Eng. Julian S. Bruno

REAL TIME DIGITAL SIGNAL PROCESSING

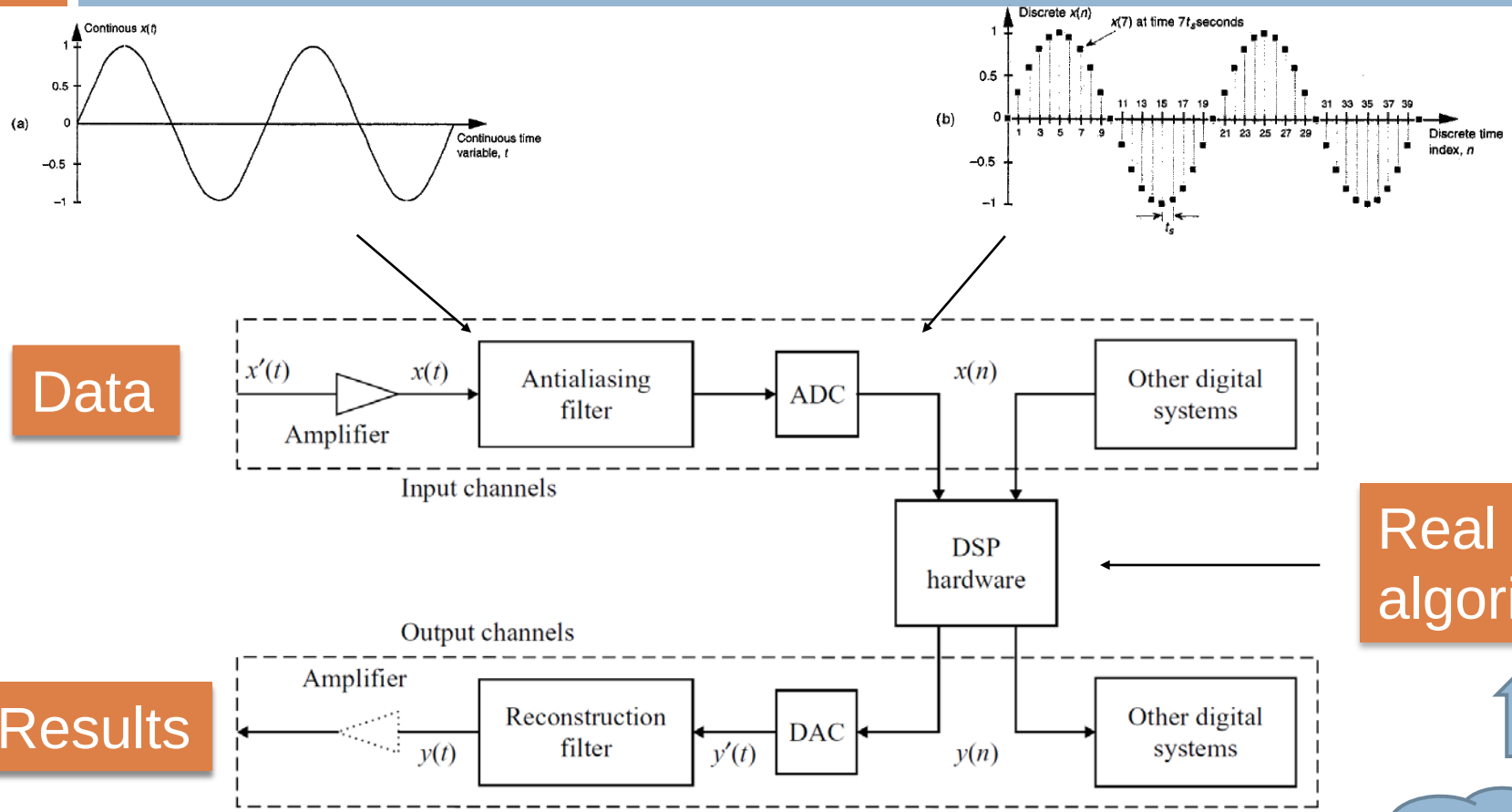
Introduction

Why Digital? A brief comparison with analog.

Advantages

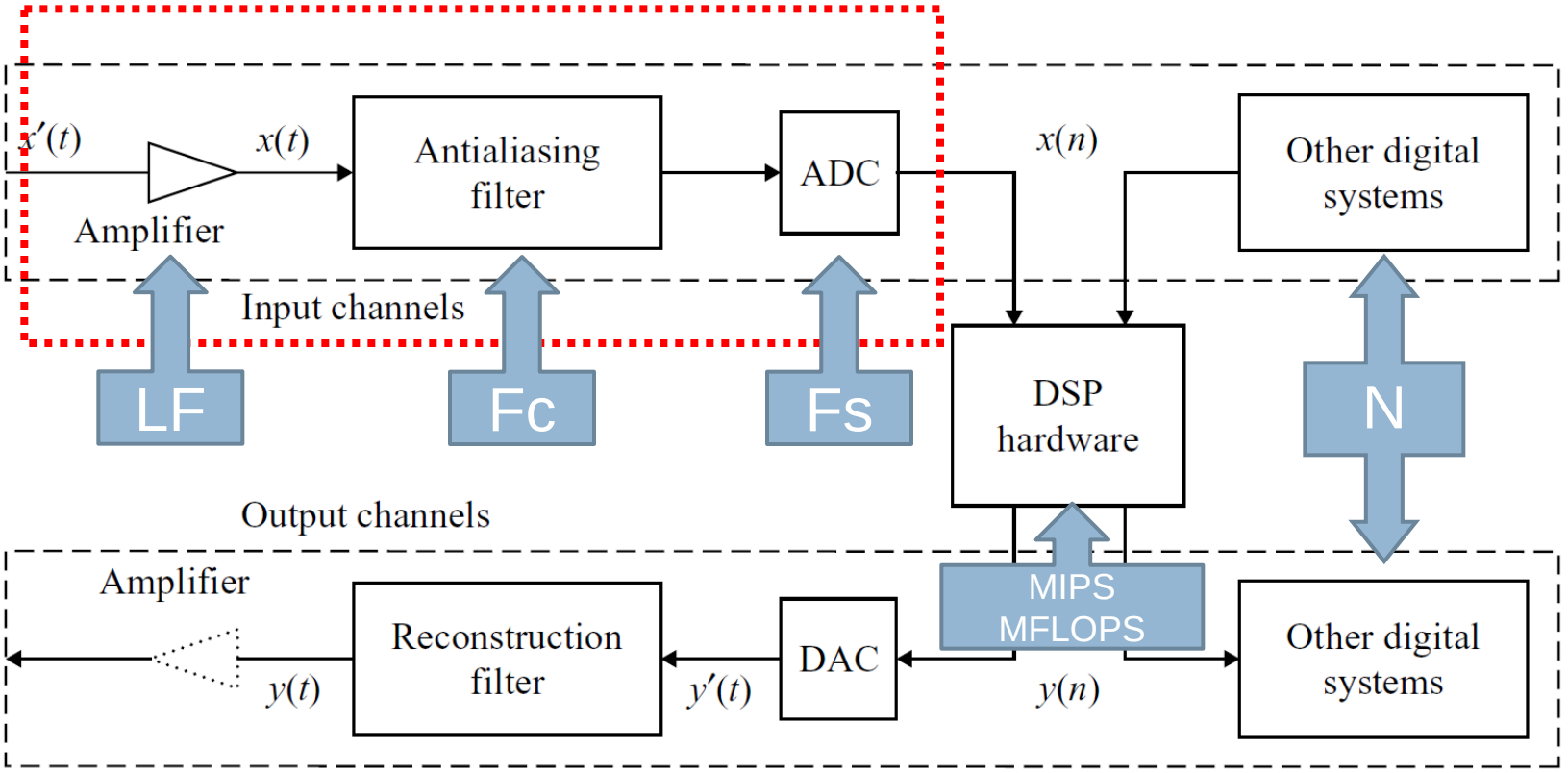
- ***Flexibility***. Easily modifiable and upgradeable.
- ***Reproducibility***. Don’t depend on components tolerance. Exactly reproduced from one unit to other.
- ***Reliability***. No age or environmental drift.
- ***Complexity***. Allows sophisticated applications in only one chip.

The BIG picture



Real Time DSP System

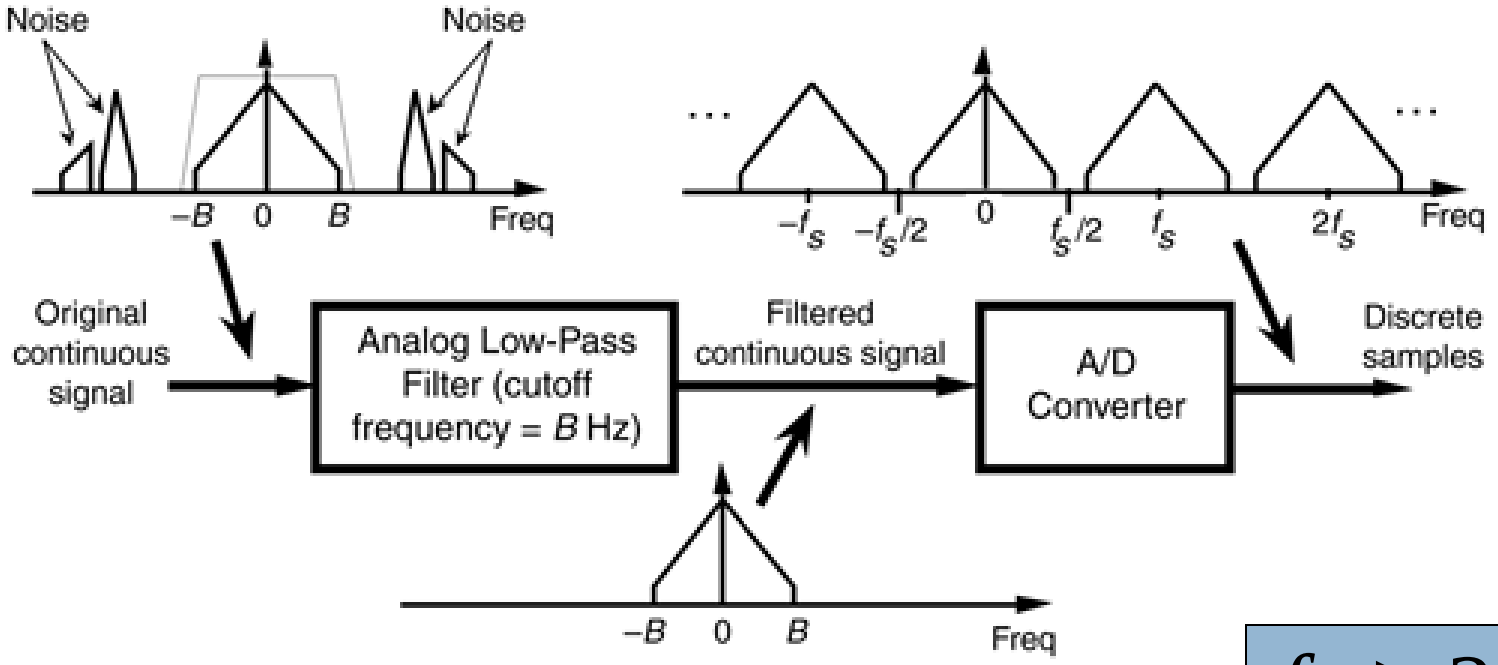
Sampling signals: A very important first step.



Real Time DSP System

Sampling low-pass signals (CT)

The sampling theorem indicates that a continuous signal can be properly sampled, only if it does not contain frequency components above one-half of the sampling rate.

