

Outline: *System Development and Programming with the ADSP-21160*

Course Name:	System Development and Programming with the ADSP-21160
Course Number:	ADST-120
Course Description:	<p>This is a practical course with ‘hands on’ training using the latest VisualDSP++ software development tools. First the core elements of the processor, which includes the Computational Units, the Data Address Generators, and the Program Sequencer, are examined in detail along with the relevant assembly code instructions. A number of simulator labs help in understanding operation of the individual elements. Memory configuration (both internal and external) is discussed next. Advanced instructions are presented with a follow on lab on code optimization. The I/O peripherals, which include the SPORTS, Link Ports, and External Port, are discussed in detail along with DMA operation between these peripherals and internal memory. This section also deals with system booting and SBSRAM interfacing. Hardware development tools, such as evaluation boards and ICE’s are introduced. Throughout the course, the various aspects of the software development process using the latest tools are discussed including setting up and building projects, assembly language programming, code debugging, simulation, tool support for code overlays and shared memory, and ‘C’ programming support.</p>
Goals/Objectives:	<p>The main course objective is to understand the architecture of the ADSP-21160 DSP sufficiently to enable DSP system designers to resolve hardware/software issues with their applications. Additional goals include gaining a thorough understanding of both assembly language programming and code development (including ‘C’ programming issues) with the latest software tools</p>
Pre-requisites:	Previous embedded microprocessor background would be an asset (hardware and/or software)
Target Audience:	System Designers needing to make informed decisions on design tradeoffs, Hardware Designers needing to develop external interfaces, and Code Developers needing to know how to get the highest performance from their algorithms
Target Duration:	3.5 days

1 Introduction

1.1 Introductions/ Course Overview

- 1.1.1 Purpose of the Course**
- 1.1.2 Course Overview**
- 1.1.3 Logistics (breaks, lunch, etc.)**
- 1.1.4 Course Handouts**
- 1.1.5 DSP at Analog**
- 1.1.6 Analog Devices Strategy**
- 1.1.7 Signal Processor Portfolio**

1.2 Introduction to ADSP-21160

- 1.2.1 Characteristics of a Good DSP**
- 1.2.2 ADSP-21000 Family Features**
- 1.2.3 Modified Harvard Architecture**
- 1.2.4 ADSP-21160 Basic System Configuration**

2 Introduction to Software Tools (VisualDSP++)

2.1 VisualDSP Overview

2.2 Software Tools Overview

- 2.2.1 Development Flow**
- 2.2.2 Invoking Tools and Switches**
- 2.2.3 IDDE (Integrated Development/Debug Environment)**
 - 2.2.3.1 Features**
 - 2.2.3.2 Projects**
 - 2.2.3.3 Property Pages**
 - 2.2.3.4 Debug Sessions**
 - 2.2.3.5 Online Help**

3 The ADSP-21K Family Core Architecture - Part I

3.1 The ADSP-21000 Family Core Internal Architecture

- 3.1.1 Registers and Data Types**
- 3.1.2 Register types: UREG, SREG, DREG - overview**
- 3.1.3 Register File**
- 3.1.4 Native data types and data word alignment**
- 3.1.5 Fixed point - 32 bit integer, fractional**
- 3.1.6 Floating point - 32 bit single precision, 40 bit extended single precision**
- 3.1.7 Simulator Exercise: registers exercise, basic simulator operation**

3.2 ALU

- 3.2.1 Features**
- 3.2.2 Instructions**
- 3.2.3 Flags**
- 3.2.4 Simulator Exercise: ALU operation**
- 3.2.5 Mini-Quiz**

3.3 Multiplier/MAC

- 3.3.1 Features**
- 3.3.2 Instructions**
- 3.3.3 Flags**
- 3.3.4 Fractional and integer math**
- 3.3.5 Simulator Exercise: MAC operation**
- 3.3.6 Mini-Quiz**

3.4 Shifter

- 3.4.1 Features**
- 3.4.2 Instructions**
- 3.4.3 Flags**
- 3.4.4 Simulator Exercise: Shifter operation**
- 3.4.5 Mini-Quiz**

4 The ADSP-21K Family Core Architecture - Part II

4.1 Internal Core Architecture Review

4.2 Data Address Generators (DAGs)

- 4.2.1 Features**
- 4.2.2 Instructions**
- 4.2.3 Immediate data move instructions**
- 4.2.4 Modulo addressing example**
- 4.2.5 Simulator Exercise: DAG operation**
- 4.2.6 Mini-Quiz**

4.3 Program Sequencer

- 4.3.1 Features**
- 4.3.2 Instructions**
- 4.3.3 Instruction pipeline**
- 4.3.4 Branching, Delayed branching**
- 4.3.5 Zero overhead looping**
- 4.3.6 Instruction cache, PM data access**
- 4.3.7 Interrupts**
- 4.3.8 Status and Mode registers / USTAT registers**
- 4.3.9 System register bit operations**

4.4 Timer

- 4.4.1 Features**

5 Memory

5.1 SHARC Memory

- 5.1.1 SHARC Memory Basics**
- 5.1.2 SHARC Memory Map**
- 5.1.3 SHARC Internal Architecture**

