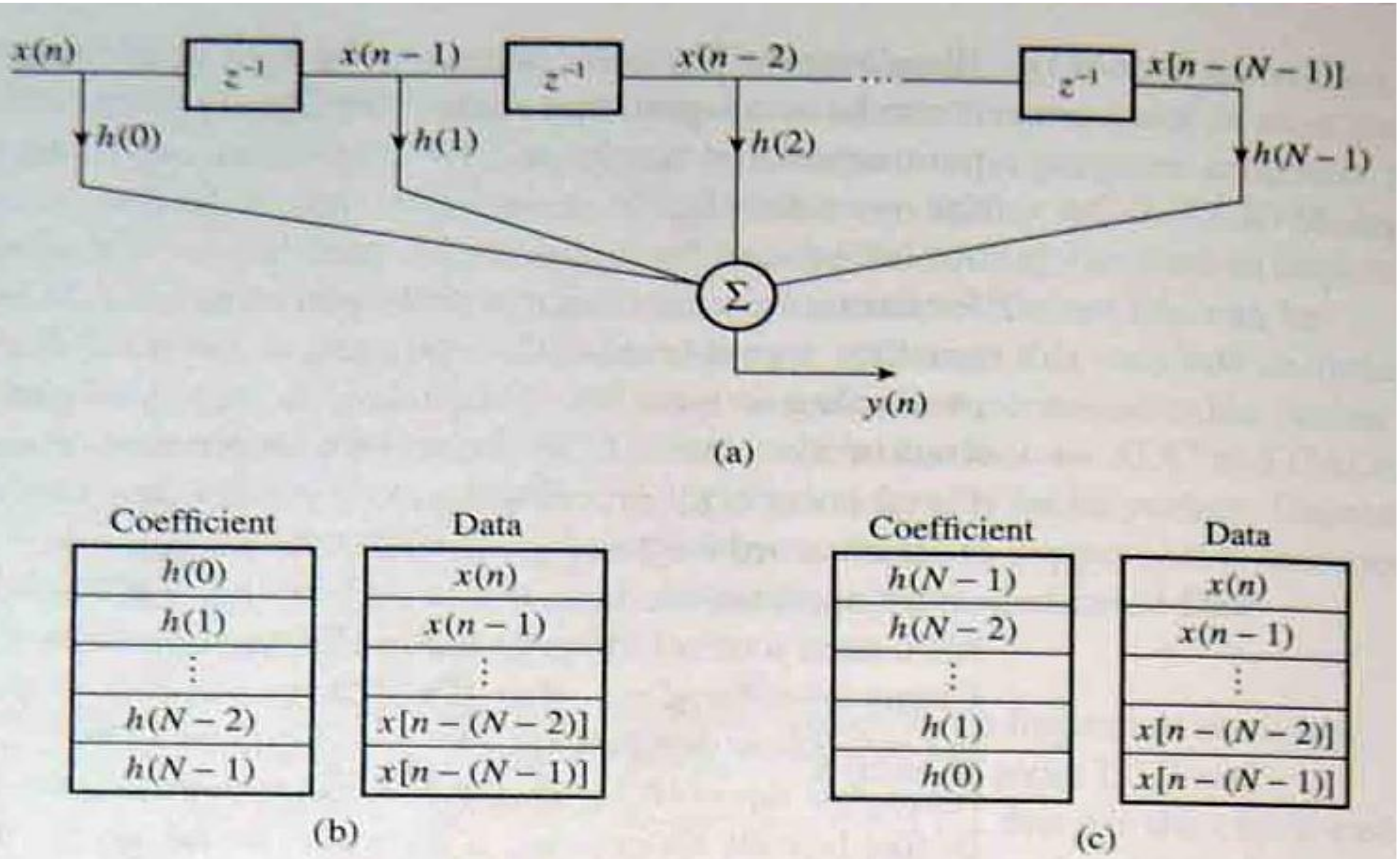
- Type of Arithmetic
  - Fixed point arithmetic
  - Floating point arithmetic
- Word Length
  - 24 bit word length for audio processing
  - Longer data word length lower the error

Implementation of DSP algorithm on general purpose Digital Signal Processor

- FIR Digital filtering



$$y(n) = \sum_{k=0}^{N-1} h(k).x(n-k)$$



Coefficient	Data
$h(0)$	$x(n)$
$h(1)$	$x(n-1)$
⋮	⋮
$h(N-2)$	$x[n-(N-2)]$
$h(N-1)$	$x[n-(N-1)]$

(b)

Coefficient	Data
$h(N-1)$	$x(n)$
$h(N-2)$	$x(n-1)$
⋮	⋮
$h(1)$	$x[n-(N-2)]$
$h(0)$	$x[n-(N-1)]$

(c)

Figure 12.24 Implementation of FIR filter: (a) filter structure; (b) coefficient and data memory maps; (c) alternative memory map.

Thank you