CLASE 10
ARQUITECTURAS DSP

DSP: Digital Signal Processor

Un DSP no es más que un **microprocesador especializado**. Tanto en su repertorio de instrucciones como en su organización. La tecnología utilizada es la misma que para los procesadores de propósitos generales.

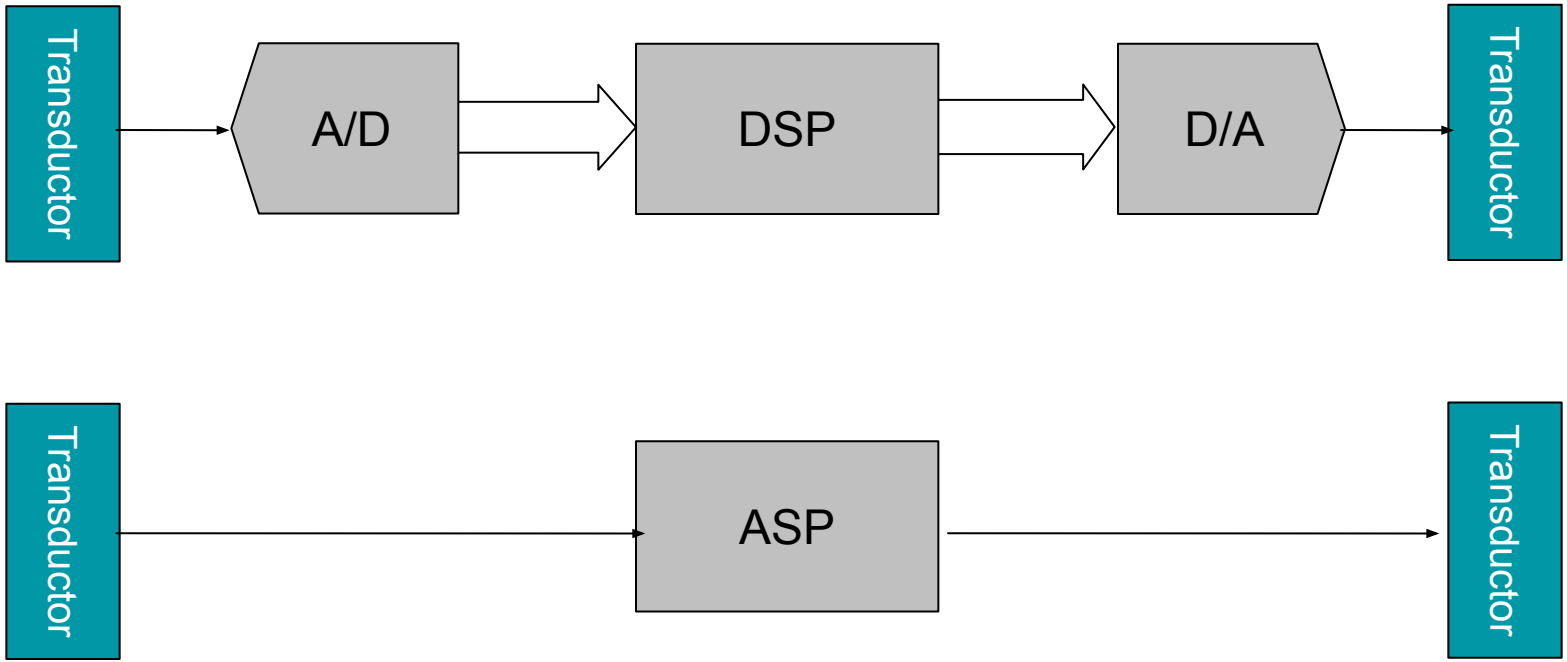
Se trata de una arquitectura optimizada para las necesidades del procesamiento digital de señales. Optimizada tanto en capacidad de cómputo como en eficiencia energética.

Aplicaciones en audio, comunicaciones, radar, reconocimiento de voz, procesamiento de imágenes médicas, televisión digital, industria automotriz, etc.

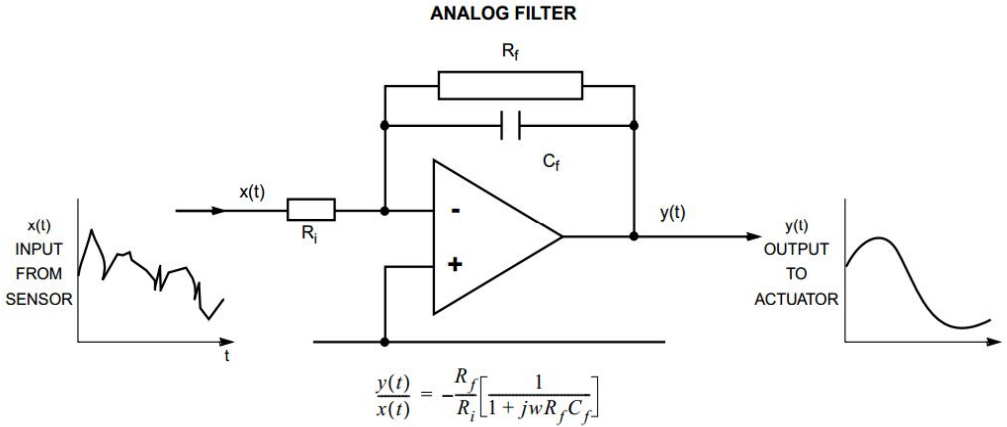
Phil Lapsley et al, DSP Processor Fundamentals, *IEEE Press 1997*
Motorola, DSP56000 User Manual, *1994*

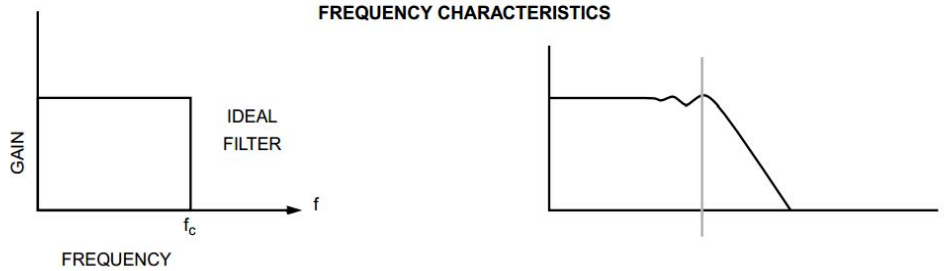
NOTA: Otro ejemplo muy común de arquitectura dedicada son las GPU (Graphics Processing Unit) → ver en Arquitectura Avanzada