5.2 SHARC Internal SRAM

- 5.2.1 Internal SRAM Architecture**
- 5.2.2 Memory Access Considerations**
- 5.2.3 Internal Memory Maps**
- 5.2.4 Configuring Internal Memory**
 - 5.2.4.1 32 bit vs 40 bit Data**
 - 5.2.4.2 48 bit Instructions**
 - 5.2.4.3 Mixing 32 bit and 48 bit words in one block**
- 5.2.5 Example LDF Memory Section**

6 Assembly Code Development with VisualDSP++

6.1 Project Development

- 6.1.1 Project Options**
- 6.1.2 Build Configurations**
- 6.1.3 Building Projects**

6.2 Assembly

- 6.2.1 Build Process**
- 6.2.2 Sections**
- 6.2.3 Controlling Assembler through Directives and Operators**
- 6.2.4 Preprocessor**
- 6.2.5 Symbolic Names**

6.3 Basic Linker Description File (LDF)

6.3.1 Introduction

6.3.2 Overview

6.3.3 Example LDF File

6.3.4 Example Commands

6.3.5 Expert Linker

6.4 VisualDSP++ Simulator

6.4.1 Overview

6.4.2 Simulator Features

6.4.3 Simulator Exercise: basic code development exercise

7 Advanced Instruction Types

7.1 Parallel Instruction Types and Multifunction Computations

7.1.1 Conditional register swap example

7.1.2 Data registers usage for multifunction computes

7.2 Reciprocal and Divide

7.3 Reciprocal Square Root and Square Root

7.4 Simulator Exercise: code optimization

8 SIMD Single Instruction Multiple Data

8.1 Terms

8.2 Features

8.3 SISD vs SIMD

8.4 Data Access

8.5 SIMD Programming Models

8.6 Status Flags

8.7 Simulator Exercise: SIMD operation

9 ADSP-21160 I/O Processor Architecture 1

9.1 Features

9.2 IOP Structure

9.2.1 IO Processor Features

9.3 DMA Unit

9.3.1 DMA Architecture

9.3.2 DMA Features

9.3.3 DMA modes & examples

9.3.4 DMA Interrupts

9.3.5 External Port DMA

10 ADSP-21160 I/O Architecture 2

10.1 External Port

10.1.1 Memory Interface

10.1.1.1 EP Configuration

10.1.1.2 SBSRAM Interfacing Example

10.1.1.3 Executing from External Memory

10.1.2 Shared Bus Multiprocessing (Cluster mode)

10.1.2.1 Bus Arbitration

10.1.2.2 Interprocessor Data Transfers

10.1.2.3 Semaphores

10.1.3 Host Interface

10.2 Link Port

10.2.1 Link Features

10.2.2 Link Pin Description & Function

10.2.3 Link Configuration—DMA & Control

10.2.4 Link Communications code example

10.3 Serial Port (SPORT)**10.3.1 Sport Features****10.3.2 Sport Pin Description****10.3.3 Sport Modes****10.3.4 Sport Configuration****10.4 System Booting****10.4.1 SHARC Powerup Boot****10.4.2 Multiprocessor booting****11 System Design****11.1 Basic system configuration****11.2 Navigating the Datasheet****11.3 Design resources/guidelines****11.4 JTAG Port overview****12 Advanced LDF Features****12.1 Shared Memory****12.1.1 Definition****12.1.2 Example****12.2 Multi-Processing****12.2.1 Definition****12.2.2 Example****12.3 Software Overlays****12.3.1 What are Overlays?****12.3.2 Defining Live and Run Spaces with LDF****12.3.3 Overlay Operation****12.3.4 Example**

- 13 C Compiler Issues/Examples ANSI C Compiler:**
 - 13.1 C Compiler Features and use in Embedded Systems**
 - 13.2 Configuring C Compiler via IDDE**
 - 13.3 Data Types**
 - 13.4 LDF for C Compiler**
 - 13.4.1 C and Section Names**
 - 13.4.2 Example C Compilation to Assembly**
 - 13.5 Stacks and Heaps**
 - 13.6 Run Time Header**
 - 13.6.1 Interrupt Handling with C Code**
 - 13.7 Assembly Language Interface**
 - 13.7.1 Register Usage and Data Types**
 - 13.7.2 C Callable Assembly Functions**
 - 13.7.3 C/Assembly Interface Example**
 - 13.8 C Code Optimization**
 - 13.8.1 C Code with SIMD**
 - 13.8.2 Simulator Exercise: C coding and optimization**

- 14 ADSP-21160 Hardware Tools**
 - 14.1 Hardware Tool Overview with part numbers**
 - 14.2 ADSP-21160 EZ-Kit Lite**
 - 14.2.1 Hardware**
 - 14.2.2 Software**
 - 14.3 JTAG ICE Emulators**
 - 14.3.1 ICE Configurator**
 - 14.3.2 ICE Debug Features**

- 15 Mini Quiz Answers**