DSP architecture

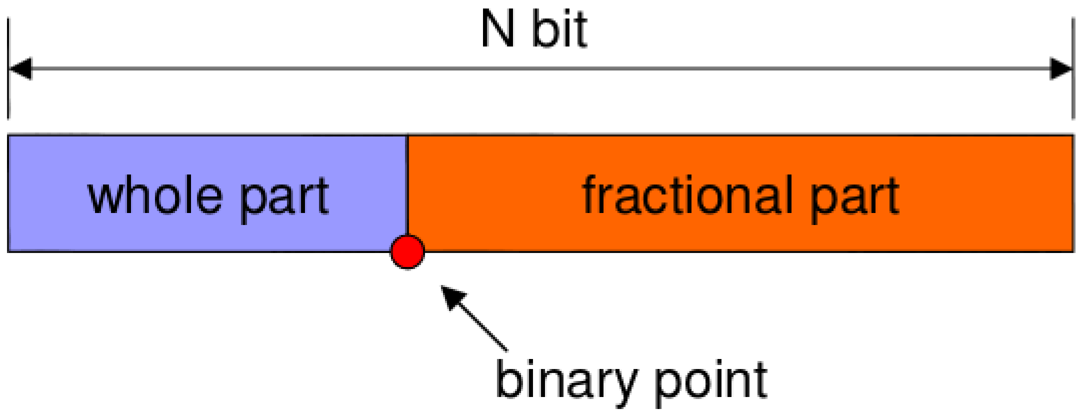
Fixed point

floating point

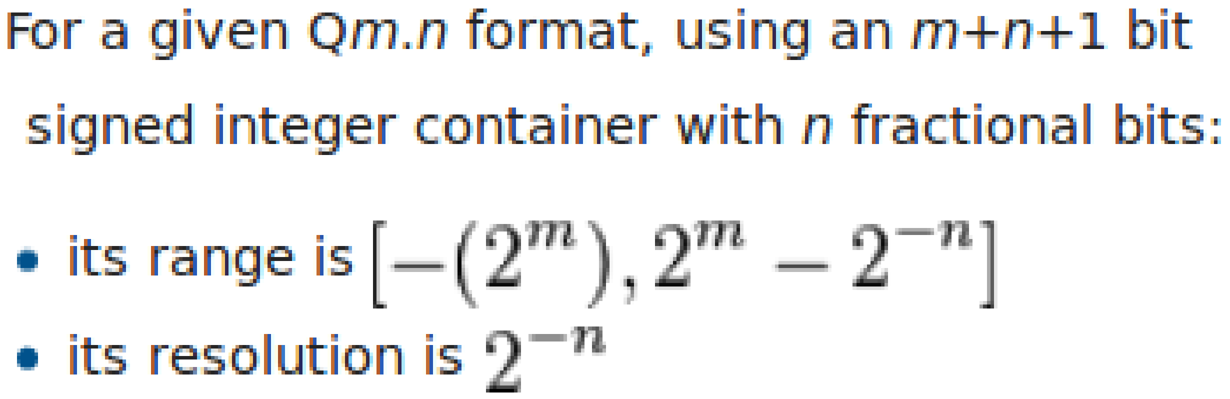
Numeric representation

- Fixed point
- Floating point

Fixed point (Q representation)