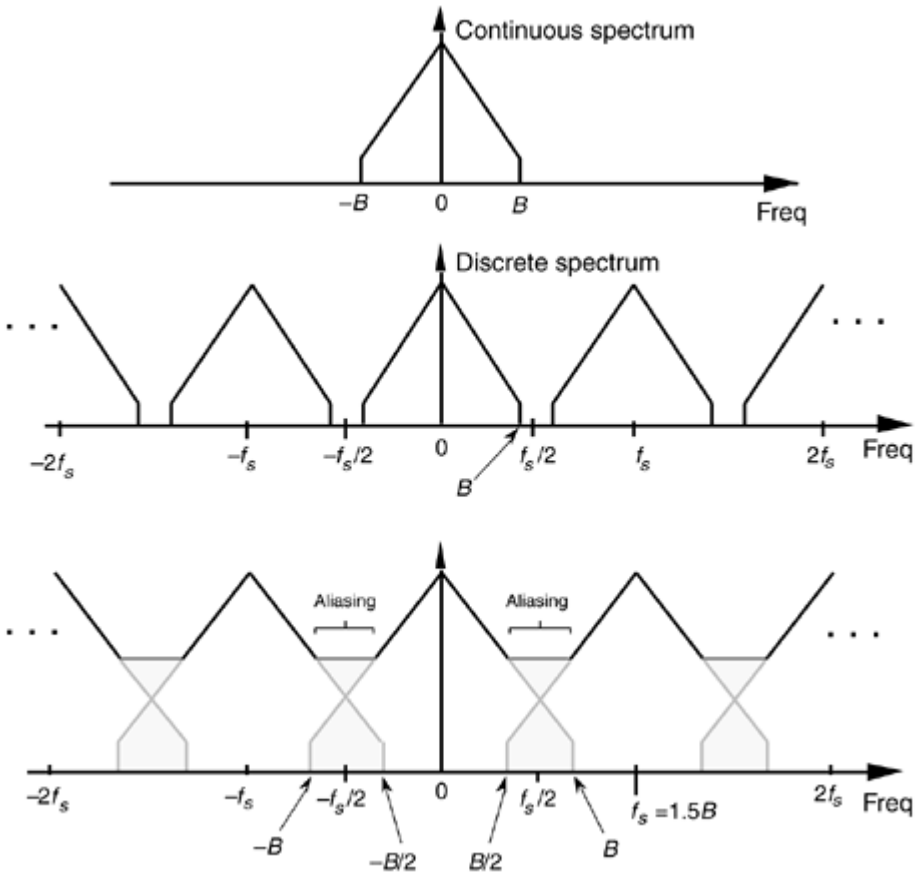


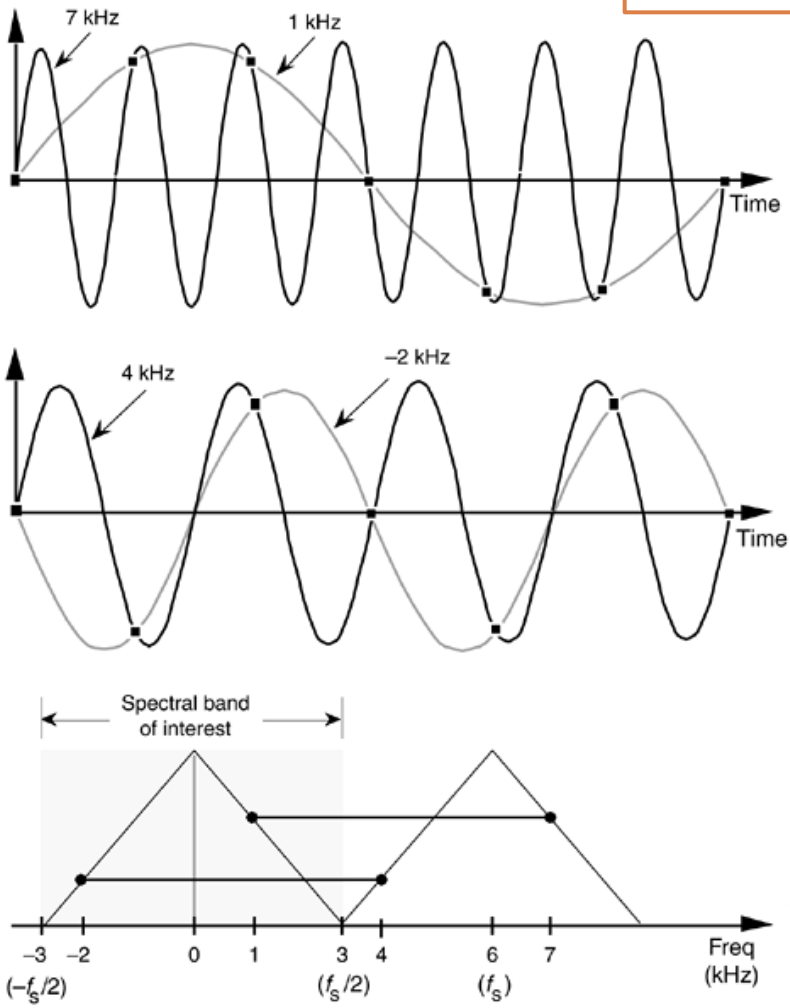
Nyquist sampling theorem

$$f_S \geq 2 f_N$$

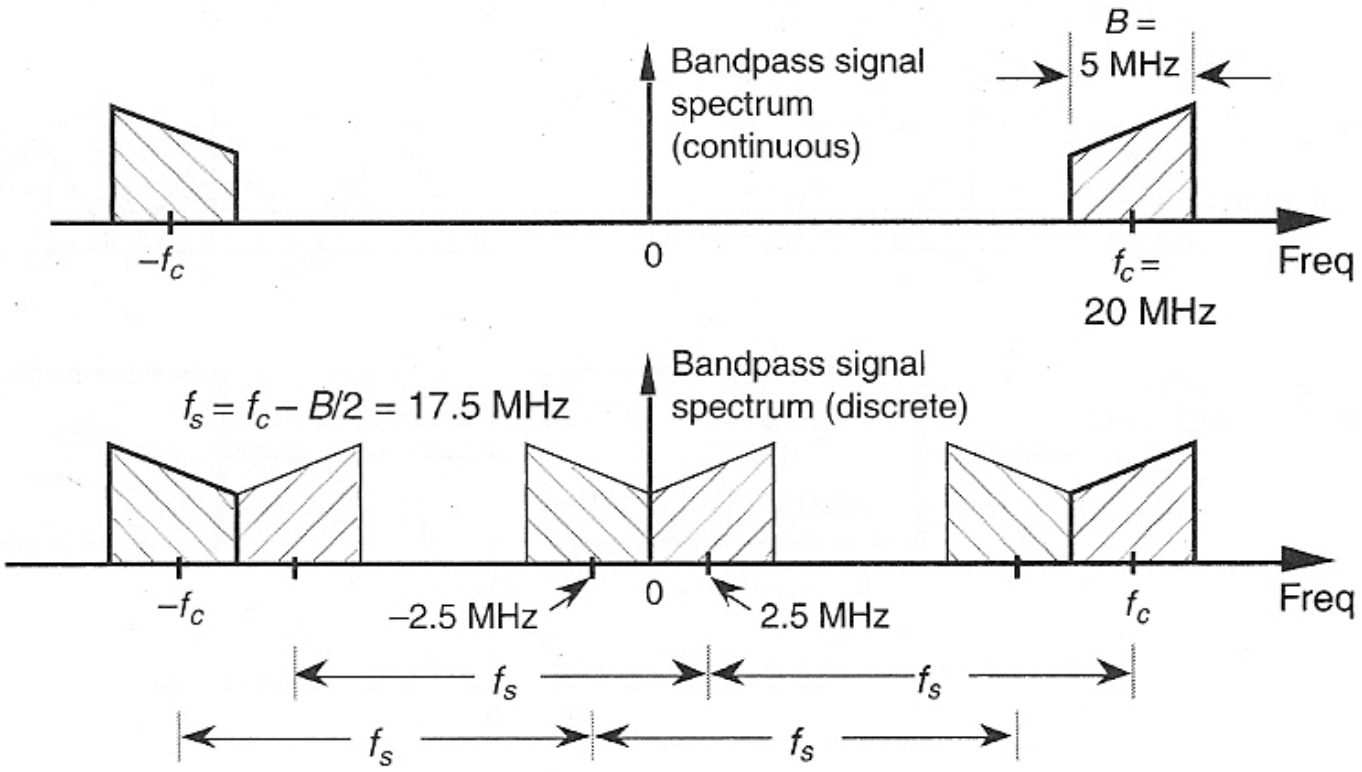
Aliasing and frequency ambiguity

Fs = 6KHz





Sampling band-pass signals



- IF sampling
- Harmonic sampling
- Sub-Nyquist sampling
- Undersampling

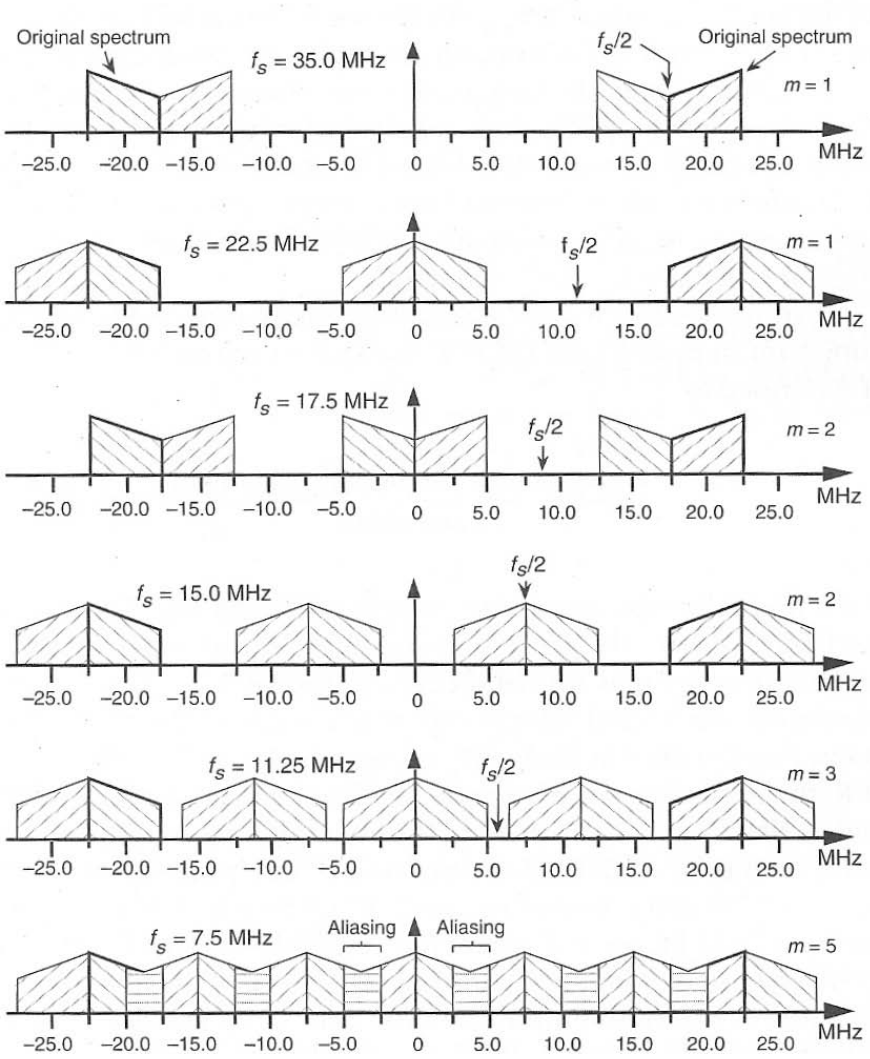
$$\frac{2f_c - B}{m} \geq f_s \geq \frac{2f_c + B}{m+1}$$

for any positive integer m, where fs ≥ 2B is accomplished.

