829 DSD

Intro to Digital Sound Processing on the dsPIC® Digital Signal Controller

Objectives

- An overview of digital sound processing
- Specific design elements of a digital sound application
- How to calculate processor utilization in a real-time system
- *Have some fun with dsPIC™ Developer Tools!*

Agenda

- Introduction
- Demo
- Application Design
- Lab #1
- Processor Utilization
- Lab #2
- Closing Remarks

Introduction

Questions to ask

- What is digital sound?
- What is a codec?
- What are typical applications?
- What are important design criteria?

MICROCHIP MASTERs

Sound Basics

- A continuous wave of pressure changes in the atmosphere
- Converted to analog electrical signal by a microphone
- Can be classified by frequency
  - Frequency is closely related to pitch
  - Low-pitch sounds -- low frequency
  - High-pitch sounds -- high frequency

Sound Frequencies

Human hearing	20 to 22,000 Hz
Most natural sounds	0 to 11,000 Hz
Voiced speech	200 to 3,400 Hz

Hz: cycles per second

Digital sound

- Sound has been *sampled* and *quantized*
  - Sampled: measured at a point in time
  - Quantized: converted to a number

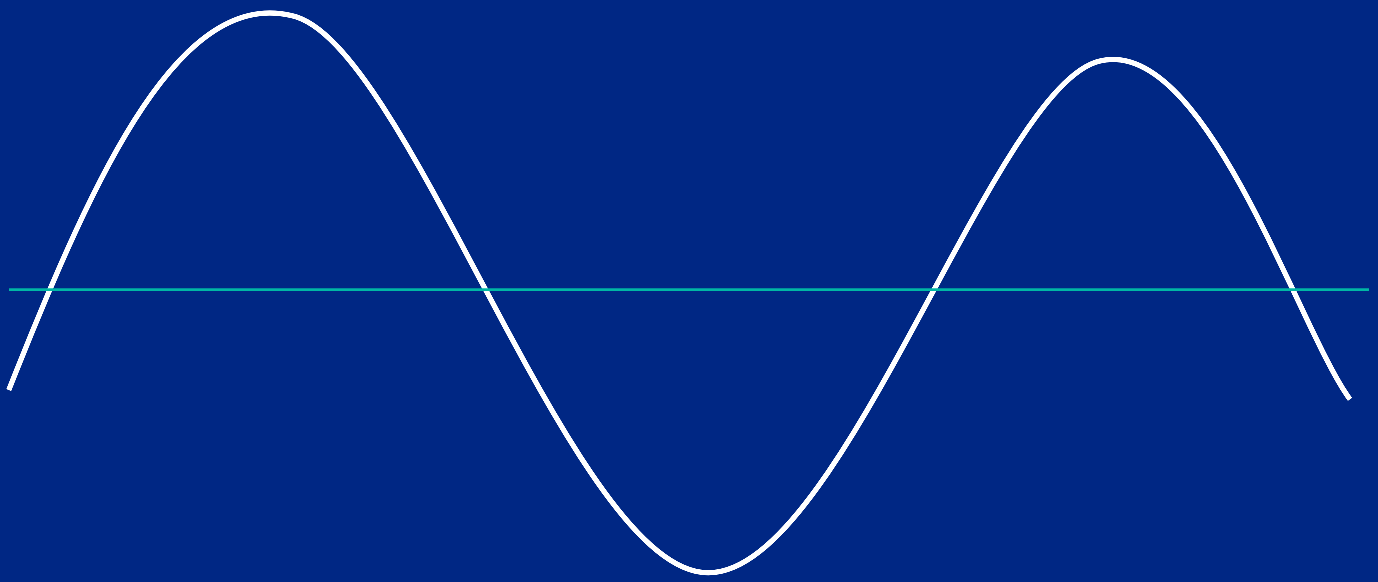
The result is a series of numbers that represents the original analog signal at discrete points in time.