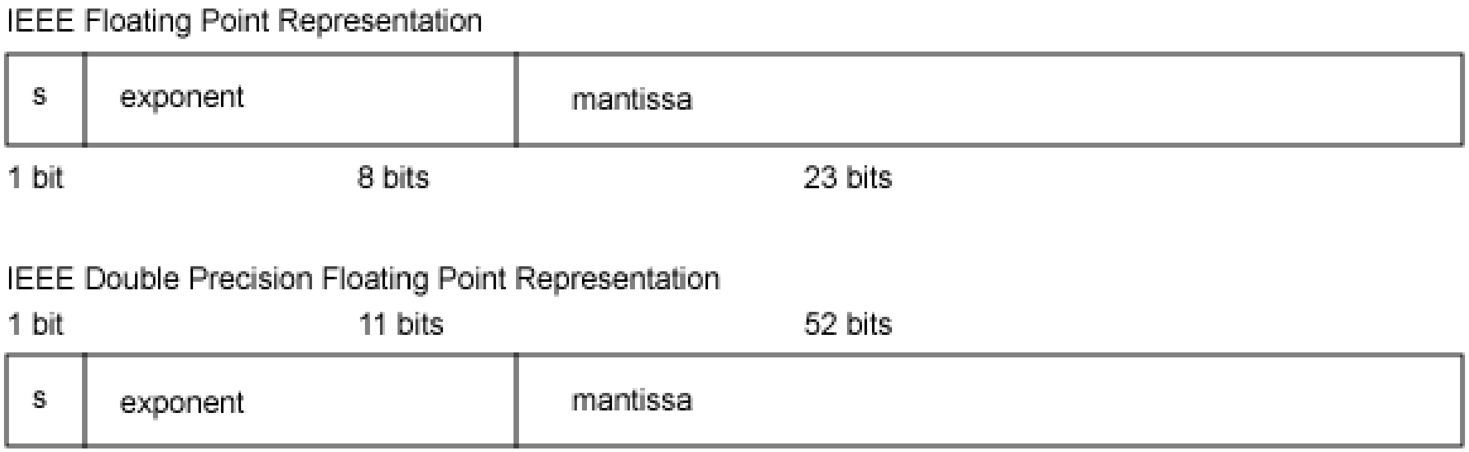
Qm.n format: m bit for whole part, n bit for fractional part.



For a given Q*m*.*n* format, using an *m*+*n*+1 bit signed integer container with *n* fractional bits:

- its range is $[-(2^m), 2^m - 2^{-n}]$
- its resolution is 2^{-n}

Floating point (IEEE 754)