Procesamiento de señales: digital vs. analógico

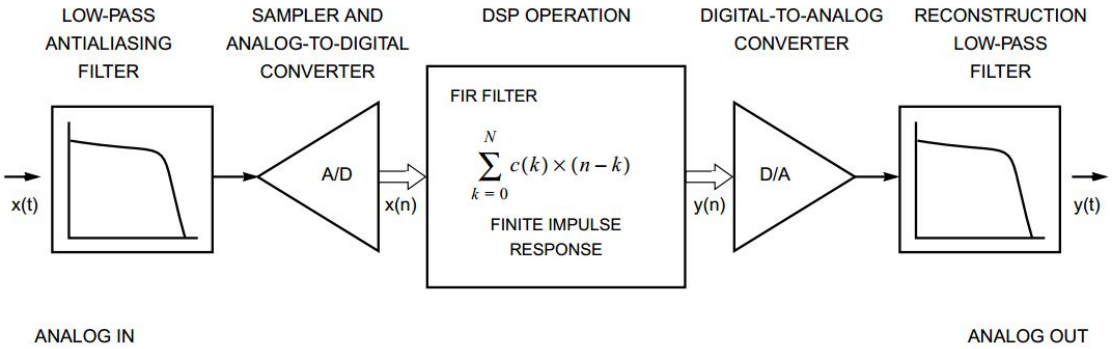


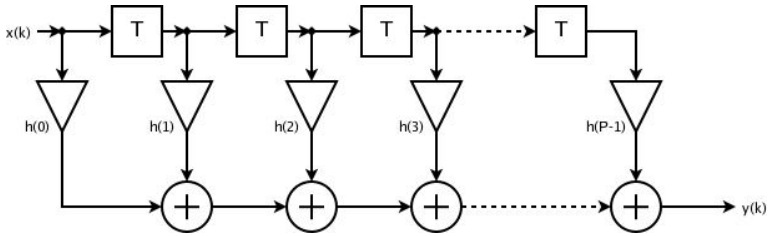
Procesamiento analógico de señales





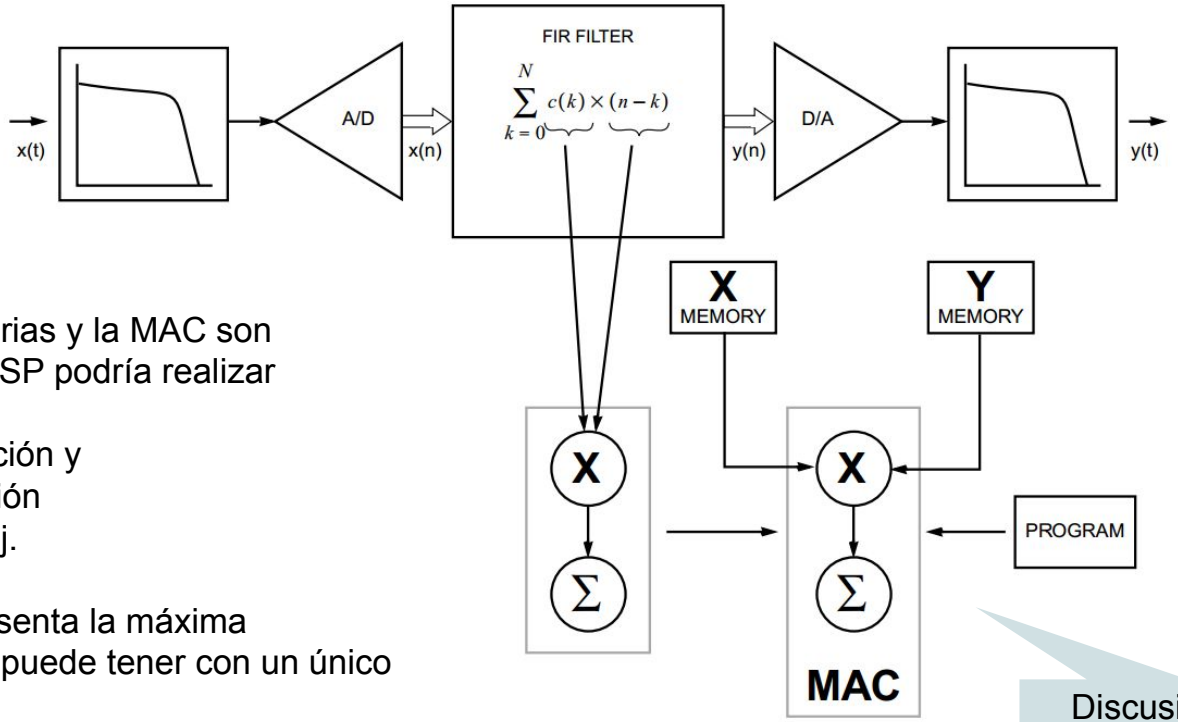
Procesamiento digital de señales





Filtro FIR (respuesta finita al impulso), N es el orden del filtro

Arquitectura Harvard + MAC



Como las dos memorias y la MAC son independientes, el DSP podría realizar

- dos cargas,
- una multiplicación y
- una acumulación

en cada ciclo de reloj.

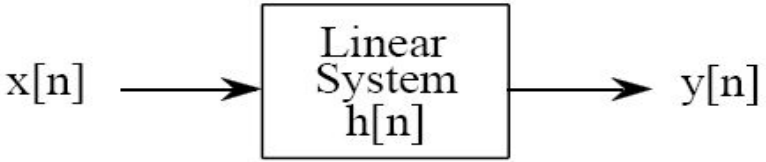
Este esquema representa la máxima performance que se puede tener con un único multiplicador.

ESTRATEGIA: FAVORECER EL CASO MÁS FRECUENTE

Discusión sobre los beneficios y diseño del acumulador

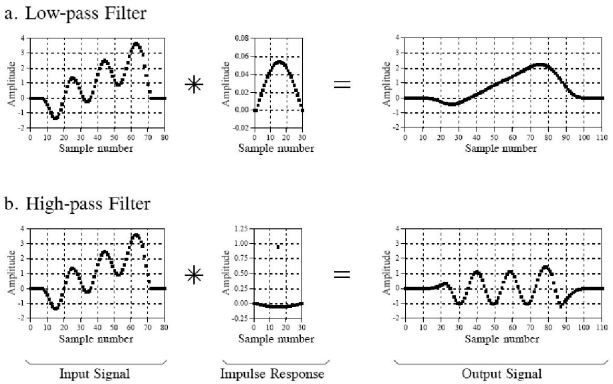
Suma de productos

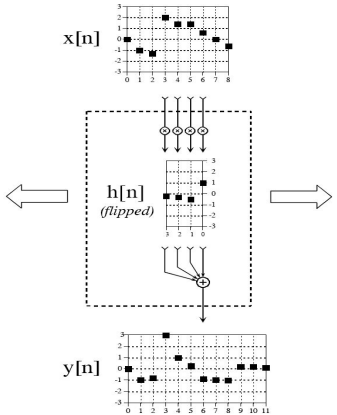
Convolución de secuencias discretas



$$y[i] = \sum_{j=0}^{M-1} h[j]\, x[i-j]$$

$$x[n] * h[n] = y[n]$$





Principales aplicaciones de los DSP

Tipos de aplicaciones

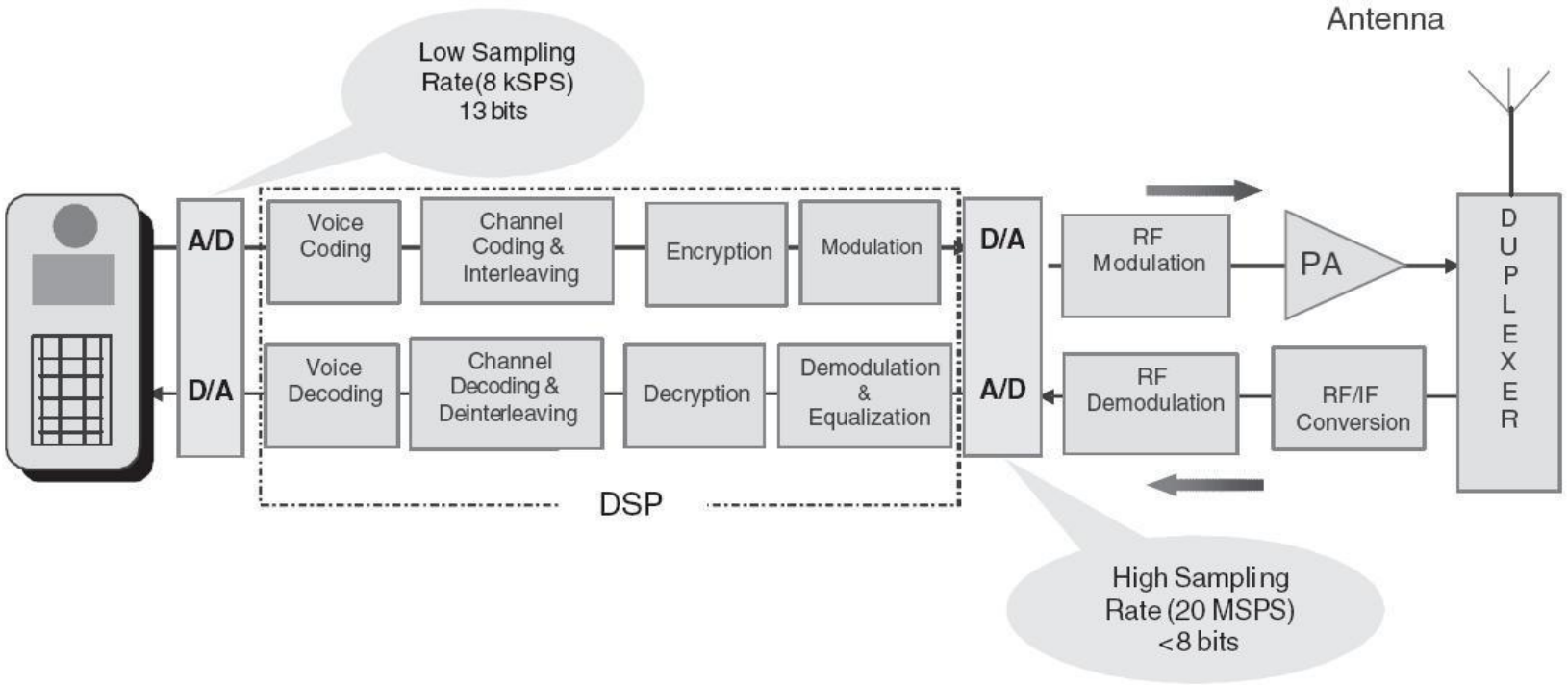
- Embedded de bajo costo (celulares, control de motores HD, automotriz)
- Aplicaciones de alta performance (algoritmos específicos demandantes)
- PC Multimedia