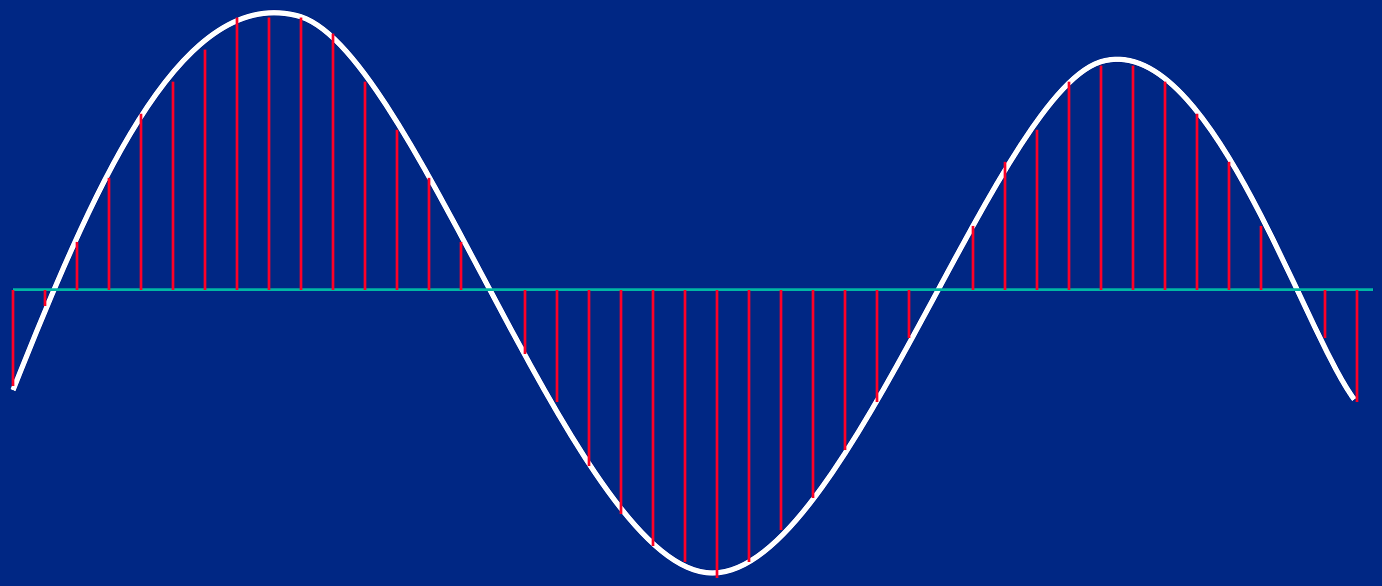
Sampling Rate

- How often the signal is measured
- Determines frequency response
  (range of frequencies that can be reproduced)

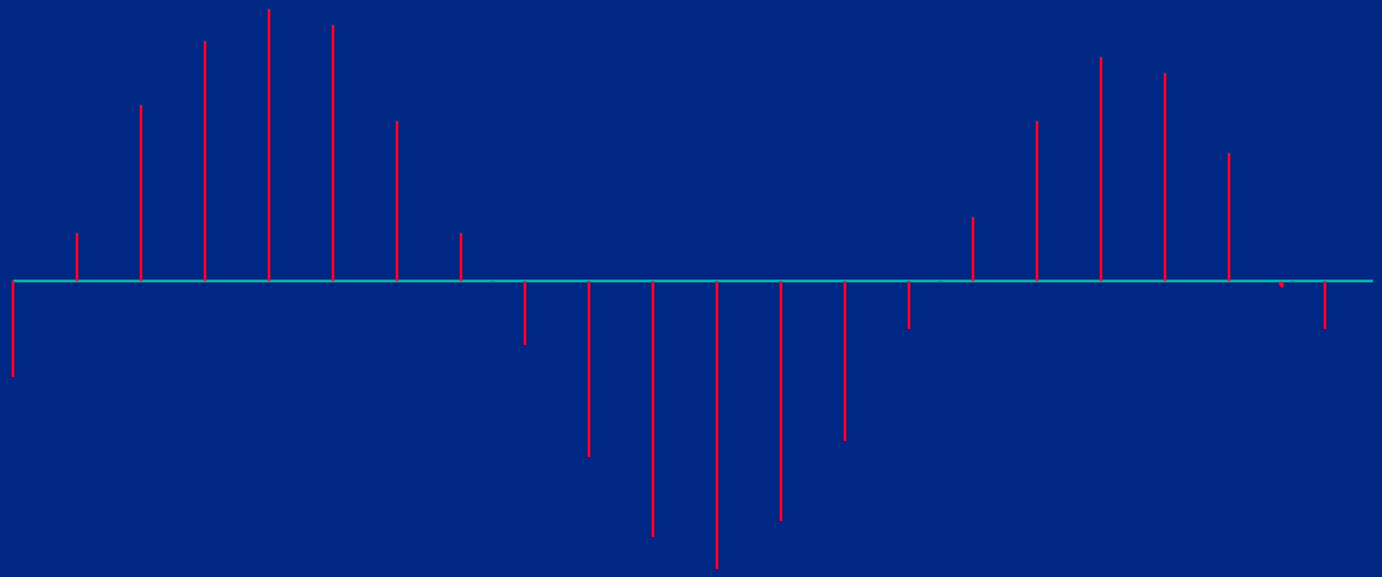
To reproduce target frequency f,
2f is the minimum sampling rate
(Nyquist Theorem)