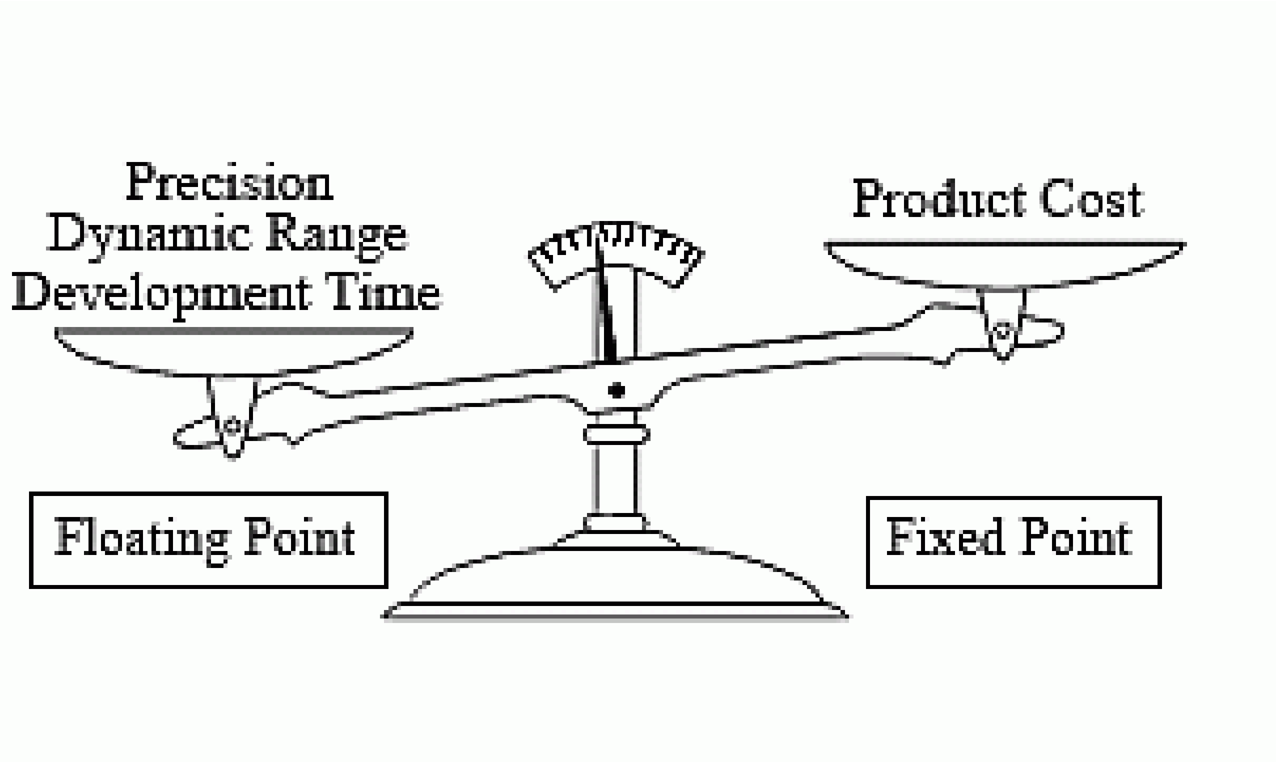
IEEE Floating Point Representation

s	exponent	mantissa
1 bit	8 bits	23 bits

IEEE Double Precision Floating Point Representation

1 bit	11 bits	52 bits
s	exponent	mantissa

Floating vs fixed point



- Fixed point
  - 16 bits registers and buffers
  - Short instruction set
  - Simple architecture

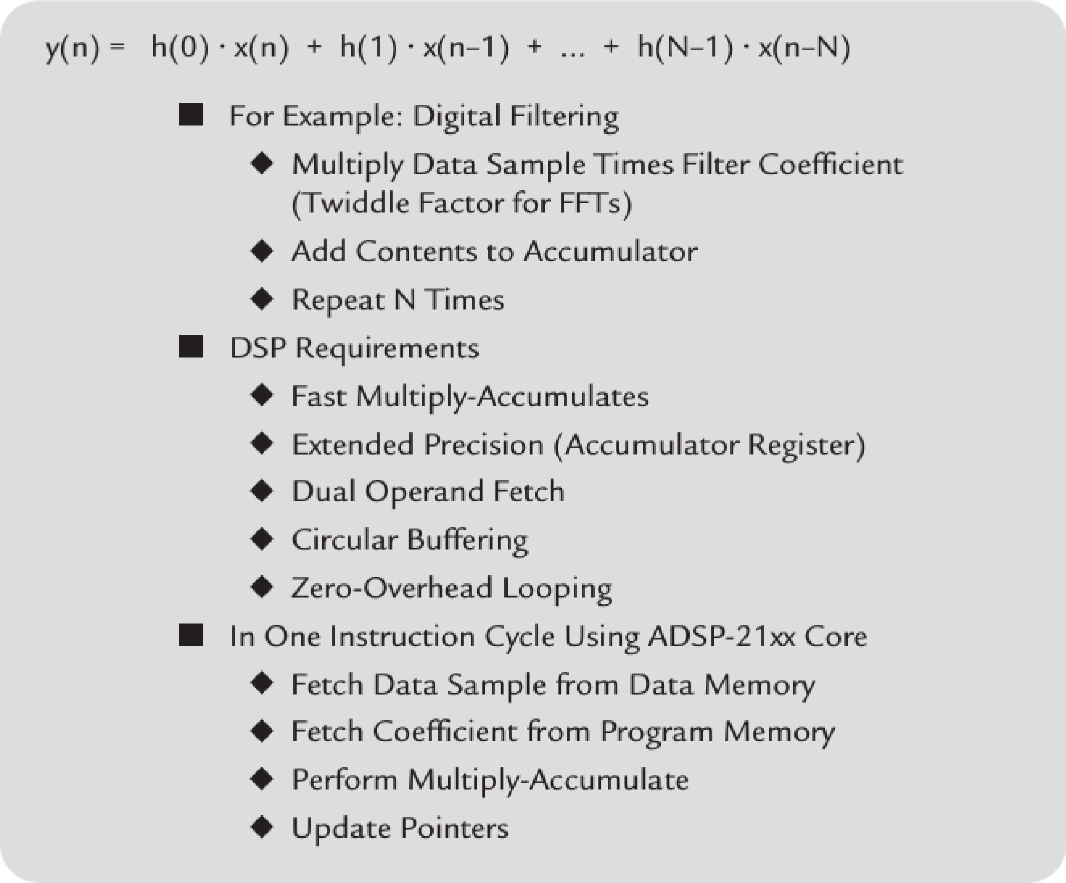
- Floating point
  - 32 bits registers and buffers
  - Multiplier and ALU must be able to quickly perform floating point arithmetic
  - Better precision
  - Shorter development cycle