Problemas típicos

- Demanda de procesamiento en tiempo real con altas frecuencias de muestreo.
- Muestreo y generación de señales analógicas.
- Aplicaciones de bajo consumo y pobres niveles de señal.
- Compresión de datos en tiempo real.

Ejemplos

- Machine vision (guia de robots y manejo autónomo), radar, software defined radios (SDR), audio, video codec, biométrica.


Antenna
Low Sampling Rate(8 kSPS) 13 bits
A/D
D/A
Voice Coding
Channel Coding & Interleaving
Encryption
Modulation
Voice Decoding
Channel Decoding & Deinterleaving
Decryption
Demodulation & Equalization
DSP
D/A
A/D
RF Modulation
PA
RF Demodulation
RF/IF Conversion
DUPLEXER
High Sampling Rate (20 MSPS) <8 bits

Consideraciones generales de diseño de la solución

- Tipos de representación numérica:
  punto fijo versus punto flotante: rango y precisión
- Algoritmo a implementar
- Frecuencia de muestreo ***fm*** necesaria

El problema

- Frecuencia de reloj ***fc*** del DSP
- Relación ***fc/fm*** (“cantidad de hardware disponible”)

El dispositivo

Cuántas instrucciones del DSP puedo ejecutar entre dos muestras de la señal

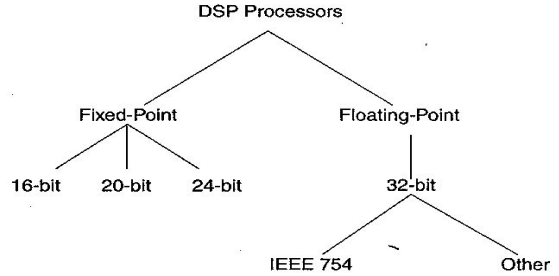
DETALLES DE DISEÑO

1. **Representación numérica** -> Punto fijo [-1,+1] vs. punto flotante.
2. **Aritmética** -> Datapath incluye acumulador, shifter normalizador, saturación y redondeo.
3. **Sistema de memoria** -> Harvard, multiple access.
4. **Acceso a los datos** -> AGU doble (address generation unit) y modos de direccionamiento especiales (circular, bit reversed).
5. **Control** -> Set de instrucciones especializado (MAC, loop, etc.).

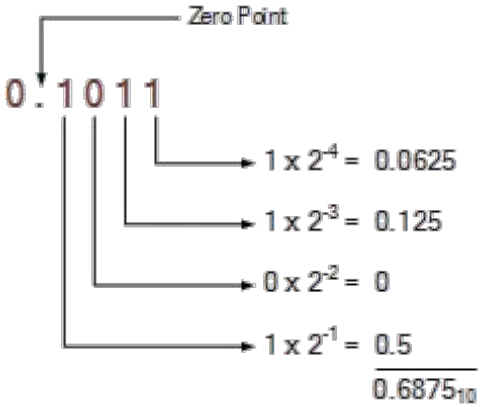
Incluyen **segmentación** profunda, características **superescalares** y **VLIW** (paralelismo)

Compiladores (C)

1. **Representación numérica → punto fijo vs. punto flotante**
2. Aritmética
3. Sistema de memoria
4. Acceso a los datos
5. Estructuras de control



Representación fraccional en punto fijo
Un mismo número en complemento a 2 puede representar un entero o un fraccional. Puede utilizarse la misma ALU para ambos formatos. El resultado se interpreta en forma diferente.



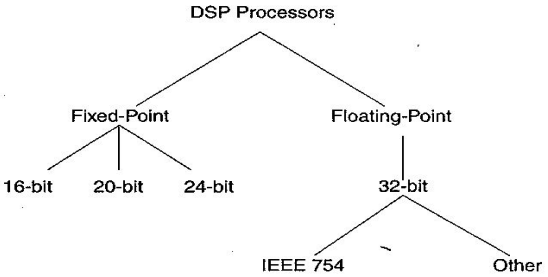
Word Value	Integer Value	Fractional Value
0x8000	-32768	-1.000000
0xA000	-24576	-0.750000
0xC000	-16384	-0.500000
0xE000	-8192	-0.250000
0x0000	0	0.000000
0x2000	8192	0.250000
0x4000	16384	0.500000
0x6000	24576	0.750000
0x7FFF	32767	0.999969

Para N = 16 bits, $2^{N-1} = 32768$

Los fraccionales son siempre menores a la unidad, por lo tanto el producto no puede tener **overflow** (solo puede aparecer en las sumas). Limitar el overflow es importante por la fase del resultado.

Los fraccionales se multiplican como si fueran enteros. Se necesitan 2n bits en el acumulador para expresar el producto de dos números de n bits. Además se requieren algunos bits adicionales si se quiere prevenir el overflow en las sumas:
DSP56000: registros de 24 bits – Acumulador de 56 bits (24+24+8)
DSP96002: registros de 32 bits – Acumulador de 96 bits (32+32+32)

1. **Representación numérica → punto fijo vs. punto flotante**
2. Aritmética
3. Sistema de memoria
4. Acceso a los datos
5. Estructuras de control