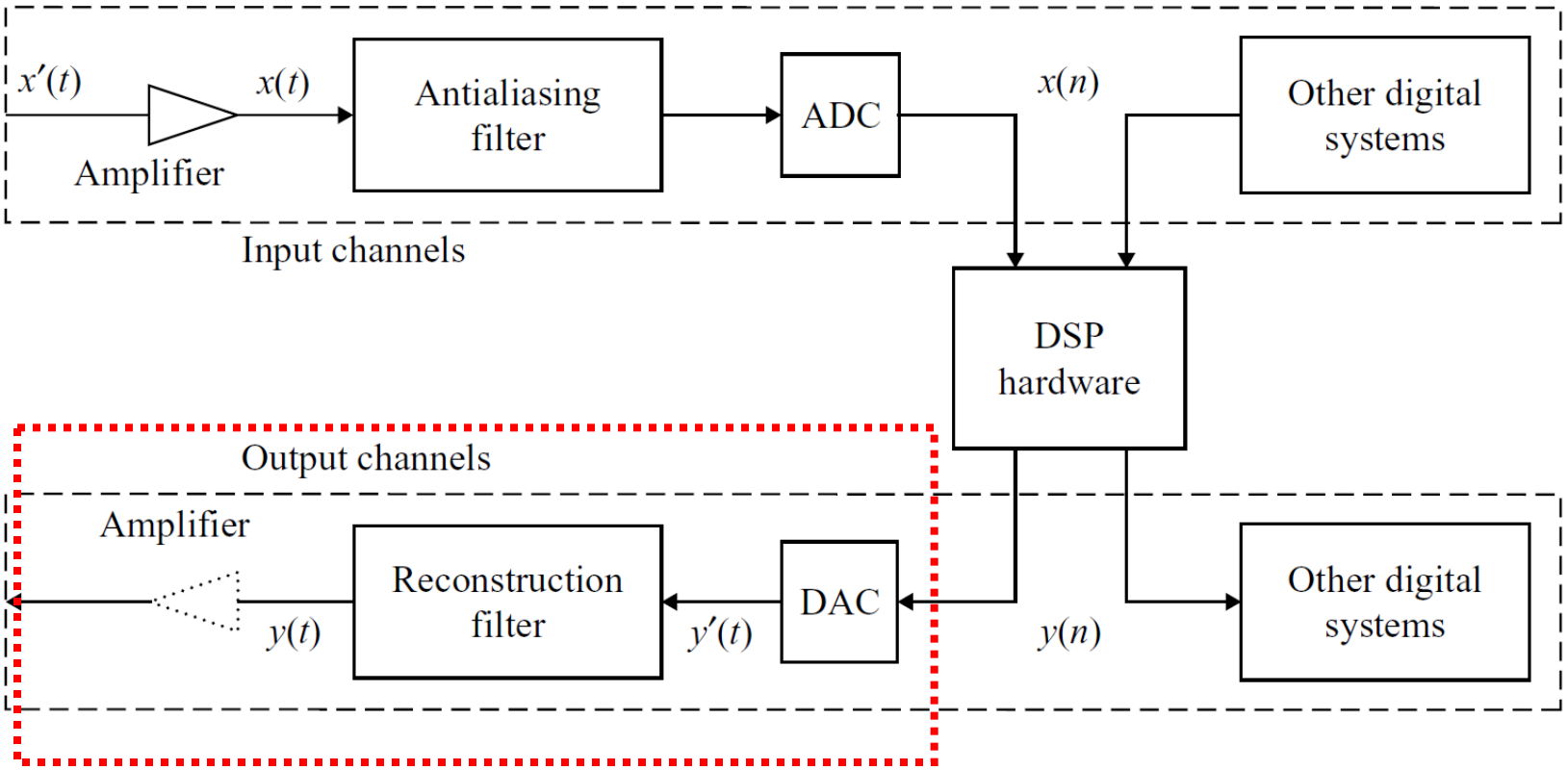
Sampling band-pass signals

m	(2Fc-B)/m	(2Fc-B)/(m+1)	Optimum Fs
1	35.0 MHz	22.5 MHz	22.5 MHz
2	17.5 MHz	15.0 MHz	17.5 MHz
3	11.66 MHz	11.25 MHz	11.25 MHz
4	8.75 MHz	9.0 MHz	-
5	7.0 MHz	7.5 MHz	-

Optimum Fs is defined here as that optimum frequency where spectral replications do no butt up against each other except at zero Hz



Reconstruction signals

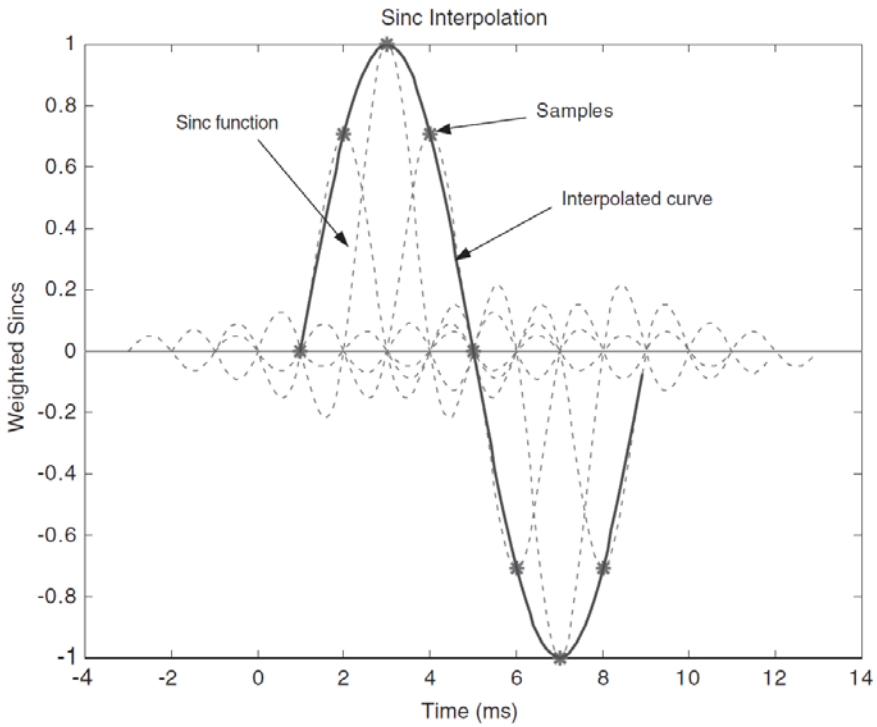


Real Time DSP System

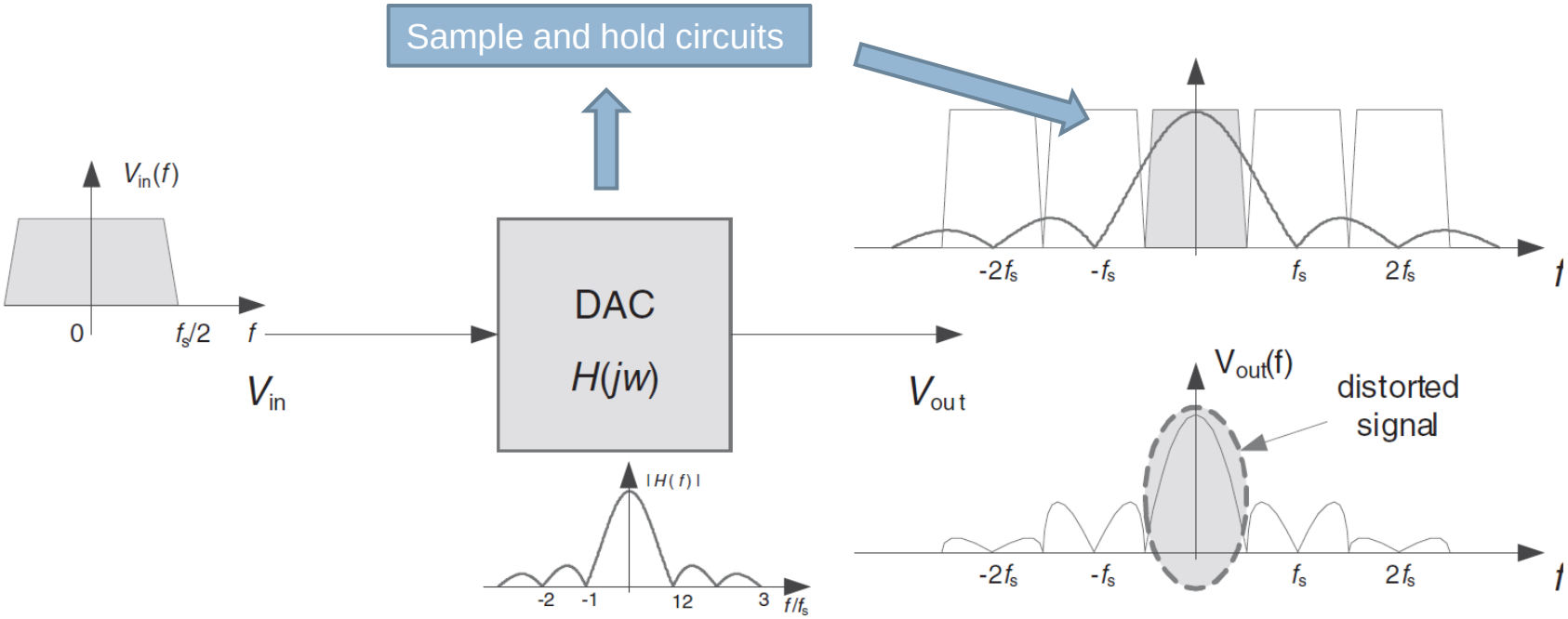
Reconstruction signals

- **Analog signal X** can be reconstructed from its samples by using the following formula:
- The **reconstruction** is based on the **interpolation** of shifted **sinc functions.**
- It is very difficult to generate sinc functions by electronic circuitry.
- An approximation of a sinc function is a pulse. **Sample and hold** circuit performs this approximation.

$$X(t) = \sum_{k=-\infty}^{\infty} X(kT_S) \cdot \mathrm{sinc}\left(\frac{t - kT_S}{T_S}\right)$$



Reconstruction Errors



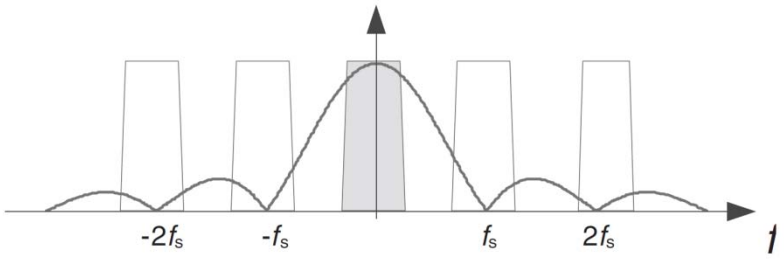
The gain in the desired central band is not constant

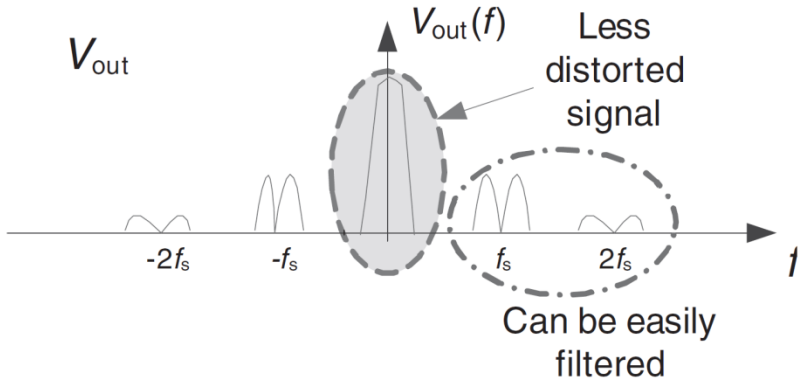
The are high-frequency replica of the signal spectrum

Reconstruction Solutions

The gain in the desired central band is not constant

It is possible to compensate for this non-ideality by using an inverse filter as part of the DSP component