architecture

- How many multiplies and adds must process a FIR filter of order 7 sampling at 400 KHz?

- DSPs are designed to maximize performance for inner loops containing a sum of products.

Fundamental Mathematical Operation in DSP



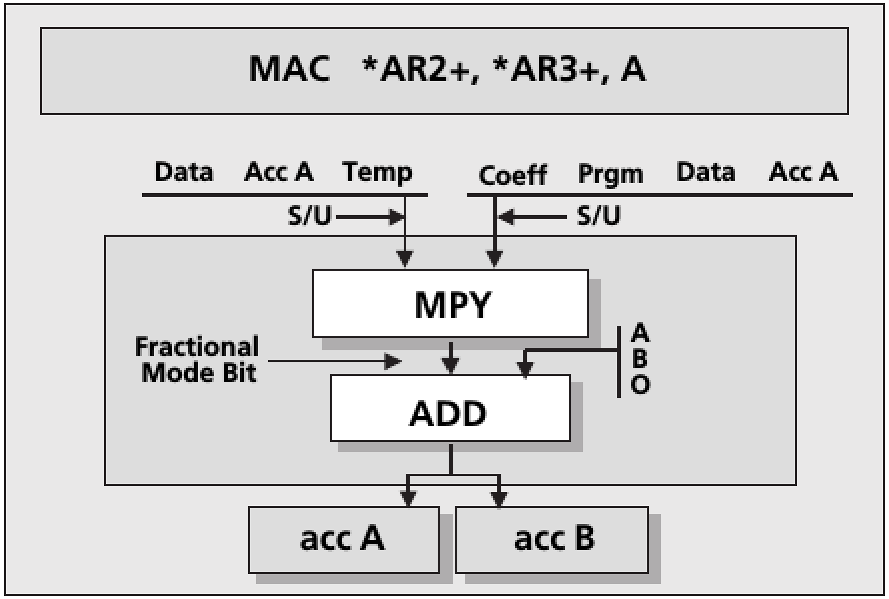
$$y(n) = h(0) \cdot x(n) + h(1) \cdot x(n-1) + \ldots + h(N-1) \cdot x(n-N)$$

- ■ For Example: Digital Filtering
  - ◆ Multiply Data Sample Times Filter Coefficient (Twiddle Factor for FFTs)
  - ◆ Add Contents to Accumulator
  - ◆ Repeat N Times
- ■ DSP Requirements
  - ◆ Fast Multiply-Accumulates
  - ◆ Extended Precision (Accumulator Register)
  - ◆ Dual Operand Fetch
  - ◆ Circular Buffering
  - ◆ Zero-Overhead Looping
- ■ In One Instruction Cycle Using ADSP-21xx Core
  - ◆ Fetch Data Sample from Data Memory
  - ◆ Fetch Coefficient from Program Memory
  - ◆ Perform Multiply-Accumulate
  - ◆ Update Pointers

