Outline: *System Development and Programming with the ADSP-BF535*

Course Name:	System Development and Programming with the ADSP-BF535
Course Number:	ADST-150
Course Description:	This is a practical course with 'hands on' training using the latest software development tools. First the core elements of the Blackfin DSP, 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, SPI, UART, Host Port, and External Port, are discussed in detail along with DMA operation between these peripherals and internal memory. System related information such as booting, operation of the PC133 SDRAM controller, PCI/USB interfaces, RTC, power management, and other hardware design issues are also discussed. 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.
Goals/Objectives:	The main course objective is to understand the architecture of the ADSP-BF535 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
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

8 Program Development

8.1 IDDE and Projects

8.2 Assembler

8.2.1 Features and Overview

8.2.2 Assembler Expressions

8.2.3 Assembler Directives

8.2.4 Definition Files

8.3 Linker / Linker Description File (LDF)

8.3.1 Features and Overview

8.3.2 Example LDF

8.3.3 LDF Commands

8.4 Using the Simulator for Code Debug

8.4.1 Features and Overview

8.4.2 Invoking the Simulator

8.5 Assembly Code Exercise

9 Advanced Instructions

9.1 8-bit ALU Instructions

9.2 Vector Operations

9.3 Issuing Parallel Instructions

9.4 Code Optimization

9.5 Optimization Exercise

10 Timers and Flags

10.1 Timers

10.2 Real Time Clock

10.3 Watch Dog Timer

10.4 Programmable Flags

11 Direct Memory Access (DMA)

11.1 Overview

11.2 Descriptor based DMA

11.3 Autobuffer based DMA

11.4 Memory DMA

12 Parallel Interface

12.1 External Bus Interface Unit (EBIU)

12.1.1 Features and Overview

12.1.2 External Memory Interface

12.2 Asynch Memory Controller

12.3 SDRAM Controller

16 System Booting

16.1 Booting Modes

16.2 Boot Kernel (2nd Stage Loader)

16.3 VisualDSP Loader Utility

16.4 Loader file formats

17 System Design

17.1 Operating modes

17.2 Dynamic Power Management

17.2.1 Clock generation / PLL

17.2.2 Power-down modes

17.3 Resetting the DSP

17.3.1 Reset Types

17.3.2 Power-up sequence

17.4 JTAG Overview

17.4.1 ICE Header

17.5 Board Design and Layout

17.5.1 Navigating the Datasheet

17.5.2 Anomaly Sheet

17.5.3 Design Guidelines

18 C/C++ Compiler

18.1 Features

18.2 Configuration

18.3 LDF Related issues

18.4 Register Usage

18.5 Run Time Environment

18.5.1 Stack and HEAP

18.5.2 Interrupt Handling

18.5.3 Assembly Code Interface

18.6 Run-time Library Overview

18.7 C++ capabilities

18.8 C Code Optimization

19 ADSP-BF535 Hardware Tools

19.1 Overview with Part Numbers

19.2 ADSP-BF535 EZ-Kit

19.3 In Circuit Emulators (ICE)