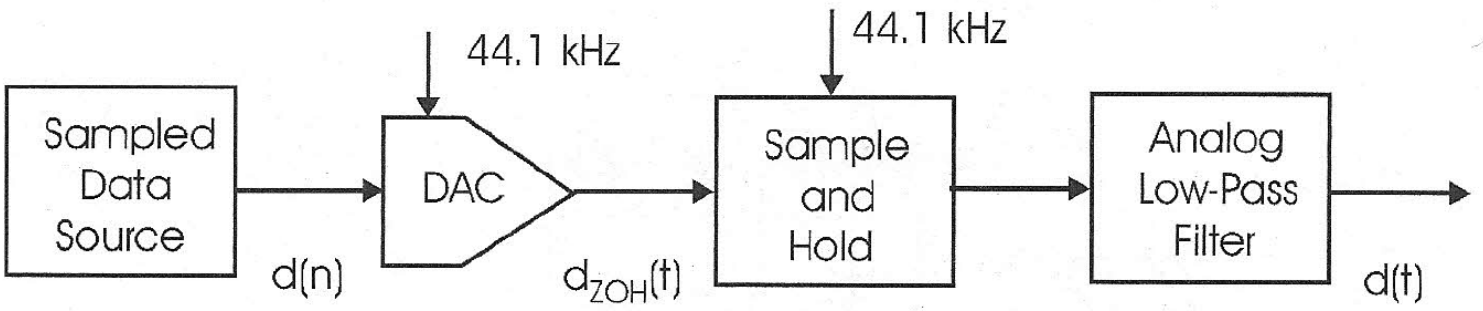


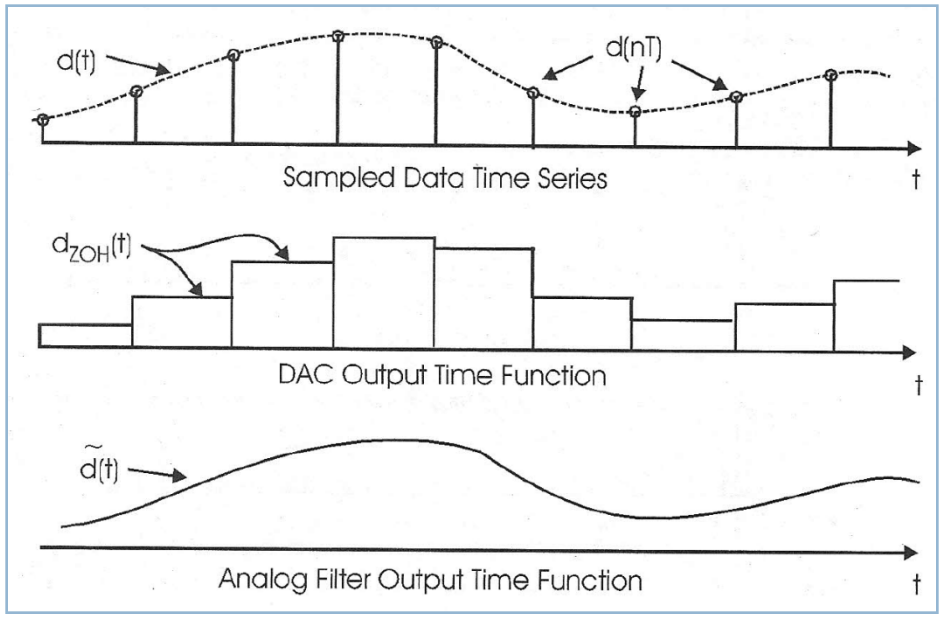


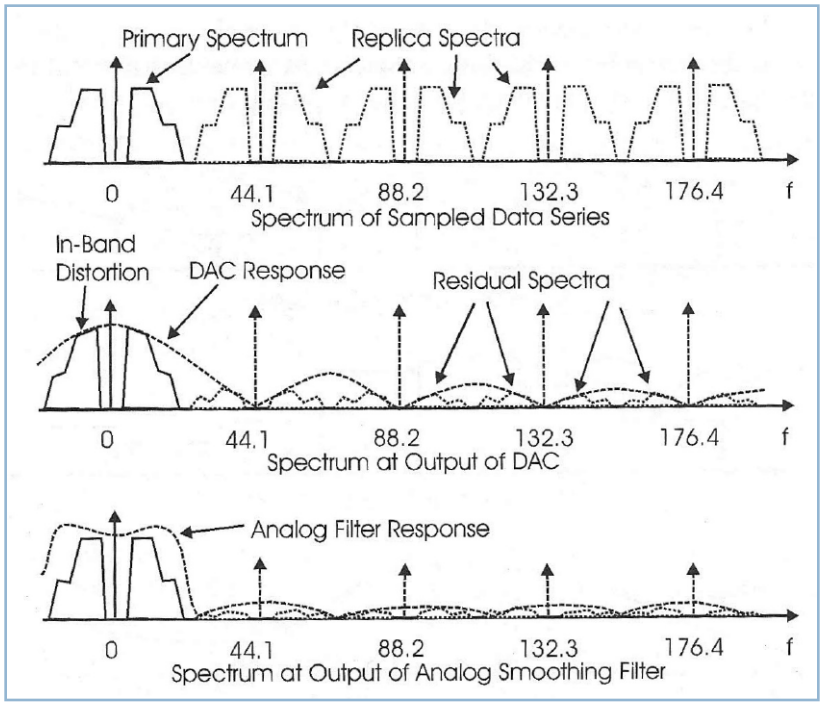
The are high-frequency replica of the signal spectrum

which can be removed by using a lowpass filter

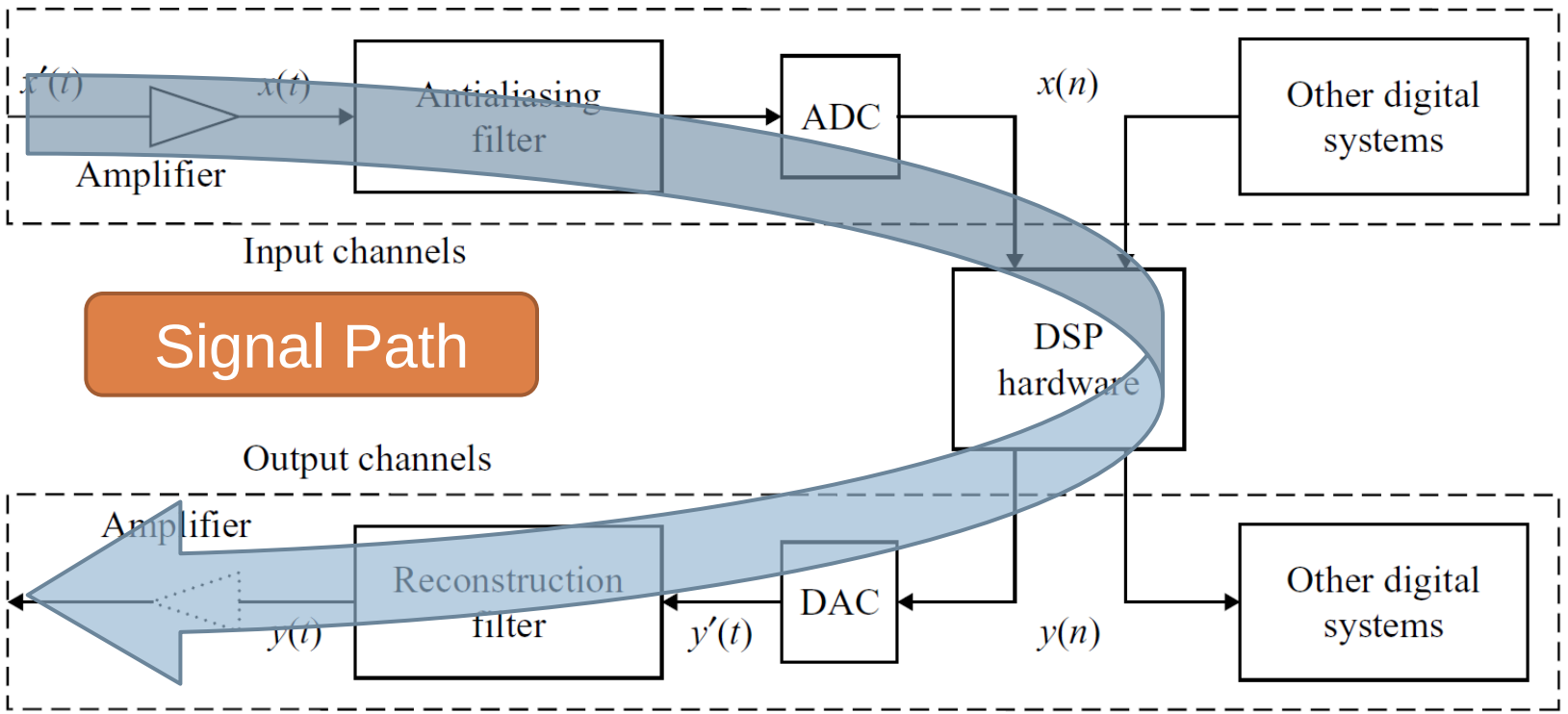
Reconstruction Errors Example







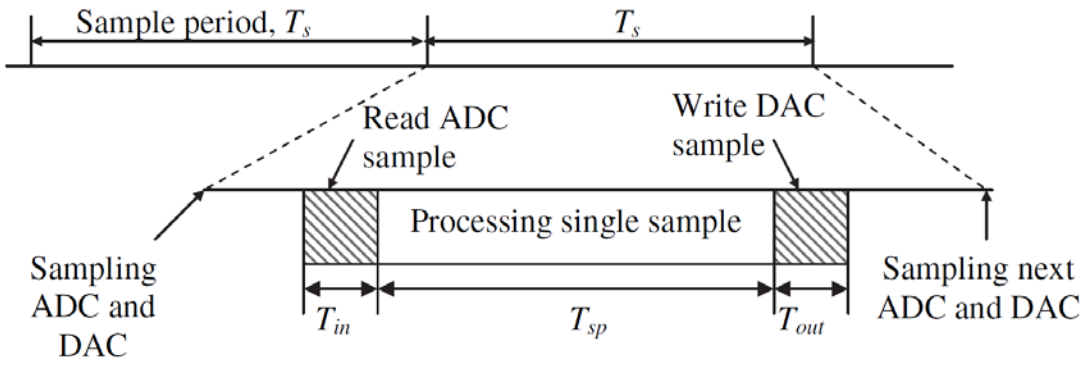
Real Time constraints



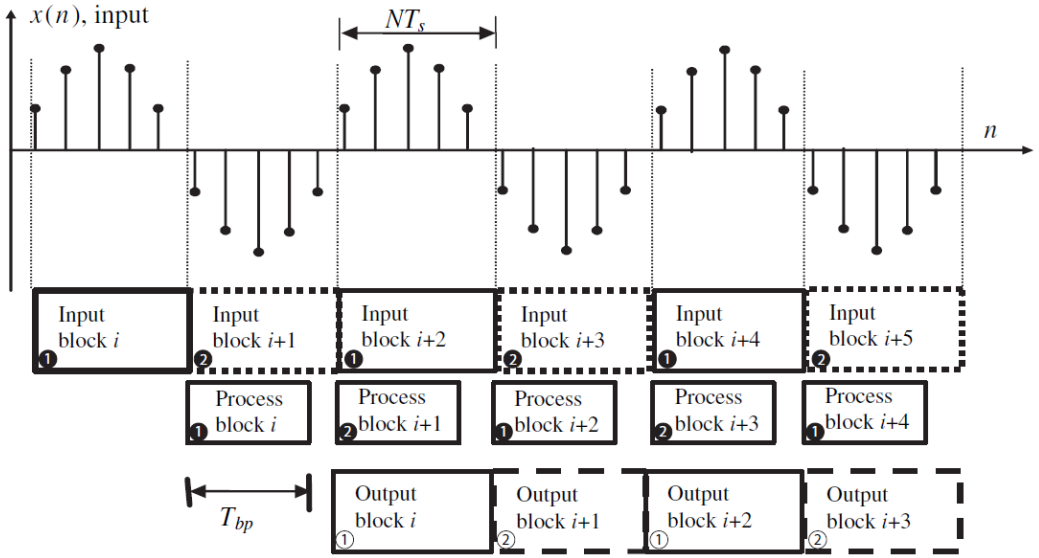
Real Time DSP System

Real time constraints

- Algorithms time (t_A) MUST fit between two consecutive sampling periods (t_S).
- Thus t_A limits the maximum frequency that a system can work.
- The definition of real time is VERY application dependant (faster speed of evolution of the system).

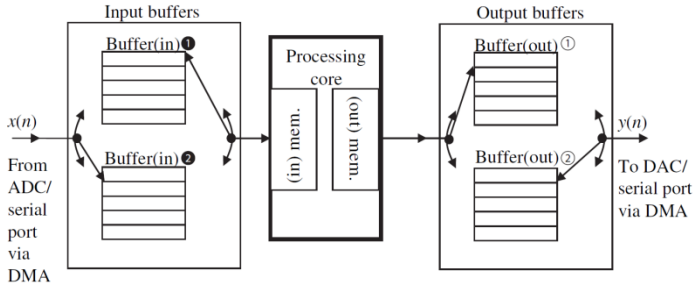


Real time constraints

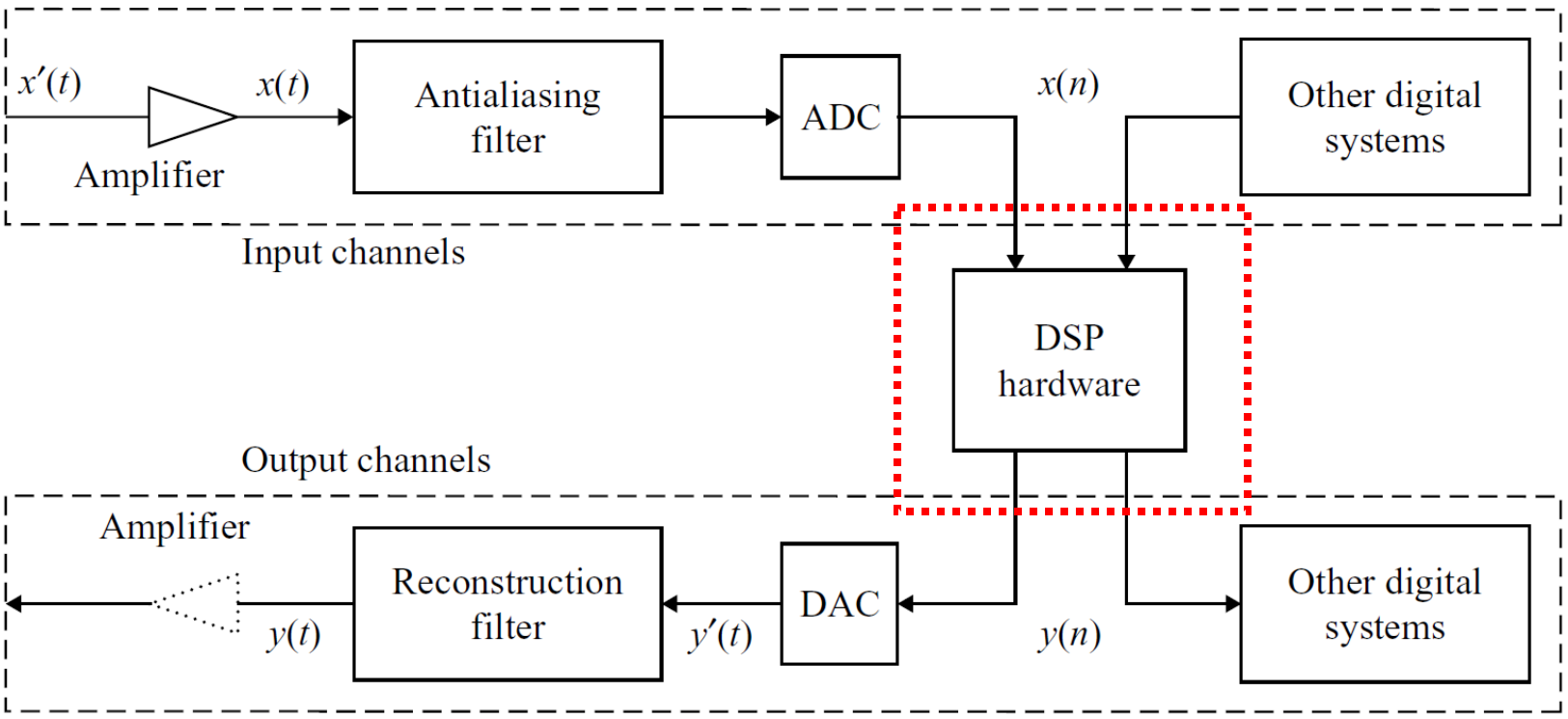


- Block Processing Mode
- 4 memory buffers of length N are required for double-buffering method.
- 2 memory buffers (in and out) are needed for internal processing by the processor.