Sampling Rate: Input Wave

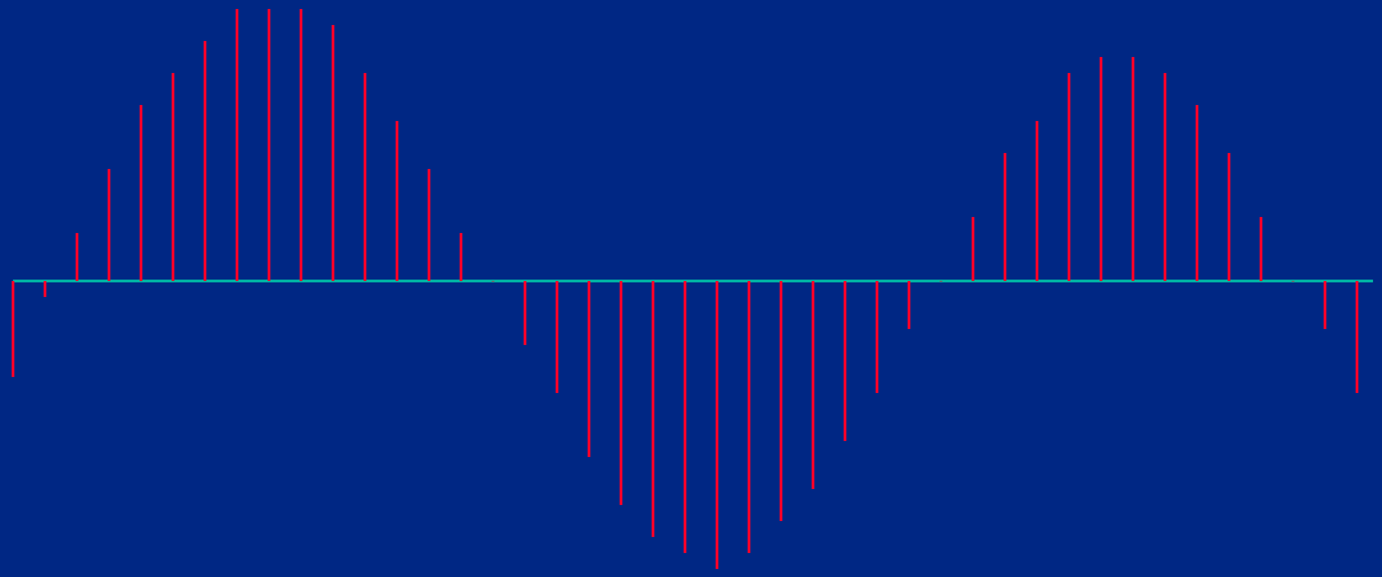




Sampling Rate: f



Sampling Rate: 2*f*



Common Sampling Rates

Music studio	96 kHz
CD audio	44 kHz
FM radio	~22 kHz