RANGO DINÁMICO

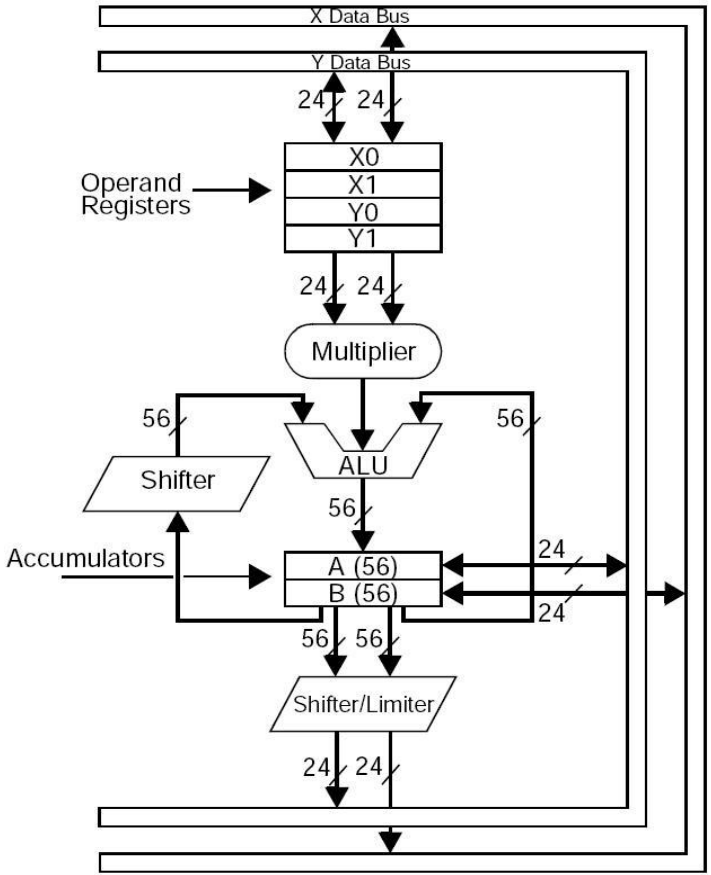
Es la relación entre el mayor y el menor número de una representación. Supongamos que la longitud de palabra es de 32 bits.

En representación de **punto fijo**, el número más pequeño es 2^{-31} y el número más grande es $1-2^{-31}$. El cociente es aproximadamente 2.15 e+9, lo que representa unos **187 db.**

En representación de **punto flotante** (24 bits de mantisa y 8 de exponente), el número más chico es 5.88 e-39 y el mayor 3.40 e+38, lo que da un rango dinámico de 5.79 e+76, unos **1535 db**.

La necesidad de rango dinámico está impuesta por la aplicación. En telecomunicaciones suele alcanzar con 50db. Audio de alta fidelidad requiere unos 90 db. Radar/IoT 120 db.

1. Representación numérica
2. **Aritmética → Datapath con acumulador, shifter normalizador, saturación y redondeo**
3. Sistema de memoria
4. Acceso a los datos
5. Estructuras de control

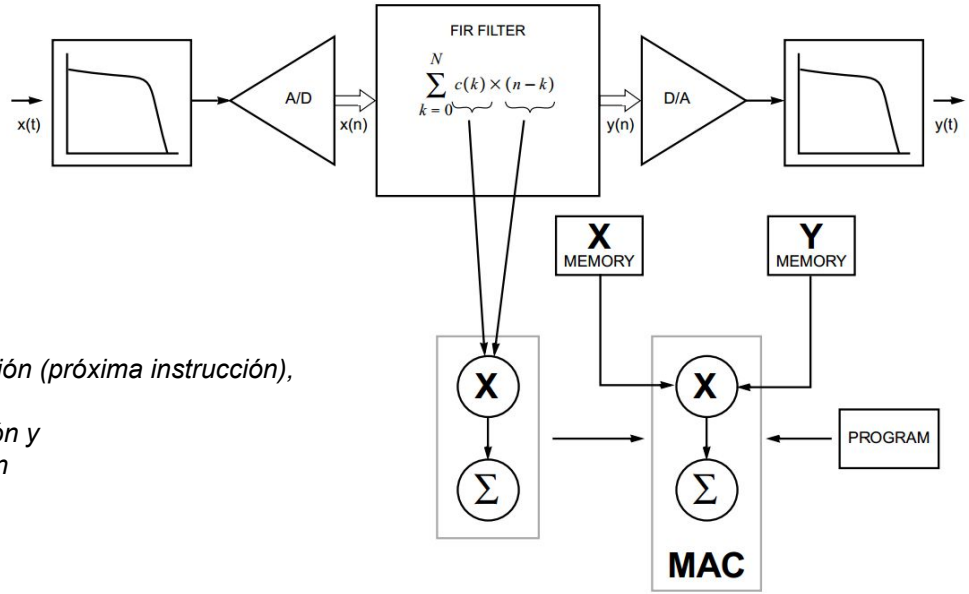


DSP56000

Registros y buses de 24 bits
Acumulador de 56 bits (24+24+8)

Datapath incluye acumulador, shifter normalizador, saturación y redondeo.

1. Representación numérica
2. Aritmética
3. **Sistema de memoria → Harvard, multiple access**
4. Acceso a los datos
5. Estructuras de control

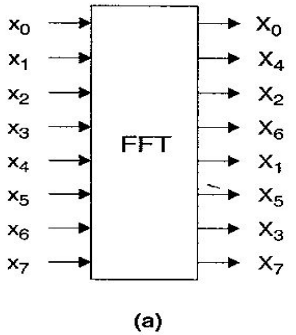


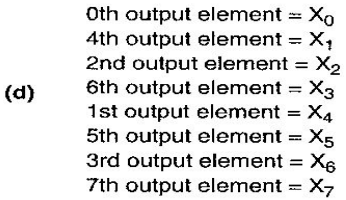
En cada ciclo de reloj:
- *Una decodificación (próxima instrucción),*
- *dos cargas,*
- *una multiplicación y*
- *una acumulación*

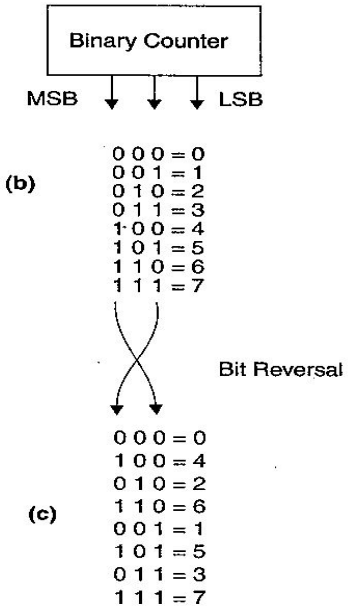
Segmentación

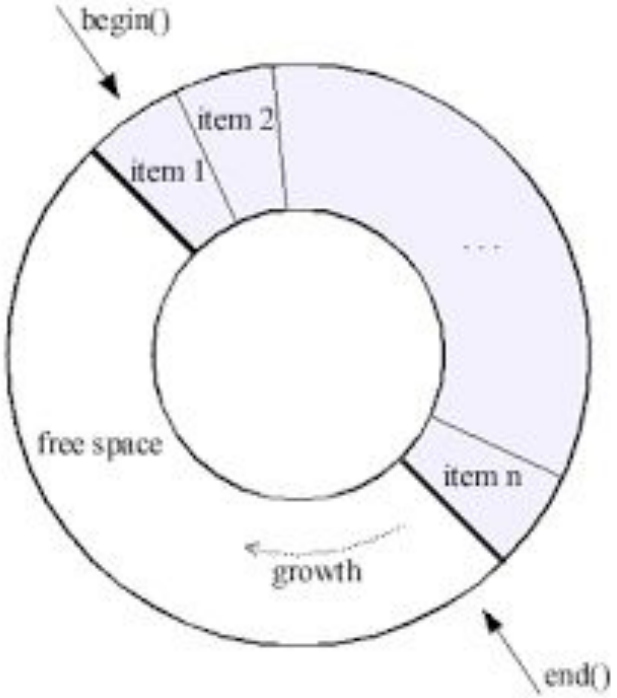
1. Representación numérica
2. Aritmética
3. Sistema de memoria
4. **Acceso a los datos →**
5. Estructuras de control

- ➢ Modos de direccionamiento indirecto vía registro con post decremento e incremento: **`MAC (R0)+,(R4)+,A`**
- ➢ Buffers circulares. Direccionamiento “módulo n”
- ➢ Direccionamiento “bit reversal” (FFT ver)
- ➢ AGU (address generation unit) doble









1. Representación numérica
2. Aritmética
3. Sistema de memoria
4. Acceso a los datos
5. **Estructuras de control → LOOP, MAC, etc.**

```
        MOVE    #16,B
LOOP:   MAC     (R0)+,(R4)+,A
        DEC     B
        JNE     LOOP
```