- A delay of 2NTs is incurred in block processing.
- More complicated programming is needed to manage the switching between buffers.
- Can be configured the ADC and DAC to transfer data samples into the internal memory of processor using the serial ports and the DMA.



DSP hardware

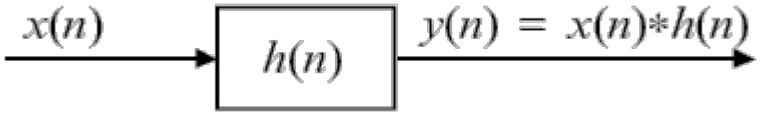


Real Time DSP System

What can we do with a DSP?

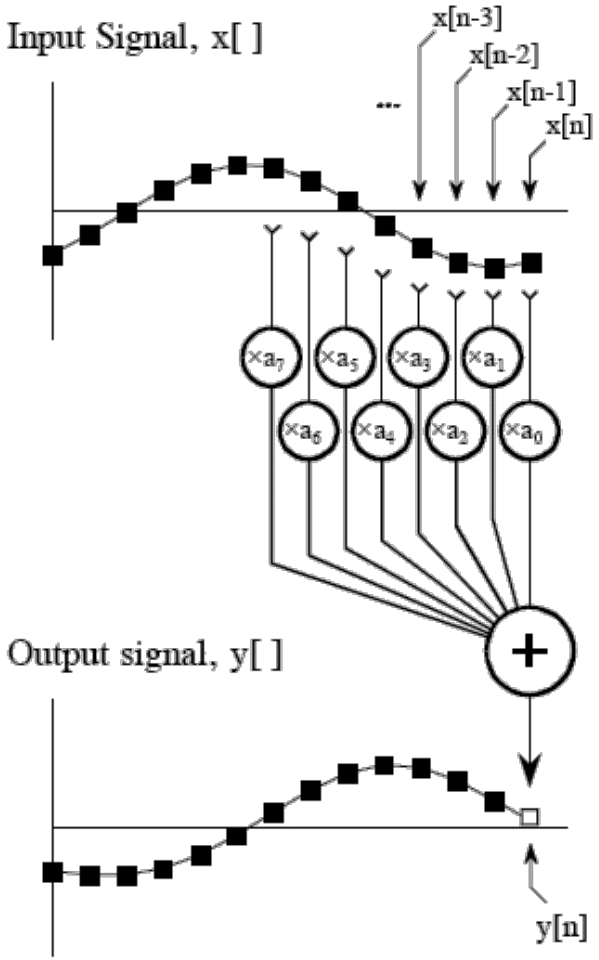
- Almost any linear and nonlinear system (PID controller).
- Digital filters (FIR-IIR).
- Adaptive systems (LMS algorithm).
- Modulators and demodulators.
- Any mathematical intensive algorithm (FFT-DCT-WT).
- Image compression algorithms
- Real time video processing

Linear systems implementation

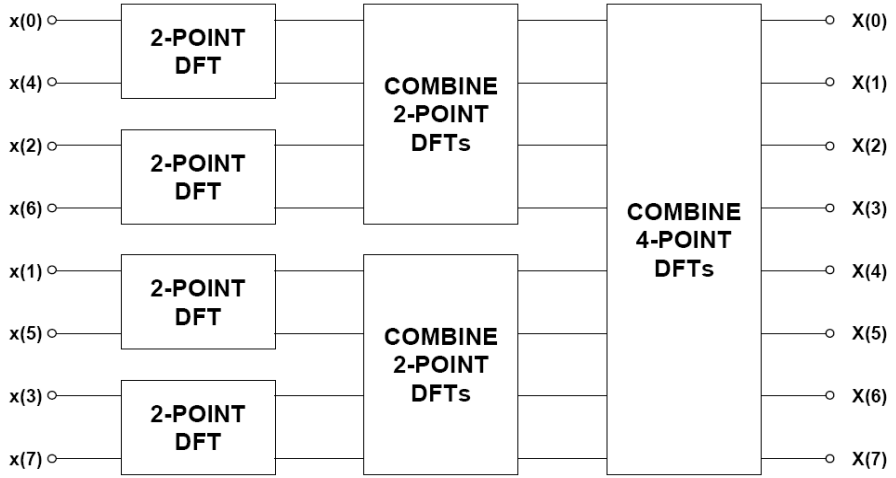


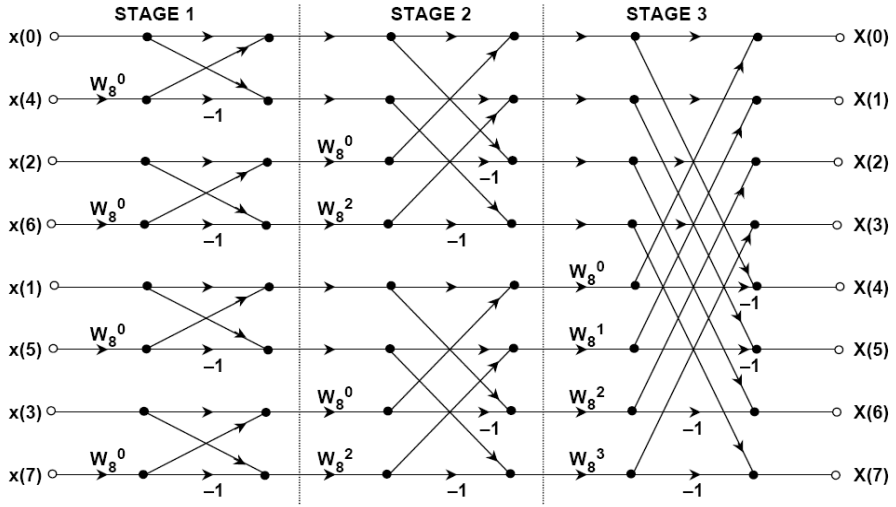
$$y(n) = x(n) * h(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k) = \sum_{k=-\infty}^{\infty} h(k)x(n-k),$$

- Being x(n) and h(n) are arrays of numbers. If we want to compute y(n) we have to multiply and sum the last M samples, being M the length of h(n). This repeated for every new sample received from de ADC.
- As you can see, any linear system uses multiplications, accumulations (sums), and loops intensively.



Fast Fourier Transform FFT





$$X(0) = A(0) + W_8^0 B(0) = 0 + j0 + (1 + j0)(0 + j0) = 0 + j0 + 0 + j0 = 0 \angle 0°,$$

$$X(1) = A(1) + W_8^1 B(1) = 0 - j1.999 + (0.707 - j0.707)(1.414 - j1.414)$$
$$= 0 - j1.999 + 0 - j1.999 = 0 - j4 = 4 \angle -90°$$

$$X(2) = A(2) + W_8^2 B(2) = 1.414 + j0 + (0 - j1)(-1.414 + j0)$$
$$= 1.414 + j0 + 0 + j1.4242 = 1.414 + j1.414 = 2 \angle 45°,$$

$$X(3) = A(3) + W_8^3 B(3) = 0 + j1.999 + (-0.707 - j0.707)(1.414 + j1.414)$$
$$= 0 + j1.999 + 0 - j1.999 = 0 \angle 0°,$$

$$X(4) = A(0) + W_8^4 B(0) = 0 + j0 + (-1 + j0)(0 + j0)$$
$$= 0 + j0 + 0 + j0 = 0 \angle 0°,$$

$$X(5) = A(1) + W_8^5 B(1) = 0 - j1.999 + (-0.707 + j0.707)(1.414 - j1.414)$$
$$= 0 - j1.999 + 0 + j1.999 = 0 \angle 0°,$$

$$X(6) = A(2) + W_8^6 B(2) = 1.414 + j0 + (0 + j1)(-1.414 + j0)$$
$$= 1.414 + j0 + 0 - j1.414 = 1.414 - j1.414 = 2 \angle -45°, \text{ and}$$

$$X(7) = A(3) + W_8^7 B(3) = 0 + j1.999 + (0.707 + j0.707)(1.414 + j1.414)$$
$$= 0 + j1.999 + 0 + j1.999 = 0 + j4 = 4 \angle 90°.$$

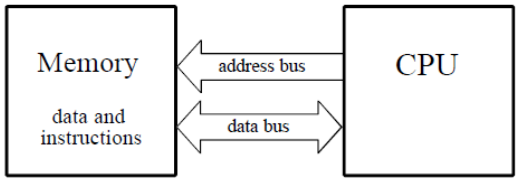
Summary of desirable features of a DSP

- Fast in mathematics operations, and combinations of them (multiply and sum specially).
- Flexible addressing modes (bit reversal, circular buffers, zero overhead loops)
- DSP specific instruction set (arithmetic shifting, saturating arithmetic, rounding, normalization)
- Minimum overhead peripherals (communications devices specially)
- DSP instructions for specific applications (Video, Control, Audio)

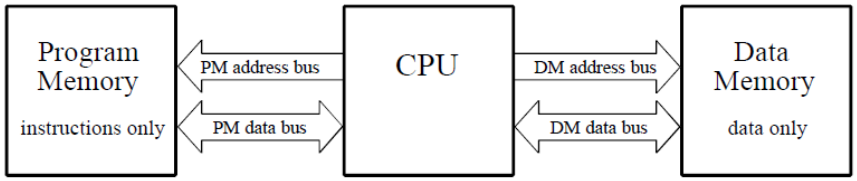
So, those are DSP math features

- Multiply and Accumulators (MAC's) units.
- ALU's (fixed and floating point).
- Barrel shifters.
- Depending on DSP application, more than one unit are present in modern DSP's, allowing parallelism.
- Harvard (modified) architecture provide multiple operations per cycle.

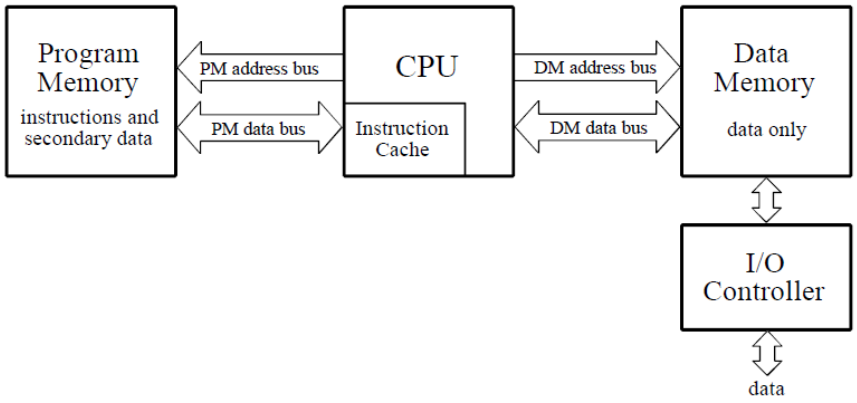
a. Von Neumann Architecture (*single memory*)



b. Harvard Architecture (*dual memory*)



c. Super Harvard Architecture (*dual memory, instruction cache, I/O controller*)



Multi-issue architectures

- Until 1997, most DSPs were very similar
  - Specialed execution units
  - Specialed, complex instruction set
    - Difficult to program in assambly
    - Unfriebdly compiler targets
  - One instruction per instruction cycle
- Multi-issue architectures are different
  - Multiple independent instruction per cycle
  - Often, simple, orthogonal instruction set

VLIW and superscalar

- In ***superscalar*** designs, the number of execution units is invisible to the instruction set. Each instruction encodes only one operation. For most superscalar designs, the instruction width is 32 bits or fewer.
- ***VLIW*** (***V****ery* ***L****ong* ***I****nstruction* ***W****ord*) is a type of MIMD (***M****ultiple* ***I****nstruction stream,* ***M****ultiple* ***D****ata stream*). One VLIW instruction encodes multiple operations; specifically, one instruction encodes at least one operation for each execution unit of the device.