Telephone	~8 kHz

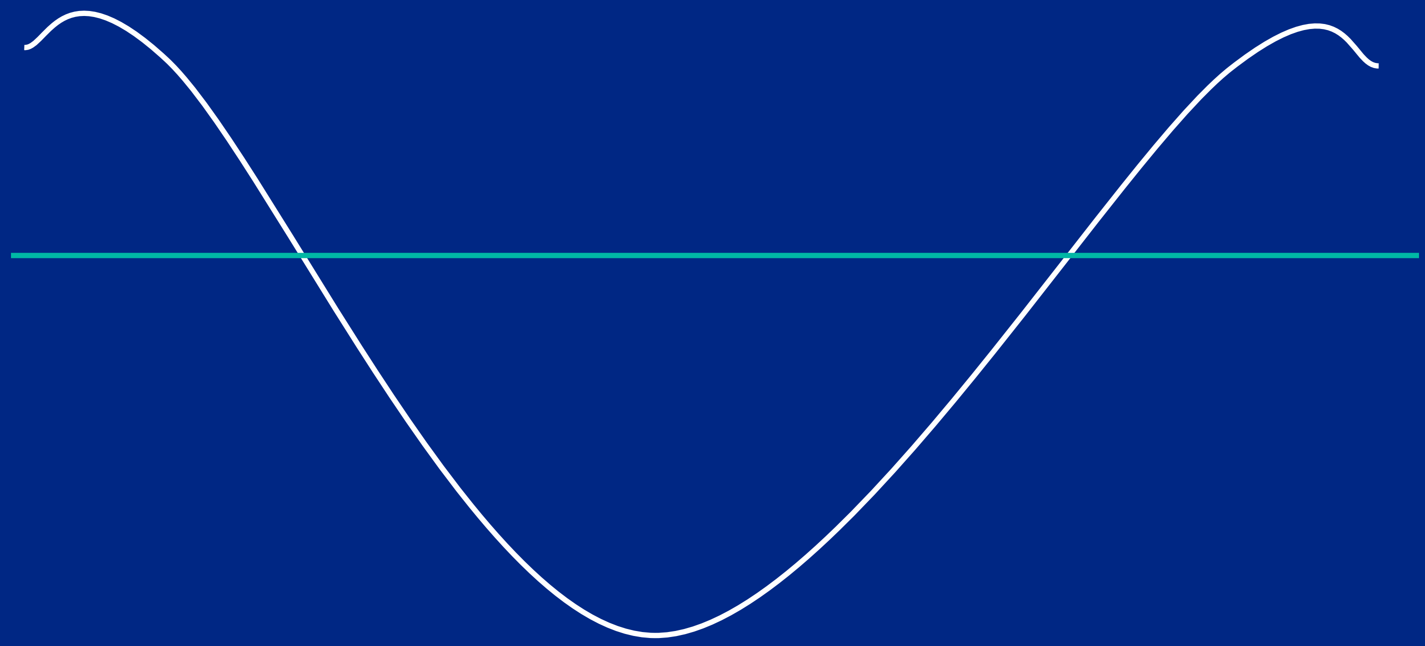
Sample Size

- How accurate is quantization
- Determines sound quality

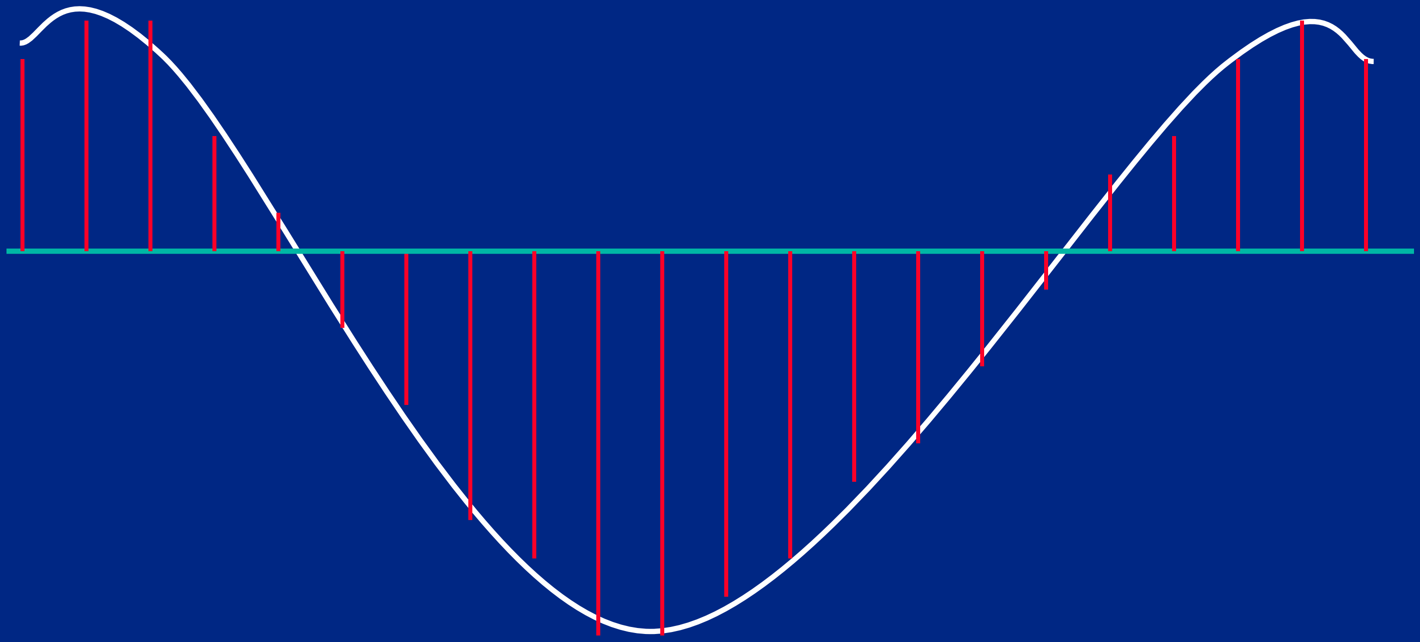
  (Quality = signal to noise ratio)