(a)

```
        RPT     #16
        MAC     (R0)+,(R4)+,A
```

(b)

16 ciclos

(c)

```
            addi r10,r0,16          ; contador
loop:       load r11,(r1)           ; vector de entrada
            load r12,(r2)           ; fir
            mul r13,r11,r12         ; multiplicar
            add r5,r5,r13           ; acumular
            addi r1,r1,1      ; incrementar puntero vector
            addi r2,r2,1      ; incrementar puntero fir
            addi r10,r10,-1         ; decrementar contador
            jnz r10,loop      ; repetir
```

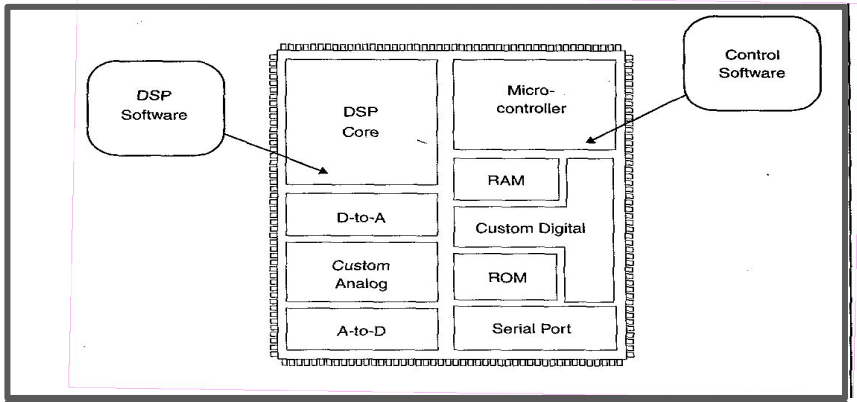
RISC (0,3)

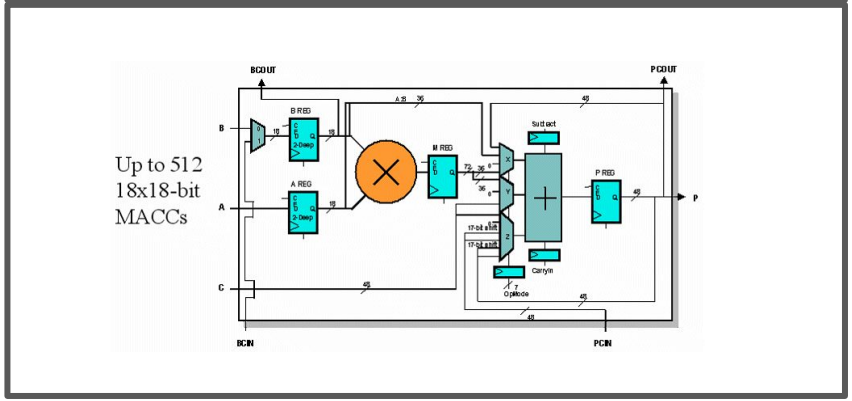
Arquitecturas híbridas

- Digital Signal Controller (DSC): Microcontrolador + DSP
  Microchip dsPIC33CH
  https://microchipdeveloper.com/16bit:ch-overview

- FPGA con módulos MAC
  Xilinx Virtex 4
  https://www.xilinx.com/products/technology/dsp.html

- Microcontroladores con instrucción MAC y saturación
  ARM Cortex-M4





EJEMPLO: Microchip dsPIC

https://microchipdeveloper.com/dsp0201:start

“The dsPIC® Central Processing Unit, or CPU, seamlessly integrates the best features of a 16-bit microcontroller (MCU) and digital signal processor (DSP). “

Caracteríasticas VLIW

```
ADD X0,A          Y0,X:(R1)+N      ; One DSP56800 Instruction
```

Opcode And Operands

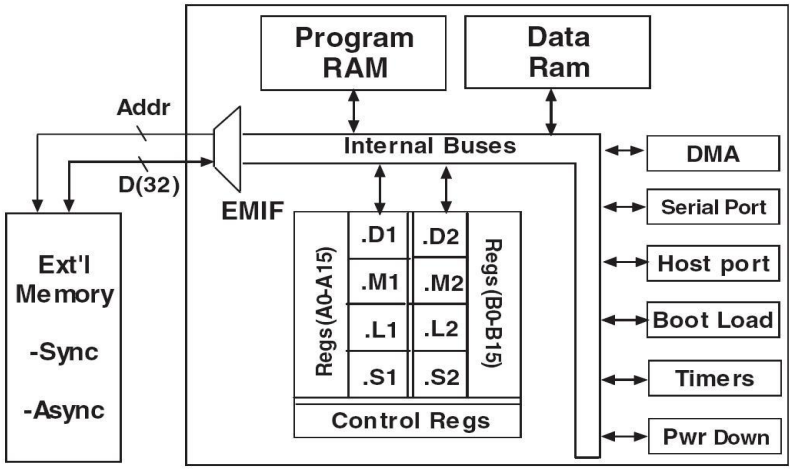
Single Parallel Move
(Uses XAB1 and CGDB)

```
MACR X0,Y0,A      X:(R0)+N,Y0      X:(R3)-,X0
```

Opcode and Operands

Primary Read
(Uses XAB1 and CGDB)

Secondary Read
(Uses XAB2 and XDB2)



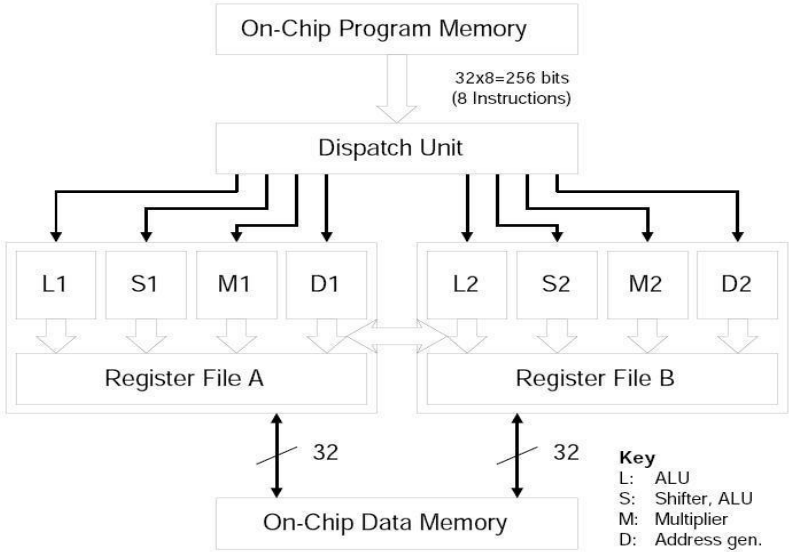
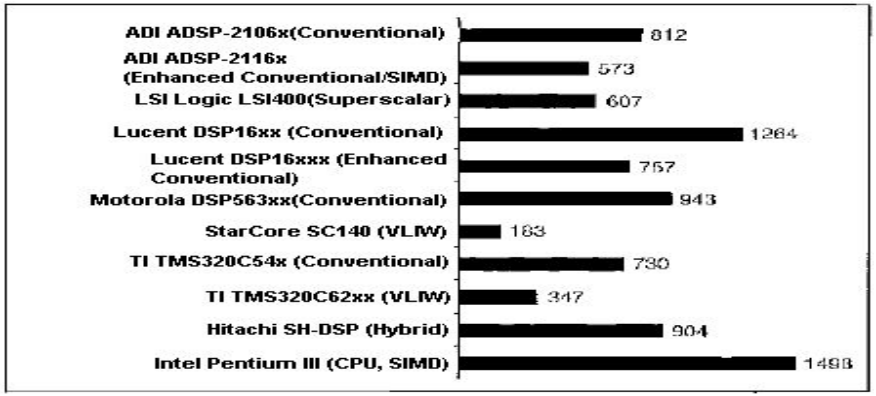
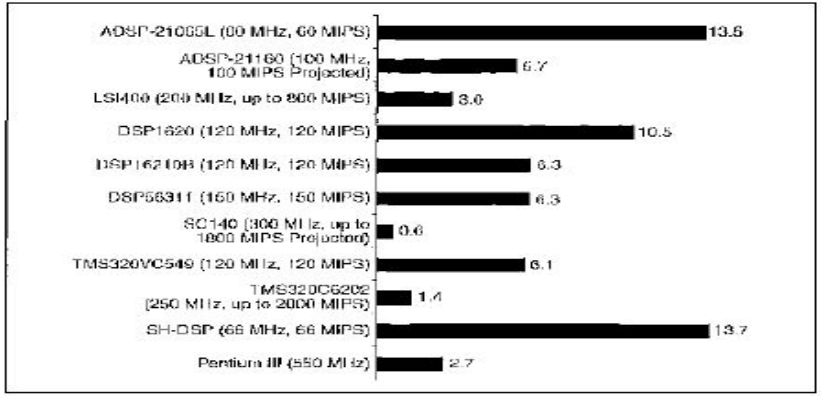