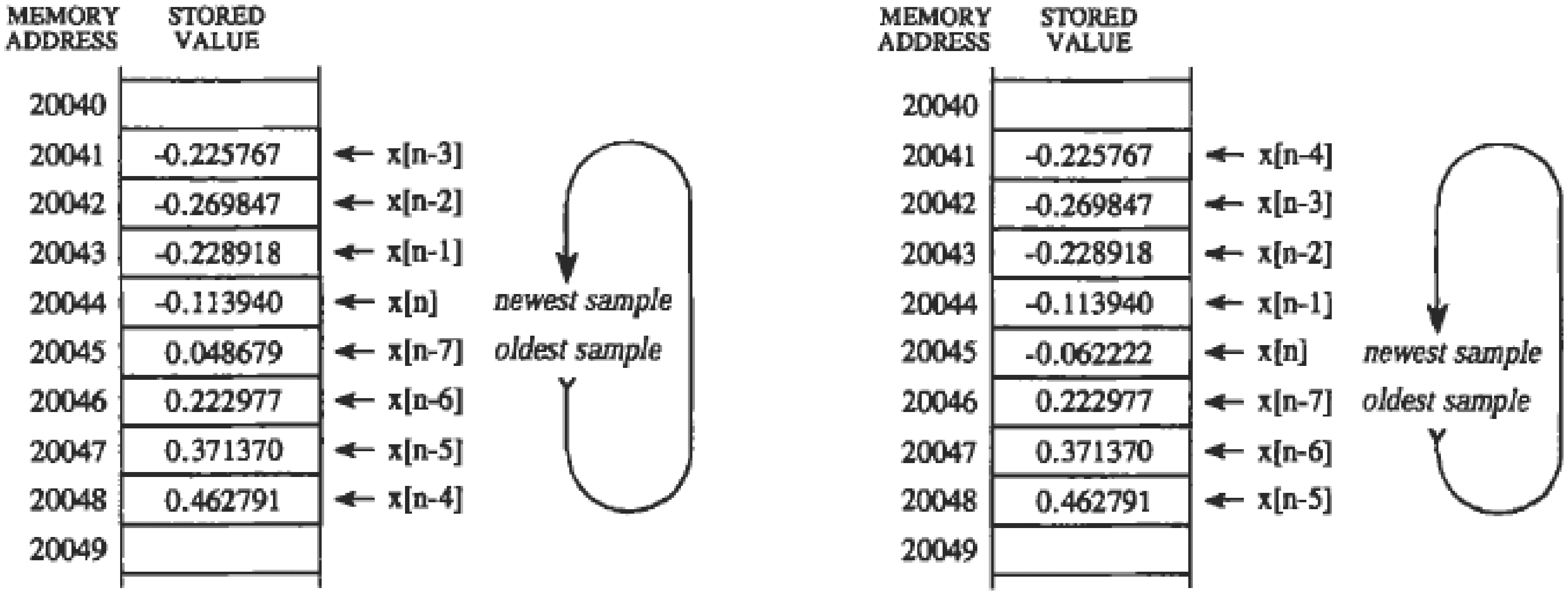
Figure 7-3: The Most Fundamental Mathematical Operation in DSP: The Sum of Products

MAC Unit

- MAC Unit (multiplier - accumulator)



Circular buffering



MEMORY ADDRESS	STORED VALUE	
20040		
20041	-0.225767	← x[n-3]
20042	-0.269847	← x[n-2]
20043	-0.228918	← x[n-1]
20044	-0.113940	← x[n]
20045	0.048679	← x[n-7]
20046	0.222977	← x[n-6]
20047	0.371370	← x[n-5]
20048	0.462791	← x[n-4]
20049		