8-bit samples are acceptable, each additional bit reduces noise by 50%

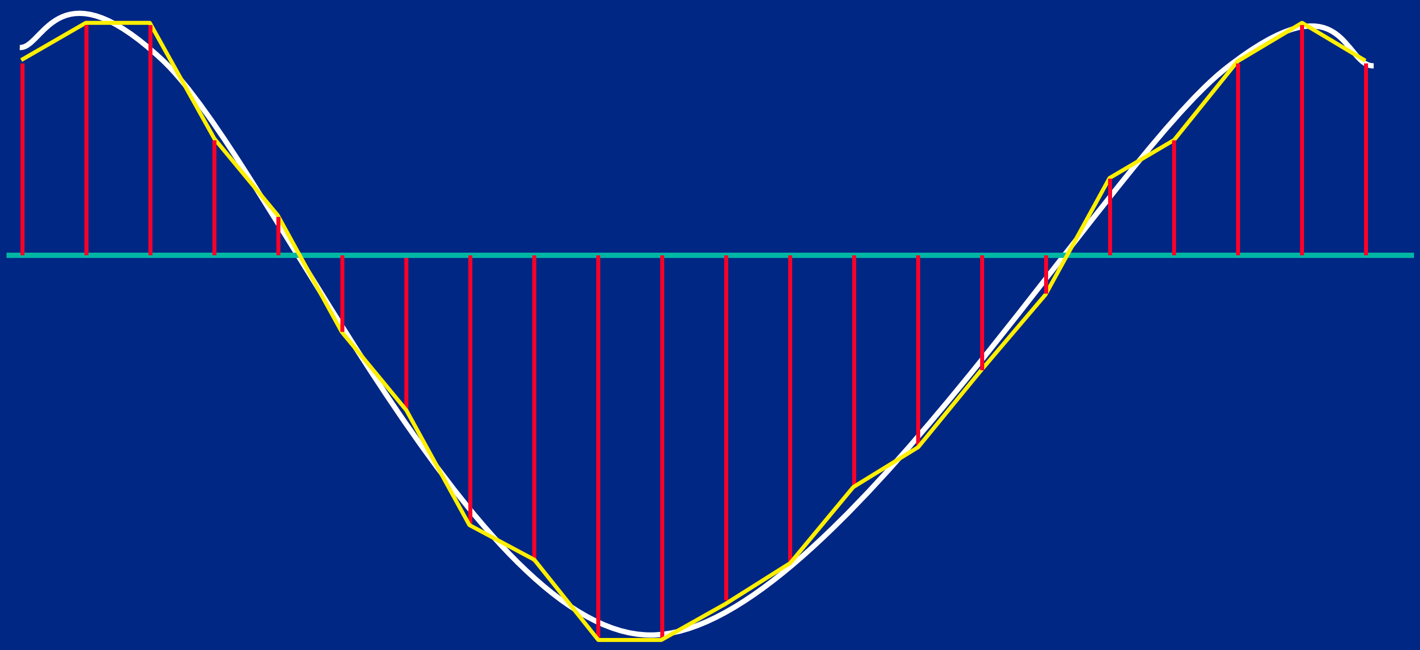
Quantization Accuracy



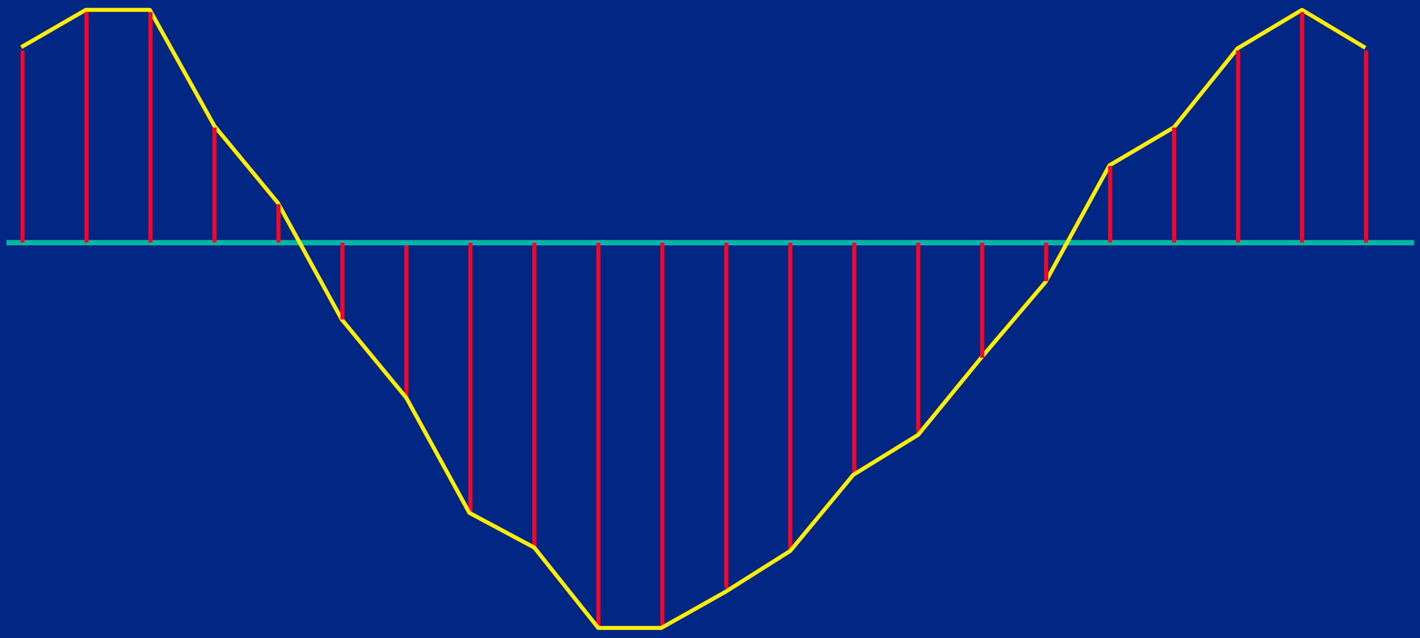
Quantization Accuracy



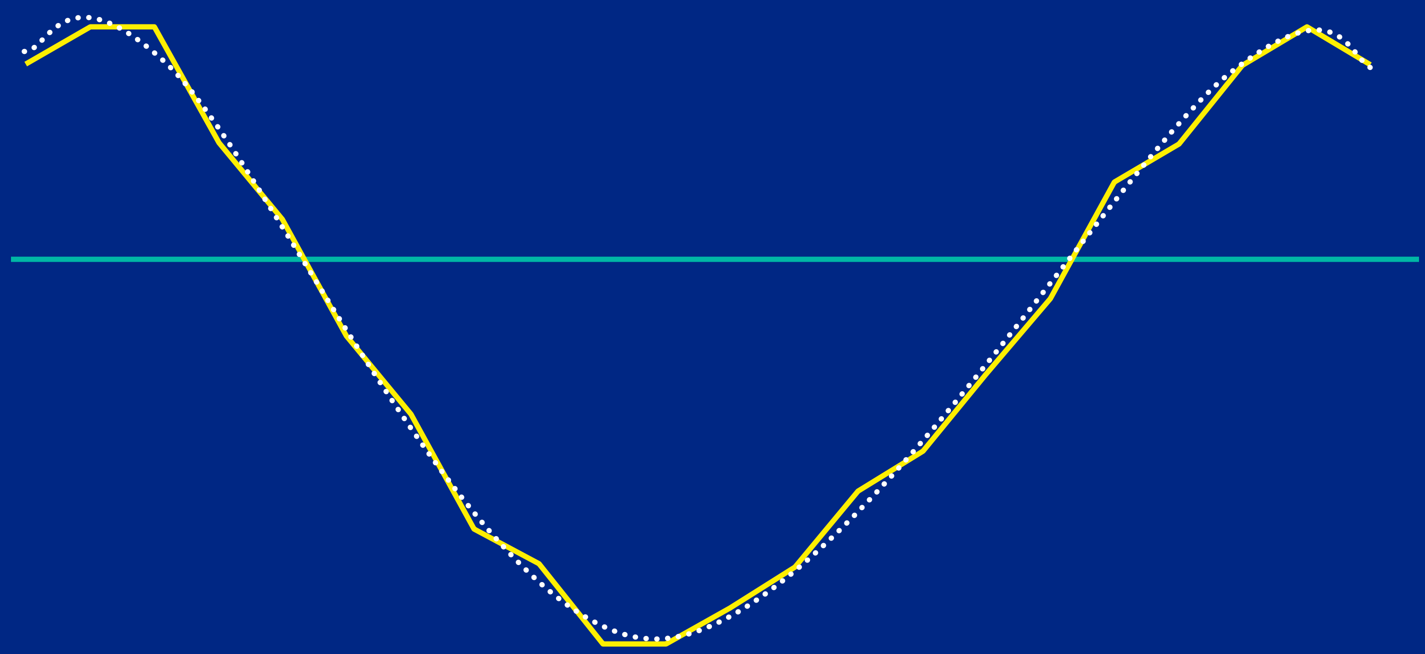
Quantization Accuracy



Quantization Accuracy



Quantization Accuracy



Common Sample Sizes

Music studio	24 bits
CD audio	16 bits
FM radio	~16 bits
Telephone	~8 bits

Codec

- Integrated circuit (coder/decoder)
- Provides A/D conversion, filtering, and programmable gain
- Switches between multiple inputs
- Data transfer is bi-directional

Si3000 Codec

- Installed on dsPICDEM™ Demo Board
- Suitable for voiceband applications
- Sample rate: up to 12 kHz
- Sample size: up to 16 bits

Typical Applications of Digital Sound Processing

- Cell phones
- Music players
- Voice recorders
- Special effects
  - Delay-based (echo, chorus, etc.)
  - Time or pitch scaling

Important Design Criteria

- Frequency response
  - Signals that can be reproduced

    A function of sample rate
- Sound quality (signal-to-noise ratio)
  - The accuracy of signal reproduction

    A function of sample size

Important Design Criteria

- Sample bandwidth
  - Maximum processing time per sample

    A function of sample rate and clock speed
- Latency
  - Maximum delay time

    A function of sample rate and memory size

Review

- What is digital sound?
- What is a codec?
- What are typical applications?
- What are important design criteria?