Figure 3. TMS320C62xx execution units and memory architecture. The TMS320C62xx has eight execution units, grouped in two sets of four.

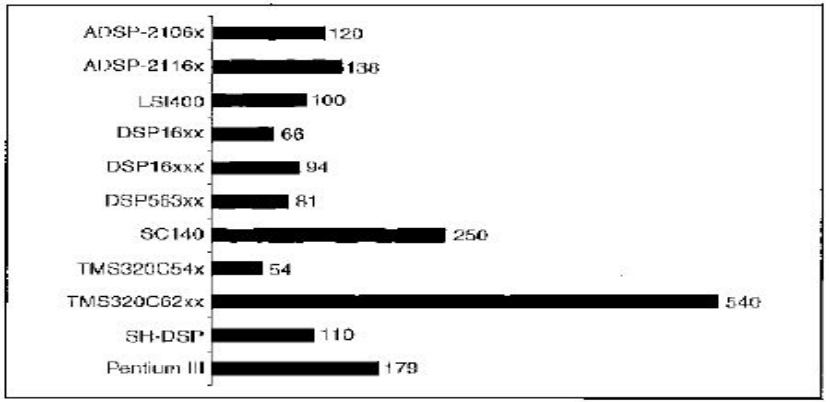
Medición de performance: FIR



Cycle counts for FIR filter



Execution times for FIR filter



Program memory usage for FIR filter

Ver también BTDImark2000 y CoreMark:
http://www.bdti.com/Resources/BenchmarkResults/BDTImark2000
https://www.eembc.org/coremark/

ANEXO: SHARC

Super Harvard ARChitecture
AD 32-Bit Floating-Point Processors

"Super" Harvard architecture extends the original concepts of separate program and data memory busses by adding an I/O processor with its associated dedicated busses.

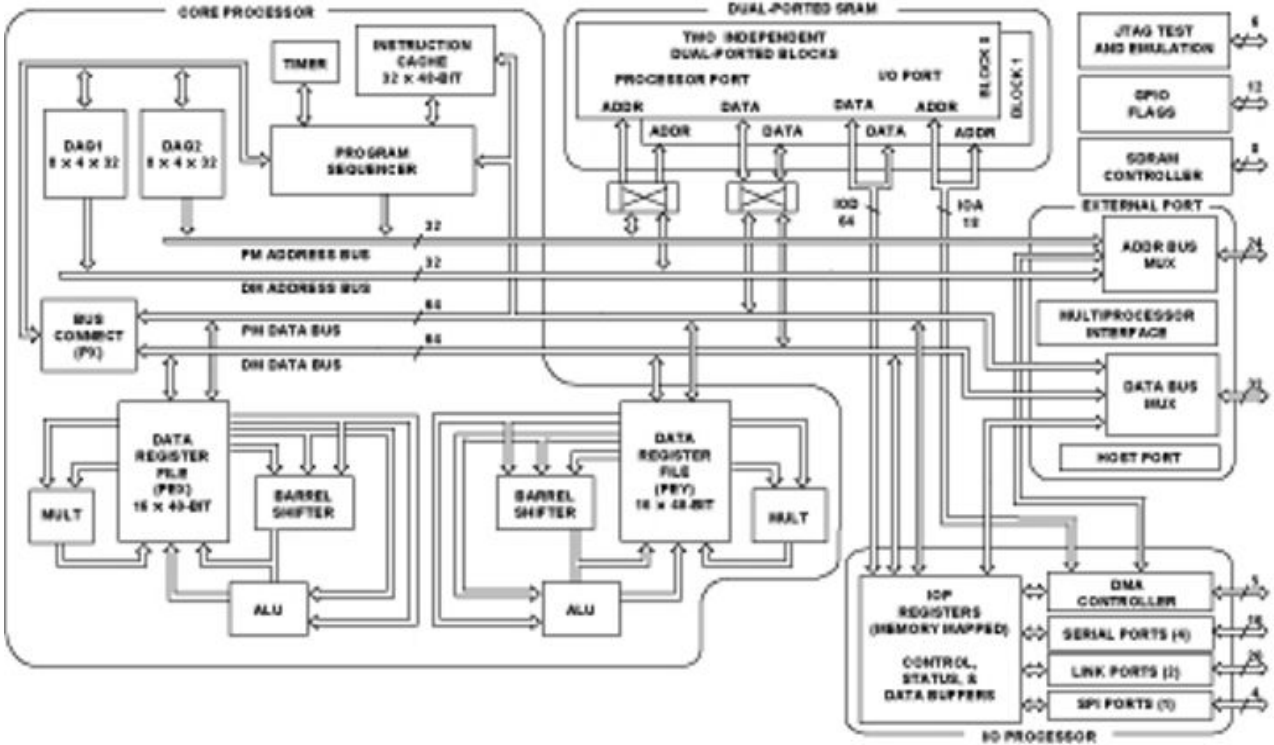
In addition to satisfying the demands of the most computationally intensive, real-time signal-processing applications, SHARC processors integrate large memory arrays and application-specific peripherals designed to simplify product development and reduce time to market.

https://www.analog.com/media/en/news-marketing-collateral/product-highlight/SHARC_Proc_Family_(C)_Final.pdf

Características de la familia

- 32/40-bit IEEE floating-point math
- 32-bit fixed-point multipliers with 64-bit product & 80-bit accumulation
- no arithmetic pipeline; all computations are single-cycle
- circular buffer addressing supported in hardware
- 32 address pointers support 32 circular buffers
- six nested levels of zero-overhead looping in hardware
- rich, algebraic assembly language syntax
- instruction set supports conditional arithmetic, bit manipulation, divide & square root, bit field deposit and extract
- dma allows zero-overhead background transfers at full clock rate without processor intervention

Diagrama funcional



Performance

First Generation	1994	66 MHz		198 MFLOPs	3
Second Generation SIMD		100 MHz		600 MFLOPs	6
Third Generation		450 MHz	900 MMACs	2700 MFLOPs	6
Fourth Generation	Actual	450 MHz	900 MMACs	2700 MFLOPs	6