a. Circular buffer at some instant

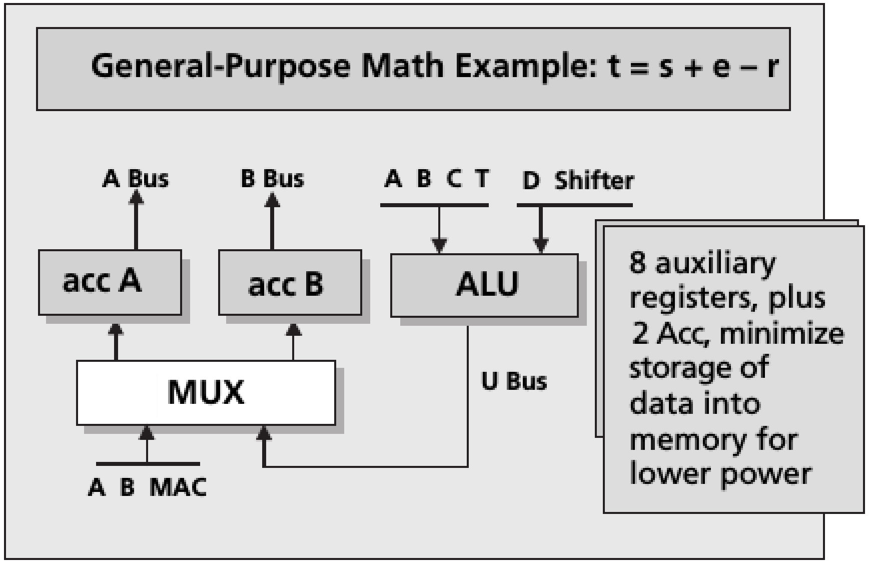
MEMORY ADDRESS	STORED VALUE	
20040		
20041	-0.225767	← x[n-4]
20042	-0.269847	← x[n-3]
20043	-0.228918	← x[n-2]
20044	-0.113940	← x[n-1]
20045	-0.062222	← x[n]
20046	0.222977	← x[n-7]
20047	0.371370	← x[n-6]
20048	0.462791	← x[n-5]
20049		

b. Circular buffer after next sample

Zero overhead looping

Zero overhead loops allow programmers to initialize loops by setting up a count value and defining the loop bounds. The processor will continue to execute this loop until the count has been reached. The looping constructs are implemented in hardware

Parallel ALUs



8 auxiliary registers, plus 2 Acc, minimize storage of data into memory for lower power

High Bandwidth Memory Architectures

- Independent memories allowing multiple operands to be fetched during the same cycle.
- Increased number of I/O ports
- DMA controller