Demo

Application Design

Application Design

- Data Structures
- Control Flow Structures

Design Considerations

- Samples must flow without interruption
  - The slightest glitch is a pop
  - Coding or timing errors can be dramatic

    Don't use headphones!
- Samples must be buffered
  - Break the data stream into chunks
  - Provide an abstraction layer for software

Delay Line Structure

- Delay Line is a series of buffers
  - Stores data stream in memory
- Creates latency
  - By providing delay between input and output
  - This time can be used for processing

Delay Line Structure

- Each buffer is a series of n samples
  - Buffer length n depends on algorithm
  - n must be a power of 2 for FFTs, etc.

- Buffer length also represents time

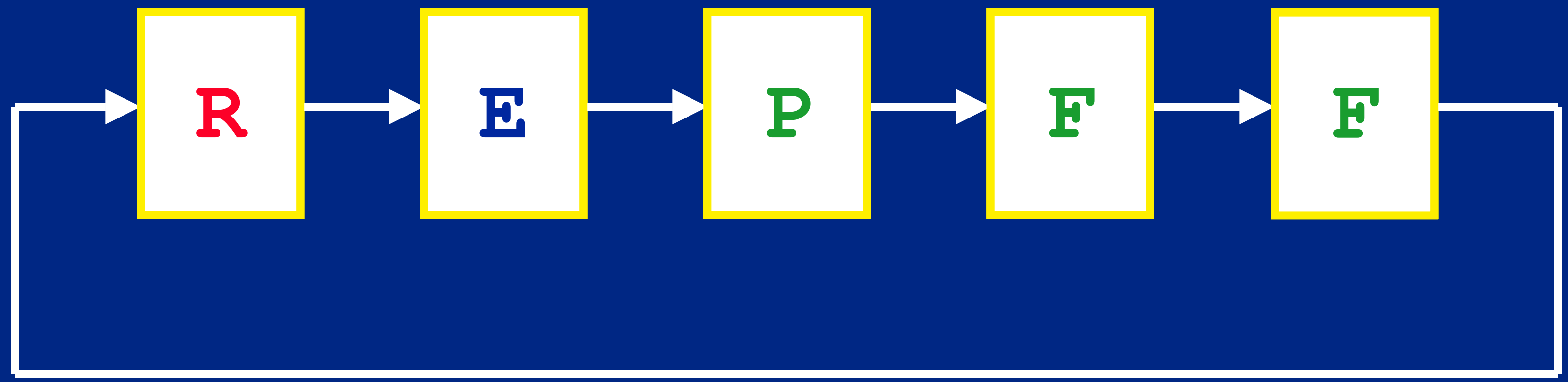
  length (samples) / sampling rate = time per buffer

  256 / 9000 = 28.4 ms

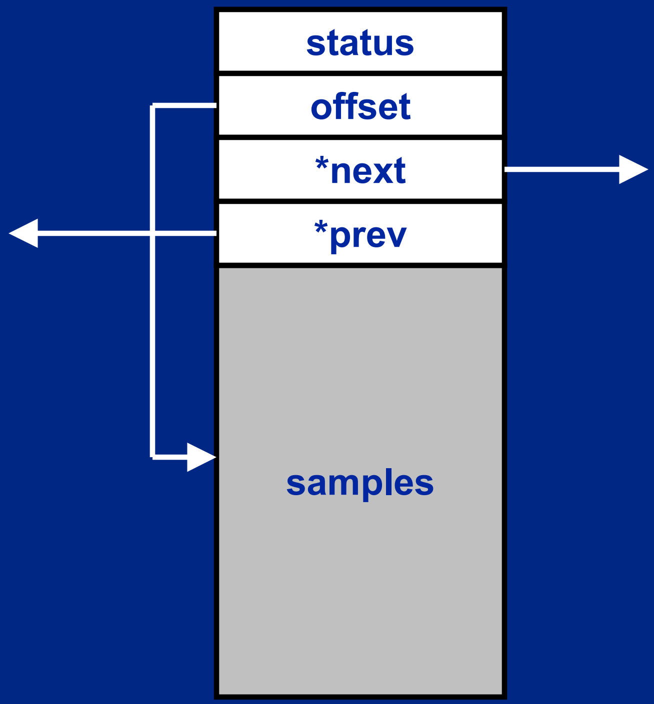
  256 / 11520 = 22.2 ms

  256 / 12000 = 21.3 ms

Delay Line Structure



Delay Line Data Structure




```
typedef struct audio_buffer {
    int status;
    int offset;
    struct audio_buffer *next;
    struct audio_buffer *prev;
    int sample[NUM_SAMPLES];
} audio_bufferT;
```

Control Flow Structures

Application Control Flow

- Background Tasks
  - Sound Input/Output
  - User Interface
- Foreground Tasks
  - Buffer Processing
  - User Interface