VLIW and superscalar

- For example, if a VLIW device has five execution units, then a VLIW instruction for that device would have five operation fields, each field specifying what operation should be done on that corresponding execution unit. VLIW instructions are usually at least 64 bits wide, and on some architectures are much wider.
- For example, the following is an instruction for the SHARC. In one cycle, it does a floating-point multiply, a floating-point add, and two autoincrement loads. All of this fits into a single 48-bit instruction.

*f12=f0*f4, f8=f8+f12, f0=dm(i0,m3), f4=pm(i8,m9);*

VLIW and superscalar

- Since the earliest days of computer architecture, some CPUs have added several additional arithmetic logic units (ALUs) to run in parallel. ***Superscalar CPUs*** use hardware to decide which operations can run in parallel.
- ***VLIW CPUs*** use software (the compiler) to decide which operations can run in parallel. Because the complexity of instruction scheduling is pushed off onto the compiler, the hardware's complexity can be substantially reduced.

Superscalar vs. VLIW

- Superscalar and VLIW: More than a single instruction can be issued to the execution units per cycle.
- Superscalar machines are able to dynamically issue multiple instructions each clock cycle from a conventional linear instruction stream.
- VLIW processors use a long instruction word that contains a usually fixed number of instructions that are fetched, decoded, issued, and executed synchronously.
- Superscalar: dynamic issue, VLIW: static issue

Architectural Features for Efficient Programming

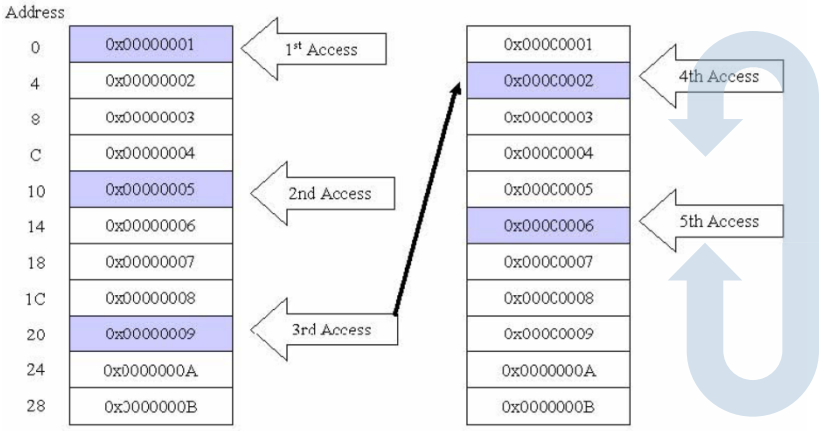
- Specialized addressing modes
- Hardware Loop Constructs
- Cacheable memories
- Multiple operations per cycle
- Interlocked pipeline
- Another important features

Architectural Features for Efficient Programming

- **Specialized addressing modes**
- Hardware Loop Constructs
- Cacheable memories
- Multiple operations per cycle
- Interlocked pipeline
- Another important features

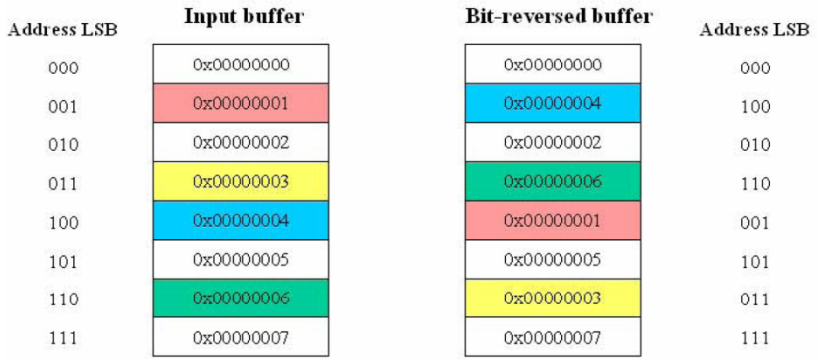
Specialized Addressing Modes

Circular buffering




```
B0 = 0x00; L0 = 44;   // Base and length
I0 = 0x00; M0 = 16;   // Index and increment
R0 = [I0++M0];   // R0=1 & I0=0x10
R0 = [I0++M0];   // R0=5 & I0=0x20
R0 = [I0++M0];   // R0=9 & I0=0x04
R0 = [I0++M0];   // R0=2 & I0=0x14
R0 = [I0++M0];   // R0=6 & I0=0x24
```

Bit-Riversal




```
B0 = 0x00; L0 = 0;   // Base and length
I0=0; M0=1;   // Index and increment
I2=256; P0 = 8;
LOOP(start, end) LC0 = P0;
start: // I0 automatically  incremented in B-R progression
R0 = [I0] || I0 += M0 (BREV);
end: // I2 point to bit-riversed buffer
 [I2++] = R0;
```

Architectural Features for Efficient Programming

- Specialized addressing modes
- **Hardware Loop Constructs**
- Cacheable memories
- Multiple operations per cycle
- Interlocked pipeline
- Another important features

Hardware Loop Constructs

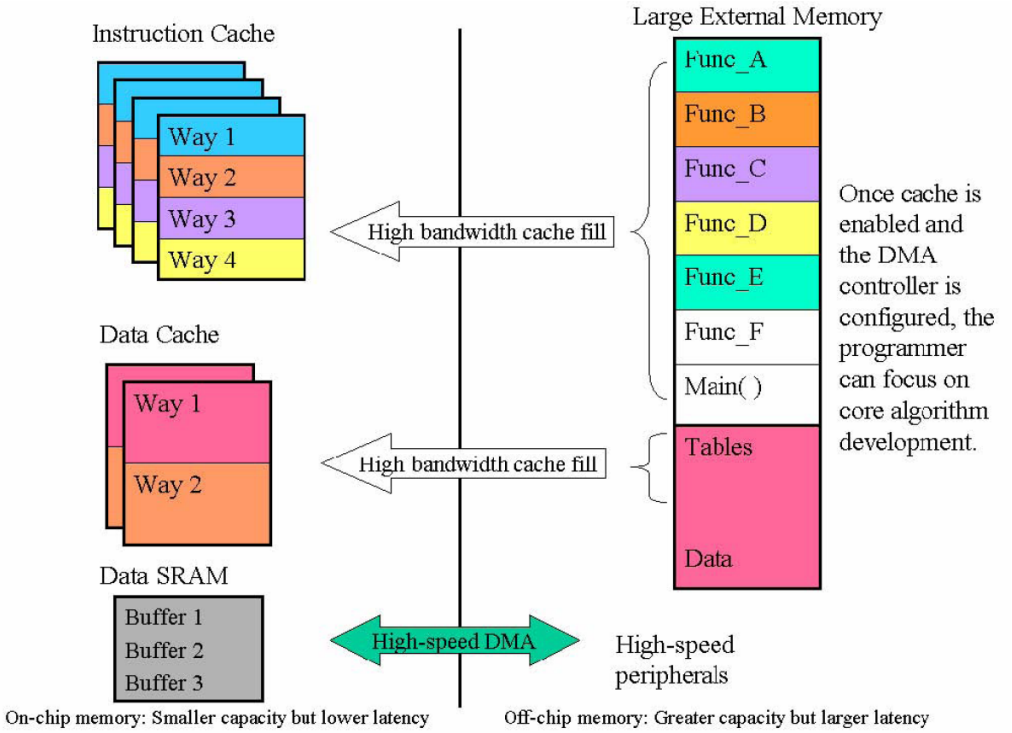
- Looping is a critical feature in communications processing algorithms.
- There are two key looping-related features that can improve performance on a wide variety of algorithms:
  - “zero-overhead hardware loop”
  - “hardware loop buffers”

Architectural Features for Efficient Programming

- Specialized addressing modes
- Hardware Loop Constructs
- **Cacheable memories**
- Multiple operations per cycle
- Interlocked pipeline
- Another important features

Cacheable memories

- Today's high-speed processors would effectively run at much slower speeds because larger applications would only fit in slower external memory.
- Programmers would be forced to manually move key code in and out of internal SRAM.
- Adding data and instruction caches into the architecture, external memory becomes much more manageable.

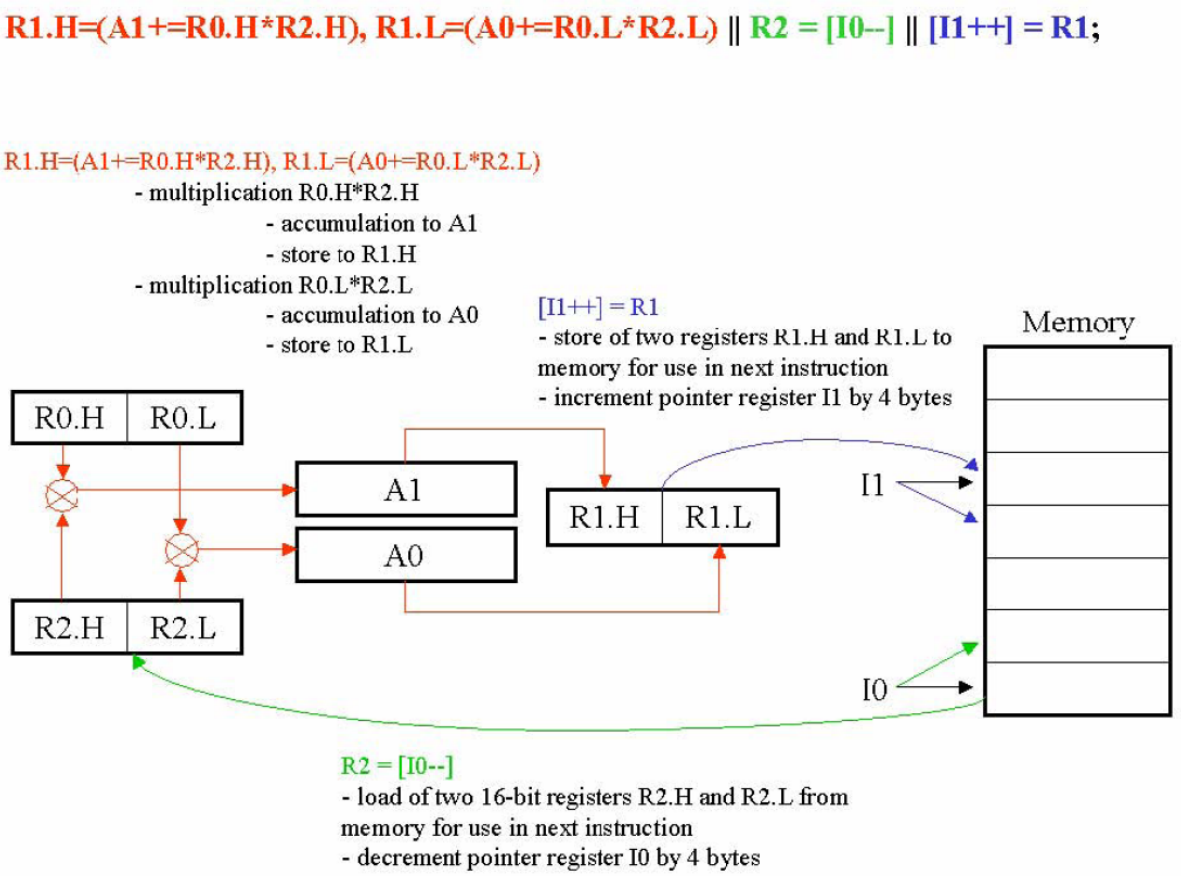


Architectural Features for Efficient Programming

- Specialized addressing modes
- Hardware Loop Constructs
- Cacheable memories
- **Multiple operations per cycle**
- Interlocked pipeline
- Another important features

Multiple operations per cycle

- In addition to performing multiple ALU/MAC operations each core processor cycle, additional data loads and stores can also be completed in the same cycle.
- The memory is typically portioned into sub-banks that can be dual-accessed by the core and optionally by a DMA controller.