Kits de desarrollo para el AD-21262 disponibles
PROYECTO FINAL

	ADSP-2116	ADSP-21261	ADSP-21262	ADSP-21371	ADSP-21364	ADSP-21368	ADSP-2146x
Clock Cycle	100 MHz	150 MHz	200 MHz	266 MHz	333 MHz	400 MHz	450 MHz
Instruction Cycle Time	10 ns	6.67 ns	5 ns	3.75 ns	3 ns	2.5 ns	2.22 ns
MFLOPS Sustained	400 MFLOPS	600 MFLOPS	800 MFLOPS	1064 MFLOPS	1332 MFLOPS	1600 MFLOPS	1800 MFLOPS
MFLOPS Peak	600 MFLOPS	900 MFLOPS	1200 MFLOPS	1596 MFLOPS	1998 MFLOPS	2400 MFLOPS	2700 MFLOPS
1024 Point Complex FFT	92 µs	61.3 µs	46 µs	34.5 µs	28 µs	23 us	20.44 µs
FIR Filter (per tap)	5 ns	3.3 ns	2.5 ns	1.88 ns	1.5 ns	1.25 ns	1.11 ns
IIR Filter (per biquad)	20 ns	13.3 ns	10 ns	7.5 ns	6 ns	5 ns	4.43 ns
Matrix Multiply (pipelined)	45 ns 80 ns	30 ns 53.3 ns	22.5 ns 40 ns	16.91 ns 30.07 ns	13.5 ns 24 ns	11.25 ns 20 ns	10.00 ns 17.78 ns
Divide (y/x)	30 ns	20 ns	15 ns	11.27 ns	9 ns	7.5 ns	6.67 ns
Inverse Square Root	45 ns	30 ns	22.5 ns	16.91 ns	13.5 ns	11.25 ns	10.00 ns

ISA

- The architecture knows nothing of 8-bit or 16-bit values since each address is used to point to a whole 32-bit word, not just a octet. It is thus neither little-endian nor big-endian, though a compiler may use either convention if it implements 64-bit data and/or some way to pack multiple 8-bit or 16-bit values into a single 32-bit word. Analog Devices chose to avoid the issue by using a 32-bit char in their C compiler.
- The word size is 48-bit for instructions, 32-bit for integers and normal floating-point, and 40-bit for extended floating-point.
- SHARC instructions may contain a 32-bit immediate operand. Instructions without this operand are generally able to perform two or more operations simultaneously.
- Many instructions are conditional, and may be preceded with "if condition " in the assembly language.
- The SHARC has a 32-bit word-addressed address space.
- There are two delay slots. After a jump, two instructions following the jump will normally be executed.
- The SHARC processor has built-in support for loop control. Up to 6 levels may be used, avoiding the need for normal branching instructions and the normal bookkeeping related to loop exit.
- The SHARC has two full sets of general-purpose registers. Code can instantly switch between them, allowing for fast context switches between an application and an OS or between two threads.

TigerSHARC

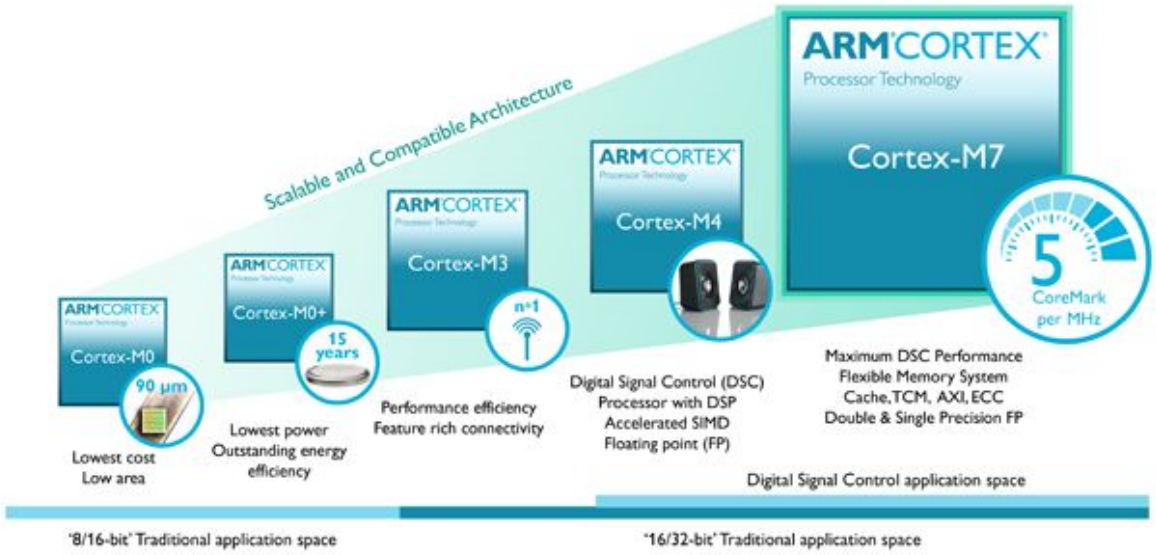
As a static superscalar DSP, the TigerSHARC Processor core can execute simultaneously from one to four 32-bit instructions encoded in a single instruction line.

With a few exceptions, an instruction line, whether it contains one, two, three or **four 32-bit instructions**, executes with a throughput of one cycle in an eight-deep processor pipeline.

ANEXO: Cortex M

Procesadores ARM para aplicaciones embebidas

Son una familia de MCU (Micro-Controller Unit). No tienen MMU (Memory Management Unit), necesaria para correr un sistema operativo de propósitos generales (Android en Cortex A).



https://en.wikipedia.org/wiki/ARM_architecture#32-bit_architecture

ISA

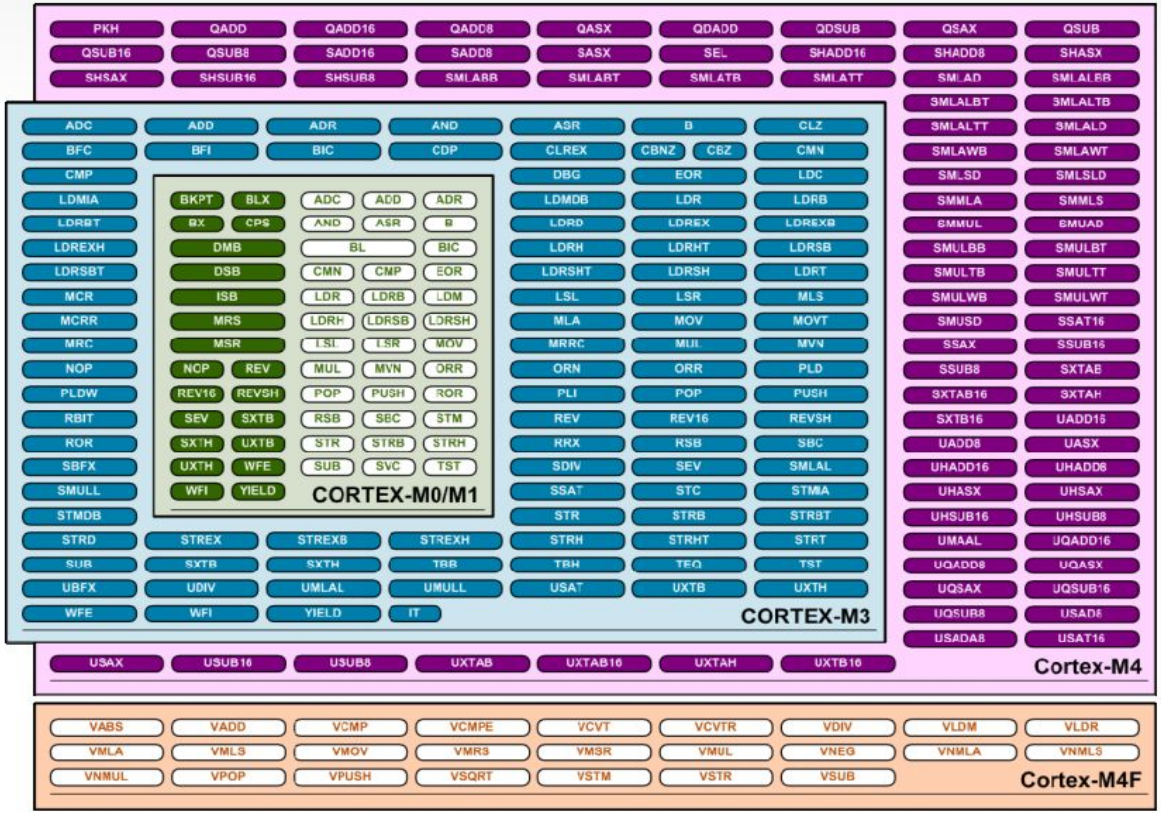
ARM Cortex-M Instruction Sets[6][7]