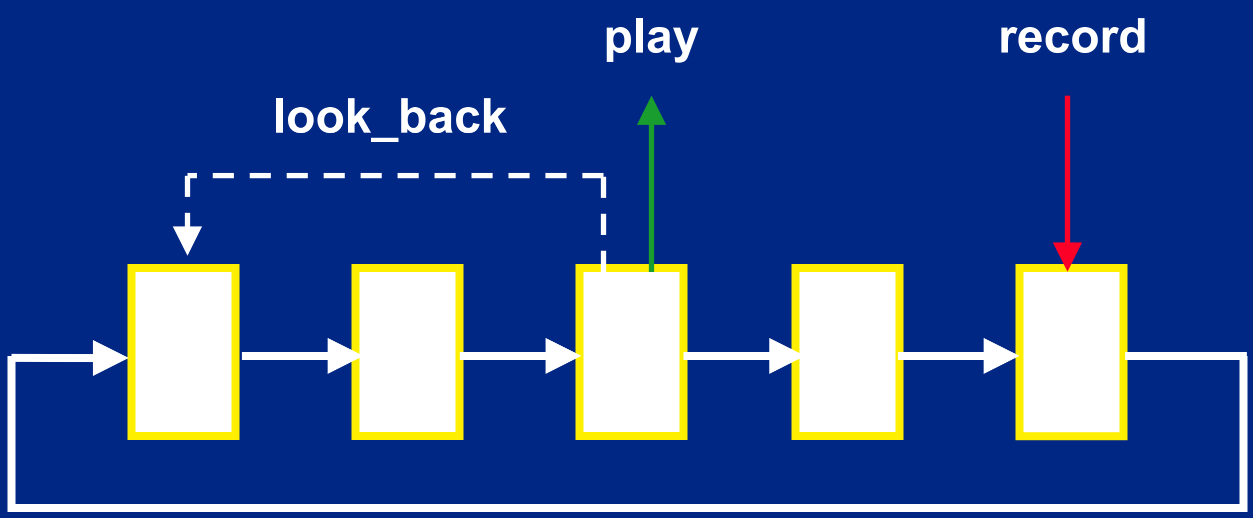
Low-Priority Interrupts

- Timer 2 interrupt
  - Provides visual cue that system is running
- A/D interrupt
  - Reads potentiometers
  - Values are stored in global variables
  - Configured to scan slowly (~n per sec)

High-Priority Interrupt

- Data Converter Interface (DCI) interrupt
  - Reads and writes data to the codec
  - Four samples processed per interrupt
  - Uses global variables for delay line pointers
  - Updates buffer status as needed

Delay Line Pointers




```
audio_bufferT *curr_rec_buf;
audio_bufferT *curr_play_buf;
audio_bufferT *look_back_buf;
```

Main Loop

- Main loop responds to state changes in the delay line
- Initiates foreground tasks as needed
  - Digital filtering
  - Spectrum analysis
  - Compression or expansion
  - Reading or writing to storage

Switches

- Polled when no foreground tasks are active
- Switch bounce must be accounted for
  - measured in ms (~100 ms)

Summary of Control Flow

- Interrupts
  - DCI reads and writes samples
  - A/D reads potentiometers
  - Timer2 blinks LED
- Main Loop
  - responds to state changes in delay line
  - polls switches
  - responds to user input

Lab #1

Lab #1

- Build sample application
- Program demo board using ICD2
- Play a sound track through demo board
- Evaluate sound quality

Processor Utilization

Processor Utilization

- Expressed as % of total cycles per unit time

  Should not exceed 50%

- Two categories of processor loading
  - Background tasks (interrupts)
  - Foreground tasks (initiated by main loop)

Background Task Loading

- Calculate cycles-per-sample interrupt
- If multiple paths exist, assume longest
- At least two values are needed
  - Maximum loading (buffer boundaries)
  - Average loading

Foreground Task Loading

- Calculate cycles-per-buffer
- Assume the slowest control path
- Task loading depends on buffer size and algorithm
- Refer to library documentation

Lab #1 Results

Instr. Rate	Sample Rate	CPU Load
15 MIPS	7.2 kHz	4.5%
15 MIPS	9.6 kHz	6.0%

(maximum background loading, opt level 1)

Critical Rule #1

- Maximum background loading must not exceed sample bandwidth

The slowest interrupt processing must complete before the next sample interrupt.

Critical Rule #2

- Maximum foreground loading plus average background loading must not exceed latency

The slowest buffer processing must complete before the next buffer is needed.

Lab #2

Lab #2

- Select instruction rate (edit sound.h)
- Rebuild app., program demo board
- Record processor utilization
- Adjust controls
- Modify sample application
  - Add a foreground task (edit main.c)
  - Change pre-recorded sounds (edit sounds.c)

Closing Remarks

Available Resources

- For information on dsPIC™ Development Tools and libraries
  - 16-Bit digital signal controller information
- For information on Si3000 Voice Codec
  - www.silabs.com/products/wireline/si3000.asp