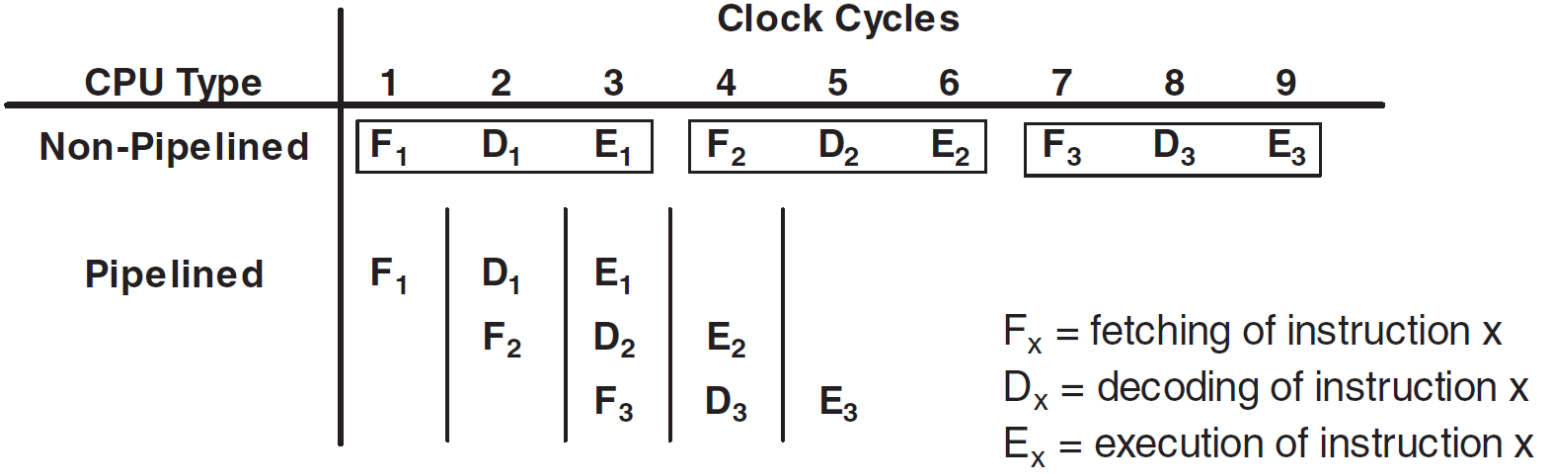


This is superscalar multi-issue architecture

Architectural Features for Efficient Programming

- Specialized addressing modes
- Hardware Loop Constructs
- Cacheable memories
- Multiple operations per cycle
- **Interlocked pipeline**
- Another important features

Interlocked pipeline



- In order to increase throughput, DSPs are designed to be *pipelined*
- When assembly programming is required, the pipeline can make programming more challenging.
- The processor automatically handles stalls and bubbles.

Architectural Features for Efficient Programming

- Specialized addressing modes
- Hardware Loop Constructs
- Cacheable memories
- Multiple operations per cycle
- Interlocked pipeline
- **Another important features**

Another important features

- RISC like registers and instruction set
- Multiple data/program buses.
- DMA controller for handling peripherals
- In traditional fixed-point DSPs, word sizes are usually fixed. However, there is an advantage to having data registers that can be treated as:
  - One 64-bit word
  - Two 32-bit word
  - Four 16-bit word
  - Eight 8-bit word

DSP clasification

- Fixed or Floating point arithmetic.
- Millions of multiply–accumulate operations per second, MMACs.
- Millions of floating-point operations per second, MFLOPS.
- Application specific features (video, audio, control, communications).
- Memory

Why DSP hardware?

	ASIC	FPGA	μP/μC	DSP processor	DSP processors with HW accelerators
Flexibility	None	Limited	High	High	Medium
Design time	Long	Medium	Short	Short	Short
Power consumption	Low	Low–medium	Medium–high	Low–medium	Low–medium
Performance	High	High	Low–medium	Medium–high	High
Development cost	High	Medium	Low	Low	Low
Production cost	Low	Low–medium	Medium–high	Low–medium	Medium

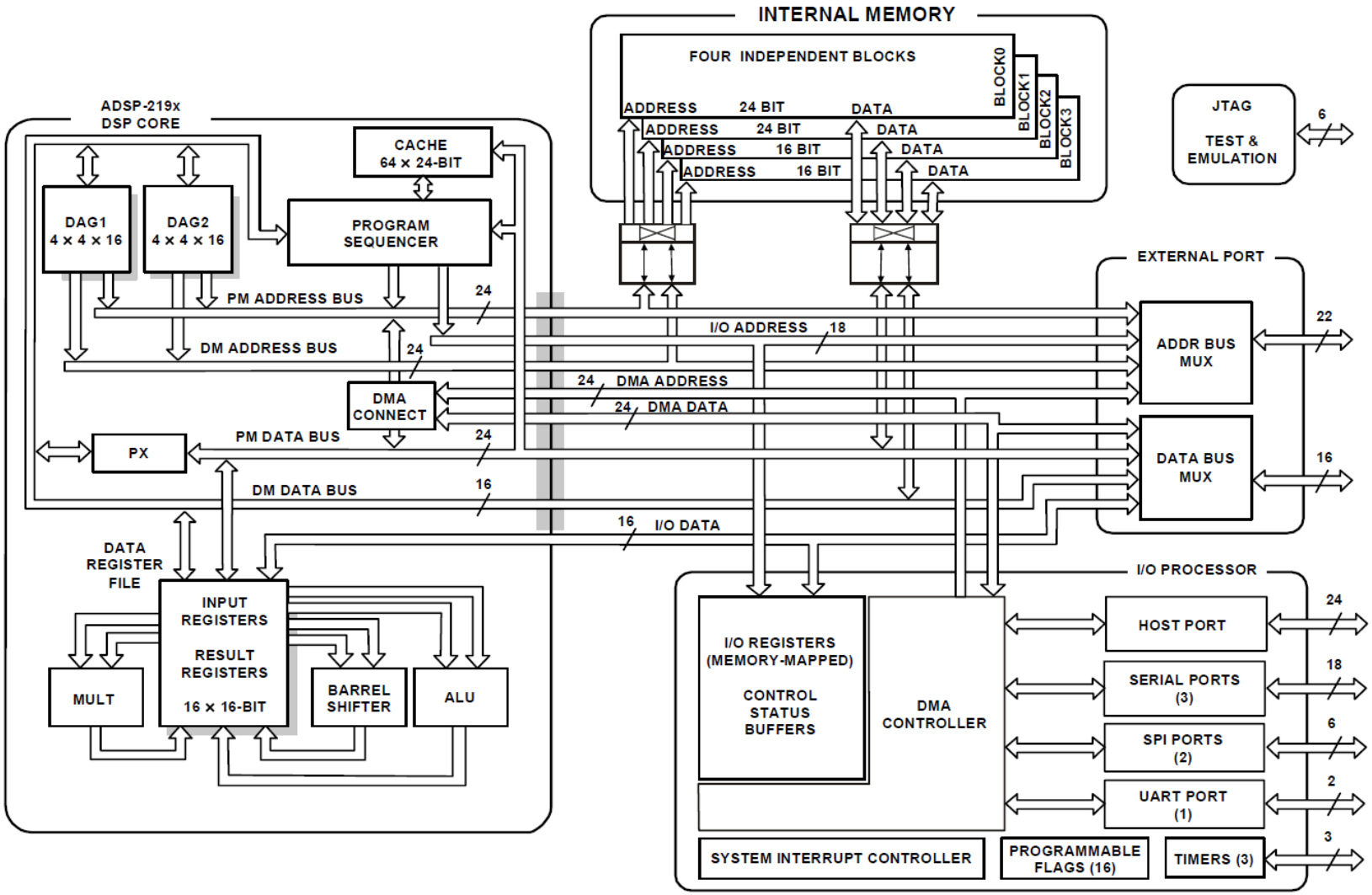
- Special-purpose (custom) chips such as application-specific integrated circuits (ASIC).
- Field-programmable gate arrays (FPGA).
- General-purpose microprocessors or microcontrollers (*μP/μC).*
- General-purpose digital signal processors (DSP processors).
- DSP processors with application-specific hardware (HW) accelerators.

ADI Processors

- **TigerSHARC® Processors**
  - 32-bit fixed-point as well as floating-point
  - Clock Speed: 250MHz to 600MHz
  - 4.8 GMACs of 16-bit performance / 3.6 GFLOPs
  - 24 Mbits of on- chip memory
  - 5 Gbytes of I/O bandwidth
- **SHARC® Processors**
  - 32-Bit floating-point
  - Clock Speed: 150MHz to 400MHz / 2.4 GFLOPs.
  - Accelerator Architecture: FIR, IIR, FFT.
- **Blackfin® Processors**
  - 16/32-bit fixed point
  - Clock Speed: 200MHz to 756MHz / 1.5 GMACs
  - Very low power consumption: 0.23mW/Mhz
  - RTOS supported. Multicore 600MHz / 2.4 GMACs.
- **ADSP-21xx Processors**
  - 16/32-bit fixed point
  - Clock Speed: 75MHz to 160MHz
  - Analog Devices brought first programmable processor to market in 1986