ARM Cortex-M	Thumb	Thumb-2	Hardware multiply	Hardware divide	Saturated math	DSP extensions	Floating-Point Unit (FPU)	ARM architecture
Cortex-M0[1]	Most	Subset	1 or 32 cycle	No	No	No	No	ARMv6-M
Cortex-M0+[2]	Most	Subset	1 or 32 cycle	No	No	No	No	ARMv6-M
Cortex-M1[3]	Most	Subset	3 or 33 cycle	No	No	No	No	ARMv6-M
Cortex-M3[4]	Entire	Entire	1 cycle	Yes	Yes	No	No	ARMv7-M
Cortex-M4[5]	Entire	Entire	1 cycle	Yes	Yes	Yes	Optional, SP	ARMv7E-M
Cortex-M7	Entire	Entire	1 cycle	Yes	Yes	Yes	Yes, SP & DP	ARMv7E-M

M4 DSP: Harvard architecture, Single cycle MAC, Floating Point, Barrel shifter, SIMD, Aritmética con saturación
M4 no-DSP: Circular and bit-reversed addressing, Zero overhead loops, Load and store operations in parallel with math operations

BENCHMARK: Decodifica MP3 con menos de 10 MHz de clock (un DSP común requiere 15 MHz y uno especializado de audio 5 MHz).

ARM Cortex-M Family Instruction Set



ARM Cortex-M4 DSP Instructions Compared

		Cycle counts		
CLASS	INSTRUCTION	ARM9E-S	CORTEX-M3	Cortex-M4
Arithmetic	ALU operation (not PC)	1 - 2	1	1
	ALU operation to PC	3 - 4	3	3
	CLZ	1	1	1
	QADD, QDADD, QSUB, QDSUB	1 - 2	n/a	1
	QADD8, QADD16, QSUB8, QSUB16	n/a	n/a	1
	QDADD, QDSUB	n/a	n/a	1
	QASX, QSAX, SASX, SSAX	n/a	n/a	1
	SHASX, SHSAX, UHASX, UHSAX	n/a	n/a	1
	SADD8, SADD16, SSUB8, SSUB16	n/a	n/a	1
	SHADD8, SHADD16, SHSUB8, SHSUB16	n/a	n/a	1
	UQADD8, UQADD16, UQSUB8, UQSUB16	n/a	n/a	1
	UHADD8, UHADD16, UHSUB8, UHSUB16	n/a	n/a	1
	UADD8, UADD16, USUB8, USUB16	n/a	n/a	1
	UQASX, UQSAX, USAX, UASX	n/a	n/a	1
	UXTAB, UXTAB16, UXTAH	n/a	n/a	1
	USAD8, USADA8	n/a	n/a	1
Multiplication	MUL, MLA	2 - 3	1 - 2	1
	MULS, MLAS	4	1 - 2	1
	SMULL, UMULL, SMLAL, UMLAL	3 - 4	5 - 7	1
	SMULBB, SMULBT, SMULTB, SMULTT	1 - 2	n/a	1
	SMLABB, SMLBT, SMLATB, SMLATT	1 - 2	n/a	1
	SMULWB, SMULWT, SMLAWB, SMLAWT	1 - 2	n/a	1
	SMLALBB, SMLALBT, SMLALTB, SMLALTT	2 - 3	n/a	1
	SMLAD, SMLADX, SMLALD, SMLALDX	n/a	n/a	1
	SMLSD, SMLSDX	n/a	n/a	1
	SMLSLD, SMLSLD	n/a	n/a	1
	SMMLA, SMMLAR, SMMLS, SMMLSR	n/a	n/a	1
	SMMUL, SMMULR	n/a	n/a	1
	SMUAD, SMUADX, SMUSD, SMUSDX	n/a	n/a	1
	UMAAL	n/a	n/a	1
Division	SDIV, UDIV	n/a	2 - 12	2 - 12

Single cycle MAC

ARM Cortex-M4 Single Cycle MAC Instructions

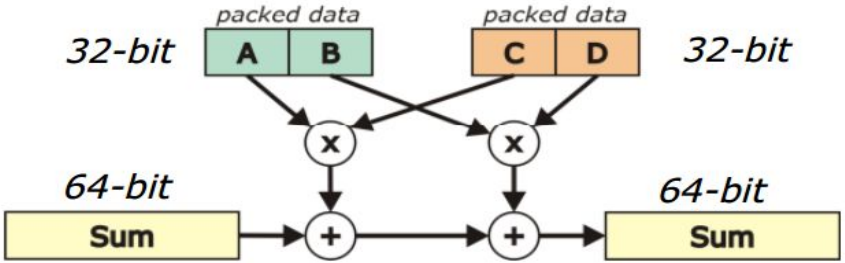
OPERATION	INSTRUCTIONS
16 x 16 = 32	SMULBB, SMULBT, SMULTB, SMULTT
16 x 16 + 32 = 32	SMLABB, SMLABT, SMLATB, SMLATT
16 x 16 + 64 = 64	SMLALBB, SMLALBT, SMLALTB, SMLALTT
16 x 32 = 32	SMULWB, SMULWT
(16 x 32) + 32 = 32	SMLAWB, SMLAWT
(16 x 16) ± (16 x 16) = 32	SMUAD, SMUADX, SMUSD, SMUSDX
(16 x 16) ± (16 x 16) + 32 = 32	SMLAD, SMLADX, SMLSD, SMLSDX
(16 x 16) ± (16 x 16) + 64 = 64	SMLALD, SMLALDX, SMLSLD, SMLSLDX
32 x 32 = 32	MUL
32 ± (32 x 32) = 32	MLA, MLS
32 x 32 = 64	SMULL, UMULL
(32 x 32) + 64 = 64	SMLAL, UMLAL
(32 x 32) + 32 + 32 = 64	UMAAL
32 ± (32 x 32) = 32 (upper)	SMMLA, SMMLAR, SMMLS, SMMLSR
(32 x 32) = 32 (upper)	SMMUL, SMMULR

All the above operations are single cycle on the Cortex-M4 processor

ARM Cortex-M4 SIMD Instructions

SIMD extensions perform multiple operations in one cycle

$$Sum = Sum + (A \times C) + (B \times D)$$



SIMD techniques operate with packed data

Performance

ARM Cortex	CoreMark/MHz	DMIPS/MHz
M7	5.04	2.14 / 2.55 / 3.23 DMIPS/MHz**
M4	3.40	1.25 / 1.52 / 1.91 DMIPS/MHz**
M3	3.32	1.25 / 1.50 / 1.89 DMIPS/MHz**
M0	2.33	0.87 / 1.02 / 1.27 DMIPS/MHz**

*** The first result abides by all of the "ground rules" laid out in the Dhrystone documentation, the second permits inlining of functions, not just the permitted C string libraries, while the third additionally permits simultaneous ("multi-file") compilation. All are with the original (K&R) v2.1 of Dhrystone*

Interesante…
M3 y M4 tienen la misma performance (Dhrystone)

https://www.eembc.org/coremark/
http://en.wikipedia.org/wiki/Coremark
http://en.wikipedia.org/wiki/Dhrystone

http://infocenter.arm.com/help/topic/com.arm.doc.dai0350a/DAI0350A_coremark_benchmarking.pdf