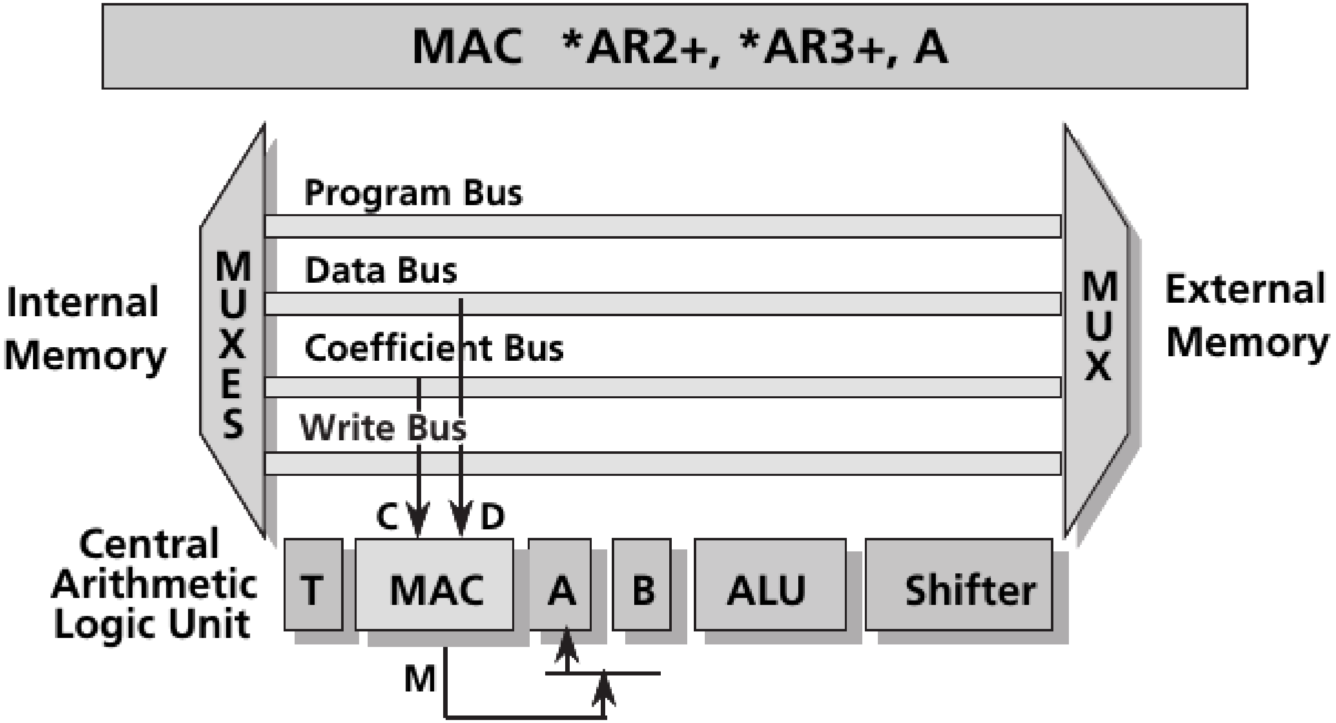


Figure 5.3 DSPs use multiple buses to read and write data (courtesy of Texas Instruments)

For the case of a simple FIR filter, each filter tap requires up to four memory accesses,

Memory Options

- RAM
  - SARAM
  - DARAM

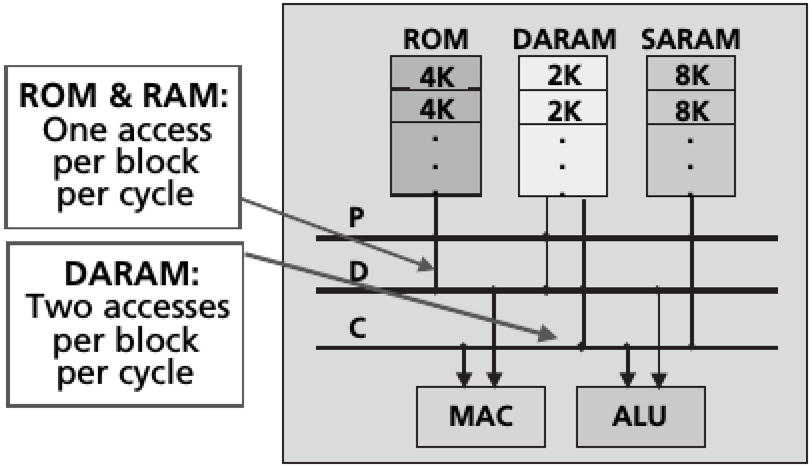


Figure 5.4 Memory options on a DSP (courtesy of Texas Instruments)

- Storing a result back to DARAM memory will not block a subsequent read from the same block of memory.

Caches for DSPs

- Repeat buffer. This cache is used explicitly with a special "repeat" instruction, it will repeat the execution of the next instruction "n" times

DMA (direct memory access)

- Direct memory access is a capability provided by DSP computer bus architectures that allows data to be sent directly from an attached device (such as external memory) to other memory locations. The DSP is freed from involvement with the data transfer, thus speeding up overall computer operation.

DMA steps

- The DMA requests the bus when it is ready to start the transfer.
- The CPU completes any memory transactions currently in operation and grants the bus to the DMA controller.
- The DMA transfers words until the transfer is complete or the CPU requests bus.
- If the DMA grants the bus back before the transfer is complete, it must request it again to finish the transaction.
- When the entire block is transferred, the DMA may optionally notify the CPU via an interrupt.