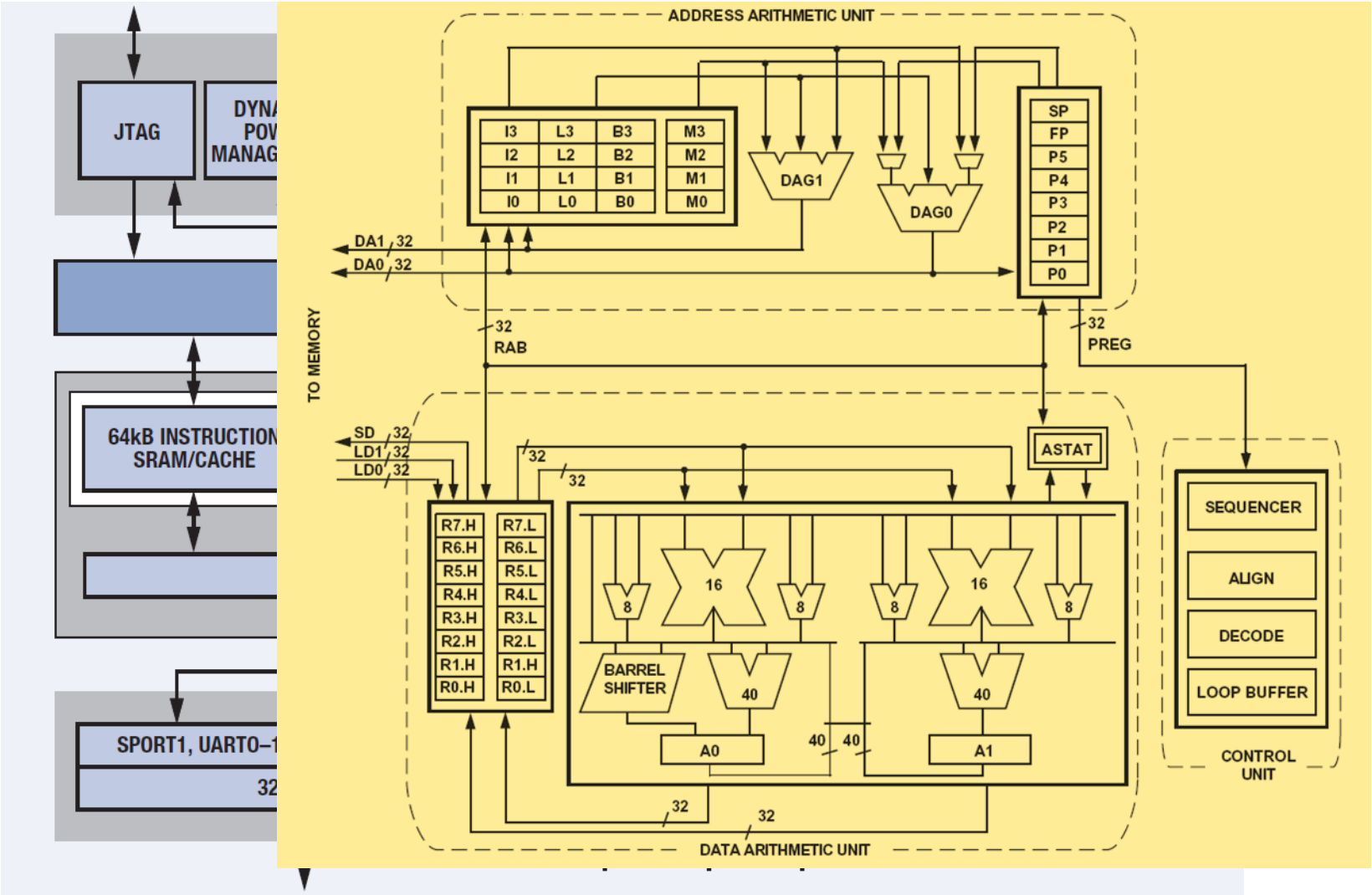
ADSP-21xx Processors

ADSP-2191 BLOCK DIAGRAM



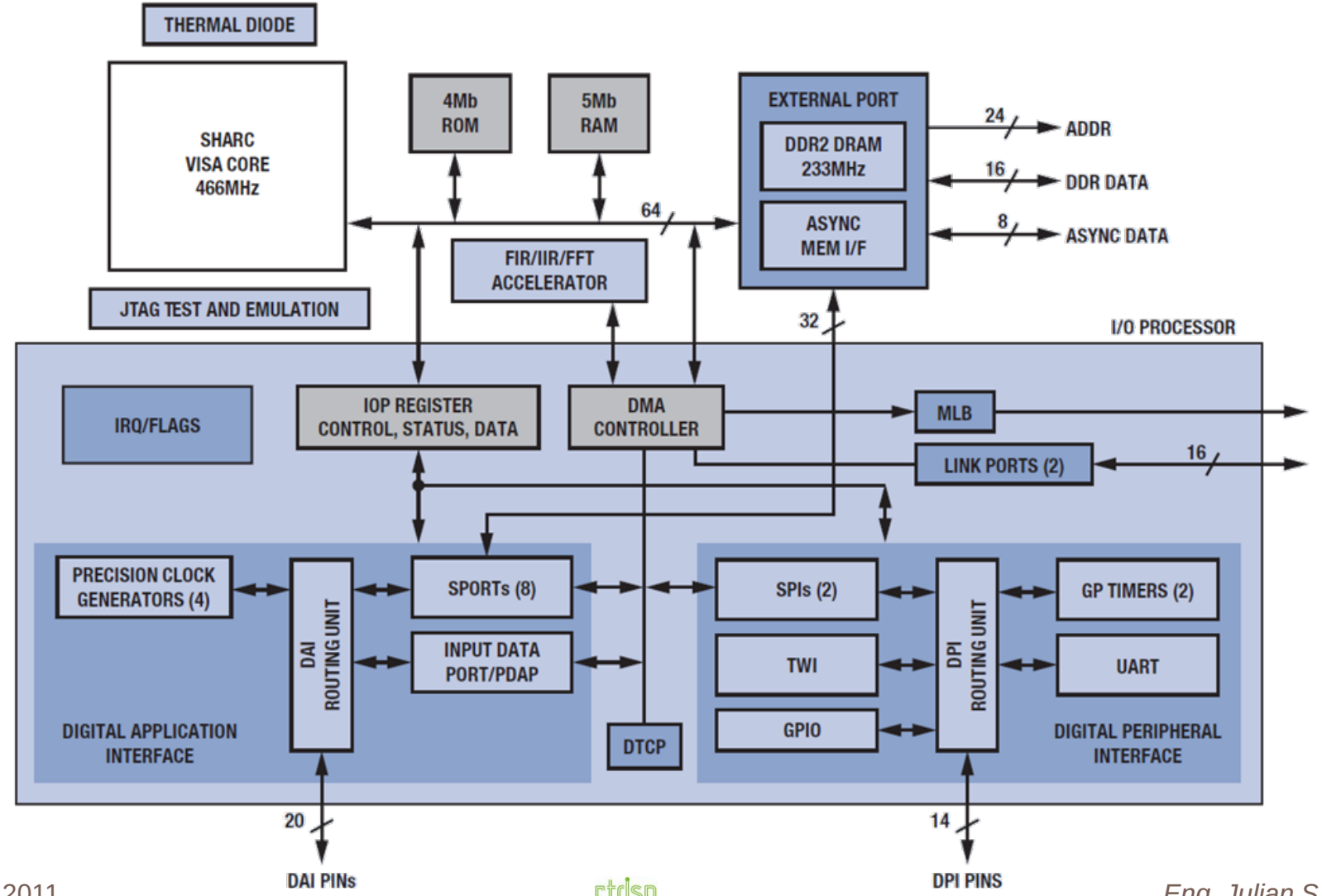
Blackfin Processors

ADSP-BF536/ADSP-BF537 BLOCK DIAGRAM



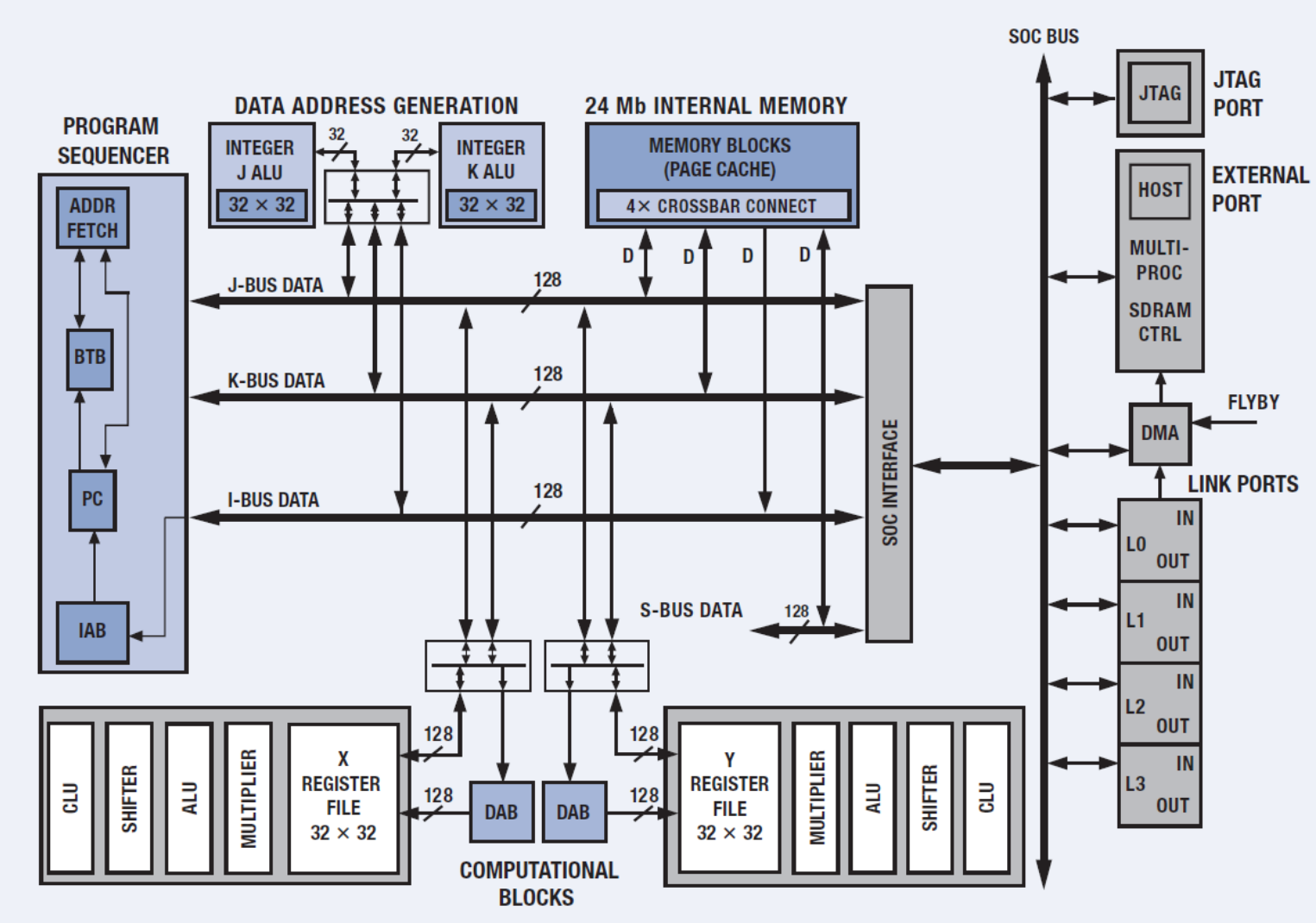
SHARC Processors

ADSP-2146x BLOCK DIAGRAM



TigerSHARC Processor

ADSP-TS201S BLOCK DIAGRAM



Markets and Applications

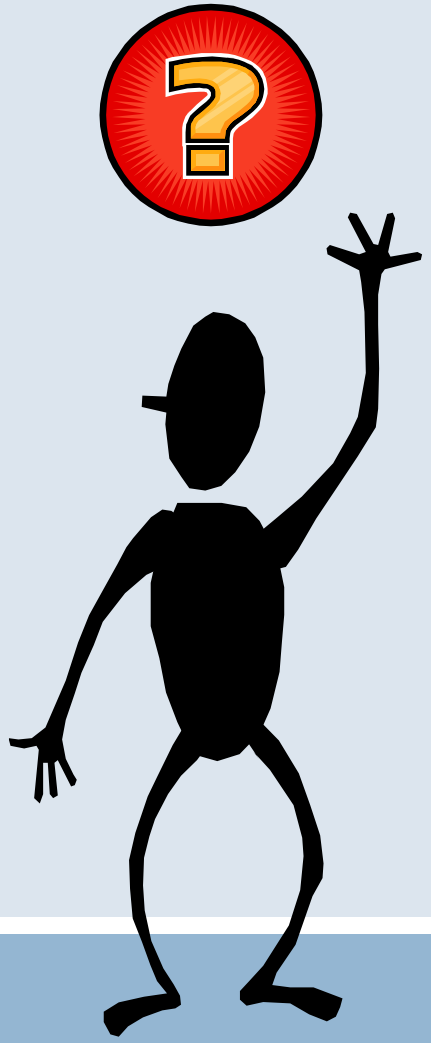
Market	Applications	Architecture/ Platform
Communications		
Broadband	Broadband over Power Lines	Blackfin
	Digital Media Gateways (VOD)	Blackfin
	Home Networking	Blackfin
	IP PBX	Blackfin
	IP Set-Top Box	Blackfin
	Media Node	Blackfin
	Multimedia over IP	Blackfin
	Video Conferencing/Phone	Blackfin
	Video Surveillance/Security	Blackfin
	Voice over IP	Blackfin
Wireless	Access (Broadband) (i.e., 802.16 ...)	Blackfin, TigerSHARC
	Base Station	TigerSHARC
	Cellular Location	Blackfin
	Satellite Phone	Blackfin
Automotive		
In Cabin	Audio Amplifier	SHARC
	Audio Jukebox	Blackfin
	Digital Radio	Blackfin
	Driver Assistance	Blackfin
	Handsfree	Blackfin
	Head Unit	Blackfin, SHARC
	Multimedia Device Interface	Blackfin
	Navigation	Blackfin
	Occupancy/Classifications	Blackfin
	Premium Audio System	SHARC
	Rear Seat Audio/Video	Blackfin

Market	Applications	Architecture/ Platform
Industrial and Instrumentation		
Medical	CT	Blackfin, SHARC, TigerSHARC
	Diagnostic	Blackfin, SHARC, TigerSHARC
	MRI	SHARC, TigerSHARC
	Patient Monitoring	Blackfin, SHARC, TigerSHARC
	Portable Medical	Blackfin, SHARC
	Ultrasound	Blackfin, SHARC, TigerSHARC
	X-Ray	SHARC, TigerSHARC
Point of Sale	Scanner	Blackfin
	Vending Machine	Blackfin
Test/ Measurement Equipment	ATE	Blackfin, SHARC, TigerSHARC
	Communications	Blackfin, SHARC, TigerSHARC
	Measurement	Blackfin, SHARC, TigerSHARC
Industrial	Data Acquisition	Blackfin, SHARC
	Factory Automation	Blackfin
	Industrial Control	Blackfin, SHARC, TigerSHARC
	Machine Control	Blackfin
	Metering	Blackfin
	Motor Control	Blackfin
	Network Management	Blackfin
	Power Control	Blackfin
	Robotics	Blackfin, SHARC, TigerSHARC
	Verification and Biometrics	Blackfin
	Video Surveillance Systems	Blackfin
	Vision Systems	Blackfin

Recommended bibliography

- RG Lyons, Understanding Digital Signal Processing 2nd ed. Prentice Hall 2004.
  - Ch2: Periodic Sampling
- SW Smith, The Scientist and Engineer's guide to DSP. California Tech. Pub. 1997.
  - Ch1: The Breadth and Depth of DSP
  - Ch3: ADC and DAC
- SM Kuo, BH Lee. Real-Time Digital Signal Processing 2nd ed. John Wiley and Sons. 2006
  - Ch1:Introduction to Real-Time Digital Signal Processing

- NOTE: Many images used in this presentation were extracted from the recommended bibliography.



Questions?

Thank you!